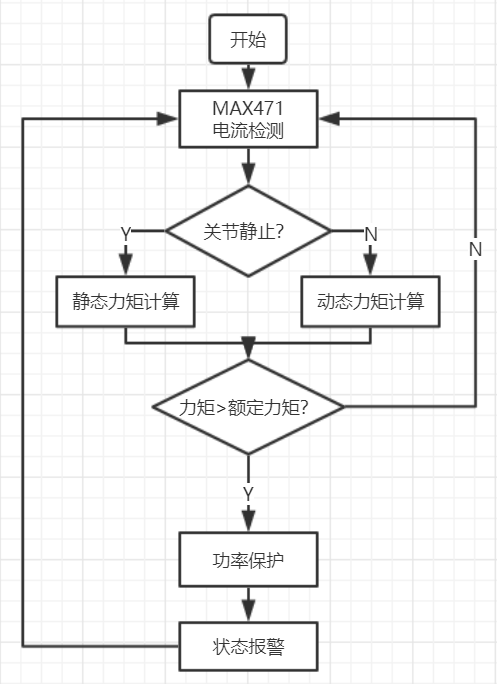
1. 基于电流反馈的关节力矩控制程序设计

在此攀爬仿生机器人的设计过程中，关节舵机作为机器人运动的主要动力部件常常出现受力较大的情况，而舵机是基于直流电机和减速齿轮组构成。由直流电机的工作原理可知，负载较大情况下，电机容易出现堵转现象，此时电机输出轴不转，电流通过电机线圈产生较大的堵转电流，非常容易使直流电机电刷烧毁或控制电路元器件烧毁。因此合理控制关节力矩，便可以有效的解决这个问题。

由电机工作原理可知，直接测量舵机输出轴力矩较为复杂，且需要给输出轴增加额外负载，无法在实际设计过程中应用。所以此攀爬放生机器人设计项目在硬件设计过程中，采用MAX471电流检测芯片，通过检测舵机工作电流间接检测舵机输出轴负载大小。然后再根据实验测试额定工作电流范围和实际工作状态电流反馈值进行程序设计，实现关节力矩控制，从而达到保护机器人关节机构，使其避免因过载而失效的问题。

而MAX471的采样原理与位置PID控制采样的原理类似，都是将被控对象（此处是电流大小）通过功能元器件及其适配电路转化成模拟量输出（电压值），再经STM32系列单片机的ADC采样功能，处理成程序计算数据。

此部分电流反馈的关节力矩控制程序设计主要考虑被控状态的不同状态，如舵机启动状态、角度维持状态、角度变换状态等等，将采样得到的电流参数I实际同原理实验得到的额定电流参数I额定进行对比，判断条件如下：

I实际 ≤ k\*I额定

其中k为短时容忍系数，即电路或电机在短时间内可以允许超过额定电流的系数。

经过比较后判断是否进行功率保护程序，与此同时若需要进行功率保护程序，则经负载超标状态进行上报，便于用户检查该关节具体情况。若力矩范围仍在而定范围内，则无需进行额外操作，完成单个控制周期。

1. 变量转换

如前文提到，在对关节力矩进行控制的过程中，由于力矩在机器人实际动态过程中是一个不易测量的物理量。但由于其力矩是由基于直流电机的伺服舵机输出的，而针对伺服舵机根据能量守恒有着近似的功率等效的关系，即：

功率输入 ≈ 功率输出

而用于力矩输出部分的功率为：

力矩输出功率 = 功率输出×机械功转换效率

所以我们可以根据以上公式，通过检测伺服舵机系统的输入功率（即输入电流）大小，来近似判断机器人关节上的负载力矩大小。

同理，我们在原理实验和程序设计的过程中，通过定参数实验，将设计额定力矩转化为电流参数，然后将其输入到程序中作为判断参数使用。所以在力矩控制的过程中，我们所达到的效果是控制关节舵机的输出力矩，使其在额定范围以保护机器人关节因过载而失效，而实质上，我们是通过变量转化的过程，以电流变量替代力矩变量，通过调节电流输入大小来实现对力矩的控制和保护。

1. 程序实现

同其他部分程序设计一样，此部分程序设计仍然是基于C语言的STM32库函数程序设计。其功能主要依赖于STM32系列单片机的ADC采样功能，所以我们在程序设计部分首先需要对单片机的ADC采样功能进行编程。

在功能配置之前，我们需要通过头文件对硬件模块和端口进行映射，指定我们使用的ADC模块编号以及其模块下GPIO所对应的通道号，MAX471.h文件编写如下：

#ifndef \_\_MAX471\_H\_\_

#define \_\_MAX471\_H\_\_ //驱动库去重复处理

#include "Global.h" //固件库头文件

#define ADC1\_PIN GPIO\_Pin\_4 //ADC采样功能端口号

#define ADC1\_GPIO GPIOA //ADC端口模块号

#define ADC1\_CH 4 //ADC采样通道编号

#define ADC1\_ADC ADC2 //ADC功能模块编号

void ADCIOInit(void); //ADC端口初始化程序声明

void ADCInit(void); //ADC功能配置初始化程序声明

u16 GetADCValue(void); //ADC采样函数声明

#endif

然后，我们在MAX471.c文件中，对ADC采样功能进行如下配置：

GPIO\_InitTypeDef GPIO\_InitStructure;

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_GPIOA, ENABLE);

GPIO\_InitStructure.GPIO\_Pin = ADC1\_PIN;

GPIO\_InitStructure.GPIO\_Mode = GPIO\_Mode\_AIN;

GPIO\_Init(ADC1\_GPIO, &GPIO\_InitStructure);

//GPIO初始化部分

ADC\_InitTypeDef ADC\_InitStructure;

//定义ADC功能结构体

RCC\_APB2PeriphClockCmd(RCC\_APB2Periph\_ADC2, ENABLE);

//打开ADC2所挂载的APB2时钟

ADC\_InitStructure.ADC\_Mode = ADC\_Mode\_Independent;

//配置ADC工作模式为ADC\_Mode\_Independent

ADC\_InitStructure.ADC\_ScanConvMode = DISABLE;

//关闭ADC单次扫描功能

ADC\_InitStructure.ADC\_ContinuousConvMode = ENABLE;

//打开ADC循环扫描功能

ADC\_InitStructure.ADC\_ExternalTrigConv = ADC\_ExternalTrigConv\_None;

ADC\_InitStructure.ADC\_DataAlign = ADC\_DataAlign\_Right;

ADC\_InitStructure.ADC\_NbrOfChannel = 16;

//配置ADC通道编号

ADC\_Init(ADC1\_ADC, &ADC\_InitStructure);

//初始化ADC结构体参数

ADC\_Cmd(ADC1\_ADC, ENABLE);

//打开ADC功能

ADC\_ResetCalibration(ADC1\_ADC);

//重置ADC

while(ADC\_GetResetCalibrationStatus(ADC1\_ADC));

//等待ADC功能就绪

ADC\_StartCalibration(ADC1\_ADC);

while(ADC\_GetCalibrationStatus(ADC1\_ADC));

ADC\_SoftwareStartConvCmd(ADC1\_ADC, ENABLE);

//启动ADC功能

至此，程序上完成了对单片机ADC功能的参数配置和初始化的过程，通过调用ADC采样函数，便可以获取当前伺服舵机系统所对应的电流输入参数。

1. 问题分析

同PID控制系统类似，系统采样存在其精度问题。此部分对电流采样仍为STM32系列单片机所提供的12位ADC采样精度。因此其电流的采样精度i电流计算公式如下：

i电流 = Imax /4096 单位：A

但此部分相较于PID控制系统的精确控制来说属于较为粗略的通过阈值比较的范围控制，主要起到保护功能，所以对采样精度没有较高的要求。其采样周期近似等于舵机的控制周期即可。