



## 心率血氧检测模块



## 销售与服务联系

### 东莞野火科技有限公司

地址：东莞市大岭山镇石大路 2 号艺华综合办公大楼 301 1 2 3 4 楼

官网：<https://embedfire.com>

论坛：<http://www.firebbs.cn>

资料：<https://doc.embedfire.com>

天猫：<https://yehuosm.tmall.com>

京东：<https://yehuo.jd.com/>

邮箱：[embedfire@embedfire.com](mailto:embedfire@embedfire.com)

电话：0769-33894118

扫码获得更多精彩



野火百科



野火电子



野火天猫店



野火京东店



野火抖音号



野火视频号



野火B站号



野火小师妹

## 第一章 产品介绍

### 1.1 模块简介

心率血氧检测模块是一种集成了红外光和可见红光 LED、光电探测器以及信号处理电路的设备，利用人体组织在血管搏动时造成透光率不同来实时监测心率和血氧饱和度，主芯片为 MAX30102，使用 IIC 接口通信

注：测量时请尽量保持模块和手指相对静止，模块不移动和抖动

### 1.2 参数特性

- ◆ 超低功耗: < 1mW
- ◆ 超低待机电流 : 0.7  $\mu$  A
- ◆ 内部采样 ADC: 15~18 位, 受 LED 脉冲宽度配置影响
- ◆ 工作电压范围: 3.3V~5V
- ◆ 工作温度范围: -40°C~+85°C

## 第二章 使用说明

### 2.1 模块说明

(建议一边打开模块原理图并且一边对着芯片数据手册看)

U3 为 LDO 低压差线性稳压器，用于将 3.3V 电压转成 1.8V 供 MAX30102 芯片使用

U1 为 MAX30102 芯片

NC0~5: 空脚，连接到 PCB 焊盘以实现机械稳定性

VDD: 芯片电源，1.8V 供电

VLED+\_0~1: LED 电源，可见红光 (RED) 和红外光 (IR)，3.3V 供电

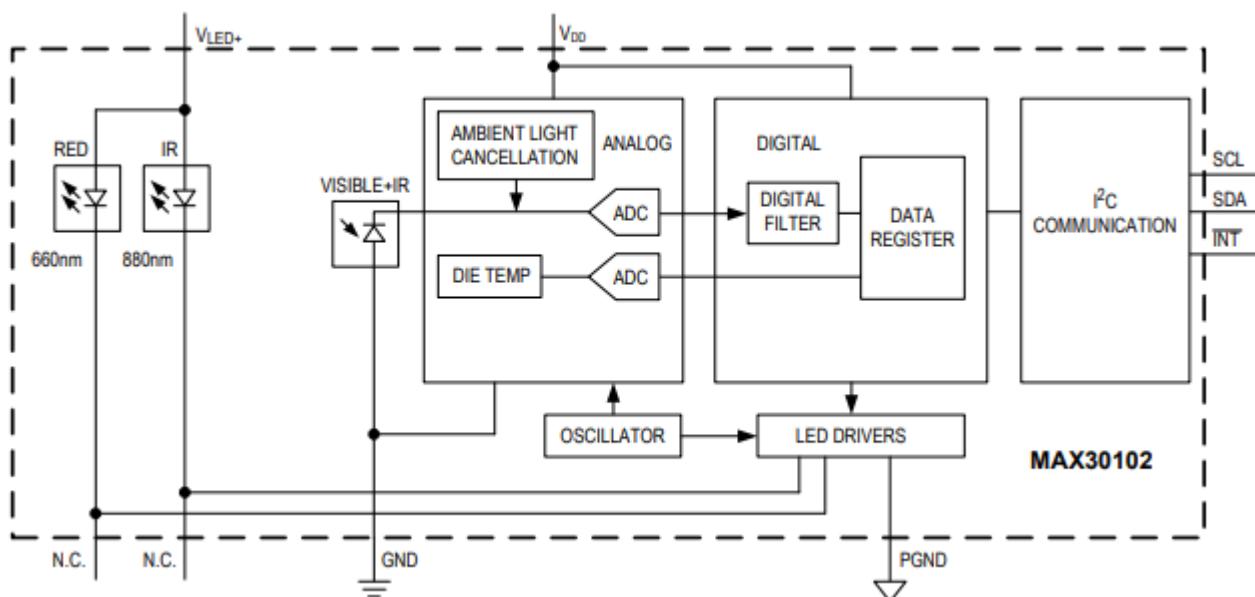
GND: 芯片接地

PGND: LED 接地

INT: 中断，低电平有效，默认接上拉电阻

SCL: I2C 的串行时钟线 SCL， 默认接上拉电阻

SDA: I2C 的双向串行数据线 SDA， 默认接上拉电阻



心率血氧模块对外发射可见红光 (RED) 和红外光 (IR)，光电探测器 (VISIBLE+IR) 会检测通过皮肤反射回来的光线并将其转换为电信号送入模块内部的信号处理电路，电路对这些反射光进行放大、环境光消除、模数转换和滤波处理，处理后将数据保存在 FIFO 数据寄存器里，主控通过 I2C 接口读取数据寄存器数据

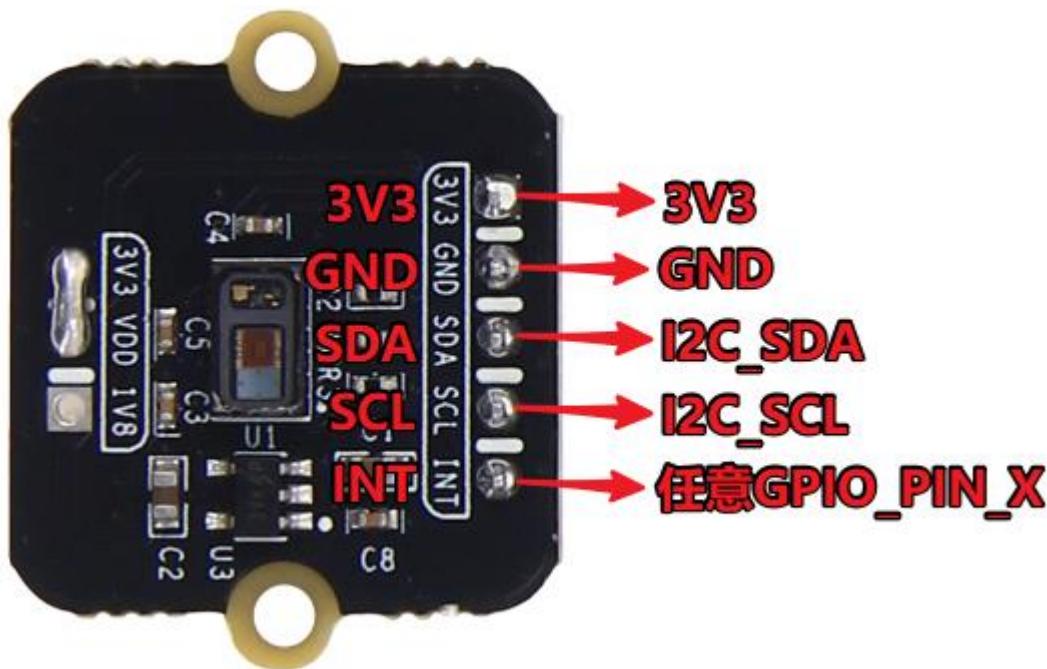
红光和红外光穿透人体组织，并使用光电探测器测量反射光量，也就是利用人体组织在血管搏动时造成透光率不同来进行心率和血氧饱和度测量

血液中有含氧血红蛋白（O<sub>2</sub>Hb）和脱氧血红蛋白（HHb），这两种血红蛋白的吸收光谱不一样，脱氧血红蛋白（HHb）吸收更多的红光，而含氧血红蛋白（O<sub>2</sub>Hb）吸收更多的红外光

**心率测量：**当心脏跳动时，血液被泵入和泵出，反射回来的光线强度会发生变化（吸收红外光 IR 的量会变化），光电探测器接收透过皮肤反射回来的光线，将其转化为电信号并产生变化的波形，通过测量这些电信号的变化来计算出心率

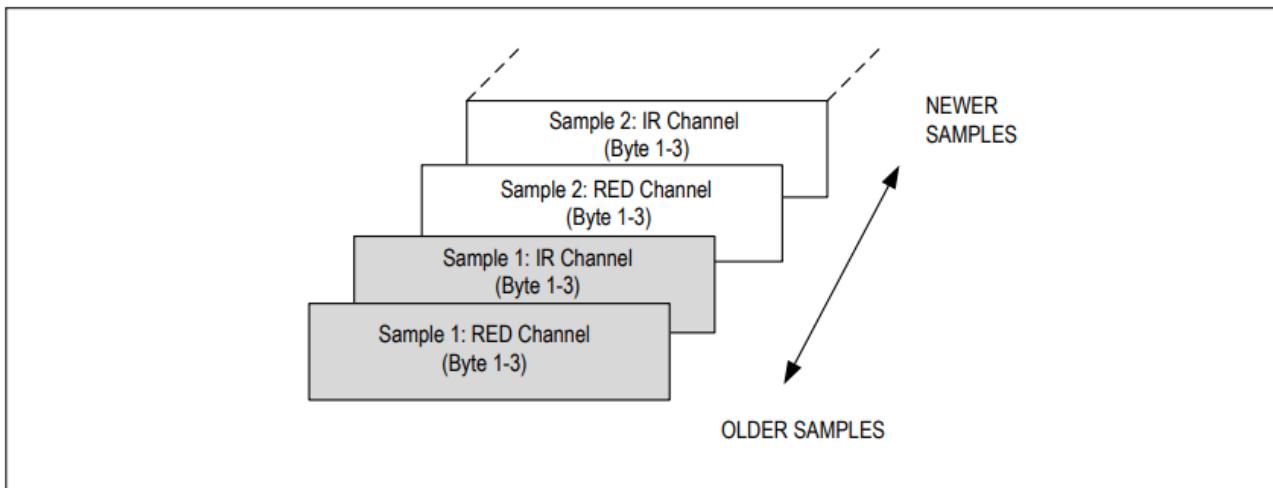
**血氧测量：**通过光电探测器检测反射回来的光线强度，可以判断红光、红外光被吸收的多少，从而推断出含氧血红蛋白（O<sub>2</sub>Hb）和脱氧血红蛋白（HHb）所占的比例可以估算血液中的氧气含量，并计算出血氧饱和度

## 2.2 模块接口



编号	心率血氧检测模块	主控
1	3V3	3V3
2	GND	GND
3	SDA	I2C_SDA
4	SCL	I2C_SCL
5	INT	任意 GPIO_PIN_X

## 2.3 程序流程



MAX30102 芯片的 FIFO 数据寄存器可存储 32 个数据样本，每个样本大小取决于配置通道的数量

当配置为 Heart Rate 模式时，每个样本就是一个 IR 通道组成，3 个字节数据大小

当配置为 SpO2 模式时，每个样本由 RED 和 IR 两个通道组成，6 个字节的数据大小，前 3 个字节为 RED 数据，后 3 个字节为 IR 数据，因此 FIFO 最多可以存储 192 个字节的数据

读取 FIFO\_DATA 寄存器时，不会自动递增 I2C 寄存器地址，而是从同一个地址反复读取数据，在 Heart Rate 模式下，一个样本为 3 个字节数据，需要调用 3 次 I2C 字节读取才能获得一个完整的样本

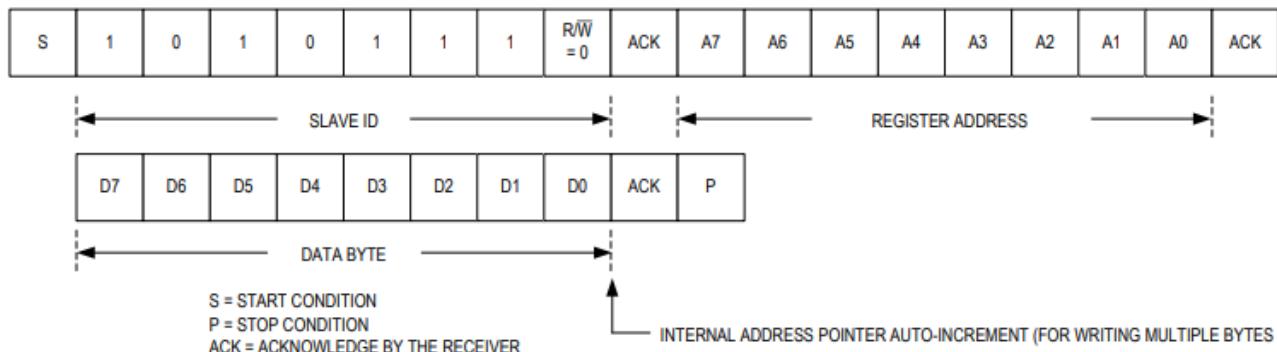
BYTE 1							FIFO_DATA[17]	FIFO_DATA[16]
BYTE 2	FIFO_DATA[15]	FIFO_DATA[14]	FIFO_DATA[13]	FIFO_DATA[12]	FIFO_DATA[11]	FIFO_DATA[10]	FIFO_DATA[9]	FIFO_DATA[8]
BYTE 3	FIFO_DATA[7]	FIFO_DATA[6]	FIFO_DATA[5]	FIFO_DATA[4]	FIFO_DATA[3]	FIFO_DATA[2]	FIFO_DATA[1]	FIFO_DATA[0]

上图为 18 位 ADC 采样一个通道的数据，并将每 bit 填充到 FIFO\_DATA 寄存器的数据排列情况，从 FIFO\_DATA[17]开始，不使用 FIFO\_DATA[18]~[23]

ADC Resolution	MSB															
	FIFO_DATA[17]	FIFO_DATA[16]	:	FIFO_DATA[12]	FIFO_DATA[11]	FIFO_DATA[10]	FIFO_DATA[9]	FIFO_DATA[8]	FIFO_DATA[7]	FIFO_DATA[6]	FIFO_DATA[5]	FIFO_DATA[4]	FIFO_DATA[3]	FIFO_DATA[2]	FIFO_DATA[1]	FIFO_DATA[0]
18-bit																
17-bit																
16-bit																
15-bit																

FIFO 数据每个通道大小都是 3 个字节，数据左对齐，无论 ADC 分辨率如何设置，MSB 始终位于同一位置，FIFO\_DATA[2]~[0]随 ADC 分辨率配置来决定使用

通过 I2C 接口通信，可以对 MAX30102 芯片的寄存器进行写入和读取数据

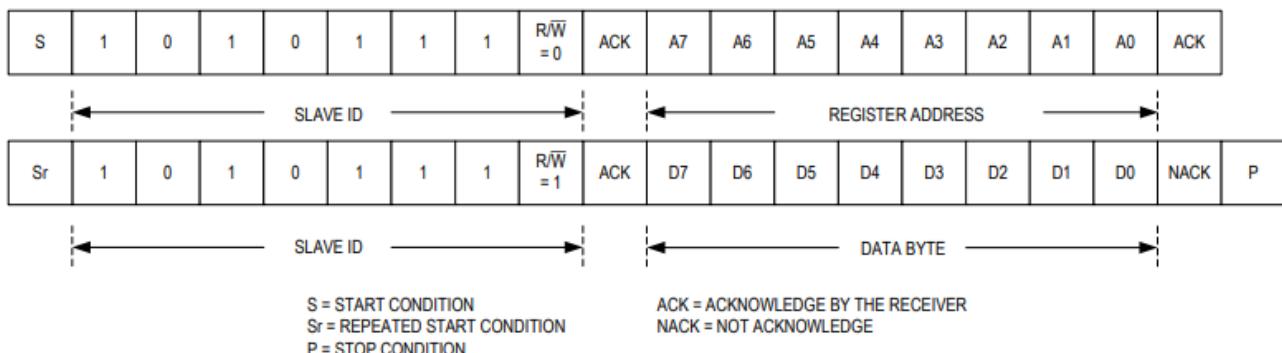


向 MAX30102 的寄存器写入一个字节数据：

- 1.发送起始信号(S)
- 2.确认 7 位固定从机地址和 1 位写入模式 (SLAVE ID)  
R/W=0 写入模式
- 3.从机应答信号 (ACK)
- 4.发送要写入的寄存器地址 (REGISTER ADDRESS)
- 5.从机应答信号 (ACK)
- 6.主机发送具体数据 (DATA BYTE)

**DATA BYTE:** 要发送的数据，8 位，一个字节大小

- 7.从机应答信号 (ACK)
- 8.发送终止信号 (P)



从 MAX30102 的寄存器读取一个字节数据：

- 1.发送起始信号(S)
- 2.确认 7 位固定从机地址和 1 位写入模式 (SLAVE ID)  
R/W=0 写入模式
- 3.从机应答信号 (ACK)
- 4.发送要写入的寄存器地址 (REGISTER ADDRESS)
- 5.从机应答信号 (ACK)
- 6.重新发送起始信号 (Sr)

7. 确认 7 位固定从机地址和 1 位读取模式 (SLAVE ID)

R/W=1 读取模式

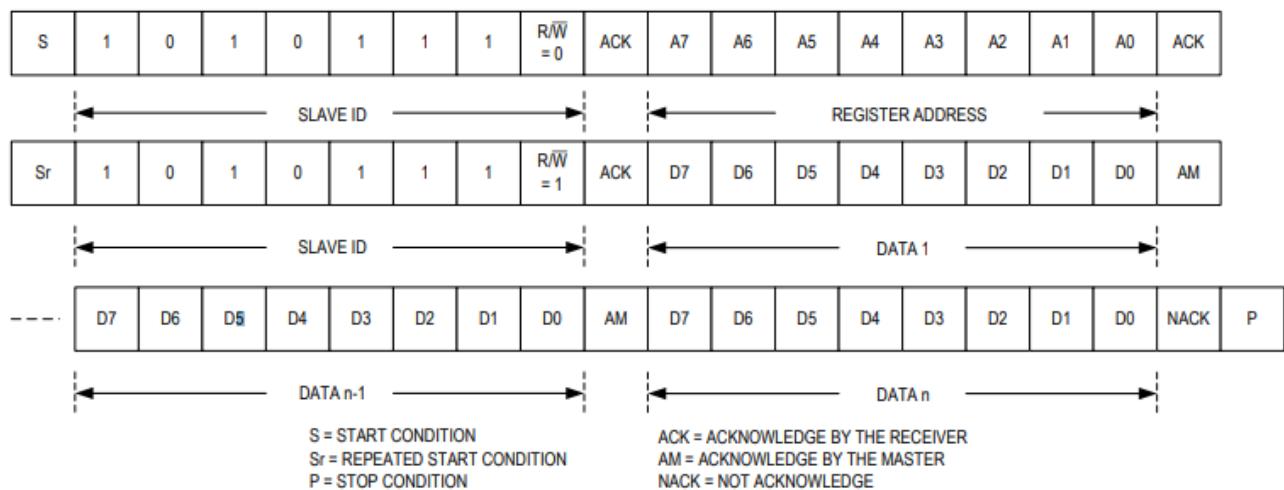
8. 从机应答信号 (ACK)

9. 从机发送具体数据 (DATA BYTE)

DATA BYTE: 要发送的数据, 8 位, 一个字节大小

10. 主机不应答信号 (NACK)

11. 发送终止信号 (P)



从 MAX30102 的寄存器读取多个字节数据:

1. 发送起始信号(S)

2. 确认 7 位固定从机地址和 1 位写入模式 (SLAVE ID)

R/W=0 写入模式, R/W=1 读取模式

3. 从机应答信号 (ACK)

4. 发送要写入的寄存器地址 (REGISTER ADDRESS)

5. 从机应答信号 (ACK)

6. 重新发送起始信号 (Sr)

7. 确认 7 位固定从机地址和 1 位读取模式 (SLAVE ID)

8. 从机应答信号 (ACK)

9. 从机发送数据 (DATA1)

10. 主机应答信号 (AM)

-----  
11. 从机发送数据 (DATA n-1)

12. 主机应答信号 (AM)

13. 从机发送数据 (DATA n)

14. 主机不应答信号 (NACK)

15. 发送终止信号 (P)

在 SpO2 模式下，一个样本包含 6 个字节的数据，需要调用 6 次 I2C 字节读取以获得一个完整的样本，要读取 FIFO\_DATA 之后的下一个寄存器，需要 I2C 写入命令来更改读取指针的位置  
下面以伪代码形式表示从 FIFO\_DATA 寄存器读取 6 个字节的样本数据

IIC_Start	发送起始信号
IIC_Send_Byt(0xAE)	发送从机地址+写模式
while (IIC_Wait_ACK)	主机等待从机应答
IIC_Send_Byt(FIFO_DATA)	发送 FIFO 寄存器地址
IIC_Start	再次发送起始信号
IIC_Send_Byt(0xAF)	发送从机地址+读模式
while (IIC_Wait_ACK)	主机等待从机应答
for (i= 0;i<6;i++)	循环 6 次
{	
DATA[i] = IIC_Read_Byt	读取 FIFO 字节数据
if (i != 6-1)	
{IIC_AM}	主机应答
else	
{IIC_NACK}	最后 1 个字节读完后主机不应答
}	
IIC_Stop	发送终止信号

用任意单片机确定使用 IIC 接口（硬件 IIC）或者任选 GPIO（软件 IIC）来驱动心率血氧模块

使用硬件 IIC：

1. 初始化 MCU 对应的 GPIO 脚，SCL 和 SDA 配置为复用开漏输出模式，INT 配置为浮空输入模式
2. 配置 IIC 结构体，初始化硬件 IIC，
3. 调用库函数编写 IIC 向寄存器写入一个字节、读取一个字节和读取多个字节的函数
4. 根据数据手册寄存器表，设置采样率，工作模式，led 电流等等参数，向寄存器写入参数完成初始化
5. 当中断发生时，从 FIFO\_DATA 寄存器读取数据，对采集到的数据样本进行滤波
6. 根据红外光信号的变化模式，使用算法计算心率
7. 根据红外光和可见红光信号的强度差异，使用算法估算血氧饱和度

使用软件 IIC:

1. 初始化 MCU 对应的 GPIO 脚，SCL 和 SDA 配置为开漏输出模式，INT 配置为浮空输入模式

2. 编写时序基本单元：起始信号、终止信号、写入一个字节、读取一个字节、主机等待从机应答、主机应答、主机不应答（SCL 和 SDA 需要拉高/拉低的动作来模拟 IIC 的硬件时序）

3. 封装编写 IIC 向寄存器写入一个字节、读取一个字节和读取多个字节的函数

4. 根据数据手册寄存器表，设置采样率，工作模式，led 电流等等参数，向寄存器写入参数完成初始化

5. 当中断发生时，从 FIFO\_DATA 寄存器读取数据，对采集到的数据样本进行滤波

6. 根据红外光信号的变化模式，使用算法计算心率

7. 根据红外光和可见红光信号的强度差异，使用算法估算血氧饱和度

更多详细内容可参考芯片数据手册和野火 STM32F1、F4 开发板教程 I2C 章节