基于嵌入式系统的运动控制器设计

课程实习报告



**班**

目录

[第一章 绪论 1](#_Toc2758)

[1.1 实习目的 1](#_Toc14215)

[1.2 实习概述 1](#_Toc9256)

[第二章 硬件结构设计 1](#_Toc8901)

[2.1 系统硬件架构 1](#_Toc16865)

[2.2 核心控制器STM32F103 2](#_Toc6757)

[2.3 TJC4832T35\_011串口屏 4](#_Toc6971)

[2.4 二维运动平台 5](#_Toc13765)

[2.5 多路稳压电源模块 6](#_Toc6772)

[第三章 软件框架设计 7](#_Toc27955)

[3.1 界面设计 7](#_Toc22912)

[3.1.1 系统主菜单界面 9](#_Toc29229)

[3.1.2 圆周运动界面 9](#_Toc11735)

[3.1.3 按键控制界面 10](#_Toc20084)

[3.1.4 SIN函数绘制界面 11](#_Toc955)

[3.2 按键交互 11](#_Toc10432)

[3.2.1 按键输入 11](#_Toc3220)

[3.2.2 按键切换界面 13](#_Toc17439)

[3.3 通讯设计 14](#_Toc8538)

[3.3.1 串口屏发送、单片机接收 14](#_Toc14148)

[3.3.2 单片机发送、串口屏接收 16](#_Toc293)

[3.4 算法接口设计 17](#_Toc848)

[3.5 遇到的问题及解决 17](#_Toc5463)

[3.5.1 问题① 17](#_Toc11737)

[3.5.2 问题② 18](#_Toc28671)

[第四章 算法设计及优化 19](#_Toc24638)

[4.1 直线插补算法 19](#_Toc7776)

[4.2 圆弧插补算法 22](#_Toc24201)

[4.3 三角函数插补算法 25](#_Toc30927)

[4.4 遇到的问题及解决 27](#_Toc31341)

[4.4.1 问题① 27](#_Toc8758)

[第五章 成果展示 28](#_Toc2848)

[5.1 直线插补展示 28](#_Toc15619)

[5.2 圆周运动展示 28](#_Toc31013)

[5.3 SIN函数绘制展示 29](#_Toc18515)

[第六章 收获与感想 29](#_Toc9534)

[第七章 意见和建议 30](#_Toc28556)

# 第一章 绪论

## 1.1 实习目的

①通过实践加深学生对控制理论、单片机原理及接口技术、电机拖动、微机接口技术、自动控制系统等课程知识的理解；

②培养学生严肃认真和实事求是的科学作风；

③使学生掌握一般运动控制的操作方法和基本技能；

④加强学生的团队协作能力；

⑤通过实习使学生了解并掌握伺服电机驱动技术、微机接口技术、单片机原理及接口技术，数控轮廓插补原理，计算机高级语言硬件编程；

⑥使学生了解一般运动控制系统常用机械结构，熟练掌握伺服电机控制方法，了解一般交流伺服电机控制方法；

⑦锻炼学生的动手实践能力。

## 1.2 实习概述

本次实习旨在通过单片机系统控制一个平面两轴驱动装置，利用各类插补算法、积分算法实现不同类型图像的绘制。使学生在实习中回顾单片机原理及接口技术、电机拖动、微机接口技术、自动控制原理等课程的知识，了解和掌握一般运动控制的方法和原理。另外，本次实习推荐采用的是AT89C51单片机，即为单片机实习中的自制电路板，与前期的课程起到了良好的衔接作用。

# 硬件结构设计

## 2.1 系统硬件架构

本控制器的核心主控部分采用STM32F103作为处理单元，采用一块3.5寸的串口屏作为人机交互接口以及信息显示界面。12V电池通过自制稳压电源模块为单片机和串口屏供电。

系统的硬件架构图如图2.1所示，实物连接图如图2.2所示。

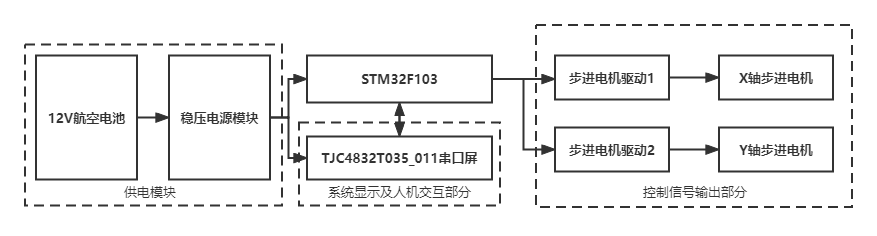


图 2.1 系统硬件架构图

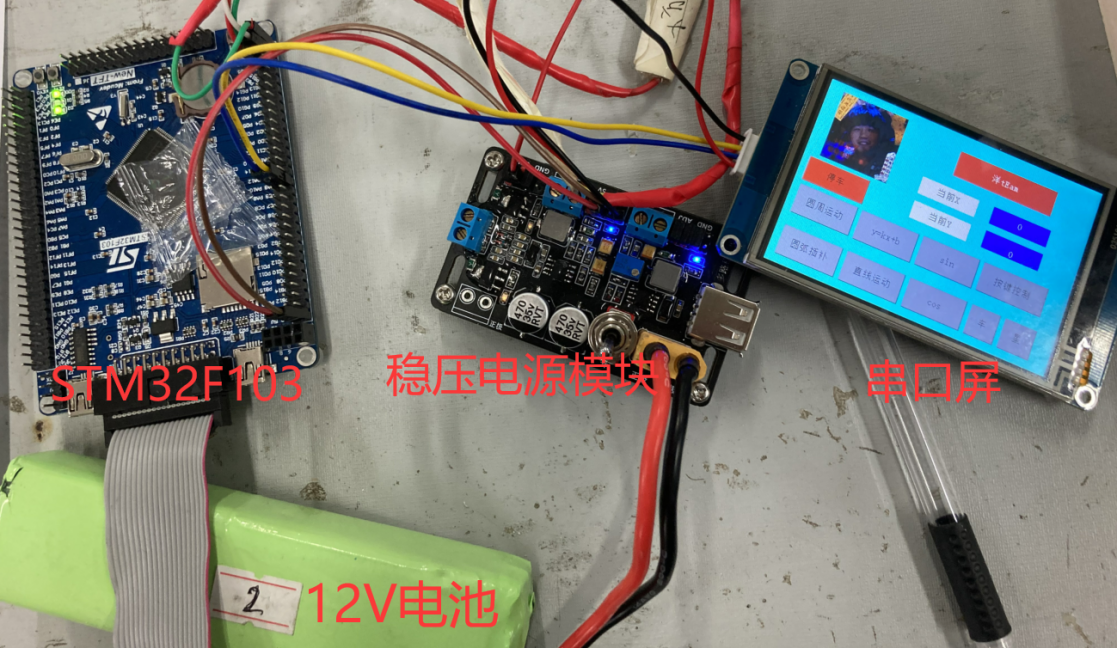


图 2.2 实物连接图

## 2.2 核心控制器STM32F103

本次实习推荐使用的单片机为AT89C51，但考虑到STM32系列的单片机有内存更大的内部ROM以及能更好的扩展外设，故我组在本次实习中采用了STM32F103系列的单片机。STM32F系列属于中低端的32位ARM[微控制器](https://baike.baidu.com/item/%E5%BE%AE%E6%8E%A7%E5%88%B6%E5%99%A8/6688343" \t "https://baike.baidu.com/item/STM32F103/_blank)，该系列芯片是意法半导体（ST）公司出品，其内核是Cortex-M3。该[系列](https://baike.baidu.com/item/%E7%B3%BB%E5%88%97/5863839" \t "https://baike.baidu.com/item/STM32F103/_blank)芯片按片内Flash的大小可分为三大类：小容量（16K和32K）、中容量（64K和128K）、大容量（256K、384K和512K）。芯片[集成定时器](https://baike.baidu.com/item/%E9%9B%86%E6%88%90%E5%AE%9A%E6%97%B6%E5%99%A8/16509212" \t "https://baike.baidu.com/item/STM32F103/_blank)Timer，[CAN](https://baike.baidu.com/item/CAN/4698193" \t "https://baike.baidu.com/item/STM32F103/_blank)，ADC，SPI，[I2C](https://baike.baidu.com/item/I2C" \t "https://baike.baidu.com/item/STM32F103/_blank)，USB，[UART](https://baike.baidu.com/item/UART" \t "https://baike.baidu.com/item/STM32F103/_blank)等多种外设功能。STM32F103实物图见图2.3，原理图见图2.4。

虽说使用STM32可以有更多扩展的可能性，但同时也引入了缺点。在原有基于AT89C51的单片机控制系统中，使用光耦将单片机电路与二维运动平台电路隔离，保证了电路的安全性，可以最大程度避免单片机电路受损；但本次实习中我们采用的STM32单片机控制系统中并没有隔离电路，故存在一定的危险性。



图 2.3 STM32F103实物图

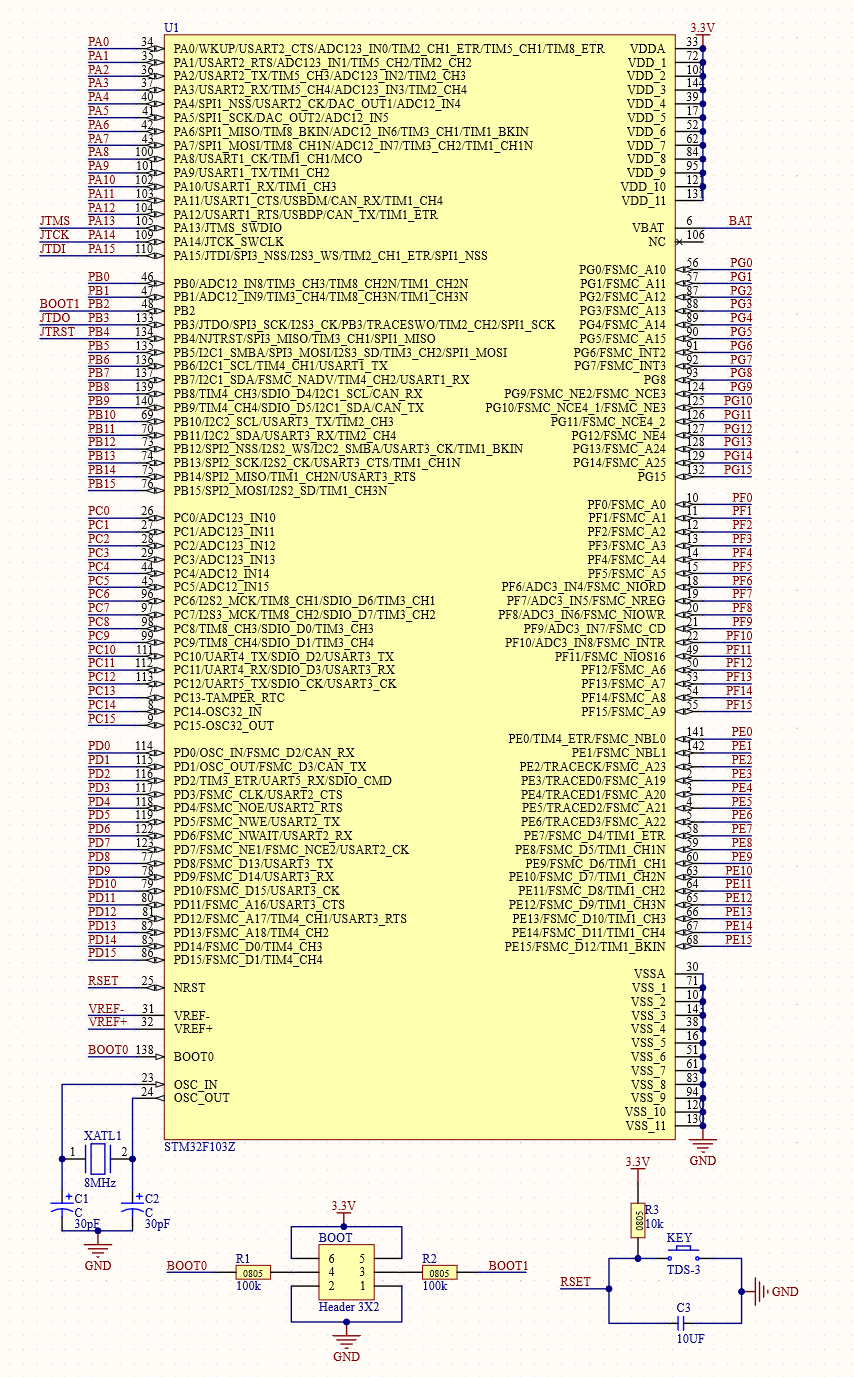


图 2.4 单片机部分原理图

## 2.3 TJC4832T35\_011串口屏

为了使用户有较好的使用交互体验，我们选用了TJC4832T35\_011串口屏，其屏幕为3.5寸LCD电容触控屏，有74\*48的分辨率，足以完成界面显示和用户输入信息的要求，在实际体验中效果良好。

在使用过程中，串口屏的TXD、RXD与单片机的RXD、TXD引脚口连接，使用串口与单片机交换数据，这也是串口屏这一名称的由来（注意串口屏的TXD连接单片机的RXD，串口屏的RXD连接单片机的TXD，不能接反）。

该系统中对二维运动平台的所有控制操作均通过串口屏来完成，串口屏实物图如图2.5所示。



图 2.5 TJC4832T35\_011串口屏实物图

## 2.4 二维运动平台

本次实习，实验室提供了一个平面两轴驱动装置，机械部分采用丝杆螺母的传动方式。驱动和检测电路板上有单片机的控制接口，使用四个GPIO口控制两个伺服电机的运动，十分方便。在其中一个伺服电机上安装了一根金属场杆，金属杆头部设置有一个金属旋进笔架，将签字笔或铅笔固定在笔架上，通过编写程序控制两个伺服电机的运动即可完成目标曲线的绘制。二维运动平台实物图如图2.6所示。

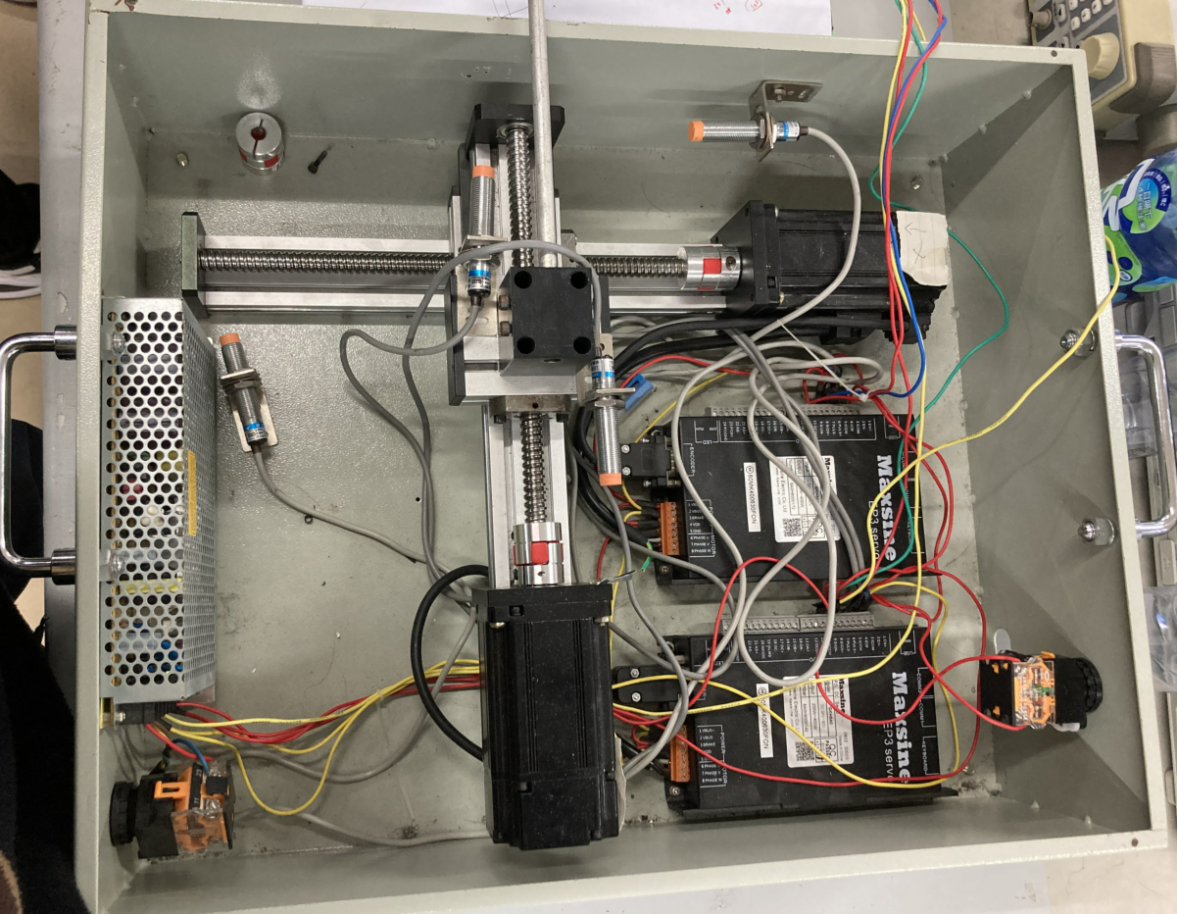


图 2.6 二维运动平台实物图

## 2.5 多路稳压电源模块

为保证系统能够独立运行，我们采用电池供电的方式为单片机及串口屏供电。由于单片机和串口屏的供电电压均为5V，而电池电压为12V，故需要采用降压芯片。本次实习使用的降压模块为自制的降压模块，使用MP2482、AMS1117-3.3两类芯片实现3.3V和5V的输出，在该系统中仅使用5V输出，故在此仅展示5V输出的原理图，5V输出的原理图如图2.7所示。多路稳压电源模块PCB图如图2.8所示。

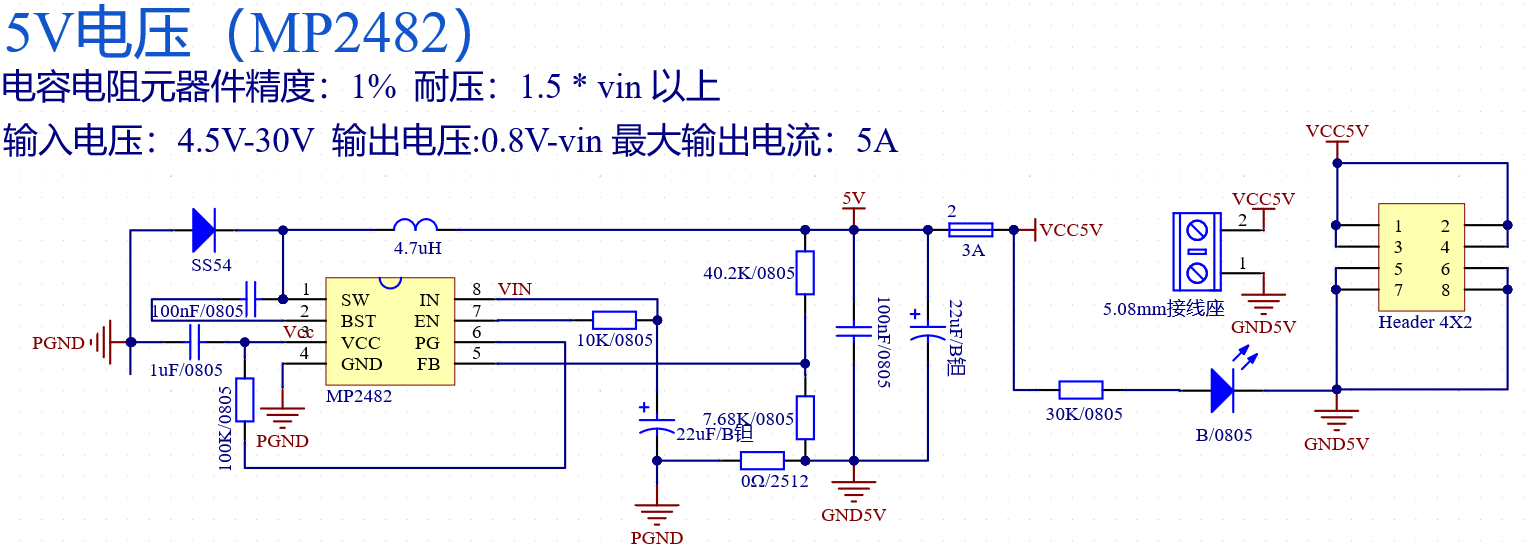


图 2.7 5V输出部分原理图

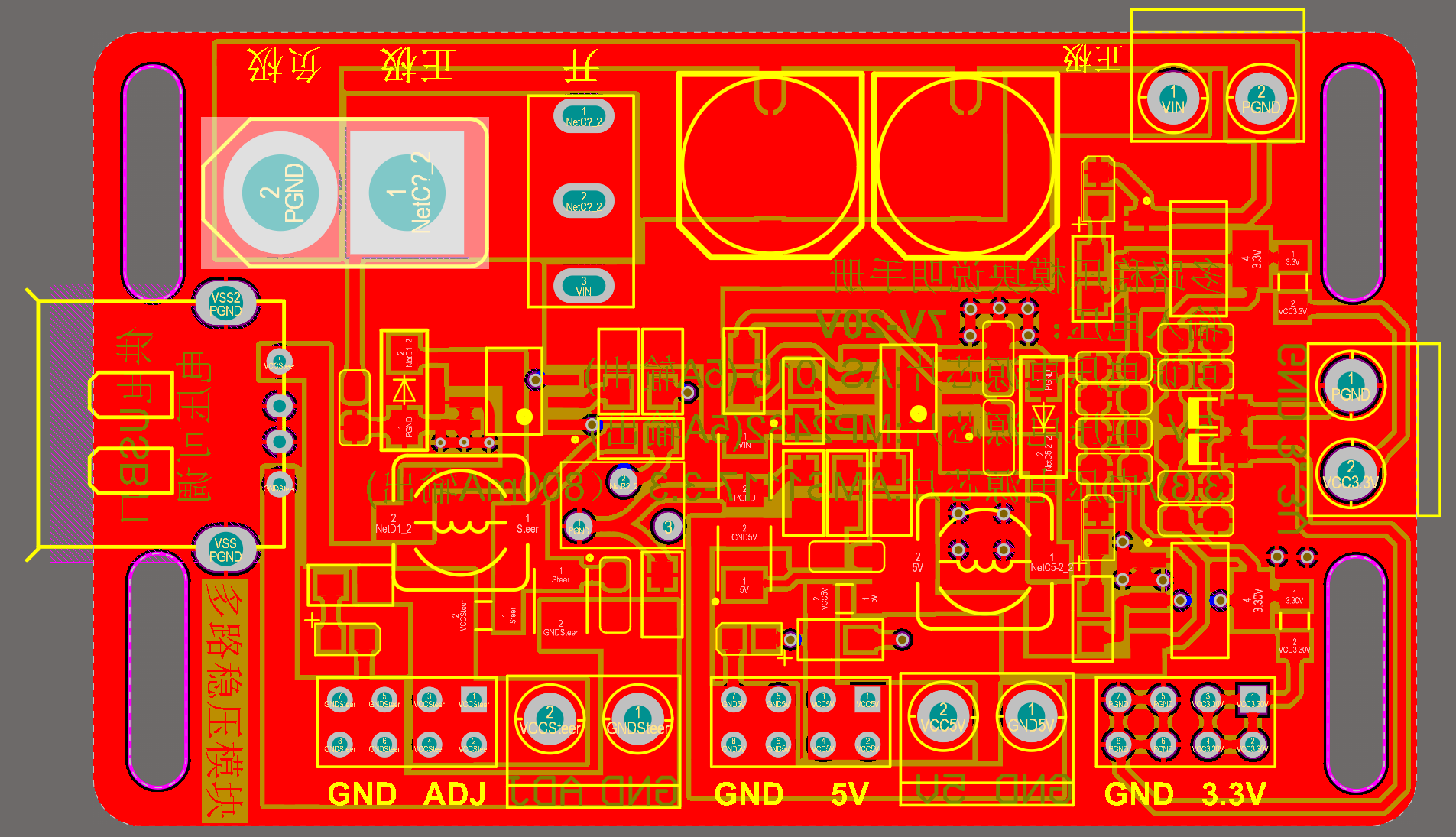


图 2.8 使用AD绘制的PCB图

# 软件框架设计

在本次实习中，我主要负责软件框架设计部分，故将在报告中将重点描述该部分完成的内容。在实习初期，我们打算采用AT89C51控制系统，但由于AT89C51单片机控制系统中的界面交互需要采用LCD屏以及实体按键进行，用户操作体验较差，故转为采用STM32F103最小系统板进行开发并使用串口屏进行人机交互及显示。由于在之前的学习过程中并没有接触过串口屏，所以本次实习也是从零开始学习串口屏的过程。在学习串口屏的过程中遇到了许多的问题和困难，通过询问同学以及查阅网上的资料，最终将所有遇到的问题和困难一一解决并顺利地完成了本次实习的软件框架设计。

## 3.1 界面设计

串口屏的界面设计软件（USART HMI）与Qt有几分相似之处，如图3.1所示。通过图3.1左侧的工具箱可以选择文本、数字、按钮、图片等模块，在该系统中主要使用的是文本模块和按钮模块。每个模块的属性可在属性栏（图3.1右下角）中被修改，例如在按钮模块中bco为按钮颜色，bco2为按钮按下后的颜色，通过修改模块的属性，我们可以DIY自己的界面。

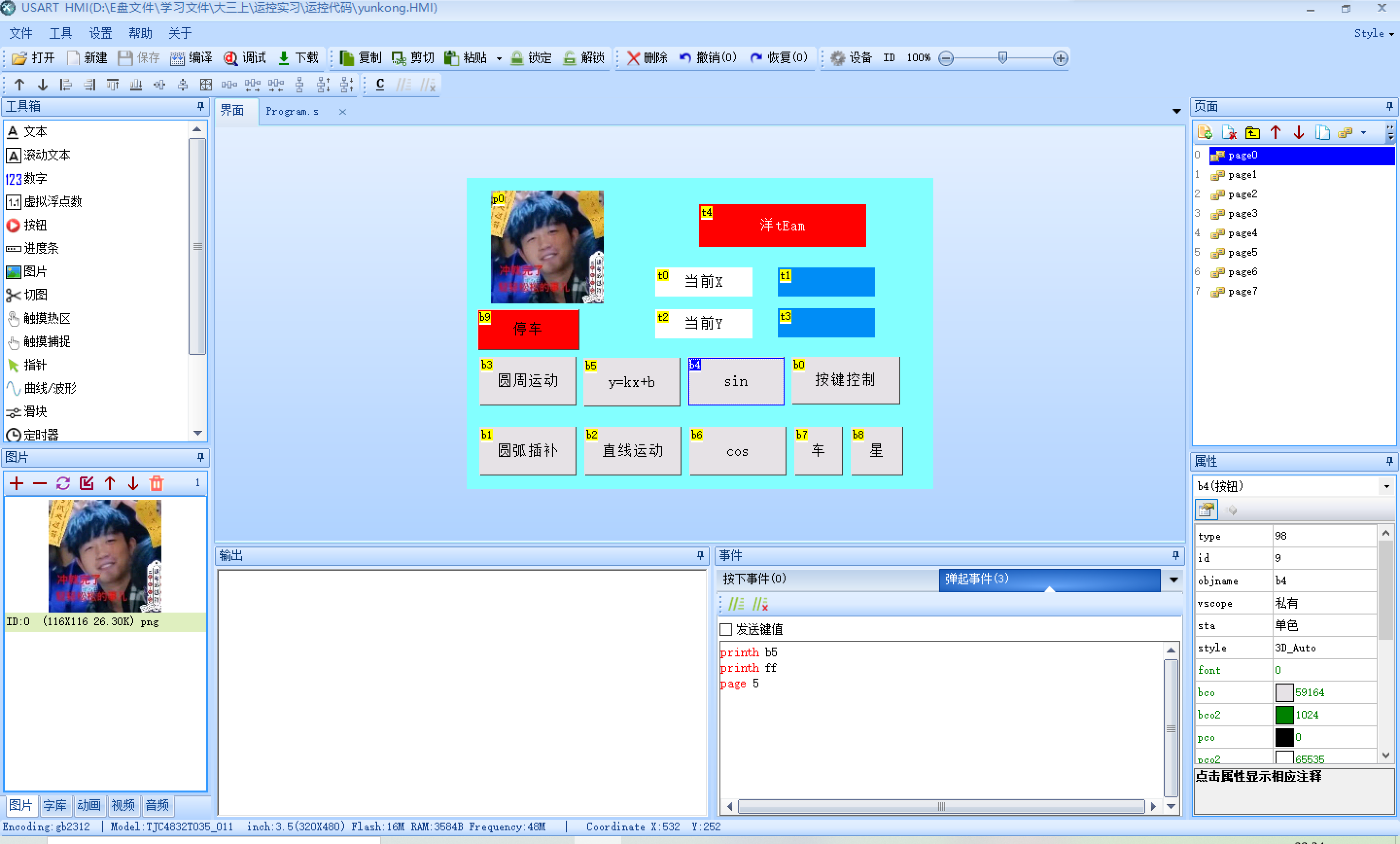


图 3.1 USART HMI软件界面图

该系统中一共有7个界面，每个界面对应一种曲线的绘制，但部分界面有相似之处，由于篇幅有限，在此仅展示4个界面。界面设计框图如图3.2所示。

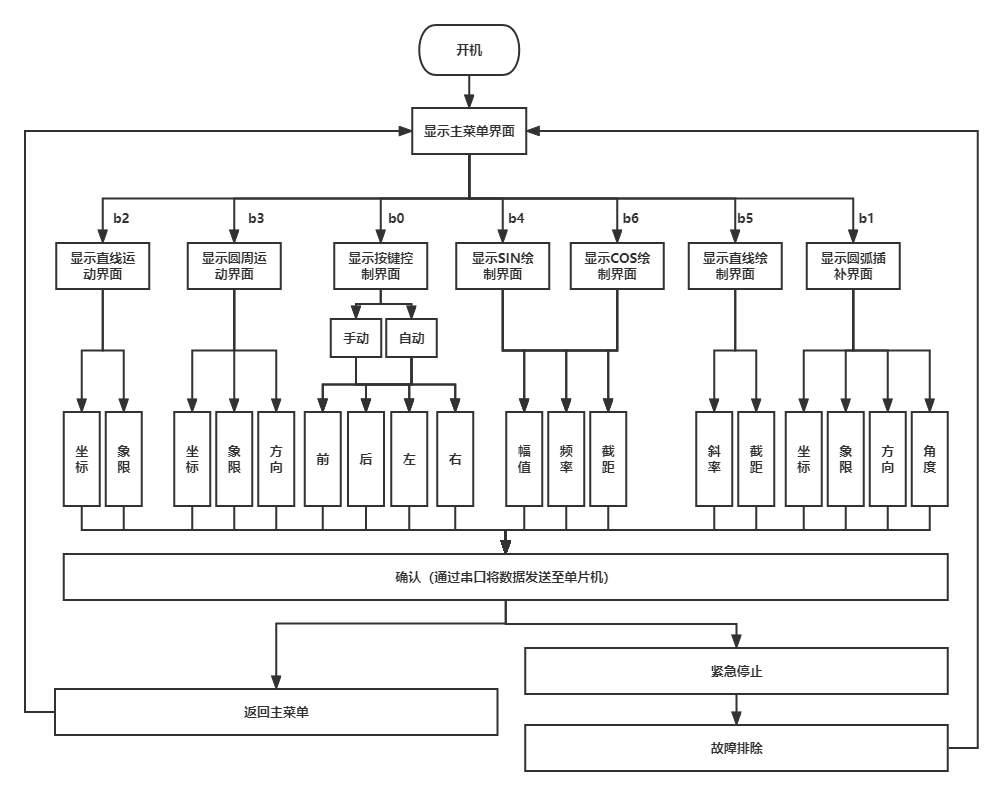


图 3.2 界面设计框图

### 3.1.1 系统主菜单界面

主界面如图3.3所示，其中t1、t3文字模块为当前XY轴坐标显示（具体实现方法将在3.3节通讯设计中描述）；每个灰色的按键用于改变工作模式及切换至其他界面，例如当sin（b4文字框）被按下时，单片机切换到SIN绘制模式并且串口屏界面切换至SIN绘制界面（如图3.6所示）。



图 3.3 主界面

### 3.1.2 圆周运动界面

圆周运动界面如图3.4所示。左侧b0-b11按钮为数字输入；右侧红色文字框分别为方向、象限及坐标输入，其中0代表正方向、1代表反方向；停车按钮用于再紧急状况下暂停曲线的绘制；清除按钮用于清除当前用户输入的所有信息；返回主界面按钮用于返回主界面（图3.3）。

用户可通过点击对应的文字框改变输入的位置，例如用户希望输入象限，需要点击文字框t11后再进行输入。同理若用户希望输入X轴坐标，则需要点击文字框t0后再进行输入。具体的输入实现方法将在3.2节按键输入中介绍。



图 3.4 圆周运动界面

### 3.1.3 按键控制界面

按键控制界面如图3.5所示。文字框t2、t3用于实时显示当前坐标；按钮b0、b2、b3、b5、b7用于控制X、Y两轴伺服电机的前后运行，在空间上即为前后左右的运动；当用户按下手动按钮（b6）后，方向控制按键工作模式为按下运动，松开停止；当用户按下自动按钮（b8）后，方向控制按键工作模式改变为按下运动，直到用户按下暂停按键后才停止，两种工作模式的切换可以提高用户的使用体验，在调试过程中也提供了极大的方便；当用户按下归零按钮（b1）时，单片机将控制伺服电机回到坐标（0，0），即开机时的初始位置。

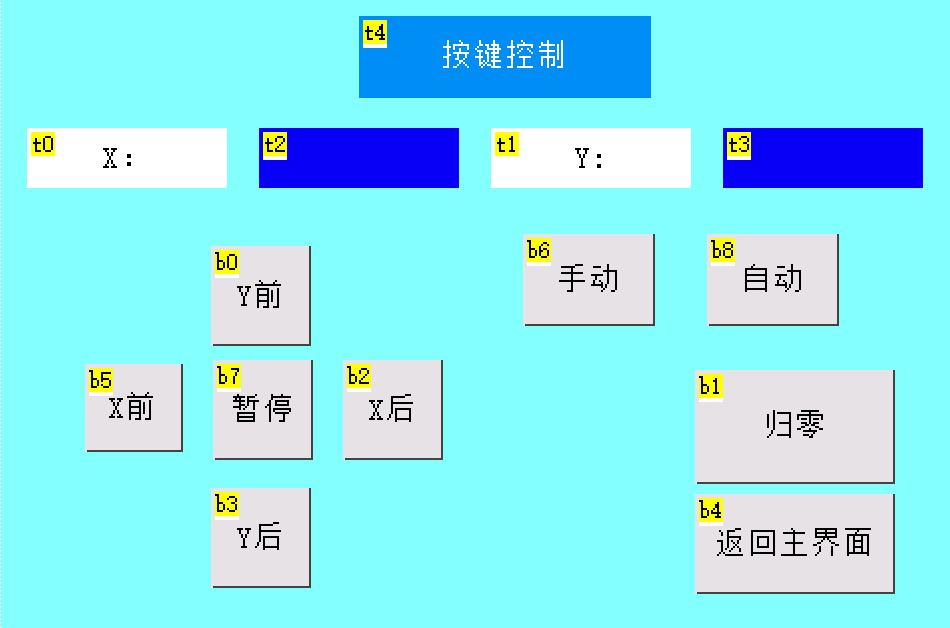


图 3.5 按键控制界面

### 3.1.4 SIN函数绘制界面

SIN函数绘制界面如图3.6所示。界面左侧为数字输入，在上文中已有介绍；界面上方为形如y=Asinωx+b的函数，用户通过点击不同的文字框，可以设置不同幅值、频率和截距的正弦函数，在实际测试中，两个正弦函数的截距之差要大于50才会有明显的区别。由于二维运动平台的两个伺服电机运动距离有限，故不宜将幅值和截距设的过大，以防电机超限并锁死。



图 3.6 SIN函数绘制界面

## 3.2 按键交互

### 3.2.1 按键输入

通过上一节界面设计的介绍我们可以得知，通过数字按钮的输入我们可以设置圆周运动的方向、象限、坐标以及SIN曲线的幅值、频率、截距等，并且点击不同的文字框可以设置不同的变量，以下将介绍按键输入的具体实现。实现流程图如图3.7所示。

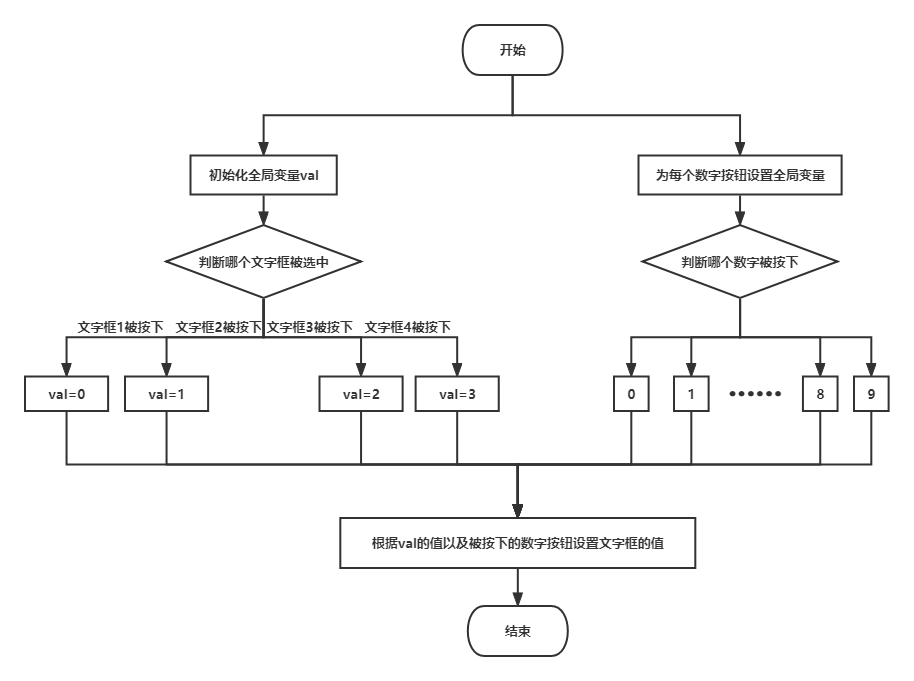


图 3.7 按键输入流程图

根据流程图我们可以得知，按键输入需要两个判断条件，一是判断哪个文字框被选中，二是判断哪个按键被按下。在描述按键输入过程之前，需要了解按钮和文字框的触发机制：①每个文字框自带一个txt变量，可用于存放变量；②在每个数字按钮或文字框按钮被按下时，都会触发一个函数，有点类似于Qt中的信号与槽的槽函数。在“槽函数”（以下均简称槽函数）中，我们可以对全局变量或者按钮中的局部变量进行操作，以实现我们的目的。

接下来是对按键输入流程的具体描述。

首先我们需要设置一个全局变量val，每当用户选中不同的文字框，在对应文字框的“槽函数”中改变全局变量val的值。接下来在每个按键的“槽函数”中加入对全局变量val的判断，判断过后将数字按钮的键值赋给对应文字框的txt变量，这样便实现了按键的输入，达到了人机交互的目的。

举例说明如下：图3.8为数字按钮1对应的槽函数截图，va0.val=0、1、2、3时分别对应X轴坐标（t0）、Y轴坐标（t1）、方向（t9）、象限（t11）。假设当X轴坐标（t0）被按下时，对应的t0对应的槽函数将全局变量va0.val的值置为0（如图3.9所示），故在图3.8数字按钮1的槽函数中执行的操作为“t0.txt=t0.txt+"1"”，即在t0的自带变量t0.txt的尾部插入1。每次按下按钮式执行一次这样的判断，即可实现按键输入。

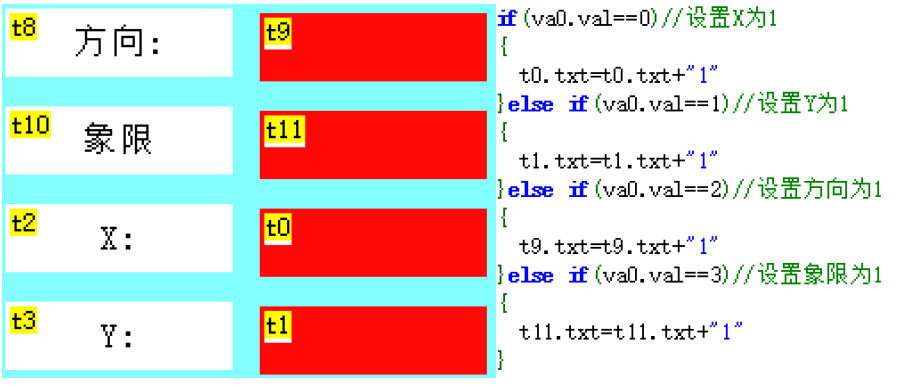


图 3.8 按键输入举例



图 3.9 X轴坐标（t0）被按下时对应槽函数执行的操作

### 3.2.2 按键切换界面

按键切换界面的实现相较于按键输入较为简单，仅需要使用内置函数page即可实现，使用格式为page+页码。

如图3.10所示，代码中的“page 2”即为跳转到界面2，“printh b3”和“printh ff”为通讯协议，用于改变单片机的工作模式，这一部分将在3.3节介绍。



图 3.10 槽函数中切换界面的操作

## 3.3 通讯设计

串口屏作为人机交互的接口，自然要与用户和单片机交流，在前面的章节中我们具体描述了串口屏如何与用户交互，在本章节将具体描述串口屏与单片机间的通信。

### 3.3.1 串口屏发送、单片机接收

在设计界面的过程中，我们为界面中涉及到的每一个操作设置了一个两位十六进制的代号，这个代号也称为帧头，并且设置了一个统一的帧尾ff（如图3.10中的“printh b3”即为帧头，“printh ff”即为帧尾）。由图3.11我们可以看出，当按键被按下，单片机串口1中断被触发，进入中断服务程序。在中断服务程序中我们可以看出，串口屏发送的数据被存在data数组里，首先我们需要判断串口屏所发送数据的第一位是否为帧头，这一点看似不起眼，但其实是必要的。在串口通信的过程中，可能会有很多意想不到的错误，在此加入对帧头的判断，基本可以消除串口通信错误对单片机的影响，这与计算机网络中的TCP协议有几分相似之处。在此我们需要注意，若数组的第一位数据不是帧头需要将t重新置为0，以便后续的判断。

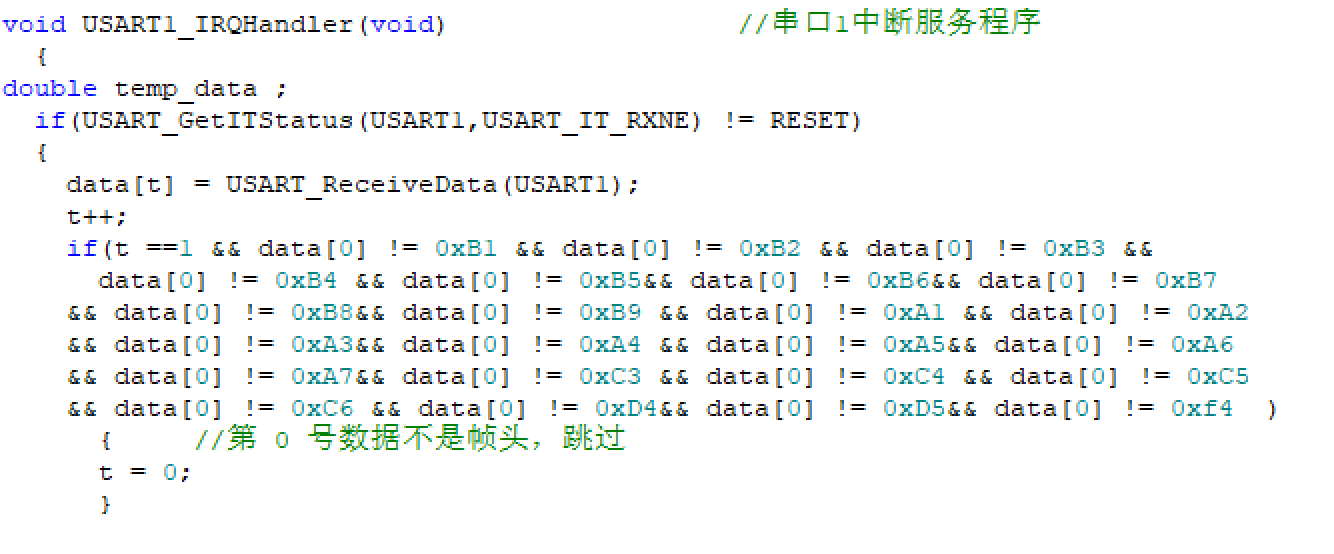


图 3.11 判断帧头的代码

接下来需要根据帧头来设置对应的工作模式，如图3.12所示，当帧头为0xB1时设置工作模式mark的值为1，在主函数中使用switch函数对全局变量mark进行判断并执行相应的操作。整体发送流程如图3.13所示。

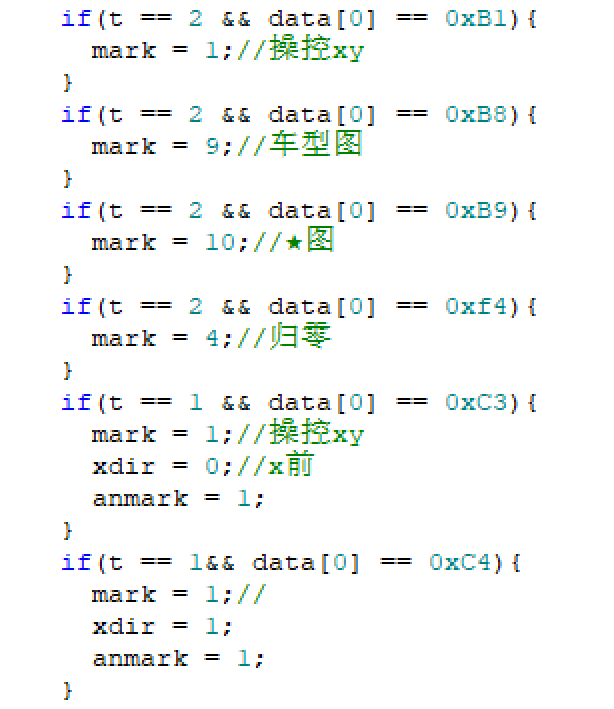


图 3.12 根据帧头设置对应的工作模式

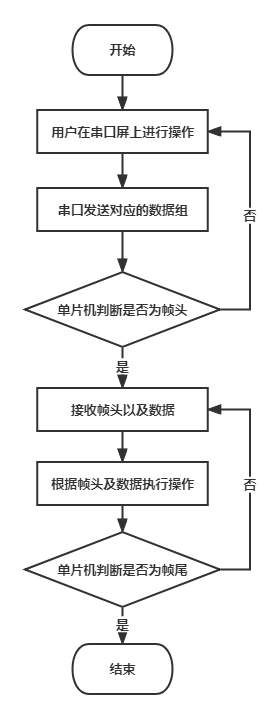


图 3.13 串口屏发送、单片机接收流程图

### 3.3.2 单片机发送、串口屏接收

在3.1节界面设计的描述中，我们介绍了当前坐标实时显示功能，该功能通过单片机向串口屏发送数据实现。本次实习中所采用的运动算法均基于xzz()、xfz()、yzz()、yfz()四个电机驱动函数（将在3.4节详细介绍），故可将坐标的更新放入电机驱动函数中。如图3.13所示，每完成一组脉冲的，便让全局变量current\_x自增一次，反转是则自减一次，每当全局变量current\_x更新后便将current\_x的值发送到串口屏中，则实现了坐标的实时显示。

由图3.14我们可以看出，单片机向串口屏发送数据是通过printf函数拼接并打印字符串实现的其中“page3.t2.txt”表示第三页中t2文字框自带变量的名称；/xff为一个十六进制数，通信协议中规定，当单片机连续发送三个/xff表示数据发送结束。至此我们实现了单片机向串口屏发送数据的功能。

具体实现流程图如图3.15所示。

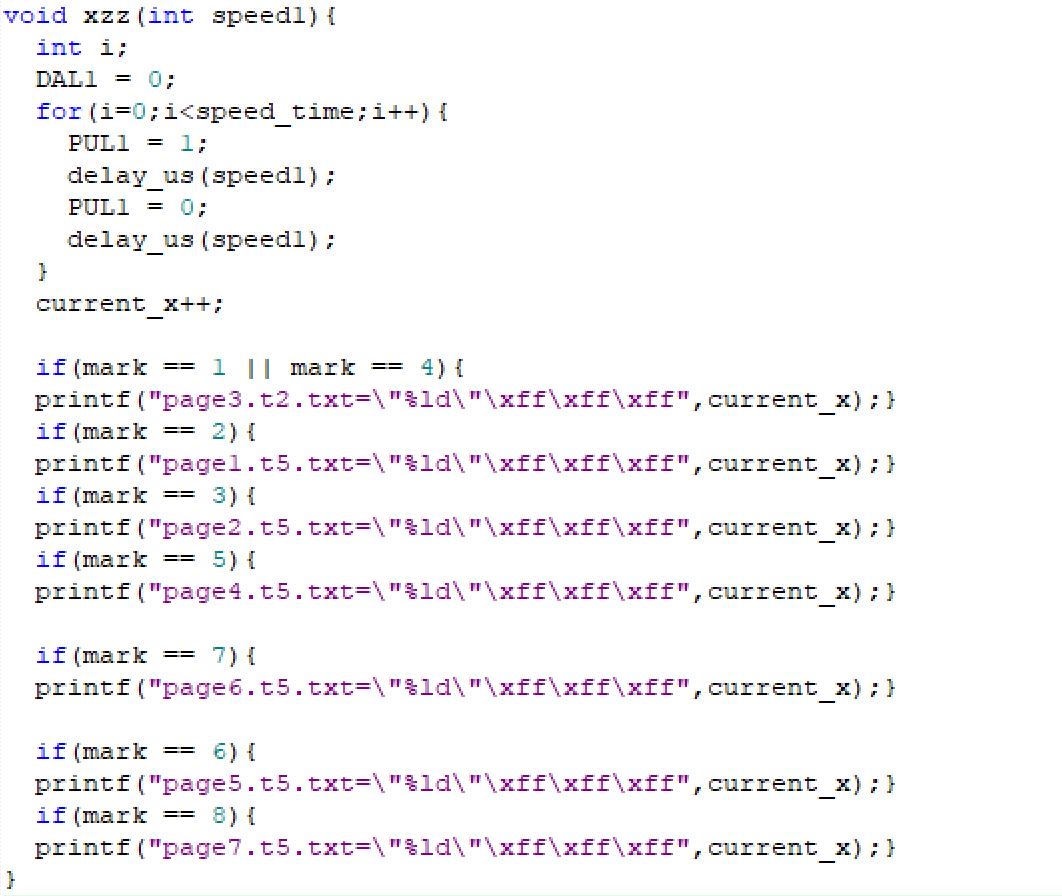


图 3.14 X轴正转函数

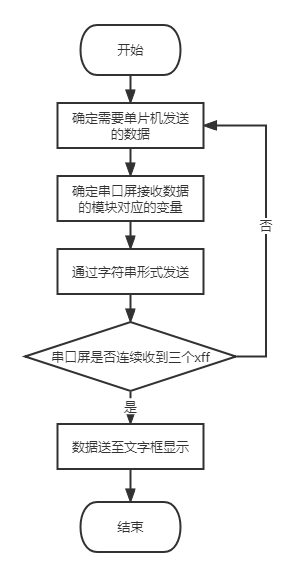


图 3.15 单片机发送、串口屏接收实现流程图

## 3.4 算法接口设计

在完成软件框架的设计后，我们需要考虑如何将现有的软件框架与算法相匹配，那么我们需要设计软件与算法的接口。在本次实习中，我们所使用的运动控制算法的本质都是控制X轴、Y轴两个伺服电机的正转与反转，让两个伺服电机按照特定规律正转、反转即可得到预置的曲线。那么所谓的算法接口设计，本质上就是为算法提供控制电机正转反转的函数。如图3.14所示，X轴正转函数在for循环中进行了引脚高低电频的变换，实现脉冲输出。通过函数的形参speed1可以控制发出脉冲的次数，即步长。同理还有X轴反转、Y轴正转、Y轴反转三个函数，这三个函数的结构与X轴正转函数相似，故在此不再做详细介绍。

## 3.5 遇到的问题及解决

### 3.5.1 问题①

**问题描述：**当一个界面需要向单片机发送多组数据时（例如幅值、截距、频率），单片机应该如何接收？

**解决方法：**如图3.16，左侧为单片机接收部分的代码，右侧为串口屏发送数据的代码。在单片机部分代码的if语句中我们可以看到，if语句对两个变量进行了判断，与图3.12中的相似，第二个数据为帧头但第一个数据t的值不同。由于单片机一次只能读取一组由串口屏发出的数据（即两个十六进制数），全局变量t用于标记读取数据的次数，当识别到帧尾时需要将t的值清零。合理选择t的值是单片机能够正常接收多组数据的关键。

由串口屏部分代码可知，串口屏一共发送了五组数据，第一组为帧头，第二至四组分别为SIN函数的幅值、频率、截距，最后一位为帧尾。故需要在单片机读取到帧尾（t=4）或读取完所有数据后（t=5），再从data数组中将我们需要的数据提取出来，即图3.16中if语句中t可以等于4或5。

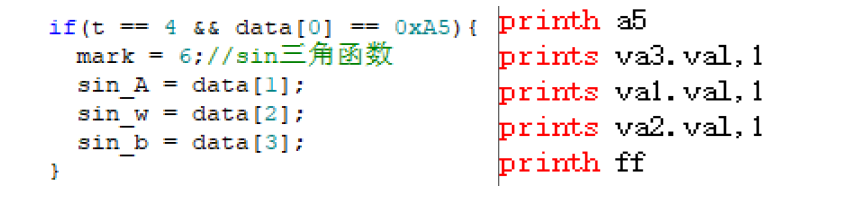


图 3.16 单片机接收部分代码（左）和串口屏发送数据代码（右）

### 3.5.2 问题②

**问题描述：**按键控制界面中的手动控制如何实现？

**解决方法：**由图3.17可知按钮的时间属性分为按下事件和弹起时间，若想实现手动控制（即按下运动，松开停止），则需要再按下事件中写入使电机转动的代码，在弹起事件中写入使电机停转的代码。



图 3.17 按钮的事件属性

# 算法设计及优化

## 4.1 直线插补算法

加工如图4.1所示的平面斜线AB，以斜线起点A的坐标为x0，y0，斜线AB的终点坐标为(xe，ye)，则此直线方程为：



取判别函数：



用逐点比较法加工时，每一次只在一个坐标方向给出一个脉冲，使运动件在该坐标方向上进给一步，因此刀具的运动轨迹是折线，而不是斜线AB。折线拐点M与斜线AB之间的位置关系有如下三种情况：

1)M点在AB线的上方．判别函数F>0;

2)M点在AB线上，F=0

3)M点在AB线的下方，F<0

为控制方便，将F>0和F=0两种情况作为F>=0一种方式判别。当判别函数F>=0时，刀具一定处在AB线的上方，或在AB线上。例如图4.1中的M1(X1,Y1)点，这时刀具只有沿+X方向进给才更接近AB线，因此，根据这个判别结果，计算机在x铀方向输出一个脉冲，使刀具在+x方向前进一个脉冲当量的距离。到达M2点。刀具在M2点的判别式为：



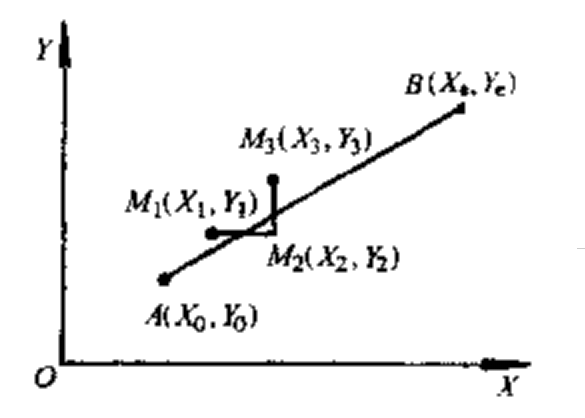


图 4.1 直线插补

F判别式中的(Xi,Yi)(i＝l,n)值应是脉冲数量值．而不是AB线的长度值。若F2<0则M2点判定处在AB线 的下方,应向y方向送出一个脉冲，使刀具向+y方向移动一步，到达从点M3,此时判别式：



由上述的M1,M2点的判别式可知：

在第一象限插补时，若沿x方向走一步即Xi+1=Xi+1则Fi+1＝Fi-(Ye-Y0)，若沿Y方向走一步，即Yi+1=Yi+1则Fi+1＝Fi+(Xe-X0)。偏差值Fi+1的计算只用到前一点的偏差值Fi和斜线的长度在坐标方向的投影(Xe-X0)，(Ye-Y0)。

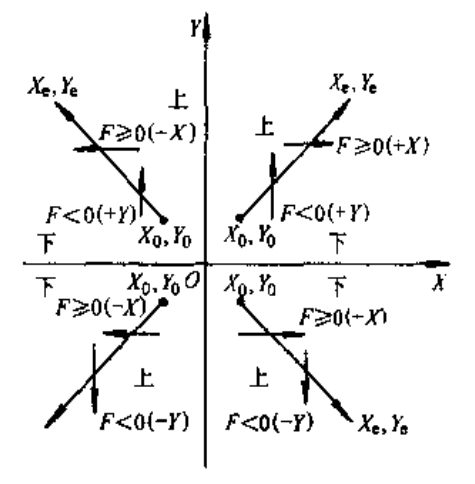


图 4.2 直线插补象限划分

图4.2是象限的划分规则，根据对线段加工方向的不同来判别它所处的象限，如表3-1对于四个象限可共用如下列的判别式：

向Xe方向走一步：



向Ye方向走一步：



上述两式(Xe-X0)，(Ye—Y0)都用绝对值，不考虑符号，但(Xe-X0)，(Ye-Y0)是有符号的，它影响刀具相对工件移动方向。对刀具相对工件的移动方向的控制可根据线段所处的象限来决定。若线段处在第三象限，在x或y方向输出脉冲时，使伺服电动机反转即可。而判别式和脉冲分配方式与第一象限相同；若线段处在第二象限可使X向电动机反转而y向电动机正转，第四象限使X向电动机反转，Y向电动机正转，如表4.1。

表 4.1 象限判别和电机方向

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 方向 | 第一象限 | 第二象限 | 第三象限 | 第四象限 |
| Xe-X0 | >0 | <0 | <0 | >0 |
| Ye-Y0 | >0 | >0 | <0 | <0 |
| X向电机 | 正 | 反 | 反 | 正 |
| Y向电机 | 正 | 正 | 反 | 反 |

直线插补的终点判断可采用如下的两种方法之一。

①每走一步都要计算|(Xi-X0)|，|(Yi-Y0)|的数值，并判断|(Xi-X0)|>= |(Xe-X0)|且|(Yi-Y0)|>= |(Ye-Y0)|是否成立，若成立则插补结束，否则继续。

②把被加工线段的Xe-X0，Ye-Y0的长度单位换算成脉冲数值(若长度单位为mm，则把上述的坐标增量值除以脉冲当量),然后求出各坐标方向所需的脉冲数总和n：即n＝|(Xe-X0)+(Ye-Y0)|，计算机无论向哪个方向输出一个脉冲都作n-1计算，直到n=0为止。用第二种方法进行终点判别的。图4.3直线插补流程图。

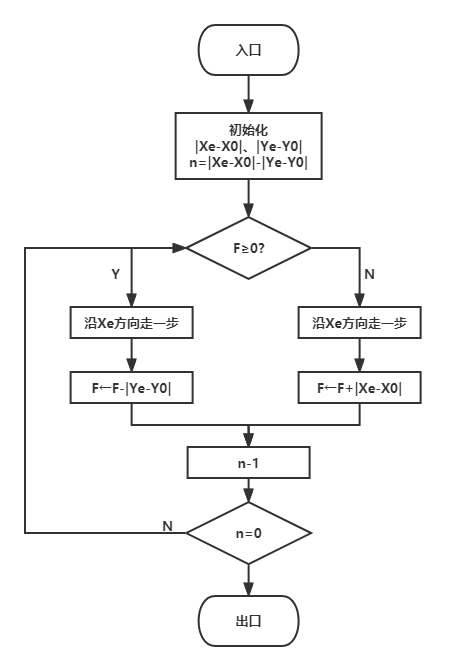


图 4.3 直线插补流程图

## 4.2 圆弧插补算法

在图4.4中AB是被加工圆弧。加工程序中给出的已知条件通常是A点B点的坐标值，圆心O’点相对圆弧起点A的增量坐标值。由图可知：圆心O’点相对A点的增量坐标值为(-Io，-Jo)。改变符号后就成为A点相对O’点的增量值Io，Jo。由此可求出圆弧的半径值R：R2=Io2+Jo2在以圆心O’点为原点的I、J坐标系中，圆的方程可表示为：I2+J2=R2设刀具已位于M1点，则Mi点对圆弧AB的位置有三种情况：

①Mi在圆弧外侧，则0’Mi>R，Ii2+Ji2>R2

②Mi在圆弧上，则0’Mi=R，Ii2+Ji2=R2

③Mi在圆弧内侧，0’Mi<R，Ii2+Ji2<R2

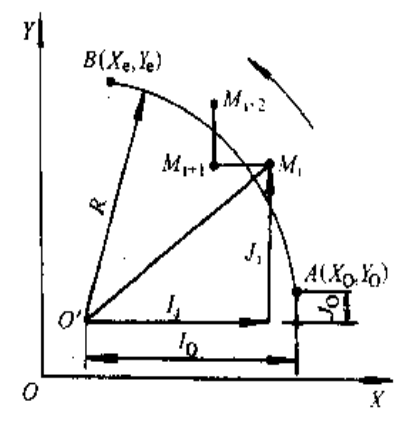


图 4.4 圆弧插补

Mi点的判别式可写成如下形式：



若Fi>0从点在圆的外侧。这时计算机应送给X方向伺服电机一个脉冲，对于图1—4中的AB且它是在第一象限逆时针方向加工，应使电机反向转一步，刀具相对工件沿—X方向走一个脉冲当量的距离，到达Mi+1点。Mi+1点对圆心O’点的坐标为Ii+1=Ii-1,Ji+1=Ji判别式为：



若Fi+1<0，则说明Mi+1点在圆内．这时计算机应送给Y向伺服电动机一个脉冲，使正转刀具沿+ y方向前进一个脉冲当量的距离，到达点Mi+2, Mi+2点的位置是Ii+2= Ii+1,Ji+2= Ji+1+1判别式为：



若Fi+2>0，则说明Mi+2点在圆外．这时计算机应送给-X向步进,Fi+2<0应送给+Y向步进。

上述为在第一象限逆时针加工圆弧(称逆圆弧)的情况，而在第一象陨顺时针加工圆弧(顺圆弧)和第二、三、四象限加工顺圆弧和逆圆弧时，判别式都不相同。带符号运算时，无论在哪个象限工作，顺圆弧或逆圆弧，归纳起来有如下四种情：

1.+X方向走一步：Ii+1=Ii+1、Fi+1=Fi+2Ii+1

2.-X方向走一步：Ii+1=Ii-1、Fi+1=Fi-2Ii+1

3.+Y方向走一步：Ji+1=Ji+1、Fi+1=Fi+2Ji+1

4.-Y方向走一步：Ji+1=Ji-1、Fi+1=Fi-2Ji+1

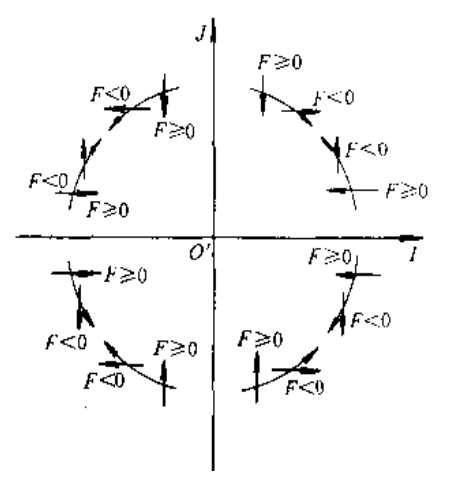


图 4.5 四个象限进给方向

图4.5是在四个象限内顺、逆圆加工时，判别式符号和进给方向关系。象限是以被加工圆弧圆心为原点的坐标系划分的。若编写零件加工程序时所用的坐标系不以圆心为原点，则象限的划分就不能用这个坐标系。以圆心为原点的坐标系是根据加工程序给出的已知条件自动建立的，圆弧加工完毕、坐标系自动取消。例如：在肝坐标平面内，给出的圆弧起点相对圆心的增量坐标值Io，Jo在插补计算过程中不断算出的Ii,Ji值，都是以圆心为原点坐标系中的坐标值。根据Ii,Ji的符号来判别圆弧所在的象限，如表4.2。过象限时的标志是Ii,Ji中的一个是零。

根据图4.5给出的判别式与加工符号关系，和表4.2结出的，Ii,Ji的符号变化规律，利用正、负方向的判别式可对四个象限的顺、逆圆弧加工编写出插补程序。

表 4.2 象限判断和电机转向

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 第一象限 | 第二象限 | 第三象限 | 第四象限 |
| Ii的符号 | | + | - | - | + |
| Ji的符号 | | + | + | - | - |
| X向电机 | 顺圆 | + | + | - | - |
| 逆圆 | - | - | + | + |
| Y向电机 | 顺圆 | - | + | + | - |
| 逆圆 | + | - | - | + |

逐点比较法圆弧插补的终点，可根据圆弧终点对圆心坐标值判定。例如：在XY平面内，圆弧终点相对圆心的坐标为Ie,Je，在插补运算的过程中，Ii,Ji的值总是不断的作+1或-l变化，当满足Je-Ji=0且Ie-Ii=0条件时，就到达了终点。

由于Ie,Je,Ii,Ji是带符号的值，因此，不管终点在哪个象限都适用。由于零件的加工程序给出的值总是以毫米为单位，因此，必须在初始化时把以毫米为单位的长度校脉冲当量除后取整，化为脉冲的数字量。

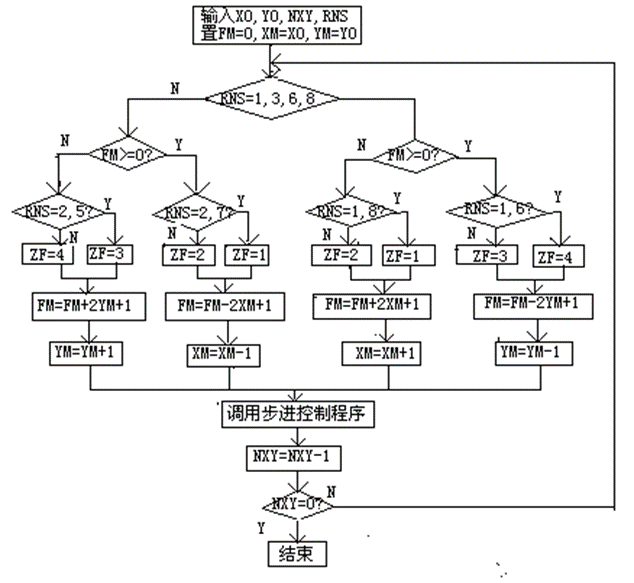


图 4.6 圆弧插补流程图

## 4.3 三角函数插补算法

三角函数的绘制基于插补思想，判断当前点与三角函数理论坐标的位置关系，确定X轴与Y轴移动的距离和方向。与圆弧插补类似，采用逐点比较发，每次在一个方向上前进一步后，判断一次当前点与三角函数理论坐标点的位置关系。理论点与实际点的位置关系有三种情况（以SIN函数为例）：

①当前点在SIN曲线的上方，F>0;

②当前点在SIN曲线上，F=0；

③当前点在SIN曲线的下方，F<0.

与直线插补相同，我们将①、②两种情况合并为一种方式判别。当F≥0时，笔尖在SIN函数的上方、或再SIN函数上（如图4.7中N1点），这时只有沿X轴正方向运动才能更接近SIN曲线，根据这个判别结果，单片机需要给出一个X轴正方向运动的脉冲。给出X轴正转脉冲后，到达N2点，此时F<0，判定N2位于SIN曲线的下方故此时应该在Y轴方向给出一个正脉冲，使得笔尖向+Y方向移动一步到达N3点。

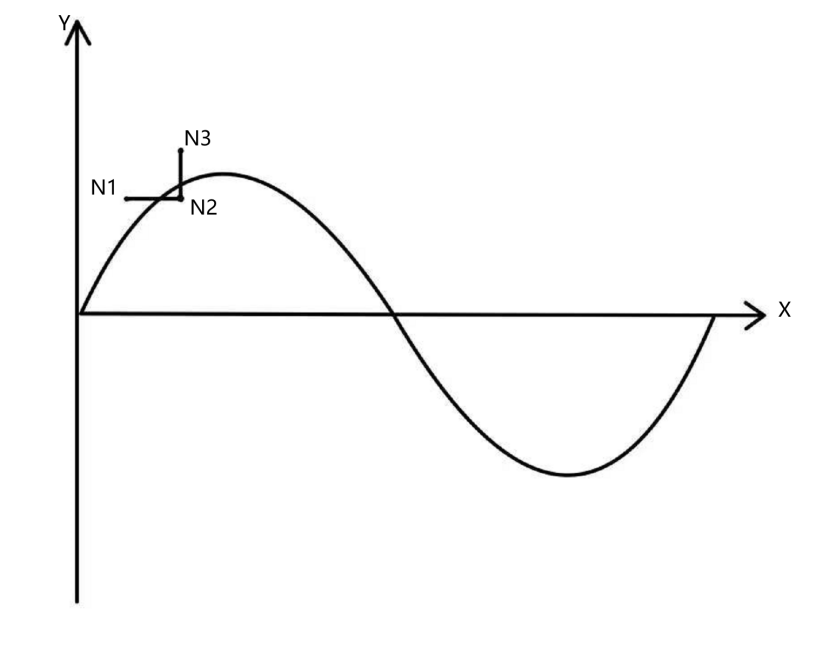


图 4.7 SIN曲线插补

在三角函数的绘制过程中，除了判断当前点与目标点的位置关系，还需要判断当前点所处在的象限。我们规定，当斜率斜率为正时（即SIN函数处于上升状态时）为第一象限；当斜率为负是（即SIN函数处于下降状态时）为第二象限。因此SIN函数的象限判别式为：



两个象限对笔尖移动的判别不同，在第二象限时，对笔尖运动的判别与第一象限的相反，见表4.3。

表 4.3 象限判别和电机方向

|  |  |  |  |
| --- | --- | --- | --- |
| F | 电机 | 第一象限 | 第二象限 |
| F≥0 | X轴电机 | +X | 0 |
| Y轴电机 | 0 | -Y |
| F<0 | X轴电机 | 0 | -X |
| Y轴电机 | +Y | 0 |

三角函数绘制的停止位置有两种，一种是绘制一个周期后自动停止，另一种是用户根据需求自动停止。

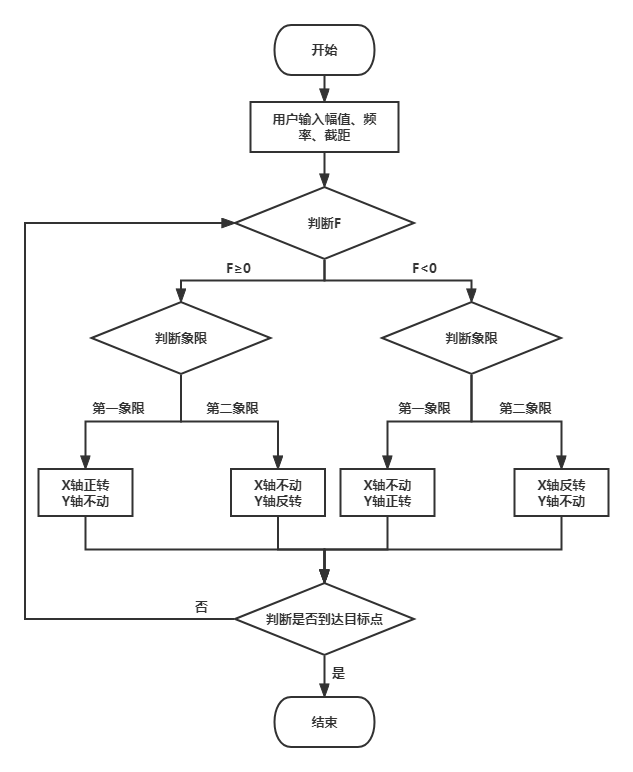


图 4.8 SIN图像绘制流程图

## 4.4 遇到的问题及解决

由于在实习过程中，我主要负责软件框架的搭建，算法部分代码的实现主要由叶洋同学完成，在算法部分我仅参与算法思想的讨论以及部分问题的解决，故在此仅对我参与的部分进行描述。

### 4.4.1 问题①

**问题描述：**在调用绘制六芒星和小车的函数时，电机移动速度极快。

**解决方法：**电机转速过快首先需要检查单片机给出脉冲的频率，但六芒星函数和小车函数中使用的运动控制函数与其他工作模式下的运动函数相同，不存在频率的问题。在解决的过程中，我们发现六芒星函数和小车函数分别使用的是主函数中switch语句的case 10和case 11，当我们将这两个函数移动到case 9及一下的判断语句中，电机的移动速度恢复正常。正常来说，case最高可以达到255，但是现在仅仅是到了10就出现了错误。最终我们采用if语句代替switch语句将该问题解决。

# 成果展示

## 5.1 直线插补展示



图 5.1 直线插补成果图

## 5.2 圆周运动展示

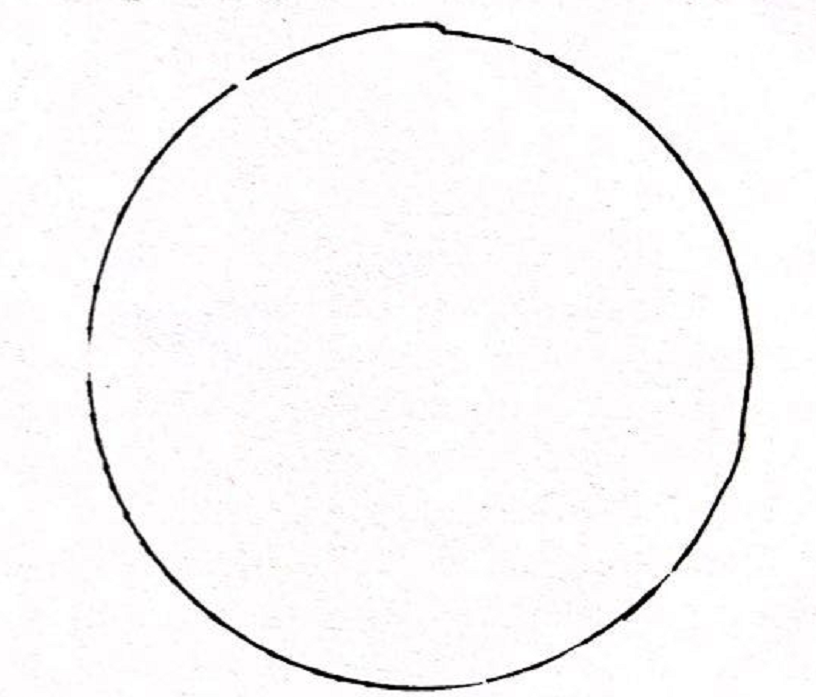


图 5.2 圆周运动成果图

## 5.3 SIN函数绘制展示

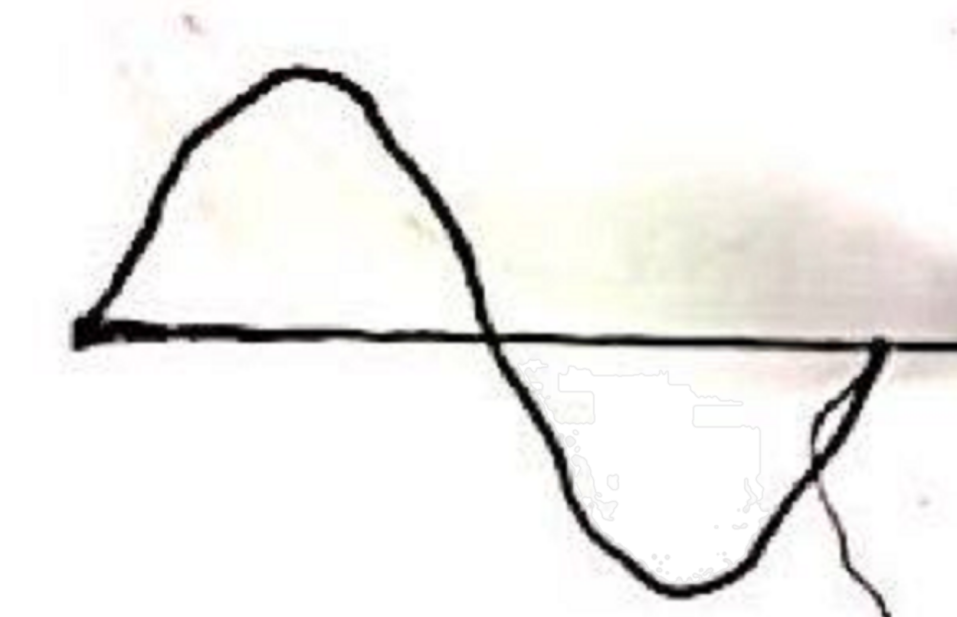


图 5.3 SIN函数绘制成果图

由于实习过程中没有注意保存图片资料，SIN函数绘制成果图（图5.3）并非是最终成果。本套系统还可绘制六芒星和小车，但由于没有拍照记录故在此无法展示。

由图5.1、图5.2、图5.3可以看出本套系统基本能完成直线、圆弧、三角函数这三类曲线的绘制。

# 收获与感想

本次实习主要完成了二维运动平台的嵌入式开发，在开发过程中我们完成了硬件调试，软件框架搭建，直线插补、圆弧插补、三角函数插补等算法的实现，最终使系统能基本满足运动要求。

实习的过程并不是一帆风顺的，在实习初期，我们决定采用51单片机进行实习，但在搭建软件框架的过程中，我们发现实现按键输入、LCD显示的过程十分繁琐。对于LCD屏幕，需要对每一个需要显示的字符取模；对于按键输入，需要设置数组，并且需要转换数据类型；并且在使用51单片机编程的过程中，进场出现超出内存范围的问题，这令我们十分头疼。偶然间听到做电赛的同学说，串口屏写起来会比LCD简单，而且STM32的性能比51强，我们产生了更换单片机的想法，但这意味着之前完成的部分全部作废。

经过深思熟虑，我们决定使用STM32进行开发。但这对于我来说是十分具有挑战的，我在此之前并没有接触过STM32，更没有学习过如何使用串口屏。不过这也是一个学习新知识的好机会。虽然做电赛的同学认为串口屏的编程较为简单，但当我真正上手做的时候才发现，由于我对串口屏的了解几乎为零，所以串口屏的编程对于我来说，难度甚至超越了51单片机系统上的LCD屏。

我认为既然已经做了更换单片机的决定，那就不能轻易的放弃。通过在网上查阅串口屏的相关资料、在b站上观看相关的教学视频以及向做电赛的同学们请教，我渐渐摸清了串口屏的工作原理以及串口屏与单片机的工作方式，最终也基本按照自己的要求完成了软件框架的搭建。

虽然在前期我花费了大量的时间在学习串口屏的编程和实现人机交互的功能上，较少参与算法部分的实现。但我参与了的后期算法调试及优化，在这过程中我基本掌握了各类算法的原理。

除了实现基本的功能，我们还构想过使用matlab上位机实现任意曲线绘制的功能，以及采用openmv模块实现寻线移动的功能。但由于期末考试临近以及各类实习安排较多，没有更多的时间实现该部分功能。

总的来说，在本次运控实习的过程中，我从零开始学习了STM32和串口屏的使用，实现了两个“从无到有”，可以说是收获颇丰。虽然说最终实现的功能不一定有其他同学优秀，或者无法和上届的学长相比，但在实习的过程中我们确实尽了自己最大的努力将实习做到最好。

# 意见和建议

由于我们在大二学年的暑假已经进行过单片机实习，故本次运控实习可以安排在学期初进行。第一，是学期初时，刚结束单片机实习的大家对51单片机系统的使用更加熟悉，不需要花费额外的时间重新回顾单片机的相关知识；第二，是学期初课程安排较少，同学们可以充分利用空闲时间来更好地完成运控实习，在时间充裕的条件下，同学们可以玩更多的“花样”，创造出更多的可能性；第三，是运控实习与课程同步进行，可以激发同学们对理论课程的学习兴趣，二者相辅相成、相互促进。