两相四线励磁式步进电机工作原理

本章将介绍在嵌入式平台UP－NETARM2410－S中步进电机的实现。步进电机在各个领域诸如机器人、智能控制、工业控制等方面都有着广泛的应用空间，本章着重介绍步进电机的工作原理及编程实现步进电机驱动的方法，主要内容如下：

l 步进电机的概述
l 步进电机的工作原理
l 和微处理器的总线连接方式
l 驱动程序的编程
l Linux 下用软件的方法实现步进电机的脉冲分配，用软件的方法代替硬件的脉冲分配器

1. 步进电机概述

步进电机是一种能够将电脉冲信号转换成角位移或线位移的机电元件，它实际上是一种单相或多相同步电动机。单相步进电动机有单路电脉冲驱动，输出功率一般很小，其用途为微小功率驱动。多相步进电动机有多相方波脉冲驱动，用途很广。

使用多相步进电动机时，单路电脉冲信号可先通过脉冲分配器转换为多相脉冲信号，在经功率放大后分别送入步进电动机各相绕组。每输入一个脉冲到脉冲分配器，电动机各相的通电状态就发生变化，转子会转过一定的角度（称为步距角）。

正常情况下，步进电机转过的总角度和输入的脉冲数成正比；连续输入一定频率的脉冲时，电动机的转速与输入脉冲的频率保持严格的对应关系，不受电压波动和负载变化的影响。由于步进电动机能直接接收数字量的输入，所以特别适合于微机控制。

1.1 步进电机的特性

步进电机转动使用的是脉冲信号，而脉冲是数字信号，这恰是计算机所擅长处理的数据类型。从20世纪80年代开始开发出了专用的IC驱动电路，今天，在打印机、磁盘器等的OA装置的位置控制中，步进电机都是不可缺少的组成部分之一。总体上说，步进电机有如下优点：

1．不需要反馈，控制简单。
2．与微机的连接、速度控制（启动、停止和反转）及驱动电路的设计比较简单。
3．没有角累积误差。
4．停止时也可保持转距。
5．没有转向器等机械部分，不需要保养，故造价较低。
6．即使没有传感器，也能精确定位。
7．根据给定的脉冲周期，能够以任意速度转____动。

但是，这种电机也有自身的缺点。

8．难以获得较大的转矩
9、不宜用作高速转动
10．在体积重量方面没有优势，能源利用率低。

11．超过负载时会破坏同步，速工作时会发出振动和噪声。

1.2 步进电机的种类

目前常用的步进电机有三类：

1、反应式步进电动机（VR）。

采用高导磁材料构成齿状转子和定子，其结构简单，生产成本低，步距角可以做的相当小，但动态性能相对较差。

2、永磁式步进电动机（PM）。

转子采用多磁极的圆筒形的永磁铁，在其外侧配置齿状定子。用转子和定子之间的吸引和排斥力产生转动，转动步的角度一般是7.5o。它的出力大，动态性能好；但步距角一般比较大。

3、混合步进电动机（HB）。

这是PM 和VR的复合产品，其转子采用齿状的稀土永磁材料，定子则为齿状的突起结构。此类电机综合了反应式和永磁式两者的优点，步距角小，出力大，动态性能好，是性能较好的一类步进电动机，在计算机相关的设备中多用此类电机。

2、步进电机的工作原理

现以反应式三相步进电机为例说明其工作原理。

三相步进电机的定子铁心上有六个形状相同的大齿，相邻两个大齿之间的夹角为60 度。每个大齿上都套有一个线圈，径向相对的两个线圈串联起来成为一相绕组。各个大齿的内表面上又有若干个均匀分布的小齿。

转子是一个圆柱形铁心，外表面上圆周方向均匀的布满了小齿。转子小齿的齿距是和定子相同的。设计时应使转子齿数能被二整除。但某一相绕组通电，而转子可自由旋转时，该相两个大齿下的各个小齿将吸引相近的转子小齿，使电动机转动到转子小齿与该相定子小齿对齐的位置，而其它两相的各个大齿下的小齿必定和转子的小齿分别错开正负1/3 的齿距，形成"齿错位"，从而形成电磁引力使电动机连续的一直转动下去。

和反应式步进电动机不同，永磁式步进电动机的绕组电流要求正，反向流动，故驱动电路一般要做成双极性驱动。混合式步进电动机的绕组电流也要求正，反向流动，故驱动电路通常也要做成双极性。

2.1、步进电机的励磁方式

步进电机有2相、4相和5相电机。在4相电机中有4组线圈，若电流按顺序通过线圈则使电机产生转动。2相电机中有2 组线圈。从图9.3可以发现，在各线圈中引出中间端子，因此若以中间端子为基准即可实现4相，称这4为A、B、C、D 的励磁相。本实验使用的就是这种方式的4相电机，而励磁方式中有1 相（单向）励磁、2 相（双向）励磁和1－2 相（单－双向）励磁方式。此外，如果转动的方向不正确，可以交替1、2端子或3、4号端子

(1). 1相励磁方式

按ABCD的顺序总是仅有一个励磁相有电流通过，因此，对应1个脉冲信号电机只会转动一步，这使电机只能产生很小的转矩并会产生振动，故很少使用。

A B C D

T1 1 0 0 0
T2 0 1 0 0
T3 0 0 1 0
T4 0 0 0 1
图2.T1－T4表示脉冲周期；ABCD表示电机的各相，1表示此时有一个脉冲，0表示没有
(2). 2相励磁方式
按AB、BC、CD、DA的方式总是只有2相励磁，通过的电流是1相励磁时通过电流的2倍，转矩也是1相励磁的2 倍。此时电机的振动较小且应答频率升高，目前仍广泛使用此种方式。
A B C D
T1 1 1 0 0
T2 0 1 1 0
T3 0 0 1 1
T4 1 0 0 1
图3.T1－T4表示脉冲周期；ABCD表示电机的各相，1表示此时有一个脉冲，0表示没有脉冲
(3). 1－2相励磁方式
即实验中所有的励磁方式，它按A、AB、B、BC、C、CD、D、DA的顺序交替进行线圈的励磁。与前述的2个线圈励磁方式相比，电机的转速是原来的1/2，应答频率范围变为原来的2倍。转子以滑动的方式转动。
A B C D
T1 1 0 0 0
T2 1 1 0 0
T3 0 1 0 0
T4 0 1 1 0
T5 0 0 1 0
T6 0 0 1 1
T7 0 0 0 1
T8 1 0 0 1
图4.T1－T8表示脉冲周期；ABCD表示电机的各相，1表示此时有一个脉冲，0表示没有脉冲

3. 与微处理器的连接方式

步进电机并不是直接与CPU相连的。由于开发板上外部设备很多，各功能模块与微处理器的连接方式有专用线路，局部总线与扩展总线。

3.1 局部总线与扩展总线

局部总线与微处理器直接相连，扩展总线通过一个总线控制器74LVCH6245与局部总线相连。
从CPU 出来的数据、地址、读写控制等信号构成局部总线。NAND FLASH、SDRAM和网卡芯片AX88796 直接挂在局部总线上的。局部总线经过四片74LVCH16245 驱动后作为扩展总线引到其他外设以及168Pin 扩展槽。由于数据线是双向的，所以16245芯片必须有方向控制信号，这里采用经

过隔离后的写控制信号OE 作为数据线所在16245 芯片的方向控制线。当OE有效时16245 芯片把扩展总线上的数据传输到局部总线上；当OE无效时反之。另外，必须注意，当系统对局部总线上的芯片读数据时OE 一样会起作用，这样就必须对局部总线和扩展总线进行总线仲裁，这里是外设所具有的四个片选信号nGCS1、3、4、5 用74HC21 相与后作为数据线所在的16245 芯片的输出使能控制线，只有当系统对扩展总线读操作，也就是上述四个片选之一有效时，16245 才能对局部总线输出数据，否则无论OE 如何都呈现高阻态。如下图：
注：LDATA表示局部总线的数据线；DATA表示扩展总线的数据线。
74LVCH16425芯片共有四块，限于篇幅仅举一片为例，其它三片芯片，一片用于数据线的低位连接，两片用于地址线的连接。
注：由于片选线使用的负逻辑电平有效，所有此处用的是与门

3.2芯片74HC573

扩展总线连接在芯片74HC573上，扩展总线的DATA0－DATA7分别接在74HC573的八个数据输入端上。74HC573芯片是由8个三态门组成的寄存器，它起到暂时保存信息和隔离总线的作用。芯片的输出I\O0—I\O3用于D\A数模转换，I\O4—I\O7用于步进电机的控制。
描述：
这个芯片采用的是八进位的D触发器，它可以驱动电容式或电阻式的负载。因此它特别适合应用于缓冲寄存器、IO端口、双向的总线控制器、和操作寄存器。
当寄存器的使能端（LE）为高电平时，Q输出端和__ _ D输入端一一对应；当LE为低电平时，输出端管脚Q输出的是寄存器中已被设定的值。当一个能开启缓冲功能的负逻辑管脚（OE）为0时，无论是在正常逻辑状态还是在高阻抗状态下，都能放置八位的输出数据。在高阻抗状态下。输出并没有负载或者进行控制总线。高阻态和改进的总线驱动可以在不拉起元件的情况下控制总线传输。OE端并不影响寄存器内部的操作。当输出端呈高阻状态时旧的数据可以被保存或者新的数据进行输入。

3.3步进电机模块的驱动电路

74HC573芯片的输出I\O4—I\O7用于用来驱动步进电机的转动。数据线要首先接在步进电机模块的一个接口上。接口对信号进行放大，使之能够驱动步进电机。放大后的信号就可以直接的来使步进电路进行工作了。

3.4、开发板中的步进电机

本开发板中使用的步进电机为四相步进电机。转子小齿数为64。
系统中采用四路I/O 进行并行控制，ARM 控制器直接发出多相脉冲信号，在通过功率放大后，进入步进电机的各相绕组。这样就不再需要脉冲分配器。脉冲分配器的功能可以由纯软件的方法实现如上图所示。
四相步距电机的控制方法有四相单四拍，四相单、双八拍和四相双四拍三种控制方式。
步距角的计算公式为：
$\theta b = 360^0 / mC_k$

其中：m 为相数，控制方法是四相单四拍和四相双四拍时C 为1，控制方法是四相单、双八拍时C 为2，Z_k为转子小齿数。

本系统中采用的是四相单、双八拍控制方法，所以步距角为360°/512。

但步进电机经过一个1/8 的减速器引出，实际的步距角应为360°/512/8。

开发平台中使用EXI/O 的高四位控制四相步进电机的四个相。按照四相单、双八拍控制方法，电机正转时的控制顺序为A→AB→B→BC→C→CD→D→DA。EXI/O 的高四位的值参见下表：

十六进制二进制通电状态

1H 0001 A

3H 0011 AB

2H 0010 B

6H 0110 BC

4H 0100 C

CH 1100 CD

8H 1000 D

9H 1001 DA

表5. 电机正转时，EXIO的高四位的值

反转时，只要将控制信号按相反的顺序给出即可。可以通过宏SETEXIOBITMASK(bit,mask)（EXIO.h）来设置扩展I/O 口，其中mask 参数为0xf0。

本实验使作的是1－2相励磁方式，还可以使用1相励磁方式和2相励磁方式。

＊1相励磁方式的顺序是ABCD，因此只要设置数组

char stepdata[]={0x10,0x20,0x40,0x80} 即可

＊2相励磁方式的顺序是AB，BC，CD，DA，因此只要设置数组

char stepdata[]={0x30,0x60,0xc0,0x90} 即可

＊要实现电机的反转，只需将上面数组的值按相反的顺序排列即可

4、驱动程序的编程

驱动程序采用C语言进行编程，下面是驱动程序中几个重要的函数。

```
static intdo_stepmotor_run(char phase) //通过调用这个函数使步进电机转起来
{
unsigned intbak; //变量bak用于存放从寄存器中读出的值
bak = readw(s3c2410_exio_base); //调用内核函数将基地址为s3c2410_exio_base寄存
//器的值读出并放在bak中
DPRINTK_STEP("s3c2410_exio_base content is %x\n", bak); //调试输出语句
tiny_delay(5); //延时函数，有延时功能
bitops_mask_bit(phase, 0xf0, &bak);
```

```
//清除4－8位然后再设置phase传进来
的位（也是4－8位）
DPRINTK_STEP("s3c2410_exio_base content is%x\n", bak); //调试输
出语句
tiny_delay(5);
writew(bak, s3c2410_exio_base); //调用内核函数将bak的值写入相应的
寄存器中
bak = readw(s3c2410_exio_base); //再次将寄存器的值读出
DPRINTK_STEP("s3c2410_exio_base content is %x\n", bak);
//利用调试语句再次将修改后的寄存器中的值输出，以验证其正确性
tiny_delay(5);
DPRINTK_STEP("\n");
return 0;
}
```

驱动程序主要通过上面这个函数来使步进电机转动。控制电机的是一个寄存器(地址是s3c2410_exio_base=0x08000100)，只要给它适当的值电机就可以运动起来。这个函数通过传递变量phase从应用程序获得数据。然后使用函数readw()把寄存器的值就读出并送给bak；
通过函数bitops_mask_bit()修改bak的值；最后由writew()写回到寄存器中。

5、应用程序的编程

1、步进电机模块

步进电机模块和DA 模块是使用Bank1 地址空间扩展出来的IO口。共同使用驱动
s3c2410-exio.o。
在驱动程序中，与步进电机相关的主要在函数s3c2410_exio_ioctl：

2、对应的应用源程序

```
#include <stdio.h>
#include <fcntl.h>
#include <string.h>
#include <sys/ioctl.h>
#define STEPMOTOR_IOCTRL_PHASE 0x13
staticintstep_fd = -1;
char *STEP_DEV="/dev/exio/0raw"; //定义一个指针指向步进电机的驱动
程序
/********* A, AB, B, BC, C CD, D, DA ***/
char stepdata[]={0x10,0x30,0x20,0x60,0x40,0xc0,0x80,0x90};//各
个相位对应的值
void Delay(int t) //延时函数
{
int i;
```

```
for(;t>0;t--)
for(i=0;i<400;i++);
}
/*************************************************************
***********/
int main(intargc, char **argv)
{
int i = 0;
if((step_fd=open(STEP_DEV, O_WRONLY))<0){
printf("Error opening /dev/exio/0raw device\n");
return 1;
}
/*
打开设备的驱动程序，由于LINUX把所有的设备都模拟成文件。
step_fd=open(STEP_DEV,0_WRONLY)实际调用的函数为：
staticint s3c2410_exio_open(structinode *inode, struct file *filp)
//驱动程序中的设备打开程序
*/
for (;;) {
for (i=0; i<sizeof(stepdata)/sizeof(stepdata[0]); i++) {
ioctl(step_fd, STEPMOTOR_IOCTRL_PHASE,
stepdata[i]);
}
/*程序进入一个死循环，这样可以使电机在没有人为停止的状况下，一直的
转动下去。
*第二层for语句循环一次即电机转动一周。函数ioctl（）对应函
数*s3c2410_exio_ioctl（）
*而这个函数最终将调用函数do_stepmotor_run((char)arg);使步进电
机转动起来。
*/
printf("Delay(100)\n");
Delay(100);
}
close(step_fd); //程序结束时关闭设备
printf("Step motor start running!\n");
return 0;
}_
```