



芯海科技
CHIPSEA

Gathering tiny bits of information together to form a vast ocean

CSU32M10 Application Notes

REV 1.0

芯海科技（深圳）股份有限公司

地址：深圳市南山区蛇口南海大道1079号花园城数码大厦A座9楼

电话：+(86 755)86169257 传真：+(86 755)86169057

网站：www.chipsea.com 邮编：518067

微信号：芯海科技

CSU32M10-02409



This document is exclusive property of CHIPSEA and shall not be reproduced or copied or transformed to any other form

without prior permission of CHIPSEA



Revision History

History	Modifications	Version Date
Rev 1.0 first version released		2019.5.16



Table of contents

Revision History..... 2

Contents 3

1How to use CSU32M10 to replace CSU32P20 4

2 Chip Applications..... 5

2.1 Analog Comparator Application Issues..... 5

2.2 Constant Current Source..... 5

2.3 EEPROM..... 6

2.4 Problem of excessive chip sleep current..... 9

2.5 Considerations for using analog comparator for short-circuit protection..... 12

3 Precautions for using IDE and CS-LINK 13

1 How to use CSU32M10 to replace CSU32P20

Use IDE_5.4.0 version IDE to open the previous CSU32P20 source program, change the chip model to CSU32M10, and regenerate the

For assembly projects, replace the header file with CSU32M10.inc, and for C language projects, replace the original program header file with

If the original SysRegDefine.c file is in the root directory of the project, the original file will be automatically relocated after the chip model is modified.

SysRegDefine.c file is replaced. If the original SysRegDefine.c is in the custom directory, a new one will be generated in the project root directory after the chip is modified.

The SysRegDefine.c file needs to be replaced by the user.

Note: After the chip is replaced, the code options are restored to the default values of the chip. Developers can reconfigure them according to their needs.

2 Chip Application

2.1 Analog comparator application issues

Compared with CSU32P20, CSU32M10 adds an analog comparator module with two analog input terminals C0P (PT3.4) and C0N

(PT3.3), you can also use the comparator built-in threshold 80mv, 200mv, 320mv, 480mv as an input of the comparator

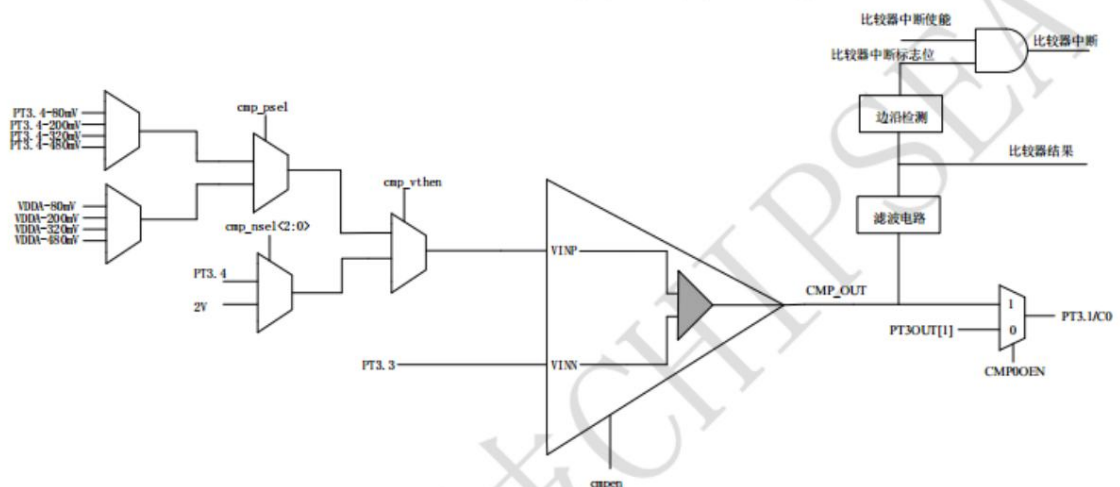
Input, C0 (PT3.1) pin can be used as the output of the comparator. This output is a digital output and there is no need to configure this port as an analog port.

CSU32M10 can turn off PWM output by changing the comparator result, which can be applied to short-circuit protection function.

The analog comparator does not work properly in SLEEP mode. Therefore, before entering SLEEP mode, the analog comparator needs to be

Device enable closed.

Analog comparator logic diagram:



Note: 1. When using a comparator, ensure that the positive and negative voltage inputs are > 2.4V, otherwise the comparator may work abnormally.

2. The output filter count of the analog comparator is not allowed to be configured as 0 instruction cycles, otherwise the analog comparator may work abnormally.

2.2 Constant current source

Compared with CSU32P20, CSU32M10 has a built-in 50mA constant current source module, which is output through the PT3.2 port and the CCSCON register

Just set the CCSOEN bit of the 1 register to 1.

位编号	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
特性	U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
CCSCON								CCSOEN

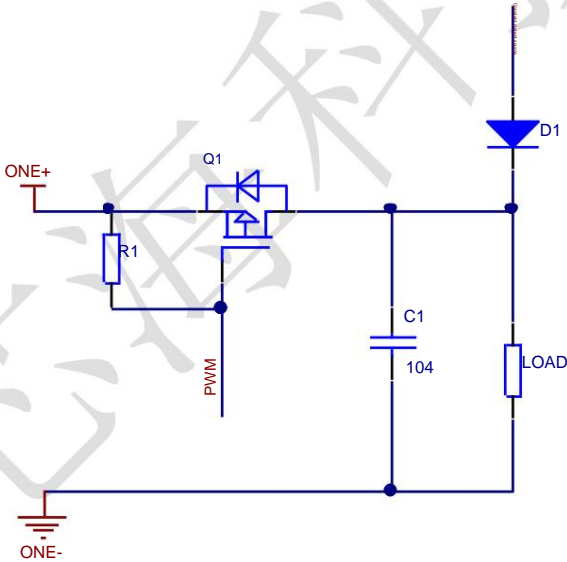
位地址	标识符	功能
7: 1	保留	保留
0	CCSOEN	电流源输出使能控制位 0: 禁止 50mA 电流源从 PT3.2 口输出 1: 使能 50mA 电流源从 PT3.2 口输出

Egŷ

```
void main()
{
    PT3PU_2 = 0; //disable pull up
    PT3PD_2 = 0; // disable pull down
    CCSOEN = 1; //enable current source
    while(1);
}
```

Note: When using a constant current source module, if the load is not a pure resistive load, it may cause the constant current source module to oscillate, so it is recommended to connect both ends of the load.

Connect a capacitor of 0.1uf or more, as shown in the following diagram:



2.3 EEPROM

CSU32M10 adds an EEPROM module compared to CSU32P20 , and the EEPROM address range is 0x2000~0x207F.

The TBLP instruction can be used to write to the EEPROM. Each EEPROM address stores one byte of data.

When performing read and write operations, only the lower 8 bits of data are valid.

When writing to the EEPROM, the write protection must be unlocked.

The WRPRT register is written to 96H, 69H, and 5AH successively. When writing to other address registers, the WRPRT register is written to 96H, 69H, and 5AH successively.

The device will be reset and the unlock function will automatically become invalid.

EEPROM write operation flow

- (1) Configure {EADRH, EADRL} write operation address.
- (2) It is recommended to turn off the global interrupt enable GIE.
- (3) Clear the wdt count value.
- (4) Write the unlock sequence to the WRPRT register to unlock the write protection.
- (5) Configure WORK write operation data.
- (6) Execute the instruction TBLP XH operation.
- (7) After the write operation is completed, check the CHKRSLT bit of the ISPCON1 register. When this bit is 0, it indicates that the write operation is verified.

Failed, the read data is inconsistent with the written data. When this bit is 1, it indicates that the write operation is successful.

EEPROM Read Operation

- (1) Configure {EADRH, EADRL} write operation address.
- (2) Execute the MOVP instruction.
- (3) Reading is completed.

Note: (1) Before unlocking, you need to disable global interrupts.

- (2) After the write unlock is successful, writing to other address registers will clear the WRPRT register to 0, causing the unlock to fail.

Therefore, in the C program, variable assignment is not allowed after unlocking successfully. Variable assignment must be placed before unlocking. The following is the solution

Lock failure case:

```
EADRH = address_buf[0]; // write address
```

```
EADRL = address_buf[1];
```

```
GIE = 0; // disable interrupt
```

```
WRPRT = 0x96; // unlock
```

```
WRPRT = 0x69;
```

```
WRPRT = 0x5A;
```

```
while(!WRPRTF);
```

```
write_data = 0xAA;
```

```
asm ("TBLP 0H"); //write data
```

```
asm ("NOP");
```

```
asm ("NOP");
```

```
if(CHKRSLT);
```

Since the write_data variable is assigned a value after write unlock, write unlock fails and data cannot be written to the EEPROM. Correct case

as follows:

```
EADRH = address_buf[0]; // write address
```

```
EADRL = address_buf[1];
```

```
GIE = 0; // disable interrupt
```

```
write_data = 0xAA;
```

```
WRPRT = 0x96; // unlock
```

```
WRPRT = 0x69;
```

```
WRPRT = 0x5A;
```

```
while(!WRPRTF);
```

```
asm ("TBLP 0H"); //write data
```

```
asm ("NOP");
```

```
asm ("NOP");
```

```
if(CHKRSLT);
```


(3) It is recommended to enable the watchdog module before unlocking to prevent the chip from being unable to recover due to abnormal conditions during the EEPROM writing process.

Since the write operation takes a long time, it is recommended to clear the wdt count to 0 before unlocking.

2.4 Problem of excessive chip sleep current

After the CPU executes the sleep instruction, all oscillators stop working until an interrupt event occurs to wake up the CPU.

The power consumption is about 1uA.

In order to minimize the power consumption of the CPU in sleep mode, before executing the sleep instruction, it is necessary to ensure that all input ports are

Connect to VDD or VSS level. The ADC module must be turned off in SLEEP mode, and the analog input channel cannot be configured as

1/8VDD (CHS[3:0] bits of SRADCON2 register cannot be configured as 0101).

Before going into sleep mode, ensure that the IO port set as digital input is not allowed to be left floating, and enable the internal pull-up resistor or pull-down resistor.

The standby power consumption is relatively high.

Eg

```
void csu32m10_enter_sleep(void)
```

```
{
```

```
    //PT1 io config
```

```
    PT1 = 0;
```

```
    PT1EN = 0xF7; // set PT1.3 input mode PT1PU =
```

```
    0x08; // PT1.3 pull up
```

```
    PT1CON = 0x14; //enable PT1.3 INT1
```

```
    PT1PD = 0;
```

```
    //PT3 io config
```

```
    PT3 = 0;
```

```
    PT3EN = 0x3F;
```

```
    PT3PU = 0;
```

```
    PT3CON = 0;
```

```
    PT3PD = 0;
```

```
    //PT5 io config
```

```
    PT5 = 0;
```

```
PT5EN = 0x03;
```

```
PT5PU = 0;
```

```
PT5CON = 0;
```

```
PT5PD = 0;
```

```
CST_WDT = 1; // disable wdt clk
```

```
TM0CON = 0; // disable timer0
```

```
TM2CON = 0; // disable timer2
```

```
TM3CON = 0; // disable timer3
```

```
SRADCON1 = 0; // disable ADC
```

```
wakeup_check:
```

```
while(!PT1_3);
```

```
E1IF = 0; //clr int1 flag
```

```
E1IE = 1; // enable int1
```

```
GIE = 1;
```

```
asm("sleep"); // enter sleep mode
```

```
delay_us();
```

```
E1IE = 0;
```

```
goto wakeup_check;
```

```
}
```

Note: If you need to enter sleep mode, you need to disable the ICD function in the program burning option when burning the code, otherwise the power consumption may increase.

Exception, as shown in the figure:

CS-Write code option configuration:



CodeOption

指令周期

- ☒ 指令周期=4个时钟周期
- ☐ 指令周期=8个时钟周期
- ☐ 指令周期=16个时钟周期
- ☐ 指令周期=32个时钟周期

复位引脚

- ☐ PT1.3作为复位引脚
- ☒ PT1.3作为普通输入输出

限流代码选项

- ☐ PT1.1、PT1.5口驱动能力配置为5mA
- ☒ 以上IO驱动能力为正常值

WWDT

- ☐ WWDT在HALT模式下继续计数，计数溢出后也会产生复位。
- ☒ WWDT在halt模式下不进行计数。

下拉代码选项

- ☐ PT3.4、PT5.1接10K下拉电阻，PT1.0接1K下拉电阻，PT1.3口接400KΩ下拉电阻
- ☒ 以上IO均不接下拉，驱动能力为正常值

WDT

- ☒ WDT模块使能和内部32K低速振荡器使能由软件配置
- ☐ WDT模块使能和内部32K低速振荡器使能固定打开，软件无法修改。

低功耗模式配置位

- ☒ SAR_ADC配置为正常功耗模式
- ☐ SAR_ADC配置为低功耗模式

内部晶振选择(ICK_SEL)

- ☐ 内部晶振频率2MHz
- ☐ 内部晶振频率4MHz
- ☐ 内部晶振频率32MHz
- ☒ 内部晶振频率8MHz
- ☐ 内部晶振频率16MHz

LVD

- ☒ 2.0V上电/掉电复位。低电压检测关闭。
- ☐ 2.0V上电/掉电复位。
- ☐ STATUS的LVD24作为2.4V的低电压检测器。
- ☐ STATUS的LVD36作为3.6V的低电压检测器。
- ☐ 2.4V上电/掉电复位。
- ☐ STATUS的LVD36作为3.6V的低电压检测器。
- ☐ 3.6V上电/掉电复位。

DRIVE

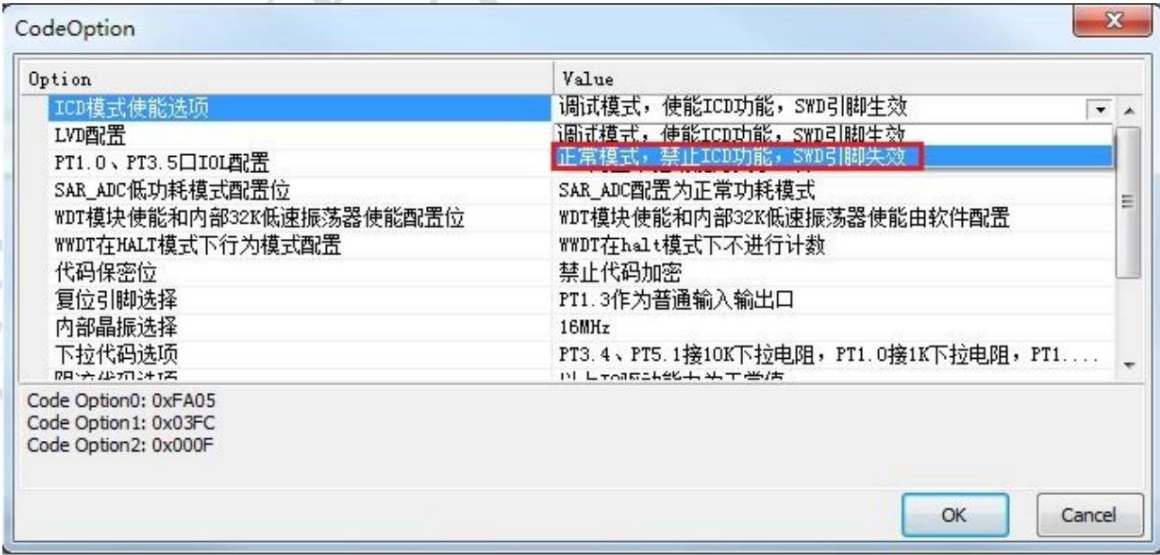
- ☒ IOL为正常驱动能力大小，即20mA
- ☐ IOL为30mA

ICD_CFG

- ☐ 使能ICD功能
- ☒ 禁止ICD功能

确定 **取消** ☐ Protect Code Memory

IDE code option configuration:



CodeOption

Option	Value
ICD模式使能选项	调试模式，使能ICD功能，SWD引脚生效
LVD配置	调试模式，使能ICD功能，SWD引脚生效
PT1.0、PT3.5口IOL配置	正常模式，禁止ICD功能，SWD引脚失效
SAR_ADC低功耗模式配置位	SAR_ADC配置为正常功耗模式
WDT模块使能和内部32K低速振荡器使能配置位	WDT模块使能和内部32K低速振荡器使能由软件配置
WWDT在HALT模式下行为模式配置	WWDT在halt模式下不进行计数
代码保密位	禁止代码加密
复位引脚选择	PT1.3作为普通输入输出
内部晶振选择	16MHz
下拉代码选项	PT3.4、PT5.1接10K下拉电阻，PT1.0接1K下拉电阻，PT1.3口接400KΩ下拉电阻

Code Option0: 0xFA05
Code Option1: 0x03FC
Code Option2: 0x000F

OK **Cancel**

2.5 Precautions for using analog comparator for short circuit protection

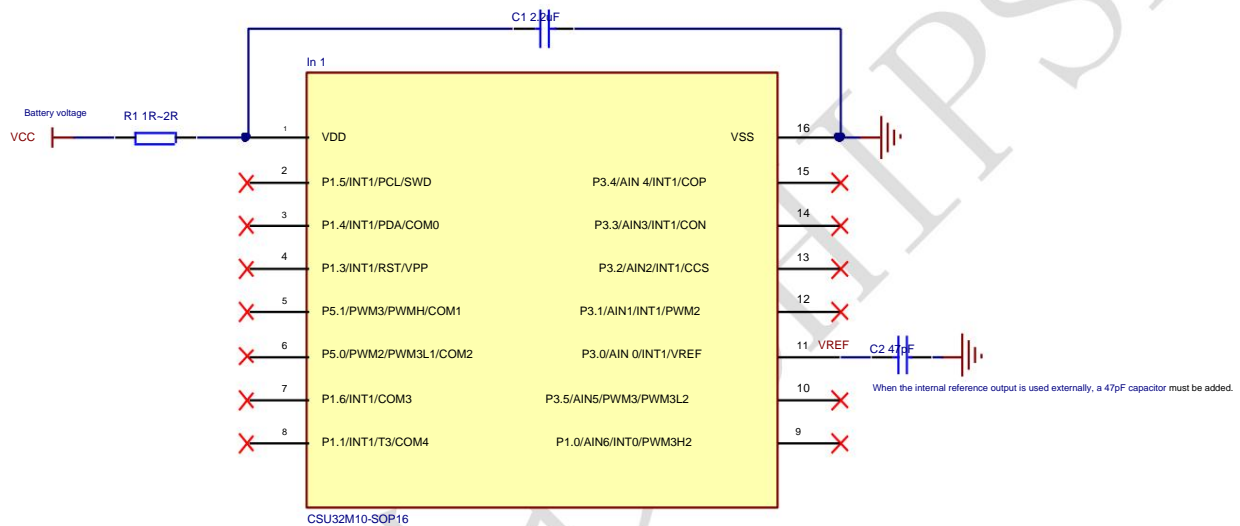
CSU32M10 has an internal integrated analog comparator, and the comparator result output can be used to trigger the PWM signal shutdown function as an electronic cigarette load

Short circuit protection, usually the protection time is 5~20us (related to the filter time of the comparator configuration). When using this module, please pay attention to the following issues:

Question: A 1~2 Ω resistor should be connected in series between the chip VDD and the power supply voltage input terminal. A filter capacitor of 2.2uF or more is recommended.

The resistance will cause the chip VDD voltage to oscillate, resulting in abnormal chip operation and a longer short-circuit protection response time. The reference circuit is shown in the figure below.

Show:



3 Precautions for using IDE and CS-LINK

1. Code option configuration ICD mode can be enabled or disabled, and ICD debugging can be performed. The difference is:

Enable: PT1.5 is still a debug port when not in debug mode, IO, interrupt and other functions cannot be used normally, and the high-speed clock is on during sleep mode.

Cannot be closed.

Disable: PT1.5 functions normally when not in debugging state, and high-speed clock is normally turned off when in sleep mode.

In this mode, the code option should disable the ICD debug function.

2. If the PT1.3 port on the debug board is not used as a reset port or has no pull-up function, disable the code option to configure the PT1.3 port as a reset port.

3. Code option configuration, [Code Confidentiality Bit] is configured as [Enable Code Encryption], ICD debugging will not be possible, and burning will not be possible.

record.

4. When debugging is stopped in the halt or sleep state, the PC cannot be modified, i.e. functions such as jumping to the cursor are disabled.

effect.

5. When the WWDT register is in debug stop state, the WWDT related registers cannot be modified in the window.

6. At a lower instruction cycle (less than 500KHz), the PC modification operation cannot be performed, that is, functions such as jumping to the cursor are invalid.

7. It is recommended to isolate the circuit of the debug port PT1.5 on the debug board and not connect other circuits when using it as a debug port.

8. CS_Link has added pull-up resistors to the debug port. It is forbidden to connect pull-down resistors to the debug port on the debug board.

9. It is not allowed to set a breakpoint at the next instruction of tb1p and movp.

10. When the clrf f instruction is executed, the address f is read and written. Therefore, when the address f meets the breakpoint condition, clrf will be triggered.

Read or write operation breakpoint.

11. Data breakpoint addresses cannot be configured as work registers.

12

The call stack function is only valid for function calls. Double-click the call stack window information to jump to each level of function call.

If you double-click the call stack information cursor to jump to a non-function call entry, the call stack information is an interrupt

This causes the call stack information to be invalid.

13. The difference between ICD debugging and ICE debugging: ICD cannot modify the system register value in the IDE window (SFR address 00-08h, but not including INTE/INTF registers, GIE in INTE cannot be modified), but ICE can be modified.

14. When the data breakpoint address is sram, when debugging stops, the next instruction meets the data breakpoint in the sram address area.

(i.e. the cursor stops at the code line), debug run will not trigger the breakpoint. As shown in the following code, the breakpoint is set to write sram 80 address

The breakpoint is triggered when the debugger stops at the second line (not executed). The breakpoint will not be triggered in debug run.

This phenomenon does not occur when the data breakpoint address is SFR.

```
movlw      0x55
```

```
movwf     sram_80
```

15. Currently, CS-LINK (firmware version: V0.6) and previous versions do not support external power supply and need to be turned off during burning or simulation.

External power supply.

16. In halt mode, since the core clock is stopped, the wwdtif flag cannot be set when the window watchdog is reset.