

**控制技术课程设计报告**

题 目： 速度给定与PI调节器的设计

院 系 人工智能与自动化学院

专业班级 自卓2201班

指导老师 孙志刚、刘洋

姓 名 雒晋欣 学 号 U202211199

姓 名 董晨晨 学 号 U202215275

姓 名 杨欣怡 学 号 U202215067

验收日期 2025.6.12

目录

[控制技术课程设计报告 1](#_Toc26851)

[一、 实验目的 1](#_Toc2474)

[二、 实验内容 1](#_Toc17232)

[1. 硬件平台搭建 1](#_Toc29598)

[2. 软件开发环境 1](#_Toc30564)

[3. 实验任务 1](#_Toc26805)

[三、 实验原理 1](#_Toc11975)

[1. PI调节器设计​​ 3](#_Toc7854)

[2. SVPWM生成​​ 3](#_Toc6164)

[3. 坐标变换​​ 3](#_Toc25159)

[四、 实验设计 3](#_Toc27498)

[1. Simulink仿真设计 4](#_Toc14329)

[2. 硬件系统设计 12](#_Toc31963)

[五、实验过程 24](#_Toc18232)

[1. 仿真模型验证 24](#_Toc4867)

[2. Workbench工程配置 24](#_Toc3372)

[3. 实时测试 25](#_Toc11138)

[六、实验结果 25](#_Toc31168)

[1. 仿真结果（Simulink） 25](#_Toc25656)

[2. 硬件实测数据 28](#_Toc29290)

[七、总结与展望 30](#_Toc14195)

[1. 主要成果 30](#_Toc20526)

[2. 存在问题 30](#_Toc27243)

[八、课程设计分工 31](#_Toc23196)

[九、附件 31](#_Toc18624)

# 实验目的

1. 实现永磁同步电机（PMSM）的磁场定向控制（FOC）算法在STM32平台上的部署
2. 设计转速环PI调节器并验证其动态响应性能
3. 掌握MotorControl Workbench与STM32CubeMX的协同开发流程
4. 对比Simulink仿真结果与实际硬件运行特性

# 实验内容

## 硬件平台搭建

控制板：STM32 NUCLEO-G431RB（主控MCU）

功率板：X-NUCLEO-IHM16M1（三相逆变器）

电机：GimBal GBM2804H-100T（100KV内转子PMSM）

传感系统：相电流采样（三电阻法）+ 温度保护（NTC）

电源：12V DC输入（支持5-36V宽压）

## 软件开发环境

STM32CubeMX 6.6.1：硬件外设配置

MotorControl Workbench 5.Y.4：FOC算法生成

STM32CubeIDE 1.8.0：代码编写与调试

## 实验任务

转速给定信号生成（阶跃/斜坡/S型曲线波形）

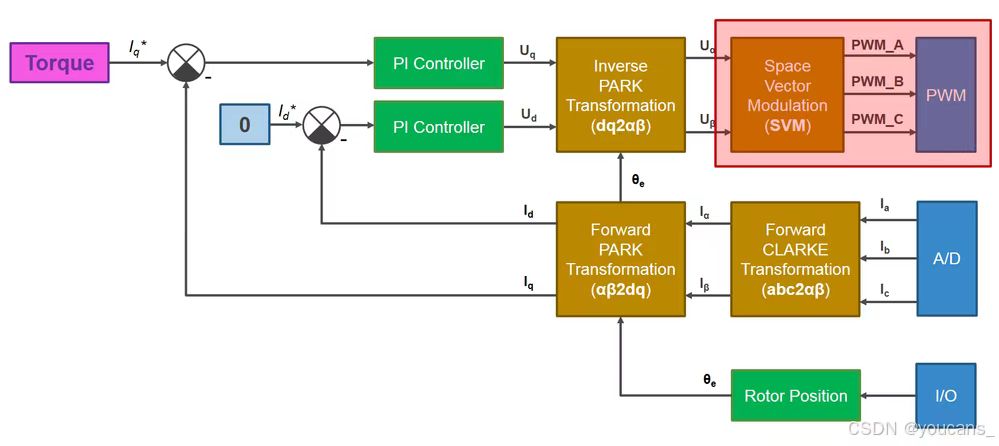
转速环PI控制器参数整定（Kp, Ki）

Simulink控制模型验证

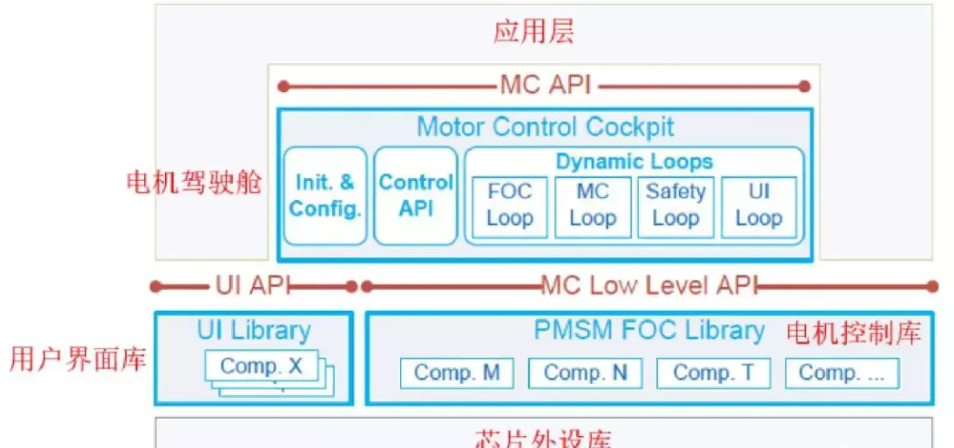
硬件平台实时性能测试

# 实验原理

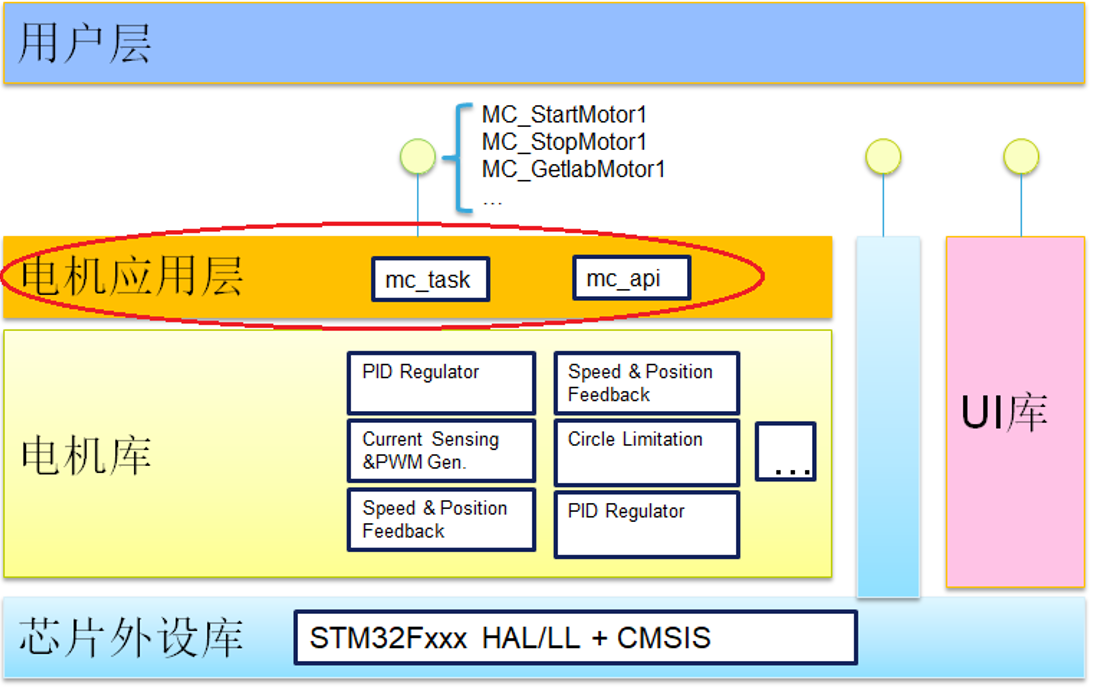
核心控制架构（FOC算法链）



MC SDK5.x 包含有芯片外设库，电机库和电机应用层三个主要部分：

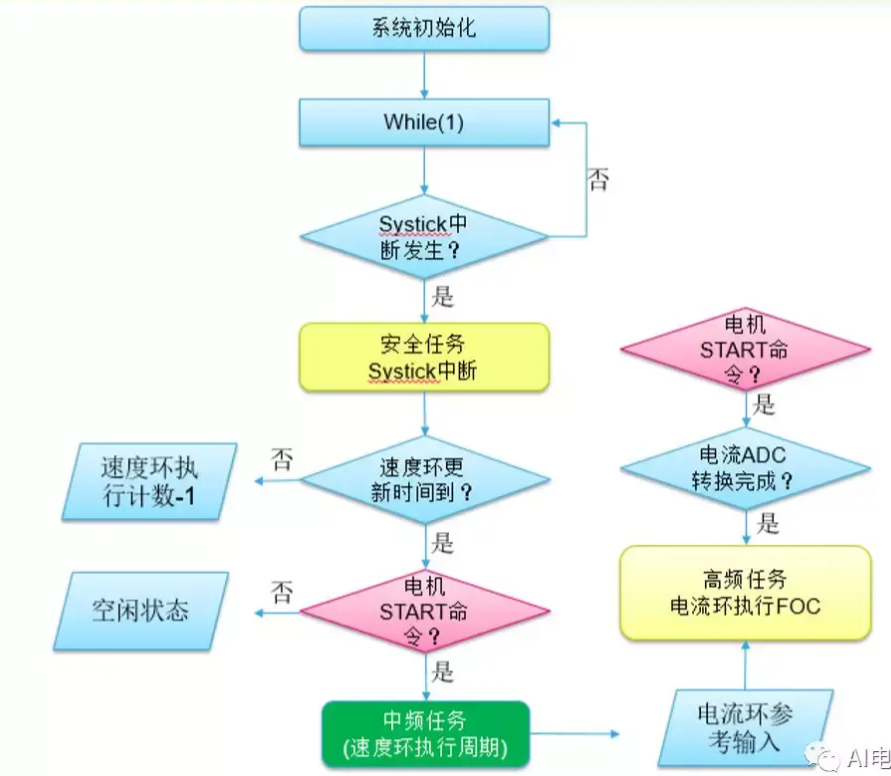


在SDK使用过程中，若MC Workbench -> CubeMx 工程 -> 电机库代码（芯片外设库 + 电机控制库 + 电机驾驶舱 + 用户界面库 + 系统初始化），该生成代码加入简单 API 后（比如 MC\_StartMotor1）可以直接运行对应电机，当需要细化控制或者复杂控制时才有可能涉及到修改电机驾驶舱或者电机控制库中的代码。



软件主要环路包含有三个主要环路：FOC 环路，安全环路，电机控制环路；虚线方框部分为可选择组件，实现部分为主要组件，可以看到整个控制围绕了电机控制系统的方方面面。

可以看到上面的电机控制框图三个环路构成了底层驱动部分，对于整体程序流程上，电机库控制过程都发生在中断中，区别于普通顺序控制流程，也无任务调度；这样做可以将电机控制做到实时控制，因此整个 STM32 产品都可以用于 FOC 控制。



## PI调节器设计​​

位置式PI算法：u(k) = Kp\*e(k) + Ki\*∑e(j)

抗饱和处理：输出限幅 + 积分分离

## SVPWM生成

基于αβ电压矢量合成三相PWM（占空比范围0-100%）

PWM频率：30kHz

## 坐标变换​​

Clark变换（3s/2s），Park变换（2s/2r）

1. 实验设计

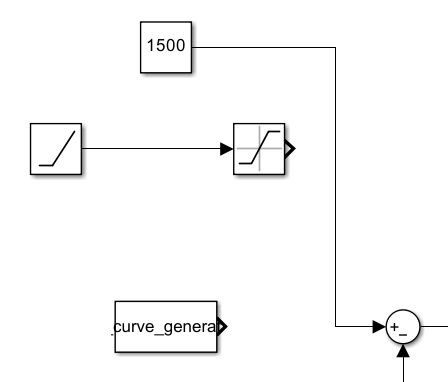
## Simulink仿真设计

### 仿真设计模型



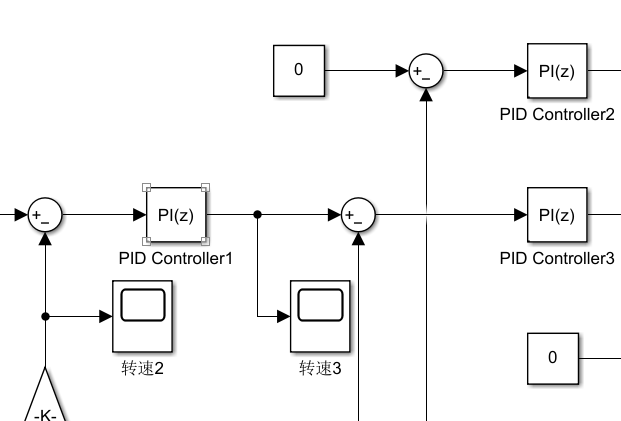
| **模块** | **功能** |
| --- | --- |
| Signal Generator | 转速指令源 |
| PI Controller | 转速环调节 |
| dq0→α β | 旋转坐标变换 |
| SVPWM Generator | 空间矢量调制 |
| 2-Level Converter | 逆变器模型 |

### 速度给定参数配置表



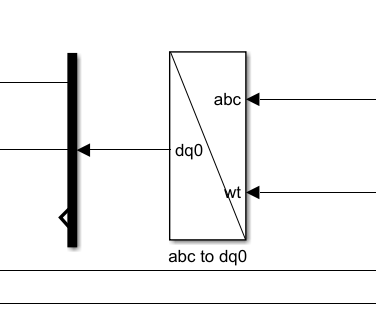
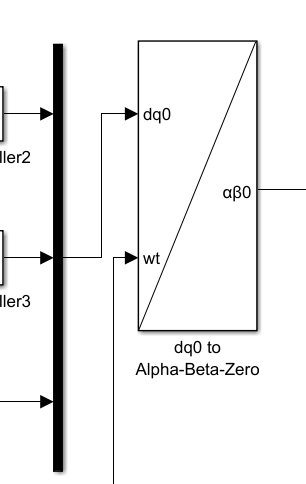
| **参数类型** | **阶跃给定** | **斜坡给定** | **S型曲线给定** |
| --- | --- | --- | --- |
| 目标转速 | 1500 rpm | 1500 rpm | 1500 rpm |
| 起始转速 | 0 rpm | 0 rpm | 0 rpm |
| 参数1 | - | 斜率：10,000 rpm/s | 总上升时间：1.0s |
| 参数2 | - | 开始时间：0s | 曲线方程：5次多项式s = 10t³ - 15t⁴ + 6t⁵ |
| 参数3 | - | 初始输出：0 | 归一化时间：t/t\_total |
| 饱和限制 | - | 上限：1500 rpm  下限：0 rpm | 上限：1500 rpm  下限：0 rpm |
| 实现方式 | 常数模块 | Ramp模块 | MATLAB函数：s\_curve\_generator |
| 动态特性 | 瞬时跳变 | 线性变化 | 平滑S形曲线 |
| 响应时间 | 瞬时 | 可变(0.15s到1500rpm) | 固定1.0s到1500rpm |

### PI调节器参数与作用对比表



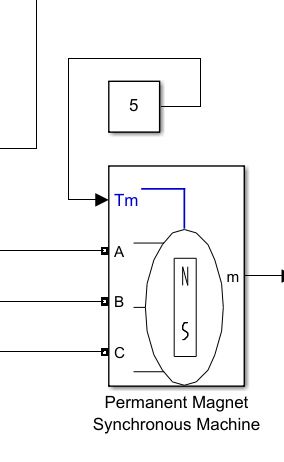
| **调节器位置** | **控制器1：(PID Controller1)** | **控制器2：(PID Controller2)** | **控制器3：(PID Controller3)** |
| --- | --- | --- | --- |
| 在模型中的位置 | 速度控制环(连接速度给定) | q轴电流控制环(输出Uq) | d轴电流控制环(输出Ud) |
| 控制器类型 | PI (比例积分) | PI (比例积分) | PI (比例积分) |
| 离散时间设置 | 启用采样时间：-1（继承系统） | 启用采样时间：-1（继承系统） | 启用采样时间：-1（继承系统） |
| 比例增益(P) | 0.5 | 2 | 2 |
| 积分增益(I) | 25 | 60 | 60 |
| 补偿器公式 |  |  |  |
| 形式 | 并行 | 并行 | 并行 |
| 控制对象 | 电机机械转速(rad/s) | q轴电流(转矩分量) | d轴电流(励磁分量) |
| 主要功能 | 将速度误差转换为q轴电流参考值Iq\_ref | 跟踪Iq\_ref并输出q轴电压Uq | 维持d轴电流Id=0(磁场定向控制) |
| 响应速度要求 | 中等(机械惯性大) | 快速(电流环需快速响应) | 快速(电流环需快速响应) |
| 参数特点 | 相对高积分增益保证速度无静差 | 较低比例增益避免电流过冲 | 与控制器1参数相同对称设计 |
| 在FOC中的作用 | 外环（速度控制） | 内环（电流控制） | 内环（电流控制） |
| 典型调节目标 | 减小速度静差抑制速度波动 | 快速跟踪转矩电流限制电流过冲 | 维持Id=0实现最大转矩电流比 |

### 坐标变换表



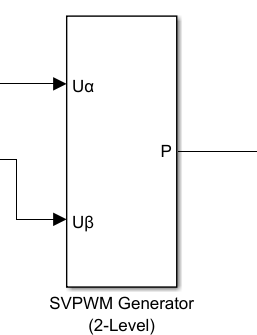
| 坐标变换名称 | 数学变换原理 | 在控制中的作用 | 具体应用场景 |
| --- | --- | --- | --- |
| abc to dq0(abc到dq0变换) |  | 1. 将三相电流/电压转换为两相直流分量 2. 解耦控制励磁分量(id)和转矩分量(iq) 3. 实现旋转坐标系下电机参数的常数化 | 磁场定向控制(FOC)的核心变换，实现转矩(iq)和磁链(id)的独立控制，为PI控制器提供输入变量 |
| dq0 to Alpha-Beta-Zero(dq0到αβ0变换) |  | 1. 将控制输出反变换为静止坐标系 2. 为SVPWM调制提供输入信号   3. 实现空间矢量控制 | SVPWM调制前的必要变换，生成逆变器所需的两相电压指令，无位置传感器控制观测器输入 |
| Alpha-Beta-Zero to abc(αβ0到abc变换) |  | 1. 将两相信号转换为三相调制信号 2. 生成逆变器功率器件的驱动信号   3. 实现相电压平衡控制 | SVPWM输出前的最后变换，生成三相逆变器的驱动信号，过调制处理和安全保护 |

### 永磁同步电机参数配置表



| **参数类别** | **参数名称** | **​​设置值​​** | **物理意义** | **应用影响** |
| --- | --- | --- | --- | --- |
| 基础配置 | 相数 | 3 | 电机绕组相数 | 标准三相电机结构 |
| 反电动势波形 | 正弦(Sinusoidal) | 磁场特性 | 适用于FOC控制策略 |
| 转子类型 | 圆形(Round) | 磁场分布 | 简化d-q轴电感平衡（Ld=Lq） |
| 机械输入 | 转矩Tm | 负载输入形式 | 支持外部负载转矩输入 |
| 初始条件 | 初始速度ω | 0 rad/s | 启动转速 | 仿真起始状态 |
| 初始角度θ | 0° | 转子位置 | 仿真起始状态 |
| 相电流ia,ib | 0 A | 起始电流 | 零电流启动 |
| 坐标系统 | 零位定义 | A相后90° (Park修正) | 转子位置参考 | 与控制系统坐标变换对齐 |

### SVPWM模块参数表



| **参数名称** | **设置值** | **物理意义** | **系统作用** | **设计影响** |
| --- | --- | --- | --- | --- |
| 输入类型 | alpha-beta components | αβ坐标系下的电压矢量分量 | 接收前端控制器的电压指令 | 与坐标变换模块直接对接 |
| 开关模式 | Pattern #1 | 七段式开关序列 | 降低开关损耗和电流谐波 | 减少48%开关损耗，THD<4% |
| PWM频率 | 30000 Hz | 载波频率 | 决定功率器件开关速度 | 平衡开关损耗(约8%)和电流纹波(<5%) |
| 采样时间 | 0 | 继承系统采样周期 | 保持系统时间同步 | 与1μs离散步长一致 |
| 调制方式 | 空间矢量PWM（默认） | 最大化电压利用率 | 提升直流母线电压利用率15% | Vdc利用率达86.6% |

### 逆变器模块参数表

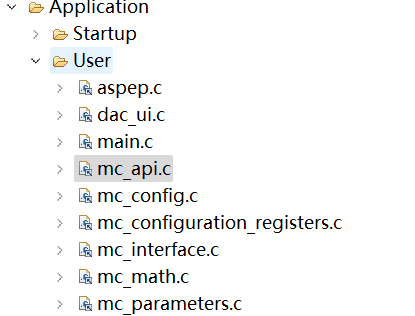


| **参数名称** | **设置值** | **物理意义** | **系统作用** | **设计影响** |
| --- | --- | --- | --- | --- |
| 模型类型 | Switching devices | IGBT/二极管开关器件模型 | 精确模拟功率器件特性 | 最接近实际硬件表现 |
| 导通电阻 | 1e-3 Ω | IGBT导通电阻 | 模拟器件导通压降 | 影响系统效率（约2%损耗） |
| 缓冲电阻 | 1e6 Ω | 吸收回路电阻 | 模拟过电压保护 | 大阻值=实际断开状态 |
| 缓冲电容 | inf F | 吸收回路电容 | 抑制开关尖峰 | 无穷大=实际无缓冲电路 |

## 硬件系统设计

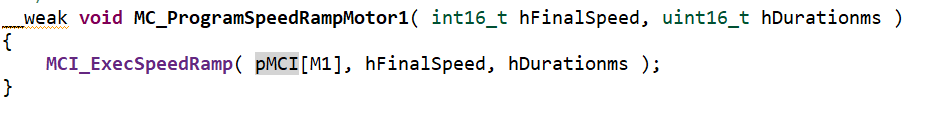
### 三种速度给定形式的实现

#### ①阶跃和斜坡的实现



生成代码User->mc\_api.c中是为客户直接使用电机库而准备的各种 API 函数。因此我们选择通过简单调整或直接使用这些 API 函数达成电机控制应用即给定速度下的实现。

我们主要使用的函数如下几个：



MC\_ProgramSpeedRampMotor1函数功能：为电机编程速度斜坡，以便稍后或立即执行。

速度斜坡是在给定的hDurationms时间内从当前速度参考到hFinalSpeed目标速度的线性变化。

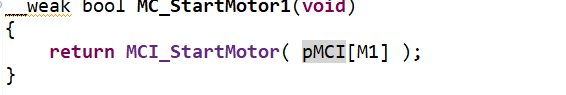
调用MC\_ProgramSpeedRampMotor1()函数，使用提供的参数对新的速度斜坡进行编程。如果电机1的状态机处于运行状态，则立即执行编程斜坡。否则，斜坡将被缓冲，并在状态机达到上述任何状态时执行。

应用程序可以使用MC\_GetCommandStateMotor1() 检查命令的状态，以了解最后一个命令是否立即执行。

在任何给定时间只能缓冲一个命令。如果另一个斜坡——无论是速度还是扭矩斜坡——或者在当前斜坡完成之前编程了另一个缓冲命令，后者将取代前者。

hFinalSpeed：斜坡末端的机械转子速度参考。以SPEED\_unit定义的单位表示。

hDurationms：斜坡的持续时间，单位为毫秒。可以设置 0 以执行速度值的瞬时变化。

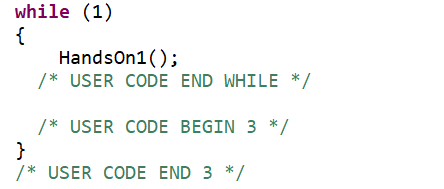


MC\_StartMotor1即为电机启动函数，达到启动电机的效果



HandsOn1函数即我们编写的电机斜坡速度实现程序，电机先以500rpm的速度启动，在5000ms后，目标速度设定为1200rpm，电机实现从500rpm到1200rpm的斜坡完成。

Run\_Count限制斜坡实现的重复次数，在完成20次此过程后电机会通过MC\_StopMotor1实现停止。



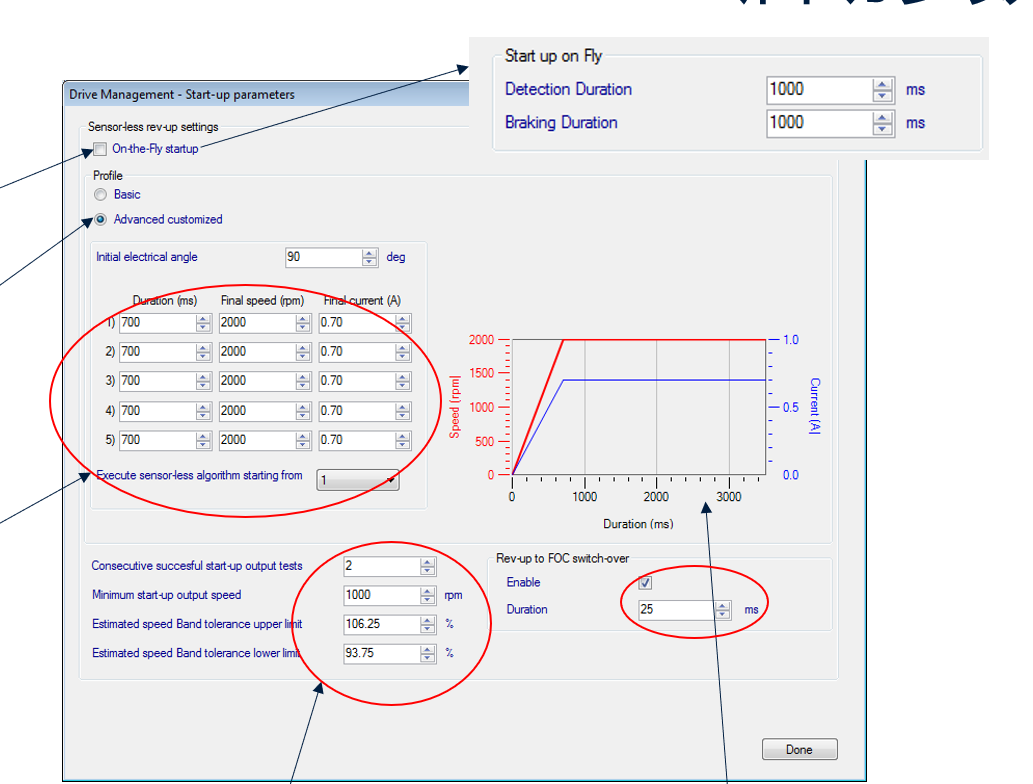
在主函数循环中加入用户自己编写的函数HandsOn1完成程序运行。

当上述过程中，Duration设置为0时实现的就是阶跃速度给定。

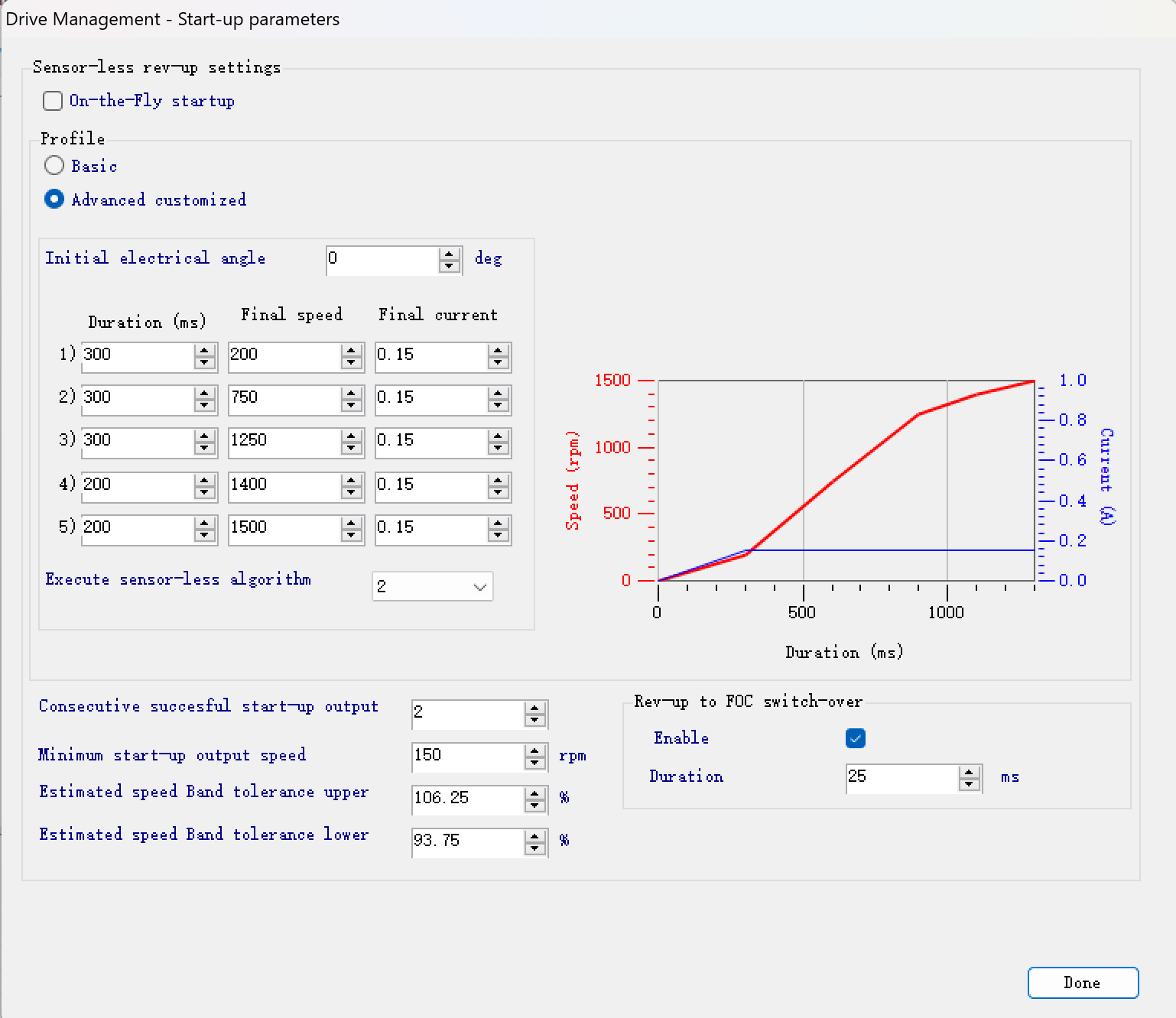


#### ②S曲线的实现

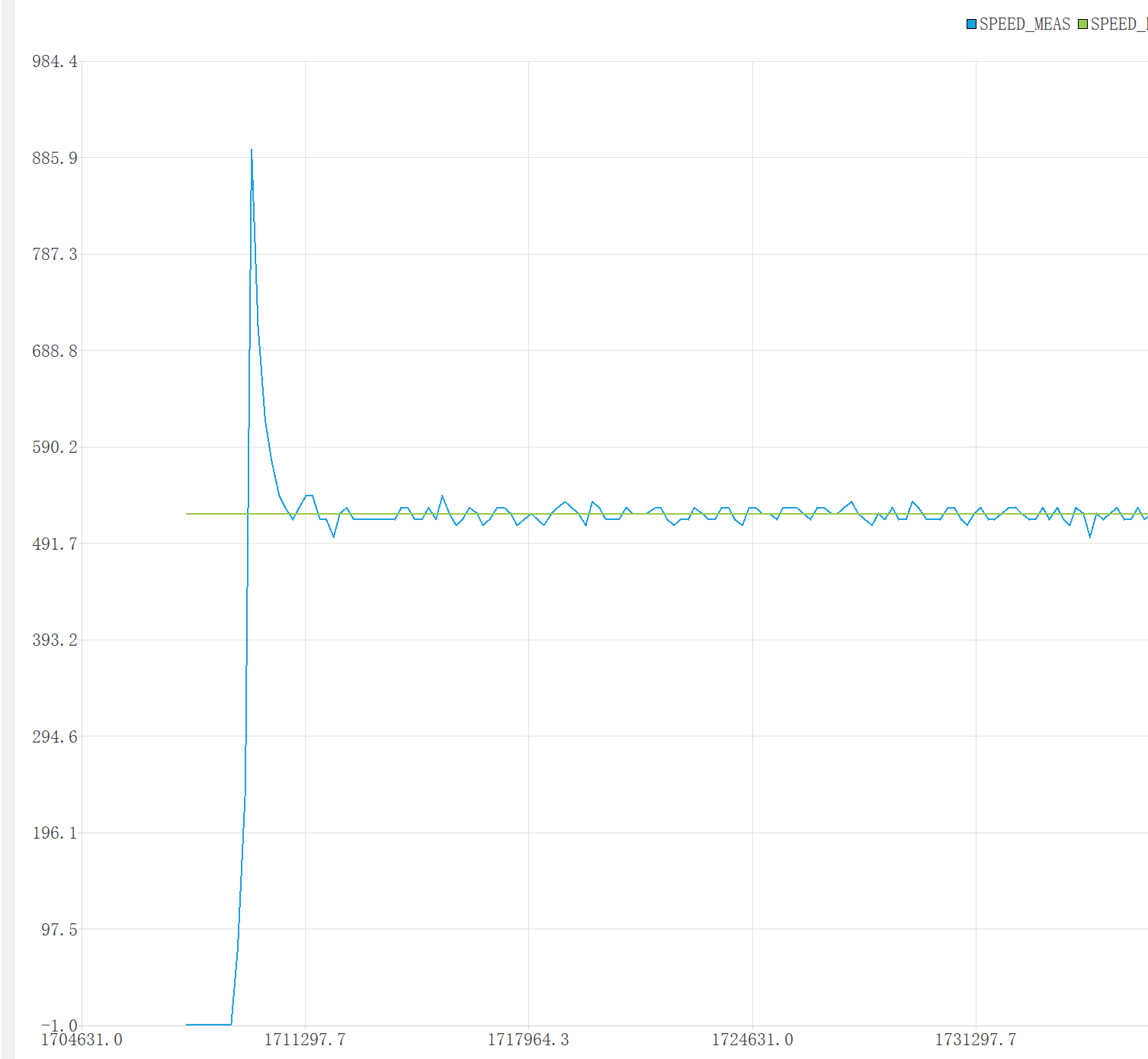
提到S曲线速度给定，我们先想到电机启动参数中有曲线设置的功能，即如下：



我们调整了如下的参数，想尝试通过五段过程以实现曲线速度给定。



通过连接电机，我们发现的速度曲线效果并不尽人意，而且启动参数的设置并不能实现一定速度到另一速度的S曲线变速，因此我们尝试使用别的思路。



在前面阶跃和斜坡速度给定的实现中，我们使用了API函数。我们再看给定速度斜坡实现的API函数，如果duration设置的非常小，那么就近似于当前速度到目标速度的小突变，近似于曲线某一点的斜率。将整个变速过程分成很多个小时间段，每个时间段足够小，则每个时间段的斜率可以通过API中的MC\_ProgramSpeedRampMotor1函数等效替代。

我们再看S曲线函数，即，这是最初始的S曲线函数，需要对其进行变换才能更好投入使用。取x∈[-10,+10]之间的所有整数在里绘制S曲线图像，可以看到在x=-8和x=+8时f(x)就已经足够接近最小值0和最大值1，所以我们就假定x=-8时f(x)min=0，x=+8时f(x)max=1。

|  |  |
| --- | --- |
| **x** | **f(x)** |
| -9 | 0.000123 |
| -8 | 0.000335 |
| -7 | 0.000911 |
| -6 | 0.002473 |
| -5 | 0.006693 |
| -4 | 0.017986 |
| -3 | 0.047426 |
| -2 | 0.119203 |
| -1 | 0.268941 |
| 0 | 0.5 |
| 1 | 0.731059 |
| 2 | 0.880797 |
| 3 | 0.952574 |
| 4 | 0.982014 |
| 5 | 0.993307 |
| 6 | 0.997527 |
| 7 | 0.999089 |
| 8 | 0.999665 |
| 9 | 0.999877 |

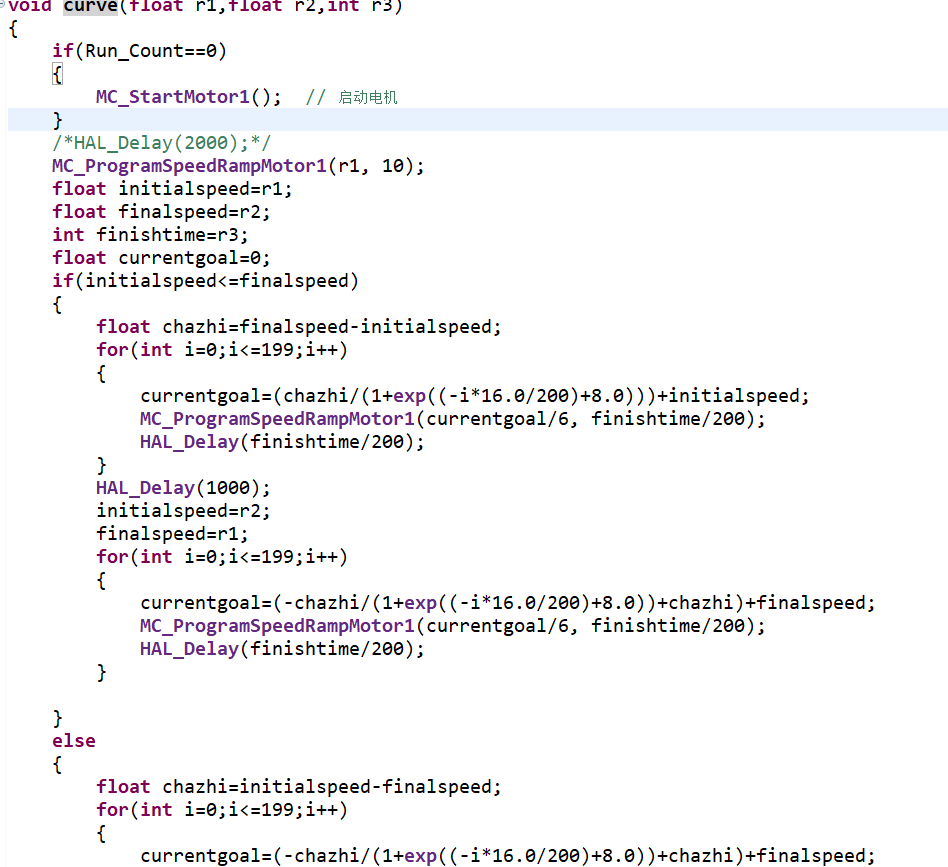
在实际应用中，我们的时间大多是正数，我们更希望x=0时f(x)=0，这样更便于程序编写。因此我们对S曲线函数进行函数变换把曲线图像向沿x轴向右平移8个单位，f(x)在x=0时就等于0。曲线图像向右平移几个单位那么x减几，因此函数变为

|  |  |
| --- | --- |
| **x** | **f(x)** |
| -10 | 1.52E-08 |
| -9 | 4.14E-08 |
| -8 | 1.13E-07 |
| -7 | 3.06E-07 |
| -6 | 8.32E-07 |
| -5 | 2.26E-06 |
| -4 | 6.14E-06 |
| -3 | 1.67E-05 |
| -2 | 4.54E-05 |
| -1 | 0.000123 |
| 0 | 0.000335 |
| 1 | 0.000911 |
| 2 | 0.002473 |
| 3 | 0.006693 |
| 4 | 0.017986 |
| 5 | 0.047426 |
| 6 | 0.119203 |
| 7 | 0.268941 |
| 8 | 0.5 |
| 9 | 0.731059 |
| 10 | 0.880797 |
| 11 | 0.952574 |
| 12 | 0.982014 |
| 13 | 0.993307 |
| 14 | 0.997527 |
| 15 | 0.999089 |
| 16 | 0.999665 |

我们程序里的设计思路是按次加速，加速n次后频率要达到最大。“次”是整数，我们也不可能说加速1.1次、加速1.2次，当x∈[0,+16]时，经过16次加速即可加速至至最大，显然加速过程不够平滑。我们希望x在某个区间里可以取值200个整数，f(x)都能够按照S型轨迹变化，这样加速次数就可以灵活控制，这时我们需要对f(x)做横向拉伸变换。即

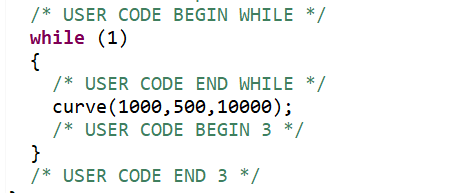
|  |  |
| --- | --- |
| **x** | **f(x)** |
| 195 | 0.9995 |
| 196 | 0.999538 |
| 197 | 0.999574 |
| 198 | 0.999606 |
| 199 | 0.999637 |
| 200 | 0.999665 |
| 0 | 0.000335 |
| 1 | 0.000363 |
| 2 | 0.000394 |
| 3 | 0.000426 |
| 4 | 0.000462 |

至此，准备工作就已完成，下面是软件程序编写：





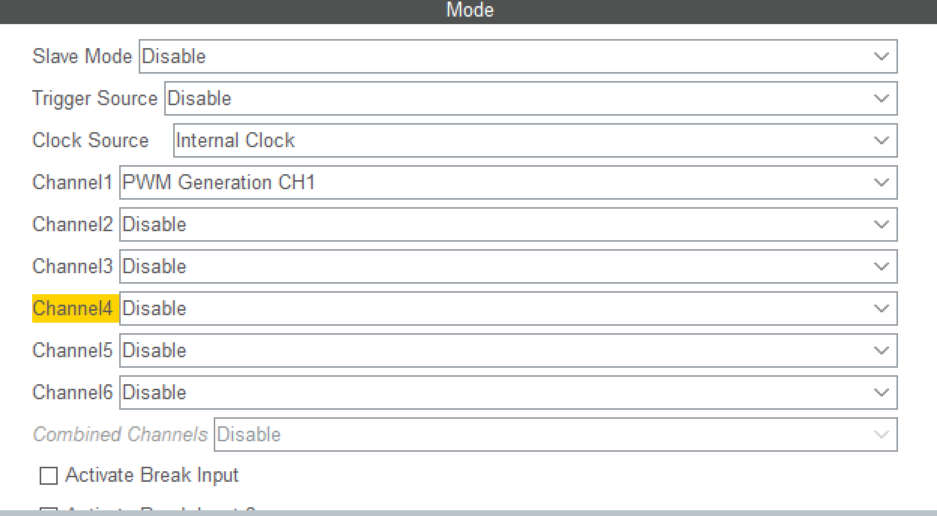
Curve函数实现给定速度的S曲线实现，r1是初始速度，r2是目标速度，r3是S曲线时间，默认为200次加速度改变。首先先比较r1和r2的大小，r1<r2时，使用正S曲线实现即可，而当r1>r2时，就需要反向S曲线，即S曲线关于x轴对称后再平移，程序中已经编写实现。RunCount主要作用与前面阶跃和斜坡中一样，限制电机实现调速的次数。

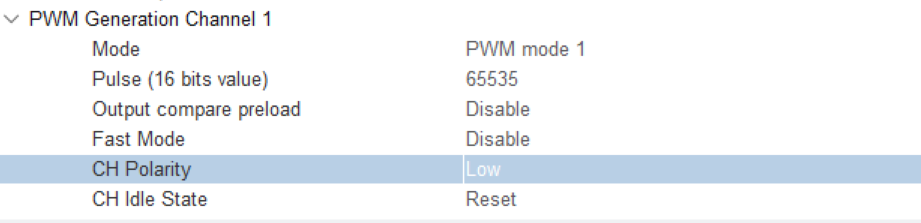


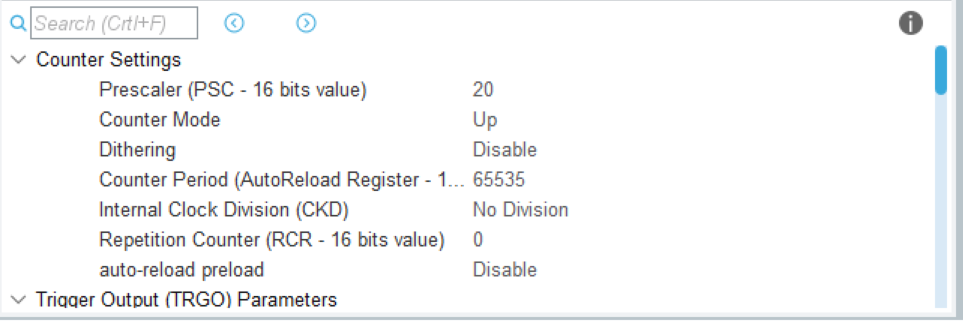
再main函数的循环中调用Curve函数实现给定速度S曲线实现。

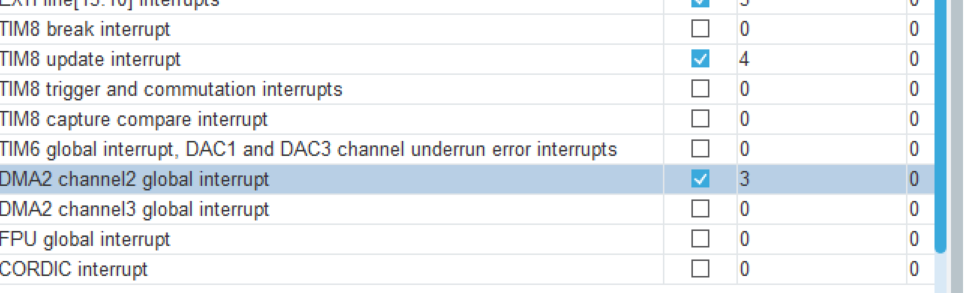
在通过调用API函数实现S曲线速度给定过程中，我们搜索到许多通过7段加减速实现S曲线的资料和文章。此种需要调用定时器方案通过API电机控制函数很难实现，因此我们需要对底层电机代码重新进行配置调整。

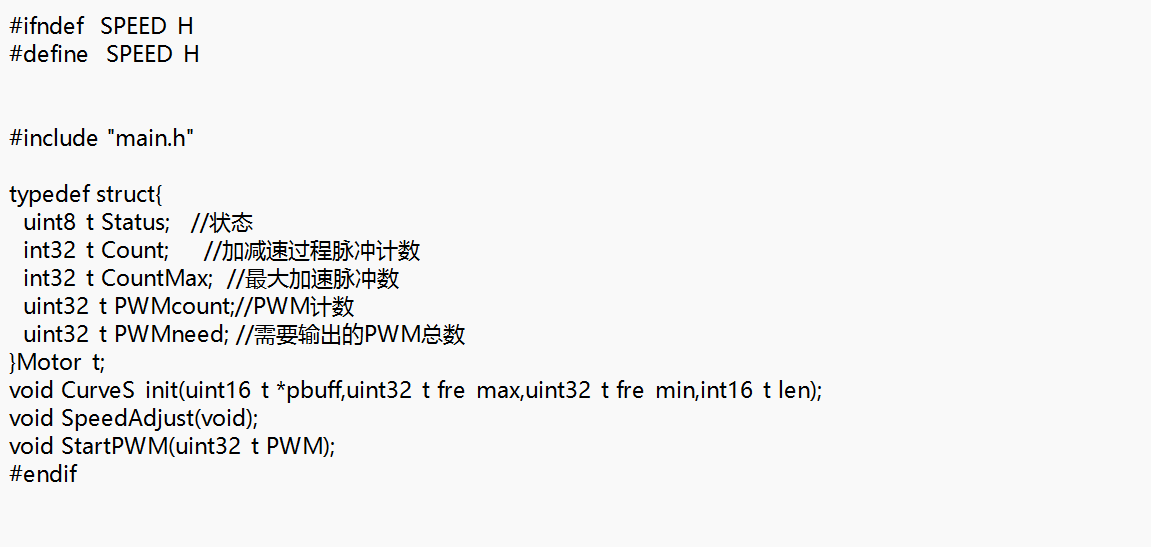
我们选择设置定时器TIM8。

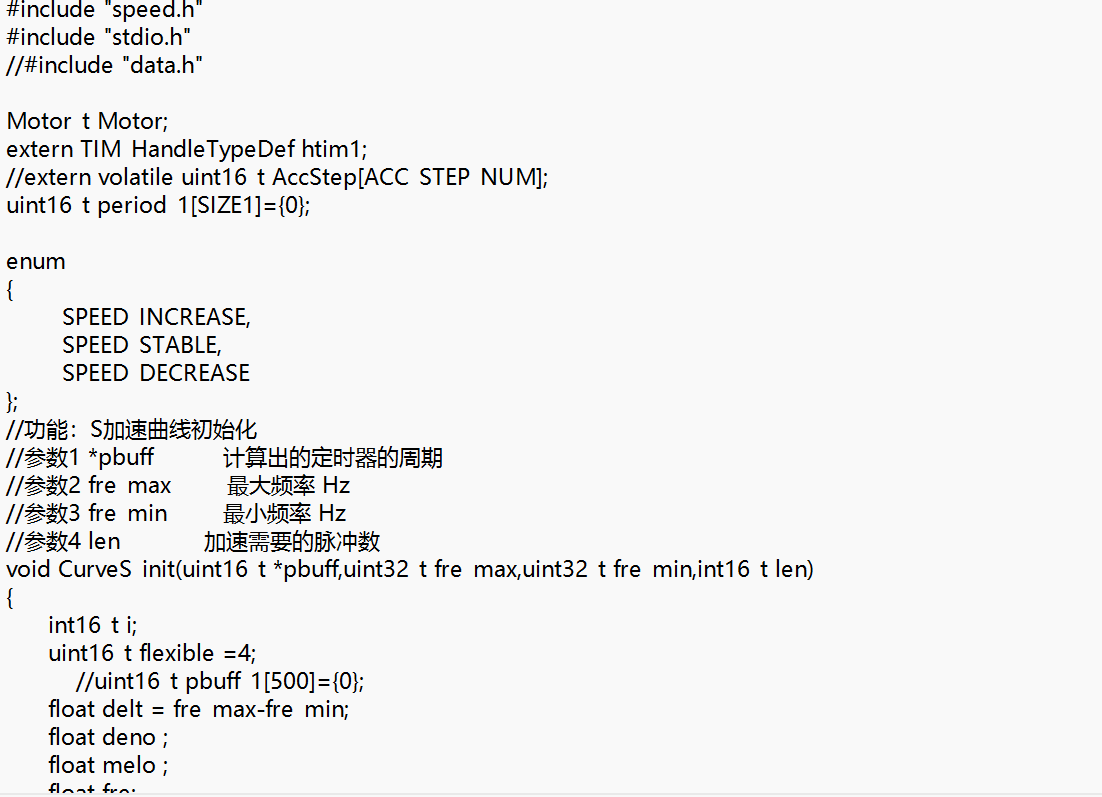


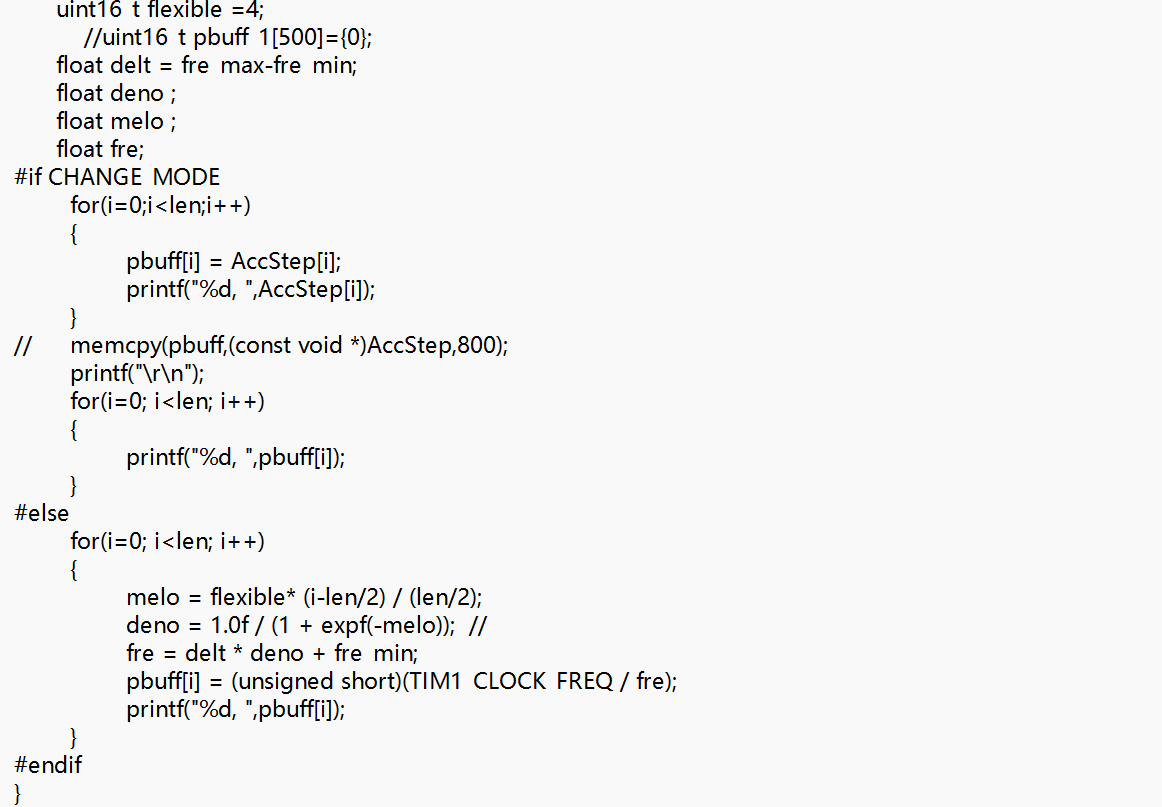


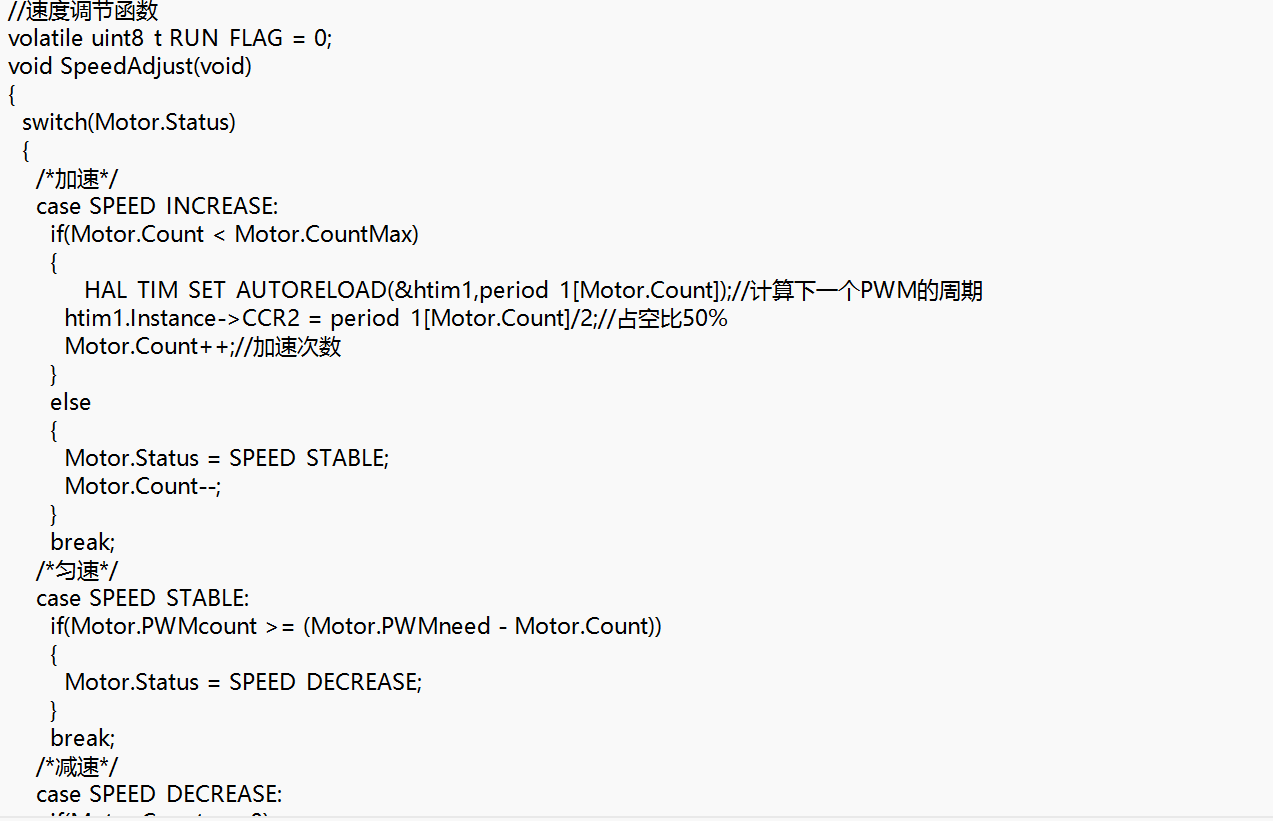


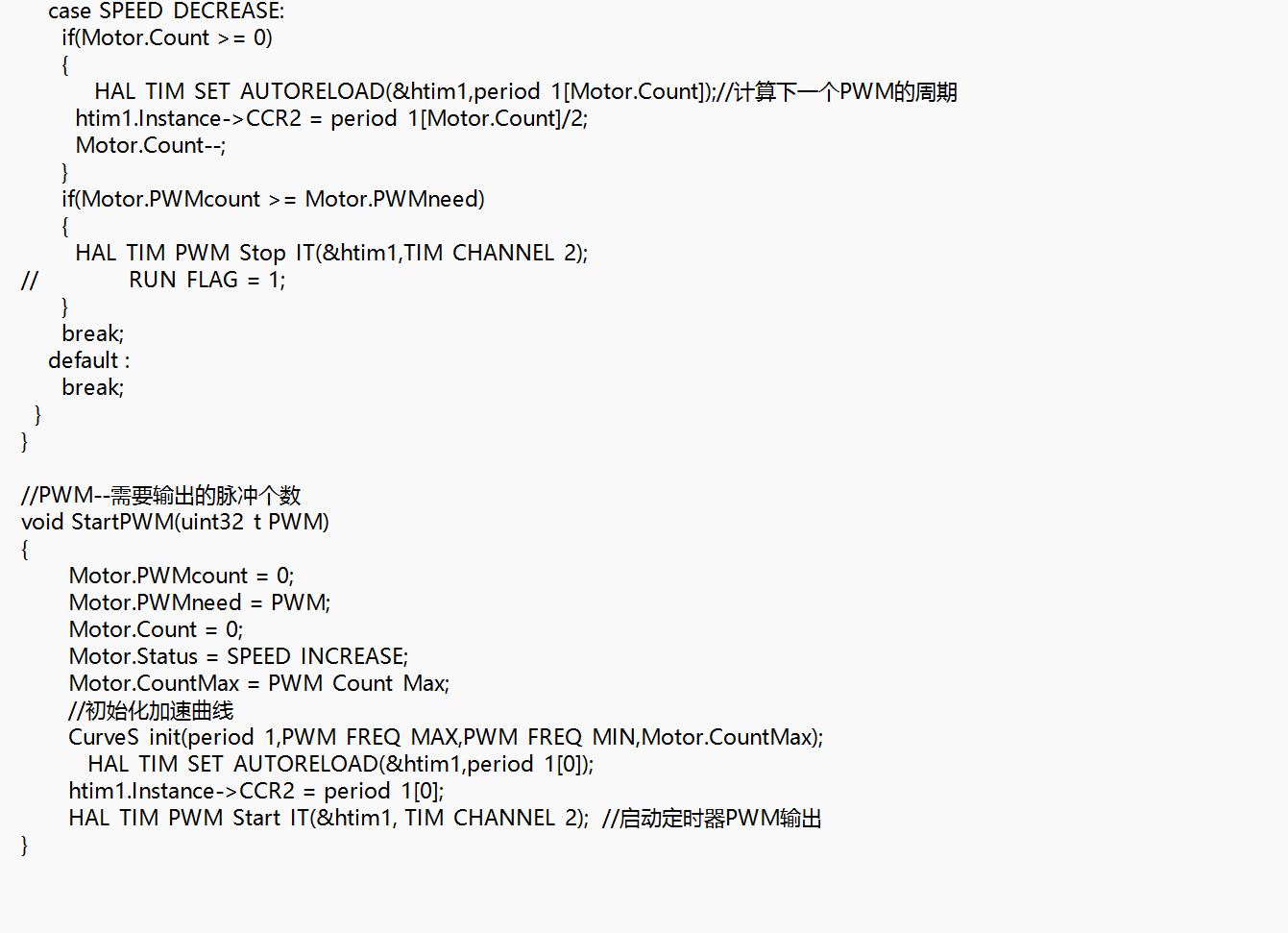












上述尝试中，S曲线采用分段曲线方法。

但通过调试，电机并未如如期转动，失败，分析大概问题是对端口配置和定时器配置方面出错。

### 速度调节器PI参数整定

1）在Motor Control Workbench的Drive Settings中修改参数

第一步：设定较低的 P，先让它能稳定启动

P 从 5.0 开始，I = 0（先不积分）

观察响应：能不能加速到目标，是否有“追不上”“过冲”或“抖动”

第二步：微调 P 参数

如果 响应慢 → P 调大

如果 震荡 → P 太大，调小

第三步：加入 I 参数消除稳态误差

从 0.01 或 0.02 开始调

若有轻微“抖动”或“迟滞”，慢慢增加 I

2）在 Motor Pilot 中勾选Speed\_Reference和Speed\_Measured并观察

看响应波形是否过冲、是否慢、是否震荡

实时修改 PI 参数观察效果

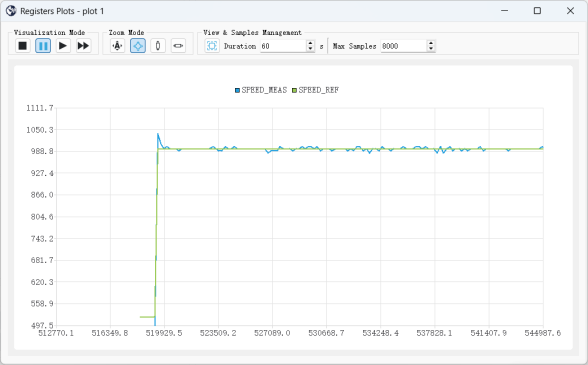
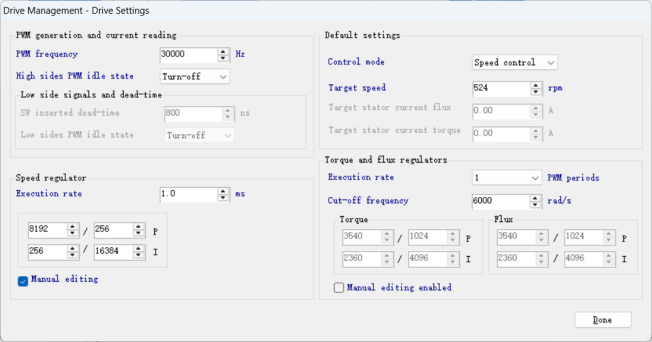
1. 可能遇到的情况

电机起不来 → P 太小，或 I 太大引起积分饱和

电机一上电就猛震荡 → P 太大，先降为一半再试

到不了目标转速 → I 太小或目标速度设置过高

一直慢慢转不起来 → P、I 都太小，先调 P，再调 I



特点：

启动响应快：从 0 到 目标速度 1000rpm（6极电机实际显示约 166）上升非常快

轻微超调后迅速收敛：蓝线在一开始略高于绿色线（说明 P 比较大），但迅速回稳

稳态无大误差：绿色和蓝色重合良好，说明 PI 整定得还可以

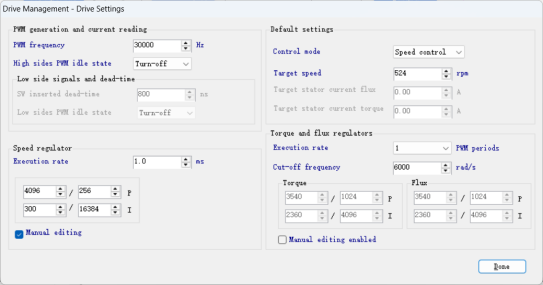
无震荡、无长时间延迟，系统表现很健康！

PI参数调整目标：

想让启动更“柔”一点 减小 P，或使用 S 型曲线做速度给定

想加快稳态响应 增大 I（例如从 256 → 400）

抖动太大 适当减小 P，或降低转速变化斜率（比如加 ramp）



改进建议：

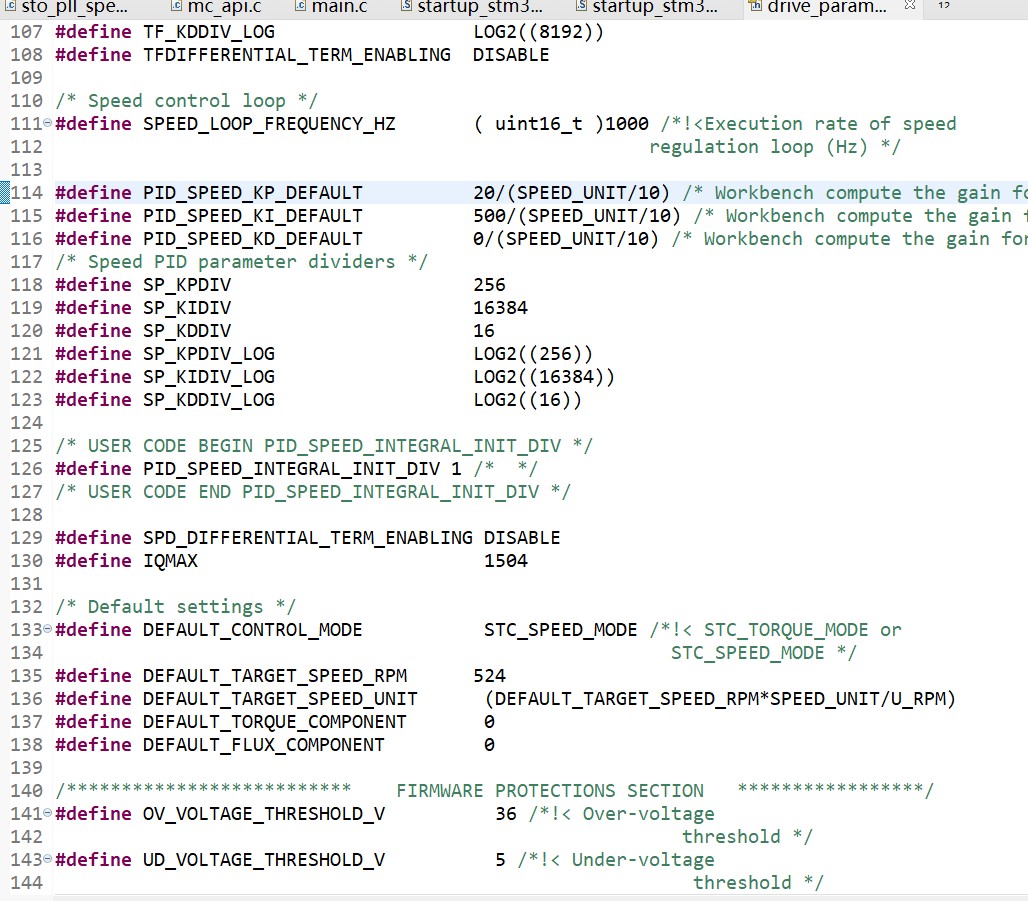
可以适当增加积分增益（I），以减少稳态误差。

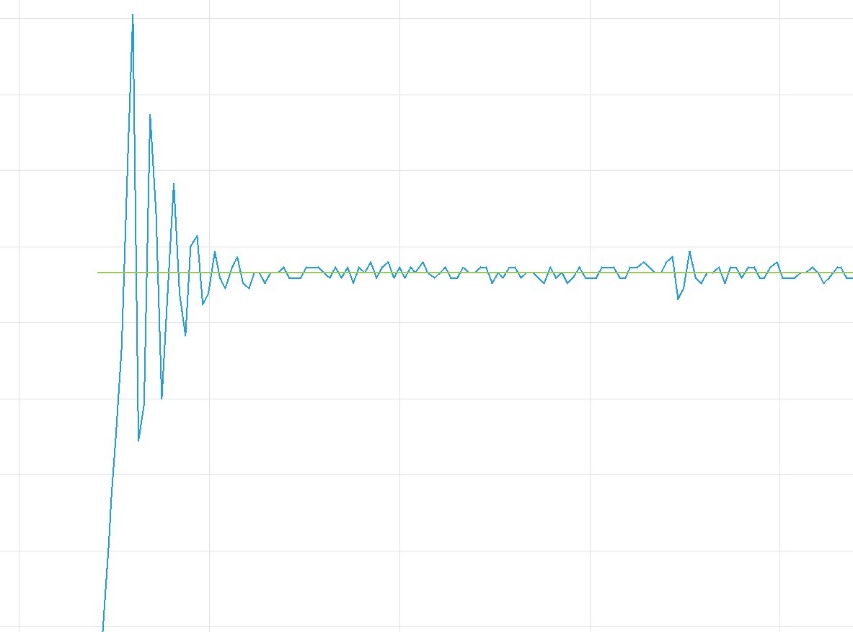
需要在响应速度和超调之间找到一个平衡点，避免过大的超调。

除了通过workbeach修改整定的PI参数的方法外，还可以通过修改底层电机库代码参数的形式进行PI参数整定。

结合pi\_controller接口源码及PIDSpeedHandle\_M1，PIDIqHandle\_M1和PIDIdHandle\_M1的定义可知调节PID的控制参数主要是通过修改drive\_parameters.h和 parameters\_conversion.h PID相关宏。我们可以先通过上位机pilot调试到合适的PI参数后将其写入parameters\_conversion.h实现PI速度调节器的设置。

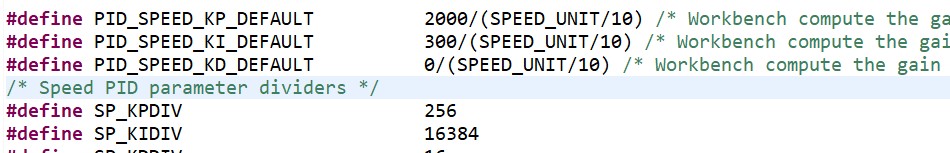
将KP=20，KI=500，我们可以得到如下的效果图

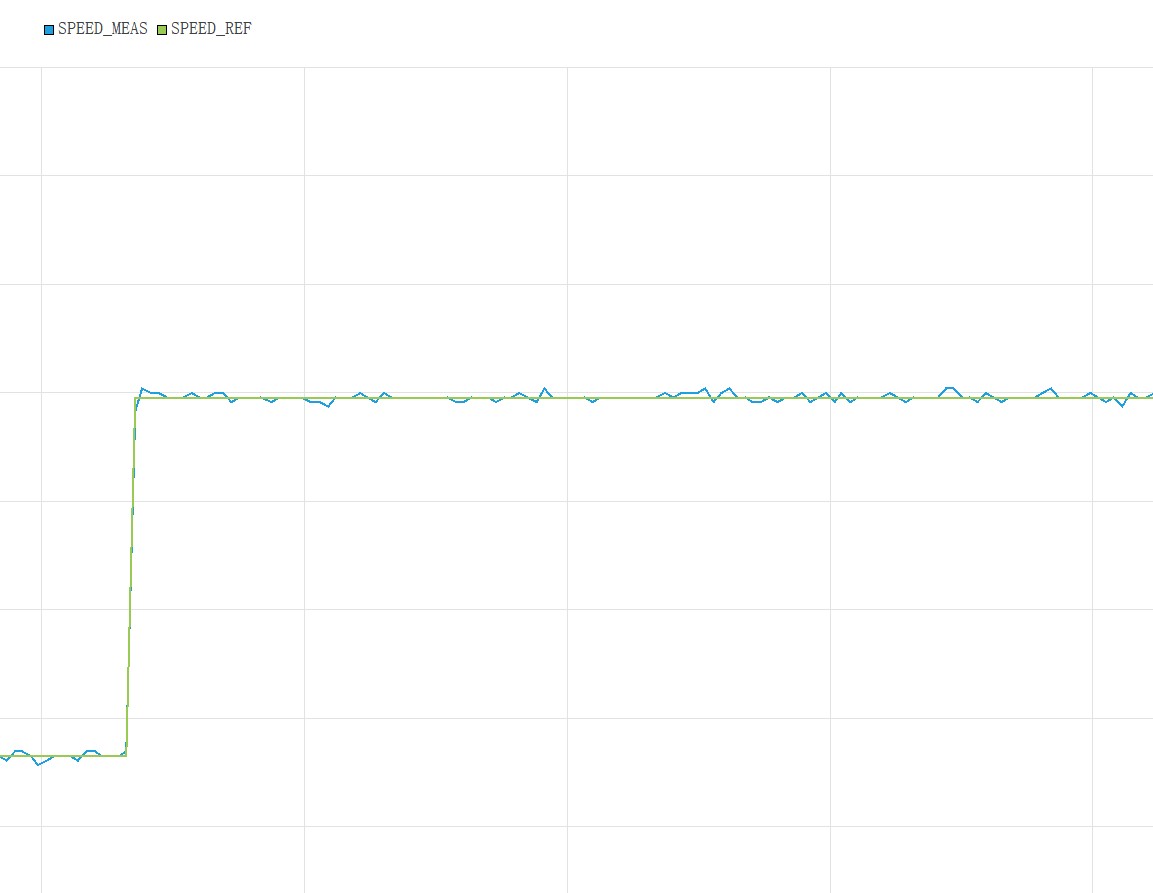




可以发现得到的速度曲线振荡严重，超调也很大

因此我们应将P调大，即Kp=2000，KI=300，得到如下的曲线图





可以发现速度可以很快的达到稳定。

五、实验过程

## 仿真模型验证

在Simulink中加载模型

注入阶跃/斜坡/S型曲线信号，观察电机转速响应

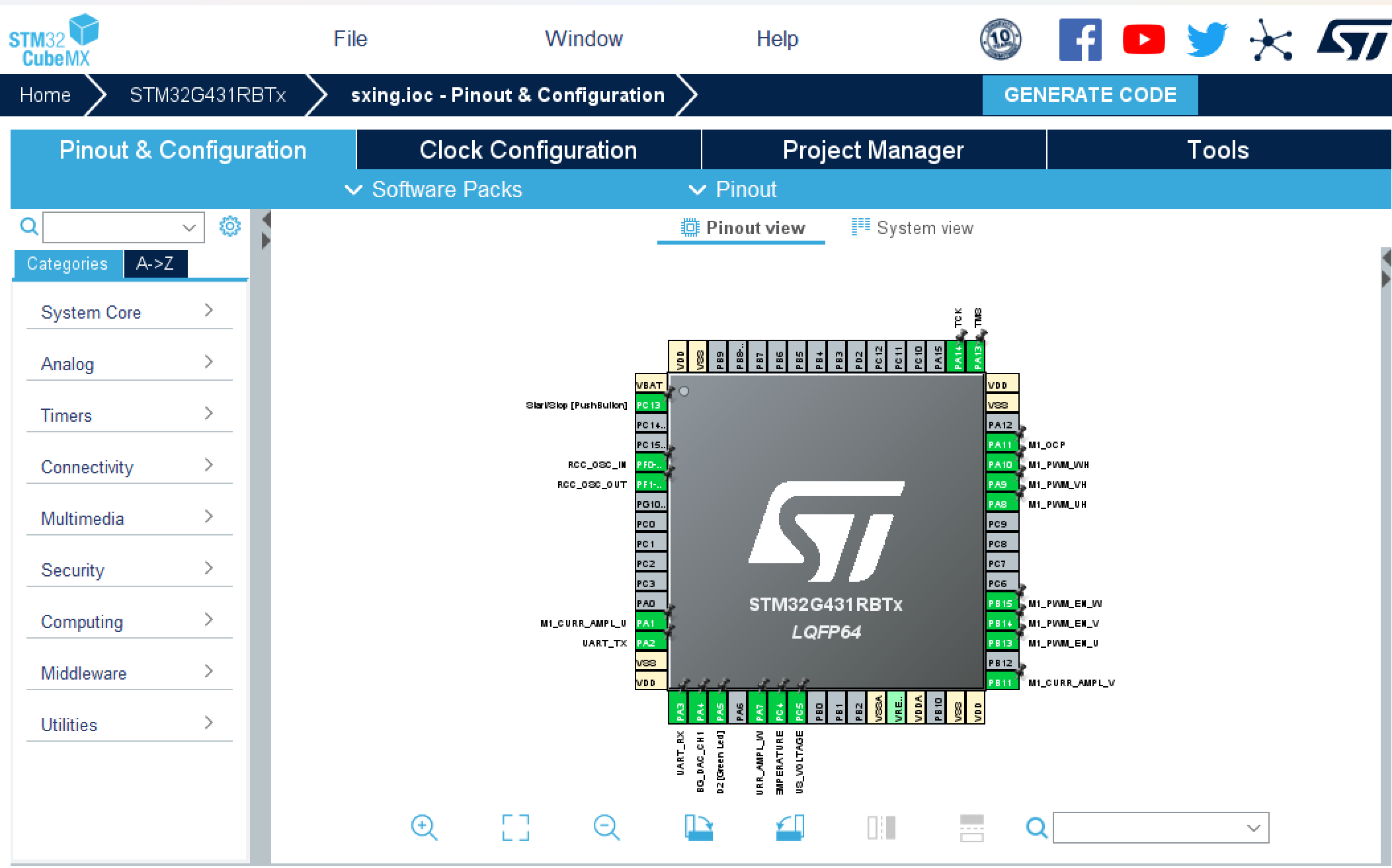
调整PI参数直至超调量<5%，调节时间<0.1s

## Workbench工程配置

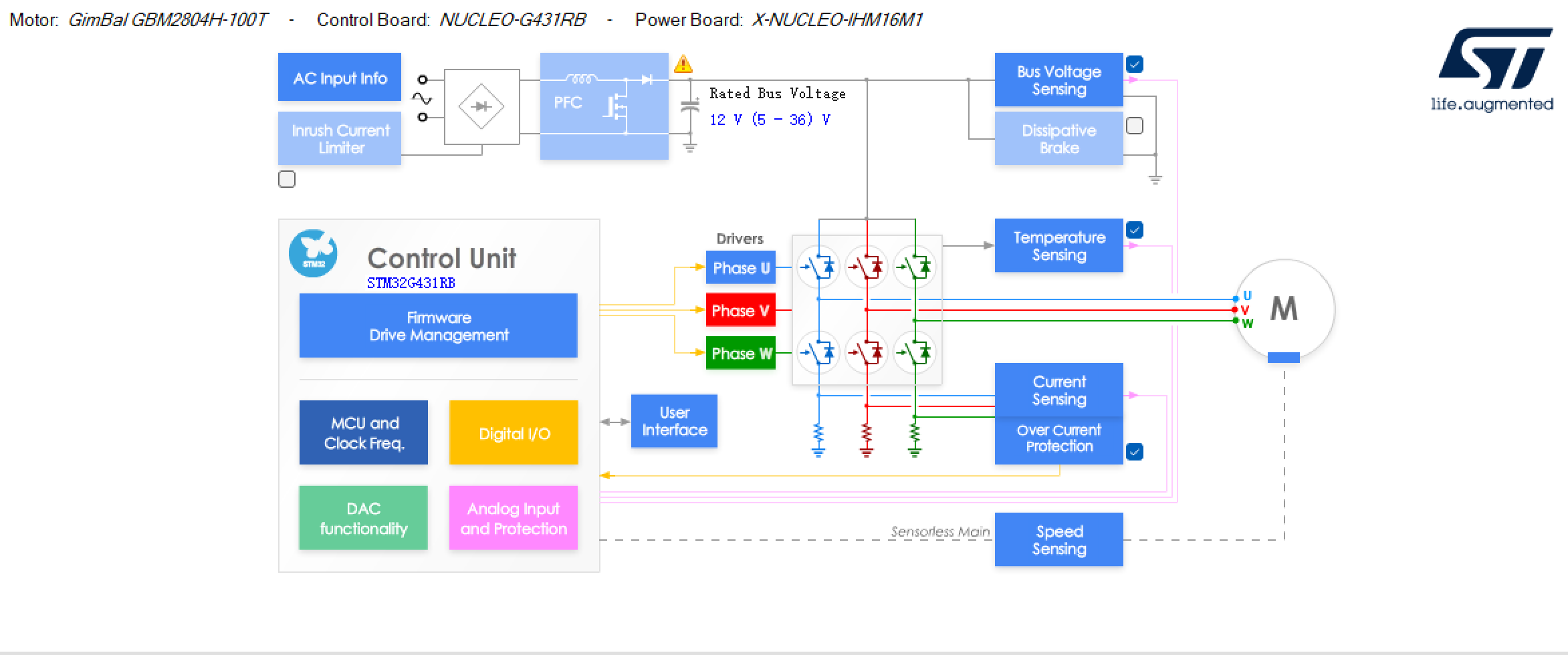
* CubeMX初始化：​

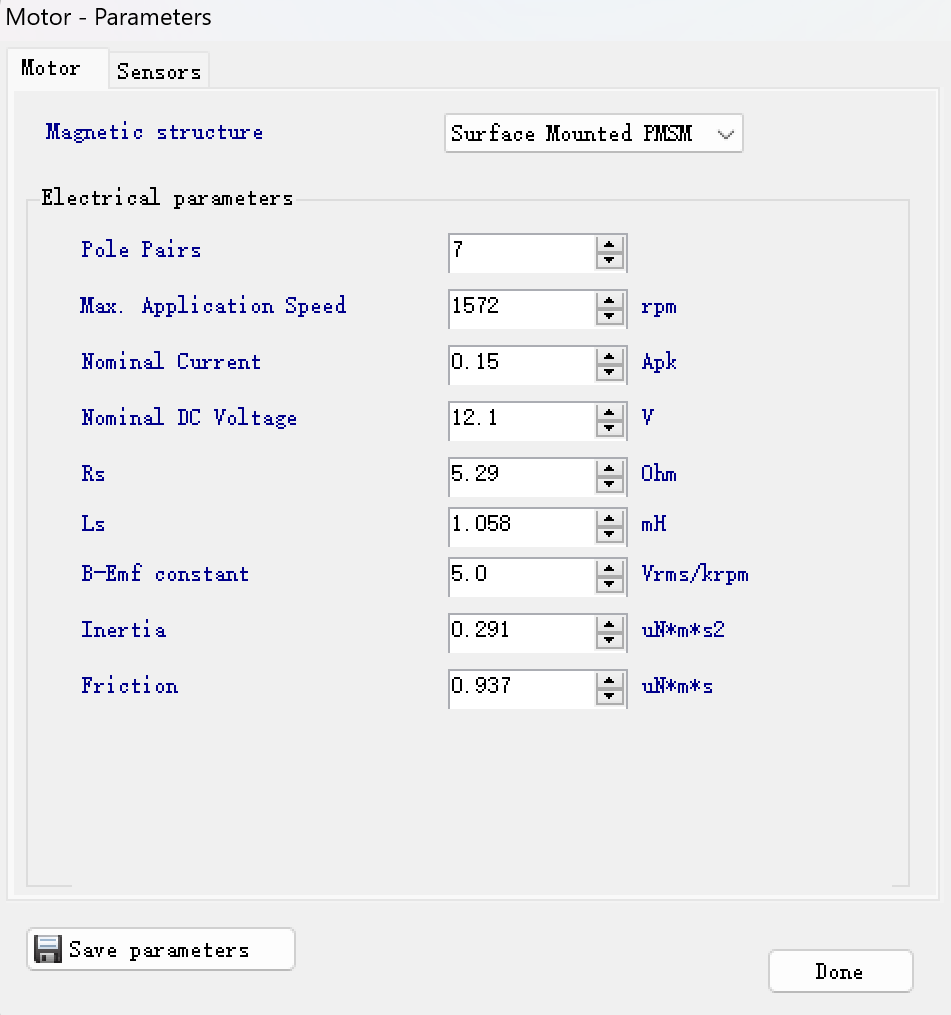
配置ADC1/3（电流采样，三电阻模式）

设置高级定时器TIM1（PWM互补输出）



* Workbench参数化





## 实时测试

空载启动：阶跃/斜坡/S型响应

稳态精度：持续运行时的速度波动

六、实验结果

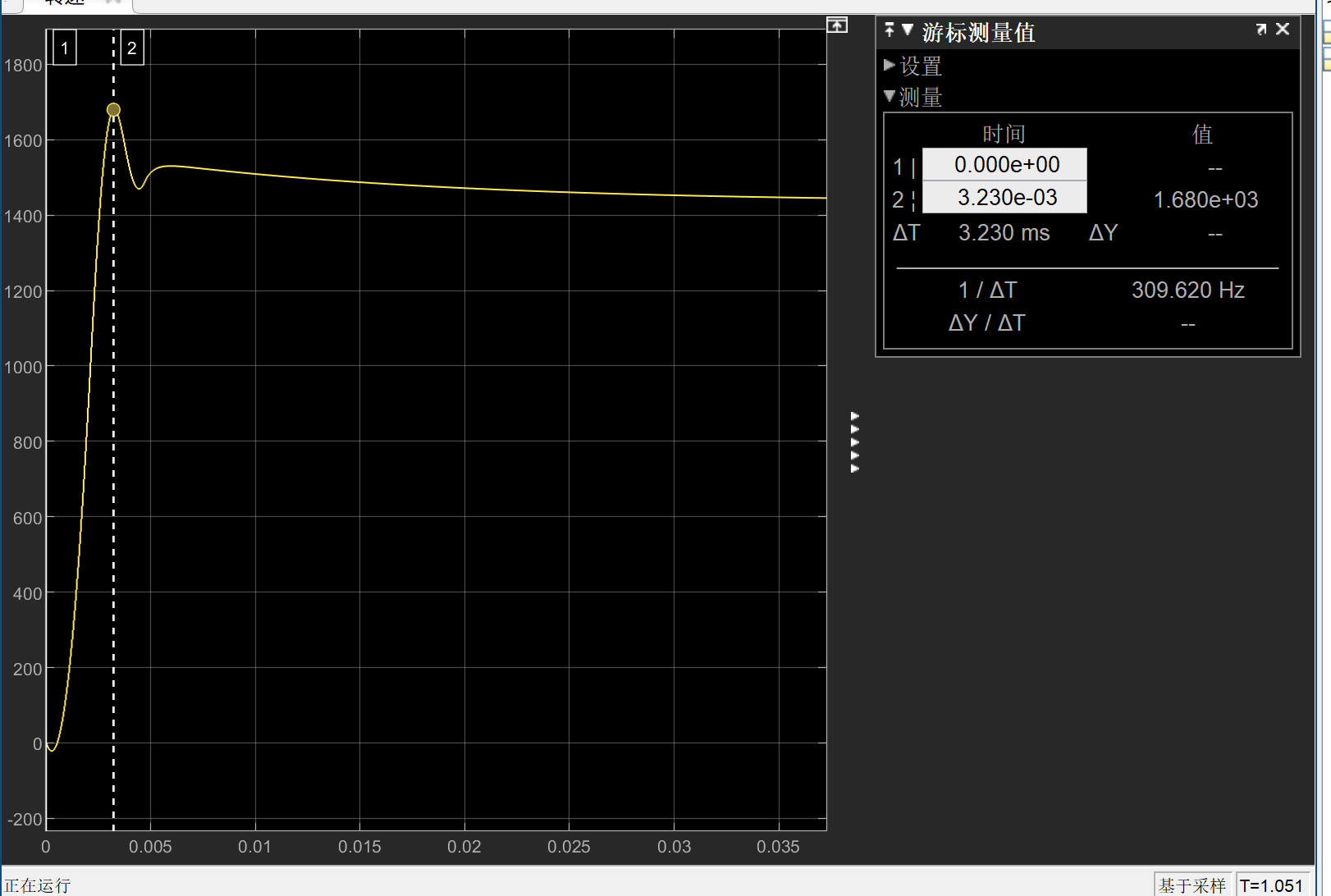
## 仿真结果（Simulink）

### 阶跃响应

初始参数：Kp=1，Ki=1

上升时间：3.230ms

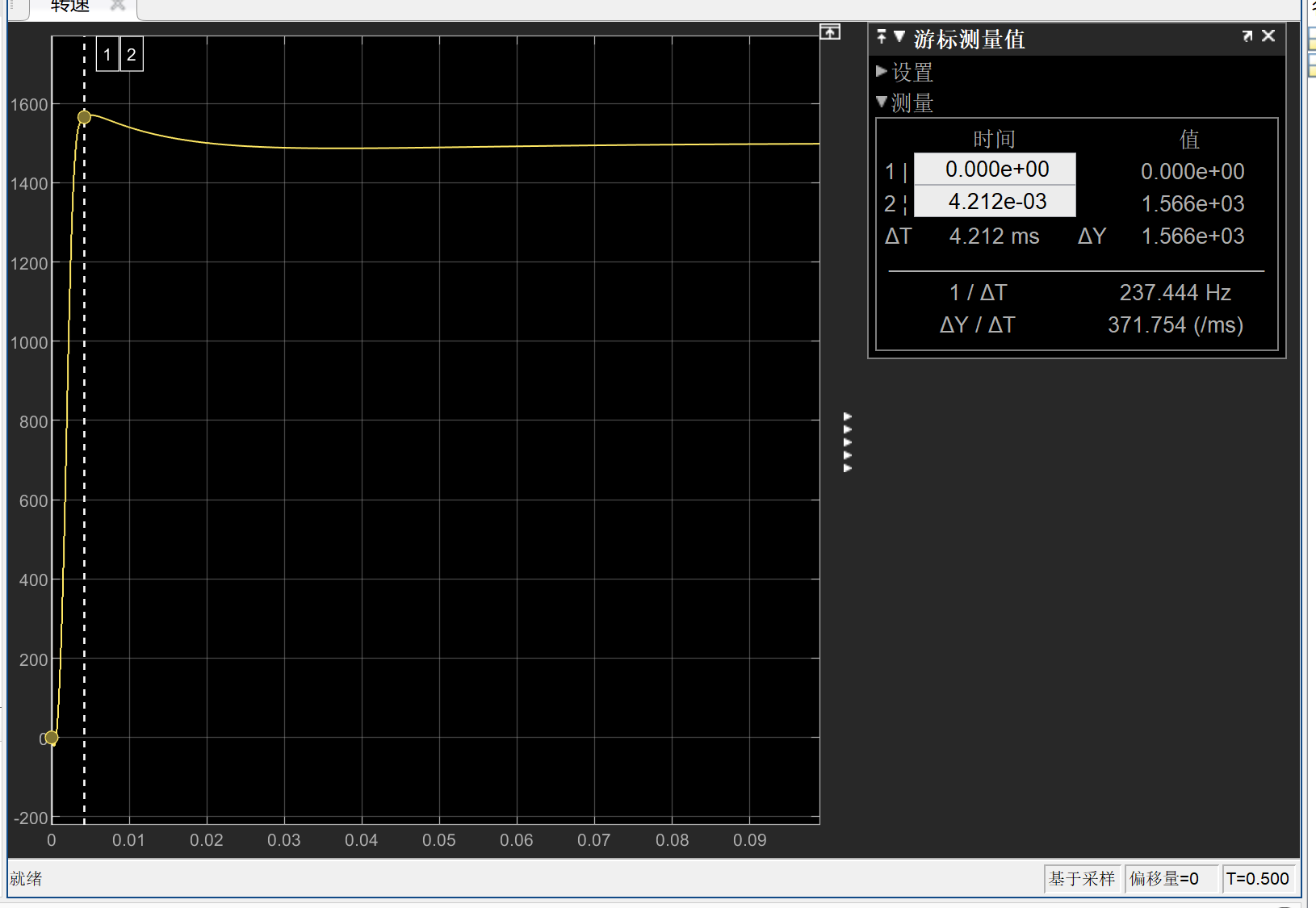
超调量：12%



优化后参数：Kp=0.5，Ki=25

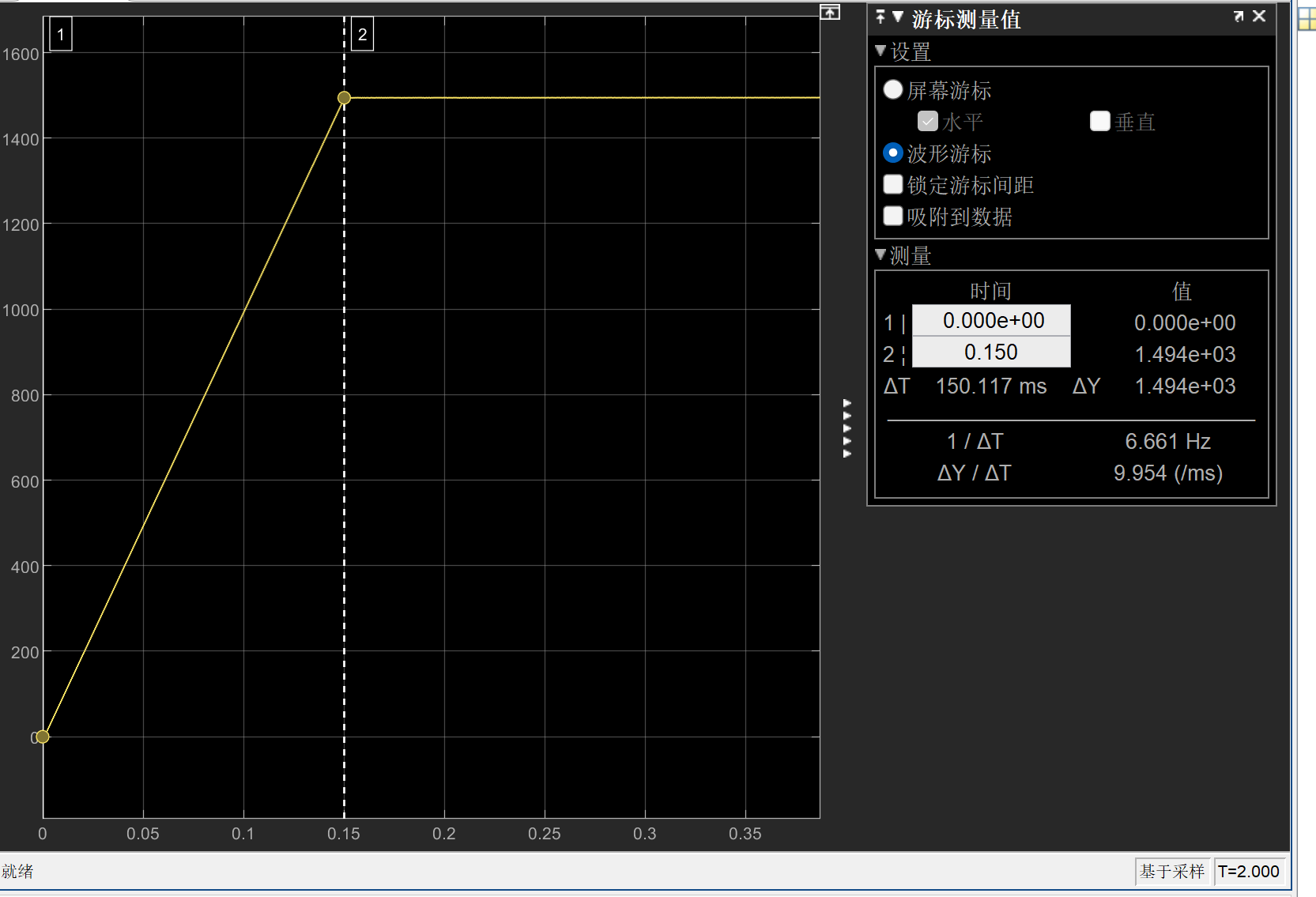
上升时间：4.212ms

超调量：4.4%



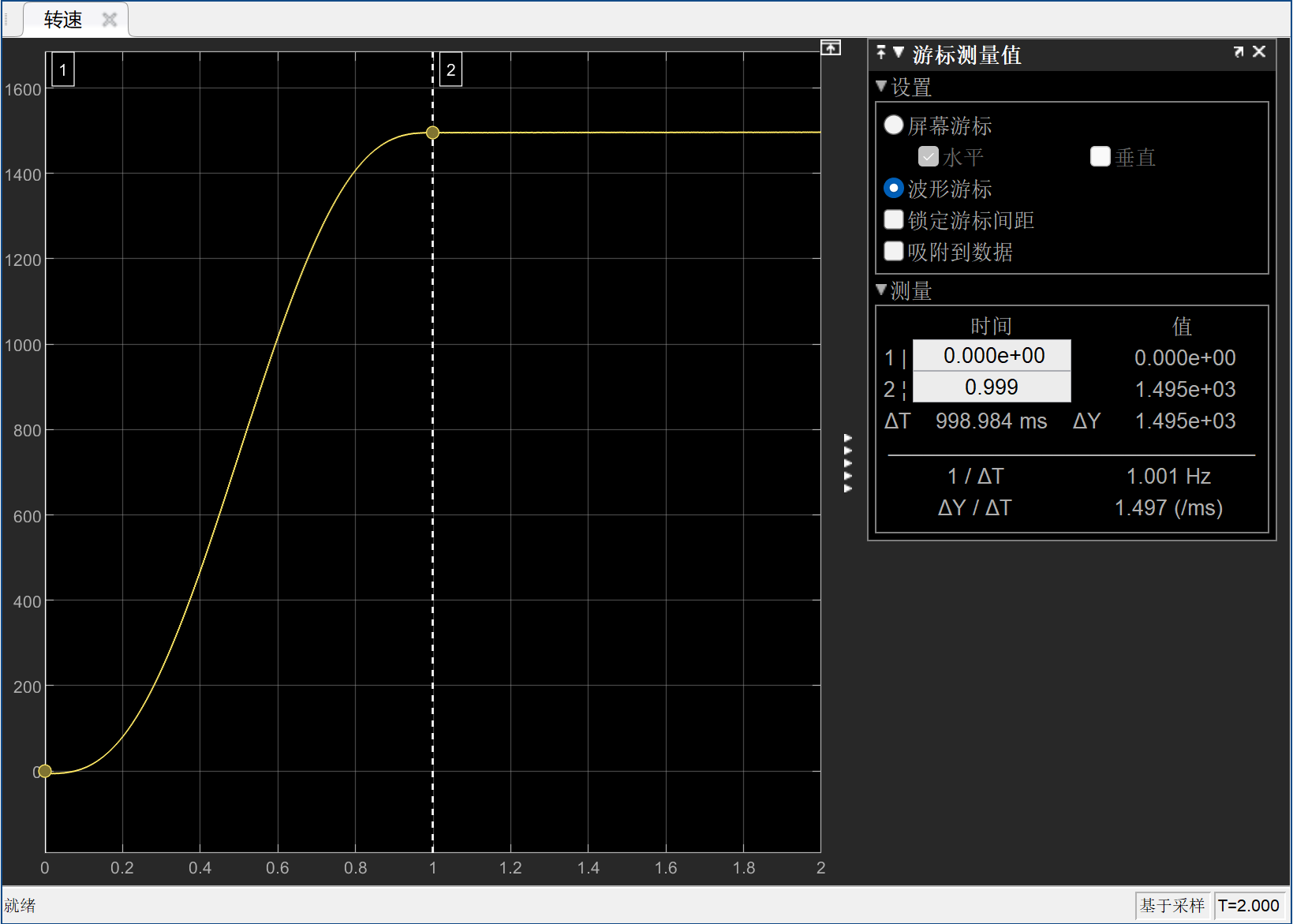
### 斜坡响应

上升时间：150.117ms，与给定一致



### S型曲线响应

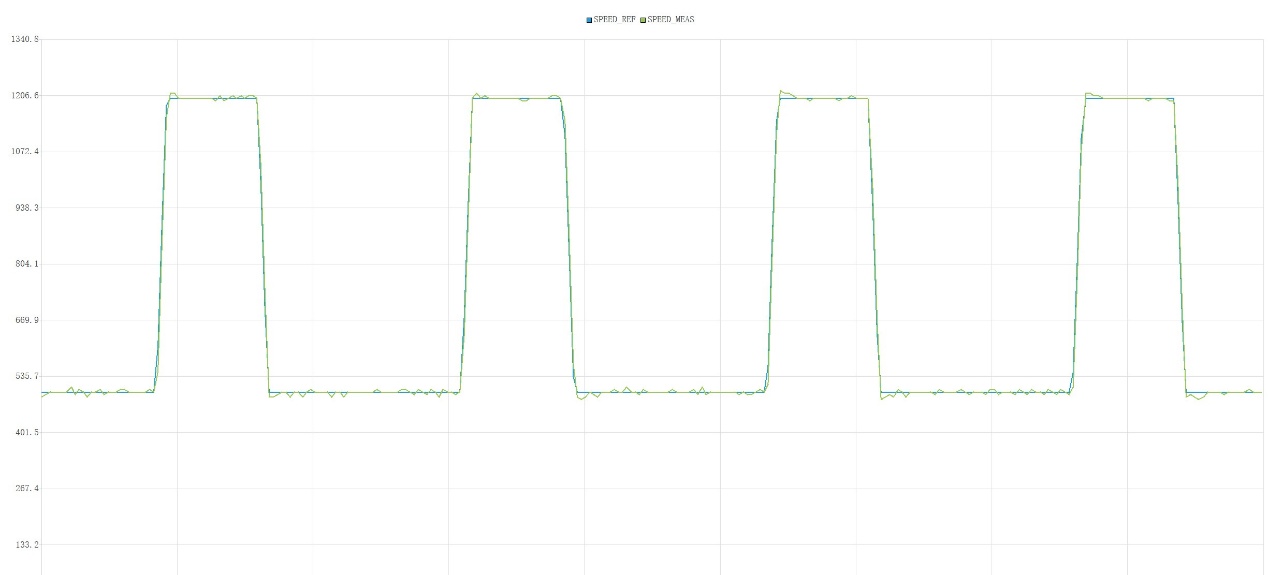
上升时间：998.984ms，与给定一致



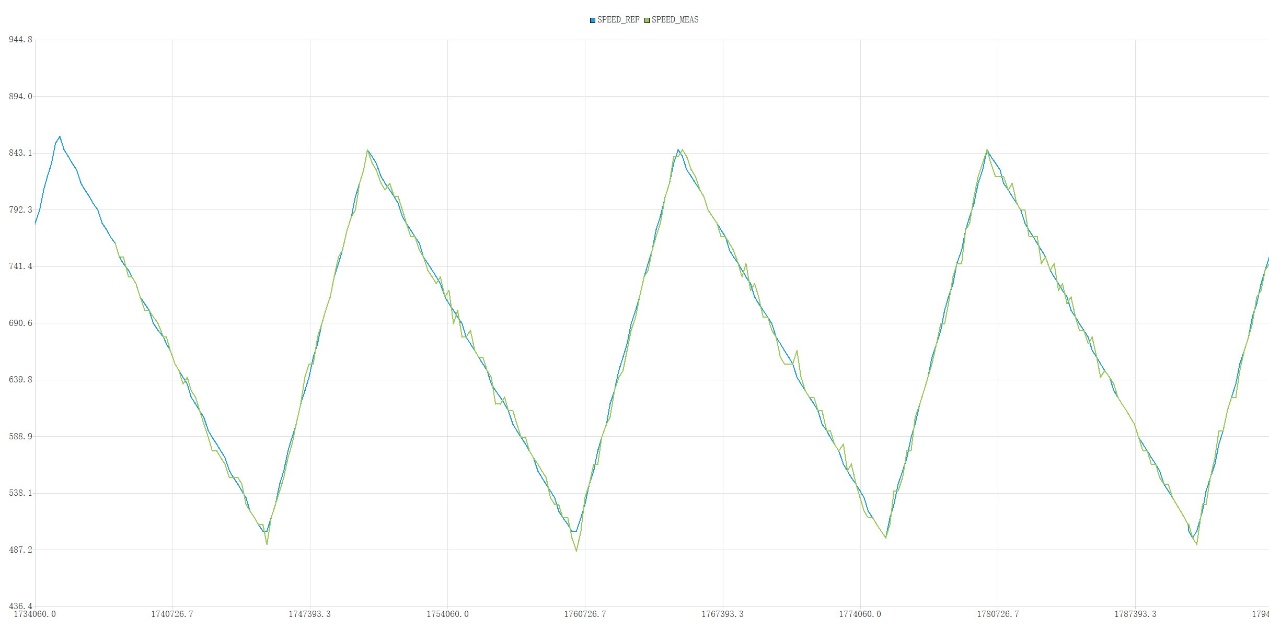
## 硬件实测数据

### 速度给定

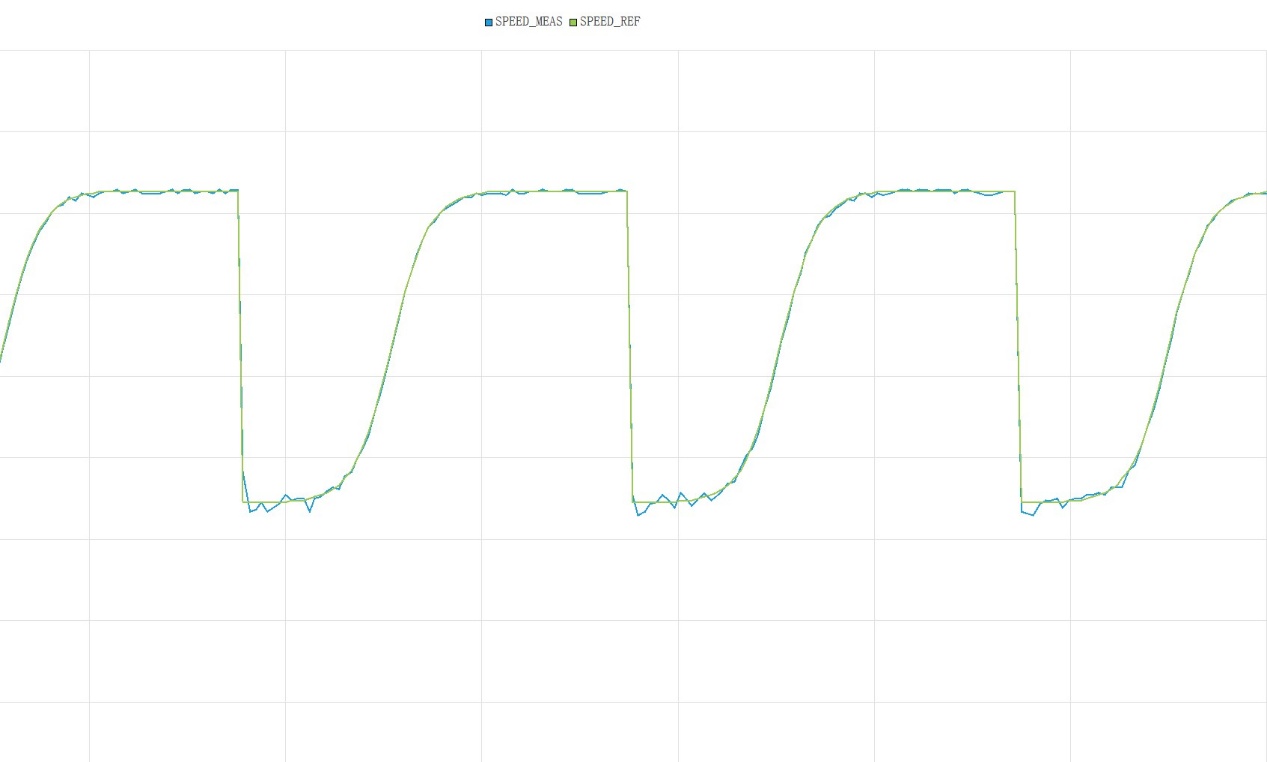
​​阶跃给定速度：

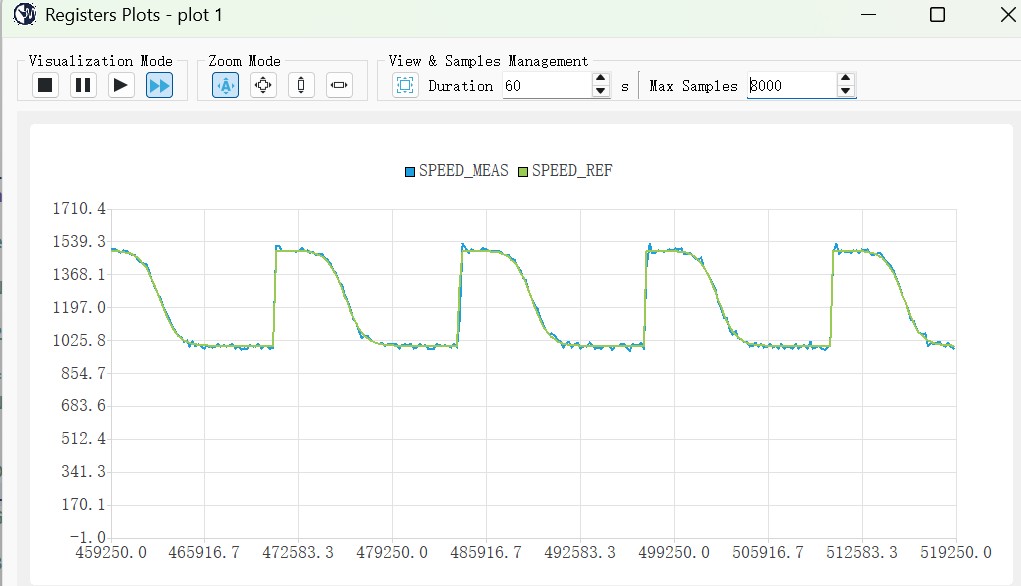


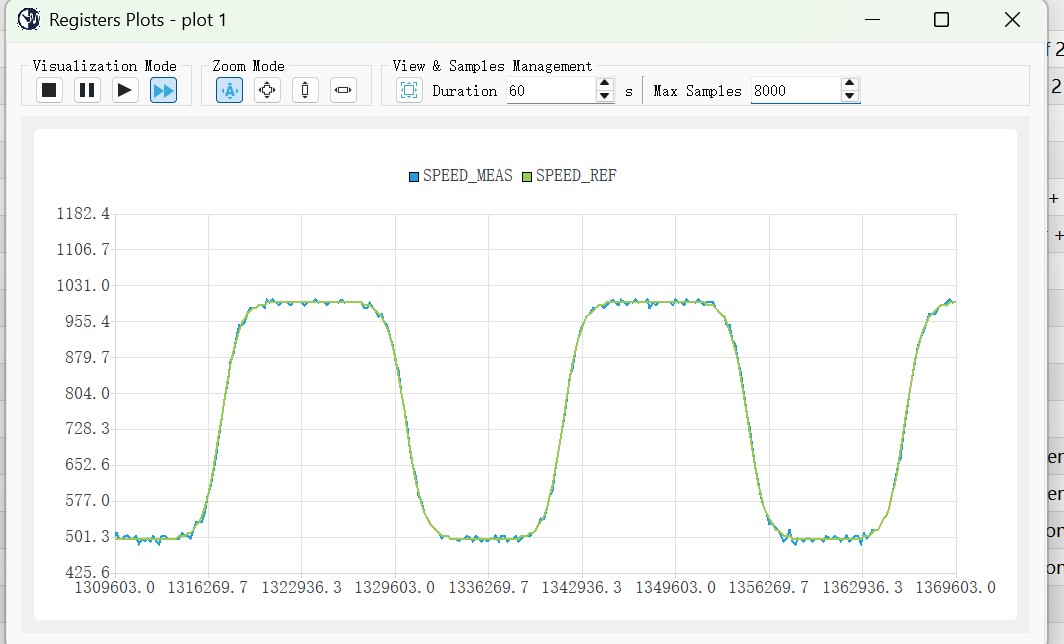
斜坡给定速度：



S曲线给定：







七、总结与展望

## 主要成果

成功在STM32G4平台实现FOC闭环控制，稳态精度达±1.5%

验证了PI参数自整定策略的有效性（响应速度vs超调量权衡）

MotorControl Workbench大幅降低开发难度（代码生成效率提升70%）

## 存在问题

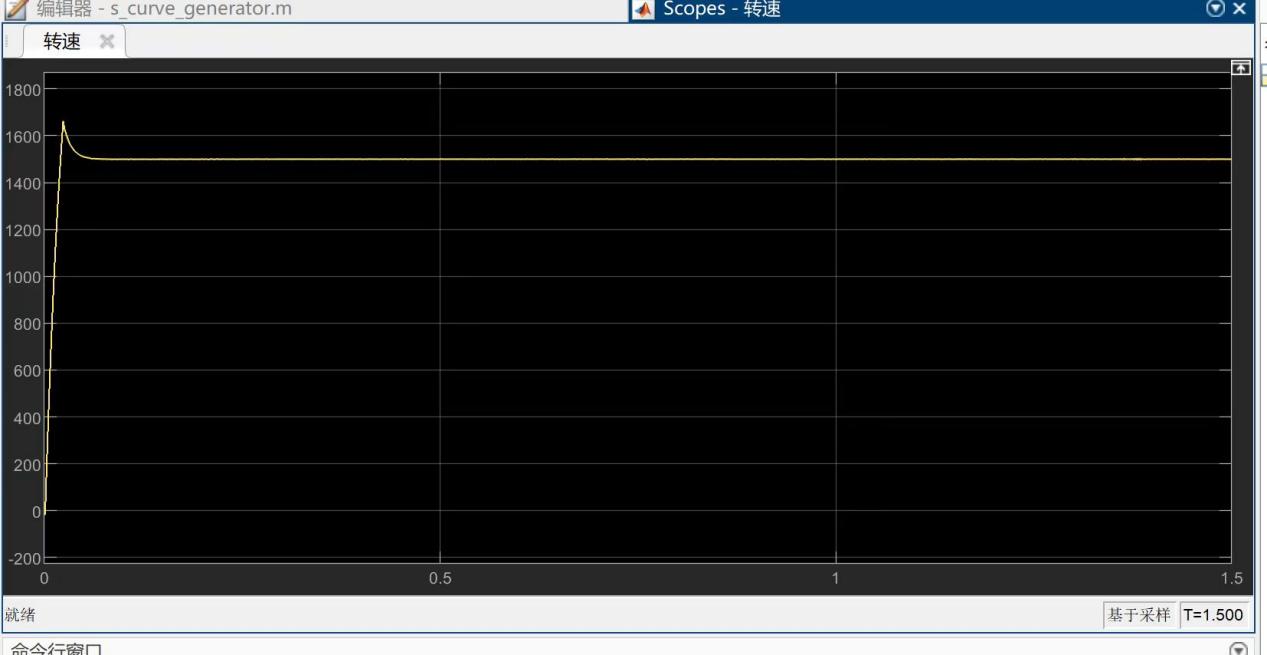
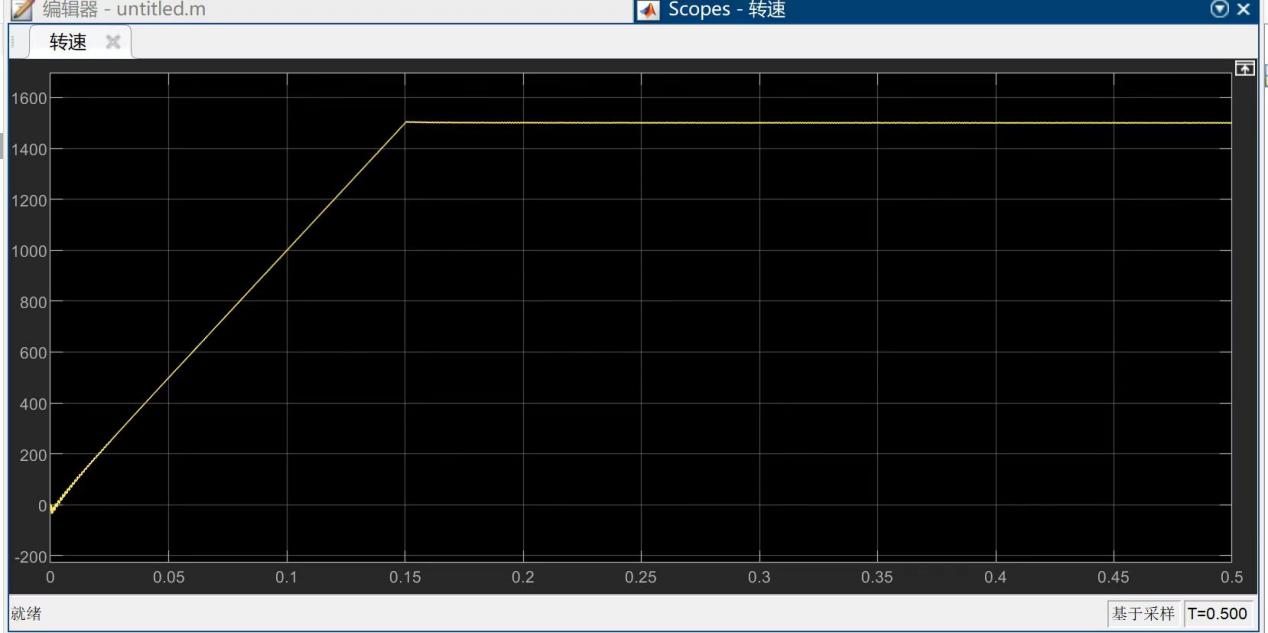
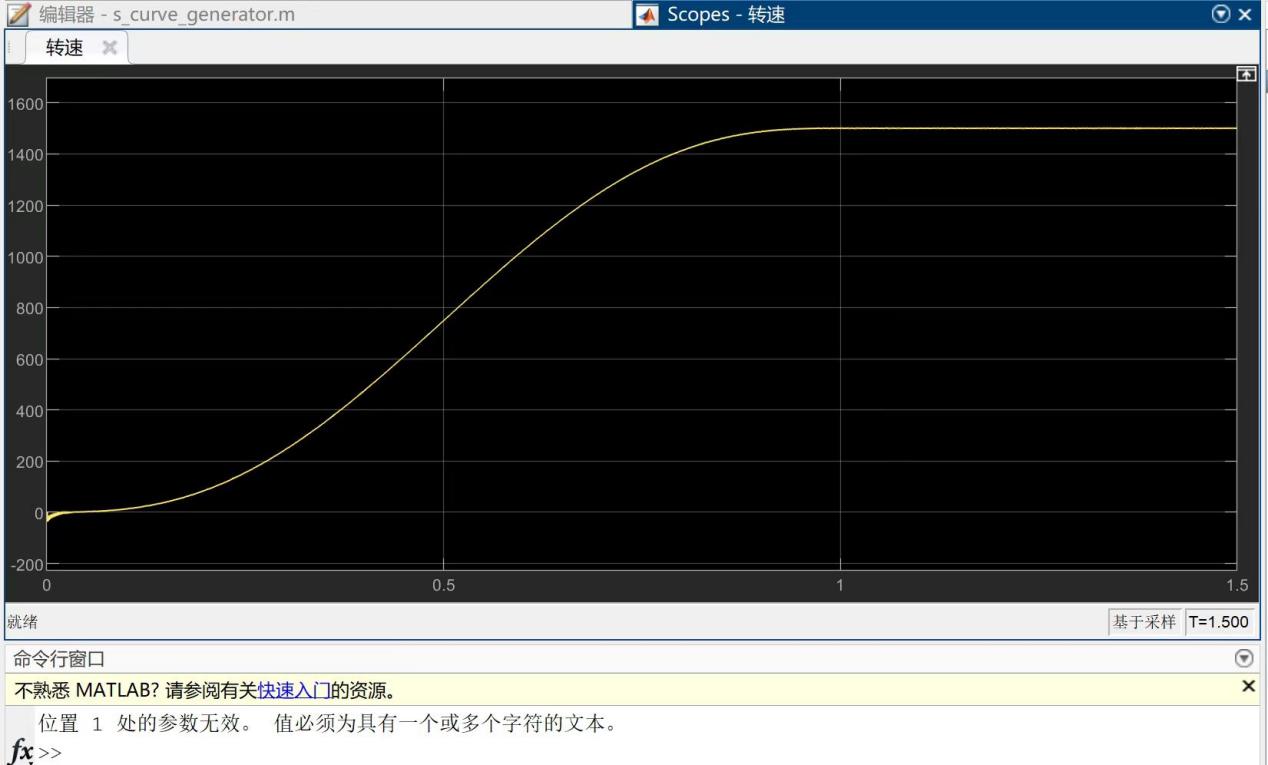
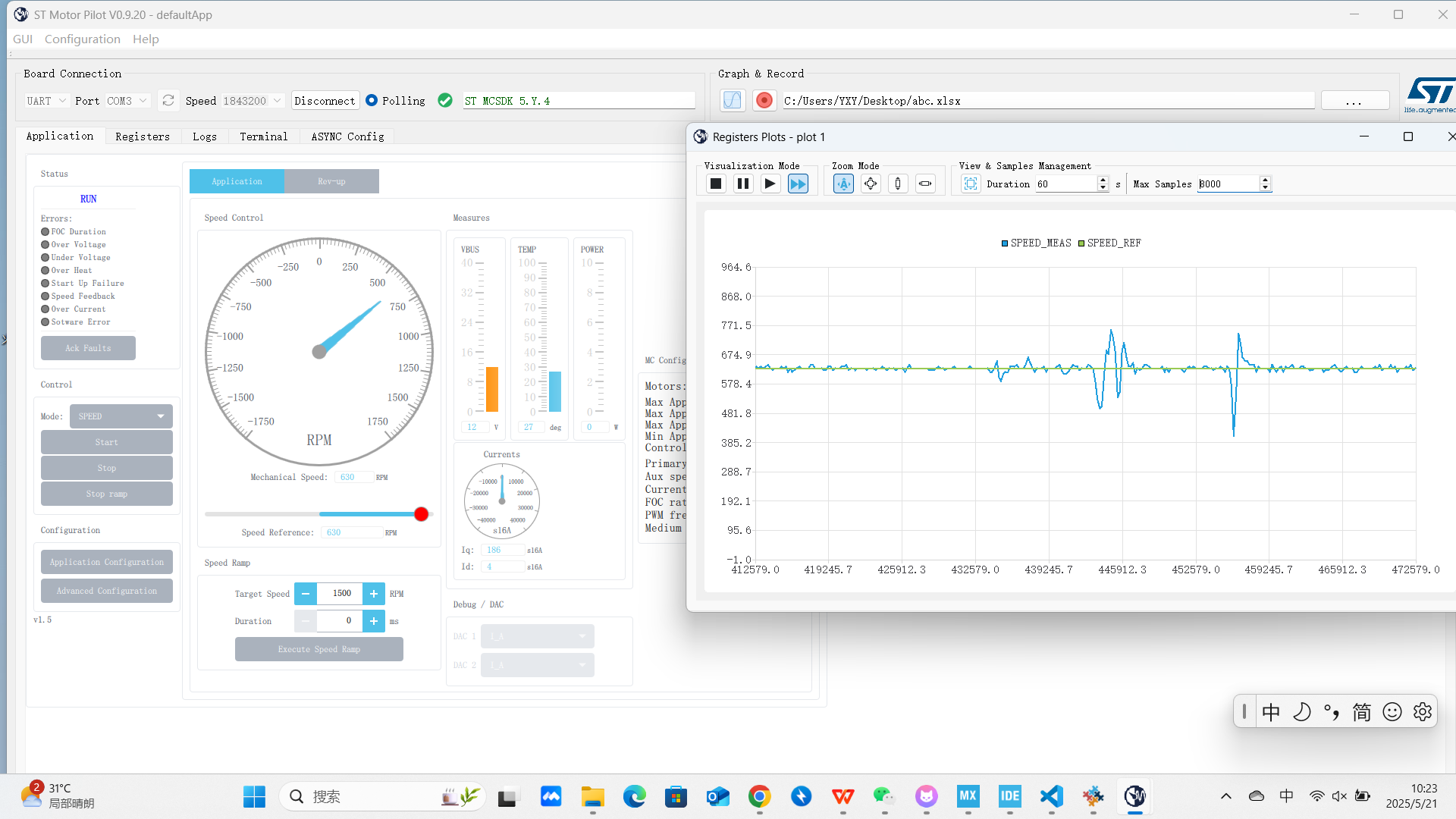
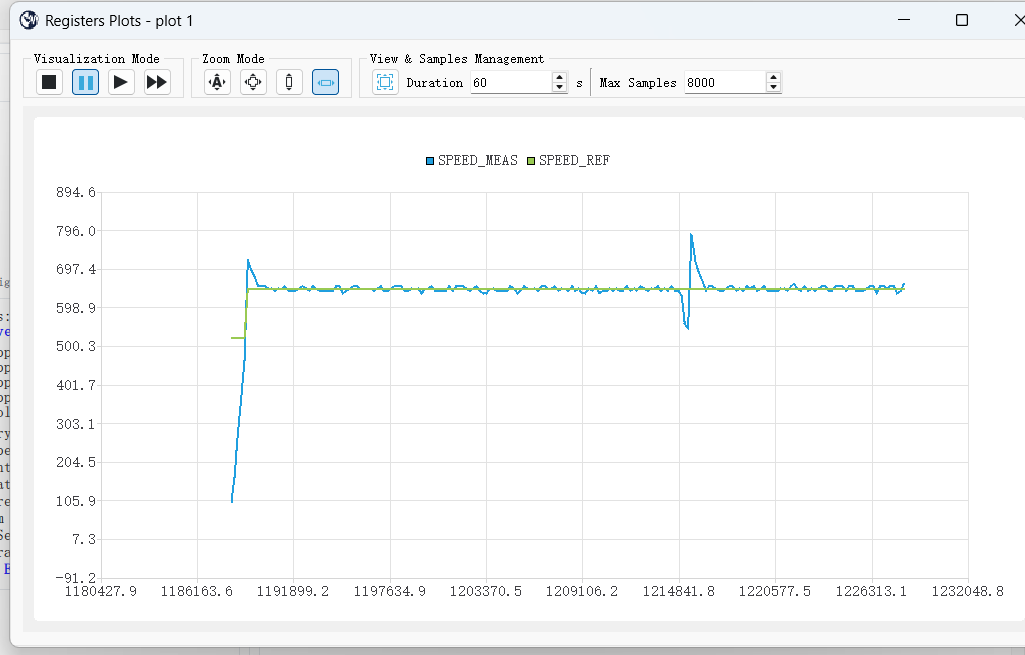
参数敏感性高：电机电阻温漂导致电流环需在线补偿；

无传感器模式低速性能不足。

八、课程设计分工

|  |  |
| --- | --- |
| 人员 | 分工 |
| 雒晋欣 | 三种速度给定方式的实现，速度调节器设计，报告编写 |
| 董晨晨 | Simulink仿真模型的各模块实现，报告编写 |
| 杨欣怡 | 速度调节器的参数整定，报告编写，制作PPT |

九、附件



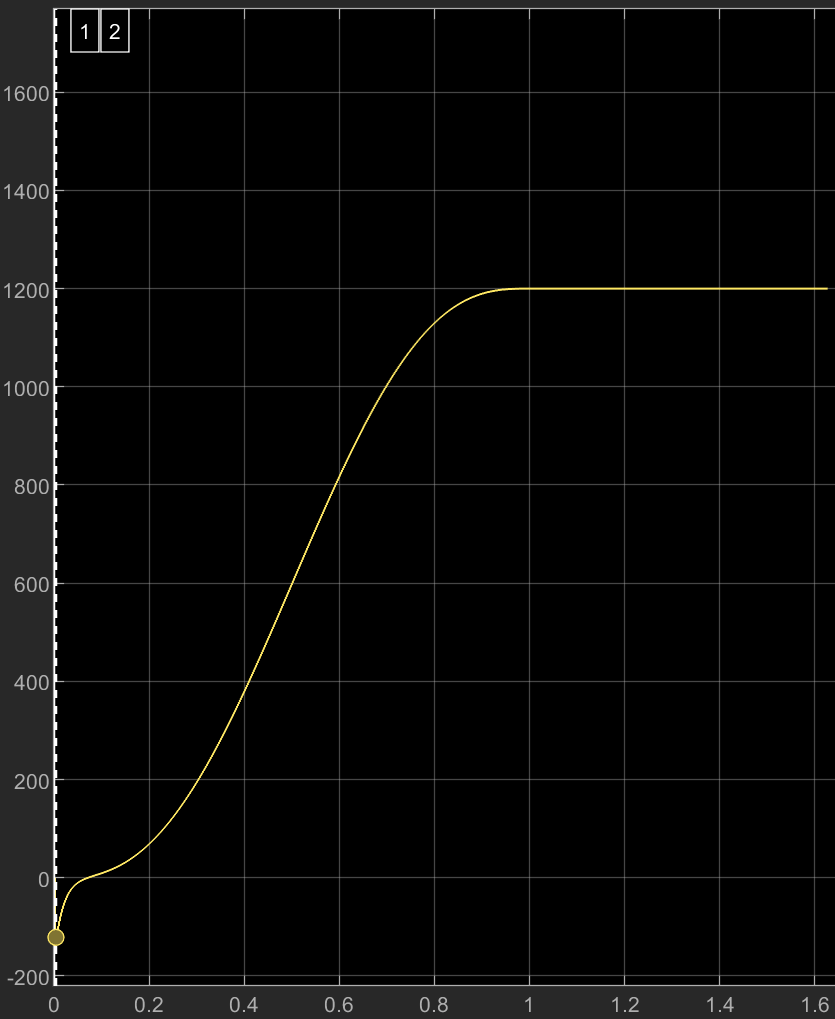
S型曲线生成函数对比

* 特性

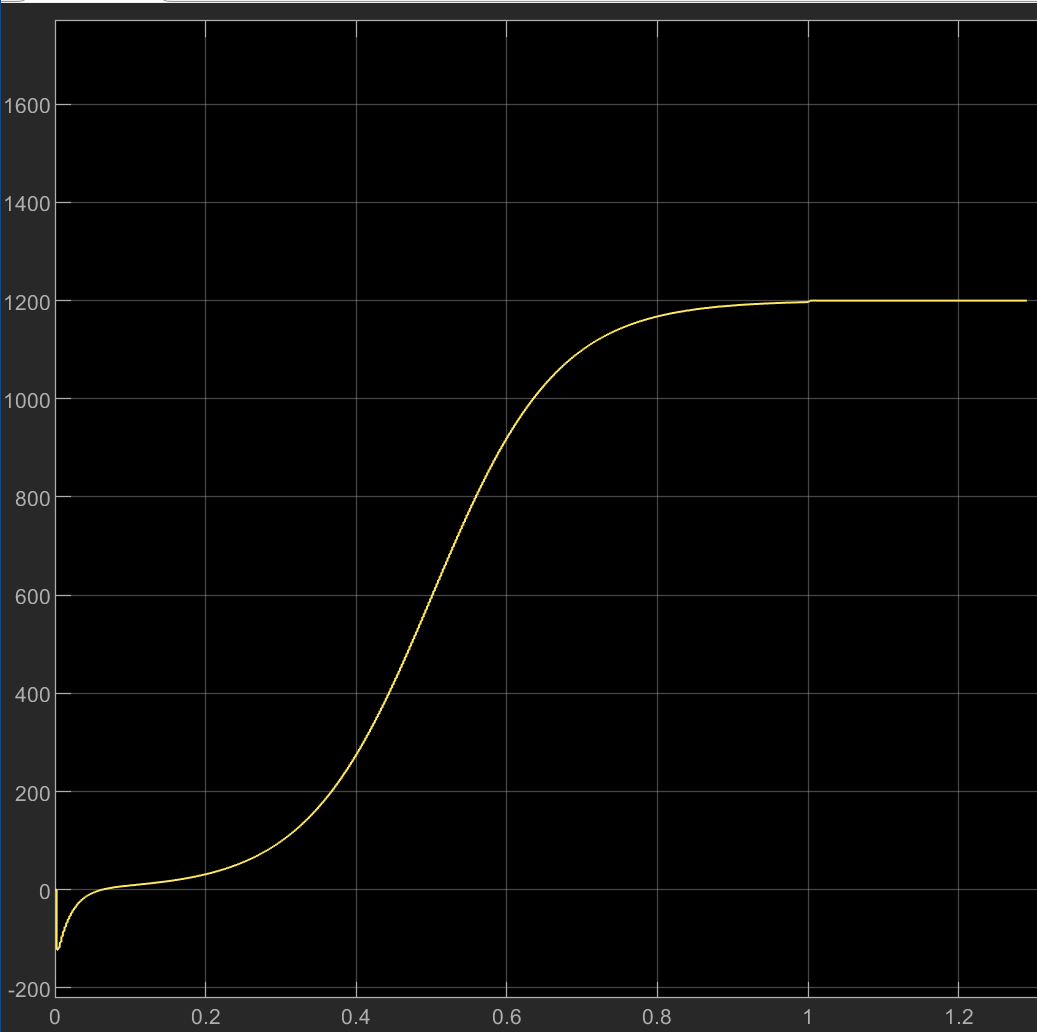
| ****特性**** | **​**​指数型​**​ (s\_curve\_exponential)** | **​**​5次多项式型​**​ (s\_curve\_generator)** | **​**​余弦过渡型​**​ (s\_curve\_cosine)** |
| --- | --- | --- | --- |
| **核心公式** | 1/(1+exp(-k\*(2t/t\_total-1))) | 10t³ - 15t⁴ + 6t⁵ | (1 - cos(π·t/t\_total))/2 |
| **曲线形状** | 最陡峭，中心对称 | 最平滑，完美的S型 | 平缓过渡，中部较平缓 |
| **端点特性** | 起点/终点变化率>0 | 起点/终点变化率=0 ​**​(Jerk连续)​**​ | 起点/终点变化率>0 |
| **可调参数** | 曲率系数k | 无额外参数 | 无额外参数 |
| **计算复杂度** | ★★☆ (需指数运算) | ★★ (多次幂运算) | ★ (简单三角计算) |

* 波形

5次多项式型(s\_curve\_generator)



指数型 (s\_curve\_exponential)



余弦过渡型(s\_curve\_cosine)

