CA51F4 Series MCU
User Guide

REV 2.4

IMPORTANT STATEMET: We reserve the right to make further clarifications regarding the reliability, functionality and design of all products listed below.We also reserve the right to make changes to all documentation for this product without notice.
Customers should ask our sales staff for the latest documentation when using this product. Hereby declare!

Table of Contents

1 Introduction

CA51F4 series is 8-bit MCU based on 1T 8051 cores and operates 10 times faster than traditional 8051 chips with definitely better performance. The Flash program memory embedded can be programmed for times and offers users with 18K FLASH which brings great convenience to software development. Not only traditional 8051 chip features, CA51F4 also includes 12 Bit ADC, Touch Key, 16-Bit PWM, UART, I²C, LCD, LVD and other functional modules. It can operate in three Power Save Modes (IDLE/STOP/LOW SPEED) in order to meet different power consumption needs. The LCD driver has a built-in voltage charge pump mode, which enables a boost function and constant LCD voltage output under different power supply conditions (including low voltage supply).With great functions configuration, CA51F4 is more superior in the products with LCD display and could be used in various fields such as remote control, thermostat and battery-powered products.

2 Basic Features

- **Core**
  - CPU: 1T 8051, with highest speed 10 times faster than traditional 8051
  - Compatible with 8051 instruction set, with double DPTR mode
- **Memory**
  - Flash: 18K byte, can be erased and overwritten for times
  - Flash could be divided into program storage and data storage. Data storage could be used to store data which need to be protected during power off and save EEPROM in the end of the day.
  - RAM:256 bytes internal RAM, 1024 bytes external RAM
- **Operating Voltage**
  - Operating Voltage: 1.8 - 5.5V
- **Clock System**
  - Internal Low Speed RC Oscillator: 131KHz
  - Internal High Speed RC Oscillator: 16MHz, with 1%(3.3V@25℃) precision.
  - External RTC Oscillator: 32.768KHz
- **RTC**
  - The internal RTC module can count hours, minutes, days and weeks. It can also be used as alarm clock
  - Supports microsecond/half second interrupt function
- **Interrupt System**
  - 15 effective interrupt source
  - Two levels for interrupt priority which also supports interrupt nesting
  - 10 external interrupt source. For each external interrupt, any of the signal pin could be configured as interrupt pin
- **Timer**
  - Three 16-bit general Timers: Timer 0, Timer 1, Timer 2

- **General Purpose Input Ouput（GPIO）**
  - Supports 46 GPIO at most and also supports push-pull, open-drain, pull-up, pull-down and high-impedance modes
  - When the sink current of 3 GPIO ports reaches 20mA@3.3V,30mA@5V(Vol=GND+0.6V), it can be used to control the backlight of LCD
  - When the sink current of 1 GPIO port reaches 400mA@5V, 300mA@3.3V(Vol=GND+1.2V), it can be used as the remote control carrier drive port
- **Touch Key**
  - Internal Touch Sensor Controller
  - Supports 16 touch channels at most
  - Touch to set the internal charging and internal reference which can effectively suppress the low-frequency interference of the power supply
  - Internal waterproof compensation mechanism
  - Excellent anti-jamming performance which conforms to EMC(CS) Standard
  - Support touch power saving mode
- **Analog/Digital Converter(ADC)**
  - Supports 8 channel 12-bit SAR ADC
  - Supports 3 Reference Voltage： VDD, Internal Reference Voltage, External Reference Voltage
  - When Internal Voltage is selected, VDD could be measured as well
- **PWM**
  - Supports 3 channel PWM, any periods or duty cycles are configurable in 16 bits
  - Support the function of directly outputting the internal clock and the special frequency of 38KHz for remote control
  - Supports PWM Interrupt
- **LCD Driver**
  - Support built-in voltage charge pump mode, charge pump voltage divider mode and resistance voltage divider mode, built-in voltage charge pump mode can realize the boost function
  - Supports 5com x 31segm、4com x 32 segm at most
  - Configurable Duty Cycle: 1/2、1/3、1/4、1/5 Duty
- **Low Voltage Detector(LVD)**
  - Configurable with four trigger voltages 2.0V、2.7V、3.7V and 4.4V
  - Low voltage reset/interrupt configurable
- **Reset mode**
  - Supports variable reset sources: Hard Reset, Soft Rest, Watch Dog Reset, LVD Reset, Power On/Down Reset
- **Watch Dog**
  - 27bit Watch Dog Timer, 16 bit precision configurable, with Watch Dog Reset and Interrupt configurable as well
- **Universal Serial Interface(UART)**
  - Supports 1 UART port

- Supports 1 byte receive buffer

- **I²C**
  - One I²C port embedded which supports Master-Slave mode and Standard/Fast/High Speed mode as well
- **Program Download and Simulation**
  - Supports ISP and IAP
  - Supports simulation online
- **Low Power Consumption**
  - For STOP Mode, current<3uA
  - For IDLE Mode, current<10uA
  - For Low Speed Mode, current<20uA
- **Package Type:** LQFP48/SOP28/SSOP28

3 Chip Model and Function Description

Table 3-1 Specific models and functional features of CA51F4 series

Models	Flash Storage[BYTE]	External Ram[BYTE]	Internal High Speed RC Oscillator [MHz]	Internal Low Speed RC Oscillator	External Low Soeed Crystal Oscillator [32.768KHz]	GPIO	UART	I²C	16 bit PWM channels	Touch key	12 bit ADC channels	LCD Drive [comx seg]	ISP	Simulation On Chip	Working Voltage[V]	Package Type
CA51F452L2	18K	1K	16	√	√	46	1	√	3	16	8	4X32 5X31	√	√	1.8-5.5	LQFP48
CA51F412L2	18K	1K	16	√	√	46	1	√	3	×	8	4X32 5X31	√	√	1.8-5.5	LQFP48
CA51F452S6	18K	1K	16	√	×	26	1	√	2	8	6	4X20 5X19	√	√	1.8-5.5	SOP28
CA51F412S6	18K	1K	16	√	×	26	1	√	2	×	6	4X20 5X19	√	√	1.8-5.5	SOP28
CA51F452P6	18K	1K	16	√	√	26	1	√	2	8	4	4X18 5X17	√	√	1.8-5.5	SSOP28
CA51F412P6	18K	1K	16	√	√	26	1	√	2	×	4	4X18 5X17	√	√	1.8-5.5	SSOP28

4 Block Diagram

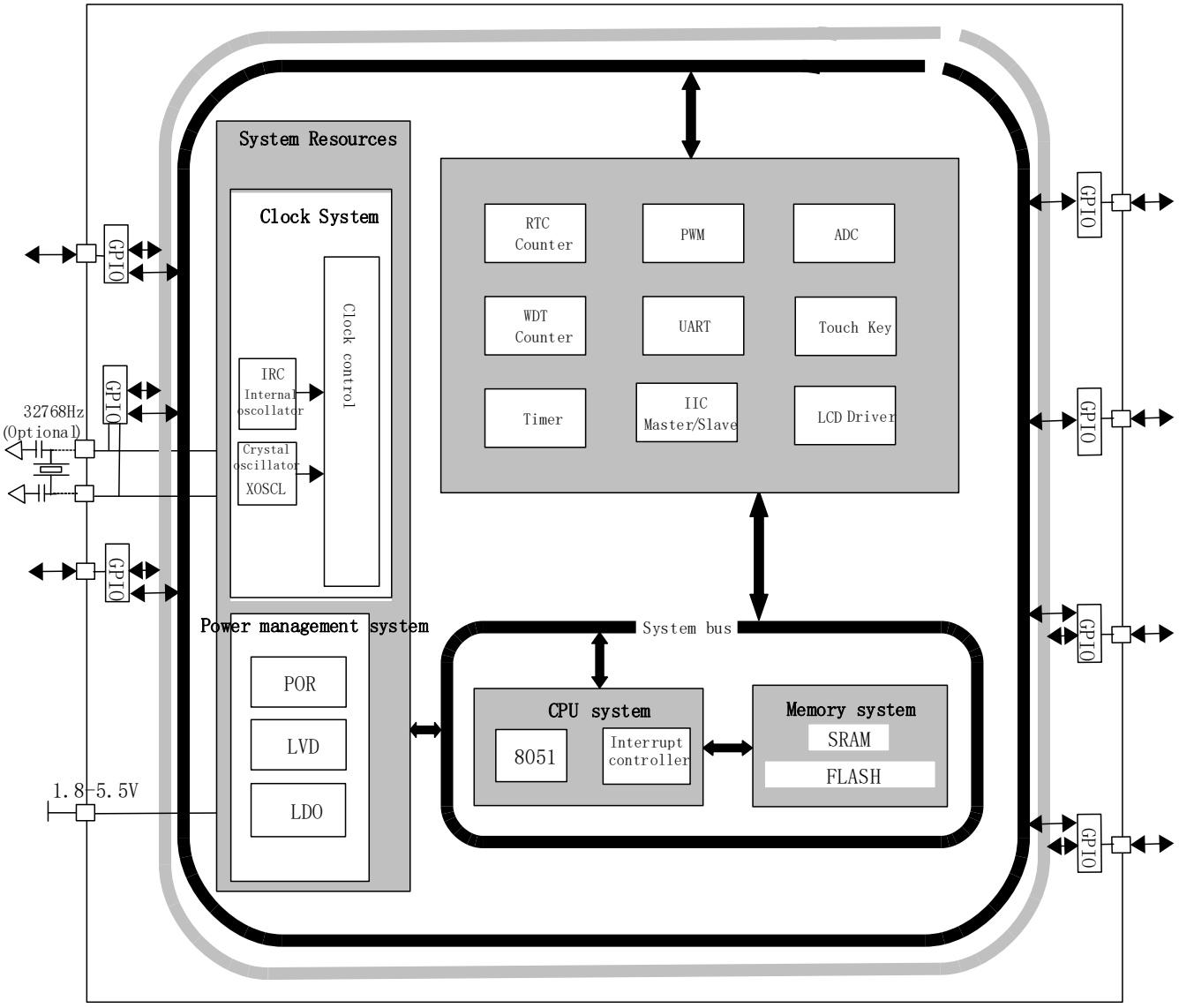


Figure 4-1-1 Chip Block Diagram

5 Pin Package and Description

5.1 Package Definition

Model: CA51F4XXL2

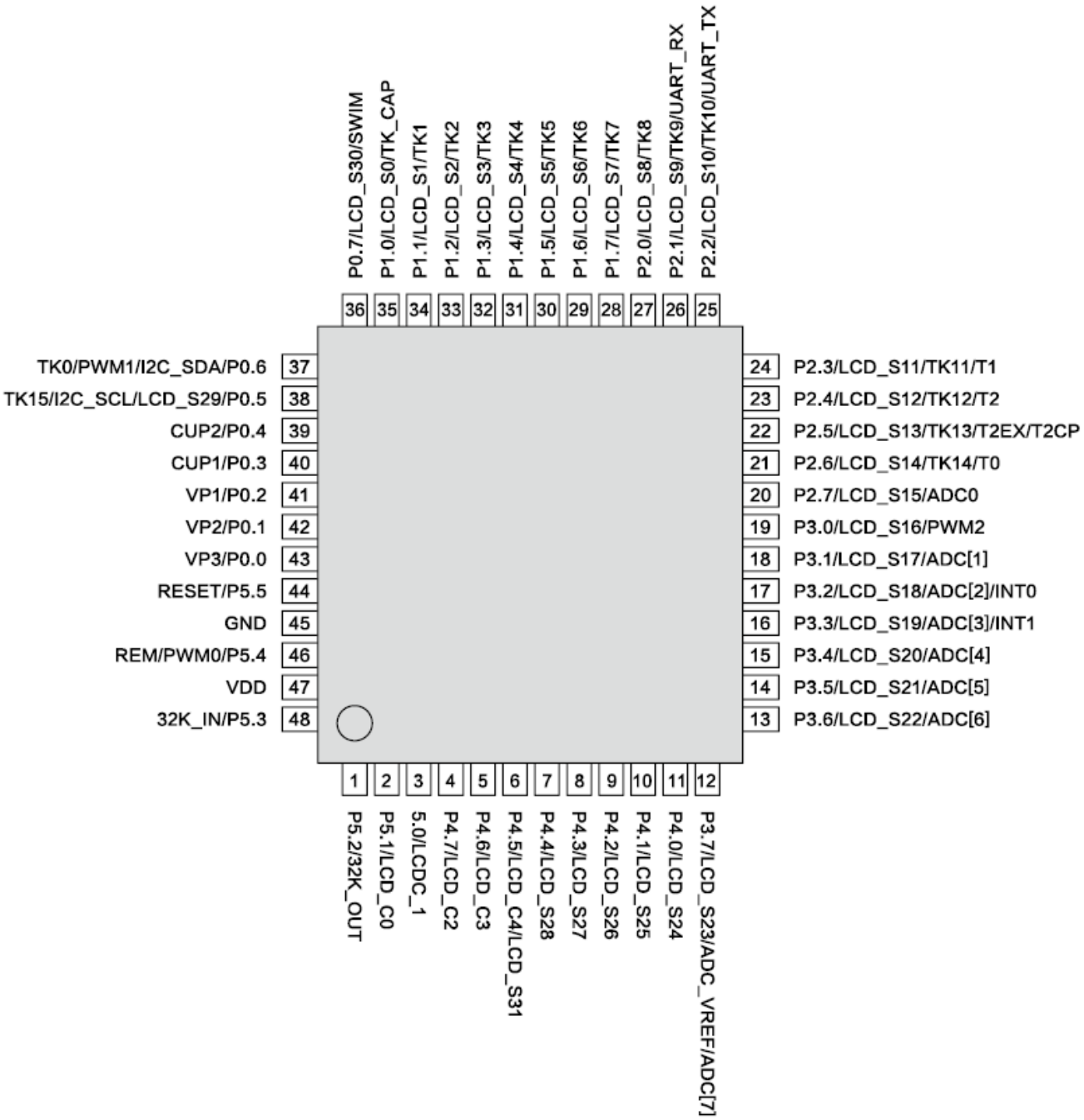


Figure 5-1-1 CA51F4XXL2(LQFP48) Package

Model: CA51F4XXS6

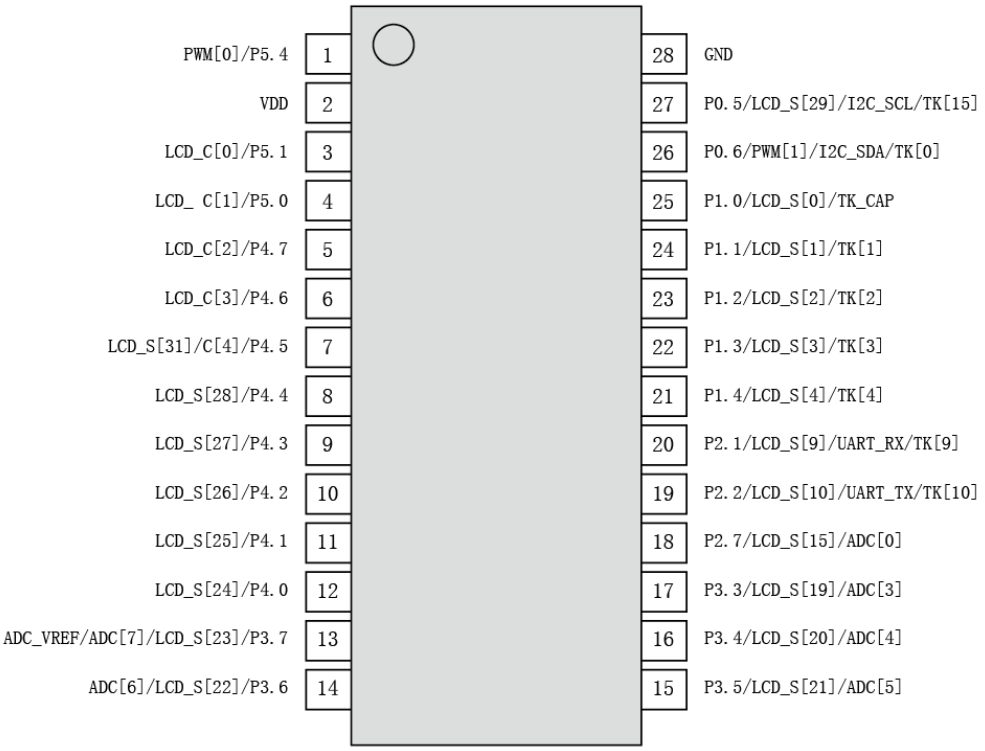


Figure 5-1-2 CA51F4XXS6(SOP28) Package

型号：CA51F4XXP6

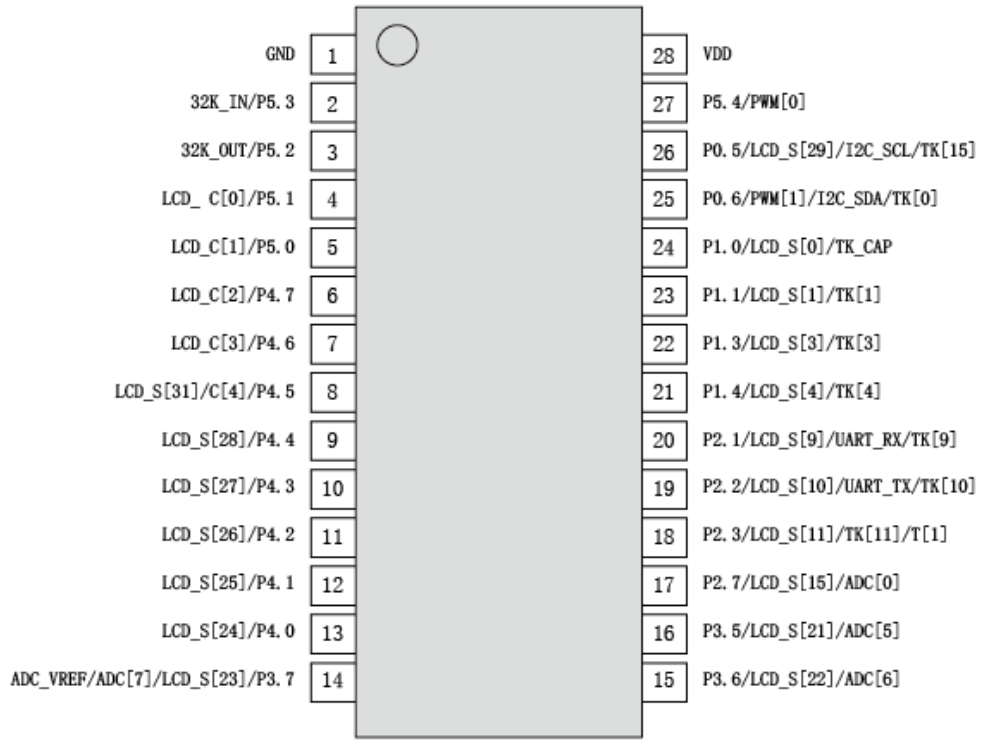


Figure 5-1-3 CA51F4XXP6(SSOP28) Package

5.2 Pin Description

Table 5-2-1 CA51F4 Pin Description

Pin Number			Pin Name	Pin Function	Default Function
LQFP48	SOP28 (S6)	SSOP28 (P6)			
1	-	3	P5.2/32K_O	General bi-directional I/O port 32K external clock output	General bi-directional I/O port
2	3	4	P5.1/LCD_C0	General bi-directional I/O port LCD driver output	General bi-directional I/O port
3	4	5	P5.0/LCD_C1	General bi-directional I/O port LCD driver output	General bi-directional I/O port
4	5	6	P4.7/LCD_C2	General bi-directional I/O port LCD driver output	General bi-directional I/O port
5	6	7	P4.6/LCD_C3	General bi-directional I/O port LCD driver output	General bi-directional I/O port
6	7	8	P4.5/LCD_C4/LCD_S31	General bi-directional I/O port LCD driver output	General bi-directional I/O port
7	8	9	P4.4/LCD_S28	General bi-directional I/O port LCD driver output	General bi-directional I/O port
8	9	10	P4.3/LCD_S27	General bi-directional I/O port LCD driver output	General bi-directional I/O port
9	10	11	P4.2/LCD_S26	General bi-directional I/O port LCD driver output	General bi-directional I/O port
10	11	12	P4.1/LCD_S25	General bi-directional I/O port LCD driver output	General bi-directional I/O port
11	12	13	P4.0/LCD_S24	General bi-directional I/O port LCD driver output	General bi-directional I/O port
12	13	14	P3.7/LCD_S23/ADC_VREF/ADC7	General bi-directional I/O port LCD driver output ADC external reference input	General bi-directional I/O port

				ADC channel input	
13	14	15	P3.6/LCD_S22/ADC6	General bi-directional I/O port LCD driver output ADC channel input	General bi-directional I/O port
14	15	16	P3.5/LCD_S21/ADC5	General bi-directional I/O port LCD driver output ADC channel input	General bi-directional I/O port
15	16	-	P3.4/LCD_S20/ADC4	General bi-directional I/O port LCD driver output ADC channel input	General bi-directional I/O port
16	17	-	P3.3/LCD_S19/ADC3/INT1	General bi-directional I/O port LCD driver output ADC channel input	General bi-directional I/O port
17	-	-	P3.2/LCD_S18/ADC2/INT0	General bi-directional I/O port LCD driver output ADC channel input	General bi-directional I/O port
18	-	-	P3.1/LCD_S17/ADC1	General bi-directional I/O port LCD driver output ADC channel input	General bi-directional I/O port
19	-	-	P3.0/LCD_S16/PWM2/CLK_IN	General bi-directional I/O port LCD driver output PWM2 channel output External standard clock input	General bi-directional I/O port
20	18	17	P2.7/LCD_S15/ADC0	General bi-directional I/O port LCD driver output ADC channel input	General bi-directional I/O port
21	-	-	P2.6/LCD_S14/T0/TK14	General bi-directional I/O port T0 port LCD driver output Touch key channel input	General bi-directional I/O port
22	-	-	P2.5/LCD_S13/T2EX/T2CP/TK13	General bi-directional I/O port T2EX/T2CP port LCD driver output Touch key channel input	General bi-directional I/O port

23	-	-	P2.4/LCD_S12/T2/TK12	General bi-directional I/O port T2 port LCD driver output Touch key channel input	General bi-directional I/O port
24	-	18	P2.3/LCD_S11/T1/TK11	General bi-directional I/O port T1 port LCD driver output Touch key channel input	General bi-directional I/O port
25	19	19	P2.2/UART_TX/LCD_S10/TK10	General bi-directional I/O port UART_TX transfer port LCD driver output Touch key channel input	General bi-directional I/O port
26	20	20	P2.1/UART_RX/LCD_S9/TK9	General bi-directional I/O port UART_TX transfer port LCD driver output Touch key channel input	General bi-directional I/O port
27	-	-	P2.0/LCD_S8/TK8	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
28	-	-	P1.7/LCD_S7/TK7	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
29	-	-	P1.6/LCD_S6/TK6	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
30	-	-	P1.5/LCD_S5/TK5	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
31	21	21	P1.4/LCD_S4/TK4	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
32	22	22	P1.3/LCD_S3/TK3	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port

33	23	-	P1.2/LCD_S2/TK2	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
34	24	23	P1.1/LCD_S1/TK1	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
35	25	24	P1.0/LCD_S0/TK_CAP	General bi-directional I/O port LCD driver output Touch key channel input	General bi-directional I/O port
36	-	-	P0.7/SWIM/ LCD_S30	General bi-directional I/O port SWIM transfer port LCD driver output	SWIM transfer port
37	26	25	P0.6/I2C_SDA/PWM1/TK0	General bi-directional I/O port I^2C data transfer port PWM1 output Touch key channel input	I^2C data transfer port
38	27	26	P0.5/I2C_SCL/TK15/ LCD_S29	General bi-directional I/O port I^2C clock transfer port LCD driver output Touch key channel input	I^2C clock transfer port
39	-	-	P0.4/CUP2	General bi-directional I/O port LCD CUP port	General bi-directional I/O port
40	-	-	P0.3/CUP1	General bi-directional I/O port LCD CUP port	General bi-directional I/O port
41	-	-	P0.2/VP1	General bi-directional I/O port LCD CUP port	General bi-directional I/O port
42	-	-	P0.1/VP2	General bi-directional I/O port LCD VP port	General bi-directional I/O port
43	-	-	P0.0/VP3	General bi-directional I/O port LCD VP port	General bi-directional I/O port
44	-	-	P5.5/RESET	General bi-directional I/O port Hardware reset input	Hardware reset port

45	28	1	GND	Ground	Ground
46	1	27	P5.4/PWM0/REM	General bi-directional I/O port PWM0 output REM output	General bi-directional I/O port
47	2	28	VDD	Chip power supply port	Chip power supply port
48	-	2	P5.3/32K_I	General bi-directional I/O port 32K external clock input	General bi-directional I/O port

Note：For signal pin's alternate function settings, please refer to Table 15-2-7 and Table 15-2-10

6 Central Processing Unit(CPU)

6.1 CPU Introduction

The core of CA51F4 Series is monocyclic 8051 CPU and make it fully compatible with original MCS-51 instruction set. A monocyclic 8051 CPU usually operates 10 times faster than standard 8051 one due to its pipeline structure.

The features of this CPU are:

- 1T 8051 CPU
- Compatible with 8051 instruction set and for more you may refer to instruction set in Appendix
- Double DPTR, so that the data could be moved quickly

6.2 Register Description

Program Counter(PC)

Program Counter(PC) is a 16-bit register without register address which is used to control the sequence of instructions.It is set to 0 after reset/power on and the machine will execute the program from zero address.

Accumulator(ACC)

Accumulator(ACC) is a special register and 'A' is used as its instruction mnemonic. It is often used to store the operand and result of logical/arithmetic computing.

General Register B

Register B cannot to be used without ACC in multiplying/dividing computing. Instruction MUL AB multiplies 8-bit unsigned number in ACC and B. The lower bytes(16 bit) and higher bytes(16 bit) of the computing result will be stored in A and B respectively. Furthermore, instruction DIV AB divides B by A, and the integer quotient will be stored in A with remainder stored in B. In addition, register B can also be used as general temporary storage register.

Stack Pointer(SP)

Stack Pointer(SP) is a 8 bit special register and indicates where the top of stack is in the internal RAM. It is initialized to

07H after a reset which makes stack actually starts from 08H. Since 08H~1FH belongs to working register group 1~3, if they are used in program development, SP is recommended to be set to 80H or even higher.

Data Pointer(DPTR)

Data pointer DPTR0/DPTR1 are two 16-bit special register with their higher stored in register DP0H/DP1H respectively and lower bytes stored in register DP0H/DP1H respectively. By setting DPS(PSW.1) either of them can be used. For each DPTR, it can be seen as one 16-bit register or two independent 8-bit registers DP0H/DP1H and DP0L/DP1L.

Program Status Word(PSW)

Program Status Word(PSW) is a register indicates the statues of the CPU. The status bit of it will change correspondingly when the CPU is doing arithmetic or logical operations.

Table 6-2-1 Accumulator ACC

E0H	7	6	5	4	3	2	1	0
ACC	ACC[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Table 6-2-2 General Register B

F0H	7	6	5	4	3	2	1	0
B	B[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Table 6-2-3 Stack Pointer SP

81H	7	6	5	4	3	2	1	0
SP	SP[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	1	1	1

Table 6-2-4 Data Pointer DP0L

82H	7	6	5	4	3	2	1	0
DP0L	DP0L[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Table 6-2-5 Data Pointer DP0H

83H	7	6	5	4	3	2	1	0
DP0H	DP0H[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Table 6-2-6 Data Pointer DP1L

84H	7	6	5	4	3	2	1	0
DP1L	DP1L[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Table 6-2-7 Data Pointer DP1H

85H	7	6	5	4	3	2	1	0
DP1H	DP1H[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Table 6-2-8 Program Status Word PSW

D0H	7	6	5	4	3	2	1	0
PSW	CY	AC	F0	RS[1:0]		OV	DPS	P
R/W	R/W	R/W	R/W	R/W		R/W	R	R
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	CY	Carry flag 0: There is no carry or borrow happened in arithmetic/logical operation 1: There is carry or borrow happened in arithmetic/logical operation
6	AC	Auxiliary Carry Flag 0: There is no auxiliary carry or borrow happened in arithmetic/logical operation 1: There is auxiliary carry or borrow happened in arithmetic/logical operation
5	F0	F0 flag It is defined by the user
4~3	RS	R0~R7 registers' page selection 00: page 0(mapping to 00H-07H) 01: page 1(mapping to 08H-0FH) 10: page 2(mapping to 10H-17H) 11: page 3(mapping to 18H-1FH)
2	OV	Overflow flag 0: no overflow 1: overflow happened
1	DPS	DPTR selector, 0 for DPTR0, 1 for DPTR1
0	P	Parity flag 0: the number of 1 in ACC is even 1: the number of 1 in ACC in odd

Table 6-2-9 Register SPMAX

8100H	7	6	5	4	3	2	1	0
SPMAX	SPMAX[7:0]							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0
Bit number	Bit symbol	Description						
7~0	SPMAX	SPMAX is used to record the maximum value of SP. Users can check this register using software to decide whether there is a risk that the stack may overflow						

7 Memory Architecture

7.1 Random Access Memory (RAM)

CA51F4 series offers both internal RAM(256 bytes) and external RAM(1024 bytes) for the users and the corresponding address are shown as follows:

- Lower 128 bytes of the internal RAM(address: 00H ~ 7FH), supports both direct addressing and indirect addressing
- Higher 128 bytes of the internal RAM(address: 80H ~ FFH), only supports indirect addressing.
- 1024 bytes external RAM(address: 0000H ~ 03FFH), supports indirect addressing by using MOVX.

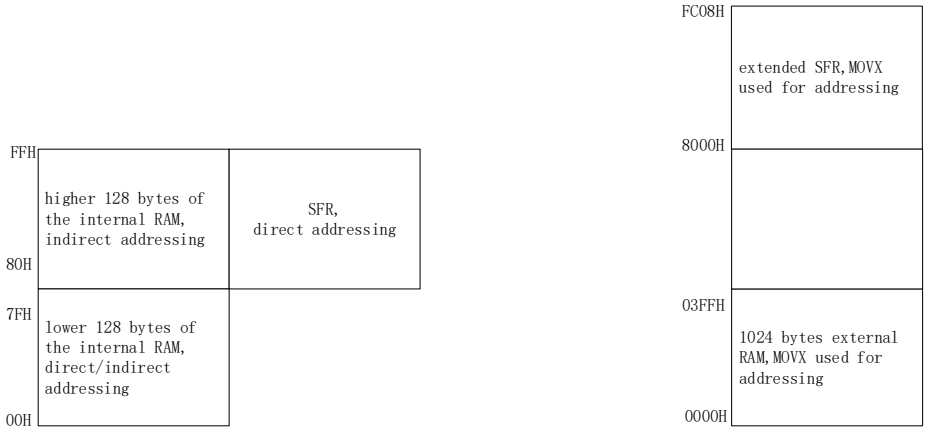


Figure 7-1-1 RAM Architecture

7.2 Special Function Register(SFR)

The SFR architecture of CA51F4 series is compatible with traditional 8051 chip. SFR and the higher 128 bytes of the internal RAM both use the address 80H ~ FFH that only supports direct addressing, SFR mapping is shown in Table 7-2-1.

Table 7-2-1 Special Function Register Mapping Table

	Bit addressable	Not bit addressable						
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
F8H	EXIP	EPIE	EPIF	EPCON	IDLSTL	IDLSTH	STPSTL	STPSTH
F0H	B	RTCON	RTCS	RTCM	RTCH	RTCDL	RTCDH	INDEX
E8H	EXIE	RTCSS	RTAS	RTAM	RTAH	RTMSS	RTCIF	LVDCON
E0H	ACC	LCCON	LCCFG	LCDAT	LCDIVL	LCDIVH	-	-
D8H	P5	-	PWMEN	-	PWMCMAX	PWMCON	PWMCKD	PWMDIVL
D0H	PSW	PWMDIVH	PWMDUTL	PWMDUTH	PWMIF	-	-	-

C8H	T2CON	T2MOD	T2CL	T2CH	TL2	TH2	TKMSL	TKMSH
C0H	P4	TKCON	TKCFG	TKMTS	TKCHS	ATKSL	ATKSH	TKIF
B8H	IP	ADCON	ADCFGL	ADCFGH	ADCDL	ADCDH	-	-
B0H	P3	I2CCON	I2CADR	I2CADM	I2CCCR	I2CDAT	I2CSTA	I2CFLG
A8H	IE	-	WDCON	WDFLG	WDVTHL	WDVTHH	-	-
A0H	P2	-	-	-	-	-	-	-
98H	-	-	S1CON	S1BUF	S1RELL	S1RELH	-	-
90H	P1	-	-	-	-	-	-	-
88H	TCON	TMOD	TL0	TL1	TH0	TH1	IT1CON	IT0CON
80H	P0	SP	DP0L	DP0H	DP1L	DP1H	PWCON	PCON

Due to limited SFR address space, CA51F4 series also added extended special function register in external RAM address space. The mapping is shown as follows.

Table 7-2-2 Extended Special Function Register Mapping Table

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
8000H	P00F	P01F	P02F	P03F	P04F	P05F	P06F	P07F
8008H	P10F	P11F	P12F	P13F	P14F	P15F	P16F	P17F
8010H	P20F	P21F	P22F	P23F	P24F	P25F	P26F	P27F
8018H	P30F	P31F	P32F	P33F	P34F	P35F	P36F	P37F
8020H	P40F	P41F	P42F	P43F	P44F	P45F	P46F	P47F
8028H	P50F	P51F	P52F	P53F	P54F	P55F	-	-
8078H	-	-	-	-	-	-	-	RCSTA
8080H	CKCON	CKSEL	CKDIV	IHCFG	-	ILCFHL	ILCFGH	-
8088H	ADCALL	ADCALH	-	-	-	-	-	ADOPC
8090H	TKMAXF	TKMINF	ATKNL	ATKNH	-	-	-	-
8098H	-	PWMHS	-	-	PWMSBC	PWMBD	-	-
80B0H	SWICON	SWIDAT	SWISTA	SWIOVT	-	-	-	-
80F8H	TSCMD	TSSTA	STPCL	STPCH				
8100H	SPMAX	-	-	TKPWC	-	-	TLEN	TLDAT
8108H	TLCON	TLFLG	TLCKS	TLCNTKL	TLCNTKH	TLCNTLL	TLCNTLH	TLDIV
8110H	-	-	-	-	-	-	LCPMP	LCCAD
8118H	UDCKS	-	-	-	-	RMCTL	FTCTL	TPCTL
8120H	P00C	P01C	P02C	P03C	P04C	P05C	P06C	P07C
8128H	P10C	P11C	P12C	P13C	P14C	P15C	P16C	P17C
8130H	P20C	P21C	P22C	P23C	P24C	P25C	P26C	P27C
8138H	P30C	P31C	P32C	P33C	P34C	P35C	P36C	P37C
8140H	P40C	P41C	P42C	P43C	P44C	P45C	P46C	P47C
8148H	P50C	P51C	P52C	P53C	P54C	P55C	-	-
FC00H	MECON	FSCMD	FSDAT	LOCK	PADRD	PTSL	PTSH	REPSET

7.3 Flash Memory

7.3.1 Function Introduction

Flash memory is 18K byte and it can be erased and overwritten repeatedly. Flash is also controlled by a group of special registers, therefore users may use these registers to erase/overwrite/set write protect to the Flash and so on.

7.3.2 Flash Architecture

- Flash consists of several sectors which are the smallest units for erasure. Each sector is 128 bytes.
- Flash can be divided into DATA area and PROGRAM area and the division unit is 128 byte. PROGRAM area is used to store use's program and DATA area is used to store data that needs to be protected during power off period.

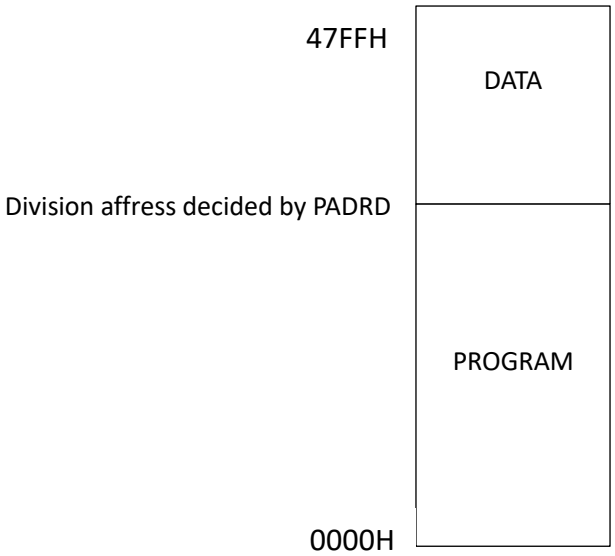


Figure 7-3-1 18K Flash Memory Structure

7.3.3 Flash Description

Table 7-3-3-1 Register MECON

FC00H	7	6	5	4	3	2	1	0
MECON	-	DPSTB	-	-	-	-	-	BOOT
R/W	-	R/W	-	-	-	-	-	R/W
Initial Value	-	0	-	-	-	-	-	0

Bit number	Bit symbol	Description
7	-	-
6	DPSTB	Flash SLEEP mode control in IDLE/STOP mode 0: Flash in NORMAL mode while IDLE/STOP 1: Flash in SLEEP mode while IDLE/STOP *Note: If DPSTB=1, when the chip enters IDLE/STOP mode, the Flash will enter SLEEP mode simultaneously and the power consumption of the Flash in SLEEP mode is 50nA. When the chip exits IDLE/STOP mode, Flash exits SLEEP mode as well.*
5~1	-	
0	BOOT	Programs start area control after soft reset 0: Program starts from FLASH after soft reset 1: Program starts from XRAM after soft reset

Table 7-3-3-2 Register FSCMD

FC01H	7	6	5	4	3	2	1	0
FSCMD	IFEN	-	-	-	-	CMD[2:0]		
R/W	R/W	-	-	-	-	R/W		
Initial Value	0	-	-	-	-	0	0	0

Bit number	Bit symbol	Description
7	IFEN	Information Area Enable, 1 indicates enable.
6~3	-	-
2~0	CMD	Command register 000: No operations 100: Erase the whole Flash 001: Read Flash DATA area 010: Write Flash DATA area 011: Erase sectors of the Flash DATA area 101: Read Flash PROGRAM area 110: Write Flash PROGRAM area 111: Erase sectors of the Flash PROGRAM area *Note:* *1. CMD will be cleared automatically after erasure command executed* *2. CMD remains unchanged after R/W commands and the R/W operations will be done by reading/writing FSDAT*

Table 7-3-3-3 Register FSDAT

FC02H	7	6	5	4	3	2	1	0
FSDAT	FSDAT[7:0]							
R/W	R/W							

Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7~0	FSDAT	Flash data register

Table 7-3-3-4 Register LOCK

FC03H	7	6	5	4	3	2	1	0
LOCK								
R		REPE			FLKF	PLKF	DLKF	ILKF
W	LOCK[7:0]							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
Write		
7~0	LOCK	28H: Unlock Flash programmable area 29H: Unlock Flash PROGRAM area 2AH: Unlock Flash DATA area AAH: Lock Flash, R/W forbidden
Read		
7、5~4	-	-
6	REPE	Information area unlocked flag and protected, write 8'h56 first and then 8'hcb
3	FLKF	Programmable area unlocked flag, 1 indicates unlocked
2	PLKF	PROGRAM area unlocked flag, 1 indicates unlocked
1	DLKF	DATA area unlocked flag, 1 indicates unlocked
0	-	-

Table 7-3-3-5 Register PADRD

FC04H	7	6	5	4	3	2	1	0
PADRD	PADRD[7:0]							
R/W	R/W							
Initial Value	1	0	0	1	0	0	0	0

Bit number	Bit symbol	Description
7~0	PADRD	PROGRAM and DATA area division configuration register The unit for division is 128 bytes and when PADRD>0: The address space for PROGRAM area: 0 ~ (PADRD×128 - 1), The address space for DATA area: (PADRD×128) ~ 47FFH . *Note：* *1. PADRD=0 indicates the whole Flash is DATA area*

		2. The maximum value for PADRD is 90H. PADRD can not be set to any values greater than the maximum

Table 7-3-3-6 Register PTS

FC05H	7	6	5	4	3	2	1	0
PTSL	PTS[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
FC06H	7	6	5	4	3	2	1	0
PTSH	-	PTS[14:8]						
R/W	-	R/W						
Initial Value	-	-	-	0	0	0	0	0

Bit number	Bit symbol	Description
15	-	-
14~0	PTS	target address pointer register

7.3.4 Flash Control Example

◆ **Divide Flash into DATA area and PROGRAM area**

For instance, if the user wants to divide a 18K Flash (128 byte DATA area and the remains for PROGRAM area) and the program may like this:

```
---------------------------------------------------------------------------------
PADRD = 0x8F; //The address for PROGRAM area will be 0~0x477F while the address for DATA area will be 0x4780~0x47FF
---------------------------------------------------------------------------------
```

Note: *This makes the physical address of the DATA area in FLASH 0x4780~0x47FF while the logical address is 0x0000~0x007F. The logical address is used for DATA area's R/W*

◆ **Sector erasure of DATA area**

Sector n of DATA area needs to be erased, for example, the program may as follows:

```
---------------------------------------------------------------------------------
FSCMD = 0;          //set CMD=0
LOCK = 0x2A;        //unlock DATA area
PTSH = (unsigned char)((n*0x80)>>8); //set the high bytes of the sector's address
PTSL = (unsigned char)(n*0x80);          //set the low bytes of the sector's address

FSCMD = 3;    //set clear
LOCK = 0xAA;        //lock FLASH
```

Note: sector number n=0、1、2…….

◆ **Write data into DATA area**

For instance, write data 0xAA to DATA area of which address is n~(n+100) and the program will be:

```
---------------------------------------------------------------------------------
unsigned char i;
FSCMD = 0;          //set CMD=0
LOCK = 0x2A;        //unlock DATA area
PTSH = (unsigned char)(n>>8); //set the high 8 bits of data's original address
PTSL = (unsigned char)n;          //set the low8 bits of data's original address
FSCMD = 2;    //set WRITE command
for(i=0;i<100;i++)
{
      FSDAT = 0xAA;     //write data continuously
}
FSCMD = 0;
LOCK = 0xAA;        //lock FLASH
---------------------------------------------------------------------------------
```

Note: 1. When data is written continuously, only original address has be set. PTS will increase automatically after writing FSDAT each time.

2. For DATA area R/W, only the logical address of the DATA area which starts from 0 needs to be set, instead of the physical address.

◆ **Read data from DATA area**

For instance, the pointer pBuf reads data from DATA area of which address is n~(n+100) and the program will be:

```
---------------------------------------------------------------------------------
unsigned char i,  *pBuf;
FSCMD = 0;          //set CMD=0
LOCK = 0x2A;        //unlock DATA area
PTSH = (unsigned char)(n>>8); //set the high 8 bits of data's original address
PTSL = (unsigned char)n;          //set the low 8 bits of data's original address
FSCMD = 1;    // set READ command
for(i=0;i<100;i++)
{
      *pBuf++ = FSDAT ;// write data continuously
}
FSCMD = 0;
LOCK = 0xAA;        // lock FLASH
---------------------------------------------------------------------------------
```

Note: When data is read continuously, only original address has be set. PTS will increase automatically after writing FSDAT each time.

7.4 External RAM Mapped to Program Area

The 1024 external RAM can be mapped as PROGRAM area as well and the mapping address is 4800H~4BFFH with the figure 7-4-1 below shows the mapping. Users may download the program to external RAM. When program is running, it jump to mapping program area to execute. Similarly, users can set BOOT(please refer to register MECON) to 1, and then soft reset. The program starts from external RAM (the mapping address is 0000H~03FFH). Mapping program area offers convenience for IAP and so on.

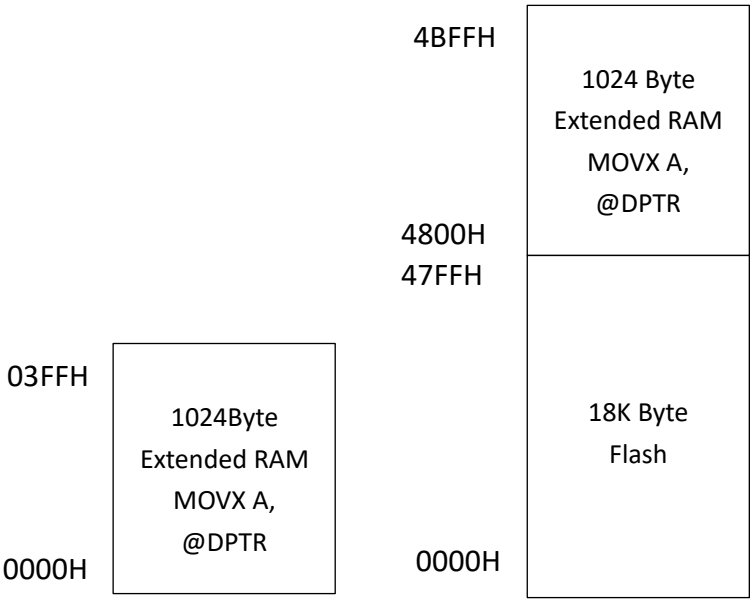


Figure 7-4-1 XRAM Address Mapping

8 Interrupt System

8.1 Function Introduction

CA51F4 series include a enhanced interrupt control system with 14 interrupt entries. For each interrupt entry, there are several interrupt sources with 2 level interrupt priorities for each source. Each interrupt source has its independent interrupt vector, priority setting, interrupt enable control and interrupt flag. CPU enters corresponding Interrupt Service Routine after responding to the interrupt. It will then returns to the former status after receiving RETI. If there are multiple valid sources requesting interrupts, CPU will respond sequentially according to the interrupt priority set before. If the sources share the same priority, CPU will respond according to their natural priority (from the smallest address to largest address of the interrupt entries).

8.2 Interrupt Logic

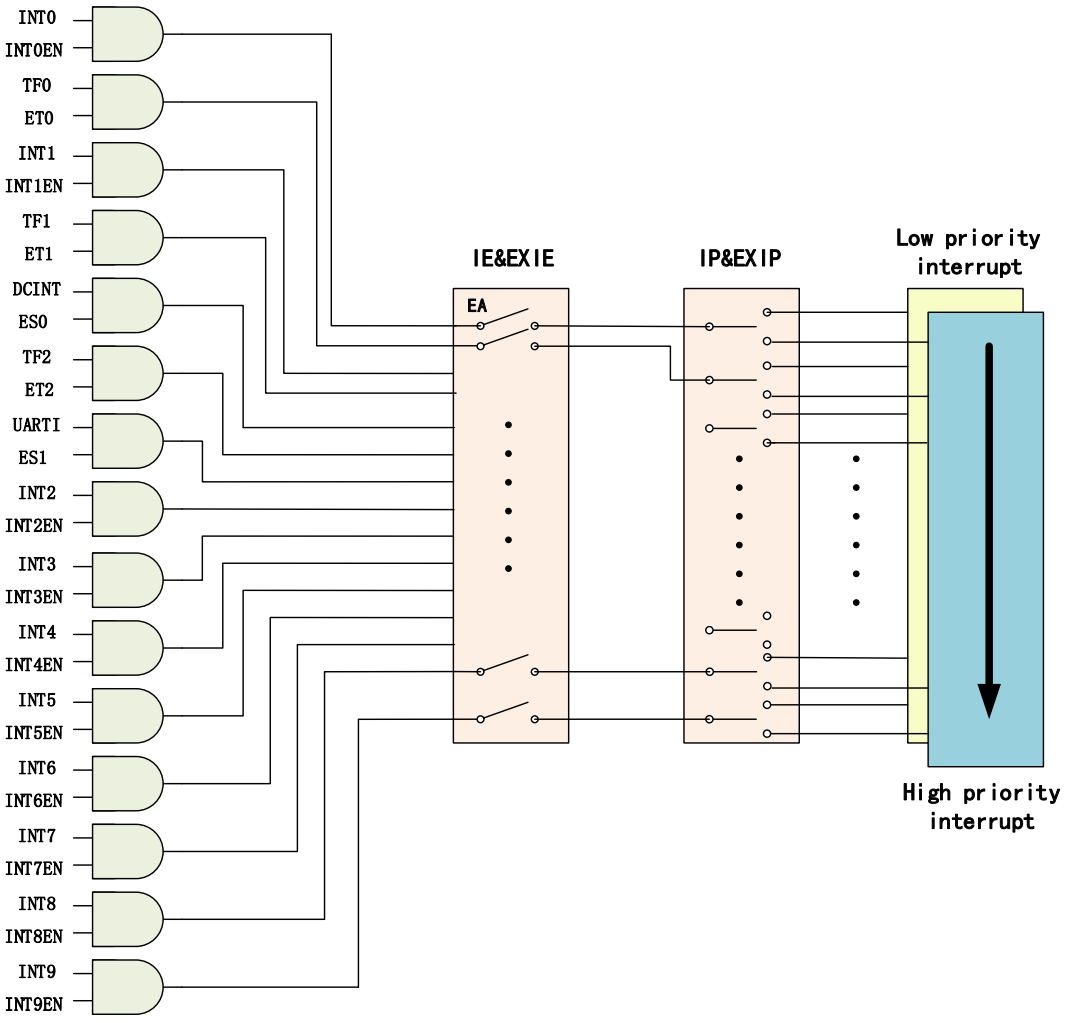


Figure 8-2-1 Interrupt Logic

8.3 Interrupt Vector Table

Table 8-3-1 Interrupt Vector Table

Interrupt	Interrupt source	Vector	Default Priority
INT0	INT0	03H	0
TF0	Timer 0	0BH	1
INT1	INT1	13H	2
TF1	Timer 1	1BH	3
TF2	Timer 2	2BH	5
UART1	UART1	33H	6
INT2	ADC/External Interrupt 2	3BH	7
INT3	UART2/TK/TL/External Interrupt 3	43H	8
INT4	LVD/External Interrupt 4	4BH	9
INT5	External Interrupt 5	53H	10
INT6	I2C/SWI/External Interrupt 6	5BH	11
INT7	WDT/External Interrupt 7	63H	12
INT8	RTC/External Interrupt 8	6BH	13
INT9	PWM/External Interrupt 9	73H	14

8.4 Interrupt Control Register

Table 8-4-1 Register IE

A8H	7	6	5	4	3	2	1	0
IE	EA	ES1	ET2	-	ET1	EX1	ET0	EX0
R/W	R/W	R/W	R/W	-	R/W	R/W	R/W	R/W
Initial Value	0	0	0	-	0	0	0	0

Bit number	Bit symbol	Description
7	EA	Global Interrupt enable control 0: disable Global Interrupt 1: enable Global Interrupt
6	ES1	UART1 Interrupt enable control 0: disable UART1 Interrupt 1: enable UART1 Interrupt
5	ET2	Timer 2 Interrupt enable control 0: disable Timer 2 Interrupt 1: enable Timer 2 Interrupt
4	-	-

3	ET1	Timer 1 Interrupt enable control 0: disable Timer 1 Interrupt 1: enable Timer 1 Interrupt
2	EX1	External Interrupt 1 enable control 0: disable External Interrupt 1 1: enable External Interrupt 1
1	ET0	Timer 0 Interrupt enable control
0	EX0	External Interrupt 0 enable control 0: disable External Interrupt 0 1: enable External Interrupt 0

Table 8-4-2 Register EXIE

E8H	7	6	5	4	3	2	1	0
EXIE	INT9EN	INT8EN	INT7EN	INT6EN	INT5EN	INT4EN	INT3EN	INT2EN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	INT9EN	Interrupt 9 enable control(Interrupt 9 is used for PWM/External Interrupt 9) 0: Disable 1: Enable
6	INT8EN	Interrupt 8 enable control(Interrupt 8 is used for RTC /External Interrupt 8) 0: Disable 1: Enable
5	INT7EN	Interrupt 7 enable control(Interrupt 7 is used for WDT/External Interrupt 7) 0: Disable 1: Enable
4	INT6EN	Interrupt 6 enable control(Interrupt 6 is used for I2C/SWI/External Interrupt 6) 0: Disable 1: Enable
3	INT5EN	Interrupt 5 enable control(Interrupt 5 is used for External Interrupt 5) 0: Disable 1: Enable
2	INT4EN	Interrupt 4 enable control(Interrupt 4 is used for LVD /External Interrupt 4) 0: Disable 1: Enable
1	INT3EN	Interrupt 3 enable control(Interrupt 3 is used for TK/External Interrupt 3) 0: Disable 1: Enable
0	INT2EN	Interrupt 2 enable control(Interrupt 2 is used for ADC/External Interrupt 2) 0: Disable 1: Enable

Note: The enable controls of EXIE corresponds to Interrupt Vector which means the enable control for each interrupt source has to be set as well. For example, if External Interrupt 2 needs to be enabled, both INT2EN and EPIE2(External Interrupt 2 enable control) need to be set to 1.

Table 8-4-3 Register IP

B8H	7	6	5	4	3	2	1	0
IP	-	PS1	PT2	PS0	PT1	PX1	PT0	PX0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	-	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	-	-
6	PS1	UART 1 priority control 0: low priority 1: high priority
5	PT2	Timer 2 priority control 0: low priority 1: high priority
4	-	-
3	PT1	Timer 1 priority control 0: low priority 1: high priority
2	PX1	External Interrupt 1 priority control 0: low priority 1: high priority
1	PT0	Timer 0 priority control 0: low priority 1: high priority
0	PX0	External Interrupt 0 priority control 0: low priority 1: high priority

Table 8-4-4 Register EXIP

F8H	7	6	5	4	3	2	1	0
EXIP	PX9	PX8	PX7	PX6	PX5	PX4	PX3	PX2
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	PX9	Interrupt INT9 priority control

		0: low priority 1: high priority
6	PX8	Interrupt INT8 priority control 0: low priority 1: high priority
5	PX7	Interrupt INT7 priority control 0: low priority 1: high priority
4	PX6	Interrupt INT6 priority control 0: low priority 1: high priority
3	PX5	Interrupt INT5 priority control 0: low priority 1: high priority
2	PX4	Interrupt INT4 priority control 0: low priority 1: high priority
1	PX3	Interrupt INT3 priority control 0: low priority 1: high priority
0	PX2	Interrupt INT2 priority control 0: low priority 1: high priority

8.5 External Interrupt

8.5.1 External Interrupt Introduction

For INT0 and INT1, arbitrary input ports can be selected as interrupt sources. The system also includes 8 extended Interrupt Entries INT2~INT9 as External Interrupt. For each interrupt entry, any input ports can be selected as interrupt source as well. Either rising or falling edge trigger can be selected independently for each Extended External Interrupt. The External Interrupt can also be waken up in STOP mode. The status register for INT2~INT9 External Interrupt is EPIF and the corresponding configuration register for INT2~INT9 which are EPCON0~ EPCON7 also can be visited by setting the index of register (INDEX=0~7).

Note: INT0 and INT1 can be triggered by rising edge or falling edge and the selection bits are IT0 and IT1 respectively. Please refer to the Register TCON for details.

8.5.2 External Interrupt Register

Table 8-5-2-1 Register IT0CON

8FH	7	6	5	4	3	2	1	0
IT0CON	-	-	IT0PS[4:0]					
R/W	-	-	R/W					
Initial Value	-	-	0	1	1	0	1	0

Bit number	Bit symbol	Description
7~5	-	-
4~0	IT0PS[4:0]	INT0 Interrupt pin selection The table for pin numbers and corresponding pins please refer to Table 8-5-2-6

Table 8-5-2-2 Register IT1CON

8EH	7	6	5	4	3	2	1	0
IT1CON	-	-	IT1PS[4:0]					
R/W	-	-	R/W					
Initial Value	-	-	0	1	1	0	1	1

Bit number	Bit symbol	Description
7~5	-	-
4~0	IT1PS[4:0]	INT1 Interrupt pin selection The table for pin numbers and corresponding pins please refer to Table 8-5-2-6

Table 8-5-2-3 Register EPIE

F9H	7	6	5	4	3	2	1	0
EPIE	EPIE9	EPIE8	EPIE7	EPIE6	EPIE5	EPIE4	EPIE3	EPIE2
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	EPIE9	External Interrupt 9 enable control
6	EPIE8	External Interrupt 8 enable control
5	EPIE7	External Interrupt 7 enable control
4	EPIE6	External Interrupt 6 enable control
3	EPIE5	External Interrupt 5 enable control
2	EPIE4	External Interrupt 4 enable control
1	EPIE3	External Interrupt 3 enable control
0	EPIE2	External Interrupt 2 enable control

Table 8-5-2-4 Register EPIF

FAH	7	6	5	4	3	2	1	0
EPIF	EPIF9	EPIF8	EPIF7	EPIF6	EPIF5	EPIF4	EPIF3	EPIF2
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	EPIF9	External Interrupt 9 Interrupt Flag, cleared when 1 is written to it
6	EPIF8	External Interrupt 8 Interrupt Flag, cleared when 1 is written to it
5	EPIF7	External Interrupt 7 Interrupt Flag, cleared when 1 is written to it
4	EPIF6	External Interrupt 6 Interrupt Flag, cleared when 1 is written to it
3	EPIF5	External Interrupt 5 Interrupt Flag, cleared when 1 is written to it
2	EPIF4	External Interrupt 4 Interrupt Flag, cleared when 1 is written to it
1	EPIF3	External Interrupt 3 Interrupt Flag, cleared when 1 is written to it
0	EPIF2	External Interrupt 2 Interrupt Flag, cleared when 1 is written to it

Table 8-5-2-5 Register EPCON

FBH	7	6	5	4	3	2	1	0
EPCON	EPPL	-	EPPS[5:0]					
R/W	R/W	-	R/W					
Initial Value	0	-	0	0	0	0	0	0

Note：EPCON is a register with index, INDEX=0~7 indicates EPCON0~EPCON7 respectively

Bit number	Bit symbol	Description
7	EPPL	External Interrupt Trigger Edge Selection 0: Rising edge 1: Falling edge
6	-	-
5~0	EPPS[4:0]	Interrupt Pin selection The table for pin numbers and corresponding pins please refer to Table 8-5-2-6

Table 8-5-2-6 Index for Interrupt Pin

Pin name	Number	Pin name	Number
P00	0	P30	24
P01	1	P31	25
P02	2	P32	26
P03	3	P33	27
P04	4	P34	28
P05	5	P35	29
P06	6	P36	30

P07	7	P37	31
P10	8	P40	32
P11	9	P41	33
P12	10	P42	34
P13	11	P43	35
P14	12	P44	36
P15	13	P45	37
P16	14	P46	38
P17	15	P47	39
P20	16	P50	40
P21	17	P51	41
P22	18	P52	42
P23	19	P53	43
P24	20	P54	44
P25	21	P55	45
P26	22		
P27	23		

8.5.3 External Interrupt Control Examples

◆ **External Interrupt 0/1 control example**

For instance, set P10 as the input pin for External Interrupt 0 and enable External Interrupt 0, the program will be:

```
---------------------------------------------------------------------------------
void INT0_init(void)
{
    P10F = 1;           //set P10 as input pin
    IT0CON = 8;         //set P10 as INT0 interrupt pin
    EX0 = 1;            //enable INT0 interrupt
    IE0 = 1;            //enable External Interrupt 0
    IT0 = 1;            //set falling edge trigger
    PX0 = 1;            //set INT0 with high priority
    EA = 1;             //enable Global Interrupt
}
void INT0_ISR (void) interrupt 0
{
    //External Interrupt 0 Interrupt Service Routine
}
---------------------------------------------------------------------------------
```

For instance, set P10 as the input pin for External Interrupt 1 and enable External Interrupt 1, the program will be:

```
---------------------------------------------------------------------------------
void INT1_init(void)
{
    P10F = 1;           //set P10 as input pin
    IT1CON = 8;         //set P10 as INT1 interrupt pin
    EX1 = 1;            //enable INT1 interrupt
    IE1 = 1;            //enable External Interrupt 1
    IT1 = 1;            //set falling edge trigger
    PX1 = 1;            //set INT1 with high priority
    EA = 1;             //enable Global Interrupt

}
void INT1_ISR (void) interrupt 2
{
    // External Interrupt 1 Interrupt Service Routine
}
---------------------------------------------------------------------------------
```

◆ **External Interrupt 2~9 control example**

Taking External Interrupt 2 for example, if P10 is set as the input pin for External Interrupt 2 and External Interrupt 2 is enable, the program may like this:

```
--------------------------------------------------------------------------------
void INT2_init(void)
{
    P10F = 1;                   // set P10 as input pin
    INDEX = 0;                  //INDEX is a register with index and set INDEX = 0 correspond to INT2
    EPCON = (1<<7) | 8;         //set P10 as INT2 external interrupt pin and falling edge trigger
    INT2EN = 1;                 //enable External Interrupt 2
    EPIE |= 0x01;               //enable INT2 interrupt
    EA = 1;                     //enable Global Interrupt
}
void INT2_ISR (void) interrupt 7
{
    if(EPIF & 0x01)             //judge the Interrupt Flag for External Interrupt 2
    {
        EPIF = 0x01;            //write 1 to the Interrupt Flag to clear it
        //External Interrupt 2 Interrupt Service Routine
        ......
    }
}
--------------------------------------------------------------------------------
```

9 Clock System

9.1 Clock System Introduction

CA51F4 Series has several clock sources as follows:

- 16MHz Internal RC Oscillator
- 131KHz Internal RC Oscillator
- Supports 32.768 KHz external Crystal Resonator

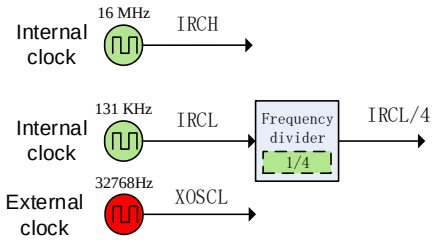


Figure 9-1-1 Clock Sources

Users can control the clock sources independently. They can disable or enable any of the clock sources in order to manage the power consumption flexibly.All the clock sources can be set as system alarm clock and assigned to various peripherals as their clock sources. For more information you may refer to the Peripherals part.

9.1.1 Clock Special Name Definition

Symbol	Description
IRCH	16MHz Internal RC Oscillator
IRCL	131KHz Internal RC Oscillator
XOSCL	32.768 KHz external Crystal Resonator

9.1.2 4 MHz Internal RC Oscillator(IRCH)

IRCH is the default the system clock after Power On and can be enabled or disabled by setting the bit IHCKE of the register CKCON.The precision can be 1% and factory frequency is 16MHz@3.3V/25℃.

9.1.3 131 KHz Internal RC Oscillator(IRCL)

IRCL can be enabled/disabled by setting the ILCKE of register CKCON. When IRCL is set as system clock the power consumption decrease as well. The precision can be 1% and factory frequency is 131KHz@3.3V/25℃.

9.1.4 32.768KHz Enternal Crystal Resonator(XOSCL)

XOSCL is mainly used as the clock source for RTC for real time timing. With the lowest power consumption even below 9uA, XOSCL is usually set as system clock when low power consumption is needed. It is enabled/disabled by setting the bit XLCKE of register CKCON. The time to start oscillation of XOSCL is not short which means it usually takes about 1 second, hence users have to wait until it is completely stabilized. The bit XLSTA of register CKCON is the flag for XOSCL clock stabilization.

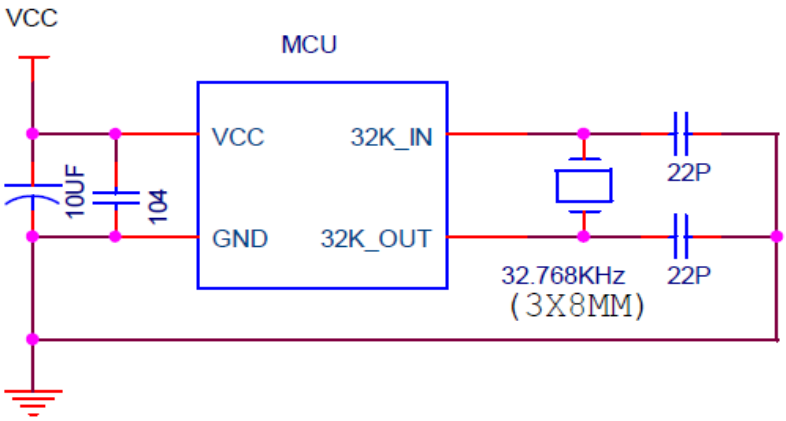


Figure 9-1-4-1 XOSCL Typical Circuit

Important reminders: 1 During hardware design, the ground of the crystal oscillator load capacitor must be connected with the chip ground, and the crystal oscillator compensation capacitor shall be close to the GND pin of the chip as far as possible. 32.768KHz quartz crystal oscillator requires crystal oscillator specifications with a diameter of 3mmx8mm.

2. The above circuit and component parameters are only for reference. When using crystal oscillators from different manufacturers, the parameters may need to be modified.

9.2 Clock Control Register Description

Table 9-2-1 Register CKCON

8080H	7	6	5	4	3	2	1	0
CKCON	ILCKE	IHCKE	-	-	XLCKE	XLSTA	-	-
R/W	R/W	R/W	-	-	R/W	R	-	-

Initial Value	0	0	-	-	0	0	-	-

Bit number	Bit symbol	Description
7	ILCKE	IRCL enable control 1：enable 0：disable *Note：* *When it is 1, the clock is enabled;* *when it is 0, if the system or other modules selected this clock source, the clock is still enabled.*
6	IHCKE	IRCH enable control 1：enable 0：disable *Note：* *When it is 1, the clock is enabled;* *when it is 0, if the system or other modules selected this clock source, the clock is still enabled.*
5~4	-	-
3	XHCKE	XOSCH enable control 1: enable 0: disable *Note：* *1.When it is 1, the clock is enabled;* *when it is 0, if the system or other modules selected this clock source, the clock is still enabled.* *2.Since XOSCH is external clock, the corresponding pin function must be set as XOSCH function to use it*
2	XHSTA	XOSCH clock stabilization flag (1 indicates it is stabilized)
1~0	-	-

Tabel 9-2-2 Register RCSTA

807FH	7	6	5	4	3	2	1	0
RCSTA	-	IHSTA	-	-	-	-	-	-
R/W	-	R	-	-	-	-	-	-
Initial Value	-	0	-	-	-	-	-	-

Bit number	Bit symbol	Description
7	-	-
6	IHSTA	IRCH clock stabilization flag, 1 enables it
5~0	-	-

Tabel 9-2-3 Register IHCFG

8083H	7	6	5	4	3	2	1	0

IHCFG	IHCFG[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
Bit number	Bit symbol	Description						
7~0	IHCFG	16MHz Internal clock configuration register *Note: This register automatically loads the value corresponding to the frequency of 16MHz after power-on. It is not recommended to modify this value except for special applications.*						

Table 9-2-4 Register ILCFGL AND ILCFGH

8085H	7	6	5	4	3	2	1	0
ILCFGL	ILCFG[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
8086H	7	6	5	4	3	2	1	0
ILCFGH	-	-	-	-	-	-	-	ILCFG[8]
R/W	-	-	-	-	-	-	-	R/W
Initial Value	-	-	-	-	-	-	-	0
Bit number	Bit symbol	Description						
8~0	ILCFG	131KHz Internal clock configuration register *Note: This register automatically loads the value corresponding to the frequency of 131KHz after power-on. It is not recommended to modify this value except for special applications.*						

9.3 System Clock

The system clock is controlled by register CKCON, CKSEL and CKDIV. Users can disable/enable any of these clock sources, divide the frequency, change the system clock and so on by using these registers.

9.3.1 System Clock Architecture

Please refer to figure 9-3-1.

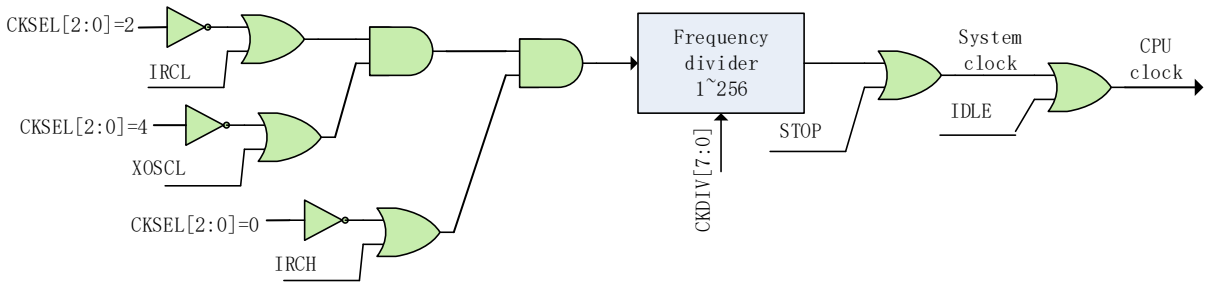


Figure 9-3-1 System Clock Architecture

9.3.2 System Clock Control Register Description

Table 9-3-2-1 Register CKSEL

8081H	7	6	5	4	3	2	1	0
CKSEL	RTCKS	-	-	-	-	CKSEL[2:0]		
R/W	R/W	-	-	-	-	R/W		
Initial Value	0	-	-	-	-	0	0	0

Bit number	Bit symbol	Description
7	RTCKS	RTC clock selection 0: XOSCL 1: IRCL *Note:when IRCL is selected, the clock's frequency will be divided by 4 first and then used for RTC*
6~3	-	-
2~0	CKSEL	System clock selection: 000: IRCH 001: Reserved 010: IRCL 011: XOSCL

		Others: IRCH *Note: if the IRCL is set as the system clock, you must wait about 1ms after enabling the IRCL clock before switching to the system clock, otherwise exceptions may occur.*

Table 9-3-2-2 Register CKDIV

8082H	7	6	5	4	3	2	1	0
CKDIV	CKDIV[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7~0	CKDIV	System clock frequency division： 00H: No division 01H: frequency divided by 2 02H: frequency divided by 3 03H: frequency divided by 4 …… FFH: frequency divided by 256

9.3.3 System Clock Control Method and Example

- **Set IRCH as the system clock**

  To set IRCH as the system clock, the program is as follows:

```
---------------------------------------------------------------------------------
#define IHCKE           (1<<6)
#define CKSEL_IRCH      0
void Sys_Clk_Set_IRCH(void)
{
    CKCON |= IHCKE;                         //enable IRCH
    CKSEL = (CKSEL&0xF8) | CKSEL_IRCH;      //set IRCH as system clock
}
---------------------------------------------------------------------------------
```

- **Set IRCL as the system clock**

  To set IRCL as the system clock, the program is as follows:

```
---------------------------------------------------------------------------------
#define ILCKE           (1<<7)
#define CKSEL_IRCL      2
void Sys_Clk_Set_IRCL(void)
{
    CKCON |= ILCKE;                         //enable IRCL
    Delay_ms(1);                            //delay 1ms and wait for IRCL stabilization
    CKSEL = (CKSEL&0xF8) | CKSEL_IRCL;      // set IRCL as system clock
}
---------------------------------------------------------------------------------
```

- **Set XOSCL as the system clock**

  To set XOSCL as the system clock, the program is as follows:

```
---------------------------------------------------------------------------------
#define XLCKE           (1<<3)
#define XLSTA           (1<<2)
#define CKSEL_XOSCL 4
void Sys_Clk_Set_XOSCL(void)
{
    P52F = 3;
    P53F = 3;
    CKCON |= XLCKE;
    while(!(CKCON & XLSTA));
    CKSEL = (CKSEL&0xF8) | CKSEL_XOSCL;
}
---------------------------------------------------------------------------------
```

10 Power Supply and Reset System

10.1 Power Supply

There is 1.8V - 5.5V source between VDD pin and VSS pin for CA51F4 series which supplies the power for the chip. VDD and LDO supply power for the analog system and LDO supplies power for the digital system.

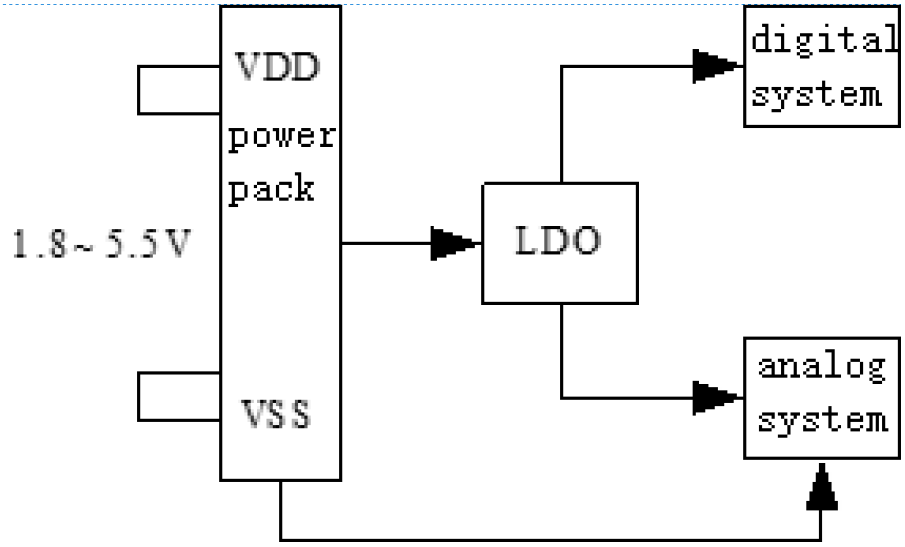


Figure 10-1-1 Power Supply Architecture

The Fig10-1-2 is the typical circuit for power supply

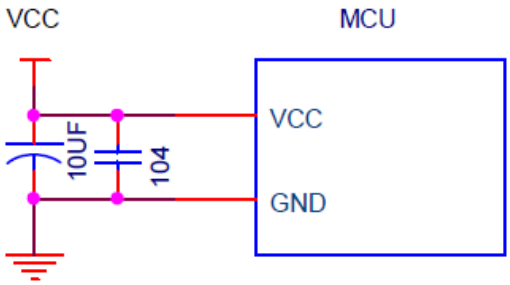


Figure 10-1-2 Typical Circuit for Power Supply

Important reminders: 1. The filter capacitors 47uF and 104 in the circuit below are the standard devices for the chip which can not be omitted, otherwise the chip may operates abnormally.

2. The above circuit and component parameters are for reference only, and may need to be modified according to the peripheral working environment and different voltage power supply parameters.

10.1.1 LDO Function Introduction

There is an internal low dropout regulator (LDO) for CA51F4 Series chip. LDO module offers supply voltage for the chip. The output voltage of LDO is set by VLEVEL (PWCON[2:0]) and the default value for VLEVEL is 5, which implies the default voltage is 1.61V. When VDD/VSS is less than the output voltage set by VLEVEL, the output voltage will be VDD directly; when VDD/VSS is greater than the output voltage set by VLEVEL, LDO output the voltage set by VLEVE. High LDO voltage is benefits to clock module' s rapid start while low LDO voltage will lower the chip' s power consumption. There are two working modes for LDO: High Power mode and Low Power mode, which is selected by VHL (PWCON[3]). The load capacity is also different in different modes. The current is higher in High Power mode but with higher power consumption, while in Low Power mode it is vice versa. For the most time when the system is operating normally, LDO is usually High Power mode. The Low Power mode is usually used for Power Save Modes such as STOP, IDLE, Low Speed Mode etc.

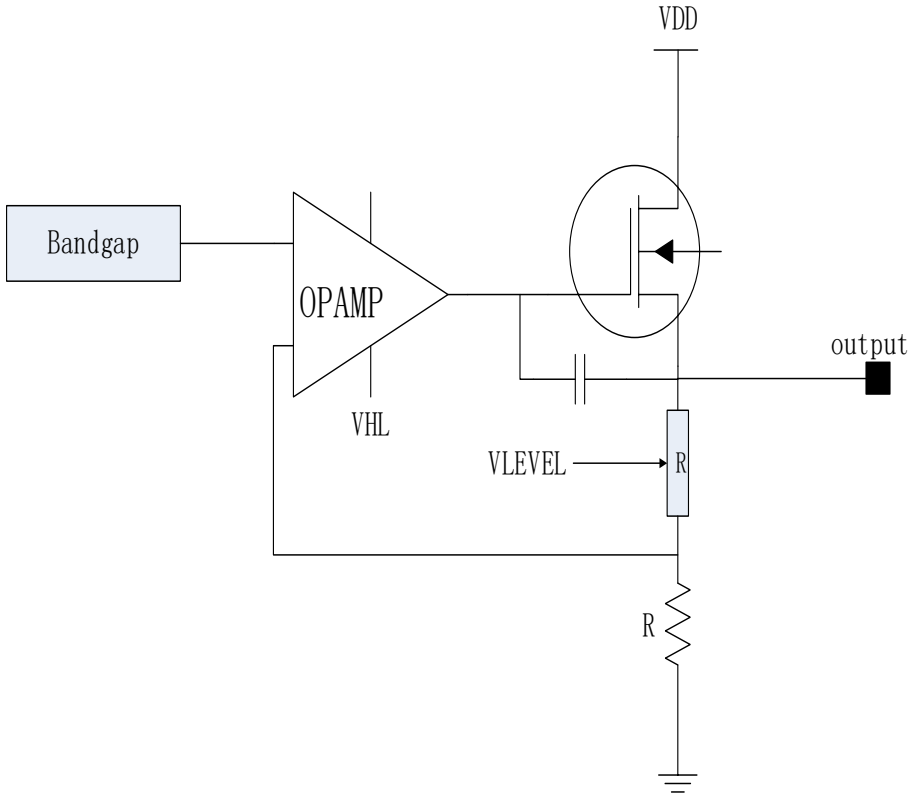


Figure 10-1-3 LDO Module Schematic

10.1.2 LDO Control Register

Table 10-1-2-1 Register PWCON

86H	7	6	5	4	3	2	1	0
PWCON	FLEVEL[3:0]				VHL	VLEVEL[2:0]		
R/W	R/W				R/W	R/W		
Initial Value	0	1	1	1	1	1	0	1

Bit number	Bit symbol	Description
7~4	FLEVEL	Internal reference voltage(Bandgap)output adjustment 0000: 0.825V 0001: 0.850V 0010: 0.875V 0011: 0.900V 0100: 0.925V 0101: 0.950V 0110: 0.975V 0111: 1.000V 1000: 1.025V 1001: 1.050V 1010: 1.075V 1011: 1.100V 1100: 1.125V 1101: 1.150V 1110: 1.175V 1111: 1.200V *Note: when the internal reference voltage is powered on, it is automatically loaded by the system and cannot be modified by the user*
3	VHL	LDO working mode selection 1: High Power mode 0: Low Power mode
2~0	VLEVEL	LDO output voltage selection 000: 1.31V 001: 1.37V 010: 1.43V 011: 1.49V 100: 1.55V 101: 1.61V 110: 1.67V 111: 1.73V *Note: 1. The internal clock circuit is powered by LDO. Changing the output voltage of LDO will cause the change of internal clock frequency. Generally, the LDO voltage can maintain the default value and is not recommended to be modified.* *2.It is not recommended to set LDO output voltage below 1.58V otherwise the chip may operates abnormally*

10.2 Reset System

There are multiple internal and external reset sources for CA51F4 Series chip as figure 10-2-1 shows.

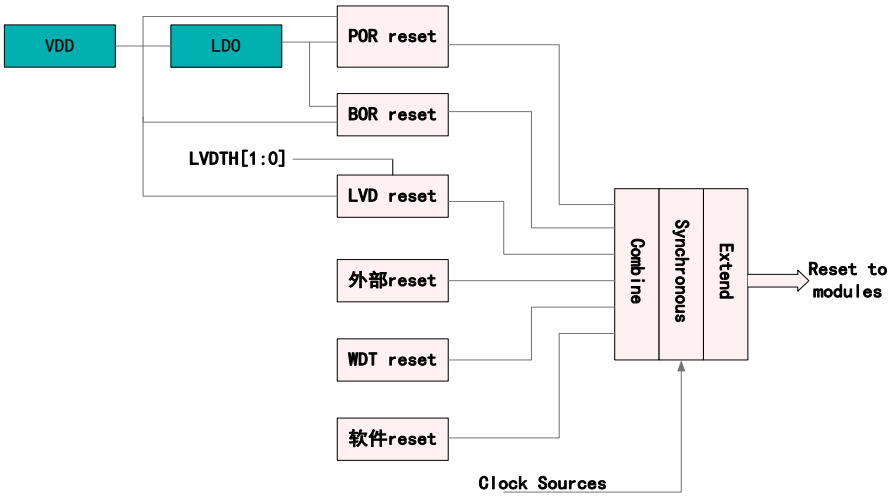


Figure 10-2-1 Reset System Architecture

- **Power On Reset(POR)**

It usually takes some time for the system to reach normal working voltage. The POR is mainly based on VDD and LDO. The POR signal is valid when the voltage is below the detection threshold.

The POR circuit ensures that the chip remains reset during the Power On period hence the chip can starts from certain stable status. The POR signal will also be expanded by the internal counter to makes sure that all the analog modules can enter stable working status after Power On stage.

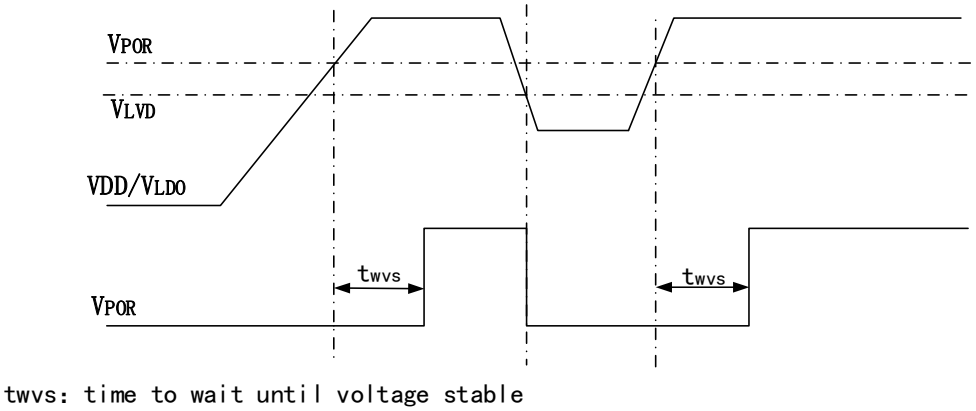


Figure 10-2-2 POR Circuit Example and Power On Stage

- **Brown Out Reset(BOR)**

BOR offers alarm signal for the chip when the voltage drops (eg. Inference or load changes). Once the VDD or internal LDO output voltage is below a certain threshold, it will reset the chip to avoid program error or

system abnormality.

- **Low Voltage Reset**

The Low Voltage Detection (LVD) can detect VDD in multiple working modes. When VDD voltage is below the threshold set by LVD for 20us it will generates reset signal (on the premise of that LVD is Reset mode).

- **External Reset**

By pulling down the reset pin(RESET), external device can reset the chip as well. RESET can reset the whole in normal working modes, while in STOP mode, the hard reset will awaken the chip first and then reset it. Usually, RESET is pulled up internally and will not influence the internal reset circuit.

- **Watchdog Reset**

The WDT (watchdog timer) is responsible for monitor the how processor do with instructions. With proper configuration, if the WDT is not refreshed in certain time, a reset signal will be generated. WDT is disabled after POR, but users can enable and configure it if necessary.

- **Soft Reset**

The program can soft reset the chip. When 1 is written to SWRST of register PCON, CPU sends out reset signal.

POR and external hard reset will reset all the circuits while LVD and WDT can reset other circuits but not reset themselves. (eg：After WDT reset, WDT registers remains former status while others are all reset) LVD/WDT and soft reset can not reset storage control circuit. Program starts from where BOOT configuration points to after soft reset. PC will point to address 0 after any reset.

11 Power Consumption Management

There are 3 low power consumption modes for CA51F4 Series: IDLE, STOP and Low Speed mode. The system power consumption for IDLE, STOP and Low Speed mode is less than 10uA, 3uA and 20uA respectively.

11.1 IDLE mode

CPU stops working in this mode. All the clocks can be disabled to save power before entering IDLE mode except the main clock. Peripherals can also be enabled/disabled before entering IDLE mode according to user's needs. Those enabled peripherals will operates normally in IDLE mode.

Register IDLST(IDLSTH and IDLSTL) needs to be checked before entering IDLE mode. If all the bits are 0, CPU will enter IDLE mode normally when the mode is set as IDLE. However, if NOT all the bits are 0, CPU will not enter IDLE mode and remains in normal working mode although the mode is set as IDLE. To deal with this situation, users must complete the IDLST corresponding interrupt processing first and then set the mode as IDLE again.

Any reset or interrupt will awake the chip. The clock will resume first and then the chip responds to the interrupt and enters the interrupt service routine after the CPU awakening. After the chip exits interrupt service routine , it will executes the instructions after the instruction which set IDLE to 1. When it exits IDLE mode, IDLE will be cleared automatically.

What must be mentioned is that there should be two "nop" instructions after setting IDLE to 1 to avoid program error.

11.2 STOP mode

The STOP mode is deeper low power consumption mode than IDLE. STOP mode is able to stop all the clocks (include the main clock) and clock generation circuits. If WDT and RTC are enabled, their clock module will still works, hence users may disable them to save power.

Similar to IDLE mode, before entering STOP mode, register STPST(STPSTH and STPSTL) has to check if all the bits are 0. If there are any 1, then they should be processed first to ensure the chip will enter STOP mode successfully.

The STOP mode can be awoken by external interrupt, LVD reset or interrupt, hard reset, RTC interrupt, WDT interrupt or reset, clock monitor interrupt and touch key interrupt. If it is awoken by an interrupt, the chip will resume clock first and respond to the interrupt, and then enters corresponding the interrupt service routine. After the chip exits the interrupt service routine, it will executes the instructions after the setting STOP to 1 instruction.The STOP will be cleared automatically when the chip exits STOP mode.

To arouse the chip better, it is recommended to set the internal clock as system clock before entering STOP

mode because it will take longer time waiting for stable status when using external clock.

When the chip enters STOP mode, the last clock edge will disable system clock and then the chip enters STOP mode entirely. What must be mentioned is that there should be three “nop” instructions after setting STOP to 1 avoid program error.

Note: 1. When it enters STOP/IDLE mode, setting LDO to low power consumption mode will reduce the power consumption effectively. However, it is a must to set LDO back to high power consumption mode when the chip exits STOP/IDLE mode, otherwise the chip will operates abnormally.

2. If the system clock is selected as IRCL, IRCL cannot be closed when entering STOP, otherwise an exception may occur when waking up from STOP.

11.3 Low Speed Mode

Since the power consumption is influenced by the its speed, so it will reduce the power consumption effectively if the main clock runs with low speed. The current will be less than 20uA if IRCL(131kHz IRCL) is set as the system clock.

11.4 Related Register Description

Table 11-4-1 Register PCON

87H	7	6	5	4	3	2	1	0
PCON	-	-	SWRST	-	-	TSMODE	STOP	IDLE
R/W	-	-	W	-	-	R	W	W
Initial Value	-	-	0	-	-	0	0	0

Bit number	Bit symbol	Description
7-6	-	-
5	SWRST	Soft reset control Setting SWRST=1 will generate soft reset signal, it will be cleared to 0 automatically after the reset
4~3	-	-
2	TSMODE	Test mode flag, 1 indicates that the chip is in test mode
1	STOP	STOP mode control, 1 enables STOP mode When STOP=1 and STPST=0, the chip will enter STOP mode. It will be cleared to 0 automatically after the chip exits STOP mode
0	IDLE	IDLE mode control, 1 enables IDLE mode When IDLE=1 and IDLST=0, the chip will enter IDLE mode. It will be cleared to 0 automatically after the chip exits IDLE mode

Table 11-4-2 Register IDLSTL and IDLSTH

FCH	7	6	5	4	3	2	1	0
IDLSTL	IDLST[7:0]							
R/W	R							
Initial Value	0	0	0	0	0	0	0	0
FDH	7	6	5	4	3	2	1	0
IDLSTH	-	IDLST[14:8]						
R/W	-	R						
Initial Value	-	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
15	-	-
14	PWMINT/EPIF[7]	Interrupt status of PWM/External Interrupt9 in IDLE mode
13	RTCINT/EPIF[6]	Interrupt status of RTC/External Interrupt8 in IDLE mode
12	WDFLG[1]/EPIF[5]	Interrupt status of WDT/External Interrupt7 in IDLE mode
11	I2CINT/SWIINT/EPIF[4]	Interrupt status of I2C/SWI/External Interrupt6 in IDLE mode
10	EPIF[3]	Interrupt status of External Interrupt5 in IDLE mode
9	LVDINT/EPIF[2]	Interrupt status of LVD/External Interrupt4 in IDLE mode
8	TKINT/TLINT/EPIF[1]	Interrupt status of TK/External Interrupt3 in IDLE mode
7	ADCINT/EPIF[0]	Interrupt status of ADC/External Interrupt2 in IDLE mode
6	U1INT	Interrupt status of UART1 in IDLE mode
5	T2INT	Interrupt status of Timer2 in IDLE mode
4	-	-
3	TCON[7]	Interrupt status of Timer1 in IDLE mode
2	PIF[1]	Interrupt status of External Interrupt1 in IDLE mode
1	TCON[5]	Interrupt status of Timer0 in IDLE mode
0	PIF[0]	Interrupt status of External Interrupt0 in IDLE mode

Table 11-4-2 Register STPSTL and STPSTH

FEH	7	6	5	4	3	2	1	0
STPSTL	STPST[7:0]							
R/W	R							
Initial Value	0	0	0	0	0	0	0	0
FFH	7	6	5	4	3	2	1	0
STPSTH	STPST[15:8]							
R/W	R							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
15	RTCWKF	Interrupt status of RTC in STOP mode

14	WDTWKF	Interrupt status of WDT in STOP mode
13	I2CWKF/SWIWKF	Interrupt status of I2C/SWI in STOP mode
12	-	-
11	LVDWKF	Interrupt status of LVD in STOP mode
10	TKWKF	Interrupt status of Touch Key in STOP mode
9	EPWKF[7]	Interrupt status of External Interrupt9 in STOP mode
8	EPWKF[6]	Interrupt status of External Interrupt8 in STOP mode
7	EPWKF[5]	Interrupt status of External Interrupt7 in STOP mode
6	EPWKF[4]	Interrupt status of External Interrupt6 in STOP mode
5	EPWKF[3]	Interrupt status of External Interrupt5 in STOP mode
4	EPWKF[2]	Interrupt status of External Interrupt4 in STOP mode
3	EPWKF[1]	Interrupt status of External Interrupt3 in STOP mode
2	EPWKF[0]	Interrupt status of External Interrupt2 in STOP mode
1	PWKF[1]	Interrupt status of External Interrupt1 in STOP mode
0	PWKF[0]	Interrupt status of External Interrupt0 in STOP mode

11.5 Low Power Consumption Control Example

- **STOP Mode Example**

The program is like:

```
---------------------------------------------------------------------------------------
void Stop(void)
{
    bit IE_EA;
    I2CCON = 0;//disable I2C for it is the default enabled, otherwise the IRCH cannot be disabled
    SWICON |= 0x01; //disable the single line communication function, otherwise the master clock cannot be disabled
    CKCON = 0;          //disable all the clocks
    PWCON &= ~0x08;//set LDO in low power consumption mode
    MECON |= (1<<6); //set FLASH in deep sleep mode
    while(STPSTH|STPSTL); //wait until all the interrupts are done
    IE_EA = EA;   //save global interrupt enable bit status
    EA = 0;
    PCON |= 0x02;       //enters STOP mode
    _nop_();
    _nop_();
    _nop_();
    EA = IE_EA;
    PWCON |= 0x08;    //the LDO must return to high power consumption mode after the chip exits STOP mode
}
---------------------------------------------------------------------------------------
```

- **IDLE Mode Example**

The program is like:

```
---------------------------------------------------------------------------------------
void Idle(void)
{
    CKCON |= (1<<7);              //enable IRCL clock
    CKSEL = (CKSEL&0xF8) | 2; //set IRCL as the system clock
    I2CCON = 0;   //disable I2C for it is the default enabled, otherwise the IRCH cannot be disabled
    SWICON |= 0x01;   //disable the single line communication function, otherwise the master clock cannot be disabled
    CKCON = 0;          //disable all the clocks
    PWCON &= ~0x08;//set LDO in low power consumption mode
    MECON |= (1<<6);
    while(IDLSTH|IDLSTL); //wait until all the interrupts are done

    PCON |= 0x01;       //enters IDLE mode

    _nop_();
    _nop_();
    PWCON |= 0x08;    //the LDO must return to high power consumption mode after the chip exits IDLE mode
}
```

Note: Since the main clock is still enabled in IDLE mode, if it is high speed clock then the power consumption remains high. Thus, it is very necessary to switch the main clock to low speed clock before entering Low Speed mode. ----------

---

◆ **Low Speed Mode Example**

The program is like:

```
---------------------------------------------------------------------------------------
void LowSpeedMode(void)
{
    CKCON |= (1<<7);              //enables IRCL clock
    CKSEL = (CKSEL&0xF8) | 2; //set IRCL as the system clock
    I2CCON = 0;   //disable I2C for it is the default enabled, otherwise the IRCH cannot be disabled
    SWICON |= 0x01;   //disable the single line communication function, otherwise the master clock cannot be disabled
    CKCON = 0;          //disable all the clocks
    PWCON &= ~0x08;//set LDO in low power consumption mode
}
```

Note: The LDO must return to high power consumption mode after the chip exits Low Speed mode, similar to STOP/IDLE example

---

12 General Timer(Timer0, Timer1, Timer2)

12.1 Timer0

12.1.1 Timer0 Introduction

The timer/counter function can be selected by CT0(TMOD[2]). When CT0=0 it operates as a timer; when CT0=1, it functions as a counter. As a timer, its clock is the system clock with frequency divided by 12. As a counter, its clock is the input clock for T0. Because it takes 2 clock cycles to detect the T0 input signal edge change, so when it operates as a counter, the maximum input baud rate is 1/2 of the internal system clock frequency. There is no limit for T0 input signal's duty cycle. However, in order to identify the 0 and 1 clearly, the signal has to keep for at least one internal system clock cycle. There are for modes for Timer0 which are selected by T0M0 and T0M1(TMOD[1:0]).

- **Mode0**

Timer0 is a 13 bit timer/counter in this mode. The higher 8 bits are stored in TH0 and the lower 5 bits are stored in TL0[4:0] with TL0[7:5] invalid. When Timer0 overflows, the interrupt flag TF0(TCON[5]) will be set to 1. TF0 will be cleared automatically after the interrupt response. When GATE0(TCON[3])=0, the timer/counter's is enabled/disabled by TR0(TCON[4]). When GATE0=1, the timer/counter's is enabled/disabled by INT0. INT0 signal with high level with enable the counting and vice verse.

- **Mode1**

Timer0 is a 16 bit timer/counter in this mode. The function is the same as Mode0.

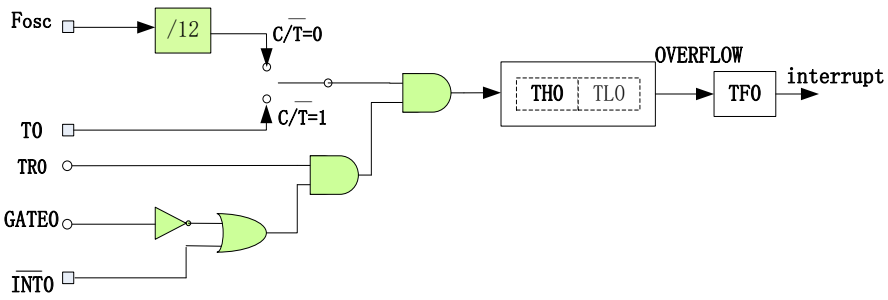


Figure 12-1-1-1 Timer0 Mode0/1

- **Mode2**

Timer0 is a 8 bit automatic reload counter/timer in this mode and only TL0 counts up automatically. When TL0 count overflows, there will be an interrupt flag TF0. The initial value for the count will be reloaded to TL0 from TH0 as well. The other settings are the same as mode0/1.

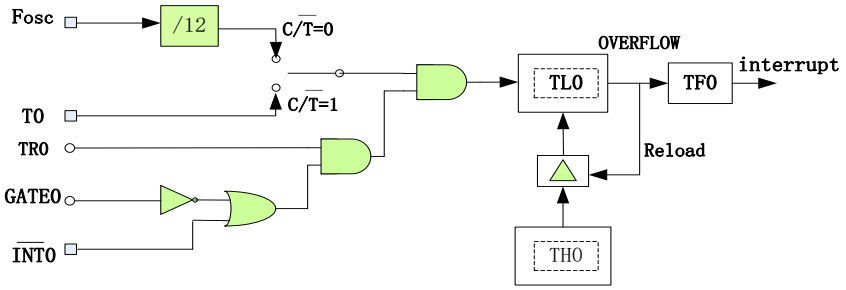


Figure 12-1-1-2 Timer0 Mode2

- **Mode3**

TL0 and TH0 are two independent 8 bit counter/timer in this mode. TL0 can be used as timer or counter while TH0 can only be used as counter. TL0 will be controlled by CT0, GATE0, TR0, TF0 and INT0 and TH0 will only be controlled by TR1 and TF1. The control method is the same as mode0/1. When Timer0 is working in mode3, Timer1 and TH0 both are controlled by TR1. Due to TF1 is used for TH0 already, at the same time, Timer1 can only be used when there is no need for interrupt. (eg, UART baud rate generation)

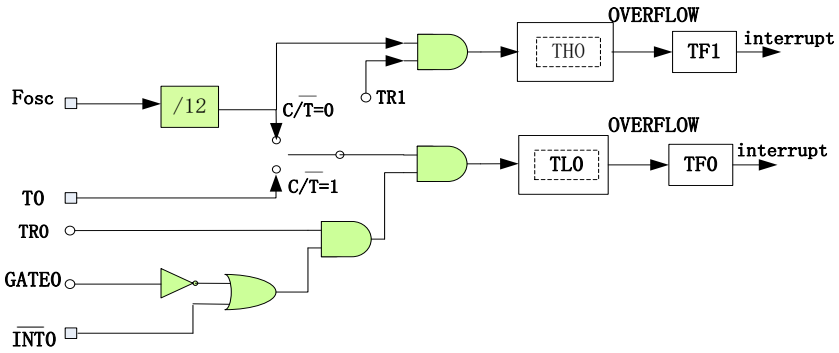


Figure 12-1-1-3 Time 0 Mode3

12.1.2 Timer0 Register Description

Table 12-1-2-1 Register TCON

88H	7	6	5	4	3	2	1	0
TCON	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	TF1	Timer0 TH0 overflow flag in mode3 /Time 1 overflow flag, it is cleared automatically after the interrupt response
6	TR1	Timer1 enable control, 1 enables it
5	TF0	Timer0 overflow flag, it is cleared automatically after the interrupt response
4	TR0	Timer0 enable control, 1 enables it
3	IE1	External Interrupt 1 enable control, 1 enables it

2	IT1	External Interrupt 1 trigger type control 0：External Interrupt 1 is triggered when input pin signal is rising edge 1：External Interrupt 1 is triggered when input pin signal comes to falling edge
1	IE0	External Interrupt 0 enable control, 1 enables it
0	IT0	External Interrupt 0 trigger type control 0：External Interrupt 0 is triggered when input pin signal is rising edge 1：External Interrupt 0 is triggered when input pin signal comes to falling edge

Table 12-1-2-2 Register TMOD

89H	7	6	5	4	3	2	1	0
TMOD	GATE1	CT1	T1M1	T1M0	GATE0	CT0	T0M1	T0M0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7	GATE1	Timer1 gating control. When it equals 1, Timer1 is enabled/disabled by INT1
6	CT1	Timer1 Counter/Timer selection 0：Timer, the clock for it is the system clock with its frequency divided by 12 1：Counter, the clock for it is T1 input clock
5	T1M1	[T1M1,T1M0] for Timer1 mode selection
4	T1M0	00：mode0, TL1 and TH1 make up a 13 bit Timer/Counter 01：mode1, TL1 and TH1 make up a 16 bit Timer/Counter 10：mode2, TL1 is a 8 bit Timer/Counter, TH1 is the automatic reload register 11：mode3, TH1/TL1 locked in this mode, it is the same as TR1=0
3	GATE0	Timer0 gating control. When it equals 1, Timer0 is enabled/disabled by INT0
2	CT0	Timer0 Counter/Timer selection 0：Timer, the clock for it is the system clock with its frequency divided by 12 1：Counter, the clock for it is T0 input clock
1	T0M1	[T0M1,T0M0] Timer0 mode selection
0	T0M0	00：mode0, TL0 and TH0 make up a 13 bit Timer/Counter 01：mode1, TL0 and TH0 make up a 16 bit Timer/Counter 10：mode2, TL0 is a 8 bit Timer/Counter, TH0 is the automatic reload register 11：mode3, TL0 and TH0 are two independent 8 bit Timer/Counter

Table 12-1-2-3 Register TL0

8AH	7	6	5	4	3	2	1	0
TL0	TL0							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7~0	TL0	Lower byte of Timer0 count value in mode0/1, count value in mode2/3

Table 12-1-2-4 Register TH0

8CH	7	6	5	4	3	2	1	0
TH0	TH0							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7~0	TH0	Higher byte of Timer0 count value in mode0/1, reload value in mode2, count value in mode3

12.2 Timer1

12.2.1 Timer1 Introduction

The timer/counter function can be selected by CT1 (TMOD[6]). When CT1=0 it operates as a timer; when CT1=1, it functions as a counter. As a counter, its clock is the input clock for T1. Because it takes 2 clock cycles to detect the T1 input signal edge change, so when it operates as a counter, the maximum input baud rate is 1/2 of the internal system clock frequency. There is no limit for T1 input signal's duty cycle. However, in order to identify the 0 and 1 clearly, the signal has to keep for at least one internal system clock cycle time. There are for modes for Timer1 which are selected by T1M0 andT1M1 (TMOD[5:4]).

- **Mode0**

Timer1 is a 13 bit timer/counter in this mode. The higher 8 bits are stored in TH1 and the lower 5 bits are stored in TL1[4:0] with TL1[7:5] invalid. When Timer1 overflows, the interrupt flag TF1(TCON[7]) will be set to 1. TF1 will be cleared automatically after the interrupt response. When GATE1(TCON[7])=0, the timer/counter's is enabled/disabled by TR1 (TCON[6]). When GATE1=1, the timer/counter's is enabled/disabled by INT1. INT1 signal with high level with enable the counting and vice verse.

- **Mode1**

Timer1 operates as a 16 bit timer/counter in this mode. TH1 stores the higher 8bits of the 16 bit timer/counter and TL1 stores the lower 8 bits. When Timer1 overflows, the interrupt flag TF1(TCON[7]) will be set to 1. TF1 will be cleared automatically after the interrupt response. When GATE1(TCON[7])=0, the Timer/Counter's is enabled/disabled by TR1(TCON[6]). When GATE1=1, the timer/counter's is enabled/disabled by INT1. INT0 signal with high level with enable the counting and vice verse.

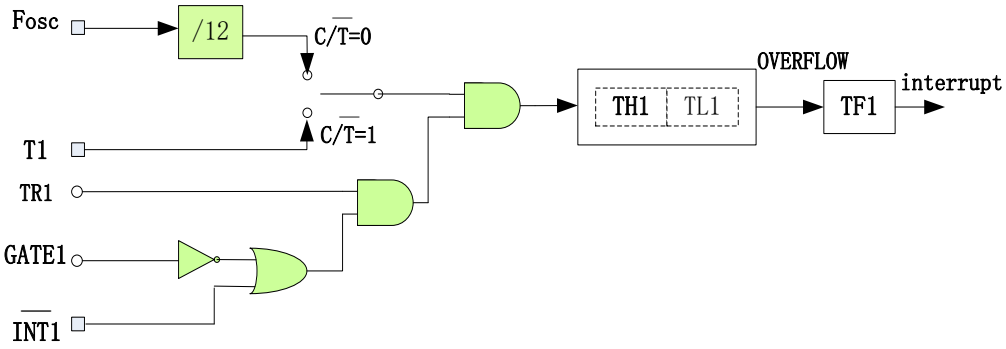


Figure 12-2-1 Timer1 Mode0/1

- **Mode2**

Timer1 is a 8 bit automatic reload counter/timer in this mode and only TL1 counts up automatically. When TL1 count overflows, there will be an interrupt flag TF1. The initial value for the count will be reloaded to TL1 from TH1 as well. The other settings are the same as mode0/1.

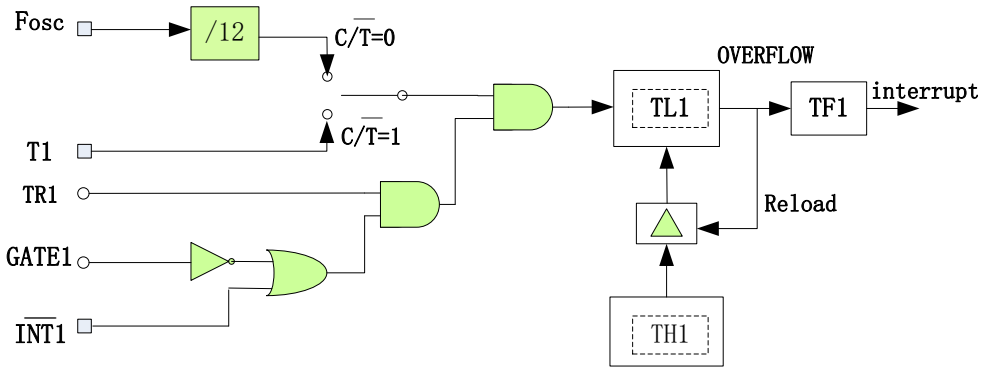


Figure 12-2-2 Timer1 Mode2

- **Mode3**

TH1 and TL1 are locked in this mode, which makes it the same as TR1=0.

12.2.2 Timer1 Register Description

For the the register TCON and TMOD please refer to Table12-1-2-1 and Table 12-1-2-2.

Table 12-2-2-1 Register TL1

8BH	7	6	5	4	3	2	1	0
TL1	TL1							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
Bit number	Bit symbol	Description						

7~0	TL1	Lower byte of Timer1 count value in mode0/1, count value in mode2/3

Table 12-2-2-2 Register TH1

8DH	7	6	5	4	3	2	1	0
TH1	TH1							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	Description
7~0	TH1	Higher byte of Timer1's count value in mode0/1, reload value in mode2, count value in mode3

12.3 Timer2

12.3.1 Timer2 Introduction

Timer2 is a 16 bit (TH2 and TL2) timer/counter. T2P0 and T2P1 can be used to select different control modes or clock sources. When T2P=0 or 3, the system clock is directly selected as the clock for Timer2 (Unlike Timer0/1, the frequency of the system clock is not divided by 12).

When T2P=0, Timer2 is enabled/disabled by TR2； when T2P=2, it is electrical level gated by T2. When the level of T2 is high, the count is enabled, and when it is low, the count stops. When T2P=1 or 2, the input signal of T2 is selected as the count clock. It counts the falling edges when T2P=1 and rising edges when T2P=2.

The working modes of Timer2 can be selected by setting T2M0 and T2M1. When T2M=0, Timer2 operates as a counter/timer. TH2 and TL2 counts up as a 16 bit counter. Two reload modes can be selected or disabled by setting T2R0 and T2R1 in this mode. T2CH and T2CL stores the reload value in reload mode. If T2R=2, Timer2 will reload the initial count value from T2CH and T2CL to TH2 and TL2 when it overflows. If T2R=3, it reloads when pin T2EX comes to falling edge. The reload flag is set to 1 after the reload. If Timer2 interrupt enables reload interrupt, RF2 can be cleared by writing 1 to it.

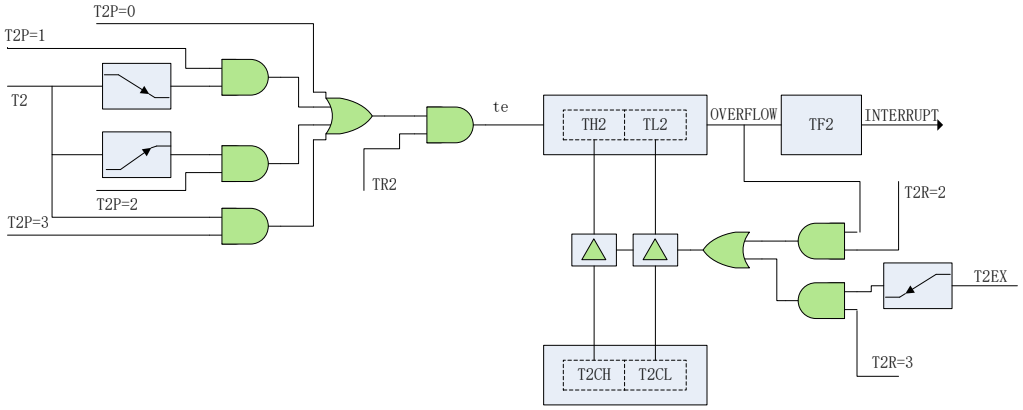


Figure 12-3-1-1 Timer2 Reload Mode

When T2M=1, Timer2 operates in compare mode. When TH2 and TL2 are greater than T2CH and T2CL, the pin T2CP output is high level, otherwise T2CP outputs low level signal.

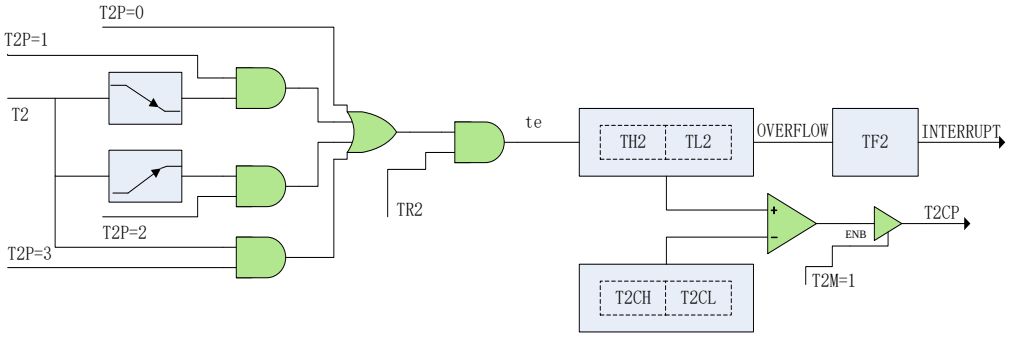


Figure 12-3-1-2 Timer2 Compare Mode Compare Mode

When T2M=2 or 3, Timer2 operates in capture mode. If T2M=2, when T2CP trigger edge comes, Timer2's count TH2 and TL2 will be latched to T2CH and T2CL. The trigger edge can be set by CCFG. The capture flag CF2 will be set to 1 after the capture happened. If Timer2 enables capture interrupt, CF2 can be cleared by writing 1 to it. When T2M=3, writing register T2CL will trigger the latch, and the value written will not be stored. Capture will not set CF2 to 1 in this mode.

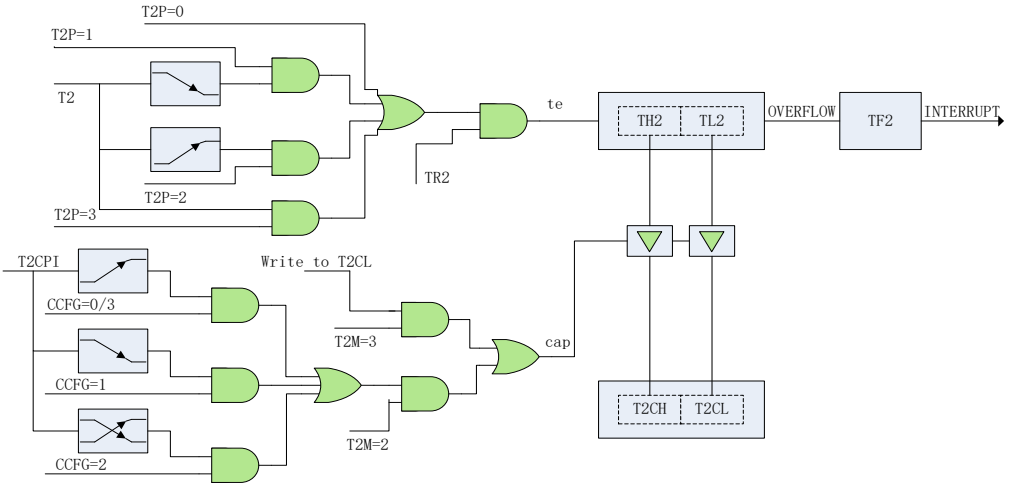


Figure 12-3-1-3 Timer2 Capture Mode

12.3.2 Timer2 Register Description

Table 12-3-2-1 Register T2CON

C8H	7	6	5	4	3	2	1	0
T2CON	-	TR2	T2R1	T2R0	T2IE	UCKS	T2P1	T2P0
R/W	-	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	-	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7	-	-
6	TR2	Timer2 enable control, 1 enables it
5	T2R1	[T2R1,T2R0]Timer2 reload mode selection 10： mode0 11： mode1 Others：disable reload mode
4	T2R0	
3	T2IE	Timer2 interrupt enable control,1 enables it
2	UCKS	UART0 clock selection 0：Timer1 overflow impulse used for UART0 clock 1：Timer2 overflow impulse used for UART0 clock
1	T2P1	[T2P1,T2P0]Timer2 pin T2 function selection 00：Timer2 uses internal system clock for count instead of T2 01：Timer2 counts T2 falling edges 10：Timer2 counts T2 rising edges 11：Timer2 uses internal system clock for count which is gated by T2
0	T2P0	

Table 12-3-2-2 Register T2MOD

C9H	7	6	5	4	3	2	1	0
T2MOD	TF2	CF2	RF2	CCFG1	CCFG0	-	T2M1	T2M0
R/W	R/W	R/W	R/W	R/W	R/W	-	R/W	R/W
Initial Value	0	0	0	0	0	-	0	0

Bit number	Bit Symbol	Description
7	TF2	Timer2 counter overflow interrupt flag, cleared by writing 1 to it
6	CF2	Capture interrupt flag, cleared by writing 1 to it
5	RF2	Automatic reload interrupt flag, cleared by writing 1 to it
4	CCFG1	[CCFG1,CCFG0] capture mode trigger selection, valid when T2M＝3 or T2M＝4 01：falling edge 10：rising or falling edge Others：rising edge
3	CCFG0	
2	-	-
1	T2M1	working mode selection 00：Timer/ counter mode 01： compare mode 10： capture mode 0 11： capture mode 1
0	T2M0	

Table 12-3-2-3 Register T2CL

CAH	7	6	5	4	3	2	1	0
T2CL	T2CL							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	T2CL	T2CL is the lower byte of reload value in reload mode T2CL is the lower byte of compare value in compare mode T2CL is the lower byte of capture value in capture mode

Table 12-3-2-4 Register T2CH

CBH	7	6	5	4	3	2	1	0
T2CH	T2CH							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	T2CH	T2CL is the higher byte of reload value in reload mode T2CL is the higher byte of compare value in compare mode T2CL is the higher byte of capture value in capture mode

Table 12-3-2-5 Register TL2

CCH	7	6	5	4	3	2	1	0
TL2	TL2							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
Bit number	Bit Symbol	Description						
7~0	TL2	Lower byte of the count value in Timer2						

Table 12-3-2-6 Register TH2

CDH	7	6	5	4	3	2	1	0
TH2	TH2							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
Bit number	Bit Symbol	Description						
7~0	TH2	Higher byte of the count value in Timer2						

13 Watchdog Timer(WDT)

13.1 Watchdog Timer(WDT) Function Introduction

The watchdog timer is a 27 bit backward counter with alternate clock sources. When the clock frequency is 16MHz, the count time can be 0.128ms –8.389s with 16 bit adjustment precision. The watchdog is mainly used for monitoring the system so that CPU will not break down due to external interference. If the software can not refresh WDT before it overflows, the watchdog will generate internal reset or interrupt. Writing A5H to register WDFLG will refresh the watchdog and reading WDFLG will get the status of the watchdog. If the watchdog is enabled in STOP mode, then the clock selected by the watchdog will works normally. In addition, if the interrupt function is also enabled for watchdog, it will awaken CPU in STOP mode.

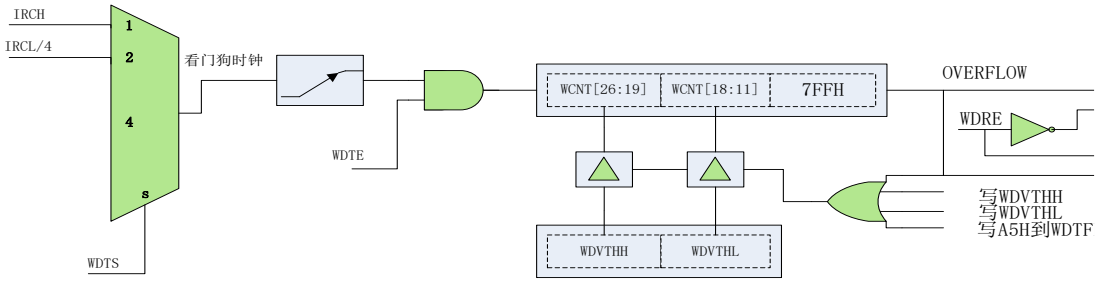


Figure 13-1-1 Watchdog Module Architecture

13.2 Watchdog Timer(WDT) Register Description

Table 13-2-1 Register WDCON

AAH	7	6	5	4	3	2	1	0
WDCON	WDTS[1:0]			-	-	-	-	WDRE
R/W	R/W			-	-	-	-	R/W
Initial Value	0	0	0	-	-	-	-	0

Bit number	Bit Symbol	Description
7~5	WDTS	WDT clock selection 001：IRCH 010：IRCL with frequency divided by 4 100：invalid Others：WDT disabled
4~1	-	

0	WDRE	WDT function selection 0：interrupt happens when WDT overflows 1：reset happens when WDT overflows

Table 13-2-2 Register WDFLG

ABH	7	6	5	4	3	2	1	0
WDFLG							WDIF	WDRF
R/W	-						R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~2	-	-
1	WDIF	WDT interrupt flag, writing A5H to the register will clear it
0	WDRF	WDT reset flag, writing A5H to the register will clear it

Table 13-2-3 Register WDVTHL、WDVTHH

ACH	7	6	5	4	3	2	1	0
WDVTHL	WDVTH[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
ADH	7	6	5	4	3	2	1	0
WDVTHH	WDVTH[15:8]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
15~0	WDVTH	WDT threshold setting, the equation is as follows: WDT trigger time ＝ (WDVTH * 800H+7FFH) * clock cycle

13.3 Watchdog Timer Control Example

◆ **Example for Watchdog interrupt mode**

For instance, IRCH is set for the watchdog clock and the frequency for it is 16MHz. The watchdog works in interrupt mode and the overflow time is one second, the program is like:

```
#define WDTS_IRCH          (1<<5)
#define WDTS_IRCL          (1<<6)
#define WDRE_reset         (1<<0)
#define WDRE_int           (0<<0)
void WDT_init(void)
{
    WDCON = WDTS_IRCH | WDRE_int;        //set the clock as IRCH and watchdog in interrupt mode
    WDVTHH = 0x1E;                       // set one second as the time for watchdog
    WDVTHL = 0x84;
    WDFLG = 0xA5;                        // refresh the watchdog
    INT7EN = 1;                          // watch dog interrupt start
    EA = 1;                              //global interrupt start
}
void WDT_ISR (void) interrupt 12
{
    if(WDFLG & 0x02)
    {
                                         // watch dog interrupt service routine
        WDFLG = 0xA5;                    // refresh the watchdog
    }
}
-------------------------------------------------------------------------------
```

◆ **Example for Watchdog reset mode**

For instance, IRCH is set for the watchdog clock and the frequency for it is 16MHz. The watchdog works in reset mode and the overflow time is one second, the program is like：

```
-------------------------------------------------------------------------------
#define WDTS_IRCH          (1<<5)
#define WDTS_IRCL          (1<<6)
#define WDRE_reset         (1<<0)
#define WDRE_int           (0<<0)
void WDT_init(void)
{
    WDCON = WDTS_IRCH | WDRE_reset; // set the clock as IRCH and watchdog in reset mode
    WDVTHH = 0x1e;                  // set one second as the time for watchdog
     WDVTHL = 0x84;
    WDFLG = 0xA5;                   // refresh the watchdog
}
-------------------------------------------------------------------------------
```

14 Real Time Clock(RTC)

14.1 RTC Function Introduction

The internal RTC is a real time clock module including millisecond, second, minute, hour, day and week registers and with alarm clock function embedded. The main clock source for it is the 32.768KHz external crystal oscillator. If the RTC time matches the time set by users, there will be an interrupt which makes it very convenient for product with (alarm) clock. In addition, RTC can set millisecond/half second interrupt with the interrupt time configurable for millisecond interrupt. Without 32.768KHz external crystal oscillator, the IRCL with frequency divided by 4 can also be the clock source for RTC when there is no need for high accuracy. In STOP/IDLE mode, RTC can be enabled and operates as the trigger source to waken the chip. Figure 14-1-1 shows the RTC architecture.

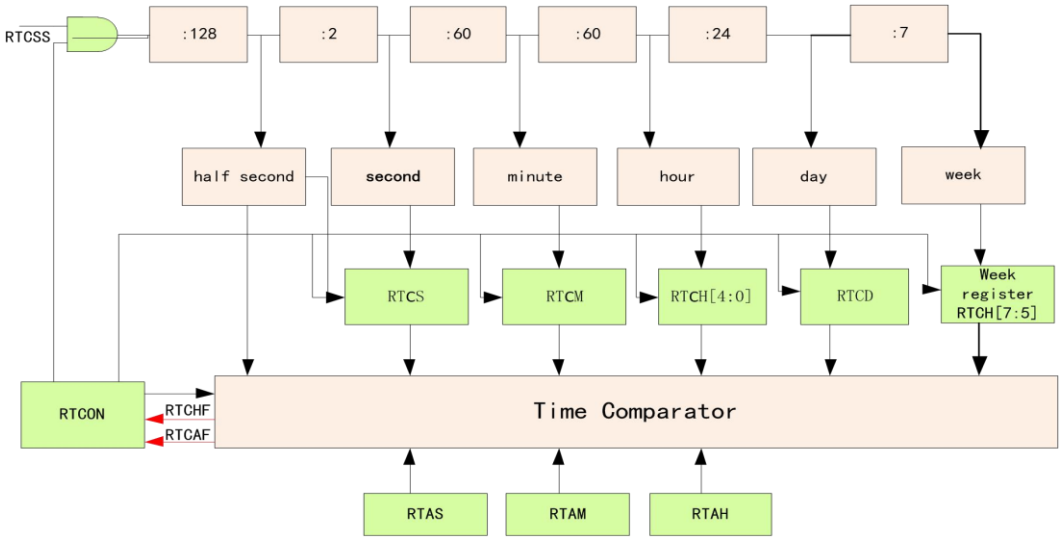


Figure 14-1-1 RTC Architecture

- **Enable/disable RTC**

RTC can be enabled/disabled by RTCE (RTCON[7]). RTC starts counting after RTCE=1 and if RTCE=0, all the registers of RTC module will be latched. It is a must to wait 300us after the RTC is enabled and then write the time register, otherwise it is invalid. Since RTC clock source is the 32.768KHz external crystal oscillator, it must wait until the oscillator works normally and then the RTC can be enabled.

- **RTC Register R/W**

RTCWE(RTCON[1]) enables/disables RTC registers (RTCSS, RTCS, RTCM, RTCH, RTCDL, RTCDH) writing.When RTCWE = 1, users have to 50us to overwrite RTC registers. RTCWE waits 50us after the writing and then changes to 0. Any illegal time which is beyond the second/minute/hour range will be seen as the maximum value of the register. The microsecond register RTCSS will be cleared when second/minute/hour/week is written. RTC registers can be read directly.

- **RTC Alarm Clock**

When RTC time matches alarm clock time, there will be an interrupt generated and the flag is RTCAF。Users can set the alarm clock time using register RTAS, RTAM and RTAH instead of setting RTCWE. Any illegal time which is beyond the second/minute/hour range will be seen as the maximum value of the register. Users may set corresponding compare enable control(HCE、MCE、SCE)to compare the values in register RTAS, RTAM and RTAH. If the enable control is set to 0, the corresponding time compare will be ignored (For instance, HCE=1, MCE=0, SCE=1, then only the hour and second will be compared, with minute default matched). In the end of the day, the alarm clock can occurs only once with all the compare enabled or several times periodically(every minute/hour/day) by select specific compares enabled.

14.2 RTC Register Description

Table 14-2-1 Register RTCON

F1H	7	6	5	4	3	2	1	0
RTCON	RTCE	MSE	HSE	SCE	MCE	HCE	RTCWE	-
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	-
Initial Value	0	0	0	0	0	0	0	-

Bit number	Bit Symbol	Description
7	RTCE	RTC clock enable control, 1 enables it
6	MSE	The millisecond interrupt enable control, 1 enables it
5	HSE	The half second interrupt enable control, 1 enables it
4	SCE	The alarm clock second compare enable control, 1 enables it
3	MCE	The alarm clock minute compare enable control, 1 enables it
2	HCE	The alarm clock hour compare enable control, 1 enables it
1	RTCWE	Clock write enable control, 1 enables it
0	-	-

Table 14-2-2 Register RTCSS

E9H	7	6	5	4	3	2	1	0
RTCSS	RTCSS[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	-

Bit number	Bit Symbol	Description
7~0	RTCE	RTC microsecond counter, 1 is added to it every 1/256 second

Table 14-2-3 Register RTCS

F2H	7	6	5	4	3	2	1	0
RTCS	-	-	RTCS[5:0]					
R/W	-	-	R/W					
Initial Value	-	-	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~6	-	-
5~0	RTCS	Second counter ranges from 0 to 59, 1 is added to it every 1 second

Table 14-2-4 Register RTCM

F3H	7	6	5	4	3	2	1	0
RTCM	-	-	RTCM[5:0]					
R/W	-	-	R/W					
Initial Value	-	-	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~6	-	-
5~0	RTCM	Minute counter ranges from 0 to 59, 1 is added to it every 1 minute

Table 14-2-5 Register RTCH

F4H	7	6	5	4	3	2	1	0
RTCH	RTCW[2:0]			RTCH[4:0]				
R/W	R/W			R/W				
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~5	RTCW	Week counter ranges from 1 to 7 which stands for Monday to Sunday, when it is set to 0, the week count function is disabled
4~0	RTCH	Hour counter ranges from 0 to 23, 1 is added to it every 1 hour

Table 14-2-6 Register RTCDL、RTCDH

F5H	7	6	5	4	3	2	1	0
RTCDL	RTCD[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
F6H	7	6	5	4	3	2	1	0
RTCDH	RTCD[15:8]							
R/W	R/W							

Initial Value	0	0	0	0	0	0	0	0
Bit number	Bit Symbol	Description						
15~0	RTCD	Day counter, 1 is added to it every day						

Table 14-2-7 Register RTAS

EAH	7	6	5	4	3	2	1	0
RTAS	-	-	RTAS[5:0]					
R/W	-	-	R/W					
Initial Value	-	-	0	0	0	0	0	0
Bit number	Bit Symbol	Description						
7~6	-	-						
5~0	RTAS	The alarm clock second setting (ranges from 0 to 59)						

Table 14-2-8 Register RTAM

EBH	7	6	5	4	3	2	1	0
RTAM	-	-	RTAM[5:0]					
R/W	-	-	R/W					
Initial Value	-	-	0	0	0	0	0	0
Bit number	Bit Symbol	Description						
7~6	-	-						
5~0	RTAM	The alarm clock minute setting (ranges from 0 to 59)						

Table 14-2-9 Register RTAH

ECH	7	6	5	4	3	2	1	0
RTAH	-	-	-	RTAH[4:0]				
R/W	-	-	-	R/W				
Initial Value	-	-	-	0	0	0	0	0
Bit number	Bit Symbol	Description						
7~5	-	-						
4~0	RTAH	The alarm clock hour setting(ranges from 0 to 23)						

Table 14-2-10 Register RTMSS

EDH	7	6	5	4	3	2	1	0

RTMSS	RTMSS[7:0]							
R/W	R/W							
Initial Value	0	0	0-	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	RTMSS	RTC millisecond interrupt threshold register, millisecond interrupt time =(RTMSS+1)× 128 × RTC clock cycle. If the RTC clock frequency is 32.768KHz, then the time unit is 128 ×(1/32.768)=3.90625ms

Table 14-2-11 Register RTCIF

EEH	7	6	5	4	3	2	1	0
RTCIF	-	-	-	-	-	RTCMF	RTCHF	RTCAF
R/W	-	-	-	-	-	R	R	R
Initial Value	-	-	-	-	-	0	0	0

Bit number	Bit Symbol	Description
7~3	-	-
2	RTCMF	RTC millisecond interrupt flag, cleared when 1 is written to it
1	RTCHF	RTC half second interrupt flag, cleared when 1 is written to it
0	RTCAF	RTC alarm clock interrupt flag, cleared when 1 is written to it

14.3 RTC Control Example

◆ **Write time to RTC**

To write hour, minute and second, the program is like：

```
//CKCON define register
#define XLCKE          (1<<3)
#define XLSTA          (1<<2)
//define RTCON
#define RTCE(N)        (N<<7)
#define MSE(N)         (N<<6)
#define HSE(N)         (N<<5)
#define SCE(N)         (N<<4)
#define MCE(N)         (N<<3)
#define HCE(N)         (N<<2)
#define RTCWE(N)       (N<<1)
//define RTCIF
#define RTC_MF         (1<<2)
#define RTC_HF         (1<<1)
#define RTC_AF         (1<<0)
```

```
void RTC_WriteHour(unsigned char hour)        //hour=0~23
{
    RTCON |= RTCWE(1);
    RTCH = (RTCH&0xE0)|hour;
    Delay_50us(1);
    RTCON &= ~RTCWE(1);
}
void RTC_WriteMinute(unsigned char minute) //minute=0~59
{
    RTCON |= RTCWE(1);
    RTCM = minute;
    Delay_50us(1);
    RTCON &= ~RTCWE(1);
}
void RTC_WriteSecond(unsigned char second) //second=0~59
{
    RTCON |= RTCWE(1);
    RTCS = second;
    Delay_50us(1);
    RTCON &= ~RTCWE(1);
}
```

◆ **Set the alarm clock time to RTC**

For instance, set the alarm clock time 11:30:0 with hour, minute and second compare enabled, the program is like：

```
---------------------------------------------------------------------------------------
void RTM_init(void)
{
    RTAH = 11;    // set the hour for the alarm clock
      RTAM = 30; //set the minute for the alarm clock
    RTAS = 00;    // set the second for the alarm clock
    RTCON |= SCE(1)|MCE(1)|HCE(1);      // enables hour, minute and second compare
}
void RTC_ISR (void) interrupt 13
{
    if(RTCIF & RTC_AF)                  // alarm clock interrupt
    {
        RTCIF = RTC_AF;
        // alarm clock interrupt service routine
    }
}
---------------------------------------------------------------------------------------
```

◆ **RTC initialization**

RTC initialization program is like：

```
--------------------------------------------------------------------------------
void RTC_init(void)
{
    P52F = 3;
    P53F = 3;
    CKCON |= XLCKE;
    while(!(CKCON & XLSTA));
    RTCON = RTCE(1) | MSE(1) | HSE(1);   // enable RTC, millisecond interrupt and half second
                                         // interrupt
    Delay_50us(6);        //time writing must after enable RTC 300us, otherwise time writing
                                          //would be invalid
    RTC_WriteHour(11);      // write the hour
    RTC_WriteMinute(29);    // write the minute
    RTC_WriteSecond(0);     // write the second
    RTM_init();     // set the alarm clock
    RTMSS = 0;     // set the time for millisecond interrupt
    INT8EN = 1;    // enable RTC interrupt
}
void RTC_ISR (void) interrupt 13
{
    if(RTCIF & RTC_MF)                  // millisecond interrupt
    {
        RTCIF = RTC_MF;

        // millisecond interrupt service routine


    }
    if(RTCIF & RTC_HF)                  // half second interrupt
    {
       RTCIF = RTC_HF;
        // half second interrupt service routine
    }
    if(RTCIF & RTC_AF)                  //   service program
    {
        RTCIF = RTC_AF;
        // alarm clock interrupt service routine
    }
}
--------------------------------------------------------------------------------
```

15 General Purpose Input/Output(GPIO) and Alternate Functions

15.1 Function Introduction

There are at most 62 I/O pins for CA51F4 Series chip package and all of the pins are alternate function pins. They can not only be independently programmed as input/output port, be also be configured as pins for other functions. For each pin, there is a function register PnxF(corresponding to pin Pnx, n=0、1、2、3、4、5, stands for P0、P1、P2、P3、P4、P5，x=0~7, stands for Pn.0~Pn.7，when n=5，x=0~5). Users can configure the main function and other options by setting register PnxF. For more information please refer to the register description.

Main features of GPIO：

- High impedance mode configurable
- The pull-up can be set independently for the I/O structure
- Open-drain or push-pull output can be selected for the output mode
- The data output latch can be read/written/revised
- Supports 1.8~5.5V voltage

Important reminder: The input voltage of all GPIO pins should not be higher than the voltage of VDD pins, otherwise it may cause abnormal chip operation.

The Figure 15-1-1 shows GPIO Push-pull Mode Structure。

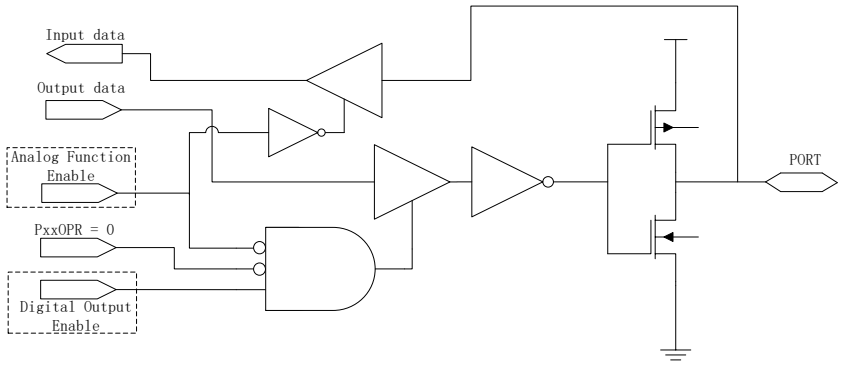


Figure 15-1-1 I/O Push-pull Mode Structure

The Figure 15-1-2 shows GPIO Open-drain Mode Structure。

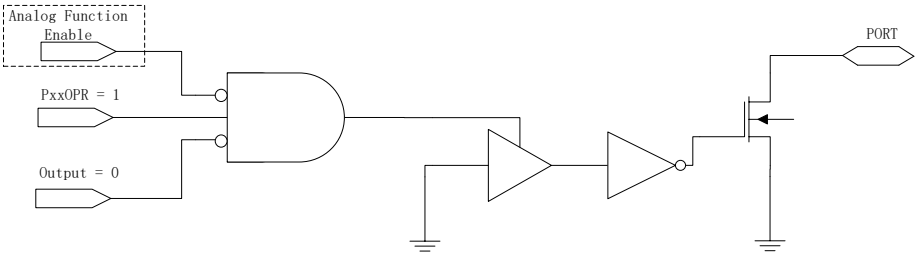


Figure 15-1-2 I/O Open-drain Mode Structure

The Figure 15-1-3 shows GPIO Pull-up Mode Structure。

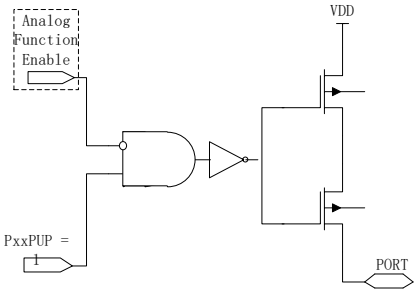


Figure 15-1-3 I/O Pull-down Mode Structure

15.2 Pin Register Description

Table 15-2-1 Register P0

80H	7	6	5	4	3	2	1	0
P0	P07	P06	P05	P04	P03	P02	P01	P00
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	P0x	Data register for pin P0x，valid when the pin function is set to GPIO 0：P0x is low level when the pin is set to input; when the pin set to output,P0x outputs low level signal 1：P0x is high level when the pin is set to input; when the pin set to output,P0x outputs high level signal

Table 15-2-2 Register P1

90H	7	6	5	4	3	2	1	0

P1	P17	P16	P15	P14	P13	P12	P11	P10
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	P1x	Data register for pin P1x， valid when the pin function is set to GPIO 0：P1x is low level when the pin is set to input; when the pin set to output,P1x outputs low level signal 1：P1x is high level when the pin is set to input; when the pin set to output,P1x outputs high level signal

Table 15-2-3 Register P2

A0H	7	6	5	4	3	2	1	0
P2	P27	P26	P25	P24	P23	P22	P21	P20
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~1	-	-
0	P2x	Data register for pin P2x，valid when the pin function is set to GPIO 0：P2x is low level when the pin is set to input; when the pin set to output,P2x outputs low level signal 1：P2x is high level when the pin is set to input; when the pin set to output,P2x outputs high level signal

Table 15-2-4 Register P3

B0H	7	6	5	4	3	2	1	0
P3	P37	P36	P35	P34	P33	P32	P31	P30
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~1	-	-
0	P3x	Data register for pin P3x, valid when the pin function is set to GPIO 0：P3x is low level when the pin is set to input; when the pin set to output,P3x outputs low level signal 1：P3x is high level when the pin is set to input; when the pin set to output,P3x outputs high level signal

Table 15-2-5 Register P4

C0H	7	6	5	4	3	2	1	0
P4	P47	P46	P45	P44	P43	P42	P41	P40
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	P4x	Data register for pin P4x， valid when the pin function is set to GPIO 0：P4x is low level when the pin is set to input; when the pin set to output,P4x outputs low level signal 1：P4x is high level when the pin is set to input; when the pin set to output,P4x outputs high level signal

Table 15-2-6 Register P5

D8H	7	6	5	4	3	2	1	0
P5	-	-	P55	P54	P53	P52	P51	P50
R/W	-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	-	-	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	P5x	Data register for pin P5x，valid when the pin function is set to GPIO 0：P5x is low level when the pin is set to input; when the pin set to output, P5x outputs low level signal 1：P5x is high level when the pin is set to input; when the pin set to output, P5x outputs high level signal

Table 15-2-7 Pin Function Control Register

8000H	7	6	5	4	3	2	1	0
P00F	P00PUP	-	P00OPR	-	-	P00S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8001H	7	6	5	4	3	2	1	0
P01F	P01PUP	-	P01OPR	-	-	P01S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8002H	7	6	5	4	3	2	1	0
P02F	P02PUP	-	P02OPR	-	-	P02S		
R/W	R/W	-	R/W	-	-	R/W		

Initial Value	0	-	0	-	-	0	0	0
8003H	7	6	5	4	3	2	1	0
P03F	P03PUP	-	P03OPR	-	-	P03S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8004H	7	6	5	4	3	2	1	0
P04F	P04PUP	-	P04OPR	-	-	P04S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8005H	7	6	5	4	3	2	1	0
P05F	P05PUP	-	P05OPR	-	-	P05S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	1	1
8006H	7	6	5	4	3	2	1	0
P06F	P06PUP	-	P06OPR	-	-	P06S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	1	1
8007H	7	6	5	4	3	2	1	0
P07F	P07PUP	-	P07OPR	-	-	P07S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	1	1
8008H	7	6	5	4	3	2	1	0
P10F	P10PUP	-	P10OPR	-	-	P10S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8009H	7	6	5	4	3	2	1	0
P11F	P11PUP	-	P11OPR	-	-	P11S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
800AH	7	6	5	4	3	2	1	0
P12F	P12PUP	-	P12OPR	-	-	P12S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
800BH	7	6	5	4	3	2	1	0

P13F	P13PUP	-	P13OPR	-	-	P13S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
800CH	7	6	5	4	3	2	1	0
P14F	P14PUP	-	P14OPR	-	-	P14S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
800DH	7	6	5	4	3	2	1	0
P15F	P15PUP	-	P15OPR	-	-	P15S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
800EH	7	6	5	4	3	2	1	0
P16F	P16PUP	-	P16OPR	-	-	P16S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
800FH	7	6	5	4	3	2	1	0
P17F	P17PUP	-	P17OPR	-	-	P17S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8010H	7	6	5	4	3	2	1	0
P20F	P20PUP	-	P20OPR	-	-	P20S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8011H	7	6	5	4	3	2	1	0
P21F	P21PUP	-	P21OPR	-	-	P21S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8012H	7	6	5	4	3	2	1	0
P22F	P22PUP	-	P22OPR	-	-	P22S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8013H	7	6	5	4	3	2	1	0
P23F	P23PUP	-	P23OPR	-	-	P23S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8014H	7	6	5	4	3	2	1	0
P24F	P24PUP	-	P24OPR	-	-	P24S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8015H	7	6	5	4	3	2	1	0
P25F	P25PUP	-	P25OPR	-	-	P25S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8016H	7	6	5	4	3	2	1	0
P26F	P26PUP	-	P26OPR	-	-	P26S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8017H	7	6	5	4	3	2	1	0
P27F	P27PUP	-	P27OPR	-	-	P27S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8018H	7	6	5	4	3	2	1	0
P30F	P30PUP	-	P30OPR	-	-	P30S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8019H	7	6	5	4	3	2	1	0
P31F	P31PUP	-	P31OPR	-	-	P31S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

801AH	7	6	5	4	3	2	1	0
P32F	P32PUP	-	P32OPR	-	-	P32S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

801BH	7	6	5	4	3	2	1	0
P33F	P33PUP	-	P33OPR	-	-	P33S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

801CH	7	6	5	4	3	2	1	0
P34F	P34PUP	-	P34OPR	-	-	P34S		

R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
801DH	7	6	5	4	3	2	1	0
P35F	P35PUP	-	P35OPR	-	-	P35S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8020H	7	6	5	4	3	2	1	0
P40F	P40PUP	-	P40OPR	-	-	P40S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8021H	7	6	5	4	3	2	1	0
P41F	P41PUP	-	P41OPR	-	-	P41S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8022H	7	6	5	4	3	2	1	0
P42F	P42PUP	-	P42OPR	-	-	P42S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8023H	7	6	5	4	3	2	1	0
P43F	P43PUP	-	P43OPR	-	-	P43S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8024H	7	6	5	4	3	2	1	0
P44F	P44PUP	-	P44OPR	-	-	P44S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8025H	7	6	5	4	3	2	1	0
P45F	P45PUP	-	P45OPR	-	-	P45S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0
8026H	7	6	5	4	3	2	1	0
P46F	P46PUP	-	P46OPR	-	-	P46S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8027H	7	6	5	4	3	2	1	0
P47F	P47PUP	-	P47OPR	-	-	P47S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8028H	7	6	5	4	3	2	1	0
P50F	P50PUP	-	P50OPR	-	-	P50S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

8029H	7	6	5	4	3	2	1	0
P51F	P51PUP	-	P51OPR	-	-	P51S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

802AH	7	6	5	4	3	2	1	0
P52F	P52PUP	-	P52OPR	-	-	P52S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

802BH	7	6	5	4	3	2	1	0
P53F	P53PUP	-	P53OPR	-	-	P53S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

802CH	7	6	5	4	3	2	1	0
P54F	P54PUP	-	P54OPR	-	-	P54S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	0	0

802DH	7	6	5	4	3	2	1	0
P55F	P55PUP	-	P55OPR	-	-	P55S		
R/W	R/W	-	R/W	-	-	R/W		
Initial Value	0	-	0	-	-	0	1	1

Note：

PXnF stands for P00F~P55F，X=0,1,2,3,4,5 stands for P0~P5，n=0,1,2,...,7(when X=5,n<=5)。

位编号	位符号	说明
7	PnxPUP	Pull-up resistor enable control 0: disable pull-up resistor 1: enable pull-up resistor
6	-	-
5	PnxOPR	Open-drain enable control, only valid when the pin is set to be digital output 0: disable open-drain

		1: enable open-drain

Table 15-2-8 Register PXnC

8120H	7	6	5	4	3	2	1	0
P00C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8121H	7	6	5	4	3	2	1	0
P01C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8122H	7	6	5	4	3	2	1	0
P02C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8123H	7	6	5	4	3	2	1	0
P03C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8124H	7	6	5	4	3	2	1	0
P04C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8125H	7	6	5	4	3	2	1	0
P05C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8126H	7	6	5	4	3	2	1	0
P06C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8127H	7	6	5	4	3	2	1	0
P07C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-

Initial Value	-	1	-	-	-	-	-	-
8128H	7	6	5	4	3	2	1	0
P10C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8129H	7	6	5	4	3	2	1	0
P11C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
812AH	7	6	5	4	3	2	1	0
P12C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
812BH	7	6	5	4	3	2	1	0
P13C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
812CH	7	6	5	4	3	2	1	0
P14C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
812DH	7	6	5	4	3	2	1	0
P15C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
812EH	7	6	5	4	3	2	1	0
P16C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
812FH	7	6	5	4	3	2	1	0
P17C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8130H	7	6	5	4	3	2	1	0

P20C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8131H	7	6	5	4	3	2	1	0
P21C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8132H	7	6	5	4	3	2	1	0
P22C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8133H	7	6	5	4	3	2	1	0
P23C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8134H	7	6	5	4	3	2	1	0
P24C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8135H	7	6	5	4	3	2	1	0
P25C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8136H	7	6	5	4	3	2	1	0
P26C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8137H	7	6	5	4	3	2	1	0
P27C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8138H	7	6	5	4	3	2	1	0
P30C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

8139H	7	6	5	4	3	2	1	0
P31C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

813AH	7	6	5	4	3	2	1	0
P32C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

813BH	7	6	5	4	3	2	1	0
P33C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

813CH	7	6	5	4	3	2	1	0
P34C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

813DH	7	6	5	4	3	2	1	0
P35C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

813EH	7	6	5	4	3	2	1	0
P36C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

813FH	7	6	5	4	3	2	1	0
P37C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

8140H	7	6	5	4	3	2	1	0
P40C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

8141H	7	6	5	4	3	2	1	0
P41C	-	SMIT_EN	-	-	-	-	-	-

R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8142H	7	6	5	4	3	2	1	0
P42C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8143H	7	6	5	4	3	2	1	0
P43C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8144H	7	6	5	4	3	2	1	0
P44C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8145H	7	6	5	4	3	2	1	0
P45C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8146H	7	6	5	4	3	2	1	0
P46C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8147H	7	6	5	4	3	2	1	0
P47C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8148H	7	6	5	4	3	2	1	0
P50C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-
8149H	7	6	5	4	3	2	1	0
P51C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

814AH	7	6	5	4	3	2	1	0
P52C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

814BH	7	6	5	4	3	2	1	0
P53C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

814CH	7	6	5	4	3	2	1	0
P54C	SINK_EN	SMIT_EN	-	-	-	-	-	-
R/W	R/W	R/W	-	-	-	-	-	-
Initial Value	0	1	-	-	-	-	-	-

814DH	7	6	5	4	3	2	1	0
P55C	-	SMIT_EN	-	-	-	-	-	-
R/W	-	R/W	-	-	-	-	-	-
Initial Value	-	1	-	-	-	-	-	-

Note：

PXnC stands for P00C~P55C，X=0,1,2,3,4,5 stands for P0~P5，n=0,1,2,...,7(when X=5, n=0,1,2,...,5)。

Bit number	Bit Symbol	Description
7	SINK_EN	For 1 the sink current enabled and for 0 not
6	SMIT_EN	1: the input SMIT enabled 0: the inverter enabled
5~0	-	-

Table 15-2-9 Register RMCTL

811DH	7	6	5	4	3	2	1	0
RMCTL	-	-	-	-	-	-	SINK_SEL	
R/W	-	-	-	-	-	-	R/W	R/W
Initial Value	-	-	-	-	-	-	0	0

Bit number	Bit Symbol	Description
7~2	-	-
1~0	SINK_SEL	Pin P54 to choose sink current： 00：125mA sink current 01：225mA sink current 10：315mA sink current 11：400mA sink current

Table 15-2-10 Pin Alternate Function Mapping

Value / Name	0	1	2	3	4	5	6	7
P00S	High impedance	Digital input	Digital output	High impedance	High impedance	High impedance	VP3	High impedance
P01S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	VP2	High impedance
P02S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	VP1	High impedance
P03S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	CUP1	High impedance
P04S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	CUP2	High impedance
P05S	High impedance	digital input	digital output	I2C_SCL	High impedance	TK15	LCD_S29	High impedance
P06S	High impedance	digital input	digital output	I2C_SDA	PWM1	TK0	TLCOM	High impedance
P07S	High impedance	digital input	digital output	SWIM	High impedance	High impedance	LCD_S30	High impedance
P10S	High impedance	digital input	digital output	High impedance	High impedance	TK_CAP	LCD_S0	High impedance
P11S	High impedance	digital input	digital output	High impedance	High impedance	TK1	LCD_S1	High impedance
P12S	High impedance	digital input	digital output	High impedance	High impedance	TK2	LCD_S2	High impedance
P13S	High impedance	digital input	digital output	High impedance	High impedance	TK3	LCD_S3	High impedance
P14S	High impedance	digital input	digital output	High impedance	High impedance	TK4	LCD_S4	High impedance
P15S	High impedance	digital input	digital output	High impedance	High impedance	TK5	LCD_S5	High impedance
P16S	High impedance	digital input	digital output	High impedance	High impedance	TK6	LCD_S6	High impedance
P17S	High impedance	digital input	digital output	High impedance	High impedance	TK7	LCD_S7	High impedance
P20S	High impedance	digital input	digital output	High impedance	High impedance	TK8	LCD_S8	High impedance
P21S	High impedance	digital input	digital output	UART1_RX	High impedance	TK9	LCD_S9	High impedance
P22S	High impedance	digital input	digital output	UART1_TX	High impedance	TK10	LCD_S10	High impedance
P23S	High impedance	digital input/T1	digital output	High impedance	High impedance	TK11	LCD_S11	High impedance

P24S	High impedance	digital input/T2	digital output	High impedance	High impedance	TK12	LCD_S12	High impedance
P25S	High impedance	digital input/T2EX	digital output	T2CP	High impedance	TK13	LCD_S13	High impedance
P26S	High impedance	digital input/T0	digital output	High impedance	High impedance	TK14	LCD_S14	High impedance
P27S	High impedance	digital input	digital output	ADC0	High impedance	High impedance	LCD_S15	High impedance
P30S	High impedance	digital input	digital output	High impedance	PWM2	CLK_IN	LCD_S16	High impedance
P31S	High impedance	digital input	digital output	ADC1	High impedance	High impedance	LCD_S17	High impedance
P32S	High impedance	digital input	digital output	ADC2	High impedance	High impedance	LCD_S18	High impedance
P33S	High impedance	digital input	digital output	ADC3	High impedance	High impedance	LCD_S19	High impedance
P34S	High impedance	digital input	digital output	ADC4	High impedance	High impedance	LCD_S20	High impedance
P35S	High impedance	digital input	digital output	ADC5	High impedance	High impedance	LCD_S21	High impedance
P36S	High impedance	digital input	digital output	ADC6	High impedance	High impedance	LCD_S22	High impedance
P37S	High impedance	digital input	digital output	ADC7	ADC_VREF	High impedance	LCD_S23	High impedance
P40S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_S24	High impedance
P41S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_S25	High impedance
P42S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_S26	High impedance
P43S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_S27	High impedance
P44S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_S28	High impedance
P45S	High impedance	digital input	digital output	High impedance	High impedance	LCD_S31	LCD_C4	High impedance
P46S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_C3	High impedance
P47S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_C2	High impedance
P50S	High impedance	digital input	digital output	High impedance	High impedance	High impedance	LCD_C1	High impedance
P51S	High	digital input	digital output	High	High	High	LCD_C0	High

	impedance			impedance	impedance	impedance		impedance
P52S	High impedance	digital input	digital output	32K_O	High impedance	High impedance	High impedance	High impedance
P53S	High impedance	digital input	digital output	32K_I	High impedance	High impedance	High impedance	High impedance
P54S	High impedance	digital input	digital output	High impedance	PWM0/REM	High impedance	High impedance	High impedance
P55S	High impedance	digital input	digital output	RESET	High impedance	High impedance	High impedance	High impedance

15.3 Pin control Example

Set the Pin function

For instance, P00 is set to be push-pull output, the program is like：

```
---------------------------------------------------------------------------------------
P00F = 2;
---------------------------------------------------------------------------------------
```

P00 is set to be open-drain output, the program is like：

```
---------------------------------------------------------------------------------------
P00F = (1<<5)|2;
---------------------------------------------------------------------------------------
```

P00 is set to be open-drain output with pull-up enabled, the program is like：

```
---------------------------------------------------------------------------------------
P00F = (1<<7) | (1<<5) | 2;
---------------------------------------------------------------------------------------
```

P00 is set to be input with pull-up enabled, the program is like：

```
---------------------------------------------------------------------------------------
P00F = (1<<7) | 1;
---------------------------------------------------------------------------------------
```

16 Universal Serial Receiver/Transmitter(UART)

16.1 UART

16.1.1 Function Introduction

UART is a full duplex synchronous/asynchronous serial data transceiver，UART receiver includes a one byte buffer，There are four working modes for UART0 which is shown as the table 16-1-1-1.

Table 16-1-1-1 UART Communication Mode

SM1	Mode	Description	Baud rate
0	A	9 bit asynchronous mode	CPUCLK/(32*(1024-SREL))
1	B	8 bit asynchronous mode	CPUCLK/(32*(1024-SREL))

There is a special baud generator in UART in which baud is set by register SRELL , SRELH.

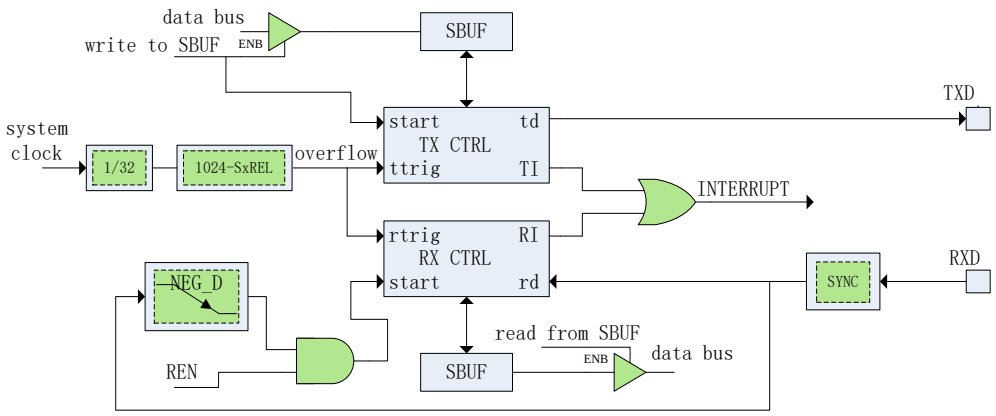


Figure 16-1-1-1 UART Schematic diagram of working principle

- **ModeA**

UART sends and receives 9 bit data asynchronously and simultaneously in modeA. Writing data to register S0BUF will starts UART transmission. The first bit transmitted is the start bit (which is 0), and then the 9 bit data follows (with the least significant bit transmitted first). The ninth data bit is the TB81 of register S0CON. The last bit transmitted is the stop bit (which is 1). When UART is used as receiver, it is synchronized by detecting the falling edges of Pin RX. The lower 8 bit data will be stored in register SBUF after the transmission is completed with the 9th data bit stored in RB81.

- **ModeB**

UART sends and receives 9 bit data which is different to ModeA, valid stop bit is stored in stop bit. Other function is same to ModeA.

- **UART Multi-computer Communication**

Multi-computer Communication can also be realized by UART0 in modeA. If SM2x of register SCON is set to 1, only when the 9th data is 1 (RB81=1) ,the slave will generate receive interrupt, which makes multi-computer communication possible. The slaves can set their SM2x to 1 and host set the 9th data bit to 1 when it transfers address to the slaves. All the slaves will generate receive interrupt and the slaves' software then compare the address received to their own addresses. It the address matches, the matched slave will set SM2x=0. The host then set the 9th data bit to 0 for the following data transmission. Due to the other slaves remain SM2x = 1, thus only the address matched slave will generate receive interrupt.

16.1.2 UART Register Description

Table 16-1-2-1 Register SCON

9AH	7	6	5	4	3	2	1	0
SCON	SM1	U1IE	SM21	REN1	TB81	RB81	TI1	RI1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7	SM1	UART0 mode selection, for more information please refer to Table 17-1-1-1
6	U1IE	UART interrupt enable control, 1 enables
5	SM21	Multi-computer communication enable control, 1 enables
4	REN1	Serial receive enable control, 1 enables
3	TB81	The 9th data bit to transmit It will be transmitted as the 9th bit of the data in mode A and it is controlled by the software(For instance, parity check or multi-computer communication)
2	RB81	The 9th bit of the data received It will be used for UART to receive the 9th bit of the data in mode A. It is the stop bit in mode B
1	TI1	Transmit interrupt flag, 1 indicates the interrupt, 0 cleared by writing 1 to it
0	RI1	Receive interrupt flag, 1 indicates the interrupt,0 cleared by writing 1 to it

Table 16-1-2-2 Register SBUF

9BH	7	6	5	4	3	2	1	0
SBUF	SBUF[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	SBUF	Receiver/Transmitter buffer Writing data to SBUF will starts the data transmission Reading SBUF will reads the data received

Table 16-2-2-3 Register SRELL,SRELH

9CH	7	6	5	4	3	2	1	0
SRELL	SRELL[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
9DH	7	6	5	4	3	2	1	0
SRELH	-	-	-	-	-	-	SREL[9:8]	
R/W	-	-	-	-	-	-	R/W	
Initial Value	-	-	-	-	-	-	0	0

Bit number	Bit Symbol	Description
9~0	SREL	Baud rate configuration register The baud rate is CPUCLK/(32 * (1024 - SREL))

Table 16-1-2-7 Register UDCKS

8118H	7	6	5	4	3	2	1	0
UDCKS	UDE	-	-	DNUM[4:0]				
R/W	R/W	-	-	R/W				
Initial Value	0	-	-	0	0	0	0	0

Bit number	Bit Symbol	Description
7	UDE	Fast baud rate setting enable control, 1 enables it ***Note：*** *When UDE=0，UART baud set by original setting, UDE=1，UART baud is set by DNUM.*
6~5	-	-
4~0	DNUM	fast baud setting register, enabled only when UDE=1 DNUM>=0 must be satisfied when sending；DNUM>=6 when receive BRx = F_{sys}*(1/((DNUM+1)*(1024-SRELL)))

17 I²C Interface

17.1 Function Introduction

I²C modules enables the chip to communicate with peripheral I²C devices by serial transmission standard which complies with standard I²C specification. It can be set to either slave or master and configured to standard/fast/high speed mode.

17.2 I²C Main Features

- Simple but strong communication port, bi-directional bus with 2 wires
- Slave/Master mode configurable
- Able to operate in receiver/transmitter mode
- 7 bit slave address
- Supports multimaster's arbitration
- Broadcast function supported

17.3 I²C Function Description

I²C modules supports I²C standard bus specification. I²C bus includes 2 wires to transfer data among devices, one is SCL(Serial Clock) and the other is SDA(Serial Data), as Figure 19-3-1 shows. Since the it is open-drain port for I²C, there must be pull-up resistor on I²C bus. The pull-up resistor can be connected externally or enabled internally. Each device that connects to the bus has its own 7-bit address.

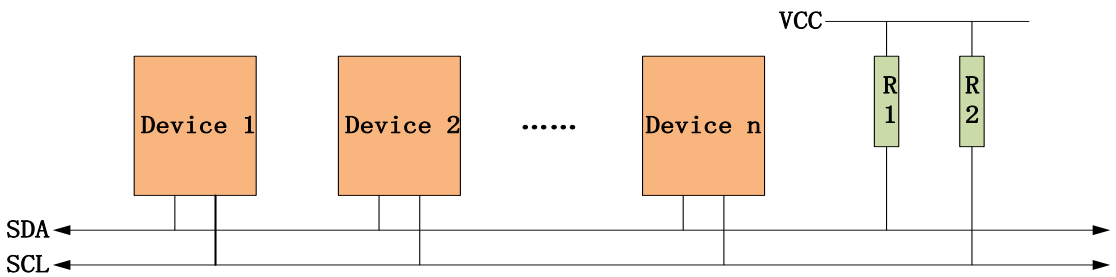


Figure 17-3-1 I2C Bus Interconnection

I²C module principle is as Figure 17-3-2 shows.

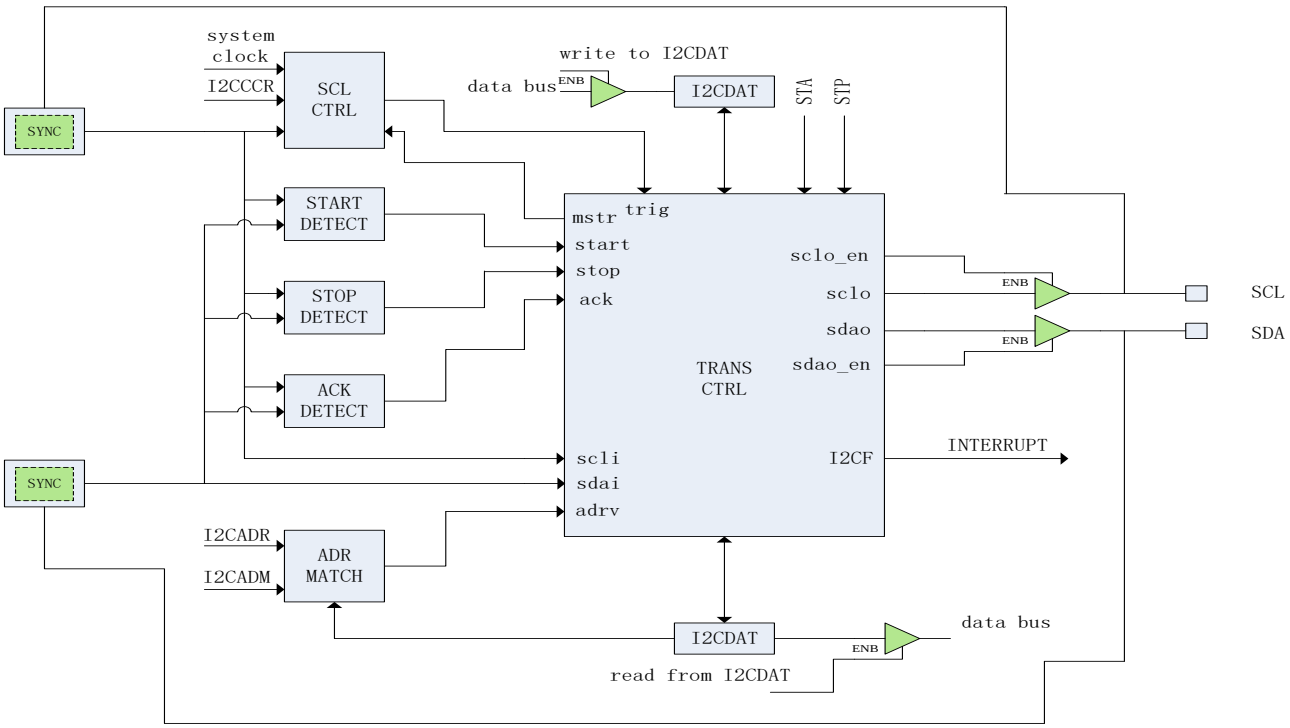


Figure 17-3-2 I²C Module Schematic

- **I²C Mode Selection**

I²C can operates in the following 4 modes：slave transmit mode, slave receive mode, master transmit mode, master receive mode. I²C operates in slave mode by default. I²C changes to master mode after the START signal generated and returns slave mode when the arbitration fails or STOP signal is generated.

- **I²C Bus Data Transmission Pattern**

There are usually 4 stages for the standard I²C communication: START signal, slave address transfer, data transmission and STOP signal. The data transmitted on I²C bus is always 8 bits with the most significant bit sent first. There must be a ACK following every one byte data. However, there is no byte limits for the data transmission. The master sends STOP signal after the transmission is over and terminates the communication.

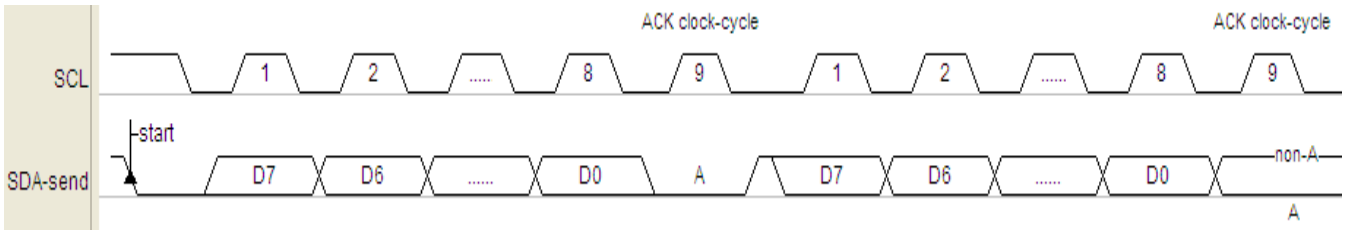


Figure 17-1-3 I2C Bus Data Transmission Format

- **Communication Process**

I²C enables the data transmission and generates the clock signal in Master mode. The serial data transmission always begins with START signal and ends with STOP signal. Both START and STOP signals are generated by the software in master mode. Setting STA=1 generates START signal and setting STP=1 generates STOP signal,

I²C can distinguish its address (7 bits) and the broadcast address in slave mode. Software can enable/disable its ability to recognize broadcast address by setting GCE.

Both address and data are transmitted in bytes. The address will be sent by the master after the START signal. The receiver must reply with a ACK signal in the 9th clock cycle after one byte information is transferred. The ACK can be set by AAK while it must be set before the one byte information transfer completes. When the one byte information is received, the ACK signal will be generated automatically.

Every time when one byte data is received/transmitted or arbitration fails (and etc.) there will be an interrupt flag I2CF. The status of the event will be indicated by register I2CSTA (for more information please refer to register I2CSTA). The software decides the next operation according to the status of the event when interrupt occurs. Clearing the interrupt flag I2CF will start the next operation.

When there occurs interrupt I2CF, if SHD=1, SCL will be set to low by slave before I2CF is cleared. After the master detects that SCL is released, it master will then continue the next operation. On the other hand, if SHD=0, SCL will not be set to low by the slave, which makes it compatible with applications when the master I²C is simulated by software. Thus, the master's software must wait long enough so that the slave can deal with the response to every one byte data.

- **I²C clock setting**

When I²C is set as slave, the master outputs SCL clock, and it has nothing to do with the slave's clock configuration. As for the slave, SCL must remain low for at least 6.5 system clock cycles periods and high for

at least 2.5 system clock cycles periods. In the end, the frequency of SCL sent by external master can be at most 1/9 of the system clock.

17.4 Register Description

Table 17-4-1 Register I2CCON

B1H	7	6	5	4	3	2	1	0
I2CCON	I2CE	I2CIE	STA	STP	SHD	AAK	CBSE	STFE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7	I2CE	I^2C module enable control, 1 enables it
6	I2CIE	I^2C interrupt enable control, 1 enables it1
5	STA	I2C START signal transfer control,valid when it is 1, it will be cleared automatically when START signal detected
4	STP	I2C STOP signal transfer control, valid when it is 1, it will be cleared automatically when STOP signal detected
3	SHD	When it is 1, if I2CF=1, I2CF will make SCL remain low after SCL becomes low
2	AAK	I2C ACK signal transfer control, 1 enables it Note： When I2C is configured as slave, this bit must be set to 1 beforehand, otherwise even the address matches it will not reply ACK
1	CBSE	CBUS compatible enable control When it is set to 1, the ACK will be ignored during the transmission to be compatible with CBUS bus.
0	STFE	When STFE=1, I2CF will be set to 1 if I^2C module detects the START signal

Table 17-4-2 Register I2CADR

B2H	7	6	5	4	3	2	1	0
I2CADR	GCE	I2CADRL[6:0]						
R/W	R/W	R/W						
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7	GCE	Broadcast address recognition(00H)enable control, 1 enables it
6~0	I2CADRL	I2C slave address, only valid when it operates as slave Note：

		(when AAK=1) when the address is 7 bits and the higher 7 bits of first received address matches I2CADR, reply with ACK and enters slave mode

Table 17-4-3 Register I2CADM

B3H	7	6	5	4	3	2	1	0
I2CADM	SPFE	I2CADML[6:0]						
R/W	R/W	R/W						
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7	SPFE	When SPFE=1, I2CF will be set to 1 if I2C module detects the STOP signal
6~0	I2CADML	I2C address mask by bit control, valid only when it operates as slave When I2CADM[n](n=0~6)=1, the corresponding address bit I2CADR[n] will not be compared (which means no matter what is received, it is seen as matched)

Table 17-4-4 Register I2CCCR

B4H	7	6	5	4	3	2	1	0
I2CCCR	SMPDIV[2:0]			I2CCKD[4:0]				
R/W	R/W							
Initial Value	0	0	1	0	0	0	0	0

Bit number	Bit Symbol	Description
7~5	SMPDIV	I2c sampling clock setting,sampling clock is equal to work clock divided by 2^smpdiv: 000：$F_{sample}=F_{i2cclk}$ 001：$F_{sample}=F_{i2cclk}/2$ 010：$F_{sample}=F_{i2cclk}/4$ … 111：$F_{sample}=F_{i2cclk}/128$
4~0	I2CCKD	I2c scl output clock frequency setting, scl output clock is equal to sampling clock divided by i2cckd+1: $F_{scl}= F_{sample}$ /(I2CCKD +1)) Note： *1.* when *SMPDIV= 0*， if *I2CCKD<9*， calculate as 9 automatically *2.* when *SMPDIV>0*， if *I2CCKD<7*， calculate as 97automatically *Note:*

		1. when I2CCCR[7:5] = 0，if a value less than 9 be wrote to I2CCCR[4:0], calculate as 9 automatically *2. when I2CCCR[7:5] =>0，if a value less than 7 be wrote to I2CCCR[4:0], calculate as 7 automatically*

Table 17-4-5 Register I2CDAT

B5H	7	6	5	4	3	2	1	0
I2CDAT	I2CDAT[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	I2CDAT	Data buffer for receiving/transmission *Note*： *When I2CF is 1, it is recommended to make I2CF remain 1 when users overwrite/read I2CDAT. I2CF should be cleared after the process is over, and then the transmission continues so that there will be no transmission errors.*

Table 17-4-6 Register I2CSTA

B6H	7	6	5	4	3	2	1	0
I2CSTA	I2CSTA[7:0]							
R/W	R							
Initial Value	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	Description
7~0	I2CSTA	I2C status register 00H: (master/slave) bus error 08H: (master/slave)START signal detected (valid only when STFE=1) 18H: (master)address and write bit sent, ACK signal received 20H: (master)address and write bit sent, no ACK signal received 28H: (master)one byte data received/transmitted, ACK signal detected 30H: (master)one byte data received/transmitted, no ACK signal detected 38H: (master)arbitration lost(master will change to slave after arbitration lost) 40H: (master)address and read bit transmitted, ACK signal received 48H: (master)address and read bit transmitted, no ACK signal received 60H: (slave)address and write bit received, with ACK signal is sent 70H: (master/slave)broadcast address received with ACK signal is sent(master/slave will become slave) 80H: (slave)one byte data received/transmitted, ACK signal detected 88H: (slave)one byte data received/transmitted, no ACK signal detected

		A0H: (master/slave)STOP signal detected(valid only when SPFE=1) A8H: (slave)address and read bit received, with ACK signal is sent F8H: (master/slave) bus is idle

Table 17-4-7 Register I2CFLG

B7H	7	6	5	4	3	2	1	0
I2CFLG	-	-	-	-	-	-	-	I2CF
R/W	-	-	-	-	-	-	-	R
Initial Value	-	-	-	-	-	-	-	0

Bit number	Bit Symbol	Description
7~1	-	-
0	I2CF	I2C interrupt flag, 1 indicates the interrupt, cleared by writing 1 to it Note： 1. I2CF will be set to 1 every time after a one-byte data or the address transmission completes (with ACK/NAK received/sent). 2. I2CF will be set to 1 when there is bus error. 3. If STFE=0, I2CF will not be set to 1 when START signal detected. 4. If SPFE=0, I2CF will not be set to 1 when STOP signal detected.

17.6 I²C Control Example

◆ **I²C as master**

For instance , the master sends 20 byte data to the slave cyclically, the program is like：

```
//I2CCON definition
#define I2CE(N)         (N<<7)
#define I2CIE(N)        (N<<6)
#define STA(N)          (N<<5)
#define STP(N)          (N<<4)
#define CKHD(N)         (N<<3)
#define AAK(N)          (N<<2)
#define CBSE(N)         (N<<1)
#define STFE(N)         (N<<0)
//I2CADR definition
#define     GCE(N)      (N<<7) //N = 0~1
//I2CFLG definition
```

```
#define I2CF                (1<<0)

#define I2C_ADDR        0xCA            //I2C slave machine address definition
unsigned char xdata WriteBuffer[20]={0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19};
void main(void)
{
unsigned char i;
    EA = 1;                                         //global interrupt start
/**********choose I2C port*****************************************************************/
    P06F = 3| (1<<7);
    P05F = 3| (1<<7);
/***********************************************************************************/
    I2CCON = I2CE(1) | I2CIE(0) | STA(0) | STP(0)| CKHD(1) | AAK(1)| CBSE(0) | STFE(1);
    I2CADR = GCE(0);
    I2CCCR = 0x4c;                                  //set I2C clock
    while(1)
    {
I2CCON |= STA(1);                   //I2C master send START signal
while(!(I2CFLG & I2CF));            //wait for interrupt signal generated
if(I2CSTA != 0x08)
{
    I2CFLG    |= I2CF;
    goto SEND_STOP;
}
I2CDAT = I2C_ADDR;                  //send I2C slave address+write bit
I2CFLG    |= I2CF;                  //clear interrupt signal
while(!(I2CFLG & I2CF));            // wait for interrupt signal generated
if(I2CSTA != 0x18)
{
    I2CFLG    |= I2CF;
    goto SEND_STOP;
}

I2CDAT = 0;                         // Address of the data register sent by the I2c master
I2CFLG    |= I2CF;                  // clear interrupt signal
while(!(I2CFLG & I2CF));            // wait for interrupt signal generated
if(I2CSTA != 0x28)
{
I2CFLG    |= I2CF;
    goto SEND_STOP;
}
for(i = 0; i < 20; i++)             //I2c master send 20 data
{
    I2CDAT =WriteBuffer[i];
```

```
        I2CFLG    |= I2CF;                    //clear interrupt signal
        while(!(I2CFLG & I2CF));             //wait for interrupt signal generated
        if(I2CSTA != 0x28)
        {
            I2CFLG    |= I2CF;
            goto SEND_STOP;
        }
    }
    SEND_STOP:
    I2CCON |= STP(1);                        //send STOP signal
    I2CFLG    |= I2CF;
    Delay_ms(100);
    }
    }
```

For instance, the master reads 20 byte data from the slave cyclically, the program is like：

```
    #define I2C_ADDR        0xCA            //I2C slave address definition
    unsigned char xdata ReadBuffer[20];
    void main(void)
    {
        unsigned char i;
        EA = 1;                                                    // global interrupt start
    /**********choose I2C port*****************************************************************/
        P06F = 3| (1<<7);
        P05F = 3| (1<<7);
    /*********************************************************************************************/
        I2CCON = I2CE(1) | I2CIE(0) | STA(0) | STP(0)| CKHD(1) | AAK(1)| CBSE(0) | STFE(1);
        I2CADR = GCE(0);
        I2CCCR = 0x4c;                                             // set I2C clock
        while(1)
        {
            I2CCON |= STA(1);                    // I2C master send START signal
            while(!(I2CFLG & I2CF));             // wait for interrupt signal generated

            if(I2CSTA != 0x08)
            {
                I2CFLG    |= I2CF;
                goto SEND_STOP;
            }
            I2CDAT = I2C_ADDR;                   // send I2C slave address+write bit
            I2CFLG    |= I2CF;                   // clear interrupt signal
```

```
while(!(I2CFLG & I2CF));                // wait for interrupt signal generated
if(I2CSTA != 0x18)
{
    I2CFLG   |= I2CF;
    goto SEND_STOP;
}

I2CDAT = 0;                             // Address of the data register sent by the I2c master
I2CFLG   |= I2CF;                       // clear interrupt signal
while(!(I2CFLG & I2CF));                // wait for interrupt signal generated
if(I2CSTA != 0x28)
{
    I2CFLG   |= I2CF;
    goto SEND_STOP;
}

I2CCON |= STA(1);                       // I2C master send START signal
I2CFLG   |= I2CF;                       // clear interrupt signal
while(!(I2CFLG & I2CF));                // wait for interrupt signal generated
if(I2CSTA != 0x08)
{
    I2CFLG   |= I2CF;
    goto SEND_STOP;
}

I2CDAT = I2C_ADDR+1;                    // send I2C slave address+write bit
I2CFLG   |= I2CF;                       // clear interrupt signal
while(!(I2CFLG & I2CF));                // wait for interrupt signal generated
if(I2CSTA != 0x40)
{
    I2CFLG   |= I2CF;
    goto SEND_STOP;
}
I2CCON |= AAK(1);                       //set answer bit

for(i = 0; i < 20; i++)
{
    I2CFLG   |= I2CF;                   // clear interrupt signal
    while(!(I2CFLG & I2CF));            // wait for interrupt signal generated
    if(I2CSTA != 0x28 && I2CSTA != 0x30)
    {
        I2CFLG   |= I2CF;
        goto SEND_STOP;
    }
```

```
                ReadBuffer[i] = I2CDAT;          //Read data into the data register
                if(i < 19)
                {
                    I2CCON |= AAK(1);            //Default ACK state if not the last byte
                }
                else
                {
                    I2CCON &= ~AAK(1);           // If it is the last byte, do not send ACK
                }
            }
SEND_STOP:
            I2CCON |= STP(1);                    //send STOP signal
            I2CFLG   |= I2CF;
            Delay_ms(100);
        }
}
```

---

18 PWM

18.1 PWM Function Introduction

CA51F4 series chip can include at most 3 channels PWM outputs. PWM period and duty cycle can be configured with 16 bit range.

18.2 PWM Function Description

There is a 16-bit counter for each PWM channel and the cycle is set by register PWMDIV. Register PWMDUT sets the corresponding PWM's duty cycle. PWM is enabled by register PWMEN with each bit of it corresponds to one channel in PWM. PWM pin can also output reversed phase by setting PWMTOG. There are multiple clock sources for PWM which is set for PWM pairs. It's corresponding control register is PWMCKS of PWMCON. The clock source can be selected by corresponding PWMCKS of register PWMCON, with the frequency division set by PWMCKD independently.

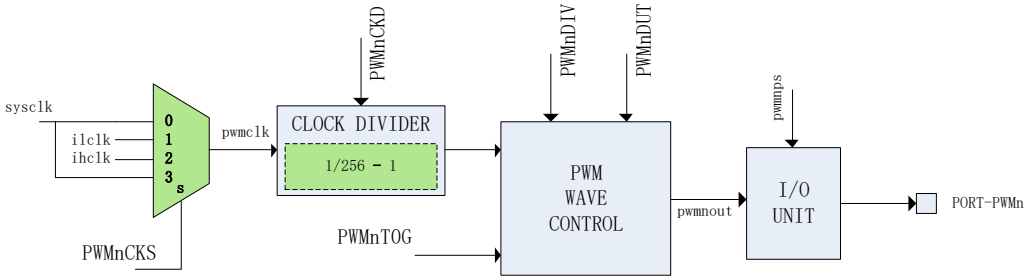


Figure 21-1-3 Schematic for PWM

- **PWM output waveform**

PWM counter starts to count from 0 after PWM is enabled. When the count valuer is less than PWMDUT, PWM pin outputs high level signal(PWMTOG=0), When the count is greater than PWMDUT, PWM pin outputs low level signal(PWMTOG=0). When the count equals PWMDIV, a PWM cycle completes and the PWM counter is cleared to start counting again. A PWM interrupt is generated at this time.

When the PWM waveform meet the condition PWMDIV>PWMDUT>0,it as shown in the figure.

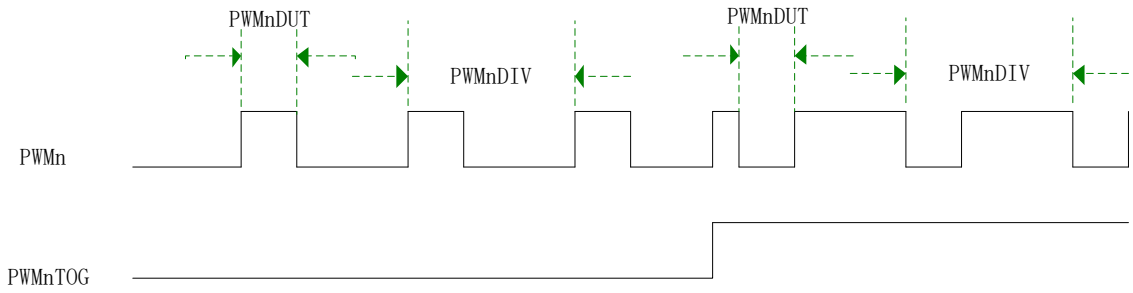


Figure 18-2-2 PWM output waveform

When PWMDIV=0, PWM pin output the PWM clock directly. If PWMCKD=0, PWM pin's output is the selected clock source. If PWMCKD! =0, PWM pin output is the selected clock source 1/ (PWMCKD+1) frequency clock signal;When PWMDIV is not 0 but PWMDUT=0, PWM pin outputs low level signal (PWMTOG＝0)；while PWMDUT>=PWMDIV>0, PWM pin outputs high level signal(PWMTOG＝0)。

- **PWM interrupt**

PWM interrupt in enabled by PWMMIE in register PWMCON. PWMIE enables the interrupt when PWM counter's count reaches the peak (which equals PWMDIV). The PWMIF register contains the interrupt flag bits for three channels.

18.3 PWM Register Description

Table 18-3-1 Register PWMEN

DAH	7	6	5	4	3	2	1	0
PWMEN	-	-	-	-	-	PWM2EN	PWM1EN	PWM0EN
R/W	-	-	-	-	-	R/W	R/W	R/W
Initial Value	-	-	-	-	-	0	0	0

Bit number	Bit Symbol	Description
7~3	-	-
2	PWM2EN	PWM2 enable control, 1 enables it
1	PWM1EN	PWM1 enable control, 1 enables it
0	PWM0EN	PWM0 enable control, 1 enables it

Table 18-3-2 Register PWMCON

DDH	7	6	5	4	3	2	1	0
PWMCON	PWM0IE	PWMTOG	-	-	-	PWMCKS[2:0]		
R/W	R/W	R/W	-	-	-	R/W		

Initial Value	0	0	-	-	-	-	0	0

Note： PWMCON is also register with index, INDEX=0/1/2 correspond to PWMCON0/1/2 respectively

Bit number	Bit Symbol	Description
7	PWMIE	PWM counter overflow interrupt enable control, 1 enables it
6	PWMTOG	PWM output negation enable control, 1 enables it
5~3	-	-
2~0	PWMCKS	PWM working clock selection 001：IRCH 010：IRCL 100：XOSCL Others：system clock

Table 18-3-3 Register PWMCKD

DEH	7	6	5	4	3	2	1	0
PWMCKD	PWMCKD[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Note：PWMCON is also register with index, INDEX=0/1/2 correspond to PWMCON0/1/2 respectively

Bit number	Bit Symbol	Description
7~0	PWMCKD	PWM working clock frequency division setting 00H： no division 01H： frequency divided by 2 02H： frequency divided by 3 ..... FEH： frequency divided by 255 FFH： frequency divided by 256

Table 18-3-4 Register PWMDIVL、PWMDIVH

DFH	7	6	5	4	3	2	1	0
PWMDIVL	PWMDIV[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
D1H	7	6	5	4	3	2	1	0
PWMDIVH	PWMDIV[15:8]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0

Note：PWMCON is also register with index, INDEX=0/1/2 correspond to PWMCON0/1/2 respectively

Bit number	Bit Symbol	Description
15~0	PWMDIV	PWM cycle configuration

Table 18-3-5 Register PWMDUTL、PWMDUTH

D2H	7	6	5	4	3	2	1	0
PWMDUTL	PWMDUT[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
D3H	7	6	5	4	3	2	1	0
PWMDUTH	PWMDUT[15:8]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
Note：PWMCON is also register with index, INDEX=0/1/2 correspond to PWMCON0/1/2 respectively								

Bit number	Bit Symbol	Description
15~0	PWMDUT	PWM duty cycle setting

Table 18-3-6 Register PWMIF

D4H	7	6	5	4	3	2	1	0
PWMIF	-	-	-	-	-	PWMIF2	PWMIF1	PWMIF0
R/W	-	-	-	-	-	R	R	R
Initial Value	-	-	-	-	-	0	0	0

Bit number	Bit Symbol	Description
7~3	-	-
2	PWMIF2	PWM2 interrupt flag, cleared when 1 is written to it
1	PWMIF1	PWM1 interrupt flag, cleared when 1 is written to it
0	PWMIF0	PWM0 interrupt flag, cleared when 1 is written to it

Table 18-3-7 Register PWMCMAX

DCH	7	6	5	4	3	2	1	0
PWMCMAX	PWMCMAX[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
Note：PWMCON is also register with index, INDEX=0/1/2 correspond to PWMCON0/1/2 respectively								

Bit number	Bit Symbol	Description
7~0	PWMCMX	PWM interrupt events number for each channel Interrupt event number=PWMCMX+1, For instance, when INDEX=0 and PWMCMX=7, then only when the interrupt trigger events happens 8 times the interrupt flag will be set to 1.

Table 18-3-8 Register PWMHS

8099H	7	6	5	4	3	2	1	0
PWMHS	-	-	-	-	-	PWMHS2	PWMHS1	PWMHS0
R/W	-	-	-	-	-	R/W	R/W	R/W
Initial Value	-	-	-	-	-	0	0	0

Bit number	Bit Symbol	Description
7~3	-	-
2	PWMHS2	PWM2 holding enable bit, 1 enables it
1	PWMHS1	PWM1 holding enable bit, 1 enables it
0	PWMHS0	PWM0 holding enable bit, 1 enables it

Table 18-3-9 Register PWMSBC

809CH	7	6	5	4	3	2	1	0
PWMSBC	-	-	-	-	-	PWMSBE2	PWMSBE1	PWMSBE0
R/W	-	-	-	-	-	R/W	R/W	R/W
Initial Value	-	-	-	-	-	0	0	0

Bit number	Bit Symbol	Description
7~3	-	-
2	PWMSBE2	PWM2 software break enable, 1enables it
1	PWMSBE1	PWM1 software break enable, 1enables it
0	PWMSBE0	PWM0 software break enable, 1enables it

Table 18-3-10 Register PWMBD

809DH	7	6	5	4	3	2	1	0
PWMBD	-	-	-	-	-	PWMBD2	PWMBD1	PWMBD0
R/W	-	-	-	-	-	R/W	R/W	R/W
Initial Value	-	-	-	-	-	0	1	0

Note：software and hardware breaking, voltage are controlled by this register

Bit number	Bit Symbol	Description
7~3	-	-
2	PWMBD2	PWM2 breaking voltage selection 0：low voltage holding when breaking 1：high voltage holding when breaking
1	PWMBD1	PWM1 breaking voltage selection 0：low voltage holding when breaking 1：high voltage holding when breaking
0	PWMFBD0	PWM0 breaking voltage selection

		0：low voltage holding when breaking 1：high voltage holding when breaking

18.4 PWM Control Example

◆ **Single channel PWM output**

For instance , PWM0 outputs 30KHz clock with duty cycle 50% and producing PWM interrupt, the program is like：

```
-------------------------------------------------------------------------------------
#define PIE(N)              (N<<7)
#define TOG(N)              (N<<6)
#define CKS_IH              (1<<0)
#define IHCKE               (1<<6)
#define PWMIF0              (1<<0)
#define PWM_CH0             0
#define PWMDIV_V    (16000000/30000)              // If the PWM clock is set to another clock frequency, modify the
                                                  //parameters accordingly
#define PWMDUT_V            (PWMDIV_V/2)      //duty cycle 50%
void PWM_init(void)
{
      CKCON |= IHCKE;                             //open IRCH clock
      INDEX = PWM_CH0;                           //set INDEX value corresponding with the PWM0
      PWMCON = PIE(1)|TOG(0)|CKS_IH;              //set IRCH as the clock source of PWM
      PWMCKD = TOG(0) | 0;
      P54F = 4;
      PWMDIVH     = (unsigned char)(PWMDIV_V>>8);          //set PWMDIV、PWMDUT
      PWMDIVL     = (unsigned char)(PWMDIV_V);
      PWMDUTH     =(unsigned char)(PWMDUT_V>>8);
      PWMDUTL     = (unsigned char)(PWMDUT_V);
      PWMEN = (1<<PWM_CH0);                         //PWM0 enable
      PWMCMX = 0;
      INT9EN = 1;
}
void INT9_ISR(void) interrupt 14
{
      if(PWMIF & PWMIF0)
      {
            PWMIF = PWMIF0;

      }
}
------------------------------------------------------------------------------------
```

◆ **PWM output clock example**

For instance , PWM0 outputs the IRCH clock , the program is like：

```
void PWM_init(void)
{
CKCON |= IHCKE;                    //open IRCH clock
INDEX = PWM_CH0;                   // set INDEX value corresponding with the PWM0
PWMCON = PIE(0)|TOG(0)|CKS_IH;     // set IRCH as the clock source of PWM
PWMCKD = TOG(0) | 0;
P54F = 4;
PWMDIVH    = 0;                    // set PWMDIV and PWMDUT to 0 to output the clock source
PWMDIVL    = 0;
PWMDUTH    = 0;
PWMDUTL    = 0;
PWMEN = (1<<PWM_CH0);              //PWM0 enable
}
```

---

19 Analog/Digital Converter (ADC)

19.1 Function Introduction

Analog/digital converter is a 12-bit successive approximation(SAR) ADC, with at most 8 input channels. The clock source for ADC is the system clock with frequency division configurable. There are ADC multiple reference voltages for ADC. When internal voltage is selected as the reference voltage, it can be used to test the power supply voltage for the chip and there will be correction to ensure the chip's consistency as well. The signal can be amplified/narrowed before the conversion when using ADC and OPAMP together.

19.2 Main Features

- 12 bit resolution
- 8 input channels at most
- Supports ADC interrupt
- ADC clock frequency division configurable
- Alternate reference voltage： internal reference voltage, VDD, external reference voltage
- Support VDD and reference ground voltage measurement
- Support automatic data correction function when selecting internal reference voltage
- Built-in operational amplifier, support detection signal reduction, reduction multiple is optional
- Input voltage range：VSS<=VIN<=VDD

19.3 Block Diagram

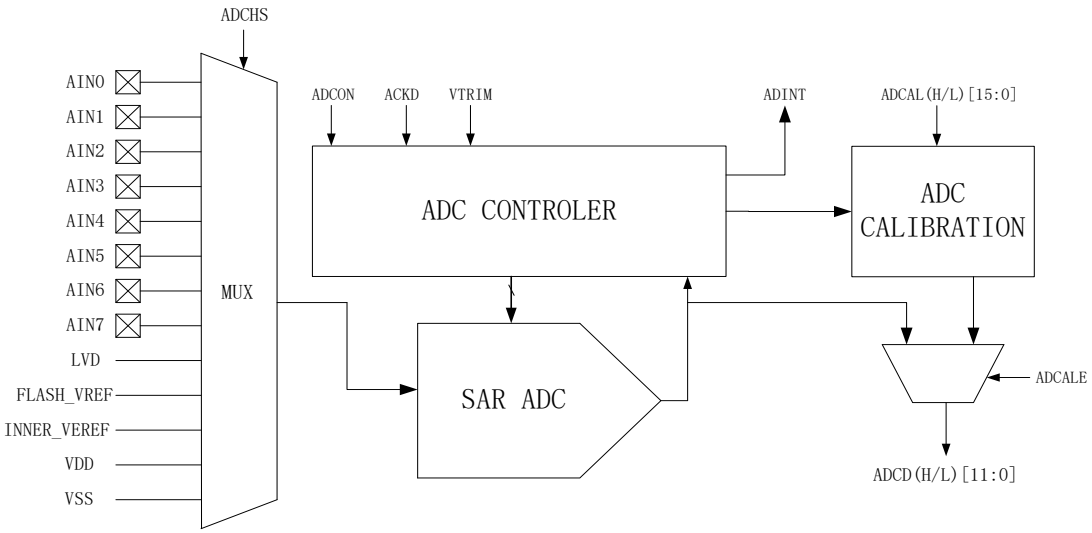


Figure 19-3-1 ADC Architecture

19.4 Function Introduction

ADC can be enabled by AST. When AST=1, the input voltage selected by ADCHS will be analog/digital converted. The clock for ADC is the system clock with frequency division set by ACKD beforehand. When ADC clock is constant, the time for single conversion is set by HTME. The conversion time is (13+2^HTME) ADC clock cycle periods. 12-bit A/D will be stored in register ADCDH and ADCDL after the conversion. AST will be cleared automatically 2.5 clock cycles later. The interrupt flag ADCIF will be set to 1 at the same time. If ADC interrupt is enabled then, ADC interrupt occurs. The shortest ADC conversion time is 0.5us. Figure 19-4-1 is the sequence diagram for ADC conversion.

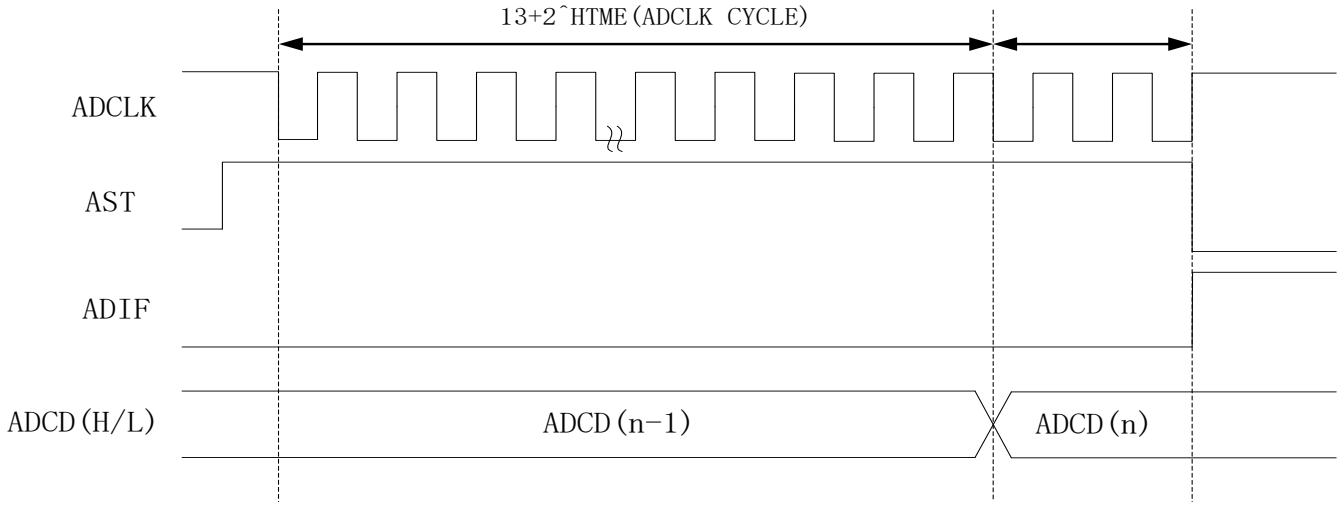


Figure 19-4-1 ADC Sequence Diagram

- **ADC Data Calibration**

  When internal voltage(1.5V) is selected as the reference voltage, due to the discreteness of the chips, the internal voltage in each chip can not be exactly the same which induces different ADC conversion results consequentially. Thus, it is necessary to correct the AD value after the conversion. The internal voltage will be tested and a correction value will be obtained when chips leave factory. When the chip's powered on, the correction value will be loaded into register ADCALL and ADCALH. The accurate AD value will be obtained by calculation according to the correction value. The final accurate result for AD will be stored in register ADCD. The function can be enabled by ADCALE. Users only need to set ADCALE=1 and the correction will be done automatically.

- **OPAMP**

  The op-amp function can reduce the detection signal of ADC, which can effectively expand the detection

signal range. The reduction multiple is set by AOPS, refer to the register section for details.

19.5 Register Description

Table 19-5-1 Register ADCON

B9H	7	6	5	4	3	2	1	0
ADCON	AST	ADIE	ADCIF	HTME			VSEL[1:0]	
R/W	R/W	R/W	R/W	R/W			R/W	
Initial Value	0	0	0	0	1	0	0	0

Bit number	Bit Symbol	Description
7	AST	ADC conversion enable control, the conversion starts when 1 is written to it, the hardware will clear it automatically after the conversion
6	ADIE	ADC interrupt enable control, 1 enables it
5	ADCIF	ADC interrupt flag, cleared when 1 is written to it
4~2	HTME	The number of sampling periods is 2 power HTME
1~0	VSEL	ADC reference voltage selection 00: internal 1.5V(INNER_VREF)as reference voltage 01: external VDD 10: external VREF 11: internal 1.5V(INNER_VREF) as reference voltage *Note： When the reference voltage is selected as an external VREF, the VREF voltage must be greater than 1.1V.*

Table 19-5-2 Register ADCFGL

BAH	7	6	5	4	3	2	1	0
ADCFGL	ACKD			ADCALE	ADCHS			
R/W	R/W			R/W	R/W			
Initial Value	0	0	0	1	0	0	0	0

Bit number	Bit Symbol	Description
7~5	ACKD	ADC clock frequency division setting 000： no division 001： frequency divided by 2 010： frequency divided by 4 … 111： frequency divided by 14
4	ADCALE	ADC calibration enable control, 1 enables it Valid only when the internal 1.5V is selected as the reference voltage.

		When ADCALE=1, ADC conversion result will be calibrated according to register ADCAL. For more information please refer to register ADCAL description
3~0	ADCHS	ADC channel enable selection 0000：disable the channels 0001：enable channel AD_CH[0](P27) 0010：enable channel AD_CH[1](P31) 0011：enable channel AD_CH[2](P32) 0100：enable channel AD_CH[3](P33) 0101：enable channel AD_CH[4](P34) 0110：enable channel AD_CH[5](P35) 0111：enable channel AD_CH[6](P36) 1000：enable channel AD_CH[7](P37) 1001：enable 1/4 VDD detection 1010：enable VSS detection Others：disable the channels

Table 19-5-3 Register ADCFGH

BBH	7	6	5	4	3	2	1	0
ADCFGH	-		VTRIM					
R/W	-		R/W					
Initial Value	-	-	1	0	0	0	1	1
Bit number	Bit Symbol		Description					
7~6	AOPS		Magnification factor selection of operational amplifier 00：无缩放 01：1/4 times 10：1/3 times 11：1/2 times					
5~0	VTRIM		Internal 1.5V reference voltage correction register, correction accuracy ±1mV					

Table 19-5-4 Register ADCAL

8088H	7	6	5	4	3	2	1	0
ADCALL	ADCAL[7:0]							
R/W	R/W							
Initial Value	0	0	0	0	0	0	0	0
8089H	7	6	5	4	3	2	1	0
ADCALH	ADCAL[15:8]							
R/W	R/W							
Initial Value	0	0	0	1	0	0	0	0

Bit number	Bit Symbol	v
15~0	ADCAL	ADC calibration register, valid only when ADCALE=1 and the internal 1.5V is selected as reference voltage. When it is valid, the ADC output is ：ADCDL=(ADC conversion result*ADCAL)/32768

Table 19-5-5 Register ADCD

BCH	7	6	5	4	3	2	1	0
ADCDL	ADCDL[3:0]				-			
R/W	R				-			
Initial Value	0	0	0	0	-	-	-	-
BDH	7	6	5	4	3	2	1	0
ADCDH	ADCDH[11:4]							
R/W	R							
Initial Value	0	0	0	0	0	0	0	0
Bit number	Bit Symbol	Description						
11~0	ADCD	ADC conversion result						

Table 19-5-6 Register ADOPC

808FH	7	6	5	4	3	2	1	0
ADOPC	-	-	-	-	-	-	GAIN[1:0]	
R/W	-	-	-	-	-	-	R/W	
Initial Value	-	-	-	-	-	-	0	0
Bit number	Bit Symbol	Description						
7~2	-	-						
1~0	GAIN	Op amp zoom factor selection bit 00: no zoom 01: Narrow 4 times 10: Narrow 3 times 11: Narrow 2 times						

19.6 ADC Control Example

For instance, external VDD is selected as the ADC reference voltage, channel 0 selected, ADC interrupt enabled, the program is like：

```
---------------------------------------------------------------------------------------------
//ADCON 定义
#define AST(N)      (N<<7)          //ADC start，  AST=0 end
#define ADIE(N)     (N<<6)          //interrupt enable
#define ADIF        (1<<5)          //interrupt signal
#define HTME(N)     (N<<2)          //N=0-7         //set the sampling time, it is the 2 power HTME clock cycle periods
#define VSEL(N)     (N)             //N=0-3         //reference voltage selection: 0-internal   1-VDD   2-external
//ADCFGL 定义
#define ACKD(N)         (N<<5)  //N=0-7                 //ADC clock division, division multiple=（ACKD+1)
#define ADCALE(N)       (N<<4)  //ADC revise，Select the internal reference voltage to be effective
#define ADCHS(N)        (N)     //N=0-15        // ADC channel selection, 1~13 corresponds to 0~12
void ADC_init(void)
{
    P27F = 3;                                           //set P27 as ADC pin function
    ADCON = AST(0) | ADIE(1) | HTME(7) | VSEL(1);       //set VDD as reference voltage of ADC
    ADCFGL = ACKD(7) | ADCALE(1) | ADCHS(1);            //select channel ADC0
    ADCON |= AST(1);                                    //AD conversion start
    INT2EN = 1;                                         //INT2 interrupt enable
}
void ADC_ISR (void) interrupt 7
{
    unsigned int    AD_Value;
    if(ADCON & ADIF)
    {
        ADCON |= ADIF;                          //clear interrupt signal
        AD_Value = ADCDH*256 + ADCDL;           //read ADC value
        AD_Value >>= 4;
        ADCON |= AST(1);                        //enable next AD conversion
    }
}
---------------------------------------------------------------------------------------------
```

20 TOUCH KEY

20.1 Function Introduction

With great anti-jamming performance, CA51F2 series chip can pass EFT,CS test and etc. The Touch Key module supports 24 channels at most.
In the application of TK_CAP pins need to be connected to a Cx capacitor, capacitance range 10nF ~ 47nF, capacitance accuracy of 10% or less, it is recommended to use polyester capacitors, X7R material capacitors or NPO material chip capacitors. cx can directly affect the touch sensitivity, the smaller the Cx capacitance, the lower the sensitivity, the larger the capacitance, the higher the sensitivity.

To meet the low power consumption requirement, it is designed to be
able to work in STOP mode.

20.2 Main Features

- Great anti-jamming performance which meets the EMC(CS) Standard
- Supports 16 channels at most
- Supports low power consumption mode
- Touch interrupt supported
- Clock division supported for charging/discharging
- Supports manual control and automatic mode
- Selective levels for comparator's threshold
- Multiplexing of touch pins with LED driver pins Touch can set internal charging and internal reference, can effectively suppress power supply low frequency interference
- Inner Waterproof compensation mechanism
- Waking up threshold configurable in STOP mode

20.3 Architecture

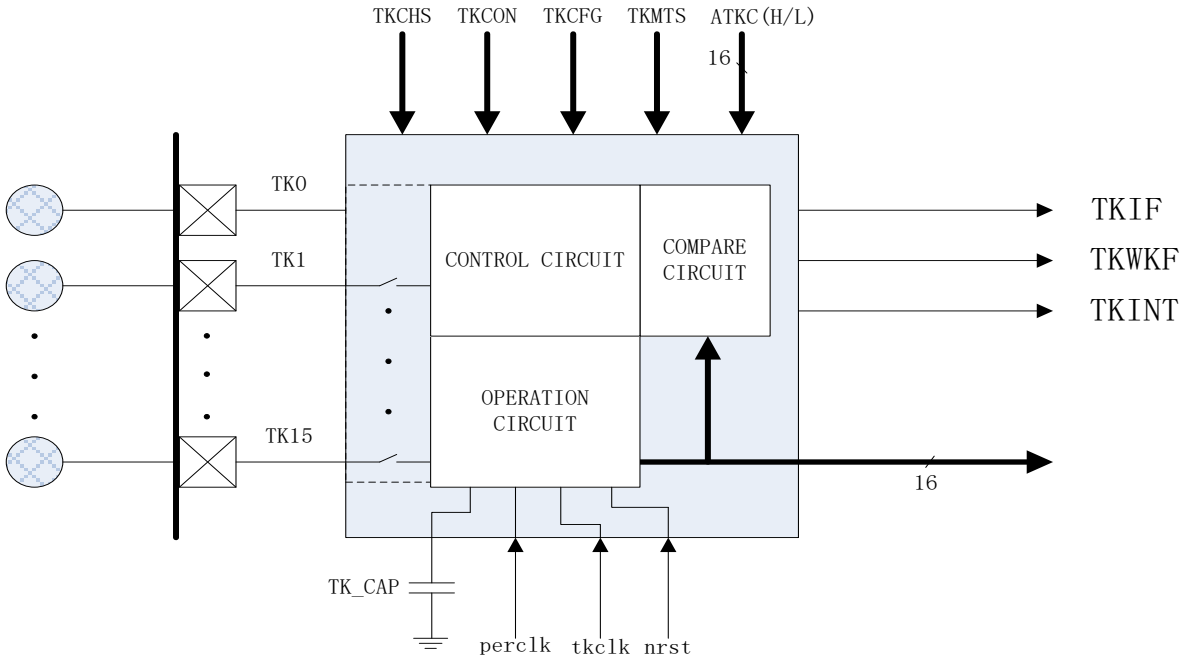


Figure20-3-1 Touch Key Module Architecture

20.4 Function Description

20.4.1 Manual Control Mode and Automatic Mode

The touch data collection can be enabled by TKST in manual control mode. When TKST=1, the module starts to collect the data through channel selected. There are at most 6 channels for one group for the channels selection, which is set by the register TKCHS with index. Every time the collection is enabled, one group of channels' data will be collected. TKST will be cleared automatically when the collection is over. The corresponding channel's interrupt flag will be set to 1. The touch data can be read from register TKMS by setting register INDEX then.

Manual control mode and automatic mode can be selected by TMEN. In automatic mode, unlike in manual control mode, the touch data collection is enabled by timer's timing. The clock source for the timer can be IRCL or XOSCL, which is selected by RTCKS in register CKSEL； the timing can be set by register TKMTS.

20.4.2 Touch Key Clock Frequency Division

The clock source for electrode charging/discharging is frenquency division by 4 of IRCH, which is extremely important for the touch module's performance. When the clock frequency for charging/discharging is too high, the touch electrode may not be charged properly, which makes the data change too small when fingers touch the key. The frequency prescale can be set by TKDIV. With proper frequency, touch module will perform even

better.

20.4.3 Low power consumption mode

As long as the clock source TFRC and low speed clock (IRCL or XOSCL) are enabled in STOP mode, the Touch Key module is able to charge/discharge normally. If TWKE=0, the data collection interrupt will awaken the CPU and then software will read data collected. The chip enters STOP mode again when data reading us over.
The module also includes threshold compare function. Users can set trigger threshold and the collected data will be compared with the threshold set by users in STOP mode. When the data exceeds the threshold, if TWKE=1 then, the interrupt will awaken CPU. CPU will collect the data and do judgement after being waken up.

20.4.4 Touch key common LED driver function description

Touch button common LED driver can be achieved by N touch button and N touch indicator control only need (N + 1) pins. Among them, the touch keys and LED driver positive control shared pins, LED negative terminal connected to COM, touch and LED control using a time-sharing approach to achieve.

Each touch has a separate control bit TLENx (x=0~19, corresponding to TK0~TK19) to enable the common LED drive function, it should be noted that the corresponding touch pin function must be turned on. After common LED enable, TLDATx (x=0~19, corresponding to LED0~LED19) can control each LED independently. COM pin is P06 (P06F is set to 6).

Touch data acquisition and LED control are implemented in a time-sharing manner, where the time for touch data acquisition is defined by TLCNTK, and the time for LED scanning is defined by TLCNTL. Note that the time defined by TLCNTK is the total time for each group of touch acquisition, and the number of touch channels in each group is 1~6. When the actual touch time is greater than the defined time, a TLERR interrupt will be generated. When the counter counts to the time defined by TLCNTK, the TLKOV interrupt is generated. Once the touch acquisition phase is completed, it enters the LED scan phase. TLCNTL defines the time of the LED scan phase, which affects the duty cycle of the LED scan, that is, it affects the brightness of the LED, and can be adjusted as needed during the application. In the LED scan phase, when the counter counts to the time defined by TLCNTL, TLLOV interrupt will be generated, so that a complete touch common LED cycle is completed. The following is a schematic of the operating phases.

Important reminder: in the touch pin and LED drive pin multiplexing mode applications, due to the existence of LED lights themselves diode junction capacitance, junction capacitance of different types of LED lights there are large differences, and this junction capacitance in the LED lights on and off when the performance may not be consistent (especially white LED lights are more obvious), this junction capacitance and its inconsistency will cause adverse effects on touch, so in the application of this mode should be used to strictly select the LED lights, and mass production can not just replace the LED light varieties.

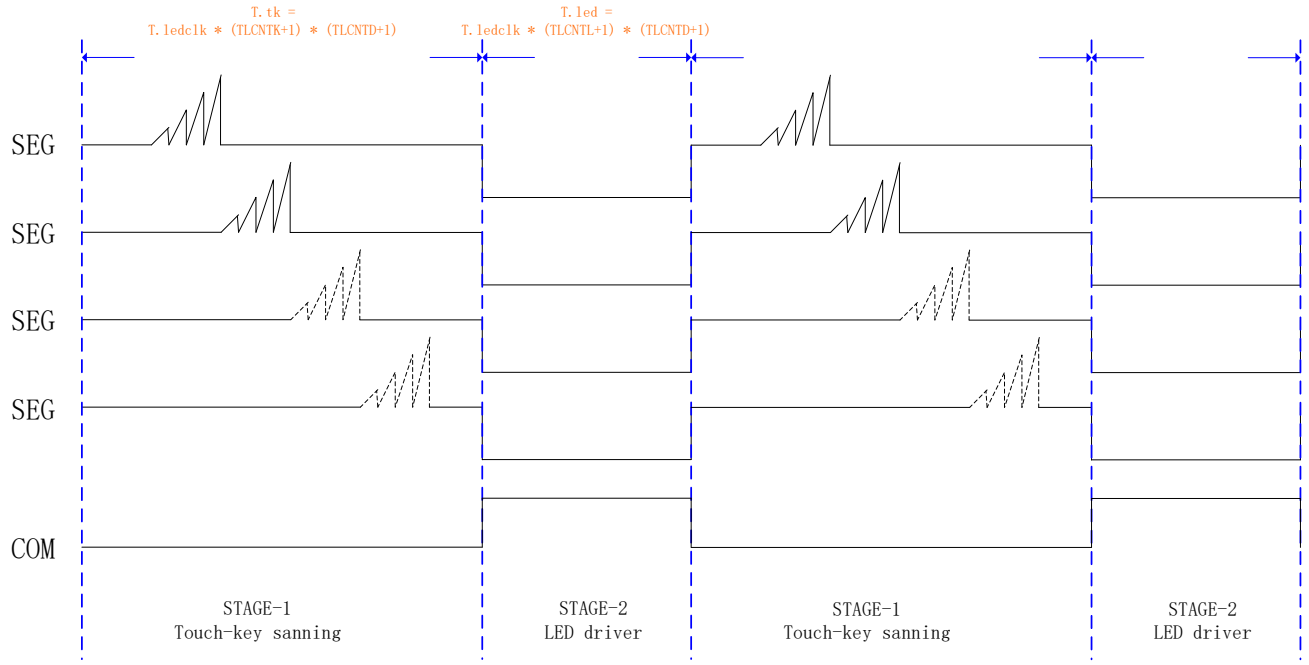


Figure20-5-1 Touch common LED driver schematic

20.4.5 Touch internal reference and internal op-amp

The touch module integrates an internal op-amp, which can be selected as the charging power supply for the touch keys via TKPWS (TKPWC [1]), and the charging voltage via VDS (TKPWC [5:4]). Alternatively, touch can also select the internal reference as the threshold voltage for the touch internal comparator via TKCVS (TKPWC [0]), and the internal reference voltage is selected via VIRS (TKPWC [3:2]).

20.4.6 Touch waterproof compensation mechanism

Touch is designed with a waterproof compensation mechanism, which can be turned on by TKPC (TKPWC [7:6]) for 2. After turning on this function, the non-selected touch pins will be synchronized to output the same compensation waveform as the charging frequency, which can effectively reduce the impact of parasitic capacitance between touch keys and achieve the effect of waterproof. Note: When turning on the waterproof compensation, the touch charging power supply should preferably be selected as external power supply.

20.5 Register Description

Table 20-5-1 Register TKCON

C1H	7	6	5	4	3	2	1	0
TKCON	TKST	TKIE	TMEN	TWKE	-	VRS[2:0]		
R/W	R/W	R/W	R/W	R/W	-	R/W		

Initial value	0	0	0	0	-	0	0	0

Bit number	Bit Symbol	Description
7	TKST	Data collection start enable control, 1 enables it, cleared automatically after the data collection
6	TKIE	TK interrupt enable control, 1 enables it
5	TMEN	Start mode selection 0: enabled by TKST 1: enabled by Timer
4	TWKE	Interrupt trigger selection 0: interrupt triggered when sampling is done 1: interrupt triggered when data collected exceeds the threshold
3	-	-
2~0	VRS	Reference voltage selection for comparator's threshold voltage (the threshold voltage is directly proportional with VDD) 0：maximum threshold voltage … 7：minimum threshold voltage

Table 20-5-2 Register TKCFG

C2H	7	6	5	4	3	2	1	0
TKCFG	TKDIV			TKTMS				
R/W	R/W			R/W				
Initial value	0	0	0	1	1	1	1	1

Bit number	Bit symbol	description
7~5	TKDIV	Frequency division selection for touch key clock 000：no division 001：frequency divided by 2 010：frequency divided by 3 … 111：frequency divided by 8
4~0	TKTMS	The discharging time setting for external modulation capacitor Discharging time = TKTMS x 128 x clock cycle period When TKDIV=0, the discharging time ranges from 32us to 992us Note：TKTMS cannot be set to 0

Table 20-5-3 Register TKPWC

8103H	7	6	5	4	3	2	1	0
TKPWC	TKPC		VDS		VIRS		TKPWS	TKCVS
R/W	R/W		R/W		R/W		R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Bit Number	Bit Symbol	description
7~6	TKPC	*Touch button undersampled channel output control* *00: No output low nor output compensation* *01: Output low* *Other: output compensation* *Notes.* *1. This function is only available for the pin configured as touch key function.* *2. If the common LED drive function is enabled, this function will be disabled for the pin selected as LED drive.*
5~4	VDS	Internal op-amp output voltage selection 00: 2V 01: 2.5V 10: 3V 11: 4V
3~2	VIRS	Internal voltage reference selection 00：1.2V 01：1.6V 10：2.0V 11：2.4V
1	TKPWS	Charging power supply selection 0: Select external power supply 1: Select internal op-amp output
0	TKCVS	Charge reference voltage selection 0：Select external voltage reference 1：Select internal voltage reference

Table 20-5-4 Register TKMTS

C3H	7	6	5	4	3	2	1	0
TKMTS	TKMTS[7:0]							
R/W	R/W							
Initial vaule	0	0	0	0	0	0	0	0

Bit number	Bit Symbol	description
7~0	TKMTS	The start time setting register in timing mode the start time=(TKMTS+1) × 32 × low speed clock cycle period If the low speed clock's frequency is 32.768K, the start time ranges from 0.977ms to 250ms.

Table 20-5-5 Register TKCHS

C4H	7	6	5	4	3	2	1	0
TKCHS	POL	NPOL	-	TKPS				

R/W	R/W	R/W	-	R/W				
Initial value	0	0	-	0	0	0	0	0

Note ： TKCHS is register with index, INDEX=0~5 indicates TKCHS0~TKCHS5 respectively

Bit number	Bit symbol	description
7	POL	ATKnC direction setting for threshold comparison 0：interrupt when the data collected is less than the threshold 1：interrupt when the data collected is greater than the threshold
6	NPOL	ATKnN direction setting for threshold comparison 0：interrupt when the data collected is less than the threshold 1：interrupt when the data collected is greater than the threshold
4~0	TKPS	Channel selection 00000：disable TK0~TK15 00001：TK0 selected 00010：TK1 selected 00011：TK2 selected ..... 10000：TK15 selected 10001：Internal reference capacitor selected

Table 20-5-6 Register ATKS

C5H	7	6	5	4	3	2	1	0
ATKSL	ATKSL[7:0]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0
C6H	7	6	5	4	3	2	1	0
ATKSH	ATKSH[15:8]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0

Note ： ATKC is register with index, INDEX=0~5 indicates ATKC0~ATKC5 respectively

Bit number	Bit symbol	description
15~0	ATKC	Compare threshold setting register, when TWKE=1, ATKC0~ATKC5 will be compared with TKMS0~TKMS5 automatically

Table 20-5-7Register ATKN

8092H	7	6	5	4	3	2	1	0
ATKNL	ATKN[7:0]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0
8093H	7	6	5	4	3	2	1	0
ATKNH	ATKN[15:8]							
R/W	R/W							

Initial value	0	0	0	0	0	0	0	0
Note ： ATKC is register with index, INDEX=0~5 indicates ATKC0~ATKC5 respectively								
Bit number	Bit Symbol	description						
15~0	ATKN	Compare threshold setting register, when TWKE=1, ATK0N~ATK5N will be compared with TK0MS~TK5MS automatically						

Table 20-5-8 Register TKMS

CEH	7	6	5	4	3	2	1	0
TKMSL	TKMS[7:0]							
R/W	R							
Initial value	0	0	0	0	0	0	0	0
CFH	7	6	5	4	3	2	1	0
TKMSH	TKMS[15:8]							
R/W	R							
Initial value	0	0	0	0	0	0	0	0
Note ： TKMS is register with index, INDEX =0~5 indicates TKMS0~TKMS5 respectively								
Bit number	Bit Symbol	description						
15~0	TKMS	Touch key sampling data register						

Table 20-5-10 Register TKIF

C7H	7	6	5	4	3	2	1	0
TKIF	-		TKIF5	TKIF4	TKIF3	TKIF2	TKIF1	TKIF0
R/W	-		R	R	R	R	R	R
Initial value	-	-	0	0	0	0	0	0
Bit number	Bit Symbol	description						
7~6	-	-						
5~0	TKIFx(x=5~0)	TK data collection interrupt flag, the bits correspond to 6 channels in order. When TWKE=1, TKIFx implies the data exceeds the ATKC or ATKN threshold						

Table 20-5-11 Register TKMAXF

8090H	7	6	5	4	3	2	1	0
TKMAXF	-	-	TKMXF5	TKMXF4	TKMXF3	TKMXF2	TKMXF1	TKMXF0
R/W	-	-	R	R	R	R	R	R
Initial value	-	-	0	0	0	0	0	0
Bit number	Bit Symbol	description						
7~6	-	-						
5~0	TKMXFx(x=5~0)	1 indicates that TKxMS exceeds ATKxC threshold, while 0 indicates TKxMS does not exceed ATKxC threshold. The polarity is set by POLx ； if TWKE=1, setting TKMXFx to 1 will set TKIFx as well； the software can not do anything						

		to it

Table 20-5-12 Register TKMINF

8091H	7	6	5	4	3	2	1	0
TKMINF	-	-	TKMNF5	TKMNF4	TKMNF3	TKMNF2	TKMNF1	TKMNF0
R/W	-	-	R	R	R	R	R	R
Initial value	-	-	0	0	0	0	0	0

Bit number	Bit Symbol	description
7~6	-	-
5~0	TKMNFx(x=5~0)	1 indicates that TKxMS exceeds ATKxN threshold, while 0 indicates TKxMS does not exceed ATKxN threshold. The polarity is set by NPOLx； if TWKE=1, setting TKMXFx to 1 will set TKIFx as well； the software can not do anything to it

Table 20-5-13 Register TLEN

8106H	7	6	5	4	3	2	1	0
TLEN (INDEX=0)	TLEN7	TLEN6	TLEN5	TLEN4	TLEN3	TLEN2	TLEN1	TLEN0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0
TLEN (INDEX=1)	TLEN15	TLEN14	TLEN13	TLEN12	TLEN11	TLEN10	TLEN9	TLEN8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Remark.

1. To enable LEDx, in addition to TLENx=1 (x=0,1,2,... ,15), the corresponding TKx pin must also be selected for the touch button function. 2.

2. The user can select any or all of the pins within the touch button pin range as the common LED driver pins.

Bit number	Bit symbol	description
7~0 (INDEX=1)	TLEN15~TLEN8	LED15~LED8 enable, 1 valid
7~0 (INDEX=0)	TLEN7~TLEN0	LED7~LED0 enable, 1 valid

Table 20-5-14 Register TLDAT

8107H	7	6	5	4	3	2	1	0
TLDAT (INDEX=0)	TLDAT7	TLDAT6	TLDAT5	TLDAT4	TLDAT3	TLDAT2	TLDAT1	TLDAT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0
TLDAT (INDEX=1)	TLDAT15	TLDAT14	TLDAT13	TLDAT12	TLDAT11	TLDAT10	TLDAT9	TLDAT8
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	description

7~0 (INDEX=1)	TLDAT15~TLDAT8	LED15~LED8 data, 1Table shows the effective level of drive
7~0 (INDEX=0)	TLDAT7~TLDAT0	LED7~LED0 data, 1Table shows the effective level of drive

Table 20-5-15 Register TLCON

8108H	7	6	5	4	3	2	1	0
TLCON	TLEIE	TLKIE	TLLIE	-	-	TLLVS		TLPOL
R/W	R	R/W	R/W	-	-	R/W		R/W
Initial value	0	0	0	-	-	0	0	0

Bit number	Bit symbol	description
7	TLEIE	TLERR interrupt enable, 1 valid
6	TLKIE	TLKOV interrupt enable, 1 valid
5	TLLIE	TLLOV interrupt enable, 1 valid
4~3	-	-
2~1	TLLVS	Touch key scan phase, COM pin level selection 00: Output high 01: Output low 10: Pull-up high Others: Reserved
0	TLPOL	LED drive phase, effective drive level selection 0: drive level high valid 1: Drive level low valid

Table 20-5-16 Register TLFLG

8109H	7	6	5	4	3	2	1	0
TLFLG	TLERR	TLKOV	TLLOV	-	-	-	-	-
R/W	R/W	R/W	R/W	-	-	-	-	-
Initial value	0	0	0	-	-	-	-	-

Bit number	Bit symbol	description
7	TLERR	Touch key scan phase time setting too short flag 0: Table indicates that the time set by the user is enough for the touch button scanning 1: Table shows that the time set by the user is not enough, the timer has finished counting and the channel has not been scanned Remark： This bit is the hardware auto-setting and auto-zeroing bit. Considering the convenience of users, this bit can also be cleared by writing 1 to it with software.
6	TLKOV	Touch key scan phase timer count full flag, 1Table show count full, software write 1 clear 0
5	TLLOV	LED drive phase timer count full flag, 1Table show count full, software write 1

		clear 0
4~0	-	-

Table 20-5-17 Register TLCKS

810AH	7	6	5	4	3	2	1	0
TLCKS	-	-	-	-	-	TLCKS[2:0]		
R/W	-	-	-	-	-	R/W		
Initial value	-	-	-	-	-	0	0	0

Bit number	Bit symbol	description
7~3	-	-
2~0	TLCKS	LED driver operating clock selection 001：IRCH 010：IRCL 011：XOSCL Others: Off

Table 20-5-18 Register TLCNTK

810BH	7	6	5	4	3	2	1	0
TLCNTKL	TLCNTK[7:0]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0
810CH	7	6	5	4	3	2	1	0
TLCNTKH	TLCNTK[15:8]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	description
15~0	TLCNTK	*Touch key scan phase time configuration Register* *T.tk = T.ledclk x (TLCNTK+1) x (TLDIV+1)* *Notes.* *1. T.tk, Table shows the time of the touch button scan phase; T.ledclk, Table shows the period of the operating clock of the LED driver circuit. 2.* *2. It takes about 1ms to scan each channel of the touch button.*

Table 20-5-19 Register TLCNTL

810DH	7	6	5	4	3	2	1	0
TLCNTLL	TLCNTL[7:0]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0

810EH	7	6	5	4	3	2	1	0
TLCNTLH	TLCNTL[15:8]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0

Bit number	Bit symbol	description
15~0	TLCNTL	LED drive phase time configuration Register T.led= T.ledclkx (TLCNTL+1) x (TLDIV+1) x N Notes. 1. T.led, Table shows the time of the LED drive phase.

Table 20-5-20 Register TLDIV

810FH	7	6	5	4	3	2	1	0
TLDIV	-	-	-	-	TLDIV[3:0]			
R/W	-	-	-	-	R/W			
Initial value	-	-	-	-	0	0	0	0

Bit number	Bit symbol	description
7~4	-	-
3~0	TLDIV	T.ledclk clock dividing frequency Register Frequency division multiplier is (TLDIV+1)

21 Low Voltage Detection (LVD)

21.1 Function Introduction

Low voltage detection (LVD) is used to monitor the chip's own power supply VDD, with four voltage levels available: 2.0V, 2.7V, 3.7V, 4.4V. When VDD is less than the set voltage value, trigger interrupt or reset can be set.

Note：Due the manufacturing process, the LVD trigger voltage may be slightly different.

Figure 22-1-1 shows the architecture of LVD.

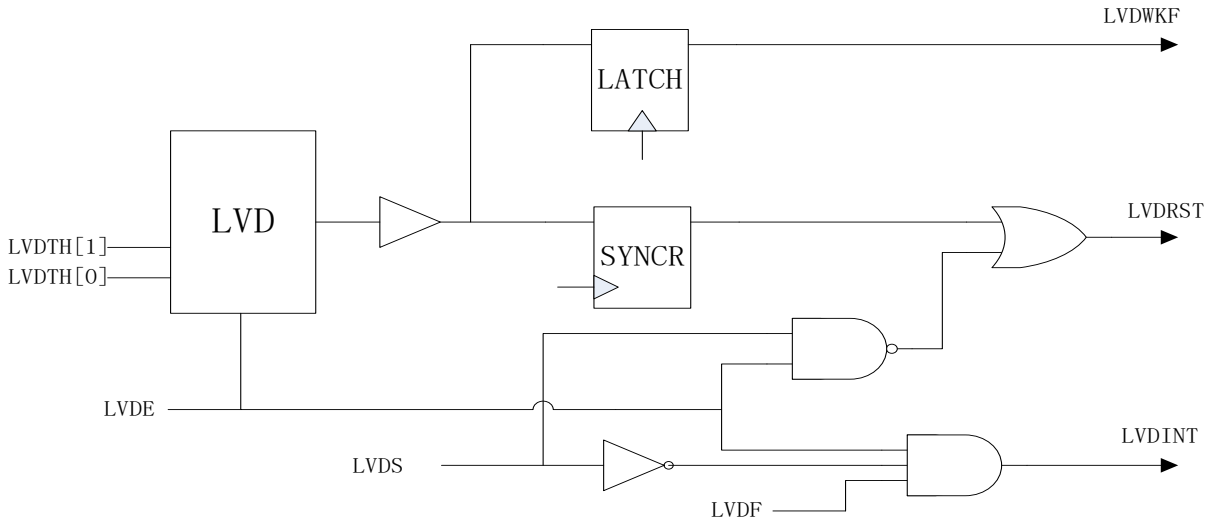


Figure22-1-1 LVD Schematic

21.2 Function Description

LVD function is enabled by LVDE and the trigger voltage is set by LVDTH. When VDD is lower than the trigger voltage, the LVDF will be set to1. If LVDS=0 then, there will be an interrupt; if LVDS=1, it will generate a reset signal. However, LVD reset signal will not reset itself, which means register LVDCON remains its status. As a result, if VDD is still lower than the trigger voltage set by users after the reset, it will be reset forever. Similarly, the interrupt will occur repeatedly if VDD is still lower than the trigger voltage set by users after the interrupt.

21.3 Register Description

Table 21-3-1 Register LVDCON

EFH	7	6	5	4	3	2	1	0
LVDCON	LVDE	LVDS	LVDF	-	-	-	LVDTH[1:0]	
R/W	R/W	R/W	R/W	-	-	-	R/W	
Initial Value	0	0	0	-	-	-	0	0

Bit number	Bit Symbol	Description
7	LVDE	LVD enable control, 1 enables it
6	LVDS	LVD function selection 0：interrupt 1：reset
5	LVDF	LVD flag, cleared when 1 is written to it
4~2	-	-
1~0	LVDTH	LVD trigger level selection 00：2.0V 01：2.7V 10：3.7V 11：4.4V

21.4 LVD Control Example

LVD interruptexample

Forinstance , set LVD to interrupt mode with trigger voltage 3V, the program is like：----------

```
#define LVDE(N)          (N<<7)    //N=0~1
#define LVDS_int         (0<<6)
#define LVDF             (1<<5)
#define LVDTH_3p7V    2
void LVD_init(void)
{
    LVDCON = LVDE(1) | LVDS_int |    LVDTH_3p7V;// enables LVD and set it to interrupt mode, set the trigger voltage to 3.7V
    INT4EN = 1;   // enables INT4 interrupt
    EA = 1;          // Turn on total interruption
}
void INT4_ISR (void) interrupt 9     // LVD LVD interrupt service routine
{
    if(LVDCON & LVDF)
    {
        LVDCON |= LVDF;        // clear LVD interrupt flag

    }
}
```

LVD reset example

For instance , set LVD to reset mode with trigger voltage 3.7V, the program is like：

```
#define LVDE(N)          (N<<7)    //N=0~1
#define LVDS_reset       (1<<6)
#define LVDTH_3p7V    2
void LVD_init(void)
{
    LVDCON = LVDE(1) | LVDS_reset| LVDTH_3p7V;// enables LVD and set it to interrupt mode, set the trigger voltage to 3.7V

}
```

---

22 LCD Driver

22.1 LCD Driver

22.1.1 Function Introduction

The built-in LCD driver can support up to 5com x 31seg, 4com x 32seg with a total of 36 output pins. The programmable duty cycle is 1/2, 1/3, 1/4, 1/5. The LCD driver has three modes: built-in voltage charge pump mode, charge pump divider mode, and resistor divider mode. For built-in voltage charge pump mode and charge pump divider mode, the bias ratio is fixed at 1/3; for resistor divider mode, the programmable bias ratio is: 1/2, 1/3, 1/4.

Figure 22-1-1-1 shows the principle of LCD

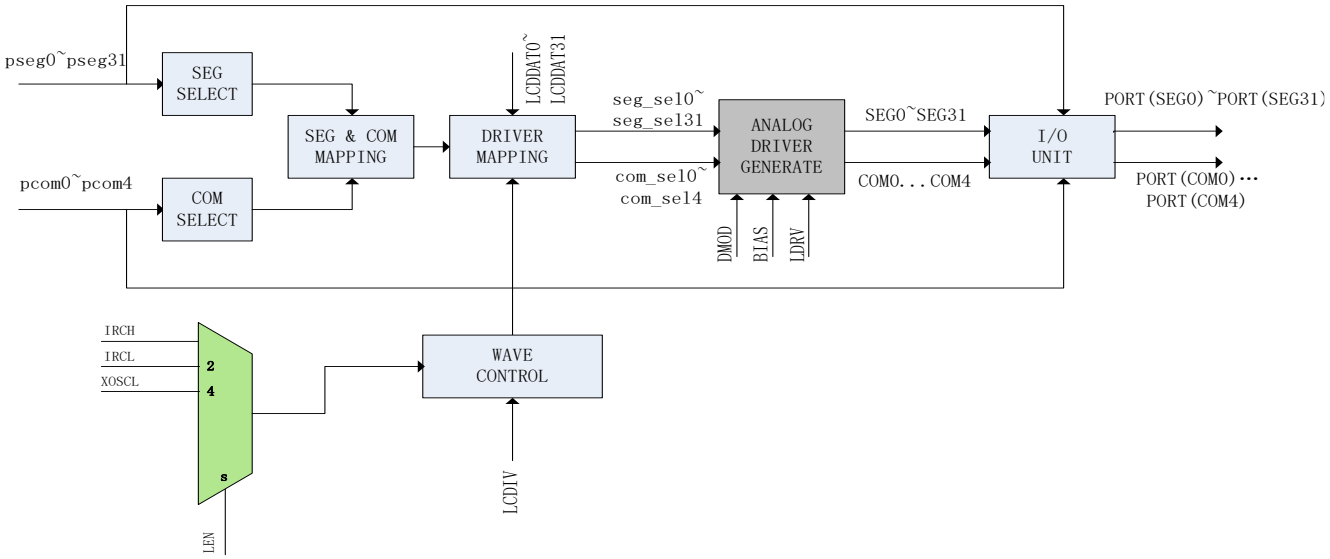


Figure22-1-1-1 LCD Schematic

22.1.2 LCD Bias Voltage

LCD programmable bias voltage can be: 1/2, 1/3, 1/4. The corresponding signals are as follows.

- **LCD bias voltage 1/2**

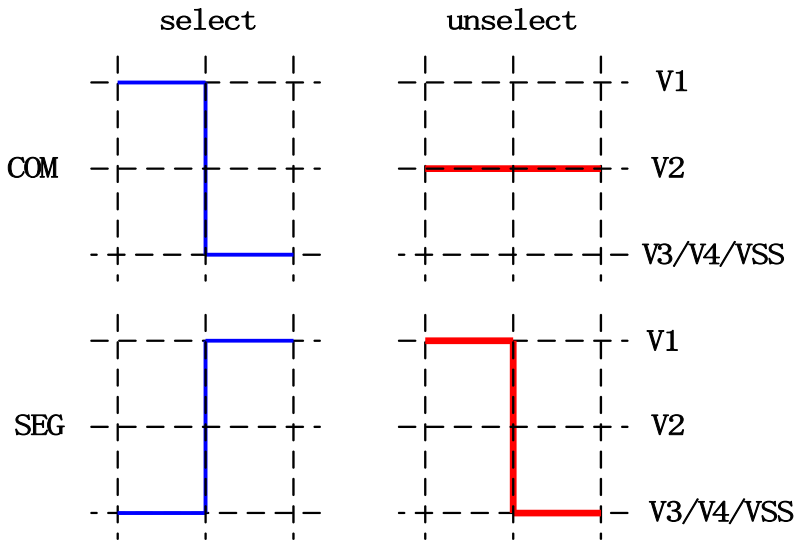


Figure22-1-1-2 LCD bias voltage 1/2

- **LCD bias voltage 1/3**

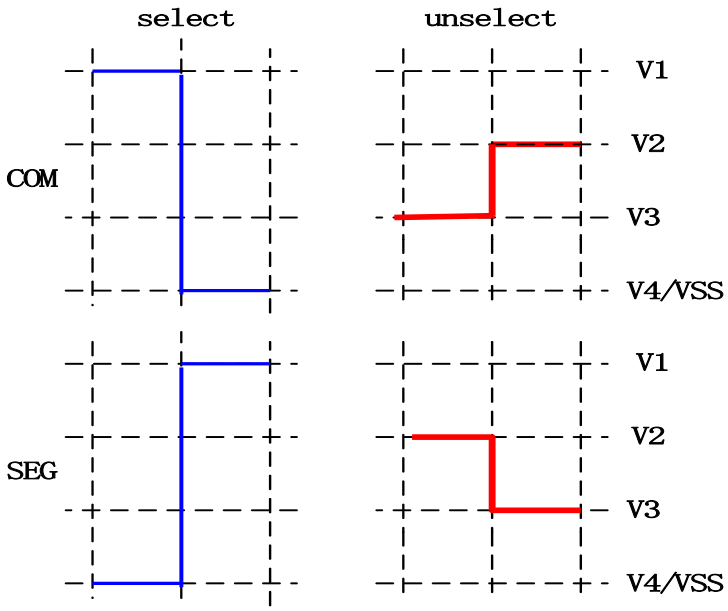


Figure22-1-1-3 LCD bias voltage 1/3

- **LCD bias voltage 1/4**

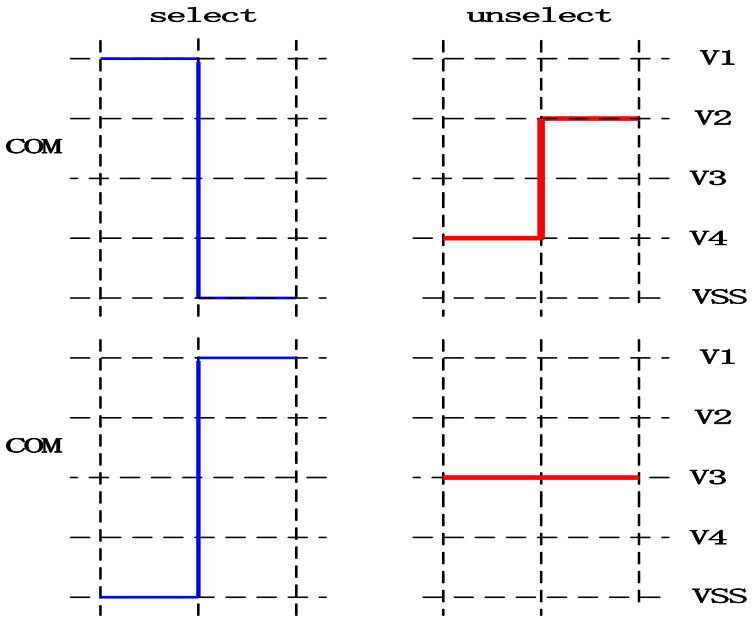


Figure22-1-1-4 LCD bias voltage 1/4

22.1.3 LCD Function Description

LCD mode can be selected by setting LMOD=1. The clock source for LCD driver can be selected by LEN. When the clock source is selected, LCD driver will be enabled simultaneously. The selected clock must be enabled and operates normally before the selection. Register LXDIV is the frequency divider for LCD clock and can be used to set different ratio for different clock sources. The typical frequency for LCD frame scanning is 64Hz.

The programmable duty cycle for LCD is：1/2、1/3、1/4、1/5、1/6、1/7、1/8, which is decided by the number of COM enabled. For instance, if 3 COM are enabled, then the duty cycle will be 1/3; if 5 COM are enabled, the duty cycle will be 1/5. Any of the COM can be enabled to and forms different combinations. However, the enabled COM pin number will correspond to real COM0, COM1, COM2... in order. For instance, pin COM1, COM3 and COM5 are the COM ports, then COM1 corresponds to real COM0, COM3 corresponds to real COM1 and COM5 corresponds to real COM2. Additionally, the duty cycle is 1/3. It is the same for SEG pins. For instance, pin SEG3, SEG5 and SEG7 are enabled, then SEG3 corresponds to real SEG0, SEG5 corresponds to real SEG1 and SEG7 corresponds to real SEG2. Those COM and SEG pins that are not enabled can still be used for other functions. It will not conflict with the LCD driver.

There is a 32byte display buffer for LCD driver. LCD display buffer corresponds to the real COM and SEG. The 32 bytes corresponds to SEG0~SEG31 in order, with COM0~COM4 corresponding to 0~4 bit in each byte. The display can be visited by index register INDEX and data register LXDAT. INDEX=0~31 correspond to SEG0~SEG31 display buffer respectively.

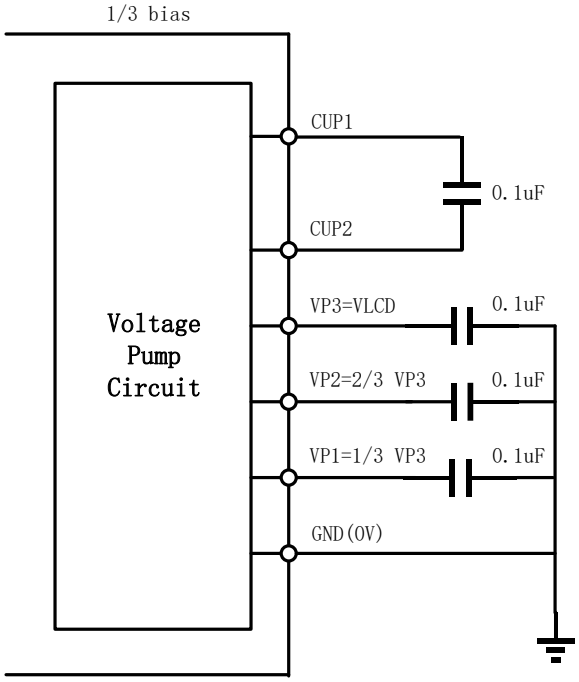
- **Resistive voltage divider mode**

In resistive voltage divider mode, the LCD driver has 8 levels of drive intensity adjusTable , set by LDRV, different

drive intensity output voltage is different, in the application should be adjusted for different LCD displays this value. To meet different power requirements, the LCD driver has 4 levels of drive current selection, set by DMOD, when the smaller the drive current, the smaller the power consumption of the driver itself, but the LCD pins also appear on the larger the noise.

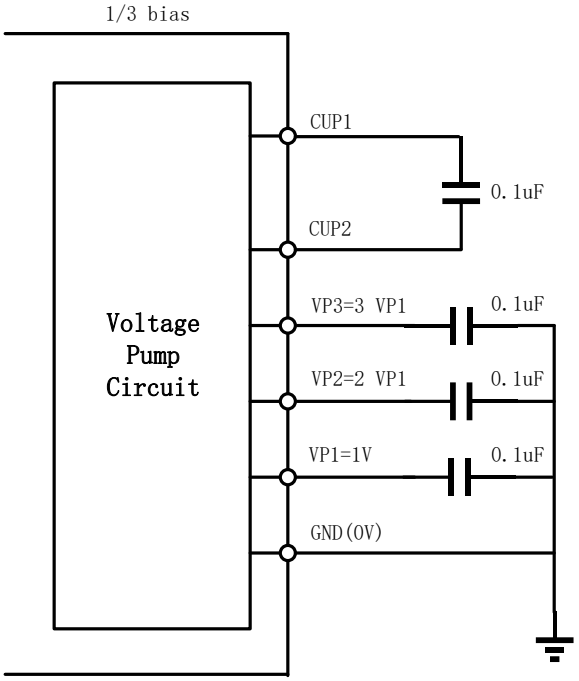
- **Charge pump voltage divider mode**

In charge pump voltage divider mode, the LCD drive voltage is VDD, the drive intensity and bias ratio are not adjust Table, and the bias ratio is constant at 1/3. Pins P00, P01, P02, P03 and P04 need to be set to LCD corresponding VP3, VP2, VP1, CUP1 and CUP2, and connected to 0.1uF capacitor, the connection method is as below. When using capacitor bias mode, you need to set LEN to select a good clock, TYPE to capacitor bias type, set PMPE to 1 to enable the charge pump, wait for at least 25ms and then turn on the LCD, i.e. LCE position 1, and light up the LCD panel.



- **Built-in voltage charge pump mode**

In the built-in voltage charge pump mode, when VDD is between 1.8V and 5V, the LCD drive voltage can be constant output, the output voltage is set by LDRV, the range is 2.4V~4.5V. In this mode, the bias ratio is constant 1/3, the drive intensity is not set. Pins P00, P01, P02, P03 and P04 need to be set to LCD corresponding to VP3, VP2, VP1, CUP1 and CUP2, connected to 0.1uF capacitor, connected as shown below. When using the built-in voltage charge pump mode, you need to set LEN to select the clock, TYPE to set the built-in voltage charge pump mode, set PMPE to 1 to enable the charge pump, wait for at least 25ms and then turn on the LCD, i.e. LCE position 1, and light up the LCD panel.



Note: Since the SOP28 package does not have the VP3, VP2, VP1, CUP1 and CUP2 pins, only resistive voltage divider mode can be applied.

.

22.2 LCD Register Description

Table 22-2-1 RegisterLCCON

E1H	7	6	5	4	3	2	1	0
LCCON	LEN [2:0]			-	-	TYPE [1:0]		LCE
R/W	R/W			-	-	R/W		R/W
Initial Value	0	0	0	-	-	0	0	0

Bit number	Bit Symbol	Description
7~5	LEN	LCD clock selection 001：IRCL 010：IRCH 011：XOSCL Others: module disabled
4~3	-	-
2~1	TYPE	Mode selection 00：Resistive voltage divider mode 01：Charge pump voltage divider mode 10：Built-in voltage charge pump mode 11：Resistive voltage divider mode
0	LCE	LCD module enable, 1 enable

Table 22-2-2 Register LCCFG

E2H	7	6	5	4	3	2	1	0
LCCFG	DMOD[1:0]		BIAS[1:0]		-	LDRV[2:0]		
R/W	R/W		R/W		-	R/W		
Initial value	0	0	0	0	-	0	0	0

Bit number	Bit Symbol	Description
7~6	DMOD	LCD resistive type drive current size selection bit 00：5uA 01：40uA 10：80uA 11：130uA *Note: This setting is only valid in resistive voltage divider mode.*
5~4	BIAS	Bias selection bit for LCD resistor type 01：1/2 Bias 10：1/3 Bias other：1/4 Bias *Note: This setting is only valid in resistive voltage divider mode.*
3	-	-
2~0	LDRV	**Resistive voltage divider mode.** LCD drive intensity control bit 000：Level 1（ Minimum） 001：Level 2 ... 111：Level 8（ Maximum ） **Built in voltage charge pump mode.** LCD output voltage control bit000：2.4V 001：2.7V 010：3.0V 011：3.3V 100：3.6V 101：3.9V 110：4.2V 111：4.5V **Charge pump voltage divider mode.** This setting is invalid.

Table 22-2-3 Register LCDIV

E4H	7	6	5	4	3	2	1	0
LCDIVL	LCDIV[7:0]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0
E5H	7	6	5	4	3	2	1	0
LCDIVH	-	-	-	-	LCDIV[11:8]			
R/W	-	-	-	-	R/W			
Initial value	-	-	-	-	0	0	0	0

Bit number	Bit Symbol	Description
15~12	-	-
11~0	LCDIV	LCD clock divider LCD scan frame frequency = LCD clock frequency ÷ ((LCDIV+1) x 512) Remark. 2. When LCD clock is selected as IRCL, the clock frequency is 1/4 of IRCL.

Table 22-2-4 Register LCDAT

E3H	7	6	5	4	3	2	1	0
LCDAT	LCDAT[7:0]							
R/W	R/W							
Initial value	0	0	0	0	0	0	0	0

Note: LCDAT is a register with index, set INDEX=0~31 corresponding to LCDAT0~LCDAT31 respectively.

Bit number	Bit Symbol	description
7~0	LCDAT	Display cache read/write registers

Table 2-5-5 LCD display cache

INDEX	SEG	COM0	COM1	COM2	COM3	COM4			
0	0	BIT0	BIT1	BIT2	BIT3	BIT4			
1	1	BIT0	BIT1	BIT2	BIT3	BIT4			
2	2	BIT0	BIT1	BIT2	BIT3	BIT4			
3	3	BIT0	BIT1	BIT2	BIT3	BIT4			
4	4	BIT0	BIT1	BIT2	BIT3	BIT4			
5	5	BIT0	BIT1	BIT2	BIT3	BIT4			
6	6	BIT0	BIT1	BIT2	BIT3	BIT4			
7	7	BIT0	BIT1	BIT2	BIT3	BIT4			
8	8	BIT0	BIT1	BIT2	BIT3	BIT4			
9	9	BIT0	BIT1	BIT2	BIT3	BIT4			
10	10	BIT0	BIT1	BIT2	BIT3	BIT4			
11	11	BIT0	BIT1	BIT2	BIT3	BIT4			
12	12	BIT0	BIT1	BIT2	BIT3	BIT4			
13	13	BIT0	BIT1	BIT2	BIT3	BIT4			

14	14	BIT0	BIT1	BIT2	BIT3	BIT4			
15	15	BIT0	BIT1	BIT2	BIT3	BIT4			
16	16	BIT0	BIT1	BIT2	BIT3	BIT4			
17	17	BIT0	BIT1	BIT2	BIT3	BIT4			
18	18	BIT0	BIT1	BIT2	BIT3	BIT4			
19	19	BIT0	BIT1	BIT2	BIT3	BIT4			
20	20	BIT0	BIT1	BIT2	BIT3	BIT4			
21	21	BIT0	BIT1	BIT2	BIT3	BIT4			
22	22	BIT0	BIT1	BIT2	BIT3	BIT4			
23	23	BIT0	BIT1	BIT2	BIT3	BIT4			
24	24	BIT0	BIT1	BIT2	BIT3	BIT4			
25	25	BIT0	BIT1	BIT2	BIT3	BIT4			
26	26	BIT0	BIT1	BIT2	BIT3	BIT4			
27	27	BIT0	BIT1	BIT2	BIT3	BIT4			
28	28	BIT0	BIT1	BIT2	BIT3	BIT4			
29	29	BIT0	BIT1	BIT2	BIT3	BIT4			
30	30	BIT0	BIT1	BIT2	BIT3	BIT4			
31	31	BIT0	BIT1	BIT2	BIT3	-			

Table 22-2-6 Register LCCAD

8117H	7	6	5	4	3	2	1	0
LCCAD	-	-	-	-	CAD_MOD[1:0]		CAD_LTH[1:0]	
R/W	-	-	-	-	R/W		R/W	
Initial value	-	-	-	-	0	1	0	1

Bit number	Bit Symbol	Description
7~4	-	-
3~2	CAD_MOD	LCD_CAD mode selection 00：turn off LCD_CAD 01： LCD_CAD length is determined by the digital signal length Others: LCD_CAD length controlled by analog signal
1~0	CAD_LTH	Analog signal to control LCD_CAD length selection, valid only when CAD_MOD=2/3 00：4us 01：8us 10：12us 11：16us

Table 22-2-7 Register LCPMP

8116H	7	6	5	4	3	2	1	0
LCPMP	-	-	-	-	PCKD			PMPE
R/W	-	-	-	-	R/W			R/W
Initial value	-	-	-	-	0	0	0	0

Bit number	Bit Symbol	description
7~4	-	- PUMP three-phase clock division conFigureuration register, valid only when PMPE=1
3~1	PCKD	PUMP three-phase clock division conFigureuration register, valid only when PMPE=1 000：1 frequency division 001：2 frequency division 010：4 frequency division 011：8 frequency division 100：16 frequency division 101：32 frequency division 110：64 frequency division 111：128 frequency division Remark. 1. The three-phase clock source is the working clock of LCD PUMP, when the LCD working clock selects 16MHz (IRCH), it will be divided into 500KHz before being used as the source clock of the three-phase clock. 2. The frequency of the final three-phase clock is the value after the frequency division and then divided by 3.
0	PMPE	When capacitive type, open PUMP, 1 valid

22.3 LCD Drive control routines

Resistive voltage divider mode

For example, the 4com*10seg, 1/3bias LCD display function in resistive voltage divider mode is programmed as follows

```
--------------------------------------------------------------------------------
//LCCON Definition
#define LEN_IRCH            (1<<5)
#define LCD_TYPE(N)         (N<<1)
#define LCE(N)              (N<<0)
//LCCFG Definition
#define DMOD_130ua          (3<<6)
#define BIAS_1_3            (2<<4)
#define LCDRV_LEV(N)        (N) //N=0-7
//LCCAD Definition
#define CAD_MOD(N)          (N<<2)   //N=0-3
#define CAD_LTH(N)          (N<<0)   //N=0-3
//LCPMP Definition
#define PCKD(N)             (N<<1)   //N=0-7
#define PMPE(N)             (N<<0)   //N=0-1
void LCD_init(void)
{
```

```
    unsigned char i;
    // Initialize COM pins
    P51F = 6;
    P50F = 6;
    P47F = 6;
    P46F = 6;
    // Initialize SEG pins
    P34F = 6;
    P35F = 6;
    P36F = 6;
    P37F = 6;
    P40F = 6;
    P41F = 6;
    P42F = 6;
    P43F = 6;
    P44F = 6;
    P45F = 5;
    // Initialize related registers
    LCDIVH = 0x01;                                  //Set the LCD clock division frequency, the target frame frequency is 64HZ
    LCDIVL = 0xdd;
    LCCFG =   DMOD_130ua | BIAS_1_3 | LCDRV_LEV(7);   //Set LCD drive current, bias, and drive intensity
    LCCON =    LEN_IRCH | LCD_TYPE(0) | LCE(1);        //Set LCD clock source, select LCD mode, enable LCD
    //LCD RAM clear
    for(i = 0; i < 10; i++)
    {
        INDEX = i;
        LCDAT = 0;
    }
}
```

Charge pump voltage divider mode

For example, the 4com*10seg, 1/3bias LCD display function in charge pump voltage divider mode is programmed as follows.

```
void LCD_init(void)
{
    unsigned char i;
    // Initialize COM pins
    P51F = 6;
    P50F = 6;
    P47F = 6;
    P46F = 6;
```

```
        // Initialize SEG pins
        P34F = 6;
        P35F = 6;
        P36F = 6;
        P37F = 6;
        P40F = 6;
        P41F = 6;
        P42F = 6;
        P43F = 6;
        P44F = 6;
        P45F = 5;
        // Initialize relevant capacitor pins
        P00F = 6;
        P01F = 6;
        P02F = 6;
        P03F = 6;
        P04F = 6;
        // Initialize related registers
        LCDIVH = 0x01;                                  //Set the LCD clock division frequency, the target frame
frequency is 64HZ
        LCDIVL = 0xdd;
        LCPMP = PCKD(2) | PMPE(1);                      // Set three-phase clock division, enable
        LCCFG =   DMOD_130ua | BIAS_1_3 | LCDRV_LEV(7);   //Set LCD drive current, bias, and drive intensity
        LCCON =    LEN_IRCH | LCD_TYPE(1) | LCE(0);     //Set LCD clock source, select LCD mode, enable LCD
        Delay_ms(30);
        LCCON |= LCE(1);
        //LCD RAM clear
        for(i = 0; i < 10; i++)
        {
            INDEX = i;
            LCDAT = 0;
        }
    }
    ------------------------------------------------------------------------------------
```

Built-in voltage charge pump mode

For example, the built-in voltage charge pump mode with 4com*10seg, 1/3bias LCD display function is programmed as follows.

```
    ------------------------------------------------------------------------------------
    void LCD_init(void)
    {
        unsigned char i;
        // Initialize COM pins
        P51F = 6;
```

```
    P50F = 6;
    P47F = 6;
    P46F = 6;
    // Initialize SEG pins
    P34F = 6;
    P35F = 6;
    P36F = 6;
    P37F = 6;
    P40F = 6;
    P41F = 6;
    P42F = 6;
    P43F = 6;
    P44F = 6;
    P45F = 5;
    // Initialize relevant capacitor pins
    P00F = 6;
    P01F = 6;
    P02F = 6;
    P03F = 6;
    P04F = 6;
    // Initialize related registers
    LCDIVH = 0x01;                                  // Set the LCD clock division frequency, the target frame
frequency is 64HZ
    LCDIVL = 0xdd;
    LCPMP = PCKD(2) | PMPE(1);                      // Set three-phase clock division, enable
    LCCFG =   DMOD_130ua | BIAS_1_3 | LCDRV_LEV(7);   // Set LCD drive current, bias, and drive intensity
    LCCON = LEN_IRCH | LCD_ TYPE(2) | LCE(0);       // Set LCD clock source, select LCD mode, enable LCD

    Delay_ms(30);
    LCCON |= LCE(1);
    //LCD RAM CLEAR
    for(i = 0; i < 10; i++)
    {
        INDEX = i;
        LCDAT = 0;
    }
}
```

---

23 Program Download and Simulation

23.1 Progrm Download

CA51F4 series chips mainly use the ISP method to download programs, and there are two kinds of ports to choose from:
Two-wire communication port:
To download through I2C port, 4 wires need to be connected: VDD, I2C_SCL(P05), I2C_SDA(P06), GND.

Single wire communication port:
For downloading through SWIM port, 3 wires need to be connected: VDD, SWIM(P07), GND.

Note: For the LQFP48 package chip, both of the above ports can be used, while the SOP28 package can only be used for the two-wire communication port as the SWIM and RESET pins are not pinout. Also, note that the above two emulation ports correspond to different downloaders.

For more details about the program download procedure, please refer to the "CA51F4 Series Development Download Tool Instructions".

23.2 Online Simulation

The CA51F4 series chips support in-circuit emulation, similar to program download, with the same two types of ports to choose from:
Two-wire communication port:
For in-circuit emulation via the I2C port, three wires need to be connected: I2C_SCL (P05), I2C_SDA (P06), and GND.

Single wire communication port:
Online emulation through SWIM port, 3 wires need to be connected: RESET, SWIM(P07), GND. It should be noted that the program inside the chip cannot set the RESET pin as GPIO function before entering the emulation, otherwise it cannot enter the single-wire emulation mode.

When TSME=0(PCON [3]), the chip is forbidden to enter simulation mode. TSMODE (PCON [2]) will be set to 1in simulation mode. The software can decide whether to enter power save mode or switch to low speed clock according to the status of TSMODE.

For the LQFP48 package chip, both of the above ports can be used, while the SOP28 package can only use the two-wire communication port for in-circuit emulation because the SWIM and RESET pins are not pinout. Also, note that the above two emulation ports correspond to different emulators. More details about the emulation functions can be found in the emulator's related documentation.

24 Electrical Specification

24.1 Limit Parameter

Parameter	Minimum	Maximum	Unit
DC voltage for power supply	-0.3	6	V
Input voltage for I/O pin	-0.3	VDD+0.3	V
Working temperature	-40	85	℃
Storage temperature	-55	125	℃
CPU working frequency	-	16	MHz

Note： When the parameters exceed the limits above, the working status of the chip is unpredicTable which may lead to severe damage to the chip. Working in such environment for a long time will influence the reliability of the chip.

24.2 DC Electrical Specification

Parameter	symbol	Operating Voltage	Minimum	Typical	Maximum	Unit	condition
Working current	Iop1	VDD=1.8V		1.87		mA	The system clock is IRCH (16MHz), with other clocks disabled. No load for all the output pins, (with LDO set to high power consumption mode, Output voltage is 1.61V). No floating for digital input pins. All the peripherals are disabled, with CPU executing instruction NOP
		VDD=3.3V		1.87			
		VDD=5V		1.88			
	Iop2	VDD=1.8V		19.7		uA	The system clock is IRCL (131kHZ), with other clocks disabled. No load for all the output pins, with LDO set to high power consumption mode, Output voltage is 1.61V. No floating for digital input pins. All the peripherals are disabled, with CPU executing instruction NOP
		VDD=3.3V		20.0			
		VDD=5V		20.2			
	Iop3	VDD=1.8V		7.6		uA	The system clock is XOSCL (32.768kHZ), with other clocks disabled. No load for all the output pins, with LDO set to high power consumption mode, Output voltage is 1.61V. No floating for digital input pins. All the peripherals are disabled, with CPU executing instruction NOP
		VDD=3.3V		7.8			
		VDD=5V		7.9			
	Iop4	VDD=1.8V		9.5		uA	The system clock is XOSCL (32.768kHZ), with other clocks disabled.set LDO low power mode, output

		VDD=3.3V		9.6			voltage is 1.61V, open LCD driver (no external LCD panel), set LCD built-in voltage charge pump mode, LCD output voltage is 3V, duty cycle is 1/4duty, LCD clock is XOSCL, LCD_CAD is off (CAD_MOD=0), all LCD pins are on, all other output pins are unloaded, all digital input pins are not floating, and other peripherals are off
		VDD=5V		9.8			
	Iop5	VDD=1.8V		12.0		uA	System clock is XOSCL (32.768kHZ), other clocks off, LDO set to low power mode, output voltage 1.61V, LCD driver on (no external LCD panel), LCD mode set to resistive voltage divider mode, bias ratio 1/3, duty cycle 1/4duty, LCD current set to minimum, drive intensity set to maximum, LCD clock is XOSCL, LCD_CAD off (CAD_MOD=0), all LCD pins on, all other output pins no load, all digital input pins not floating, other peripherals off
		VDD=3.3V		13.9			
		VDD=5V		16.1			
	Iop6	VDD=1.8V		24.0		uA	System clock is IRCL (131kHZ), other clocks off, LDO set to low power mode, output voltage 1.61V, LCD driver on (no external LCD panel), LCD mode set to built-in voltage charge pump mode, LCD output voltage 3V, duty cycle 1/4duty, LCD clock is IRCL, LCD_CAD off (CAD_MOD=0), all LCD pins on, all other output pins no load, all digital input pins not floating, other peripherals off
		VDD=3.3V		24.2			
		VDD=5V		24.4			
	Iop7	VDD=1.8V		23.1		uA	System clock is IRCL (161kHZ), other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to resistor divider mode, bias ratio is 1/3, duty cycle is 1/4duty, LCD current is set to minimum, LCD clock is XOSCL, LCD_CAD off (CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins are not floating, and other peripherals are turned off
		VDD=3.3V		25.0			
		VDD=5V		27.3			
Current for STOP mode	I_{stp1}	VDD=1.8V		2.2		uA	All clocks are off, all output pins have no load, all digital input pins are not floating, all peripherals are off, LDO is set to low power mode, Flash goes to sleep mode, and CPU goes to STOP mode.
		VDD=3.3V		2.3			
		VDD=5V		2.4			
	I_{stp2}	VDD=1.8V		5.5		uA	Except for XOSCL (32.768kHZ), other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to built-in voltage charge pump mode, LCD output voltage is 3V, duty cycle is 1/4duty, LCD clock is XOSCL, LCD_CAD is off
		VDD=3.3V		5.6			

		VDD=5V		5.7			(CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
	I_{stp3}	VDD=1.8V		7.8		uA	Except for XOSCL (32.768kHZ), other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to resistive voltage divider mode, bias ratio is 1/3, duty cycle is 1/4duty, LCD current is set to minimum, LCD clock is XOSCL, LCD_CAD off (CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
		VDD=3.3V		9.7			
		VDD=5V		11.9			
	I_{stp4}	VDD=1.8V		5.3		uA	Except for IRCL (131kHZ), all other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to built-in voltage charge pump mode, LCD output voltage is 3V, duty cycle is 1/4duty, LCD clock is XOSCL, LCD_CAD is off (CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
		VDD=3.3V		5.4			
		VDD=5V		5.5			
	I_{stp5}	VDD=1.8V		7.6		uA	Except for IRCL (131kHZ), all other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to resistive voltage divider mode, bias ratio is 1/3, duty cycle is 1/4duty, LCD current is set to minimum, LCD clock is XOSCL, LCD_CAD is off (CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
		VDD=3.3V		9.3			
		VDD=5V		11.6			
Current for IDLE mode	I_{idl1}	VDD=1.8V		0.816		mA	The system clock is set to IRCH (16MHz), other clocks are off, all output pins are unloaded, all digital input pins are not floating, all peripherals are off, LDO is set to low power mode, Flash goes to sleep mode, and CPU goes to IDLE mode.
		VDD=3.3V		0.817			
		VDD=5V		0.821			
	I_{idl2}	VDD=1.8V		9.4		uA	The system clock is set to IRCL (131KHz), other clocks are off, all output pins are unloaded, all
		VDD=3.3V		9.5			

		VDD=5V		9.6			digital input pins are not floating, all peripherals are off, the LDO is set to low power mode, and the CPU enters IDLE mode.
	I_{idl3}	VDD=1.8V		6.6		uA	The system clock is set to XOSCL (32.768KHz), other clocks are off, all output pins have no load, all digital input pins do not float, all peripherals are off, LDO is set to low power mode, Flash goes into sleep mode, and CPU goes into IDLE mode.
		VDD=3.3V		6.6			
		VDD=5V		6.7			
	I_{idl4}	VDD=1.8V		6.6		uA	Except for XOSCL (32.768kHZ), other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to built-in voltage charge pump mode, LCD output voltage is 3V, duty cycle is 1/4duty, LCD clock is XOSCL, LCD_CAD is off (CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
		VDD=3.3V		6.7			
		VDD=5V		6.8			
	I_{idl5}	VDD=1.8V		8.9			Except for XOSCL (32.768kHZ), other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to resistive voltage divider mode, bias ratio is 1/3, duty cycle is 1/4duty, LCD current is set to minimum, LCD clock is XOSCL, LCD_ CAD off (CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
		VDD=3.3V		10.7			
		VDD=5V		12.9			
	I_{idl6}	VDD=1.8V		9.5			Except for IRCL (131kHZ), all other clocks are off, LDO is set to low power mode, output voltage is 1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to built-in voltage charge pump mode, LCD output voltage is 3V, duty cycle is 1/4duty, LCD clock is XOSCL, LCD_CAD is off (CAD _MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
		VDD=3.3V		9.6			
		VDD=5V		9.7			
	I_{idl7}	VDD=1.8V		11.8		uA	Except for IRCL (131kHZ), all other clocks are off, LDO is set to low power mode, output voltage is

		VDD=3.3V		13.6			1.61V, LCD driver is turned on (no external LCD panel), LCD mode is set to resistive voltage divider mode, bias ratio is 1/3, duty cycle is 1/4duty, LCD current is set to minimum, LCD clock is XOSCL, LCD_CAD is off (CAD_MOD=0), all LCD pins are turned on, all other output pins have no load, all digital input pins do not float, other peripherals are turned off, Flash enters sleep mode, and CPU enters STOP mode.
		VDD=5V		15.8			
High voltage for IO port Input (Schmitt mode on)	V_{hi1}	VDD=1.8V	0.75	-	1.8	V	-
		VDD=3.3V	1.20		3.3		
		VDD=5V	1.50		5		
High voltage for IO port input (Schmitt mode off)	V_{hi2}	VDD=1.8V		0.5*VDD	VDD	V	-
		VDD=3.3V					
		VDD=5V					
Low voltage for IO port input (Schmitt mode on)	V_{lo1}	VDD=1.8V	0	-	0.62	V	-
		VDD=3.3V	0	-	0.85		
		VDD=5V	0	-	1.20		
Low voltage for IO port input (Schmitt mode off)	V_{lo2}	VDD=1.8V	0	0.5*VDD		V	-
		VDD=3.3V					
		VDD=5V					
Sink current for IO port	I_{pu}	VDD=3.3V	-	2.4	-	mA	set IO push-pull output mode, Vol=VDD-0.3V
		VDD=5V	-	3.3	-		
Sink current for IO port (Excluding P06, P22, P23)	I_{ol}	VDD=3.3V	-	13.34	-	mA	set IO push-pull output mode, Vol=VDD+0.3V
		VDD=5V	-	19.05	-		
Sink current for IO port (P06, P22, P23)	I_{ol}	VDD=3.3V	-	11	-	mA	set IO push-pull output mode, Vol=VDD+0.3V
		VDD=5V	-	15	-		
Sink current for Pin REM(P5.4)	I_{rem}			400		mA	VDD=3.0V, Vol = GND+1.2V, set Irrigation current the highest level
IO port pull-up resistor	R_{u2}	VDD=1.8~5.5V		10		KΩ	

Note: The above parameters are typical chip test results randomly selected for reference only.

24.3 AC Electrical Specification

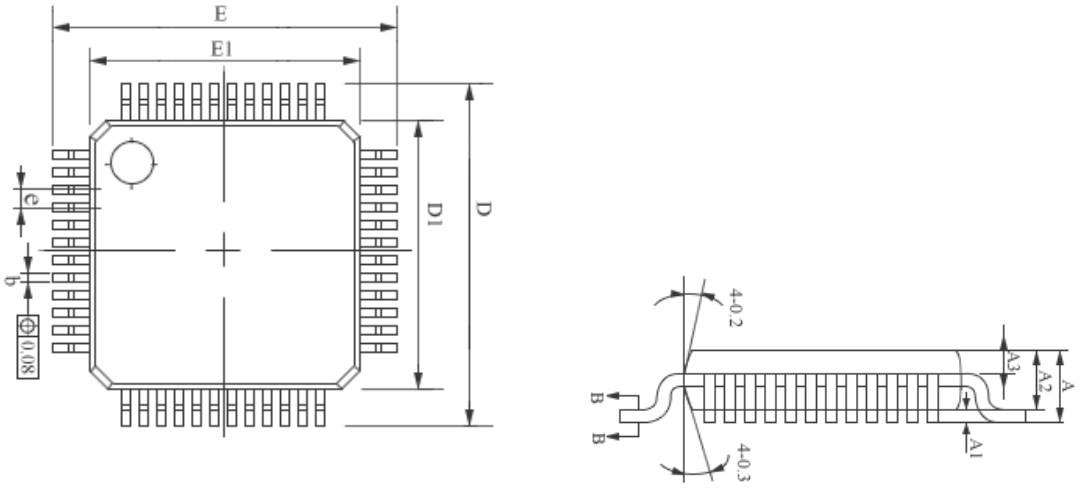
AC Electrical Specification(VDD=1.8-5.5V, TA=25℃, unless there are other explanations)

Parameter	Symbol	Minimum	Typical	Maximum	Unit	Condition
Time to start oscillation for IRCL	Trc1	-	50	-	us	IRCL frequency 131KHz
Time to start oscillation for IRCH	Trc2	-	10	-	us	IRCH frequency 16MHz
Time of the reset pulse	Trst	-	0.5	-	us	

Note ： VDD=3.3V,TA=25 ℃ , the factory frequency for internal high speed clock is 3.6864MHz, with deviation less than 1%..

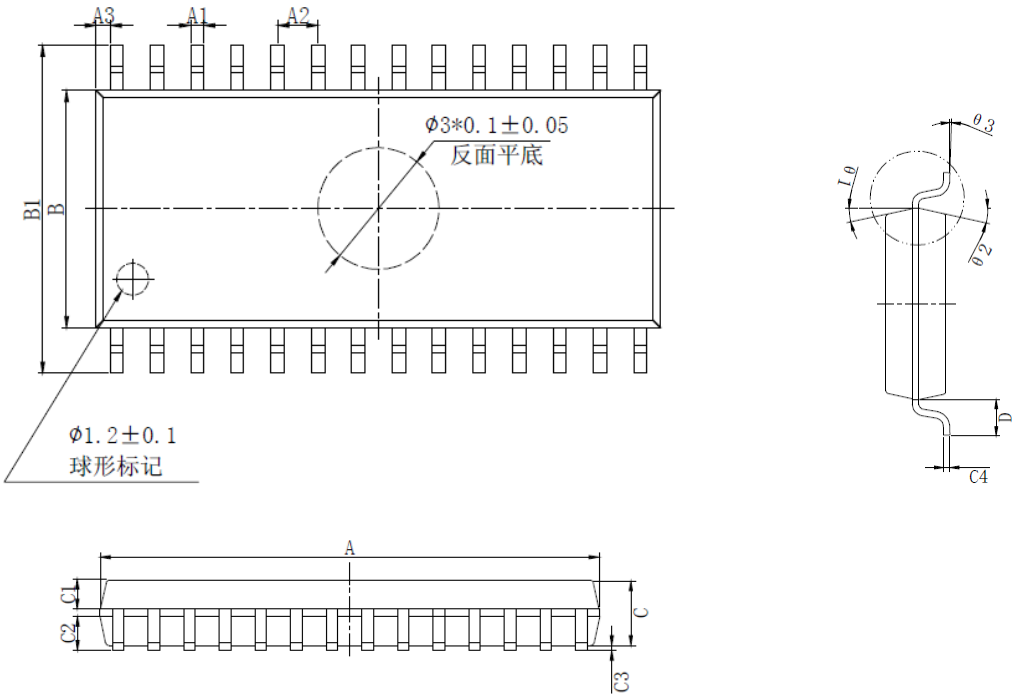
25 Package Type

Package (LQFP48)



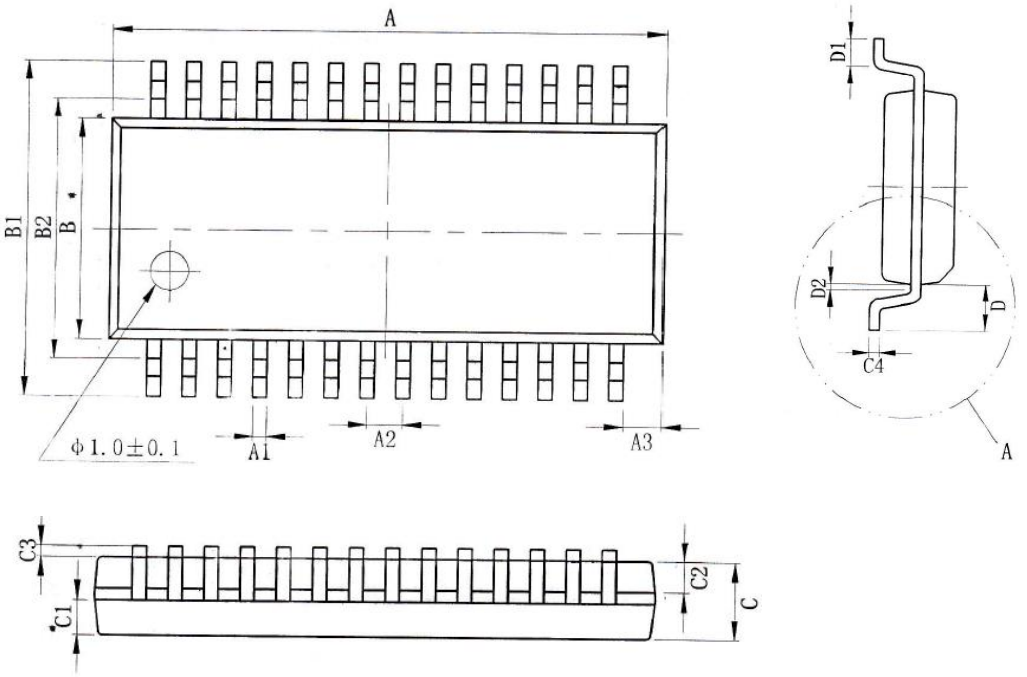
Sequence number	Minimum (mm)	Standard (mm)	Maximum (mm)
A	————	————	1.60
A1	0.05	————	0.15
A2	1.35	1.40	1.45
A3	0.59	0.54	0.69
b	0.18	————	0.27
D	8.80	9.00	9.20
D1	6.90	7.00	7.10
E	8.80	9.00	9.20
E1	6.90	7.00	7.10
e	0.50		

Package (SOP28)



序号	最小值(mm)	标准值(mm)	最大值(mm)
A	17.90	18.00	18.10
A1	0.356	0.40	0.456
A2	1.24	1.27	1.30
A3	---	0.542 TYP	---
B	7.40	7.50	7.60
B1	10.206	10.30	10.406
C	2.18	2.23	2.28
C1	0.938	1.0	1.038
C2	0.938	1.0	1.038
C3	0.03	0.09	0.17
D	1.353	1.40	1.453
C4	0.244	0.25	0.264

Package (SSOP28)



序号	最小值(mm)	标准值(mm)	最大值(mm)
A	9.80	9.90	10.00
A1	---	0.254TYP	---
A2	---	0.635TYP	---
A3	---	0.695TYP	---
B	3.85	3.90	3.95
B1	5.85	6.00	6.24
B2	---	5.00TYP	---
C	1.40	1.50	1.60
C1	0.61	0.66	0.71
C2	0.54	0.59	0.64
C3	0.05	0.15	0.25
C4	0.203	0.215	0.233
D	---	1.05TYP	---
D1	0.40	0.55	0.70
D2	0.15	0.20	0.25

26 Appendix

Appendix 1 Instruction Set Quick Reference Table

Mnemonic	Description	Description	Cycles
DATA TRANSFER			
MOV A,Rn	Move register to A	(A) ← (Rn)	1
MOV A,direct	Move direct byte to A	(A) ← (direct)	1
MOV A,@Ri	Move indirect RAM to A	(A) ← ((Ri))	1
MOV A,#data8	Move 8-bit immediate data to A	(A) ← #data	1
MOV Rn,A	Move A to register	(Rn) ← (A)	1
MOV Rn,direct	Move direct byte to register	(Rn) ← (direct)	2
MOV Rn,#data8	Move 8-bit immediate data to register	(Rn) ← #data	1
MOV direct,A	Move A to direct byte	(direct) ← (A)	1
MOV direct,Rn	Move register to direct byte	(direct) ← (Rn)	2
MOV direct,direct	Move direct byte to direct byte	(direct) ← (direct)	2
MOV direct,@Ri	Move indirect RAM to direct byte	(direct) ← ((Ri))	2
MOV direct,#data8	Move 8-bit immediate data to direct byte	(direct) ← #data	2
MOV @Ri,A	Move A to indirect RAM	((Ri)) ← (A)	1
MOV @Ri,direct	Move direct byte to indirect RAM	((Ri)) ← (direct)	2
MOV @Ri,#data8	Move 8-bit immediate data to indirect RAM	((Ri)) ← #data	1
MOV DPTR,#data16	Load Data Pointer with 16-bit constant	(DPTR) ← #data116	2
MOV A,@A+DPTR	Move Code byte relative to DPTR to A	(A) ← ((A)) + (DPTR)	2
MOV A,@A+PC	Move Code byte relative to FFe to A	(PC) ← (PC) + 1 (A) ← ((A) + (PC))	2
MOVX A,@Ri	Move External RAM (8-bit addr) to A	(A) ← ((Ri))	2
MOVX A,@DPTR	Move External RAM (16-bit addr) to A	(A) ← ((DPTR))	2
MOVX @Ri,A	Move A to External RAM (8-bit addr)	((Ri)) ← (A)	2
MOVX @DPTR,A	Move A to External RAM (16-bit addr)	(DPTR) ← (A)	2
PUSH direct	Push direct byte onto stack	(SP) ← (SP) + 1 ((SP)) ← (direct)	2
POP DIRECT	Pop direct byte from stack	(direct) ← ((SP)) (SP) ← (SP) - 1	2
XCH A,Rn	Exchange register with A	(A) ↔ (Rn)	1
XCH A,direct	Exchange direct byte with A	(A) ↔ (direct)	1
XCH A,@Ri	Exchange indirect RAM with A	(A) ↔ ((Ri))	1
XCHD A,@Ri	Exchange low-order Digit indirect RAM with A	(A.3,...,A.0) ↔ ((Ri).3,...,(Ri).0)	1
SWAP A	Swap nibbles within A	(A.3,...,A.0) ↔	1

		(A.7,...,A.4)	
ARITHMETIC OPERATIONS			
ADD A, Rn	Add register to A	(A) ← (A) + (Rn)	1
ADD A, direct	Add direct byte to A	(A) ← (A) + (direct)	1
ADD A, @Ri	Add indirect RAM to A	(A) ← (A) + ((Ri))	1
ADD A, #data8	Add 8-bit immediate data to A	(A) ← (A) + #data	1
ADDC A, Rn	Add register to A with Carry	(A) ← (A) + (C) + (Rn)	1
ADDC A, direct	Add direct byte to A with Carry	(A) ← (A) + (C) + (direct)	1
ADDC A, @Ri	Add indirect RAM to A with Carry	(A) ← (A) + (C) + ((Ri))	1
ADDC A, #data8	Add 8-bit immediate data to A with Carry	(A) ← (A) + (C) + #data	1
SUBB A, Rn	Subtract register from A with Borrow	(A) ← (A) - (C) - (Rn)	1
SUBB A, direct	Subtract direct byte from A with Borrow	(A) ← (A) - (C) - (direct)	1
SUBB A, @Ri	Subtract indirect RAM from A with Borrow	(A) ← (A) - (C) - ((Ri))	1
SUBB A, #data8	Subtract immediate data from A with Borrow	(A) ← (A) - (C) - #data	1
INC A	Increment A	(A) ← (A) + 1	1
INC Rn	Increment register	(Rn) ← (Rn) + 1	1
INC direct	Increment direct byte	(direct) ← (direct) + 1	1
INC @Ri	Increment indirect RAM	((Ri)) ← ((Ri)) + 1	1
INC DPTR	Increment Data Pointer	(DPTR) ← (DPTR) + 1	2
DEC A	Decrement A	(A) ← (A) – 1	1
DEC Rn	Decrement register	(Rn) ← (Rn) - 1	1
DEC direct	Decrement direct byte	(direct) ← (direct) - 1	1
DEC @Ri	Decrement indirect RAM	((Ri)) ← ((Ri)) - 1	1
MUL AB	Multiply A & B (A × B => BA)	temp16 ← (A) X (B) (A)←(temp.7,temp.6,...,temp.0) (B)←(temp.15,temp.14,...,temp.8)	4
DIV AB	Divide A by B(A/B => A +B)	QUO ← (A) / (B) .....REM	4

		(A) ← QUO (B) ← REM	

DA A	Decimal Adjust A	IF (A.3,...,A.0) > 9
		|| AC = 1
		THEN
		temp16 ← (A) +
		0x06
		(A) ←
		(temp.7,...,temp.0)
		
		IF (temp16) >
		0xFF
		THEN
		CY ← 1
		
		IF (A.7,...,A.4) > 9
		|| CY = 1
		THEN
		temp16 ← (A) +
		0x60
		(A) ←
		(temp.7,...,temp.0)
		
		IF (temp16) >
		0xFF
		THEN
		CY ← 1
			1

LOGICAL OPERATIONS			
ANL A, Rn	AND register to A	(A) ← (A) & (Rn)	1
ANL A, direct	AND direct byte to A	(A) ← (A) & (direct)	1
ANL A, @Ri	AND indirect RAM to A	(A) ← (A) & ((Ri))	1
ANL A, #data8	AND 8-bit immediate data to A	(A) ← (A) & #data	1
ANL direct, A	AND A to direct byte	(direct) ← (direct) & (A)	1
ANL direct, #data8	AND 8-bit immediate data to direct byte	(direct) ← (direct) & #data	2
ORL A, Rn	OR register to A	(A) ← (A) | (Rn)	1
ORL A, direct	OR direct byte to A	(A) ← (A) | (direct)	1
ORL A, @Ri	OR indirect RAM to A	(A) ← (A) | ((Ri))	1
ORL A, #data8	OR 8-bit immediate data to A	(A) ← (A) | #data	1
ORL direct, A	OR A to direct byte	(direct) ← (direct) | (A)	1
ORL direct, #data8	OR 8-bit immediate data to direct byte	(direct) ← (direct) |	2

		#data	
XRL A, Rn	Exclusive-OR register to A	(A) ← (A) ^ (Rn)	1
XRL A, direct	Exclusive-OR direct byte to A	(A) ← (A) ^ (direct)	1
XRL A, @Ri	Exclusive-OR indirect RAM to A	(A) ← (A) ^ ((Ri))	1
XRL A, #data8	Exclusive-OR 8-bit immediate data to A	(A) ← (A) ^ #data	1
XRL direct, A	Exclusive-OR A to direct byte	(direct) ← (direct) ^ (A)	1
XRL direct, #data8	Exclusive-OR 8-bit immediate data to direct byte	(direct) ← (direct) ^ #data	2
CLR A	Clear A	(A) ← 0	1
CPL A	Complement A	(A) ← /(A)	1
RL A	Rotate A Left	(A) ← (A.6,A.5,...,A.0,A.7)	1
RLC A	Rotate A Left through Carry	C ← A.7 (A) ← (A.6,A.5,...,A.0,C)	1
RR A	Rotate A Right	(A) ← (A.0,A.7,...,A.2,A.1)	1
RRC A	Rotate A Right through Carry	C ← A.0 (A) ← (C,A.7,...,A.2,A.1)	1
PROGRAM AND MACHINE CONTROL			
ACALL addr11	Absolute subroutine call	(PC) ← (PC) + 2 (SP) ← (SP) + 1 ((SP)) ← (PC7-0) (SP) ← (SP) + 1 ((SP)) ← (PC15-8) (PC10-0) ← page address	2
LACLL addr16	Long subroutine call	(PC) ← (PC) + 3 (SP) ← (SP) + 1 ((SP)) ← (PC7-0) ((SP)) ← (PC15-8) (PC) ←addr15-0	2
RET	Return from subroutine	(PC15-8) ← ((SP)) (SP) ← (SP) - 1 (PC7-0) ← ((SP)) (SP) ← (SP) - 1	2
RETI	Return from interrupt	(PC15-8) ← ((SP)) (SP) ← (SP) - 1 (PC7-0) ← ((SP)) (SP) ← (SP) - 1	2

AJMP addr11	Absolute Jump	(PC) ← (PC) + 2 (PC10-0) ← page address	2
LJMP addr16	Long Jump	(PC) ← (PC) + 3 (SP) ← (SP) + 1 ((SP)) ← (PC7-0) (SP) ← (SP) + 1 ((SP)) ← (PC15-8) (PC10-0) ←addr15-0	2
SJMP rel	Short Jump (relative addr)	(PC) ← (PC) + 2 (PC) ← (PC) + rel	2
JMP @A+DPTR	Jump indirect relative to DPTR	(PC) ← (A) + (DPTR)	2
JZ rel	Jump if A is Zero	(PC) ← (PC) + 2 IF (A) = 0 THEN (PC) ← (PC) + rel	2
JNZ rel	Jump if A is Not Zero	(PC) ← (PC) + 2 IF (A) <> 0 THEN (PC) ← (PC) + rel	2

CJNE A, direct, rel	Compare direct to A & Jump if Not Equal	(PC) ← (PC) + 3
		IF (A) <> (direct)
		THEN
		(PC) ← (PC) + relative offset
		IF (A) < (direct)
		THEN
		(C) ← 1
		ELSE
		(C) ← 0
			2

CJNE A, #data8, rel	Compare 8-bit immediate to A & Jump if Not Equal	(PC) ← (PC) + 3
		IF (A) <> data
		THEN
		(PC) ← (PC) + relative offset
		IF (A) < data
		THEN
		(C) ← 1
		ELSE
		(C) ← 0
			2

CJNE Rn, #data8, rel	Compare 8-bit immediate to reg. & Jump if Not Equal	(PC) ← (PC) + 3 IF (Rn) <> data	2

		THEN
		(PC) ← (PC) +
		relative offset
		IF (Rn) < data
		THEN
		(C) ← 1
		ELSE
		(C) ← 0	

CJNE @Ri, #data8, rel	Compare 8-bit immediate to ind. & Jump if Not Equal	(PC) ← (PC) + 3
		IF ((Ri)) <> data
		THEN
		(PC) ← (PC) +
		relative offset
		IF ((Ri)) < data
		THEN
		(C) ← 1
		ELSE
		(C) ← 0
			2

DJNZ Rn, rel	Decrement register & Jump if Not Zero	(PC) ← (PC) + 2 (Rn) ← (Rn) - 1 IF (Rn) <> 0 THEN (PC) ← (PC) + rel	2
DJNZ direct, rel	Decrement direct byte & Jump if Not Zero	(PC) ← (PC) + 2 (direct) ← (direct) - 1 IF (direct) <> 0 THEN (PC) ← (PC) + rel	2
NOP	No operation	(PC) ← (PC) + 1	1
BOOLEAN VARIABLE MANIPULATION			
CLR C	Clear Carry flag	(C) ← 0	1
CLR bit	Clear direct bit	(bit) ← 0	1
SETB C	Set Carry flag	(C) ← 1	1
SETB bit	Set direct bit	(bit) ← 1	1
CPL C	Complement Carry flag	(C) ← /(C)	1
CPL bit	Complement direct bit	(bit) ← /(bit)	1
ANL C, bit	AND direct bit to Carry flag	(C) ← (C) & (bit)	2
ANL C, /bit	AND complement of direct bit to Carry flag	(C) ← (C) & /(bit)	2
ORL C, bit	OR direct bit to Carry flag	(C) ← (C) | (bit)	2
ORL C, /bit	OR complement of direct bit to Carry flag	(C) ← (C) | /(bit)	2
MOV C, bit	Move direct bit to Carry flag	(C) ← (bit)	1
MOV bit, C	Move Carry flag to direct bit	(bit) ← (C)	2
JC rel	Jump if Carry flag is set	(PC) ← (PC) + 2	2

		IF (C) = 1 THEN (PC) ← (PC) + rel	
JNC rel	Jump if No Carry flag	(PC) ← (PC) + 2 IF (C) = 0 THEN (PC) ← (PC) + rel	2
JB bit, rel	Jump if direct Bit is set	(PC) ← (PC) + 3 IF (bit) = 1 THEN (PC) ← (PC) + rel	2
JNB bit, rel	Jump if direct Bit is Not set	(PC) ← (PC) + 3 IF (bit) = 0 THEN (PC) ← (PC) + rel	2
JBC bit, rel	Jump if direct Bit is set & Clear bit	(PC) ← (PC) + 3 IF (bit) = 1 THEN (bit) ← 0 (PC) ← (PC) + rel	2