

《在 STM32 上驱动使用高通字库芯片》

— 产品规格书 —



如何在 STM32 上驱动使用高通字库芯片

目录

1. 高通字库芯片是什么	3
2. 如何驱动高通点阵字库芯片	3
2.1 硬件连接	3
2.2 软件驱动	4
2.3 如何使用高通字库接口	6
2.4 如何验证数据正确性	7
3. 数据异常自检排查	8
3.1 排查步骤	8
3.2 初始化和驱动函数是否正常的验证方法:	8

如何在 32 位 MCU 上使用高通点阵字库:

https://www.bilibili.com/video/BV1aG41117uH/?spm_id_from=333.999.0.0

高通字库使用教程（1）硬件连接与注意事项:

https://www.douyin.com/user/MS4wLjABAAAALZKQhGxpXFKZjq0mrp9tsvmvTn97WmQVB2XqYz-YXpz6o.jpKjy63wjJictca4gh3?modal_id=6852599188291620104

高通字库使用教程（2）SPI 底层函数使用:

https://www.douyin.com/user/MS4wLjABAAAALZKQhGxpXFKZjq0mrp9tsvmvTn97WmQVB2XqYz-YXpz6o.jpKjy63wjJictca4gh3?modal_id=6852616522930621710

高通字库使用教程（3）SPI 底层函数验证:

https://www.douyin.com/user/MS4wLjABAAAALZKQhGxpXFKZjq0mrp9tsvmvTn97WmQVB2XqYz-YXpz6o.jpKjy63wjJictca4gh3?modal_id=6852618057332837646

高通字库使用教程（4）关于库函数的讲解:

https://www.douyin.com/user/MS4wLjABAAAALZKQhGxpXFKZjq0mrp9tsvmvTn97WmQVB2XqYz-YXpz6o.jpKjy63wjJictca4gh3?modal_id=6854008040983694599

在 STM32 上如何驱动使用高通点阵字库芯片

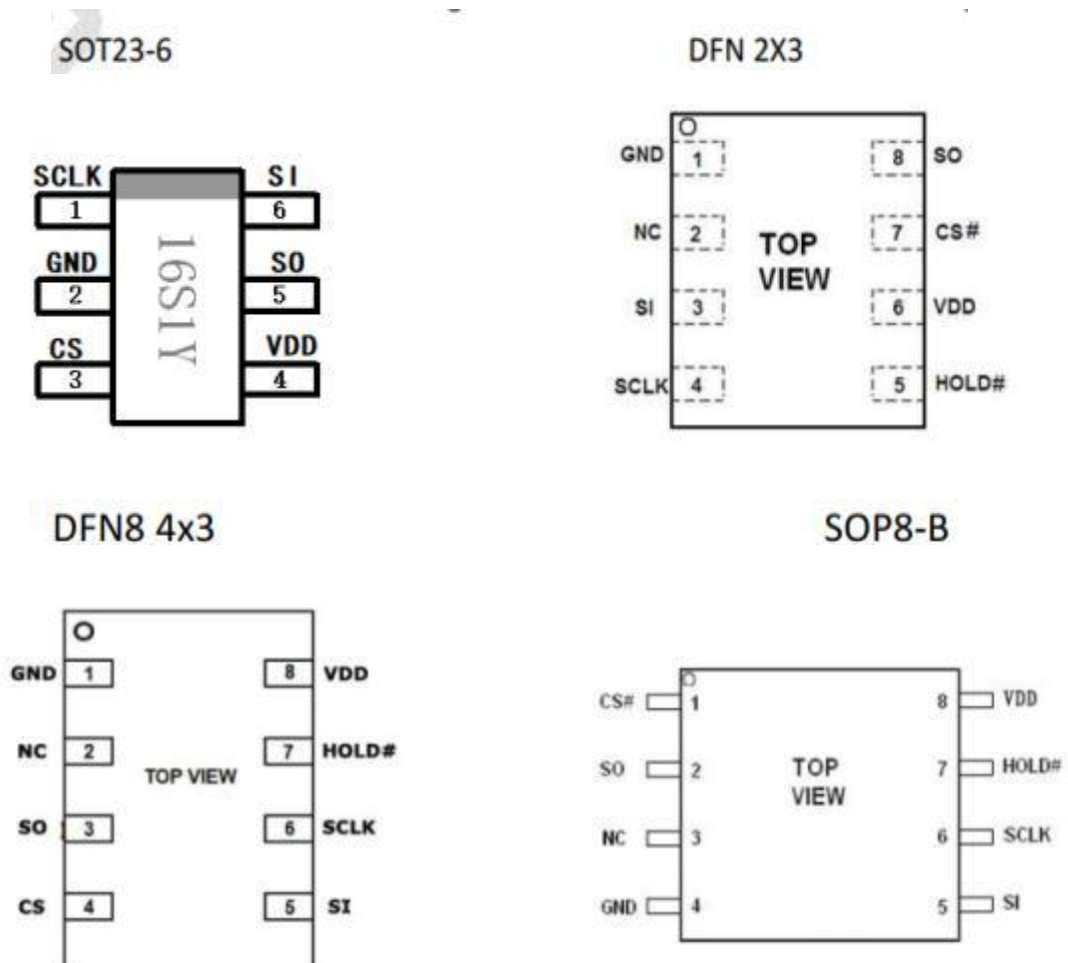
1. 高通字库芯片是什么

高通字库芯片是深圳高通半导体有限公司生产的搭载标准专业字库的 ic 元件。按照搭载的字库是否支持矢量字库，高通字库芯片可划分为点阵字库芯片与矢量字库芯片。本次我们介绍点阵字库芯片的驱动方式。

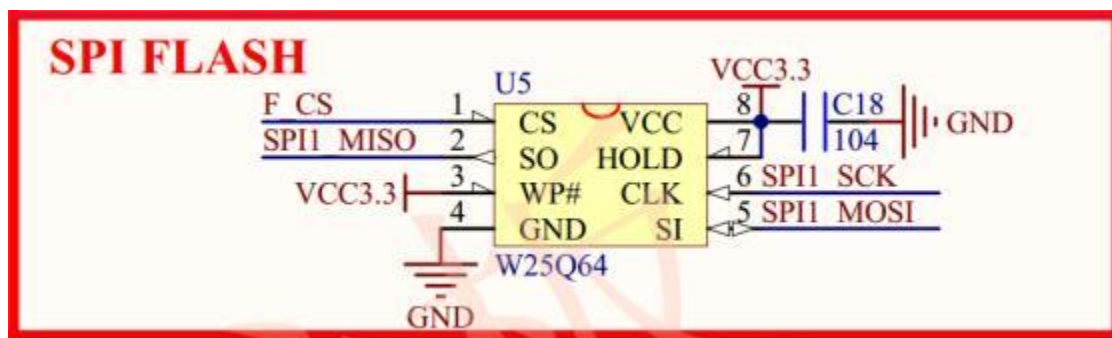
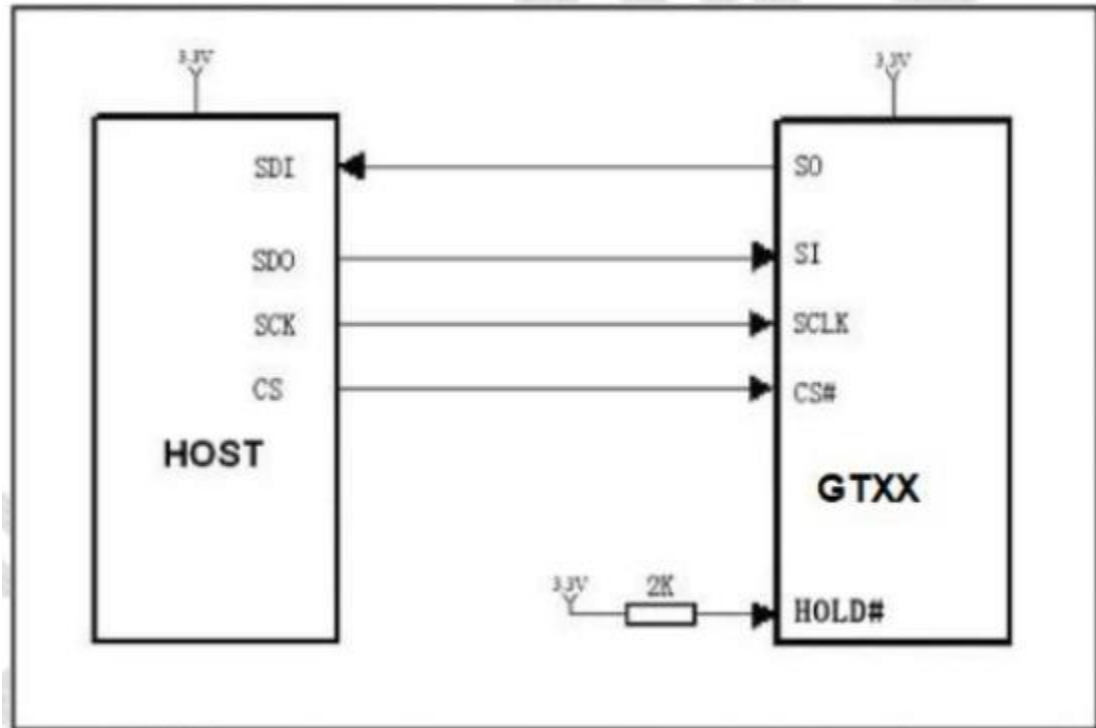
2. 如何驱动高通点阵字库芯片

2.1 硬件连接

高通字库芯片通常使用 SPI 通讯接口，所以需要与主控 mcu 的硬件 spi 引脚连接建立硬件 SPI 联系，或者使用 io 口模拟软件 spi 方式驱动。需要注意不同型号的字库芯片采用的封装不同导致芯片引脚的排布有差异，在连接时参考字库芯片规格书中的引脚排布图确保引脚正确相连。常用的封装为 SOT23-6, DFN 2X3, DFN8 4X3 以及 SOP8，其对应引脚排布如下图：



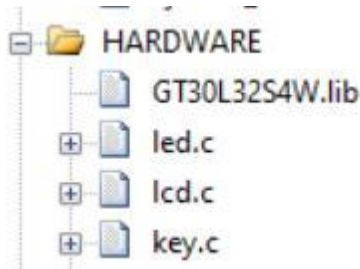
我们以正点原子的探索者开发板为例演示驱动过程，该开发板 MCU 采用 STM32F407ZGT6 芯片。该芯片提供 3 个 spi 通讯接口，开发板本身有一个 SOP8 封装的 W25Q64 的 flash 芯片与 SPI1 连接，为简化开发进程，我们选择与 flash 共用 SPI1 接口（即 SCK-PB3, MISO-PB4 以及 MOSI-PB5 引脚共用，通过不同的 CS 片选脚区分 flash-PB14 与字库芯片 -PG7），这样我们可以直接使用正点原子提供的 SPI 接口示例代码。以型号为 GT30L32S4W 的字库芯片为例，该芯片与 flash 一样为 SOP-8 封装，参照字库芯片规格书和 STM32 开发板 flash 中的示意图连接好芯片与 mcu 电路。



2.2 软件驱动

在硬件连接完成后进行软件驱动代码的实现，目前高通字库芯片统一使用静态库提供调用接口函数方式调用文字，您在购买字库芯片后，请您联系高通公司的技术支持人员索要符合您编译环境的静态库文件与参考资料。通常提供的资料中包含.lib 或.a 后缀的库文件，.h 后缀的接口函数声明头文件以及.h 的说明文件等参考资料。库文件是.lib 或.a 后缀与你使用的编译环境有关，例如我们使用 keil 软件开发，使用 armcc 编译，故库文件为.lib 后缀，若使用 gcc 编译则库文件通常为.a 后缀。

在使用上.lib 或.a 库文件与.c 文件是等效的，分别将库文件与.h 头文件添加到 keil 工程 项目中。



随后实现.h 头文件中带有 extern 关键字声明的外部驱动函数(外部函数声明有几个就需 要实现几个)，即可调用.h 中声明的接口函数获取字形数据。

GT30L32S4W 型号芯片中声明了两个驱动函数

```
/* 外部函数声明 */
extern unsigned long r_dat_bat(unsigned long address,unsigned long
DataLen,unsigned char *pBuff);
extern unsigned char gt_read_data(unsigned char* sendbuf , unsigned
char sendlen , unsigned char* receivebuf, unsigned int receivelen);
```

GT30L32S4W 型号芯片两个驱动函数的实现，实现方式不唯一，关键在于实现函数功能即可， r_dat_bat 函数是从 address 地址开始读取 DataLen 个字节的数据， 存储到 pBuff 中， 而 gt_read_data 函数则是先发送 sendlen 个字节的 sendbuf 数据，再接收 receivelen 个字节 的数据，存储到 receivebuf 中。实现示例如下(其中的 SPI1_ReadWriteByte 函数为正点原子 示例代码实现的 SPI 读取发送函数)，

```
static void SPI_Address(unsigned char AddH,unsigned char AddM,unsigned
char AddL) {
    SPI1_ReadWriteByte(AddH);
    SPI1_ReadWriteByte(AddM);
    SPI1_ReadWriteByte(AddL);
}
unsigned long r_dat_bat(unsigned long address,unsigned long
DataLen,unsigned char *pBuff){
    unsigned long i;
    unsigned char addrHigh;
    unsigned char addrMid;
    unsigned char addrLow;

    addrHigh=address>>16;
    addrMid=address>>8;
    addrLow=(unsigned char)address;
    Rom_csl;          //片选选中字库芯片

    SPI1_ReadWriteByte(0x03);    //普通读取首先发送 0X03,然后发送地址高八位
```

```

        pBuff[i]=SPI1_ReadWriteByte(0x00); //开始读取数据
    }
    Rom_csh;
}
unsigned char gt_read_data(unsigned char* sendbuf , unsigned char
sendlen , unsigned char* receivebuf, unsigned int
receivelen){ unsigned char i;
    Rom_csl;          //片选选中字库芯片

    for(i=0;i<sendlen;i++){
        SPI1_ReadWriteByte(sendbuf[i]);
    }
    for(i=0;i<receivelen;i++){
        receivebuf[i]=SPI1_ReadWriteByte(0x00);
    }
}

```

实现上述底层驱动函数后，如果.h 头文件中声明有 GT_Font_Init 函数则需要先调用该函数进行字库初始化，没有声明则不需要处理，直接调用接口函数获取字形数据。

```

/* ----- */
//字库初始化(在调用字形接口前，请务必在初始化 SPI 后调用以进行字库初始化)
int GT_Font_Init(void);
/* ----- */

```

2.3 如何使用高通字库接口

在完成上述步骤驱动字库芯片后，按照你所需要的文字编码，传入接口函数中调取，例如调用字母 A, 其编码为 ASCII 码 0x41, 各接口函数调用说明可以参考资料中提供的.h 头文件说明文件或者咨询高通公司的技术支持，之后将读取出来的字形数据进行显示即可。

```

GT_SPI_init();          //字库芯片 SPI 初始化
GT_Font_Init();         //初始化字库芯片

ASCII_GetData(0x41,ASCII_8X16,DZ_Data);//调用 8*16 大小的 ASCII 编码为
0x41 即 A 的字形数据 存放于 DZ_Data 数组中

```

其中的显示函数参考高通公司技术支持提供的参考文件实现如下，其中 LCD_Fast_DrawPoint 为将某(x,y)点设置为相应的颜色，替换为适合本身的描点函数即可。

```

//横置横排显示
void Display_W(unsigned char *pBits,unsigned int x,unsigned int
y,unsigned int widt,unsigned int high)
{
    unsigned int i,j,k,n = 0;
    unsigned char temp;
}

```

```
{
    for( j = 0;j < ((width+7)>> 3);j++)
    {
        temp = pBits[n++];
        for(k = 0;k < 8;k++)
        {
            if(((temp << k)& 0x80) == 0 )
            {
                /* 显示一个像素点 */
                LCD_Fast_DrawPoint(    x+k+(j*8),    y+i,
                WHITE); }else{
                /* 显示一个像素点 */
                LCD_Fast_DrawPoint(    x+k+(j*8),    y+i,
                BLACK); }
        }
    }
}
```

2.4 如何验证数据正确性

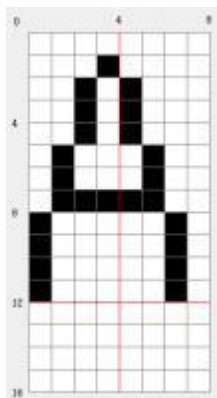
您可以从字库芯片的规格书或者咨询高通公司的技术支持获取验证数据，将读取到的数据与正确数据进行比较验证以确定驱动程序的正确性，上述 GT30L32S4W 型号字库芯片的 8*16 点阵大小的 W 排置（横置横排）的 ASCII 字母“A”的正确数据如下：（16 进制）

排置：W（横置横排）点阵大小 8X16

字母"A"

点阵数据：00 10 28 28 28 44 44 7C 82 82 82 82 00 00 00 00

字符字形效果如下图所示



3 数据异常自检排查

3.1 排查步骤

若读取数据验证异常，请按以下步骤进行自检排查：

- 1, 查看接口调用说明文件， 确保调用接口传入的参数正确。
- 2, 确认是否在初始化 SPI 后调用 GT_Font_Init 函数进行字库初始化， 若确实调用， 查看 GT_Font_Init 函数返回值是否为非 0， 若为 0 则表示字库初始化失败。
- 3, 联系高通公司技术支持人员提供验证数据进行对比验证外部声明函数功能是否正常， 如上述 GT30L32S4W 型号的 r_dat_bat 和 gt_read_data 函数。
- 4, 确认芯片引脚是否与 MCU 正确相连， 芯片是否存在虚焊现象。
- 5, 有任何其他问题， 可咨询高通技术人员提供技术支持。

3.2 初始化和驱动函数是否正常的验证方法：

1: GT_Font_Init 返回为 0 及 gt_read_data 函数的解决验证办法：

```
uint8_t tmp_buf[8] = {0};

tmp_buf[0] = 0x9F;

gt_read_data(tmp_buf, 1, tmp_buf, 8); //验证函数

for(int i=0;i<8;i++)
{

    printf("tmp_buf:%x\r\n",tmp_buf[i]); //打印返回的值

}
```

打印出来的 tmp_buf 里面如果全是 00 或者 FF 则是错误的。里面应该会比较有规律的数字。看 tmp_buf 里面的数据。如果 tmp_buf 里面的数据不是 0 或 0xff， 而是有规律的一组数字，前三位的最后一位在 12-19 范围内或者是 C9， 则是大概率是正确的。

2: r_dat_bat() 函数验证方法：

要验证 r_dat_bat() 函数，可以根据不同的芯片型号读取 0xc0/0xb0/0xa0/0x2c0 地址的 16 位数据，并参考以下提供的数据进行对比。如果没有需要的数据，请向 FAE 索要对应型号字库芯片该地址的数据，然后对比读出来的数据是否正确。

以下是各个字库型号的 0xc0/0xb0/0xa0/0x2c0 地址 16 字节数据以及对应型号的地址和地址数据的参考：

GT5DL14P1Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00

GT5DL14S2Y: c0h: FC 02 02 FC 00 00 01 02 02 01 00 00 00 04 FE 00

GT5DL16M2Y: c0h: 21 72 21 73 21 74 21 75 21 76 21 77 21 78 21 79

GT5DL16S1W: c0h: 00 00 00 00 00 00 0c 60 0c 60 00 00 00 00 00 00

GT5DL24A1Y80: c0h: 00 00 E0 30 18 18 FE 18 18 10 E0 00 00 00 00 00

GT5DL24A2W: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00

GT5DL28K2W: b0h: CE 00 7B 00 00 00 00 03 00 00 C0 00 C0 00 C0 00

GT5DL32A3W: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00

GT5HL24A2W: 2c0h: A8 AB 00 00 A8 AF 00 00 A8 B3 00 00 A8 B5 00 00

GT5HL32S3W: 2c0h: A8 AB 00 00 A8 AF 00 00 A8 B3 00 00 A8 B5 00 00

GT5HL20K2W: c0h: A8 A4 A8 A2 00 00 00 00 00 00 00 00 00 00 00 00

GT5HL16S2W: 2c0h: A8 B6 00 00 A8 B7 00 00 A8 B8 00 00 00 00 00 00

GT5SDL24A40: c0h: 00 05 00 00 06 00 11 05 00 00 06 00 22 05 00 00

GT5SL24K3W40: c0h: 68 B0 01 02 78 B0 01 02 88 B0 01 03 A0 B0 01 02

GT5SL24K4W: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00

GT5SLAD2E1A: c0h: 63 15 00 00 06 00 81 15 00 00 06 00 9F 15 00 00

GT5SLAD3BFA: c0h: 63 15 00 00 06 00 81 15 00 00 06 00 9E 15 00 00

GT5SLCD2E1A: c0h: F7 04 00 00 06 00 08 05 00 00 06 00 19 05 00 00

GT5SLCD2S2A: c0h: 63 15 00 00 06 00 81 15 00 00 06 00 9F 15 00 00

GT5SLCD2S4A: c0h: 68 B0 01 02 78 B0 01 02 88 B0 01 03 A0 B0 01 02

GT5SUAD2E: c0h: 7C 49 01 02 8C 49 01 02 9C 49 01 03 B4 49 01 02

GT5SUAD3BFA: c0h: 63 15 00 00 06 00 81 15 00 00 06 00 9F 15 00 00

GT5SUCD2E1A: c0h: E7 04 00 00 06 00 08 05 00 00 06 00 19 05 00 00

XT5YL14U1Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00

GT5YL24A1Y520: c0h: 00 00 E0 30 18 18 FE 18 18 10 E0 00 00 00 00 00

GT24L24A2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00

GT20L16J1Y: b0h: 00 00 00 00 00 00 00 03 03 00 00 00 00 00 00 00

GT20L16P1Y: c0h: 00 00 00 00 00 00 00 00 00 00 00 16 0E 00 00 00

GT20L16S1Y: c0h: 00 00 00 00 18 18 00 00 00 18 18 00 00 00 00 00

GT20L24F6Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00

GT21L16S2W: c0h: 00 00 00 00 00 00 0C 60 0C 60 00 00 00 00 00 00

GT21L16S2Y: c0h: 00 00 00 00 18 18 00 00 00 18 18 00 00 00 00 00

GT21L16T1W: c0h: 00 00 00 00 00 00 0C 60 0C 60 00 00 00 00 00 00

GT21L24S1W: c0h: 10 40 00 10 40 00 10 40 00 08 80 00 07 00 00 00

GT22L16A1Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00

GT22L16A2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT22L16K1Y40: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT22L16M1Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT22L16U1Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT22L16V2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT22L24S3W: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT22U16A2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT23L16U2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT24L16A2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT24L16K1Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT24L16M1Y20: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT24L24A2W: c0h: 00 00 00 E0 10 08 08 08 10 E0 00 00 00 00 00 00
GT24L24A2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT24L32M4W80: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT24U24A2Y: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT30L16M2Y: c0h: 00 40 30 1C 17 12 90 70 38 10 00 00 00 00 00 01
GT30L16U2W: c0h: 18 00 04 00 0A 00 31 80 C0 60 00 00 7E C0 01 80
GT30L24A2W: c0h: 0C 1C 10 10 10 10 10 1F 0F 00 00 00 00 0A 80 C0
GT30L24A3W: c0h: 18 C0 00 18 C0 00 18 C0 00 0F 80 00 07 00 00 00
GT30L24M1W: c0h: 10 40 00 10 40 00 10 40 00 08 80 00 07 00 00 00
GT30L24M1Z: a0h: 40 10 00 40 10 00 40 10 00 80 08 00 00 07 00 00
GT30L24T3Y: c0h: 00 40 30 1C 17 12 90 70 38 10 00 00 00 00 00 01
GT30L32A1W80: d0h: 0c 00 00 00 03 00 00 00 01 C0 00 00 00 E0 00 00
GT30L32M4W80: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT30L32S4W: c0h: 08 00 0C 00 18 00 10 80 3E C0 21 80 41 00 02 00
GT30L32S4Y: c0h: 00 40 30 1C 17 12 90 70 38 10 00 00 00 00 00 01
GT31L16M1Y80: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT31L16S2W80: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00
GT31L24M1W16: c0h: 10 40 00 10 40 00 10 40 00 08 80 00 07 00 00 00
GT31L24M3W40: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT32L24A180: c0h: 10 40 00 10 40 00 10 40 00 08 80 00 07 00 00 00
GT32L24F0210: c0h: 36 49 49 49 36 00 00 00 06 49 49 29 1E 00 00 00

GT32L24M1Y80: c0h: 00 00 07 0F 18 30 30 30 18 0F 07 00 00 00 00 00
GT32L24M0140: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT32L32M4W40: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT32L32M0180: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT32L32S0140: c0h: 70 88 88 70 88 88 70 00 70 88 88 78 08 10 60 00
GT60L16M2K4: c0h: 00 00 00 00 00 00 00 00 00 00 18 18 18 30 00 00 00
GT60M2: c0h: 68 B0 01 02 78 B0 01 02 88 B0 01 03 A0 B0 01 02
GT61L24M3K4: c0h: 00 00 00 00 00 00 00 00 00 00 18 18 18 30 00 00 00
XT21L12S1Y12: c0h: c0 02 C0 01 00 00 00 00 00 00 20 00 20 00 20 00
XT21L12S1Y40: c0h: c0 02 C0 01 00 00 00 00 00 00 20 00 20 00 20 00
XT21L20S2W60: 2c0h: 00 00 00 00 00 00 00 00 00 00 00 00 30 30 10 20 00
XT30L24M1W: c0h: 10 40 00 10 40 00 10 40 00 08 80 00 07 00 00 00