目 录

第1章 让你的智能机器人动起来

1.1 欢迎进入 AS-MII 的世界

提起机器人，我们都不陌生，我们脱口就能说出一大串机器人的名字：铁臂阿童木、霹雳五号、奥托曼等，这些都是小说和影视作品里的主人翁，他们的音容笑貌我们仍旧记忆犹新。而现在我们真正的机器人朋友 AS-MII 诞生了！他梦幻般的造型和神奇的智慧将把您带入到智能机器人的奇妙领域。

AS-MII 有一个功能很强的“ 大脑 ”和一组灵敏的“ 感觉 ”器官。它不仅可以随着外部环境敏捷地作出反应，而且还可以与你进行交流。它有听觉、视觉、和触觉，它还会象人一样使用动作和声音，来表达与它周围世界互动时的感觉。如：

AS-MII 会唱歌、会跳舞；
在早晨，他会叫你起床 ；
出去散步的时候他会跟在你后面，边走边唱；
放了学回家，他会哼着小曲欢迎你；
夜晚，他又可以不知疲惫地充当你的忠实卫士，守候在你的身边
保护你；
……

当然，更有趣的是，我们的机器人 AS-MII 就象是一个刚出生的婴儿，大脑一片空白，他可以按照你的意愿成长。他完全听从你的命令，就像一个忠实的仆人顺从着你，陪伴着你，理解你，表达你。

AS-MII 是我们给机器人起的名字。AS 是 abilitystorm 的缩写，意思是能力风暴，M 表示中学版，II- 第二版本。

AS-MII 是不是很有趣，下面就让我们一起仔仔细细地来了解他！

1.2 AS-MII 的外形结构

AS-MII 的外形结构参见图 1-1 ：

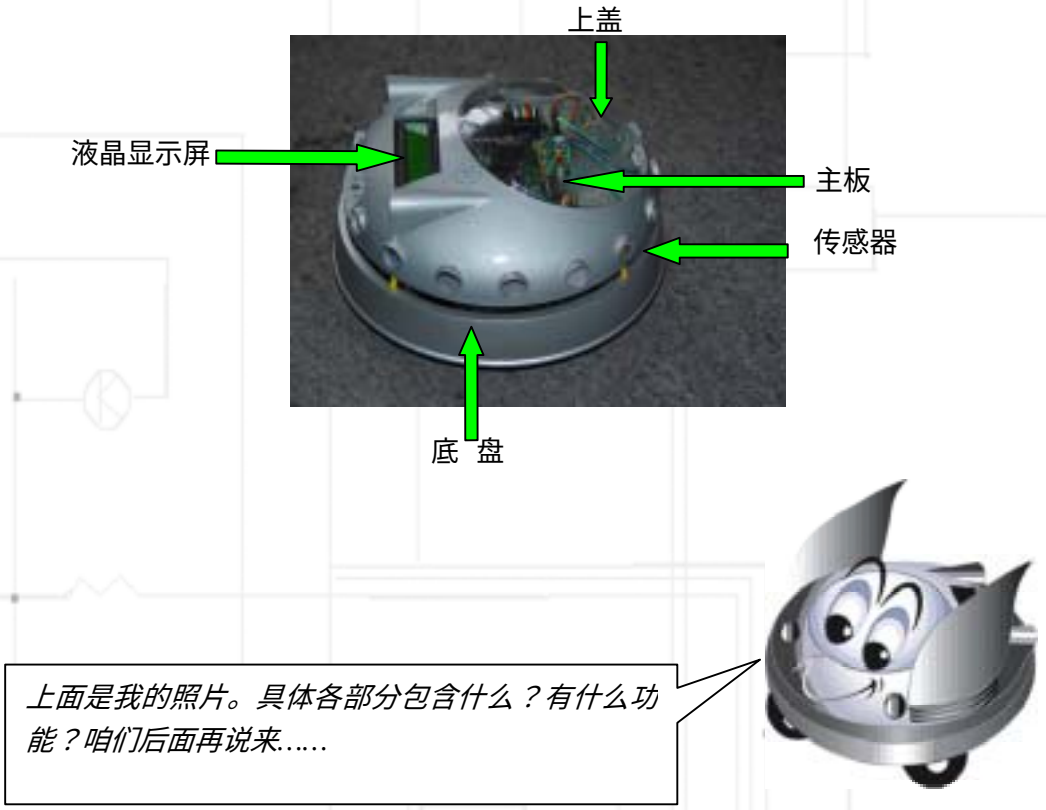


图 1-1 AS-MII 结构简图

1.3 AS-MII 的控制面板

看见 AS-MII 背后的控制面板了吗？就象下面这个样子。在控制面板旁有 2 个小灯，它们会告诉你 AS-MII 所处的状态。

图 1-2 控制按键部分

开关

开关顾名思义就是控制 AS-MII 电源的按钮，按此按钮可以开机或关机，也就是打开或

关闭机器人电源。

电源指示灯

电源指示灯的颜色是绿色。开机时，这个灯会发光，告诉你机器人已经进入工作状态了！关机时，电源指示灯熄灭。

充电指示灯

AS-MII 也有电量不足的时候，就像我们“ 饿 ”了一样，它也会“ 饿 ”，这时需要及时地补充能源。当你给机器人充电时，充电指示的红灯发光。

充电口

要给 AS-MII 充电，充电口这时就派上了用场。只要将充电器的直流输出端插在充电口上，再将另一端接到 220V 电源上即可。具体使用方法见“ 1.4 AS-MII 的充电 ”。

下载口

下载口用于下载程序到机器人主板上，使用时只需将串口通信线的一端接下载口，另一端连接在电脑机箱后面的一个九针串口上，机器人与计算机就连接起来了。具体使用方法见“ 1.6 与 AS-MII 进行交流 ”。

“ 复位/ASOS ”按钮

这是个复合按钮，用于下载操作系统和复位。

复位功能：在机器人运行程序的过程中，按下此按钮，机器人就会中断程序的运行。这时，如果要重新运行程序，须按运行键，或关机再按运行键。

下载操作系统功能：连接好串口通信线，打开机器人电源开关，在 VJC1.5 流程图编辑界面中选择“ 工具（T）--更新操作系统 ”命令，然后按下此按钮，即可下载操作系统。

“ 运行 ”键

机器人开机后，按击“ 运行 ”键，就可以运行最近下载的程序。

通信指示灯

通信指示灯位于机器人主板的前方，是一个黄色的小灯。在给机器人下载程序时，这个黄灯闪烁，表明下载正常，程序正在进入机器人的“ 大脑 ”。

1.4 AS-MII 的充电

AS-MII 可以在线充电，也就是不用取出电池，直接为机器人充电。在线充电又可分为开机充电和关机充电。

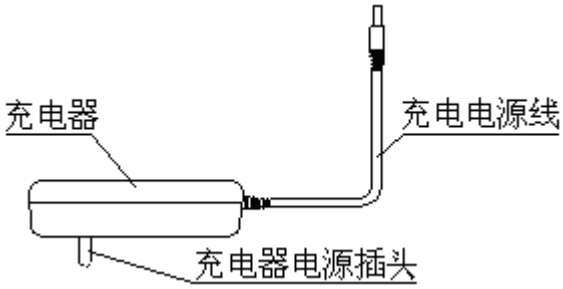


图 1-3 充电器充电示意图

1.4.1 开机充电

AS-MII 可以一边充电一边活动，这样很方便，不会影响你与它的交流。您要采用这种方式给 AS-MII 充电时，只需：

1. 将充电器取出；
2. 充电器直流端接机器人充电口；
3. 充电器另一端插在标准电源插座上（220V，50Hz）。

1.4.2 关机充电

AS-MII 也可以关机充电。只需将机器人关机，再用以上三个步骤给机器人充电。关机充电 1.5 小时即可充满。

1.4.3 更换电池

电池充满电时电压 8.4V，额定工作电压 7.2V，最低工作电压 5V。电池可重复充电 1000 次以上。因为 AS-MII 用的是锂电池，没有记忆和充爆问题，所以可以随用随充。

如需要更换电池，可按下面步骤进行：

1. 关闭 AS-MII 的电源；
2. 将电池取出（须将电池上的小卡片压紧，贴住池身，以脱离卡槽）；
3. 装上新电池。

1.5 AS-MII 的连接和检测

1.5.1 连接串口通信线

在很多情况下，AS-MII 是要和计算机连接使用的。连接串口通信线是一项基本操作，下面是连接的标准步骤：

取出串口通信线。一 端接 AS-MII 的下载口 ，另一 端接电脑机箱后的 9 针串口。如果你的电脑后面没有空余 9 针串口，请咨询电脑维护人员。（可以把暂时不用的设备移开，腾出一个串口。）

1.5.2 运行自检程序

双击 VJC1.5 图标，在打开的对话框中选择“ 新建－ 流程图程序 ”，进入了流程图编辑界面。

在此界面中你可以发现工具栏中有“ 自检 ”按钮  ，点击此按钮，即可下载自检程序。

自检程序下载完毕后，我们就可以检测 AS-MII 了。进行自检时，请拔下串口通信线，将机器人带到安全的地方（空旷，无障碍平地，2 米×2 米大小即可）。

按下机器人电源开关，会听到“ 嘟 ”的一声，LCD 上显示出“ ASOS2002 Grandar Ability Storms ”，同时右下角有太极状的图标在跳。太极图跳动表示 AS-MII 的系统运行正常。按一下“ 运行 ”键，机器人就开始自检了，LCD 上会显示“ AS-MII Intelligent Robot Test ”。自检内容共有九项，一项内容自检完成后， 再按一下运行键，将进行下一项检测内容。下面就让我们逐项看一看：

1. LCD 液晶显示是否正常？ ☐Yes ☐No
   字符显示清晰，16×2 个字符不应有缺行、缺列现象。

2. 扬声器（喇叭）是否正常？ ☐Yes ☐No
   扬声器所播放的乐曲应清晰洪亮，无明显噪声。

3. 光敏传感器是否正常？ ☐Yes ☐No
   左右光敏传感器的感应数值随光强不同而变化，其范围为 0～255。光强越弱，数值越大，光强越强，数值越小。在相同光强条件下，左右两光敏传感器数值偏差小于 10。
   如：(photo L172 R210) 表示左边的光线强。

4. 红外传感器是否正常？ ☐Yes ☐No
   在前方 10cm～80cm 范围内，有 A4 纸大小的障碍物时，在 LCD 上会有“ <<<< ”符号显示，并指明障碍物所在的方位（左前、右前或者正前）。
   如： <<<<
   IR Test
   表示机器人左前方有障碍。

5. 话筒是否正常？ ☐Yes ☐No
   对着 AS-MII 话筒槽孔（蜂窝状小孔）说话，看 LCD 上的 > 是否增加。

6. 碰撞传感器是否正常？ ☐Yes ☐No
   按动机器人下部的碰撞环，在 LCD 上能显示碰撞方位（英文）。

7. 运动系统是否正常？ ☐Yes ☐No
   机器人可移动、转弯，同时在 LCD 上显示光电编码器累计计数值和瞬时电机转速。
   如：

```
Motor     30        L   100
Test      31        R   100
```

表示左电机速度 100，右电机速度 100，左轮转过 30 个单位，右轮转过 31 个单位。

8. 光电编码器是否正常？ □Yes □No

机器人左、右轮子分别转动 1 圈，轮子内侧码盘也随之转动 1 圈，LCD 上显示光电编码器的计数值约为 33。轮子连续转动，LCD 上则显示光电编码器的累计计数值 。

9. 扩展电机是否正常？

扩展电机通常接在机器人主板的 MDCA3 接口上。接扩展电机时，要注意 PA3 的跳线选择（参见图 3.37）。

自检程序全部完成后，按一下复位键，机器人就会停止运行。最后关闭电源开关。也可以不按复位键，直接关闭电源开关。

通过上述检测，我们可以大体了解 机器人各部分的功能，为以后编制程序打下基础。

如果你拿到的 AS-MII 已经被使用过，可能内存里已经没有了自检程序。请按照下一节“ VJC1.5 编程软件简介 ”中介绍的方法下载自检程序，然后再进行自检。

1.6 VJC1.5 编程软件简介

图形化交互式 C 语言(简称 VJC)是用于 AS-MII 的专用开发系统。VJC1.5 运行在 Windows 95/98 /2000/XP 和 Windows NT 4.0 以上版本的操作系统上。VJC1.5 由流程图编辑界面和 JC 代码编辑界面组成 。具体应用参考第四章。

双击桌面上的 VJC1.5 图标，进入流程图编辑界面(如图 1-4)，可以看到这样几个部分 ：菜单栏、工具栏、模块库（包括执行器模块库、传感器模块库、控制模块库和程序模块库)、垃圾箱、流程图生成区、JC 代码显示区。通过点击工具栏中的 “ 编辑 JC 代码程序 ” 快捷按钮（界面切换按钮）可以切换到 JC 代码编辑界面（如图 1-5)。

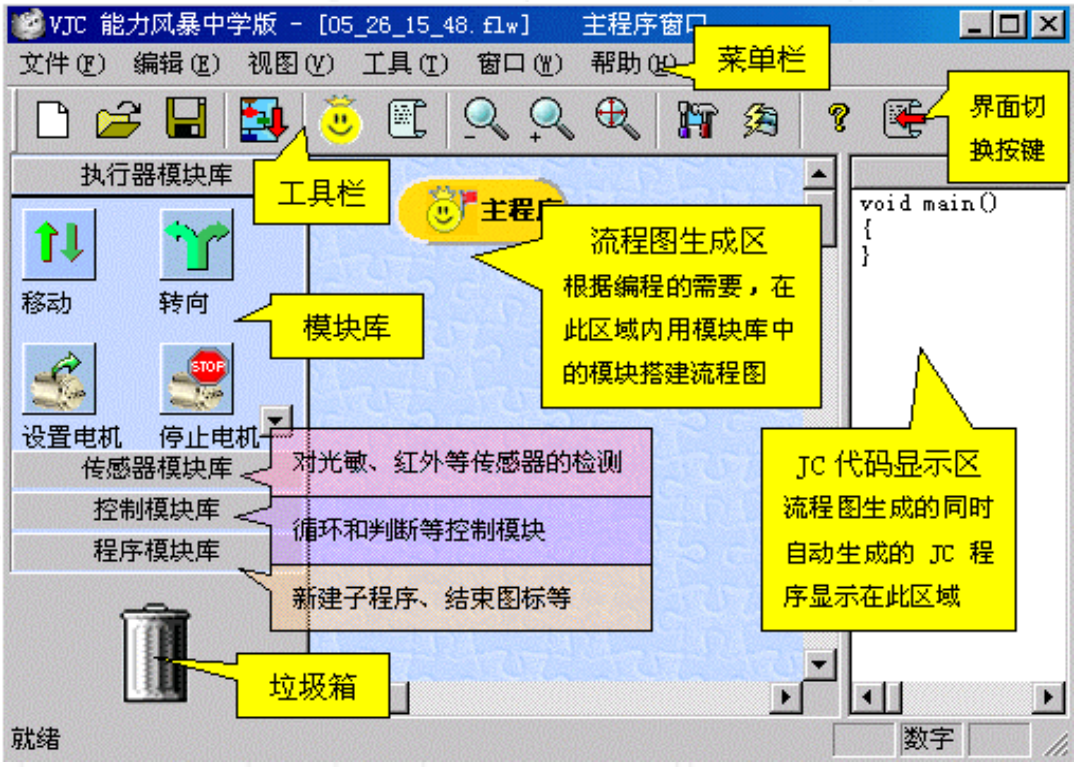


图 1-4 流程图编辑界面

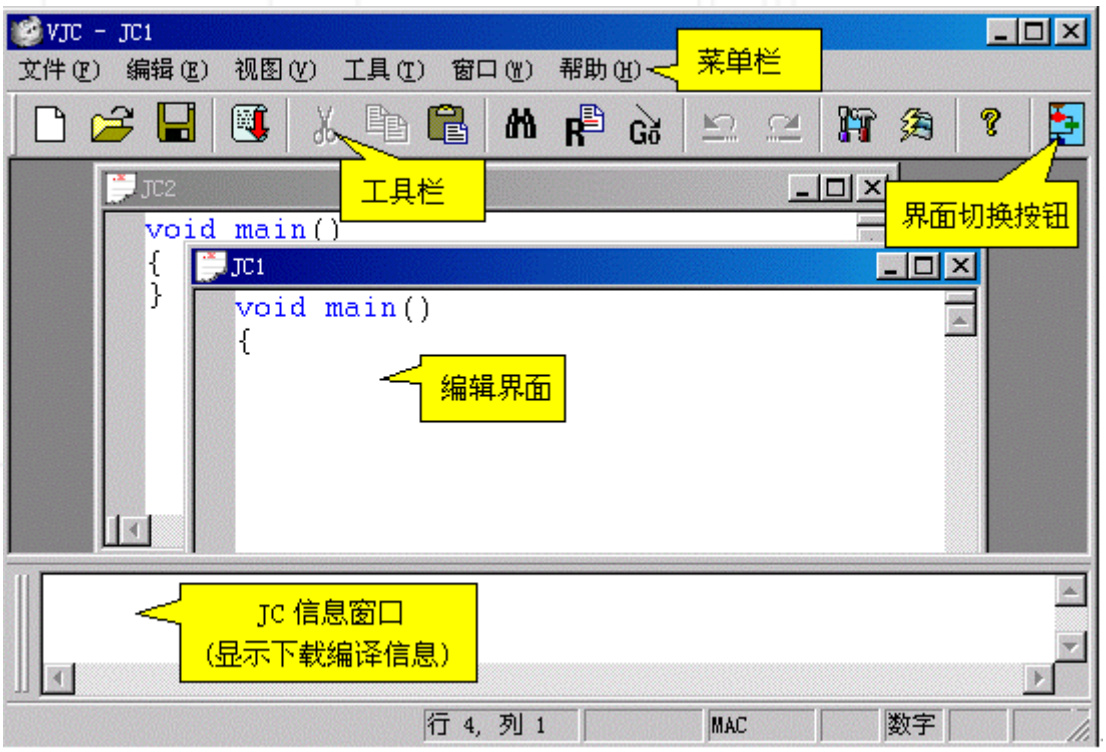


图 1-5 JC 代码编辑界面

VJC1.5 的 JC 代码编辑界面 由这样几个部分组成：菜单栏、工具栏、编辑界面、JC 信息窗口。通过点击工具栏中的“ 流程图窗口 ” 快捷按钮（界面切换按钮）就可以切换到流程图编辑界面。

我们先看看 流程图编辑界面：

新建程序： 要编写流程图程序，可以在流程图生成区直接编写 。如果编辑过之后，还想再新建一个程序，那么可以选择菜单栏中“ 文件—新建 ”命令，也可以利用工具栏里的“ 新建 ”快捷按钮，直接新建一个新程序，见图 1-6：

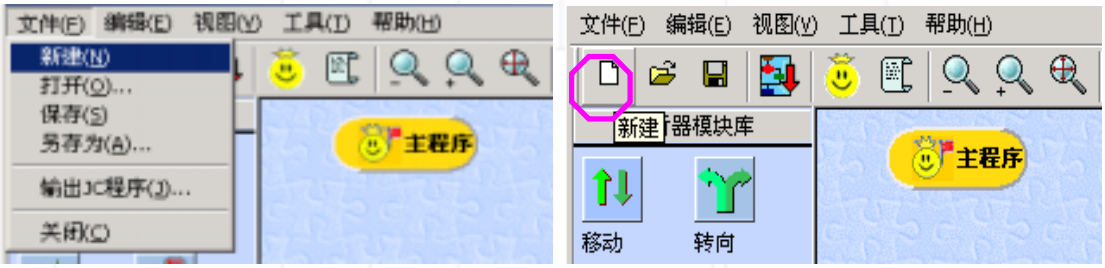


图 1-6 新建程序菜单、图标

打开程序：可以选择菜单栏上“ 文件—打开 ”命令（或点击工具栏中的“ 打开 ”快捷按钮），来查看或编辑以前保存的程序。

下载程序：写好的应用程序必须下载到 AS-MII 上运行。可以选择菜单栏中“ 工具（T）—下载当前程序 ”来下载程序（或点击工具栏中的“ 下载 ”快捷按钮）。如图 1-7：

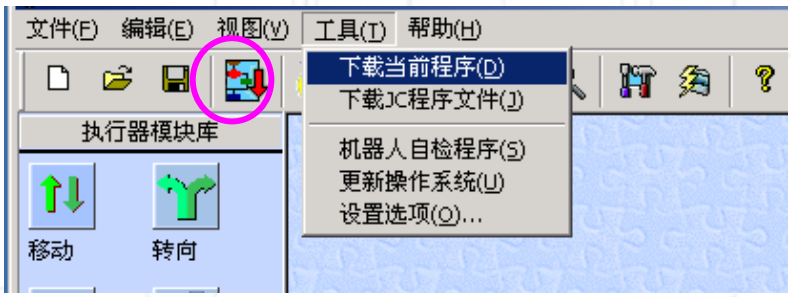


图 1-7 下载当前程序菜单

下载程序时，会弹出如图 1-8 所示“ 智能下载程序 ”对话框，对话框中有两条提示，当你按提示操作无误后，两项提示前端会打勾，然后出现蓝色条码显示编译下载的进程。

图 1-8 下载窗口

下载过程中，你可以看到主板上通讯指示灯（黄色）在闪动，表示数据在传送。如果编译下载都正确，VJC1.5 会提示下载成功，如图 1-10 所示：

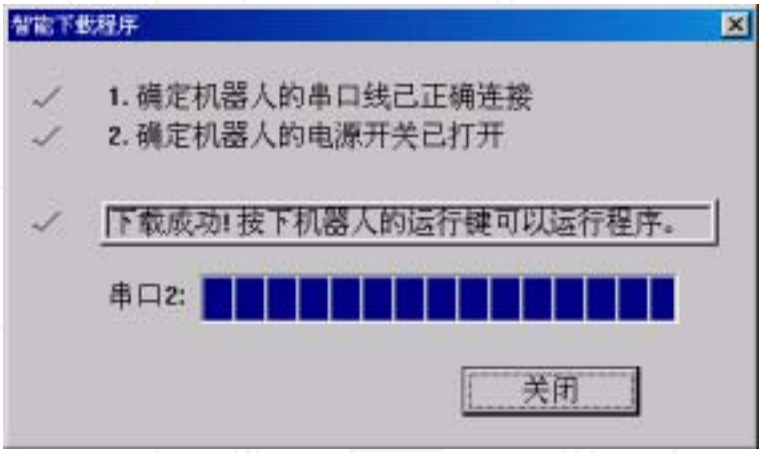


图 1-10 下载成功

下面再看看 JC 代码编辑界面：

新建程序： 可以在 JC 代码编辑界面中直接编写 JC 代码程序。如果你还想再新建一个程序，那么可以选择菜单栏中“ 文件—新建 ”命令，然后在新建的窗口里编写程序。你也可以利用工具栏里的“ 新建 ”快捷按钮，直接新建一个 JC 代码程序，见图 1-5。

打开程序：可以选择菜单栏上的“ 文件— 打开 ”命令，来查看或编辑以前保存的程序。

下载程序：写好的应用程序必须下载到机器人上运行。可以选择菜单栏中“ 工具（T）—下载当前程序 ”来下载程序（或点击工具栏中的“ 下载 ”快捷按钮）。VJC1.5 会在 JC 信息窗口中显示程序的编译下载过程。下载过程中，你可以

看到主板前面的黄灯在闪动，表示数据在传送。

调试程序：所编写的 JC 程序如果有语法错误，那么在编译下载时就会在 JC 信息窗口中显示程序的语法错误，提示错误可能在程序的第几行（用括号注明），并提示可能的错误原因。这时你使用工具栏中的 Go 按钮，就会出现跳转对话框。将出错的行数写入此对话框，光标就会自动跳转到该错误行，那么你就可以找出错误，并修改它。这样反复编译下载，直到没有编译错误为止，就可下载成功。用这种方式可以加快调试程序的过程。

在实际应用中，两种编辑界面切换是经常要用到的。现将编辑界面切换方法总结如下：

流程图编辑界面切换到 JC 代码编辑界面的途径：

- 在菜单栏的“窗口”选项卡中选择任一个“JC 程序”
- 在菜单栏的“编辑”选项卡中选择“编辑 JC 代码”
- 点击工具栏中“编辑 JC 代码”快捷按钮

JC 代码编辑界面切换到流程图编辑界面的途径：

- 在菜单栏的“窗口”选项卡中选择“流程图窗口”
- 点击工具栏中“流程图窗口”快捷按钮
- 打开文件时选择文件类型为“流程图文件*.flw”。

使用键盘上的 F12 键也可以在两个界面之间来回切换。

VJC1.5 的详细操作方法请参见《VJC1.5 操作手册》或直接查询 VJC1.5 的帮助。

第2章 AS-MII 身体的构成

本章分为三个部分，主要描述了 :AS-MII 的身体结构 ，AS-MII 的传动机构，以及 AS-MII 的动力与驱动。

2.1 AS-MII 的身体结构

2.1.1 控制部分——主板和控制面板

控制部分主要是指我们在和 AS-MII 机器人进行交流时，对它进行直接操作的部件，见图 2-1 。从图中我们可以清楚地看到 AS-MII 的控制部分主要由以下两个部分组成：主板和控制面板。

图 2-1 AS-MII 控制部分

1. 主板

主板是 AS-MII 的大脑 ，它由很多电子元器件组成。跟人的大脑一样，它在控制身体的运动时，要完成接收信息、处理信息、发出指令等一系列过程。

AS-MII 的大脑有记忆功能，这主要由主板上的内存来实现。至于“ 大脑 ”的分析、判断、决策功能则由主板上的众多芯片共同完成。

2. 控制面板

位于 AS-MII 背部的控制面板，是 AS-MII 机器人 的按钮和接口集中的地方，它的组成和功能在前面 1.3 中 已做了介绍，这里就不再重复了。

2.1.2 感官部分——传感器

感官部分是 AS-MII 机器人用来同环境进行交流——采集环境信息的一组器件，我们称之为传感器。如图 2-2 所示，AS-MII 上用到的传感器有以下 5 种：碰撞传感器，红外传感器，光敏传感器，话筒，光电编码器。

1．碰撞传感器

AS-MII 机器人的碰撞传感器能够检测到 360°范围内的碰撞，使 AS-MII 机器人遭遇到碰撞之后，能够转弯避开，或作出其它反应。

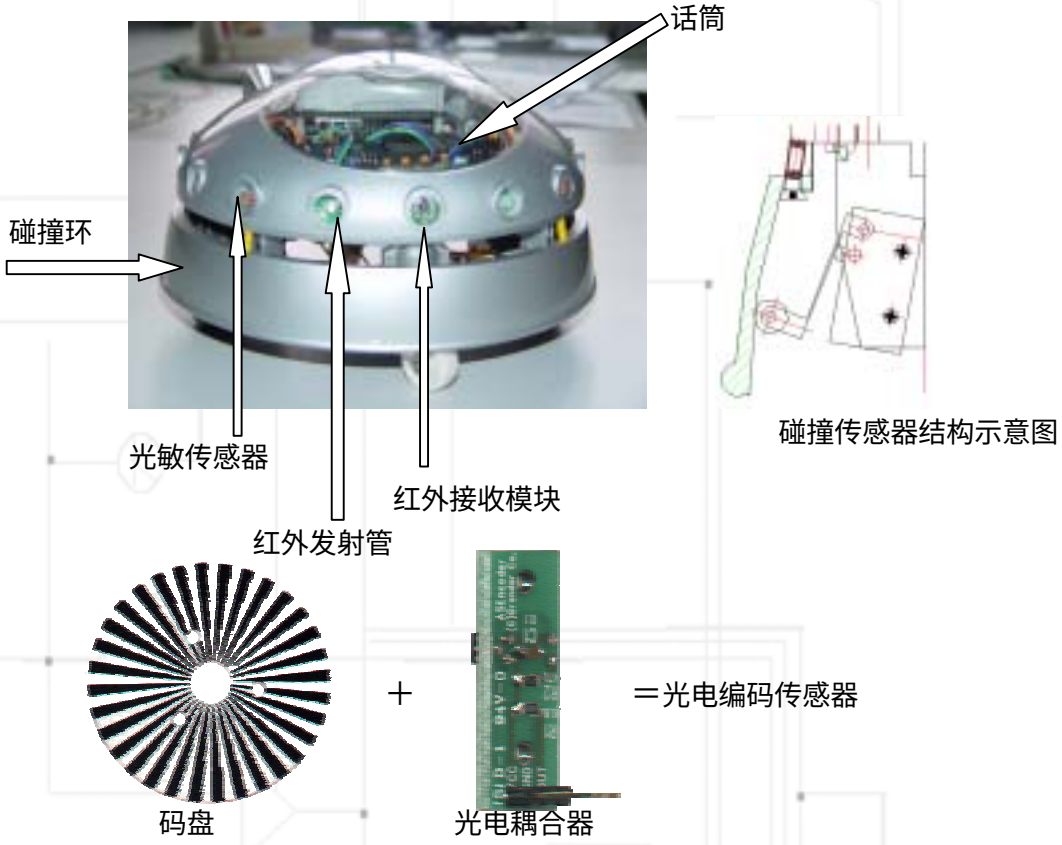


图 2-2　AS-MII,AS-UII 感官部分

2．红外传感器

红外传感器能够和人眼一样“看见”前方的障碍物，然后通知“大脑”作出反应。

红外传感器共包含两种器件：红外发射管和红外接收模块，如图 2-2 所示。红外接收模块位于 AS-MII 机器人的正前方，两只红外发射管紧靠在红外接收模块的两侧，它们共同组成了 AS-MII 机器人的“眼睛”。

红外发射管可以发出红外线，红外线在遇到障碍后反射回来，红外接收模块接收到被反射回来的红外线以后，发出电信号给机器人主板，这样机器人就“看见”东西了。

人的眼睛有时发现不了太小、太近或太远的东西，这是因为人眼有一定的可视范围。AS-MII 机器人的“眼睛”也一样。AS-MII 的“眼睛”能够看到前方 10cm~80cm ，90°范围内的障碍物，障碍物面积须在 210mmx150mm 以上。如果障碍物太小太细、或者在可视范围以外，它可就没法看到了，注意这点哟！

在 AS-MII 机器人的可视距离是可以调整的， 参见后面传感器部分 3．２．２。

3．光敏传感器

光敏传感器是由两个光敏电阻组成，位于机器人的左前和右前方（参见图 2-2）。

光敏传感器不但能够探测光线强弱，而且我们可以让它看见特定的颜色 。 我们可以在光敏传感器上罩一层滤光纸，通过滤光纸的颜色来决定机器人能探测到什么颜色的光线。

4．话筒 （麦克风）

话筒在主板上位于喇叭的内侧，液晶显示屏的下方。 话筒可以检测到声音，也可叫做声音传感器。

话筒能听见的声音频率跟人能听到的大致一样，频率范围大约是 16Hz~20000Hz。

5．光电编码器

光电编码器由码盘（在主动轮内侧）和光电耦合器（在主动轮支架内侧）组成。光电耦合器通过发射红外线，然后测定随轮轴一起转动的码盘的反射脉冲，得出轮子转动的圈数，从而测定机器人行走的距离。

2.1.3 执行部分——喇叭、液晶屏、轮子、电机

AS-MII 机器人的执行部分是执行具体功能时所要用到的部件，如图 2-3 所示。AS-MII 机器人的执行部分共有以下五种：喇叭，液晶显示屏，主动轮，从动轮，电机。

图 2-3 AS-MII 执行部分

1. 喇叭（扬声器）

AS-MII 机器人也可以说话，它的嘴就是喇叭。

当然现在还不能让 AS-MII 机器人的嘴讲“ 人话 ”， 喇叭只能按照输入的频率和时间来发声。

想一想，你是不是可以让 AS-MII 高歌一曲呢！

2. LCD — 液晶显示屏

人类会因为各种原因不能通过声音语言来交流，这时候我们会借助手势或是文字来交流。AS-MII 机器人则利用一种特殊的方式来表达自己，这种方式就是液晶显示屏，简称 LCD。

LCD 显示屏可以显示英文、数字等字符， 告诉你它遇到了什么、正在做什么或是想干什么？别小看这些信息，在你调试程序时它们可有用了。

3. 主动轮及其驱动机构

主动轮有两只，电机输出的动力通过齿轮箱传给主动轮，带动整个机器人运动 。AS-MII 是平面移动机器人，它能够完成直行、走弧线、左转、右转、原地打转这些技术动作 。机器人采用的是差动驱动方式，即每只主动轮由独立的电机驱动，这使得机器人有较高的灵活性。

4. 从动轮

如果 AS-MII 机器人只靠两个主动轮，难以保持平衡。有了从动轮作为支撑，就能实现动态和静态的平衡了。

从动轮共 2 只，与机身弹性连接，可以在垂直于地面的方向上下移动，以保持机器人动态平衡，并能实现一定的越障功能。

5. 直流电机

AS-MII 机器人上有两个直流电机，可以将电池提供的电能转化为动能，让机器人动起来。机器人的运动速度是通过电机来调节的。

2.1.4 AS-MII 的能源—电池

AS-MII 机器人的底盘下安装有电池 ，见图 2-4 。 机器人的能量就来自于这个电池 。

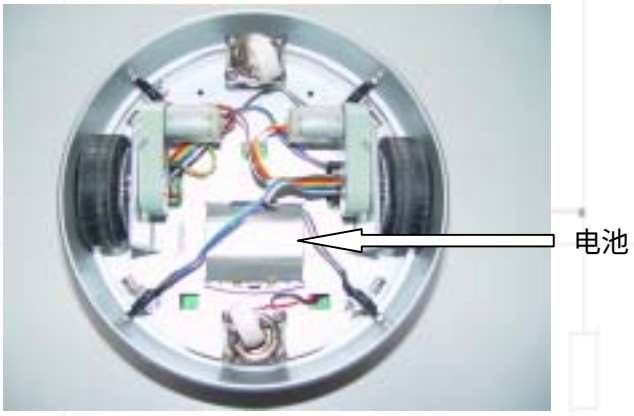


图 2-4 AS-MII 的电池

2.2 AS-MII 的传动机构

2.2.1 齿轮传动机构

齿轮，顾名思义就是带了齿的轮子。齿轮传动机构就是以齿轮为主要传动件的传动机构。工作时， 主动轮先转动起来，与从动轮的齿一个一个 地咬合下去，带动从动轮旋转，这叫做齿轮的啮合传动。

齿轮的种类很多，它们通过不同的配合可以实现不同的传动类型， 图 2-7 是传动类型示意图：

(a) (b) (c)

图 2-7 传动类型示意图

图 2-7(a)所示的传动类型能够将旋转运动变成直线运动。齿轮作旋转运动，下方的齿条则作直线运动。

图 2-7(b)所示传动类型能够改变旋转运动的速度。小齿轮带动大齿轮则减速，大齿轮带动小齿轮则增速。

图 2-7(c)所示传动类型不仅改变旋转运动的速度，而且改变旋转运动的方向。

2.2.2 AS-MII 的齿轮箱

如图 2－8 所示，机器人的齿轮箱是三级减速的，即用了三对齿轮减速机构。也许你会感到纳闷了 :为什么齿轮箱里的传动系统不能用一级减速机构呢？这是因为如果只用一对齿轮，为了达到降低速度的要求，就必须一个齿轮超小，一个齿轮超大，这样不仅会造成空间上的浪费，而且对小齿轮性能指标的要求很高。所以，为了将输出的速度降低，并将输出的扭矩增大，我们采用了三级减速机构，一级一级地改变速度和扭矩，直到将输出轴的速度和扭矩调整到所需要的指标。

对于改变速度的传动形式来说，就有一个传动比的概念。 齿轮传动的传动比 k 可以用两个齿轮的齿数来定义：

$$k = z_1 / z_2$$

Z_1 为主动齿轮的齿数，Z_2 为从动齿轮的齿数

从动轮的转速 $V_{从动轮}$ 可表达为：

$$V_{从动轮} = V_{主动轮} \times k$$

AS-MII 机器人的齿轮箱展开图见图 2-8：

(注：为了能看的更清楚，我们将齿轮箱展开了，实际情况会略有不同)

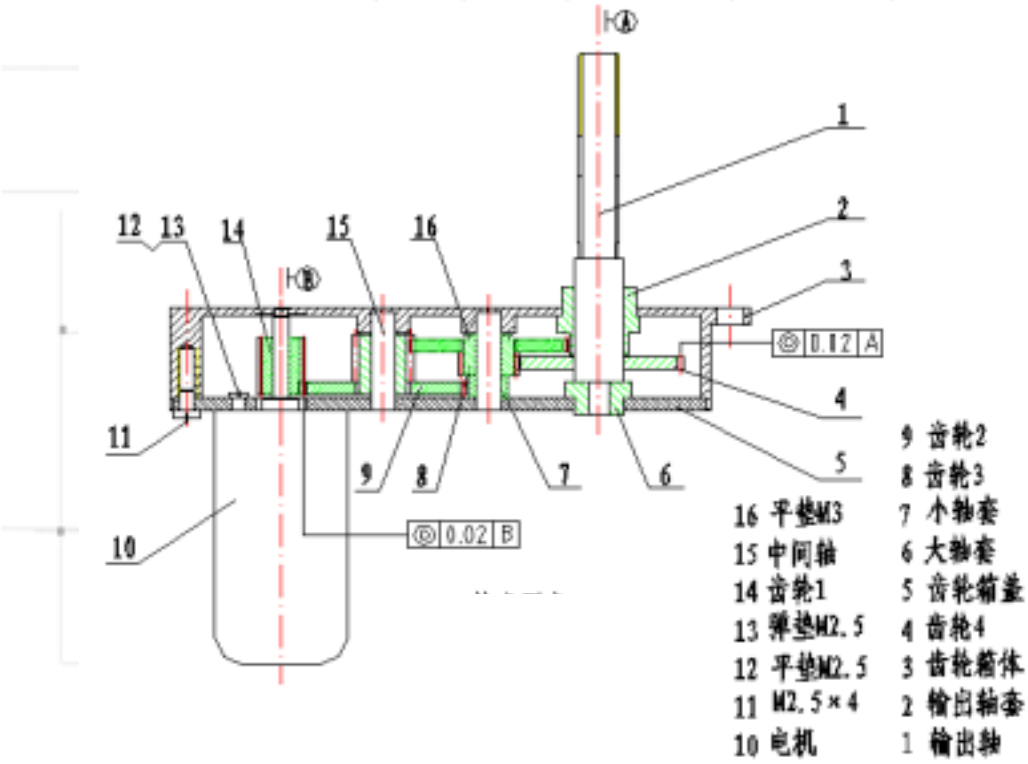


图 2-8 齿轮箱展开图

当然，AS-MII 机器人的传动机构除了齿轮箱以外，还有其他部件，轴就是其中一个关键的部件。

轴在日常生活中是比较常见的，它与轴承配合，支撑齿轮、涡轮等零件，并与之一起旋转，以传递运动、扭矩 。

观察上图，我们可以发现在齿轮箱里有四根轴，它们分别是电机轴、中间轴、中间轴和输出轴。在整个齿轮传动中， 轴的功能有三种：

1. 输入轴---直接与动力源（电机）联接；
2. 输出轴---直接与执行器（轮子）联接；
3. 中间轴---在齿轮箱中仅作为齿轮支架，起过渡的作用。

2.3 AS-MII 的动力与驱动

2.3.1 AS-MII 的动力

AS-MII 的动力来源于位于机器人底盘内的电池。

电池提供电能，而机器人运动需要的是动能。这两种能量是怎么转化的呢？

电能转化为动能须利用电机。电机是现代工业必不可少的，是工业电气化的标志。

2.3.2 AS-MII 中的直流电机

以电为原动力产生机械旋转动力的装置叫做电动机。电动机如果是依靠直流电源工作，则称为直流电机。

在 AS-MII 中，直流电机将轴的旋转运动输入到齿轮箱，齿轮箱将转速降低，将扭矩增大，最后将动力输出给机器人的主动轮，从而驱动整个机器人运动。

机器人的调速是通过调节电机的平均电压实现的，因为直流电机上的电压大小影响它的转速和扭矩。观察下图：

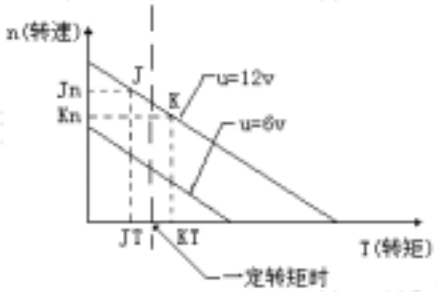


图 2-9 直流电动机 T-N 图

直流电机在一定电压下，其转速与扭矩成反比。在图 2-9 中 12V 转速扭矩线上取两个点 J、K，可以看出 Jn > Kn，JT < KT。由此我们可以发现，当电机电压一定时，转速（n）变小 ，则扭矩（T）增大；转速（n）变大，则扭矩（T）减小，这就叫转速与扭矩成反比。从图中还可以看出，如果电压降低，则转速扭矩线向下方移动。

在智能机器人负载一定时（即扭矩一定时），降低电压，转速扭矩线向下方移动，从而可以降低转速；提高电压，转速扭矩线向上方移动，从而可以增大转速，这样就可以实现调速。在智能机器人里正是采用改变电压的方式，来改变电机转速的。

智能机器人电机上得到的电压是方波，不同的方波的脉冲宽度不同，平均电压也就不同（参见图 2-10），我们就利用这一点来进行电机的速度控制。采用不同的脉冲宽度，调节平均电压的高低，进而调节电机的转速，这种方式叫做脉宽调制（PWM，Pulse Width Modulation）。智能机器人通过主板发出信号，改变脉冲宽度，实现脉宽调制。

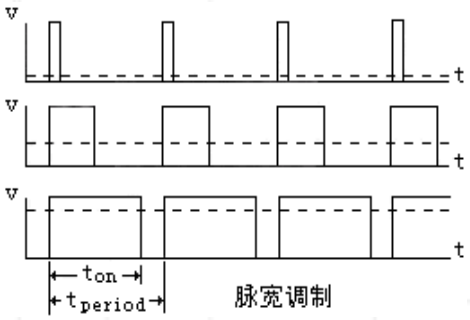


图 2-10 不同宽度的方波（虚线表示平均电压）

2.3.3 AS-MII 的驱动方式

AS-MII 机器人的驱动方式是差动驱动。

差动是指将两个有差异的或两个独立的运动合成为一个运动。当我们把两个电机的运动合成为一个运动时，就形成了差动驱动。

仔细观察智能机器人的底盘，你会发现机器人有两个直流电机。两个直流电机分别 控制两个主动轮。在运行时，我们可以分别设置两个电机的转速，组合起来就能使机器人实现 各种运动方式，如直行、转弯、走弧线等，这就是差动驱动。

想一想，既然我们的智能机器人是差动方式驱动的，由两个直流电机分别控制。那么，智能机器人能走出多少种不同的路线来呢？

下面我们以 JC 代码编程为例，介绍两个库函数 ，以便更深刻地理解机器人的差动驱动方式。这两个库函数是 motor()和 drive()。

库函数 motor（a，b）应用时应注意：

1. 库函数 motor（a，b）只控制单个的电机转速；
2. 库函数 motor（a，b）有两个参数 a、b，都是整型数；
3. 库函数 motor（a，b）中 a 指定是左轮或是右轮，a=1 代表左轮，a=2 代表右轮；b 指定转速，b 的取值范围是－100～＋100。

例 1：在 JC 代码编辑界面中写入以下程序

```
void main()
{
  motor( 1 , 40 );
  motor( 2 , 80 );
  wait( 0.500000 );
  stop();
 }
```

然后下载运行，你会发现，机器人以左轮 40 右轮 80 的速度前进了 0.5 秒钟，划了一条弧线。

库函数 drive（a，b）应用时应注意：

1. 此库函数是复合语句，同时控制左右两个电机的转速；
2. 此语句有两个参数 a、b，都是整型数；
3. a 指定平移的速度，b 指定旋转的速度。左轮的速度＝a＋b，右轮的速度＝a－b。

观察下面两个例子，看看它们有什么不同：

例 2：
```
void main()
{
   drive( 80 ,0);
   wait( 0.500000 );
   stop();
 }
```

例 3：
```
void main()
{
   drive( 60 ,20);
   wait( 0.500000 );
   stop();
}
```

请下载这两个程序运行，验证您观察的结论。

VJC1.5 的详细操作方法请参见《VJC1.5 使用手册》或直接查询 VJC1.5 的帮助。在下面的第 5 章《尝试迷人的机器人项目》里，我们还将作进一步的介绍。

第3章 感觉、大脑与执行器

3.1. 智能机器人的三大要素

人对周围环境的反应过程是：感觉→大脑思考→作出反映。机器人的信息处理流程也是如此。机器人的信息处理流程由以下三大要素来完成：传感器，微控制器和执行器。

能力风暴智能机器人配有 5 种十几个传感器，另外还可以根据需要扩展其他传感器。能力风暴对环境的感知能力很强。感知环境的能力是产生智能行为的前提，因此能力风暴能产生许多智能性行为。

能力风暴通过微控制器（microcontroller）来思维。我们采用的是 Motorola 公司 8 位单片机中功能最强、集成功能最全的高档机种 68HC11E1。它的可靠性很高，有程序自下载功能。

计算机硬件决定了机器人的极限潜能，去开发这种潜能是软件的工作。为了充分利用 68HC11E1 的功能，我们为用户提供了图形化交互式 C 语言—VJC1.5，它使开发能力风暴的高层行为充满了乐趣。

能力风暴智能机器人的执行器有：二只高性能直流电机；一只喇叭；一只 2*16 字符的液晶显示器。

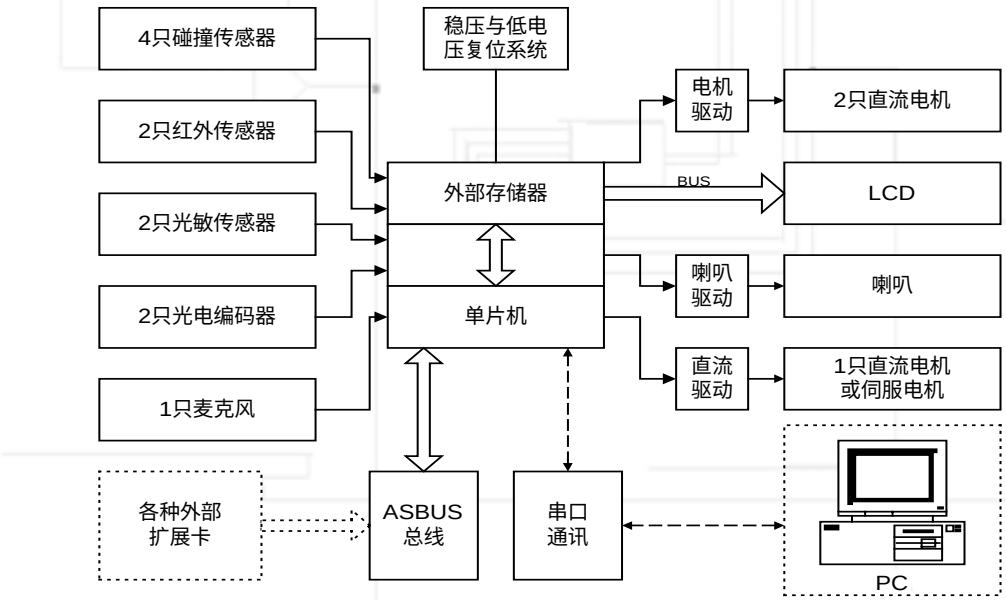


图 3．1 能力风暴智能机器人的系统结构

3.2. 能力风暴的传感器及其处理电路

3.2.1. 碰撞传感器

碰撞传感器是感知碰撞信息的传感器。在能力风暴智能机器人的左前、右前、左后、右后设置有四个碰撞开关，它们与碰撞环共同构成了碰撞传感器(见图 3.2)。碰撞环与底盘柔性连接，在受力后与底盘产生相对位移，触发固连在底盘上的碰撞开关，使之闭合。

图 3.2 碰撞开关及碰撞环

我们把来自四周的碰撞分为八个方向(见图 3.3)。

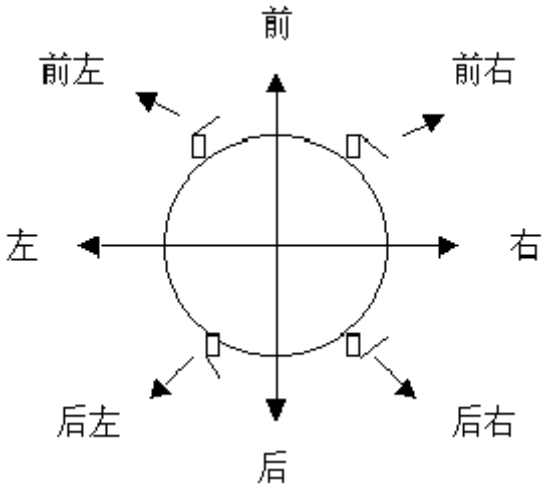


图 3.3 碰撞传感器方位

应用

我们在 VJC 环境中，编写一个碰撞检测程序，来理解如何在程序中使用碰撞开关。VJC 流程图编辑界面参见图 1-4：

1 进入 VJC 的 流程图编辑界面，将“ 控制模块库 ”中的“ 永远循环 ”模块拖入到流程图生成区，并与“ 主程序 ”模块相连；

2 将“ 传感器模块 ”库中“ 碰撞检测 ”模块连接到循环体内部，见图 3-6；

3 将“ 执行器模块库 ”中的“ 显示模块 ”连接在“ 碰撞检测 ”模块下方；

4 设置“ 显示 ”模块。在“ 显示 ”模块上右击鼠标，在弹出的对话框中选择“ 引用变量”，会出现“ 变量百宝箱”对话框（见图 3-5），点击“ 碰撞 ”图标，在碰撞变量引用中点击“ 碰撞变量一 ”，按“ 确定 ”；

图 3-5 VJC 碰撞显示变量设置图

5 在“ 显示 ”模块正下方，连接“ 延时等待 ”模块，在“ 延时等待 ”模块上右击鼠标，设置时间为 2 秒；

6 将“ 程序模块库 ”中的 “ 任务结束 ”模块连接在循环体的下方，完成碰撞检测程序的编写。

图 3-6 碰撞检测程序流程图

下载并运行此程序，观察 LCD 上的显示：

bmp_1=0　　（表示此时没有碰撞）

若左后方受到碰撞，LCD 上显示：

bmp_1=4

在其它方向施加碰撞，LCD 显示的值将不同，八个方向发生碰撞时返回值的意义为：

1 左前，2 右前，4 左后，8 右后，3 前，12 后，5 左，10 右。

上面使用的是流程图程序，下面我们再用 JC 语言（交互式 C 语言）编写程序来理解碰撞传感器是如何工作的。

在 JC 语言中，碰撞传感器的库函数是 bumper()，在程序运行过程中**此库函数仅在被调用到时执行一次，即采集数据一次。**因此要连续查询碰撞传感器的状态就要在 JC 代码编辑界面中间一行编辑框（JC 对话窗口）中输入如下程序块（while(1)表示永远循环）：

```
{while(1) {printf("bump=%d\n",bumper());wait(0.1);}}
```

按回车（Enter），JC 能立即编译这一段程序并下载运行（须连接串口通信线并开机），LCD 上显示：

bump=0　　（表示此时没有碰撞）

按左前碰撞环，LCD 上显示：bump=1。在其他方向施加碰撞，显示的值将不同。

（JC 代码编辑中的这种交互式能力特别便于调试、测试和学习）

安装　以下是碰撞传感器的接线图

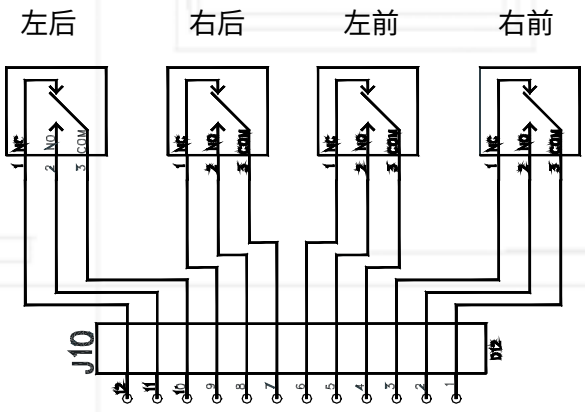


图 3.4　碰撞传感器的接线图

用户在使用时须注意四个碰撞开关的安装位置，然后将组合插针以正确的方向插入主板相应的接口（参见图 3－5）。

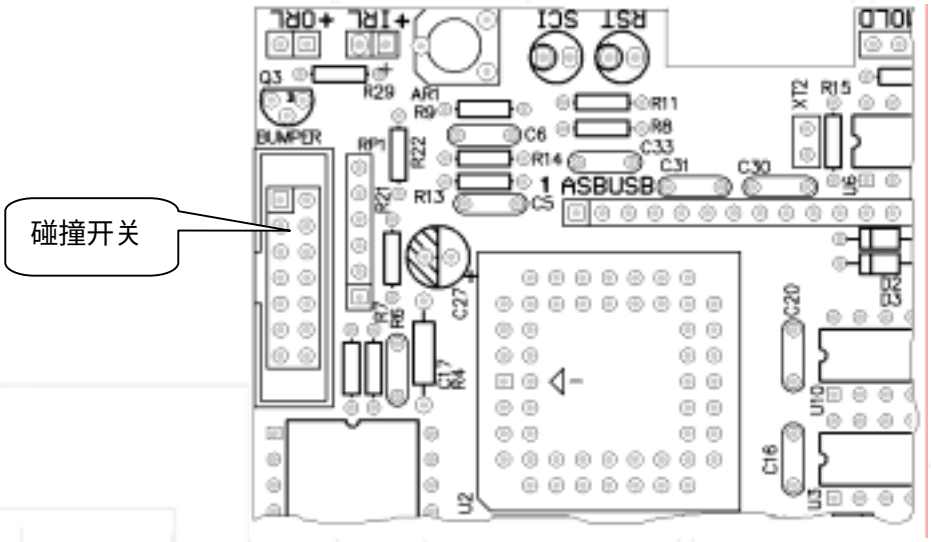


图 3.5　碰撞传感器的插针位置图

原理

到此，碰撞传感器已经能够被用户直接使用。但是，对应于每一个方向的碰撞，用户是怎样得到一个二进制数值的呢？

在能力风暴智能机器人里，四个碰撞开关接在一个电阻网络里，通过采集模拟口 PE3 上电压值的变化，来识别出哪些碰撞开关闭合，从而判断出哪个方向有碰撞。 为了帮助理解，让我们在 JC 代码编辑界面中间一行编辑框（JC 对话窗口）中输入如下程序块：

```
{while(1) {printf("pe3=%d\n",analogport(3));wait(0.1);}}
```

按回车（Enter），JC 能立即编译这一段程序并下载运行（须连接串口通信线并开机），LCD 上显示：

```
pe3=0   (表示此时无碰撞)
```

在各个方向上按压碰撞环，pe3 返回值各不相同（参见图 3－6），这些值是 pe3 上电压值通过模数转换（模拟量转化为数字量）得到的结果。它们不直观，也不方便记忆。对此在库函数 bumper()中进行了变换，将模拟口上采得的电压值变为直观表示各个方位的四位二进制数（参看 VJC1.5\robots\AS-MII\libs\lib_AS_MII.jc 中的子函数 int bumper()。），这样就得到了在应用中的对应关系、计算方式和调用方法。

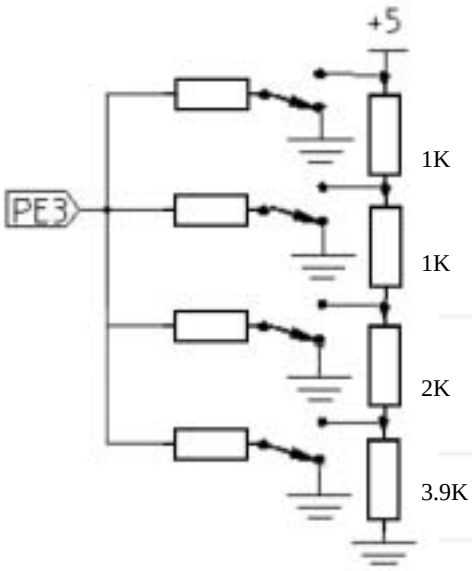


图 3.6　碰撞传感器的电路图

范例

```
void main()
{
    int  bumpvalue;              /*  定义整形变量bumpvalue */
    while(1)                     /*  永远循环 */
    {
      bumpvalue=bumper();        /*库函数bumper () 返回碰撞检测值给bumpvalue*/
      printf("bump=%b\n",bumpvalue);
                                 /*以二进制形式打印出碰撞传感器的检测值*/
      wait(0.5);                 /*等待0.5秒*/
     }
}
```

运行以上程序，前后左右按碰撞环，可直接从显示屏上读出显示值。

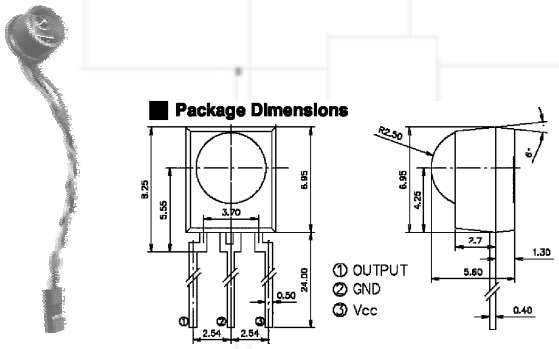
3.2.2. 红外传感器

能力风暴运用了 2 只红外发射管（970nm）和一只红外接收模块构成红外传感系统（见图 3.7），主要用来检测前方、左前方和右前方的障碍，检测距离范围为 10～80cm。

用户可以通过调节两个电位器（主板上靠近红外传感器接口的黄色旋钮）来调节左右两个红外的检测距离，顺时针红外发射强，检测距离远，逆时针红外发射弱，检测距离近。逆时针将电位器旋转到底，将关闭红外发射管。

主板中的 XT2 为 38kHz 的晶体，它将红外光发射的调制频率固化在 38KHz 左右，这是红外接收模块中带通滤波器的中心频率。

红外光发射管



红外接收模块

图 3.7 红外传感器

从图 3-7 中可以看出红外发射管的头部象一个发光二极管，它是两针的；红外接收模块的头部是个集成块，它是三针的。

安装

红外传感器的插针是有方向性的，用户自己安装时应注意方向。

红外线发射接口 IRR 和 IRL 的正极已经标出。插反不会损坏元件，但传感器会不工作。

红外线接收模块的正确接法是将紫色线朝向主板中间的缺口。如果插反不会损坏器件，但错位有可能损坏红外接收模块。

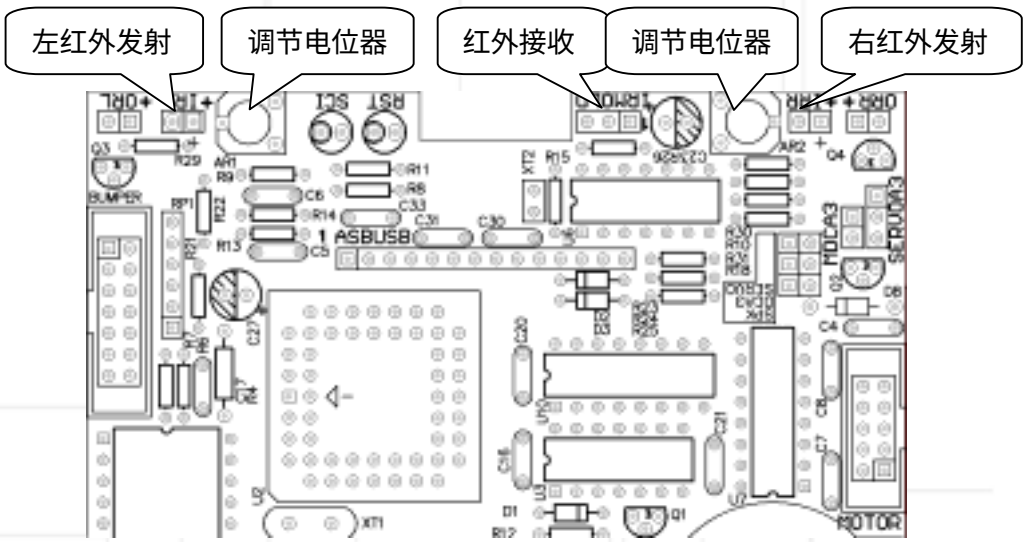


图 3.8　红外传感器插针位置图

应用

在 JC 语言中，红外传感器的库函数是 ir_detector()，在程序运行过程中**此库函数仅在被调用到时执行一次，即采集数据一次。**

在 JC 代码编辑界面中间一行编辑框（JC 对话窗口）中输入如下程序块：

```
{while(1) {printf("ir=%b\n", ir_detector());wait(0.5);}}
```

按回车，JC 能立即编译这一段程序并下载运行，LCD 上显示：

ir=0　　（表示此时没有障碍）

用一张白纸分别挡在能力风暴智能机器人的前方、左方和右方，液晶显示屏上显示的 ir 的值都不一样，可总结如下：

	无障碍	左方	右方	前方
十进制表示：	0	1	2	4

原理

红外接收模块集成了红外接收管、前置放大器、限幅放大器、带通滤波器、峰值检波器、整前电路和输出放大电路，灵敏度很高。有时从红外管侧面和后面漏出的红外光也会被接收模块探测到。在能力风暴智能机器人上，两个红外发射管和一个红外接收模块都是先装在套管里再固定在外壳上的，有效地避免了这种情况的发生。用户在自己扩展红外传感器时，如果遇到这种情况，只需用黑胶布把发射管的侧面和后部包住即可。

红外传感器是靠发射红外光，并接收由障碍物反射回来的红外光来判断是否有障碍的（如图 3.9）。

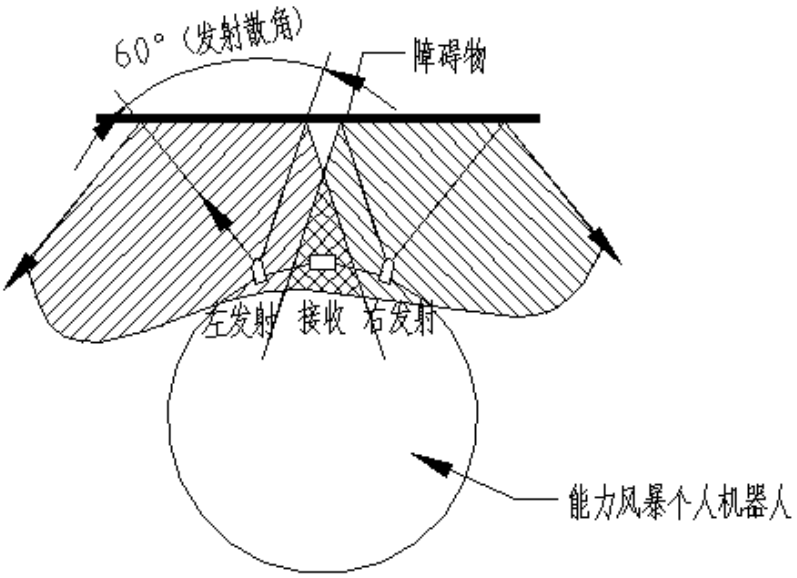


图 3.9 红外测障原理图

图 3.9 中所示即为前方有障碍时的情况，红外光以 60 度的散角向外发射，阴影区域分别是左右两个红外的反射光区域，而红外接收模块正好处于反射光区域内，能接收到左右两个红外发射管的反射光，由此即判断出前方有障碍。

应特别注意，红外接收模块只是在接收到了一定强度的红外光时才有反应， 判断有障碍。所以，当障碍物太细时，机器人会检测不到；当障碍物是黑色或深色时，会吸收大部分红外光， 只反射回一小部分，不足以产生有障碍的信号。

下面是红外传感器的电路图：

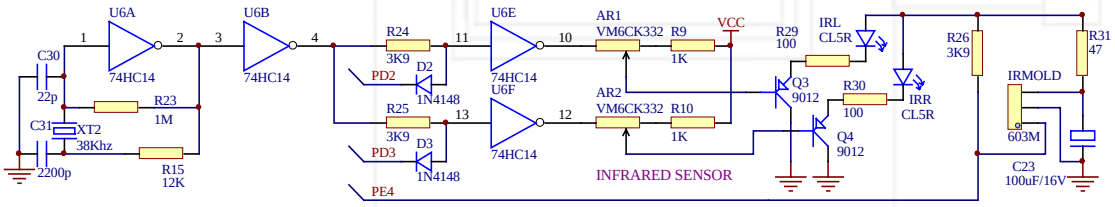


图 3.10 红外传感器电路图

从图 3.10 中可以看出，I/O 口 PD2，PD3 分别控制左右红外光发射管的关闭和打开，AR1 和 AR2 控制左右红外光发射管的发射强度。

红外传感器检测障碍的过程如下：

1. 首先，左右发射管均关闭，红外探测器探测一次当前信号，并保存下来，以跟后面采集到的数据比较。
2. 当程序中调用库函数 ir_detector()时，红外探测系统启动。 左红外发射管先发射一次，延时 1ms 后红外接收模块探测一次信号；然后，右红外发射管发射一次，延时 1ms 后红外接收模块探测一次信号。

红外接收模块采样一次信号的时间为 0.064ms。红外接收模块通过 PE4 口采样当前值，并保存下来。由于先后时间的不同，就可以分别探测左右两边的红外信号。

3. 每调用一次 ir_detector()函数，红外探测系统开启一次。完成后，左右红外发射管关闭 。

根据采集的数据可以判断是否有反射。 在初始探测无反射而第二次探测有反射时，左反射管才是有反射的，这样系统认为左方有障碍。同理，初始探测无反射而第三次探测有反射时，右反射管才是有反射的，右方被认为有障碍。采用这种方法可以抑制许多环境红外噪音。

范例

```
void main()
{
   int ir;                  /*定义一个整型变量ir*/
   while(1)
    {
      ir=ir_detector();     /* 把库函数ir_detector()的检测值赋给ir */
      if(ir!=0)             /*判断有否障碍*/
         {
          drive(-60,-30);
          wait(0.1);        /*有障碍就倒退并转向 0.1 秒*/
         }
      drive(40,0);          /*否则前进*/
     }
}
```

运行此程序，将会看到，在机器人前进过程中，当前方(左前，右前，正前方)有障碍时，能力风暴智能机器人会作出反应。

3.2.3. 光敏传感器

能力风暴智能机器人上有 2 只光敏传感器（见图 3.11），在机器人左前和右前方，可以检测到光线的强弱。

图 3.11 光敏传感器

光敏传感器其实是一个光敏电阻，它的阻值随光线强弱而变化。能力风暴智能机器人所用的光敏电阻的阻值在很暗的环境下为几百 KΩ，室内照度下几 KΩ，阳光或强光下几十 Ω。

安装

光敏传感器是可变电阻，它的接插方式没有方向性，它在主板上的位置如图 3.12 所示。

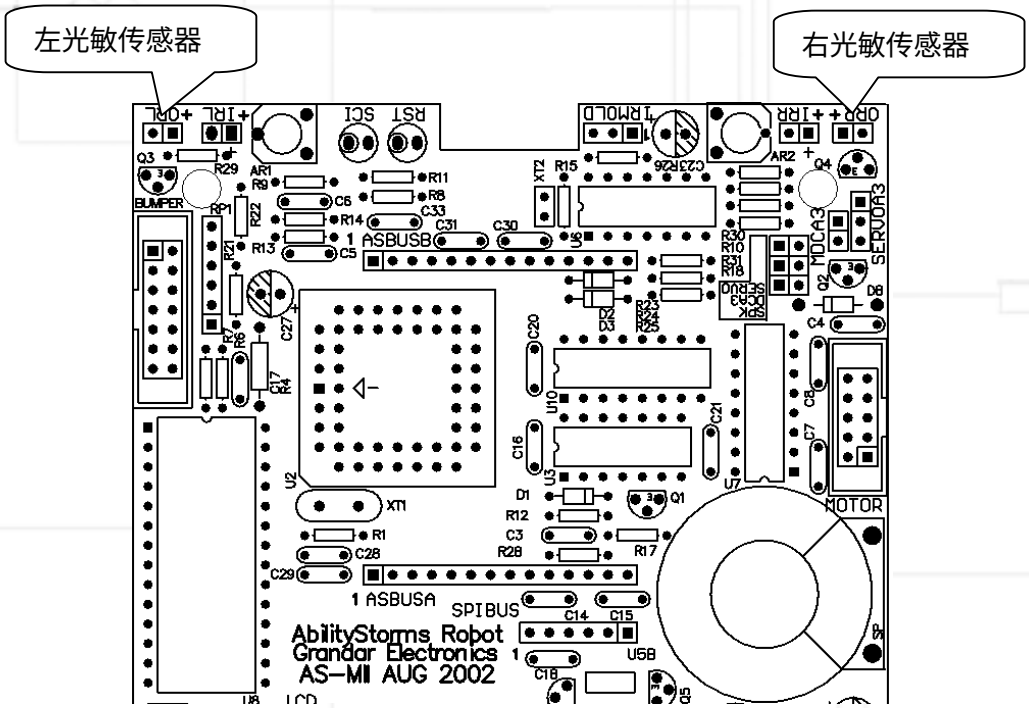


图 3.12 光敏传感器插针位置图

应用

在 JC 语言中，光敏传感器的库函数是：左光敏 photo(1)，右光敏 photo(2)。在程序运行过程中光敏**库函数仅在被调用到时执行一次，即采集数据一次。**

在 JC 代码编辑界面中间一行编辑框（JC 对话窗口）中输入如下程序块：

```
{while(1) {printf("photoleft=%d\n",photo(1));wait(0.5);}}
```

（程序中仅采集了左光敏的读数）

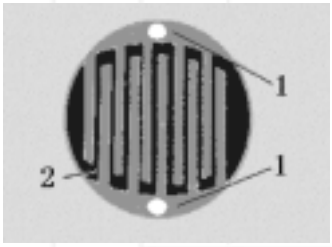
按回车（Enter），JC 能立即编译这一段程序并下载运行，LCD 上会有如下显示：

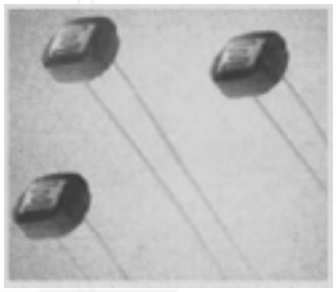
photoleft=180　　（表示此时照射在左光敏上的光强值是 180）

注意观察读数的变化。光越暗，数字越大，光越强，数字越小。

原理

以下是光敏电阻的结构及工作原理图





1．电极　　2．光导体

图 3.13　光敏电阻的电极图案

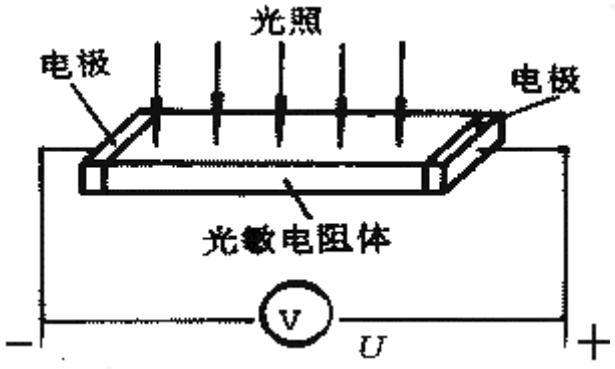


图 3.14　光敏电阻的工作原理图

光敏传感器 工作原理如下：光敏电阻和 10K 的电阻 R13，R14 相连后构成分压器。左右两个光敏电阻分别与模拟输入口 PE0，PE1 相连，在系统中采集的是光敏电阻上的电压值。光暗时，光敏电阻上的电压接近 5V，光强时，接近 0V，模数转换为 8 位数字量后的范围为 0-255。参见光敏传感器的电路图（图 3－15）：

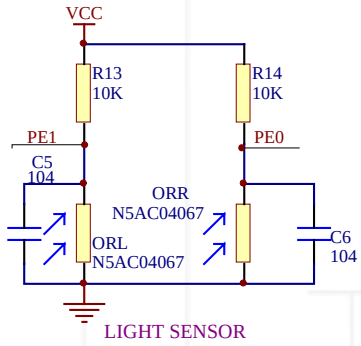


图 3.15 光敏传感器的电路图

范例

在代码编辑界面中写入以下程序：

```
void main()
{
  int right;
  while(1)
    {
      right=photo(2);
      if(right>200)
          {beep();
           beep();
          }
    }
}
```

下载运行此程序，当光线暗于一定值时，能力风暴智能机器人将叫两声。

3.2.4. 麦克风（话筒）

能力风暴智能机器人上的麦克风(microphone)是能够检测声强大小的声音传感器（见图 3.16），麦克风的安装位置在主板上喇叭的内侧，液晶显示屏的下方。

图 3.16 麦克风

应用

在 JC 语言中，声音传感器的库函数是 microphone( )，在程序运行过程中**此库函数仅在被调用到时执行一次，即采集数据一次。**

在 JC 对话窗口中输入如下程序块：

```
{while(1) {printf("mic=%d\n", microphone());wait(0.5);}}
```

按回车，JC 能立即编译这一段程序并下载运行，如果周围的环境很静 LCD 上显示：

mic = 12 （表示此时很安静）

对着麦克风发出声音，可以看到显示值不断变化。它的变化范围是 0～255。

原理

麦克风采集到的信号通过 LM386（U1）进行放大，放大倍数为 200（由 C25 确定），输出信号接至 PE2。没有声音时，电压为 2．5V 左右，转换为 8 位二进制数后得到的十进制整数为 127 左右（如图 3.18），库函数 microphone()对数据进行处理，使返回值为 0～255。

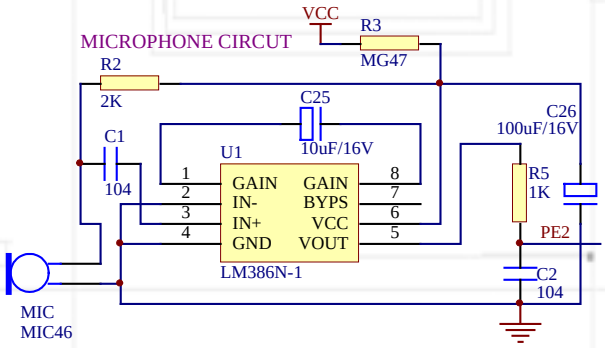


图 3.18 麦克风电路图

当有声音时，LM386 的输出电压在 2.5V 上下波动。PE2 测得的电压和 2.5 V 相减的绝对值越大，则声音越大。R5、C2 构成高频滤波，滤去线路板其他元器件产生的高频噪声。

范例

```
void main()
{
int b=1;                                /*状态标志位*/
int mic;
while(1)
    {
     mic= microphone();                 /*检测麦克风*/
     printf("mic=%d\n",mic);            /*显示检测值*/
     wait(0.5);                         /*延时0.5秒，可调整*/
        if(mic>40) b=b*(-1);            /*条件满足，改变运动状态*/
        if(b==-1) drive( 20 , 0 );      /*前进*/
        if(b==1) stop();                /*停止*/
    }
}
```

运行上面的程序，第一次声音大于 40 时， 机器人前进；第二次声音大于 40 时，机器人停止运动；如此循环。

3.2.5. 光电编码器

光电编码器是一种能够传递位置信息的传感器，它由码盘和光电编码模块组成（见图 3.19），分别安装在主动轮内侧和轮子支架内侧。光电编码模块运用反射式红外发射接收模块。反射器(即码盘)是黑白相间的铝合金圆片，黑白条纹把圆分成 66 等分。当码盘随轮子旋转时，光电编码模块发出的红外线照射在码盘上，黑条和白条反射回来的信号状态不同，从而产生一个脉冲。轮子转一圈共产生 33 个脉冲，每个脉冲对应角度约为 10.91 度。

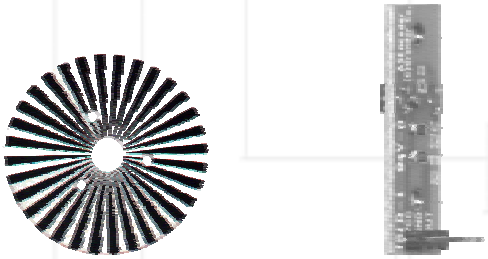


图 3.19 码盘及光电编码模块外形

安装

码盘装在轮子的内侧，光电编码模块的插针位置如下图所示：

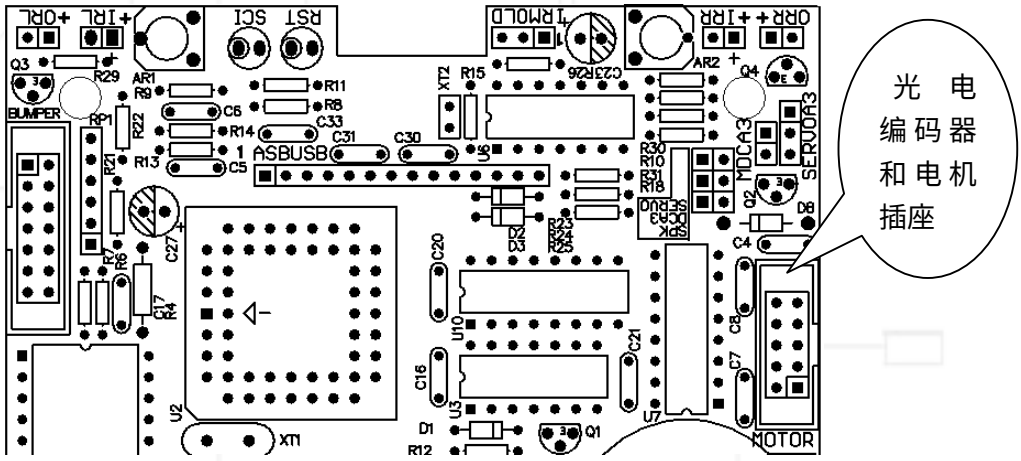


图 3.20 光电编码芯片的插针位置示意图

应用

检测左右光电编码器当前状态的库函数为：encoder (1)，encoder (2)。

在 JC 对话窗口中输入如下程序块：

```
{while(1){printf("encoder_1=%d\n",encoder(1));wait(1.0);}}
```

按回车（Enter)，JC 能立即编译这一段程序并下载运行，LCD 上会显示 0 或 1。0 表示当前无反射信号，码盘片的黑格正对编码器；1 表示当前有反射信号，码盘片的白格正对编码器。然后缓慢转动左轮，看转一圈是否有 33 个脉冲。

库函数 rotation(1)、rotation(2)可以读出左右光电编码器脉冲累计值。

原理

光电编码器是靠发射与接收红外光来工作的。能力风暴智能机器人上用的光电编码模块集成了发射与接收功能，参见图 3．２１。

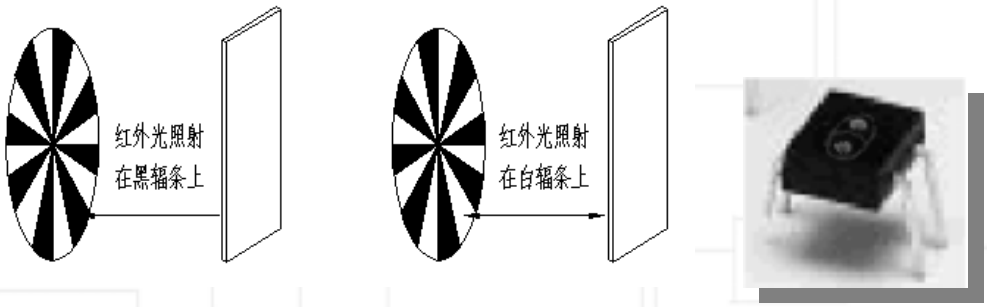


图 3.21 光电编码器的工作原理图

从图中可以看出：红外光射在黑色辐条上时没有反射信号，因为红外光大部分已经被黑色辐条吸收 ;当红外光射在白色辐条上时有反射信号,因为红外光在白色辐条上反射强烈。

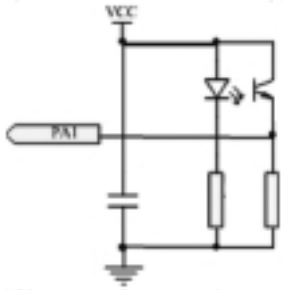


图 3.22 光电编码器电路图

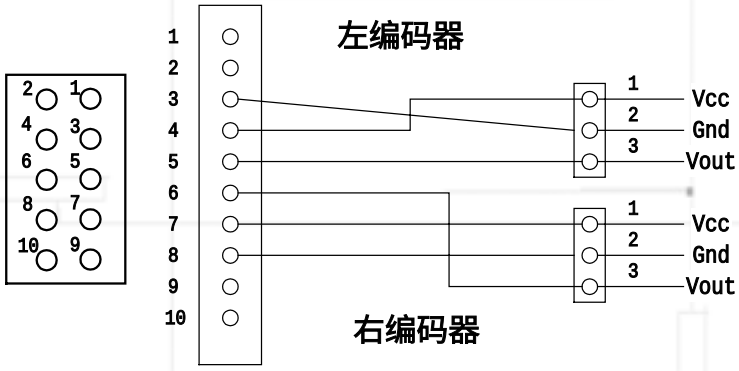


图 3.22 光电编码器和 motor/encoder 插座的接线图

范例：

```
void main()
{
  rotation(1);                  /*调用一次，目的是为了将脉冲记数清零*/
  drive(70,0);                  /*前进*/
  wait(2.0);                    /*延时2秒*/
  printf("left=%f\n",(float)rotation(1)/33.0);
                                /*读出的脉冲数值除以33即得左轮旋转圈数*/
  stop();                       /*停止*/
 }
```

以上是 int rotation(int index)函数的应用。运行此程序，不同的速度值（在此为 70）可观测到不同的结果。

3.2.6. 其他传感器

能力风暴还能集成很多其他的传感器，插在 ASBUS（参见 3.5 节）上即可使用。下面作简单介绍。

人体热释片传感器

人体热释片传感器对移动的人体热源敏感，加上菲涅耳透镜后可以探测 10 米外的人体。能力风暴装上一个或几个人体热释片传感器后，你可以让他一看见你，就向你迎过来，让他跟着你走。

超声传感器

超声传感器是机器人测距的专业传感器，测量距离一般为 20cm-6m，测量精度为 1%，通过测量声波发射与收到回波之间的时间差来测量距离。运用能力风暴本身的传感器在房间里找到门不容易，但运用超声传感器对房间扫描一周后，就能较方便地找到房门。

连续测距红外传感器

SHARP 公司推出了创新的 GP2D02/ GP2D12 连续测距红外传感器，测量范围为 10cm-80cm，参加灭火比赛时，用它来找房间门非常棒。

数字指南针

自主机器人的导航至今仍是世界性难题，借助数字指南针，可以使能力风暴辨别方向。

温度传感器

想让机器人动态地告诉你气温吗？加一个温度传感器是个好方法。

视觉

让能力风暴处理视觉是有些力不从心了，但你可以用它来作移动的监视平台。你可以在能力风暴上装微型摄像头，把视频信号发射出来，用 计算机接收后进行图象处理。

火焰传感器

用来发现火焰，参加机器人灭火比赛最好使用火焰传感器。

还有许多传感器，可以让能力风暴拥有特殊本领。大部分传感器可以方便地运用 ASBUS 接在能力风暴上，并用 JC 来编驱动程序。个别传感器需要用汇编语言编写驱动程序。详细信息请看广茂达网站：http://www.grandar.com 。

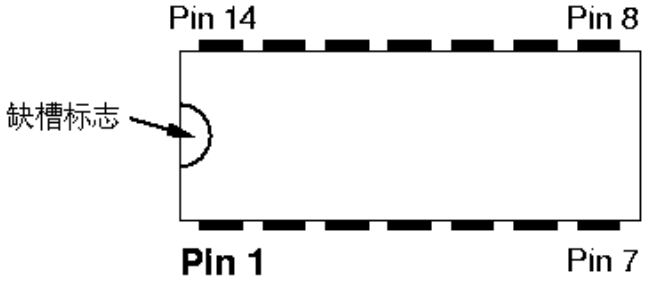
能力风暴的魅力在于你能控制所有的资源，直接领悟信息采集与处理的机制，以及如何处理现实情况的复杂性和难以预测性。

3.3. 能力风暴的计算机硬件

能力风暴计算机硬件的设计策略是尽量选择功能齐全、可靠、周边设备集成度高的微控制器，价格也需控制，能让中国的学生以可以承受的价格获得世界上先进的智能机器人平台。Motorola 生产的 68HC11E1，使我们以极少的周边芯片获得了齐全的功能：8 个模拟口，5 个输入捕捉，3 个 PWM 输出，16 位地址，8 位数据总线，串口，以及 4 个通用 I/O。

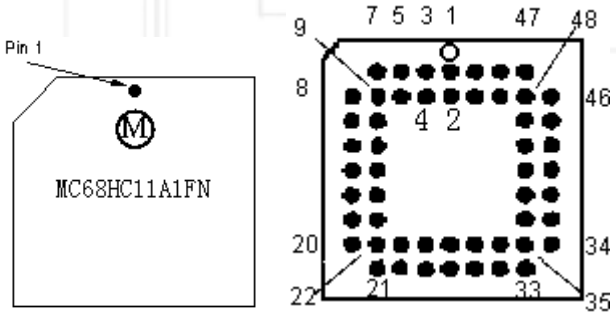
同时，考虑软件开发工具问题。没有优秀的软件开发工具，硬件只能成为专有系统，而无法成为开发平台。68HC11E1 的自下载功能，使我们拥有了纯软件开发调试的工具 JC。JC 是优秀的软件开发工具，可用于开发高层应用软件，又便于开发低层驱动，还能交互调试。

在观察能力风暴控制板时要注意识别芯片管脚，特别是 1 脚位置。常见芯片的封装形式有两种，双列直插（DIP）和塑料扁平封装（PLCC），参见下图。



正面视图

图 3.24 DIP 芯片管脚



正面视图

图 3.25 PLCC 芯片管脚和插座引脚

3.3.1. 微控制器

68HC11E1 由 CPU、片内存储器、定时器系统、串行口、A/D、并行 I/O 口、中断和复位系统组成，见图 3.26。

CPU

计算机的计算原理是什么？68HC11E1 的 CPU 有两个 8 位累加器，两个 16 位变址寄存器、一个 8 位条件寄存器，一个 16 位堆栈指针和程序计数器；M6800/M6801 指令系统，共 300 多条指令；16 位加、减、乘、除指令。学习使用单片机，可以真切感受计算原理，指令的构成，时钟周期，指令周期等等。

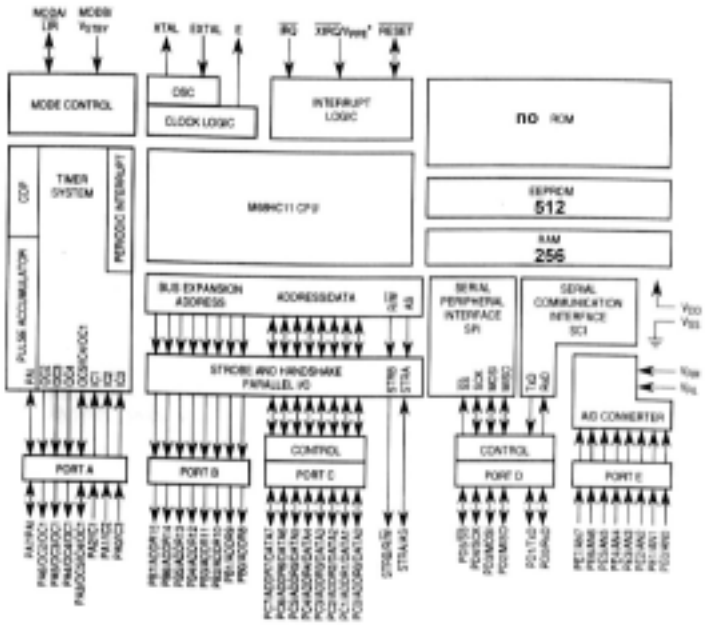


图 3.26 68HC11E1 原理结构图

片内存储器

68HC11E1 无内部 ROM，有 512 字节 EEPROM，可重定位的 256 字节 RAM。

定时器和脉冲电路

16 位高性能定时器系统，8M 晶振，定时器频率为 2MHZ（周期 0.5μs），3 个输入捕捉，可测量脉冲数量，脉冲周期、宽度和相位等；5 个输出比较，可输出 PWM 信号，可以完成各种定时控制功能，有定时器溢出中断功能。高性能定时器是 68HC11E1 的特色，能力风暴中用输入捕捉计码盘信号，用输出比较功能控制直流电机。

串行口

串行通讯接口 SCI，能力风暴用于和 PC 机通信。全双工同步串行外围接口 SPI（图 3.27），Motorola 单片机独有的串口标准，速度可达 2Mbps 以上，用于扩展外围芯片和多机通讯。能力风暴中已将其用于其他设备驱动中。

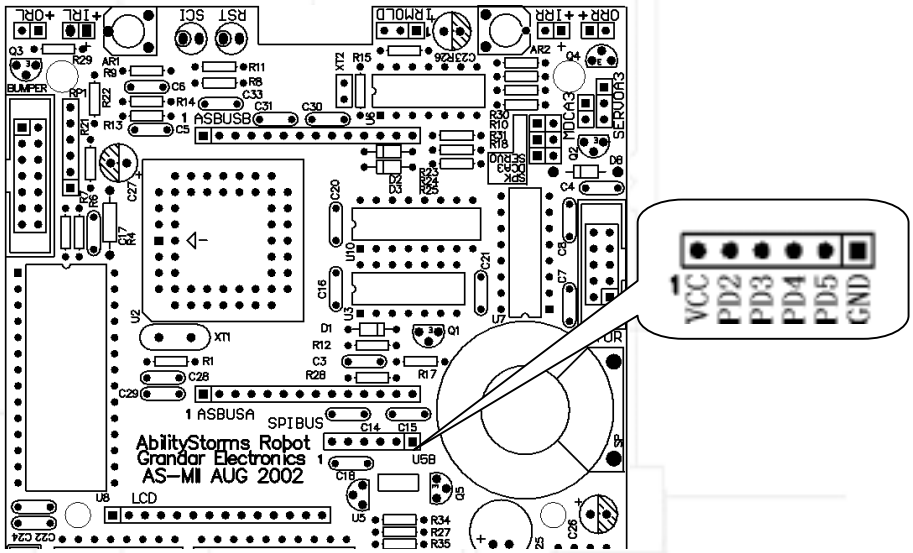


图 3.27　SPI 全双工同步串行接口引脚排列

A/D 转换器

8 个输入通道和四个转换结果寄存器，具有一次完成四路 A/D 转换或连续对同一路采样转换 4 次的功能。后一种功能可以方便实施去掉最大、最小、取均位的滤波方法。能力风暴中碰撞传感器、声音传感器均使用 A/D 转换器，非常方便，这也是 68HC11E1 的特色。

中断

有 16 个硬件中断和两个软件中断，它们各有独立的中断向量和中断允许位，响应中断时能自动保护所有的 CPU 寄存器。另外具有实时中断电路，可每隔指定的时间产生一次中断。而 8051 单片机则只有 5 个硬件中断。

并行 I/O 口

单片方法工作时，有 38 个 I/O；扩展方式时：有 8 位数据单线和 16 位地址总线，可扩展 64K 存储器。另有 2 个 8 位 I/O 口和一个 I/O 口。

复位系统和电源

计算机有多种复位方式：上电自动复位；外部 RESET 复位；看门狗复位（软件工作不正常时）；时钟监视复位。能力风暴采用前两种复位方式。

68HC11E1 的管脚

68HC11E1 的一大特色是功耗低，工作电流小于 15mA，有 WAIT 和 STOP 两种方式进行省电。68HC11E1 的管脚图见图 3.28。

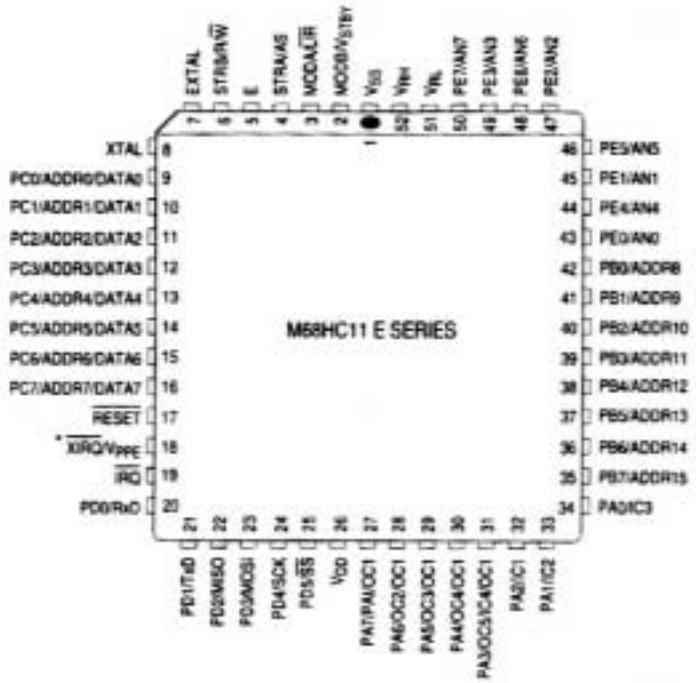


图 3.28　68HC11E1 的管脚

VDD，VSS----电源脚，VDD 为+，VSS 为 －

MODB/VSTBY，MODA/LIR----运行方式选择输入脚，备用电源供给脚。

EXTAL，EXTAL----晶振脚。能力风暴接 8M 晶振。

E----总线时钟输出脚。能力风暴中为 2MHZ。

RESET----复位脚，低电压复位。

XIRQ，IRQ --- XIRQ 不可屏蔽中断请求输入。

IRQ 外部中断输入脚

VREFL，VREFH ---- A/D 参考电压输入。能力风暴中 VREFL=0，VREFH=5V。

PE（PE0-PE7）----A/D 模拟输入或通用输入

PA（PA0-PA7）---- PA0-PA2，输入捕捉脚。

PA3-PA6，定时器输出比较脚，PA7，能作通用 I/0，或脉冲累加器输入或比较输出。

PD0-PD5----可作通用 I/0 脚，或者 PD0 为串口 RXD 输入，PD1，串口 TXD 输出，PD2-P5 分别为 MISO，MOSI，SCK，SS，构成 SP2 串口总线。

STRA，STRB，PB，PCD----单片时，可作通用 I/O 口，扩展方地址选择 AS 和读（R/W）控制线。

能力风暴运行于扩展工作方式，扩展了 32K 静态 RAM 和 8 个 I/O 口。注意：为了省引脚，单片机中一般以低 8 位地址和 8 位数据以时分多路方式合用 PC 的 8 只引脚。

能力风暴充分利用了 68HC11E1 的全部硬件资源。

3.3.2.　外部存储器

能力风暴智能机器人扩展了 32K 的静态不挥发 RAM。其优点是既有静态 RAM 的速度和

方便（70ns），又有 EEPROM 或 FlashRom 的掉电不丢失性，从而能将程序和数据合用一个芯片。AS62256 写入的数据可保存十年以上，同时具有可靠的上电、掉电、强静电等数据保护功能。

选址和并行口扩展

32KRAM 用了 AO-A14 共 15 根地址线，构成 32K 的地址空间，A15 为高电平时和 E，RESET 等信号复合片选 32K RAM，因此 32K RAM 的地址空间为 OX8000-OXFFFF。另一根地址线 A15 和其余地址线及读/写线复合扩展 4 个输入控制线，4 个输出控制线，见图 3.29。

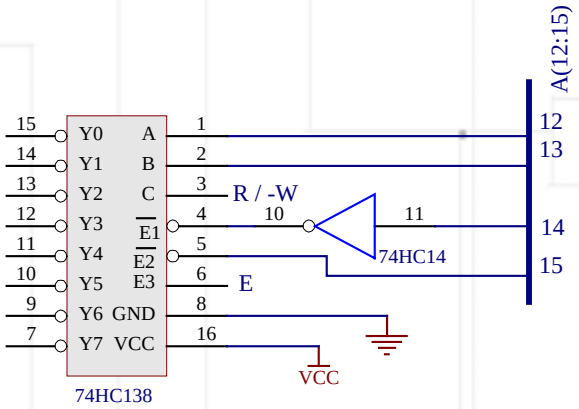


图 3.29　扩展地址选择线

地址空间与 8 个读写控制线选择对应表：

	写操作	读操作
OX4000—OX4FFF	OS0	IS0
OX5000—OX5FFF	OS1	IS1
OX6000—OX6FFF	OS2	IS2
OX7000—OX7FFF	OS3	IS3

对地址 OX4000 进行写操作，OS0 口线被选中，OS0 作为片选信号，即把数据总线上的数据写往 OSO 选中的并行接口芯片上。对地址 OX4000 进行读操作，IS0 口线被选中，IS0 作为片选信号，即把选中的芯片上的 8 位数据通过并行口读至 CPU。

32K 空间用来选择四对输入输出控制线，在能力风暴在 ASBUS 上能扩展 4 块平行输入卡，4 块平行输出卡。目前 PC 机上也只有 4-5 个 PCI 扩展槽。只要使用一对输入输出选择线就可以扩展 8 个输出口，8 个输入口。

能力风暴地址空间的分配如图 3.30。

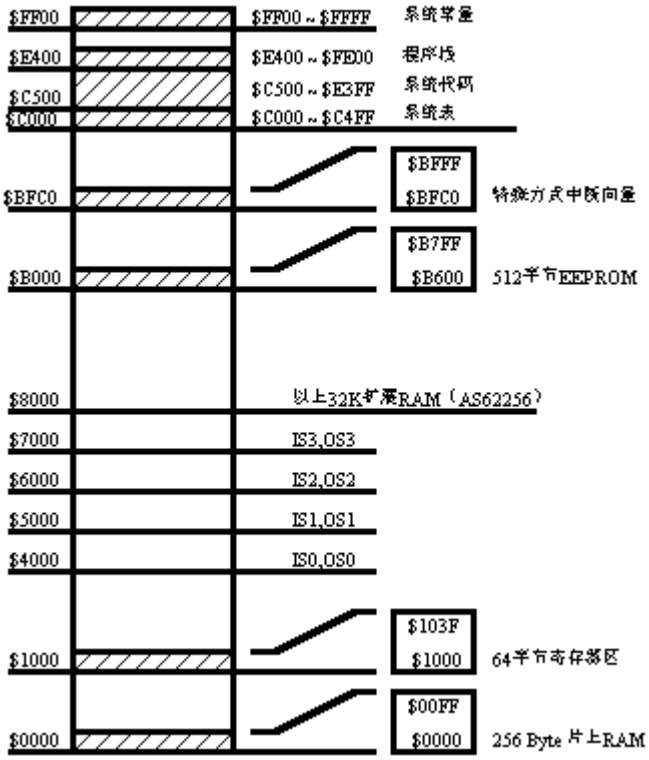
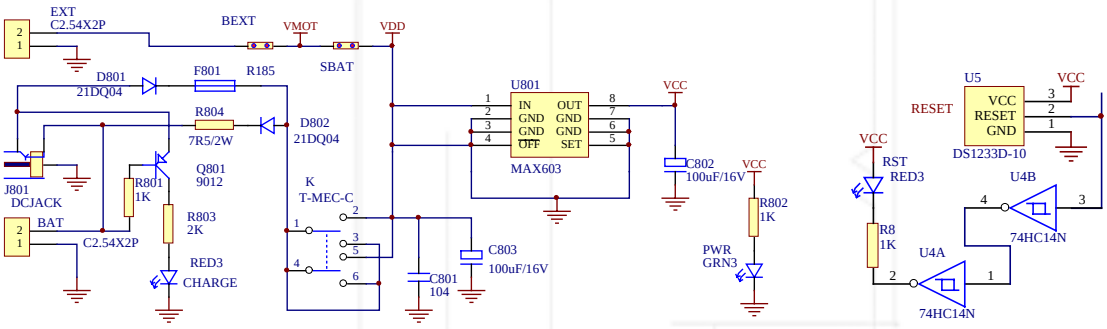


图 3.30 能力风暴智能机器人的地址空间，图中地址为 16 进制

能力风暴智能机器人的操作系统 ASOS 放在高端内存 10K 左右，往下到 8000 之上有近 20K 左右是用户程序空间。在下载用户程序的时候，JC 会显示编译后代码有多少字节，存放的位置。你可以计算一下 20K 能放多少行 JC 程序。

3.3.3. 电源与复位电路

能力风暴控制板采用 Maxim603 稳压芯片，提供 500MA，5V 电压，该芯片自身的功耗很低。低电压复位保护电路采用 DS1233D-10，当电压低于 4.5V，将产生复位信号，同时红色 RST 发光二极管变亮，见图 3.31。



未使用扩展电池电机的供电是由系统电源提供的。

使用扩展电池，系统电源只为主板电路提供电能。

图 3.31 电源与复位电路

主板上有两个电池连接插座，扩展电池接口只用来为电机（两只轮子驱动电机、MDCA3 直流电机、伺服电机）提供电源，使用时要将主板左下脚的短路跳线帽从 SBAT（无扩展电池）位置移到 BEXT（使用扩展电池）位置（如图 3.31，为了便于与主板图对照没有将示意图翻转。），这时所有电机的电源都来自扩展电池。**注意电池的电压不能超过电机的额定电压，例如大多数伺服电机的额定电压为 9V，并注意认真核对两个电池插口的正负符号（与背面大块铜皮相连的引脚为“－”负）。**

3.3.4. 通信

能力风暴采用 MAX202 串口驱动芯片， PC 发过来的信号经过 U4E 和 U4F 去驱动 SCI 发光二极管，因此 PC 传送数据给能力风暴时，黄色 SCI 发光二极管会闪动。见图 3.32。

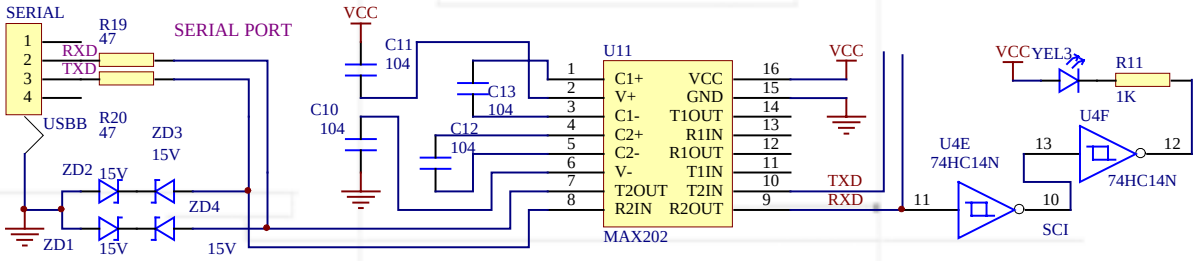


图 3.32 通信模块

3.4. 驱动器

机器人获取环境信息，计算处理后，会作用于环境。能力风暴作用于环境的驱动器有三个。

- 直流电机
- 喇叭
- LCD

两只直流电机构成能力风暴智能机器人的行走装置，采用差动驱动方式。

喇叭是能力风暴的嘴，碰到障碍可以发出警告声，可以唱歌，可以呼叫同伴。

LCD 用于显示能力风暴运行的信息，在状态检测和故障诊断时特别有用。

3.4.1. 电机驱动电路

电机驱动采用直流电机驱动芯片 U7（SN754410 ），其连接电路如图 3.33。

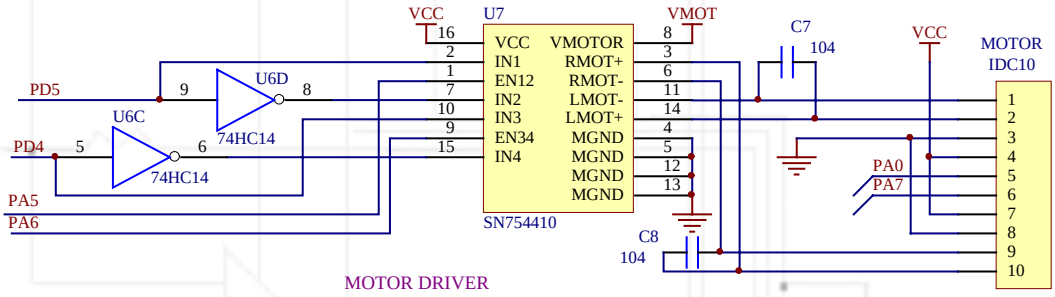


图 3.33 电机驱动电路

直流电机在一定电压下，转速与转矩成反比；如果改变电压，则转速转矩线随着电压的升降而升降（如图 3.34）。在能力风暴智能机器人负载一定时（即转矩一定时），降低电压，对应的转速也跟着降低， 这样就可实现用电机调速，参见图 3.34。

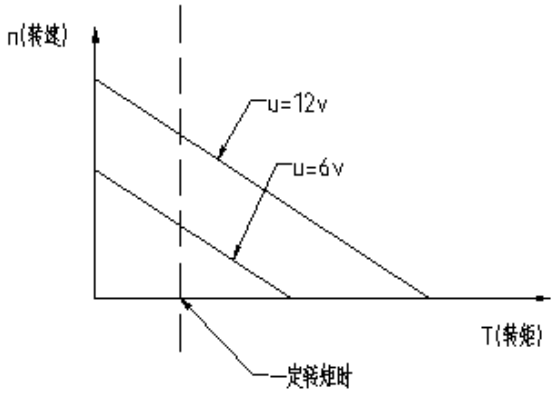


图 3.34 转矩、转速与电压关系图

在能力风暴智能机器人里采用的是改变电机电压的方式来改变电机的转速。能力风暴智能机器人提供给电机的信号是方波，不同的方波平均电压不同（如图 3.35），我们就利用这一点来进行能力风暴智能机器人的速度控制。采用不同的脉宽调节平均电压的高低，进而调节电机的转速，即脉宽调制（PWM，Pulse Width Modulation）。PA5，PA6 分别给 SN754410 发脉宽调制信号，通过改变脉冲宽度来调节输入到电机的平均电压。SN754410 每路电机控制输出为 1A，并有超载保护功能（如图 3.33）。

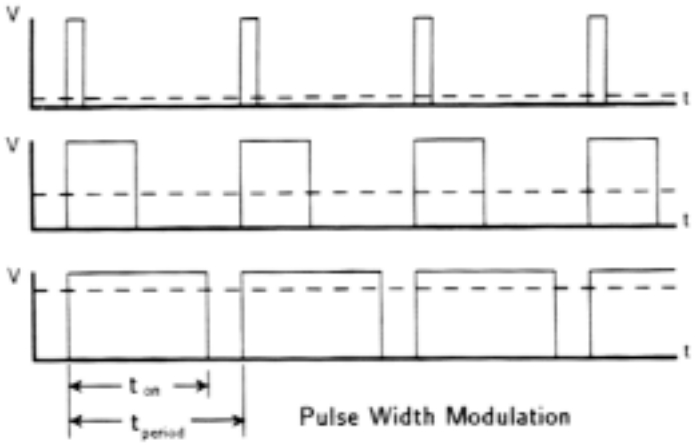


图 3.35 不同宽度的方波实现 PWM 控制

电机通过减速器将转动传给轮子，将高速转化为低速。能力风暴通过三级直齿轮传动减速，以满足运行的速度和转矩要求。

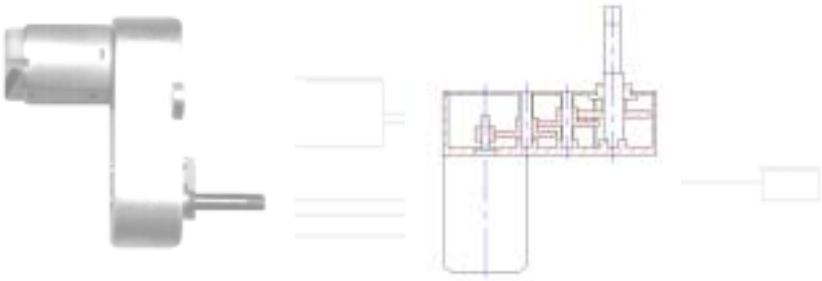
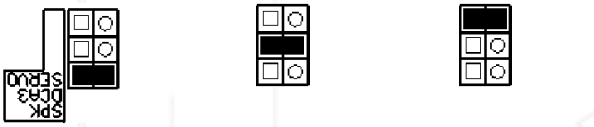


图 3.36 三级减速器（齿轮头）

3.4.2. 喇叭 （扩展电机）

喇叭由 PA3 控制。能力风暴在主板上扩展了一个直流电机接口和一个伺服电机接口，都由 PA3 控制。因此喇叭、直流电机和扩展电机三者不能同时使用，须用短路端子加以选择。如图 3.37 所示。



出厂状态，单片机 PA3 口驱动扬声器

单片机 PA3 口驱动直流电机（如灭火风扇）

单片机 PA3 口驱动伺服电机（如机器手爪）

图 3.37　PA3 跳线选择

3.5. 硬件扩展总线 ASBUS

能力风暴控制板设计了 ASBUS 总线（图 3.38），类似于 ISA 和 PCI 总线。采用堆叠式的 ASBUS 扩展卡可以方便扩展控制板的功能。

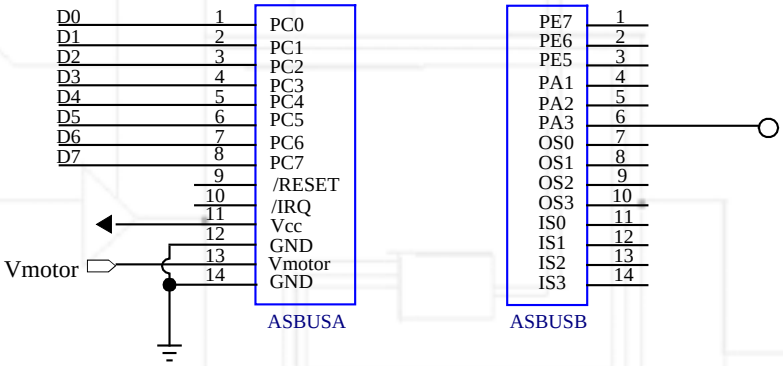


图 3.38 ASBUS 信号线

ASBUSA 和 ASBUSB 分别有 14 个信号线。

PC0-PC7：数据总线

RESET：复位信号

IRQ：外部中断输入脚

VCC：+5V 电源（负载不要超过 300MA）

Vmotor：电机电压，也即电池电压，可接较大负载

GND：地

IS0-IS3：输入选择线 0-3

OS0-0S3：输出选择线 0-3

PA1-PA2：输入捕捉口

PA3：输出比较口，已被喇叭、DC3，servo 使用，

PE5-PE7：模拟输入口

下面举例介绍具体的扩展方法。

3.5.1. 扩展 2 个光敏传感器

ASBUS 上还有 3 个模拟输入口 PE5-PE7，选择 PE5，PE6 作为 2 个光敏传感器输入口，见图 3.39。具体实施时，可采用 ASDIY 通用实验板，在上面布置电阻和插座，2 个光敏电阻焊在附带的圆形传感器板 PCB 上，焊上插头，装入传感器管，滴一点 502 胶水固定，然后装在能力风暴外壳空闲传感器孔内，即完成 2 个光敏传感器的扩展。

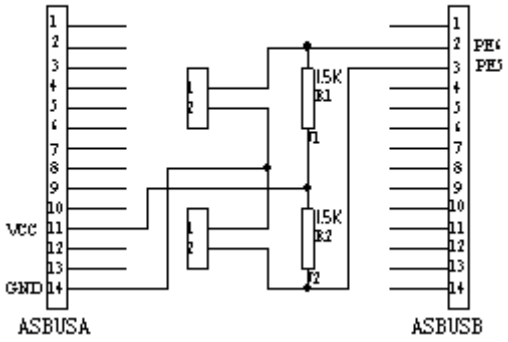


图 3.39 扩展两个光敏传感器

调用库函数 analogport (5)，analogport (6)，可以检测新扩展的光敏传感器的光强值。扩展温度传感器、力传感器等模拟量输入传感器均可采用这种方法。

3.5.2. 扩展红外接收传感器

利用 ASBUSB 上的模拟口或数字口可方便地扩展红外接收传感器。红外接收传感器的扩展电路见图 3.40，其中+5V 接 ASBUSA 的 Vcc，地接 Gnd，输入口可以接在 ASBUSB 的任一个模拟口或数字口。例如在 PA2 口扩展一个红外接收传感器,调用 digitalport (2) 获取数据。如果在 PE5 口扩展一个红外接收传感器,则调用 analogport (5) 获取数据。扩展的红外接收传感器可用于扩展其他方向的测障能力，或感知其他能力风暴伙伴发来的红外信号，可用在捉迷藏游戏中。

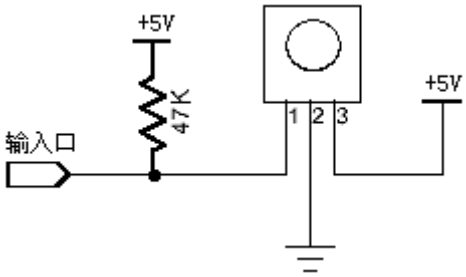


图 3.40 扩展一个红外接收传感器

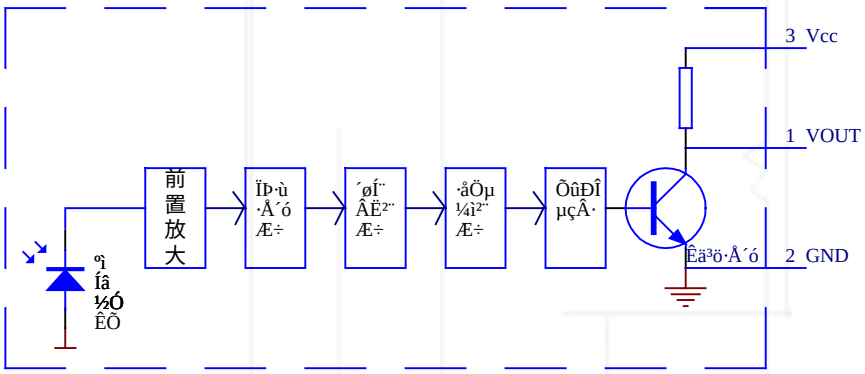


图 3.41　能力风暴红外接收传感器模块原理图

PA1，PA2 可用来扩展数字或脉冲输入的传感器，如光电编码器，超声传感器等。研究一下怎样扩展一个超声传感器，使能力风暴看得更远。

3.5.3.　扩展 8 个数字输出口

想再增加 2 个 LED 来装饰能力风暴，想增加一个步进电机，想增加 2 个伺服电机制作的手爪，需要扩展更多的输出口。

利用 ASBUS 的并行口扩展功能，采用地址锁存芯片 74HC373，用 OS0 线进行片选。数据总线送到 74HC373 后将被锁存，从而输出给外部设备，数字输出口的状态可由发光二极管的状态显示，outputo=1，则 LED0 亮。Outputo0-Outputo7 加驱动电路后可控制电机、继电器、灯泡等等。

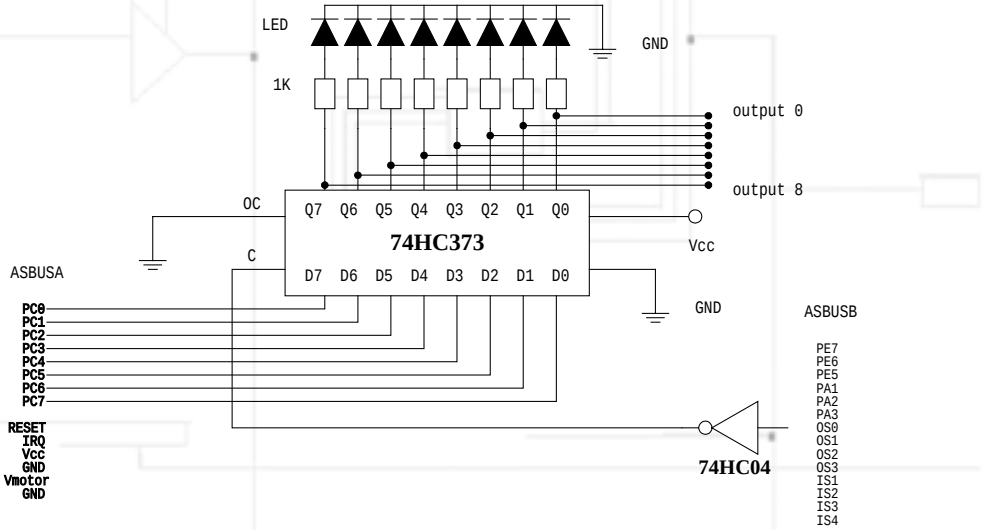


图 3.42 扩展 8 个数字输出口

想一下，怎样扩展 8 个数字输入口？怎样扩展每个伺服电机控制接口？怎样扩展一片键盘控制芯片？怎样扩展语音芯片？

ASBUS 的扩展潜力很大，如果你开发出自己的 ASBUS 卡，请告诉我们。

第4章 编程—赋予能力风暴智慧

4.1 第一个 VJC 程序，Hello robot！

双击桌面上的 VJC1.5 图标，可以打开一个对话框。VJC1.5 支持流程图程序和交互式 C 语言（JC）程序，可在对话框中加以选择，然后进入机器人编程界面。

下面我们就用 VJC1.5 的流程图（参见 1．6 节）为机器人编写第一个程序 ，在能力风暴的液晶显示屏 LCD 上显示“ Hello,robot！”

我们要编写的程序如图 4-1 左图所示：

图 4-1 Hello,robot！

a) 点击工具栏中的“ 新建 ” 按钮，新建一个程序；

b) 用鼠标将“ 执行器模块库 ”中的“ 显示 ”模块拖入到流程图生成区，放在主程序模块正下方，看看“ 主程序 ”模块与“ 显示 ”模块之间是不是有箭头连接，有则说明连接上了，否则再调整一下“ 显示 ”模块的位置。

c) 右键点击“ 显示 ”模块，就会出现如图 4-1 右图 所示的对话框，请将显示信息由 HI 改写为 Hello,robot!设定完毕，按“ 确定 ”；

d) 在“ 程序模块库 ”中选择“ 结束 ”模块，将它连在“ 显示 ”模块之后，这样整个程序就编写完成了。

e) 将机器人与计算机连接起来（用串口通信线，一端接计算机的九针串口，一端接机器人控制面板上的下载口）。

f) 将机器人的“ 开关 ”按钮打开，使机器人处于开机的状态。

g) 按 VJC 界面中的“ 下载 ” 按钮，待看到“ 下载成功 !” 字样时，取下串口通信线，按机器人 控制面板上的“ 运行 ”按钮，机器人的液晶显示屏上就会显示出 Hello,robot!

每个 VJC 的图形模块都代表一组 JC 代码，将流程图编辑界面切换为 JC 代码编辑界面就可看出来（参见 1.6 节）。也可在流程图编辑界面中，使用工具栏中的“ JC 代码 ” 快捷按钮，直接显示与流程图对应的 JC 代码。

下面我们用 JC 语言编同样的程序，并对 JC 编程的相关知识作简单介绍：

```
void main( )
{
  printf("Hello robot!\n");
 }
```

- main 是主函数，每一个 JC 程序总是从 main 函数开始执行的；main 函数的开始和结尾分别有个“ { ”和“ } ”；
- void 可以理解为“ 不带返回值 ”；所以第一句就可以理解为一个程序的“ 开头 ”；
- printf 函数的作用是：向输出设备（这里是指 LCD 显示屏）输出数据。那么 printf 一句的含义就是把 Hello,robot!输出到机器人的液晶显示屏上。
- 程序中每一句结尾都要加“ ；”这是 C 语句结束的标志。

如果你把上面这段程序下载到机器人中去，机器人就会显示 Hello,robot!

4.2 让机器人动起来

下面我们尝试着让机器人动起来。（ 相关的知识介绍请参看《VJC1.5 使用手册》 ）

在 VJC1.5 执行器模块库中有一专门控制机器人“ 直行 ”的模块，这里我们就用它来编写一个机器人直线行走的程序。要求：先让机器人以速度 100 前进 3 秒，再让机器人以速度 -60 后退 5 秒，再在原地以功率 80 旋转 1 秒。（如图 4-2 所示）

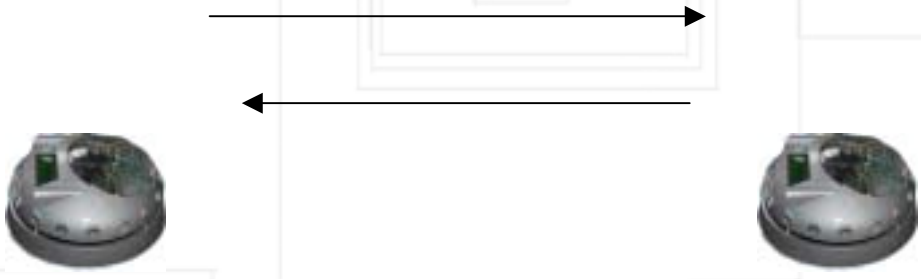


图 4-2 机器人移动

示范操作步骤如下：

4.2.1 编写流程图

a) 用鼠标将“ 执行器模块库 ”中的“ 直行 ”模块移到流程图生成区，并与“ 主程序 ”模块连接上；

b) 右击“ 直行 ”模块，在弹出的对话框中输入速度为 100、时间为 3 秒；

c) 再用鼠标将“ 执行器模块库 ”中的“ 直行 ”模块移到流程图生成区并连接在第一个

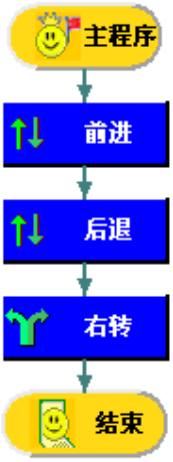
“直行”模块的下面；

d) 设置第二个“ 直行” 模块， 在对话框中输入移动速度为-60、时间为 5 秒；

e) 再将“执行器模块库”中的“转向”模块连接到程序中，在模块上点击右键，在弹出的对话框中设置速度和时间分别为：80 和 1

f) 再将“程序模块库”中的“任务结束”模块移入到流程图生成区，并连接在程序的末尾。

下面就是编好的流程图：




```
void main()
{
    drive( 100 ,0);
    wait( 3.000000 );
    stop();
    drive( -60 ,0);
    wait( 5.000000 );
    stop();
    drive( 0 , 80);
    wait( 1.000000 );
    stop();

}
```

图 4-3 机器人移动 VJC 程序

点击“工具栏”中的“编辑 JC 程序代码”按钮，切换到 JC 代码界面，可以对此界面中显示的 JC 代码进行修改，比如我们可以让机器人后退 7 秒，那么我们就可以把程序中对应的参数 5.000000 修改为 7.000000。

4.2.2 保存源代码程序

点击工具栏中的“保存”按钮，输入文件名，按“确定”，即可将编好的程序保存起来。

4.2.3 程序下载

下载程序时，能力风暴要处于开机状态，并且串口通信线已与计算机连接。 在菜单栏里“工具（T）”选项卡中选择“下载当前程序”，就会出现“智能下载程序”对话框，并显示下载进程， 待出现“下载成功！”字样， 程序就已经下载到能力风暴里了。

4.2.4 运行程序

将串口通信线取下，将机器人带到宽敞平坦的地方，按机器人身上的“运行”键，能力风暴智能机器人就会运行程序，快速前进 3 秒，再后退 7 秒，然后再旋转 1 秒。

4.2.5 走出规则轨迹

下面我们让能力风暴在地上走出一个规则的轨迹。

例 1：
```
void round()
    {
      drive(60,30);
      wait(10.000);
      stop();
    }
```
以上程序让能力风暴顺时针走直径约一米的圆形路径。

例 2：
```
void rectangle()
    {
      int i;
      for(i = 0; i < 4; i++)
         {
          drive(100,0);
          wait(1.0);
          drive(0,60);
          wait(0.2);
         }
      stop();
     }
```
以上程序让能力风暴逆时针走约一米见方的正方形路径。其中 wait(0.2)是能力风暴转 90 弯所需要的时间。 以上参数仅供参考，需要实际调整 。

4.3 让机器人感知环境信息

在 VJC1.5 的“ 传感器模块库 ”中，有让机器人感知环境信息的模块，这些模块的调用能够带给机器人感觉。比如说，“ 光敏检测 ”能够让机器人感觉到外界光线的强弱 ；“ 红外测障 ”能够让机器人检测前/左前/右前方的障碍等等。下面我们就试着让机器人感知外界环境。

例：让机器人对外界的光线进行检测，并在 LCD 上显示左右光敏所检测到的外界光线强度的平均值。

示范操作步骤如下：

4.3.1 编写流程图

a) 用鼠标将“ 控制模块库 ”中的“ 永远循环 ”模块移到流程图生成区并与“ 主程序 ”模块连接上；
b) 用鼠标将“ 传感器模块库 ”中的“ 光敏检测 ”模块移到“ 永远循环 ”模块内部并连接上；
c) 设置时，右击“ 光敏检测 ”模块，在弹出的对话框中选择“ 平均 ”，确定；
d) 再用鼠标将“ 执行器模块库 ”中的“ 显示 ”模块移到流程图生成区，并连接在“ 光敏检测 ”模块的后面；
e) 设置“ 显示 ”模块，点击右键在弹出的对话框中选择“ 引用变量 ”，就会出现一个“ 变量百宝箱 ”，点击“ 光敏 ”图标， 选择“ 光敏变量一 ”， 确定 。这样“ 显示 ”模块中的“ 显示信息 ”就为“ 光敏变量一 ”；
f) 再将“ 程序模块库 ”中的“ 任务结束 ”模块移入到流程图生成区，并连接在程序的末尾。
编好的程序如下图所示：

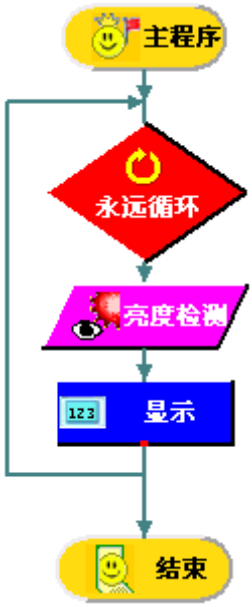


图 4-4 机器人感光 VJC 程序

我们也可以点击“ 工具栏 ”中的“ 编辑 JC 程序代码 ”按钮，切换到 JC 代码编辑界面，对 JC 代码进行修改。

4.3.2 保存程序

点击工具栏中的“ 保存 ”按钮，文件名输入：感光，按“ 确定 ”。此时，“ 感光.flw ”流程图程序文件就保存起来了。

4.3.3 程序下载

下载程序时，能力风暴要处于开机状态，并且串口连接线已与计算机连接。 在菜单栏里“ 工具（T）”选项卡中选择“ 下载当前程序 ”，就会出现“ 智能下载程序 ”对话框，并显示下载进程，当出现“ 下载成功！”字样， 程序就已经下载到能力风暴里了。

4.3.4 运行程序

将串口通信线取下，将机器人带到安全的地方，按机器人身上的“运行”键，在机器人的 LCD 上就会显示出外界光线强度的平均值。

4.4 JC 程序的基本程序结构

下面是 JC 程序中最常见的一种程序模式。

```
 while(1)      /*循环检测*/
{
  ir=ir_detector(); /*对环境采样*/
  if(ir>0)      /*条件判断*/
  {
    语句 A;       /*做相应的处理
   }
  else
  {
    语句 B;
  }
  wait(0.1);        /*采样周期*/
}
```

4.4.1 “台球”——碰撞传感器的使用

下面结合“台球”程序来学习碰撞传感器的使用。

```
void main ()
{
 int bill_trans=0; /* 预设的前进速度 */
 int bill_rot=0;   /* 预设的旋转速度 */
 int bmpr=0;
  while(1)          /*无限循环检测*/
  {
     bmpr=bumper();          /*检测碰撞传感器*/
      if(bmpr!=0)
      {
       if(bmpr==0b0011)      /*正前方发生碰撞*/
       {
        bill_trans=-80;      /*后退*/
        bill_rot=0;
       }
```

```
        else if(bmpr==0b1100)    /*正后方发生碰撞*/
        {
         bill_trans=80;          /*前进*/
         bill_rot=0;
        }
        else if(bmpr & 0b0101)   /*左侧发生碰撞，&是按位与运算符，
                                 故 bmpr & 0b0101 有零和非零两种情况*/
        {
         bill_trans=0;
         bill_rot=80;
         drive(bill_trans,bill_rot); /*顺时针转一个角度*/
         wait(0.5);
         bill_trans=80;              /*前进*/
         bill_rot=0;
        }
        else if(bmpr & 0b1010)       /*右侧发生碰撞*/
        {
         bill_trans=0;
         bill_rot=-80;
         drive(bill_trans,bill_rot); /*逆时针转一个角度*/
         wait(0.5);
         bill_trans=80;              /*前进*/
         bill_rot=0;
        }
      drive(bill_trans,bill_rot);    /*电机驱动库函数，参见 2.3.3*/
      }
   }
}
```

注：if()括号中的值有两种情况---零与非零，if()即据此进行判断。

在以上程序中有一个常用结构

while(1){ 检测传感器；作出响应；}

该结构实现了对传感器的循环检测，使能力风暴能够对周围环境的变化及时作出响应。能力风暴作出的响应可以是改变运动方式，也可以是改变一个内部变量。在这个例子里，能力风暴响应来自外部四个方向上的碰撞，并相应地改变运动方向。程序刚开始运行时，能力风暴静止。这时你只要给它一个初始碰撞，它就不停地运动起来，在障碍物之间撞来撞去。

如果你有不止一个能力风暴，试着给每个能力风暴下载这样的程序，把它们摆在一起，让其中一个能力风暴从远处撞过来，那场面一定象打台球，非常有趣。

4.4.2 跟人走——红外传感器的使用

能力风暴的红外传感器可以说是它最有用的武器。通过红外传感器，能力风暴可以探测到前方和左右两侧 10－80 厘米以内的障碍物。红外传感器就象一对“ 眼睛 ”。下面我们看一个有趣的程序，看看如何使用能力风暴的“ 眼睛 ”。

```
void main()
{ int ir = 0;                    /*用于储存红外检测结果的变量 */
  int bmp = 0;                   /* 用于储存碰撞检测结果的变量 */
  int old_bmp = 0;               /*用于储存 前一次的碰撞检测结果的变量 */
  int fol_trans_def =60;           /* 预设的前进速度 */
  int fol_rot_def =30;             /* 预设的旋转速度 */
  printf("Follow\n");              /*在 LCD 屏幕上显示 Follow*/
  while(1)
{ ir = ir_detector();            /* 取红外系统检测结果 */
    bmp = bumper() & 0b0011;  /* 检测左前、右前方向上的碰撞 ，
                                          &是按位与运算符*/
    if (old_bmp && bmp)     /*如果连续两次发生同样碰撞，
                                        &&是逻辑与运算符 */
      {
       stop();
       wait(0.5);             /* 等一会 */
       }
    else if (bmp)             /* 如果前方有碰撞 ，即 bmp 不为零。
                                 碰撞可能发生在左前、右前或正前方*/
      stop();                 /* 停止运动 */
    else if (ir == 0)             /* 前方没有物体 */
      stop();                     /* 停止运动 */
    else if (ir == 4)             /* 前方有物体 */
      drive(fol_trans_def,0);     /* 往前追 */
    else if (ir == 1)                        /* 物体在左侧 */
      drive(fol_trans_def, (- fol_rot_def));     /* 转向左 */
    else if (ir == 2)                         /* 物体在右侧 */
      drive(fol_trans_def, fol_rot_def); /* 转向右 */
    wait(0.1);                /* 让运动持续一会儿*/
    old_bmp = bmp;
   }
}
```

注：if()括号中的值有两种情况---零与非零，if()即据此进行判断。

这个例子实现了跟随前方物体。能力风暴可以跟随前方移动的人或物；如果撞上前方的物体，就停一停；如果在红外传感器探测范围内，前方没有物体，它就停下来。物体的面积须大一点，在 30 厘米×30 厘米以上。

4.5 JC 程序的高级编程

JC 支持多任务。多任务允许能力风暴同时做多件事情，如检测光源的同时避开障碍。在 JC 中实现多任务是很简单的，只要调用 start_process ()，括号中填写作为进程的函数名。任何一个函数都能作为一个进程来运行。在 JC 中进程的内存空间不是独立的，所有的进程共用一个内存空间，因此全局变量在任何进程里都可以访问。

4.5.1 第一个多进程程序

下面我们把 4.4.1 节的“台球”程序改成多进程的程序。

```
int bill_trans=0;  /* 预设的前进速度，全局变量 */
int bill_rot=0;     /* 预设的旋转速度 ，全局变量*/
void main()        /*主程序*/
{
 start_process(billiards_drive());    /*创建电机驱动进程*/
 start_process(billiards());          /*创建碰撞处理进程*/
}

void billiards_drive()           /*电机驱动函数*/
{
 while(1)
    drive(bill_trans,bill_rot); /*驱动电机*/
}

void billiards ()          /*碰撞处理函数*/
{
  int bmpr=0;
  while(1)                 /*无限循环检测*/
  {
     bmpr=bumper();        /*检测碰撞传感器*/
      if(bmpr!=0)
      {
       if(bmpr==0b0011)    /*正前方发生碰撞*/
       {
        bill_trans=-80;    /*后退*/
        bill_rot=0;
```

```
        }
        else if(bmpr==0b1100)    /*正后方发生碰撞*/
        {
         bill_trans=80;          /*前进*/
         bill_rot=0;
        }
        else if(bmpr & 0b0101)   /*左侧发生碰撞*/
        {
         bill_trans=0;
         bill_rot=80;
         wait(0.2);              /*顺时针转一个角度*/
         bill_trans=80;          /*前进*/
         bill_rot=0;
        }
        else if(bmpr & 0b1010)   /*右侧发生碰撞*/
        {
         bill_trans=0;
         bill_rot=-80;
         wait(0.2);              /*逆时针转一个角度*/
         bill_trans=80;          /*前进*/
         bill_rot=0;
        }
       }
      }
    }
```

“台球”改成多进程后，运行的效果没变，但结构已经完全不一样了。电机驱动进程 billiards_drive()专门设置电机速度，碰撞处理进程 billiards()判断碰撞并改变电机速度。两个进程之间通过全局变量 bill_trans 和 bill_rot 进行通讯。能力风暴的操作系统自动调度两个进程，给它们分配时间片 0.005 秒，两个进程交替运行。从执行效果来讲，就相当于这两个进程并列运行。这样的结构非常便于增加新的进程，同时便于处理更多的外部信息。

4.5.2 添加一个新进程

我们给前面的程序增加一个红外避障进程，改变它的行为，而其它部分不做大的变动。改动后的程序如下。

```
int bill_trans=0;          /* 预设的前进速度，全局变量 */
int bill_rot=0;            /* 预设的旋转速度 ，全局变量*/
int bmpr=0;                /*碰撞变量*/
int forward=0;      /*标志后退与否的变量*/
int running=0;     /*标志位移运动状态的变量 */
```

```
void main()
{
 start_process(billiards_drive());   /*创建电机驱动进程*/
 start_process(billiards_ir());      /*创建红外避障进程*/
 start_process(billiards());         /*创建碰撞处理进程*/
}


void billiards_drive()           /*电机驱动函数*/
{
 while(1)
  {  running = bill_trans;        /*running 标志位移运动状态*/
    drive(bill_trans,bill_rot);   /*驱动电机*/
  }
}

void billiards_ir()           /*红外避障函数*/
{
 int ir;
 while(1)
  {
  if(running)                  /*能力风暴一旦运动，就开始检测障碍*/
    { ir=ir_detector();        /*检测红外传感器*/
      if(bmpr==0 && forward)   /* 未发生碰撞且前进时，开始红外避障*/
        {
             if(ir==2)                /*右侧有障碍，向左绕*/
            {  bill_trans=20;
               bill_rot=-80;}         /*逆时针转*/

            else if(ir==1)           /*左侧有障碍，向右绕*/
            {  bill_trans=20;
              bill_rot=80;}          /*顺时针转*/

            else if(ir==0)           /*前方没有障碍，恢复直行*/
            {  bill_trans=80;
              bill_rot=0;}

        }
    wait(0.1);
    }
  }
}
```

```
void billiards ()          /*碰撞处理函数*/
{
  while(1)    /*无限循环检测*/
  {
      bmpr=bumper();        /*检测碰撞传感器*/
      if(bmpr!=0)
      {if(bmpr==0b0011)     /*正前方发生碰撞*/
       {forward=0;
        bill_trans=-80;     /*后退*/
        bill_rot=0;}

       else if(bmpr==0b1100)    /*正后方发生碰撞*/
       {forward=1;
        bill_trans=80;          /*前进*/
        bill_rot=0;}

       else if(bmpr & 0b0101)   /*左侧发生碰撞*/
       {bill_trans=0;
        bill_rot=80;
        wait(0.2);              /*顺时针转一个角度*/
        forward=1;
        bill_trans=80;          /*前进*/
        bill_rot=0;}

       else if(bmpr & 0b1010)   /*右侧发生碰撞*/
       {bill_trans=0;
        bill_rot=-80;
        wait(0.2);              /*逆时针转一个角度*/
        forward=1;
        bill_trans=80;          /*前进*/
        bill_rot=0;}
      }
  }
}
```

改动以后的程序增加了红外避障行为。下载运行这个程序，与前面的“台球”程序比较一下。

进程间的通讯和同步是多进程编程的难点，不解决好这个问题，多进程程序可能会产生一些奇怪的行为。在这个程序里，由于碰撞处理进程和红外避障进程都要设置电机速度，都要使用两个全局变量（bill_trans,bill_rot），这是进程间发生冲突的根源。如果不加以限制，两个进程同时修改电机速度，必然会出现一片混乱。

本程序中把 **bmpr** 改为全局变量，通过 **bmpr** 来划分两个进程生效的时间。即发生碰撞时，只有碰撞处理进程可以修改电机速度；在其他时间里，只有红外避障进程才有可能修改电机速度，碰撞处理进程只是在不断检测碰撞传感器。我们可以看到，碰撞处理进程的优先级高于红外避障进程。增加的另两个全局变量 running 和 forward 是出于红外避障的需要。**running** 是能力风暴位移运动的标志，红外避障进程要等待位移发生后才能起作用。**forward** 反映能力风暴运动方向， 红外避障进程在后退（forward=0）时不须处理前方障碍。

第5章 机器人项目

5.1 进门比赛

尝试编一个程序，控制机器人从墙外走进门内，看谁的机器人先进门，见图 5-1。

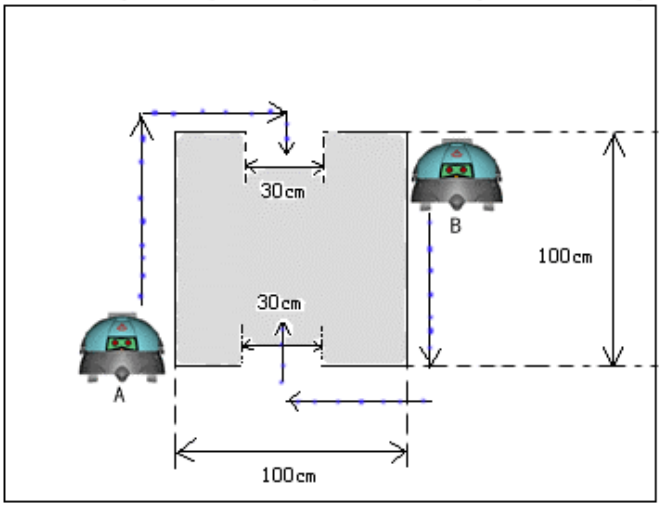


图 5-1 机器人进门

大家见过自动化工厂中的自动导引小车（一种运货机器人）吗？工程师为自动导引小车编程，使自动导引小车能自动从一个工位运货至另一个工位。机器人进门这个活动，就是模拟了这个情景。

5.1.1 活动准备

活动准备有两个目的，一是明确活动内容和要求，二是准备好活动器材和场地。

活动内容是运用 VJC 语言给能力风暴设计程序，控制机器人从墙外走进墙内，比赛谁的机器人先进门。

需要准备的活动器材如下：

能力风暴智能机器人一台；

装有 VJC1.5 的计算机一台；

面积为 2 平方米的比赛场地；

粉笔及抹布。

5.1.2 方案设计

首先画出机器人进门的路径图，并标出运动的步骤，见图 5-2。

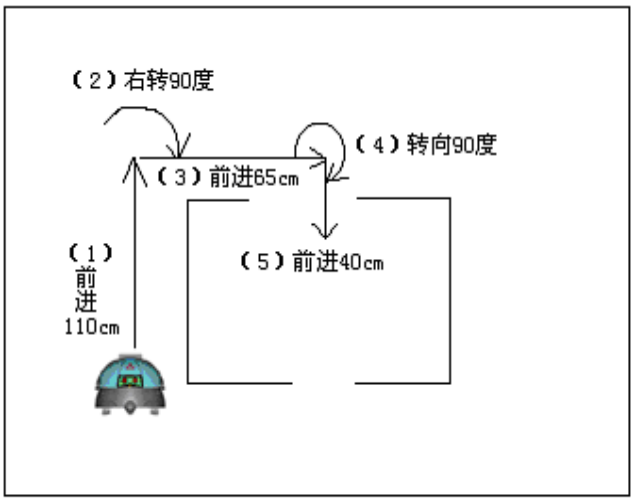


图 5-2　进门的运动步骤

图 5-3 所示流程图控制机器人的运动过程，调整“ 直行 ”模块和“ 转向 ”模块对话框中的速度和时间，测试机器人移动的距离，多试几次，就能走准你所要的距离。记录实验参数如下：

前进:　左右轮速度_____，前进 110cm，耗时_____秒；
　　左右轮速度_____，前进 65cm，　耗时_____秒；
　　左右轮速度_____，前进 40cm，　耗时_____秒；
右转：　速度_____，右转 90 度，耗时_____秒。
方案设计清楚后，就可以方便地设计出程序。

5.1.3　程序设计

按图 5-2 所示的路径图，设计出程序。图 5-3 是参考程序。

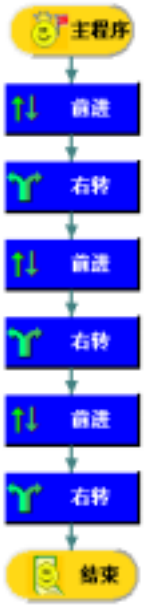


图 5-3 机器人进门参考程序

图 5－3 的参考步骤:

a) 双击桌面上的 VJC1.5 图标；

b) 单击执行器模块库中“ 直行 ”模块，将其移至流程图生成区中，连接在“ 主程序 ”模块下方；

c) 右击“ 直行 ”模块，输入机器人前进 110cm 的速度与时间。

d) 单击模块库中“ 转向 ”模块，将其移至流程图生成区中，连接在“ 直行 ”模块下方；

e) 右击转向模块，输入机器人转向速度与时间,使机器人原地转 90 度；

f) 类似可添加其余模块；

g) 打开程序模块库，将“ 结束 ”模块连接在流程图最下方。

能力风暴智能机器人中的单片机只懂得特定的机器语言,故用流程图编写的程序须先翻译为机器语言（图 5-6），然后才能下载运行。

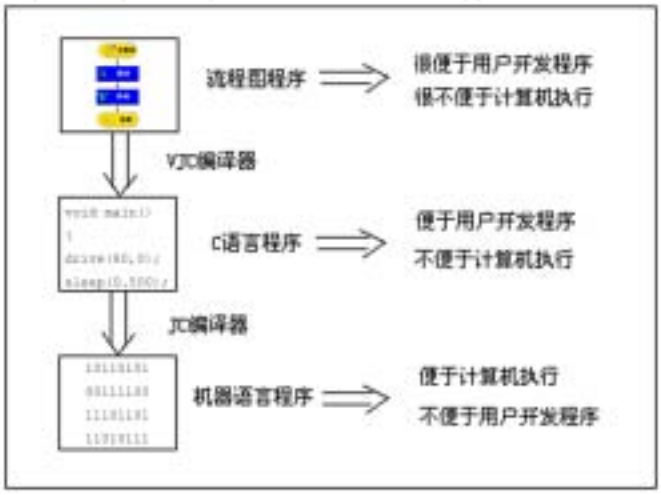


图 5-6　程序翻译过程

5.1.4　运行调试

程序编写好之后，按以下步骤，下载到机器人中去运行：

a) 用串口通信线将机器人与计算机连接起来；

b) 打开机器人的电源开关；

c) 选择 VJC 菜单栏中“ 工具（T）--下载当前程序 ”命令，即可下载程序 ；

d) 拔掉串口通信线，按一下机器人的运行键，能力风暴就会自动运行。

仔细观察运行结果，分析程序的什么地方需要修改，修改后再运行，直至满意为止。

5.2 机器人的行为控制

行为控制方法目前是机器人学的前沿研究领域之一。借助于多进程功能，能力风暴机器人也可以实施行为控制方法。

行为控制的基本思想是建构行为能力，而非功能模块。首先把要实现的机器人的行为特点描述下来，表达成基本的行为构成。如要做一只机器飞蛾，一边到处找食物，一边向较强光飞去，发生碰撞则向相反方向逃离，发现障碍后要能自动避障。用行为控制来实现如图 5-7。

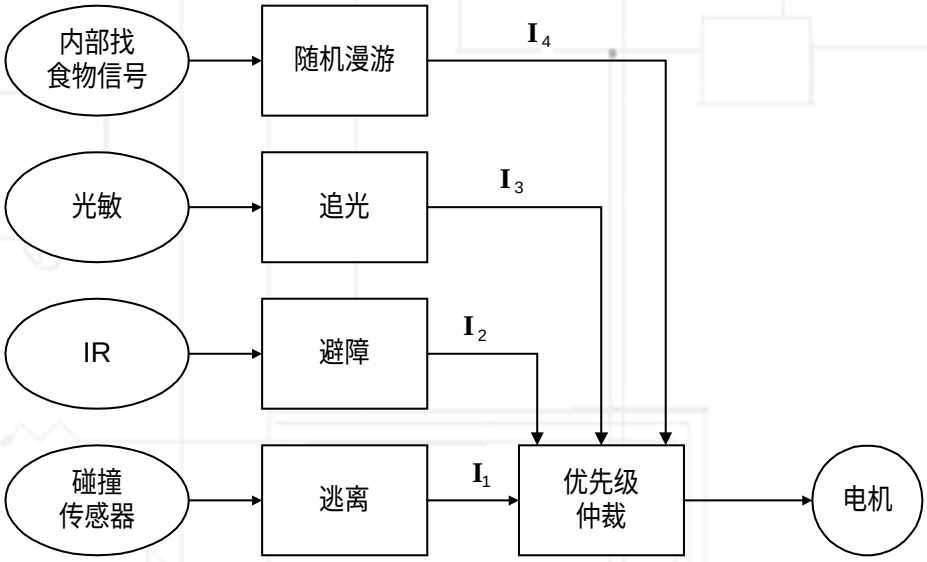


图 5.7 行为控制

四个基本行为结构很清晰，都是信号刺激→行为计算→执行。

由于执行机构（行走机能）只有一个，因此只有一个基本行为的输出能执行，这就需要优先级仲裁机制。$I_1 - I_4$ 是优先极，I_1 最高。谁的优先级高，先运行谁的执行信号。以下的例子可用能力风暴机器人来完成。

JC 的程序结构示意如下：

```
int  escape_output1, escape_output2;/* 逃离行为变量，输出给电机 */
int escape_output_flag;             /*逃离标志变量，供仲裁  */
void   escape ()                    /*逃离函数*/
{  ..... }
```

```
int    avoid_output1, avoid_output2;/*避障行为变量，输出给电机*/
int    avoid_output_flag;            /*避障标志变量，供仲裁*/
void   avoid ()                      /*避障函数*/
{ ..... }

int seeklight_output1, seeklight_output2；
                                  /*追光行为变量，输出给电机*/
int  seeklight_output_flag；    /*追光标志变量，供仲裁*/
void  seeklight ()              /*追光函数*/
{ ..... }

int seekfood_output1, seekfood_output2；
                                /*寻找食物行为变量，输出给电机*/
int seekfood_output_flag;   /*寻找食物标志变量，供仲裁*/
void   seekfood ()           /*寻找食物函数*/
{ ..... }
```

以上四个基本行为函数，完成信息采集、处理、输出反应的工作。

优先级仲裁如下：

```
void  arbitrate () /*仲裁函数*/
{
While (1)
{
 if(escape_flag==1)
    {a=escape_output1;
     b=escape_output2;}
else if(avoid_flag==1)
    {a=avoid_output1;
     b=avoid_output2;}
else if (seeklight_output_flag==1)
    { a=seeklight_output1;
      b=seeklight_output2;}
else if (seekfood_output_flag==1)
   { a=seekfood_output1；
     b=seekfood_output2;}
 wait(5.000);     /* 让优先行为持续 5 秒钟 */
 }
 }
```

优先级高的行为如果激活，则它的输出将覆盖低优先级低的行为输出。可以看出，以上程序中优先次序为：逃离，避障，追光，寻找食物。

主函数如下：

```
void main( )
 {
 start_process(motor_driver());
 start_process(escape());
 start_process(avoid());
 start_process(seeklight());
 start_process(seekfood());
 start_process(arbitrate());
}

int  a,b;
void  motor_driver( )
{
 drive(a,b);
}
```

在 JC 中主函数就特别简洁，以上 6 个任务通过 CPU 的调度在进行，外部看就象 6 个任务在并行运行。

请试着写全这个机器飞蛾的程序，运行后，看一看，是不是有全新的感觉，机器好象有生命了。对，这就是人工生命！

5.3 群鸭过河

[任　　务]

以小组的形式展开比赛，机器人作小鸭，它们能够一个跟在一个后面，一起安全“游”过“河”。安全过“河”的机器人个数最多的队获胜。如图 5-11 所示

图 5-11 小鸭过河

[任务分析]

鸭妈妈带着一队小鸭过河，那么就需要为小鸭妈妈编写“过河”程序，其他小鸭能够一个跟着一个，那么就要为它们编写“跟随”程序。

5.3.1 活动准备

a) 物品准备

能力风暴智能机器人	5 台
计算机	1 台

b) 让机器人移动要用到库函数 drive(a,b)，在前面已经作过相应的介绍，在这里就不重复说明了。

c) 判断与循环语句的使用。

```
while(1)
{
 …
  }
```

这个语句是一个无限循环语句。while 语句不断对其后（）中的内容进行判定。当判断的值为 1 时就将 {} 内的语句执行一遍，然后再接着对（）中的值进行判断。而在这里 while 后面（）里的内容为 1，所以这个语句会无限循环下去。

```
if ()
…
else 或者 else if ()
…
```

也是比较常见的判断语句，这种语句能够将问题的多种情形都罗列出来，分别进行处理。

d) 红外传感器可以检测到正前方、左前方、右前方是否有障碍物，调用一次库函数 ir_detector()，红外传感器就进行一次检测，并有返回值。返回值具体意义为 0=>没有障碍，1=>左边有障碍，2=>右边有障碍，4=>前方有障碍；

机器人的碰撞传感器，能够检测到前、后、左、右四个方向的碰撞，调用一次 bumper() 函数，就进行一次碰撞检测，有返回值。其具体意义为： 1 左前，2 右前，4 左后，8 右后，3 前，12 后，5 左，10 右。

5.3.2 方案设计

方案的设计分为“过河”和“跟随”两个部分。

a)“过河”

过河的方案有很多，比如：直着过河、斜着过河、波浪式过河等，不同的过河方法，程序编写的难度有所不同，随后的小鸭“跟随”的难度也有所不同，波浪式过河，后面跟随的小鸭就比较容易跟丢，带小鸭成功过河的可能性就比较小。

b)“跟随”

1) 要让小鸭能够跟随，这就需要用到红外传感器，当小鸭“看到”前方有物体时，就会主动跟上前去。这时候，如果小鸭的前进速度比较慢，它很可能就跟不上了，速度快又会撞到前面的小鸭，因此速度的设置比较重要。

2) 如果小鸭撞到了前面的鸭子，该怎么办？可以让它停下来，再去检测前面有没有可以跟随的鸭子。

3) 如果小鸭迷失了方向该怎么办呢？也让它停下来，在原地等着鸭妈妈。

具体的方法由大家自己来设计，能够完成整个活动的方案就是好方案！

5.3.3 画流程图

试着根据思路画流程图，画这个流程图时遇到的判断比较多，难度增加了。要仔细地思考。条理清淅会有利于编程。

根据流程图或 JC 代码所学知识编写程序，再将编好的程序下载、运行。先从鸭妈妈带一个小鸭过河开始，逐渐增加小鸭的个数，看哪个队的鸭妈妈一次能带过河的小鸭最多！

若未达到活动效果，则说明程序还需要调试，该改动哪些地方呢？请填写表 5-1：

序号	出现的现象	解决方法
1		
2		
3		
4		
5		

表5－1 现象及调试

第6章 附录

附录 1：JC1.5 库函数

1．执行器输出

void stop()	关闭左右两个电机，停止运动；
void stop_motor(int m)	关闭电机m，0为所有电机，1为左电机，2为右电机，3为扩展电机；
void motor(int m, int speed)	以功率级别speed(-100～100)启动电机m(1为左电机，2为右电机，3为扩展电机)；
void drive(int move, int turn)	同时设定两个电机的速度，move 为平移速度，turn 为旋转速度。

2．传感器输入

int photo()	光敏传感器检测。photo(1)为检测左光敏，photo(2)为检测右光敏，返回值为0～255的数字量；
int microphone()	声音传感器检测。返回值为0～255的数字量；
int ir_detector()	红外传感器检测。返回值的意义：0=> 没有障碍，1=>左边有障碍，2=>右边有障碍，4=>前方有障碍；
int bumper()	碰撞传感器检测。返回值的意义：1 左前，2 右前，4 左后，8 右后，3 前，12 后，5 左，10 右。
int rotation(int index)	光电编码器脉冲累计读数，rotation(1)为检测左光电编码器，rotation(2)为检测右光电编码器。
int digitalport(int channel)	读数字口上传感器的值。channel的范围是0～7。返回值：从传感器数字硬件读到的值为零伏或逻辑零时，返回1；否则返回0
int analogport(int channel)	读模拟口上传感器的值。channel的范围是0～7。返回值是0到255间的整数值
int encoder(int index)	读取光电编码器的当前状态。0为低电平/1为高电平（分别对应码盘的黑条和白条）。

3．时间

void wait(float sec)	延时sec 秒后再执行后面的语句。sec是一个浮点数
void resettime()	将系统时间复位清零
float seconds()	以秒的形式返回系统时间，它是一个浮点数，精度为0.001秒

4．声音

void beep()	产生一段0.3秒500赫兹的音频信号。发声结束后返回。
void tone(float frequency, float length)	产生一个length秒长、音调为frequency赫兹的音频信号。

5．电池

int battery()	检测电池电量，返回值为0～255的数字量；

6. 其他函数

void asosreset()	软件复位函数。与按下能力风暴的复位开关的效果一样，ASOS操作系统重启动，程序停止运行。
int runbutton()	读运行键状态。运行键在能力风暴智能机器人的头顶上，可以在程序中使用该按钮。返回值：0没按，1按下
int abs(int val)	取整数val的绝对值
int max(int x, int y)	求两个整数的最大值
int min(int a, int b)	求两个整数的最小值
float rand()	返回0～1之间的随机浮点数
int random(int scale)	返回0～scale之间的随机整数

附录 2：产品的主要技术性能和参数

参数名称	性能指标
外形尺寸 mm	直径φ220 高 H150
质量(重量) Kg	≈1.1
运行	可实现 R=0 转弯，速度可调，Vmax≥60cm/s
电池	额定电压 7.2V，锂电池。 重复充电次数：1000 次。
连续工作时间	≥2h
串口速率	9600bps
人机交互	可通过程序的下载，执行界定的功能并将信息反馈，实现人机交互
扩 展	可进行机械、电子、软件的扩展、ASBUS＋扩展
负载能力	
最大爬坡角度	

附录 3：产品的使用条件和使用环境要求

1 大气条件

a) 环境温度：0℃ ～ 40℃；

b) 相对温度：30% ～ 90%；

c) 大气压力：86 KPa ～ 106 KPa.

2 电源条件

充电器电源：交流 220V,50Hz 。

3 操作系统条件

Windows 95/98/ME/NT4/2000/XP，有光驱、至少有一个九针串口的 PC 机。

产品的安全警示说明和使用注意事项

1 在使用充电器充电时，应该由成年人将充电器插到 220V 标准电源上。

附录 4：常见故障及解决办法

常见故障及解决办法详见后面的列表。

	现　　象	原　因	解决方法	说　明
电源故障	机器人稍微受到震动或冲击会掉电或者复位	电池没有完全卡到电池槽内，或者电池卡损坏，电池无法卡紧	将电池卡到底（有些电池可能有点紧，但卡入后不会损坏机器）	
	接充电器时主板正常工作，不接充电器时主板不上电	充电插座坏，导致电池对主板供不上电	更换充电插座	
下载程序错误	下载时没有进度显示	操作系统崩溃	重新下载操作系统	下载开始，PC 机送能力风暴一个串行信号，能力风暴收到后会返回一个信号，接收到这个返回信号后，开始显示下载进程
		机器人死机	重新复位或者重新开机	
	下载到一半后提示中断错误	下载过程中按下复位键	重新下载一次	
LCD 显示屏故障	LCD 黑屏	操作系统崩溃	重新下载操作系统	
		死机	重新开机	
	太极图不闪烁	死机或操作系统崩溃	重新下载操作系统	

光敏传感器故障	光敏读数始终为 255	光敏探头的导线脱落	重新焊接导线	
红外传感器故障	探测不到障碍物	红外发射或红外接收的导线脱落	重新焊接脱落导线	
		调节电位器损坏	更换调节电位器	更换时注意不要损坏焊盘，建议焊接前最好用平口剪将电位器的三个引脚剪掉
碰撞传感器故障	在没有碰撞的时候认为有碰撞	碰撞开关摆动臂过于外翘，导致碰撞环在正常位置时，碰撞开关仍然处于闭合状态	用尖嘴钳将碰撞开关摆动臂向里面弯曲，使碰撞开关处于开启状态	
	有碰撞的时候认为没有碰撞	碰撞开关摆动臂过于内凹，导致碰撞环在压迫碰撞开关时，开关仍然处于开启状态	用尖嘴钳将碰撞开关摆动臂向外弯曲，使碰撞环压迫碰撞开关时能够闭合	

运动系统故障	**驱动轮不转**	**AS611 芯片烧毁**	**更换 AS611**	区别的标志是在软件里启动电机，用手触摸齿轮箱上的电机轴是否转动，如果电机不转动证明是单片机坏或者 AS611 坏，否则认为是齿轮箱坏
		齿轮箱齿轮滑牙	**更换齿轮箱**	
	驱动轮一直转	**AS611 芯片烧毁**	**更换 AS611**	
	机器人不走直线	**齿轮箱摩擦大**	**更换齿轮箱**	通常 1 米内偏差 20 厘米为正常
		光电编码器损坏	**更换新的编码器**	
编码器故障	**光电编码器没有读数或读数不变化**	**光电便码器插头插反**	**将插头有金属片的一面朝线路板**	当光电编码器导线插反时，其很容易被烧掉
		光电编码器损坏	更换新的编码器	

附录 5：AS-MII 的主板布局图