**物料分拣流水线技术文档**

1. **项目名称**

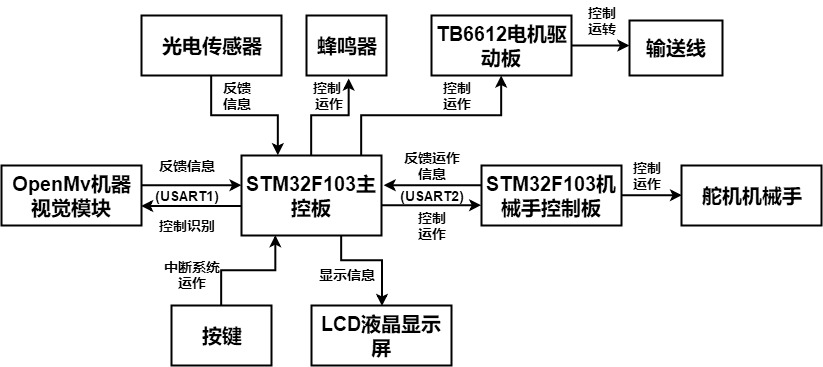
**物料分拣流水线。**

1. **设计要求**

**将单条输送线上的不同颜色或不同形状的物料进行分拣，并将其放入到指定的位置。**

1. **项目架构的设计**

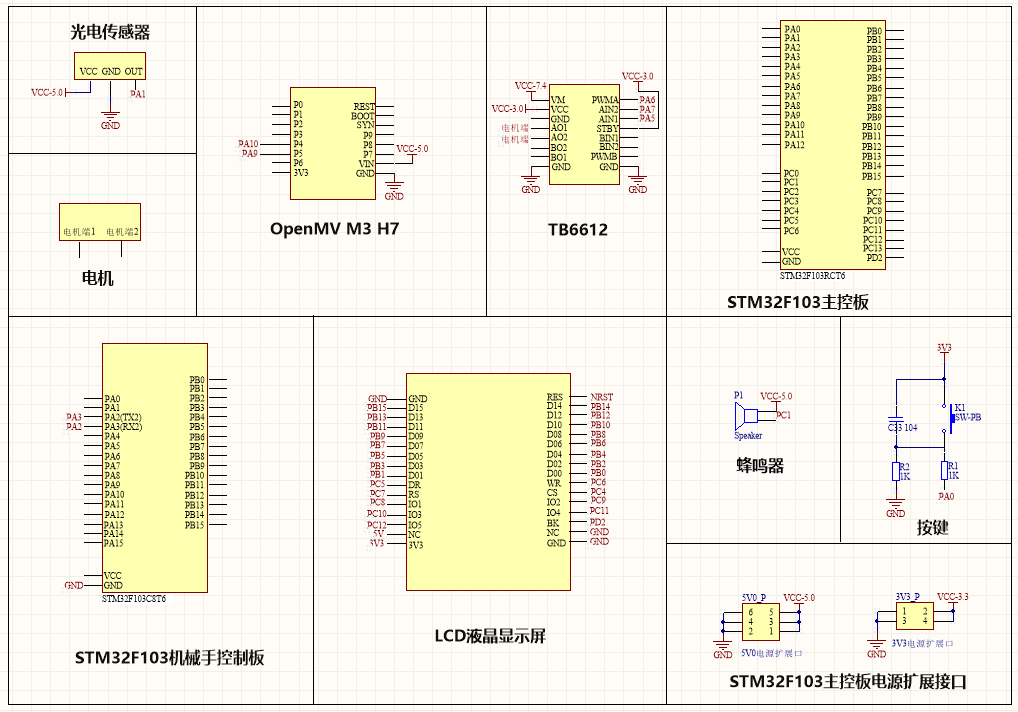
**本项目涉及到十大模块，分别是TB6612电机驱动板、物料输送线、光电传感器、STM32F103主控板、STM32F103机械手控制板，舵机机械手、OpenMv机器视觉模块、按键、蜂鸣器以及LCD液晶显示屏；**

****

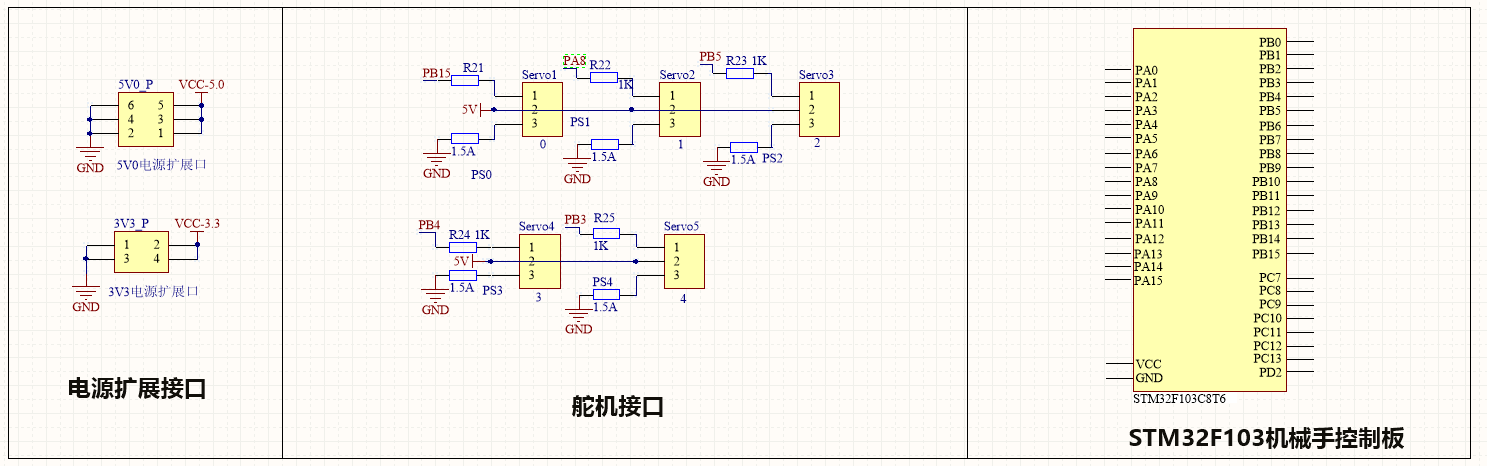
**（项目架构图）**

**模块功能：**

1. **TB6612电机驱动板：驱动物料输送线中的直流电机转动；**
2. **物料输送线：输送物料;**
3. **光电传感器：检测物料是否到达机器视觉识别的区域；**
4. **STM32F103主控板：控制输送带运转、OpenMv机器视觉模块识别、LCD液晶显示屏显示操作信息、蜂鸣器的鸣响以及STM32F103机械手控制板的运作。接收光电传感器、OpenMv机器视觉模块以及STM32F103机械手控制板反馈的信息。STM32F103主控板为项目运行的大脑**
5. **机器视觉：对物料进行颜色或形状的识别；**
6. **STM32F103机械手控制板：控制舵机机械手的运作；**
7. **舵机机械手：对物料进行抓取以及放置；**
8. **LCD液晶显示屏：显示系统的工作状态，有利于故障的排查；**
9. **按键：按键按下中断整个系统的运作，可以有效避免在系统运行中出现不可挽回的错误；**
10. **蜂鸣器：当系统通信超时或系统停止运作时鸣响，起到报警作用；**
11. **项目详细方案的设计**
12. **电路图设计**

****

**（物料分拣流水线电路原理图）**

****

**（STM32机械手模块电路原理图）**

1. **系统软件设计**
2. **STM32F103主控板程序流程框图**

**超链接：** [**程序流程图\物料分拣流水线STM32F103主控板程序流程图.jpg**](程序流程图/物料分拣流水线STM32F103主控板程序流程图.jpg)

**STM32F103主控板程序流程框图文字说明**

1. **主程序流程框图文字描述：**

**首先开机系统初始化，STM32主控板通过USART1向OpenMv发送RTU检测数据帧，开启系统计时器TIM6，判断USART1\_RX\_Over是否置1（即USART1是否接收OpenMv返回的RTU检测数据帧）。若USART1\_RX\_Over不为 1，则判断TIM6\_Count计时变量是否为100（即计时时间是否为5S），若TIM6\_Count也不为100，则继续等待接收OpenMv返回的数据帧，若TIM6\_Count计时变量为100（即计时时间为5S），关闭系统计时器TIM6，TIM6\_ Count计时变量清0，判断USART1\_Send \_Count变量是否为5（即重复发送检测数据次数为5次），若不为5次，STM32主控板重新通过USART1向OpenMv发送RTU检测数据帧（提高通成功率），USART1\_Send\_Count变量加1，开启系统计时器TIM6，继续等待接收OpenMv返回的数据帧。若USART1\_Send \_Count变量为5时，LCD液晶显示屏显示“向视觉模块发送检测指令超时”， 蜂鸣器鸣响。若USART1\_RX\_Over为 1，关闭系统计时器TIM6，TIM6\_Count计时变量清0，USART1\_RX\_Over置0，校验USART1\_RX\_ Pack数组存放的RTU数据帧是否正确，若校验不正确，则清空USART1\_RX\_ Pack数组，继续开启系统计时器TIM6，继续等待接收OpenMv返回的数据帧，若校验正确，USART1\_Send \_Count变量清0，清空USART1\_RX\_Pack数组，LCD液晶显示屏显示“视觉模块响应正常”，显示时间为2S。**

**STM32主控板通过USART2向机械手发送RTU检测数据帧，开启系统计时器TIM6，判断USART2\_RX\_Over是否置1（即USART2是否接收机械手返回的RTU检测数据帧）。若USART2\_RX\_Over不为 1，则判断TIM6\_Count计时变量是否为100（即计时时间是否为5S），若TIM6\_Count也不为100，则继续等待接收机械手返回的数据帧，若TIM6\_Count计时变量为100（即计时时间为5S），关闭系统计时器TIM6，TIM6\_ Count计时变量清0，判断USART2\_Send \_Count变量是否为5（即重复发送检测数据次数为5次），若不为5次，STM32主控板重新通过USART2向机械手发送RTU检测数据帧（提高通成功率），USART2\_Send\_Count变量加1，开启系统计时器TIM6，继续等待接收机械手返回的数据帧。若USART2\_Send \_Count变量为5时，LCD液晶显示屏显示“向机械手发送检测指令超时”， 蜂鸣器鸣响。若USART2\_RX\_Over为 1，关闭系统计时器TIM6，TIM6\_Count计时变量清0，USART2\_RX\_Over置0，校验USART2\_RX\_ Pack数组存放的RTU数据帧是否正确，若校验不正确，则清空USART2\_RX\_ Pack数组，开启系统计时器TIM6，继续等待接收机械手返回的数据帧，若校验正确，USART2\_Send \_Count变量清0，清空USART2\_RX\_Pack数组，LCD液晶显示屏显示“机械手控制板响应正常”，显示时间为2S。**

**开启光电传感器，输送带运作，光电传感器是否检测到物体，Check\_Object是否为1，若检测不到物体，输送带继续运作，光电传感器继续检测，若检测到物体，关闭光电传感器，Check\_Object置0（避免下次调用出错），输送带0.45S后停止运作（保证物体停止在视觉模块识别范围）。**

**STM32主控板通过USART1向OpenMv发送RTU识别数据帧，开启系统计时器TIM6，判断USART1\_RX\_Over是否置1（即USART1是否接收OpenMv返回的RTU识别数据帧）。若USART1\_RX\_Over不为 1，则判断TIM6\_Count计时变量是否为100（即计时时间是否为5S），若TIM6\_Count也不为100，则继续等待接收OpenMv返回的数据帧，若TIM6\_Count计时变量为100（即计时时间为5S），关闭系统计时器TIM6，TIM6\_ Count计时变量清0，判断USART1\_Send \_Count变量是否为5（即重复发送检测数据次数为5次），若不为5次，STM32主控板重新通过USART1向OpenMv发送RTU识别数据帧（提高通成功率），USART1\_Send\_Count变量加1，开启系统计时器TIM6，继续等待接收OpenMv返回的数据帧。若USART1\_Send \_Count变量为5时，LCD液晶显示屏显示“向视觉模块发送识别指令超时”， 蜂鸣器鸣响。若USART1\_RX\_Over为 1，关闭系统计时器TIM6，TIM6\_Count计时变量清0，USART1\_RX\_Over置0，校验USART1\_RX\_ Pack数组存放的RTU数据帧是否正确，若校验不正确，则清空USART1\_RX\_ Pack数组，开启系统计时器TIM6，继续等待接收OpenMv返回的数据帧，若校验正确，USART1\_Send \_Count变量清0，清空USART1\_RX\_Pack数组，LCD液晶显示屏显示“视觉模块识别开始”。**

**开启系统计时器TIM6，判断USART1\_RX\_Over是否置1（即USART1是否接收OpenMv返回的RTU颜色数据帧）。若USART1\_RX\_Over不为 1，则判断TIM6\_Count计时变量是否为100（即计时时间是否为5S），若TIM6\_Count也不为100，则继续等待接收OpenMv返回的数据帧，若TIM6\_Count计时变量为100（即计时时间为5S），关闭系统计时器TIM6，TIM6\_ Count计时变量清0，判断USART1\_Send \_Count变量是否为5（即重复发送检测数据次数为5次），若不为5次，STM32主控板重新通过USART1向OpenMv发送RTU识别数据帧，USART1\_Send\_Count变量加1，开启系统计时器TIM6，继续等待接收OpenMv返回的数据帧。若USART1\_Send \_Count变量为5时，LCD液晶显示屏显示“接收视觉模块识别数据超时”， 蜂鸣器鸣响。若USART1\_RX\_Over为 1，关闭系统计时器TIM6，TIM6\_Count计时变量清0，USART1\_RX\_Over置0，校验USART1\_RX\_ Pack数组存放的RTU数据帧是否正确，若校验不正确，则清空USART1\_RX\_ Pack数组，开启系统计时器TIM6，继续等待接收OpenMv返回的数据帧，若校验正确，USART1\_Send \_Count变量清0。STM32主控板通过USART1向OpenMv发送RTU"OK"数据帧，解析接收到视觉模块返回的RTU颜色数据帧，并在LCD液晶显示屏显示识别到的颜色字符。**

**STM32主控板根据USART1接收到视觉模块识别颜色的数据通过USART2向机械手发送RTU“执行抓取指定颜色物料”数据帧, 开启系统计时器TIM6，判断USART2\_RX\_Over是否置1（即USART2是否接收机械手返回的RTU执行数据帧）。若USART2\_RX\_Over不为 1，则判断TIM6\_Count计时变量是否为100（即计时时间是否为5S），若TIM6\_Count也不为100，则继续等待接收机械手返回的数据帧，若TIM6\_Count计时变量为100（即计时时间为5S），关闭系统计时器TIM6，TIM6\_ Count计时变量清0，判断USART2\_Send \_Count变量是否为5（即重复发送检测数据次数为5次），若不为5次，STM32主控板重新通过USART2向机械手发送RTU执行数据帧，USART2\_Send\_Count变量加1，开启系统计时器TIM6，继续等待接收机械手返回的数据帧。若USART2\_Send \_Count变量为5时，LCD液晶显示屏显示“向机械手发送执行指令超时”， 蜂鸣器鸣响。若USART2\_RX\_Over为 1，关闭系统计时器TIM6，TIM6\_Count计时变量清0，USART2\_RX\_Over置0，校验USART2\_RX\_Pack数组存放的RTU数据帧是否正确，若校验不正确，则清空USART2\_RX\_ Pack数组，继续开启系统计时器TIM6，继续等待接收机械手返回的数据帧，若校验正确，USART2\_Send \_Count变量清0，清空USART2\_RX\_Pack数组，LCD液晶显示屏显示“机械手执行操作”**

**开启系统计时器TIM6，判断USART2\_RX\_Over是否置1（即USART2是否接收机械手返回的RTU执行完毕数据帧）。若USART2\_RX\_Over不为 1，则判断TIM6\_Count计时变量是否为100（即计时时间是否为5S），若TIM6\_Count也不为100，则继续等待接收机械手返回的数据帧，若TIM6\_Count计时变量为100（即计时时间为5S），关闭系统计时器TIM6，TIM6\_ Count计时变量清0，判断USART2\_Send \_Count变量是否为5（即重复发送执行数据次数为5次），若不为5次，USART2\_Send\_Count变量加1，开启系统计时器TIM6，继续等待接收机械手返回的数据帧。若USART2\_Send \_Count变量为5时，LCD液晶显示屏显示“接收机械手执行数据超时”，蜂鸣器鸣响。若USART2\_RX\_Over为 1，关闭系统计时器TIM6，TIM6\_Count计时变量清0，USART2\_RX\_Over置0，校验USART2\_RX\_Pack数组存放的RTU数据帧是否正确，若校验不正确，则清空USART2\_RX\_ Pack数组，继续开启系统计时器TIM6，继续等待接收机械手返回的数据帧，若校验正确，USART2\_Send \_Count变量清0，清空USART2\_RX\_Pack数组，LCD液晶显示屏显示“机械手执行完毕”清空USART1\_RX\_Pack数组，继续开启光电传感器，输送带继续运行，以此循环，实现对输送线上的物料进行分拣功能；**

**中断程序流程图文字描述：**

1. **USART1接收中断：中断接收OpenMv发送的RTU数据帧，并将数据保存到USART1\_RX\_Pack数组里，USART1\_RX\_Over置1（即证明接收完毕OpenMv发送的数据）返回主程序；**
2. **USART2接收中断：中断接收机械手发送的RTU数据帧，并将数据保存到USART2\_RX\_Pack数组里，USART2\_RX\_Over置1（即证明接收完毕机械手发送的数据）返回主程序；**
3. **计时中断：TIM6定时器每50ms中断一次，TIM6\_Count计数值加1（TIM6\_Count为100时，即计时时间为5S），返回主程序；**
4. **K1按键中断：设置按键按下停止整个系统的运行，输送线停止运作，主控板向机械手发送“停止运作”指令，LCD液晶显示屏显示“系统停止运行！”，蜂鸣器鸣响，系统程序停止运行。可以有效避免在系统运行中出现不可挽回的错误；**
5. **光电传感器EXTI中断：当检测到物体时，触发EXTI中断，将Check\_Object置1（可以利用该变量判断是否检测到物体），返回主程序；**
6. **OpenMv程序流程框图**

**超链接：**[**程序流程图\OpenMv模块程序流程图.jpg**](程序流程图/OpenMv模块程序流程图.jpg)

**OpenMv程序流程框图文字描述：**

**首先开机OpenMv系统初始化，判断是否接收到STM32主控板发送的RTU检测数据帧，若接收不到数据帧，则继续等待接收数据，若接收到数据帧，则校验data数组存放的RTU数据帧是否正确，若校验数据出错，则系统返回等待接收STM32主控板发送的检测指令状态，若校验数据成功，则将接收到的RTU检测数据帧发送回STM32主控板上，并清除data数组里的数据。**

**判断是否接收到STM32主控板发送的RTU识别数据帧，若接收不到数据帧，则继续等待接收数据，若接收到数据帧，则校验data数组存放的RTU数据帧是否正确，若校验数据出错，则系统返回等待接收STM32主控板发送的识别指令状态，若校验数据成功，则将接收到的RTU检测数据帧发送回STM32主控板上，并清除data数组里的数据，等待10ms（保证STM32主控板进入接收数据状态）。**

**OpenMv模块识别物料，并将识别到的颜色数据打包成RTU数据帧发送回STM32主控板上，等待接收STM32主控板发送的响应指令，若接收不到数响应指令，继续等待接收数据，若接收到数响应指令，校验data数组存放的RTU数据帧是否正确，若校验出错，则系统恢复到等待接收STM32主控板发送的响应指令状态，若校验成功，则解析data数组存放的RTU数据帧，并判断RTU数据帧中是否存在OK数据，若不存在OK数据（则证明STM32主控板接收识别颜色数据出错），则清除data数组里的数据，视觉模块恢复到识别颜色状态，重新发送识别颜色数据。若存在OK数据，则清除data数组里的数据，视觉模块恢复到等待接收STM32主控板发送的识别指令状态，以此循环，实现对物料的颜色识别；**

1. **STM32F103机械手控制板程序流程框图**

**超链接：**[**程序流程图\STM32F103机械手控制板程序流程图.jpg**](程序流程图/STM32F103机械手控制板程序流程图.jpg)

**1.STM32F103机械手控制板程序流程框图描述：**

**首先开机系统初始化，进入等待接收STM32主控板发送的RTU检测数据帧状态, 并判断USART\_RX\_Over是否置1（即USART2是否接收完STM32主控板发送的RTU数据帧）,若USART\_RX\_Over为0，则重新等待接收数据，若USART\_RX\_Over为1，则将USART\_RX\_Over置0（避免下次调用该变量时出错），同时校验Receive\_Master\_Data数组存放的RTU数据帧是否正确，若校验错误，则清除Receive\_Master\_Data数组中存放的数据，系统恢复到等待接收STM32主控板发送的RTU检测数据帧状态，若校验成功，则解析Receive\_Master \_Data数组里的数据。**

**判断操作类型，若操作类型为“IRON\_HAND\_CHACK”，则将接收到的RTU检测数据帧发送回STM32主控板上，清除Receive\_Master \_Data数组里的数据，系统恢复到等待接收STM32主控板发送的RTU检测数据帧状态，若操作类型为“IRON\_HAND\_EXECUTE”，则将接收到的RTU执行操作数据帧发送回STM32主控板上。判断执行动作组类型，若为类型为“DATA\_NONE”，则机械手不执行任何动作，若为“DATA\_GREEN”，则控制机械手抓取物料，并将其放入到绿色存储区，若为“DATA\_RED”，则控制机械手抓取物料，并将其放入到红色存储区。**

**动作组执行完毕后，将RTU动作执行完毕数据帧发送回STM32主控板上，清除Receive\_Master \_Data数组中存放的数据，系统恢复到等待接收STM32主控板发送的RTU检测数据帧状态，以此循环，实现机械手对物料进行抓取和放置。**

**2. 中断程序流程图文字描述：**

**UASRT2接收中断：中断接收STM32主控板发送RTU数据帧，并将数据保存到Receive\_Master\_Data组里，同时将USART\_RX\_Over置1（即证明接收完毕STM32主控板发送的数据）。检查接收到的数据是否为停止数据，若数据为停止数据，则机械手停止运作，若数据不为停止数据，则返回主程序；**

**四、自测方式**

1. **设置运行急停按钮：按键按下停止整个系统的运运行（输送线停止运转，机械手停止运作），可以有效避免在系统运行中出现不可挽回的错误；**
2. **通信超时处理：在STM32F103主控板（主机）与OpenMv（从机）、机械手控制板（从机）通信时，当主机发送数据五秒后仍接收不到从机返回的信息，则重新发送操作指令（该做法可以保证较高的通信成功率）。若发送次数为5次时，则证明主机与从机通信超时。当出现通信超时时，LCD液晶显示屏会显示具体某一次通信超时的信息，有利于故障的排查，提高查找bug的效率；**
3. **通信方式使用Modbus协议：在一个主机控制多个从机的情况下，建议使用Modbus协议。相比传输ASCII码，Modbus协议传输的数据帧具有数据量小，信息大，传输方便等特点。Modbus协议中规定了数据帧末端需要添加了数据校验码，每次传输数据完成后都需要校验，提高了数据传输的准确性。在此建议使用Modbus协议中的RTU模式进行通信；**

**参考文献网站：**

[**https://baike.so.com/doc/5414879-5653021.html**](https://baike.so.com/doc/5414879-5653021.html)

[**http://www.360doc.com/content/14/0120/10/7991404\_346584755.shtml**](http://www.360doc.com/content/14/0120/10/7991404_346584755.shtml)

[**https://blog.csdn.net/byxdaz/article/details/77892778**](https://blog.csdn.net/byxdaz/article/details/77892778)

1. **串口发送数据注意事项：通过串口发送数据帧时，数据中间不能出现0，即每8位的数据不能有0x00，否则数据帧发送不完全。在串口发送数据的过程中系统如果检测到0x00停止发送数据，若数据帧的中间数据出现0x00，发送的数据帧会失真。**
2. **数据处理注意事项：****重复使用的变量必须得清楚在何时要将其恢复到初始值，如果两个或两个以上变量都来自同一个文件，且都需要恢复到初始值，那么需要新建一个专门的函数，这样下次调用时可以避免忘记将某个变量恢复初始值，可以避免出现隐藏的bug；**
3. **引脚模拟PWM波输出：利用定时器控制引脚高低电平时间，可以模拟固件库PWM输出，该做法可以提高程序的可移植性，PWM波输出不受固件库的限制；**
4. **判断串口是否接收数据完毕的方法：利用串口总线空闲中断方式可以判断串口是否接收数据完毕，不需要额外添加定时器作数据接收超时检测处理，不仅可以减少了TIM定时器外设的使用，提高系统性能，而且可以增强程序的可读性；**