

JAN-MCUBUS-0B02 用户手册

[illegible]

目 录

1. 特性参数.....	1
2. 操作说明.....	2
2.1 引脚说明.....	2
2.2 接口时序.....	2
2.2.1 TFT 驱动接口时序.....	2
2.2.2 触摸屏驱动接口时序.....	3
2.3 寄存器说明.....	3
2.3.1 CUR_Y (0x00)、CUR_X (0x01)	4
2.3.2 PIXELS (0x02)	4
2.3.3 END_X(0x03).....	4
2.3.4 END_Y(0x04).....	5
2.3.5 PREF(0x05).....	5
2.3.6 RVS_MASK (6)	6
2.3.7 MIRROR (7)	7
2.3.8 STATE	8
3. MCU 操作示例	9
3.1 基本读写操作.....	9
3.1.1 指针方式.....	9
3.1.2 IO 模拟总线.....	10
3.2 高级操作.....	11
3.2.1 设置显示参数.....	11
3.2.2 矩形域填充.....	11
3.2.3 页拷贝操作.....	12
3.2.4 上电复位.....	13
附录 A 选型表.....	15
附录 B 结构尺寸图.....	16
B.1 JAN-MCUBUS-0A01	16
B.2 JAN-MCUBUS-0B0X	16

1. 特性参数

JAN-MCUBUS-0B01 是一款高性能的 16 位真彩 TFT 控制器,控制器上集成了 16Mbytes 的显示缓存,能够提供 8 个显示分页,并且支持各个分页之间的数据拷贝操作。同时还提供了背光管理、自动反色、镜像翻转等高级功能,使用非常灵活、方便。JAN-MCUBUS-0B02 除了具备 JAN-MCUBUS-0B01 的所有功能外还集成了触摸屏驱动电路。JAN-MCUBUS-0B01 和 JAN-MCUBUS-0B02 的各项参数如表 1.1、表 1.2 和表 1.3 所示。

表 1.1 JAN-MCUBUS-0B01&0B02 基本特性

项目	说明
接口类型	Intel8080-8
颜色格式	RGB565
显存页数	8 页
显存容量	16MBytes
可驱动 TFT	AT070TN92

表 1.2 JAN-MCUBUS-0B01&0B02 功能特性

功能	说明
定点写数据	将指定数据写入指定坐标
X 坐标自动累加	每写入一个数据点,当前 X 坐标会自动加一
X 坐标自动返回	当 X 坐标累加到用户预设的 X 结束坐标后,自动返回用户预设的 X 起始坐标
Y 坐标自动累加	X 坐标自动返回时,Y 坐标自动加一
当前显示页切换	屏上显示的数据在 8 页显存中任意切换
当前操作页切换	以 8 页显存中任意页作为目标,写入数据
页拷贝	在任意两页显存之间实现任意区域的数据拷贝操作
自动反色	对任意分页、任意区域进行自动反色操作
背光控制	PWM 背光信号 64 级可调
镜像翻转	对 TFT 上显示的图像进行水平或垂直镜像翻转
状态标识	通过总线接口读取控制器的状态位

表 1.3 JAN-MCUBUS-0B01&0B02 电气特性

项目	说明
电源电压	$5 \pm 0.5V$
功耗 ¹	180mA~750mA
IO 电平 ²	3.3V LVTTTL

注 1: 180mA 对应着背光关闭时的功耗,750mA 对应着背光最亮时的功耗,此数据是在电源电压为 5V 时测出的,实际应用中功耗会由于电源电压的波动而略微变化。

注 2: 通常情况下,如果用 3.3V 的 IO 输出驱动 5V 的 IO 是可以直接驱动的,如果用 5V 的 IO 输出驱动 3.3V 的 IO,推荐您将 5V 的 IO 设置成弱上拉的模式,这样可以避免由于电平不兼容而导致的 IO 电流过大。

2. 操作说明

2.1 引脚说明

JAN-MCUBUS-0B01&0B02 对外有 21 个引脚,关于各个引脚的详细说明请参见表 2.1。

表 2.1 JAN-MCUBUS-0B01&0B02 引脚说明

序号	名称	说明
1	CK ¹	触摸屏驱动时钟输入
2	CS ¹	触摸屏片选信号
3	DI ¹	触摸屏驱动数据输入
4	BY ¹	触摸屏驱动忙标志
5	DO ¹	触摸屏驱动数据输出
6	GND	电源地
7	IR ¹	触摸屏驱动中断输出
8	RE	TFT 驱动读使能信号
9	WE	TFT 驱动写使能信号
10	CE	TFT 驱动片选信号
11	RST	TFT 驱动复位
12	A0	TFT 驱动地址信号
13	D7	TFT 驱动数据总线
14	D6	TFT 驱动数据总线
15	D5	TFT 驱动数据总线
16	D4	TFT 驱动数据总线
17	D3	TFT 驱动数据总线
18	D2	TFT 驱动数据总线
19	D1	TFT 驱动数据总线
20	D0	TFT 驱动数据总线
21	+5V	+5V 电源输入

注 1: 与触摸屏驱动相关的信号, 仅在 JAN-MCUBUS-0B02 有效

2.2 接口时序

2.2.1 TFT 驱动接口时序

JAN-MCUBUS-0B01&0B02 采用 8 位 8080 总线接口与主控 MCU 进行交互, 具体接口时序如图 2.1、图 2.2 所示。

图 2.1 为总线写的时序, 当地址线为 A0 为 0 时表示写入的是地址寄存器, 该寄存器用于对 JAN-MCUBUS-0B01&0B02 中的各个寄存器进行寻址, 取值范围为 0~7。当地址线 A0 为 1 时表示写入的是寄存器值, 关于各个寄存器的作用请参见 2.3。

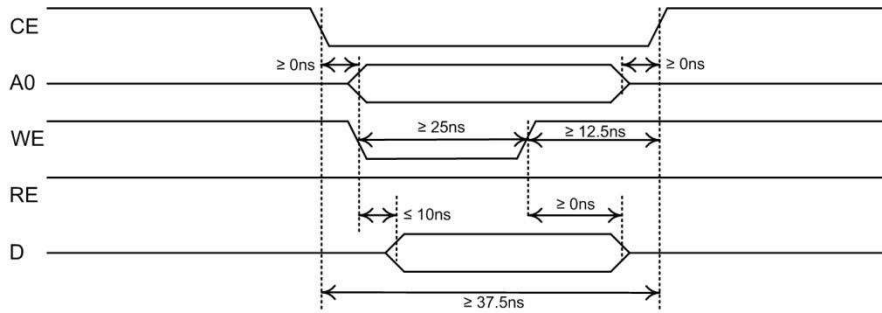


图 2.1 总线写时序

图 2.2 为总线读的时序，在 JAN-MCUBUS-0B01&0B02 中可读的寄存器只有一个，因此为了方便操作，任何读操作都被自动指向了该寄存器，期间 A0 信号以及地址寄存器的状态都会被忽略。

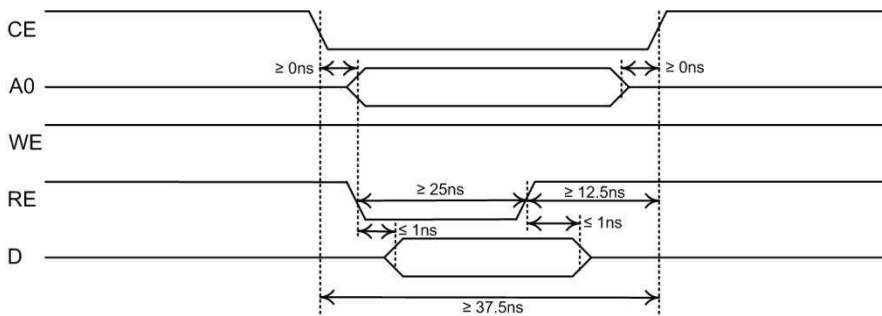


图 2.2 总线读时序

2.2.2 触摸屏驱动接口时序

仅在 JAN-MCUBUS-0B02 中才集成有触摸屏驱动电路，该电路由 ADS7843 实现，具体的接口时序及使用方法请参考 ADS7843 的相关资料。

2.3 寄存器说明

JAN-MCUBUS-0B01&0B02 各个寄存器的地址和功能简介如表 2.2 所示，其中有 7 个 16 位寄存器和 1 个 8 位寄存器，对于 16 位的寄存器需要进行两次写操作才能完成一个寄存器的设置，在进行写操作时必须先写入高八位再写入低八位，而且写操作必须要成对出现，对于 8 位寄存器，只需一次写操作即可完成设置。

表 2.2 寄存器地址和功能简介

操作	位宽	地址	名称	功能简介	复位值
只写	16	0x00	CUR_Y	设置屏幕的 Y 坐标	0x0000
	16	0x01	CUR_X	设置屏幕的 X 坐标	0x0000
	16	0x02	PIXELS	写入像素数据	0x0000
	16	0x03	END_X	设置 X 方向自动返回的坐标，以及页拷贝时 X 方向的结束坐标	0x031f
	16	0x04	END_Y	设置页拷贝时 Y 方向结束坐标	0x01df
	16	0x05	PREF	设置当前显示页、当前操作页，背光等	0x0000
	16	0x06	RVS_MASK	设置反色掩码	0x0000
	8	0x07	MIRROR	镜像翻转	0x01

只读	8	—	STATE	状态寄存器	0x00
----	---	---	-------	-------	------

2.3.1 CUR_Y (0x00)、CUR_X (0x01)

寄存器 CUR_Y 和 CUR_X 用于设置待操作像素点的坐标，TFT 屏幕上坐标的排列如图 2.3 所示，当 CUR_Y 和 CUR_X 的值确定后，像素点 A 的位置便被唯一的确定了，随后的写入的像素数据会被准确的放置在 A 点。

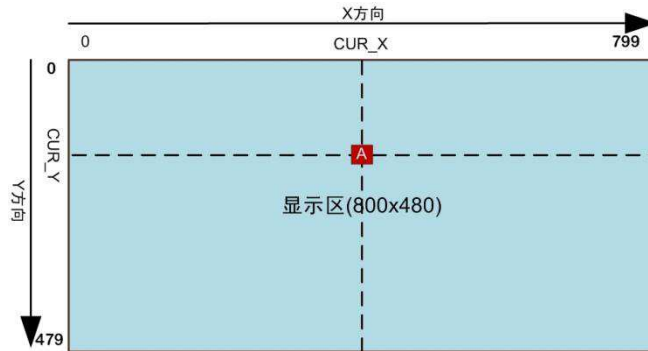


图 2.3 坐标排列

2.3.2 PIXELS (0x02)

寄存器 PIXELS 对应着 16 位的颜色数据，如果当前显示页与当前操作页相同，那么写入 PIXELS 的数据会被立即呈现在由 CUR_X 和 CUR_Y 选中的当前激活点上，如果当前显示页与当前操作页不相同，那么写入 PIXELS 的数据不会被立即呈现出来。JAN-MCUBUS-0B01&0B02 的颜色格式为 RGB565，具体的颜色位对应关系如表 2.3 所示。

表 2.3 颜色位对应关系

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0

2.3.3 END_X(0x03)

为了提高像素数据连续写的效率，当设置好 CUR_X 和 CUR_Y 后，每写入一个像素，当前激活点的 X 坐标就会自动加一，当激活点的 X 坐标等于 END_X 后，便会自动返回 CUR_X 同时 Y 坐标自动加一。如图 2.4 所示，假设 CUR_X、CUR_Y、END_X 分别为 400、200、500，A 点、B 点、C 点、D 点的坐标分别为 (400, 200)、(500, 200)、(400, 201)、(500, 201)。设置好 CUR_X、CUR_Y 后，第一个像素写到了 A 点，第 100 个像素写到 B 点，第 101 个像素写到 C 点，第 200 个像素写到 D 点，依此类推。

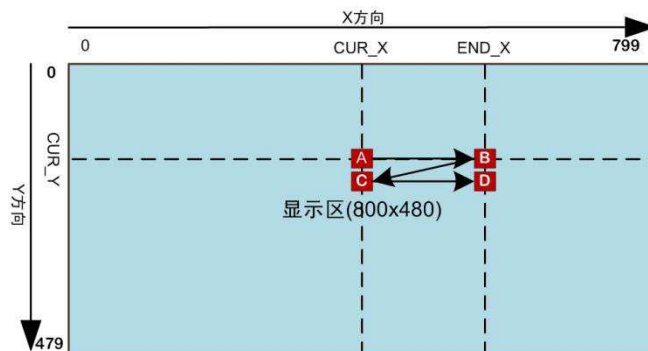


图 2.4 X 自动返回示意

借助 END_X 寄存器,可以简化 MCU 批量数据写的流程,假设 MCU 需要以(100, 200)为起始坐标写入一个 10×20 的矩形,那么只需要将 CUR_X 设为 100, CUR_Y 设为 200, END_X 设为 210,然后进行 200 次的像素点写操作即可,期间不需要再进行坐标设置操作,所有的坐标都会被自动推算。

2.3.4 END_Y(0x04)

END_Y 寄存器需要配合 CUR_X、CUR_Y 和 END_X 使用,在页拷贝和反色操作时,这四个寄存器用于界定操作范围,如图 2.5 所示,A 点坐标为 (CUR_X, CUR_Y), B 点坐标为 (END_X, CUR_Y), C 点坐标为 (CUR_X, END_Y), D 点坐标为 (END_X, END_Y),页拷贝和反色操作的作用范围都是由 A、B、C、D 四点所界定的。

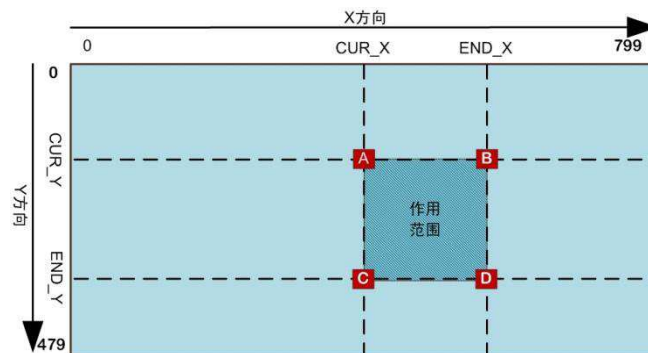


图 2.5 操作范围界定

2.3.5 PREF(0x05)

PREF 寄存器用于设置当前显示页、当前操作页、页拷贝源和 TFT 背光,各个位的具体含义如表 2.4 所示。

表 2.4 PREF 寄存器位定义

位	名称	功能简介	复位值
b5~b0	BK_PWM	背光控制	0
b8~b6	COPY_SRC	页拷贝时的源	0
b11~b9	CUR_PAGE	当前显示的页	0
b14~b12	OPT_PAGE	当前操作的页	0
b15	保留	——	0

1. 背光控制

BK_PWM 用于设置背光信号的占空比,从而调节 TFT 背光的亮度,取值范围为 0~63,0 代表背光关闭,63 代表背光最亮。上电复位后 BK_PWM 的值默认为 0,也就是背光关闭,在 MCU 对 BK_PWM 赋以非零值后,背光亮起。

2. 页拷贝时的源

COPY_SRC 用于设置页拷贝时的数据源,取值范围为 0~7,对应于显存中的 8 个分页,关于页拷贝操作的示意如图 2.6 所示。

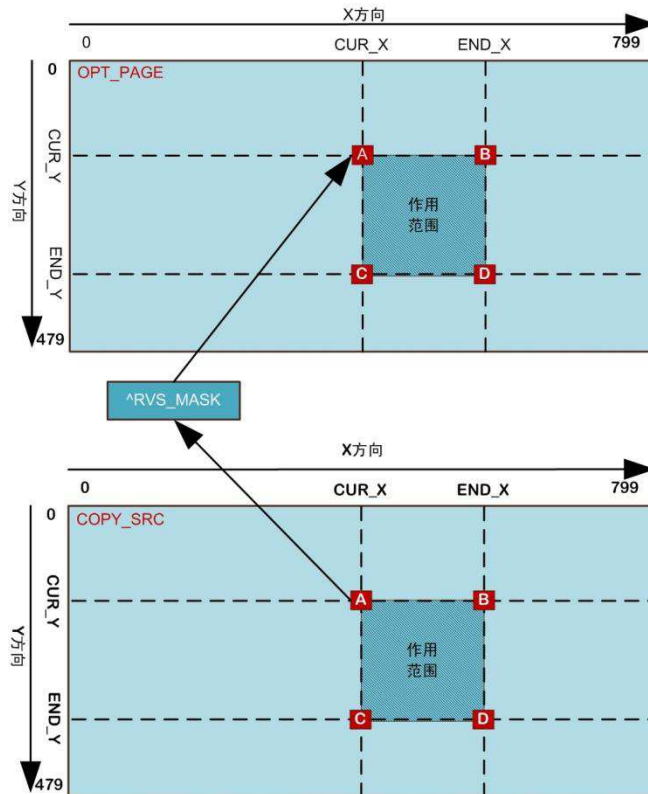


图 2.6 页拷贝操作示意

图 2.6 中示意了两个分页，上面的分页为 OPT_PAGE，表示当前正在操作的页，下面为 COPY_SRC，表示储存着拷贝操作数据源的页，当发起页拷贝操作后，主控逻辑会从 COPY_SRC 所指定的页中将 A、B、C、D 四点所界定的范围中的点逐个读出、与 RVS_MASK 异或，然后写入 OPT_PAGE 中的对应位置。如果 RVS_MASK 的值为 0，那么这一操作只是单纯的数据搬移，如果 RVS_MASK 的值不为 0，那么在数据搬移的过程中像素的颜色值会以 RVS_MASK 为掩码进行反色，如果 OPT_PAGE 和 COPY_SRC 指向了同一页，同时 RVS_MASK 不为 0，那么数据搬移操作便演化成了单纯的反色操作。关于页拷贝操作的进一步说明请参见 2.3.6。

3. 当前显示/操作页

当前显示页由 CUR_PAGE 指定，表示屏幕上实际显示的显存分页，当前操作页由 OPT_PAGE 指定，表示写数据操作、反色操作以及页拷贝操作所对应的显存分页。如果 CUR_PAGE 与 OPT_PAGE 指向同一显存分页，那么写数据、反色等操作的结果会被立即呈现在屏幕上，如果 CUR_PAGE 与 OPT_PAGE 指向不同的显存分页，那么对 OPT_PAGE 的任何操作都不会影响屏幕上的显示内容，只有在 CUR_PAGE 切换到 OPT_PAGE 后，OPT_PAGE 中数据才会被显示出来。

2.3.6 RVS_MASK (6)

RVS_MASK 用于设置 16 位的反色掩码，反色掩码的作用是标识在反色操作时需要进行反转的颜色位，RVS_MASK 的位定义如表 2.5 所示。

表 2.5 RVS_MASK 位定义

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
R4	R3	R2	R1	R0	G5	G4	G3	G2	G1	G0	B4	B3	B2	B1	B0

反色操作由 MIRROR 寄存器中的 RVS 位启动(详见 2.3.7)。如果是需要进行反色操作，那么首先要让 OPT_PGAE 和 COPY_SRC 同时指向需要反色的页，然后设置 CUR_X、CUR_Y、END_X 和 END_Y，界定需要反色的区域并向 RVS_MASK 写入反色掩码，然后向 MIRROR 寄存器中的 RVS 位写入 1 即可。反色的掩码的具体值可以随意指定，例如 0xf800 可以用于对所有的红色位反色，0x07e0 可以用于对所有绿色位反色，0x001f 可以用于对所有蓝色位反色，依此类推。

如果是需要进行页拷贝操作，那么要让 OPT_PAGE 和 COPY_SRC 指向不同的页，然后设置 CUR_X、CUR_Y、END_X 和 END_Y 界定需要拷贝的区域并向 RVS_MASK 写入 0x0000（也可以是非零值），然后向 MIRROR 寄存器的 RVS 位写入 1 即可。拷贝操作完成后，COPY_SRC 页中对应区域的数据将被拷贝到 OPT_PAGE 的对应区域中。需要注意的是，也可以向 RVS_MASK 写入非零值，不同之处是，拷贝到 OPT_PAGE 中的数据并非是 COPY_SRC 页中对应区域的原始数据，而是以 RVS_MASK 为掩码经过反色的数据。

2.3.7 MIRROR (7)

MIRROR 寄存器用于实现图像的水平和垂直镜像翻转，以及控制页拷贝和反色操作的启动，该寄存器各位的具体含义如表 2.6 所示。

表 2.6 MIRROR 寄存器位定义

位	名称	功能简介	复位值
b7~b3	保留	——	0
b2	RVS	反色以及页拷贝操作启动位	0
b1	UD	控制垂直镜像翻转	0
b0	LR	控制水平镜像翻转	1

RVS 位用于启动页拷贝和反色操作，在向 RVS 位写入 1 之前要向先设置好 CUR_X、CUR_Y、PREF 等寄存器设置好待操作的页、待操作的区域以及反色掩码等参数。在启动反色或页拷贝操作后，RVS 位会自动清零。

UD 位用于控制显示画面的垂直翻转，LR 位用于控制显示画面的水平翻转，操作 UD 位和 LR 位会影响 TFT 上的像素点位置与显存中数据地址的映射关系，但不会改变显存中的数据，不同的 UD 和 LR 值所对应的显示效果如图 2.7 所示。

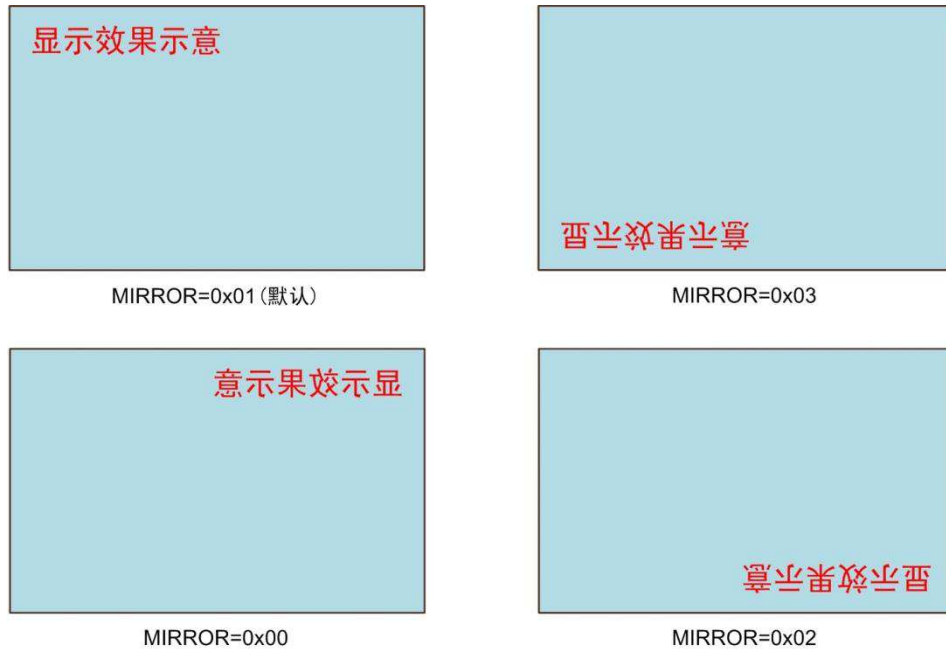


图 2.7 显示效果

2.3.8 STATE

STATE 是 JAN-MCUBUS-0B01&0B02 中唯一的可读的寄存器，因此对总线的所有读操作均默认为读 STATE 寄存器，读期间的 A0 信号和地址寄存器均会被忽略。STATE 寄存器的宽度为 8bit，通过读取 STATE 寄存器，可以获知控制器当前的状态，如果从 STATE 寄存器中读回的值为零，表示控制器处于空闲状态，可以接收并处理新的操作，如果从 STATE 寄存器中读回的值为非零，表示控制器正进行页拷贝或反色操作，此时控制器不可以接收 MCU 的任何写操作，否则会导致页拷贝或反色操作出错。

3. MCU 操作示例

3.1 基本读写操作

对于兼容 8080 总线的 MCU，可将 JAN-MCUBUS-0B01&0B02 映射成一个存储器件，以指针方式进行读写访问，对于不兼容 8080 总线或不具备外部总线接口的 MCU，可以用 IO 模拟总线的方式进行读写操作，下面以 8051 单片机为例分别说明，其端口连接如图 3.1 所示。

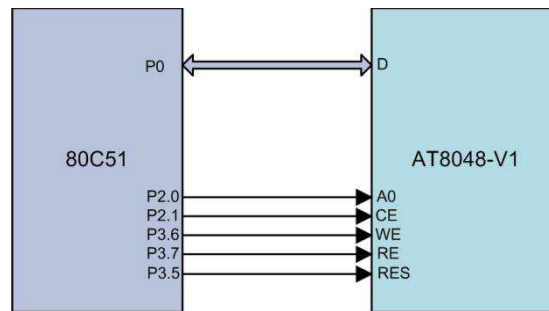


图 3.1 端口连接示意图

3.1.1 指针方式

对于图 3.1 所示的端口连接方式，用指针实现基本读写操作的代码示例如程序清单 3.1 所示。

程序清单 3.1 指针实现基本读写操作

```
#include "AT89X52.h"
#define RES    P3_5
unsigned char  xdata *pTFT_RegAddr = (unsigned char *)0x0000;
unsigned char  xdata *pTFT_RegData = (unsigned char *)0x0100;
static unsigned char udIr = 0x01;
// 写寄存器地址
void TFT_WRegAddr(unsigned char a)
{
    *pTFT_RegAddr = a;
}
// 写寄存器内容
void TFT_WRegData(unsigned char d)
{
    *pTFT_RegData = d;
}
// 读寄存器内容
unsigned char TFT_RData()
{
    unsigned char temp;
    temp = *pTFT_RegData;
    return temp;
}
```

```
}
```

3.1.2 IO 模拟总线

对于图 3.1 所示的端口连接方式，用 IO 模拟总线实现基本读写操作的代码如程序清单 3.2 所示。

程序清单 3.2 IO 模拟总线

```
#include "AT89X52.h"
#define CE    P2_1
#define A0    P2_0
#define D      P0
#define WE    P3_6
#define RD    P3_7
#define RES    P3_5
// 写寄存器地址
void TFT_WRegAddr(unsigned char a)
{
    CE = 0;
    A0 = 0;
    D = a;
    WE = 0;
    WE = 1;
    CE = 1;
}
// 写寄存器内容
void TFT_WRegData(unsigned char d)
{
    CE = 0;
    A0 = 1;
    D = d;
    WE = 0;
    WE = 1;
    CE = 1;
}
// 读寄存器内容
unsigned char TFT_RData()
{
    unsigned char temp;
    D = 0xff;
    CE = 0;
    RD = 0;
    temp = D;
    RD = 1;
    CE = 1;
    return temp;
}
```

```
}
```

3.2 高级操作

3.2.1 设置显示参数

JAN-MCUBUS-0B01&0B02 可以方便的对显示缓存和背光进行管理，具体示例如程序清单 3.3 所示。

程序清单 3.3 设置显示参数

```

/*****
* 函数名      : TFT_SetPref
* 描述       : 设置当前显示页、当前操作页、页拷贝源以及背光
* 参数       :
                --cur_page : 当前显示页
                --opt_page : 当前操作页
                --copy_src  : 页拷贝的源
                --bk_pwm   : 背光
*返回值      : 无
*****/

void TFT_SetPref( unsigned char cur_page,unsigned char opt_page,
                  unsigned char copy_src,unsigned char bk_pwm    )
{
    int temp;
    temp = bk_pwm | (copy_src<<6) | (opt_page <<12) | (cur_page<<9);
    TFT_WRegAddr(5);          // 地址寄存器指向 PREF
    TFT_WRegData(temp>>8);    // 数据写入 PREF
    TFT_WRegData(temp);
}

```

3.2.2 矩形域填充

在进行清屏、图片显示等操作时，会用到矩形域填充操作，JAN-MCUBUS-0B01&0B02 对矩形域填充操作进行了优化，在填充时 MCU 只需要设置好起点坐标和终点坐标即可，填充过程中所有点的坐标都会被自动推算，最大限度的保证矩形域填充的效率，矩形域填充的示例如程序清单 3.4 所示。

程序清单 3.4 矩形域填充

```

/*****
* 函数名      : TFT_RectFill
* 描述       : 用指定颜色填充 TFT 上的指定矩形域
* 参数       :
                --start_x  : 矩形域的起始 X 坐标
                --start_y  : 矩形域的起始 Y 坐标
                --end_x    : 矩形域的结束 X 坐标
                --end_y    : 矩形域的结束 Y 坐标
                ---color   : 待填充的颜色

```

```

*返回值      : 无
*****/

void TFT_RectFill(int start_x,int start_y,int end_x,int end_y,int color)
{
    int i,j,w,h;
    TFT_WRegAddr(0);          // 地址寄存器指向 CUR_Y
    TFT_WRegData(start_y>>8); // 设置 Y 起始坐标
    TFT_WRegData(start_y);
    TFT_WRegAddr(1);          // 地址寄存器指向 CUR_X
    TFT_WRegData(start_x>>8); // 设置 X 起始坐标
    TFT_WRegData(start_x);
    TFT_WRegAddr(3);          // 地址寄存器向 END_X
    TFT_WRegData(end_x>>8);   // 设置 END_X
    TFT_WRegData(end_x);
    TFT_WRegAddr(2);          // 地址寄存器指向 PIXELS
    h=end_y-start_y+1;        // 计算矩形域高度
    w=end_x-start_x+1;        // 计算矩形域宽度
    for(i=0;i<h;i++)
    {
        for(j=0;j<w;j++)
        {
            // 循环填充数据
            TFT_WRegData(color>>8);
            TFT_WRegData(color);
        }
    }
}

```

3.2.3 页拷贝操作

JAN-MCUBUS-0B01&0B02 提供了 8 页的显示缓存，可以在任意页之间对指定区域进行数据拷贝，数据拷贝操作由硬件完成，拷贝过程中不需要 MCU 的介入。对于低速 MCU，当刷新大面积区域时，会出现拉幕现象，灵活的运用页拷贝操作可以有效的避免这种现象，从而使画面显示更加流畅，页拷贝操作的示例如程序清单 3.5 所示。

程序清单 3.5 页拷贝操作

```

/*****
* 函数名      : TFT_PageCopy
* 描述       : 在页之间进行数据拷贝
* 参数       :
                --start_x      : 待拷贝区域的起始 X 坐标
                --start_y      : 待拷贝区域的起始 Y 坐标
                --end_x        : 待拷贝区域的结束 X 坐标
                --end_y        : 待拷贝区域的结束 Y 坐标
                ---rvs_mask    : 反色掩码
*返回值      : 无
*****/

```

```
void TFT_PageCopy(int start_x,int start_y,int end_x,int end_y,int rvs_mask)
{
    unsigned char temp;
    TFT_WRegAddr(0);          // 地址寄存器指向 CUR_Y
    TFT_WRegData(start_y>>8); // 设置 Y 起始坐标
    TFT_WRegData(start_y);
    TFT_WRegAddr(1);          // 地址寄存器指向 CUR_X
    TFT_WRegData(start_x>>8); // 设置 X 起始坐标
    TFT_WRegData(start_x);
    TFT_WRegAddr(3);          // 地址寄存器指向 END_X
    TFT_WRegData(end_x>>8);   // 设置 X 结束坐标
    TFT_WRegData(end_x);
    TFT_WRegAddr(4);          // 地址寄存器指向 END_Y
    TFT_WRegData(end_y>>8);   // 设置 Y 结束坐标
    TFT_WRegData(end_y);
    TFT_WRegAddr(6);          // 地址寄存器指向 RVS_MASK
    TFT_WRegData(rvs_mask>>8); // 写该寄存器启动页拷贝操作
    TFT_WRegData(rvs_mask);
    TFT_WRegAddr(7);          // 地址寄存器指向 MIRROR
    TFT_WRegData(0x04|udlr)   // 启动页拷贝操作
    while(1)                  // 等待页拷贝结束
    {
        temp = TFT_RData();
        if(temp == 0)
            break;
    }
}
```

需要注意的是，在进行页拷贝操作前先要调用 `TFT_SetPref` 函数，设置好当前操作页以及拷贝源。如果在拷贝时 `RVS_MASK` 的值不为 0，那么拷贝过去的数据是经过反色的。当前操作页和拷贝源也可以指向同一页，此时通过设置反色掩码可以实现单纯的反色操作。

3.2.4 上电复位

JAN-MCUBUS-0B01&0B02 的上电复位操作非常简单，上电复位的示例如程序清单 3.6 所示，首先 MCU 将 JAN-MCUBUS-0B01&0B02 的 RES 引脚拉低 1ms 以上，然后 MCU 再等待 1ms 即可开始对 JAN-MCUBUS-0B01&0B02 发起其它写操作。

程序清单 3.6 上电复位操作

```
/******
* 函数名      : TFT_Init
* 描述        : 上电初始化
* 参数        : 无
* 返回值      : 无
*****/

void TFT_Init()
```

```
{  
    unsigned int i;  
    RES = 0;  
    for(i=0;i<10000;i++);    // 延时至少 1ms  
    RES = 1;  
    for(i=0;i<10000;i++);    // 延时至少 1ms  
    TFT_SetPref(0,0,0,63);    // 打开背光  
}
```


附录A 选型表

型号	触摸驱动	可驱动 TFT	尺寸	分辨率	显存分页
JAN-MCUBUS-0A01	否	AT070TN83 V.1	7 inch	800x480	8 页
JAN-MCUBUS-0B01	否	AT070TN92	7 inch	800x480	8 页
JAN-MCUBUS-0B02	是	AT070TN92	7 inch	800x480	8 页
JAN-MCUBUS-0B03	否	AT080TN52 V.1	8 inch	800x600	4 页
JAN-MCUBUS-0B04	是	AT080TN52 V.1	8 inch	800x600	4 页

