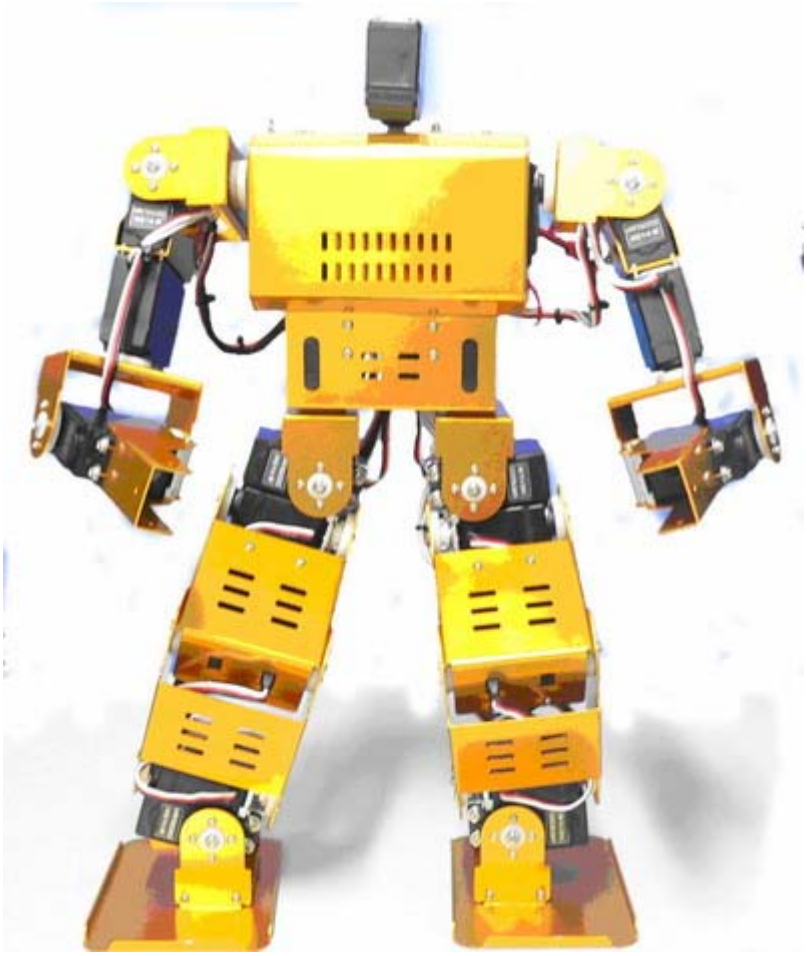
Xmen （测试版）调试说明书

——Xmen-test 人形机器人



亿学通™

北京纳克斯科技发展有限公司

目录

1. 系统概述

1.1 舵机与机器人

我们都知道，机器人有许多个关节，每一个关节我们称为一个自由度。一般的机体，都有十几个自由度，这样才能够保证动作的灵活性。在机器人机体上，我们通常使用舵机作为每一个关节的连接部分。它可以完成每个关节的定位和运动。舵机的控制信号相对简单，控制精度高，反应速度快，而且比伺服电机省电。这些优点是非常突出的。在下面的论述中，许多地方都会涉及到舵机相关的知识，读者应反复详细阅读。

1.2 舵机的基本认识

舵机的外观入下图所示：

图 1-1

这里可以看到，舵机体积十分小巧。机器人使用它是非常合适的。

一般的舵机可以旋转 185 左右，我们这里留一些余量，算做 180 度。八位单片机的精度是 256，我们也留一些余量，算作 250。这样我们可以得到一个基本的对应关系：

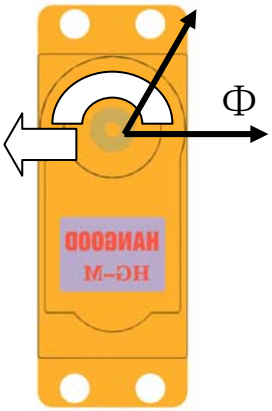


图 1-2

舵机转动角度Φ：0-180 度
单片机数值　N：0-250

1.3 舵机 PWM 信号介绍

1.3.1 PWM 信号定义

PWM 信号为脉宽调制信号，其特点在于他的上升沿与下降沿之间的时间宽度。具体的时间宽窄协议参考下列讲述。我们目前使用的舵机主要依赖于模型行业的标准协议，随着机器人行业的渐渐独立，有些厂商已经推出全新的舵机协议，这些舵机只能应用于机器人行业，已经不能够应用于传统的模型上面了。

目前，北京亿学通科技的 HG14-M 舵机可能是这个过渡时期的产物，它采用传统的 PWM 协议，优缺点一目了然。优点是已经产业化，成本较低，旋转角度大（目前所生产的都可达到 185 度）；缺点是控制比较复杂，毕竟采用 PWM 格式。

但是它是一款数字型的舵机，其对 PWM 信号的要求较低：

（1） 不用随时接收指令，减少 CPU 的疲劳程度；

（2） 可以位置自锁、位置跟踪，这方面超越了普通的步进电机；

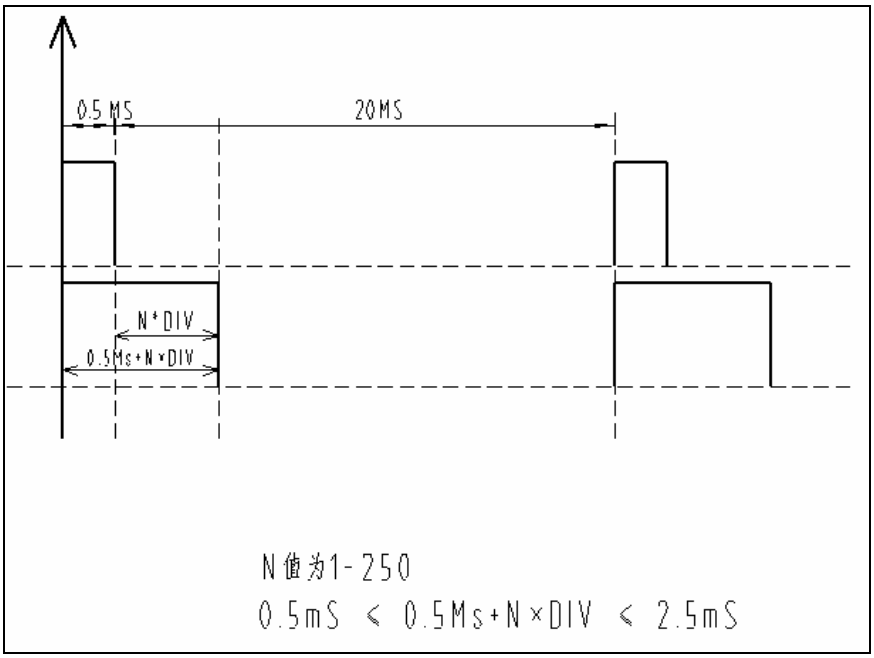


图 1-3

其 PWM 格式注意的几个要点：

（1） 上升沿最少为 0.5mS，为 0.5-2.5mS 之间；（对应舵机旋转 0-180 度）

（2） HG14-M 数字舵机下降沿时间没要求，目前采用 0.5Ms 就行；也就是说 PWM 波形可以是一个周期 1mS 的标准方波；

（3） HG0680 为塑料齿轮模拟舵机，其要求连续供给 PWM 信号；它也可以输入一个周期为 1mS 的标准方波，这时表现出来的跟随性能很好、很紧密。

1.3.2 PWM 信号控制精度制定

上面已经提到了八位单片机，我们的舵机需要的是方波信号。单片机的精度直接影响了舵机的控制精度，这里就详细的说明一下。

我们采用的是 8 位 STC12C5410ADCPU，其数据分辨率为 256，那么经过舵机极限参数实验，得到应该将其划分为 250 份。

那么 0.5mS---2.5Ms 的宽度为 2mS = 2000uS。

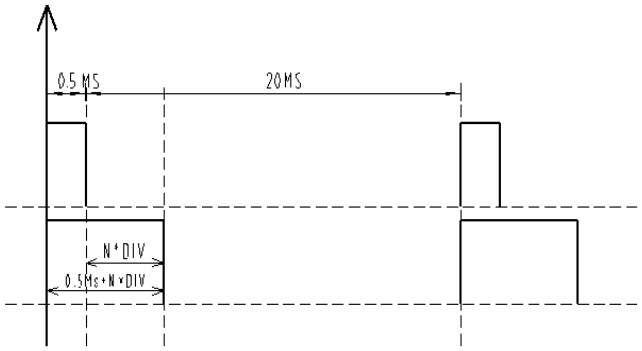
2000uS÷250=8uS

则：**PWM 的控制精度为 8us**

我们可以以 8uS 为单位递增控制舵机转动与定位。

舵机可以转动 185 度，那么 185 度÷250=0.74 度，

则：**舵机的控制精度为 0.74 度**



N值为1-250

0.5mS ≤ 0.5Ms+N×DIV ≤ 2.5mS

图 1-4

我们在这里做了一些名词上的定义。DIV 是一个时间位置单位，一个 DIV 等于 8us，关系入公式：

1 DIV = 8uS　　250DIV=2mS

时基寄存器内的数值为：（#01H）01 ———（#0FAH）250。

共 185 度，分为 250 个位置，每个位置叫 1DIV。则：

185÷250 = 0.74 度 / DIV

PWM 上升沿函数：　0.5mS + N×DIV

0uS ≤ N×DIV ≤ 2mS

0.5mS ≤ 0.5Ms+N×DIV ≤ 2.5mS

根据这些知识。我们就可以开始编程，并做一些初步的实验，学会舵机控制是研究机器人的一个比较技术手段，需要完全掌握。

2. 控制系统硬件定义

首先我们对控制板做简单的介绍，而后会对机器人的机体做一个说明。机器人的所有关节都会与控制板相连接。具体定义见下文：

2.1 STC 控制板简介

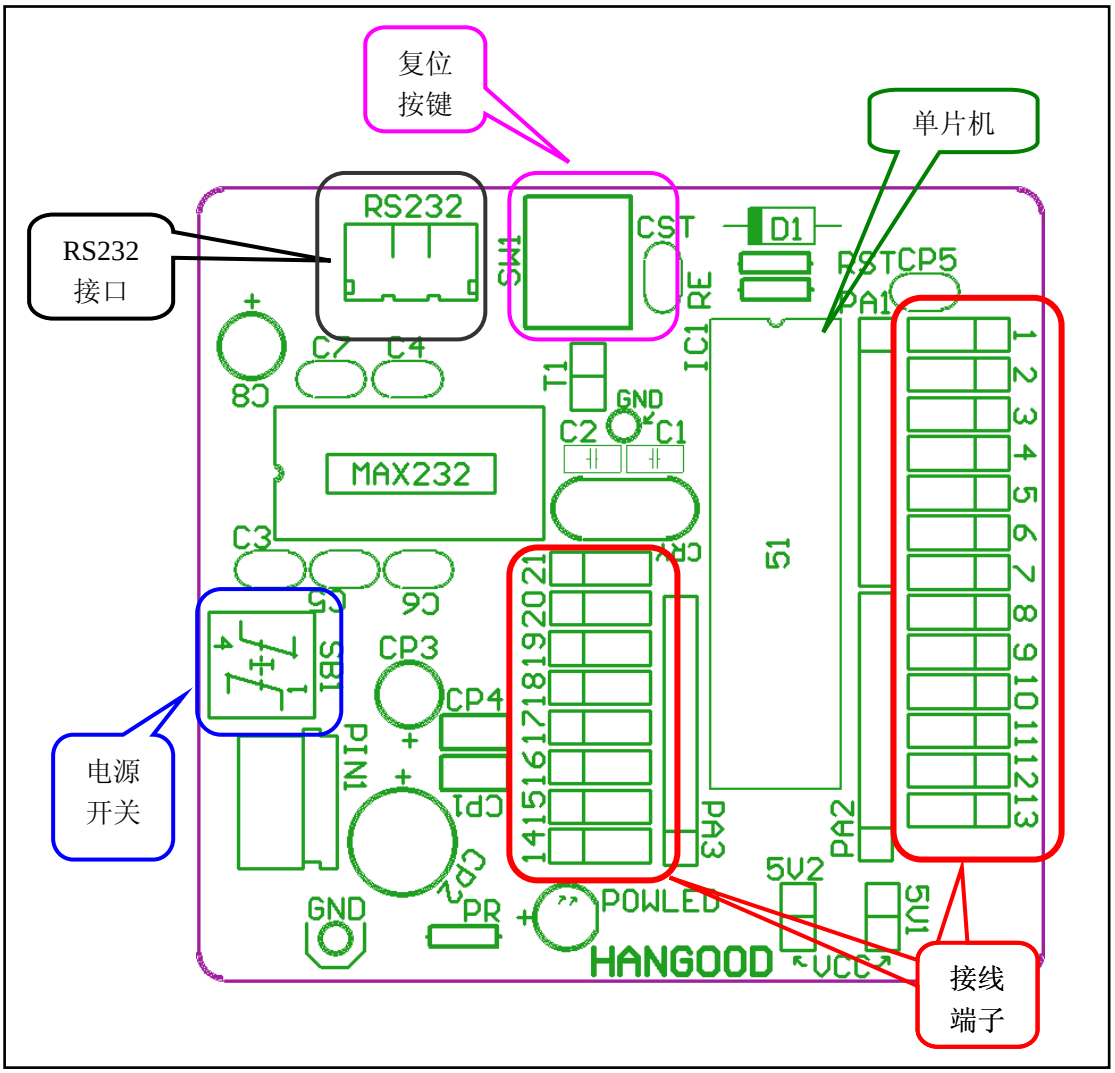


图 2-1

2.2 机器人机体简介

双足 XMEN 机器人共有 21 个关节，即所谓的每一自由度由一个舵机控制。这 21 个舵机都使用 STC 控制板完成控制，使机器人完成特定动作。下图是双足机器人下蹲拍摄的图片。

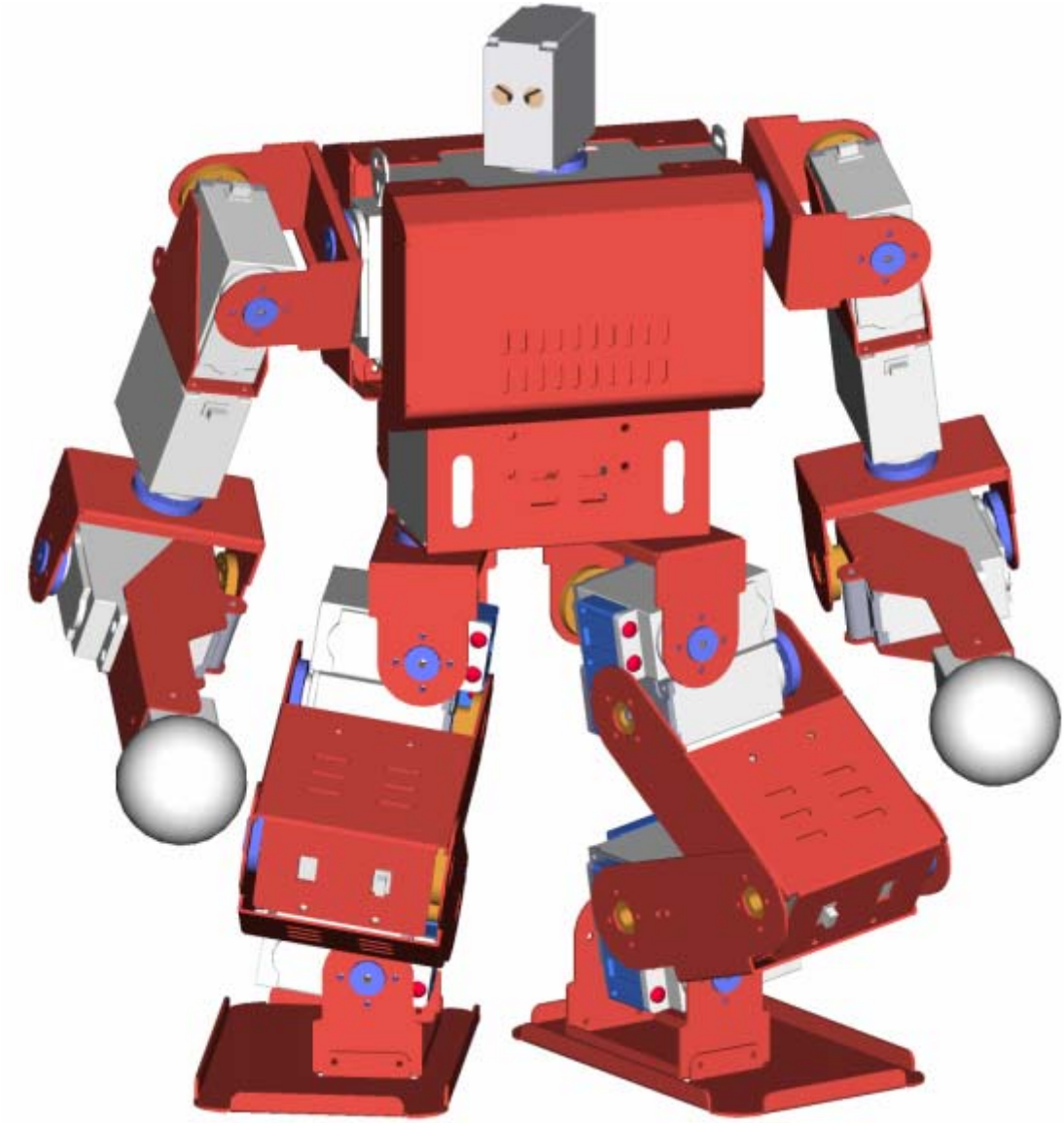


图 2-2

下面对机器人的每一个部分做示意说明

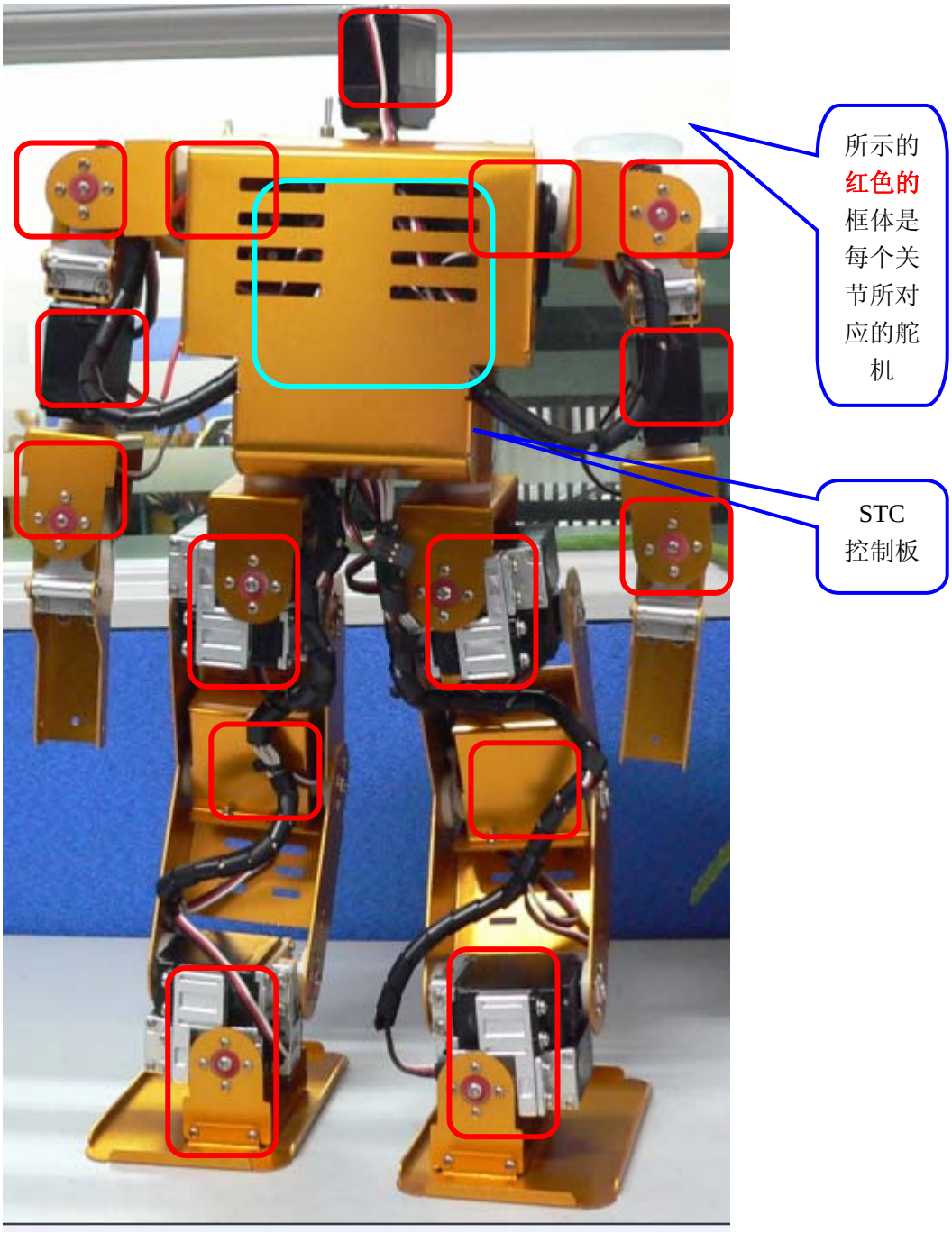


图 2-3

2.3 连接关系说明

控制板上共可以接 21 个舵机，由于一些客户需要加入其他传感器，故将控制板 14 至 21 接线端子暂不作为舵机输出端口，对于双足机器人来说，共有 21 个关节，这二十一个关节的舵机分别接在对应端口上。具体定义如下图所示：

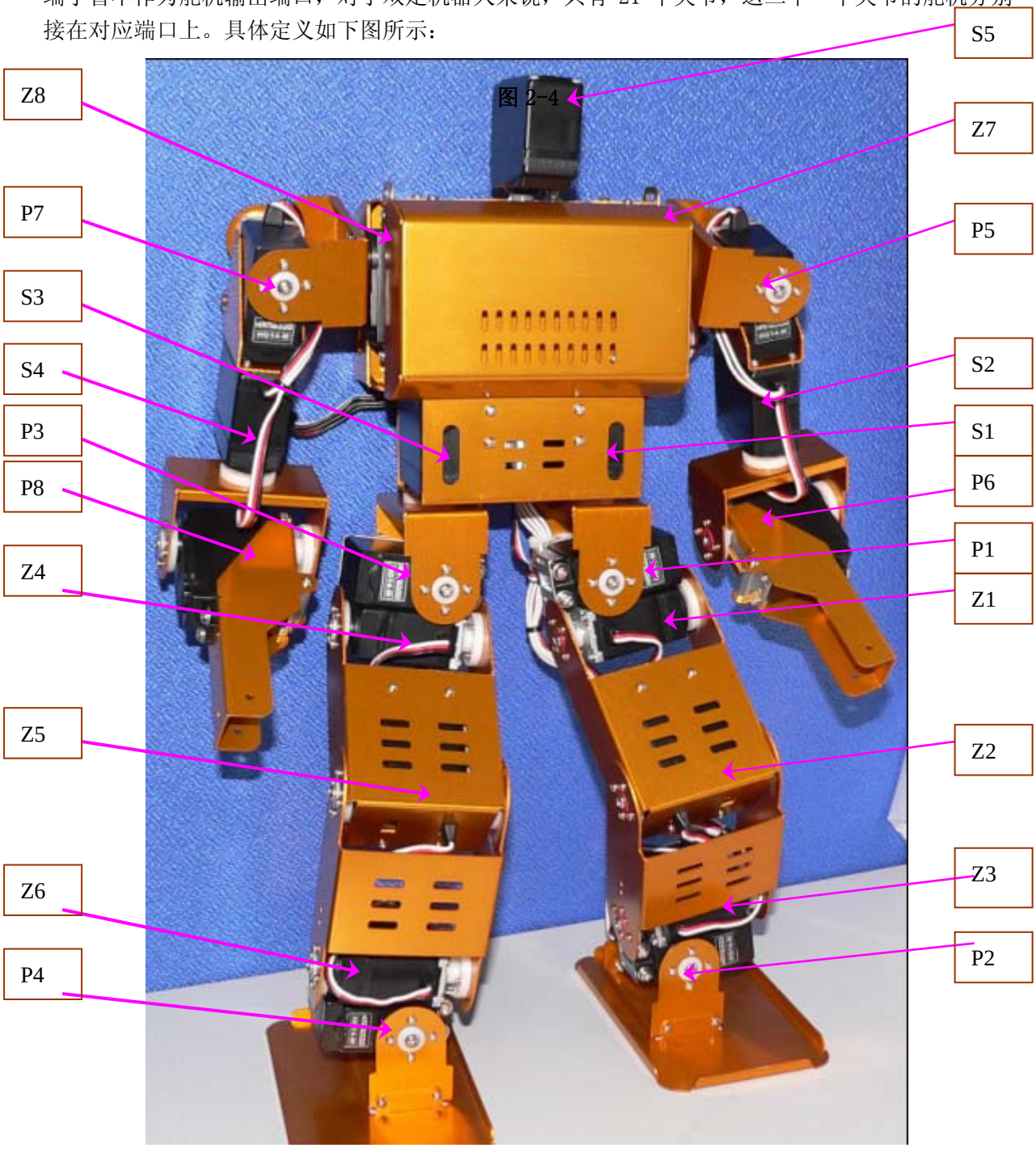


图 2-4

其中舵机的三端点接线定义如下：

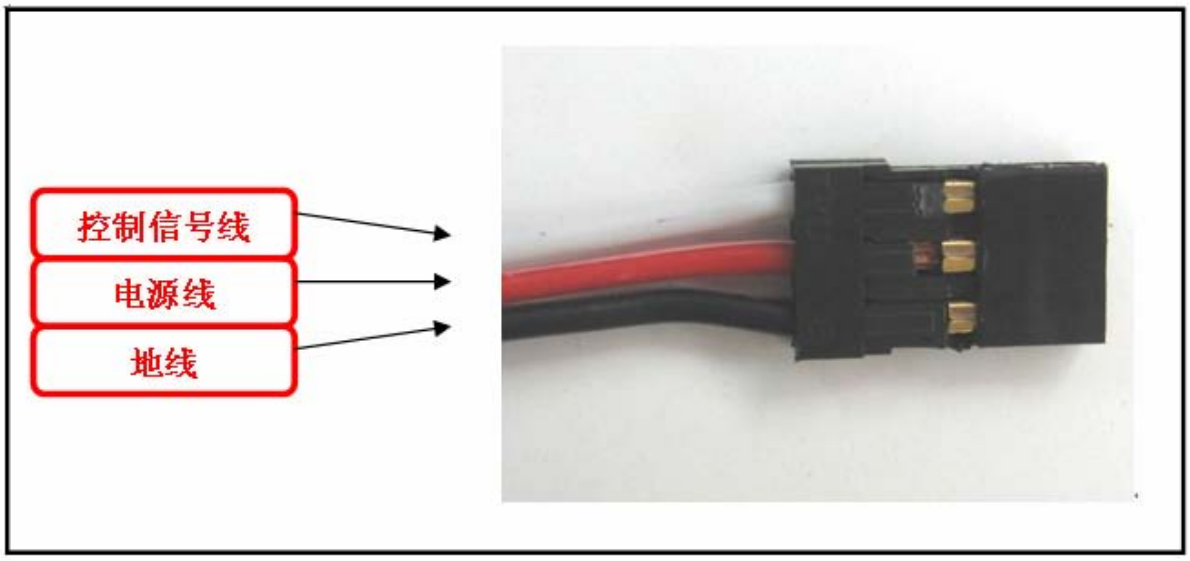


图 2-5

注意：舵机电源可以是 5V-7.4V，直流；
舵机信号线电压 5V，PWM 信号；

为了清晰，我们将 12 个舵机摆在一排，然后链接至控制板对应端口上。如下图：

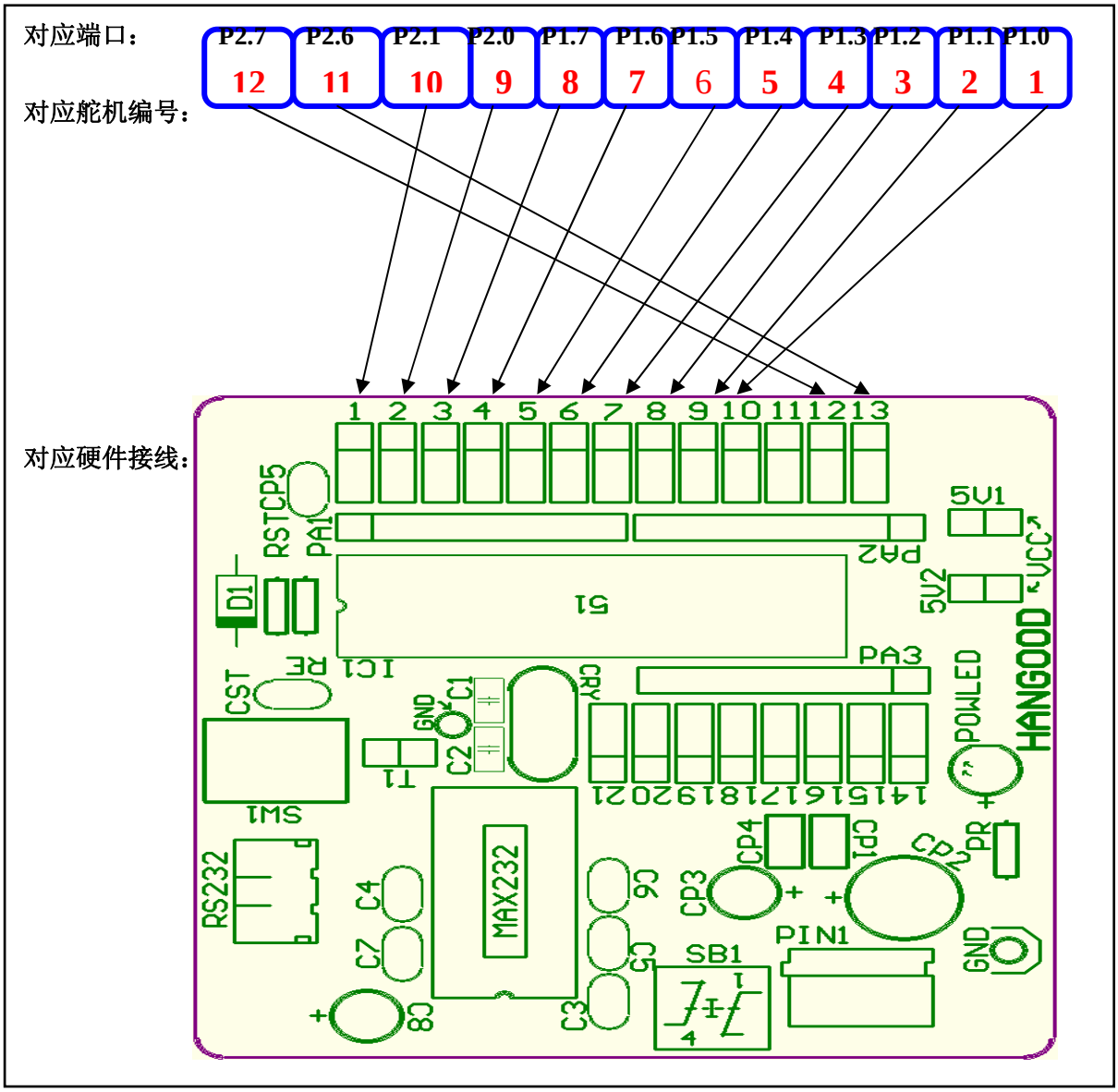


图 2-6

注意：在控制板上的三端出线遵守统一规则，离单片机近的是控制信号端，然后是舵机正电源，离单片机最远的是地。在舵机连接电路板的时候应该使白色控制信号线查到离单片机近的三端接口上。

3. 开发软件介绍

STC 单片机是基于 51 控制核的高速单片机。对于程序的编译和链接，我们可以使用 KEIL C 帮助完成。

在对芯片进行编程时，我们使用 STC 公司提供的烧录软件 STC_ISP_V3.5。

3.1 KEIL C

我们使用的单片机是 STC12C5410D，是 51 的内核。指令周期都优大幅度的缩减，运行速度自然提高。由于我们使用的是 51 内核，所以我们可以使用支持 C51 的开发软件帮助编辑程序和编译链接程序。KEIL C 就是这样一款软件。

KEIL C 的软件界面如下：

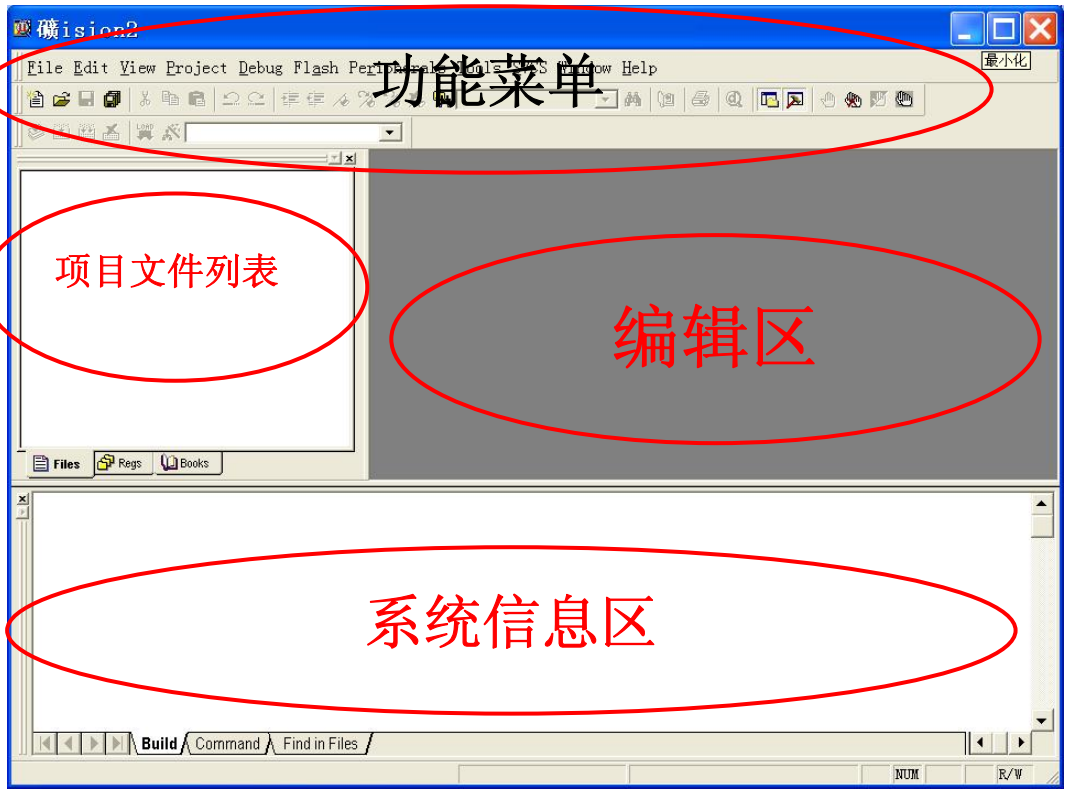


图 3-1

在项目中我们可以建立自己的工程，根据工程添加编辑程序文件。最终使用“完全编译”功能完成编译。在烧录单片机之前还应该生成对应的“.HEX”INTEL 二进制文件。有了这个文件，我们才能使用 STC 提供的软件进行烧写单片机的操作。

3.2 STC_ISP_V3.5

STC _ISP_V3.5 是由 STC 开发的程序烧写测试综合软件。它可以通过普通的串口（COM）烧写单片机的程序。软件运行稳定。操作相对方便。软件的操作界面如下：

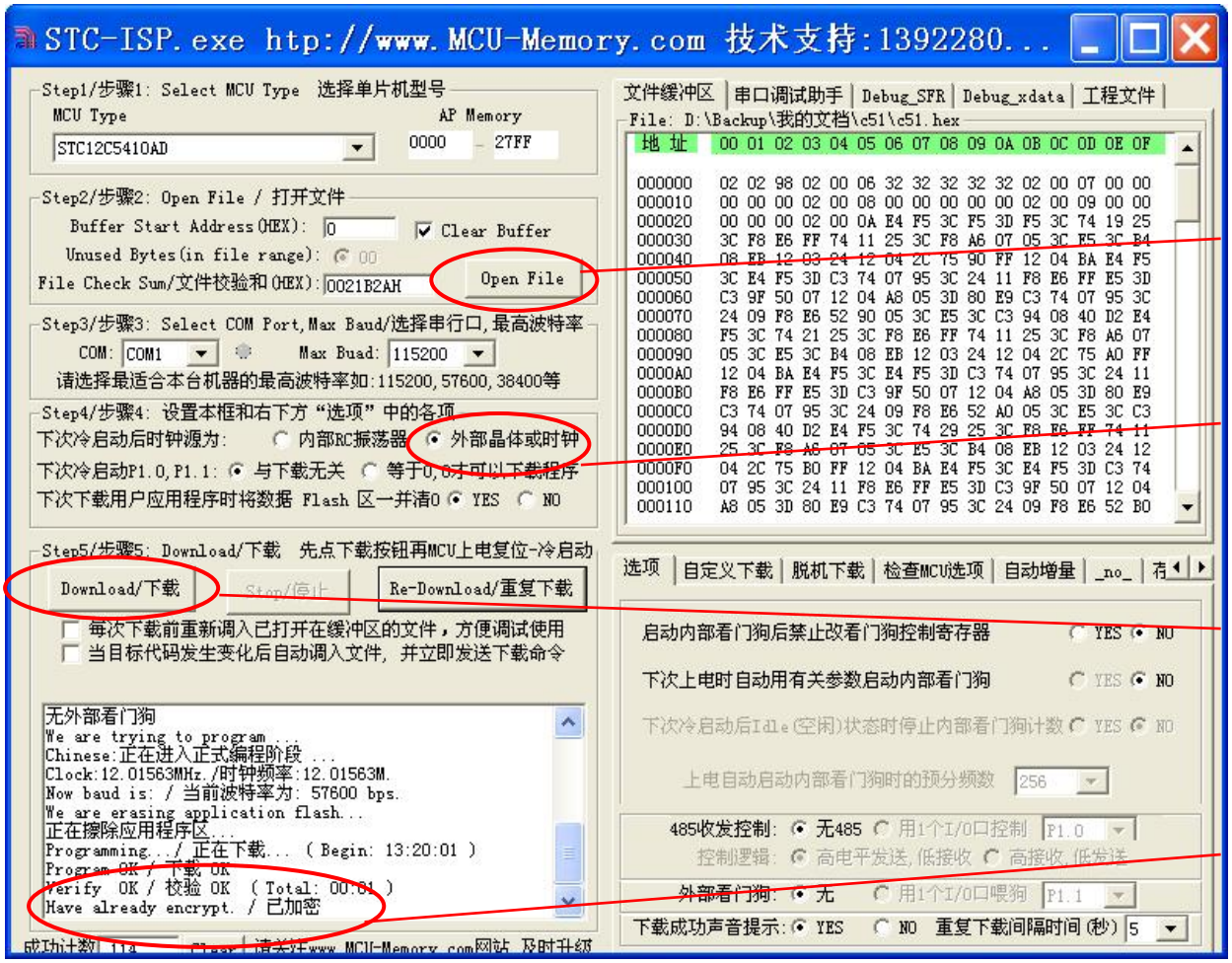


图 3-2

下面给出操作的具体步骤：

1. 选择芯片类型；
2. 选择要烧写的 HEX 文件；
3. 设置串行端口和波特率，这里要注意端口号，波特率的选择比较任意一般为 38400；
4. 选择外部晶振；与下载无关；清 FLASH 区；
5. 点击下载按钮后，打开单片机供电电源。在上电后单片机会自动进入编程状态。通过提示可以判断是否下载完成。

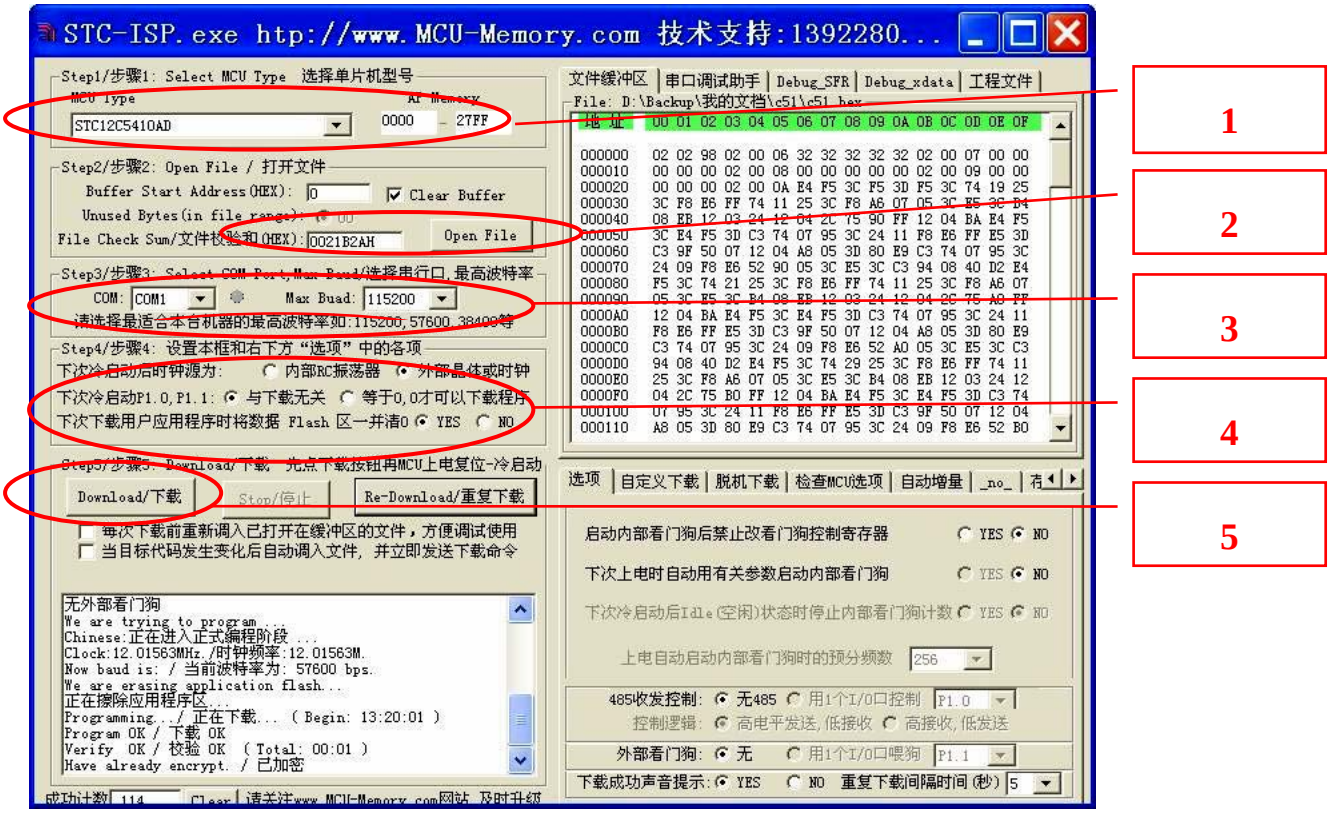


图 3-3

注意：

1. **在选择芯片时一定要看清型号，很容易选错。**
2. **应选择外部晶振，如果选择内部晶振，单片可以工作，但是舵机的控制会出现问题。**
3. **一定要先点编程按钮，然后在开控制板上的电源，要不下载不会成功。**

4.语言程序架构解析

4.1 C 语言嵌入汇编

由于机器人要求一个很准的时钟，并通过这个时钟做准确的延时，所以我们需要使用 51 的汇编语言帮助。但是在一些处理不严格的地方，我们可以 C 语言的模块化编程方法帮助完成。在这里我们使用了 C 语言嵌入汇编的技术。按文件分类，我们一共有两个文件：一个是 ASM 文件，还有一个是 C 的文件。他们需要包含在一个工程文件中才能编译通过。

4.2 汇编语言被嵌入的说明

在次机器人程序中，由于部分是使用的汇编语言编写，为了清晰和直观。我们将汇编程序单独摘出来做成 “.asm”的文件，在工程中，我们应该将文件关联到工程目录下，一旦关联，C 程序就可以调用汇编子函数了。我们采用模块化编程方法，在调用汇编程序时也是按照子函数的方式直接调用的。使用十分方便。

4.3 KEIL C 下的具体操作步骤

4.3.1 在 KEIL C 下建立工程（project）

在“project”菜单下选择“New project”，这时会有对话框询问你将工程保存在什么目录下，并询问工程名称，这个名称将使系统相关的文件都已这个名字为开头，包括我们将要用到 HEX 文件。如图所示：

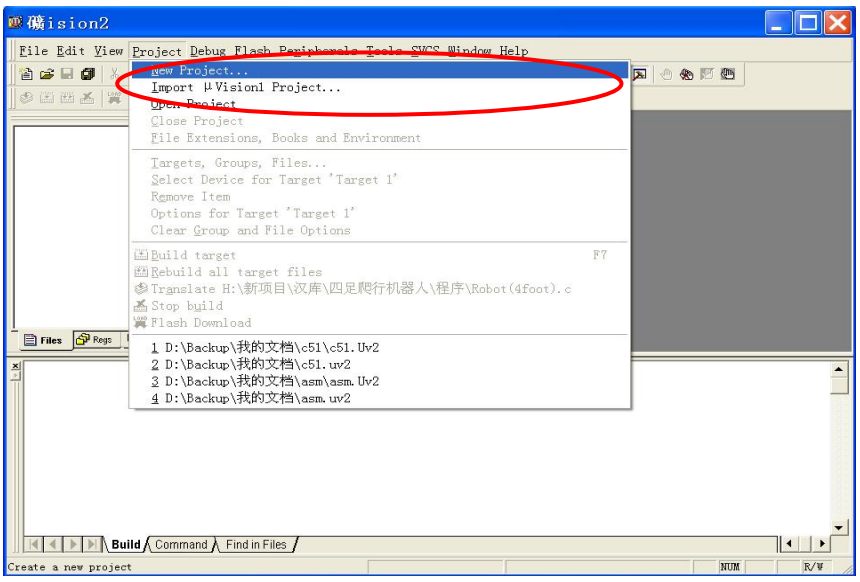


图 4-1

4.3.2 选择开发芯片类型

在设置好文件名和目录以后，系统会询问你所使用的芯片类型。我们这里只使用标准的 AT89S52 即可。下图是我们的选择。点击“确定”完成工程向导。

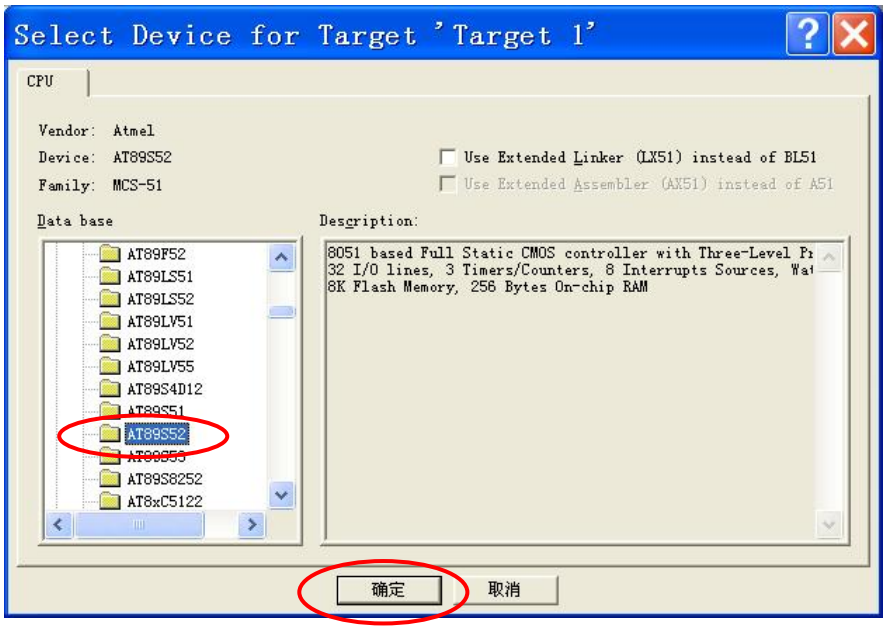


图 4-2

4.3.3 添加程序文件到工程

将制作好的 C 程序和汇编文件（必须以“.C”“.ASM”为后缀文件名）加入到新的工程中去。我们可以在下图中的对应位置，点击右键并左键点击对应选项添加文件。

这里需要说明一下，如果添加的文件只有 C 程序而没有添加汇编文件。则在编译的时候会出错。

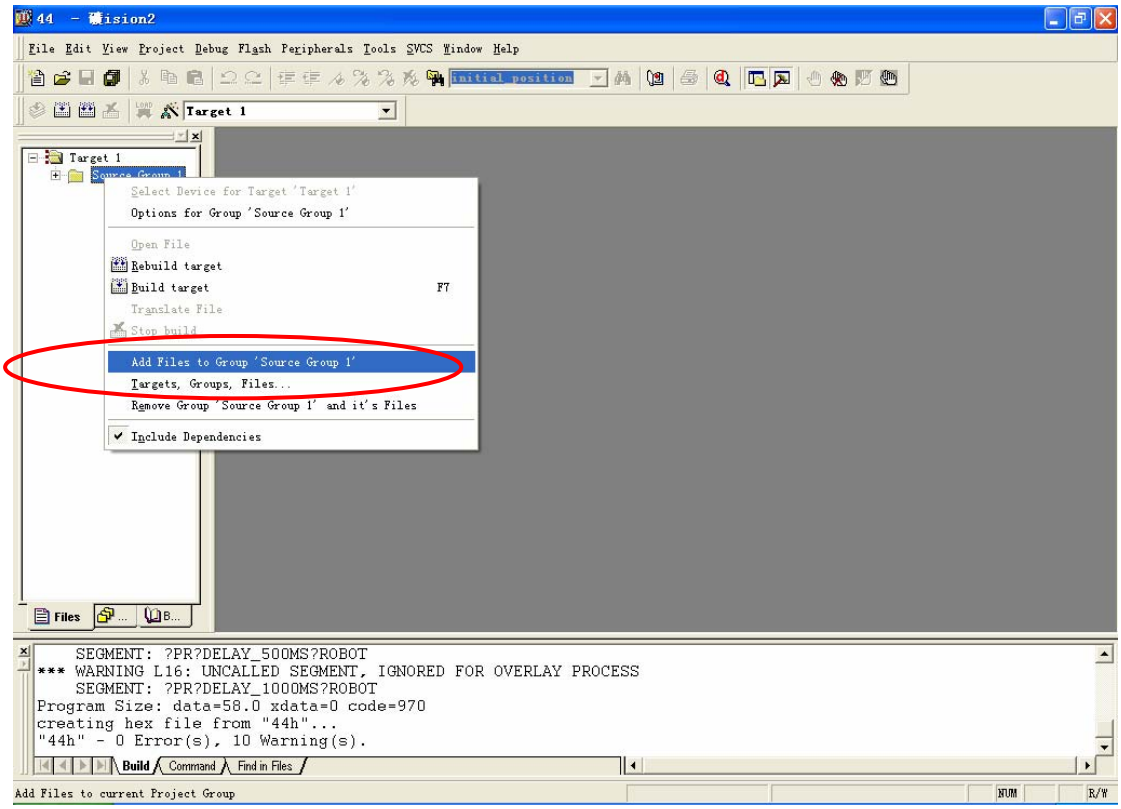


图 4-3

4.3.4 设置工程属性

每个工程的要求不一样，在 KEIL C 中为这些属性做了一个属性选项。我们可以设置好工程属性以方便开发。

首先在“project”菜单下选择“options for File‘xxxxx’”。如下图所示：

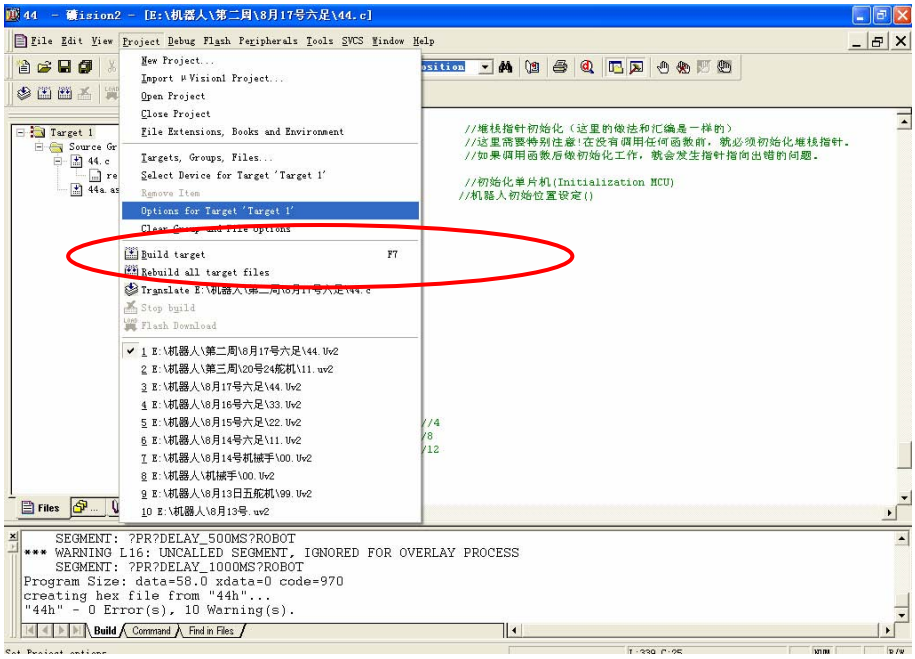


图 4-4

之后就会弹出工程属性对话框，在第三个“Output”选项卡下，将生成 HEX 文件的选项打上对勾，如下图所示：

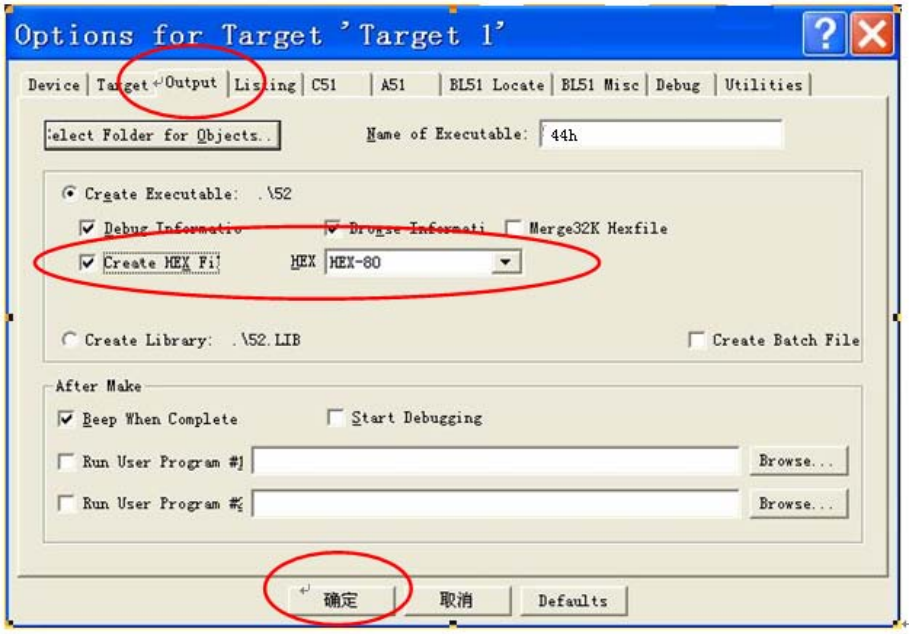


图 4-5

点击确定完成工程属性的调整，这样我们在之后生成的过程中就会生成 HEX 文件。对于下面的烧录软件来说，需要的就是 HEX 文件。我们可以选择完全编译，然后看看是否生成了 HEX 文件。如下图所示：

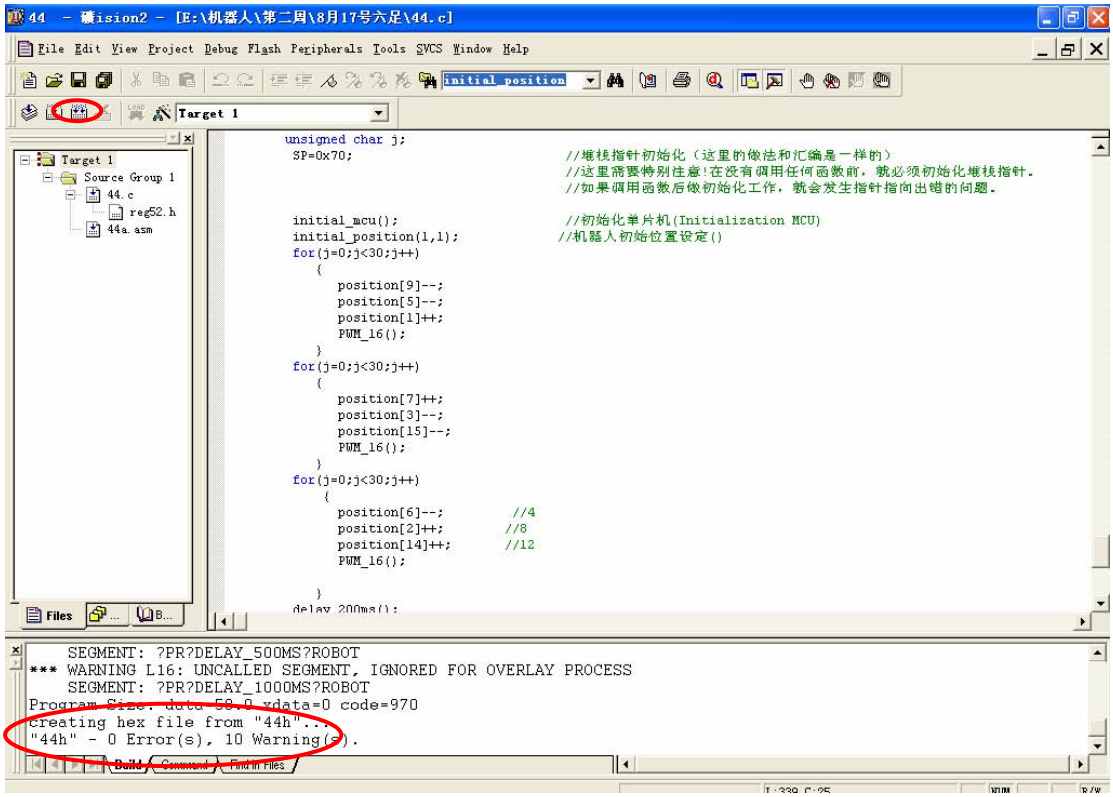


图 4-6

这样我们就可以使用工程中的 HEX 文件通过 STC _ISP_V3.5 烧录软件烧录 STC12C5410AD 了。

4.3.5 C 的底层函数说明

在 C 程序中有一些函数是我们控制机器人所必需的，下面就对其中最重要的函数做一个说明，这样可以帮助用户开发新的程序。

我们管这个程序叫“PWM_16()”，这个函数没有入口参数，但是有一个数组是与之相关的。这个数组是 uchar position[24]={0};

PWM_16()的源程序如下：

```
void PWM_16( )
{       uchar i=0,j=0;

        for(i=0;i<=7;i++)           //给排序数组赋值
        {
                paixu_ncha[i]=position[i];
        }

        sorting( );                 //调用排序函数
        N_value( );                 //调用 N 差函数
        P1=0xff;                    //使口 P1 全部拉高
        delay_500us();              //调用延时 500us 函数

        for(i=0;i<8;i++)            //延时输出到口 P1（八路）
        {
                for(j=0;j<paixu_ncha[7-i];j++)
                {
                        delay_8us();
```

```
                }
                P1=P1&kouchu[7-i];
        }


        for(i=0;i<8;i++)          //给排序数组赋值
        {
                paixu_ncha[i]=position[i+8];
        }
        sorting( );                //调用排序函数
        N_value( );                //调用 N 差函数
        P2=0xff;                   //使口 P2 全部拉高
        delay_500us();             //调用延时 500us 函数

        for(i=0;i<8;i++)           //延时输出到口 P2（八路）
        {       for(j=0;j<paixu_ncha[7-i];j++)
                {
                        delay_8us();
                }
                P2=P2&kouchu[7-i];
        }

        for(i=0;i<saowei;i++)
        delay_20us();
}
```

这个函数会使用延时的方式输出 PWM 波形，对应的端口是 P1 口和 P2 口。我们在文章开始部分说过。对于一个舵机的 0 至 180 度转动，我们在单片机内部是对应 1-250 的。这样，延时程序首先先延时 0.5ms，形成 PWM 高电平。然后这时根据对应舵机的值拉低电平。每一格需要延时 8us。这个数值也是在上面计算出来的。我们通过排序（我们称之为排序函数），判定出哪一个口最先到达延时时间。最先到达的，则最先拉低。他们之间的间距是通过对舵机值做差得到的（我们称之为 N 差函数）。这样我们就会看到我们需要的八路波形。对 P1 口做完这些事情后，在对 P2 口做。

```
void main(void)
{
 uchar flag=3,i=0,count=4;
 SP=0x70;          //堆栈指针初始化（这里的做法和汇编是一样的）
                   需要特别注意!在没有调用任何函数前，就必须初始化堆栈指针。
                   //如果调用函数后做初始化工作，就会发生指针指向出错的问题。
 initial_mcu();                    //初始化单片机(Initialization MCU)
 initial_position(1,1);           //调用机器人初始位置设定第 1 族子程序
 delay_500ms();
while(1)
        {//       r_pyi(3);                 //右平移   九
         //       l_pyi(3);                移 十
                  front_jh_bs_zu2(5);   //摆手交互式第二类步伐子程  一
        //        delay_1000ms();             //延时 1000ms
           //       l_zuan_bs(3);           //左转+摆手模式子程第  二
        //        delay_1000ms();             //延时 1000ms
           //       r_zuan_bs(3);           //右转+摆手模式子程第  三
        //        delay_1000ms();             //延时 1000ms
        //      goal_r(2);                //右脚踢球(站立)子程序  四
        //       delay_1000ms();             //延时 1000ms
        //      goal_l(2);                //左脚踢球(站立)子程序  五
          //          delay_1000ms();
//              fwc(5);                   //俯卧撑
         }
       while(1);
}
```

此函数为 HGR-3M-C 机器人软件中的主函数，其内部函数的嵌套调用如图说明。

4.3.6 C 的编程说明

程序的最底下是 main()主函数。这是整个程序开始的地方。我们可以编制相应的函数。最后通过主程序调用的方式来开发自己的动作函数库。请详见其他相关手册。

5.遇到的问题及解决方法

A&Q

1. 问： 如何判断单片机已经坏了？
   答： 有几种情况可以断定单片机已经坏了：
   ①单片机在任何编程器上，都不能编程，并会提示芯片错误之类的警告信息。
   ②单片机本身已经被“烧”的“面目全非”的时候。
   ③单片机在正确连接晶振、复位、ROM 选择、电源后，仍不能操作此芯片。

   这里只是举了一些例子，单片机烧坏的可能性多种多样，当我们在判断一颗芯片坏调之前，一定要用一颗好的芯片做同样的操作，看看是不是自己的操作存在问题。芯片工作正常，才敢断定芯片已坏。有时候单片机本身没有问题，而是在焊接或者电路板上存在问题没有发现罢了。

2. 问：我想知道串口和串口线的好坏，应该如何做？
   答：你可以下载一个串口调试软件，然后根据手头上的工具选择测试方式：
   ① 有示波器：这是最真实的测试方式，我们用示波器的探头测试串口的 2 号、3 号管脚（对地，也就是 5 号管脚，这里说的是标准的 9 针串口的定义），在电脑上打开串口调试软件，向外面发数据，示波器能看到波形就认为发送是好的，但是串口接收数据无法测试。一般来说，如果发是好的，收也是好的。
   ② 电脑上有两个串口：将串口线（2、3 线已经交叉）接到电脑的两个串口上，然后使用两个同样的软件设置好串口后，做收发实验，应该会在接收窗口看到发送的数据，这说明了串口和线都是好的。**（注意串口号的正确选择）**
   ③ 电脑上只有一个串口：这时就需要两台电脑来完成测试了，方法与②类似。

   这里面要注意软件设置的问题，波特率、停止位数、奇偶校验等信息都要设置相同，端口号也要正确选择后才有可能成功测试串口。

3. 问：KEIL 软件因为什么原因,会通不过编译？
   答：有些书写格式和内容会影响到编译和连接（LINK），在使用时注意使用的关键字和标号，使用不正确会通不过编译。例如我们在程序结尾写：“EEND: END”的结束汇编指令时，就有可能通不过编译。我们将“EEND:”删掉就可以通过编译。在编程时，应该慢慢的收集一些造成编译通不过的原因，为日后编程提供方便。总结经验教训并记录，与学习新的知识同样重要！
   还有一些问题是 KEIL 软件版本引起来的，有些版本没有经过破解，不能编译过大的文件，会提示你的程序超过了多少字节。这些问题可以通过下载破解软件来解决。
   还有一点需要注意，在写函数的时候，我们应该在函数开始就定义变量。如果不这样做就编译不通过。这并不是 C 语言的标准。需要特别留心。可能是 KEIL C 特意制作的吧。记住一句顺口溜：**函数开始就定义！**