ANALOG DEVICES
超越一切可能™

开源、可重复使用的软件堆栈助力实现实时处理和CbM算法开发

Travis Collins，高级算法工程师

CN0549状态监控平台简介

在本文中，我们将重点介绍CN0549的不同组件可用的软件生态系统、数据分析工具和软件集成，以及工程师和数据专家如何使用它们进行应用开发。我们分两部分来介绍使用CN0549开发平台进行状态监控(CbM)和预测性维护(PdM)应用，这是该系列文章的第二篇。新平台旨在加快定制CbM解决方案从原型制作到生产的整个开发流程。第一部分主要介绍MEMS振动技术，以及为CbM应用捕捉高质量的振动数据。

从概念到生产的整个过程，以及如何加快这个过程！

在构建状态监控解决方案时，它们必须包含传感器、本地处理、连接、某些形式的软件或硬件，使其能够正常运行。CN0549提供可自定义的硬件和软件选项，让工程师和软件开发人员能够使用常用的工具和基础设施，并根据应用设计做出权衡取舍，以解决所有这些挑战。例如，如果您想选择特定的微控制器或FPGA进行处理，想要使用Python进行编码，或有喜欢的、想要重复使用的传感器。这让CN0549成为一个强大的平台，让希望构建优化CbM解决方案的人员能够根据自己的需求来自定义处理、功率、性能、软件和数据分析。

嵌入式系统的开发流程

我们来看看嵌入式系统从生成概念到生产的整个开发流程。图1概要描述这个抽象化的过程。

在图1所示的设计流程中，第1步是“数据研究”阶段。在这个阶段，用户将他们的要求转化到应用对硬件和软件的不同要求。从硬件的角度来看，可能涉及抗冲击性、模拟信号带宽或测量范围等参数。在考虑对软件的要求时，样本数量、采样速率、频谱、过采样和数字滤波都是CbM应用的重要参数。该平台非常实用灵活，允许研究人员使用不同的传感器组合，并调节数据采集参数，以满足其应用需求。

“数据研究”阶段之后是“算法开发”阶段，这个阶段主要是验证系统的应用或使用。这通常需要在高级工具中开发模型或设计算法，并最终移植到嵌入式系统中。但是，在优化设计之前，必须使用真实数据和硬件环路进行验证，这正是CN0549的优势所在，因为它不仅能与热门的高级分析工具直接集成，还支持硬件环路验证。

设计得到验证之后，就开始进行优化和嵌入所需软件组件的工作。在“嵌入式设计细化”阶段，可能需要重新实施某些算法或软件层，以便在FPGA或资源有限的微控制器中使用。必须小心谨慎地不断验证设计，因为我们会将它移植到原型或将要投入生产的硬件中进行最后验证。

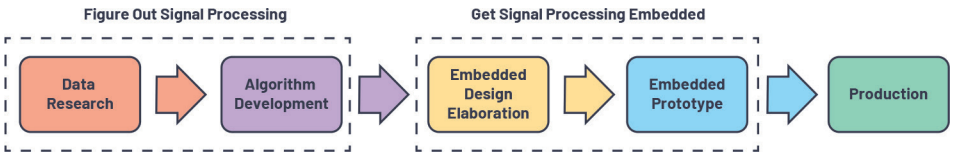

图1. 嵌入式系统的开发流程。

最后是到达“生产”阶段，这个阶段可能与设计开始使用的原始开发环境没有什么相似之处，但仍然要满足同样的要求。由于最终的系统可能与原始的研究系统相去甚远，所以可能无法或很难运行相同的代码或测试。这可能导致产生生产测试问题和设备故障，很可能需要花费额外的时间和资金投入来进行补救。

通过最大程度的重复使用来降低风险

在设计过程中，降低风险最简单的方法之一是尽可能在每个阶段重复使用更多的硬件和软件组件，CN0549为开发人员提供许多开箱即用的资源，可以在开发流程的每个阶段直接使用。CN0549解决方案提供原理图和电路板布局文件，提供一个适用于优化和全功能环境的开源软件堆栈，以及更高等级工具（例如MATLAB®和Python）可用的集成选项。最终用户可以使用ADI经过验证的组件，并在研究阶段到生产阶段期间选择想要维护或更改的组件。这样最终用户就能集中精力进行软件开发和系统集成，不必去绘制ADI组件的原理图或进行基础的软件开发。利用硬件模块和重复使用软件层，例如ADI提供的设备驱动程序、HDL或应用固件，可以减少构建系统所需的开发时间，并大大加快上市时间。

软件开发流程和过程

在开发期间，CN0549为工程师们提供多种选项，允许他们使用通用语言，包括C或C++，同时使用他们熟悉的数据分析工具，例如MATLAB或Python。这主要是通过利用和基于开源标准，以及支持不同制造商的多种嵌入式平台的现有解决方案进行构建而实现。

CN0549系统堆栈

图2所示的系统堆栈概述了构成CN0549系统的不同组件。左上角的深蓝色方框表示传感器和数据采集(DAQ)电路板，浅蓝色和紫色方框表示用于数据处理的FPGA分区。该平台直接支持Intel DE10-Nano和Xilinx® CoraZ7-07s，涵盖两大FPGA供应商。绿色方框表示与主机PC的连接。这为算法开发提供了从硬件到高级数据分析工具的直接数据访问。

所有硬件描述语言(HDL)代码都是开源的，允许开发人员进行修改，将数字信号处理(DSP)插入可编程逻辑(PL)的数据流中，如图2所示。这可以是从滤波器到状态机甚至机器学习等任何内容，具体由您的系统分区决定，这一步也可以在用户空间或应用层完成。由于代码是公开提供的，它可以移植到不同制造商的其他FPGA，或不同处理器系列中，具体取决于终端应用的需求。

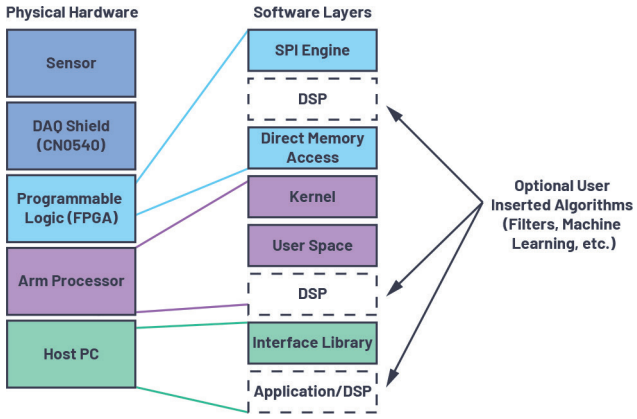

图2. CN0549平台的系统堆栈。

Arm®处理器内部提供两种软件选项。具体使用哪种，由具体的用例决定，大多数开发人员可能会使用：

- Linux®：内核驱动程序，可用于在内核中的输入输出工业(IIO)框架中构建的DAQ屏蔽。它与一个名为Kuiper Linux的完全嵌入式Linux发行版相结合，这个版本在Arm内核用户空间中运行，基于树莓派OS。
- 无操作系统(No-OS)：裸机项目，使用与Linux内核中同样的驱动程序，可以在Xilinx或Intel的SDK中使用。它也可以作为替代方案，在实时操作系统(RTOS)环境中实现。

建议开发人员从Linux开始学习并使用其系统进行开发，因为Linux提供的工具最多。Linux还提供大量开发包和驱动程序，构成了所需的开发环境。在系统设计稳定并准备进行优化时，通常会转向无操作系统环境，只提供必要的软件。但是，这主要取决于应用，许多制造商会交付完整的Linux系统，以保持他们要提供的灵活性。

与用于可编程逻辑的HDL一样，整个内核源代码、Kuiper Linux镜像和No-OS项目都是完全开源的，让最终用户能够按照自己的意愿更改组件。如果需要，还可以将这些代码库移植到不同的处理器系统或不同的运行时环境中。

图2所示的最后一个组件是与主机PC的连接，如绿色方框所示。在运行该系统时，可以对设备进行配置，并将数据流备份到主机系统进行分析，开发人员将利用MATLAB或TensorFlow等标准工具在主机上创建算法。最终将这些算法转移到嵌入目标中，让他们能够使用本地处理能力来加快算法开发迭代。

访问CbM数据——使用入门

使用Arm处理器和PL一般发生在设计流程较为靠后的阶段，也就是要对系统实施优化进行部署时。所以，对于开发人员来说，最开始常用的切入点都是从工作站远程连接至嵌入式系统。在嵌入式系统上运行Linux时，因为基础设施的设计方式，在工作站上远程或本地运行代码是一个相对透明的过程。这主要是因为名为libIIO的开源库。libIIO是一个接口库，允许在内核的Linux IIO框架内构建适用于不同设备驱动程序的简单、一致的访问模型。这个库是能够灵活使用CbM平台的核心，并提供数据流传输和设备控制功能。

libIIO本身主要分成两个部分：

- libIIO库，这是一个C语言库，用于访问不同的IIO驱动程序属性或函数。这包括向设备（例如ADC、DAC和传感器）传输数据流或从中输出数据流。
- IIO daemon(iiod)利用实际的驱动程序的库和内核接口来管理libIIO库或客户端之间的访问。

libIIO和iiod本身是从不同的组件写入，可以使用不同的方法来访问驱动程序，即所谓的后端。后端允许本地和远程用户对libIIO进行控制和提供数据流，而且，由于它们已形成组件，所以可以将新后端添加到系统中。目前，libIIO支持四个后端：

- 本地：允许访问连接至同一设备的硬件的本地可访问驱动程序。
- USB：通过使用libusb，此后端允许通过USB链接远程控制驱动程序。
- 串行：为通过串行连接的电路板提供更通用的接口。UART是最常见的用例。
- 网络：最常用的远程后端，基于IP来访问网络中的驱动程序。

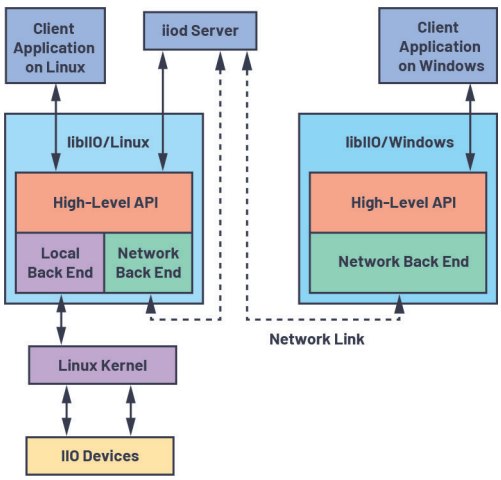

图3. 使用网络后端的libIIO系统概述。

图3从系统层面概述如何使用libIIO组件，以及如何将它们集成到整个系统中。图中左侧是嵌入式系统，它已安装libIIO库，运行iiod daemon。在嵌入式系统中，用户可以访问本地后端，甚至网络后端。他们可以通过更改一行代码来确认任一后端的地址，在两个后端之间切换。无需对目标代码进行其他更改。

图3左侧显示的是远程主机，可以运行任何操作系统。提供Windows、macOS、Linux和BSD等官方软件包。该图显示使用了基于网络或IP的后端，也可能是使用串行、USB或PCIe连接。从用户的角度来看，可以从C语言库本身，或者从其他语言的许多可用绑定来使用libIIO，包括：Python、C#、Rust、MATLAB和Node.js。为需要与应用中的不同驱动程序交互的用户提供多种选择。

应用和工具

当开始使用一个新设备时，通常不建议直接使用libIIO。所以，有很多基于libIIO构建的更高等级的应用，它们通过命令行和GUI格式为IIO设备提供基本的配置能力。它们分别是IIO工具和IIO示波器。

IIO工具是一组与libIIO一起发布的命令行工具，对于通过脚本执行的低等级调试和自动化任务来说非常有用。例如，在执行实验室测试时，它可以在不同的采样率模式下设置平台，以及收集一些数据。利用几行bash，或通过使用IIO工具的批处理脚本可轻松完成这些操作。图5显示了一个简单示例，可以在本地或远程运行，以更改采样速率和ADC的地输入模式。这个示例使用名为iio_attr的IIO工具，让用户能够轻松更新设备的配置。

```
C:\>iio_attr -u ip:analog.local -d ad7768-1 sampling_frequency 128000
128000

C:\>iio_attr -u ip:analog.local -d ad7768-1 common_mode_voltage 1V9
1V9
```

图5. IIO工具的iio_attr部分的使用示例。

但是，对用户来说，最常见的切入点是GUI应用IIO示波器，一般被称为OSC。与IIO工具一样，OSC是通用的，可以管理任意IIO驱动程序，而且，因为它是基于libIIO构建，所以它可以远程运行或在电路板上运行。但是，它也包含一个插件系统，可以为特定的驱动程序或驱动程序组合添加专用选项卡。图6显示自动加载到基于CN0540的电路板上的插件选项卡，包括控制和监控选项卡。这些选项卡提供了一个简单的界面，可以访问CN0540的ADC、DAC和控制引脚的低级功能，以及数据采集板和测试点监控的基本示意图。如需了解其他可用的默认选项卡和插件信息，可以访问ADI公司Wiki查看更多OSC文档。

OSC的最后一个重要方面是捕获窗口。捕获窗口可以根据从ADC或基于libIIO的缓冲区收集的数据进行绘图。图7显示在频域模式下使用的捕获窗口，这是基于频谱数据信息绘制。也可以绘制其他图，包括时域图、相关图和星座图。这对于抽检设备、调试或评估非常有用。这些图提供常用工具，例如标记、峰值检测、谐波检测，甚至相位估计。由于OSC也是开源的，任何人都可以添加更多插件或绘图，甚至更改现有功能，对其进行扩展。

```
# Connect To IP context
ctx = iio.Context("ip:192.168.2.1")
# Get ADC
adc = ctx.find_device("ad7768-1")
# Set sample rate
adc.attr['sampling_frequency'].value = '128000'
```

```
# Connect To local context
ctx = iio.Context("local:")
# Get ADC
adc = ctx.find_device("ad7768-1")
# Set sample rate
adc.attr['sampling_frequency'].value = '128000'
```

图4. libIIO远程与本地示例。

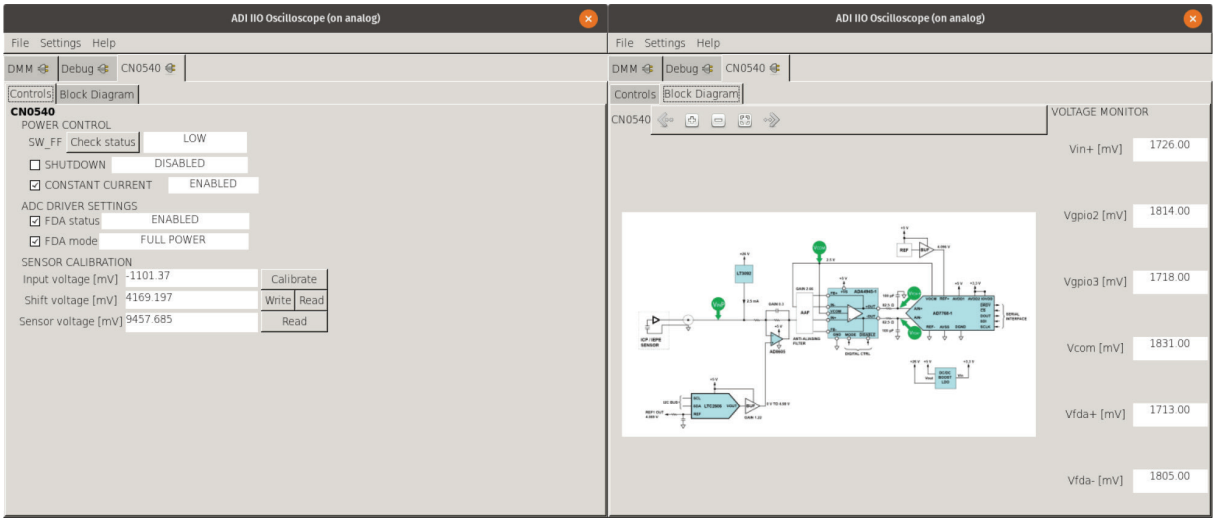

图6. CN0540 IIO示波器插件选项卡。

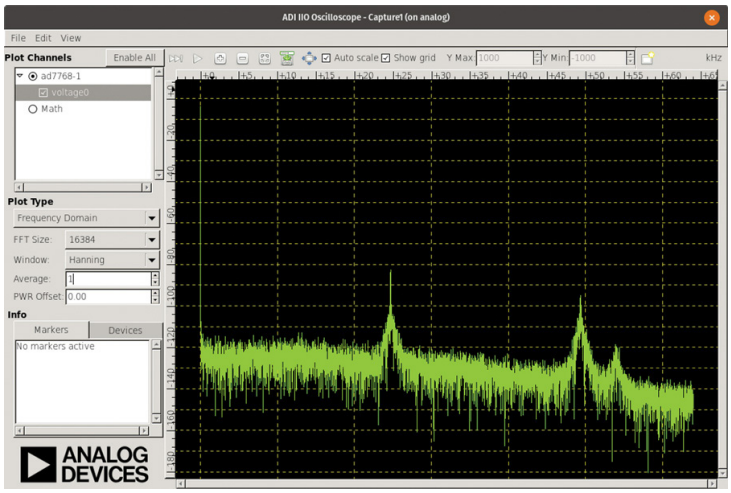

图7. 频域模式下的IIO示波器捕获窗口。

算法开发环境集成

至此，我们已经介绍了大部分工程师在首次使用CN0549时会采用的低等级重要工具。首先理解这些是很重要的，这样开发人员才能理解系统的灵活性以及他们可以使用的不同选择或接口。但是，在设置和运行基线系统后，开发人员希望使用MATLAB或Python等工具将数据快速迁移至算法开发。这些程序可以从硬件导入数据。必要时可以设计附加控制逻辑。

在机器学习开发周期中，开发人员通常会遵循通用的流程，该流程与他们想要用于处理数据的软件环境无关。图8简要显示了该流程的一个示例，其中涉及数据收集、分割数据用于测试和训练、开发模型和算法，最后部署模型进行现场推理。在实际服务中，会持续执行这整个流程，将新学习内容集成到生产模型中。TensorFlow、PyTorch，或MATLAB Machine Learning Toolbox等工具都可以采用此流程。这个流程有其作用，但是，通常会忽视或完全忽略收集和整理数据，以及管理数据这种复杂任务。为了简化这项任务，我们使用这些相关工具和软件包设计出相关的软件生态系统。

Python集成——连接到Python分析工具

首先，从Python开始，可以通过模块PyADI-IIO获得CN0549的设备特定类别。图6显示了一个通过以太网配置设备的采样速率和提取缓冲区的简单示例。这里没有复杂的寄存器序列、模糊的存储器控制调用，或要记忆的随机位。而是由板上运行的驱动程序、libIIO和PyADI-IIO在工作站，甚至在云中进行远程管理。

PyADI-IIO可以通过pip和conda进行安装，将控制按钮表现为易于使用和归档记录的属性。它还按易于理解的形式（例如NumPy阵列或原生形式）提供数据，在必要时，还会处理设备的数据流转换。这使PyADI-IIO易于添加到Jupyter Notebook之类的环境中，无需通过不同的工具或复杂的数据转换即可轻松将数据传输到机器学习管道中，让开发人员可以集中精力开发算法，而不是处理某些困难的API或数据转换。

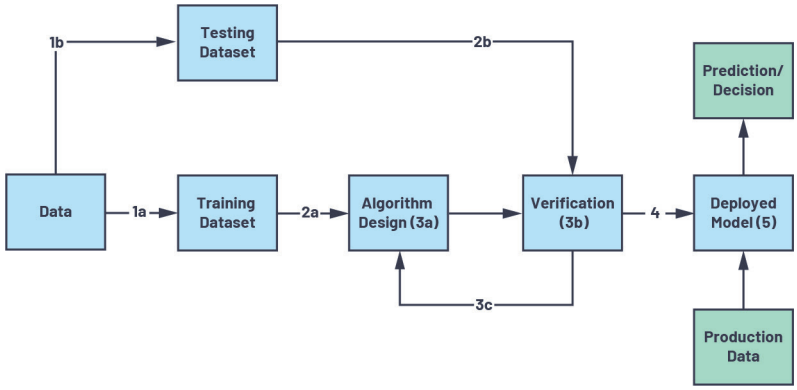

图8. 机器学习模型开发流程。

```
import adi # Import module
xl = adi.cn0532("ip:192.168.2.1") # Create object
xl.sample_rate = 12800 # Update sample rate
data = xl.rx() # Collect data
```

图9. PyADI-IIO示例。

MATLAB集成——连接到MATLAB

在MATLAB方面，通过Analog Devices Sensor Toolbox提供对CN0549及其组件的支持。这个工具箱与PyADI-IIO类似，提供针对不同组件的特性类别，将它们实施为MATLAB系统对象(MSO)。MSO是MathWorks开发人员可以用来连接硬件和不同软件组件的一种标准化方式，提供先进功能，帮助执行代码生成、Simulink支持和一般状态管理。许多MATLAB用户能够在不了解的情况下，使用实施为MSO的MATLAB的各种功能，例如示波器或信号生成器。在图10中，我们使用CN0532接口和DSP频谱分析仪示波器，两者都实施为MSO。同样，和PyADI-IIO一样，提供一个易于使用的接口供传统的MATLAB用户使用。

除了硬件连接之外，Sensor Toolbox还集成适用于HDL和C/C++的代码生成工具。这些工具适用于开发、模拟和部署IP，甚至不熟悉HDL设计或工具，但了解MATLAB和Simulink的人员也可以使用。

```
%% Configure object interface
xl = adi.CN0532('uri','ip:192.168.2.1');
xl.SampleRate = '1024';

%% Create scope
sa = dsp.SpectrumAnalyzer('PlotAsTwoSidedSpectrum',false);
for k=1:10
    sa(xl());
end
```

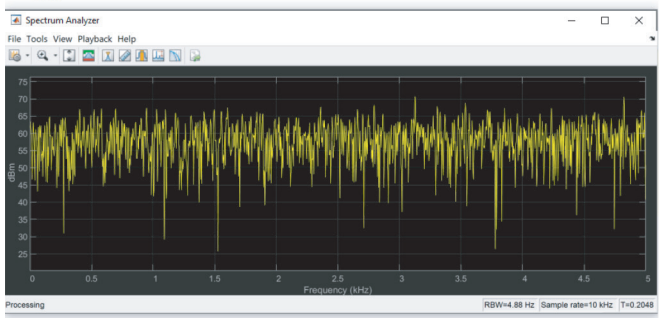

图10. 使用示波器的Sensor Toolbox流传输示例。

使用TensorFlow的分类示例

CN0549套件提供几个示例，从基本数据流传输到机器学习分类示例。关于时间序列数据的机器学习，例如来自CN0532的振动数据，可以从几个不同角度进行理解。这可能包括支持向量机(SVM)、长短时记忆网络(LSTM)模型，如果将数据直接解译为时间序列的话，甚至包括自动编码器。但是，在许多情况下，将时间序列问题转换为成像处理问题，并利用在该应用领域开发的工具和丰富知识可能更为方便。

我们在Python中看看这种方法。在随PyADI-IIO提供的一个示例中，将CN0532安装到振荡风扇上，然后进行了一些测量。这些测量在不同的风扇设置（Sleep、General、Allergen）下进行，在每种模式下都会捕捉409,600个样本。在图11中查看这个数据时，可以轻松确定Allergen用例的时域，但其他两个用例则比较难以区分。虽然可以通过检测来确认这些用例，但在时域中使用算法来确认这些用例会很容易出错。

为了帮助更好地区分这些用例，会将数据转化为频域，并使用频谱图来描绘不同频率随时间变化的浓度。与图11相比，图12所示的频谱图在数据上有更明显的差异，但在时间维度上是一致的。这些频谱图是有效的图像，现在可以使用传统的图像分类技术进行处理。

将数据集拆分为训练集和测试集，将频谱图分别输入仅由神经网络(NN)构成的模型（包含三个致密层）和更小一些的卷积神经网络(CNN)模型。这两种方法都是在TensorFlow中实现的，可以在不到100次的周期内轻松收敛到接近100%测试验证。CNN使用大约1%的可调参数在大约一半的时间内收敛，是目前最高效的设计。图13提供关于精度和周期的培训收敛图，以概述CNN的快速收敛。

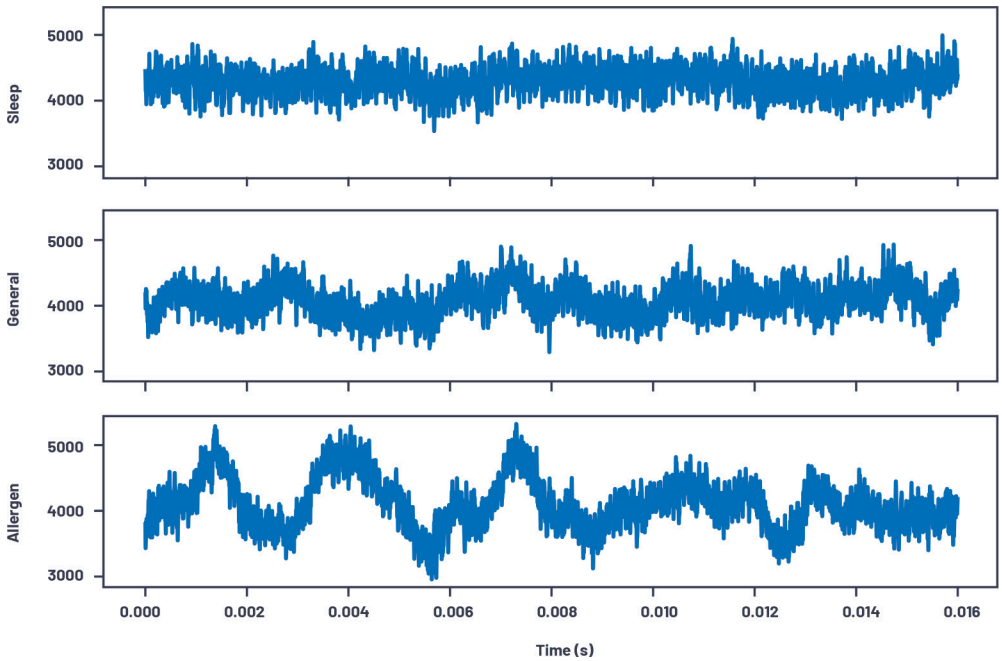

图11. 时间序列中的风扇振荡数据。

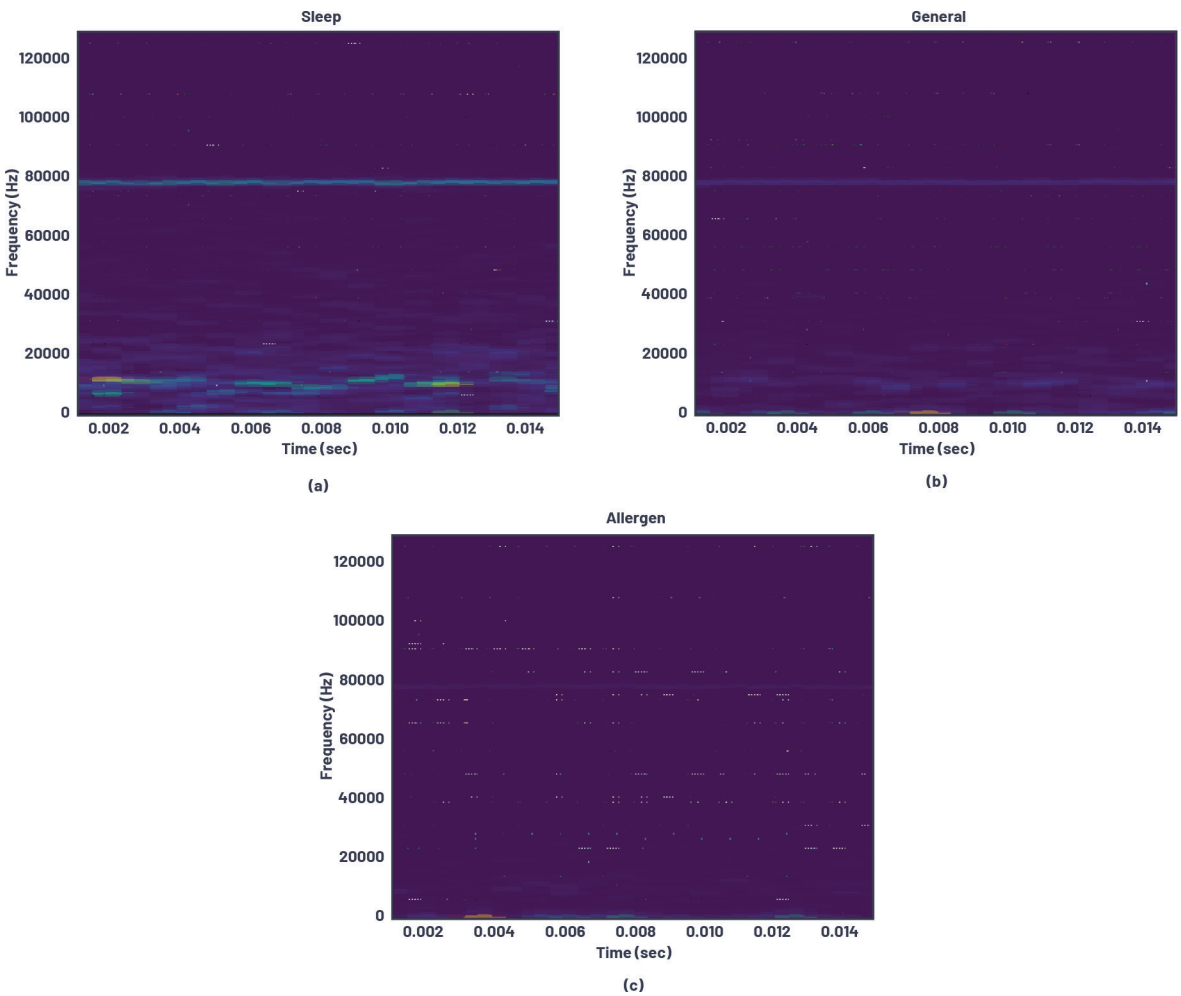

图12. 捕捉的振动数据的频谱图。

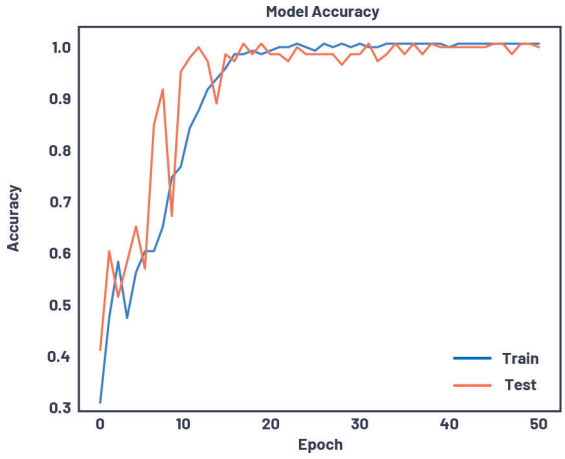

图13. 随时间变化的CNN训练精度（用于绘制振动频谱图）。

在GitHub的PyADI-IIO源代码树下提供了此示例的所有Python脚本、手册和数据集。由于提供了数据集，甚至可以在不使用CN0549硬件的情况下使用TensorFlow来展示示例。但是，使用硬件时，可以将训练模型用于实时推理。

边缘到云：转向嵌入式解决方案

创建模型后，可将其部署用于推理或决策。采用CN0549时，它可以安装在远程PC上，从CN0540传输数据流，或是直接在嵌入式处理器上运行。根据实施方案，将模型放到处理器中需要更多的工程工作，但可以将功效提高一个数量级，且能够实时运行。幸运的是，在过去几年里，用于部署机器学习模型的工具和软件都取得了很大的发展。

使用FPGA

赛灵思公司和英特尔都提供高阶合成(HLS)工具，将高阶语言转化成在FPGA上运行的HDL代码。它们通常会与TensorFlow、PyTorch或Caffe等Python框架集成，以帮助将模型转换为IP内核，从而允许工程师将IP部署到DE10-Nano、Cora Z7-07S或自定义系统上。然后，可以将这些IP内核集成到ADI提供的开源HDL参考设计中。图14显示Vivado提供的Cora Z7-07S CN0540的屏幕截图，其中包含注释，其中侧重显示数据路径。在该设计中，来自CN0540的数据通过SPI引脚读取，24位样本由SPI引擎解译，传输到DMA控制器，再进入存储器。任何DSP或机器学习模型都可以直接插入数据路径中这个管道。

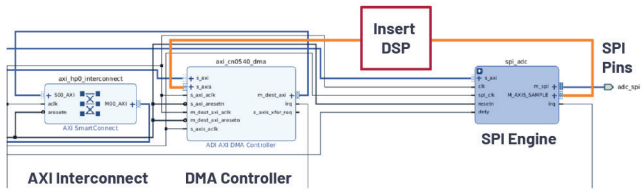

图14. Vivado（2019年1月）显示的Cora Z7-07S HDL参考设计数据路径。

使用微处理器

它们无需将算法转化为HDL层，而是可以直接在Arm内核中运行。根据数据速率和算法的复杂性，这个开发流程很合理，也更加简单。相比HDL，为Arm内核开发C代码甚至Python所耗费的开发资源和时间都更少，通常也更易于维护。

MATLAB Embedded Coder这样的工具甚至可以简化此流程，自动将MATLAB转化为可嵌入且优化的C代码，供Arm内核使用。或者，TensorFlow提供TensorFlow Lite等工具，它们是Python代码库的可嵌入的C版本，能够更轻松地转换为嵌入式目标。

智能决策拓扑

状态监控并非适用于所有硬件和软件配置，所以CN0549采用了灵活的设计。我们在考虑CbM异常检测之类的问题时，通常可以从两个时间量程角度来解决：在一个时间量程，我们需要立即做出反应，例如在安全相关的场景中，在长期时间量程，更多的是关于维护或设备更换。两者需要使用不同的算法、处理能力和方法。

在理想情况下，机器操作员将会拥有很大的数据湖来训练模型，可以无需干扰事件来处理短期检测，也可以持续从运行设备传输数据流，以便进行未来的维护预测。但是，对大多数操作员来说，情况并非如此，数据湖严重干涸。由于安全考量、地理位置、网络或拓扑等要求，有些现成的解决方案也很难执行数据收集。面对这些困难，我们需要自定义程度更高的解决方案。

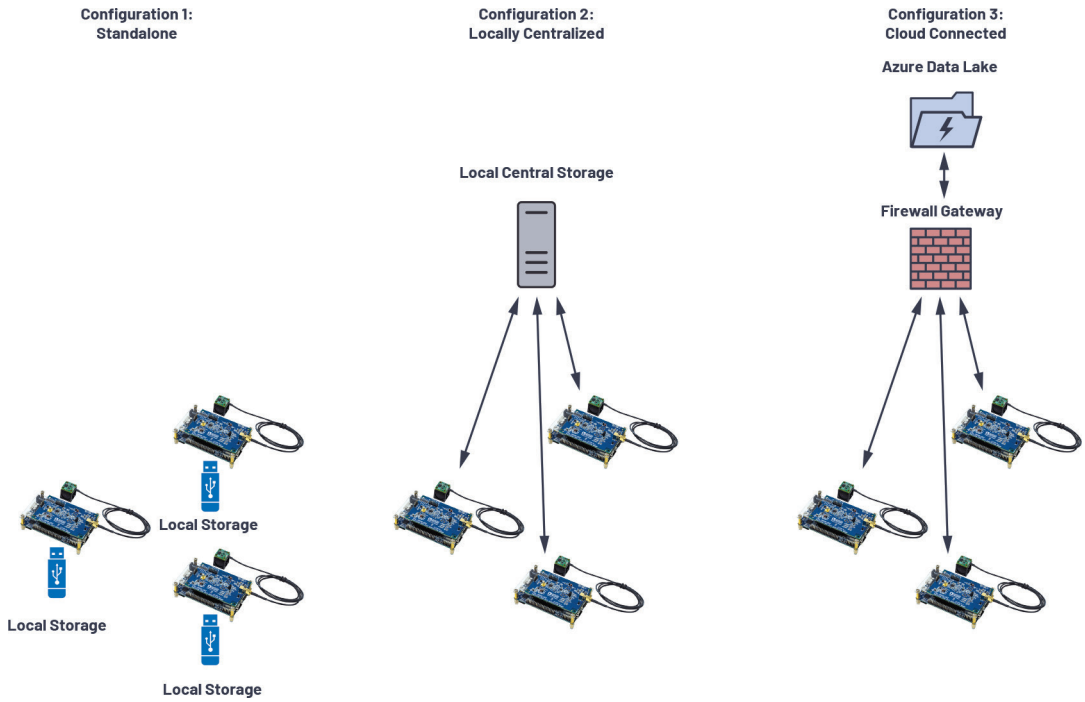

图15. CbM网络拓扑。

CN0549是一个独立系统，提供多种连接选项。它运行标准的Linux，所以传统的网络堆栈（例如以太网和Wi-Fi）可以开箱即用，甚至能在必要时连接蜂窝调制解调器。在实际应用中，可以使用几种出色的典型拓扑，如图15所示。

图15最左侧的配置是脱机收集数据示例，一般发生在偏远位置或无法联网的地方。在这种情况下，平台会配备大型存储媒介，并按照计划来收集数据。或者，其他两个选项是将数据流传输至同一个端点。图15中间的配置是隔离网络，可能仅供组织内部使用，或者是偏远位置的一组用于集中收集数据的平台。出于安全考虑，或者在无法联网时，可能需要这种配置。在这些配置下，CN0549易于设置，且能够根据终端部署的特定需求来自定义。

最后一个配置是直接云选项，每个平台直接访问互联网，并将测量数据推送至云。CN0549在Linux上运行，所以该平台可以通过Python等语言轻松使用不同的云供应商（例如Microsoft Azure IoT或Amazon IoT Greengrass）的API，提供一种为新连接的设备构建数据湖的简单方式。

云和本地流程之间保持稳定连接时，如我们之前所探讨的那样，可以对不同算法进行划分，哪些是需要或可以在本地运行的，哪些是可以在云中运行的。然后自然地针对算法复杂性处理能力、事件延迟和云传输带宽限制等的要求进行权衡和取舍。但是，由于非常灵活，因此这些因素很容易考虑决断。

结论

CN0549 CbM平台为设计人员开发应用提供了系统灵活性和大量软件资源。本文深入探讨软件堆栈，并围绕如何使用不同组件来实施CbM和预测性维护(PdM)开发展开讨论。由于软件、HDL、原理图以及与数据科学工具集成的开放性，设计人员可以在整个堆栈中充分利用其终端系统所需的组件。总之，这种状态监控设计提供了一款易于使用的开箱即用解决方案，包括开源软件和硬件，以提供灵活性，让设计人员能够在更短时间内实现更好的自定义设计。

作者简介

Travis Collins拥有伍斯特理工学院电气和计算机工程博士学位和硕士学位。他的研究侧重于小型蜂窝参考建模、相控阵测向和软件定义无线电的高性能计算。他目前就职于ADI公司的系统开发部，主要负责通信、雷达和通用信号处理应用。联系方式：travis.collins@analog.com。

在线支持社区

ADI EngineerZone™
中文技术论坛

访问ADI在线支持社区，与ADI技术专家互动。提出您的棘手设计问题、浏览常见问题解答，或参与讨论。

请访问ez.analog.com/cn

ANALOG DEVICES
超越一切可能™

TA23098sc-8/21

请访问analog.com/cn