

在机器控制应用中使用高级原型构建硬件及软件

作者: Erik Goethert

公司: Boston Engineering

行业:

机器/机械

产品

CompactRIO、LabVIEW、用于 ADI Blackfin 的 LabVIEW 嵌入式模块、LabVIEW FPGA、LabVIEW RT

挑战:

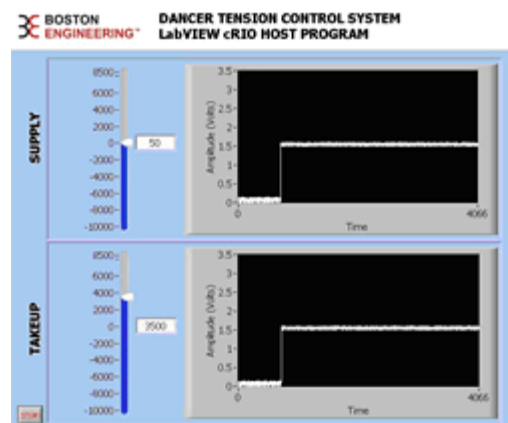
开发嵌入式微处理器系统，不仅要完成机械控制功能，还应通盘考虑整个生产过程。

解决方案:

转向使用更高级的建模工具，促使“按照规范修改”代码的生成。在这样的环境中，即使工程师们的编程经验有限，他们依然可以对开发中的软件进行评估，并且对其进行增量式、交互式的修改。

Boston Engineering 提供多种学科领域的工程服务，帮助需要使用基于微处理器的面向控制机械运动的嵌入式系统来设计电气、机械系统的公司。这就需要横跨多种传统编程语言、调试器、链接器和 IDE，同样还超越了简单的板卡原型验证。

为满足我们客户的需求，我们需要性价比很高的开发工具，用于开发基于计算机的机械、电气和嵌入式软件控制的仿真，图形用户界面（GUI）和系统功能原型验证，之后我们能有效地将它们转化成最终产品。



构建张力控制系统

我们使用多面法为数字打印亭的底片洗印构建张力控制器。在底片打印机中，由驱动电机将彩色媒体卷轴送入打印头，其中用收紧电机和进给电机控制张力。绞刀头的振动、一次照片打印数量的变化以及两个电机中任一个电机的转速变动都会影响基质的张力。开发低成本产品是这个项目的关键目标之一。

我们用两个模拟霍尔效应传感器监视测量两个活动辊位置，从而间接控制基质张力。控制系统调节两个电机的位置，使进给张力和收紧张力保持一个确定的设定值。如果张力没有被准确地控制，配准误差会导致照片色彩发生偏差。

系统仿真

机械建模需要多次重复，以便让用户更好地定义子系统挂接和容量分配。我们还根据电机尺寸、运动部件最大惯性和传感器选择等控制系统规范进一步定义了机械模型。

仿真定义了系统的开环、闭环特性。当我们确定我们需要减少活动辊纵向尺寸和基质运动距离后，我们开始了三维模型的修正循环。通过建模和反馈，很明显通过简单的 PID 控制器不能按照要求的稳定裕量提供闭环带宽。为增加稳定性，我们需要较小的 PID 增益，但是这样我们就不能得到所需要的 20 Hz

闭环带宽。

因此，我们需要更复杂的控制算法。我们使用带积分器的四阶相位超前控制器，用 **bi-quad** 形式实现。我们选择这种实现方式以降低由定长的 **16** 位字长及其内在的算术舍入产生的误差，这些误差会导致不稳定。

当我们创建带有用户界面的系统时，我们通常使用 **NI LabVIEW** 进行用户界面原型构建。我们将这个图形化工具用于设计过程的后期原型构建和部署阶段。我们使用 **NI LabVIEW** 可以方便快速地将界面元素拖动至面板，从而使客户能够自定义界面的布置、内容和操作。这减少了界面调整的重复次数，还减少了最终代码的变动机会。我们发现这种方式还能够用于嵌入式系统中，尽管其中通常没有真正的用户界面。我们使用原型构建界面辅助系统调节，以达到最大稳定性。

系统原型构建

设计阶段的下一步是为系统构建物理原型。原型构建完成多个重要功能。最重要的是，原型构建确保我们的设计者能够完全理解问题。通过将仿真的操作和原型的操作相比较，我们可以观察到差异，并分析出差异产生的原因。有时候，它们是我们可能直到原型准备运行时才发现的高阶效应。

我们通常使用现成的开发板，一般带有可编程 **DSP**、**FPGA** 或微处理器，在原型构建阶段可以重复配置。但是这些板卡通常不带有适合严格时间控制的 **I/O**，或 **I/O** 对应的信号调理。甚至即便使用了可编程元件，例如 **FPGA**，我们还碰到了许多在回归式开发过程中引发问题的因素，包括使用专利语言、缺少技术文档和技术支持、控制器缺乏灵活性、**I/O** 端口数量不足以及我们不能轻易修改的封闭控制器算法。

在这个项目中，我们取而代之使用了带有控制器的 **LabVIEW FPGA** 快速原型构建硬件平台，以及通过模块化 **I/O** 背板上的 **FPGA** 连接到控制器的模块化 **I/O** 系统。我们还使用美国国家仪器公司的 **CompactRIO** 原型构建系统，它带有四槽或八槽的背板，可以接受数字 **I/O**、模拟 **I/O** 或通信总线模块。

在我们的设计中，张力控制硬件需要两个脉冲宽度调制器输出来控制两个电机；两个编码器提供两个电机的速度反馈；两个模拟输入通道供霍尔效应传感器探测活动辊位置；两条数字线传送信号以及供温度和空气读数使用的通道。

我们使用定制的信号调理电路，将这些模块与包含两个控制器的 **NI CompactRIO** 模块上的硬件相连。第一个控制器是运行于 **266 MHz** 的嵌入式微处理器，它与以太网控制器和固态磁盘驱动器相连。第二个控制器是位于机箱背板的 **1M** 门电路 **FPGA**，用于连接 **I/O** 模块和嵌入式微处理器。

我们利用 **LabVIEW** 图形化数据流语言不仅进行模块化嵌入式微处理器编程，还用于管理背板中的 **FPGA**。由于代码是比 **C** 语言更加高级的语言，我们的控制、机械和电气工程师们可以直接在 **MCU** 中的代码上工作。

我们选择在嵌入式控制器上运行监督程序，在 **FPGA** 上运行电机控制算法，以便在系统原型和最终系统编程模型之间提供最大的相似性。为在 **FPGA** 上运行控制算法，我们将零极点增益模型转换为类似于 **LabVIEW** 数字滤波器设计工具包中的滤波器。我们方便地利用工具包将滤波器转换为能够在 **FPGA** 上运行的代码。由于在 **FPGA** 上，浮点算术十分消耗资源，我们使用工具包自动生成定点运算的代码。之后，我们测试量化选项使得最终的滤波器能够稳定。

在系统界面设计和底层硬件中使用了 **LabVIEW** 为我们带来许多好处。每一个 **LabVIEW** 函数包含一个

表示代码的程序框图和一个带有输入控件和显示控件的 GUI。在交叉目标中，GUI 在设计系统中以窗口形式出现，可用于监视系统的内部状态或调整程序参数。利用运行于嵌入式微处理器的代码前面板在运行中调整系统，这使我们受益匪浅。

使用开放式、高生产效率的 LabVIEW 图形化系统设计平台，我们可以在设计到原型构建直至部署阶段，节省大量时间。在我们行业中，节省时间就意味着节省成本，LabVIEW 嵌入式技术使我们对客户而言更有价值，在市场中更有竞争力。要完成复杂的嵌入式运动控制系统，我们使用标准设计工具，并把它们在 LabVIEW 中与实际数据的仿真整合在一起，从而优化设计。

与建模、设计、测试和目标的众多工具和开发环境相比，LabVIEW 嵌入式模块，现在是用于 ADI Backfin 处理器的 NI LabVIEW 嵌入式模块为我们提供了一套整合工具链，使我们把时间花费在工程上，而不再是语法上。

要获取更多信息，请联系：

Erik Goethert, egoethert@boston-engineering.com