SONiX Technology Co., Ltd.

SN8F5814 Series Datasheet

8051-based Microcontroller

SN8F5814
SN8F5813
SN8F5812

1 Device Overview

1.1 Features

- **Enhanced 8051 microcontroller** with reduced instruction cycle time (up to 12 times 80C51)
- Up to 8 MHz flexible CPU frequency
- Internal 32 MHz Clock Generator (IHRC), 1 MHz to 16 MHz crystal, and external synchronous clock source selections
- 1-set external low clock 32.768 kHz crystal
- **16 KB non-volatile flash memory (IROM)** with in-system program support
- **256 bytes internal RAM (IRAM)**
- **512 bytes external RAM (XRAM)**
- **15 interrupt sources with priority levels control and unique interrupt vectors**
- 12 internal interrupts
- 3 external interrupts: INT0, INT1, INT2
- 1 set of DPTR
- 2 set 8/16-bit timers with 4 operation modes
- 1 set 16-bit timers with 4 comparison output (PWM) and capture channels
- **1 set 16-bit PWM generators**:
  PWM generator has 4 output channels with inverters and dead-band control, and error detect function
- **12-bit SAR ADC** with 13 external and 1 internal channels, and 4 internal reference voltages
- **4*16 R-type LCD Driver with static mode**
- **16 channel Capacitive touch**
- **SPI, UART, I2C** interface with SMBus Support
- **On-Chip Debug Support**:
  Single-wire debug interface
  2 hardware breakpoints
  Unlimited software breakpoints
  ROM data security/protection
- Watchdog and programmable external reset
- 1.8V low voltage detectors
- Wide supply voltage (1.8 V – 5.5 V) and temperature (-40 °C to 85 °C) range

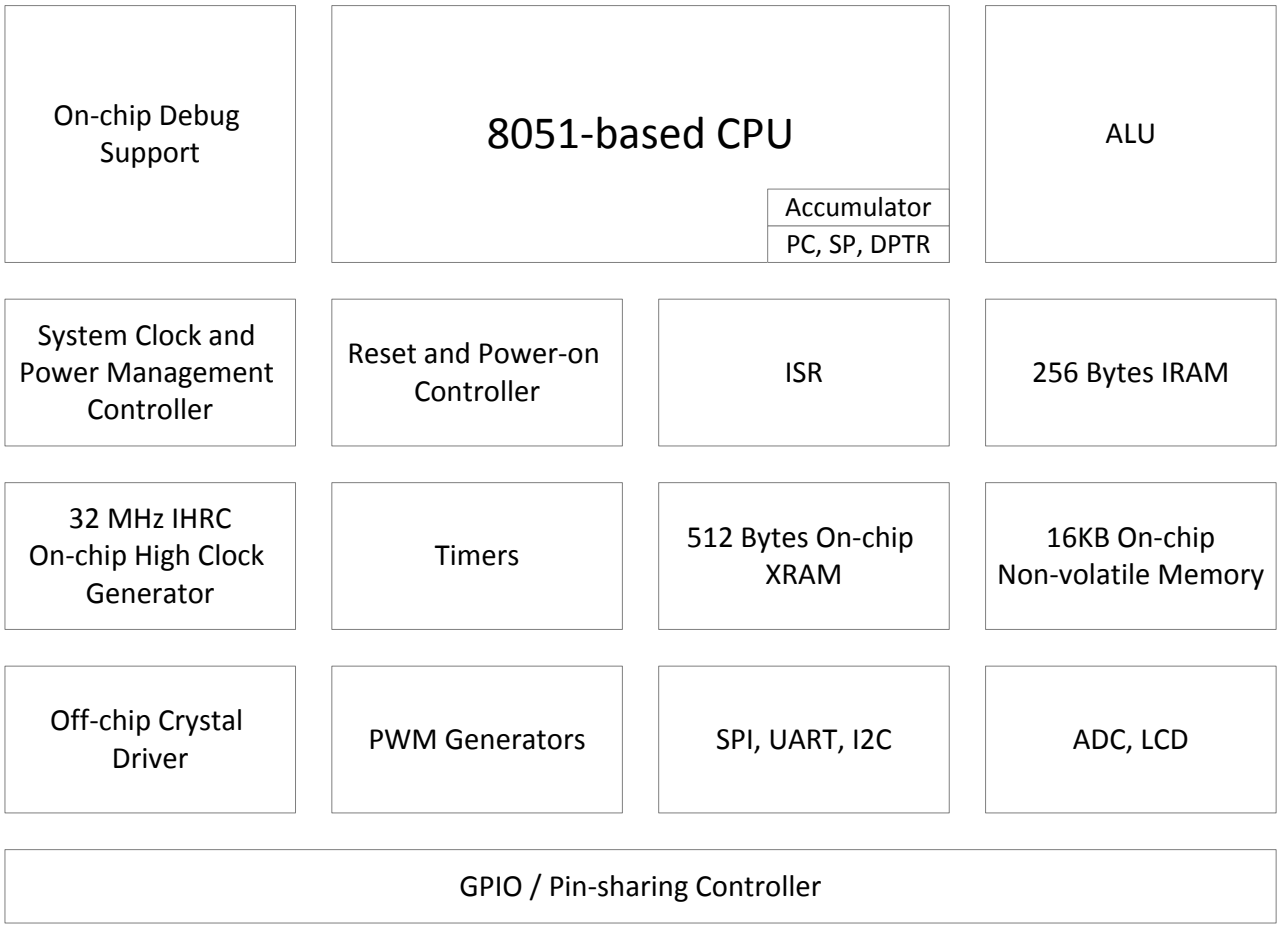
1.2 Applications

- Home automation
- Touch Key

1.3 Features Selection Table

	I/O	PWM Channels	I2C	SPI	UART	ADC ext. Channels	LCD	Touch	Ext. INT	Package Types
SN8F5814	26	8	V	V	V	13	4*16	16	3	SOP28 SSOP28 TSSOP28
SN8F5813	22	6	V	V	V	13	4*14	14	3	SOP24 TSSOP24
SN8F5812	18	6	V	V	V	11	4*10	10	3	SOP20 TSSOP20

1.4 Block Diagram



2 Table of Contents

3 Revision History

Revision	Date	Description
1.0	Mar. 2017	First issue.
1.1	Mar. 2017	Modify ROM Programming Pin description.
1.2	Mar. 2017	Modify Request Flag Clearance section.
1.3	Apr. 2017	1. Modify UART Baud Rate Control section. 2. Modify Features Selection Table. 3. Modify Pin Assignments chapter.
1.4	Apr. 2017	1. Modify MP5 Writer Transition Board Socket Pin Assignment. 2. Modify Pin Assignments chapter.
1.5	May. 2017	1. In-System Program chapter removes byte write operation. 2. Adds Pin Characteristic section.
1.6	May. 2017	Repair an error, omission, etc.
1.7	Sep. 2017	Add detail content and description in all of section.
1.8	Nov. 2017	Modify LVD related content.
1.9	Jul. 2018	1. Repair an error, omission, etc. 2. Modify UART baud rate table. 3. Modify I2C clock rate table. 4. Add SN5814 Starter-kit chapter.
2.0	Sep. 2018	1. Repair an error, omission, etc. 2. MP5 Writer Programming Pin Mapping adds normal mode and high speed mode sections. 3. Modify SOP24 outline description.
2.1	Nov. 2018	1. Repair an error, omission, etc. 2. Modify system clock section description. 3. Modify Pin Circuit Diagrams section. 4. Add stack and power-on sequence description. 5. Timer 0 and Timer 1 chapter adds Function Enable and Disable section.
2.2	Jan. 2019	1. Repair an error, omission, etc. 2. Modify normal mode and IDLE mode supply current value. 3. Remove SN8F5814K pin assignment.
2.3	Feb. 2019	1. Repair an error, omission, etc. 2. Update Package Information dimensions and illustrations. 3. Modify ADC input offset range. 4. Modify power on sequence and system clock timing. 5. Modify Timer0/ Timer1 section description.

SONIX reserves the right to make change without further notice to any products herein to improve reliability, function or design. SONIX does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights nor the rights of others. SONIX products are not designed, intended, or authorized for us as components in systems intended, for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the SONIX product could create a situation where personal injury or death may occur. Should Buyer purchase or use SONIX products for any such unintended or unauthorized application. Buyer shall indemnify and hold SONIX and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, cost, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use even if such claim alleges that SONIX was negligent regarding the design or manufacture of the part.

4 Pin Assignments

4.1 SN8F5814S/X/T (SOP28/SSOP28/TSSOP28)

VSS	1	U	28	VDD
SCL/T2COM0/INT1/XIN/P00	2		27	P30/CM0/AIN0/SEG15/INT0/AVREFH/PWMFL
SDA/T2COM1/XOUT/P01	3		26	P31/AIN1/SWAT
PWM10/T2CC2/LXIN/P02	4		25	P20/CM1/AIN2/SEG14/PWM11
PWM20/T2CC3/LXOUT/P03	5		24	P21/CM2/AIN3/SEG13/PWM21
INT2/RST/P04	6		23	P22/CM3/AIN4/SEG12/MOSI
T2CC0/COM0/P05	7		22	P23/CM4/AIN5/SEG11/MISO
T2CC1/COM1/P06	8		21	P24/CM5/AIN6/SEG10/SCK
T2COM2/COM2/P07	9		20	P25/CM6/AIN7/SEG9/SSN
T2COM3/COM3/P10	10		19	P26/CM7/AIN8/SEG8/UTX
SEG0/CM15/P11	11		18	P27/CM8/AIN9/SEG7/URX
SEG1/CM14/P12	12		17	P17/CM9/AIN10/SEG6
SEG2/CM13/P13	13		16	P16/CM10/AIN11/SEG5
SEG3/CM12/P14	14		15	P15/CM11/AIN12/SEG4

4.2 SN8F5813S/T (SOP24/TSSOP24)

VSS	1	U	24	VDD
SCL/T2COM0/INT1/XIN/P00	2		23	P30/CM0/AIN0/SEG15/INT0/AVREFH/PWMFL
SDA/T2COM1/XOUT/P01	3		22	P31/AIN1/SWAT
INT2/RST/P04	4		21	P20/CM1/AIN2/SEG14/PWM11
T2CC0/COM0/P05	5		20	P21/CM2/AIN3/SEG13/PWM21
T2CC1/COM1/P06	6		19	P22/CM3/AIN4/SEG12/MOSI
T2COM2/COM2/P07	7		18	P23/CM4/AIN5/SEG11/MISO
T2COM3/COM3/P10	8		17	P24/CM5/AIN6/SEG10/SCK
SEG0/CM15/P11	9		16	P25/CM6/AIN7/SEG9/SSN
SEG1/CM14/P12	10		15	P26/CM7/AIN8/SEG8/UTX
SEG4/AIN12/CM11/P15	11		14	P27/CM8/AIN9/SEG7/URX
SEG5/AIN11/CM10/P16	12		13	P17/CM9/AIN10/SEG6

4.3 SN8F5812S/T (SOP20/TSSOP20)

VSS	1	U	20	VDD
SCL/T2COM0/INT1/XIN/P00	2		19	P30/CM0/AIN0/SEG15/INT0/AVREFH/PWMFL
SDA/T2COM1/XOUT/P01	3		18	P31/AIN1/SWAT
INT2/RST/P04	4		17	P20/CM1/AIN2/SEG14/PWM11
T2CC0/COM0/P05	5		16	P21/CM2/AIN3/SEG13/PWM21
T2CC1/COM1/P06	6		15	P22/CM3/AIN4/SEG12/MOSI
T2COM2/COM2/P07	7		14	P23/CM4/AIN5/SEG11/MISO
T2COM3/COM3/P10	8		13	P24/CM5/AIN6/SEG10/SCK
SEG5/AIN11/CM10/P16	9		12	P26/CM7/AIN8/SEG8/UTX
SEG6/AIN10/CM9/P17	10		11	P27/CM8/AIN9/SEG7/URX

4.4 Pin Descriptions

Power Pins

Pin Name	Type	Description
VDD	Power	Power supply.
VSS	Power	Ground (0 V).

Port 0

Pin Name	Type	Description
P0.0	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
XIN	Analog Input	System clock: external clock input.
SCL	Digital I/O	I2C: clock output (master) clock input (slave).
T2COM0	Digital Output	Timer 2: compare 0 output.
INT1	Digital Input	INT1: external interrupt 1.
P0.1	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
XOUT	Analog Output	System clock: drive external crystal/resonator.
SDA	Digital I/O	I2C: data pin.
T2COM1	Digital Output	Timer 2: compare 1 output.
P0.2	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
LXIN	Analog Input	Low clock: external clock input.
T2CC2	Digital Input	Timer 2: capture 2 input.
PWM10	Digital Output	PWM: programmable PWM output.
P0.3	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
LXOUT	Analog Output	Low clock: drive external crystal.
T2CC3	Digital Input	Timer 2: capture 3 input.
PWM20	Digital Output	PWM: programmable PWM output.
P0.4	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
Reset	Digital Input	System reset (active low).
INT2	Digital Input	INT2: external interrupt 2.
P0.5	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
T2CC0	Digital Input	Timer 2: capture 0 input.
COM0	Analog Output	LCD: Common 0 output.

P0.6	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
T2CC1	Digital Input	Timer 2: capture 1 input.
COM1	Analog Output	LCD: Common 1 output.
P0.7	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
T2COM2	Digital Output	Timer 2: compare 2 output.
COM2	Analog Output	LCD: Common 2 output.

Port 1

Pin Name	Type	Description
P1.0	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
T2COM3	Digital Output	Timer 2: compare 3 output.
COM3	Analog Output	LCD: Common 3 output.
P1.1	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
SEG0	Analog Output	LCD: Segment 0 output.
CM15	Analog Input	CM15: Cap-sensing input channel 15.
P1.2	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
SEG1	Analog Output	LCD: Segment 1 output
CM14	Analog Input	CM14: Cap-sensing input channel 14
P1.3	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
SEG2	Analog Output	LCD: Segment 2 output.
CM13	Analog Input	CM13: Cap-sensing input channel 13.
P1.4	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
SEG3	Analog Output	LCD: Segment 3 output.
CM12	Analog Input	CM12: Cap-sensing input channel 12.
P1.5	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
SEG4	Analog Output	LCD: Segment 4 output.
AIN12	Analog Input	ADC: input channel 12.
CM11	Analog Input	CM11: Cap-sensing input channel 11.
P1.6	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.

SEG5	Analog Output	LCD: Segment 5 output.
AIN11	Analog Input	ADC: input channel 11.
CM10	Analog Input	CM10: Cap-sensing input channel 10.
P1.7	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters. Level change wake-up.
SEG6	Analog Output	LCD: Segment 6 output.
AIN10	Analog Input	ADC: input channel 10.
CM9	Analog Input	CM9: Cap-sensing input channel 9.

Port 2

Pin Name	Type	Description
P2.0	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
PWM11	Digital Output	PWM: programmable PWM output.
SEG14	Analog Output	LCD: Segment 14 output.
AIN2	Analog Input	ADC: input channel 2.
CM1	Analog Input	CM1: Cap-sensing input channel 1.
P2.1	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
PWM21	Digital Output	PWM: programmable PWM output.
SEG13	Analog Output	LCD: Segment 13 output.
AIN3	Analog Input	ADC: input channel 3.
CM2	Analog Input	CM2: Cap-sensing input channel 2.
P2.2	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
MOSI	Digital I/O	SPI: transmission pin (master) reception pin (slave).
SEG12	Analog Output	LCD: Segment 12 output.
AIN4	Analog Input	ADC: input channel 4.
CM3	Analog Input	CM3: Cap-sensing input channel 3.
P2.3	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
MISO	Digital I/O	SPI: reception pin (master) transmission pin (slave).
SEG11	Analog Output	LCD: Segment 11 output.
AIN5	Analog Input	ADC: input channel 5.
CM4	Analog Input	CM4: Cap-sensing input channel 4.
P2.4	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
SCK	Digital I/O	SPI: clock output (master) clock input (slave).

SEG10	Analog Output	LCD: Segment 10 output.
AIN6	Analog Input	ADC: input channel 6.
CM5	Analog Input	CM5: Cap-sensing input channel 5.
P2.5	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
SSN	Digital I/O	SPI: slave selection pin (slave mode).
SEG9	Analog Output	LCD: Segment 9 output.
AIN7	Analog Input	ADC: input channel 7.
CM6	Analog Input	CM6: Cap-sensing input channel 6.
P2.6	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
UTX	Digital Output	UART: transmission pin.
SEG8	Analog Output	LCD: Segment 8 output.
AIN8	Analog Input	ADC: input channel 8.
CM7	Analog Input	CM7: Cap-sensing input channel 7.
P2.7	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
URX	Digital Input	UART: reception pin.
SEG7	Analog Output	LCD: Segment 7 output.
AIN9	Analog Input	ADC: input channel 9.
CM8	Analog Input	CM8: Cap-sensing input channel 8.

Port 3

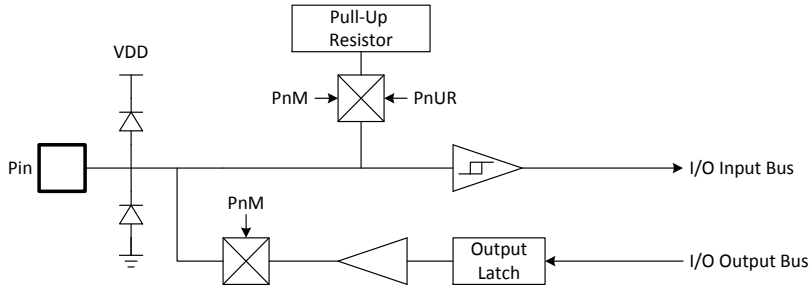
Pin Name	Type	Description
P3.0	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
INT0	Digital Input	INT0: external interrupt 0.
PWMFL	Digital Input	PWMFL: PWM fail detect pin.
SEG15	Analog Output	LCD: Segment 15 output.
AIN0	Analog Input	ADC: input channel 0.
AVREFH	Analog Input	ADC: external reference voltage.
CM0	Analog Input	CM0: Cap-sensing input channel 0.
P3.1	Digital I/O	GPIO: Bi-direction pin. Schmitt trigger structure as input mode. Built-in pull-up resisters.
SWAT	Digital I/O	Debug interface.
AIN1	Analog Input	ADC: input channel 1.

4.5 Pin Characteristic

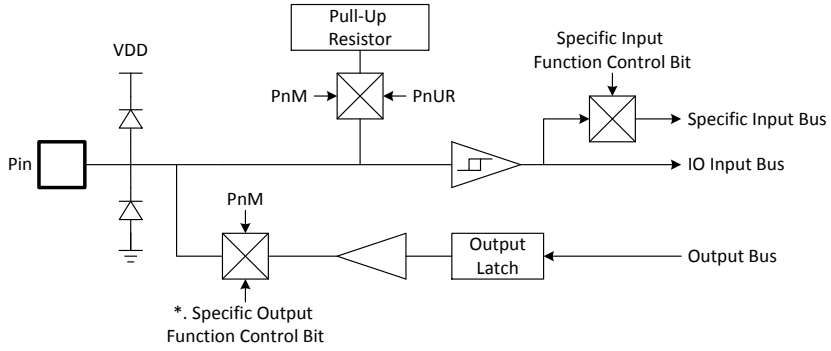
Port	Open-Drain	Sink Current 200mA VSS+1.5V	External Interrupt	Wakeup (Level change)	Shared Pin
P0.0	-	-	V	V	XIN/T2COM0/SCL/INT1
P0.1	-	-	-	V	XOUT/T2COM1/SDA
P0.2	-	-	-	V	LXIN/T2CC2/PWM10
P0.3	-	-	-	V	LXOUT/T2CC3/PWM20
P0.4	-	-	V	V	INT2/RST
P0.5	-	V	-	V	COM0/T2CC0
P0.6	-	V	-	V	COM1/T2CC1
P0.7	-	V	-	V	COM2/T2COM2
P1.0	-	V	-	V	COM3/T2COM3
P1.1	-	-	-	V	CM15/SEG0
P1.2	-	-	-	V	CM14/SEG1
P1.3	-	-	-	V	CM13/SEG2
P1.4	-	-	-	V	CM12/SEG3
P1.5	-	-	-	V	CM11/AIN12/SEG4
P1.6	-	-	-	V	CM10/AIN11/SEG5
P1.7	-	-	-	V	CM9/AIN10/SEG6
P2.0	-	-	-	-	CM1/AIN2/SEG14/PWM11
P2.1	-	-	-	-	CM2/AIN3/SEG13/PWM21
P2.2	V	-	-	-	CM3/AIN4/SEG12/MOSI
P2.3	V	-	-	-	CM4/AIN5/SEG11/MISO
P2.4	V	-	-	-	CM5/AIN6/SEG10/SCK
P2.5	-	-	-	-	CM6/AIN7/SEG9/SSN
P2.6	V	-	-	-	CM7/AIN8/SEG8/UTX
P2.7	V	-	-	-	CM8/AIN9/SEG7/URX
P3.0	-	-	V	-	CM0/AIN0/SEG15/INT0/AVREFH/PWMFL
P3.1	-	-	-	-	SWAT/AIN1

4.6 Pin Circuit Diagrams

Normal Bi-direction I/O Pin.

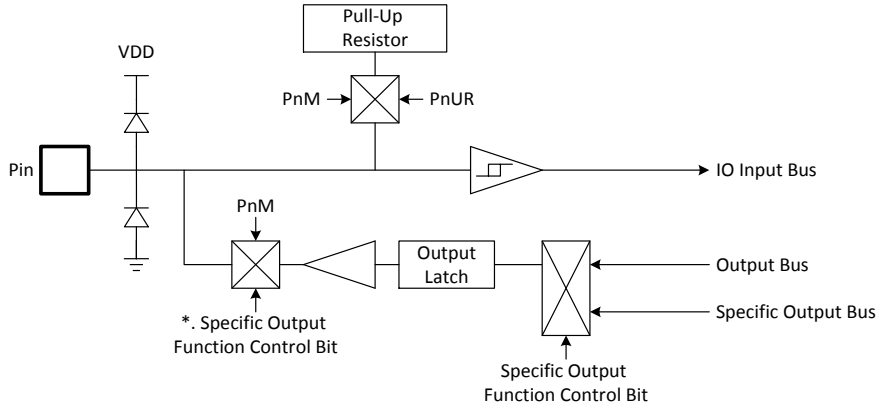


Bi-direction I/O Pin Shared with Specific Digital Input Function, e.g. INT2.



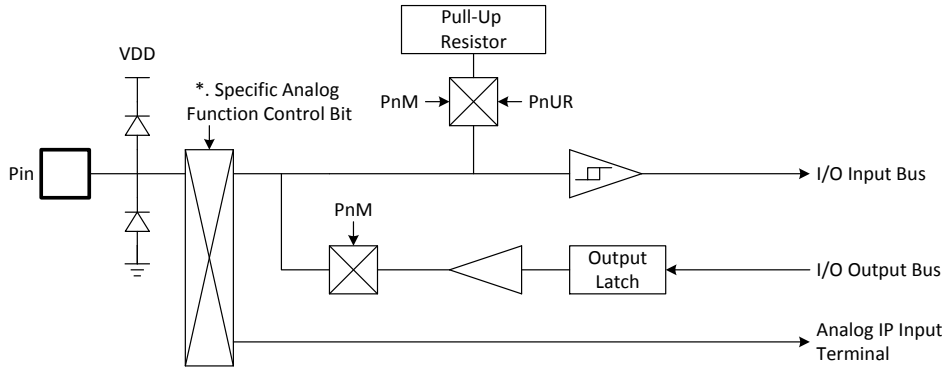
***. Some specific functions switch I/O direction directly, not through PnM register.**

Bi-direction I/O Pin Shared with Specific Digital Output Function, e.g. PWM, SIO, UART.



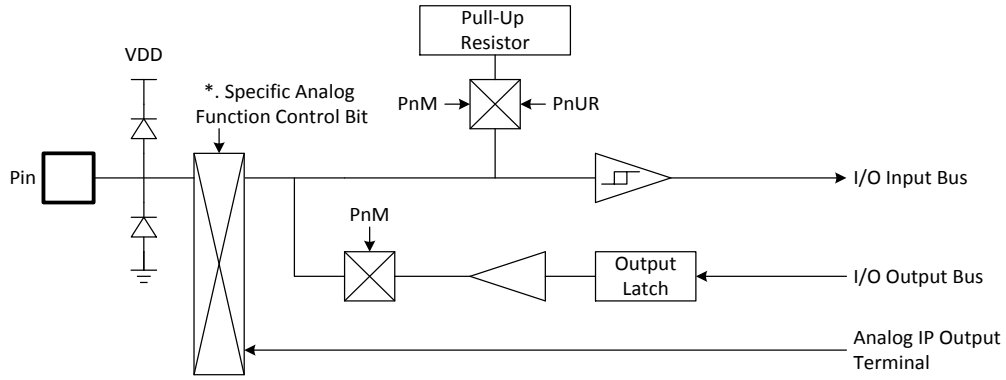
***. Some specific functions switch I/O direction directly, not through PnM register.**

Bi-direction I/O Pin Shared with Specific Analog Input Function, e.g. XIN, ADC.



***. Some specific functions switch I/O direction directly, not through PnM register.**

Bi-direction I/O Pin Shared with Specific Analog Output Function, e.g. XOUT…



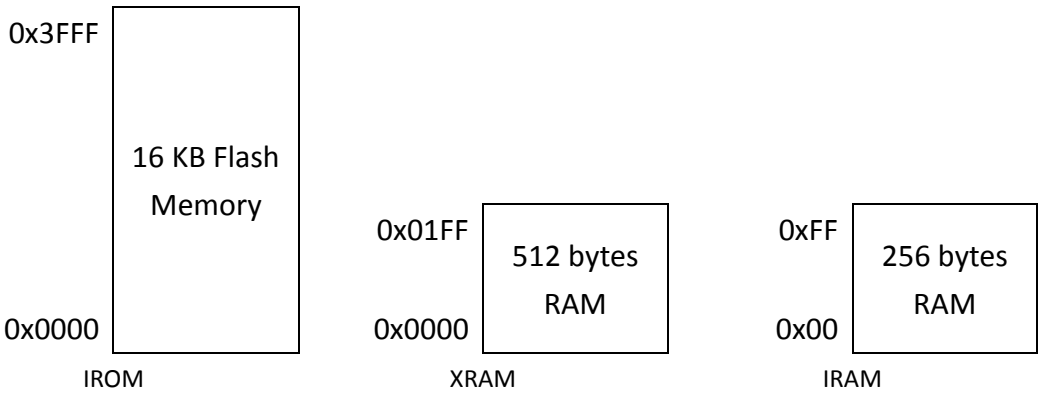
***. Some specific functions switch I/O direction directly, not through PnM register.**

5 CPU

SN8F5000 family is an enhanced 8051 microcontroller (MCU). It is fully compatible with MCS-51 instructions, hence the ability to cooperate with modern development environment (e.g. Keil C51). SN8F5000 CPU has 9.4 to 12.1 times faster than the original 8051 at the same frequency.

5.1 Memory Organization

SN8F5814 builds in three on-chip memories: internal RAM (IRAM), external RAM (XRAM), and program memory (IROM). The internal RAM is a 256-byte RAM which has higher access performance (direct and indirect addressing). By contrast, the external RAM has 512-byte of size, but it requires a longer access period. The program memory is a 16 KB non-volatile memory and has a maximum 8 MHz speed limitation.



5.2 Internal RAM (IRAM)

256 X 8-bit RAM (Internal Data Memory)

Address	RAM Location		
000h 01Fh	Work Register Area		00h-7Fh of RAM is direct and indirect access RAM
020h 02Fh	Bit Addressable Area		
030h … … 07Fh	General Purpose Area		
080h … … … 0FFh	General Purpose Area (Indirect Access)	Special Function Register (Direct Access)	080h-0FFh store special function registers. End of Bank 0

The 256-byte data RAM in internal data memory is a standard 8051 RAM access configuration. The upper 128-byte RAM is general purpose RAM and can configure by direct addressing access and indirect addressing access. The lower 128-byte can be indirect access RAM in general purpose or direct access RAM in special function register (SFR).

- 0x0000-0x007F: General purpose RAM contains work register area and bit addressable area. In this area, direct or indirect addressing can be used.
- 0x0000-0x001F: Work register area includes 4-bank. Each bank has 8 work registers (R0 - R7) which is selected by RS0/RS1 in PSW register.
- 0x0020-0x002F: Bit addressable area.

In the bit addressable area, user can read or write any single bit in this range by using the unique address for that bit. Supports 16bytes bit addressable RAM area giving 128 addressable bits. Each bit has individual address in the range from 00H to 7FH. Thus, the bit can be addressed directly. Bit0 of the byte 20H has bit address 00H and Bit 7 of the byte 20H has bit address 07H. Bit0 of the byte 2FH has bit address 78H and Bit 7 of the byte 2FH has bit address 7FH. When set “SETB 42H”, it means the bit2 of the byte 28H is set.

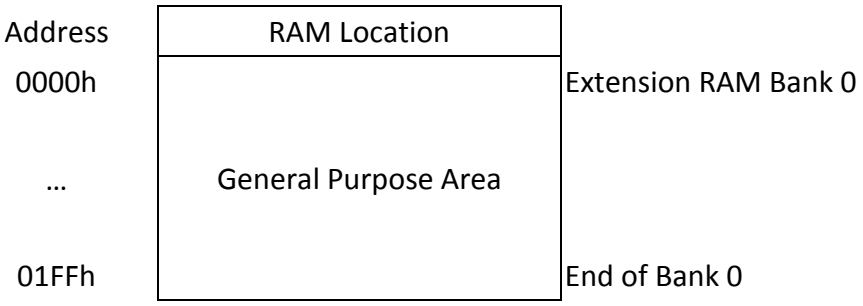
	Byte Address	Bite 0	Bite 1	Bite 2	Bite 3	Bite 4	Bite 5	Bite 6	Bite 7
Bit Addressable Area	**0x20**	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07
	0x21	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
	0x22	0x10	0x11	0x12	0x13	0x14	0x15	0x16	0x17
	0x23	0x18	0x19	0x1A	0x1B	0x1C	0x1D	0x1E	0x1F
	0x24	0x20	0x21	0x22	0x23	0x24	0x25	0x26	0x27
	0x25	0x28	0x29	0x2A	0x2B	0x2C	0x2D	0x2E	0x2F
	0x26	0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37
	0x27	0x38	0x39	0x3A	0x3B	0x3C	0x3D	0x3E	0x3F
	0x28	0x40	0x41	0x42	0x43	0x44	0x45	0x46	0x47
	0x29	0x48	0x49	0x4A	0x4B	0x4C	0x4D	0x4E	0x4F
	0x2A	0x50	0x51	0x52	0x53	0x54	0x55	0x56	0x57
	0x2B	0x58	0x59	0x5A	0x5B	0x5C	0x5D	0x5E	0x5F
	0x2C	0x60	0x61	0x62	0x63	0x64	0x65	0x66	0x67
	0x2D	0x68	0x69	0x6A	0x6B	0x6C	0x6D	0x6E	0x6F
	0x2E	0x70	0x71	0x72	0x73	0x74	0x75	0x76	0x77
	0x2F	0x78	0x79	0x7A	0x7B	0x7C	0x7D	0x7E	0x7F

- 0x0080~0x00FF: General purpose area in indirect addressing access or special function register in direct addressing access.

5.3 External RAM (XRAM)

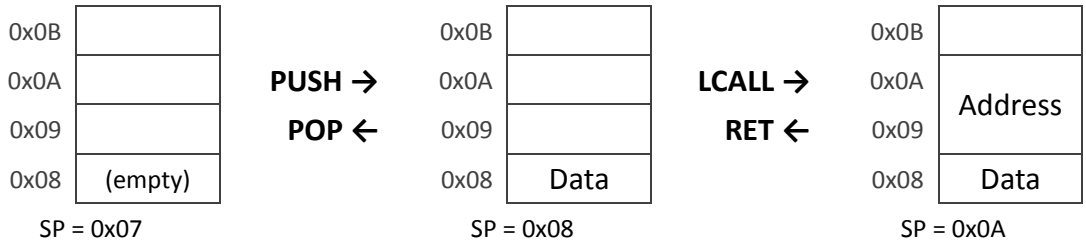
512 X 8-bit SRAM (Extension Data Memory)

The external RAM enlarges the capacity of variables; it is the lowest access performance in the contrast of internal RAM. Since frequently used variables and local variables are expected to store in internal RAM, the vast majority of external RAM usages are specific. It can be allocated as a variable storage area for lower priority tasks, or look-up table preloaded from ROM to speed up the access period.



5.4 Stack

Stack can be assigned to any area of internal RAM (IRAM). However, it requires manual assignment to ensure its area does not overlap other RAM's variables. An overflow or underflow stack could also mistakenly overwrite other RAM's variables; thus, these factors should be considered while arrange the size of stack.



By default, stack pointer (SP register) points to 0x07 which means the stack area begin at IRAM address 0x08. In other word, if a planned stack area is assigned from IRAM address 0xC0, the appropriate SP register is anticipated to set at 0xBF after system reset.

An assembly PUSH instruction costs one byte of stack. LCALL, ACALL instructions and interrupt respectively costs two bytes stack. POP-instruction decreases one count, and a RET/RETI subtract two counts of stack pointer.

✷ ***Note: Stack and IRAM share the same area, Keil C51 compiler will not display "error" or "warning" when overlap condition is occurred so user must pay attention.***

5.5 Program Memory (IROM)

The program memory is a non-volatile storage area where stores code, look-up ROM table, and other data with occasional modification. It can be updated by debug tools like SN-Link3, and a program can also self-update via in-system program process (refer to In-system Program).

Address	ROM	Comment
0000H	Reset vector	Reset vector
0001H 0002H	General purpose area	User program
0003H	**INT0 Interrupt vector**	**Interrupt vector**
000BH	**TIMER0 Interrupt vector**	
0013H	**INT1 Interrupt vector**	
001BH	**TIMER1 Interrupt vector**	
0023H	**UART Interrupt vector**	
002BH	**TIMER2 Interrupt vector**	
0043H	**I2C Interrupt vector**	
004BH	**SPI Interrupt vector**	
0053H	**T2COM0 Interrupt vector**	
005BH	**T2COM1 Interrupt vector**	
0063H	**T2COM2 Interrupt vector**	
006BH	**T2COM3 Interrupt vector**	
0083H	**INT2 Interrupt vector**	
008BH	**ADC Interrupt vector**	
0093H	**PWM1 Interrupt vector**	
00ECH	General purpose area	User program End of user program
3FF6H 3FF7H . 3FFDH 3FFEH 3FFFH	Reserved	

The ROM includes reset vector, Interrupt vector, general purpose area and reserved area. The reset vector is program beginning address. The interrupt vector is the head of interrupt service routine when any interrupt occurring. The general purpose area is main program area including main loop, sub-routines and data table.

- 0x0000 Reset vector: Program counter points to 0x0000 after any reset events (power on reset, reset pin reset, watchdog reset, LVD reset…).
- 0x0001~0x0002: General purpose area to process system reset operation.
- 0x0003~0x00EB: Multi interrupt vector area. Each of interrupt events has a unique interrupt vector.
- 0x00EC~0x3FDF: General purpose area for user program and ISP (EEPROM function).
- 0x3FE0~0x3FF6: General purpose area for user program. Do not execute ISP.
- 0x3FF6~0x3FFF: Reserved area. Do not execute ISP.

5.6 Program Memory Security

The SN8F5814 provides security options at the disposal of the designer to prevent unauthorized access to information stored in FLASH memory. When enable security option, the ROM code is secured and not dumped complete ROM contents. ROM security rule is all address ROM data protected and outputs 0x00.

5.7 Data Pointer

A data pointer helps to specify the XRAM and IROM address while performing MOVX and MOVC instructions. The microcontroller has one set of data pointer (DPH/DPL). The DPC register controls automatically increase/decrease DPTR function.

The automatically increase/decrease DPTR function can make an increment or decrement after perform MOVX @DPTR instruction. As a result, it enables a continuous external RAM access without re-specified DPTR value.

5.8 Stack and Data Pointer Register

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SP	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0
DPL	DPL7	DPL6	DPL5	DPL4	DPL3	DPL2	DPL1	DPL0
DPH	DPH7	DPH6	DPH5	DPH4	DPH3	DPH2	DPH1	DPH0
DPC	-	-	-	-	-	ATMS	ATMD	ATME

SP Register (0x81)

Bit	Field	Type	Initial	Description
7..0	SP	R/W	0x07	Stack pointer

DPL Register (0x82)

Bit	Field	Type	Initial	Description
7..0	DPL[7:0]	R/W	0x00	Low byte of DPTR0

DPH Register (0x83)

Bit	Field	Type	Initial	Description
7..0	DPH[7:0]	R/W	0x00	High byte of DPTR0

DPC Register (0x93)

Bit	Field	Type	Initial	Description
7..3	Reserved	R	0x0	
2..1	ATMS/ATMD	R/W	00	Automatically increase/decrease DPTR (if ATME applied) 00: +1 after any MOVX @DPTR instruction 01: -1 after any MOVX @DPTR instruction 10: +2 after any MOVX @DPTR instruction 11: -2 after any MOVX @DPTR instruction
0	ATME	R/W	0	Automatically increase/decrease DPTR function 0: Disable 1: Enable

6 Special Function Registers

6.1 Special Function Register Memory Map

BIN / HEX	000	001	010	011	100	101	110	111
F8	-	P0M	P1M	P2M	P3M	-	-	PFLAG
F0	B	P0UR	P1UR	P2UR	P3UR	-	-	SRST
E8	-	-	-	-	-	-	-	-
E0	ACC	SPSTA	SPCON	SPDAT	P2OC	CLKSEL	CLKCMD	TCON0
D8	S0CON2	-	I2CDAT	I2CADR	I2CCON	I2CSTA	SMBSEL	SMBDST
D0	PSW	-	ADM	ADB	ADR	VREFH	P2CON	P3CON
C8	T2CON	T2CON0	CRCL	CRCH	TL2	TH2	-	-
C0	IRCON	CCEN	CCL1	CCH1	CCL2	CCH2	CCL3	CCH3
B8	IEN1	IP1	S0RELH	PW1DL	PW1DH	PW1A	PW1CH	IRCON2
B0	P3	-	-	-	-	-	-	PWFLM
A8	IEN0	IP0	S0RELL	PW1M	PW1YL	PW1YH	PW1BL	PW1BH
A0	P2	LCDCON	LCOMOD	LCDSEG	LCDBUF	LCDADR	LCDSEG1	-
98	S0CON	S0BUF	IEN2	-	-	-	P0CON	P1CON
90	P1	P1W	-	DPC	PECMD	PEROML	PEROMH	PERAM
88	TCON	TMOD	TL0	TL1	TH0	TH1	CKCON	PEDGE
80	P0	SP	DPL	DPH	-	-	WDTR	PCON

✷ ***Note: All SFRs in the left-most column are bit-addressable. (Every 0x0/0x8-ending SFR addresses are bit-addressable).***

6.2 Special Function register Description

0x80 - 0x9F Registers Description

Register	Address	Description
P0	0x80	Port 0 data buffer.
SP	0x81	Stack pointer register.
DPL	0x82	Data pointer low byte register.
DPH	0x83	Data pointer high byte register.
-	0x84	-
-	0x85	-
WDTR	0x86	Watchdog timer clear register.
PCON	0x87	System mode register.
TCON	0x88	Timer 0 / 1 controls register.
TMOD	0x89	Timer 0 / 1 mode register.
TL0	0x8A	Timer 0 counting low byte register.
TL1	0x8B	Timer 1 counting low byte register.
TH0	0x8C	Timer 0 counting high byte register.
TH1	0x8D	Timer 1 counting high byte register.
CKCON	0x8E	Extended cycle controls register.
PEDGE	0x8F	External interrupt edge controls register.
P1	0x90	Port 1 data buffer.
P1W	0x91	Port 1 wake-up controls register.
-	0x92	-
DPC	0x93	Data pointer controls register.
PECMD	0x94	In-System Program command register.
PEROML	0x95	In-System Program ROM address low byte
PEROMH	0x96	In-System Program ROM address high byte
PERAM	0x97	In-System Program RAM mapping address
S0CON	0x98	UART control register.
S0BUF	0x99	UART data buffer.
IEN2	0x9A	Interrupts enable register
-	0x9B	-
-	0x9C	-
-	0x9D	-
P0CON	0x9E	Port 0 configuration controls register.
P1CON	0x9F	Port 1 configuration controls register.

0xA0 - 0xBF Registers Description

Register	Address	Description
P2	0xA0	Port 2 data buffer
LCDCON	0xA1	LCD controls register.
LCDMOD	0xA2	LCD mode control.
LCDSEG	0xA3	LCD segment pin control.
LCDBUF	0xA4	LCD data buffer.
LCDADR	0xA5	LCD data buffer address.
LCDSEG1	0xA6	LCD segment pin control.
-	0xA7	-
IEN0	0xA8	Interrupts enable register
IP0	0xA9	Interrupts priority register.
S0RELL	0xAA	UART reload low byte register.
PW1M	0xAB	PW1 controls register.
PW1YL	0xAC	PW1 cycle controls buffer low byte.
PW1YH	0xAD	PW1 cycle controls buffer high byte.
PW1BL	0xAE	PW1 B point dead band controls buffer low byte.
PW1BH	0xAF	PW1 B point dead band controls buffer high byte.
P3	0xB0	Port 3 data buffer.
-	0xB1	-
-	0xB2	-
-	0xB3	-
-	0xB4	-
-	0xB5	-
-	0xB6	-
PWFLM	0xB7	PWM fail detect control register.
IEN1	0xB8	Interrupts enable register
IP1	0xB9	Interrupts priority register.
S0RELH	0xBA	UART reload high byte register.
PW1DL	0xBB	PW1 duty controls buffer low byte.
PW1DH	0xBC	PW1 duty controls buffer high byte.
PW1A	0xBD	PW1 A point dead band controls buffer.
PW1CH	0xBE	PW1 channel enable register.
IRCON2	0xBF	Interrupts request register.

0xC0 - 0xDF Registers Description

Register	Address	Description
IRCON	0xC0	Interrupts request register.
CCEN	0xC1	Timer 2 Compare /capture enable register.
CCL1	0xC2	Timer 2 Compare /capture module 1 low byte register.
CCH1	0xC3	Timer 2 Compare /capture module 1 high byte register.
CCL2	0xC4	Timer 2 Compare /capture module 2 low byte register.
CCH2	0xC5	Timer 2 Compare /capture module 2 high byte register.
CCL3	0xC6	Timer 2 Compare /capture module 3 low byte register.
CCH3	0xC7	Timer 2 Compare /capture module 3 high byte register.
T2CON	0xC8	Timer 2 controls register.
T2CON0	0xC9	Timer 2 high clock source control register.
CRCL	0xCA	Timer 2 Compare/capture module 0 & reload function low byte register.
CRCH	0xCB	Timer 2 Compare/capture module 0 & reload function high byte register.
TL2	0xCC	Timer 2 counting low byte register.
TH2	0xCD	Timer 2 counting high byte register.
-	0xCE	-
-	0xCF	-
PSW	0xD0	System flag register.
-	0xD1	-
ADM	0xD2	ADC controls register.
ADB	0xD3	ADC data buffer.
ADR	0xD4	ADC resolution selects register.
VREFH	0xD5	ADC reference voltage controls register.
P2CON	0xD6	Port 2 configuration controls register.
P3CON	0xD7	Port 3 configuration controls register.
S0CON2	0xD8	UART baud rate controls register.
-	0xD9	-
I2CDAT	0xDA	I2C data buffer.
I2CADR	0xDB	Own I2C slave address.
I2CCON	0xDC	I2C interface operation control register.
I2CSTA	0xDD	I2C Status Code.
SMBSEL	0xDE	SMBus mode controls register.
SMBDST	0xDF	SMBus internal timeout register.

0xE0 - 0xFF Registers Description

Register	Address	Description
ACC	0xE0	Accumulator register.
SPSTA	0xE1	SPI statuses register.
SPCON	0xE2	SPI control register.
SPDAT	0xE3	SPI data buffer.
P2OC	0xE4	Open drain controls register.
CLKSEL	0xE5	Clock switch selects register.
CLKCMD	0xE6	Clock switch controls Register.
TCON0	0xE7	Timer 0 / 1 clock controls register.
-	0xE8	-
-	0xE9	-
-	0xEA	-
-	0xEB	-
-	0xEC	-
-	0xED	-
-	0xEE	-
-	0xEF	-
B	0xF0	Multiplication/ division instruction data buffer.
P0UR	0xF1	Port 0 pull-up resister controls register.
P1UR	0xF2	Port 1 pull-up resister controls register.
P2UR	0xF3	Port 2 pull-up resister controls register.
P3UR	0xF4	Port 3 pull-up resister controls register.
-	0xF5	-
-	0xF6	-
SRST	0xF7	Software reset controls register.
-	0xF8	-
P0M	0xF9	Port 0 input/output mode register.
P1M	0xFA	Port 1 input/output mode register.
P2M	0xFB	Port 2 input/output mode register.
P3M	0xFC	Port 3 input/output mode register.
-	0xFD	-
-	0xFE	-
PFLAG	0xFF	Reset flag register.

6.3 System Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ACC	ACC7	ACC6	ACC5	ACC4	ACC3	ACC2	ACC1	ACC0
B	B7	B6	B5	B4	B3	B2	B1	B0
PSW	CY	AC	F0	RS1	RS0	OV	F1	P

ACC Register (0xE0)

Bit	Field	Type	Initial	Description
7..0	ACC[7:0]	R/W	0x00	The ACC is an 8-bit data register responsible for transferring or manipulating data between ALU and data memory. If the result of operating is overflow (OV) or there is carry (C or AC) and parity (P) occurrence, then these flags will be set to PSW register.

B Register (0xF0)

Bit	Field	Type	Initial	Description
7..0	B[7:0]	R/W	0x00	The B register is used during multiplying and division instructions. It can also be used as a scratch-pad register to hold temporary data.

PSW Register (0xD0)

Bit	Field	Type	Initial	Description
7	CY	R/W	0	Carry flag. 0: Addition without carry, subtraction with borrowing signal, rotation with shifting out logic “0”, comparison result < 0. 1: Addition with carry, subtraction without borrowing, rotation with shifting out logic “1”, comparison result ≥ 0.
6	AC	R/W	0	Auxiliary carry flag. 0: If there is no a carry-out from 3rd bit of Accumulator in BCD operations. 1: If there is a carry-out from 3rd bit of Accumulator in BCD operations.
5	F0	R/W	0	General purpose flag 0. General purpose flag available for user.
4..3	RS[1:0]	R/W	00	Register bank select control bit, used to select working register bank. 00: 00H – 07H (Bnak0) 01: 08H – 0FH (Bnak1) 10: 10H – 17H (Bnak2) 11: 18H – 1FH (Bnak3)
2	OV	R/W	0	Overflow flag. 0: Non-overflow in Accumulator during arithmetic Operations. 1: overflow in Accumulator during arithmetic Operations.
1	F1	R/W	0	General purpose flag 1. General purpose flag available for user.
0	P	R	0	Parity flag. Reflects the number of ‘1’s in the Accumulator. 0: if Accumulator contains an even number of ‘1’s. 1: Accumulator contains an odd number of ‘1’s.

6.4 Register Declaration

SN8F5814 has many registers to control various functions, but SFR name is not predefined in the C51 / A51 compiler. To make programming easier and therefore need to add header files to declare SFR name.

When using the assembly code programs, please add the following sentence.

```
1 $NOMOD51 ;Do not recognize the 8051-specific predefined special register.
2 #include <SN8F5814.H>
```

When using the C code programs, please add the following sentence.

```
1 #include <SN8F5814.H>
```

After adding the header file, user can use name of registers to program. During compilation, the compiler will register name translate into register position through the header file.

Different devices need to use a different header file to declare, but the option file is to use the same.

Device	Header file	Options file
SN8F5814	SN8F5814.h	OPTIONS_SN8F5814.A51
SN8F5813	SN8F5813.h	OPTIONS_SN8F5814.A51
SN8F5812	SN8F5812.h	OPTIONS_SN8F5814.A51

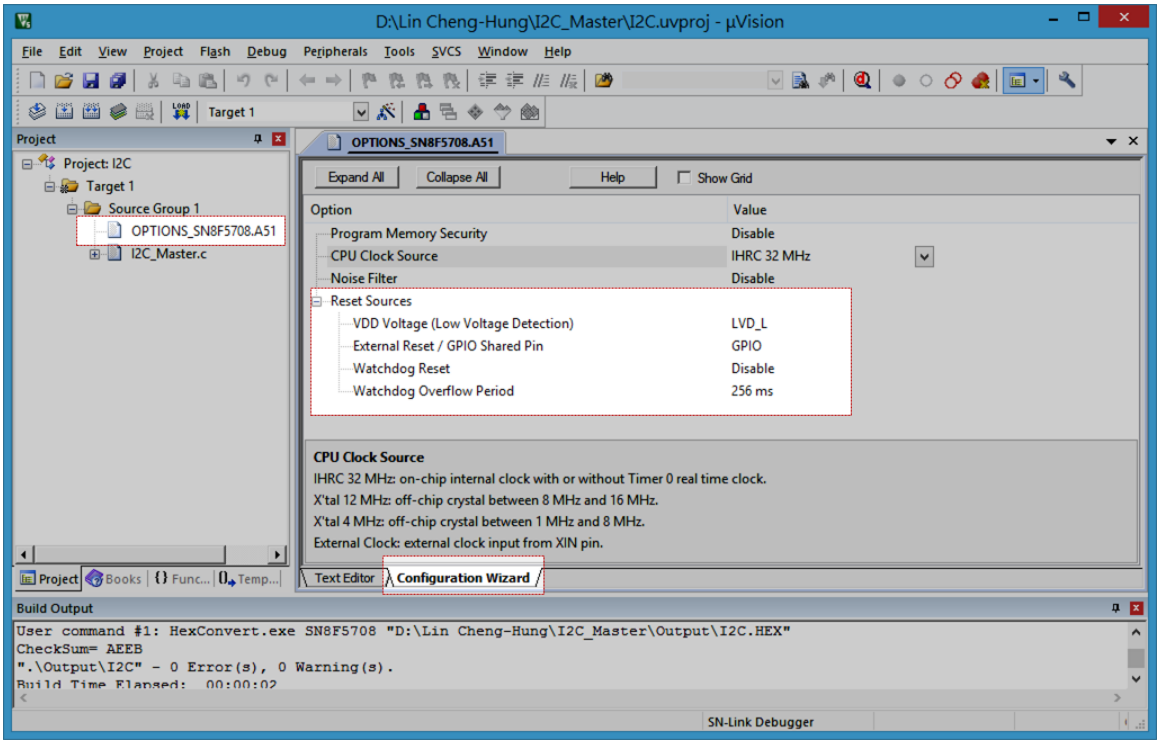
7 Reset and Power-on Controller

The reset and power-on controller has four reset sources: low voltage detectors (LVDs), watchdog, programmable external reset pin, and software reset. The first three sources would trigger an additional power-on sequence. Subsequently, the microcontroller initializes all registers and starts program execution with its reset vector (ROM address 0x0000).

7.1 Configuration of Reset and Power-on Controller

SONiX publishes a *SN8F5814_OPTIONS.A51 file* in *SN-Link Driver for Keil C51.exe* (downloadable on cooperative website: www.sonix.com.tw). This *options file* contains appropriate parameters of reset sources and CPU clock source selection, and is strongly recommended to add to Keil project. *SN8F5000 Debug Tool Manual* provides the further detail of this configuration. The option items are as following:

- Program Memory Security
- CPU Clock Source
- Noise Filter
- Reset Source : VDD Voltage (Low Voltage Detection)
- Reset Source : External Reset / GPIO Shared Pin
- Reset Source : Watchdog Reset & Overflow Period

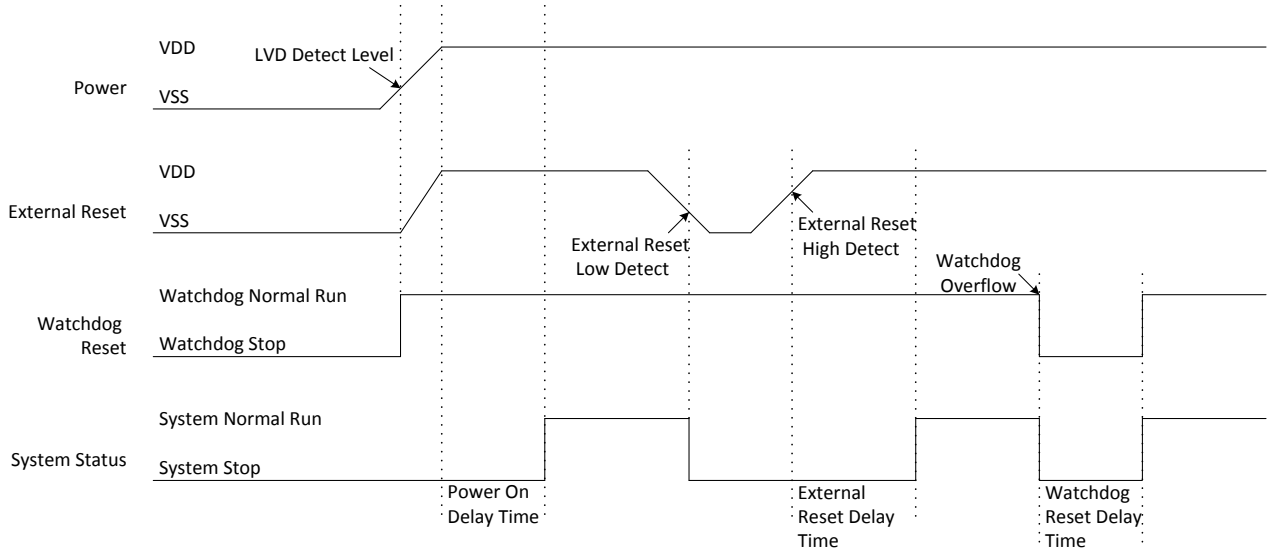


The code option is the system hardware configurations including oscillator type, noise filter option, watchdog timer operation, LVD option, reset pin option and flash ROM security control. The code option items are as following table:

Code Option	Content	Function Description
Program Memory Security	Disable	Disable ROM code Security function
	Enable	Enable ROM code Security function
CPU Clock Source	IHRC 32MHz	High speed internal 32MHz RC. XIN/XOUT pins are bi-direction GPIO mode
	X'tal 12MHz	High speed crystal /resonator (e.g. 12MHz) for external high clock oscillator
	X'tal 4MHz	Standard crystal /resonator (e.g. 4M) for external high clock oscillator
	External Clock	XIN pin connect external clock (1M ~32M), XOUT pin is bi-direction GPIO mode
Noise Filter	Disable	Disable Noise Filter
	Enable	Enable Noise Filter
LVD	LVD_L	LVD will reset chip if VDD is below 1.8V
External Reset	Reset with De-bounce	Enable External reset pin with De-bounce
	Reset without De-bounce	Enable External reset pin without De-bounce
	GPIO with P04	Enable P04
Watchdog Reset	Always	Watchdog timer is always on enable even in STOP mode and IDLE mode
	Enable	Enable watchdog timer. Watchdog timer stops in STOP mode and IDLE mode
	Disable	Disable Watchdog function
Watchdog Overflow Period	64ms	Watchdog timer clock source F_{ILRC} /4
	128ms	Watchdog timer clock source F_{ILRC} /8
	256ms	Watchdog timer clock source F_{ILRC} /16
	512ms	Watchdog timer clock source F_{ILRC} /32

7.2 Power-on Sequence

A power-on sequence would be triggered by LVD, watchdog, and external reset pin. It takes place between the end of reset signal and program execution. Overall, it includes two stages: power stabilization period, and clock stabilization period.

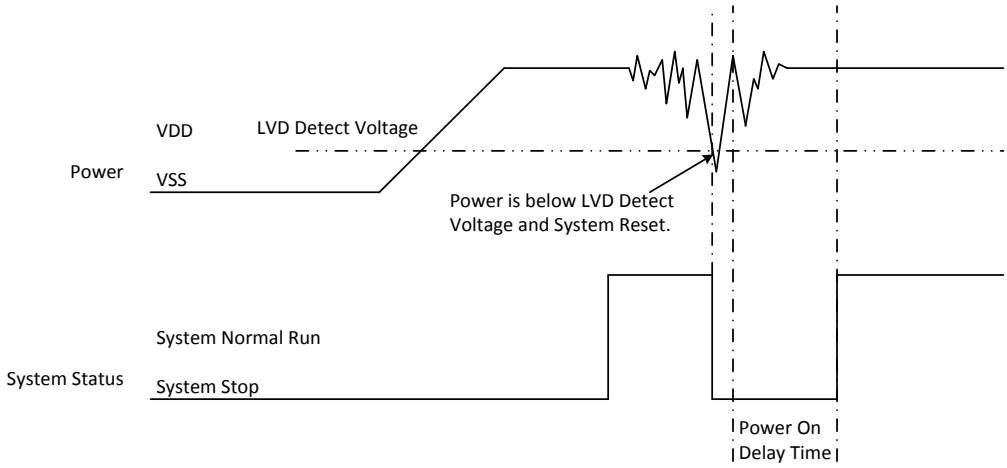


The power stabilization period spends 4.6 ms in typical condition. Afterward the microcontroller fetches CPU Clock Source selection automatically. The selected clock source would be driven, and the system counts 2048 times of the clock period and 5 times of the internal low-speed oscillator clocks to ensure its reliability.

✷ ***Note: In high power noise environment, user can put 10ohm resistor in the front of 0.1uF capacitor & VDD PAD to suppress power noise and avoid IC damage.***

7.3 LVD Reset

The low voltage detectors monitor VDD pin's voltage at only one level: 1.8 V. Depend on low voltage detection configuration, the comparison result can be seen as a system reset signal. The table below lists low voltage detection configuration, LVD_L, and the results of VDD pin's condition.



Condition	LVD_L
VDD ≤ 1.8 V	Reset

7.4 Watchdog Reset

Watchdog is a periodic reset signal generator for the purpose of monitoring the execution flow. Its internal timer is expected to be cleared in a check point of program flow; therefore, the actual reset signal would be generated only after a software problem occurs. Writing 0x5A to WDTR is the proper method to place a check point in program.

```
1     WDTR = 0x5A;
```

Watchdog timer interval time = 256 * 1/ (Internal Low-Speed oscillator frequency/WDT Pre-scalar)
= 256 / (F_{ILRC} /WDT Pre-scaler) …sec

Internal low-speed oscillator	WDT pre-scaler	Watchdog interval time
F_{ILRC} =16 kHz	F_{ILRC} /4	256/(16000/4)=64ms
	F_{ILRC} /8	256/(16000/8)=128ms
	F_{ILRC} /16	256/(16000/16)=256ms
	F_{ILRC} /32	256/(16000/32)=512ms

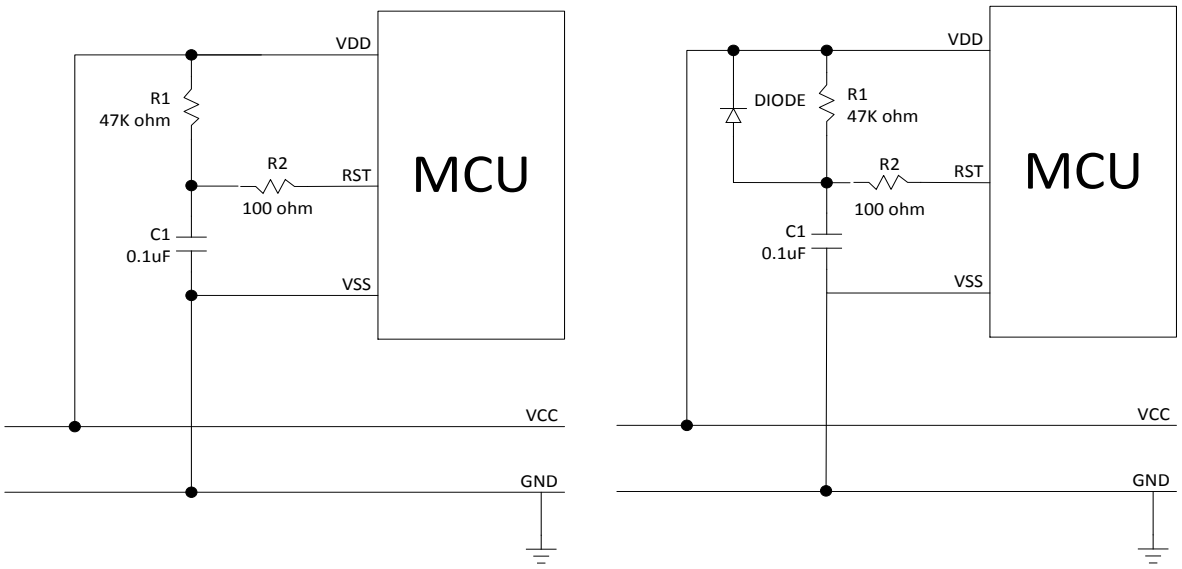
The operation mode of watchdog is configurable in options file:
Always mode counts its internal timer in all CPU operation modes (normal, IDLE, SLEEP);
Enable mode counts its internal timer during CPU stays in normal mode, and it would not trigger watchdog reset in IDLE and STOP modes;
Disable mode suspends its internal timer at all CPU modes, and the watchdog would not trigger in this condition.

When watchdog is operating in always mode, the system will consume additional power.

7.5 External Reset Pin

Programmable external reset pin is configurable in *options file*. Once it is enabled, it monitors its shared pin's logic level. A logical low (lower than 30% of VDD) would immediately trigger system reset until the input is recovered to high (lager than 70% of VDD).

An optional de-bounce period can improve reset signal's stability. Instead of immediate reset, the system reset requires an 8-ms-long logic low to avoid bouncing from a button key. Any signal lower than de-bounce period would not affect the CPU's execution.



✸ ***Note:***

1. The reset circuit is no any protection against unusual power or brown out reset on the left side of the figure.

2. The R2 100 ohm resistor of "Simply reset circuit" and "Diode & RC reset circuit" is necessary to limit any current flowing into reset pin from external capacitor C in the event of reset pin breakdown due to Electrostatic Discharge (ESD) or Electrical Over-stress (EOS) on the right side of the figure.

7.6 Software Reset

A software reset would be generated after consecutively set SRSTREQ register. As a result, this procedure enables firmware's ability to reset microcontroller (e.g. reset after firmware update). The following sample C code repeatedly set the least bit of SRST register to perform software reset.

```
1    SRST = 0x01;
2    SRST = 0x01;
```

7.7 Reset and Power-on Controller Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PFLAG	POR	WDT	RST	WARM	-	-	-	-
SRST	-	-	-	-	-	-	-	SRSTREQ
WDTR	WDTR7	WDTR6	WDTR5	WDTR4	WDTR3	WDTR2	WDTR1	WDTR0

PFLAG Register

Bit	Field	Type	Initial	Description
7	POR	R	-	This bit is automatically set if the microcontroller has been reset by LVD.
6	WDT	R	-	This bit is automatically set if the microcontroller has been reset by watchdog.
5	RST	R	-	This bit is automatically set if the microcontroller has been reset by external reset pin.
4	WARM	R	0	This bit is automatically set if the Ext. 32.768kHz crystal warm up is ready.
3..0	Reserved	R	0	

SRST Register

Bit	Field	Type	Initial	Description
7..1	Reserved	R	0	
0	SRSTREQ	R/W	-	Read: This bit is automatically set if the microcontroller has been reset by software reset. Write: Consecutively set this bit for two times to trigger software reset.

WDTR Register (0x86)

Bit	Field	Type	Initial	Description
7..0	WDTR[7:0]	W	-	Watchdog clear is controlled by WDTR register. Moving 0x5A data into WDTR is to reset watchdog timer.

8 System Clock and Power Management

For power saving purpose, the microcontroller built in three different operation modes: normal, IDLE, and STOP mode.

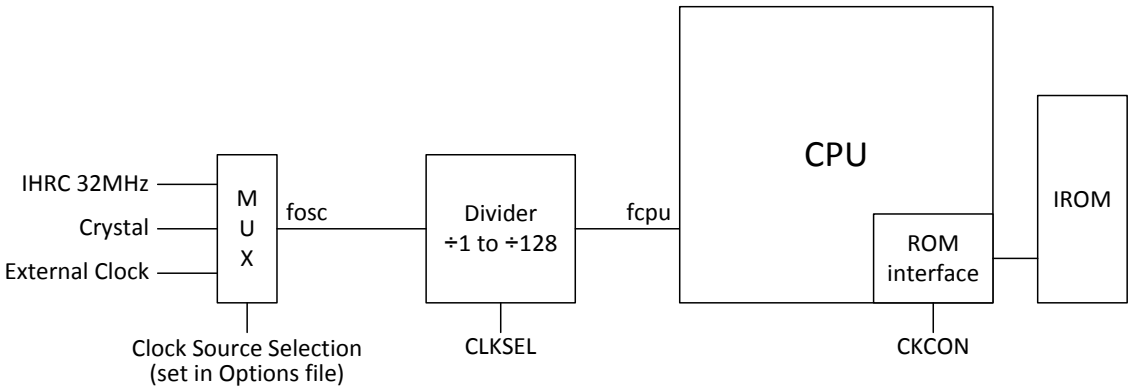
The normal mode means that CPU and peripheral functions are under normally execution. The system clock is based on the combination of source selection, clock divider, and program memory wait state. IDLE mode is the situation that temporarily suspends CPU clock and its execution, yet it remains peripherals' functionality (e.g. timers, PWM, SPI, UART, and I2C). STOP mode disables all functions and clock generator until a wakeup signal to return normal mode.

8.1 System Clock

The microcontroller includes an on-chip clock generator (IHRC 32MHz), crystal/resonator driver, and an external clock input. The reset and power-on controller automatically loads clock source selection during power-on sequence. Therefore, the selected clock source is seen as 'fosc' domain which is a fixed frequency at any time.

Subsequently, the selected clock source (fosc) is divided by 1 to 128 times which is controlled by CLKSEL register. The CPU input the divided clock as its operation base (named fcpu). Applying CLKSEL's setting when CLKCMD register be written 0x69.

```
CKCON = 0x70;     // For change safely the system clock
CLKSEL = 0x85;    // set fcpu = fosc / 4
CLKCMD = 0x69;    // Apply CLKSEL's setting
CKCON = 0x00;     // IROM fetch = fcpu / 1
```



ROM interface is built in between CPU and IROM (program memory). It optionally extends the data fetching cycle in order to support lower speed program memory.

IROM fetching cycle (Instruction cycle) ≦ 8MHz

✷ ***Note: For user develop program in C language or assembly, the first line of the program "must be set" CKCON = 0x70, CLKSEL = 0x87~0x80, CLKMD = 0x69 and then set CKCON = 0x00~0x70, this priority cannot be modified.***

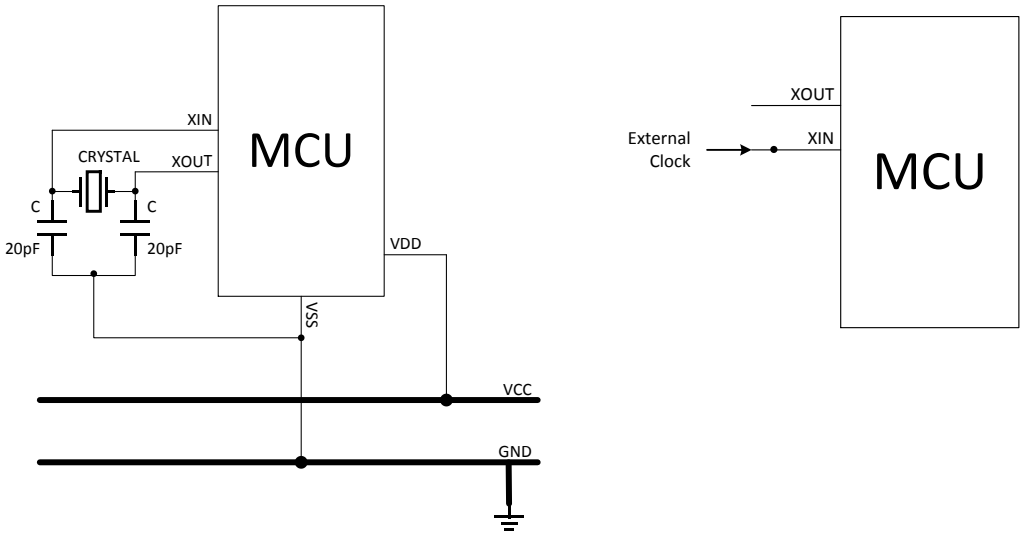
System clock rate and program memory extended cycle limitation as follows.

Code Option CPU Clock Source	Fcpu = CLKSEL[2:0]	IROM Fetch = CKCON[6:4]
IHRC 32M External Clock (16-32MHz)	**Only Support** 000 = fosc / 128 001 = fosc / 64 010 = fosc / 32 011 = fosc / 16 100 = fosc / 8 101 = fosc / 4	**Support** **000 = fcpu / 1 => Recommend!** 001 = fcpu / 2 010 = fcpu / 3 011 = fcpu / 4 100 = fcpu / 5 101 = fcpu / 6 110 = fcpu / 7 111 = fcpu / 8
X'tal 12M (Crystal 8-16MHz) External Clock (8-16MHz)	**Only Support** 000 = fosc / 128 001 = fosc / 64 010 = fosc / 32 011 = fosc / 16 100 = fosc / 8 101 = fosc / 4 110 = fosc / 2	
X'tal 12M (Crystal 4-8MHz) X'tal 4M (Crystal 1-4MHz) External Clock (1-8MHz)	**Support** 000 = fosc / 128 001 = fosc / 64 010 = fosc / 32 011 = fosc / 16 100 = fosc / 8 101 = fosc / 4 110 = fosc / 2 111 = fosc / 1	

8.2 High Speed Clock

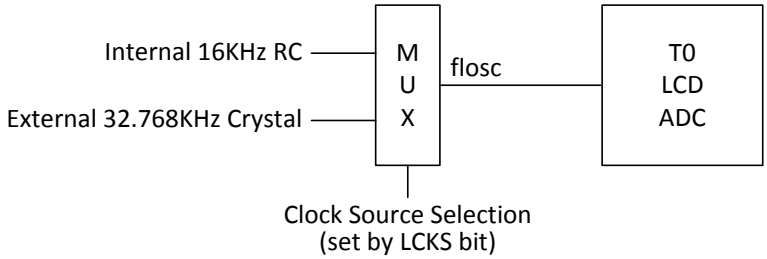
High-speed clock has internal and external two-type. The external high-speed clock includes 4MHz, 12MHz crystal/ceramic and external clock input mode. The internal high-speed oscillator is 32MHz RC type. These high-speed oscillators are selected by *SN8F5814_OPTIONS.A51*.

- IHRC 32M: The system high-speed clock source is internal high-speed 32MHz RC type oscillator. In the mode, XIN and XOUT pins are bi-direction GPIO mode, and not to connect any external oscillator device.
- X'tal 12M: The system high-speed clock source is external high-speed crystal/ceramic. The oscillator bandwidth is 4MHz~16MHz and connected to XIN/XOUT pins with 20pF capacitors to ground.
- X'tal 4M: The system high-speed clock source is external high-speed crystal/resonator. The oscillator bandwidth is 1MHz~4MHz and connected to XIN/XOUT pins with 20pF capacitors to ground.
- External Clock: The system high-speed clock source is external clock input mode. The input signal only connects to XIN pin, and the XOUT pin is bi-direction GPIO mode.



8.3 Low Speed Clock

SN8F5814 supplies low speed clock (flosc) for specific functions, such as T0, LCD and ADC. The flosc has two clock sources selection: internal 16KHz RC (ILRC) and external 32.768KHz crystal, which is controlled by LCKS bit.



In external 32.768 kHz crystal mode, LXIN and LXOUT pin switch to crystal mode to drive an off-chip 32.768 kHz crystal after LCKS bit is cleared. Microcontroller supplies WARM bit (the 4th bit of PFLAG) to indicate crystal warm-up status. The WARM bit is set when crystal warm-up procedure ready and cleared when crystal stop. If 32.768 kHz crystal is abnormal stop, re-clear LCKS bit can re-start crystal warm-up procedure.

8.4 Noise Filter

The Noise Filter controlled by Noise Filter option is a low pass filter and supports crystal mode. The purpose is to filter high rate noise coupling on high clock signal from external oscillator. In high noisy environment, enable Noise Filter option is the strongly recommendation to reduce noise effect.

8.5 Power Management

After the end of reset signal and power-on sequence, the CPU starts program execution at the speed of fcpu. Overall, the CPU and all peripherals are functional in this situation (categorized as normal mode).

The least two bits of PCON register (IDLE at bit 0 and STOP at bit 1) control the microcontroller's power management unit.

If IDLE bit is set by program, only CPU clock source would be gated. Consequently, peripheral functions (such as timers, PWM, and I2C) and clock generator (IHRC 32 MHz/crystal driver) remain execution in this status. Any change from P0/P1 input and interrupt events can make the microcontroller turns back to normal mode, and the IDLE bit would be cleared automatically.

If STOP bit is set, by contrast, CPU, peripheral functions, and clock generator are suspended. Data storage in registers and RAM would be kept in this mode. Any change from P0/P1 can wake up the microcontroller and resume system's execution. STOP bit would be cleared automatically.

For user who is develop program in C language, IDLE and STOP macros is strongly recommended to control the microcontroller's system mode, instead of set IDLE and STOP bits directly.

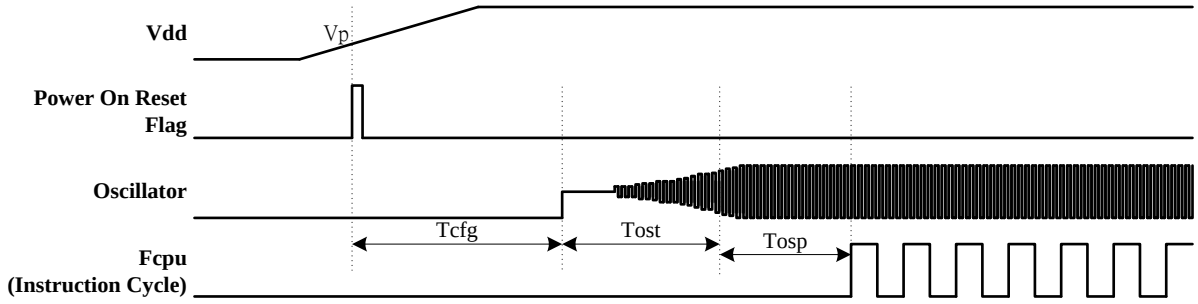
```
1     IDLE();
2     STOP();
```

SN8F5814 build in STWK bit (the 4th bit in PCON register) to enable or disable flosc clock in STOP mode. If STWK=0, both fosc and flosc are suspended in STOP mode. If STWK=1, flosc clock keeps running in STOP mode.

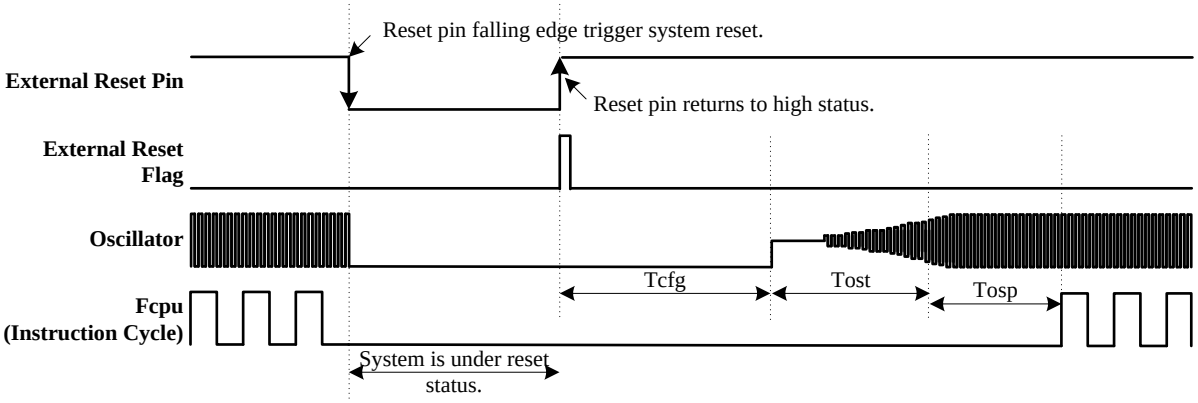
8.6 System Clock Timing

Parameter	Symbol	Description	Typical
Hardware configuration time	Tcfg	8* F_{ILRC} +2^{17}*F_{IHRC}	4.6ms @ F_{ILRC} = 16kHz & F_{IHRC} = 32MHz
Oscillator start up time	Tost	The start-up time is depended on oscillator's material, factory and architecture. Normally, the low-speed oscillator's start-up time is lower than high-speed oscillator. The RC type oscillator's start-up time is faster than crystal type oscillator.	-
Oscillator warm-up time	Tosp	Oscillator warm-up time of reset condition. 2048*F_{hosc} + 5*F_{ILRC} (Power on reset, LVD reset, watchdog reset, external reset pin active.)	825us @ F_{hosc} = 4MHz 441us @ F_{hosc} = 16MHz 377us @ F_{hosc} = 32MHz
		Oscillator warm-up time of power down mode wake-up condition. 2048*F_{hosc} + 5*F_{ILRC} ……Crystal/resonator type oscillator, e.g. 32768Hz crystal, 4MHz crystal, 16MHz crystal… 64*F_{hosc} + 5*F_{ILRC}……RC type oscillator, e.g. internal high-speed RC type oscillator.	X'tal: 825us @ F_{hosc} = 4MHz 441us @ F_{hosc} = 16MHz RC: 315us @ F_{hosc} = 32MHz

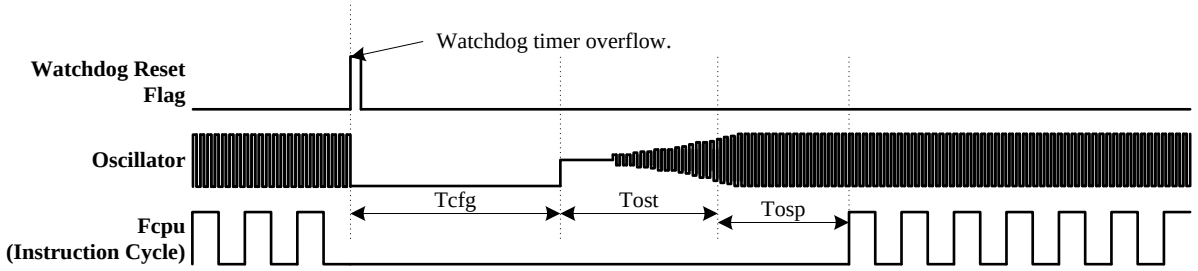
- Power On Reset Timing



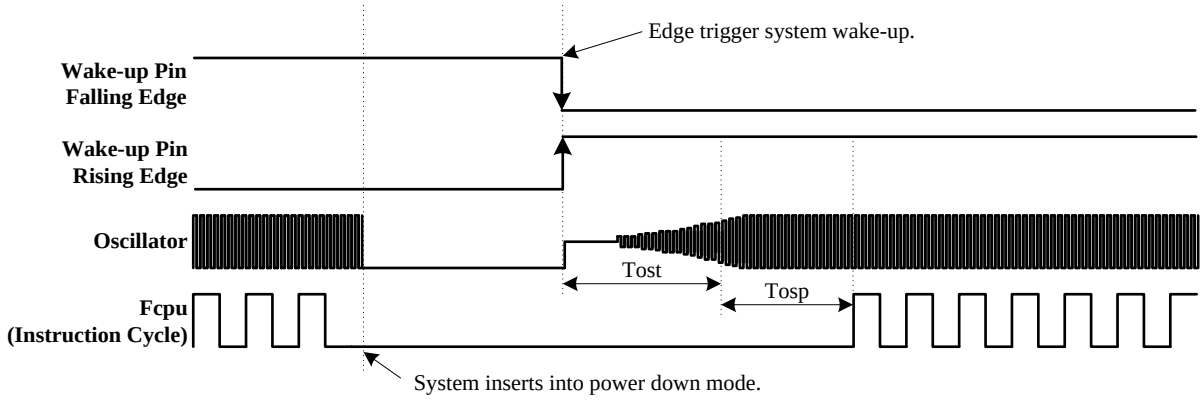
- External Reset Pin Reset Timing



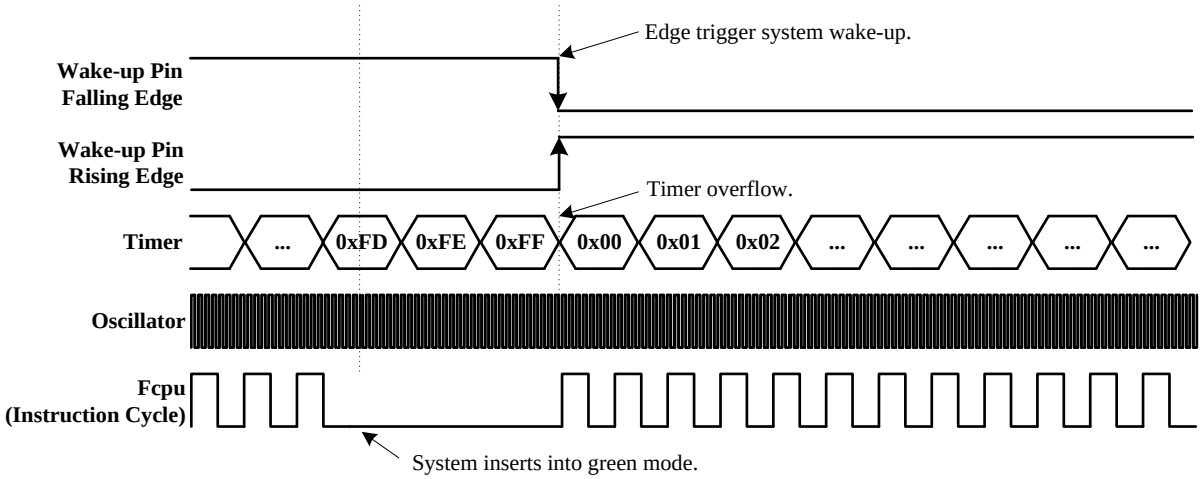
- Watchdog Reset Timing



- STOP Mode Wake-up Timing

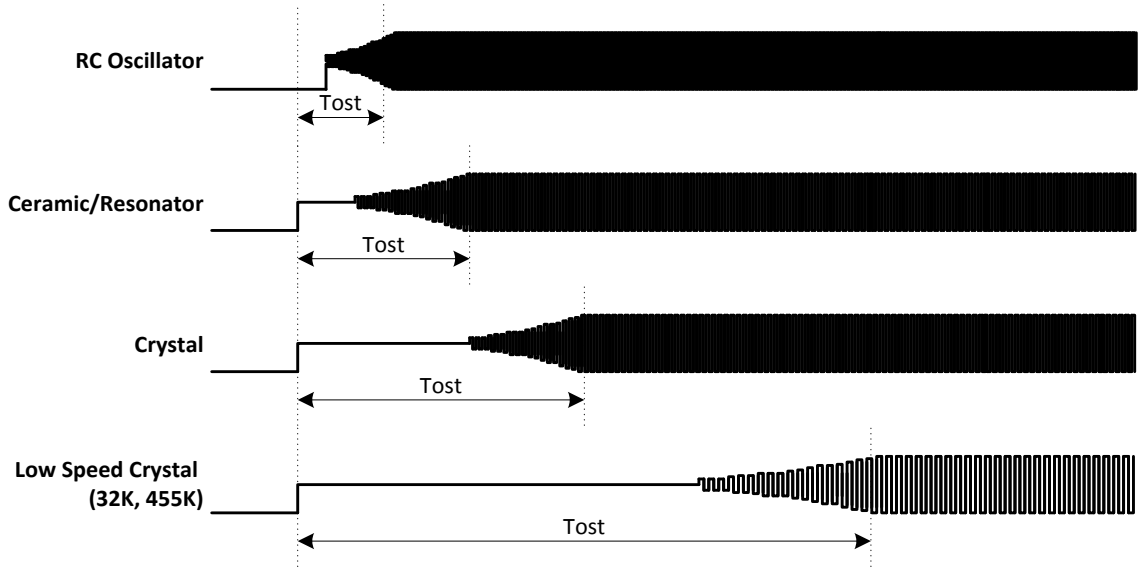


- IDLE Mode Wake-up Timing



- Oscillator Start-up Time

The start-up time is depended on oscillator's material, factory and architecture. Normally, the low-speed oscillator's start-up time is lower than high-speed oscillator.



8.7 System Clock and Power Management Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CKCON	-	PWSC2	PWSC1	PWSC0	ESYN	EWSC2	EWSC1	EWSC0
CLKSEL	LCKS	-	-	-	-	CLKSEL2	CLKSEL1	CLKSEL0
CLKCMD	CMD7	CMD6	CMD5	CMD4	CMD3	CMD2	CMD1	CMD0
PCON	SMOD	-	-	STWK	P2SEL	GF0	STOP	IDLE
P1W	P17W	P16W	P15W	P14W	P13W	P12W	P11W	P10W

CKCON Register (0x8E)

Bit	Field	Type	Initial	Description
7	Reserved	R	0	
6..4	PWSC[2:0]	R/W	111	Extended cycle(s) applied to reading program memory 000: non 001: 1 cycle 010: 2 cycles 011: 3 cycles 100: 4 cycles 101: 5 cycles 110: 6 cycles 111: 7 cycles
3	ESYN	R/W	0	Extended extra cycles to write XRAM
2..0	EWSC[2:0]	R/W	001	Extended cycle(s) applied to reading XRAM 000: non 001: 1 cycle 010: 2 cycles 011: 3 cycles 100: 4 cycles 101: 5 cycles 110: 6 cycles 111: 7 cycles

CLKCMD Register (0xE6)

Bit	Field	Type	Initial	Description
7..0	CMD[7:0]	W	0x00	Writing 0x69 to apply CLKSEL's setting.

P1W Register (0x91)

Bit	Field	Type	Initial	Description
7..0	P1nW	R/W	0	0: Disable P1.n wakeup functionality 1: Enable P1.n wakeup functionality

CLKSEL Register (0xE5)

Bit	Field	Type	Initial	Description
7	LCKS	R/W	1	Low clock source (flosc) select bit 0: flosc = External 32.768KHz crystal and LXIN/LXOUT pins drive 32.768kHz crystal. 1: flosc = Internal 16KHz RC (ILRC).
6..3	Reserved	R	0x00	
2..0	CLKSEL[2:0]	R/W	111	CLKSEL would be applied by writing CLKCMD. 000: fcpu = fosc / 128 001: fcpu = fosc / 64 010: fcpu = fosc / 32 011: fcpu = fosc / 16 100: fcpu = fosc / 8 101: fcpu = fosc / 4 110: fcpu = fosc / 2 111: fcpu = fosc / 1

PCON Register (0x87)

Bit	Field	Type	Initial	Description
7				Refer to other chapter(s)
6..5	Reserved	R	0x00	
4	STWK	R/W	0	0: Both fosc and flosc are suspended in STOP mode. 1: flosc keep running in STOP mode*.
3	P2SEL	R/W	1	High-order address byte configuration bit. Chooses the higher byte of address (“XRAM [15:8]”) during MOVX @Ri operations 0: The “XRAM[15:8]” = “P2REG”. The “P2REG” is the contents of Port2 output register. 1: The “XRAM[15:8]” = 0x00.
2	GF0	R/W	0	General Purpose Flag
1	STOP	R/W	0	1: Microcontroller switch to STOP mode
0	IDLE	R/W	0	1: Microcontroller switch to IDLE mode

* Before entering STOP mode, the STWK bit setting must be earlier than the STOP bit.

9 System Operating Mode

The chip builds in three operating mode for difference clock rate and power saving reason. These modes control oscillators, op-code operation and analog peripheral devices' operation.

- Normal mode: System high-speed operating mode
- IDLE mode: System idle mode (Green mode)
- STOP mode: System power saving mode (Sleep mode)

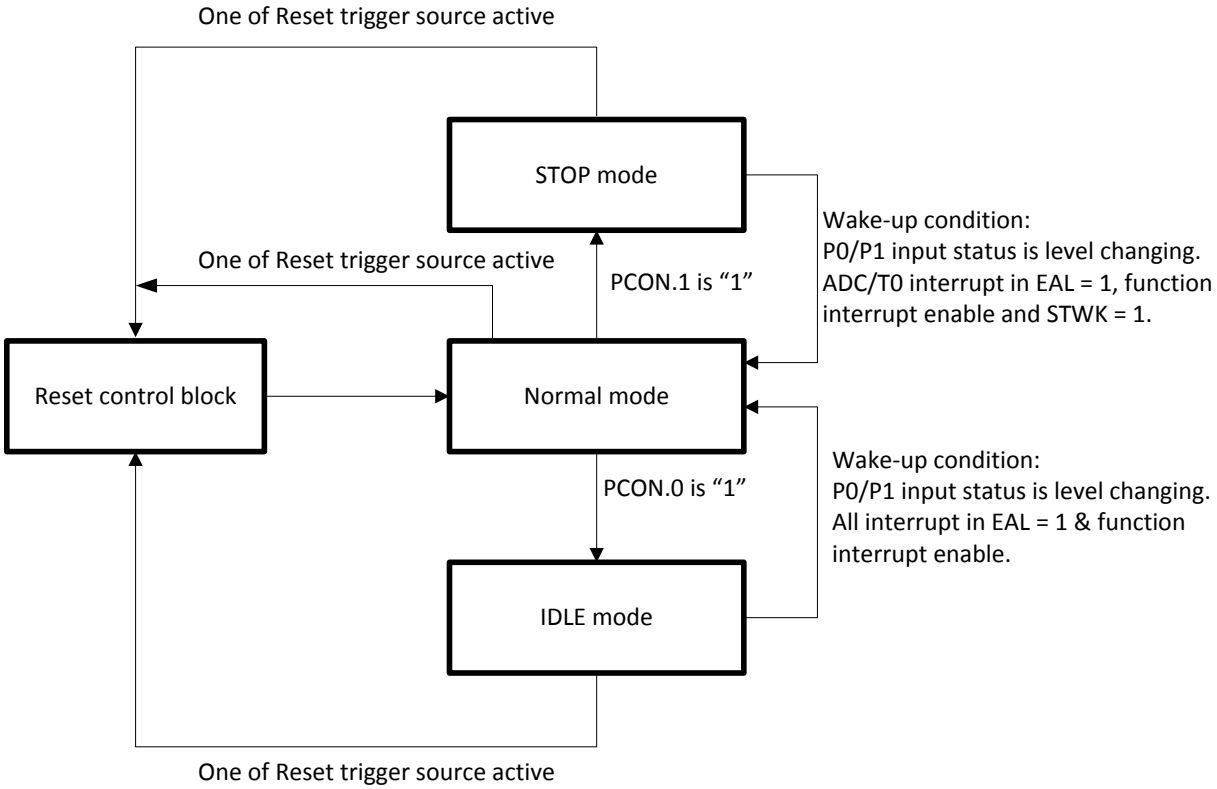


Table 9-1 The operating mode clock control

Operating Mode	Normal Mode	IDLE Mode	STOP Mode
IHRC	IHRC: Running Ext. OSC: Disable	IHRC: Running Ext. OSC: Disable	Stop
ILRC	Running	Running	STWK=1 & LCKS=1: Running Watchdog always: Running
Ext. OSC	IHRC: Disable Ext. OSC : Running	IHRC: Disable Ext. OSC : Running	Stop
Ext. low OSC	Active as LCKS = 0	Active as LCKS = 0	STWK=1 & LCKS=0: Running

CPU instruction	Executing	Stop	Stop
Timer 0 (Timer, Event counter)	Active by TR0	Active by TR0	Active as Timer 0 clock source is Flosc (ILRC or Ext. 32kHz) & STWK=1
Timer 1 (Timer, Event counter)	Active by TR1	Active by TR1	Inactive
Timer 2 (Timer, Event counter, compare, capture)	Active by T2I0 & T2I1	Active by T2I0 & T2I1	Inactive
PWM1	Active as enable	Active as enable	Inactive
SIO/I2C/UART	Active as enable	Active as enable	Inactive
LCD	Active as enable	Active as enable	Active as STWK=1
ADC	Active as enable	Active as enable	Active as ADC clock source is Flosc (ILRC or Ext. 32kHz) & STWK=1
Watchdog timer	By Watchdog Code option	By Watchdog Code option	By Watchdog Code option
Internal interrupt	All active	All active	ADC/T0 interrupt is active when STWK = 1 and clock source is Flosc. Other inactive.
External interrupt	All active	All active	All inactive
Wakeup source	-	P0, P1, Reset, All interrupt in EAL = 1 & function interrupt enable	P0, P1, Reset, ADC/T0 enable & clock source is Flosc (ILRC or Ext. 32kHz) & STWK=1 & function interrupt enable.

- Ext. OSC: External high-speed oscillator (XIN/XOUT).
- Ext. low OSC: External low-speed oscillator (LXIN/LXOUT).
- IHRC: Internal high-speed oscillator RC type.
- ILRC: Internal low-speed oscillator RC type.

9.1 Normal Mode

The Normal Mode is system high clock operating mode. The system clock source is from high speed oscillator. The program is executed. After power on and any reset trigger released, the system inserts into normal mode to execute program. When the system is wake-up from STOP/IDLE mode, the system also inserts into normal mode. In normal mode, the high speed oscillator actives, and the power consumption is largest of all operating modes.

- The program is executed, and full functions are controllable.
- The system rate is high speed.
- The high speed oscillator and internal low speed RC type oscillator active.
- Normal mode can be switched to other operating modes through PCON register.
- STOP/IDLE mode is wake-up to normal mode.

9.2 STOP Mode

The STOP mode is the system ideal status. No program execution and oscillator operation. Only internal regulator actives to keep all control gates status, register status and SRAM contents. The STOP mode is waked up by P0/P1 hardware level change trigger. P0 wake-up function is always enables. The STOP mode is wake-up to normal mode. Inserting STOP mode is controlled by stop bit of PCON register. When stop = 1, the system inserts into STOP Mode. After system wake-up from STOP mode, the stop bit is disabled (zero status) automatically.

- The program stops executing, and full functions are disabled.
- All oscillators including external high speed oscillator, internal high speed oscillator and internal low speed oscillator stop.
- Only internal regulator actives to keep all control gates status, register status and SRAM contents.
- The system inserts into normal mode after wake-up from STOP mode.
- The STOP mode wake-up source is P0/P1 level change trigger, ADC/T0 enable & clock source is Flosc (ILRC or Ext. 32kHz) & STWK=1 & function interrupt enable.

9.3 IDLE Mode

The IDLE mode is another system ideal status not like STOP mode. In STOP mode, all functions and hardware devices are disabled. But in IDLE mode, the system clock source keeps running, so the power consumption of IDLE mode is larger than STOP mode. In IDLE mode, the program isn't executed, but the timer with wake-up function actives as enabled, and the timer clock source is the non-stop system clock. The IDLE mode has 2 wake-up sources. One is the P0/P1 level change trigger wake-up. The other one is any interrupt in EAL = 1 & function interrupt enable. That's mean users can setup any function with interrupt enable, and the system is waked up until the interrupt issue. Inserting IDLE mode is controlled by idle bit of PCON register. When idle = 1, the system inserts into IDLE mode. After system wake-up from IDLE mode, the idle bit is disabled (zero status) automatically.

- The program stops executing, and full functions are disabled.
- Only the timer with wake-up function actives.
- The oscillator to be the system clock source keeps running, and the other oscillators operation is depend on system operation mode configuration.
- If inserting IDLE mode from normal mode, the system insets to normal mode after wake-up.
- The IDLE mode wake-up sources are P0/P1 level change trigger.
- If the function clock source is system clock, the functions are workable as enabled and under IDLE mode, e.g. Timer, PWM, event counter…
- All interrupt in EAL = 1 & function interrupt enable can wake-up in IDLE mode.

9.4 Wake up

Under STOP mode (sleep mode) or idle mode, program doesn't execute. The wakeup trigger can wake the system up to normal mode. The wakeup trigger sources are external trigger (P0/P1 level change) and internal trigger (any interrupt in EAL = 1 & function interrupt enable). The wakeup function builds in interrupt operation issued request flag and trigger system executing interrupt service routine as system wakeup occurrence.

When the system is in STOP mode the high clock oscillator stops. When waked up from STOP mode, MCU waits for 2048 external high-speed oscillator clocks + 5 internal low-speed oscillator clocks and 64 internal high-speed oscillator clocks + 5 internal low-speed oscillator clocks as the wakeup time to stable the oscillator circuit. After the wakeup time, the system goes into the normal mode.

The value of the external high clock oscillator wakeup time is as the following.

The Wakeup time = 1/Fosc * 2048 (sec) + 1/Flosc * 5 + high clock start-up time

Example: In STOP mode (sleep mode), the system is waked up. After the wakeup time, the system goes into normal mode. The wakeup time is as the following.

The wakeup time = 1/Fosc * 2048 + 1/Flosc * 5 = 0.825 ms (Fosc = 4MHz)

The total wakeup time = 0.825 ms + oscillator start-up time

The value of the internal high clock oscillator RC type wakeup time is as the following.

The Wakeup time = 1/Fosc * 64 (sec) + 1/Flosc * 5 + high clock start-up time

Example: In STOP mode (sleep mode), the system is waked up. After the wakeup time, the system goes into normal mode. The wakeup time is as the following.

The wakeup time = 1/Fosc * 64 +1/Flosc * 5= 315 us (Fhosc = 32MHz)

✷ ***Note: The high clock start-up time is depended on the VDD and oscillator type of high clock.***

Under STOP mode and green mode, the I/O ports with wakeup function are able to wake the system up to normal mode. The wake-up trigger edge is level changing in rising edge or falling edge. The Port 0 and Port 1 have wakeup function. Port 0 wakeup functions always enables, but the Port 1 is controlled by the P1W register.

P1W Register (0x91)

Bit	Field	Type	Initial	Description
7..0	P1nW	R/W	0	0: Disable P1.n wakeup functionality 1: Enable P1.n wakeup functionality

10 Interrupt

The MCU provides 15 interrupt sources (3 external and 12 interrupt) with 4 priority levels. Each interrupt source includes one or more interrupt request flag(s). When interrupt event occurs, the associated interrupt flag is set to logic 1. If both interrupt enable bit and global interrupt (EAL=1) are enabled, the interrupt request is generated and interrupt service routine (ISR) will be started. Some interrupt request flags must be cleared by software. However, most interrupt request flags can be cleared by hardware automatically. In the end, ISR is finished after complete the RETI instruction. The summary of interrupt source, interrupt vector, priority order and control bit are shown as the table below.

Table 10-1 The interrupt list

Interrupt	Enable Interrupt	Request (IRQ)	IRQ Clearance	Priority / Vector
System Reset	-	-	-	0 / 0x0000
INT0	EX0	IE0	Automatically	1 / 0x0003
INT2	EX2	IE2	Automatically	2 / 0x0083
I2C	EI2C	SI	By firmware	3 / 0x0043
Timer 0	ET0	TF0	Automatically	4 / 0x000B
ADC	EADC	ADCF	Automatically	5 / 0x008B
SPI	ESPI	SPIF / MODF	By firmware	6 / 0x004B
INT1	EX1	IE1	Automatically	7 / 0x0013
PWM1	EPWM1	PWM1F	Automatically	8 / 0x0093
T2COM0	ET2C0	TF2C0	Automatically	9 / 0x0053
Timer 1	ET1	TF1	Automatically	10 / 0x001B
T2COM1	ET2C1	TF2C1	Automatically	11 / 0x005B
UART	ES0	TI0 / RI0	By firmware	12 / 0x0023
T2COM2	ET2C1	TF2C2	Automatically	13 / 0x0063
Timer 2	ET2 / ET2RL	TF2 / TF2RL	Automatically	14 / 0x002B
T2COM3	ET2C3	TF2C3	Automatically	15 / 0x006B

10.1 Interrupt Operation

Interrupt operation is controlled by interrupt request flag and interrupt enable bits. Interrupt request flag is interrupt source event indicator, no matter what interrupt function status (enable or disable). Both interrupt enable bit and global interrupt (EAL=1) are enabled, the system executes interrupt operation when each of interrupt request flags actives. The program counter points to interrupt vector (0x03 – 0x93) and execute ISR.

10.2 Interrupt Priority

Each interrupt source has its specific default priority order. If two interrupts occurs simultaneously, the higher priority ISR will be service first. The lower priority ISR will be serviced after the higher priority ISR completes. The next ISR will be service after the previous ISR complete, no matter the priority order.

For special priority needs, 4-level priority levels (Level 0 – Level 3) are used. All interrupt sources are classified into 6 priority groups (Group0 – Group5). Each group can be set one specific priority level. Priority level is selected by IP0/IP1 registers. Level 3 is the highest priority and Level 0 is the lowest. The interrupt sources inside the same group will share the same priority level. With the same priority level, the priority rule follows default priority.

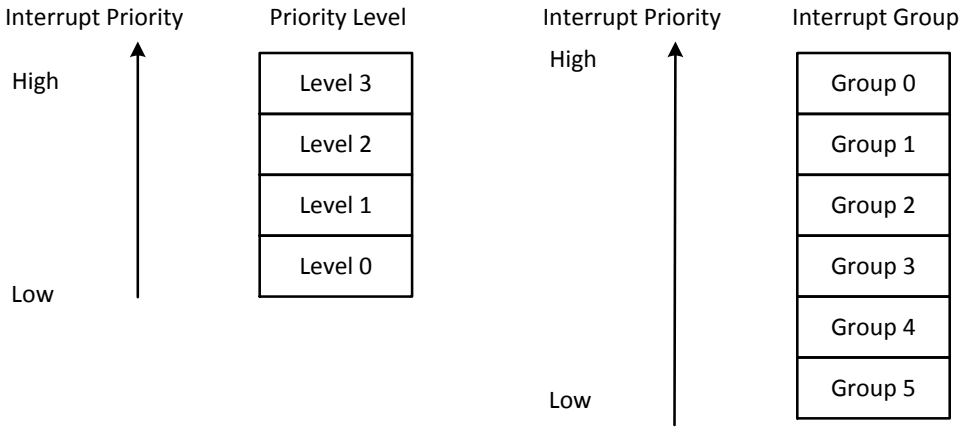
Priority Level	IP1.x	IP0.x
Level 0	0	0
Level 1	0	1
Level 2	1	0
Level 3	1	1

The ISR with the higher priority level can be serviced first; even can break the on-going ISR with the lower priority level. The ISR with the lower priority level will be pending until the ISR with the higher priority level completes.

Group	Interrupt Source			
Group 0	INT0	INT2	I2C	
Group 1	Timer 0	ADC	SPI	
Group 2	INT1	PWM1	T2COM0	
Group 3	Timer 1		T2COM1	
Group 4	UART		T2COM2	
Group 5	Timer 2		T2COM3	

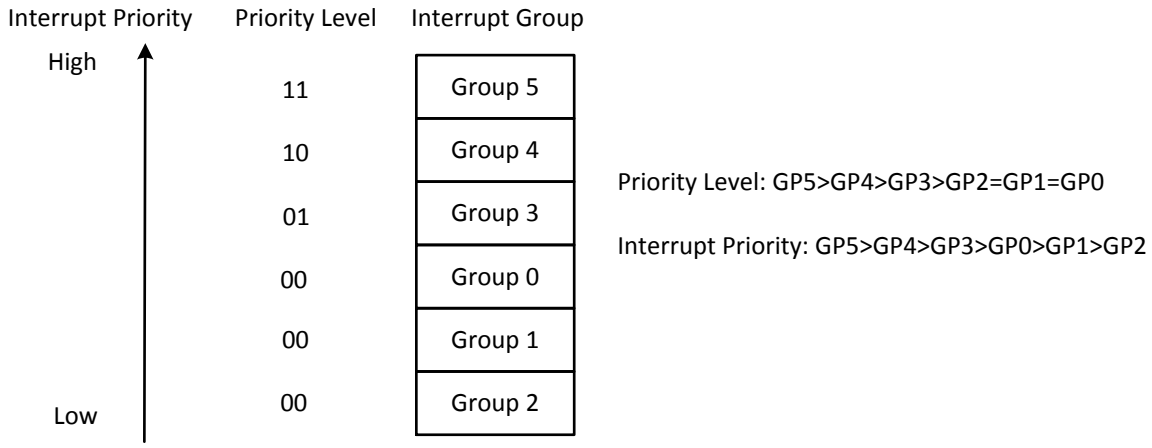
When more than one interrupt request occur, the highest priority request must be executed first. Choose the highest priority request according natural priority and priority level. The steps are as the following:

1. Choose the groups which have the highest priority level between all groups.
2. Choose the group which is the highest nature priority between the groups with the highest priority level.
3. Choose the ISR which has the highest nature priority inside the group with the highest priority.



As the example, group5 has the highest priority level and group0~group2 have the lowest priority level. It means the interrupt vector in group5 has the highest interrupt priority, the 2nd interrupt priority in group4 and the 3rd interrupt priority in group3. Group0~ group2 have the same priority level thus the nature priority rule will be followed. Therefore, interrupt priority will be group5> group4> group3> group0> group1> group2.

```
MOV     IP0, #00101000B         ; Set group0 - group5 in different priority level.
MOV     IP1, #00110000B
```



IP0, IP1 Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IP0	-	-	IP05	IP04	IP03	IP02	IP01	IP00
IP1	-	-	IP15	IP14	IP13	IP12	IP11	IP10

IP0 Register (0XA9)

Bit	Field	Type	Initial	Description
5..0	IP0[5:0]	R/W	0	Interrupt priority. Each bit together with corresponding bit from IP1 register specifies the priority level of the respective interrupt priority group.
Else	Reserved	R	0	

IP1 Register (0XB9)

Bit	Field	Type	Initial	Description
5..0	IP1[5:0]	R/W	0	Interrupt priority. Each bit together with corresponding bit from IP0 register specifies the priority level of the respective interrupt priority group.
Else	Reserved	R	0	

10.3 Request Flag Clearance

When the interrupt function is enabled, most of the interrupt request flags can be cleared by hardware automatically. Some still need to use the program to clear. However, when the interrupt function is turned off, the interrupt request flag only be cleared by program.

Most cases can be cleared by the ANL instruction. But some special cases, if the adjacent flag is issued asynchronously, it may be accidentally cleared.

```
IRCON &= 0xF7;    // It is possible to cause adjacent flags to be cleared.
TF2C1 = 0;        // It is possible to cause adjacent flags to be cleared.
```

If you want to avoid the above, it is recommended to use the interrupt bit characteristics. Most of the interrupt request flag can't be written 1 by which you can avoid clearing asynchronous adjacent flags.

For user who is develop program in C language, the flag clear macros is strongly recommended to clear request flag, instead of clear request flag bits directly.

```
TCONCLR(bit);     // The marco can clear the flag of TCON. bit is 0~7.
IRCONCLR(bit);    // The marco can clear the flag of IRCON. bit is 0~7.
IRCON2CLR(bit);   // The marco can clear the flag of IRCON2. bit is 0~7.
```

10.4 Interrupt Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
IEN1	-	-	ET2C3	ET2C2	ET2C1	ET2C0	ESPI	EI2C
IEN2	-	-	-	-	EPWM1	EX2	-	EADC
IRCON	-	TF2	TF2C3	TF2C2	TF2C1	TF2C0	-	-
IRCON2	-	-	-	-	PWM1F	IE2	-	ADCF
TCON	TF1	TR1	TF0	TR0	IE1	-	IE0	-
S0CON	SM0	SM1	SM20	REN0	TB80	RB80	TI0	RI0
SPSTA	SPIF	WCOL	SSERR	MODF	-	-	-	-
I2CCON	CR2	ENS1	STA	STO	SI	AA	CR1	CR0

IEN0 Register (0XA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Enable all interrupt control bit. 0: Disable all interrupt function. 1: Enable all interrupt function.
5	ET2	R/W	0	T2 timer interrupt control bit 0: Disable T2 interrupt function. 1: Enable T2 interrupt function.
4	ES0	R/W	0	UART interrupt control bit. 0: Disable UART interrupt function. 1: Enable UART interrupt function.
3	ET1	R/W	0	T1 timer interrupt control bit. 0: Disable T1 interrupt function. 1: Enable T1 interrupt function.
2	EX1	R/W	0	External P0.0 interrupt (INT1) control bit. 0: Disable INT1 interrupt function. 1: Enable INT1 interrupt function.
1	ET0	R/W	0	T0 timer interrupt control bit. 0: Disable T0 interrupt function. 1: Enable T0 interrupt function
0	EX0	R/W	0	External P3.0 interrupt (INT0) control bit. 0: Disable INT0 interrupt function. 1: Enable INT0 interrupt function.
Else	Reserved	R	0	

IEN1 Register (0XB8)

Bit	Field	Type	Initial	Description
5	ET2C3	R/W	0	T2 Timer COM3 interrupt control bit. 0: Disable T2COM3 interrupt function. 1: Enable T2COM3 interrupt function.
4	ET2C2	R/W	0	T2 Timer COM2 interrupt control bit. 0: Disable T2COM2 interrupt function. 1: Enable T2COM2 interrupt function.
3	ET2C1	R/W	0	T2 Timer COM1 interrupt control bit. 0: Disable T2COM1 interrupt function. 1: Enable T2COM1 interrupt function.
2	ET2C0	R/W	0	T2 Timer COM0 interrupt control bit. 0: Disable T2COM0 interrupt function. 1: Enable T2COM0 interrupt function.
1	ESPI	R/W	0	SPI interrupt control bit 0: Disable SPI interrupt function. 1: Enable SPI interrupt function.
0	EI2C	R/W	0	I2C interrupt control bit. 0: Disable I2C interrupt function. 1: Enable I2C interrupt function.
Else	Reserved	R	0	

IEN2 Register (0X9A)

Bit	Field	Type	Initial	Description
3	EPWM1	R/W	0	PWM1 interrupt control bit. 0 = Disable PWM1 interrupt function. 1 = Enable PWM1 interrupt function.
2	EX2	R/W	0	External P0.4 interrupt (INT2) control bit. 0: Disable INT2 interrupt function. 1: Enable INT2 interrupt function.
0	EADC	R/W	0	ADC interrupt control bit. 0: Disable ADC interrupt function. 1: Enable ADC interrupt function.
Else	Reserved	R	0	

IRCON Register (0xC0)

Bit	Field	Type	Initial	Description
6	TF2	R/W*	0	T2 timer interrupt request flag. 0: None T2 interrupt request. 1: T2 interrupt request.
5	TF2C3	R/W*	0	T2 Timer COM3 interrupt request flag. 0: None T2COM3 interrupt request. 1: T2COM3 interrupt request.
4	TF2C2	R/W*	0	T2 Timer COM2 interrupt request flag. 0: None T2COM2 interrupt request. 1: T2COM2 interrupt request.
3	TF2C1	R/W*	0	T2 Timer COM1 interrupt request flag. 0: None T2COM1 interrupt request. 1: T2COM1 interrupt request.
2	TF2C0	R/W*	0	T2 Timer COM0 interrupt request flag. 0: None T2COM0 interrupt request. 1: T2COM0 interrupt request.
Else	Reserved	R	0	

* This bit can't write '1' value. The IRCONCLR macro is strongly recommended to clear request flag.

IRCON2 Register (0XBF)

Bit	Field	Type	Initial	Description
3	PWM1F	R/W*	0	PWM1 interrupt request flag. 0: None PWM1 interrupt request 1: PWM1 interrupt request.
2	IE2	R/W*	0	External P0.4 interrupt (INT2) request flag 0: None INT2 interrupt request. 1: INT2 interrupt request.
0	ADCF	R/W*	0	ADC interrupt request flag. 0: None ADC interrupt request. 1: ADC interrupt request.
Else	Reserved	R	0	

* This bit can't write '1' value. The IRCON2CLR macro is strongly recommended to clear request flag.

TCON Register (0X88)

Bit	Field	Type	Initial	Description
7	TF1	R/W*	0	T1 timer external reload interrupt request flag. 0: None T1 interrupt request 1: T1 interrupt request.
5	TF0	R/W*	0	T0 timer external reload interrupt request flag. 0: None T0 interrupt request 1: T0 interrupt request.
3	IE1	R/W*	0	External P0.0 interrupt (INT1) request flag 0: None INT1 interrupt request. 1: INT1 interrupt request.
1	IE0	R/W*	0	External P3.0 interrupt (INT0) request flag 0: None INT0 interrupt request. 1: INT0 interrupt request.
Else				Refer to other chapter(s)

* This bit can't write '1' value. The TCONCLR macro is strongly recommended to clear request flag.

S0CON Register (0X98)

Bit	Field	Type	Initial	Description
1	TI0	R/W	0	UART transmit interrupt request flag. It indicates completion of a serial transmission at UART. It is set by hardware at the end of bit 8 in mode 0 or at the beginning of a stop bit in other modes. It must be cleared by software. 0: None UART transmit interrupt request. 1: UART transmit interrupt request.
0	RI0	R/W	0	UART receive interrupt request flag. It is set by hardware after completion of a serial reception at UART. It is set by hardware at the end of bit 8 in mode 0 or in the middle of a stop bit in other modes. It must be cleared by software. 0: None UART receive interrupt request. 1: UART receive interrupt request.
Else				Refer to other chapter(s)

SPSTA Register (0XE1)

Bit	Field	Type	Initial	Description
7	SPIF	R	0	SPI complete communication flag Set automatically at the end of communication Cleared automatically by reading SPSTA, SPDAT registers
4	MODF	R	0	Mode fault flag
Else				Refer to other chapter(s)

I2CCON Register (0XDC)

Bit	Field	Type	Initial	Description
7	SI	R/W	0	Serial interrupt flag The SI is set by hardware when one of 25 out of 26 possible I2C states is entered. The only state that does not set the SI is state F8h, which indicates that no relevant state information is available. The SI flag must be cleared by software. In order to clear the SI bit, '0' must be written to this bit. Writing a '1' to SI bit does not change value of the SI.
Else				Refer to other chapter(s)

10.5 Example

Defining Interrupt Vector. The interrupt service routine is following user program.

```
ORG     0           ; 0000H
JMP     START       ; Jump to user program address.
…
ORG     0X0003      ; Jump to interrupt service routine address.
JMP     ISR_INT0
ORG     0X000B
JMP     ISR_T0
…
ORG     0X0083
JMP     ISR_INT2
…
```

```
                ORG     0X00ECH
START:                          ; 00ECH, The head of user program.
                …               ; User program.
                …
                …
                JMP     START   ; End of user program.
                …
ISR_ INT0:                      ; The head of interrupt service routine.
                PUSH    ACC     ; Save ACC to stack buffer.
                PUSH    PSW     ; Save PSW to stack buffer.
                …
                POP     PSW     ; Load PSW from stack buffer.
                POP     ACC     ; Load ACC from stack buffer.
                RETI            ; End of interrupt service routine.
ISR_T0:                         ;
                PUSH    ACC     ; Save ACC to stack buffer.
                PUSH    PSW     ; Save PSW to stack buffer.
                …
                POP     PSW     ; Load PSW from stack buffer.
                POP     ACC     ; Load ACC from stack buffer.
                RETI            ; End of interrupt service routine.
                …
ISR_INT2                        ;
                PUSH    ACC     ; Save ACC to stack buffer.
                PUSH    PSW     ; Save PSW to stack buffer.
                …
                POP     PSW     ; Load PSW from stack buffer.
                POP     ACC     ; Load ACC from stack buffer.
                RETI            ; End of interrupt service routine.


                END             ; End of program.
```

11 GPIO

The microcontroller has up to 26 bidirectional general purpose I/O pin (GPIO). Unlike the original 8051 only has open-drain output, SN8F5814 builds in push-pull output structure to improve its driving performance.

11.1 Input and Output Control

The input and output direction control is configurable through P0M to P2M registers. These bits specify each pin that is either input mode or output mode.

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P0M	P07M	P06M	P05M	P04M	P03M	P02M	P01M	P00M
P1M	P17M	P16M	P15M	P14M	P13M	P12M	P11M	P10M
P2M	P27M	P26M	P25M	P24M	P23M	P22M	P21M	P20M
P3M	-	-	-	-	-	-	P31M	P30M
P2OC	P27OC	P26OC	-	P24OC	P23OC	P22OC	-	-

P0M: 0xF9, P1M: 0xFA, P2M: 0XFB, P3M: 0xFC

Bit	Field	Type	Initial	Description
7	P07M	R/W	0	Mode selection of P0.7 0: Input mode 1: Output mode
6	P06M	R/W	0	Mode selection of P0.6 0: Input mode 1: Output mode
5	P05M	R/W	0	Mode selection of P0.5 0: Input mode 1: Output mode
4..0				et cetera

P2OC Register (0xE4)

Bit	Field	Type	Initial	Description
7..5		R/W	000	Refer to PWM chapter
7	P27OC	R/W	0	P2.7 open-drain control bit 0: Disable 1: Enable
6	P26OC	R/W	0	P2.6 open-drain control bit 0: Disable 1: Enable
4	P24OC	R/W	0	P2.4 open-drain control bit 0: Disable 1: Enable
3	P23OC	R/W	0	P2.3 open-drain control bit 0: Disable 1: Enable
2	P22OC	R/W	0	P2.2 open-drain control bit 0: Disable 1: Enable
Else	Reserved	R	0	

11.2 Input Data and Output Data

By a read operation from any registers of P0 to P3, the current pin's logic level would be fetch to represent its external status. This operation remains functional even the pin is shared with other function like UART and I2C which can monitor the bus condition in some case.

A write P0 to P3 register value would be latched immediately, yet the value would be outputted until the mapped P0M – P3M is set to output mode. If the pin is currently in output mode, any value set to P0 to P3 register would be presented on the pin immediately.

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P0	P07	P06	P05	P04	P03	P02	P01	P00
P1	P17	P16	P15	P14	P13	P12	P11	P10
P2	P27	P26	P25	P24	P23	P22	P21	P20
P3	-	-	-	-	-	-	P31	P30

P0: 0x80, P1: 0x90, P2: 0xA0, P3: 0xB0

Bit	Field	Type	Initial	Description
7	P07	R/W	1	Read: P0.7 pin's logic level Write 1/0: Output logic high or low (applied if P07M = 1)
6	P06	R/W	1	Read: P0.6 pin's logic level Write 1/0: Output logic high or low (applied if P06M = 1)
5	P05	R/W	1	Read: P0.5 pin's logic level Write 1/0: Output logic high or low (applied if P05M = 1)
4..0				et cetera

11.3 On-chip Pull-up Resisters

The P0UR to P3UR registers are mapped to each pins' internal 100k ohm (in typical value) pull-up resister.

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P0UR	P07UR	P06UR	P05UR	P04UR	P03UR	P02UR	P01UR	P00UR
P1UR	P17UR	P16UR	P15UR	P14UR	P13UR	P12UR	P11UR	P10UR
P2UR	P27UR	P26UR	P25UR	P24UR	P23UR	P22UR	P21UR	P20UR
P3UR	-	-	-	-	-	-	P31UR	P30UR

P0UR: 0xF1, P1UR: 0xF2, P2UR: 0xF3, P3UR: 0XF4

Bit	Field	Type	Initial	Description
7	P07UR	R/W	0	On-chip pull-up resister control of P0.7 0: Disable* 1: Enable
6	P06UR	R/W	0	On-chip pull-up resister control of P0.6 0: Disable* 1: Enable
5	P05UR	R/W	0	On-chip pull-up resister control of P0.5 0: Disable* 1: Enable
4..0				et cetera

* Recommended disable pull-up resister if the pin is output mode or analog function

11.4 Pin Shared with Analog Function

The microcontroller builds in analog functions, such as AD and LCD. The Schmitt trigger of input channel is strongly recommended to switch off if the pin's shared analog function is enabled.

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
P0CON	P0CON7	P0CON6	P0CON5	-	-	-	-	-
P1CON	P1CON7	P1CON6	P1CON5	P1CON4	P1CON3	P1CON2	P1CON1	P1CON0
P2CON	P2CON7	P2CON6	P1CON5	P2CON4	P2CON3	P2CON2	P2CON1	P2CON0
P3CON	-	-	-	-	-	-	P3CON1	P3CON0

P0CON: 0x9E, P1CON: 0x9F, P2CON: 0XD6, P3CON: 0XD7

P0CON Register (0x9E)

Bit	Field	Type	Initial	Description
7..5	P0CON[7:5]	R/W	0x00	P0 configuration control bit*. 0: P0 can be analog input pin or digital GPIO pin. 1: P0 is pure analog input pin and can't be a digital GPIO pin.

* P0CON [7:0] will configure related Port0 pin as pure analog input pin to avoid current leakage.

P1CON Register (0x9F)

Bit	Field	Type	Initial	Description
7..0	P1CON[7:0]	R/W	0x00	P1 configuration control bit*. 0: P1 can be analog input pin or digital GPIO pin. 1: P1 is pure analog input pin and can't be a digital GPIO pin.

* P1CON [7:0] will configure related Port1 pin as pure analog input pin to avoid current leakage.

P2CON Register (0XD6)

Bit	Field	Type	Initial	Description
7..0	P2CON[7:0]	R/W	0x0	P2 configuration control bit*. 0: P2 can be analog input pin or digital GPIO pin. 1: P2 is pure analog input pin and can't be a digital GPIO pin.

* P2CON [7:0] will configure related Port2 pin as pure analog input pin to avoid current leakage.

P3CON Register (0XD7)

Bit	Field	Type	Initial	Description
1..0	P3CON[1:0]	R/W	0x0	P3 configuration control bit*. 0: P3 can be analog input pin or digital GPIO pin. 1: P3 is pure analog input pin and can't be a digital GPIO pin.

* P3CON [7:0] will configure related Port3 pin as pure analog input pin to avoid current leakage.

12 External Interrupt

INT0, INT1 and INT2 are external interrupt trigger sources. Build in edge trigger configuration function and edge direction is selected by PEDGE register. When both external interrupt (EX0/EX1/EX2) and global interrupt (EAL) are enabled, the external interrupt request flag (IE0/IE1/IE2) will be set to “1” as edge trigger event occurs. The program counter will jump to the interrupt vector (ORG 0x0003/0x0013/0x0083) and execute interrupt service routine. Interrupt request flag will be cleared by hardware before ISR is executed.

12.1 External Interrupt Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PEDGE	-	-	EX2G1	EX2G0	EX1G1	EX1G0	EX0G1	EX0G0
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
TCON	TF1	TR1	TF0	TR0	IE1	-	IE0	-
IEN2	-	-	-	-	EPWM1	EX2	-	EADC
IRCON2	-	-	-	-	PWM1F	IE2	-	ADCF

PEDGE Register (0X8F)

Bit	Field	Type	Initial	Description
5..4	EX2G[1:0]	R/W	10	External interrupt 2 trigger edge control register. 00: Reserved. 01: Rising edge trigger. 10: Falling edge trigger (default) 11: Both rising and falling edge trigger
3..2	EX1G[1:0]	R/W	10	External interrupt 1 trigger edge control register. 00: Reserved. 01: Rising edge trigger. 10: Falling edge trigger (default) 11: Both rising and falling edge trigger
1..0	EX0G[1:0]	R/W	10	External interrupt 0 trigger edge control register. 00: Reserved. 01: Rising edge trigger. 10: Falling edge trigger (default) 11: Both rising and falling edge trigger
Else	Reserved	R	0	

IEN0 Register (0XA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Enable all interrupt control bit. 0: Disable all interrupt function. 1: Enable all interrupt function.
2	EX1	R/W	0	External P0.0 interrupt (INT1) control bit. 0: Disable INT1 interrupt function. 1: Enable INT1 interrupt function.
0	EX0	R/W	0	External P3.0 interrupt (INT0) control bit. 0: Disable INT0 interrupt function. 1: Enable INT0 interrupt function.
Else				Refer to other chapter(s)

TCON Register (0X88)

Bit	Field	Type	Initial	Description
3	IE1	R/W*	0	External P0.0 interrupt (INT1) request flag 0: None INT1 interrupt request. 1: INT1 interrupt request.
1	IE0	R/W*	0	External P3.0 interrupt (INT0) request flag 0: None INT0 interrupt request. 1: INT0 interrupt request.
Else				Refer to other chapter(s)

* This bit can't write '1' value. The TCONCLR macro is strongly recommended to clear request flag.

IEN2 Register (0X9A)

Bit	Field	Type	Initial	Description
2	EX2	R/W	0	External P0.4 interrupt (INT2) control bit. 0: Disable INT2 interrupt function. 1: Enable INT2 interrupt function.
Else				Refer to other chapter(s)

IRCON2 Register (0XBF)

Bit	Field	Type	Initial	Description
2	IE2	R/W*	0	External P0.4 interrupt (INT2) request flag 0: None INT2 interrupt request. 1: INT2 interrupt request.
Else				Refer to other chapter(s)

* This bit can't write '1' value. The IRCON2CLR macro is strongly recommended to clear request flag.

12.2 Sample Code

The following sample code demonstrates how to perform INT0/INT1/INT2 with interrupt.

```
#define INT0Rsing     (1 << 0)  //INT0 trigger edge is rising edge
#define INT0Falling   (2 << 0)  //INT0 trigger edge is falling edge
#define INT0LeChge    (3 << 0)  //INT0 trigger edge is level chagne
#define EINT0         (1 << 0)  //INT0 interrupt enable

#define INT1Rsing     (1 << 2)  //INT1 trigger edge is rising edge
#define INT1Falling   (2 << 2)  //INT1 trigger edge is falling edge
#define INT1LeChge    (3 << 2)  //INT1 trigger edge is level chagne
#define EINT1         (1 << 2)  //INT1 interrupt enable

#define INT2Rsing     (1 << 4)  //INT2 trigger edge is rising edge
#define INT2Falling   (2 << 4)  //INT2 trigger edge is falling edge
#define INT2LeChge    (3 << 4)  //INT2 trigger edge is level chagne
#define EINT2         (1 << 2)  //INT2 interrupt enable


void EnableINT(void)
{
  // INT0 rising edge, INT1 falling edge, INT2 level change
  PEDGE = INT0Rising | INT1Falling | INT2LeChge;

  // Enable INT0/INT1 interrupt
  IEN0 |= EINT0 | EINT1;
  // Enable INT2 interrupt
  IEN2 |= EINT2
  // Enable total interrupt
  IEN0 |= 0x80;

  P1 = 0x00;
  P1M = 0x07;
}

void INT0Interrupt(void) interrupt ISRInt0  //0x03
{ //IE0 clear by hardware
  P10 = ~P10;
}

void INT1Interrupt(void) interrupt ISRInt1  //0x13
{ //IE1 clear by hardware
  P11 = ~P11;
}

void INT2Interrupt(void) interrupt ISRInt2  //0x83
{ //IE2 clear by hardware
  P12 = ~P12;
}
```

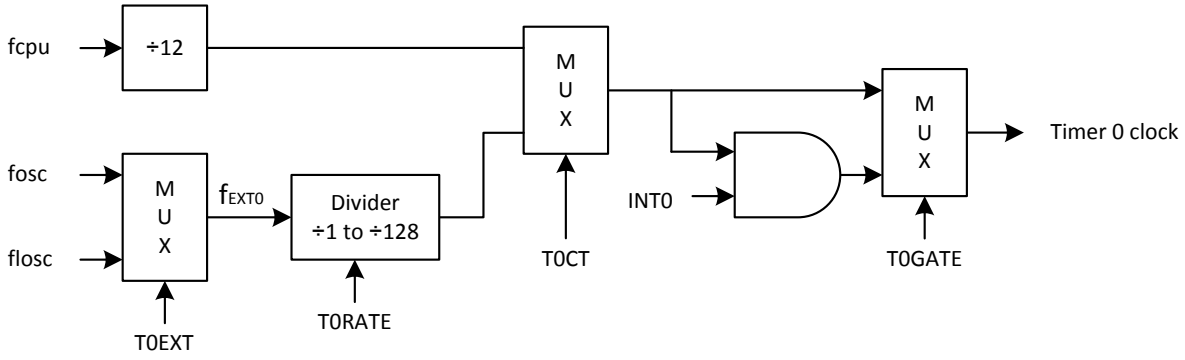
13 Timer 0 and Timer 1

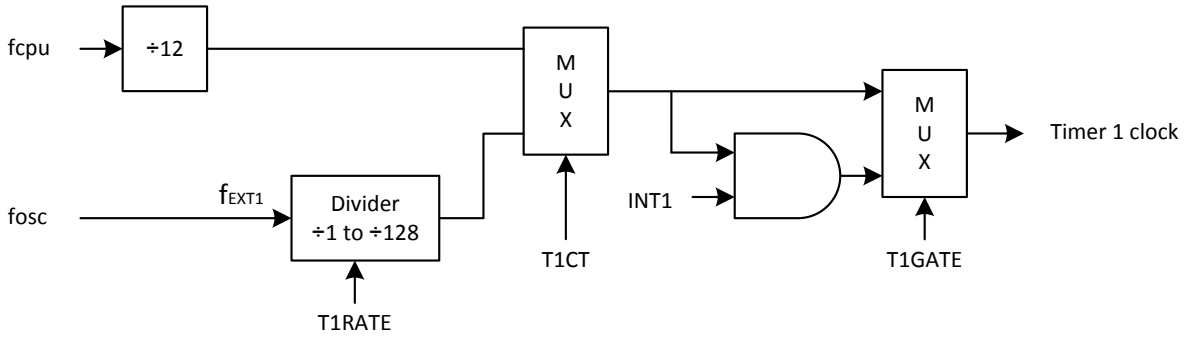
Timer 0 and Timer 1 are two independent binary up timers. Timer 0 has four different operation modes: (1) 13-bit up counting timer, (2) 16-bit up counting timer, (3) 8-bit up counting timer with specified reload value support, and (4) separated two 8-bit up counting timer. By contrast, Timer 1 has only mode 0 to mode 2 which are same as Timer 0. Timer 0 and Timer 1 respectively support ET0 and ET1 interrupt function.

When Timer 0 clock source is flosc and STWK=1, Timer 0 can work in stop mode and waked up from stop mode by Timer 0 interrupt.

13.1 Timer 0 and Timer 1 Clock Selection

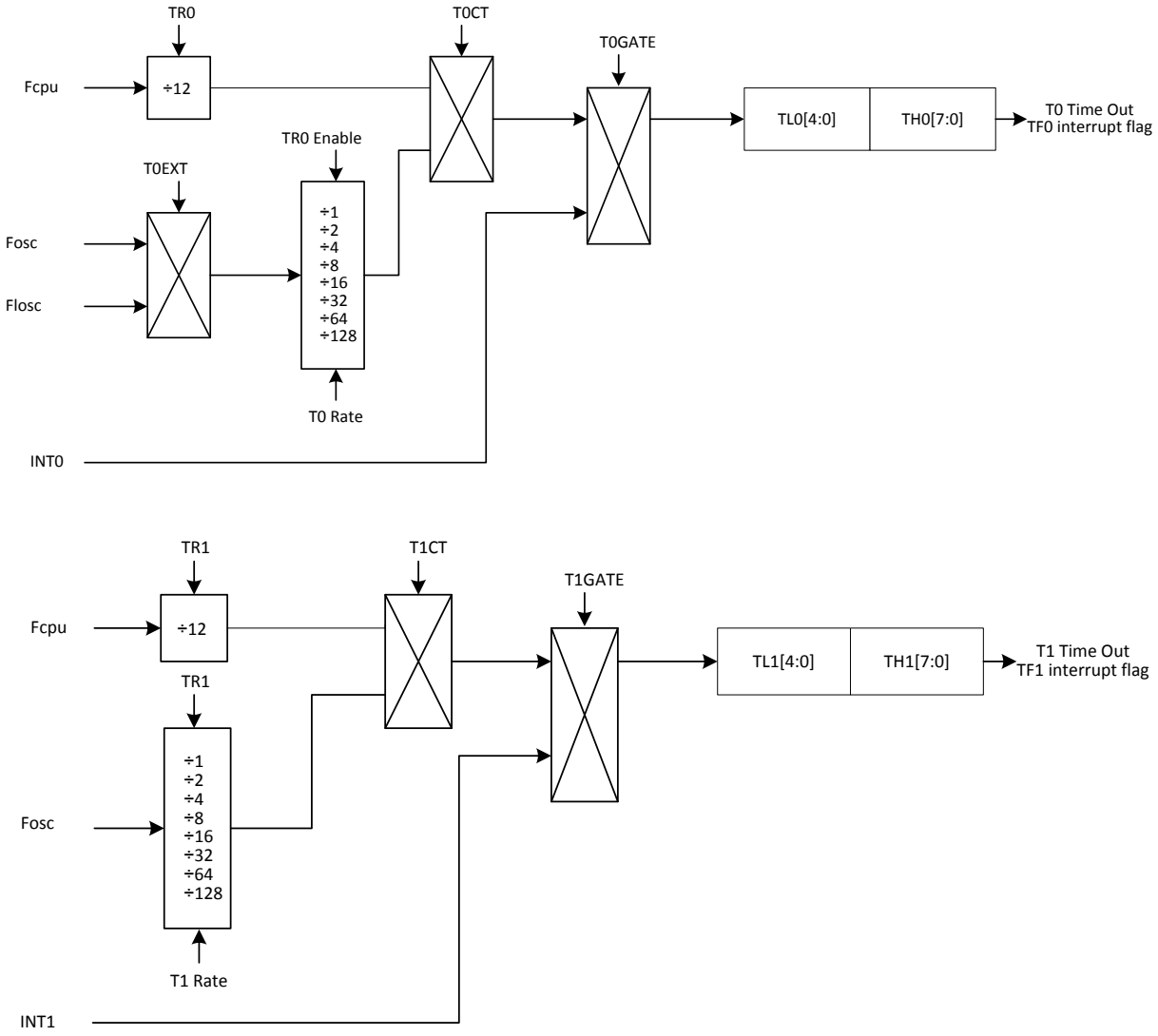
The figures below illustrate the clock selection circuit of Timer 0 and Timer 1. Timer 0 has three clock sources selection: fcpu, fosc, and flosc. All clock sources can be gated (pause) by INT0 pin if T0GATE is applied. Timer 1 clock sources selection: fcpu and fosc. All clock sources can be gated (pause) by INT1 pin if T1GATE is applied. Overall, the major difference between the two timers is that Timer 0 additionally supports flosc clock source (low speed clock).





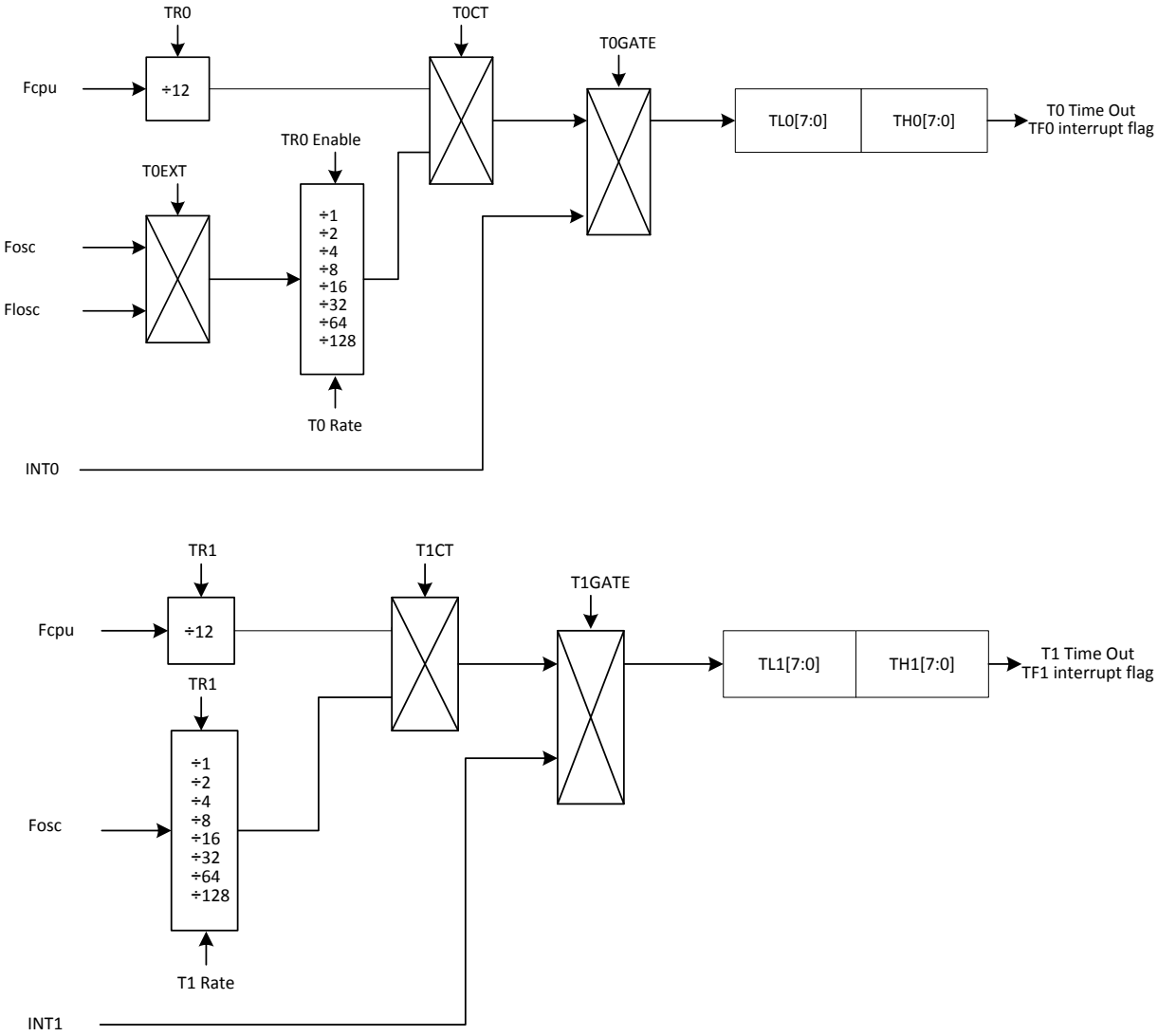
13.2 Mode 0: 13-bit Up Counting Timer

Timer 0 and Timer 1 in mode 0 is a 13-bit up counting timer (the upper 3 bits of TL0 is suspended). Once the timer's counter is overflow (counts from 0xFF1F to 0x0000), TF0/TF1 flag would be issued immediately. This flag is readable by firmware if ET0/ET1 does not apply, or can be handled by interrupt controller if ET0/ET1 is applied.



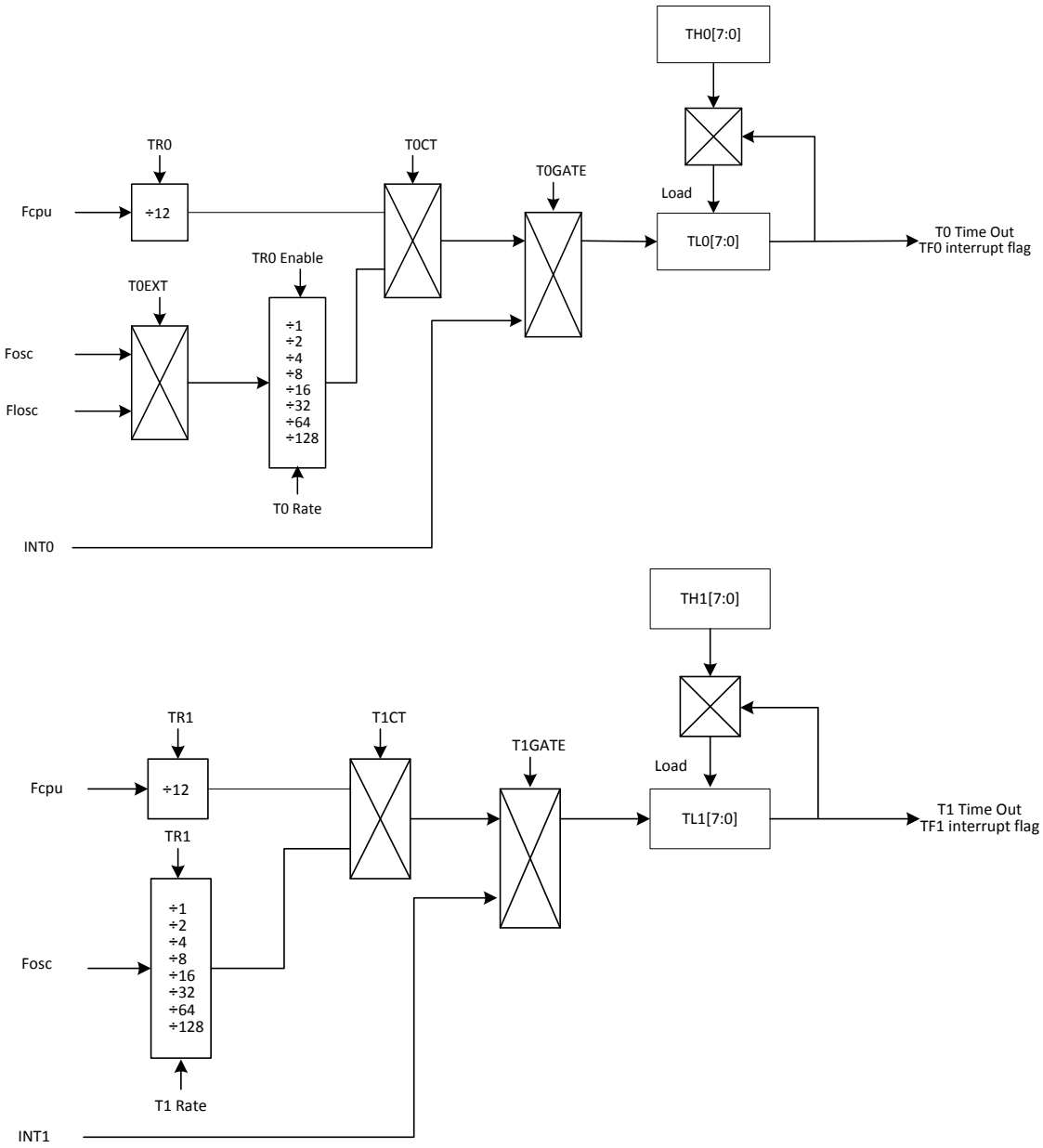
13.3 Mode 1: 16-bit Up Counting Timer

Timer 0 and Timer 1 in mode 1 is a 16-bit up counting timer. Once the timer's counter overflow is occurred (from 0xFFFF to 0x0000), TF0/TF1 would be issued which is readable by firmware or can be handled by interrupt controller (if ET0/ET1 applied).



13.4 Mode 2: 8-bit Up Counting Timer with Specified Reload Value Support

Timer 0 and Timer 1 in mode 2 is an 8-bit up counting timer (TL0/TL1) with a specifiable reload value. An overflow event (TL0/TL1 counts from 0xFF to 0x00) issues its TF0/TF1 flag for firmware or interrupt controller; meanwhile, the timer duplicates TH0/TH1 value to TL0/TL1 register in the same time. As a result, the timer is actually counts from 0xFF to the value of TH0/TH1.

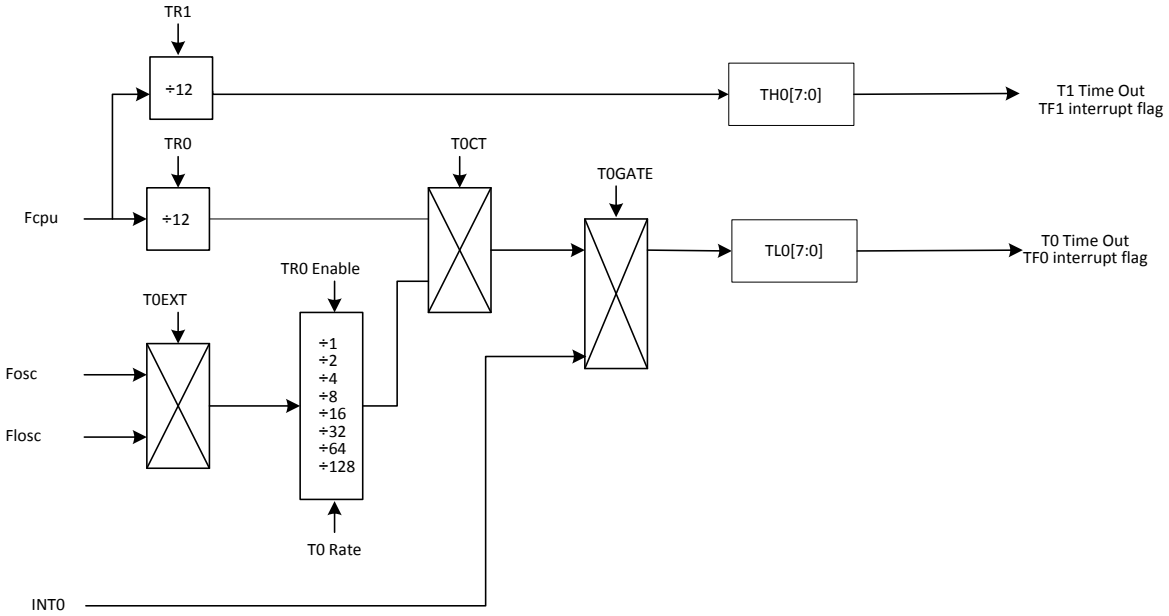


13.5 Mode 3 (Timer 0 only): Separated Two 8-bit Up Counting Timer

Mode 3 treats TH0 and TL0 as two separated 8-bit timers. TL0 is an 8-bit up counting timer with RTC support or two clock sources selection (fcpu and fosc), whereas TH0 clock source is fixed at fcpu/12. Only TL0 clock source can be gated (pause) by INT0 pin if T0GATE is applied.

In this mode TL0 counter is enabled by TR0, and its overflow signal is reflected in TF0 flag. TH0 counter is controlled by TR1, and TF1 flag is also occupied by TH0 overflow signal.

Timer 1 cannot issue any overflow event in this situation, and it can be seen as a self-counting timer without flag support.



13.6 Function Enable and Disable

Most cases can be set or cleared control bit by ORL or ANL instruction. But some special cases, if the adjacent flag is issued asynchronously, it may be accidentally cleared.

```
TCON &= 0xEF;    // It is possible to cause adjacent flags to be cleared.
TR1 = 0;         // It is possible to cause adjacent flags to be cleared.
TCON |= 0x40;    // It is possible to cause adjacent flags to be cleared.
TR0 = 1;         // It is possible to cause adjacent flags to be cleared.
```

If you want to avoid the above, it is recommended to use the interrupt bit characteristics. Most of the interrupt request flag can't be written 1 by which you can avoid clearing asynchronous adjacent flags.

For user who is develop program in C language, Timer 0 and Timer 1 can be enabled and disabled by TCONSET and TCONCLR macros. The macros are strongly recommended to change control bit, instead of set and clear control bits directly.

```
TCONSET(bit);    // The marco can set the bit of TCON. bit is 0~7.
TCONCLR(bit);    // The marco can clear the bit of TCON. bit is 0~7.
```

13.7 Timer 0 and Timer 1 Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
TCON	TF1	TR1	TF0	TR0	IE1	-	IE0	-
TCON0	T0EXT	T0RATE2	T0RATE1	T0RATE0	-	T1RATE2	T1RATE1	T1RATE0
TMOD	T1GATE	T1CT	T1M1	T1M0	T0GATE	T0CT	T0M1	T0M0
TH0	TH07	TH06	TH05	TH04	TH03	TH02	TH01	TH00
TL0	TL07	TL06	TL05	TL04	TL03	TL02	TL01	TL00
TH1	TH17	TH16	TH15	TH14	TH13	TH12	TH11	TH10
TL1	TL17	TL16	TL15	TL14	TL13	TL12	TL11	TL10

IEN0 Register (0xA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Interrupts enable. Refer to Chapter Interrupt
3	ET1	R/W	0	Timer 1 interrupt 0: Disable 1: Enable
1	ET0	R/W	0	Timer 0 interrupt 0: Disable 1: Enable
Else				Refer to other chapter(s)

TH0 / TH1 Registers (TH0: 0x8C, TH1: 0x8D)

Bit	Field	Type	Initial	Description
7..0	TH0/TH1	R/W	0x00	High byte of Timer 0 and Timer 1 counter

TL0 / TL1 Register (TL0: 0x8A, TL1: 0x8B)

Bit	Field	Type	Initial	Description
7..0	TL0/TL1	R/W	0x00	Low byte of Timer 0 and Timer 1 counter

TCON Register (0x88)

Bit	Field	Type	Initial	Description
7	TF1	R/W*(1)	0	Timer 1 overflow event 0: Timer 1 does not have any overflow event 1: Timer 1 has overflowed This bit can be cleared automatically by interrupt handler, or manually by firmware
6	TR1*(2)	R/W	0	Timer 1 function 0: Disable 1: Enable
5	TF0	R/W*(1)	0	Timer 0 overflow event 0: Timer 0 does not have any overflow event 1: Timer 0 has overflowed This bit can be cleared automatically by interrupt handler, or manually by firmware
4	TR0*(2)	R/W	0	Timer 0 function 0: Disable 1: Enable
Else				Refer to other chapter(s)

*(1) This bit can't write '1' value. The TCONCLR macro is strongly recommended to clear request flag.

*(2) TCONSET and TCONCLR macros are strongly recommended to change control bit.

TCON0 Register (0xE7)

Bit	Field	Type	Initial	Description
7	T0EXT	R/W	0	Timer 0 f_{EXT0} clock source selection. 0: fosc 1: flosc
6..4	T0RATE[2:0]	R/W	000	Clock divider of Timer 0 external clock source 000: f_{EXT0} / 128 001: f_{EXT0} / 64 010: f_{EXT0} / 32 011: f_{EXT0} / 16 100: f_{EXT0} / 8 101: f_{EXT0} / 4 110: f_{EXT0} / 2 111: f_{EXT0} / 1
3	Reserved	R	0	
2..0	T1RATE[2:0]	R/W	000	Clock divider of Timer 0 external clock source 000: f_{EXT1} / 128 001: f_{EXT1} / 64 010: f_{EXT1} / 32 011: f_{EXT1} / 16 100: f_{EXT1} / 8 101: f_{EXT1} / 4 110: f_{EXT1} / 2 111: f_{EXT1} / 1

TMOD Register (0x89)

Bit	Field	Type	Initial	Description
7	T1GATE	R/W	0	Timer 1 gate control mode 0: Disable 1: Enable, Timer 1 clock source is gated by INT1
6	T1CT	R/W	0	Timer 1 clock source selection 0: f_{Timer1} = fcpu / 12 1: $f_{Timer\ 1}$ = f_{EXT1} / T1RATE (refer to T1RATE) *(1)
5..4	T1M[1:0]	R/W	00	Timer 1 operation mode 00: 13-bit up counting timer 01: 16-bit up counting timer 10: 8-bit up counting timer with reload support 11: Reserved
3	T0GATE	R/W	0	Timer 0 gate control mode 0: Disable 1: Enable, Timer 0 clock source is gated by INT0
2	T0CT	R/W	0	Timer 0 clock source selection 0: f_{Timer0} = fcpu / 12 1: f_{Timer0} = f_{EXT0} / T0RATE (refer to T0RATE) *(2)
1..0	T0M[1:0]	R/W	00	Timer 0 operation mode 00: 13-bit up counting timer 01: 16-bit up counting timer 10: 8-bit up counting timer with reload support 11: Separated two 8-bit up counting timer

*(1) f_{EXT1} = fosc.

*(2) f_{EXT0} = fosc or flosc.

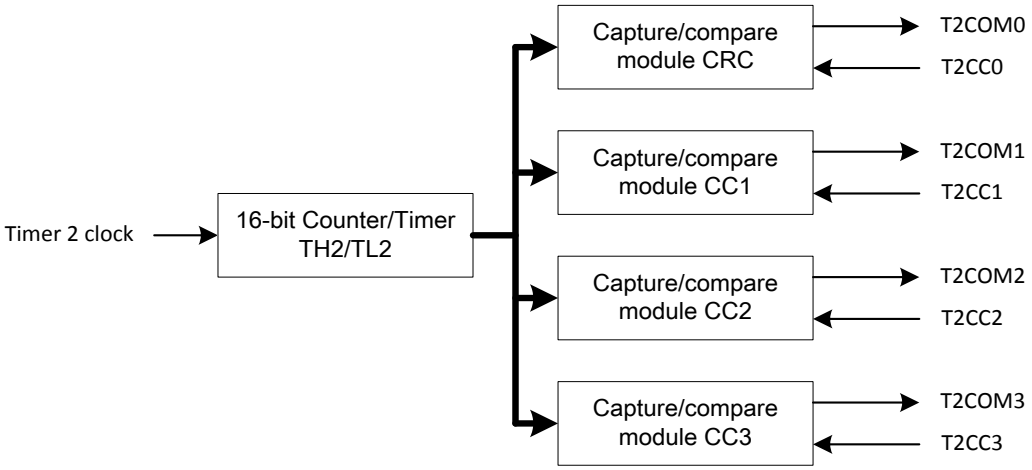
13.8 Sample Code

The following sample code demonstrates how to perform T0/T1 with interrupt.

```
#define T0Mode0     (0 << 0)  //T0 mode0, 13-bit counter
#define T0Mode1     (1 << 0)  //T0 mode1, 16-bit counter
#define T0Mode2     (2 << 0)  //T0 mode2, 8-bit auto-reload counter
#define T0Mode3     (3 << 0)  //T0 mode3, T0 two 8-bit counter/T1 no flag
#define T0GATE      (8 << 0)  //T0 gating clock by INT0
#define T0ClkFcpu    (0 << 0)  //T0 clock source from Fcpu/12
#define T0ClkExt     (4 << 0)  //T0 clock source from Fosc or FRTC
#define T0ExtFosc    (0 << 4)  //T0 clock source from Fosc
#define T0ExtFlosc    (8 << 4)  //T0 clock source from Flosc

#define T1Mode0     (0 << 4)  //T1 mode0, 13-bit counter
#define T1Mode1     (1 << 4)  //T1 mode1, 16-bit counter
#define T1Mode2     (2 << 4)  //T1 mode2, 8-bit auto-reload counter
#define T1Mode3     (3 << 4)  //T1 mode3, T1 stop
#define T1GATE      (8 << 4)  //T0 gating clock by INT1
#define T1ClkFcpu    (0 << 4)  //T0 clock source from Fcpu/12
#define T1ClkExt     (4 << 4)  //T0 clock source from Fosc

void InitT0T1(void)
{
  // T0/T1_Initial
  TH0 = 0x00;
  TL0 = 0x00;
  TH1 = 0x00;
  TL1 = 0x00;
  // T0 mode0 with gating clock by INT0, clock source from Fosc or Flosc
  TMOD |= T0Mode0 | T0GATE | T0ClkExT;
  // T0 clock source = FRTC/1
  TCON0 |= T0ExtFlosc | 0x70;
  // T1 mode1, clock source from Fcpu/12
  TMOD |= T1Mode1 | T1ClkFcpu;
  // Clear TF0/TF1
  TCON = 0x00;
  // Timer 0/1 enable
  TCONSET(4);
  TCONSET(6);
  // Enable T0/T1 interrupt
  IEN0 |= 0x0A;
  // Enable total interrupt
  IEN0 |= 0x80;

  P0 = 0x00;
  P0M = 0x03;
}

void T0Interrupt(void) interrupt ISRTimer0  //0x0B
{ //TF0 clear by hardware
  P00 = ~P00;
}
void T1Interrupt(void) interrupt ISRTimer1  //0x1B
{ //TF1 clear by hardware
  P01 = ~P01;
}
```

14 Timer 2

Timer 2 is a 16-bit up counting timer which has several optional extensions: specified reload value, comparison output (PWM) and capture function. Timer 2 consists of a dedicated 16-bit counter/timer and four 16-bit capture/compare modules. Each capture/compare module has its own associated I/O when enabled. Each capture/compare module may be configured to operate independently in one of 3 modes: compare, capture with rising edge, or capture with register be written.



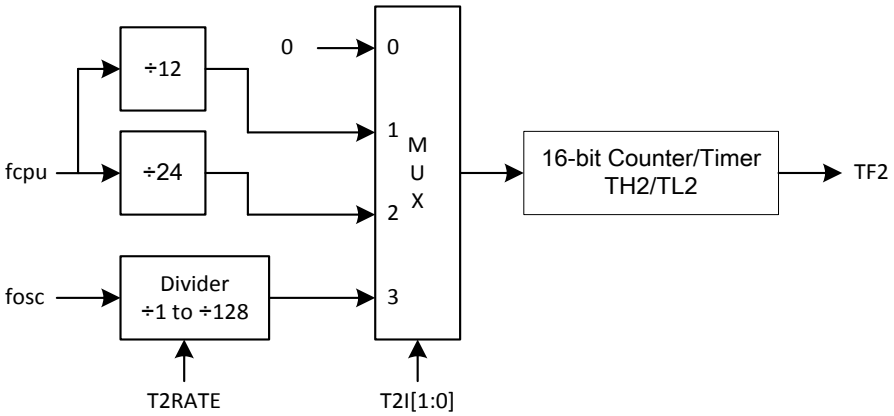
14.1 Timer 2 Up-counting Control

Timer 2 has three operation modes by its clock source: specify fcpu clocks (fcpu/12 and fcpu/24), and fosc clock (fosc/1 ~ fosc/128). The table below categorizes these three operation modes and its related registers (T2I1 and T2I0). Once the timer's counter is overflow (counts from 0xFFFF to 0x0000), TF2 would be issued immediately which can read/write by firmware. Timer 2 interrupt function is controlled by ET2.

T2I1	T2I0	Timer 2 Clock Source
0	0	Disable Timer 2
0	1	fcpu/12
1	0	fcpu/24
1	1	fosc/1 ~ fosc/128 (by T2RATE bits control)

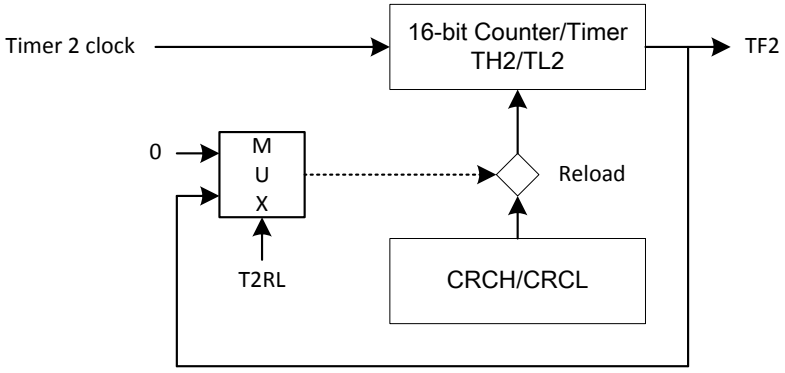
The clock speed of timer 2 can't be faster than the fcpu, otherwise the comparison output will be wrong. For example if the fosc runs at 32 MHz and the fcpu runs at 8 MHz, the divider must be divided by 4 when clock source is fosc.

$$\text{Timer 2 clock speed} \leq \text{fcpu}$$



14.2 Specified Timer 2 Reload Value

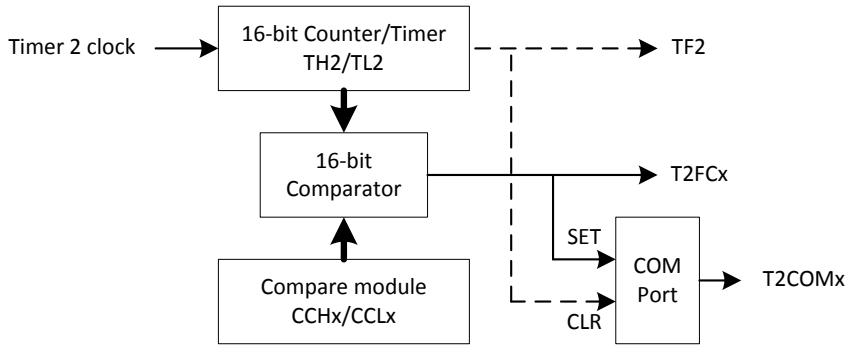
The specified reload value is an optional function which can reload Timer 2 counter by overflow. If overflow-to-reload is selected, Timer 2 duplicates CRCH/CRCL value to its counter (TH2/TL2) automatically by overflow signal. As a result, Timer 2 would repeatedly counts from CRCH/CRCL value to 0xFFFF.

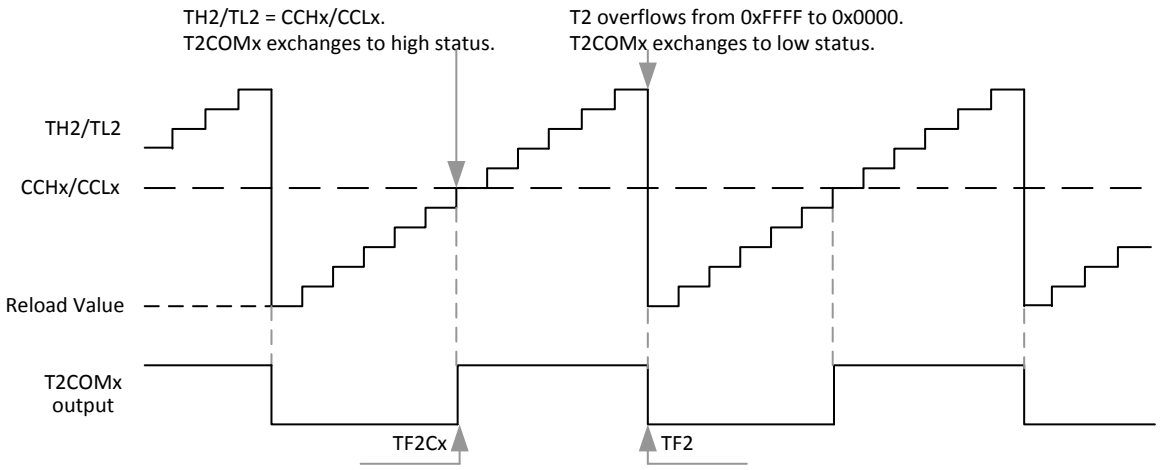


14.3 Comparison Output (PWM)

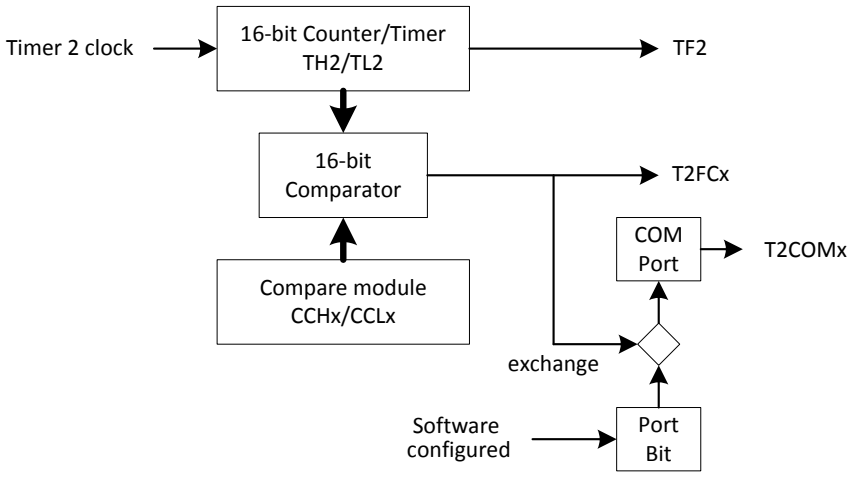
Timer 2 has up to four set of comparison output. Each set (CRC/CC1/CC2/CC3) independently compares its value to Timer 2 counter (TH2/TL2) and outputs the comparison result on T2COM0 to T2COM3 pins (shared with P0.0, P0.1, P0.7 and P1.0). The comparison result has two output methods: directly output and indirectly output.

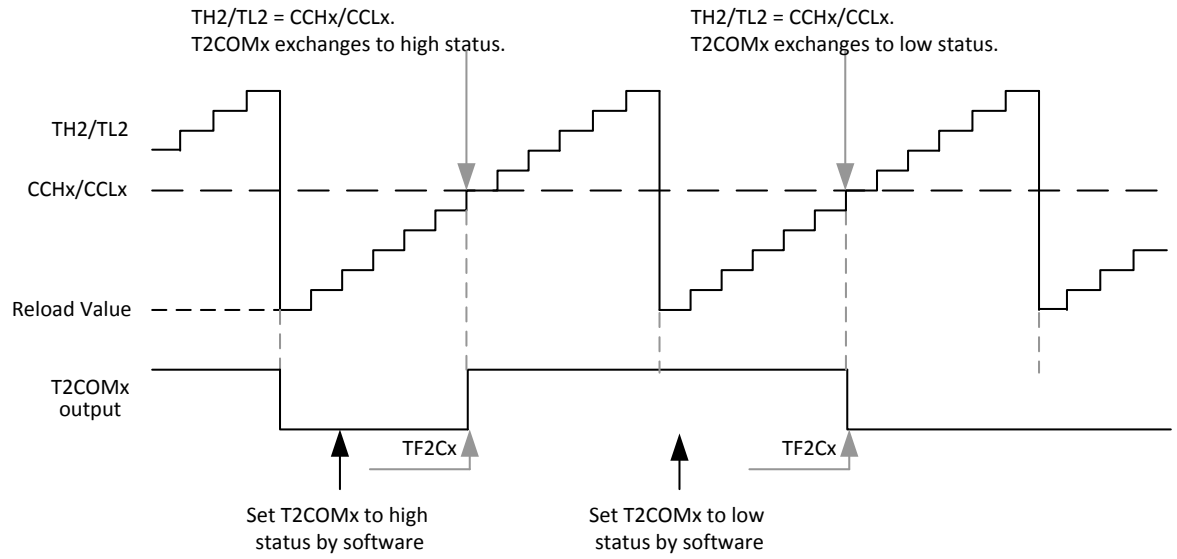
The directly method is that the mapped pin outputs low status if Timer 2 counter is lower than CRC/CC1/CC2/CC3 register, whereas it outputs high status if Timer 2 counter is equal/lager than CRC/CC1/CC2/CC3 register. Thus, the output status is changed twice at crossover points. As CRC/CC1/CC2/CC3 register is equal to Timer 2 counter, a TF2C0/TF2C1/TF2C2/TF2C3 flag is issued which can read/write by firmware. Compare interrupt function is controlled by ET2C0/ET2C1/ET2C2/ET2C3.





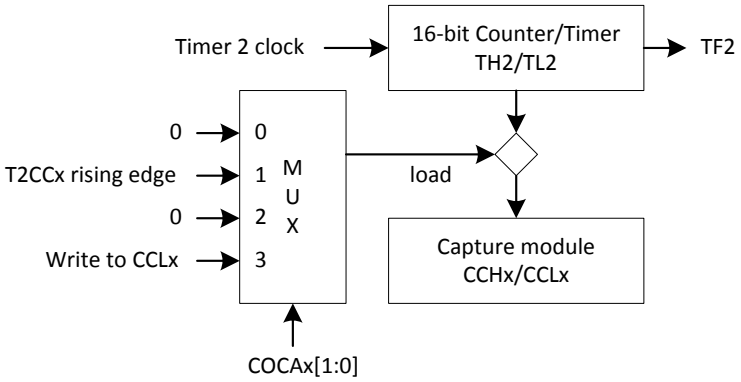
By contrast, the indirectly output method is an event which keep the mapped pin's previous output setting until Timer 2 counter overtakes CRC/CC1/CC2/CC3 register value. In this mode, the transition of the output signal can be configured by software. In other word, the P0.0 register bit would be affect T2COM0/P0.0 pin when TH2/TL2 equal to CRC registers. A Timer 2 overflow causes no output change.



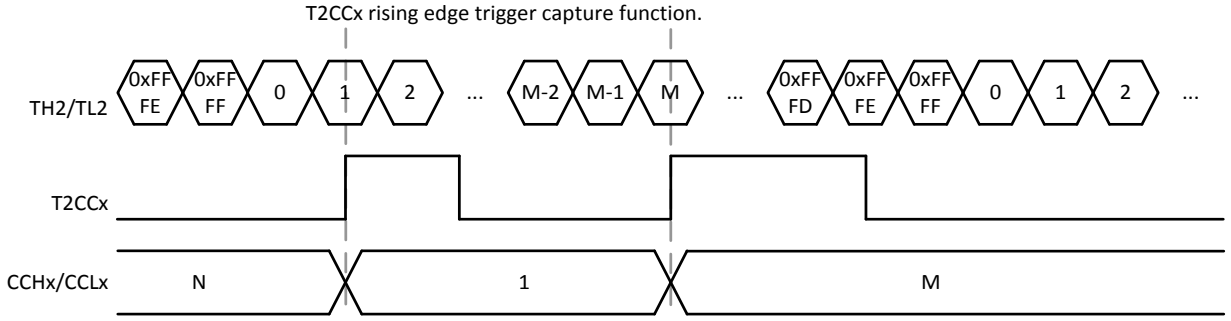


14.4 Capture Function

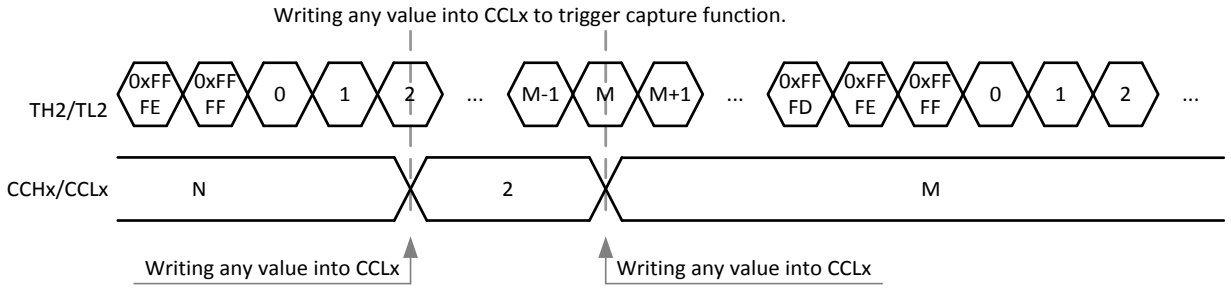
The capture function is similar to split/lap button of a stopwatch. While Timer 2 counter (TH2/TL2) routinely count up, a split event records counter value in CRC/CC1/CC2/CC3 register(s).



The split event can from hardware or software. The T2CC0 pin (shared with P0.5) can trigger a hardware split event that duplicates TH2/TL2 value to CRCH/CRCL registers, whereas T2CC1 (P0.6), T2CC2 (P0.2) and T2CC3 (P0.3) respectively control CC1 to CC3 registers.



A software split event is triggered by writing any value into CRCL/CCL1/CCL2/CCL3 register. While perform a writing instruction to these registers, the present TH2/TL2 value would be record in the paired registers instead.



14.5 Timer 2 Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
T2CON	-	I3FR	-	T2RL	-	T2CM	T2I1	T2I0
T2CON0	-	-	-	-	-	T2RATE2	T2RATE1	T2RATE0
CCEN	COCA31	COCA30	COCA21	COCA20	COCA11	COCA10	COCA01	COCA00
TH2	TH27	TH26	TH25	TH24	TH23	TH22	TH21	TH20
TL2	TL27	TL26	TL25	TL24	TL23	TL22	TL21	TL20
CRCH	CRCH7	CRCH6	CRCH5	CRCH4	CRCH3	CRCH2	CRCH1	CRCH0
CRCL	CRCL7	CRCL6	CRCL5	CRCL4	CRCL3	CRCL2	CRCL1	CRCL0
CCH3	CCH37	CCH36	CCH35	CCH34	CCH33	CCH32	CCH31	CCH30
CCL3	CCL37	CCL36	CCL35	CCL34	CCL33	CCL32	CCL31	CCL30
CCH2	CCH27	CCH26	CCH25	CCH24	CCH23	CCH22	CCH21	CCH20
CCL2	CCL27	CCL26	CCL25	CCL24	CCL23	CCL22	CCL21	CCL20
CCH1	CCH17	CCH16	CCH15	CCH14	CCH13	CCH12	CCH11	CCH10
CCL1	CCL17	CCL16	CCL15	CCL14	CCL13	CCL12	CCL11	CCL10
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
IEN1	-	-	ET2C3	ET2C2	ET2C1	ET2C0	ESPI	EI2C
IRCON	-	TF2	TF2C3	TF2C2	TF2C1	TF2C0	-	-

T2CON Register (0xC8)

Bit	Field	Type	Initial	Description
6	I3FR	R/W	0	In compare mode: 0: The COM0 interrupt would be generated when the TH2/TL2 becomes not equal to the CRC register (e.g. Timer 2: 0x8081, CRC: 0x8080). 1: The COM0 interrupt would be generated when the TH2/TL2 becomes equal to the CRC register. In capture mode 0: 0: The timer 2 content would be latched into CRC register by T2CC0 is falling edge. 1: The timer 2 content would be latched into CRC register by T2CC0 is rising edge.
5	Reserved	R/W	0	
4	T2RL	R/W	0	Specified Timer 2 reload value 0: Disable 1: Load CRCH/CRCL to TH2/TL2 by counter overflow
2	T2CM	R/W	0	Timer 2 comparison output 0: Directly output method 1: Indirectly output, next output status can be specified
1..0	T2I[1:0]	R/W	00	Timer 2 up counting control 00: Disable 01: Clock source is fcpu/12 10: Clock source is fcpu/24 11: Clock source is fosc / T2RATE (refer to T2RATE)
Else	Reserved	R	0	

T2CON0 Register (0xC9)

Bit	Field	Type	Initial	Description
2..0	T2RATE[2:0]	R/W	000	Clock divider of Timer 2 fosc clock source 000: fosc / 128 001: fosc / 64 010: fosc / 32 011: fosc / 16 100: fosc / 8 101: fosc / 4 110: fosc / 2 111: fosc / 1
Else	Reserved	R	0	

CCEN Register (0xC1)

Bit	Field	Type	Initial	Description
7..6	COCA3[1:0]	R/W	00	Comparison and capture function of CC3 00: Disable 01: Capture by T2CC3 pin rising edge 10: Comparison function 11: Capture by writing CCL3 register
5..4	COCA2[1:0]	R/W	00	Comparison and capture function of CC2 00: Disable 01: Capture by T2CC2 pin rising edge 10: Comparison function 11: Capture by writing CCL2 register
3..2	COCA1[1:0]	R/W	00	Comparison and capture function of CC1 00: Disable 01: Capture by T2CC1 pin rising edge 10: Comparison function 11: Capture by writing CCL1 register
1..0	COCA0[1:0]	R/W	00	Comparison and capture function of CRC 00: Disable 01: Capture by T2CC0 pin rising edge 10: Comparison function 11: Capture by writing CRCL register

TH2/TL2 Registers (TH2: 0xCD, TL2: 0xCC)

Bit	Field	Type	Initial	Description
7..0	TH2/TL2	R/W	0x00	Timer 2 16-bit counter registers.

CRC Registers (CRCH: 0xCB, CRCL: 0xCA)

Bit	Field	Type	Initial	Description
7..0	CRCH[15:0]	R/W	0x00	16-bit compare/capture registers.

CCH3/CCL3 Registers (CCH3: 0xC7, CCL3: 0xC6)

Bit	Field	Type	Initial	Description
7..0	CCH3/CCL3	R/W	0x00	16-bit compare/capture registers.

CCH2/CCL2 Registers (CCH2: 0xC5, CCL2: 0xC4)

Bit	Field	Type	Initial	Description
7..0	CCH2 /CCL2	R/W	0x00	16-bit compare/capture registers.

CCH1/CCL1 Registers (CCH1: 0xC3, CCL1: 0xC2)

Bit	Field	Type	Initial	Description
7..0	CCH1/CCL1	R/W	0x00	16-bit compare/capture registers.

IEN0 Register (0xA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Interrupts enable. Refer to Chapter Interrupt
5	ET2	R/W	0	Enable Timer 2 interrupt
Else				Refer to other chapter(s)

IEN1 Register (0xB8)

Bit	Field	Type	Initial	Description
5	ET2C3	R/W	0	T2 Timer COM3 interrupt control bit 0: Disable 1: Enable
4	ET2C2	R/W	0	T2 Timer COM2 interrupt control bit 0: Disable 1: Enable
3	ET2C1	R/W	0	T2 Timer COM1 interrupt control bit 0: Disable 1: Enable
2	ET2C0	R/W	0	T2 Timer COM0 interrupt control bit 0: Disable 1: Enable
Else				Refer to other chapter(s)

IRCON Register (0xC0)

Bit	Field	Type	Initial	Description
6	TF2	R/W*	0	T2 timer interrupt request flag. 0: None T2 interrupt request. 1: T2 interrupt request.
5	TF2C3	R/W*	0	T2 Timer COM3 interrupt request flag. 0: None T2COM3 interrupt request. 1: T2COM3 interrupt request.
4	TF2C2	R/W*	0	T2 Timer COM2 interrupt request flag. 0: None T2COM2 interrupt request. 1: T2COM2 interrupt request.
3	TF2C1	R/W*	0	T2 Timer COM1 interrupt request flag. 0: None T2COM1 interrupt request. 1: T2COM1 interrupt request.
2	TF2C0	R/W*	0	T2 Timer COM0 interrupt request flag. 0: None T2COM0 interrupt request. 1: T2COM0 interrupt request.
Else	Reserved	R	0	

* This bit can't write '1' value. The IRCONCLR macro is strongly recommended to clear request flag.

14.6 Sample Code

The following sample code demonstrates how to perform T2 compare function with interrupt.

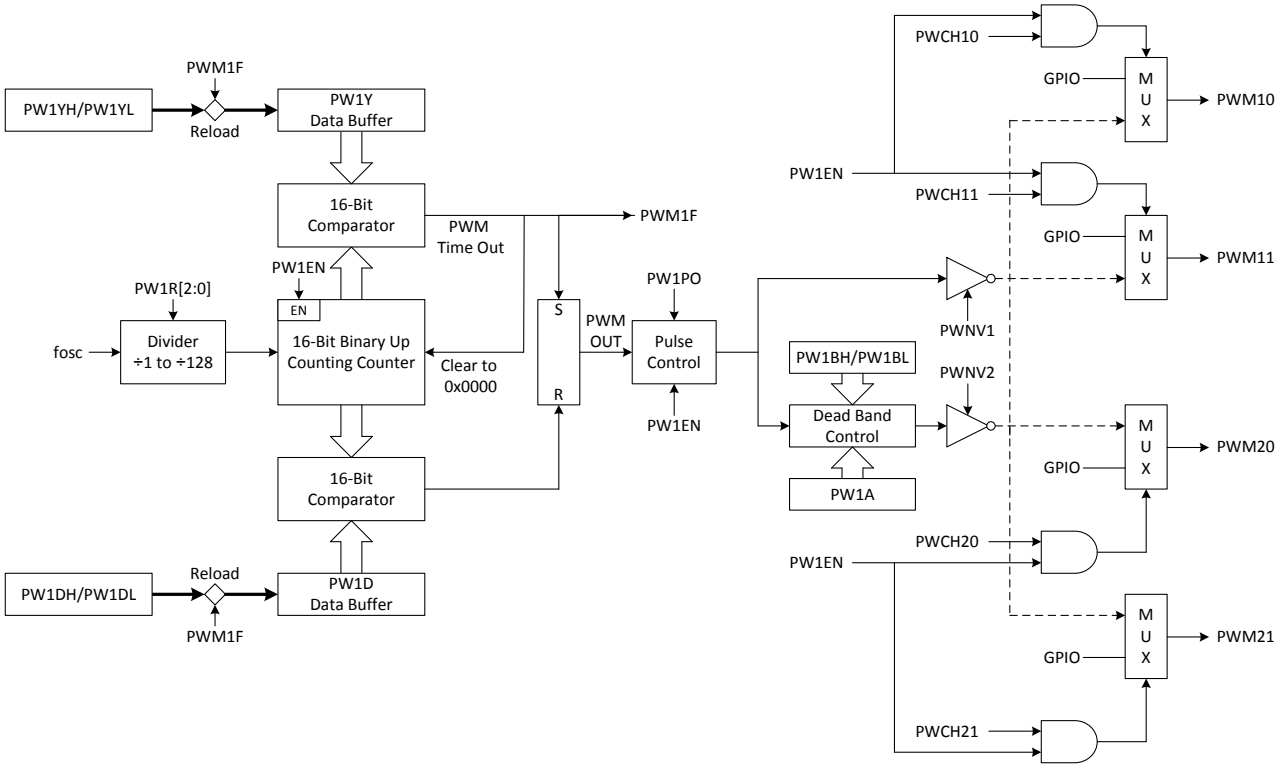
```
#define T2ClkFcpu12  (1 << 0)  //T2 clock from Fcpu/12
#define T2ClkFcpu24  (2 << 0)  //T2 clock from Fcpu/24
#define T2ClkFosc    (3 << 0)  //T2 clock from Fosc/N
#define T2RLMode     (2 << 3)  //T2 reload mode = auto-reload
#define ComMode0     (0 << 2)  //Compare mode = directly method
#define ComMode1     (1 << 2)  //Compare mode = indirectly output method
#define T2COM0EdNE   (0 << 6)  //T2COM0 interrupt edge = no equle CRC
#define T2COM0EdE    (1 << 6)  //T2COM0 interrupt edge = equle CRC
#define T2COM0En     (2 << 0)  //T2COM0 compare funcion enable
#define T2COM1En     (2 << 2)  //T2COM1 compare funcion enable
#define T2COM2En     (2 << 4)  //T2COM2 compare funcion enable
#define T2COM3En     (2 << 6)  //T2COM3 compare funcion enable

void InitT2(void)
{
  // T2_Initial
  TH2 = 0x00;
  TL2 = 0x00;
  CRCH = 0x80;
  CRCL = 0x00;
  CCH1 = 0xC0;
  CCL1 = 0x00;
  CCH2 = 0xE0;
  CCL2 = 0x00;
  CCH3 = 0xF0;
  CCL3 = 0x00;

  // T2 clock from Fosc/2
  // Reload mode = auto-reload
  // Compare mode = directly method
  // T2COM0 interrupt trigger = equle CRC
  T2CON |= T2ClkFosc | T2RLMode | ComMode0 | T2COM0EdE;
  // Clock divider of Timer fosc clock source
  T2CON0 |= 0x06;

  // Compare function T2COM0/1/2/3 enable
  CCEN |= T2COM0En | T2COM1En | T2COM2En | T2COM3En;

  // Enable T2COM0/1/2/3 interrupt
  IEN1 |= 0x3C;

  // Enable total/Timer2 interrupt
  IEN0 |= 0xA0;

  P2 = 0x00;
  P2M = 0x3D;
}

void T2Interrupt(void) interrupt ISRTimer2  //0x2B
{ //TF2 clear by hardware
  P20 = ~P20;
}

```

```
54 void T2COM0Interrupt(void) interrupt ISRCom0  //0x53
55 { //TF2C0 clear by hardware
56   P22 = ~P22;
57 }
58
59 void T2COM1Interrupt(void) interrupt ISRCom1  //0x5B
60 { //TF2C1 clear by hardware
61   P23 = ~P23;
62 }
63
64 void T2COM2Interrupt(void) interrupt ISRCom2  //0x63
65 { //TF2C2 clear by hardware
66   P24 = ~P24;
67 }
68
69 void T2COM3Interrupt(void) interrupt ISRCom3  //0x6B
70 { //TF2C3 clear by hardware
71   P25 = ~P25;
72 }
```

15 PWM

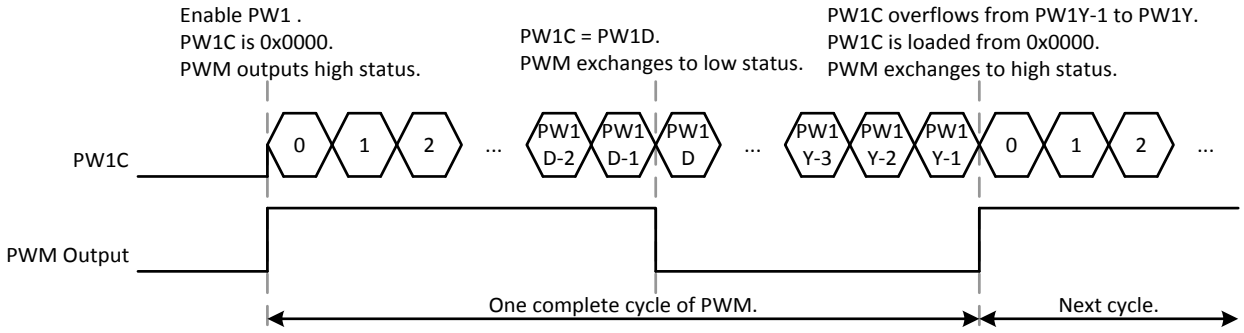
The PW1 timer includes a 16-bit binary up 4-channel PWM, and one pulse PWM functions. By the counter reaches the up-boundary value (PW1Y), it clears its counter and triggers an interrupt signal. PWM's duty cycle is controlled by PW1D register.

The PWM also support one pulse output signal which can disables itself by the end of first PWM cycle. Thus, only one pulse would be generated in this condition. The PWM has four programmable channels shared with GPIO pins and controlled by PW1CH register. The output operation must be through enabled each bit/channel of PW1CH register. The enabled PWM channel exchanges from GPIO to PWM output. When the bits of PW1CH register disables the PWM channel returns to last status of GPIO mode. The PWM build in IDLE Mode wake-up function if interrupt enable. When PW1 timer overflow occurs (counts from PW1Y-1 to PW1Y), PWM1F would be issued immediately which can read/write by firmware. PWM clock source is fosc, and divided by 1 to 128 times which is controlled by PW1R[2:0] bits. PW1 interrupt function is controlled by EPWM1.



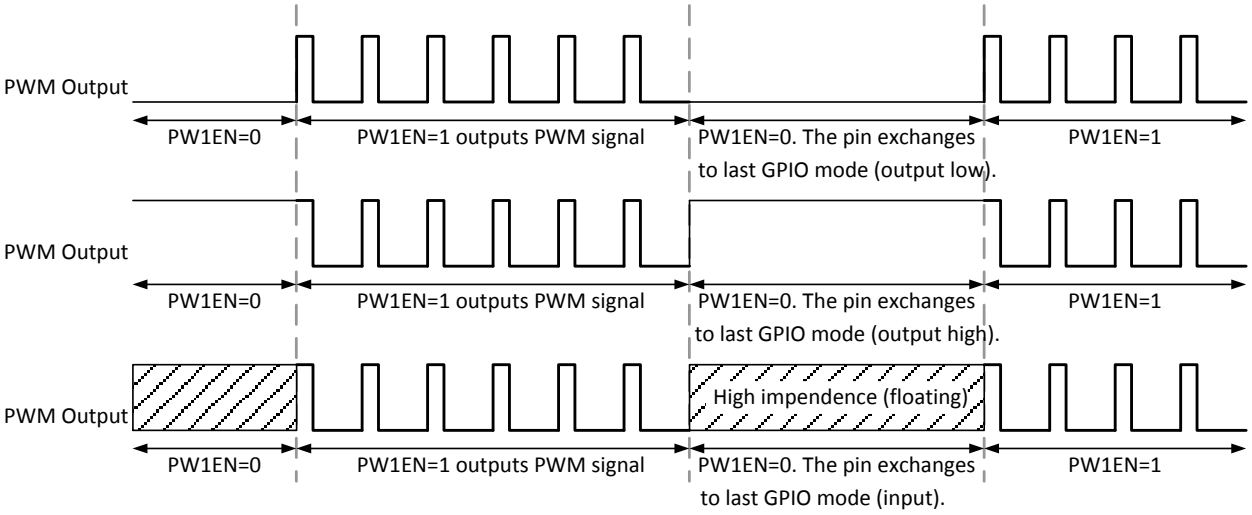
15.1 General PWM

PW1 timer builds in PWM function controlled by PW1EN and PW1CH register. PWM10, PWM11, PWM20 and PWM21 are output pins (shared with P0.2, P0.3, P2.0 and P2.1). The PWM output pins are shared with GPIO pin controlled by PW1CH register. When output PWM function, we must be set PW1EN = 1. When PWM output signal synchronize finishes, the PWM channel exchanges from GPIO to PWM output. When PW1EN = 0, the PWM channel returns to GPIO mode and last status. PWM signal is generated from the result of PW1Y and PW1D comparison combination. When PW1C starts to count or returns to 0x0000, the PWM outputs high status which is the PWM initial status. PW1C is loaded new data from PW1Y register to decide PWM cycle and resolution. PW1C keeps counting, and the system compares PW1C and PW1D. When PW1C = PW1D, the PWM output status exchanges to low PW1C keeps counting. When PWM timer overflow occurs (PW1Y-1 to PW1Y), and one cycle of PWM signal finishes. PW1C is reloaded from 0x0000 automatically, and PWM output status exchanges to high for next cycle. PW1D decides the high duty duration, and PW1Y decides the resolution and cycle of PWM. PW1D can't be larger than PW1Y, or the PWM signal is error.



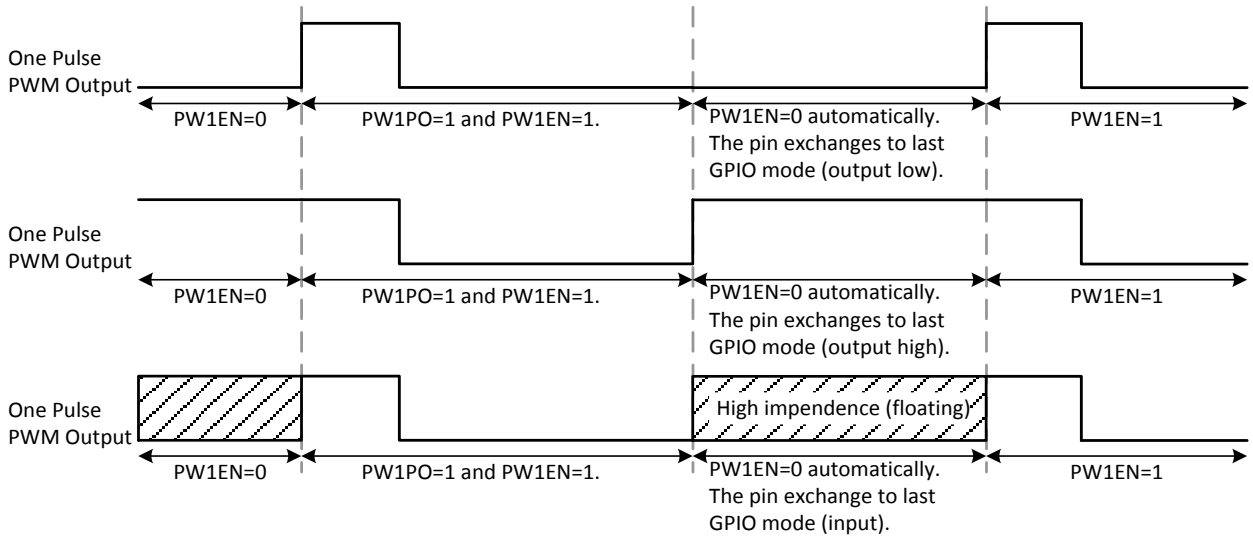
PWM Period = PW1Y

PWM duty = (PW1D): (PW1Y-PW1D)



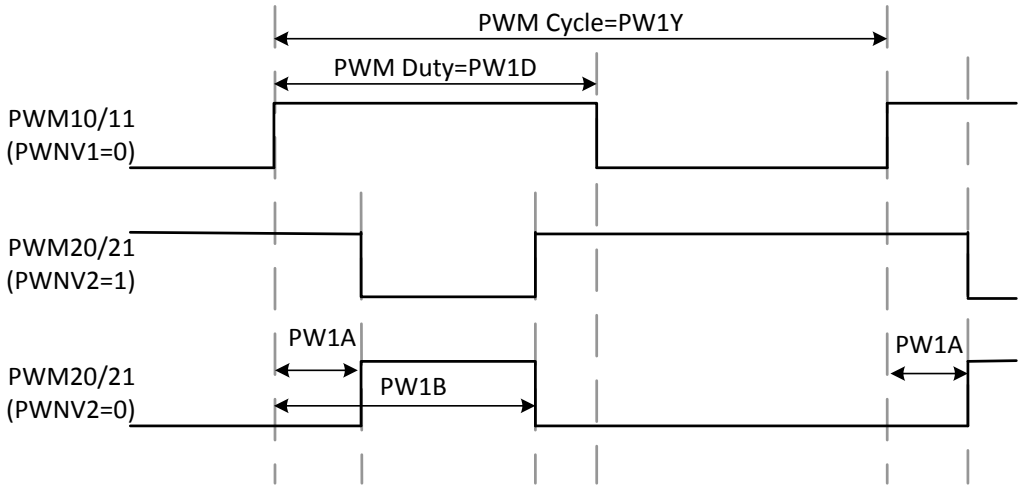
15.2 One Pulse PWM

When PW1PO = 0, PW1 is PWM function mode. When PW1PO = 1 and PW1EN = 1, PW1 will output one pulse PWM function and the PWM1F is issued as PW1 counter overflow. PW1EN bit is cleared automatically and pulse output pin returns to idle status. To output next pulse is to set PW1EN bit by program again. One pulse PWM channels selected by PW1CH register. When output one pulse PWM function, we must be set PW1PO = 1 and PW1EN=1. When one pulse PWM output signal synchronize finishes, the PWM channel exchanges from GPIO to PWM output. When one pulse PWM output finishes, PW1EN = 0, the PWM channel returns to GPIO mode and last status.



15.3 Inverse and Dead Band

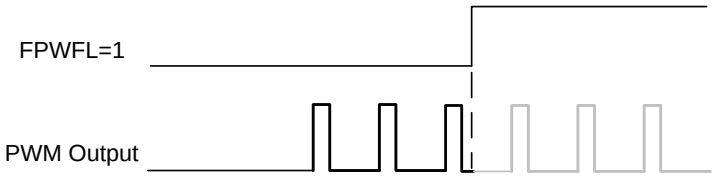
The PWM builds in inverse output function. The PWM has one inverse PWM signal as PWNV =1. When PWNV = 1, the PWM outputs the inverse PWM signal of PW1. When PWNV = 0, the PWM outputs the non-inverse PWM signal of PW1. The inverse PWM output waveform is below diagram.



The PWM dead band occurs in PWM high pulse width, and the dead band period is programmable from PW1A and PW1D-PW1B registers. The dead band period is symmetrical at left-right terminal of PWM pulse width or not. If the bead band period is longer than PWM duty, the PWM is no output.

15.4 Fail Detect

The PWM builds in fail detect function. PWM fail detect function share the same signal input pin, edge select bit and interrupt vector with INT0. When PWFLEN=1, fail detect function is enabled. The fail event is triggered by external signal from PWFL pin (P3.0, shared with INT0). Trigger edge is selected by PEDGE[1:0]. If fail event occurs, the PWM fail detect flag FPWFL is set by hardware. In the same time, PWM function is disabled and PW1EN bit is cleared automatically. FPWFL is only clear by software.



Fail detect function supports interrupt only if INT0 interrupt function is also enabled. When EX0=1 and PWFLEN=1, the program counter is pointed to 0x03 to execute ISR after INT0/PWMFL pin trigger event occurrence. Check FPWFL flag to identify if PWM event or not.

15.5 PWM Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PW1M	PW1EN	PW1R2	PW1R1	PW1R0	PWNV2	PWNV1	-	PW1PO
PW1CH	-	-	PWCH21	PWCH20	-	-	PWCH11	PWCH10
PW1YH	PW1Y15	PW1Y14	PW1Y13	PW1Y12	PW1Y11	PW1Y10	PW1Y9	PW1Y8
PW1YL	PW1Y7	PW1Y6	PW1Y5	PW1Y4	PW1Y3	PW1Y2	PW1Y1	PW1Y0
PW1DH	PW1D15	PW1D14	PW1D13	PW1D12	PW1D11	PW1D10	PW1D9	PW1D8
PW1DL	PW1D7	PW1D6	PW1D5	PW1D4	PW1D3	PW1D2	PW1D1	PW1D0
PW1BH	PW1B15	PW1B14	PW1B13	PW1B12	PW1B11	PW1B10	PW1B9	PW1B8
PW1BL	PW1B7	PW1B6	PW1B5	PW1B4	PW1B3	PW1B2	PW1B1	PW1B0
PW1A	PW1A7	PW1A6	PW1A5	PW1A4	PW1A3	PW1A2	PW1A1	PW1A0
PWFLM	-	-	-	-	-	-	FPEFL	PWFLEN
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
IEN2	-	-	-	-	EPWM1	EX2	-	EADC
IRCON2	-	-	-	-	PWM1F	IE2	-	ADCF

PW1M Registers (PW1M: 0xAB)

Bit	Field	Type	Initial	Description
7	PW1EN	R/W	0	PW1 function 0: Disable 1: Enable*
6..4	PW1R[2:0]	R/W	000	PWM timer clock source 000: fosc / 128 001: fosc / 64 010: fosc / 32 011: fosc / 16 100: fosc / 8 101: fosc / 4 110: fosc / 2 111: fosc / 1
3	PWNV2	R/W	0	PWM20/21/22 pins output control 0: Non-inverse 1: Inverse
2	PWNV1	R/W	0	PWM10/11/12 pins output control 0: Non-inverse 1: Inverse
1	Reserved	R/W	0	
0	PW1PO	R/W	0	One pulse function 0: Disable 1: Enable

* When the period is setting 0x0000, after PWM is set enable bit, the PWM will stop and the period can’t update.

PW1CH Register (0xBE)

Bit	Field	Type	Initial	Description
7..6	Reserved	R/W	0	
5 4	PWCH21 PWCH20	R/W	0	PWM1 shared-pin control 0: GPIO 1: PWM output (shared with P0.3/ P2.1)
3..2	Reserved	R/W	0	
1 0	PWCH11 PWCH10	R/W	0	PWM1 shared-pin control 0: GPIO 1: PWM output (shared with P0.2/ P2.0)

PW1YH/PW1YL Registers (PW1YH: 0xAD, PW1YL: 0xAC)

Bit	Field	Type	Initial	Description
7..0	PW1YH/L	R/W	0x00	16-bit PWM1 period control*.

* The period configuration must be setup completely before starting PWM function.

PW1DH/PW1DL Registers (PW1DH: 0xBC, PW1DL: 0xBB)

Bit	Field	Type	Initial	Description
7..0	PW1DH/L	R/W	0x00	16-bit PWM1 duty control.

PW1BH/PW1BL Registers (PW1BH: 0xAF, PW1BL: 0xAE)

Bit	Field	Type	Initial	Description
7..0	PW1BH/L	R/W	0x00	16-bit PWM1 dead band control.

PW1A Register (PW1A: 0xBD)

Bit	Field	Type	Initial	Description
7..0	PW1A	R/W	0x00	8-bit PWM1 dead band control.

PWFLM Register (0XB7)

Bit	Field	Type	Initial	Description
1	FPWFL	R/W	0	PWM1 fail detect flag. 0: None fail detect request. 1: PWM1 fail detect request.
0	PWMFLM			PWM1 fail detect function 0: Disable 1: Enable
Else				Refer to other chapter(s)

IEN0 Register (0xA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Interrupts enable. Refer to Chapter Interrupt
Else				Refer to other chapter(s)

IEN2 Register (0X9A)

Bit	Field	Type	Initial	Description
3	EPWM1	R/W	0	PWM1 interrupt control bit. 0 = Disable PWM1 interrupt function. 1 = Enable PWM1 interrupt function.
Else				Refer to other chapter(s)

IRCON2 Register (0XBF)

Bit	Field	Type	Initial	Description
3	PWM1F	R/W*	0	PWM1 interrupt request flag. 0: None PWM1 interrupt request 1: PWM1 interrupt request.
Else				Refer to other chapter(s)

* This bit can't write '1' value. The IRCON2CLR macro is strongly recommended to clear request flag.

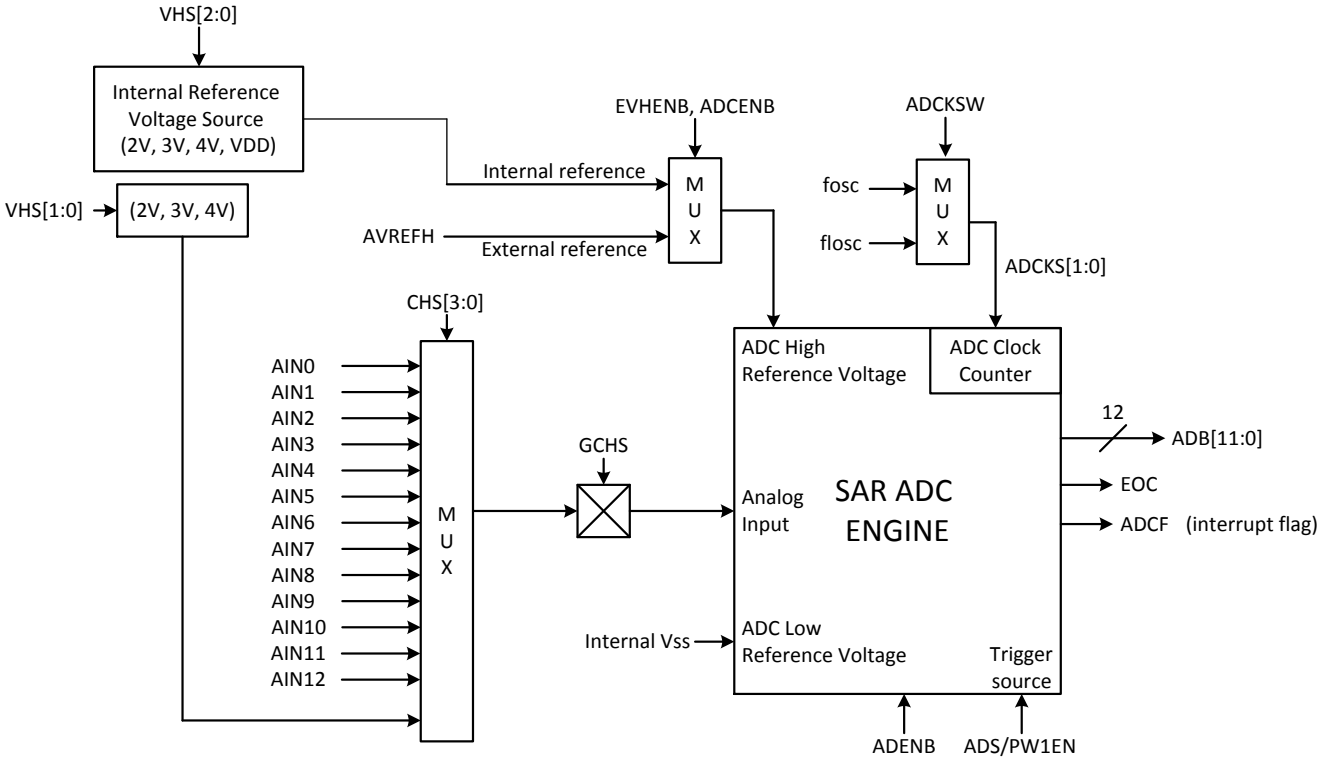
15.6 Sample Code

The following sample code demonstrates how to perform PW1 with interrupt.

```
#define PW1Inv1      (1 << 2)  //PWM10/11 output inverse
#define PW1Inv2      (1 << 3)  //PWM20/21 output inverse
#define PW1OnePu     (1 << 0)  //Enable PW1 pulse output function
#define PWM10En      (1 << 0)  //Enable PWM10 output function
#define PWM11En      (2 << 0)  //Enable PWM11 output function
#define PWM20En      (1 << 4)  //Enable PWM20 output function
#define PWM21En      (2 << 4)  //Enable PWM21 output function
#define PW1En         (1 << 7)  //Enable PWM1 function

void InitPWM(void)
{
  // PWM1_Initial
  PW1YH = 0x80;
  PW1YL = 0x00;
  PW1DH = 0x40;
  PW1DL = 0x00;
  PW1BH = 0x20;
  PW1BL = 0x00;
  PW1A = 0x80;

  // PW10/11/20/21 channel enable
  PW1CH = PWM10En | PWM11En | PWM20En | PWM21En;

  // PWM1 enable, P10/11 output inverse, clock = Fosc/32
  PW1M = PW1En | PW1Inv1 | 0x20;

  // Enable PWM1 interrupt & clear PWM1F
  IEN2 = 0x08;
  IRCIN1 = 0x00;

  // Enable total interrupt
  IEN0 |= 0x80;

  P0 = 0x00;
  P0M |= 0x01;
}

void PW1Interrupt(void) interrupt ISRPwm1  //0x93
{ //PWM1F clear by hardware
  P00 = ~P00;
}
```

16 ADC

The analog to digital converter (ADC) is SAR structure with 13-input sources and up to 4096-step resolution to transfer analog signal into 12-bits digital buffers. The ADC builds in 13-channel input source to measure 13 different analog signal sources. The ADC resolution is 12-bit. The ADC has four clock rates to decide ADC converting rate. The ADC reference high voltage includes 5 sources. Four internal power source including VDD, 4V, 3V and 2V. The other one is external reference voltage input pin from AVREFH pin. The ADC builds in P1CON/P2CON/P3CON registers to set pure analog input pin. After setup ADENB and ADS bits, the ADC starts to convert analog signal to digital data. Besides ADS bit can start to convert analog signal, PW1EN also have convert analog signal ADC function. ADC can work in idle mode. After ADC operating, the system would be waked up from idle mode to normal mode if interrupt enable. When ADC clock source is flosc and STWK=1, ADC can work in stop mode and waked up from stop mode by ADC interrupt.



16.1 Configurations of Operation

These configurations must be setup completely before starting ADC converting. ADC is configured using the following steps:

1. Choose and enable the start of conversion ADC input channel. (By CHS[3:0] bits and GCHS bit)
2. The GPIO mode of ADC input channel must be set as input mode. (By PnM register)
3. The internal pull-up resistor of ADC input channel must be disabled. (By PnUR register)
4. The configuration control bit of ADC input channel must be set. (By PnCON register)
5. Choose ADC high reference voltage. (By VREFH register)
6. Choose ADC Clock Source and Clock Rate. (By ADCKSW and ADCKS[1:0] bits)
7. After setup ADENB bits, the ADC ready to convert analog signal to digital data.

16.1.1 Start to Conversion

When ADC IP is enabled by ADENB bit, it is necessary to make an ADC start-up by program. Writing a 1 to the ADS bit of register ADM. After setup ADENB and ADS bits, the ADC starts to convert analog signal to digital data. The ADS bit is reset to logic 0 when the conversion is complete. When the conversion is complete, the ADC circuit will set EOC and ADCF bits to “1” and the digital data outputs in ADB and ADR registers. If ADC interrupt function is enabled (EADC = 1), the ADC interrupt request occurs and executes interrupt service routine when ADCF is “1” after ADC converting. Clear ADCF by hardware automatically in interrupt procedure. Note that when ADPWS bit is “1”, if PWM enable trigger be used as the conversion source, the ADC will continuous conversions until PWM is disabled.

16.2 ADC input channel

The ADC builds in 13-channel input source (AIN0 – AIN12) to measure 13 different analog signal sources controlled by CHS[3:0] and GCHS bits. AIN13 channel is reserved. The AIN14 is internal 2V or 3V or 4V input channel. There is no any input pin from outside. In this time ADC reference voltage must be internal VDD and External voltage, not internal 2V or 3V or 4V. AIN14 can be a good battery detector for battery system. To select appropriate internal AVREFH level and compare value, a high performance and cheaper low battery detector is built in the system.

CHS[3:0]	Channel	Pin name	Remark
0000	AIN0	P3.0	-
0001	AIN1	P3.1	-
0010	AIN2	P2.0	-
0011	AIN3	P2.1	-
0100	AIN4	P2.2	-
0101	AIN5	P2.3	-
0110	AIN6	P2.4	-
0111	AIN7	P2.5	-
1000	AIN8	P2.6	-
1001	AIN9	P2.7	-
1010	AIN10	P1.7	-
1011	AIN11	P1.6	-
1100	AIN12	P1.5	-
1101	AIN13	-	Reserved
1110 - 1111	AIN14	Internal 2V or 3V or 4V	Battery detector channel

16.2.1 Pin Configuration

ADC input channels are shared with Port1, Port2 and Port3. ADC channel selection is through CHS[3:0] bit. Only one pin of Port1, Port2 and Port3 can be configured as ADC input in the same time. The pins of Port1, Port2 and Port3 configured as ADC input channel must be set input mode, disable internal pull-up and enable P1CON/P2CON/P3CON first by program. After selecting ADC input channel through CHS[3:0], set GCHS bit as “1” to enable ADC channel function.

ADC input pins are shared with digital I/O pins. Connect an analog signal to CMOS digital input pin, especially, the analog signal level is about 1/2 VDD will cause extra current leakage. In the power down mode, the above leakage current will be a big problem. Unfortunately, if users connect more than one analog input signal to Port1, Port2 or Port3 will encounter above current leakage situation. Write “1” into PnCON register will configure related pin as pure analog input pin to avoid current leakage.

Note that When ADC pin is general I/O mode, the bit of P1CON, P2CON and P3CON must be set to “0”, or the digital I/O signal would be isolated.

16.3 Reference Voltage

The ADC builds in five high reference voltage source controlled through VREFH register. There are one external voltage source and four internal voltage source (VDD, 4V, 3V, 2V). When EVHENB bit is “1”, ADC reference voltage is external voltage source from AVREFH/P3.0. In the condition, P3.0 GPIO mode must be set as input mode and disable internal pull-up resistor.

If EVHENB bit is “0”, ADC reference high voltage is from internal voltage source selected by VHS[1:0] bits. If VHS[1:0] is “11”, ADC reference high voltage is VDD. If VHS[1:0] is “10”, ADC reference high voltage is 4V. If VHS[1:0] is “01”, ADC reference high voltage is 3V. If VHS[1:0] is “00”, ADC reference high voltage is 2V. The limitation of internal high reference voltage application is VDD can’t below each of internal high voltage level, or the level is equal to VDD. If AIN14 channel is selected as internal 2V or 3V or 4V input channel. There is no any input pin from outside. In this time ADC reference high voltage must be internal VDD or External voltage, not internal 2V/3V/4V.

16.3.1 Signal Format

ADC sampling voltage range is limited by high/low reference voltage. The ADC low reference voltage is VSS. The ADC high reference voltage includes internal VDD/4V/3V/2V and external reference voltage source from P3.0/AVREFH pin controlled by EVHENB bit. ADC reference voltage range limitation is “(ADC high reference voltage - low reference voltage) ≥ 2V”. ADC low reference voltage is VSS = 0V. So ADC high reference voltage range is 2V to VDD. The range is ADC external high reference voltage range.

- ADC Internal Low Reference Voltage = 0V.
- ADC Internal High Reference Voltage = VDD/4V/3V/2V. (EVHENB=0)
- ADC External High Reference Voltage = 2V to VDD. (EVHENB=1)

ADC sampled input signal voltage must be from ADC low reference voltage to ADC high reference. If the ADC input signal voltage is over the range, the ADC converting result is error (full scale or zero).

- ADC Low Reference Voltage ≤ ADC Sampled Input Voltage ≤ ADC High Reference Voltage

16.4 Converting Time

The ADC converting time is from ADS=1 (Start to ADC convert) to EOC=1 (End of ADC convert). The converting time duration is depend on ADC clock rate. 12-bit ADC’s converting time is
1/ (ADC clock /4)*16 sec. ADC has two clock sources: fosc of flosc, which is controlled by ADCKSW bit. ADCKS[1:0] bits: 00 = fosc/16 or flosc/16, 01 = fosc/8 or flosc/8, 10 = fosc/1or flosc/1, 11 = fosc/2 or flosc/2.

The ADC converting time affects ADC performance. If input high rate analog signal, it is necessary to select a high ADC converting rate. If the ADC converting time is slower than analog signal variation rate, the ADC result would be error. So to select a correct ADC clock rate to decide a right ADC converting rate is very important.

$$\text{12 bits ADC conversion time} = \frac{16}{\text{ADC clock rate}/4}$$

When ADCKSW=0:

ADCKS[1:0]	ADC clock rate	fosc = 16MHz		fosc = 32MHz	
		Converting time	Converting rate	Converting time	Converting rate
00	fosc/16	1/(16MHz/16/4)*16 = 64us	15.625kHz	1/(32MHz/16/4)*16 = 32us	31.25kHz
01	fosc/8	1/(16MHz/8/4)*16 = 32us	31.25kHz	1/(32MHz/8/4)*16 = 16us	62.5kHz
10	fosc	1/(16MHz/4)*16 = 4us	250kHz	1/(32MHz/4)*16 = 2us	500kHz
11	fosc/2	1/(16MHz/2/4)*16 = 8us	125kHz	1/(32MHz/2/4)*16 = 4us	250kHz

When ACKSW=1:

ADCKS[1:0]	ADC clock rate	flosc = 16KHz		flosc = 32.768KHz	
		Converting time	Converting rate	Converting time	Converting rate
00	flosc/16	1/(16KHz/16/4)*16 = 64ms	15.625Hz	1/(32.768KHz/16/4)*16 = 31.25ms	32Hz
01	flosc/8	1/(16KHz/8/4)*16 = 32ms	31.25Hz	1/(32.768KHz/8/4)*16 = 15.625ms	64Hz
10	flosc	1/(16KHz/4)*16 = 4ms	250Hz	1/(32.768KHz/4)*16 = 1.953ms	512Hz
11	flosc/2	1/(16KHz/2/4)*16 = 8ms	125Hz	1/(32.768KHz /2/4)*16 = 3.906ms	256Hz

16.5 Data Buffer

ADC data buffer is 12-bit length to store ADC converter result. The high byte is ADB register, and the low-nibble is ADR[3:0] bits. The ADB register is only 8-bit register including bit 4 – bit 11 ADC data. To combine ADB register and the low-nibble of ADR will get full 12-bit ADC data buffer. The ADC data buffer is a read-only register and the initial status is unknown after system reset.

Table 16-1 The AIN input voltage vs. ADB output data

AIN n	ADB11	ADB10	ADB9	ADB8	ADB7	ADB6	ADB5	ADB4	ADB3	ADB2	ADB1	ADB0
0/4096*VREFH	0	0	0	0	0	0	0	0	0	0	0	0
1/4096*VREFH	0	0	0	0	0	0	0	0	0	0	0	1
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.	.	.	.	.
4094/4096*VREFH	1	1	1	1	1	1	1	1	1	1	1	0
4095/4096*VREFH	1	1	1	1	1	1	1	1	1	1	1	1

16.6 ADC Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
ADM	ADENB	ADS	EOC	ADCKSW	CHS3	CHS2	CHS1	CHS0
ADB	ADB11	ADB10	ADB9	ADB8	ADB7	ADB6	ADB5	ADB4
ADR	-	GCHS	ADCKS1	ADCKS0	ADB3	ADB2	ADB1	ADB0
VREFH	EVHENB	-	-	ADPWS	-	VHS2	VHS1	VHS0
P1CON	P1CON7	P1CON6	P1CON5	P1CON4	P1CON3	P1CON2	P1CON1	P1CON0
P2CON	P2CON7	P2CON6	P2CON5	P2CON4	P2CON3	P2CON2	P2CON1	P2CON0
P3CON	-	-	-	-	-	-	P3CON1	P3CON0
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
IEN2	-	-	-	-	EPWM1	EX2	-	EADC
IRCON2	-	-	-	-	PWM1F	IE2	-	ADCF

ADM Register (0xD2)

Bit	Field	Type	Initial	Description
7	ADENB	R/W	0	ADC control bit. In stop mode, disable ADC to reduce power consumption. 0: Disable 1: Enable
6	ADS	R/W	0	ADC conversion control Write 1: Start ADC conversion (automatically cleared by the end of conversion)
5	EOC	R/W	0	ADC status bit. 0: ADC progressing 1: End of conversion (automatically set by hardware; manually cleared by firmware)
4	ADCKSW	R/W	0	ADC clock source select bit 0: fosc 1: flosc
3..0	CHS[3:0]	R/W	0x00	ADC input channel select bit. 0000: AIN0, 0001: AIN1, 0010: AIN2, 0011: AIN3, 0100: AIN4, 0101: AIN5, 0110: AIN6, 0111: AIN7, 1000: AIN8, 1001: AIN9, 1010: AIN10, 1011:AIN11, 1100: AIN12, 1101: Reserved, Others: AIN14*(1).

*(1) The AIN14 is internal 2V or 3V or 4V input channel. There is no any input pin from outside. In this time ADC reference voltage must be internal VDD and External voltage, not internal 2V or 3V or 4V.

ADB Register (0xD3)

Bit	Field	Type	Initial	Description
7..0	ADB[11:4]	R	-	ADC Result Bit [11:4]* in 12-bit ADC resolution mode.

* ADC data buffer is 12-bit length to store ADC converter result. The high byte is ADB register, and the low-nibble is ADR[3:0] bits.

ADR Register (0xD4)

Bit	Field	Type	Initial	Description
7	Reserved	R	0	
6	GCHS	R/W	0	ADC global channel select bit. 0: Disable AIN channel. 1: Enable AIN channel.
5..4	ADCKS[1:0]	R/W	00	ADC's clock rate select bit. 00 = fosc/16, 01 = fosc/8, 10 = fosc/1, 11 = fosc/2 or 00 = flosc/16, 01 = flosc/8, 10 = flosc/1, 11 = flosc/2
3..0	ADB[3:0]	R	-	ADC Result Bit [3:0]* in 12-bit ADC resolution mode.

* ADC data buffer is 12-bit length to store ADC converter result. The high byte is ADB register, and the low-nibble is ADR[3:0] bits.

VREFH Register (0xD5)

Bit	Field	Type	Initial	Description
7	EVHENB	R/W	0	ADC internal reference high voltage control bit. 0: Enable ADC internal VREFH function. AVREFH/P2.0 pin is GPIO. 1: Disable ADC internal VREFH function. AVREFH/P2.0 pin is external AVREFH*(1) input pin.
4	ADPWS	R/W	0	PWM trigger ADC start control bit. 0: Disable PWM trigger ADC start. 1: Enable PWM trigger ADC start.
2..0	VHS[2:0]	R/W	00	ADC internal reference high voltage selects bits. *(2) 000: VREFH = 2.0V. 001: VREFH = 3.0V. 010: VREFH = 4.0V. 011: VREFH = VDD. 100: VREFH = VDD and AIN14 = 2.0V. 101: VREFH = VDD and AIN14 = 3.0V. 110: VREFH = VDD and AIN14 = 4.0V. Others: Reserved.
Else	Reserved	R/W	0	

*(1) The AVREFH level must be between the VDD and 2.0V.

*(2) If AIN14 channel is selected as internal 2V or 3V or 4V input channel. There is no any input pin from outside. In this time ADC reference high voltage must be internal VDD or External voltage, not internal 2V/3V/4V.

P1CON Register (0x9F)

Bit	Field	Type	Initial	Description
7..0	P1CON[7:0]	R/W	0x00	P1 configuration control bit*. 0: P1 can be analog input pin (ADC input pin) or digital GPIO pin. 1: P1 is pure analog input pin and can't be a digital GPIO pin.

* P1CON [7:0] will configure related Port1 pin as pure analog input pin to avoid current leakage.

P2CON Register (0XD6)

Bit	Field	Type	Initial	Description
7..0	P2CON[7:0]	R/W	0x0	P2 configuration control bit*. 0: P2 can be analog input pin (ADC input pin) or digital GPIO pin. 1: P2 is pure analog input pin and can't be a digital GPIO pin.

* P2CON [7:0] will configure related Port2 pin as pure analog input pin to avoid current leakage.

P3CON Register (0XD7)

Bit	Field	Type	Initial	Description
1..0	P3CON[1:0]	R/W	0x0	P3 configuration control bit*. 0: P3 can be analog input pin (ADC input pin) or digital GPIO pin. 1: P3 is pure analog input pin and can't be a digital GPIO pin.

* P3CON [7:0] will configure related Port3 pin as pure analog input pin to avoid current leakage.

IEN0 Register (0xA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Interrupts enable. Refer to Chapter Interrupt
Else				Refer to other chapter(s)

IEN2 Register (0x9A)

Bit	Field	Type	Initial	Description
0	EADC	R/W	0	ADC interrupt control bit. 0: Disable ADC interrupt function. 1: Enable ADC interrupt function.
Else				Refer to other chapter(s)

IRCON2 Register (0xBF)

Bit	Field	Type	Initial	Description
0	ADCF	R/W*	0	ADC interrupt request flag. 0 = None ADC interrupt request. 1 = ADC interrupt request.
Else				Refer to other chapter(s)

* This bit can't write '1' value. The IRCON2CLR macro is strongly recommended to clear request flag.

16.7 Sample Code

The following sample code demonstrates how to perform ADC to convert AIN5 with interrupt.

```
#define ADCAIN14_VDD   (3 << 0)  //AIN14 = VDD
#define ADCAIN14_4V    (2 << 0)  //AIN14 = 4.0V
#define ADCAIN14_3V    (1 << 0)  //AIN14 = 3.0V
#define ADCAIN14_2V    (0 << 0)  //AIN14 = 2.0V
#define ADCInRefVDD    (1 << 2)  //internal reference from VDD
#define ADCExHighRef   (1 << 7)  //high reference from AVREFH/P3.0
#define ADCClkFosc     (0 << 4)  //ADC clock source = fosc
#define ADCClkFlosc    (1 << 4)  //ADC clock source = flosc
#define ADCSpeedDiv16  (0 << 4)  //ADC clock = fosc/16 or flosc/16
#define ADCSpeedDiv8   (1 << 4)  //ADC clock = fosc/8 or flosc/8
#define ADCSpeedDiv1   (2 << 4)  //ADC clock = fosc/1 or flosc/1
#define ADCSpeedDiv2   (3 << 4)  //ADC clock = fosc/2 or flosc/2
#define ADCChannelEn   (1 << 6)  //enable ADC channel
#define SelAIN5        (5 << 0)  //select ADC channel 5
#define ADCStart       (1 << 6)  //start ADC conversion
#define ADCEn          (1 << 7)  //enable ADC
#define EADC           (1 << 0)  //enable ADC interrupt
#define ClearEOC       0xDF;

unsigned int  ADCBuffer;    // data buffer

void ADCInit(void)
{
  P1  = 0x00;
  P1M = 0x80;
  // set AIN5 pin's mode at pure analog pin
  P2CON |= 0x08;     //AIN5/P23
  P2M   &= 0XF7;    //input mode
  P2UR  &= 0Xf7;    //disable pull-high

  // configure ADC channel, ADC clock source and enable ADC.
  ADM = ADCEn | SelAIN5 | ADCClkFosc;
  // enable channel and select conversion speed
  ADR = ADCChannelEn | ADCSpeedDiv1;
  // configure reference voltage
  VREFH = ADCInRefVDD;

  // enable ADC interrupt
  IEN2  |= EADC;
  IEN0  |= 0x80;    //enable global interrupt

```

```
42    // start ADC conversion
43    ADM   |= ADCStart;
44  }
45
46  void ADCInterrupt(void) interrupt ISRAdc  //0x8B
47  {
48    //ADCF clear by hardware
49    P17 = ~P17;
50    ADCBuffer = (ADB << 4) + (ADR & 0x0F);
51    ADM &= ClearEOC;
52    ADM |= ADCStart;
53  }
```

17 UART

The UART provides a flexible full-duplex synchronous/asynchronous receiver/transmitter. The serial interface provides an up to 1MHz flexible full-duplex transmission. It can operate in four modes (one synchronous and three asynchronous). Mode0 is a shift register mode and operates as synchronous transmitter/receiver. In Mode1-Mode3 the UART operates as asynchronous transmitter/receiver with 8-bit or 9-bit data. The transfer format has start bit, 8-bit/ 9-bit data and stop bit. Transmission is started by writing to the S0BUF register. After reception, input data are available after completion of the reception in the S0BUF register. TB80/RB80 bit can be used as the 9th bit for transmission and reception in 9-bit UART mode. Programmable baud rate supports different speed peripheral devices.

The UART features include the following:

- Full-duplex, 2-wire synchronous/asynchronous data transfer.
- Programmable baud rate.
- 8-bit shift register: operates as synchronous transmitter/receiver
- 8-bit / 9-bit UART: operates as asynchronous transmitter/receiver with 8 or 9-bit data bits and programmable baud rate.

17.1 UART Operation

The UART UTX and URX pins are shared with GPIO. In synchronous mode, the UTX/URX shared pins must set output high by software. In asynchronous mode (8-bit/9-bit UART), the UTX shared pins must set output high and URX set input high by software. Thus, URX/UTX pins will transfers to UART purpose. When UART disables, the UART pins returns to GPIO last status.

The UTX/URX pins also support open-drain structure. The open-drain option is controlled by PnOC bit. When PnOC=0, disable UTX/URX open-drain structure. When PnOC=1, enable UTX/URX open-drain structure. If enable open-drain structure, UTX/URX pin must set high level (IO mode control will be ignored) and need external pull-up resistor.

The UART supports interrupt function. ES0 is UART transfer interrupt function control bit. ES0=0, disable both transmitter and receiver interrupt function. ES0=1, enable both UART transmitter and receiver interrupt function. When UART interrupt function enable, the program counter points to interrupt vector to do UART interrupt service routine after UART operating. TI0/RI0 is UART interrupt request flag, and also to be the UART operating status indicator when interrupt is disabled. TI0 and RI0 must clear by software.

UART provides four operating mode (one synchronous and three asynchronous) controlled by S0CON register. These modes can be support in different baud rate and communication protocols.

SM0	SM1	Mode	Synchronization	Clock Rate	Start Bit	Data Bits	Stop Bit	UART pins' mode and data
0	0	0	Synchronous	Fcpu/12	X	8	X	UTX pin: P05M=1 and P05=1 URX pin: Transmitter: P06M=1 and P06=1 Receiver: P06M=0 and P06=1
0	1	1	Asynchronous	Baud rate generator or T1 overflow rate	1	8	1	UTX pin: P05M=1 and P05=1 URX pin: P06M=0
1	0	2	Asynchronous	Fcpu/64 or Fcpu/32	1	9	1	
1	1	3	Asynchronous	Baud rate generator or T1 overflow rate	1	9	1	

17.2 Mode 0: Synchronous 8-bit Receiver/Transmitter

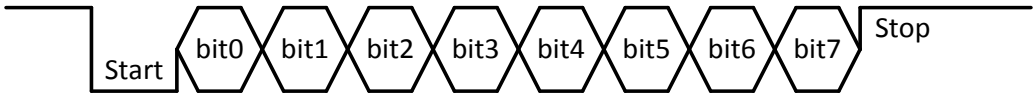
Mode0 is a shift register mode. It operates as synchronous transmitter/receiver. The UTX pin output shift clock for both transmit and receive condition. The URX pin is used to transmit and receive data. 8-bit data will be transmit and receive with LSB first. The baud rate is fcpu/12. Data transmission is started by writing data to S0BUF register. In the end of the 8th bit transmission, the TI0 flag is set. Data reception is controlled by REN0 bit. When REN0=1, data transmission starts and the RI0 flag is set at the end of the 8th bit reception.

17.3 Mode 1: 8-bit Receiver/Transmitter with Variable Baud Rate

Mode1 supports an asynchronous 8-bit UART with variable baud rate. The transfer format includes 1 start bit, 8 data bits (LSB first) and 1 stop bit. Data is transmitted by UTX pin and received by URX pin. The baud rate clock source can be baud rate generator or T1 overflow controlled by BD bit. When BD=0, the baud rate clock source is from T1 overflow. When BD=1, the baud rate clock source is from baud rate generator controlled by S0RELH and S0RELL. Additionally, the baud rate can be doubled by SMOD bit.

Data transmission is controlled by REN0 bit. After transmission configuration, load transmitted data into S0BUF, and then UART starts to transmit the pocket. The TI0 flag is set at the beginning of the stop bit.

Data reception is controlled by REN0 bit. When REN0=1, data reception function is enabled. Data reception starts by receiving the start bit for master terminal, URX detects the falling edge of start bit, and then the RI0 flag is set in the middle of a stop bit. Until reception completion, input data is stored in S0BUF register and the stop bit is stored in RB80.



17.4 Mode 2: 9-bit Receiver/Transmitter with Fixed Baud Rate

Mode2 supports an asynchronous 9-bit UART with fixed baud rate. The transfer format includes 1 start bit, 9 data bits (LSB first) and 1 stop bit. Data is transmitted by UTX pin and received by URX pin. The baud rate clock source is fixed to fcpu/64 or fcpu/32 and is controlled by SMOD bit. When SMOD=0, baud rate is fcpu/64. When SMOD=1, baud rate is fcpu/32.

Data transmission is controlled by REN0 bit. After transmission configuration, load transmitted data into S0BUF, and then UART starts to transmit the pocket. The 9th data bit is taken from TB80. The TI0 flag is set at the beginning of the stop bit.

Data reception is controlled by REN0 bit. When REN0=1, data reception function is enabled. Data reception starts by receiving the start bit for master terminal, URX detects the falling edge of start bit, and then the RI0 flag is set in the middle of a stop bit. Until reception completion, lower 8-bit input data is stored in S0BUF register and the 9th bit is stored in RB80.

17.5 Mode 3: 9-bit Receiver/Transmitter with Variable Baud Rate

Mode3 supports an asynchronous 9-bit UART with variable baud rate. The transfer format includes 1 start bit, 9 data bits (LSB first) and 1 stop bit. Data is transmitted by UTX pin and received by URX pin. The different between Mode2 and Mode3 is baud rate selection. In the Mode3, the baud rate clock source can be baud rate generator or T1 overflow controlled by BD bit. When BD=0, the baud rate clock source is from T1 overflow. When BD=1, the baud rate clock source is from baud rate generator controlled by S0RELH and S0RELL. Additionally, the baud rate can be doubled by SMOD bit.

Data transmission is controlled by REN0 bit. After transmission configuration, load transmitted data into S0BUF, and then UART starts to transmit the pocket. The 9th data bit is taken from TB80. The TI0 flag is set at the beginning of the stop bit.

Data reception is controlled by REN0 bit. When REN0=1, data reception function is enabled. Data reception starts by receiving the start bit for master terminal, URX detects the falling edge of start bit, and then the RI0 flag is set in the middle of a stop bit. Until reception completion, lower 8-bit input data is stored in S0BUF register and the 9th bit is stored in RB80.

17.6 Multiprocessor Communication

UART supports multiprocessor communication between a master device and one or more slaver device in Mode2 and Mode3 (9-bit UART). The master identifies correct slavers by using the 9th data bit. When the communication starts, the master transmits a specific address byte with the 9th bit is set “1” to selected slavers, and then transmits a data byte with the 9th bit is set “0” in the following transmission.

Multiprocessor communication is controlled by SM20 bit. When SM20=0, disable multiprocessor communication. When SM20=1, enable multiprocessor communication. If SM20 is set, the UART reception interrupt is only generated when the 9th received bit is “1” (RB80). The slavers will compare received data with its own address data by software. If address byte is match, the slavers clear SM20 bit to enable interrupt function in the following data transmission. The slavers with unmatched address, their SM20 keep in “1” and will not generate interrupt in the following data transmission.

17.7 Baud Rate Control

The UART mode 0 has a fixed baud rate at fcpu/12, and the mode 2 has two baud rate selection which is chosen by SMOD register: fcpu/64 (SMOD = 0) and fcpu/32 (SMOD = 1).

The baud rate of UART mode 1 and mode 3 is generated by either S0RELH/S0RELL registers (BD = 1) or Timer 1 overflow period (BD = 0). The SMOD bit doubles the frequency from the generator.

If the S0RELH/S0RELL is selected (BD = 1) in mode 1 and 3, the baud rate is generated as following equation.

$$\text{Baud Rate} = 2^{\text{SMOD}} \times \frac{\text{fcpu}}{64 \times (1024 - \text{S0REL})} bps$$

Table 17-1 Recommended Setting for Common UART Baud Rates (fcpu = 8 MHz)

Baud Rate	SMOD	S0RELH	S0RELL	Accuracy
4800	0	0x03	0xE6	0.16 %
9600	0	0x03	0xF3	0.16 %
19200	1	0x03	0xF3	0.16 %
38400	1	0x03	0xF9	-6.99 %
56000	1	0x03	0xFB	-10.71 %
57600	1	0x03	0xFC	8.51 %
115200	1	0x03	0xFE	8.51 %
128000	1	0x03	0xFE	-2.34 %
250000	1	0x03	0xFF	0 %

If the Timer 1 overflow period is selected (BD = 0) in mode 1 and 3, the baud rate is generated as following equation. The Timer 1 must be in 8-bit auto-reload mode which can generate periodically overflow signals.

$$\text{Baud Rate} = 2^{\text{SMOD}} \times \frac{\text{T1 clock rate}}{32 \times (256 - \text{TH1})} bps$$

Table 17-2 Recommended Setting T1 overflow period (T1 clock=32M) for Common UART Baud Rates (fcpu = 8 MHz)

Baud Rate	SMOD	Timer Period	TH1/TL1	Accuracy
4800	0	6.510 us	0x30	0.16 %
9600	1	6.510 us	0x30	0.16 %
19200	1	3.255 us	0x98	0.16 %
38400	1	1.628 us	0xCC	0.16 %
56000	1	1.116 us	0xDC	-0.80 %
57600	1	1.085 us	0xDD	-0.80 %
115200	1	0.543 us	0xEF	2.08 %
128000	1	0.488 us	0xF0	-2.40 %

✷ ***Note:***

1. When baud rate generator source is T1 overflow rate, the max counter value is 0xFB. (Only supports 0x00~0xFB).

2. When baud rate generator source is T1 overflow rate, the system clock fcpu must be greater or equal to T1 clock rate.

17.8 Power Saving

The UART module has clock gating function for saving power. When REN0 bit is 0, the UART module internal clocks are halted to reduce power consumption. UART relevant register (S0CON, S0CON2, S0BUF, S0RELL, S0RELH and SMOD bit) are unable to access.

Conversely, when REN0 bit is 1, UART internal clocks are run, and registers can access. The REN0 bit must be set to 1, before the initial setting UART.

17.9 UART Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
S0CON	SM0	SM1	SM20	REN0	TB80	RB80	TI0	RI0
S0CON2	BD	-	-	-	-	-	-	-
S0BUF	S0BUF7	S0BUF6	S0BUF5	S0BUF4	S0BUF3	S0BUF2	S0BUF1	S0BUF0
PCON	SMOD	-	-	STWK	P2SEL	GF0	STOP	IDLE
S0RELH	-	-	-	-	-	-	S0REL9	S0REL8
S0RELL	S0REL7	S0REL6	S0REL5	S0REL4	S0REL3	S0REL2	S0REL1	R0REL0
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
P2OC	P27OC	P26OC	-	P24OC	P23OC	P22OC	-	-
P2M	P27M	P26M	P25M	P24M	P23M	P22M	P21M	P20M
P2	P27	P26	P25	P24	P23	P22	P21	P20

S0CON Register (0x98)

Bit	Field	Type	Initial	Description
7..6	SM[0:1]	R/W	00	UART mode selection 00: Mode 0 01: Mode 1 10: Mode 2 11: Mode 3
5	SM20	R/W	0	Multiprocessor communication (mode 2, 3) 0: Disable 1: Enable
4	REN0	R/W	0	UART module (and reception function) 0: Disable for power saving* 1: Enable for UART operating
3	TB0	R/W	0	The 9th bit transmission data (mode 2, 3)
2	RB0	R/W	0	The 9th bit data from reception
1	TI0	R/W	0	UART interrupt flag of transmission
0	RI0	R/W	0	UART interrupt flag of reception

* When REN0 bit is 0, UART relevant register are unable to access, and the module internal clocks are halted.

S0CON2 Register (0xD8)

Bit	Field	Type	Initial	Description
7	BD	R/W	0	Baud rate generators selection (mode 1, 3) 0: Timer 1 overflow period 1: Controlled by S0RELH, S0RELL registers
6..0	Reserved	R	0x00	

S0BUF Register (0x99)

Bit	Field	Type	Initial	Description
7..0	S0BUF	R/W	0x00	Action of writing data triggers UART communication (LSB first). Reception data is available to read by the end of packages.

PCON Register (0x87)

Bit	Field	Type	Initial	Description
7	SMOD	R/W	0	UART baud rate control. In UART mode 0: Unused. In UART mode 1, 3: The baud rate is generated as the equation in section 17.7 (Baud Rate Control). In UART mode 2: 0: fcpu/64 1: fcpu/32
6..0				Refer to other chapter(s)

S0RELH/S0RELL Registers (S0RELH: 0xBA, S0RELL: 0xAA)

Bit	Field	Type	Initial	Description
15..10	Reserved	R	0x00	
9..0	S0REL[9:0]	R/W	0x00	S0RELH[1:0] & S0RELL[7:0]. UART Reload Register is used for UART baud rate generation.

IEN0 Register (0xA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Interrupts enable. Refer to Chapter Interrupt
4	ES0	R/W	0	Enable UART interrupt
Else				Refer to other chapter(s)

P2OC Register (0xE4)

Bit	Field	Type	Initial	Description
7	P27OC	R/W	0	0: Switch P2.7 (URX) to input mode (required) ~~1: Switch P2.7 (URX) to open-drain mode~~*
6	P26OC	R/W	0	0: Switch P2.6 (UTX) to push-pull mode 1: Switch P2.6 (UTX) to open-drain mode
Else				Refer to other chapter(s)

* Setting P27OC as high causes URX cannot receive data.

P2M Register (0xFB)

Bit	Field	Type	Initial	Description
1	P27M	R/W	0	0: Set P2.7 (URX) as input mode (required) ~~1: Set P2.7 (URX) as output mode~~*
0	P26M	R/W	0	~~0: Set P2.6 (UTX) as input mode~~* 1: Set P2.6 (UTX) as output mode (required)
Else				Refer to other chapter(s)

* The URX and UTX respectively require input and output mode selection to receive/transmit data appropriately.

P2 Register (0xA0)

Bit	Field	Type	Initial	Description
1	P27	R/W	0	This bit is available to read at any time for monitoring the bus statue.
0	P26	R/W	0	~~0: Set P2.6 (UTX) always low~~* 1: Make P2.6 (UTX) can output UART data (required)
Else				Refer to other chapter(s)

* Setting P26 initially high because UART block drive the shared pin low signal only.

17.10 Sample Code

The following sample code demonstrates how to perform UART mode 1 with interrupt.

```
#define SYSUartSM0    (0 << 6)
#define SYSUartSM1    (1 << 6)
#define SYSUartSM2    (2 << 6)
#define SYSUartSM3    (3 << 6)
#define SYSUartREN    (1 << 4)
#define SYSUartSMOD   (1 << 7)
#define SYSUartES0    (1 << 4)

void SYSUartInit(void)
{
  // set UTX, URX pins' mode at here or at GPIO initialization
  P26 = 1;
  P2M = P2M | 0x40 & ~0x80;
  // configure UART mode between SM0 and SM3, enable URX
  S0CON = SYSUartSM1 | SYSUartREN;
  // configure UART baud rate
  PCON = SYSUartSMODE1;
  S0CON2 = SYSUartBD1;
  S0RELH = 0x03;
  S0RELL = 0xFE;

  // enable UART interrupt
  IEN0 |= SYSUartES0;
  // send first UTX data
  S0BUF = uartTxBuf;
}

void SYSUartInterrupt(void) interrupt ISRUart  //0x23
{
  if (TI0 == 1) {
    S0BUF = uartTxBuf;
    TI0 = 0;
  } else if (RI0 == 1) {
    uartRxBuf = S0BUF;
    RI0 = 0;
  }
}
```

18 SPI

The SPI a serial communicate interface for data exchanging from one MCU to one MCU or other hardware peripherals. It is a simple 8-bit interface without a major definition of protocol, packet or control bits. The SPI transceiver includes three pins, clock (SCK), data input and data output (MISO/ MOSI) to send data between master and slaver terminals. An optional slave select pin (SSN) can be enabled by register in slave mode. The SPI interface builds in 4-mode which are the clock idle status and the clock phases.

- Full-duplex, 3-wire synchronous data transfer.
- Master (SCK is clock output) or Slave (SCK is clock input) operation.
- Seven SPI Master baud rates.
- Slave Clock rate up to fcpu/8.
- 8-bit data transmitted MSB first, LSB last.
- Serial clock with programmable polarity and phase.
- Master Mode fault error flag with MCU interrupt capability.
- Write collision flag protection.

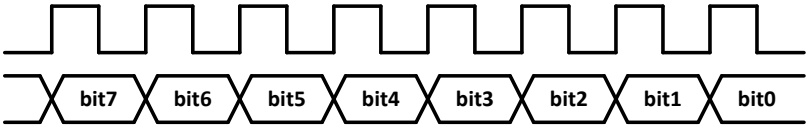
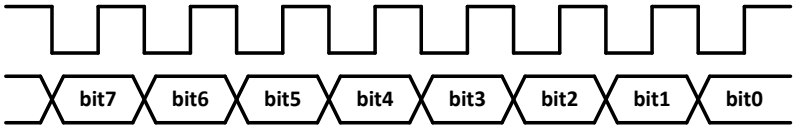
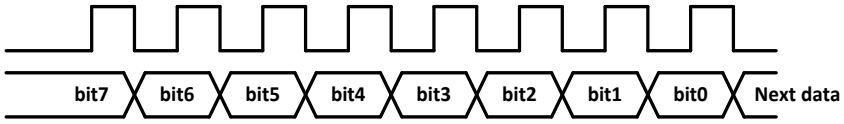
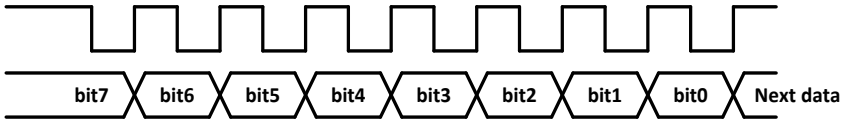
18.1 SPI Operation

The SPCON register can control SPI operating function, such as: transmit/receive, clock rate, data transfer direction, SPI clock idle status and clock control phase and enable this circuit. This SPI circuit will transmit or receive 8-bit data automatically by setting SPEN in SPCON register and write or read SPDAT register.

CPOL bit is designed to control SPI clock idle status. CPHA bit is designed to control the clock edge direction of data receive. CPOL and CPHA bits decide the SPI format. The SPI data transfer direction is MSB bit to LSB bit.

The SPI supports 4-mode format controlled by CPOL and CPHA bits. The edge direction is “Data Transfer Edge”. When setting rising edge that means to receive and transmit one bit data at SCK rising edge, and data transition is at SCK falling edge. When setting falling edge, that means to receive and transmit one bit data at SCK falling edge, and data transition is at SCK rising edge.

“CPHA” is the clock phase bit controls the phase of the clock on which data is sampled. When CPHA=1, the SCK first edge is for data transition, and receive and transmit data is at SCK 2nd edge. When CPHA=0, the 1st bit is fixed already, and the SCK first edge is to receive and transmit data. The SPI data transfer timing as following figure:

CPOL	CPHA	Diagrams	Description
0	1		SCK idle status = Low. The transfer first bit = MSB. SCK data transfer edge = Falling edge.
1	1		SCK idle status = High. The transfer first bit = MSB. SCK data transfer edge = Rising edge.
0	0		SCK idle status = Low. The transfer first bit = MSB. SCK data transfer edge = Rising edge.
1	0		SCK idle status = High. The transfer first bit = MSB. SCK data transfer edge = Falling edge.

The SPI supports interrupt function. ESPI is SPI interrupt function control bit. ESPI=0, disable SPI interrupt function. ESPI=1, enable SPI interrupt function. When SPI interrupt function enable, the program counter points to interrupt vector to do SPI interrupt service routine after SPI operating. SPIF is SPI interrupt request flag, and also to be the SPI operating status indicator when ESPI= 0, but cleared by reading the SPSTA, SPDAT registers.

SPI builds in chip selection function to implement SPI multi-device mode. One master communicating with several slave devices in SPI bus, and the chip selection decides the pointed device. The chip selection pin is SSN pin.

The SPI pins also support open-drain structure. The open-drain option is controlled by PnOC bits. When PnOC=0, disable SPI open-drain structure. When PnOC=1, enable SPI open-drain structure. If enable open-drain structure, SPI pins must be set input mode and need external pull-up resistor.

18.2 SPI Master

The SPI master mode has seven types of clock generator from fcpu/2 to fcpu/128. Generated clock is outputted through SCK pin (shared with P2.4) and its idle status is controlled by CPOL.

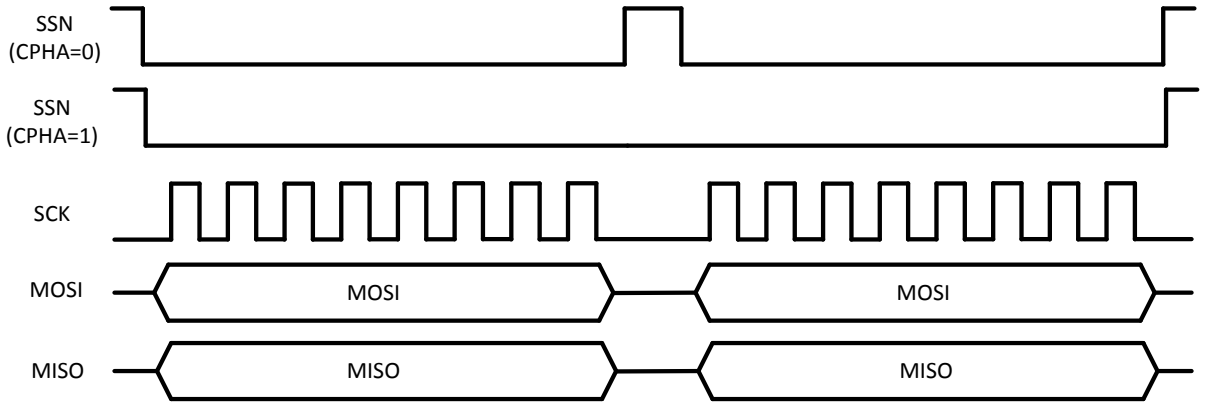
The phase of data input and output is automatically specified by CPHA register. In master mode MOSI pin (shared with P2.2) plays the role of data output, and MISO pin (shared with P2.3) fetches data from slave device. A SPI communication is started by writing SPDAT register; the received data from MISO is available to read after the end of data transmission.

The master mode has two status flags with interrupt function:
SPIF register indicates the end of one byte data communication. An interrupt would be issued at the same time if ESPI bit is enabled.
MODF is issued by SSN (shared with P2.5) low status while transmission. This interrupt source can be masked by setting SSDIS bit.

18.3 SPI Slave

The SPI slave mode monitors SCK pin to control its MISO and MOSI communication. However, the maximum clock rate is limited at fcpu/8. Slave device(s) are expected to specify its CPOL and CPHA setting as the same configuration of the connected SPI bus.

The slave mode treats MOSI pin as its data input, and MISO pin as its data transmission. By default, the SSDIS register is low which means the slave select pin (SSN) is functional. A SPI communication would be processed if the SSN is low status. Thus, a slave device is suspended if its SSN is high status. But in CPHA = 0, Strictly SSN must follow each 8-bit data needs to be included with falling edge and rising edge, CPHA=1 is not limitation.



The slave mode has two status flags with interrupt function:
SPIF indicates the end of one byte data communication. The original SPDAT's value has been transmitted, and the received data from MOSI is ready to be read on SPDAT.
MODF indicates that the slave select pin (SSN) has turned high before a completion of one byte communication. In other word, the last time of SPI communication is broken.

18.4 Power Saving

The SPI module has clock gating function for saving power. When SPEN bit is 0, the SPI module internal clocks are halted to reduce power consumption. SPI relevant register (SPCON, SPSTA and SPDAT) are unable to access. Conversely, when SPEN bit is 1, SPI internal clocks are run, and registers can access. The SPEN bit must be set to 1, before the initial setting SPI.

18.5 SPI Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SPCON	SPR2	SPEN	SSDIS	MATR	CPOL	CPHA	SPR1	SPR0
SPSTA	SPIF	WCOL	SSERR	MODF	-	-	-	-
SPDAT	SPDAT7	SPDAT6	SPDAT5	SPDAT4	SPDAT3	SPDAT2	SPDAT1	SPDAT0
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
IEN1		-	ET2C3	ET2C2	ET2C1	ET2C0	ESPI	EI2C
P2OC	P27OC	P26OC	-	P24OC	P23OC	P22OC	-	-
P2M	P27M	P26M	P25M	P24M	P23M	P22M	P21M	P20M

SPCON Register (0xE2)

Bit	Field	Type	Initial	Description
7,1,0	SM[2:0]	R/W	000	SPI baud rate generator (master mode only) 000: fcpu/2 001: fcpu/4 010: fcpu/8 011: fcpu/16 100: fcpu/32 101: fcpu/64 110: fcpu/128 ~~111: reserved~~
6	SPEN	R/W	0	SPI communication function 0: Disable for power saving* 1: Enable for SPI operating
5	SSDIS	R/W	0	Slave select pin function (MSTR = 0, CPHA = 0 only) 0: Enable slave selection pin (SSN) function 1: Disable slave select pin (SSN) function
4	MSTR	R/W	1	SPI mode 0: Slave mode 1: Master mode
3	CPOL	R/W	0	SCK pin idle status 0: SCK idle low 1: SCK idle high
2	CPHA	R/W	1	Clock phase of data latch control 0: Data latched by the first of clock edge 1: Data latched by the second of clock edge

* When SPEN bit is 0, SPI relevant register are unable to access, and the module internal clocks are halted.

SPSTA Register (0xE1)

Bit	Field	Type	Initial	Description
7	SPIF	R	0	SPI complete communication flag Set automatically at the end of communication Cleared automatically by reading SPSTA, SPDAT registers
6	WCOL	R	0	Write collision flag Set automatically if write SPDAT during communication Cleared automatically by reading SPSTA, SPDAT registers
5	SSERR	R	0	Synchronous slave select pin error Set automatically if SSN error controlling Cleared automatically by clear SPEN
4	MODF	R	0	Mode fault flag
3..0	Reserved	R	0x00	

SPDAT Register (0xE3)

Bit	Field	Type	Initial	Description
7..0	SPDAT	R/W	0x00	Master mode: action of writing data triggers SPI communication; reception data is readable after the end of one byte communication (SPIF automatically set). Slave mode: written data would be transmitted by SCK input; reception data is available to read after the end of one byte communication (SPIF automatically set).

IEN0 Register (0xA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Interrupts enable. Refer to Chapter Interrupt
Else				Refer to other chapter(s)

IEN1 Register (0xB8)

Bit	Field	Type	Initial	Description
1	ESPI	R/W	0	Enable SPI interrupt
Else				Refer to other chapter(s)

P2OC Register (0xE4)

Bit	Field	Type	Initial	Description
4	P24OC	R/W	0	0: Switch P2.4 (SCK) to input or output mode 1: Switch P2.4 (SCK) to open-drain mod
3	P23OC	R/W	0	0: Switch P2.3 (MISO) to input or output mode 1: Switch P2.3 (MISO) to open-drain mode
2	P22OC	R/W	0	0: Switch P2.2 (MOSI) to input or output mode 1: Switch P2.2 (MOSI) to open-drain mod
Else				Refer to other chapter(s)

P2M Register(0xFB)

Bit	Field	Type	Initial	Description
5	P25M	R/W	0	0: Set P2.5 (SSN) as input mode* 1: Set P2.5 (SSN) as output mode*
4	P24M	R/W	0	0: Set P2.4 (SCK) as input mode slave mode 1: Set P2.4 (SCK) as output mode master mode
3	P23M	R/W	0	0: Set P2.3 (MISO) as input mode master mode 1: Set P2.3 (MISO) as output mode slave mode
2	P22M	R/W	0	0: Set P2.2 (MOSI) as input mode slave mode 1: Set P2.2 (MOSI) as output mode master mode
Else				Refer to other chapter(s)

[1]Setting SCK as input mode is essential in slave mode; setting as output mode is recommended in master mode.

[2]Setting MISO as input mode is essential in master mode; setting as output mode is recommended in slave mode.

[3]Setting MOSI as input mode is essential is slave mode; setting as output mode is recommended in master mode.

*If slave mode with SSN function: essentially to set SSN as input mode.

18.6 Sample Code

The following sample code demonstrates how to perform SPI Master with interrupt.

```
#define SpiMaster      (1 << 4)  //SPI = Master mode
#define SpiSlave       (1 << 4)  //SPI = Slave mode
#define SpiMode0       (0 << 2)  //SCK idle low, data latch at rising edge
#define SpiMode1       (1 << 2)  //SCK idle low, data latch at falling edge
#define SpiMode2       (2 << 2)  //SCK idle high, data latch at falling edge
#define SpiMode3       (3 << 2)  //SCK idle high, data latch at rising edge
#define SpiEn          (1 << 6)  //Enable SPI
#define SpiSSNEn       (0 << 5)  //SSN pin function enable
#define SpiSSNDis      (1 << 5)  //SSN pin function disable

unsigned char  u8SpiData = 0;    // data buffer
unsigned char  u8TxCompleted = 0;

void SpiMaster(void)
{
  unsigned char u8RcvData = 0;

  //SCK & MOSI = output, MISO = input
  P2M |= 0x14;
  //Enable Spi, Master mode, SSN pin disable, Fclk/128
  //SCK idle low, data latch at falling edge
  SPCON = SpiEn | SpiMaster | SpiMode1 | SpiSSNDis | 0x82;
  //Enable Global/SPI interrupt
  IEN1  |= 0x02;
  IEN0  |= 0x80;    //enable global interrupt

  while (1) {
    SPDAT = 0x55;
    while(!u8TxCompleted);      // wait end of transmition
    u8TxCompleted = 0;         // clear sw flag
    u8RcvData = u8SpiData;      // receive 0x66

    SPDAT = 0x99;
    while(!u8TxCompleted);      // wait end of transmition
    u8TxCompleted = 0;         // clear sw flag
    u8RcvData = u8SpiData;      // receive 0xAA
  }
}

void SpiInterrupt(void) interrupt ISRSpi //0x4B
{
  switch ( SPSTA )              // Clear SPI flag (SPIF) by reading
  {
    case 0x80:
      u8SpiData = SPDAT;
      u8TxCompleted = 1;
      break;
    case 0x10:
      // Mode Fault
      break;
  }
}
```

19 I2C

The I2C is a serial communication interface for data exchanging from one MCU to one MCU or other hardware peripherals. The device can transmit data as a master or a slave with two bi-directional IO, SDA (Serial data output) and SCL (Serial clock input).

When a master transmit data to a slave, it's called "WRITE" operation; when a slave transmit data to a master, it's called "READ" operation. It also supports multi-master communication and keeps data transmission correctly by an arbitration method to decide one master has the control on bus and transmit its data.

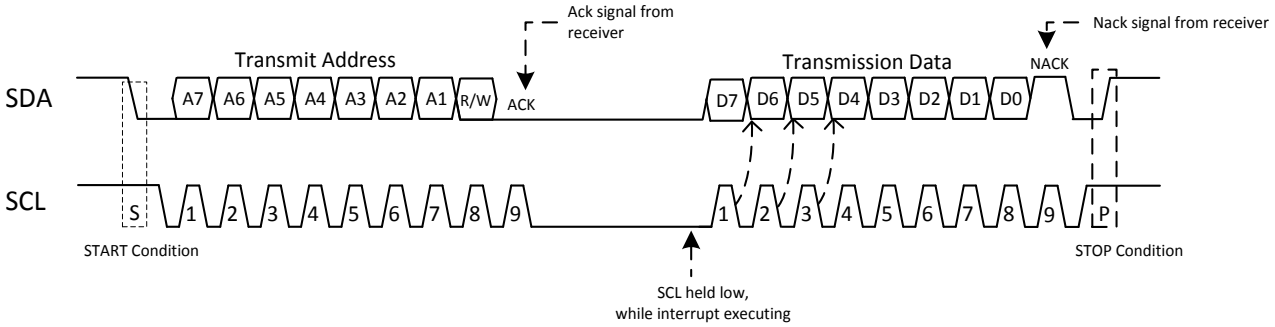
- Master Tx, Rx Mode
- Slave Tx, Rx mode (with general address call) for multiplex slave in single master situation.
- 2-wire synchronous data transfer/receiver.
- Support 100K/400K clock rate.

19.1 I2C Protocol

I2C transmission structure includes a START(S) condition, 8-bit address byte, one or more data byte and a STOP (P) condition. START condition is generated by master to initial any transmission.

Data is transmitted with the Most Significant Bit (MSB) first. In address byte, the higher 7-bit is address bit and the lowest bit is data direction (R/W) bit. When R/W=0, it assigns a "WRITR" operation. When R/W=1, it assigns a "READ" operation.

After each byte is received, the receiver (a master or a slave) must send an acknowledge (ACK). If transmitter can't receive an ACK, it will recognize a not acknowledge (NACK). In WRITE operation, the master will transmit data to the slave and then waits for ACK from slave. In READ operation, the slave will transmit data to the master and then waits for ACK from master. In the end, the master will generate a STOP condition to finish transmission.

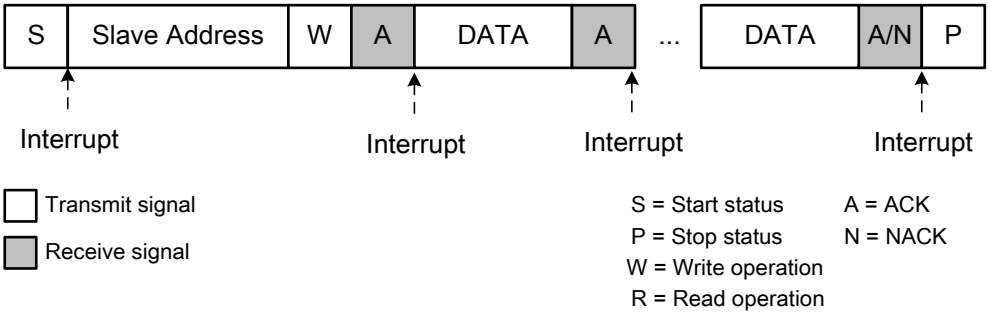


19.2 I2C Transfer Modes

The I2C can operate as a master/slave to execute the 8-bit serial data transmission/reception operation. Thus, the module can operate in one of four modes: Master Transmitter, Master Receiver, Slave Transmitter and Slave Receiver.

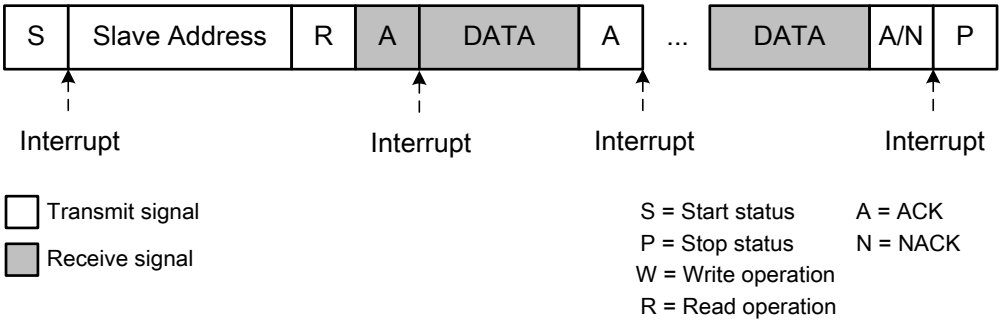
19.2.1 Master Transmitter Mode

The master transmits information to the slave. The serial data is output via SDA while the serial clock is output on SCL. Data transmission starts via generate a START(S) signal. After the START signal, the specific address byte of slave device is sent. The address byte includes 7-bit address bit and an 8th data direction (R/W) bit. The R/W is set “0” to enable the master transmission. In the following, the master transmits one or more data byte to the slaver. After each data is transmitted, the master waits for the acknowledge (ACK) from the slave. In the end, the master generates a STOP (P) signal to terminate the data transmission.



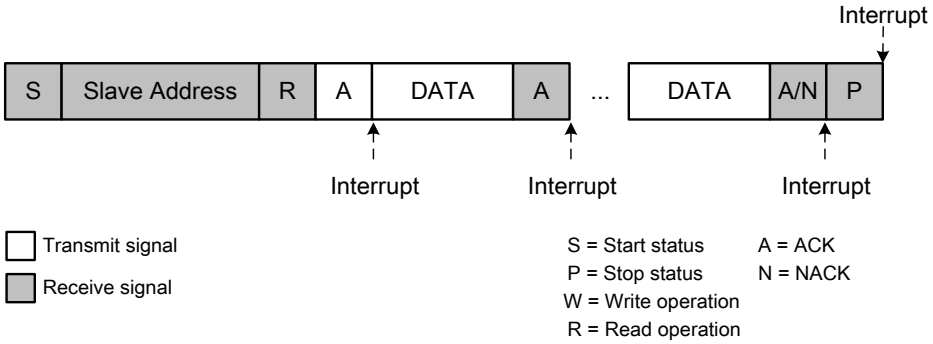
19.2.2 Master Receiver Mode

The master receives the information from the slave. The serial data input via SDA while the serial clock output on SCL. Data reception starts via generate a START(S) signal. After the START signal, the specific address byte of slave device is sent. The address byte includes 7-bit address bit and an 8th data direction (R/W) bit. The R/W is set “1” to enable the master reception. In the following, the master receives one or more data byte from the slaver. After each data is received, the master generates the acknowledge (ACK) or not acknowledge (NACK) to the slave via the status of AA bit. In the end, the master generates a STOP (P) signal to terminate the data transmission.



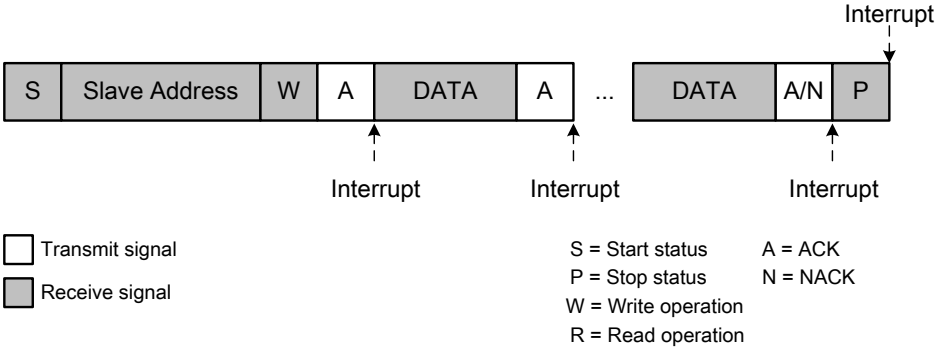
19.2.3 Slave Transmitter Mode

The slave transmits information to the master. The serial data output via SDA while the serial clock input on SCL. Data transmission starts via receive a START(S) signal from the master. After the START signal, the specific address byte of slave device is received. The address byte includes 7-bit address bit and an 8th data direction (R/W) bit. The R/W is set “1” to enable the slave transmission. If the received address byte match the address in I2CADR register, the slave generate an acknowledge (ACK). Otherwise, if general call address condition is set (GC=1), the slave also generate an acknowledge (ACK) after general call address (0x00) is received. In the following, the slave transmits one or more data byte to the master. After each data is transmitted, the slave waits for the acknowledge (ACK) from the master. In the end, the slave receives a STOP (P) signal from the master to terminate the data transmission.



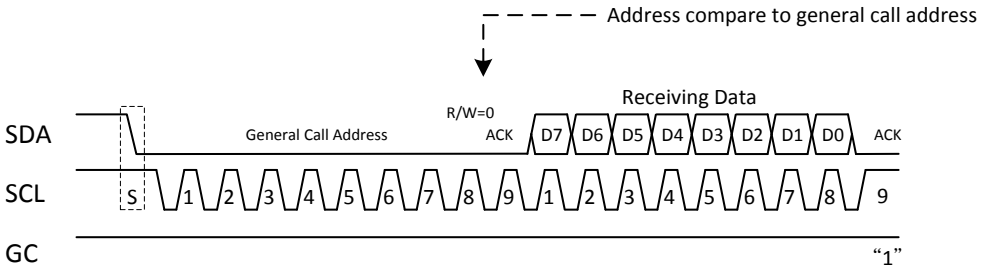
19.2.4 Slave Receiver Mode

The slave receives information from the master. Both the serial data and the serial clock are input on SDA and SCL. Data reception starts via receive a START(S) signal from the master. After the START signal, the specific address byte of slave device is received. The address byte includes 7-bit address bit and an 8th data direction (R/W) bit. The R/W is set “0” to enable the slave reception. If the received address byte match the address in I2CADR register, the slave generate an acknowledge (ACK). Otherwise, if general call address condition is set (GC=1), the slave also generate an acknowledge (ACK) after general call address (0x00) is received. In the following, the slave receives one or more data byte from the master. After each data is receives, the slave generates the acknowledge (ACK) or not acknowledge (NACK) to the master via the status of AA bit. In the end, the slave receives a STOP (P) signal from the master to terminate the data transmission.



19.3 General Call Address

In I2C bus, the first 7-bit is the slave address. Only the address matches slave address, the slave will response an ACK. The exception is the general call address which can address all slave devices. When this address occur, all devices should response an acknowledge (ACK). The general call address is a special address which is reserved as all “0” of 7-bit address. The general call address function is control by GC bit. Set this bit will enable general call address and clear it will disable. When GC=1, the general call address will be recognized. When GC=0, the general call address will be ignored.



19.4 Serial Clock Generator

In master mode, the SCL clock rate generator’s is controlled by CR[2:0] bit of I2CCON register.

When CR[2:0]=000~110, SCL clock rate is from internal clock generator.

$$\text{SCL Clock Rate} = \frac{\text{Fcpu}}{\text{Prescaler}} (\text{Prescaler} = 256\sim60)$$

When CR[2:0]=111, SCL clock rate is from Timer 1 overflow rate .

$$\text{SCL Clock Rate} = \frac{\text{Timer 1 Overflow}}{8}$$

The table below shows the clock rate under different setting.

CR2	CR1	CR0	I2C	Bit Frequency (kHz)	
			Prescaler	6MHz	8MHz
0	0	0	256	23	31
0	0	1	224	27	36
0	1	0	192	31	42
0	1	1	160	37	50
1	0	0	960	6.25	8
1	0	1	120	50	67
1	1	0	60	100	133
1	1	1	(Timer 1 overflow rate)/8		

✷ *Note:*

1. The first step of I2C operation is to setup the I2C pins' mode. Must be set "input mode" in SDA/SCL pins.

2. When clock generator source is T1 overflow rate, the max counter value is 0xFB. (Only supports 0x00~0xFB). And in this time if T1 clock rate is IHRC_32MHz, SCL maximum clock rate is 800kHz.

3. If user wants to generate SCL clock rate is 100kHz/400kHz, you can set T1 counter value is 0xD8/0xF6 easily.

19.5 Synchronization and Arbitration

In multi-master condition, more than one master may transmit on bus in the same time. It must be decided which master has the control of bus and complete its transmission. Clock synchronization and arbitration are used to configure multi-master transmission. Clock synchronization is executed by synchronizing the SCL signal with anther devices.

When two masters want to transmit data in the same, the clock synchronization will start by the High to Low transition on the SCL. If master 1 clock set LOW first, it holds the SCL in LOW status until the clock transit to HIGH status. However, if anther master clock still keep LOW status, the Low to High transition of master 1 may not change SCL status (SCL keep LOW). In the other word, SCL keep LOW by the master with the longest clock time in LOW status. The SCL will transit from LOW to HIGH when the all devices clock transit to HIGH status. In the duration, the master1 will keep in HIGH status and wait for SCL transition (from LOW to HIGH), then continue its transmission. After clock synchronization, all devices clock and SCL clock are the same. Arbitration is used to decide which master can complete its transmission by SDA signal. Two masters may send out a START condition and transmit data on bus in the same time. They may influence by each other. Arbitration will force one master to lose the control on bus. Data transmission will keep until master output different data signal. If one master transmits HIGH status and anther master transmits LOW status, the SDA will be pull low. The master output High will detect the different with SDA and lose the control on bus. The mater with LOW status wins the bus control and continues its transmission. There is no data miss during arbitration.

19.6 System Management Bus Extension

The optional System Management Bus (SMBus) protocol hardware supports 3 types timeout detection: (1) Tmext Timeout Detection: The cumulative stretch clock cycles within one byte. (2)Tsext Timeout Detection: The cumulative stretch clock cycles between start and stop condition. (3)Timeout Detection: The clock low measurement.

Timeout detection is controlled by SMBSEL and SMBDST registers. The SMBEXE bit of SMBSEL is SMBus extension function enable bit. When SMBEXE=1, SMBus extension function is enabled. Otherwise, Disable SMBus extension function. Timeout type and period setting is controlled by SMBTOP[2:0] and SMBDST. The period of SMBus timeout is controlled by three 16-bit buffers of Tmex, Tsext and Tout. The equation is as following.

$$\text{Tmext/Tsext/Tout} = \frac{\text{Timeout Period(sec)xFcpu(Hz)}}{1024}$$

Tmext is support by two 8-bit register of Tmext_L and Tmext_H . Tmext_L hold the low byte and Tmext_H hold high byte. Tsext is support by two 8-bit register of Tsext_L and Tsext_H . Tsext_L hold the low byte and Tsext_H hold high byte. Tout is support by two 8-bit register of Tout_L and Tout_H . Tout_L hold the low byte and Tout_H hold high byte.

Type	Time out period	Fcpu=8MHz	
		DEC	HEX
Tmext	5ms	39	27
Tsext	25ms	195	C3
Tout	35ms	273	111

By the setting of SMBTOP[2:0] to choose register type (as the table below), and write to register by write data to SMBDST register.

SMBTOP[2:0]	SMBDST	Description
000	Tmext_L	Select the low byte of Tmext register.
001	Tmext_H	Select the high byte of Tmext register.
010	Tsext_L	Select the low byte of Tsext register.
011	Tsext_H	Select the high byte of Tsext register.
100	Tout_L	Select the low byte of Tout register.
101	Tout_H	Select the high byte of Tout register.

When the SMBus extension function is enabled the lower 3-bit of I2CSTA hold the information about time out as the table below.

I2CSTA	Description
XXXX X000	No timeout errors.
XXXX XXX1	Tout timeout error.
XXXX XX1X	Tsext timeout error.
XXXX X1XX	Tmext timeout error.

19.7 Power Saving

The I2C module has clock gating function for saving power. When ENS1 bit is 0, the I2C module internal clocks are halted to reduce power consumption. I2C relevant register (I2CDAT, I2CADR, I2CCON, I2CSTA, SMBSEL and SMBDST) are unable to access. Conversely, when ENS1 bit is 1, I2C internal clocks are run, and registers can access. The ENS1 bit must be set to 1, before the initial setting I2C.

19.8 I2C Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2CDAT	I2CDAT7	I2CDAT6	I2CDAT5	I2CDAT4	I2CDAT3	I2CDAT2	I2CDAT1	I2CDAT0
I2CADR	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0	GC
I2CCON	CR2	ENS1	STA	STO	SI	AA	CR1	CR0
I2CSTA	I2CSTA7	I2CSTA6	I2CSTA5	I2CSTA4	I2CSTA3	I2CSTA2	I2CSTA1	I2CSTA0
SMBSEL	SMBEXE	-	-	-	-	SMBSTP2	SMBSTP1	SMBSTP0
SMBDST	SMBD7	SMBD6	SMBD5	SMBD4	SMBD3	SMBD2	SMBD1	SMBD0
IEN0	EAL	-	ET2	ES0	ET1	EX1	ET0	EX0
IEN1		-	ET2C3	ET2C2	ET2C1	ET2C0	ESPI	EI2C
P0M	P07M	P06M	P05M	P04M	P03M	P02M	P01M	P00M

I2CDAT Register (0xDA)

Bit	Field	Type	Initial	Description
7:0	I2CDAT[7:0]	R/W	0x00	The I2CDAT register contains a byte to be transmitted through I2C bus or a byte which has just been received through I2C bus. The CPU can read from and write to this 8-bit, directly addressable SFR while it is not in the process of byte shifting. The I2CDAT register is not shadowed or double buffered so the user should only read I2CDAT when an I2C interrupt occurs.

I2CADR Register (0xDB)

Bit	Field	Type	Initial	Description
7:1	I2CADR[6:0]	R/W	0x00	I2C slave address
0	GC	R/W	0	General call address (0X00) acknowledgment 0: ignored 1: recognized

I2CCON Register (0xDC)

Bit	Field	Type	Initial	Description
7,1,0	CR[2:0]	R/W	0	I2C clock rate 000: fcpu/256 001: fcpu/224 010: fcpu/192 011: fcpu/160 100: fcpu/960 101: fcpu/120 110: fcpu/60 111: Timer 1 overflow-period/8
6	ENS1	R/W	0	I2C functionality 0: Disable for power saving* 1: Enable for I2C operating
5	STA	R/W	0	START flag 0: No START condition is transmitted. 1: A START condition is transmitted if the bus is free.
4	STO	R/W	0	STOP flag 0: No STOP condition is transmitted. 1: A STOP condition is transmitted to the I2C bus in master mode.
3	SI	R/W	0	Serial interrupt flag The SI is set by hardware when one of 25 out of 26 possible I2C states is entered. The only state that does not set the SI is state F8h, which indicates that no relevant state information is available. The SI flag must be cleared by software. In order to clear the SI bit, '0' must be written to this bit. Writing a '1' to SI bit does not change value of the SI.
2	AA	R/W	0	Assert acknowledge flag 0: A NACK will be returned when a byte has received 1: An ACK will be returned when a byte has received

* When ENS1 bit is 0, I2C relevant register are unable to access, and the module internal clocks are halted.

I2CSTA Register (0xDD)

Bit	Field	Type	Initial	Description
7:3	I2CSTA[7:3]	R	11111	I2C Status Code
2..0	I2CSTA[2:0]	R	000	SMBus Status Code

I2C status code and status

Mode	Status Code	Status of the I2C	Application software response					Next action taken by I2C hardware
			To/from I2CDAT	TO I2CCON				
				STA	STO	SI	AA	
Master Transmitter/ Receiver	08H	A START condition has been transmitted	Load SLA+R	X	0	0	X	SLA+R/W will be transmitted; ACK will be received
	10H	A repeated START condition has been transmitted.	Load SLA+R	X	0	0	X	SLA+R/W will be transmitted; ACK will be received
			Load SLA+W					SLA+W will be transmitted; I2C will be switched to MST/TRX mode.
Master Transmitter	18H	SLA+W has been transmitted; ACK has been received	Load data byte	0	0	0	X	Data byte will be transmitted; ACK will be received.
			No action	1	0	0	X	Repeated START will be transmitted.
			No action	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
			No action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
	20H	SLA+W has been transmitted; not ACK has been received	Load data byte*	0	0	0	X	Data byte will be transmitted; ACK will be received.
			No action	1	0	0	X	Repeated START will be transmitted.
			No action	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
			No action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
	28H	Data byte in I2CDAT has been transmitted; ACK has been received	Load data byte	0	0	0	X	Data byte will be transmitted; ACK bit will be received.
			No action	1	0	0	X	Repeated START will be transmitted.
			No action	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
			No action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
	30H	Data byte in I2CDAT has been transmitted; not ACK has been received	Load data byte*	0	0	0	X	Data byte will be transmitted; ACK will be received.
			No action	1	0	0	X	Repeated START will be transmitted.
			No action	0	1	0	X	STOP condition will be transmitted; STO flag will be reset.
			No action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset.
Master Receiver	40H	SLA+R has been transmitted; ACK has been received	No action	0	0	0	0	Data byte will be received; not ACK will be returned
			No action	0	0	0	1	Data byte will be received; ACK will be returned
	48H	SLA+R has been transmitted; not ACK has been received	No action	1	0	0	X	Repeated START condition will be transmitted
			No action	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
			No action	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset
	50H	Data byte has been received; ACK has been returned	Read data byte	0	0	0	0	Data byte will be received; not ACK will be returned
			Read data byte	0	0	0	1	Data byte will be received; ACK will be returned
	58H	Data byte has been received; not ACK has been returned	Read data byte	1	0	0	X	Repeated START condition will be transmitted
			Read data byte	0	1	0	X	STOP condition will be transmitted; STO flag will be reset
			Read data byte	1	1	0	X	STOP condition followed by a START condition will be transmitted; STO flag will be reset

Mode	Status Code	Status of the I2C	Application software response					Next action taken by I2C hardware
			To/from I2CDAT	TO I2CCON				
				STA	STO	SI	AA	
Slave Receiver	60H	Own SLA+W has been received; ACK has been returned	No action	X	0	0	0/1	Data byte will be received and not ACK/ACK will be returned
	68H	Arbitration lost in SLA+R/W as master; own SLA+W has been received, ACK returned	No action	X	0	0	0/1	Data byte will be received and not ACK/ACK will be returned
	70H	General call address (00H) has been received; ACK has been returned	No action	X	0	0	0/1	Data byte will be received and not ACK/ACK will be returned
	78H	Arbitration lost in SLA+R/W as master; general call address has been received, ACK returned	No action	X	0	0	0/1	Data byte will be received and not ACK/ACK will be returned
	80H	Previously addressed with own SLV address; DATA has been received; ACK returned	Read data byte	X	0	0	0/1	Data byte will be received and not ACK/ACK will be returned
	88H	Previously addressed with own SLA; DATA byte has been received; not ACK returned	Read data byte	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address
			Read data byte	0	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized
			Read data byte	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address; START condition will be transmitted when the bus becomes free
			Read data byte	1	0	0	1	Switched to not addressed SLV mode; own SLA or general

								call address will be recognized; START condition will be transmitted when the bus becomes free
	90H	Previously addressed with general call address; DATA has been received; ACK returned	Read data byte	X	0	0	0/1	Data byte will be received and not ACK/ACK will be returned
	98H	Previously addressed with general call address; DATA has been received; not ACK returned	Read data byte	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address
			Read data byte	0	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized
			Read data byte	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address; START condition will be transmitted when the bus becomes free
			Read data byte	1	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized; START condition will be transmitted when the bus becomes free
	A0H	A STOP condition or repeated START condition has been received while still addressed as SLV/REC or SLV/TRX	No action	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address
			No action	0	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized
			No action	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address; START condition will be transmitted when the bus becomes free
			No action	1	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized; START condition will be transmitted when the bus becomes free
Slave Transmitter	A8H	Own SLA+R has been received; ACK has been returned	Load data byte	X	0	0	0	Last data byte will be transmitted and ACK will be received
			Load data byte	X	0	0	1	Data byte will be transmitted; ACK will be received.
	B0H	Arbitration lost in SLA+R/W as master; own SLA+R has been received, ACK has been returned.	Load data byte	X	0	0	0	Last data byte will be transmitted and ACK will be received
			Load data byte	X	0	0	1	Data byte will be transmitted; ACK will be received.
	B8H	Data byte has been transmitted; ACK will be received.	Load data byte	X	0	0	0	Last data byte will be transmitted and ACK will be received
			Load data byte	X	0	0	1	Data byte will be transmitted; ACK will be received.
	C0H	Data byte has been transmitted; not ACK has been received.	No action	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address.
			No action	0	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized.
			No action	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address; START condition will be transmitted when the bus becomes free.
			No action	1	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized; START condition will be transmitted when the bus becomes free.
	C8H	Last data byte has been transmitted; ACK has been received.	No action	0	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address.
			No action	0	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized.
			No action	1	0	0	0	Switched to not addressed SLV mode; no recognition of own SLA or general call address; START condition will be transmitted when the bus becomes free.
			No action	1	0	0	1	Switched to not addressed SLV mode; own SLA or general call address will be recognized; START condition will be transmitted when the bus becomes free.
Miscellaneous	F8H	No relevant state information available; SI=0	No action	No action				Wait or proceed current transfer
	38H	Arbitration lost	No action	0	0	0	X	I2C will be released; A start condition will be transmitted.
			No action	1	0	0	X	When the bus becomes free. (enter to a master mode)
	00H	Bus error during MST or selected slave modes	No action	0	1	0	X	Only the internal hardware is affected in the MST or addressed SLV modes. In all cases, the bus is released and I2C is switched to the not addressed SLV mode. STO flag is reset.

"SLA" means slave address, "R" means R/W=1, "W" means R/W=0

*For applications where NACK doesn't mean the end of communication.

SMBSEL Register (0xDE)

Bit	Field	Type	Initial	Description
7	SMBEXE	R/W	0	SMBus extension functionality 0: Disable 1: Enable
2..0	SMBSTP[2:0]	R/W	000	SMBus timeout register

SMBDST Register (0xDF)

Bit	Field	Type	Initial	Description
7..0	SMBD[7:0]	R/W	0x00	This register is used to provide a read/write access port to the SMBus timeout registers. Data read or written to that register is actually read or written to the Timeout Register which is pointed by the SMBSEL register.

IEN0 Register (0xA8)

Bit	Field	Type	Initial	Description
7	EAL	R/W	0	Interrupts enable. Refer to Chapter Interrupt
Else				Refer to other chapter(s)

IEN1 Register (0xB8)

Bit	Field	Type	Initial	Description
0	EI2C	R/W	0	Interrupts enable. Refer to Chapter Interrupt
Else				Refer to other chapter(s)

P0M Register (0xF9)

Bit	Field	Type	Initial	Description
1	P01M	R/W	0	0: Set P0.1 (SDA) as input mode (required) ~~1: Set P0.1 (SDA) as output mode~~*
0	P00M	R/W	0	0: Set P0.0 (SCL) as input mode (required) ~~1: Set P0.0 (SCL) as output mode~~*
Else				Refer to other chapter(s)

* The P00M and P01M require be set input mode.

19.9 Sample Code

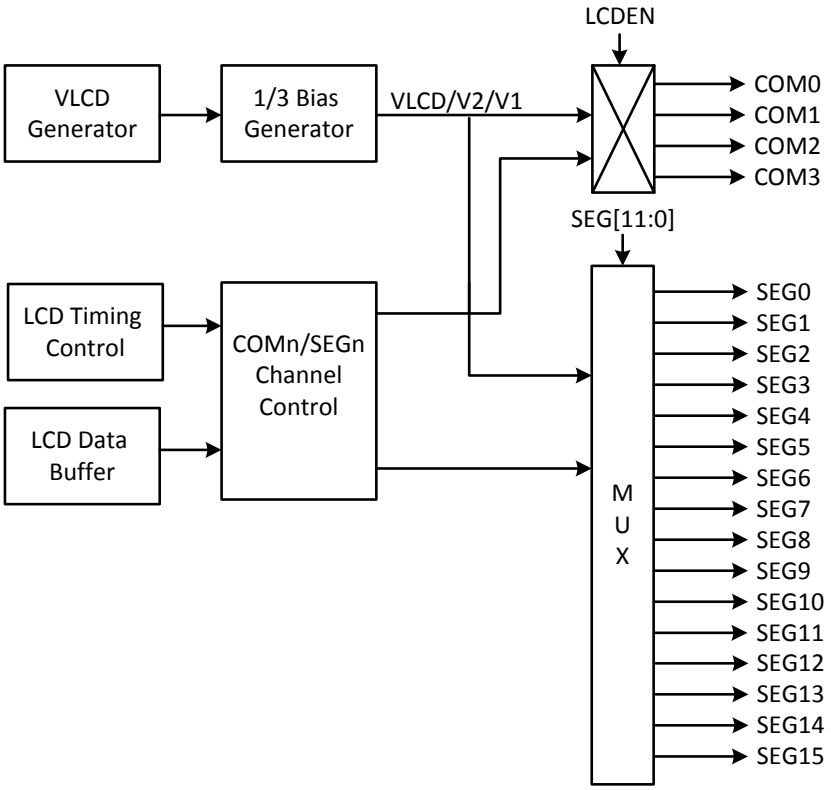
The following sample code demonstrates how to perform I2C with interrupt.

```
unsigned int  I2CAddr;
unsigned int  I2C_TXData0;
unsigned int  I2C_TXDatan;
unsigned int  I2C_RXData0;
unsigned int  I2C_RXDatan;

void I2CInit(void)
{
  P0M &= 0xFC;      // P00 and P01 as input

  // configure I2C clock(T1) and enable I2C.
  I2CCON = 0xC3;
  TMOD = 0x60;   // auto reload
  TCON0 = 0x07;  // Fosc/1
  TH1 = 0xF6;    //400kHz
  TL1 = 0xF6;    //400kHz or
  TH1 = 0xD8;    //100kHz
  TL1 = 0xD8;    //100kHz
  TR1 = 1;

  // enable I2C interrupt
  EI2C = 1;
  EAL = 1;       //enable global interrupt

  I2CCON |= 0x20;            // START (STA) = 1
}

void I2cInterrupt(void) interrupt ISRI2c  //0x43
{
    switch (I2CSTA)
    {
        // tx mode
        case 0x08:
            I2CCON &= 0xDF;           // START (STA) = 0
            I2CDAT = I2CAddr;        // Tx/Rx addr
            break;
        case 0x18:                    // write first byte
            I2CDAT = I2C_TXData0;
            break;
        case 0x28:                    // write n byte
            I2CDAT = I2C_TXDatan;
            break;
        case 0x30:                    // STOP (STO)
            I2CCON |= 0x10;
            break;
        //    rx mode
        case 0x40:                    // get slave addr
            I2CCON |= 0x04;           // AA = 1
            break;
        case 0x50:                    // read n byte
            I2C_RXData0 = I2CDAT;
            I2CCON &= 0xFB;           // AA = 0
            break;
        case 0x58:                    // read last byte & stop
```

```
54              I2C_RXDatan = I2CDAT;
55              I2CCON |= 0x10;           // STOP (STO)
56              break;
57          default:
58              I2CCON |= 0x10;            // STOP (STO)
59      }
60
61      I2CCON &= 0xF7;                    // Clear I2C flag (SI)
62  }
```

20 LCD

The LCD driver generates the control to drive a static or multiplexed LCD panel, with support for up to 16 segments multiplexed with up to 4 commons. The LCD also provides control of the LCD pixel data. The LCD driver supports a static mode controlled by LSTC control bit. If LSTC=0, the LCD driver is 1/3 bias, 1/4 duty LCD mode. If LSTC=1, the LCD driver is selected to static mode. The LCD has two clock source and four clock rates to decide LCD frame rate from 15.625Hz to 256Hz. The clock source is from internal 16KHz RC or external 32.768KHz oscillator crystal or RC type and controlled by LCDCKS bit.



In LCD mode, the LCD builds in one internal bias circuit to adjust LCD power and bias voltage. There are 16-pin GPIO shared with COM pins and SEG pins which controlled by LCDSEG and LCDSEG1 registers. The LCD builds in P0CON/P1CON/P2CON/P3CON registers to set pure analog input pin. After setup LCDEN bit, the LCD starts output analog data. In static mode, the selected COM/SEG pins switch to like GPIO output mode and only output 0V and VDD voltage.

LCD can work in idle mode. If LCD clock source is flosc and STWK=1, LCD also can work in stop mode.

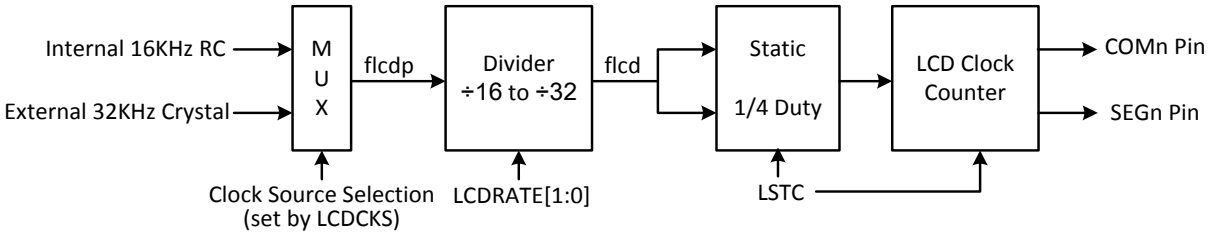
20.1 Configurations of Operation

These configurations must be setup completely before starting LCD. LCD is configured using the following steps:

1. The GPIO mode of LCD output channel must be set as low status. (By Pn register)
2. The GPIO mode of LCD output channel must be set as output mode. (By PnM register)
3. The configuration control bit of LCD output channel must be set. (By PnCON register)
4. Choose and enable the LCD input channel. (By LCDSEG[7:0] bits and LCDSEG1[7:0] bits)
5. Choose LCD mode. (By LSTC register)
6. Choose LCD Clock Source and Clock Rate. (By LCDCKS and LCDRATE[1:0] bits)
7. Choose LCD VLCD voltage and bias resistance. (By VLCD[3:0] and LCDBIA[3:0] bits, can be ignored in static mode)
8. Program LDC data to specific data buffer. (By LCDADR and LCDBUF register)
9. After setup LCDEN bits, the LCD ready to generate waveform.

20.2 LCD Timing Control

The LCD timing control generates the clock for LCD frame rate. The LCD has two clock sources (flcdp): internal 16KHz RC and external 32.768KHz crystal, controlled by LCDCKS bit. There are 4-level pre-scaler to divide the flcdp to obtain LCD frame clock source flcd. Each LCD frame length is 8*flcd.



LCD Frame Rate			
LCDRATE[1:0]	LCD clock rate	flcdp = Int. 16KHz	flcdp = Ext. 32.768KHz
00	flcdp/16	125 Hz	256 Hz
01	flcdp/32	62.5 Hz	128 Hz
10	flcdp/64	31.25 Hz	64 Hz
11	flcdp/128	15.625 Hz	32 Hz

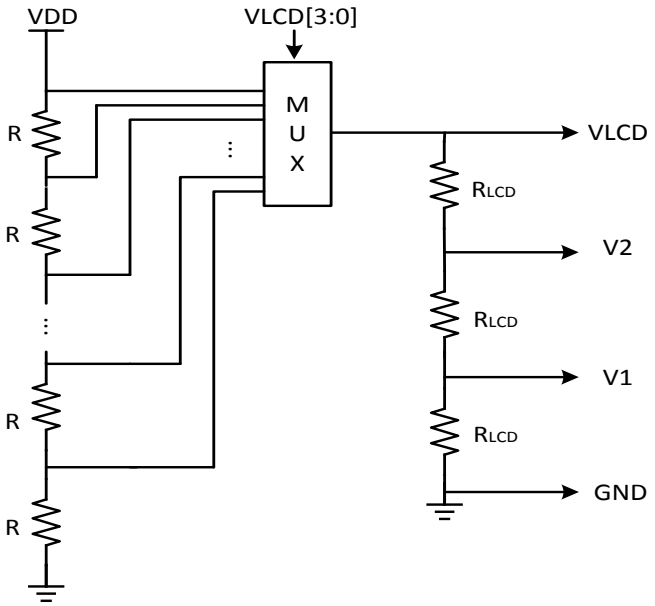
20.3 LCD Channel Control

The LCD include 4-COM pins and 16-SEG pins. Each of the LCD channels is shared with GPIO. If the LCD driver is enabled (LCDEN=1), the 4-COM pins are controlled to LCD COM pin mode, but the 16-SEG pins are controlled by SEG[15:0] selection bits (By LCDSEG[7:0] bits and LCDSEG1[7:0] bits). If the LCD driver is disabled (LCDEN=0), all COM/SEG pins are controlled to GPIO mode. Each of the LCD channels has a bias switch circuit to select a bias voltage for COM or SEG driving waveform. The bias voltage is from the LCD bias generator. The bias switch is controlled by LCD COM/SEG output control.

In LCD static mode, the bias switch of COM/SEG pins is disabled. The selected COM/SEG pins switch to like GPIO output mode and only output 0V and VDD voltage. The LCD COM/SEG output control circuits control the GPIO output latch of COM/SEG pins directly.

20.4 LCD Bias Generator

The LCD bias generator include LCD contrast control and 1/3 bias control. The LCD contrast control is used to set the voltage of VLCD. There are 16-stage VLCD voltage from VDD to VDD*0.5 and is controlled by VLCD[3:0]. The 1/3 bias control circuit has three resistances (R_{LCD}) for difference VLCD controlled by LCDBIA[2:0]. The R_{LCD} is range from 17.65KΩ to 300KΩ. The final 4-level LCD bias voltage source is VLCD/ V2/ V1/ GND and supply to LCD COM and SEG pins. When the static mode active (LSTC=1), both LCD contrast control and 1/3 bias control will be turned off.



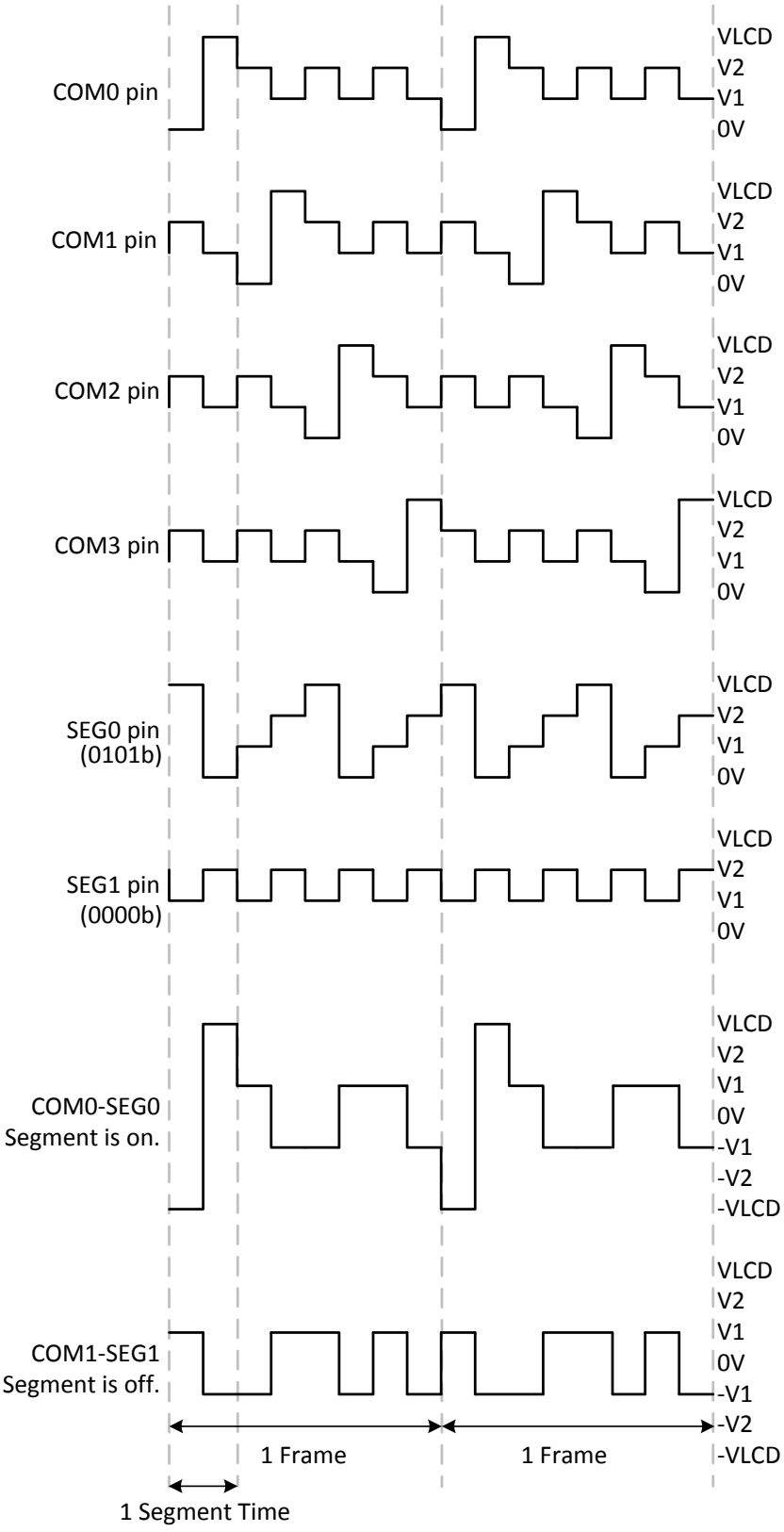
20.5 LCD Data Buffer

The LCD data buffer stores LCD map data and the density is 16x4-bit. The LCD data buffer is indirectly address memory and accessed through LCDADR and LCDBUF control registers. The low-nibble of LCDADR indicate the address of each LCD segment data buffer from SEG0~SEG15. The high-nibble of LCDADR is useless. Program the LCD data must set address (LCDADR) first and then write data to LCDBUF. One LCD data includes four common bits data. COM0~COM3 is in low-nibble of one LCDBUF. The high-nibble of LCDBUF is useless. The LCD RAM map is as following.

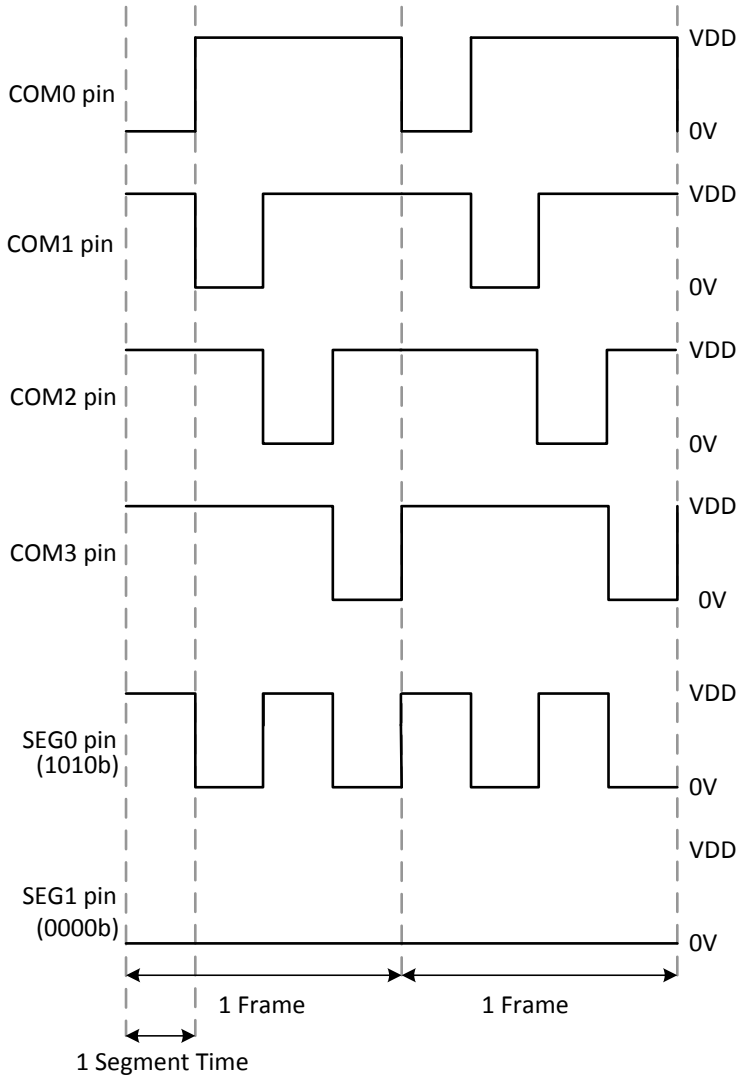
LCD	LCDADR[3:0]	LCDBUF[3:0]			
		COM3	COM2	COM1	COM0
SEG0	0000	C3S0	C2S0	C1S0	C0S0
SEG1	0001	C3S1	C2S1	C1S1	C0S1
SEG2	0010	C3S2	C2S2	C1S2	C0S2
SEG3	0011	C3S3	C2S3	C1S3	C0S3
SEG4	0100	C3S4	C2S4	C1S4	C0S4
SEG5	0101	C3S5	C2S5	C1S5	C0S5
SEG6	0110	C3S6	C2S6	C1S6	C0S6
SEG7	0111	C3S7	C2S7	C1S7	C0S7
SEG8	1000	C3S8	C2S8	C1S8	C0S8
SEG9	1001	C3S9	C2S9	C1S9	C0S9
SEG10	1010	C3S10	C2S10	C1S10	C0S10
SEG11	1011	C3S11	C2S11	C1S11	C0S11
SEG12	1100	C3S12	C2S12	C1S12	C0S12
SEG13	1101	C3S13	C2S13	C1S13	C0S13
SEG14	1110	C3S14	C2S14	C1S14	C0S14
SEG15	1111	C3S15	C2S15	C1S15	C0S15

20.6 LCD Waveform Generation

Waveform in LCD mode:



Waveform in static mode:



20.7 LCD Registers

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LCDCON	LCDEN	LSTC	LCDRATE1	LCDRATE0	VLCD3	VLCD2	VLCD1	VLCD0
LCDMOD	-	-	-	-	LCDCKS	LCDBIA2	LCDBIA1	LCDBIA0
LCDSEG	SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0
LCDSEG1	SEG15	SEG14	SEG13	SEG12	SEG11	SEG10	SEG9	SEG8
LCDBUF	-	-	-	-	LCDDAT3	LCDDAT2	LCDDAT1	LCDDAT0
LCDADR	-	-	-	-	SADR3	SADR2	SADR1	SADR0
P0CON	P0CON7	P0CON6	P0CON5	-	-	-	-	-
P1CON	P1CON7	P1CON6	P1CON5	P1CON4	P1CON3	P1CON2	P1CON1	P1CON0
P2CON	P2CON7	P2CON6	P2CON5	P2CON4	P2CON3	P2CON2	P2CON1	P2CON0
P3CON							P3CON1	P3CON0

LCDCON Register (0XA1)

Bit	Field	Type	Initial	Description
7	LCDEN	R/W	0	LCD driver control bit. 0: Disable 1: Enable
6	LSTC	R/W	0	LCD static mode select bit. 0: LCD mode. 1: Static mode.
5..4	LCDRATE[1:0]	R/W	0	LCD clock rate select bit 00: flcd/16 01: flcd/32 10: flcd/64 11: flcd/128
3..0	VLCD[3:0]	R/W	0x00	LCD contrast adjustment bit. 0000: VLCD=VDD, 0001: VLCD=VDD*0.97, 0010: VLCD=VDD*0.93, 0011: VLCD=VDD*0.9, 0100: VLCD=VDD*0.87, 0101: VLCD=VDD*0.83, 0110: VLCD=VDD*0.8, 0111: VLCD=VDD*0.77, 1000: VLCD=VDD*0.73, 1001: VLCD=VDD*0.7, 1010: VLCD=VDD*0.67, 1011: VLCD=VDD*0.63, 1100: VLCD=VDD*0.6, 1101: VLCD=VDD*0.57, 1110: VLCD=VDD*0.53, 1111: VLCD=VDD*0.5

LCDMOD Register (0XA2)

Bit	Field	Type	Initial	Description
3	LCDCKS	R/W	0	LCD clock source (flcd) select bit. 0: Internal 16KHz RC 1: External 32KHz crystal*
2..0	LCDBIA[2:0]	R/W	000	LCD bias resistance select bit 000: Disable 001: RLCD = 25KΩ 010: RLCD = 75KΩ 011: RLCD = 18.75KΩ 100: RLCD = 300KΩ 101: RLCD = 23.08KΩ 110: RLCD = 60KΩ 111: RLCD = 17.65KΩ
Else	Reserved	R	0	

*Before choose External 32KHz crystal as LCD clock source, the LCKS bit must be set to enable 32KHz crystal function.

LCDSEG Register (0XA3)

Bit	Field	Type	Initial	Description
7..0	SEG[7:0]	R/W	0x00	LCD SEGn channel control bit. 0: GPIO pin 1: SEG pin.

LCDSEG1 Register (0XA6)

Bit	Field	Type	Initial	Description
7..0	SEG[15:8]	R/W	0x00	LCD SEGn channel control bit. 0: GPIO pin 1: SEG pin.

LCDBUF Register (0XA4)

Bit	Field	Type	Initial	Description
3..0	LCDDAT[3:0]	R/W	0000	LCD SEG data buffer.
Else	Reserved	R	0	

LCDADR Register (0XA5)

Bit	Field	Type	Initial	Description
3..0	SADR[3:0]	R/W	0000	LCD SEG data buffer address.
Else	Reserved	R	0	

P0CON Register (0x9E)

Bit	Field	Type	Initial	Description
7..5	P0CON[7:5]	R/W	0x00	P0 configuration control bit*. 0: P0 can be analog input pin (LCD pin) or digital GPIO pin. 1: P0 is pure analog input pin and can't be a digital GPIO pin.

* P0CON [7:5] will configure related Port0 pin as pure analog input pin to avoid current leakage.

P1CON Register (0x9F)

Bit	Field	Type	Initial	Description
7..0	P1CON[7:0]	R/W	0x00	P1 configuration control bit*. 0: P1 can be analog input pin (LCD pin) or digital GPIO pin. 1: P1 is pure analog input pin and can't be a digital GPIO pin.

* P1CON [7:0] will configure related Port1 pin as pure analog input pin to avoid current leakage.

P2CON Register (0XD6)

Bit	Field	Type	Initial	Description
7..0	P2CON[7:0]	R/W	0x0	P2 configuration control bit*. 0: P2 can be analog input pin (LCD pin) or digital GPIO pin. 1: P2 is pure analog input pin and can't be a digital GPIO pin.

* P2CON [7:0] will configure related Port2 pin as pure analog input pin to avoid current leakage.

P3CON Register (0XD7)

Bit	Field	Type	Initial	Description
1..0	P3CON[1:0]	R/W	0x0	P3 configuration control bit*. 0: P3 can be analog input pin (LCD pin) or digital GPIO pin. 1: P3 is pure analog input pin and can't be a digital GPIO pin.

* P3CON [7:0] will configure related Port3 pin as pure analog input pin to avoid current leakage.

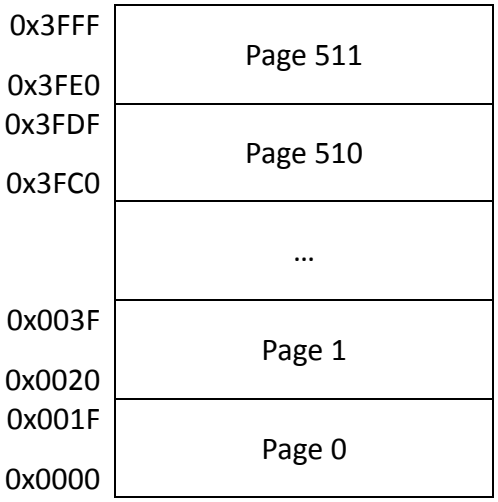
20.8 Sample Code

The following sample code demonstrates how to perform LCD with COM0~COM3 and SEG5.

```
#define LCDBias         (2 << 0)   //RLCD = 75KΩ
#define LCDClkILRC      (0 << 3)   //LCD clock source = ILRC
#define LCDClk32K       (1 << 3)   //LCD clock source = Ext. 32KHz
#define LCDVLCD         (0 << 0)   //VLCD = VDD
#define LCDSpeedDiv16   (0 << 4)   //LCD clock = flcd/16
#define LCDSpeedDiv32   (1 << 4)   //LCD clock = flcd/32
#define LCDSpeedDiv64   (2 << 4)   //LCD clock = flcd/64
#define LCDSpeedDiv128  (3 << 4)   //LCD clock = flcd/128
#define SelSEG5         (5 << 0)   //select SEG5
#define LCDStatic       (1 << 6)   //Set static mode
#define LCDEn           (1 << 7)   //enable LCD

void ADCInit(void)
{
  P0  = 0x00;
  P1  = 0x00;
  P0M = 0xE0;
  P1M = 0x41;
  // set COM0~COM3 & SEG5 pin's mode at pure analog pin
  P0CON |= 0xE0;     //COM0/P05, COM1/P06 & COM2/P07
  P1CON |= 0x41;     //COM3/P10 & SEG5/P16

  // configure LCD channel, LCD clock source and LCD Bias.
  LCDMOD = LCDClkILRC | LCDBias;
  LCDSEG = SelSEG5;

  // configure LCD clock divider and VLCD.
  LCDCON = LCDSpeedDiv128 | LCDVLCD;

  // Program SEG5 LCD data
  LCDADR = 0x05;
  LCDBUF = 0x5A;

  // enable LCD
  LCDCON |= LCDEn;
}
```

21 In-System Program

SN8F5814 builds in an on-chip 16 KB program memory, aka IROM, which is equally divided to 512 pages (32 bytes per page). The in-system program is a procedure that enables a firmware to freely modify every page's data; in other word, it is the channel to store value(s) into the non-volatile memory and/or live update firmware.



Program memory (IROM)

21.1 Page Program

Because each page of the program memory has 32 bytes in length, a page program procedure requires 32 bytes IRAM as its data buffer.

ISP ROM MAP		ROM address bit0~bit4 (hex) =0
ROM address bit5~bit15 (hex)	0000	These pages include reset vector and interrupt sector. We strongly recommend to reserve the area not to do ISP erase.
	0020	
	0040	
	…	
	00C0	
	00E0	
	0100	One ISP Program Page
	0120	One ISP Program Page
	…	One ISP Program Page
	1000	One ISP Program Page
	1020	One ISP Program Page
	1040	One ISP Program Page
	…	One ISP Program Page
	3FA0	One ISP Program Page
	3FC0	One ISP Program Page
	3FE0	This page includes ROM reserved area. We strongly recommend to reserve the area not to do ISP erase.

These configurations must be setup completely before starting Page Program. ISP is configured using the following steps:

1. Save program data into IRAM. The data continues for 32 bytes.
2. Set the start address of the content location to PERAM.
3. Set the start address of the anticipated update area to PEROM [15:5]. (By PEROMH/PRROML registers)
4. Write '0x5A' into PECMD [7:0] to trigger ISP function.
5. Write 'NOP' instruction twice.

As an example, assume the 510th page of program memory (IROM, 0x3FC0 – 0x3FDF) is the anticipated update area; the content is already stored in IRAM address 0x60 – 0x7F. To perform the in-system program, simply write starting IROM address 0x3FC0 to EPROMH/EPROML registers, and then specify buffer starting address 0x60 to EPRAM register. Subsequently, write '0x5A' into PECMD [7:0] registers to duplicate the buffer's data to 510th page of IROM.

In general, every page has the capability to be modified by in-system program procedure. However, since the first and least pages (page 0 and 511) respectively stores reset vector and information for power-on controller, incorrectly perform page program (such as turn off power while programming) may cause faulty power-on sequence / reset.

✷ ***Note:***

1. Watch dog timer should be clear before the Flash write (program) operation, or watchdog timer would overflow and reset system during ISP operating.

2. Don't execute ISP flash ROM program operation for the first page and the last page, or affect program operation.

21.2 In-system Program Register

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PERAM	PERAM7	PERAM6	PERAM5	PERAM4	PERAM3	PERAM2	PERAM1	PERAM0
PEROMH	PEROM15	PEROM14	PEROM13	PEROM12	PEROM11	PEROM10	PEROM9	PEROM8
PEROML	PEROM7	PEROM6	PEROM5	-	-	-	-	-
PECMD	PECMD7	PECMD6	PECMD5	PECMD4	PECMD3	PECMD2	PECMD1	PECMD0

PERAM Register (0x97)

Bit	Field	Type	Initial	Description
7..0	PERAM[7:0]	R/W	0x00	The first address of data buffer (IRAM)

PEROMH Register (0x96)

Bit	Field	Type	Initial	Description
7..0	PEROM[15:8]	R/W	0x00	The first address (15^{th}– 8^{th} bit) of program page (IROM)

PEROML Register (0x95)

Bit	Field	Type	Initial	Description
7..0	PEROM[7:5]	R/W	000	The first address (7^{th}– 5^{th} bit) of program page (IROM)
Else	Reserved	R	0	

PECMD Register (0x94)

Bit	Field	Type	Initial	Description
7..0	PECMD[7:0]	W	-	0x5A: Start page program procedure ~~Else values: Reserved~~ *(1)

*(1) Not permitted to write any other to PECMD register.

21.3 Sample Code

```
unsigned cahr idata dataBuffer[32] _at_ 0xE0;  // IRAM 0xE0 to 0xFF

void SYSIspSetDataBuffer(unsigned char address, unsigned char data)
{
  dataBuffer[address & 0x1F] = data;
}

void SYSPageIspStart(unsigned int pageAddress)
{
  PISP(pageAddress, 0xE0);  //Page program, data save 0xE0 to 0xFF
}
```

22 Electrical Characteristics

22.1 Absolute Maximum Ratings

Voltage applied at VDD to VSS ..... - 0.3V to 6.0V
Voltage applied at any pin to VSS..... - 0.3V to VDD+0.3V
Operating ambient temperature.....-40°C to 85°C
Storage ambient temperature ..... -40°C to 125°C
Junction Temperature..... -40°C to 125°C

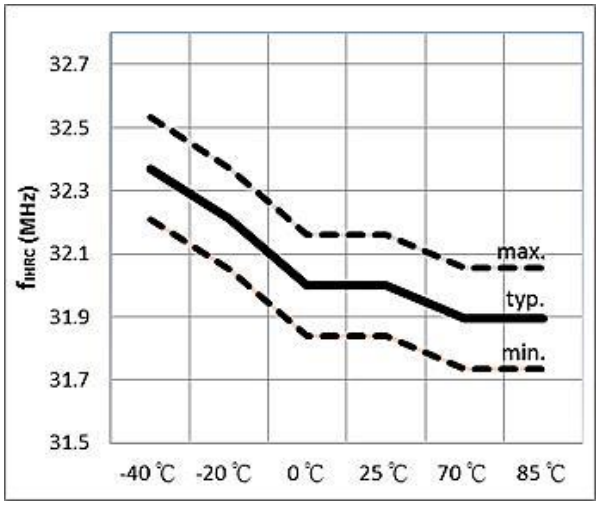
22.2 System Operation Characteristics

	Parameter	Test Condition	Min	TYP	MAX	UNIT
VDD	Operating voltage	fcpu = 1MHz	1.8		5.5	V
V_{DR}	RAM data retention Voltage		1.5			V
V_{POR}	VDD rising rate*		0.05			V/ms
I_{DD1}	Normal mode supply current (CKCON = 0x00, 32MHz IHRC)	VDD = 3V, fcpu = 1MHz		2.77		mA
		VDD = 5V, fcpu = 1MHz		2.78		mA
		VDD = 3V, fcpu = 4MHz		3.24		mA
		VDD = 5V, fcpu = 4MHz		3.26		mA
		VDD = 3V, fcpu = 8MHz		3.87		mA
		VDD = 5V, fcpu = 8MHz		3.89		mA
	Normal mode supply current (CKCON = 0x00, 16MHz Crystal)	VDD = 3V, fcpu = 1MHz		2.52		mA
		VDD = 5V, fcpu = 1MHz		3.18		mA
		VDD = 3V, fcpu = 4MHz		3.06		mA
		VDD = 5V, fcpu = 4MHz		3.72		mA
		VDD = 3V, fcpu = 8MHz		3.70		mA
		VDD = 5V, fcpu = 8MHz		4.29		mA
	Normal mode supply current (CKCON = 0x00, 4MHz Crystal)	VDD = 3V, fcpu = 1MHz		2.04		mA
		VDD = 5V, fcpu = 1MHz		2.24		mA
		VDD = 3V, fcpu = 4MHz		2.58		mA
		VDD = 5V, fcpu = 4MHz		2.78		mA
I_{DD2}	STOP mode supply current	VDD = 3V		4	8.5	µA
		VDD = 5V		4.9	9.0	µA
I_{DD3}	STOP mode supply current (LCD enable, no panel)	VDD = 3V, 32KHz Crystal		108		µA
		VDD = 5V, 32KHz Crystal		124		µA
		VDD = 3V, Internal 16K		101		µA
		VDD = 5V, Internal 16K		112		µA

I_{DD4}	IDLE mode supply current (fcpu = 1MHz)	VDD = 3V, 32MHz IHRC		1.02		mA
		VDD = 5V, 32MHz IHRC		1.04		mA
		VDD = 3V, 16MHz Crystal		0.77		mA
		VDD = 5V, 16MHz Crystal		1.41		mA
		VDD = 3V, 4MHz Crystal		0.33		mA
		VDD = 5V, 4MHz Crystal		0.53		mA
F_{IHRC}	Internal high clock generator	VDD = 1.8V to 5.5V, 25°C	31.84	32	32.16	MHz
		VDD = 1.8V to 5.5V, 25°C to 85°C	31.68	-	31.99	MHz
		VDD = 1.8V to 5.5V, -40°C to 25°C	32.31	-	32.64	MHz
F_{ILRC}	Internal low clock generator	VDD = 5.0V, 25°C	12	16	24	kHz
V_{LVD18}	LVD18 detect voltage	25°C	1.7	1.8	1.9	V
		-40°C to 85°C	1.6	1.8	2.0	V

* Parameter(s) with star mark are non-verified design reference. Ambient temperature is 25°C.

- IHRC Frequency - Temperature Graph



22.3 GPIO Characteristics

Parameter		Test Condition	Min	TYP	MAX	UNIT
V_{IL}	Low-level input voltage		VSS		0.3VDD	V
V_{IH}	High-level input voltage		0.7VDD		VDD	V
I_{LEKG}	I/O port input leakage current	V_{IN} = VDD			2	µA
R_{UP}	Pull-up resister	VDD = 3V	100	200	300	kΩ
		VDD = 5V	50	100	150	kΩ
I_{OH}	I/O output source current	VDD = 5V, V_O = VDD-0.5V	12	19		mA

I_{OL1}	I/O sink current (P00 – P04 , P11 – P17, P2, P3)	VDD = 5V, V_O = VSS+0.5V	15	22		mA
I_{OL2}	I/O sink current (P05 – P07, P10)	VDD = 5V, V_O = VSS+1.5V	200			mA

* Ambient temperature is 25°C.

22.4 ADC Characteristics

	Parameter	Test Condition	Min	TYP	MAX	UNIT
V_{ADC}	Operating voltage		2.0		5.5	V
V_{AIN}	AIN channels input voltage	VDD = 5V	0		V_{REFH}	V
V_{REFH}	AVREFH pin input voltage	VDD = 5V	2		VDD	V
V_{IREF}	Internal VDD reference voltage	VDD = 5V		VDD		V
	Internal 4V reference voltage	VDD = 5V	3.92	4	4.08	V
	Internal 3V reference voltage	VDD = 5V	2.94	3	3.06	V
	Internal 2V reference voltage	VDD = 5V	1.96	2	2.04	V
I_{AD}	ADC current consumption	VDD = 3V		0.69		mA
		VDD = 5V		0.79		mA
f_{ADCLK}	ADC clock	VDD = 5V			32	MHz
f_{ADSMP}	ADC sampling rate	VDD = 5V			500	kHz
t_{ADEN}	ADC function enable period	VDD = 5V	100			µs
DNL	Differential nonlinearity*	f_{ADSMP} = 62.5kHz		±1		LSB
		f_{ADSMP} = 250kHz		±1		LSB
		f_{ADSMP} = 500kHz		±3		LSB
INL	Integral Nonlinearity*	f_{ADSMP} = 62.5kHz		±2		LSB
		f_{ADSMP} = 250kHz		±2		LSB
		f_{ADSMP} = 500kHz		±4		LSB
NMC	No missing code*	f_{ADSMP} = 62.5kHz	10	11	12	Bit
		f_{ADSMP} = 250kHz		10		Bit
		f_{ADSMP} = 500kHz		9		Bit
V_{OFFSET}	Input offset voltage	Non-trimmed	-10	0	10	mV

* Parameters with star mark: VDD = 5V, V_{REFH} = 2.4V, 25°C.

22.5 Flash Memory Characteristics

Parameter		Test Condition	Min	TYP	MAX	UNIT
V_{dd}	Supply voltage		1.8		5.5	V
T_{en}	Endurance time	25°C		*100K		cycle
I_{wrt}	Write current	25°C		3	4	mA
T_{wrt}	Write time	Write 1 page=32 bytes, 25°C		6	8	ms

* Parameters with star mark are non-verified design reference.

23 Instruction Set

This chapter categorizes the SN8F5814 microcontroller's comprehensive assembly instructions. It includes five categories—arithmetic operation, logic operation, data transfer operation, Boolean manipulation, and program branch—which are fully compatible with standard 8051.

Symbol description

Symbol	Description
`Rn`	Working register R0 - R7
`direct`	One of 128 internal RAM locations or any Special Function Register
`@Ri`	Indirect internal or external RAM location addressed by register R0 or R1
`#data`	8-bit constant (immediate operand)
`#data16`	16-bit constant (immediate operand)
`bit`	One of 128 software flags located in internal RAM, or any flag of bit-addressable Special Function Registers
`addr16`	Destination address for LCALL or LJMP, can be anywhere within the 64-Kbyte page of program memory address space
`addr11`	Destination address for ACALL or AJMP, within the same 2-Kbyte page of program memory as the first byte of the following instruction
`rel`	SJMP and all conditional jumps include an 8-bit offset byte. Its range is +127/-128 bytes relative to the first byte of the following instruction
`A`	Accumulator

Arithmetic operations

Mnemonic	Description
ADD A, Rn	Add register to accumulator
ADD A, direct	Add directly addressed data to accumulator
ADD A, @Ri	Add indirectly addressed data to accumulator
ADD A, #data	Add immediate data to accumulator
ADDC A, Rn	Add register to accumulator with carry
ADDC A, direct	Add directly addressed data to accumulator with carry
ADDC A, @Ri	Add indirectly addressed data to accumulator with carry
ADDC A, #data	Add immediate data to accumulator with carry
SUBB A, Rn	Subtract register from accumulator with borrow
SUBB A, direct	Subtract directly addressed data from accumulator with borrow
SUBB A, @Ri	Subtract indirectly addressed data from accumulator with borrow
SUBB A, #data	Subtract immediate data from accumulator with borrow
INC A	Increment accumulator
INC Rn	Increment register
INC direct	Increment directly addressed location
INC @Ri	Increment indirectly addressed location
INC DPTR	Increment data pointer
DEC A	Decrement accumulator
DEC Rn	Decrement register
DEC direct	Decrement directly addressed location
DEC @Ri	Decrement indirectly addressed location
MUL AB	Multiply A and B
DIV	Divide A by B
DA A	Decimally adjust accumulator

Logic operations

Mnemonic	Description
ANL A, Rn	AND register to accumulator
ANL A, direct	AND directly addressed data to accumulator
ANL A, @Ri	AND indirectly addressed data to accumulator
ANL A, #data	AND immediate data to accumulator
ANL direct, A	AND accumulator to directly addressed location
ANL direct, #data	AND immediate data to directly addressed location
ORL A, Rn	OR register to accumulator

ORL A, direct	OR directly addressed data to accumulator
ORL A, @Ri	OR indirectly addressed data to accumulator
ORL A, #data	OR immediate data to accumulator
ORL direct, A	OR accumulator to directly addressed location
ORL direct, #data	OR immediate data to directly addressed location
XRL A, Rn	Exclusive OR (XOR) register to accumulator
XRL A, direct	XOR directly addressed data to accumulator
XRL A, @Ri	XOR indirectly addressed data to accumulator
XRL A, #data	XOR immediate data to accumulator
XRL direct, A	XOR accumulator to directly addressed location
XRL direct, #data	XOR immediate data to directly addressed location
CLR A	Clear accumulator
CPL A	Complement accumulator
RL A	Rotate accumulator left
RLC A	Rotate accumulator left through carry
RR A	Rotate accumulator right
RRC A	Rotate accumulator right through carry
SWAP A	Swap nibbles within the accumulator

Data transfer operations

Mnemonic	Description
MOV A, Rn	Move register to accumulator
MOV A, direct	Move directly addressed data to accumulator
MOV A, @Ri	Move indirectly addressed data to accumulator
MOV A, #data	Move immediate data to accumulator
MOV Rn, A	Move accumulator to register
MOV Rn, direct	Move directly addressed data to register
MOV Rn, #data	Move immediate data to register
MOV direct, A	Move accumulator to direct
MOV direct, Rn	Move register to direct
MOV direct1, direct2	Move directly addressed data to directly addressed location
MOV direct, @Ri	Move indirectly addressed data to directly addressed location
MOV direct, #data	Move immediate data to directly addressed location
MOV @Ri, A	Move accumulator to indirectly addressed location
MOV @Ri, direct	Move directly addressed data to indirectly addressed location
MOV @Ri, #data	Move immediate data to in directly addressed location

MOV DPTR, #data16	Load data pointer with a 16-bit immediate
MOVC A, @A+DPTR	Load accumulator with a code byte relative to DPTR
MOVC A, @A+PC	Load accumulator with a code byte relative to PC
MOVX A, @Ri	Move external RAM (8-bit address) to accumulator
MOVX A, @DPTR	Move external RAM (16-bit address) to accumulator
MOVX @Ri, A	Move accumulator to external RAM (8-bit address)
MOVX @DPTR, A	Move accumulator to external RAM (16-bit address)
PUSH direct	Push directly addressed data onto stack
POP direct	Pop directly addressed location from stack
XCH A, Rn	Exchange register with accumulator
XCH A, direct	Exchange directly addressed location with accumulator
XCH A, @Ri	Exchange indirect RAM with accumulator
XCHD A, @Ri	Exchange low-order nibbles of indirect and accumulator

Boolean manipulation

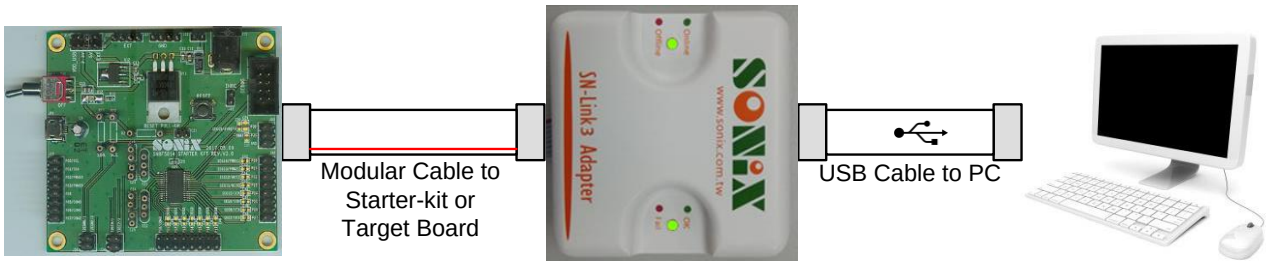
Mnemonic	Description
CLR C	Clear carry flag
CLR bit	Clear directly addressed bit
SETB C	Set carry flag
SETB bit	Set directly addressed bit
CPL C	Complement carry flag
CPL bit	Complement directly addressed bit
ANL C, bit	AND directly addressed bit to carry flag
ANL C, /bit	AND complement of directly addressed bit to carry
ORL C, bit	OR directly addressed bit to carry flag
ORL C, /bit	OR complement of directly addressed bit to carry
MOV C, bit	Move directly addressed bit to carry flag
MOV bit, C	Move carry flag to directly addressed bit

Program branches

Mnemonic	Description
ACALL addr11	Absolute subroutine call
LCALL addr16	Long subroutine call
RET	Return from subroutine
RETI	Return from interrupt
AJMP addr11	Absolute jump
LJMP addr16	Long jump
SJMP rel	Short jump (relative address)
JMP @A+DPTR	Jump indirect relative to the DPTR
JZ rel	Jump if accumulator is zero
JNZ rel	Jump if accumulator is not zero
JC rel	Jump if carry flag is set
JNC rel	Jump if carry flag is not set
JB bit, rel	Jump if directly addressed bit is set
JNB bit, rel	Jump if directly addressed bit is not set
JBC bit, rel	Jump if directly addressed bit is set and clear bit
CJNE A, direct, rel	Compare directly addressed data to accumulator and jump if not equal
CJNE A, #data, rel	Compare immediate data to accumulator and jump if not equal
CJNE Rn, #data, rel	Compare immediate data to register and jump if not equal
CJNE @Ri, #data, rel	Compare immediate to indirect and jump if not equal
DJNZ Rn, rel	Decrement register and jump if not zero
DJNZ direct, rel	Decrement directly addressed location and jump if not zero
NOP	No operation for one cycle

24 Development Environment

SONIX provides an Embedded ICE emulator system to offer SN8F5814 firmware development. The platform is an in-circuit debugger and controlled by Keil C51 IDE software on Microsoft Windows platform. The platform includes SN-Link3, SN8F5814 Starter-kit and Keil C51 IDE software to build a high-speed, low cost, powerful and multi-task development environment including emulator, debugger and programmer. To execute emulation is like run real chip because the emulator circuit integrated in SN8F5814 to offer a real development environment.



24.1 Minimum Requirement

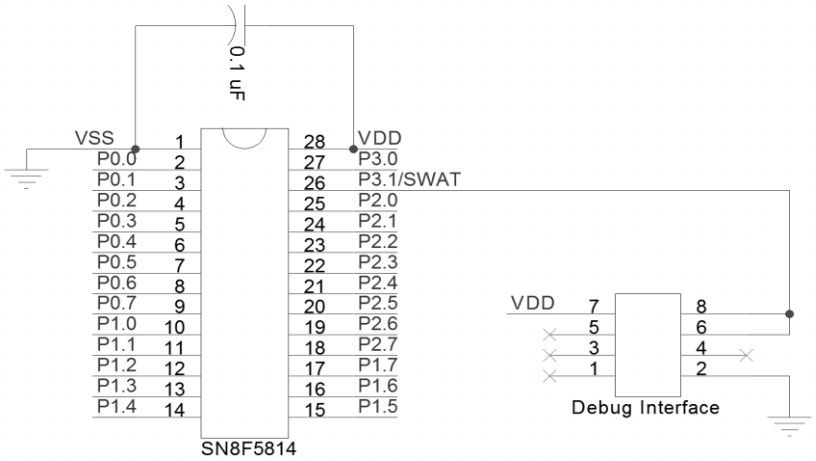
The following items are essential to build up an appropriate development environment. The compatibility is verified on listed versions, and is expected to execute perfectly on later version. SN-Link related information is available to download on SONiX website (www.sonix.com.tw);
Keil C51 is downloadable on www.keil.com/c51.

- **SN-Link3 Adapter** with updated firmware version 1.02
- **SN-Link Driver for Keil C51** version 1.00.317
- **Keil C51** version 9.50a and 9.54a or greater.

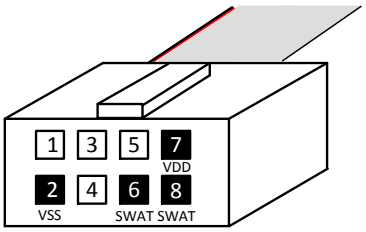
24.2 Debug Interface Hardware

The circuit below demonstrates the appropriate method to connect microcontroller's SWAT pin and SN-Link3 Adapter.

Before starting debug, microcontroller's power (VDD) must be switched off. Connect the SWAT to both 6th and 8th pins of SN-Link, and respectively link VDD and VSS to 7th pin and 2nd pin. A handshake procedure would be automatically started by turn on the microcontroller, and SN-Link's green LED (Run) indicates the success of connection (refer *SN8F5000 Debug Tool Manual* for further detail).



example circuit

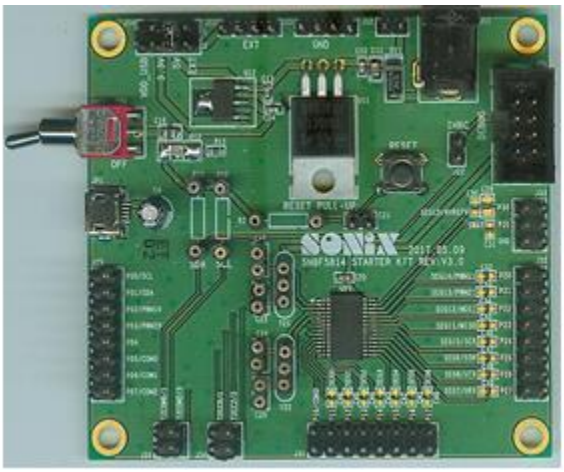


SN-Link header

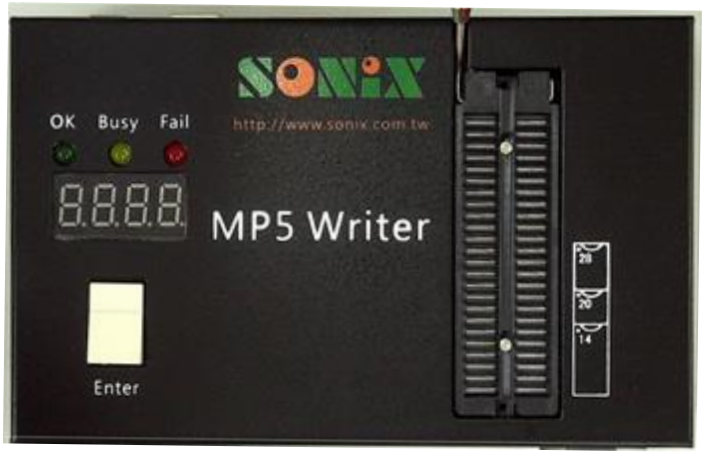
24.3 Development Tool

SN-Link3 Adapter

Starter-Kit support SN8F5814, SN8F5813, SN8F5812



MP5 Writer



25 SN8F5814 Starter-Kit

SN8F5000 Starter-Kit provides easy-development platform. It includes SN8F5000 family real chip and I/O connectors to input signal or drive device of user's application. It is a simple platform to develop application as target board not ready. The Starter-Kit can be replaced by target board, because SN8F5000 family integrates embedded ICE in-circuit debugger circuitry.

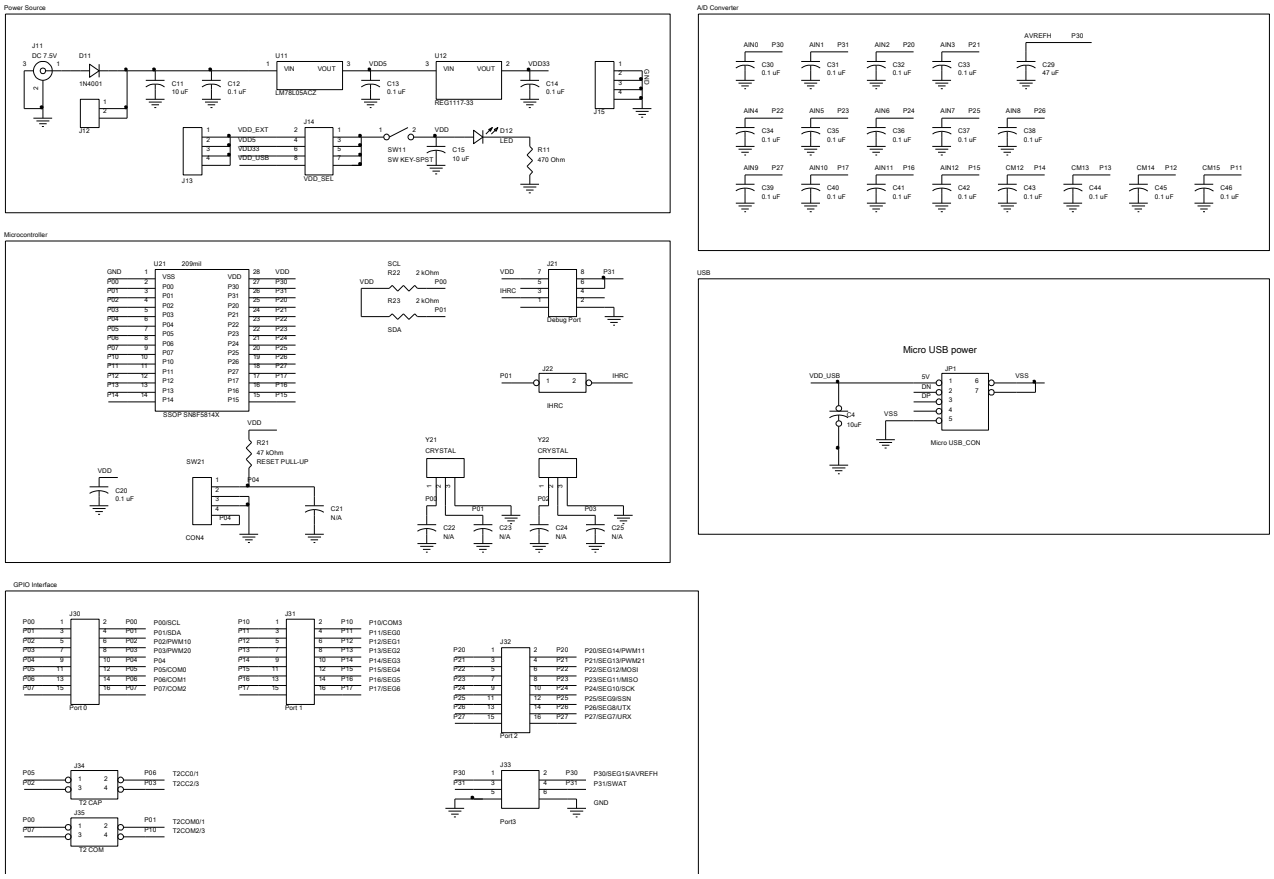
25.1 Configurations of Circuit

These configurations must be setup completely before starting Starter-Kit developing.

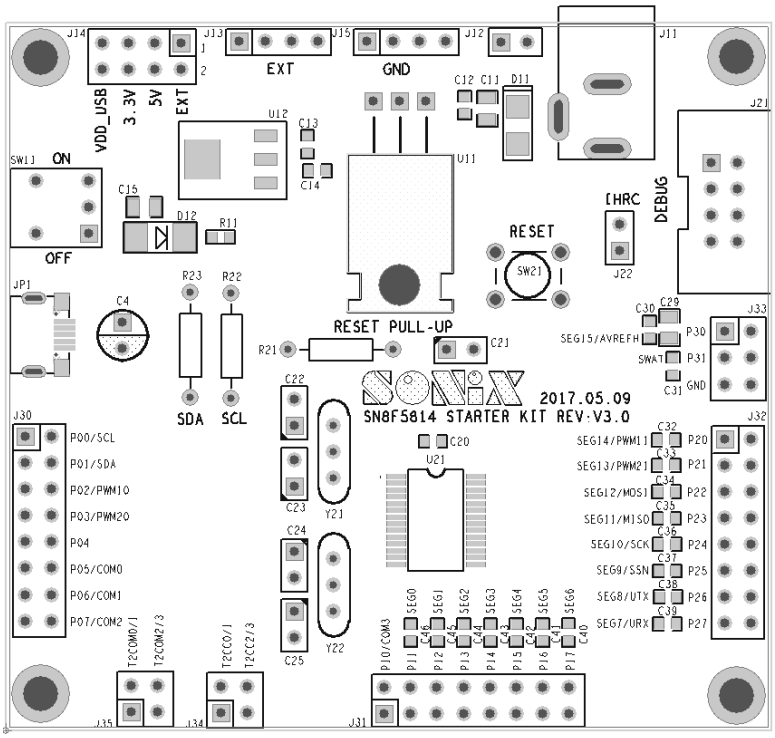
1. Confirm to the circuit board whether elements are complete.
2. The power source of Starter-Kit circuit is chosen from 5.0V, 3.3V, external power or Micro USB via jumper.
3. The power source comes from 5.0V or 3.3V which must be connect to DC 7.5V power adapter.
4. If the power source is chosen from external power, then external power source connects to EXT pin.
5. The "RST" pin needs to connect pull high resister to VDD when external reset is chosen to use.
6. The "XIN" pin and the "XOUT" pin need to connect crystal/resonator oscillator components when system clock is setting crystal or RTC mode.
7. The "XIN" pin needs to connect external clock source when system clock is setting external clock input mode.
8. The Debug Port can connect SN-LINK Adapter for emulation or download code.

The MCU LED will light up and SN8F5000 family chip will be connected to power when power (VDD) is switched on.

25.2 Schematic



25.3 Floor Plan of PCB layout



25.4 Component Description

Number	Description
C30 – C42	13-ch ADC capacitors.
C29	AVREFH capacitor.
D12	MCU LED
J11	DC 7.5V power adapter
J13/J15	External power source.
SW21	External reset trigger source
J14	VDD power source is 5.0V, 3.3V or external power.
J21	Debug Port
J30 – J33	I/O connector.
J34	Timer 2 capture connector.
J35	Timer 2 compare connector.
R21, C21	External reset pull-high resister and capacitor.
R22, R23	I2C pull-high resisters.
SW11	Target power (VDD) switch
U21	SN8F5814X real chip (Sonix standard option).
Y21, C22, C23, Y22, C24, C25	External crystal/resonator oscillator components.

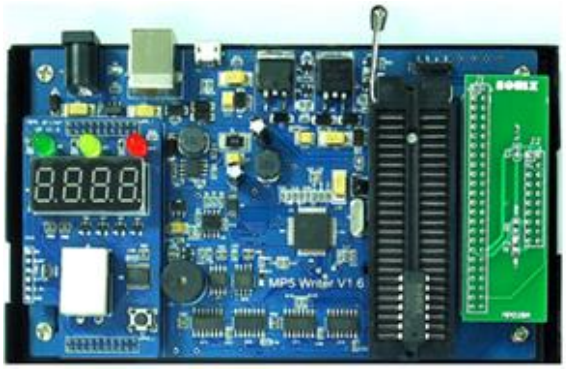
26 ROM Programming Pin

SN8F5814 Series Flash ROM erase/program/verify support SN-Link and MP5 Writer

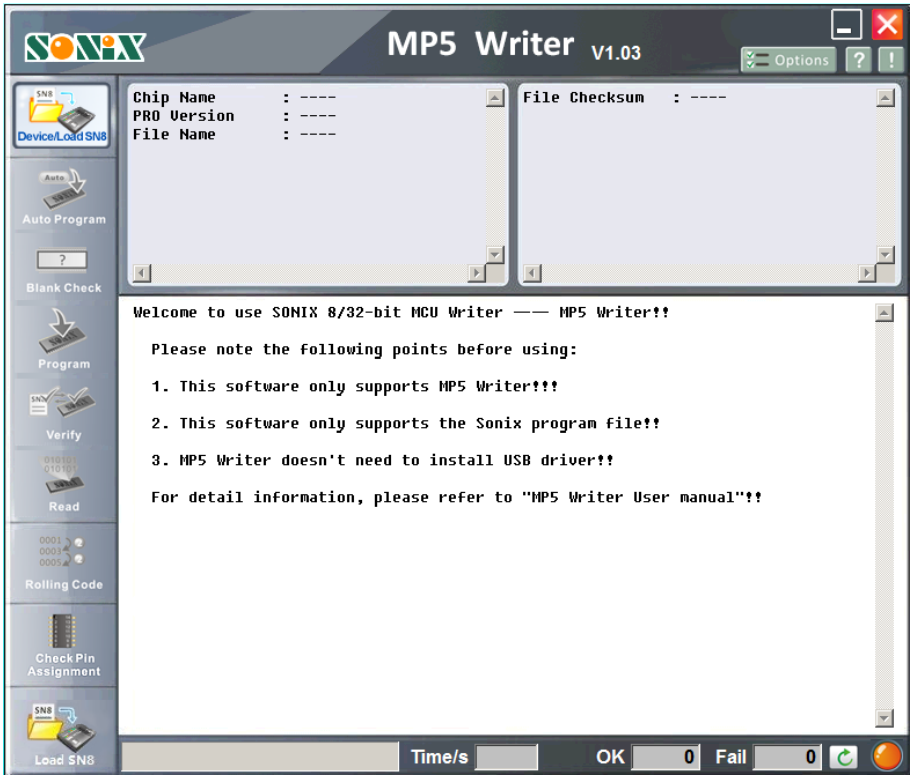
- SN-Link: Debug interface and on board programming.
- MP5 Writer: For SN8F5814 series version mass programming.

26.1 MP5 Hardware Connecting

Different package type with MCU programming connecting is as following, DIP and SOP/TSSOP Illustration.

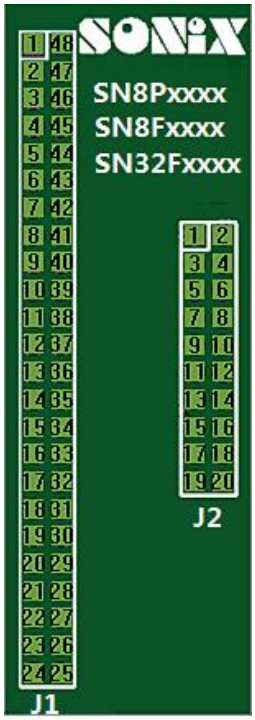


MP5 Software operation interface is as following.



26.2 MP5 Writer Transition Board Socket Pin Assignment

MP5 Writer Transition Board:



26.3 MP5 Writer Programming Pin Mapping

There are two modes of MP5 writer programming: normal mode and high speed mode. Normal mode requires four pins to program the code. The high speed mode requires eight pins to program the code for fast programming.

Normal mode can meet most programming needs. However, if you want to shorten the programming time, you can use the high speed mode. High speed mode requires a more stable connection environment. Please confirm whether the environment can meet the requirements before use.

26.3.1 Normal Mode

Writer Connector		MCU Pin Number	SN8F5814S/X/T		SN8F5813S/T		SN8F5812S/T	
J2 Pin Number	J2 Pin Name		MCU Pin Number	J1 Pin Number	MCU Pin Number	J1 Pin Number	MCU Pin Number	J1 Pin Number
1	VDD	VDD	28	38	24	36	20	34
2	GND	VSS	1	11	1	13	1	15
7	SWAT	P3.1	26	36	22	34	18	32
9	SWAT	P3.1	26	36	22	34	18	32
20	PDB	P0.1	3	13	3	15	3	17

26.3.2 High Speed Mode

Writer Connector		MCU Pin Number	SN8F5814S/X/T		SN8F5813S/T		SN8F5812S/T	
J2 Pin Number	J2 Pin Name		MCU Pin Number	J1 Pin Number	MCU Pin Number	J1 Pin Number	MCU Pin Number	J1 Pin Number
1	VDD	VDD	28	38	24	36	20	34
2	GND	VSS	1	11	1	13	1	15
3	DFTCLK	P2.0	25	35	21	33	17	31
5	SEL	P2.1	24	34	20	32	16	30
7	SWAT	P3.1	26	36	22	34	18	32
9	SWAT	P3.1	26	36	22	34	18	32
11	DAH	P2.2	23	33	19	31	15	29
13	DAL	P2.3	22	32	18	30	14	28
20	PDB	P0.1	3	13	3	15	3	17

26.4 SN-Link ISP Programming

SN-Link ISP programming hardware and software are as following.

26.5 SN-Link ISP Programming Pin Mapping

SN-Link Connector		MCU Pin Number	SN8F5814S/X/T	SN8F5813S/T	SN8F5812S/T
Pin Number	Pin Name		Pin Number	Pin Number	Pin Number
7	VDD	VDD	28	24	20
2	GND	VSS	1	1	1
6	SWAT	P3.1	26	22	18
8	SWAT	P3.1	26	22	18

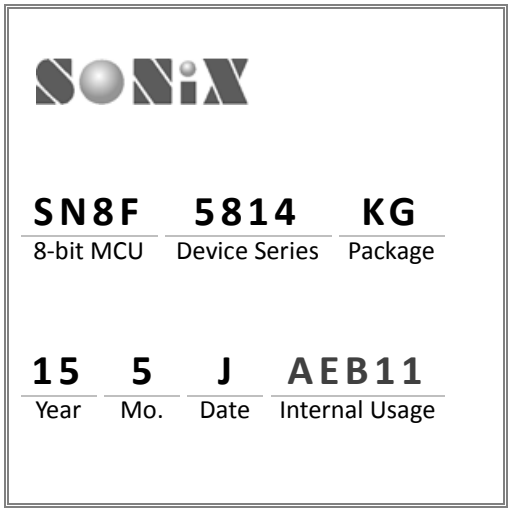
27 Ordering Information

A typical surface of SONiX microcontroller is printed with three columns: logo, device's full name, and date code.

SONiX Logo

Full Name

Date Code



27.1 Device Nomenclature

Full Name	Packing Type
S8F5814W	Wafer
SN8F5814H	Dice
SN8F5814SG	SOP, 28 pins, Green package
SN8F5814XG	SSOP, 28 pins, Green package
SN8F5814TG	TSSOP, 28 pins, Green package
SN8F5813SG	SOP, 24 pins, Green package
SN8F5813TG	TSSOP, 24 pins, Green package
SN8F5812SG	SOP, 20 pins, Green package
SN8F5812TG	TSSOP, 20 pins, Green package

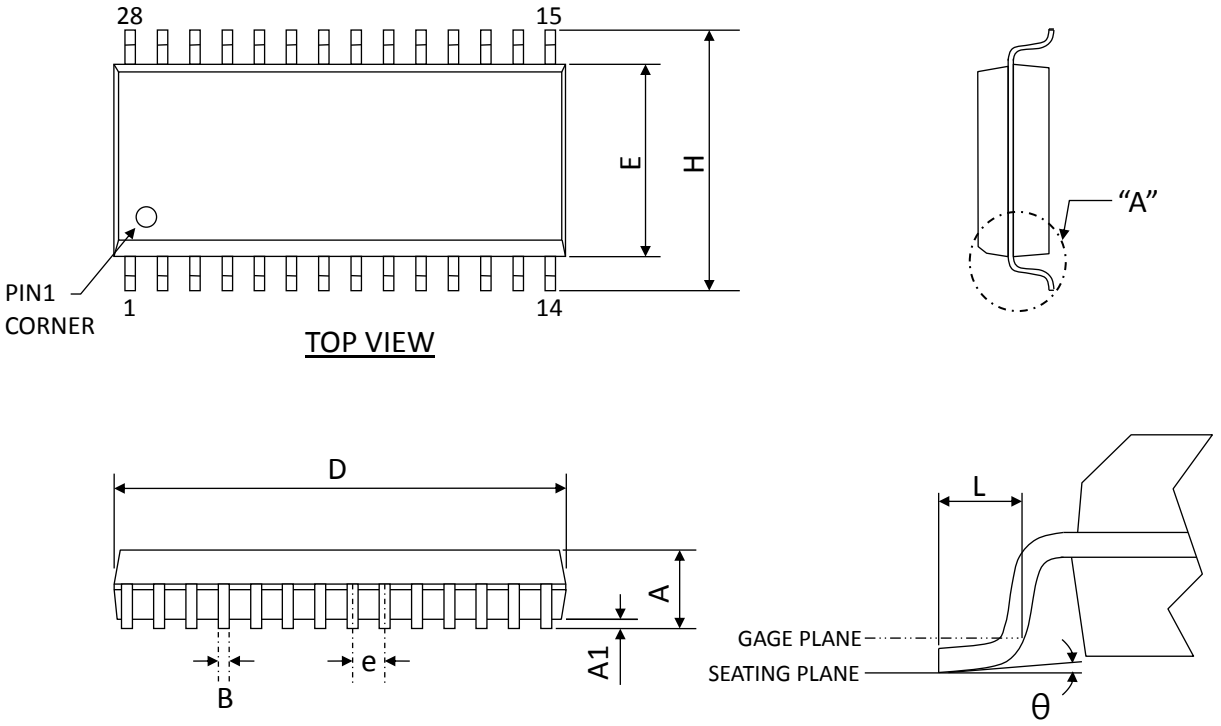
27.2 Date Code

The date code includes two parts: date of manufacture and production serial code. The first part is public information which is encoded by following principles.

Year	15: 2015 16: 2016 17: 2017 et cetera
Month	1: January 2: February 3: March A: October B: November C: December et cetera
Date	1: 01 2: 02 3: 03 A: 10 B: 11 et cetera

28 Package Information

28.1 SOP28

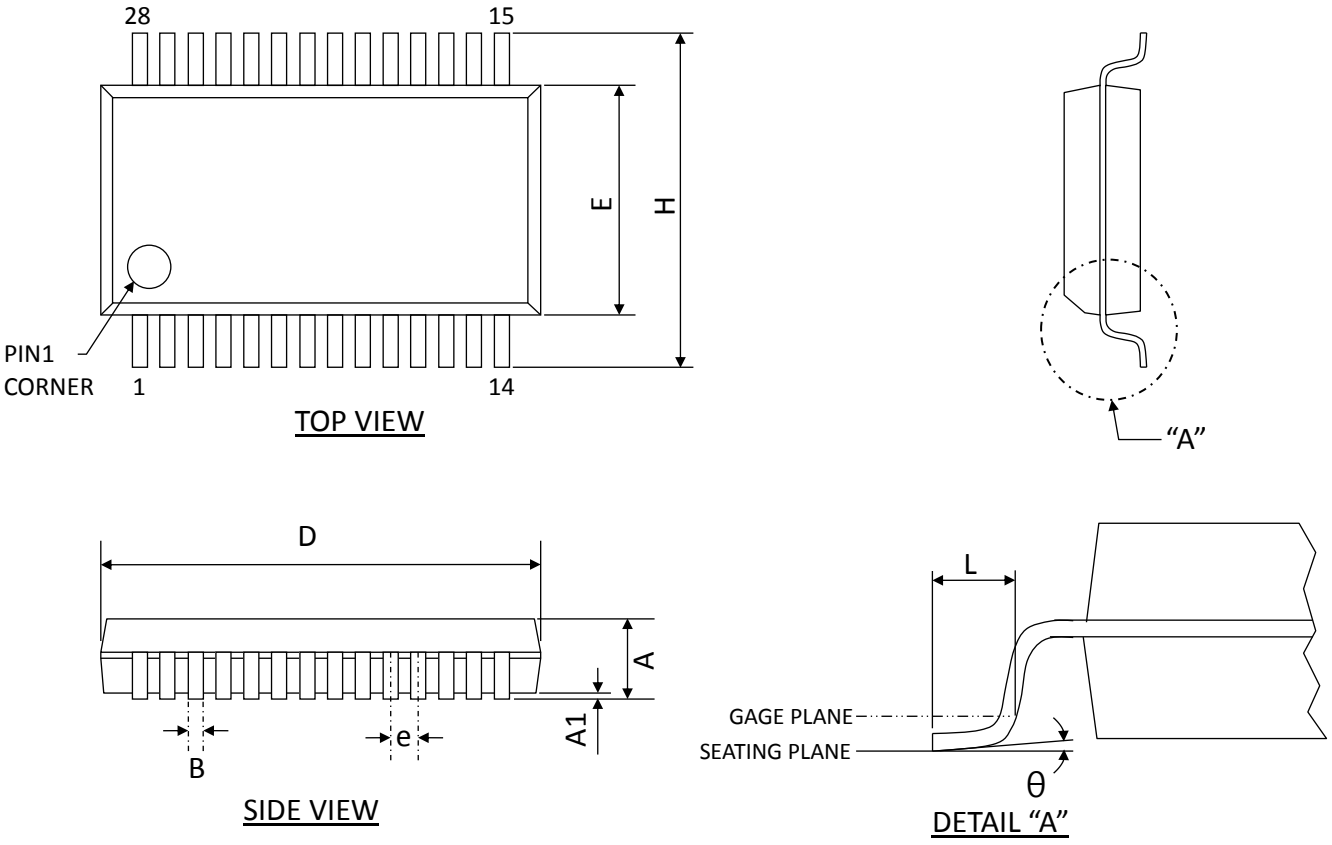


SYMBOLS	Dimension in mm			Dimension in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	--	--	2.65	--	--	0.104
A1	0.10	--	0.30	0.004	--	0.011
B	0.31	0.41	0.51	0.012	0.016	0.020
D	17.70	18.20	18.70	0.697	0.716	0.736
E	7.50 BSC			0.295 BSC		
e	1.27 BSC			0.050 BSC		
H	10.30 BSC			405 BSC		
L	0.40	--	1.27	0.016	--	0.050
θ	0°	4 °	8°	0°	4 °	8°

Notes：

1. CONTROLLING DIMENSION：mm
2. JEDEC OUTLINE : MO-119 AB

28.2 SSOP28

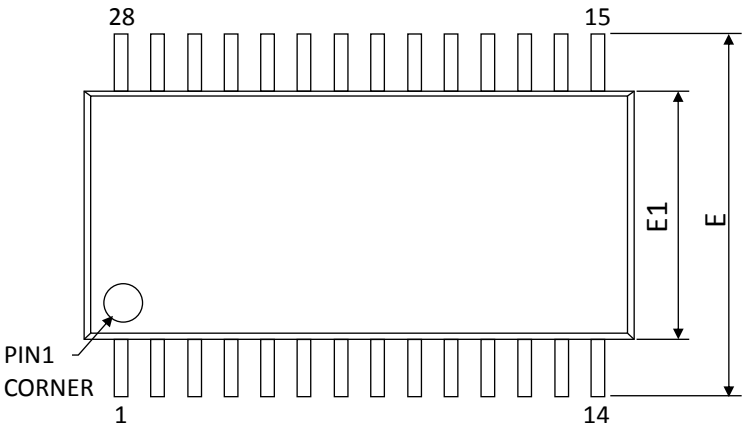


SYMBOLS	Dimension in mm			Dimension in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	--	--	2.0	--	--	0.079
A1	0.05	--	--	0.002	--	--
B	0.22	--	0.38	0.009	--	0.015
D	10.05	10.20	10.50	0.396	0.402	0.413
E	5.00	5.30	5.60	0.197	0.209	0.220
e	0.65 BSC.			0.026 BSC.		
H	7.65	7.80	7.90	0.301	0.307	0.311
L	0.55	0.80	1.05	0.022	0.031	0.041
θ	0°	4°	8°	0°	4°	8°

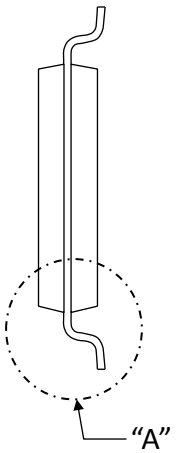
Notes：

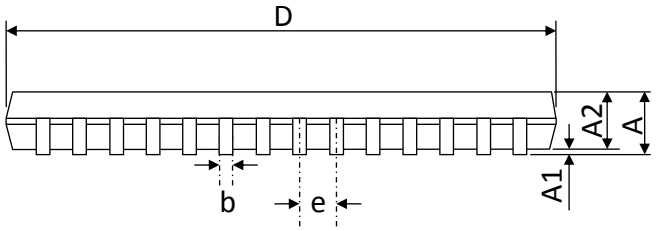
1. CONTROLLING DIMENSION：mm
2. JEDEC OUTLINE : MO-105 AH

28.3 TSSOP28

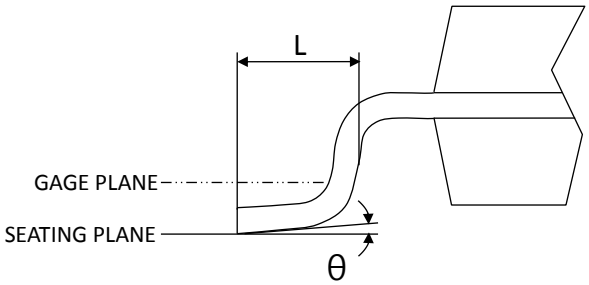


TOP VIEW





SIDE VIEW



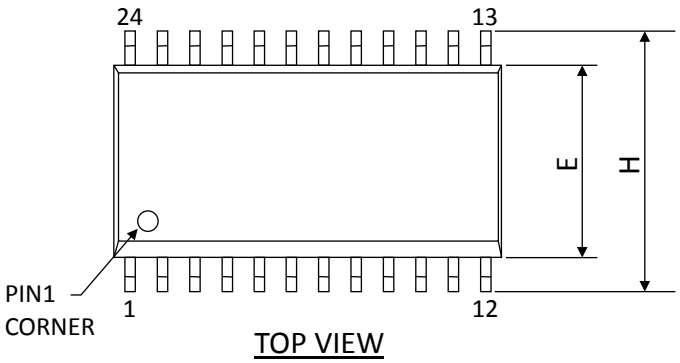
DETAIL "A"

SYMBOLS	Dimension in mm			Dimension in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	--	--	1.20	--	--	0.047
A1	0.00	--	0.15	0.000	--	0.006
A2	0.80	1.00	1.05	0.031	0.039	0.041
b	0.19	--	0.30	0.007	--	0.012
D	9.60	9.70	9.80	0.378	0.382	0.386
E		6.40 BSC.			0.252 BSC.	
E1	4.30	4.40	4.50	0.169	0.173	0.177
e		0.65 BSC.			0.026 BSC.	
L	0.45	0.60	0.75	0.018	0.024	0.030
θ	0°	--	8°	0°	--	8°

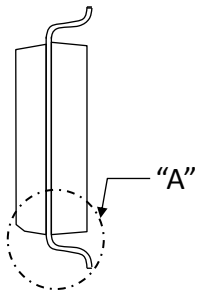
Notes：

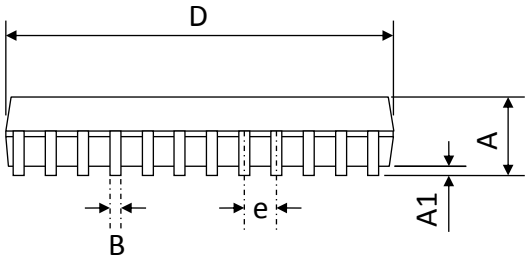
1. CONTROLLING DIMENSION：mm
2. JEDEC OUTLINE : MO-153
3. DIMENSION 'D' DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BERRES.
4. DIMENSION 'E1' DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION.
5. DIMENSION 'b' DOES NOT INCLUDE DAMBAR PROTRUSION.

28.4 SOP24

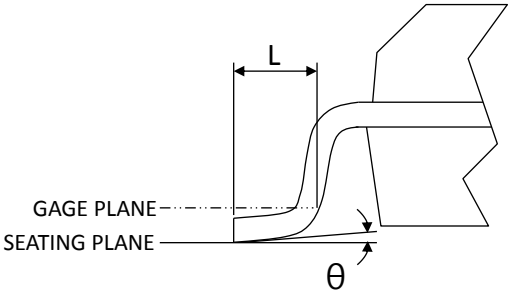


TOP VIEW





SIDE VIEW



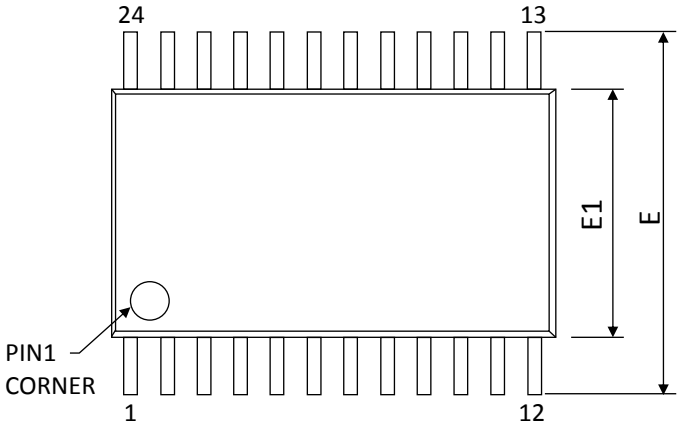
DETAIL "A"

SYMBOLS	Dimension in mm			Dimension in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	--	--	2.65	--	--	0.104
A1	0.10	--	0.30	0.004	--	0.011
B	0.31	0.41	0.51	0.012	0.016	0.020
D	15.30	15.50	15.70	0.602	0.610	0.618
E	7.50 BSC			0.295 BSC		
e	1.27 BSC			0.050 BSC		
H	10.30 BSC			0.405 BSC		
L	0.4	--	1.27	0.015	--	0.05
θ	0°	4 °	8°	0°	4 °	8°

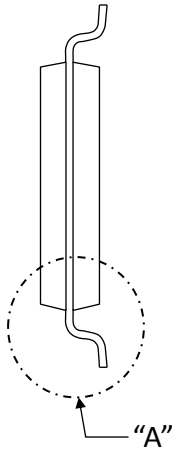
Notes：

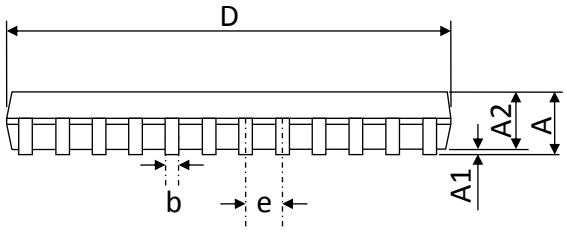
1. CONTROLLING DIMENSION：mm
2. JEDEC OUTLINE : MO-119 AA

28.5 TSSOP24

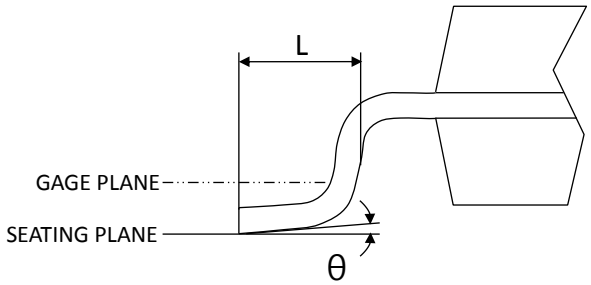


TOP VIEW





SIDE VIEW



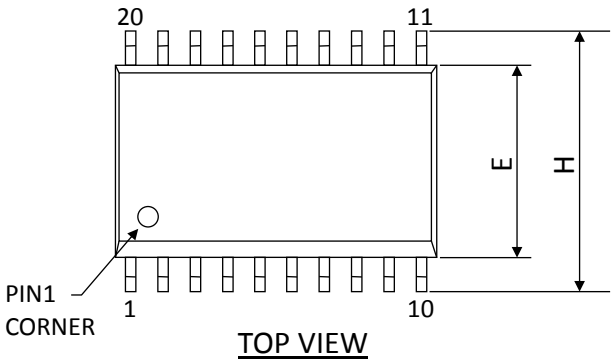
DETAIL "A"

SYMBOLS	Dimension in mm			Dimension in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	--	--	1.20	--	--	0.047
A1	0.00	--	0.15	0.000	--	0.006
A2	0.80	1.00	1.05	0.031	0.039	0.041
b	0.19	--	0.30	0.007	--	0.012
D	7.70	7.80	7.90	0.303	0.307	03.11
E	6.40 BSC.			0.252 BSC.		
E1	4.30	4.40	4.50	0.169	0.173	0.177
e	0.65 BSC.			0.026 BSC.		
L	0.45	0.60	0.75	0.018	0.024	0.030
θ	0°	--	8°	0°	--	8°

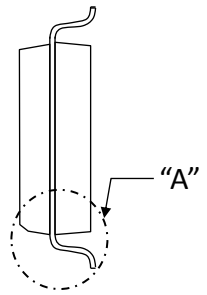
Notes：

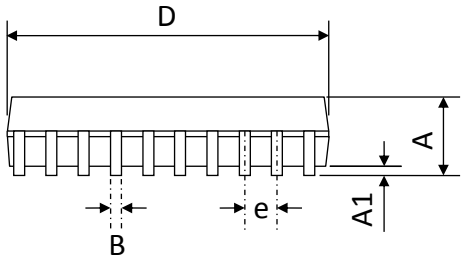
1. CONTROLLING DIMENSION：mm
2. JEDEC OUTLINE : MO-153
3. DIMENSION 'D' DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BERRES.
4. DIMENSION 'E1' DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION.
5. DIMENSION 'b' DOES NOT INCLUDE DAMBAR PROTRUSION.

28.6 SOP20

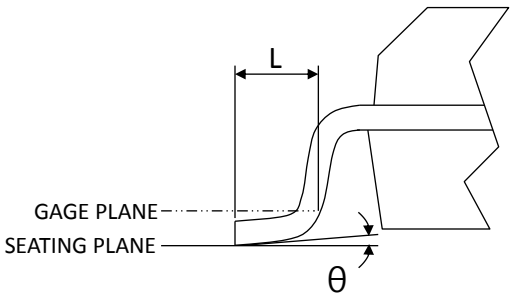


TOP VIEW





SIDE VIEW



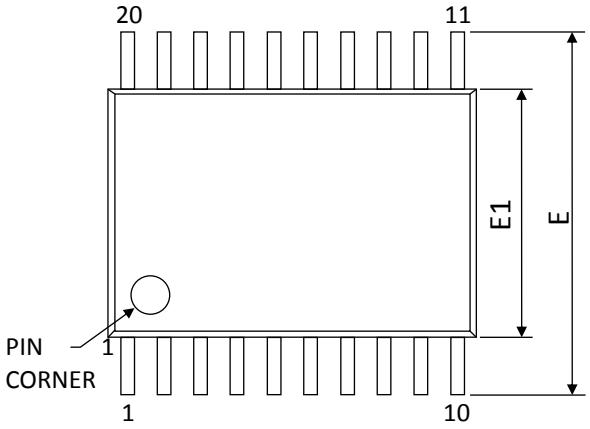
DETAIL "A"

SYMBOLS	Dimension in mm			Dimension in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	--	--	2.65	--	--	0.104
A1	0.10	--	0.30	0.004	--	0.012
B	0.31	0.41	0.51	0.012	0.016	0.020
D	12.80 BSC			0.503		
E	7.50 BSC			0.295		
e	1.27 BSC			0.050 BSC		
H	10.30 BSC			0.405		
L	0.40	--	1.27	0.016	--	0.050
θ	0°	4°	8°	0°	4°	8°

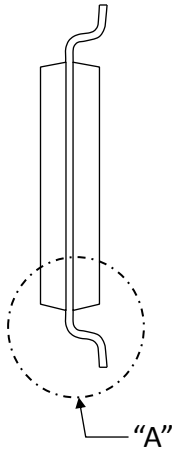
Notes：

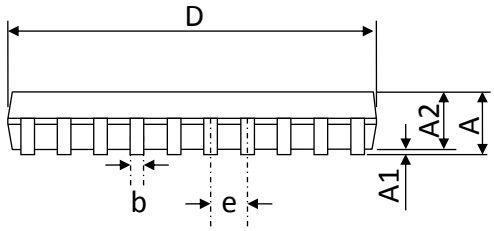
1. CONTROLLING DIMENSION：mm
2. JEDEC OUTLINE : MO-013 AC

28.7 TSSOP20

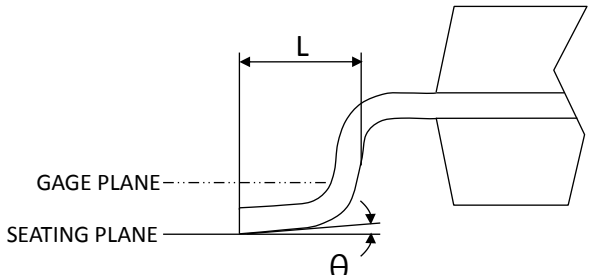


TOP VIEW





SIDE VIEW



DETAIL "A"

SYMBOLS	Dimension in mm			Dimension in inch		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	--	--	1.20	--	--	0.047
A1	0.05	--	0.15	0.002	--	0.006
A2	0.80	--	1.05	0.031	--	0.041
b	0.19	--	0.30	0.007	--	0.012
D	6.40	6.50	6.60	0.252	0.256	0.260
E		6.40 BSC.			0.252 BSC.	
E1	4.30	4.40	4.50	0.169	0.173	0.177
e		0.65 BSC.			0.026 BSC.	
L	0.45	0.60	0.75	0.018	0.024	0.030
θ	0°	--	8°	0°	--	8°

Notes：

1. CONTROLLING DIMENSION：mm
2. JEDEC OUTLINE : MO-153
3. DIMENSION 'D' DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BERRES.
4. DIMENSION 'E1' DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION.
5. DIMENSION 'b' DOES NOT INCLUDE DAMBAR PROTRUSION.

29 Appendix: Reference Document

Sonix provides reference document for users to help them quickly familiar SN8F5000 family (downloadable on cooperative website: www.sonix.com.tw).

Document Name	Description
SN8F5000 Starter-Kit User Manual	This documentation introduces SN8F5000 family all Starter-Kit, providing the user selects an appropriate starter-kit for development.
SN8F5000 Family Instruction Set	The document details the 8051 instruction set, and a simple example illustrates operation.
SN8F5000 Family Instruction Mapping Table	This document supplies the information about mapping assembly instructions from 8-Bit Flash/ OTP Type to 8051 Flash Type.
SN8F5000 Packaging Information	This documentation introduces SN8F5000 family microcontrollers' mechanical data, such as height, width and pitch information.
SN8F5000 Debug Tool Manual	This document teaches the user to install software Keil C51, and helped create a new project to be developed.

SN8F5814 Series Datasheet

8051-based Microcontroller

Corporate Headquarters
10F-1, No. 36, Taiyuan St.
Chupei City, Hsinchu, Taiwan
TEL: +886-3-5600888
FAX: +886-3-5600889

Taipei Sales Office
15F-2, No. 171, Songde Rd.
Taipei City, Taiwan
TEL: +886-2-27591980
FAX: +886-2-27598180
mkt@sonix.com.tw
sales@sonix.com.tw

Hong Kong Sales Office
Unit 2603, No. 11, Wo Shing St.
Fo Tan, Hong Kong
TEL: +852-2723-8086
FAX: +852-2723-9179
hk@sonix.com.tw

Shenzhen Contact Office
High Tech Industrial Park,
Shenzhen, China
TEL: +86-755-2671-9666
FAX: +86-755-2671-9786
mkt@sonix.com.tw
sales@sonix.com.tw

USA Office
TEL: +1-714-3309877
TEL: +1-949-4686539
tlightbody@earthlink.net

Japan Office
2F, 4 Chome-8-27 Kudanminami
Chiyoda-ku, Tokyo, Japan
TEL: +81-3-6272-6070
FAX: +81-3-6272-6165
jpsales@sonix.com.tw

FAE Support via email
8-bit Microcontroller Products:
sa1fae@sonix.com.tw
All Products: fae@sonix.com.tw

X-ON Electronics

Largest Supplier of Electrical and Electronic Components

Click to view similar products for 32-bit Microcontrollers - MCU ***category:***

Click to view products by SONIX ***manufacturer:***

Other Similar products are found below :

MCF51AC256AVFUE MCF51AC256BCFUE MCF51AC256BVFUE MB91F464AAPMC-GSE2 R5S726B0D216FP#V0 MB91F248PFV-GE1 MB91243PFV-GS-136E1 SAK-TC1782F-320F180HR BA TC364DP64F300WAAKXUMA1 R5F566NNDDFP#30 R5F566NNDDFC#30 R5F566NNDDBD#20 MC96F8216ADBN A96G181HDN A96G140KNN A96G174FDN A31G213CL2N A96G148KNN A96G174AEN AC33M3064TLBN-01 V3s T3 A40i-H V526 A83T R11 V851s A133 V833 F1C100S T3L T507 A33 A63 T113-i H616 V853 V533 V536-H A64-H V831 V3LP T113-S3 F1C200S F133-A R128-S2 ADUCM360BCPZ128-TR APT32S003F8PT AT32F435VMT7 AT32F435CGT7