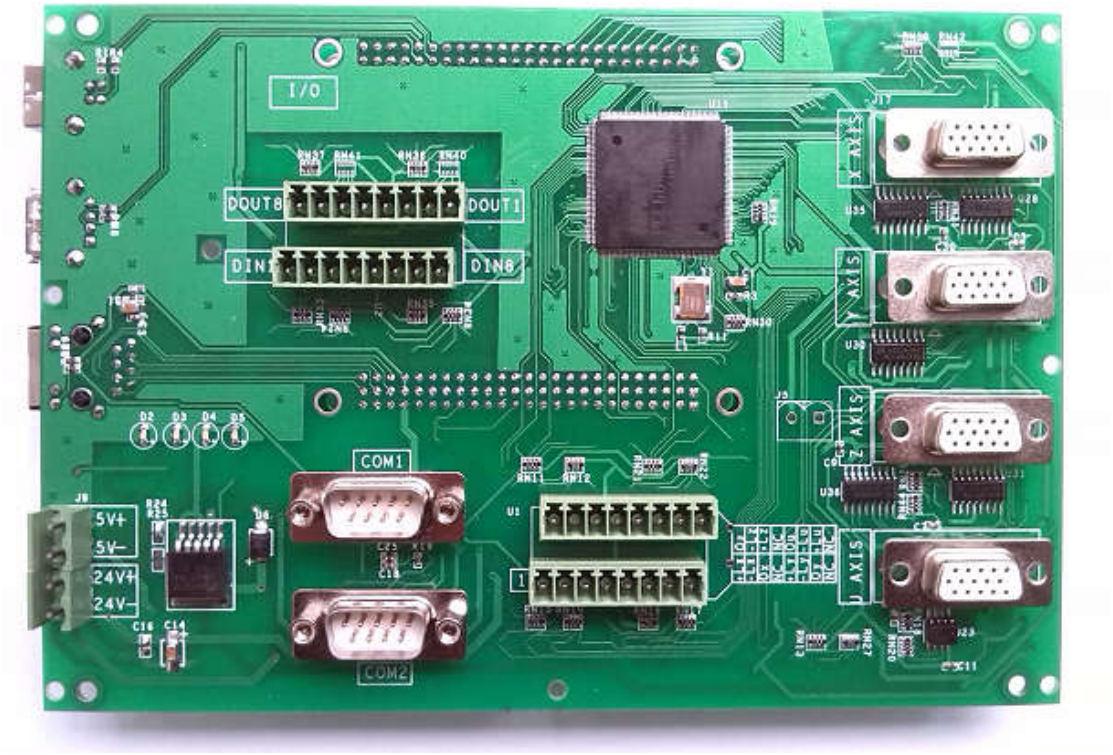
STM0408 使用说明书



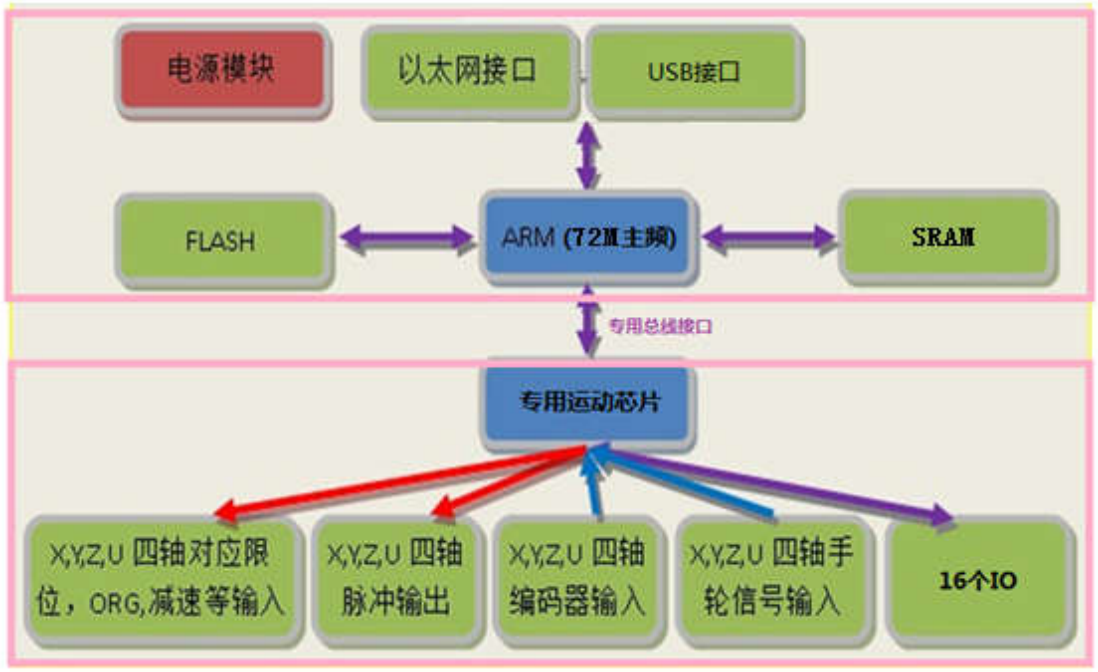
目录

一.简介

STM0408 型号解释：

- ETH6045　　控制器类型（系列）
- STM　　控制器内部架构为 STM 架构
- 0416　　'04'标识为 4 轴，'16'标识输入输出各 16 个，
  0408　　'04'标识为 4 轴，'8'标识输入输出各 8 个

STM0408 型运动控制器，是嵌入式 CPU（ARM 架构）与运动控制器相结合，软硬件一体化方案，软件操作系统采用 **KEILC，C 语言编程，提供 DEMO 源码和二进制 lib 库**，硬件采用 32 位 ARM 芯片 CORTEX-M3 架构与板载专用运动芯片方案。



嵌入式 ARM 处理器参数：

- 处理器主频：72M Hz
- 内存：　64KB 内置，可外扩 SRAM
- 闪存：512KB 内置，可外扩 NAND FLASH,SPI FLASH
- 16 个通用输出，16 个通用输入，全光耦隔离
- 2 个串口，2 个 USB 接口
- 1 个手轮接口
- 1 个 100M 网口

运动控制器（专用运动芯片）参数：

控制轴数： 4 轴

脉冲输出最大频率：6.5Mbps

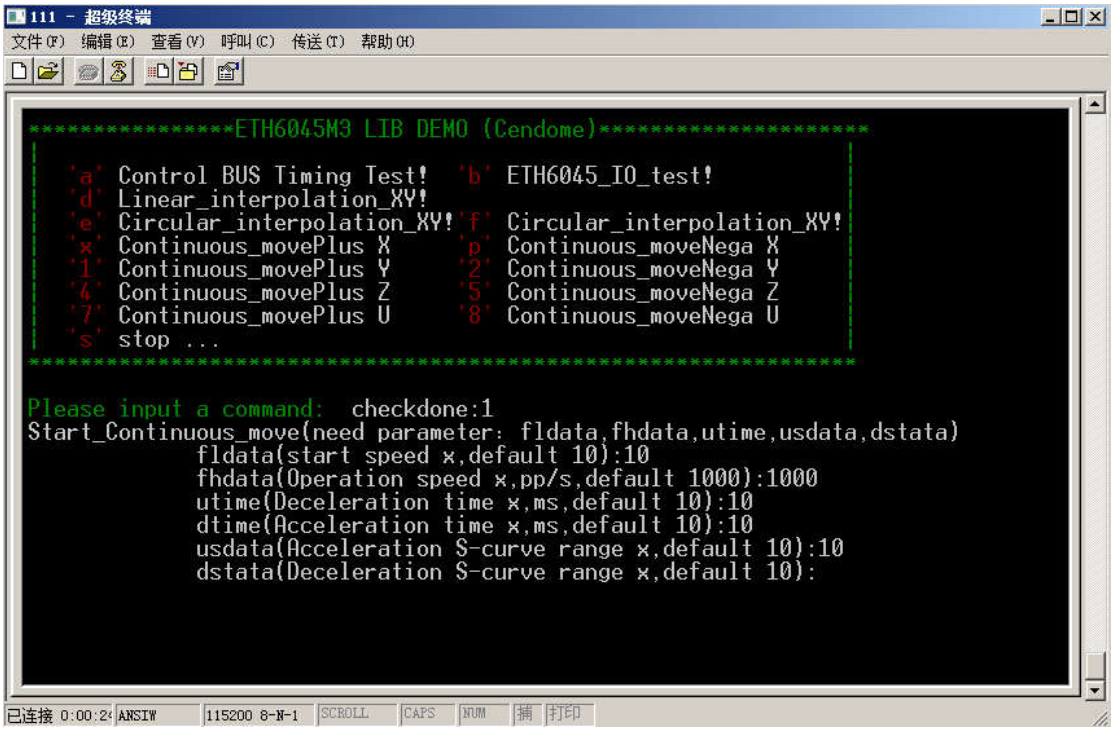
加减速：支持 S 曲线加减速和 T 型加减速

插补控制：任意 2～4 轴直线插补、圆弧插补

其他特性：

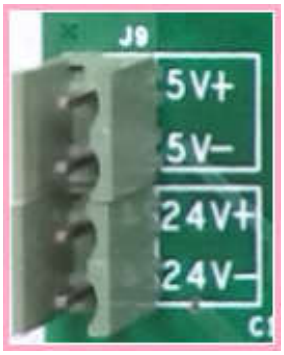
- 带 4 个编码器计数器可用于实现全闭环或“速度模式（脉冲方向）”
- 动作中的速度以及目标位置变更
- 4 个轴可以同时做 2 组不同的直线运动或 1 组直线运动+1 组圆弧运动
- 脉冲方向差分输出

- ETH6045m3-demo(售后提供源码)是本公司针对该运动控制器开发的应用开发调试软件，用于辅助用户加快运动控制应用的开发过程，该软件都能在静态链接函数库（M3_lib.lib）中找到对应的函数。



二.接口说明

电源接口



控制器由**隔离**的直流 5V（不小于 15 瓦）和 24V（不小于 100 瓦）电源供电。推荐电源品牌：台湾明纬.

以太网口

以太网口速度为 100M，满足各种应用。

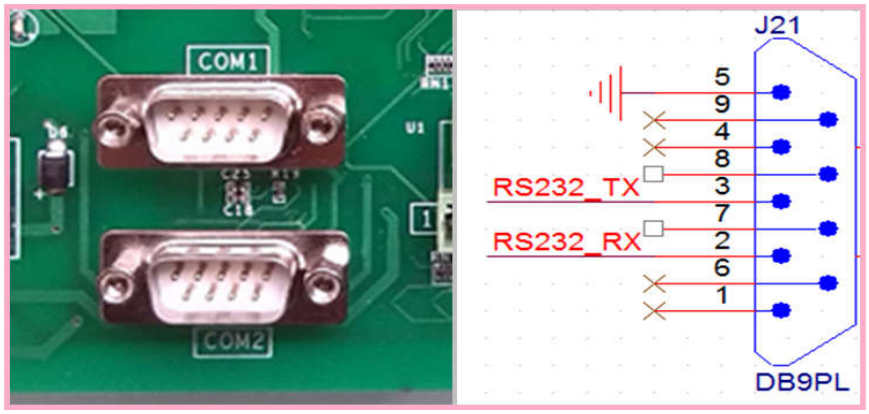
USB 主口

用于连接 U 盘，USB 鼠标，USB 键盘。扩展控制器的相关功能。

USB 从口

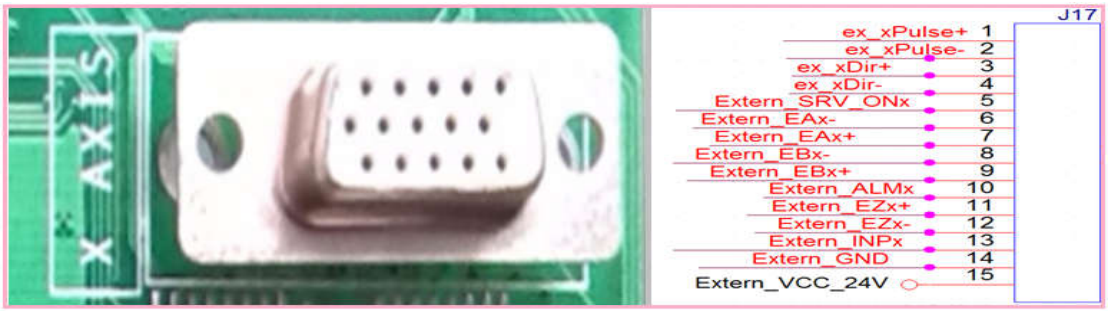
用于连接 PC 机。

RS232 串口



RS232 串口（三线串口）用于连接其他串口模块或 PC 电脑，作为扩展接口使用。与 PC 电脑相连时使用串口交叉线。

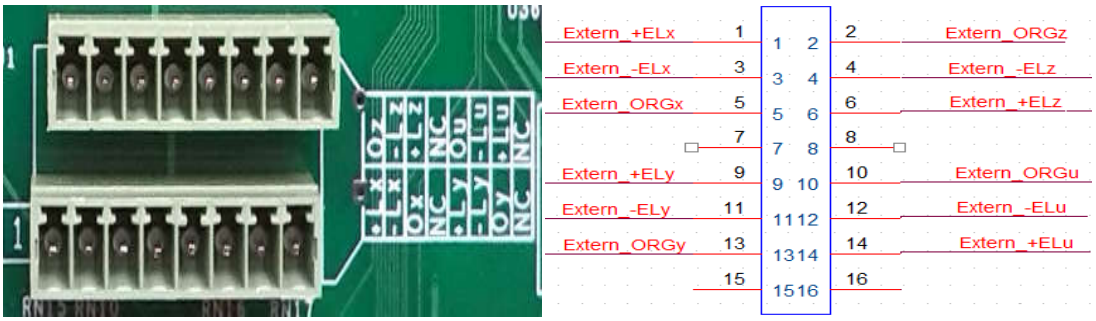
四个轴脉冲接口



该接口使用三排 DB15 接口。引脚定义如图。

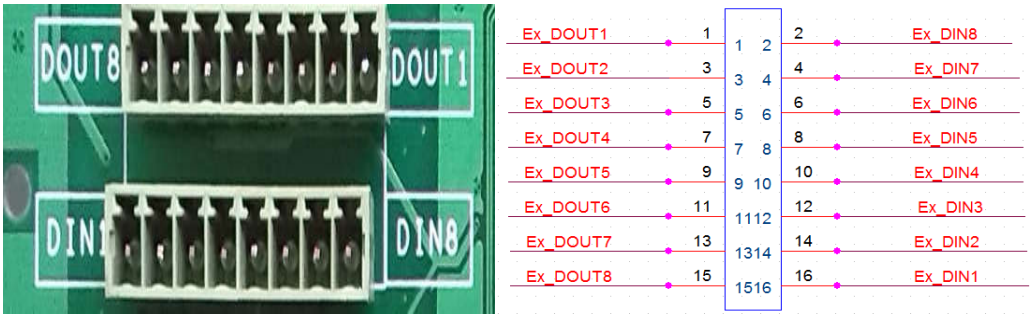
引脚编号	引脚名称	方向	备注
1	ex_xPulse+	输出	脉冲+输出，接入电机驱动器
2	ex_xPulse-	输出	脉冲-输出，接入电机驱动器
3	ex_xDir+	输出	方向+输出，接入电机驱动器
4	ex_xDir-	输出	方向-输出，接入电机驱动器
5	Extern_SRV_ONx	输出	伺服使能，用于接入电机驱动器
6	Extern_EAx-	输入	伺服驱动器编码器输出的 A-
7	Extern_EAx+	输入	伺服驱动器编码器输出的 A+
8	Extern_EBx-	输入	伺服驱动器编码器输出的 B-
9	Extern_EBx+	输入	伺服驱动器编码器输出的 B+
10	Extern_ALMx	输入	伺服驱动器输出的报警
11	Extern_EZx+	输入	伺服驱动器编码器输出的 Z+
12	Extern_EZx-	输入	伺服驱动器编码器输出的 Z-
13	Extern_INPx	输入	伺服驱动器编码器输出的到位信号
14	Extern_GND	输出	24V-输出，用于接入伺服驱动器
15	Extern_VCC_24V	输出	24V+输出，用于接入伺服驱动器

四个轴限位原点接口

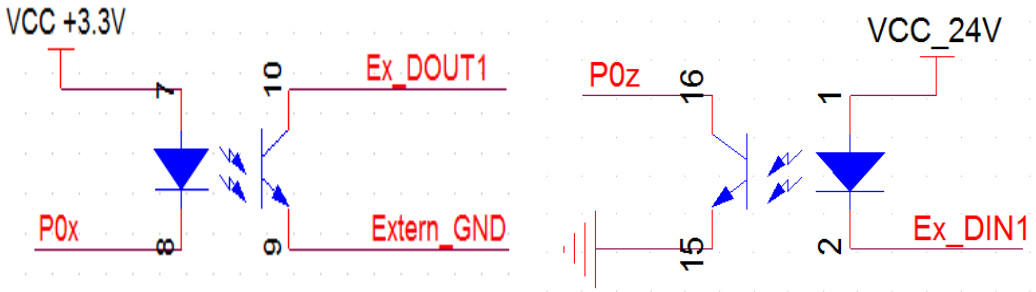


四个轴 XYZU，每个轴 3 个信号：正限位(Extern_+ELx)，负限位(Extern_-ELx)，原点(Extern_+ORGx)。使用 3.81mm 间距端子排座。

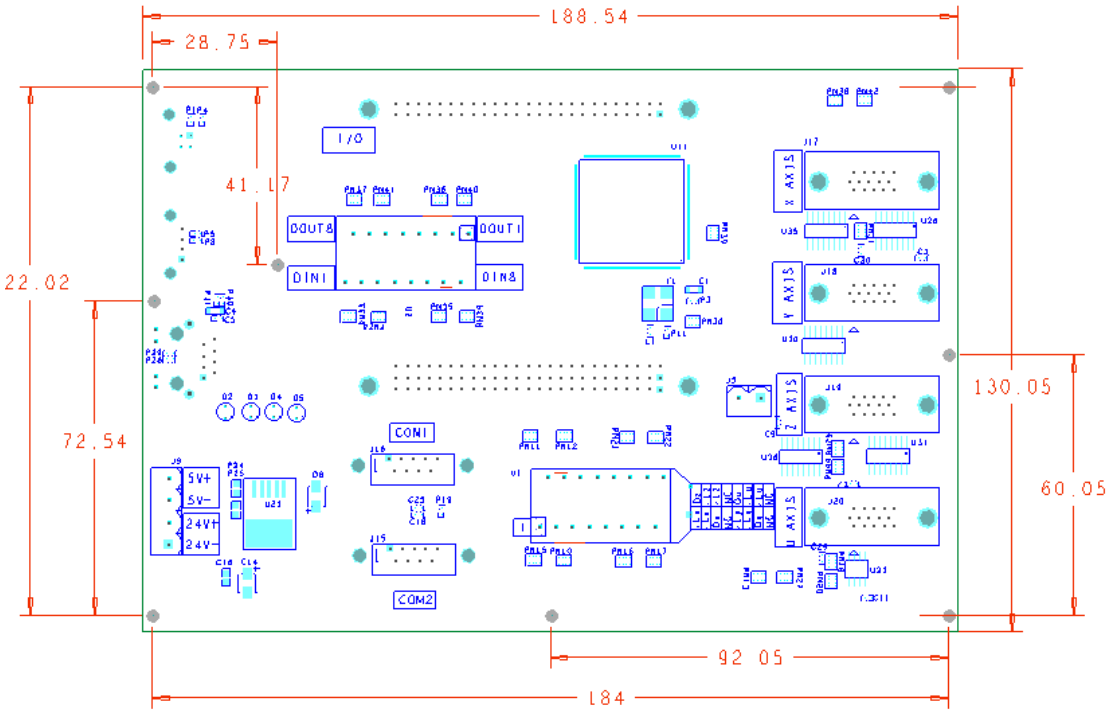
输入输出接口



输入输出各 8 路，全光耦隔离，示意图如下。使用 3.81mm 间距端子排座。



三.尺寸(单位：mm)



如上图 PCB 板尺寸为 188.54 x 130.05mm。

四. 二次开发 LIB 库函数

为二次开发提供的 dll 库可以用于 VB,VC 的工程中。函数列表如下

函数名称	说明
C45CE_CardInit	控制器初始化，必须要调用
C45CE_CardClose	
C45CE_set_pulse_outmode	
C45CE_set_profile	
C45CE_pmove	定长运动函数
C45CE_check_done	
C45CE_change_speed	
C45CE_reset_target_position	
C45CE_vmove	连续运动函数
C45CE_decel_stop	减速停止函数
C45CE_sudden_stop	立即停止函数
C45CE_set_HOME_PIN_logic	
C45CE_config_home_mode	
C45CE_home_move	回原点
C45CE_line2	
C45CE_line3	

C45CE_line4	
C45CE_axis_io_status	
C45CE_Get_DebugStatus	
C45CE_get_position	
C45CE_set_position	
C45CE_rel_arc_move	
C45CE_set_backlash	
C45CE_write_outbit	
C45CE_read_inbit	
C45CE_handwheel_move	
C45CE_counter_config	
C45CE_get_encoder	
C45CE_set_encoder	
C45CE_Bline2	
C45CE_set_AGPIO	

```
//! 初始化函数，使用其它函数之前调用
//! 返回值：
//!            0       ------       成功
//!            1       ------       没找到控制卡(无运动器件)
EXTERN_C int STDCALLDEF C45CE_CardInit(void);/*= NULL*/
EXTERN_C int STDCALLDEF C45CE_CardClose(int iLaserOnDelayUs);
// 脉冲输出模式设置 ，该函数如果不调用outmode=0 ，如果要某轴反向
outmode=2。
//! 脉冲/方向 ,还是双脉冲，脉冲/方向模式 可以通过设置参数来反向
//! 参数：axis 取值0,1,2,3 分别对应X,Y,Z,U轴
//!          outmode 取值0～5 与雷泰相同含义
//! 返回值：
//!            0       ------       成功
//!            1       ------       不支持的模式
//!            2       ------       axis 参数输入不正确
//!
EXTERN_C int STDCALLDEF C45CE_set_pulse_outmode(u16 axis, u16
outmode );

//梯形速度曲线设置函数
//! 功能：设定梯形速度曲线的起始速度、运行速度、加速时间、减速时间
//! 参数：axis： 轴号,取值0,1,2,3 分别对应X,Y,Z,U轴
//!       Max_Vel：运行速度，或简称为高速 ,单位pps
//!       acc： 加速时间，单位秒
//!       dec： 减速时间，单位秒
//!       BacklashSpeed： 间隙补偿速度，单位pps
//! 返回值：
```

```
//!              0        ------        成功
//!              1        ------        不支持的模式
//!              2        ------        axis 参数输入不正确
//EXTERN_C int STDCALLDEF C45CE_set_profile(u16 axis, UINT32 Max_Vel,
double acc, double dec);
EXTERN_C int STDCALLDEF C45CE_set_profile(u16 axis, UINT32 Max_Vel,
double acc, double dec,UINT32 BacklashSpeed); //=500

//梯形速度曲线 位移控制函数(定长运动)
//! 功能：让指定轴作点位运动
//! 参数：    axis：  轴号,取值0,1,2,3 分别对应X,Y,Z,U轴
//!        Dist：运动距离（Distance）
//!        posi_mode：坐标模式，相对位移为 0，绝对位移为 1。
//!
//! 返回值：
//!              0        ------        成功
//!              1        ------        不支持的模式
//!              2        ------        axis 参数输入不正确
EXTERN_C int STDCALLDEF C45CE_pmove(u16 axis, int Dist, u16 posi_mode );

//!功能：检测指定轴的运动状态，是运行还是停止。
//!参数：axis：轴号,取值0,1,2,3 分别对应X,Y,Z,U轴
//!返回值：
//!            0 表示指定轴正在运行
//!            1 表示指定轴停止运行
//!            2 表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_check_done(u16 axis);

//!功能：单轴运行中改变当前运行速度(速度正在改变时不要调用此函数)
//!          当指定轴在作连续运动时，调用此函数可以改变当前的运动速度，并
立即按所指定的
//!          速度连续运行
//!参数：
//!        Axis：轴号
//!        Curr_Vel：新的运动速度
//!返回值：
//!            0 表示成功
//!            1 无意义
//!            2 表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_change_speed(u16 axis,U32 Curr_Vel);

//!功能：改变目标位置
//!          在运动中改变目标位置。若目标位置比当前位置远，则继续向前运动，
到达新目标位置
```

```
//!          后，停止脉冲输出；若目标位置比当前位置近，控制卡将先停止当前运动，然后向反方
//!          向运动至目标位置
//!参数：
//!       axis：轴号,取值0,1,2,3  分别对应X,Y,Z,U轴
//!       dist：新的目标位置值
//!返回值：
//!          0  表示成功
//!          1  无意义
//!          2  表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_reset_target_position(u16 axis,int dist);


//!功能：单轴连续运动
//!          让指定轴加速到指定的运行速度后，连续运行。
//!参数：
//!       axis：轴号,取值0,1,2,3  分别对应X,Y,Z,U轴
//!       dir：指定运动的方向，其中  0  表示负方向，1  表示正方向
//!返回值：
//!          0  表示成功
//!          1  表示方向参数错误
//!          2  表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_vmove(u16 axis,u16 dir);
//!功能：减速停止
//!          指定轴减速停止。调用此函数后立即减速，到达起始速度后停止
//!
//!
//!参数：
//!       axis：轴号,取值0,1,2,3  分别对应X,Y,Z,U轴
//!返回值：
//!          0  表示成功
//!          1  无意义
//!          2  表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_decel_stop(u16 axis);

//!功能：立即停止
//!          指定轴立即停止。调用此函数后立即停止
//!
//!
//!参数：
//!       axis：轴号
//!返回值：
//!          0  表示成功
//!          1  无意义
```

```
//!            2  表示轴参数错误
EXTERN_C     int STDCALLDEF C45CE_sudden_stop(WORD axis);


//!功能：设置原点信号有效电平
//!
//!参数：
//!        axis：轴号,取值0,1,2,3  分别对应X,Y,Z,U轴
//!        org_logic：原点信号的有效电平,
//!                        0—低电平有效
//!                        1—高电平有效
//!返回值：
//!            0  表示成功
//!            1  表示org_logic无效
//!            2  表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_set_HOME_PIN_logic(u16 axis,u16
org_logic);

//!功能：设定回原点模式
//!        提供了多种不同的回原点模式，实现精确定位到原点的方案，
//!        通过调用此函数便可以选择其中一种模式。
//!参数：
//!        axis：轴号,取值0,1,2,3  分别对应X,Y,Z,U轴
//!        home_dir  回零方向，  1  正向, 2:负向
//!        vel  回零速度     (这个不起作用) pps
//!        mode  回原点的信号模式
//!             1 -  一次回零
//!             2 -  二次回零
//!             3 -  一次回零加回找  （不支持）
//!             10  -  以  EZ  作为原点进行一次回零
//!             11  -  以  EZ  作为原点进行一次回零，碰到限位后自动反找。
//!返回值：
//!            0  表示成功
//!            1  表示home_dir,mode无效
//!            2  表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_config_home_mode(u16 axis,u16 home_dir,
U32 vel, u16 mode);

//!功能：回原点
//!
//!参数：
//!        axis：轴号,取值0,1,2,3  分别对应X,Y,Z,U轴
//!返回值：
//!            0  表示成功
```

```
//!            1 无意义
//!            2 表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_home_move(u16 axis);

//!功能：二轴直线插补
//!        让指定的两轴作对称的梯形加减速插补运动。当 posi_mode 为 0 时，作相对位移运动，
//!运动方向由 Dist 的正负值确定；为 1，作绝对位移运动，运动方向由 Dist 与当前位置
//!的差值决定。
//!参数：
//!        axis1，2：第一、二轴轴号
//!        Dist1，Dist2：第一、二轴距离
//!        posi_mode：位置模式，  (不起作用，只能实现绝对位移)
//!        0－相对位移
//!        1－绝对位移
//!返回值：
//!            0 表示成功
//!            1 表示posi_mode错误
//!            2 表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_line2(u16 axis1,int Dist1,u16 axis2,int Dist2,u16 posi_mode);

//!功能：指定任意三轴做直线插补运动
//!        让指定的三轴作对称的梯形加减速插补运动。当 posi_mode 为 0 时，作相对位移运动，
//!运动方向由 Dist 的正负值确定；为 1，作绝对位移运动，运动方向由 Dist 与当前位置
//!的差值决定。
//!参数：
//!        axis 轴号列表的指针
//!        Dist1 指定 axis[0]轴的位移值，单位：脉冲数
//!        Dist2 指定 axis[1]轴的位移值，单位：脉冲数
//!        Dist3 指定 axis[2]轴的位移值，单位：脉冲数
//!        posi_mode 位移模式设定：0 表示相对位移，1 表示绝对位移
//!返回值：
//!            0 表示成功
//!            1 表示posi_mode错误
//!            2 表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_line3(u16 *axis,int Dist1,int Dist2,int Dist3,u16 posi_mode);


//!功能：指定任意四轴做直线插补运动
```

```
//!       让指定的四轴作对称的梯形加减速插补运动。当 posi_mode 为 0 时，作相对位移运动，
//!运动方向由 Dist 的正负值确定；为 1，作绝对位移运动，运动方向由 Dist 与当前位置
//!的差值决定。
//!参数：
//!       Dist1 指定 axis[0]轴的位移值，单位：脉冲数
//!       Dist2 指定 axis[1]轴的位移值，单位：脉冲数
//!       Dist3 指定 axis[2]轴的位移值，单位：脉冲数
//!       posi_mode 位移模式设定：0 表示相对位移，1 表示绝对位移
//!返回值：
//!           0 表示成功
//!           1 表示posi_mode错误
//!           2 表示轴参数错误
EXTERN_C int STDCALLDEF C45CE_line4(int Dist1, int Dist2, int Dist3, int Dist4,u16 posi_mode);

//!功能：读取指定轴有关运动信号的状态，包含指定轴的专用 I/O 状态
//!
//!参数：
//!       axis：轴号
//!返回值：
//!       位号        信号名称        描述
//!       0           ALM             ALM 信号          有效为1
//!       1           EL+         EL+信号           有效为1
//!       2           EL-         EL-信号           有效为1
//!       3           EMG             EMG 信号          有效为1
//!       4           HOME            Home 信号         有效为1
//!       5           SD          减速信号          有效为1
//!       6           SL+         软限位信号，最大值  (目前不支持此状态)
//!       7           SL-         软限位信号，最小值  （目前不支持此状态）
//!      8~15         P0~P7           每轴的8个IO ，输入电平状态，0--低电平，1--高电平
//!       其它位              保留

EXTERN_C unsigned short STDCALLDEF C45CE_axis_io_status(u16 axis);



//!功能：调试的时候用，获取运动部分底层细节状态信息
//!
//!参数：
//!       axis：轴号
EXTERN_C     unsigned short STDCALLDEF C45CE_Get_DebugStatus(WORD
```

```
axis,UINT32 *statusArr);
//!功能：读取指定轴的指令脉冲位置
//!
//!参数：
//!        axis：轴号
//!返回值：指定运动轴的命令脉冲数，单位：脉冲
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_get_position(u16 axis);


//!功能：设定指定轴的指令脉冲位置
//!
//!参数：
//!        axis：轴号
//!        current_position：设定位置
//!返回值：指定运动轴的命令脉冲数，单位：脉冲
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_set_position(u16 axis,long
current_position);
//!功能：二轴相对位置插补
//!        让指定的二轴作相对位置圆弧插补运动
//!参数：
//!        参数：  axis：轴号列表
//!                rel_pos：目标位置列表（指定圆弧终点）
//!                rel_cen：圆心位置列表
//!                arc_dir：圆弧方向，
//!                        0－顺时针
//!                        1－逆时针
//!返回值：
//!           0 表示成功
//!           1 表示arc_dir错误
//!           2 表示轴参数错误
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_rel_arc_move(u16 *axis,long
*rel_pos,long *rel_cen, u16 arc_dir);


//!功能：打开缓冲区
```

```
//!        让指定的二轴作相对位置圆弧插补运动
//!参数：
//!        参数：  buffnum：缓冲区变号，目前不用
//!返回值：
//!            0 表示成功
//!            1 表示错误
//!            2 表示错误
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_conti_open_list(int buffnum);



//!功能：关闭缓冲区
//!        让指定的二轴作相对位置圆弧插补运动
//!参数：
//!        参数：  buffnum：缓冲区变号，目前不用
//!返回值：
//!            0 表示成功
//!            1 表示错误
//!            2 表示错误
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_conti_close_list(int buffnum);


//!功能：开始执行缓冲区
//!        让指定的二轴作相对位置圆弧插补运动
//!参数：
//!        参数：  buffnum：缓冲区变号，目前不用
//!返回值：
//!            0 表示成功
//!            1 表示错误
//!            2 表示错误
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_conti_start_list(int buffnum);


//!功能：缓冲区连续直线插补函数
//!
```

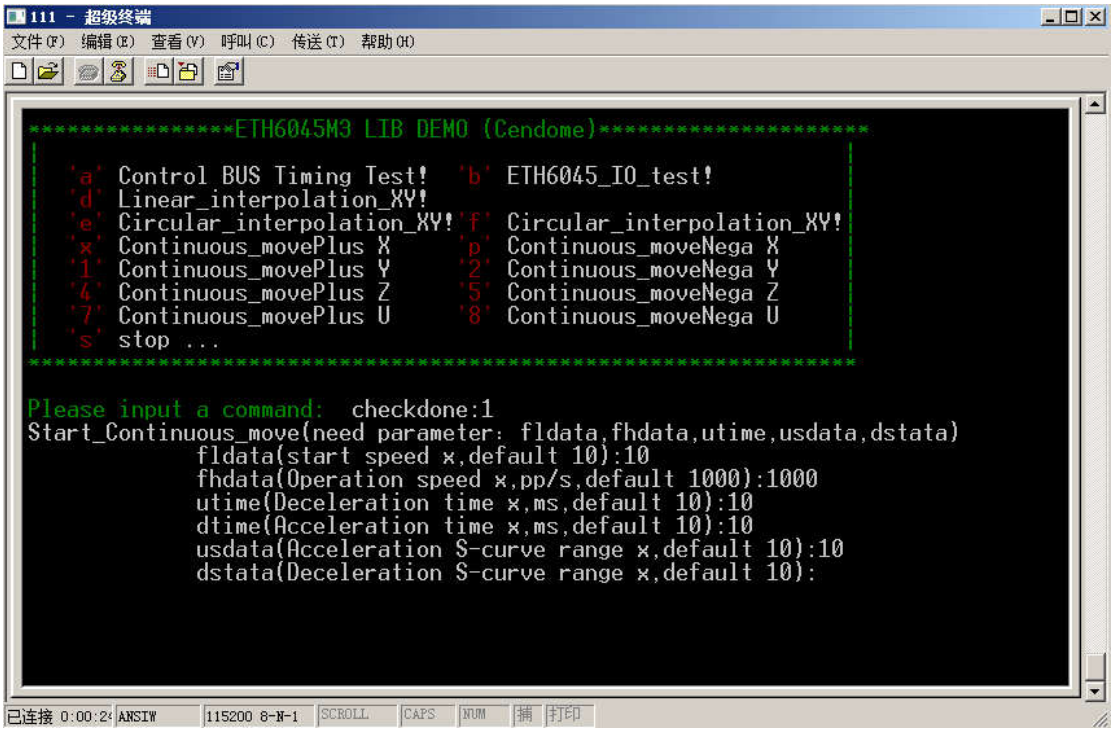
```
//!参数：
//!        axisNum  轴数
//!        piaxisList  轴号列表,
//!        pPosList  位置列表
//!        posi_mode 0 -  相对, 1-绝对位置模式
//!返回值：
//!            0  表示成功
//!            1  表示缓冲区未打开
//!            2  表示错误
EXTERN_C unsigned int STDCALLDEF C45CE_conti_lines (u16 axisNum, u16
*piaxisListw, long *pPosList, u16 posi_mode);


//!功能：连续插补中减速停止
//!
//!参数：
//!        参数：  buffnum：缓冲区变号，目前不用
//!返回值：
//!            0  表示成功
//!            1  表示错误
//!            2  表示错误
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_conti_decel_stop_list (int
buffnum);

//!功能：连续插补中立即停止
//!
//!参数：
//!        参数：  buffnum：缓冲区变号，目前不用
//!返回值：
//!            0  表示成功
//!            1  表示错误
//!            2  表示错误
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_conti_sudden_stop_list(int
buffnum);



//!功能：设置间隙补偿值
```

```
//!    反向间隙补偿速度 由速度设置的最后一个参数指定
C45CE_set_profile(u16 axis, UINT32 Max_Vel, double acc, double dec,UINT32
BacklashSpeed=500)
//!
//!
//!
//!参数：
//!        参数：  axis：轴号 （取值0,1,2,3 分别对应X,Y,Z,U轴）
//!        参数：  backlash：间隙补偿值， 单位：脉冲 取值范围（0~4095）
//!返回值：
//!            0 表示成功
//!            1 表示错误
//!            2   axis 参数输入不正确
//!
//!
//!
EXTERN_C unsigned int STDCALLDEF C45CE_set_backlash(u16 axis, UINT32
backlash);


//!功能：置位指定卡的指定输出口
//!
//!参数：
//!        参数：  bitno：输出口位号（1～16）
//!        参数：  on_off: 输出电平，0 表示输出低电平，1 表示输出高电平
//!返回值： 无

EXTERN_C void C45CE_write_outbit (WORD bitno,WORD on_off);


//!功能：读取指定卡的指定输入口
//!
//!参数：
//!        参数：  bitno：输入口位号（1～16）
//!        参数：  on_off: 输出电平，0 表示输出低电平，1 表示输出高电平
//!返回值：0 表示低电平；1 表示高电平

EXTERN_C    int C45CE_read_inbit(WORD bitno) ;

//!功能：启动指定轴的手轮脉冲运动
//!参数：axis：轴号,取值0,1,2,3 分别对应X,Y,Z,U轴
//!返回值：
//!            0 表示成功
//!            1 表示指定轴停止运行
```

```
//!            2 表示轴参数错误
EXTERN_C     int    C45CE_handwheel_move(WORD axis);


//!功能：设置编码器反馈输入模式函数，同时允许EA/EB输入
//!反馈位置计数器是一个 28 位正负计数器，对通过控制卡编码器接口 EA，EB
输入的脉冲（如
//!编码器、光栅尺反馈脉冲等）进行计数。
//!可以配置两种模式的脉冲输入：（1）非 A/B 相（脉冲+方向模式）；（2）
AB相输入模式
//!参数：axis：轴号,取值0,1,2,3 分别对应X,Y,Z,U轴
//!          mode：编码器反馈输入模式
//!              0     1 倍 A/B 相脉冲信号
//!              1     2 倍 A/B 相脉冲信号
//!              2     4 倍 A/B 相脉冲信号
//!              3     非 A/B 相, 为脉冲+方向
//!返回值：
//!            0 表示成功
//!            1 表示指定轴停止运行
//!            2 表示轴参数错误
EXTERN_C     void    C45CE_counter_config(WORD axis, WORD mode);

//!功能：读取编码器反馈的脉冲计数值。范围：28 位有符号数。
//!反馈位置计数器是一个 28 位正负计数器，对通过控制卡编码器接口 EA，EB
输入的脉冲（如
//!编码器、光栅尺反馈脉冲等）进行计数。
//!参数：axis：轴号
//!返回值：编码器的计数值

EXTERN_C     int    C45CE_get_encoder(WORD axis);


//!功能：设置编码器的脉冲计数值。范围：28 位有符号数。
//!反馈位置计数器是一个 28 位正负计数器，对通过控制卡编码器接口 EA，EB
输入的脉冲（如
//!编码器、光栅尺反馈脉冲等）进行计数。
//!参数：axis：轴号
//!          encoder_value：编码器脉冲计数的设定值。
//!返回值：无

EXTERN_C     void    C45CE_set_encoder (WORD axis, long encoder_value);


//!功能：二轴直线插补2
```

```
//!       为了同时进行2个二轴直线插补增加
//!让指定的两轴作对称的梯形加减速插补运动。当 posi_mode 为 0 时，作相对位移运动，
//!运动方向由 Dist 的正负值确定；为 1，作绝对位移运动，运动方向由 Dist 与当前位置
//!的差值决定。
//!参数：
//!       axis1，2：第一、二轴轴号
//!       Dist1，Dist2：第一、二轴距离
//!       posi_mode：位置模式，   (不起作用，只能实现绝对位移)
//!       0－相对位移
//!       1－绝对位移
//!返回值：
//!           0 表示成功
//!           1 表示posi_mode错误
//!           2 表示轴参数错误
EXTERN_C    int STDCALLDEF C45CE_Bline2(WORD axis1,long Dist1,WORD axis2,long Dist2,WORD posi_mode);

// ARM的4个IO输出设置 ， number:0~3 , value:0/1
EXTERN_C    BOOL   C45CE_set_AGPIO (int number, int value);
```

五.二次开发 ETH6045m3-demo（售后提供源码）



ETH6045m3-demo 程序界面如上图，可以设置启动速度，加速度等等参数后，进行 X,Y,Z,U 四个轴的运动测试，包括直线插拔，圆弧插补等等。

- 命令 a
  所有的 DOUT 全部输出 1
- 命令 b
  所有的 DOUT 全部输出 0
- 命令 d
  XY 轴做直线插补
- 命令 e
  XY 轴做顺时针圆弧插补
- 命令 f
  XY 轴做逆时针圆弧插补
- 命令 x
  X 轴做正向连续运动
- 命令 p
  X 轴做负向连续运动
- 命令 1
  Y 轴做正向连续运动
- 命令 2
  Y 轴做负向连续运动
- 命令 4
  Z 轴做正向连续运动
- 命令 5

Z 轴做负向连续运动

- 命令 7
  U 轴做正向连续运动
- 命令 8
  U 轴做负向连续运动

示例

X 轴连续运动，T 型加速时间 2 秒， 起始速度 100，匀速速度 3000。

```
ETH6045_start_Continuous_move(need parameter:fldata,fhdata,utime,usdata,dstata)
            fldata(start speed x,default 10):100
            fhdata(Operation speed x,pp/s,default 1000):2000
            utime(Deceleration time x,ms,default 10):2000
            dtime(Acceleration time x,ms,default 10):10
            usdata(Acceleration S-curve range x,default 10):10
            dstata(Deceleration S-curve range x,default 10):10
```

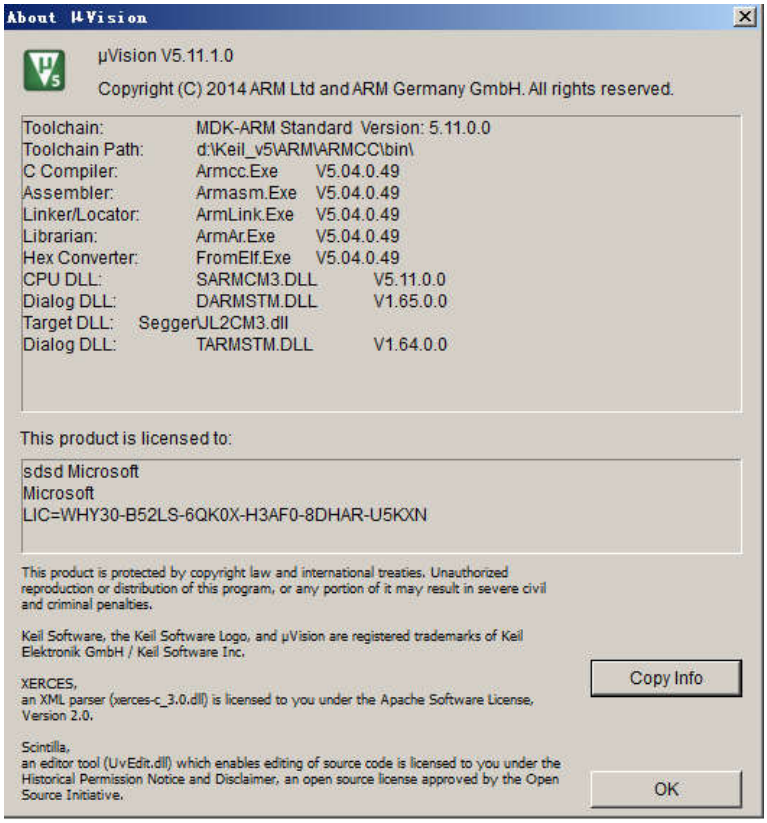
在上面菜单处按 x 后，分别输入

起始速度（start speed）：100 （输入后要回车）
匀速速度（Operation speed）：3000 （输入后要回车）
加速时间（utime）2000 （输入后要回车）
减速时间（dtime）2000 （输入后要回车）
S 曲线参数（usdata） 不输入直接回车
S 曲线参数（dstata） 不输入直接回车

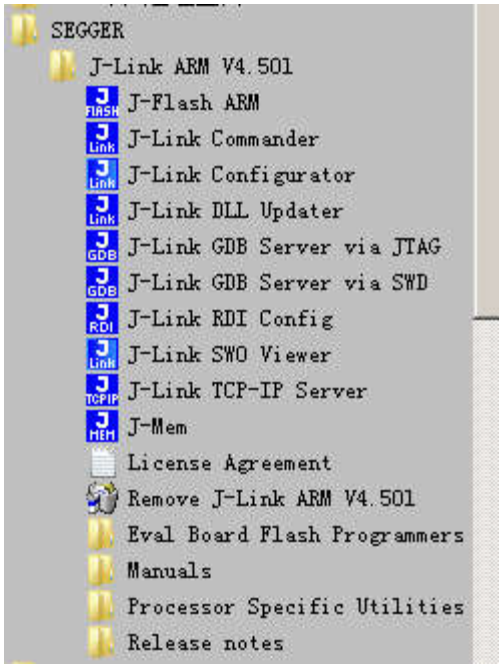
六.KEILC 开发步骤

1.安装 KEIL-MDK 与 JLINK usb 驱动

- 根据相关教程安装好 KEIL-MDK



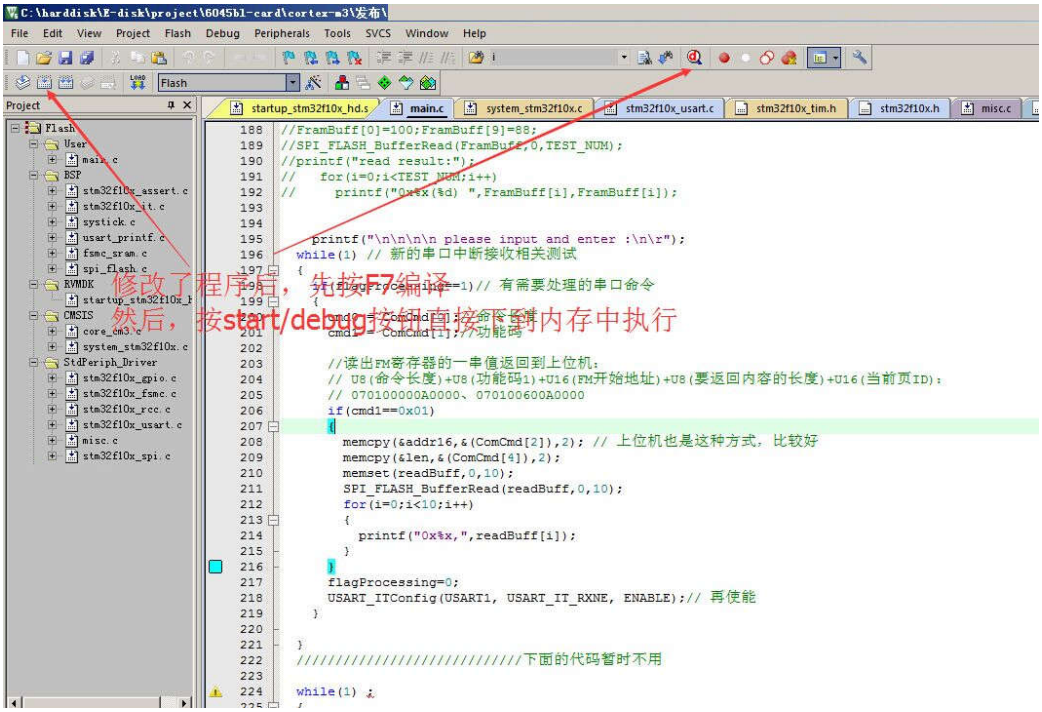
- 然后安装 JLINK USB 驱动



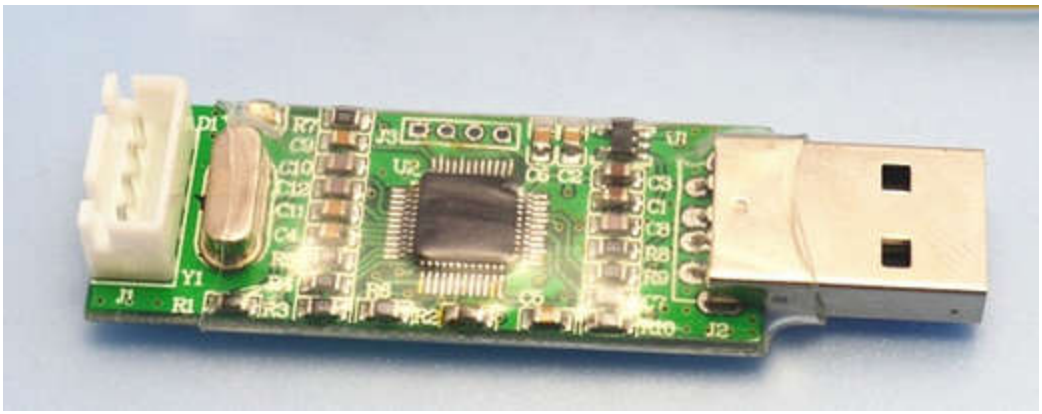
2.打开 ETH6045m3-demo 源码，进行修改

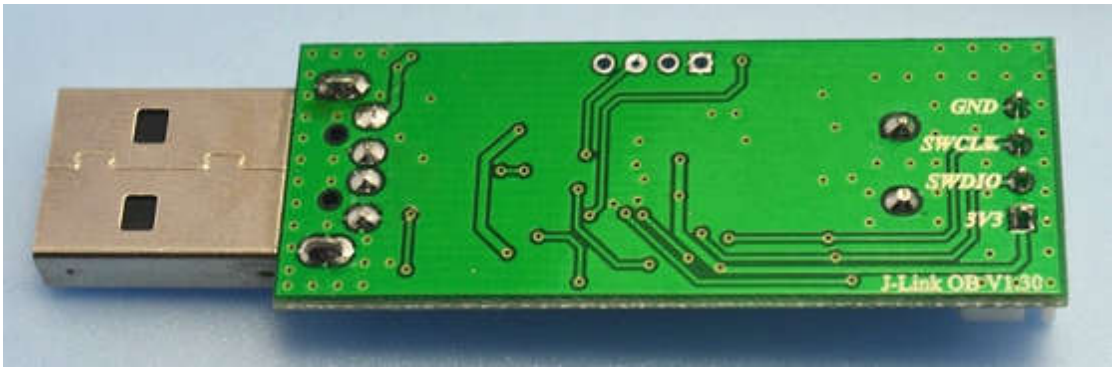
修改了程序代码再测试时，最便捷的调试方法，可以不用下载， 直接用 start/debug 按钮，下到内存中执行，前提是：

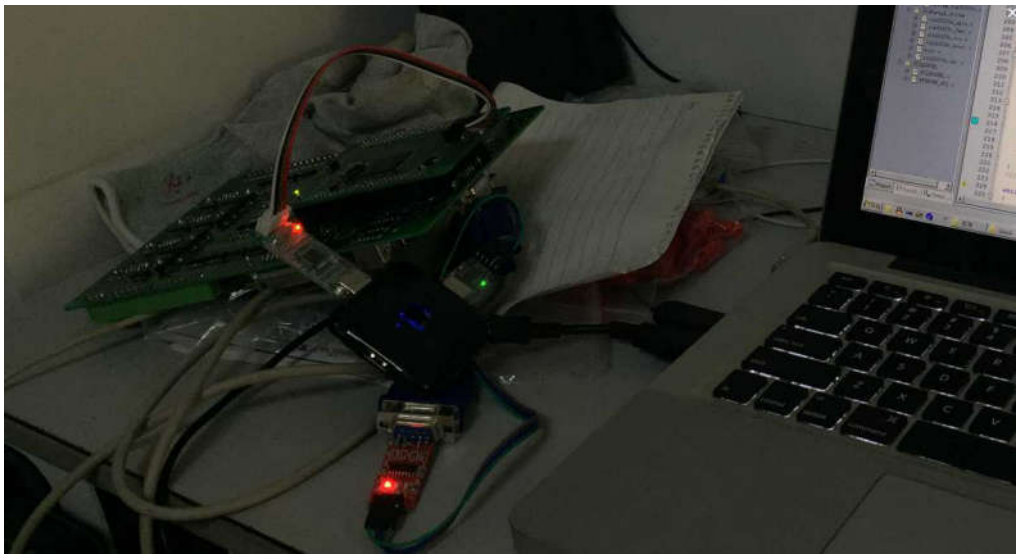
- 接好调试线(烧写器)
- F7 编译程序



调试线的接法如下：

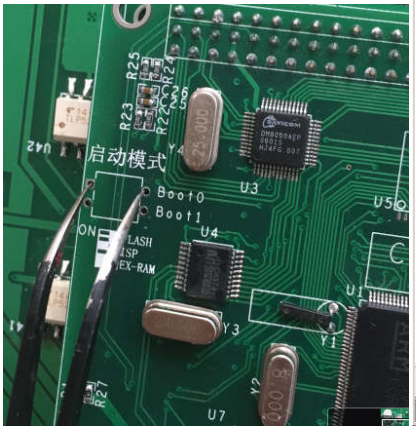


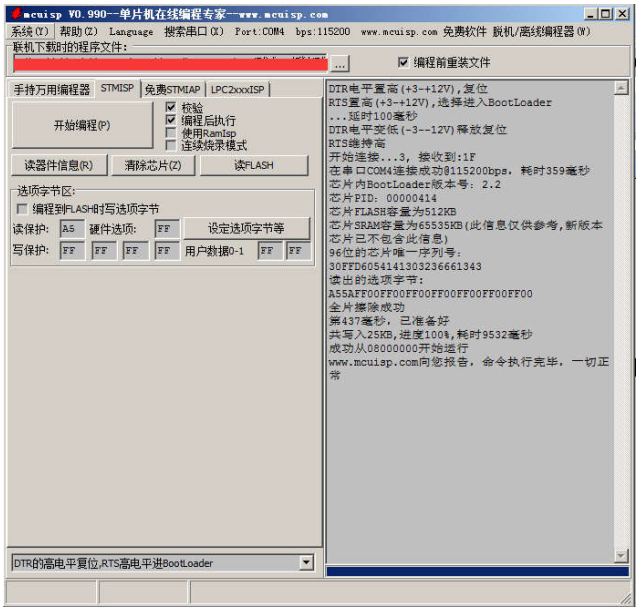




3.通过串口烧写 Hex 文件

1.首先让 M3 进入下载模式(镊子短路 BOOT0 信号)并上电，串口 COM1 下载 "USB Mass Storage(SD 卡和 NAND 模拟 U 盘)"工程的 hex 文件。





2.重新上电控制器，通过串口(COM1,波特率 115200)可以看到 M3 内部软件的变化