



深圳驰芯微电子有限公司

地址：深圳市南山区高新区南区粤兴三道 8 号中国地质大学中地大楼 C606

电话：0755-66632617

传真：0755-66635972

CW1802

Datasheet



1.1 高性能的 RISC CPU

-4K words OTP 芯片

- 除跳转指令外的所有指令都是单周期的
- 工作速度：
- 振荡器/ 时钟的输入频率为 DC – 24 MHz
- 指令周期为 DC – 167 ns
- 中断功能
- 8 级深硬件堆栈
- 直接、间接和相对寻址模式

单片机特性

- 高精度内部振荡器：
- 出厂时精度校准为 $\pm 1\%$
- 可用软件选择的频率范围为 125 kHz 到 24 MHz
- 节能的休眠模式
- 宽工作电压范围。（2.0V-5.5V）
- 上电复位（Power-on Reset ,

POR）

- 与上拉/ 输入引脚复用的主复位功能

低功耗特性

- 待机电流：
- 当电压为 2.0V 时，典型值为 5uA
- 工作电流：
- 当频率为 125 kHz、电压为 2.0V 时，典型值为 150uA
- 当频率为 1 MHz、电压为 2.0V 时，典型值为 400uA
- 看门狗定时器电流：
- 当电压为 2.0V 时，典型值为 5uA

外设特性

- 具有 6/9/10/11 个独立方向控制的 I/O
- 引脚：
- 高灌/ 拉电流可直接驱动 LED
- 引脚电平变化中断
- 独立的可编程弱上拉
- A/D 转换器：
- 10 位分辨率
- Timer0：带 8 位可编程预分频器的 8 位定时器/ 计数器
- 增强型 Timer1：
- 带有预分频器的 16 位定时器/ 计数器
- Timer1 的时钟源可选外部时钟。
- Timer2：带 8 位周期寄存器、预分频器和后分频器的 8 位定时器/ 计数器
- PWM 模块：
- 10 位 PWM 模块，最大频率为 20 kHz
- USB2.0 Full Speed Device
- SPI Master
- T key 扫描 电路



深圳驰芯微电子有限公司

地址：深圳市南山区高新区南区粤兴三道 8 号中国地质大学中地大楼 C606

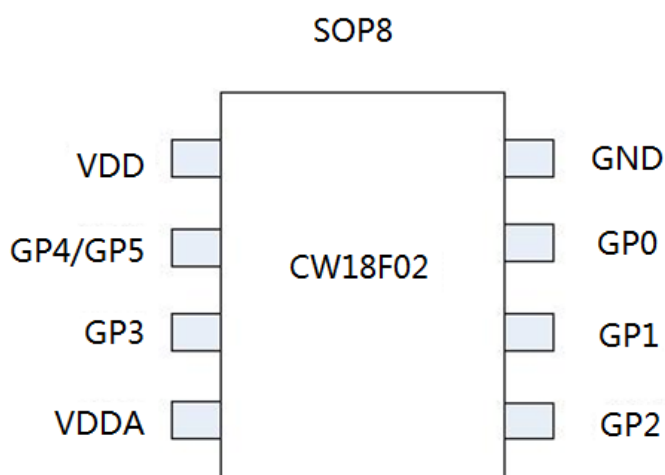
电话：0755-66632617

传真：0755-66635972

1.2 CW18FXX 系列为您提供以下型号及功能

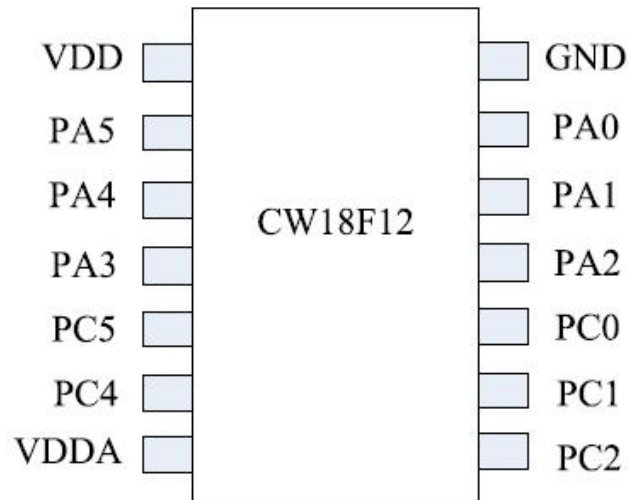
功能 型号	工作 电压	USB	ADC	I/O	OTP (words)	Sram (bytes)	上/掉 电 复 位	PWM	定时器 8/16 位	封装
CW18F02	2.0-5.5 V		4	6	4K	256	√	1	2/1	Sop8
CW18F12	2.0-5.5 V		8	11	4K	256	√	2	2/1	Sop14
CW18F22	2.0-5.5 V		8	11	4K	256	√	2	2/1	Sop16
CW18F32	2.0-5.5 V	√	7	12	4K	256	√	2	2/1	Sop16

1.3 封装尺寸图

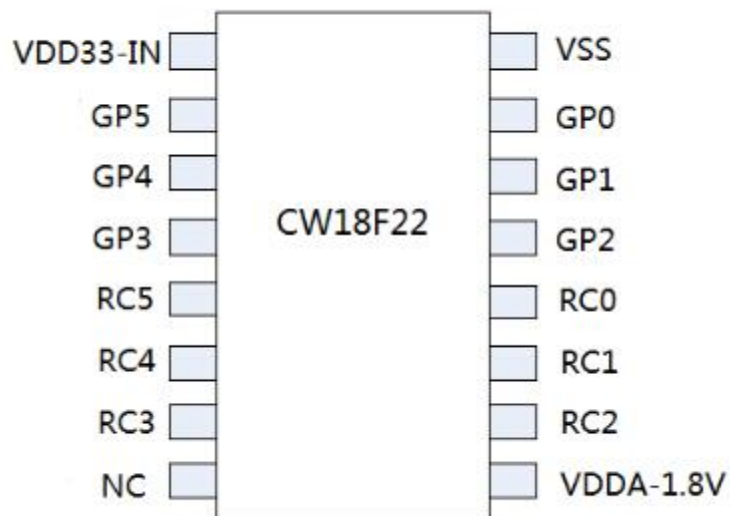




SOP14

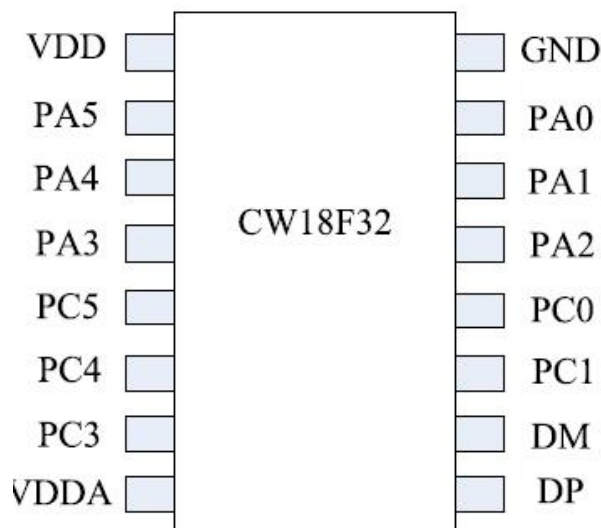


SOP16





SOP16



2 简介

CW1802 的 Core 兼容 16F684 所有指令，兼容 16F684 的大部分外围模块（硬件和软件），并在这基础上加入了：

- USB2.0 Full Speed Device。
- SPI Master。
- T key 扫描 电路。
- 增加了 2 个 GPIO，就是 USB 的 DP/DM。

CW1802 有 16F684 的大部分外围模块，它的 Memory Map 也基本一样，但下面不一样：

- Flash 换成了 OTP，并扩展到 4K Words。
- 没有了 EEPROM。
- 没有了电压比较器。
- SOP8 封装使用要注意：PIN2 可以选 GP4 或者 GP5 使用，选 GP4, GP5 要设置成输入；如果选 GP5, GP4 就要设置成输入。
- RortC 口上拉设置：95H (WPU) 寄存器的 Bit3 (0-7) 要设成 1，RC 会同时上拉。
- PortC 口输入设置：8DH 寄存器的 bit4 (0-7) 要设成 0，从 86H 寄存器读取 IO 值。
- PortC 口设置 IO 输出电平时不能用位操作。
- USB、SPI 和 T 型按键有部分寄存器不能用位操作：9DH, 98H, 19H, 88H, 93H, 写 9AH 和 9CH 时不能用位操作，但读 9AH 和 9CH 时，可以用

BTFSS 和 BTFSC 指令

- 程序超过 2K 时需要切换 ROM BANK，操作如下：

1) 使能切换 ROM BANK，把 89H 寄存器的最高位置 1。MOVLW 0x80 ---->MOVWF 89H

2) 把 PCLATH 的第 3 位置 1。MOVLW 0x08 ---->IORWF PCLATH, F

3) 通过跳转指令 CALL 或者 GOTO 就能切换到第二个 ROM BANK.

- 4) 想切换回到第一个 ROM BANK 时，先把 PCLATH 的第 3 位清 0:

MOVLW 0xF7 ----> ANDWF PCLATH, F 。再通过跳转指令 GOTO 或者 RETUEN，切换到第一个 ROM BANK。

3 USB Device

USB 有 4 个 endpoint, 是固定设置，不能改变。设置如下：

- 第一个是 control，其它 3 个是 IN
- 最大的 Packet 是 8 个 byte
- 第一个 endpoint 有一个输出 FIFO，一个输入 FIFO，各 8 bytes
- 其他的 endpoint 有一个 IN FIFO，各 8 bytes
- 软件把 IN FIFO 写满或写一部分，等空了（中断或标志查询）再写。如果发小于 8 个数据，把数据写到 FIFO 的后半部分，最后再写一次第一个要发的地址，把内部指针指向在第一个要发的数据，当 IN token 一来，就从这个数据一直发送到第 8 个数据。指针先增加，后写数据

这是写 endpoint 3 的例子，只发 4 个数据。

```
;; lock ds
MOVLW 10H
MOVWF 9CH
; set fifo addr starting point
MOVLW 1BH ; 注意，这里指向低 3 个数据，先加指针
MOVWF 9DH
MOVLW 34H ; 注意，这里写了低 4 个数据
MOVWF 9BH
MOVLW 35H
MOVWF 9BH
MOVLW 36H
MOVWF 9BH
MOVLW 37H
MOVWF 9BH
; set where should the star point,note it is start pointer-1
MOVLW 1B ; here set the start pointer to 3,
will send bytes 4 to 7
MOVWF 9DH
```

```
;; do 1 dummy write
MOVLW 34H
MOVWF 9BH
```

- 软件等 OUT FIFO 满了（中断或标志查询），可以把数据读出，做判断，等的时候可以做什么事。OUT 还有一个非空标志，软件可以判断有数据在 FIFO 里，就读，这主要是为小于 8 个数据的包做处理。

软件只需要等到收到 Package，读出 package 和一些状态位，对 package 进行判断，然后把响应的 package 写到对应的 FIFO 里。

要写/读 FIFO，先设 FIFO 地址 register, 0x9D

Bit 5-7	Bit 3-4	Bit 0-2
Not Used	FIFO 对应的 Endpoint number，有效数是 0 到 3，第一个 endpoint 有两个 FIFO，都应设 0，写是写到 OUT FIFO，读是读 IN FIFO，其他的 endpoint 只能写	一个 FIFO 里 byte offset，设了 offset 后，可以连续读/写，offset 会自动增减，到了 7，会回到 0.

设了 FIFO 的地址后，就可以通过 0x9B 读/写 FIFO.

Register 0x9A 是第一个 USB configure register, 读的时候可以读出 USB 状态，定义如下：

USB Configure Register 1 (0x9A)			
Bit number	name	Defaults	描述
0	Enable PHY/FIFO0 IN empty	0	写 1: 使能 USB PHY，写 0: power down PHY; 读: 1 是 endpoint 0 IN FIFO empty
1	USB speed/ FIFO0 OUT FULL	1	写 1: full speed, 写 0: low speed; 读: 1 是 endpoint 0 OUT FIFO full
2	PHY suspend/ FIFO1 IN empty	0	写 1: suspend PHY, 读: 1 是 endpoint 1 IN FIFO empty
3	DM pull up/ FIFO0 OUT not empty	0	写 1: enable DM 内部 pull up 读: 1 是 endpoint 0 OUT FIFO has data
4	Resume/ FIFO2 IN empty	0	写 1: 在 suspend 后, resume USB,需呀手动清零; 读: 1 是 endpoint 2 IN FIFO empty
5	Soft reset	0	写 1: 复位整个 USB，一定要写

			0 才能放开;
6	Pull Down/ FIFO3 IN empty	1	写 1: pull down DP/DM; 读: 1 是 endpoint 3 IN FIFO empty
7			;

Register 0x9C 是第二个 USB configure register, 读的时候可以读出 USB 状态, 它有 4 个 registers, 要读/写其中一个, 要先设定 0x9D 地址 register 来选定 offset, 定义如下:

USB Configure Register 2 (0x9C, offset=0)			
Bit number	name	Defaults	描述
0	Stall EN0/Setup	0	写 1: Stall endpoint 0; 读: 1 是指收到的 package 是 Setup
1	Stall EN1/Softreset	0	写 1: Stall endpoint 1; 读: 1 是指收到了 softreset 指令
2	Stall EN2/USBreset	1	写: Stall endpoint 2 读: 1 是 USB bus 在 reset 状态
3	Stall EN3/suspend	0	写 1: Stall endpoint 3; 读: 1 是 USB bus 在 suspend 状态
4	LOCK FIFO/NACK	1	写 1: 要是 USB 要求 FIFO, 并且 FIFO_ADDR[4:3] 也指向这个 endpoint, 就回 NAK, USB 不会对 FIFO 读/写, 这样防止 FIFO 同时读/写产生错误。 读: 1 是收到了 NACK
5	DPPU_LO/ACK	只读	写 1: 使能 DP weak pull up 1: 是收到了 NACK
6	Toggle/SOF	只读	写 1, FIFO_ADDR[4:3] 指向的 endpoint 会被强迫翻转 data0/data1。 注意: FIFO_ADDR[4:3]=0 指向 EP1 FIFO_ADDR[4:3]=2 指向 EP2 FIFO_ADDR[4:3]=3 指向 EP3 1: 是收到 SOF (star of frame)
7	Disable OTP power down	0	写 1, 在睡眠状态时, 不关闭 OTP

USB Configure Register 2 (0x9C, offset=1)			
Bit	name	Defaults	描述

number			
2:0	Alternate Interface	只读	Set_Interface 命令设的 Alternate Interface number
4:3	Interface	只读	Set_Interface 命令 设的 Interface number
6:5	Configure	只读	Set Configure 命令设的 configure value
7	Not Used	0	

USB Configure Register 2 (0x9C, offset=2)			
Bit number	name	Defaults	描述
7: 0	Time Stamp	只读	当检测到 SOF 时，Bit 0 to 7 of Time stamp

USB Configure Register 2 (0x9C, offset=3)			
Bit number	name	Defaults	描述
2: 0	Time Stamp	只读	当检测到 SOF 时，Bit 8 to 10 of Time stamp
5: 3	Not Used		
6	DM	只读	DM 的值，当把 DM 当 GPIO 用时， 可通过这位来读 DM 脚的值
7	DP	只读	DP 的值，当把 DM 当 GPIO 用时， 可通过这位来读 DP 脚的值

USB Configure Register 2 (0x9C, offset=6)			
Bit number	name	Defaults	描述
0	NAK0	只读	Endpoint 0 有 NAK (IN 没有 data 在 FIFO 里， OUT 时 FIFO 不空， 读时清 0)
1	NAK1	只读	Endpoint 1 有 NAK (IN 没有 data 在 FIFO 里)， 读时清 0
2	NAK2	只读	Endpoint 2 有 NAK (IN 没有 data 在 FIFO 里)， 读时清 0
3	NAK2	只读	Endpoint 3 有 NAK (IN 没有 data 在 FIFO 里)， 读时清 0

USB 中断

所有 USB 的 Full 和 Empty 状态产生一个 USB 总中断， 但每个 USB 中断源可以单独使能。

总中断的标准位是 0x0C (PIR1) 寄存器的 bit 2 (最低位是 0)， 必须要往这位写 “0” 才能清掉。

USB 里的中断源有

- Endpoint 0-3. IN FIFO 空
- Endpoint 0, OUT FIFO 满。
- Endpoint 0-3 有 NAK
- USB Host reset CW1802
- 收到 setup 包

要使能 USB 的总中断，要往 **0x8C (PIE1)** 寄存器的 bit 2 (最低位是 0) 写 “1”，要单独使能各个中断，配置下边的寄存器。

USB Configure Register 2 (0x98)			
Bit number	name	Defaults	描述
0	Empty 0 EN	0	1: 使能 Endpoint 0 IN FIFO 空中断
1	Empty 1 EN	0	1: 使能 Endpoint 1 IN FIFO 空中断
2	Empty 2 EN	0	1: 使能 Endpoint 2 IN FIFO 空中断
3	Empty 3 EN	0	1: 使能 Endpoint 3 IN FIFO 空中断
4	Full 0 EN	0	1: 使能 Endpoint 0 OUT FIFO 满中断
5	保留	0	
6	CLR ACK INT FLAG	0	写 0 来清除 ACK INT 标志位，不需要清除时设为 1
7	EN NEW INT	0	写 1 来使能新的中断

USB Interrupt Enable Register 2 (0x19)			
Bit number	name	Defaults	描述
0	NAK EN	0	1: 使能 NAK 中断, 任何一个 endpoint NAK 都会产生中断
1	5V LDO Disable	0	1: 禁用 5V LDO, 省功耗
2	保留	0	1: 使能 Endpoint 2 IN FIFO 空中断
3	Setup EN	0	1: 使能 USB setup 中断
4	ACK INT EN	0	1: 使能 USB ACK 中断

注意：使能 USB ACK 中断时，98H.6 位应设为 1，进入中断后，98H.6 位设为 0，清掉中断标志位。

USB 强迫 Data0/Data1 翻转

如果应用需呀强行把某个 endpoint data0/data1 的包换成 data1/data0，把 fifo-add[4:3] 设置到这个 endpoint，把 0x9C 的 bit 6 设为 1。

DP 和 DM 也可以用来做普通的 GPIO，但要用它们，必须要把 USB 使能，这会增加 8ma 的功耗。参考下边红色的寄存器设计

PORT C (0x87)			
Name	Bit	Reset Values	Descriptions
RC pin number	[5:0]	0	PORTC 的方向（输出、输入，和 684 一样的）
USB GPIO mode	6	0	1: 会强迫 DP/DM 做为 GPIO，输出模式

DP DM Value (0x85)			
Name	Bit	Reset Values	Descriptions
DP	6	0	DP 数出的值
DM	7	0	DM 数出的值

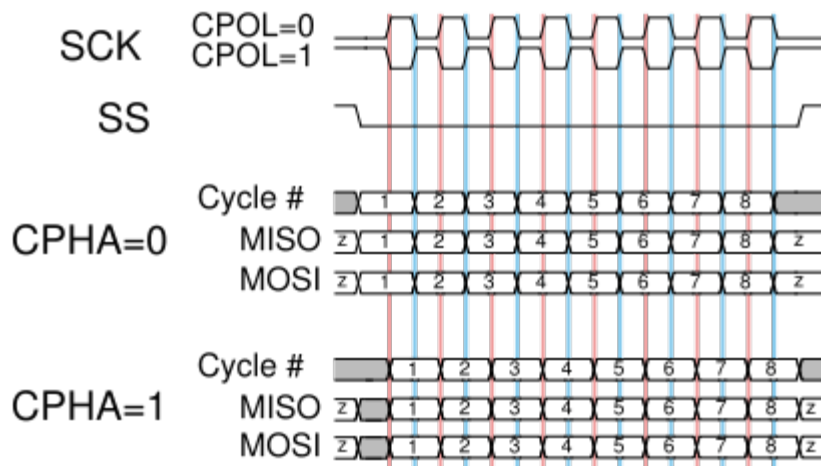
要是 DP/DM 作为 GPIO 输入，不用设置，只要把 USB enable, 读 USB Configure Register 2 (0x9C, offset=3, bit 6 和 7 就好

1. LDO

CW1802 有两个 LDO，一个输出 1.8V，用作 core 供电，输入电压从 2V 到 5V，另一个输出 3.3V，给 USB 供电，同时会输出 3.3V 到管脚，可以给其他芯片供电，最大电流是 100ma，同时也可以把他的输出电压接到 1.8V LDO 的输入。如果不用 USB，可以不用 5.5V LDO，用 0x9C 寄存器（USB Configure Register 2, x9C, offset=0）的 bit 7 把他关掉，做到功耗最低。

2. SPI Master

SPI module is SPI master. Like SPI standard, the clock phase and polarity are configurable via configure registers. The following figure shows the timing diagram of SPI with 4 different configure combinations.



SPI has 4 interface pins: chip select, clock, data output and data input.

SPI generates interrupts when it completes sending and receiving. By default, this interrupt is masked. SPI can run up to 16MHz. SPI speed can be changed by "Clock divider bits" in "SPI Control Registers".

SPI Control Register 0 (0x88)			
Name	Bit	Reset Values	Descriptions
Reserved	[7:6]	0x00	
SPI Output enable	5	0	1: enable all SPI outputs
Read/write	4	1	1: read mode, 0: write mode; Should write 0, 看注解
CPOL	3	0	Clock polarity.
CPHA	2	0	Clock phase. Clock phase and clock polarity combinations have 4 different timing.
Clock divider low	[1:0]	0	Combine with the 'clock divider high' in SPI Control 1, it determines the SPI clock rate. This setting is low 2 bits. 0x0: 2 0x1: 8 0x2: 16 0x3: 32

SPI Status Register 0 (Read Only 0x89)			
Name	Bit	Reset Values	Descriptions
Reserved	[7:5]	0x000	
Interrupt	4	0	Interrupt flag, clear on write
RBUF full	3	0	1: read buffer is full; 0: not empty.
WBUF empty	2	0	1: write buffer is empty; 0: not empty.
Interrupt count	1:0	0	当 R FIFO 收到了这么多次的数据时，产生中断。

SPI Read Buffers (0x94)			
Name	Bit	Reset Values	Descriptions
Write	[7:0]	0	Data received

SPI Write Buffers (0x93)			
Name	Bit	Reset Values	Descriptions
Write	[7:0]	0	Data to send

注:

- 1、SPI 是全双工的，就是发和收都会同时进行，所以应该把 SPI 放在写模式，SPI 只有在写 FIFO 有数据才开启，要收，也往写 FIFO 里写一个数，RBU full 和 WRBUF empty 两个标志位只需要看一个就可以，因为 FULL 和 empty 总是同时发生
- 2、SPI 默认是 RA 口，使用 RC 口时，需要把 OPTION 的第 7 位（0-7）清零，并打开 RC 上拉，95H(WPU)的第 3 位（0-7）置 1。

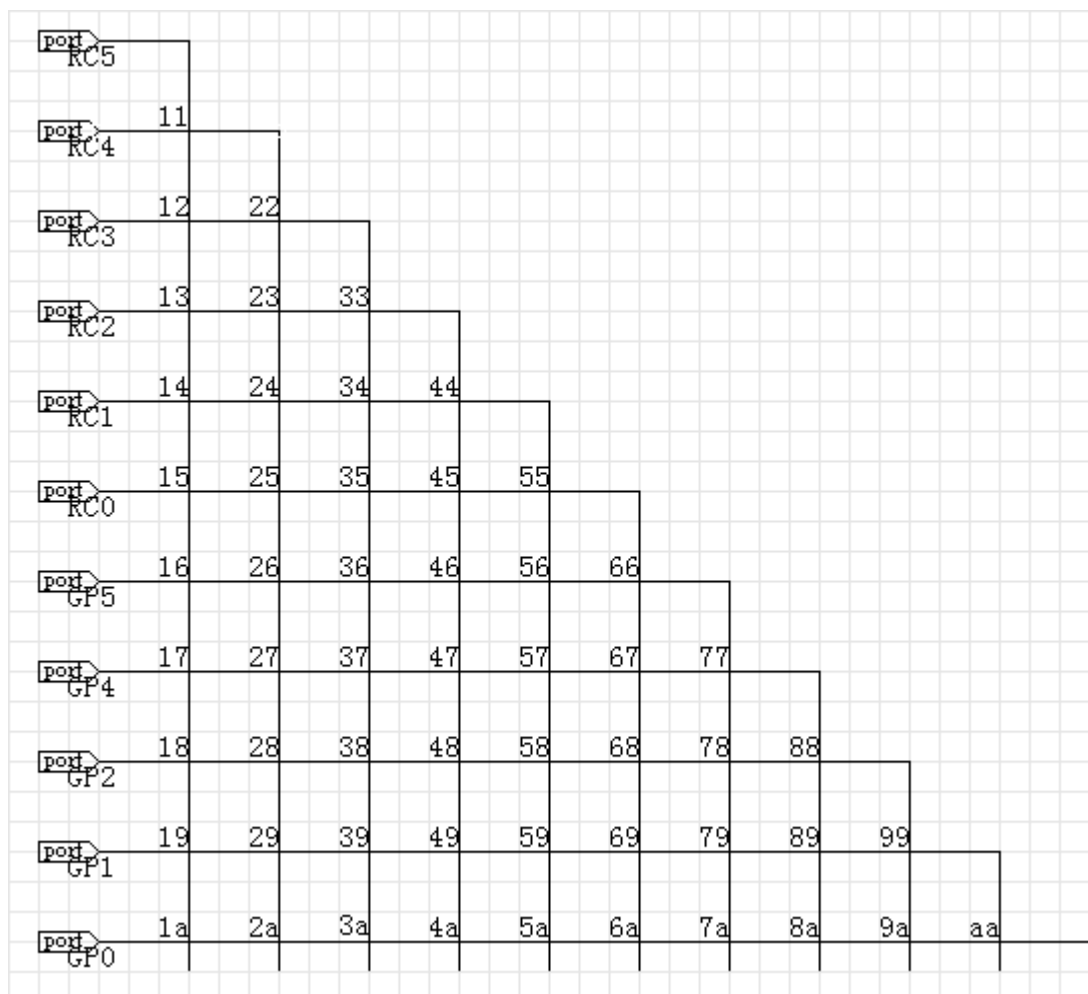
3. T-Key Scan

梯形按键扫描共有 11 个 IO 口，IO 的排列是 RC5，RC4，RC3，RC2，RC1，RC0，GP5，GP4，GP2，GP1，GP0，8DH【3:0】设置的越多，GPIO 用于扫描的就越多。T-key scan 模块可以 configure 多少个这些 GPIO 来用作 T-key scan。如设置 N 个 GPIO 用作 scan，前边的 N 个 GPIO 会被使用，后边的可以用作别的用途。

Scan Configure (0x8D)			
Name	Bit	Reset Values	Descriptions
Scan pin number	[3:0]	0	多少 GPIO 用来做 SCAN，有效值 0-11,0 是 1 个
Scan enable	4	0	1: 使能 T-scan
SRAM Address bit 7	5	0	SRAM 有 256 bytes，要使用后边 128 bytes，先设这位为 1.
Strong Drive GP0	6	1	1: 使能 GP0 能大电流驱动（200ma）
USB test mode	7	0	1: 把 USB 的内部信号送到 GPIO 上（RC0-RC5）

Scan Results (0x86)			
Name	Bit	Reset Values	Descriptions
Keys	[7:0]	0	Key scan 结果

外接电路如下图，每个交叉点代表一个按键，交叉点的数值是当按下按键时地址 86H 就被写成对应的值



DP 和 DM 也可以用来做普通的 GPIO，但要用它们，必须要把 USB 使能，这会增加 8ma 的功耗。参考下边红色的寄存器设计

PORT C (0x87)			
Name	Bit	Reset Values	Descriptions
RC pin number	[5:0]	0	PORTC 的方向（输出、输入，和 684 一样的）
USB GPIO mode	6	0	1: 会强迫 DP/DM 做为 GPIO，输出模式

DP DM Value (0x85)			
Name	Bit	Reset Values	Descriptions
DP	6	0	DP 数出的值
DM	7	0	DM 数出的值

要是 DP/DM 作为 GPIO 输入，不用设置，只要把 USB enable, 读 USB Configure Register 2 (0x9C, offset=3, bit 6 和 7 就好)。

