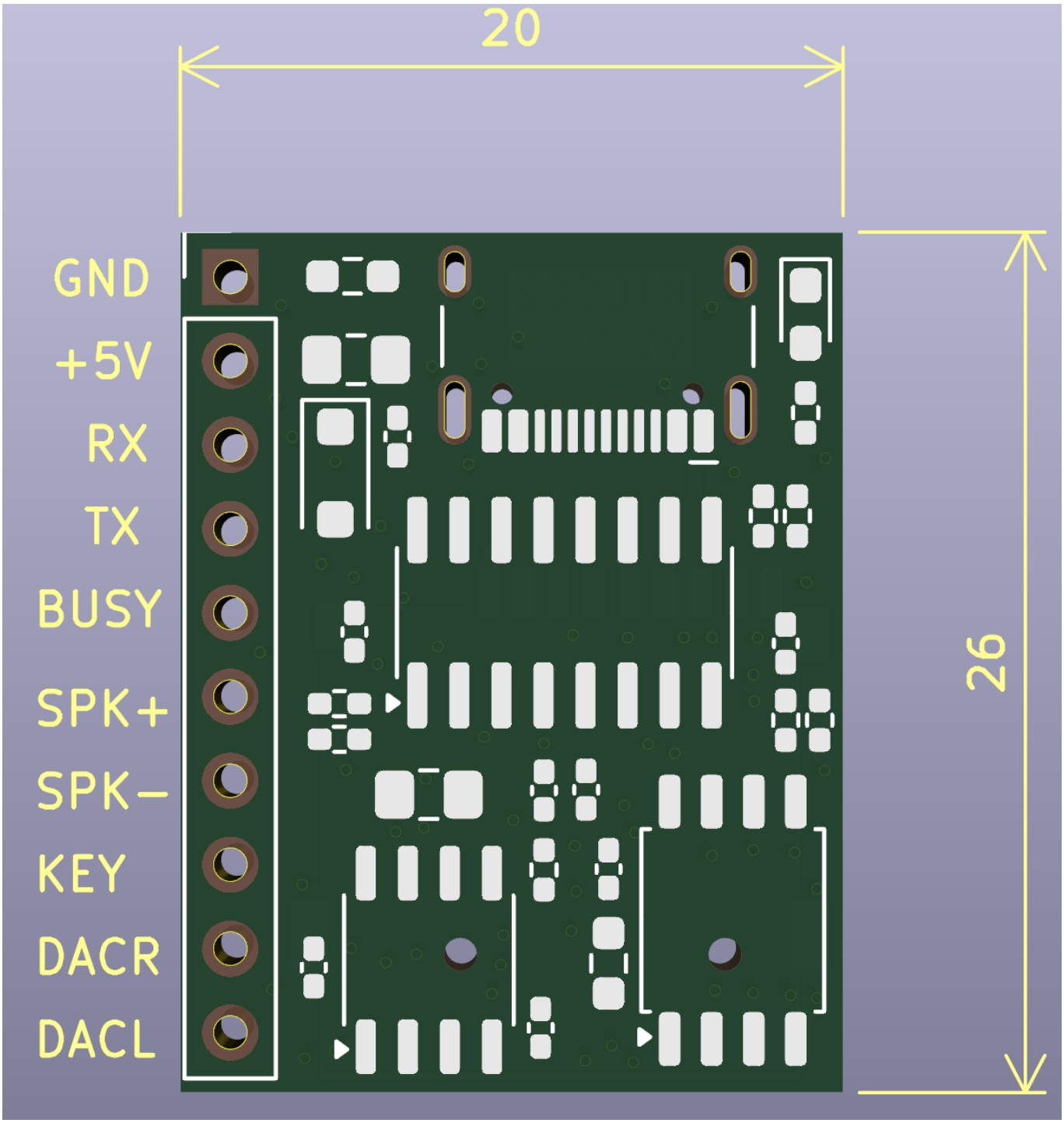
引脚排列



UART 通讯协议 115200 N 8 1

- 发送

head	length	CMD	DATA	XOR	tail
02	04	A0	01	FF	03

- 回复

head	length	CMD	ERR	XOR	tail

head	length	CMD	ERR	XOR	tail
02	04	A0	00	FF	03

- 解释：
  - heat：帧头 一个字节 0x02
  - tail：帧尾 一个字节 0x03
  - length：长度 一个字节 从长度到校验的长度，包含长度和校验
  - CMD：命令字 一个字节
  - DATA：数据区，不定长
  - XOR：校验，一个字节的xor校验,调试可以填写一个字节FF，当此字节为FF时，为不校验
- 指令列表：
  - 0xA0 播放
  - 0xA1 暂停
  - 0xA2 播放/暂停
  - 0xA3 下一曲
  - 0xA4 上一曲
  - 0xA5 停止
  - 0xA6 音量加
  - 0xA7 音量减
  - 0xA8 设置音量
  - 0xA9 静音
  - 0xAA 解除静音
  - 0xAB 播放指定曲目(通配符)
  - 0xAC 播放指定曲目（带扩展名）
  - 0xAD 播放指定曲目（数字）

指令详述

- 系统复位：0xA0/A1/A2/A3/A4/A5/A6/A7/A9/AA

HEAD	LENGTH	CMD	CHK	TAIL
0x02	1Byte	0xA0	1Byte	0x03

- 播放/暂停：0xC2

HEAD	LENGTH	CMD	CHK	TAIL
0x02	1Byte	0xC2	1Byte	0x03

- 设置音量：0xA8

HEAD	LENGTH	CMD	VOLUME	CHK	TAIL
0x02	1Byte	0xA8	0-30	1Byte	0x03

  - VOLUME： 0-30
- 播放指定曲目(通配符)：AB

HEAD	LENGTH	CMD	STA	NAME	END	CHK	TAIL
0x02	1Byte	0xAB	'$'	'abc'	'$'	1Byte	0x03

  - STA： 名字起始，字符'$'(0x24)。
  - END : 名字结束，字符'$'(0x24)。

    - NAME： 名字字符串，支持通配符,注:会自动增加扩展名，实例为3字节'abc'（61 62 63）,实际上是播放 "abc.*"，程序自动增加 ".*" 通配符扩展名，"ab*"也是合法的，会播放"ab*.*"
    - 文件必须存在否则播放失败
- 播放指定曲目(全路径)：AC

HEAD	LENGTH	CMD	STA	NAME	END	CHK	TAIL
0x02	1Byte	0xAC	'$'	'abc.mp3'	'$'	1Byte	0x03

    - STA： 名字起始，字符'$'(0x24)。
    - END : 名字结束，字符'$'(0x24)。
    - NAME： 名字字符串,必须写全文件名，例如：文件名采用8+3格式，只识别文件名的前2个字符，例如：01.mp3和010.mp3是不同的
    - 文件必须存在否则播放失败
- 播放指定曲目(文件号)：AD

HEAD	LENGTH	CMD	FH	FL	CHK	TAIL
0x02	1Byte	0xAD	1Byte	1Byte	1Byte	0x03

    - FH，FL：组合成以恶搞16bit的整形，例如 FH=0x00 FL=0x02 实际上是播放0x0002曲目，曲目的顺序号是拷贝进磁盘的顺序
    - 文件必须存在否则播放失败
- 指定文件号播放：0xC8

HEAD	LENGTH	CMD	NUM	LOOP	CHK	TAIL
0x02	1Byte	0xC8	1-99	1Byte	1Byte	0x03

    - NUM： 文件名号码。注：文件名采用8+3格式，只识别文件名的前2个字符，例如：01.mp3和01月光小夜曲.mp3是相同的，月光小夜曲可有可无，但前面序号必须有，存放文件时需注意。
    - LOOP： 循环次数，0为单曲无限循环
    - 文件必须存在否则播放失败

附录1：校验

- 校验方式为校验和，计算方式为数据累计和取反加1的方式
- 举例发送系统复位指令A0：0x02 0x02 0xA0 CheckSum 0x03
    - 其中头尾分别为02 03
    - 计算：第二个02 + 指令A0，相加，结果取反再+1，得到CheckSum
    - 校验：数据段+校验后的值为0
    - 详细算法

```
uint8_t buf[16];
int index = 0;
buf[index++] = 0x02;     //头
buf[index++] = 0x00;     //长度，临时赋值0
buf[index++] = 0xA0;     //指令

buf[1] = index - 1;      //数据填充完成后，重新赋值长度(0x02)

uint16_t sum = 0;

for(int i = 0; i < index - 1; i++)
```

```
    {
        sum += buf[1+i];    //从第一个字节开始累加，去除头，
    }

    //此时累加完成，sum：0x02 + 0xA0 = 0xA2

    sum = ~sum + 1;          //取反加1
    //sum = 0xff5D + 1 = 0xff5E

    buf[index++] = (u8)sum; //强制转换成uint8_t类型,截断后为 0x5E
    buf[index++] = 0x03;    //尾

    //此时buf = {0x02,0x02,0xA0,0x5E,0x03}

    //下面校验
    //0x02 + 0xA0 + 0x5E = 0x0100
    //将0x100强转为uint8_t后，截断后 = 0，校验通过
```

附录2： ERRCODE 错误定义：

- 0：指令正确，数据正确，且执行正确
- 1：一个通用的错误，可能是一个未定义的错误
- 2：未识别的指令
- 3：参数错误
- 4：帧错误，此包数据未找到帧头，或者未找到帧尾
- 5：长度错误
- 6：校验错误