第 35 卷 第 1 期 计 算 机 工 程 2009 年 1 月
Vol.35 No.1 Computer Engineering January 2009
·工程应用技术与实现· 文章编号：1000—3428(2009)01—0229—03 文献标识码：A 中图分类号：TP311

基于 WinCE5.0 的高速绣花机步进电机驱动

白瑞林，刘 洋，曲明波
(江南大学智能控制研究所，无锡 214122)

摘 要：高速绣花机控制系统必须在高速的情况下保证绣品的质量，同时维持较低的噪音。难点是要在动态限定的时间内控制 *X*/*Y* 步进电机，使之平稳精确。该文在分析系统特点的基础上，以 ARM9 处理器结合 CPLD 构成步进电机控制单元，提高控制效率并降低编程复杂度。以旋转动力学方程和步进电机的矩频特性为依据，分析步进控制脉冲数据的选取、中断的处理和流驱动的结构，实现 WinCE5.0 下步进电机流驱动程序的开发。

关键词：绣花机；步进电机；流驱动

Step Motor Driver for High Speed Embroidery Machine Based on WinCE5.0

BAI Rui-lin, LIU Yang, QU Ming-bo
(Institute of Intelligent Control, Jiangnan University, Wuxi 214122)

【Abstract】High speed embroidery machine has to produce high quality product at high speed, meanwhile, to keep low noise. The difficulty lays in how to control the step motor *X*/*Y* precisely and steadily within the dynamically limited time. ARM9 controller together with CPLD is used to construct the control system to improve efficiency of control and reduce the complexity of coding. Based on the research of embroidery control system and according to the kinetics function of the whirling objects and the moment-frequency feature of the step motor, the selection of control pulses, interrupt handling and architecture of the stream driver are analyzed, and the development of the stream driver for the step motors under WinCE5.0 is completed.

【Key words】embroidery machine; step motor; stream driver

1 概述

我国绣花机的产量约占全球的 70%，在由缝纫机、编织机等 40 多个品种组成的缝制设备行业中，绣花机产值约占 1/4。但国产绣花机基本上属于中低档机，平均价格比国外先进机型相差很多，在速度、噪声、品质、效率和功能方面也有不少差距。高速绣花机的开发不仅是市场的需求，更有助于进一步提升行业产值，提升民族品牌价值。高速绣花机的开发要从刺绣速度、运行噪声、产品质量、效率和功能等多方面着手，不仅要求有高的刺绣速度来提高生产率，而且要保证绣品的高质量。同时，刺绣过程中的噪声要尽可能降低。一般而言，噪声正比于速度，质量反比于速度。因此，高速绣花机开发的关键包括：步进电机的细分精度及加/减速控制；齿轮等机械传动装置的加工及装配精度；对断线检测、自动剪线装置等系统的协同控制。其中，步进电机加/减速控制的平稳与否起决定作用。

2 绣花机步进电机控制的分析

2.1 绣花机步进电机控制的特点

在绣花机中，*X*/*Y* 步进电机的角位移通过传动机构转换为 *X*/*Y* 方向的水平位移，从而控制绣框水平运动。根据花样文件，1 针位移范围为 0.1 mm~12.7 mm。每绣 1 针，须发送的脉冲数为 $n(n\in[1, 127])$。为保证刺绣平稳，要经过加速→匀速→减速或加速→减速阶段(n 接近 1 时除外)。刺绣速度为 0~1 000 s/min(s 代表针)，对应主轴速度为 0～1 000 r/min。绣针处于布面上的时间占主轴旋转 1 圈 360°中的 200°。对不同的刺绣速度，完成每针 n 个脉冲所允许的时间(绣针处于布面上的时间)是不同的。

在相同刺绣速度下，随着发送脉冲数 n 的不同，加减速过程中的脉冲分配也不同。这正是绣花机控制系统中步进电机控制与一般步进电机升降速控制的不同，即难点所在，既非单纯速度控制，也非单纯位置控制，而是限定时间内完成升降速过程的强实时控制。为获得绣针相对布面的垂直位置，即允许步进电机动作的 200°的起始和结束时刻，须对主轴伺服光电编码器返回脉冲计数。

2.2 步进电机特性分析

步进电机控制的关键是要确保不失步或过冲，维持加减速过程平稳，以达到精确定位和降低噪声的目的。因此，须根据步进电机的动态特性和矩频特性来计算脉冲的频率和个数[1]。

步进电机的动态特性，可通过旋转动力学方程来描述[2]：

$$J\frac{\theta\times 2\pi}{360}\times\frac{\mathrm{d}f}{\mathrm{d}t}=T-T_L-T_f \tag{1}$$

其中，$J=J_r+J_L$；$T_f=K_4\theta f$；J_r 为转子惯量；J_L 为负载惯量；θ 为步距角；T 为步进电机转矩；T_L 为恒定摩擦和负载转矩；T_f 为随转速变化的摩擦和负载转矩。

作者简介：白瑞林(1955－)，男，教授，主研方向：智能控制，嵌入式系统；刘 洋、曲明波，硕士研究生

收稿日期：2008-05-23 **E-mail：**bairuilin@hotmail.com

以 110BYG350B 型步进电机为例，该步进电机的矩频特性如图 1 所示。

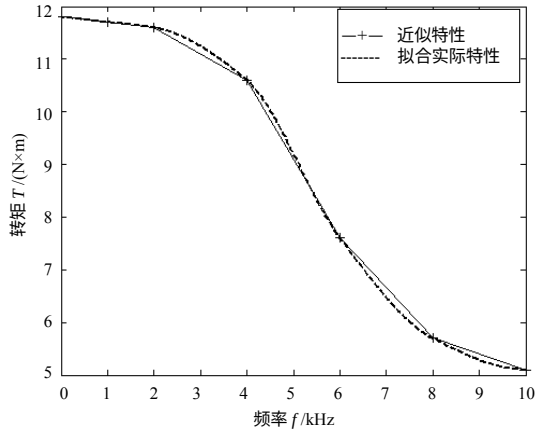

图 1　110BYG350B 矩频特性

将图 1 近似矩频特性(实线)的方程带入式(1)得(为保证较大的转矩，不考虑频率大于 6 kHz 的情况)：

$$f = f_t - (f_t - f_0)\mathrm{e}^{\frac{C_1}{A_1}(t-t_0)} \tag{2}$$

其中，f_0 为每个线段的起始处频率；$f_t = \dfrac{T_{m1} - T_L}{k_1 + k_4\theta}$；$C_1 = -(k_1 + k_4\theta)$；$A_1 = J\dfrac{\theta \times 2\pi}{360}$。且满足：

$$\begin{cases} f_0 = 0,\ t_0 = T_0,\ T_{m1} = 11.8,\ k_1 = 0.1 & (0 \le f < 2) \\ f_0 = 2,\ t_0 = T_1,\ T_{m1} = 12.6,\ k_1 = 0.5 & (2 \le f < 4) \\ f_0 = 4,\ t_0 = T_2,\ T_{m1} = 16.6,\ k_1 = 1.5 & (4 \le f \le 6) \end{cases}$$

2.3　步进加减速控制数据选取

由式(2)可计算得到理想的升降速控制数据。但由于模型的不精确以及参数的偏差，该结果须结合实际运行情况对升降速控制数据作适当的修正和调整。

根据上述分析，将刺绣速度分成 10 档，间隔 100 s/min。以每个档内的最大速度为准计算允许时间，以满足绣针位于布面上的要求。对每一档，根据脉冲个数 $n(n \in [1,127])$不同，再以 10 为间隔分成 13 级，共 130 种情况。参考式(2)取脉冲数据分别存入 130 个数组。对 n 小于 10 的情况，直接以匀速处理。在实际刺绣时，根据读取的花样数据和刺绣速度判断应采用哪一组控制数据。这样根据不同针长和不同主轴速度分别选取最佳控制数组，可大幅度提高控制的精度和平稳性，降低噪声，从而提高绣品质量。

3　产生精确步进控制脉冲的控制系统

通常步进控制脉冲的输出是由控制器自身的定时器或专用的定时器/计数器芯片(如 8254)产生，通过往初始值寄存器写不同初值来控制脉冲频率。该方法受到控制器自身工作频率的限制，产生的控制脉冲精度不够。改用 CPLD 接外部晶振来产生 *X*/*Y* 步进电机的控制脉冲，可精确灵活控制输出脉冲的频率和占空比同时减少换算工作。

CPLD 还可实现对主轴伺服控制器的光电编码器返回脉冲计数来识别主轴位置，从而确定绣针相对于水平布面的位置。系统相应结构见图 2。X_INT 和 Y_INT 为 *X*/*Y* 步进脉冲发送完的中断请求，分别接 S3C2440A 的 EINT14 和 EINT15；CNT_INT 是光电计数比较中断请求，接 EINT2。S3C2440A 通过 Bank4 地址的 0x2000 0002~0x2000 0011 给 EPM1270 发送控制 *X*/*Y* 步进的脉冲分频值和脉冲个数。CPLD 对 2 MHz 晶振分频，分频后的脉冲送步进电机驱动器。脉冲发完后由 CPLD 产生 X_INT 和 Y_INT 中断，步进电机流驱动收到中断事件会继续发送剩下的脉冲或者停止电机。

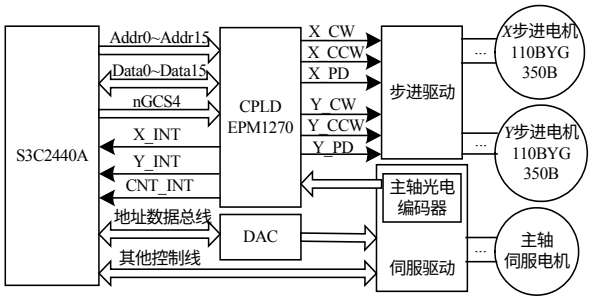

图 2　步进控制系统结构

4　实时响应多线程步进电机的流驱动

WinCE强大的功能来源于对多种外设的支持。驱动程序实际上就是为操作系统访问外设提供的接口。流驱动灵活方便，是WinCE外设使用最多的驱动类型。流驱动一般表现为动态连接库(dll)文件，在系统起动的时候由设备管理器Device.exe或图形窗口事件子系统GWES.exe模块加载[3]。

4.1　驱动程序结构

流驱动具有固定的接口形式，应用程序可通过文件系统来访问流接口函数[4]。步进驱动主要接收并执行来自应用程序的命令。取流驱动的前缀名为XYM(XY Motor driver)，需要具体实现的流接口函数有：

XYM_Init()：初始化步进电机驱动，由 Device.exe 调用；

XYM_Deinit()：设备管理器卸载电机驱动，由 Device.exe 调用；

XYM_Open()：打开一个步进电机驱动，应用程序通过 *CreateFile*() 调用；

XYM_Close()：在驱动关闭时，应用程序通过 *CloseHandle*() 调用；

XYM_IOContorl()：上层软件通过 *DeviceIoControl*() 函数调用，向步进电机发送命令；

对于其他接口函数 *XYM_Read*(), *XYM_Write*()等只须直接返回无效值。流驱动的架构形式如图 3 所示。

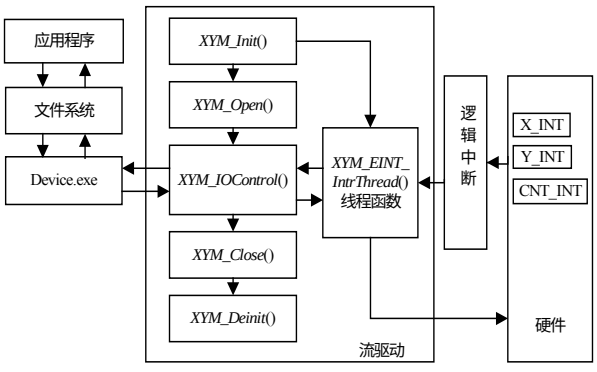

图 3　流驱动架构

来自应用程序的控制命令和参数通过文件系统提供的接口和设备管理器发送到流驱动，流驱动接收到命令，根据命令的类型和输入参数作处理。如果是步进的控制脉冲命令，则进一步触发线程函数内部的脉冲控制来完成脉冲发送。关键是中断处理部分，包括线程函数 *XYM_EINT_IntrThread*() 和 *XYM_IOControl*()函数。

4.2 虚拟地址映射

WinCE 通过虚拟地址访问地址空间，因此，驱动要完成对所访问物理地址到虚拟地址的映射，这主要由 2 个函数完成：*VirtualAlloc*()申请保留一段虚拟地址空间，*VirtualCopy*()完成待映射的物理地址到由 *VirtualAlloc*()分配的虚拟地址的映射。

为访问 CPLD 的控制寄存器，须为其分配虚拟地址空间，代码如下：

```
    v_pCPLDRegs=
(volatile CPLD_REG*)VirtualAlloc(0,sizeof (CPLD_REG),
    MEM_RESERVE,PAGE_NOACCESS);
    if
(!VirtualCopy((PVOID)v_pCPLDRegs,(PVOID)(S3C2440A_BASE_R
EG_PA_CPLD>>8),
    sizeof(CPLD_REG),PAGE_PHYSICAL|PAGE_READWRITE|PA
GE_NOCACHE))
    {VirtualFree((PVOID)v_pCPLDRegs,0,MEM_RELEASE);}
    v_pCPLDRegs=NULL; }
```

4.3 中断处理

WinCE的中断处理分成 2 个阶段：ISR(Interrupt Service Routine)和IST(Interrupt Service Thread)。ISR处于内核模式，而IST处于用户模式，ISR与IST是一对多关系[5]。外部物理中断请求IRQ(Interrupt ReQuest)通过外部中断引脚向S3C2440A发送中断信号，OAL(OEM Adaption Layer)将物理中断信号映射成逻辑中断信号，逻辑中断号被关联到相应事件，驱动程序等待与逻辑中断关联的事件，当事件被触发时，根据事件类型和时间做相应的处理。中断处理模型如图 4 所示。中断处理是驱动程序编写的关键，中断处理效率的高低直接关系到系统的实时性。

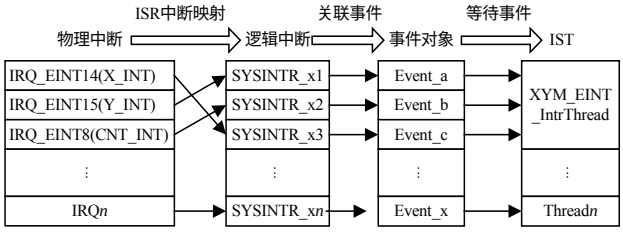

图 4　中断处理模型

4.3.1　申请逻辑中断号并与事件关联

为 CPLD 的 X_INT 注册逻辑中断，通过 *KernelIoControl*()发送 IOCTL_HAL_REQUEST_SYSINTR 命令获得与 g_MotorX_T1_Irq 即 EINT14 对应的逻辑中断号，将所获得逻辑中断号存入 g_MotorX_T1_SysIntr，程序代码如下：

```
    KernelIoControl(IOCTL_HAL_REQUEST_SYSINTR,
&g_MotorX_ T1_Irq, sizeof(UINT32), &g_MotorX_T1_SysIntr, sizeof
(UINT32), NULL)
```

同理，可得到 Y_INT 和光电编码中断 CNT_INT 的逻辑中断号。

关联逻辑中断 g_MotorX_T1_SysIntr 与 local_WaitEvent_XYM[2]事件，当 g_MotorX_T1_SysIntr 中断发生时，WinCE 将会触发 local_WaitEvent_XYM[2]事件。等待这个事件的线程将会判断中断的类型，代码如下：

```
    InterruptInitialize(g_MotorX_T1_SysIntr,local_WaitEvent_XYM
[2],0,0)
```

4.3.2　中断处理线程函数

在 *XYM_Init*()函数中通过 *CreateThread*()函数建立中断处理线程，线程函数中的 *WaitForMultipleObjects* 将线程挂起，等待要处理的所有事件，如果等待的所有事件中的任何 1 个发生，则判断是哪个事件并作相应处理。函数结构见图 5。

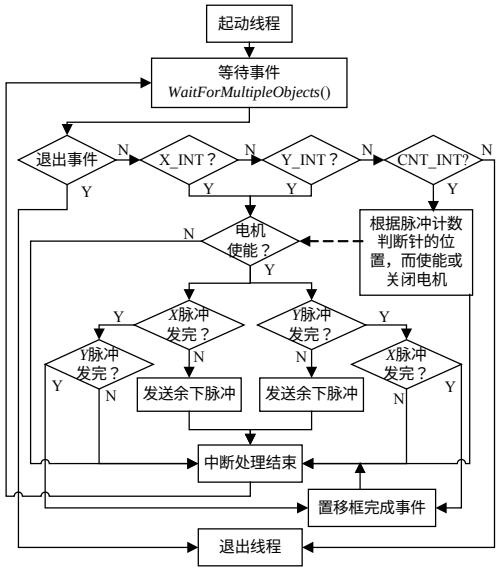

图 5　中断处理线程函数

4.4 来自应用程序命令的处理

驱动最主要功能就是处理应用程序的命令，通过 *XYM_IOControl*()函数来完成此功能，其中，*hOpenContext* 是驱动实例句柄，不同的命令通过输入参数 *dwCode* 区分，*pBufIn*, *dwLenIn* 是输入参数，最后 3 个参数是命令执行的结果。伪代码如下：

```
    XYM_IOControl( DWORD hOpenContext, DWORD dwCode,
PBYTE pBufIn, DWORD dwLenIn, PBYTE pBufOut, DWORD
dwLenOut, PDWORD pdwActualOut )
    {   判断入参有效性；
     switch (dwCode)
     { //正常刺绣处理
    case IOCTL_XYM_NORMAL_MOVE:
                获取 XY 方向要走的距离；
                //计算频率数组指针
              tmp_FreqArrayX=GetFreqArrayOf(tmp_X_Pulses)；
    tmp_FreqArrayY=GetFreqArrayOf(tmp_Y_Pulses)；
                //等待上次脉冲发送完成
    WaitForSingleObject(XY_MOVE_FINISHED, INFINITE);
              设置各个电机正反转并使能；
                起动 CPLD 发送控制脉冲；
                //脉冲发送会在 XYM_EINT_IntrThread 线程完成
          //按键拉杆移框处理
          case IOCTL_XYM_MANUAL_MOVE:
                    控制电机开始升速；
                    等待停止事件；
                    降速至停止电机；
          case IOCTL_XYM_SET_CMP_VALUE:
                    设置光电计数比较值；
        }
     }
```

当驱动完成后，编译驱动工程，会在工程的 release 目录下生成驱动的 XYMotor.dll 文件。可通过修改工程的 bib 文件

（下转第 234 页）