

千万GD32机器人指南

V1.0 2022.12.17

《千万GD32机器人指南》是千万撰写的，用于说明如何在千万的GD32框架下，基于GD32 Firmware库进行机器人控制。本文档仅适用于参加机器人竞赛的同学学习，禁止用于一切商业用途。

本文档提供的一系列类，设计核心是灵活地对特定外部设备做出调整，方便后来者继承，因此SPI、IIC等类只提供基本功能。具体使用还需开发。

千万是本指南的作者，读者可以通过[邮箱](mailto:qianwan233@foxmail.com)联系到千万。

本文档撰写基于GigaDevice.GD32F30x\_DFP.2.2.1与Firmware V2.1.0。

|  |  |
| --- | --- |
| 更新日期 | 内容 |
| 2022/12/17 | 新建文档 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

目录

[阻塞型延时 3](#_Toc122207871)

[文件通览 3](#_Toc122207872)

[函数解析 3](#_Toc122207873)

[串行外设接口(软件) 4](#_Toc122207874)

[文件通览 4](#_Toc122207875)

[函数解析 5](#_Toc122207876)

[串行外设接口(硬件) 7](#_Toc122207877)

[文件通览 7](#_Toc122207878)

[函数解析 8](#_Toc122207879)

# 阻塞型延时

## 文件通览

|  |  |  |
| --- | --- | --- |
| 文件名 | 版本 | 内容 |
| Delay.c | V1.0\_2022/11/20 | 延迟函数体 |
| Delay.h | V1.0\_2022/11/20 | 延迟函数声明 |

本功能为其他

|  |  |
| --- | --- |
| 函数名 | 内容 |
| void qDelay\_init(uint8\_t SYSCLK) | 初始化系统定时器 |
| void qDelay\_ms(uint16\_t nms) | 阻塞延时nms毫秒 |
| void qDelay\_us(uint32\_t nus) | 阻塞延时nus微妙 |

本章共包含2个文件，3个函数。

## 函数解析

阻塞型函数延迟时间将略长于设置时间，如若需要准确延时，需要在固定代码优化等级下手动测算并修正。

**void qDelay\_init(uint8\_t SYSCLK)**

**简介：**初始化系统时钟

**输入：**系统定时器频率，单位MHz

**返回：**无

**void qDelay\_ms(uint16\_t nms)**

**功能：**以阻塞形式延时一段时间

**输入：**阻塞延迟时间，单位ms

**返回：**无

**void qDelay\_us(uint32\_t nus)**

**功能：**以阻塞形式延时一段时间

**输入：**阻塞延迟时间，单位us

**返回：**无

# 串行外设接口(软件)

## 文件通览

|  |  |  |
| --- | --- | --- |
| 文件名 | 版本 | 内容 |
| SPI\_Software\_GD.cpp | V1.4\_2022/11/21 | SPI函数 |
| SPI\_Software\_GD.h | V1.4\_2022/11/21 | SPI基类 |

内联函数位于头文件中。

|  |  |
| --- | --- |
| 函数名 | 内容 |
| void SPI\_Init( uint32\_t MOSI\_Port,  uint32\_t MOSI\_Pin, uint32\_t MISO\_Port,  uint32\_t MISO\_Pin, uint32\_t SCL\_Port,  uint32\_t SCL\_Pin, uint32\_t CS\_Port,  uint32\_t CS\_Pin) | 初始化函数，为SPI基类指定管脚 |
| inline void CS\_0(void) | 片选引脚置0 |
| inline void CS\_1(void) | 片选引脚置1 |
| inline void MISO\_OD(void) | MISO引脚设置为开漏 |
| inline void MISO\_PP(void) | MISO引脚设置为推挽 |
| uint8\_t SPI\_ExchangeOneByte(  uint8\_t Data) | 通过MOSI引脚发送1字节数据，并通过MISO引脚读取1字节数据 |

本章共包含2个文件，6个用户函数。

## 函数解析

软件SPI函数默认100Kbit波特率，这是通过qDelay\_us延迟函数控制的。受延时函数精度及代码优化等级影响，波特率将略小于预设，需要调整波特率时，可在调整SPI\_Software\_GD.h头文件中声明的cSPI:Delay函数实现方式以达到预期。

**void SPI\_Init( uint32\_t MOSI\_Port, uint32\_t MOSI\_Pin,**

**uint32\_t MISO\_Port, uint32\_t MISO\_Pin,**

**uint32\_t SCL\_Port, uint32\_t SCL\_Pin,**

**uint32\_t CS\_Port, uint32\_t CS\_Pin )**

**简介：**指定MOSI、MISO、SCL、CS功能的实现引脚。

**输入：**MOSI、MISO、SCL、CS引脚端口与位号

**返回：**无

**补充：**对于不需要的功能引脚，建议指向空闲引脚而非未知地址以避免发生溢出。使用三线半双工SPI时，可以将MOSI与MISO指向同一引脚，并通过变更引脚的工作模式与发送数据内容实现半双工。

**inline void CS\_0(void)**

**简介：**将片选引脚置0

**输入：**无

**返回：**无

**inline void CS\_1(void)**

**简介：**将片选引脚置1

**输入：**无

**返回：**无

**inline void MISO\_OD(void)**

**简介：**将MISO引脚设置为开漏模式

**输入：**无

**返回：**无

**补充：**本函数适用范围是三线半双工SPI，在接收信号时将MISO引脚设置为开漏模式并拉高，从而将信号线控制权交予从设备以实现数据接收。

**inline void MISO\_PP(void)**

**简介：**将MISO引脚设置为推挽模式

**输入：**无

**返回：**无

**补充：**本函数适用范围是三线半双工SPI，在发送信号时将MISO引脚设置为推挽模式，从而控制信号线以实现数据发送。

**uint8\_t SPI\_ExchangeOneByte(uint8\_t Data)**

**简介：**从MOSI引脚发送1字节数据，并从MISO引脚接收1字节数据

**输入：**发送数据

**返回：**接收数据

**补充：**本函数实现SPI的基本数据交换功能，本函数不会自动完成片选，需要手动调用函数。使用四线全双工SPI时可以使用本函数完成数据的同时收发。使用三线半双工SPI时，本函数需要配合MISO\_OD函数和MISO\_PP函数以实现数据的发送或接收。另外，半双工发送时，函数返回的数据等同于发送的数据；半双工接收时，函数必须传入0xFF，返回的数据为接收到的数据。

# 串行外设接口(硬件)

## 文件通览

|  |  |  |
| --- | --- | --- |
| 文件名 | 版本 | 内容 |
| SPI\_GD.cpp | V1.1\_2022/11/30 | SPI函数 |
| SPI\_GD.h | V1.1\_2022/11/30 | SPI基类 |

内联函数位于头文件中。

|  |  |
| --- | --- |
| 函数名 | 内容 |
| void SPI\_Init( uint32\_t SPI,  uint32\_t CS\_Port, uint32\_t CS\_Pin ) | 不启用DMA时的初始化函数，函数将初始化SPI类 |
| void SPI\_Init( uint32\_t SPI,  uint32\_t CS\_Port, uint32\_t CS\_Pin,  uint32\_t DMA,  dma\_channel\_enum DMA\_CH) | 启用发送或接收DMA时的初始化函数，函数将初始化SPI类 |
| void SPI\_Init( uint32\_t SPI,  uint32\_t CS\_Port, uint32\_t CS\_Pin,  uint32\_t DMAt,  dma\_channel\_enum DMA\_CHt,  uint32\_t DMAr,  dma\_channel\_enum DMA\_CHr) | 同时启用发送和接收DMA时的初始化函数，函数将初始化SPI类 |
| inline void CS\_0(void) | 片选引脚置0 |
| inline void CS\_1(void) | 片选引脚置1 |
| uint8\_t SPI\_ExchangeOneByte(  uint8\_t Data) | 与从设备交换1字节数据 |
| uint8\_t Transmit\_DMA(  uint8\_t \*data,uint16\_t num) | 使用DMA方式传输指定长度数据 |
| uint8\_t IRQ\_Tx(void) | DMA传输完成中断处理函数 |
| uint16\_t Receive\_DMA(  uint8\_t \*data,uint16\_t num) | 使用DMA方式接收指定长度数据 |
| uint8\_t IRQ\_Rx(void) | DMA传输完成中断处理函数 |

本章共包含2个文件，10个用户函数。

## 函数解析

本类中的DMA收、发功能分别可以通过设置SPI\_GD.h头文件中SPI\_USE\_TX\_DMA和SPI\_USE\_RX\_DMA宏定义的值来使能或失能。调整宏定义也会导致SPI\_Init函数的重载，下面的三种SPI\_Init函数只有一种会被启用。

**void SPI\_Init(uint32\_t SPI, uint32\_t CS\_Port, uint32\_t CS\_Pin)**

**简介：**收发DMA都未启用时的初始化函数

**输入：**SPI外设指针、片选引脚端口、片选引脚位号

**返回：**无

**void SPI\_Init(uint32\_t SPI, uint32\_t CS\_Port, uint32\_t CS\_Pin,**

**uint32\_t DMA,dma\_channel\_enum DMA\_CH)**

**简介：**收或发DMA启用时的初始化函数

**输入：**SPI外设指针、片选引脚端口、片选引脚位号、DMA指针、DMA通道

**返回：**无

**void SPI\_Init(uint32\_t SPI, uint32\_t CS\_Port, uint32\_t CS\_Pin,**

**uint32\_t DMAt, dma\_channel\_enum DMA\_CHt,**

**uint32\_t DMAr,dma\_channel\_enum DMA\_CHr)**

**简介：**收与发DMA都启用时的初始化函数

**输入：**SPI外设指针、片选引脚端口、片选引脚位号、发DMA指针、发DMA通道、收DMA指针、收DMA通道

**返回：**无

**inline void CS\_0(void)**

**简介：**将片选引脚置0

**输入：**无

**返回：**无

**inline void CS\_1(void)**

**简介：**将片选引脚置1

**输入：**无

**返回：**无

**uint8\_t SPI\_ExchangeOneByte(uint8\_t Data)**

**简介：**与从设备交换1字节数据

**输入：**发送数据

**返回：**接收数据

**补充：**本函数不会控制片选信号，需要用户手动调整