RR 配置 1602

这几天在 RR 上折腾实现 1602 液晶，将配置过程记录下来，备考。
本文以 GPIO 直接驱动 1602，考虑到引脚的数量，采用的是 4 位驱动模式。

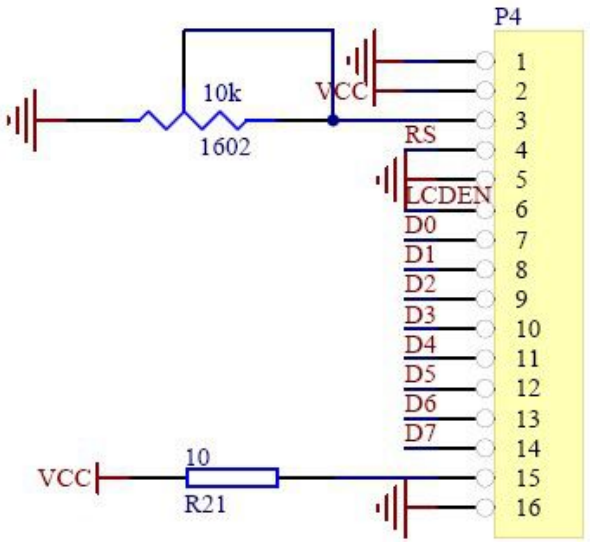
预备知识

首先要熟悉 GPIO 的基本用法，RR 提供了两组 GPIO 接口，分别是 J8 及 J12，关于各引脚的编号及含义请参考官方文档，本文不赘述。
其次要熟悉 LCD1602 的硬件参数及连接方法，下面择其要点进行介绍。

接口说明

接口信号说明：

编号	符号	引脚说明	编号	符号	引脚说明
1	VSS	电源地	9	D2	Data I/O
2	VDD	电源正极	10	D3	Data I/O
3	VL	液晶显示偏压信号	11	D4	Data I/O
4	RS	数据/命令选择端（H/L）	12	D5	Data I/O
5	R/W	读/写选择端（H/L）	13	D6	Data I/O
6	E	使能信号	14	D7	Data I/O
7	D0	Data I/O	15	BLA	背光源正极
8	D1	Data I/O	16	BLK	背光源负极



为了调整 LCD 的背光，在引脚 15 处最好接一个可变电阻，以调整背光亮度，或使用 PWM 实现软件背光调整。

1602 命令集

	指令	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0
1	清屏	0	0	0	0	0	0	0	0	0	1
2	光标返回	0	0	0	0	0	0	0	0	1	*
3	输入模式	0	0	0	0	0	0	0	1	I/D	S
4	显示控制	0	0	0	0	0	0	1	D	C	B
5	光标/字符移位	0	0	0	0	0	1	S/C	R/L	*	*
6	功能	0	0	0	0	1	DL	N	F	*	*
7	置字符发生器地址	0	0	0	1	字符发生存贮器地址					
8	置数据存贮器地址	0	0	1	显示数据存贮器地址						
9	读忙标志和地址	0	1	BF	计数器地址						
10	写数据到指令7.8所设地址	1	0	要写的数据							
11	从指令7.8所设的地址读数据	1	1	读出的数据							

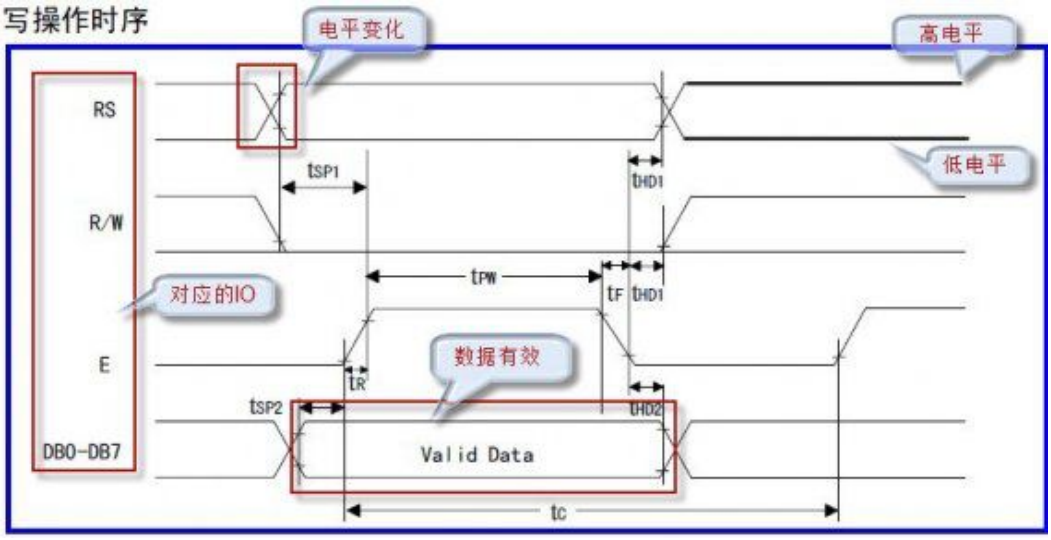
指令 1：清显示，光标复位到地址 00H 位置。
指令 2：光标复位，光标返回到地址 00H。
指令 3：光标和显示模式设置 I/D：光标移动方向，高电平右移，低电平左移，S:屏幕上所有文字是否左移或者右移。高电平表示有效，低电平则无效。
指令 4：显示开关控制。 D：控制整体显示的开与关，高电平表示开显示，低电平表示关显示 C：控制光标的开与关，高电平表示有光标，低电平表示无光标 B：控制光标是否闪烁，高电平闪烁，低电平不闪烁。
指令 5：光标或显示移位 S/C：高电平时移动显示的文字，低电平时移动光标。R/L，高向

左，低向右。
指令 6：功能设置命令 DL：高电平时为 4 位总线，低电平时为 8 位总线 N：低电平时为单行显示，高电
平时双行显示 F: 低电平时显示 5x7 的点阵字符，高电平时显示 5x10 的点阵字符。（有些模块是 DL：高电平时为 8 位总线，低电平时为 4 位总线）
指令 7：字符发生器 RAM 地址设置，地址：字符地址*8+字符行数。(将一个字符分成 5*8 点阵，一次写入一行，8 行就组成一个字符)

指令 8：置显示地址，第一行为：00H——0FH,第二行为：40H——4FH。
指令 9：读忙信号和光标地址 BF：为忙标志位，高电平表示忙，此时模块不能接收命令或者数据，如果为低电平表示不忙。
指令 10：写数据。
指令 11：读数据。

1602 时序图

2. 写操作时序



3. 时序参数

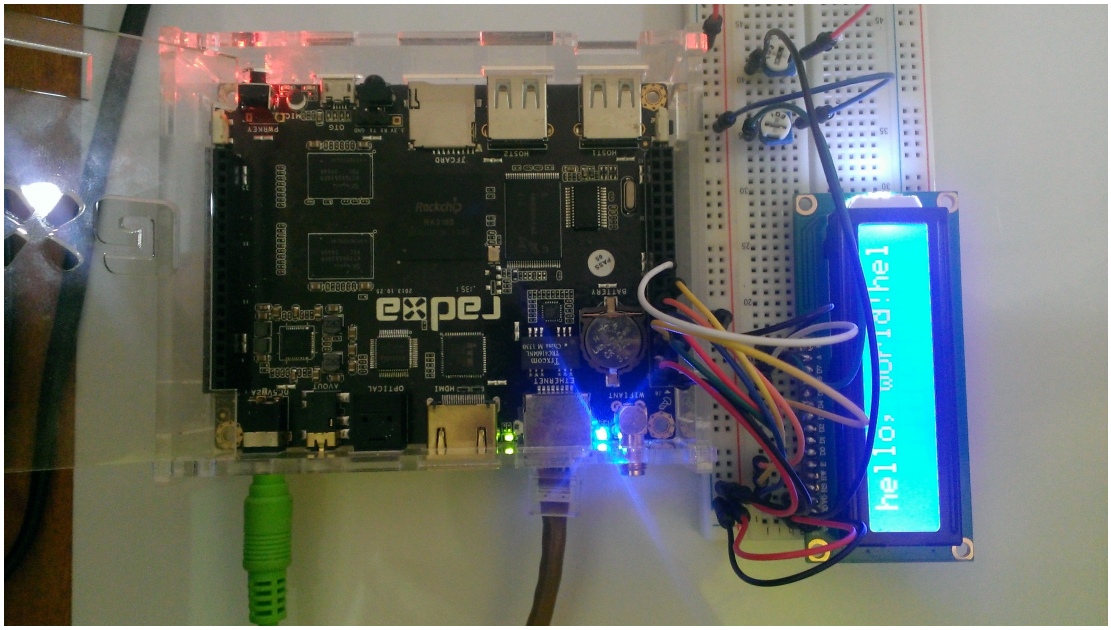
时序参数	符号	极限值			单位	测试条件
		最小值	典型值	最大值		
E 信号周期	t_C	400	–	–	ns	引脚 E
E 脉冲宽度	t_{PW}	150	–	–	ns	
E 上升沿/下降沿时间	t_R, t_F	–	–	25	ns	
地址建立时间	t_{SP1}	30	–	–	ns	引脚 E、RS、R/W
地址保持时间	t_{HD1}	10	–	–	ns	
数据建立时间(读操作)	t_D	–	–	100	ns	引脚 DB0～DB7
数据保持时间(读操作)	t_{HD2}	20	–	–	ns	
数据建立时间(写操作)	t_{SP2}	40	–	–	ns	
数据保持时间(写操作)	t_{HD2}	10	–	–	ns	

RR 连接 1602

使用 RR 上 J8 的 GPIO 口来连接 1602，用到的的各 GPIO 引脚如下

PIN 编号	GPIO 名称	1602 引脚	GPIO 内部编号	备注
7	GPIO0_A7	D4	167	
8	GPIO0_A6	D5	166	
9	GPIO0_B1	D6	169	
20	GPIO1_B5	D7	205	
12	GPIO3_D4	RS	284	
15	GPIO1_A2	RW	194	
14	GPIO1_A1	E	193	
	GND	VSS		
	5V	VDD		
	5V	A		可变电阻（可选）
	GND	VO		可变电阻（可选）

说明：RR 内部对部分 GPIO 口实现了复用，所以有些标示为 GPIO 的接口被其它程序占用，所以上面的连接看起来有点乱。



程序代码

说明：程序用到了 gpio 迷你库，请参考论坛中 GPIO 迷你库。
GPIO 迷你库：http://www.leikeji.com/thread-793-1-1.html

```
#include "rr_gpio.h"
#include <stdio.h>
#include <string.h>
#include <unistd.h>

#define D4 167
#define D5 166
#define D6 169
#define D7 205

#define RS 284
#define RW 194
#define E 193

void initPins(void)
{
   pinMode(RS, OUTPUT);
   pinMode(RW, OUTPUT);
   pinMode(E, OUTPUT);

   pinMode(D4, OUTPUT);
   pinMode(D5, OUTPUT);
   pinMode(D6, OUTPUT);
   pinMode(D7, OUTPUT);

}

void write4bits(char value)
{
   int dataPins[] = {D4, D5, D6, D7};
   int i;
   for(i = 0; i < 4; i ++)
   {
      digitalWrite(dataPins[i], (value >> i) & 0x01);
   }
}


void lcdWriteComm(char comm)
{
   digitalWrite(E, LOW);
   digitalWrite(RS, LOW);
   digitalWrite(RW, LOW);
```

```
    write4bits(comm >> 4);
    usleep(1);

    digitalWrite(E, HIGH);
    usleep(5);
    digitalWrite(E, LOW);

    write4bits(comm);
    usleep(1);

    digitalWrite(E, HIGH);
    usleep(5);
    digitalWrite(E, LOW);


}

void lcdWriteData(char data)
{
    digitalWrite(E, LOW);
    digitalWrite(RS, HIGH);
    digitalWrite(RW, LOW);

    write4bits(data >> 4);
    usleep(1);

    digitalWrite(E, HIGH);
    usleep(5);
    digitalWrite(E, LOW);

    write4bits(data);
    usleep(1);

    digitalWrite(E, HIGH);
    usleep(5);
    digitalWrite(E, LOW);

}

void initLCD(void)
{

    lcdWriteComm(0x28);
    usleep(40);
```

```
    lcdWriteComm(0x28);
    usleep(40);
    lcdWriteComm(0x28);
    usleep(40);

    digitalWrite(E, HIGH);
    usleep(40);
    lcdWriteComm(0x28);
    lcdWriteComm(0x0c);
    lcdWriteComm(0x01);
    usleep(5000);

}
void main(void)
{
    char s[] = "hello, world!";

    initPins();
    initLCD();
    int length = strlen(s);
    printf("%d\n", length);

    int i;
    for(i = 0; i < length; i ++)
        lcdWriteData(s[i]);


    sleep(2);
    lcdWriteComm(0x08);
    sleep(2);
    lcdWriteComm(0x0c);
    for(i = 0; i < length; i ++)
        lcdWriteData(s[i]);

}
```

效果图