

**课 程 设 计**

（2019 / 2020 学年 第 二 学期）

|  |  |  |  |
| --- | --- | --- | --- |
| 课程名称 | 嵌入式课程设计 | | |
| 实验名称 | 基于单片机的直流电动机控制系统设计 | | |
| 实验时间 | 2020年6月1日 | 至 | 2020年6月12日 |
| 指导单位 | 自动化学院、人工智能学院 | | |
| 指导教师 | 范保杰 | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 学生姓名 | 陈力 | 班级学号 | B17050322 |
| 学院(系) | 自动化学院、  人工智能学院 | 专 业 | 自动化 |

摘要

本次课程设计主要利用MCS-51系列单片机控制PWM信号从而实现了对直流电机转速进行控制。课程仿真设计总共包含5部分，单片机控制模块、直流电机与L298电机驱动模块、速度检测模块、独立键盘输入模块、LCD显示模块。本文介绍了PWM（脉冲宽度调制）信号的原理以及如何使用AT89C51单片机输出PWM信号，如何通过软件对PWM信号的占空比进行调节，从而控制电机转速。

**关键字: MCS-51 、 AT89C51 、 PWM信号、 电机调速**

目录

[1 设计目的及任务要求 1](#_Toc43542141)

[2 设计内容 1](#_Toc43542142)

[3 设计与实现 1](#_Toc43542143)

[3.1 系统模块 1](#_Toc43542144)

[3.2 系统方案 3](#_Toc43542145)

[3.2.1 速度加减的实现: 3](#_Toc43542146)

[3.2.2 正反转的控制: 3](#_Toc43542147)

[3.2.3 电机速度的显示 : 3](#_Toc43542148)

[3.3 原理分析 3](#_Toc43542149)

[3.3.1 PWM调速原理 3](#_Toc43542150)

[3.3.2 PI算法 5](#_Toc43542151)

[3.3.3 程序流程 5](#_Toc43542152)

[4 测试和结果 6](#_Toc43542153)

[4.1 设计仿真原理图 6](#_Toc43542154)

[4.2 正反转实现 7](#_Toc43542155)

[4.3 加减速实现 8](#_Toc43542156)

[4 遇到的问题 9](#_Toc43542157)

[5 实验心得 9](#_Toc43542158)

[6 参考文献 10](#_Toc43542159)

[附录代码 10](#_Toc43542160)

1 设计目的及任务要求

利用ARM硬件平台、速度检测电路、电机驱动模块和直流电动机来实现直流电动机控制系流。该系流需实现下列要求:

1．使用定时器产生可控PWM波,通过按键改变PWM波的占空比，从而控制直流电动机的转速。

2.通过按键控制电动机的启动和停止、加速、减速以及设定转动方向和转速。

3.实时测量电机的实际转速并在LED显示出来

4.对电动机实现PI调速，使其转速趋近于设定值。

2 设计内容

完成单片机的直流电动机控制系流总体设计包括直流电动机的驱动方式、控制方式、单片机电路、案件电路、电机驱动电路、显示电路等硬件设计及相应的驱动设计。设计出的控制系统应能使直流电机正反转加减速调节，并可以从案件键盘输入进行相应工作模式的切换和加、减速的调节，用显示屏显示当前的状态，如速度和正反转等信息。

直流电动机PWM控制系统的主要内容包括：控制直流电动机的加、减速以及正、反转和急停.并且可以调整电机的转速。能够方便地实现对电动机的智能控制。

主体电路：这部分电路要由AT89C51单片机的I/O端口、定时计数器、外部中断扩展等控制直流电动机的加速、减速以及电动机的正、反转并且可以调节转速。其原理为通过单片机产生PWM信号并输入到L298驱动芯片来控制直流电动机工作。

该直流电动机的PWM控制系统由以下部分组成：

设计输入部分，这一部分主要是利用带中断的独立式键盘来实现对直流电动机的加减速和正反转以及急停的控制。

设计控制部分:主要由单助机的外部中断扩展电路组成。直流电机PWM控制实现部分要由一些二极管、电机和L298直流电机驱区动模块组成。

3 设计与实现

## 3.1 系统模块

1. **单片机模块:**

本系统采用AT89C51单片机作为系统的主模块进行开发，负责检测与控制电机。AT89C51是一个低电历、高性能CMOS的8位片机，内含8 bytes的可反复擦写的flash只读程序有储器和256 bytes的随机存储的数据在分诸器。器件采用ATMEL公司的高密度、非易损失性存储技术生产养容标准MCS-51指令系统，片内有通用的8位中央处理器和flash存储单元，该单片机在电子行业已有非常广泛应用、

该单片机在本系统中负责的内容如下:

①读取按键输入从而读取、设定转速值和选定工作模式

②输出PWM信号并控制PWM波的占空比从而控制转速

③在液晶屏上显示设定的转速/占空比和当前转速/占空比

④进行PI调节

1. **电机驱动模块**

本系依采用L298作为直流电机的驱动芯片，其内部包含4通道逻辑驱动电路，是-种二相电机和四相电机专用的驱动器，内含2个H电桥和高压大电流双全桥式驱动电路，接收标准的TTL逻辑电平信号,可驱动46V、2A以下的电机。桥式电路保证了可以简单地实现转速和方向的控制；电子开关的速度很快，稳定性也极佳，是一种广泛采用的PWM技术。

1. **显示模块**

本系统采用液晶显示屏LCD作为输出器件。其有以下优点:

* 1. 液晶显示屏每一个点阵列在收列信号后就一直保持色彩和高度，能恒定发光，而不像阴极射线管显示器(CRT)那样需要不断刷新亮点，因此液晶显示屏画质高且不会闪烁。
  2. 液晶显示器是数字式的，和单片机系统的接口连接简单，可靠操作方便
  3. 液晶显示器的功耗主要在其内部的电极和驱动IC上，功耗较低

1. **按键模块**

按键按下闭合时电路接通，松开时电路断开。把按键引脚一端接他，另一端与单片机的某个I/0口相连。初始情况下将电平拉高，单片机在该I/O端口输出高电平，随后让单片机不断检测该I/O端口的电平情况。当按键按下时，此时该I/0相当于直接与地相连电平被置为低，检测到了按键发生从而单片机会执行相应程序。

1. **电机与测速偏码器**

测速编码器选择的是仿真工程中自带的直流电动机机测速编码器，其两个引脚可分别输出两路相差一定相位的测速脉冲，单片机通计采集这两路脉冲的频率和相位就可以得到电机的转速和转向。

## 3.2 系统方案

### 3.2.1 速度加减的实现:

单片机通过控制 L298 的使能端“允许”或者“禁止”即改变 a脉冲宽度的值，从而达到控制 PWM 脉冲宽度调节电机转速的目的，即用 P1.1 通过软件延时得到 PWM 信号与 ENA 引脚相连，来调节电机的加、减速。

### 3.2.2 正反转的控制:

单片机通过 L298 中的 H 桥，从 AT89C51中的 P1.0 输出控制信号与 L298 的 IN2 相连，同时 P1.0 输出的电信号再与非门相连再输入到 IN1 来达到控制 BJT 的基极电压， IN1 与 IN2 具有互锁的控制 L298 中 H 桥的 BJT 通断，从而达到控制电机转向的目的。

### 3.2.3 电机速度的显示 :

单片机通过 P3.4即T0口接受电机发出的脉冲信号（在仿真是采用 proteus 里 MOTOR-encoder 电机，用这个可以简单的测得电机的转速，主要原理是编码器可以根据电机转一圈输出脉冲数，根据统计的脉冲量得到电机的转数，中间的是编码器转一周，高电平一次。根据这可以测出转速。左右两边是检测左转还是右转，哪边先高电平电机就是往哪边转）。在本次设计采用检测每转动一周高电平一次来简单地计算电机的转速。单片机 T1 采用模式一定时中断模式， T0 采用模式一计数中断模式，由高 8 位 TH0 和低 8 位 TL0 两个 8 位寄存器组成，当设定计算值为 65536-50000=15536 （D）即十六进制 3CB0(H) ，此时， TH0=3C ， TL0=B0 ，定时器 T1 计数 50ms ，此时电机每转动一周 P3.4 高电平一次，存储电机转动的转数，同时采集 2.5S 之后将电机的转数进行 LCD 显示。

## 3.3 原理分析

### 3.3.1 PWM调速原理

改变两端的电压，从而达到控制要求的一种电压调整方法。PWM可以应用在许多方面，比如：电机调PWM（脉冲宽度调制）是通过控制固定电压的直流电源开关频率，改变负速、温度控制、压力控制等等。

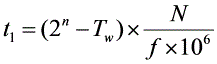
在PWM驱动控制的调整系统中，按一个固定的频率来接通和断开电源，并且根据需要改变一个周期内“接通”和“断开”时间的长短。通过改变直流电机电枢上电压的“占空比”来达到改变平均电压大小的目的，从而来控制电动机的转速。

设电机始终接通电源时，电机转速最大为Vmax，设占空比为D= t / T，则电机的平均速度为Va = Vmax \* D，其中Va指的是电机的平均速度；Vmax 是指电机在全通电时的最大速度；D = t / T是指占空比。

由上面的公式可见，当我们改变占空比D=t1/T时，就可以得到不同的电机平均速度Vd,从而达到调速的目的。严格来说，平均速度Vd与占空比D并非严格的线性关系，但是在一般的应用中，我们可以将其近似的看成是线性关系。

PWM 信号的产生通常有两种方法：一种是软件的方法；另一种是硬件的方法。本文主要介绍利用单片机对 PWM 信号的软件实现方法。 51 系列典型产品 AT89C51 具有两个定时器 和 。通过控制定时器初值 和 ，从而可以实现从 C51 的任意输出口输出不同占空比的脉冲波形。 大致的的编程思路是这样的： T0 定时器中断是让一个 I/O 口输出高电平，在这个定时器 T0 的中断当中起动定时器 T1 ，而这个 T1 是让 I/O 口输出低电平，这样改变定时器 T0 的初值就可以改变频率，改变定时器 T1 的初值就可以改变占空比。

如果单片机的时钟频率为 ，定时器／计数器为 位，则定时器初值与定时时间的关系为：



—— 定时器定时初值；

—— 一个机器周期的时钟数；

本设计以 AT89C51 单片机为核心，以 5 个弹跳按钮（"ON/C"、"="、"+"、"-"）作为输入达到控制直流电机的反转、正转、加速、减速。设计中采用 PWM 技术对电机进行控制，通过对占空比的计算达到调速的目的。

### 3.3.2 PI算法

模拟PI控制器的基本算式为:

对上式离散化得

### 3.3.3 程序流程

主程序主要完成单片机以及系统外设的初始化和系统运行的循环控制、PWM信号发生。初始化包括定时器初始化、中断初始化、按键初始化、LCD模块、L298模块初始化。循环控制主要检测键盘状态是否改变，根据按键情况修改占空比从而改变转速和转向。整个程序的主流程图如下：

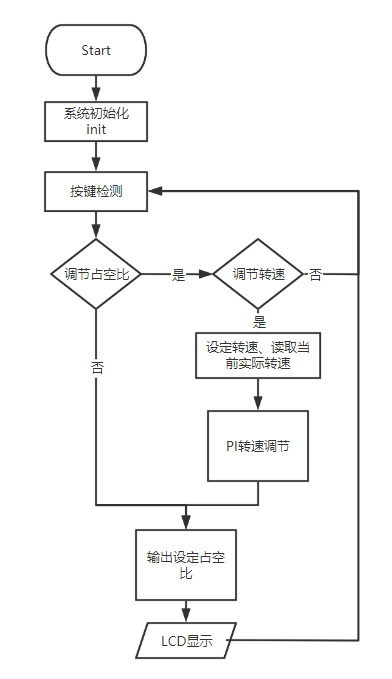


图3-1 主程序流程图

4 测试和结果

## 4.1 设计仿真原理图

Protues未按下仿真按钮时，系统仿真的初始情况如图4-1所示。

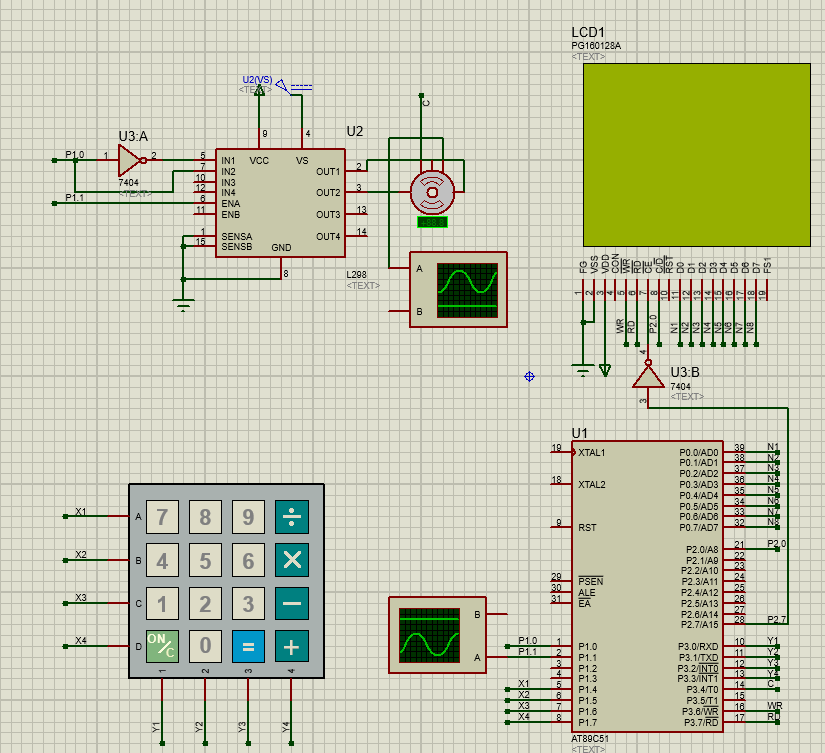


图4-1 系统仿真初始图

当启动 Protues 进行仿真后，系统运行效果如图示4-2，此时LCD显示屏显示处于上电启动状态。

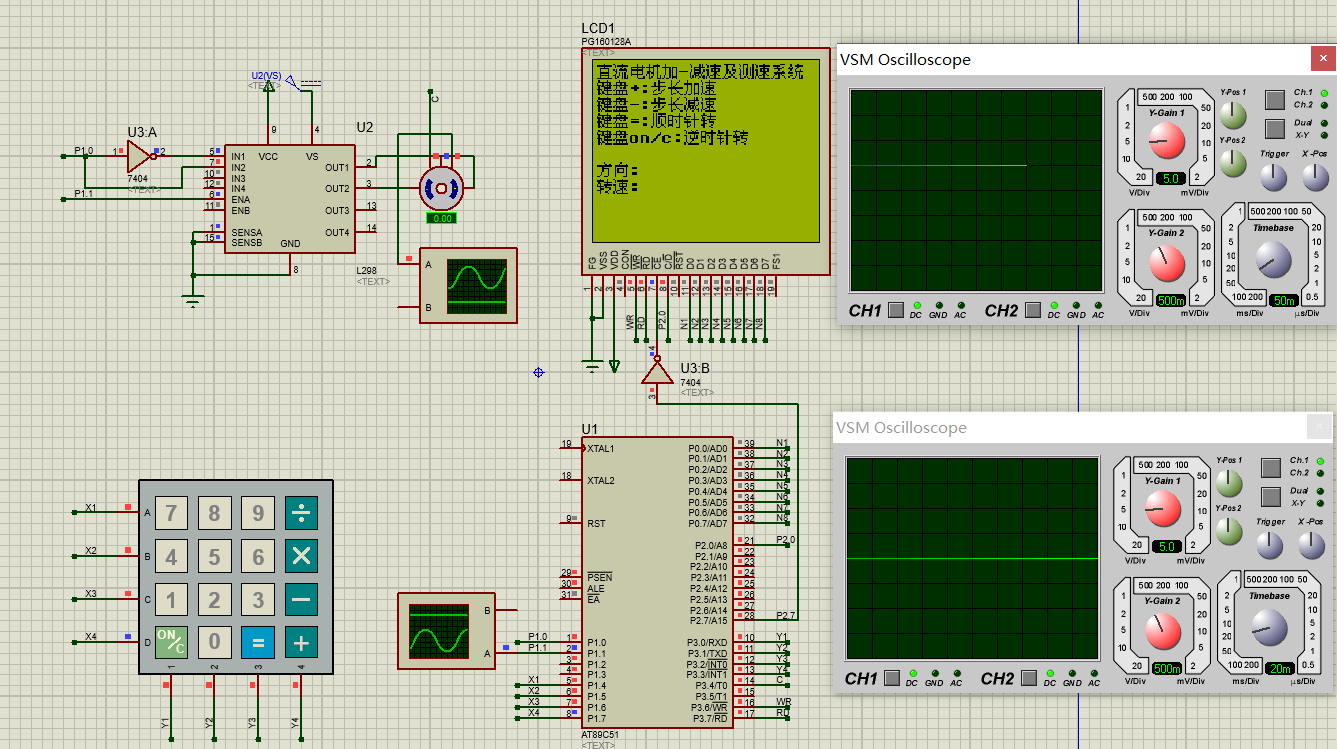


图4-2 启动仿真时的效果图

## 4.2 正反转实现

正常运行时，直流电机有图4-3的运行结果。由图可以看出电机的转速显示为顺时针120 r/min。

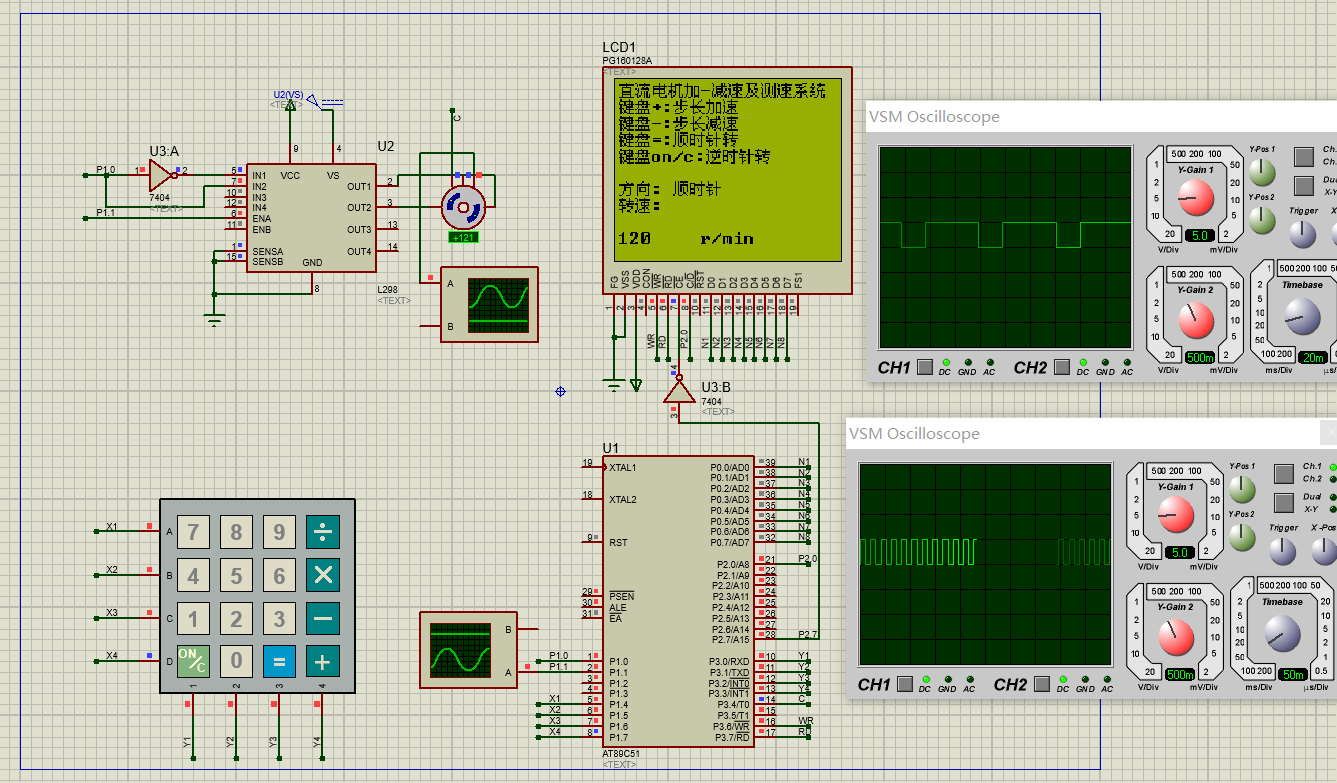


图4-3 按下启动按钮启动时的效果图

按下反转键，可以看到直流电机开始向反方向转动即逆时针, 此时LCD显示转速为124 r/min ，有图3-4的运行结果。

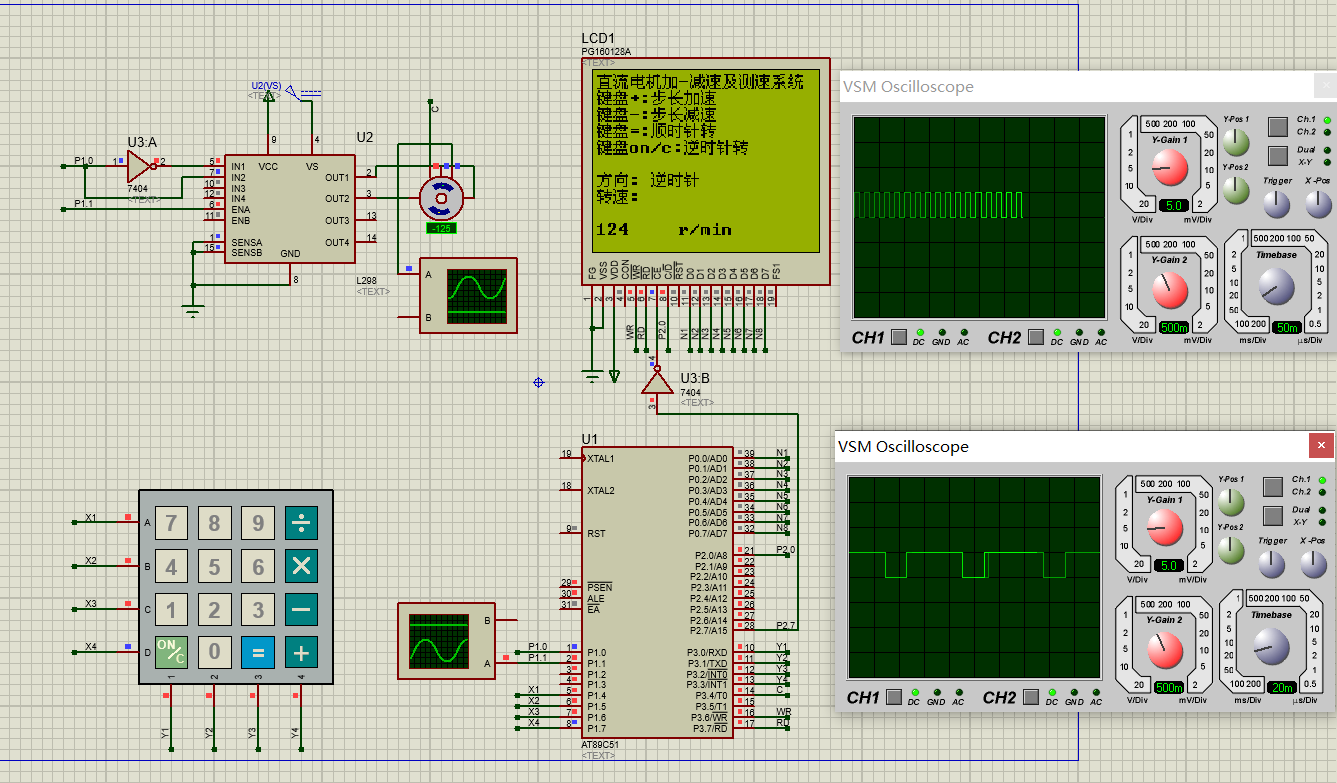


图4-4 按下反转按钮(“ON/C”)时的效果图

## 4.3 加减速实现

按下顺时针按钮(“=”)，此时电机恢复之前的顺时针120 r/min。此时按下“+”按键后，可以看到转速提高到135 r/min。

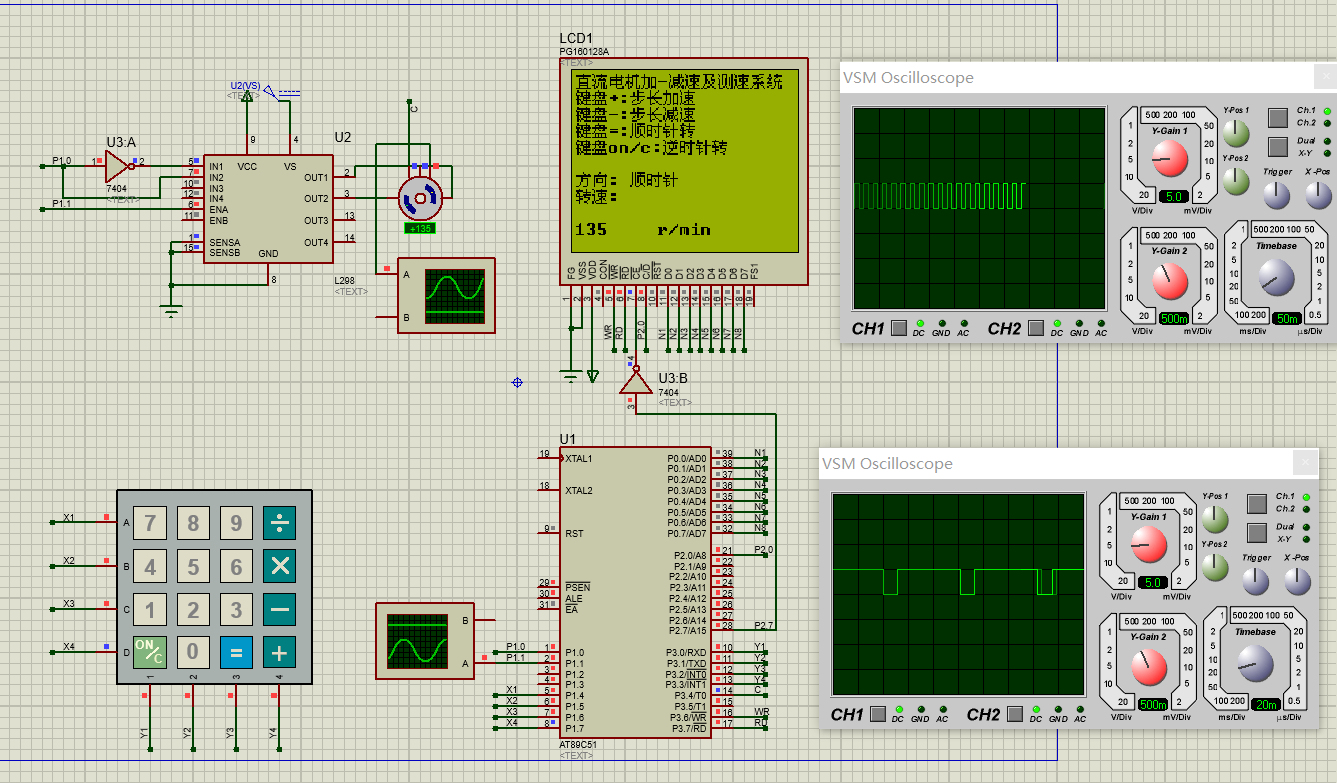


图4-5 按下步长加速（“+”）按钮时的效果图

之后按下步长减速按键“-”转速又恢复了120 r/min， 再按一次“-”可以看到转速降为了112 r/min。

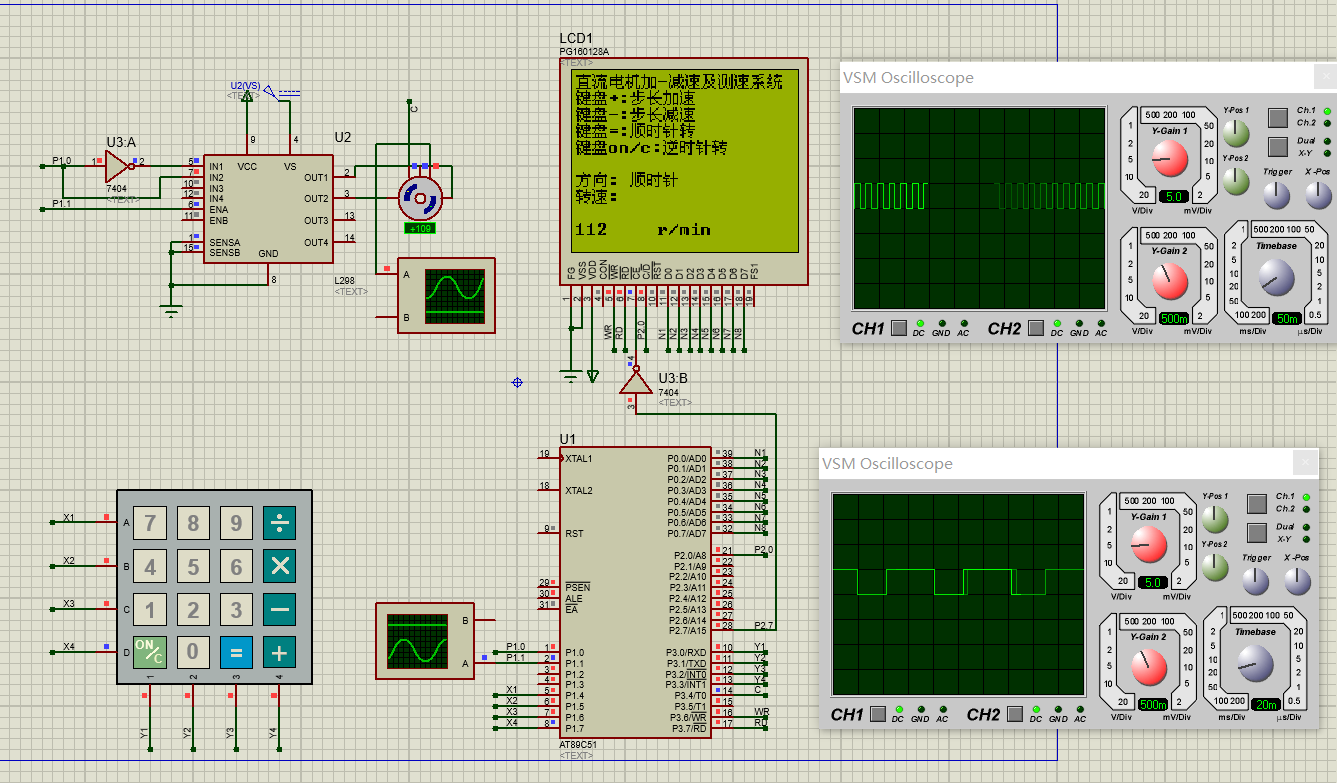


图4-5 按下步长减速（“-”）按钮时的效果图

4 遇到的问题

第一次使用Protues有很多功能都不是很清楚，找测速元件找了挺久。以及总是有引脚忘记引出，导致调试了很久。

示波器的波形显示第一次关闭后，之后找不到打开，后来才发现可以从菜单栏的DEBUG->Reset Debug popup windows,按下确定后再次运行仿真示波器界面就又出来了。

5 实验心得

该直流电机调速系统是以51单片机为核心的，采用PWM软件方法来实现的调速过程具有更大的灵活性和更低的成本，它能够充分发挥单片机的效能，对于简易速度控制系统的实现提供了一种有效的途径。而在软件方面，采用PI算法来确定闭环控制的补偿量也是由数字电路组成的直流电机闭环调速系统所不能及的。本次采用比较可靠实用的L298芯片为驱动电路主控芯片，这使我更加了解了这些东西，对我自身的提高有极大的帮助。

通过这次设计，使我的设计思路有了很大的提高，使我的分析与解决实际问题能力以及动手操作能力得到了很大的巩固和提高。通过该系统设计的学习，更熟悉的了解和掌握单片机应用系统设计的基本方法、设计思路、设计步骤等全过程，达到了理论联系实际，学以致用的目的。

6 参考文献

[1] 周立功 . 直流电机原理与驱动 [M]. 西安：西安电子科技大学出版社 .2008.1

[2] 江志红 .51 单片机技术与应用系统开发开发系统案例精选 [M]. 北京：清华大学出版社， 2008.12

[3] 王选民 . 智能仪器原理与设计 [M]. 北京：清华大学出版社 .2008.7

[4] 杨欣 , 王玉凤 .51 单片机应用从零开始 [M]. 北京：清华大学出版 .2008 年

[5] 李泉溪 . 单片机原理与应用实例仿真 [M]. 北京：北京航空航天大学出版社 .2009.8

[6] 刘保录 . 基于单片机的电机综合参数测试仪设计 [M]. 第 10 卷第 2 期

[7] 岳东海 . 直流电机 PWM 无级调速控制系统设计 [J] . 价值工程2010(2)135-136

附录代码

Main.c主程序代码

1. #include "showfun.h"
2. #include <stdio.h>
4. **extern** **char** fnLCMInit();                                 // LCM 初始化
5. **extern** **void** at(unsigned **char** x,unsigned **char** y);          // 设定文本x,y值
6. **extern** **void** cls();                                         // 清屏
7. **extern** **void** charout(unsigned **char** \*str);              //ASCII(8\*8) 显示函数
8. **extern** **void** fnSetPos(unsigned **char** urow, unsigned **char**  ucol);      // 设置当前地址
9. **extern** uchar dprintf(uchar x,uchar y,**char** \*fmt);                    // ASCII(8\*16) 及 汉字(16\*16) 显示函数
10. **extern** uchar fnPR12(uchar uCmd);                         // 写无参数的指令
11. **extern** uchar fnPR13(uchar uData);                            // 写数据
12. **extern** unsigned **int** Adc0832(unsigned **char** channel);
13. **extern** **void** Line( unsigned **char** x1, unsigned **char** y1, unsigned **char** x2, unsigned **char** y2, bit Mode);
14. **extern** **void** Pixel(unsigned **char** PointX,unsigned **char** PointY, bit Mode);

17. uchar dsp[10]={0,0,0,0,0,0,0,0,0,0,};
18. **char** abc[3]={0,0,0,};
19. uchar key=0;
20. uint a=100;
21. uchar n=5;
22. uchar count=1;
23. uint k1=0;
25. uchar GeyKey();
26. **void** delay(uchar i);
27. **void** control();

30. main()
31. {
32. fnLCMInit();
33. fnSetPos(0,0);
34. dprintf(0,0,"直流电机加-减速及测速系统");
35. dprintf(0,12,"键盘+:步长加速");
36. dprintf(0,24,"键盘-:步长减速");
37. dprintf(0,36,"键盘=:顺时针转");
38. dprintf(0,48,"键盘on/c:逆时针转");
40. dprintf(0,72,"方向:");
41. dprintf(0,84,"转速:");
43. P1\_1=0;
44. TMOD=0x15;
45. TH1=0x3c;
46. TL1=0xb0;
47. TH0=0x00;
48. TL0=0x00;
49. ET0=1;
50. ET1=1;
51. TR0=1;
52. TR1=1;
54. **while**(1)
55. {
56. key=GeyKey();
57. **switch**(key)
58. {
59. **case** '1': {          a=10;
60. //sprintf(abc,"%3.2f",a/255.0);
61. dprintf(0,96,"  5    r/min");
62. //dprintf(0,96,abc);
63. dprintf(60,96,"r/min");
64. **break**;      }
65. **case** '2': {          a=25;dprintf(0,96,"  25    r/min");    **break**;      }
66. **case** '3': {          a=40; dprintf(0,96," 40    r/min");   **break**;      }
67. **case** '4': {          a=55; dprintf(0,96," 60    r/min");   **break**;      }
68. **case** '5': {          a=70; dprintf(0,96," 80    r/min");   **break**;      }
69. **case** '6': {          a=90; dprintf(0,96,"100    r/min");   **break**;      }
70. **case** '7': {          a=110; dprintf(0,96,"120    r/min");   **break**;      }
71. **case** '8': {          a=130; dprintf(0,96,"135    r/min");   **break**;      }
72. **case** '9': {          a=150; dprintf(0,96,"150    r/min");   **break**;      }
73. **case** '+': {          dprintf(0,72,"方向: 顺时针");
74. control();
75. **break**;
76. }
77. **case** '-': {         P1\_0=0;
78. dprintf(0,72,"方向: 逆时针");
79. control();
80. **break**;
81. }
82. **case** '=': {          P1\_0=1;dprintf(0,72,"方向: 顺时针");**break**;}
83. **case** 'c': {          P1\_0=0;dprintf(0,72,"方向: 逆时针");**break**;}
84. **case** '/': {          dprintf(0,72,"方向: 顺时针");
85. control();                               }
86. **default**:  **break**;
87. }
88. }
89. }
91. /\*\*
92. 键盘输入处理
93. \*/
94. uchar GeyKey()
95. {
96. P1\_4=0;
97. P1\_5=1;
98. P1\_6=1;
99. P1\_7=1;
100. P3\_0=1;
101. P3\_1=1;
102. P3\_2=1;
103. P3\_3=1;
104. \_nop\_();\_nop\_();
105. **if**(!P3\_0)**return** '7';
106. **if**(!P3\_1)**return** '8';
107. **if**(!P3\_2)**return** '9';
108. **if**(!P3\_3)**return** '/';
109. P1\_4=1;
110. P1\_5=0;
111. P1\_6=1;
112. P1\_7=1;
113. \_nop\_();\_nop\_();
114. **if**(!P3\_0)**return** '4';
115. **if**(!P3\_1)**return** '5';
116. **if**(!P3\_2)**return** '6';
117. **if**(!P3\_3)**return** '\*';
119. P1\_4=1;
120. P1\_5=1;
121. P1\_6=0;
122. P1\_7=1;
123. \_nop\_();\_nop\_();
124. **if**(!P3\_0)**return** '1';
125. **if**(!P3\_1)**return** '2';
126. **if**(!P3\_2)**return** '3';
127. **if**(!P3\_3)**return** '-';
128. P1\_4=1;
129. P1\_5=1;
130. P1\_6=1;
131. P1\_7=0;
132. \_nop\_();\_nop\_();
133. **if**(!P3\_0)**return** 'c';
134. **if**(!P3\_1)**return** '0';
135. **if**(!P3\_2)**return** '=';
136. **if**(!P3\_3)**return** '+';
138. **return** 0;
139. }

142. /\*\*
143. 延时
144. \*/
145. **void** delay(uchar i)
146. {
147. uchar j,k;
148. **for**(;i>0;i--)
149. **for**(j=15;j>0;j--)
150. **for**(k=11;k>0;k--);
151. }
153. /\*\*
154. 中断, 显示当前转速
155. \*/
156. **void** time()interrupt 3
157. {
158. TR1=0;
159. count++;
160. k1+=TL0;
161. **if**(count==51)
162. {
163. sprintf(dsp,"%3d",k1);
164. dprintf(0,108,dsp);
165. dprintf(60,108,"r/min");
166. count=1;
167. k1=0;
168. }
169. TH1=0x3c;
170. TL1=0xb0;
171. TH0=0x00;
172. TL0=0x00;
173. TR1=1;
174. }
176. /\*\*
177. 控制实现模块: 转方向和步长加/减速
178. \*/
179. **void** control()
180. {
181. EA=1;
182. **while**(1){
183. **if**(a>=150)
184. a=150;
185. **if**(a<=10)
186. a=10;
187. P1\_1=0;
188. delay(160-a);
189. P1\_1=1;
190. delay(a);
191. key=GeyKey();
192. **if**(key=='-') a-=n;
193. **else**
194. **if**(key=='+') a+=n;
195. **else**
196. **if**(key=='=')
197. {  P1\_0=1;
198. dprintf(0,72,"方向: 顺时针");
199. }
200. **else**
201. **if**(key=='c')
202. {  P1\_0=0;
203. dprintf(0,72,"方向: 逆时针");
204. }
205. **else**
206. **if**(key=='\*')
207. {   P1\_1=0;
208. **break**;
209. }
210. **else**
211. **if**(key!=0)
212. **break**;
213. }
214. EA=0;
215. }