**\*根据脉冲计算电机的转速**

步进电机常见的步进角是1.8°或者3.6°。意思是说每个脉冲步进电机转动这么多角度。相应的分别需要200个脉冲和100个脉冲转1圈。

分频：2分频时每输入两个脉冲步进电机才会转一步，8分频和16分频同理。

通过调整代码中高低电平的延迟时间可以调整输入脉冲的频率，进而控制电机的转速。

\*脉冲数和步进电机的位置坐标之间的关系

步进电机处于什么位置是通过相对于零位置（上电瞬间电机所在的位置）旋转到当前位置需要的脉冲数确定的（和步进电机的步进角以及分频的模式有关）

**\*AccelStepper的库文件**

由于arduino自带的stepper库只能同时控制一个步进电机，因此需要accelstepper库实现对多个步进电机的同时控制。

**\*关于学长发的arduino\_test代码的解读**（黑色的是源代码，红色而是解读）

#include <AccelStepper.h> 调用AccelStepper.h库

#include <MultiStepper.h>

//分割字符串\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

#define numdata\_length 5

String comdata = "";

long numdata[numdata\_length] = {0};

int flag = 0;

//变量声明\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

下面定义了五个关节上步进电机的引脚，说明对于MEGA2560来说是通过下面定义的引脚输出相应的每个步进电机的方向，使能和脉冲数。所以我推测这体现的就是RAMPS1.4和Arduino之间的对应关系。例如下面的Joint1：

// Joint 1

#define E1\_STEP\_PIN 36

#define E1\_DIR\_PIN 34

#define E1\_ENABLE\_PIN 30

Arduino上的引脚36对应着RAMPS1.4上E1的step；Arduino上的引脚34对应着RAMPS1.4上E1的Dir；Arduino上的引脚30对应着RAMPS1.4上E1的EN；下面同理

// Joint 2

#define Z\_STEP\_PIN 46

#define Z\_DIR\_PIN 48

#define Z\_ENABLE\_PIN 62

#define Z\_MIN\_PIN 18

#define Z\_MAX\_PIN 19

// Joint 3

#define Y\_STEP\_PIN 60

#define Y\_DIR\_PIN 61

#define Y\_ENABLE\_PIN 56

#define Y\_MIN\_PIN 14

#define Y\_MAX\_PIN 15

// Joint 4

#define X\_STEP\_PIN 54

#define X\_DIR\_PIN 55

#define X\_ENABLE\_PIN 38

// Joint 5

#define E0\_STEP\_PIN 26

#define E0\_DIR\_PIN 28

#define E0\_ENABLE\_PIN 24

上面是对五个关节上的电机的引脚的定义

// EG X-Y position bed driven by 2 steppers

// Alas its not possible to build an array of these with different pins for each :-(

AccelStepper joint1(1,E1\_STEP\_PIN, E1\_DIR\_PIN);

AccelStepper joint2(1,Z\_STEP\_PIN, Z\_DIR\_PIN);

AccelStepper joint3(1,Y\_STEP\_PIN, Y\_DIR\_PIN);

AccelStepper joint4(1,X\_STEP\_PIN, X\_DIR\_PIN);

AccelStepper joint5(1, E0\_STEP\_PIN, E0\_DIR\_PIN);

上面是对5个电机的定义，E1,Z.Y.X.E0连接的电机分别是joint1到joint5.上述语句调用了AccelStepper.h库中的函数，我推测其中的1应该表示Arduino中每产生一个脉冲则电机控制板tb6560就会收到一个脉冲；如果改成2那么Arduino中每产生一个脉冲

// Up to 10 steppers can be handled as a group by MultiStepper

MultiStepper steppers;

//test with uint8 converted to long

unsigned int x = 1000;

void setup() {

Serial.begin(250000);

// Configure each stepper

joint1.setMaxSpeed(50);

joint2.setMaxSpeed(50);

joint3.setMaxSpeed(50);

joint4.setMaxSpeed(50);

joint5.setMaxSpeed(80);

定义每个电机的最大速度

// Then give them to MultiStepper to manage

后面的函数来自于库MultiStepper.h

steppers.addStepper(joint1);

steppers.addStepper(joint2);

steppers.addStepper(joint3);

steppers.addStepper(joint4);

steppers.addStepper(joint5);

}

创建电机组

void loop()

{

long positions[5]; // Array of desired stepper positions

// Back of the envelope calculation for microsteps/revolution, where positions[i] is the number of steps (or microsteps).

positions[0] = 0; //4100 microsteps is 1/8 revolutions ----> 32800 microsteps/rev

positions[1] = 0; //2000 is 40/360 revolutions ---> 18000 microsteps/rev

positions[2] = 0; //4000 is 20/360 revolutions ---> 72000 microsteps/rev

positions[3] = 0; //820 is 1/4 revolution (200steps/revolution \* 16microsteps/step (since microstepping) ~= 32800 microsteps/rev)

positions[4] = 0; //2000 is 50/360 revolution ---> 14400

创建位置数组然后赋初值0，每进行一次loop循环进行一次上述过程

**steppers.moveTo(positions);**

**steppers.runSpeedToPosition(); // Blocks until all are in position**

机械臂回到初始位置，我认为上面两行代码是导致机械臂每运行到一个地方之后都会返回的问题。

delay(1);

//输入下一个位置//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int j = 0;

//不断循环检测串口缓存，一个个读入字符串

while (Serial.available() > 0)

{

comdata += char(Serial.read());

delay(2);

flag = 1;

}

//如果接收到数据则执行comdata分析操作，否则什么都不做

检测串口缓存，接受输入的字符串

if(flag == 1) {

for(int i = 0; i < comdata.length() ; i++){

if(comdata[i] == ','){

j++;

}

else{

numdata[j] = **numdata[j] \* 10 +** (comdata[i] - '0');

}

}

找出其中的数字“字符”，然后转换成对应的数字，存入数组numdata

comdata = String("");

flag = 0;

for(int i = 0; i < numdata\_length; i++){

Serial.println(numdata[i]);

}

输出读取到的位置坐标

}

// Move to a different coordinate

positions[0] = numdata[0];

positions[1] = numdata[1];

positions[2] = numdata[2];

positions[3] = numdata[3];

positions[4] = numdata[4];

把读取到的位置坐标存入positions数组

**steppers.moveTo(positions);**

**steppers.runSpeedToPosition(); // Blocks until all are in position**

移动到输入的位置；

for(int i = 0; i < numdata\_length; i++){

numdata[i] = 0;

}

数组numdata清零；

delay(1000);

}