WP502 (v1.0) 2018 年 6 月 15 日

Python 生产力价值：赛灵思 Zynq 产品系列的前沿优势分析

作者：Giulio Corradi 博士

赛灵思® PYNQ 框架能在 Zynq® 产品系列中实现对 Python 语言及运行时的全面支持与集成。直接在 Zynq SoC 架构上利用 Python 的生产力优势，用户能够充分发挥可编程逻辑和微处理器的长处，更容易为人工智能、机器学习和信息技术应用构建设计。

摘要

从工程设计、科研、数据科学、机器学习、信息技术到人工智能，Python 开源编程语言已经成为各类应用中的不成文标准。

当在嵌入式应用中使用现代片上系统 (SoC) 时，就能够运行 Python 执行复杂的分析算法，其性能接近台式机工作站，但外形尺寸显著缩小，功耗要求也显著降低。通过预处理从传感器读取的数据，赛灵思® Zynq 产品系列大幅度提高性能和确定性，同时降低时延。

这种被称为 PYNQ 框架方案，能从应用处理器有效卸载不必要占用处理器带宽的大量重要但重复的操作。这种卸载功能对于满足工业物联网中边缘应用提高的智能需求有重要意义。

嵌入式计算的新范例

近期的 IEEE 调查报告称 2017 年最流行的两种编程语言分别是 Python 和 C 语言。在嵌入式计算领域，C 语言一直以来都是中坚力量。传统上来说，我们一直将 Python 语言用于网络或台式机计算，而从未用作嵌入式计算语言；但是这种情况正在发生改变。

Python 及其相关框架能支持用于数据分析、机器学习（ML）和人工智能（AI）应用的复杂算法的开发。当然，这些应用属于嵌入式计算领域的热点话题，而且它们正在促使 Python 得到采用，特别是在边缘工业物联网 (IIoT) 领域的普及。

C、C++ 和 Python 紧密相连，因为 Python 本身也依赖 C 和 C++ 用来提供最核心的库。但是，C 和 C++ 属于编译型语言，能够在裸机上执行。Python 在这点上则与之不同，是一种解释型语言。这种差异在嵌入式计算中为自身带来了挑战：例如，Python 需要操作系统（一般是 Linux），另外还需要易失性和非易失性存储器资源。

在工业物联网边缘嵌入式计算领域，ML 和 AI 的实现日趋倾向于发挥数字孪生体 [1] 与物理致动器的功用。因此，解决方案必须能够实时以低确定性时延做出响应。此外，工业物联网解决方案也必须能够支持其他行业趋势，例如：

- 根据待解决问题在实时处理器、应用处理器和专用处理单元间进行分区
- 为专用处理器卸载引擎创建接口，从而为性能关键内核提速
- 使用 Linux 等标准操作系统
- 提供具备调度功能和确定性的解决方案
- 为原型设计和生产提供高生产力框架
- 覆盖标准和传统网络通信接口与协议，包括 IT、OT 融合网络。
- 为机器学习和分析提供丰富的库
- 功能安全性
- 网络安全

构建工业物联网平台绝非易事。从物理环境的边缘到云（包括 AI 和 ML）的整个链条是复杂的，并需要多种专业能力。因此，开发工业物联网就要求使用更高水平的抽象，而这种抽象水平又要与项目涉及的不同工作职能相关联，才能让开发在可接受的时间预算和成本预算内完成。参见图 1。

1. **数字孪生**是真实资产、流程和系统的数字副本，用于为真实行为建模，提供监控、预测和诊断功能。

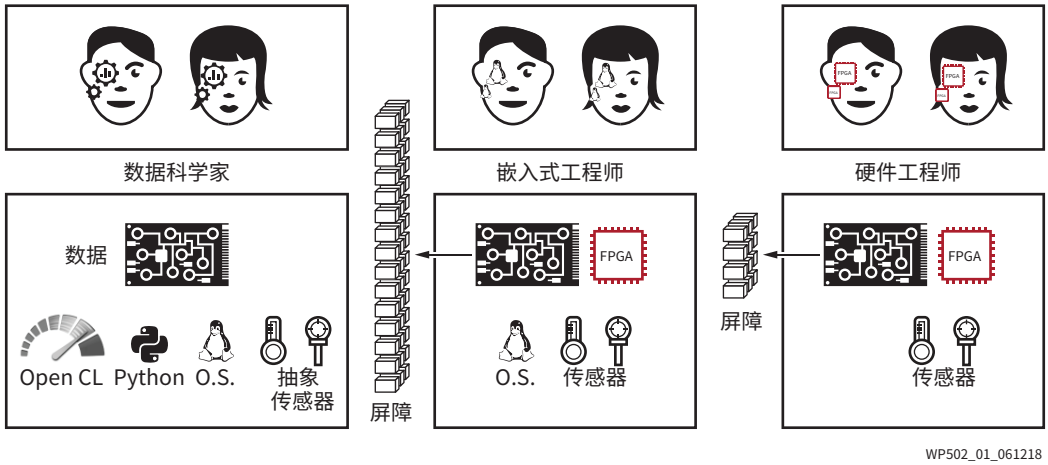

图 1： **同一平台的不同抽象水平**

在工业物联网中探索机器学习

工业物联网解决方案越来越多地在边缘纳入嵌入式智能。对于众多应用而言，这意味着机器学习推断的实现。实现后，ML 算法会利用其经验，根据一套输入数据得出结论。在 ML 中，经验可通过名为培训的学习过程来获得。ML 应用的培训可使用下列两种方法之一执行：(1) 人工监督或 (2) 实现判断功能。两种方法都需要将由正反例构成的大数据集应用于 ML 网络。在 ML 算法得到充分培训后，就能将其部署在工业物联网边缘，根据新输入和未知输入进行推断。

工业物联网解决方案应用从 ML 的使用中获益匪浅，尤其是对于那些传统方法不能提供可接受的性能的应用或者需要大量人工干预的应用，例如再校准、维护、诊断和故障安全保护操作。这些应用具体包括：

- 传感器与测量系统
- 系统识别
- 机器学习系统控制
- 高级信号处理
- 自主系统的强化学习
- 图像处理

因此，开发 ML 解决方案需要具备有足够通用性的平台和生态系统，才能支持完整的开发模型，而不仅仅支持解决方案的独立单元。

PYNQ 框架

赛灵思 Zynq-7000 SoC 包含用来为现代 SoC 提供不成文标准特性的双核 Arm® Cortex®-A9 处理器系统 (PS) 和可编程逻辑 (PL)，同时它还提供独特的高度差异化灵活性，支持将关键任务卸载到 PL 。Zynq UltraScale+ MPSoC 和 Zynq UltraScale+ RFSoC 使用四核 Arm Cortex-A53 PS、PL 和其他特定部件型号的处理块进一步扩展这一模型。

PS 和 PL 间的紧密耦合能实现比传统方法响应性更好、可重配置性更强、能效更高的系统。基于 CPU 的传统方法需要使用外部存储器来共享镜像等大型数据结构。因为需要片外仲裁和通信，这样做会降低确定性并增大功耗与时延，Zynq 产品系列中提供的异构片上系统器件允许设计人员在器件的 PL 内为功能提速。这样，提供的解决方案不仅拥有确定性响应时间，还能降低时延，优化功耗。参见图 2。

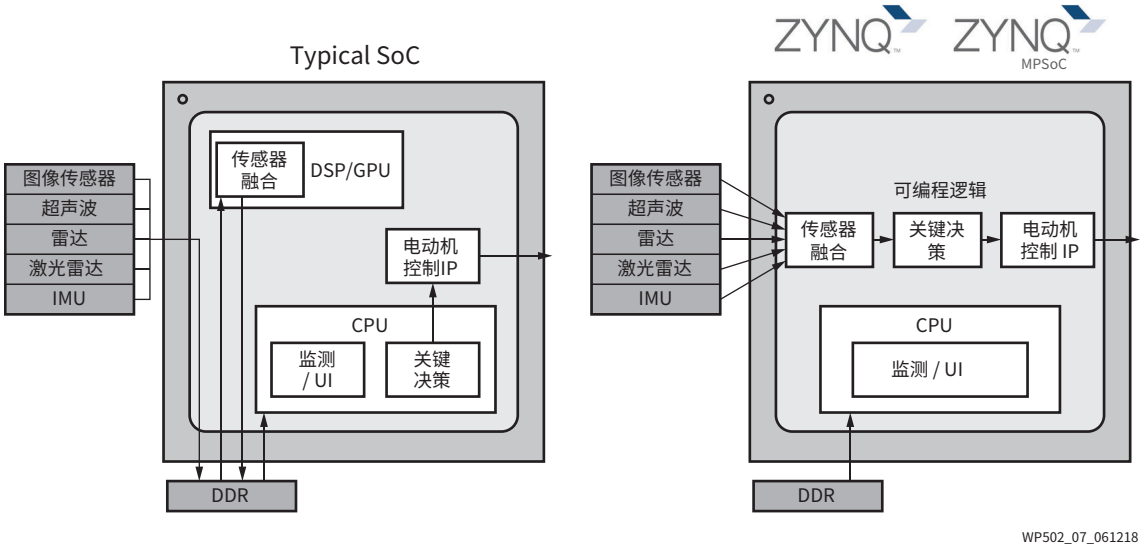

图 2： **与典型 SoC 相比 Zynq 产品系列的优势**

使用 PL 能提供比传统 CPU 方法更丰富的接口功能，因为后者只提供固定接口。PL I/O 结构的灵活性可支持业界标准、专有或传统接口，从而实现任意连接。

在高级综合工具 SDSoC™ 和 Vivado® HLS 的帮助下，开发人员能够把 C 和 C++ 编译代码直接转化为硬件。

结合可编程逻辑中功能的加速，还能够实例化一个或更多的 MicroBlaze™ 32 位软核处理器。MicroBlaze 核允许执行实时关键应用。这些功能让 Zynq 产品系列可用来满足支持 Python 的嵌入式平台的所有要求。

PYNQ 框架支持 Python 能与 Zynq 产品系列一起使用，可充分发挥可编程逻辑提供的加速功能。为实现这一点，PYNQ 采用了将 PL 叠加封装成混合库的方式。

混合库是一种全新形式的库，包含叠加比特流和与之相关视硬件而定的 C 代码和 Python API。混合库是实现重复使用的关键机制，使用 PIP Install（用于安装和管理使用 Python 编写的软件包的封装管理系统）就能轻松安装。

因此，PYNQ 框架为数据科学家、嵌入式工程师、硬件工程师和系统工程师等所有开发人员提供了所需的必要抽象层次。参见图 3。

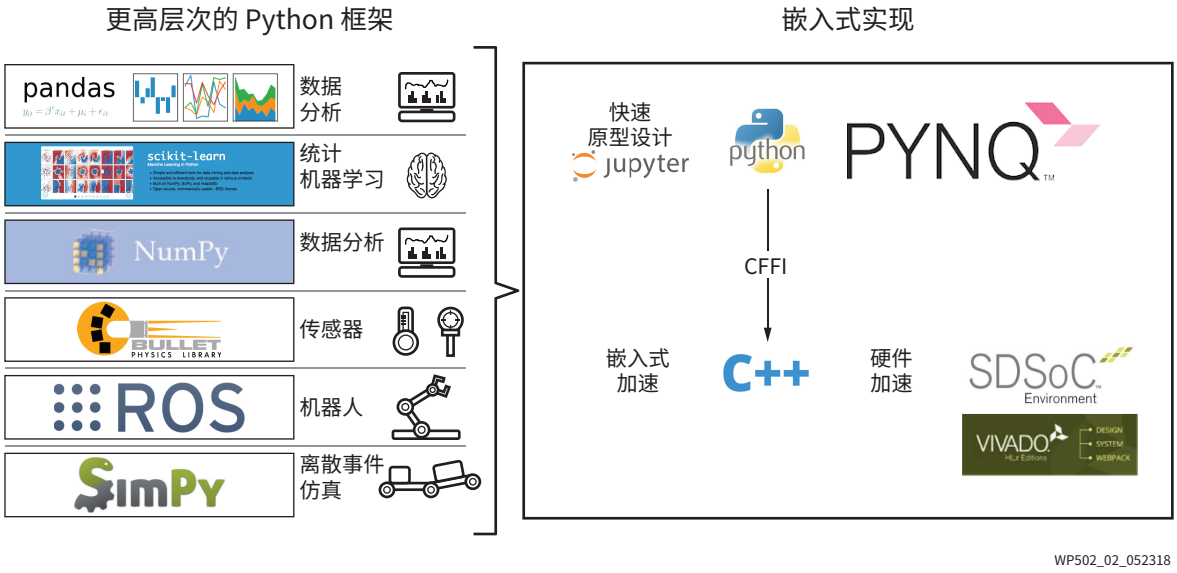

图 3： **PYNQ 框架中不断提高的抽象水平**

PYNQ 框架率先结合下列要素，能简化和改进基于 Zynq 产品系列的设计：

- 高层次生产力语言 (**Python**)
- 在可编程逻辑内实现加速的**混合库**
- 受嵌入式处理器支持的**基于 Web 的架构**
- **Jupyter Notebook** 框架

与不能使用可编程逻辑的传统 SoC 方法相比，依托这些要素的 PYNQ 框架拥有显著优势。

数据科学家能立刻把由赛灵思用熟悉的封装在 Python 框架里创建的系统投入使用。图 3 展示了可以使用一些标准封装的堆栈。Panda、Scikit-learn 和 NumPy 位于顶层，其他封装则提供专门功能，比如用于机器人的 ROS 和用于仿真的 SimPy。

Python 能够导入一个或多个随时可用的预配置硬件模块，为应用加速并消除瓶颈。所有封装都能在 Jupyter 环境中进行尝试和使用，并可使用基于 FPGA 叠加的硬件库接口连接到可编程逻辑。

PYNQ 框架是围绕开源社区设计的，开发人员在该社区可以创建和共享叠加。使用这种方法，开发人员在确认需要新叠加之前不需要构建叠加。也就是说，在运行所需功能的叠加不存在时才需要构建它。每个新叠加都应严格遵循为工业物联网确立的“设计模式”。这些设计模式包括（但不仅限于）下列模式：

- **加速器**为计算提速-与主处理器共享和/或交换数据。根据数据大小和性能要求，数据交换可以在 PL 存储器内（块 RAM）、片上存储器内 (COM)、L2 高速缓存内和 DDR 存储器内进行。
- **日志记录器**负责采集数据，一般是与主处理器共享的原始数据。根据数据大小和性能要求，日志记录可以在 PL 存储器内（块 RAM）、片上存储器内 (COM)、L2 高速缓存内和 DDR 存储器内进行。采集过程可以通过明确触发来自主处理器的事件或通过捕获外部事件来启动。
- **定序器**生成逻辑值的自动序列；根据复杂程度，这其中可包含用于实现布尔函数、FSM 和/或仲裁数字模式的可编程生成器实例。
- **运行器**提供可用 C/C++ 语言进行编程的基于 MicroBlaze 32 位软核处理器的实时控制功能，用于从主处理器卸载重复性任务以确保确定性。
- 可以把运行器用作**安全模块**。可配置 MicroBlaze 处理器用于锁步操作。虽然整个 Zynq 产品系列都针对功能安全性进行了精心设计，但当产品从最初的原型设计进入到可生产阶段时，这一功能尤为重要。

传感器与测量系统

传感器是任何工业系统，尤其是工业物联网解决方案的关键组成部分。从简单的温度测量热电偶，到结合多个异构传感器的用来测量特定物理量的复杂传感器融合，工业物联网解决方案采用多种不同的传感器模态。在工业物联网解决方案中实现 ML，有助于开发人员让给定传感器发挥出最佳性能，同时提高下列操作的效率：

- 传感器数据采集（例如：振动分析）
- 传感器数据标准化
- 传感器线性化
- 传感器诊断
- 高级传感器
- 传感器融合
- 校准与自校准

传感器诊断

因为老化原因，传感器性能在整个工作寿命期间会发生变化。传感器在恶劣环境中使用时尤其如此，此时老化会影响可靠性，并带来偏离和偏差问题。此外，如果将传感器用于安全应用，传感器诊断功能同样极为有用；在此情况下，正确的诊断流程也是安全系统的组成部分。

通过使用 ML，设计人员能够创建传感器模型，或更为普遍的说法，即数字孪生，在持续监控实际传感器输出的同时预测传感器输出。在额定条件下，传感器信号遵循某种已知模式，且伴随系统与测量噪声导致的一定程度的不确定性。但是，如果传感器失效，观察到的输出就会与预测输出相左，并且当偏差越过指定时序或阈值时，就可以明确声明传感器失效。由于分析冗余技术允许使用工作在额定条件下的传感器提供的信息为动态系统创建模型，因此“数字孪生”不会老化或失效，会永久存续。

预测性维护用例：用于诊断与安全的滚珠轴承故障检测

封装材料行业已经认识到“全面生产维护”作为提高设备可靠性的积极方法体系的重要意义。逐渐发生的轴承失效是行业故障最主要的原因之一。因此，尽早地检测这些故障对确保可靠高效的运营而言至关重要。单个包装机往往就装有 8 部以上的电动机和众多主轴，存在可能导致生产线停运的多个故障源。

滚珠轴承能确保轴以最小摩擦自由转动，同时让轴保持在正确位置上。如果旋转系统的滚珠轴承失效，后果将不堪设想。造成轴承受损的原因有很多，包括：

- 轴承错位
- 微动磨损，对轴承接触面的一种侵蚀性破坏
- 变频驱动，产生导致点蚀、槽蚀和弧坑的轴电流
- 轴承润滑不当
- 轴承侵蚀
- 轴承疲劳
- 高温和其他因素

在故障发生之前检测此类故障，对于降低运营成本和维护成本而言有极为重要的意义。幸运的是，低成本加速计的问世提供高带宽测量，能支持功能极为强大的振动数据采集系统。将这些加速计与先进的电动机控制功能共同部署，便可基于 PYNQ 框架实现侦测系统，实时检测可能的失效。

对于本应用而言，PYNQ 架构部署在 ARTY-Z7 电路板上。该电路板支持基于 PYNQ 的叠加，但这只仅仅是 PYNQ 框架的可能的用途之一。在本例中，系统设置包含一个高性能电动机控制系统和一个由 5 个 Kionix(1) 三轴加速计构成的采集系统。

可以使用多种技术进行振动监测，其中包括振动测量、声学测量、温度测量和磨损分析。使用硅微加速计为感应全部三个轴向的振动提供了低成本的方法，从而能够全面掌握系统中发生的所有可能的振动模式。具体如图 4所示。

为了确保测量准确，加速计位置应贴近电动机、齿轮箱和主滑轮内的滚珠轴承。

电动机电流、定子电压和轴角度位置等系统参数均在监测范围内。将电动机模型用于估算角度和内部电压。

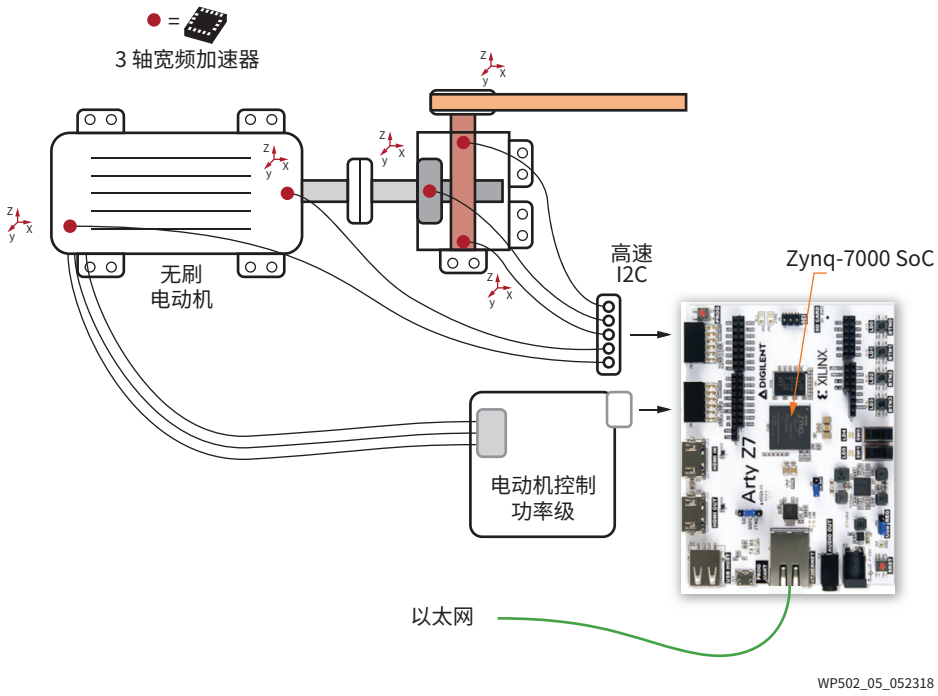

图 4： **故障检测与电动机控制**

1. 1.Kionix 公司 www.kionix.com 是一家微电子机械 (MEMS) 系统制造商，产品包括加速度计、陀螺仪和传感器融合惯性传感器技术。

如图 5 所示，使用下列单元为 PYNQ 环境创建了新的叠加：

- **电动机控制系统：**实例化为使用 QDESYS（www.qdesys.com）高性能电动机控制 IP 集的 HDL 块。
- **日志记录器：**实例化为 Vivado HLS 模块，并连接高性能 AXI 总线 0。
- **即时高速傅立叶转换 (FFT)：**实例化为 Vivado HLS 模块，并连接高性能 AXI 总线 0。
- **运行器：**用于解耦和管理加速度计，采用 MicroBlaze 软核处理器实现并以 DDR 存储器内共享段的形式对外提供。MicroBlaze 处理器：从 ARM 处理器分担加速度计管理工作。I2C 协议对这些单元进行区别对待，分别补偿偏置、灵敏度、温度和比率误差。

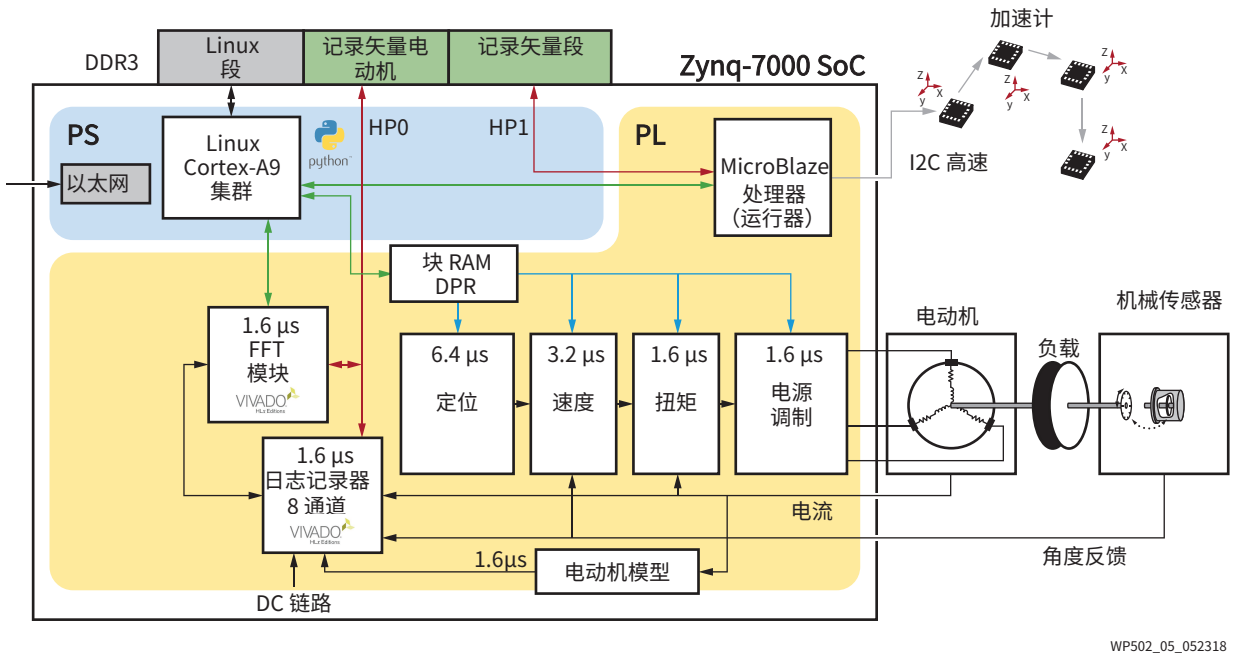

*图 5：**捕获与电动机控制内部模块***

为实现能与 PYNQ 框架配合的新叠加，使用了Python C 外部函数接口 (CFFI)。在使用 Python 时，该接口支持与几乎任何 C 代码进行交互。可编程逻辑加速功能置于 PYNQ 框架内，并且可以访问该框架下基于 C/C++ 的软件驱动程序。PYNQ 框架也提供在开发应用时有所帮助的补充功能。PYNQ 的两大核心功能包括：

(a) **mmio**：用于实现存储器映射 I/O，以及 **xlnk**：将 DDR 存储器分配为 NumPy（使用 Python 开展科学计算的基本库）可见的缓存空间。**xlnk** 负责为 PL（用于映射登记的向量）获取虚拟地址和物理地址。

举个简单的例子，可以借助观察通过电动机的电流，来建立加速度计输出的振动信号与检测到的振动信号间的关联关系，因此开展传统的振动分析或是更先进的信号特征分析就可以建立这一关联。

具体指标及整体系统的描述将占尽单独一本白皮书的篇幅，故不在本白皮书中详述。作为展示，图 6 体现了超过阈值的滚珠轴承频率限值信号由图 7 中看似正常的电流波形中提取的。

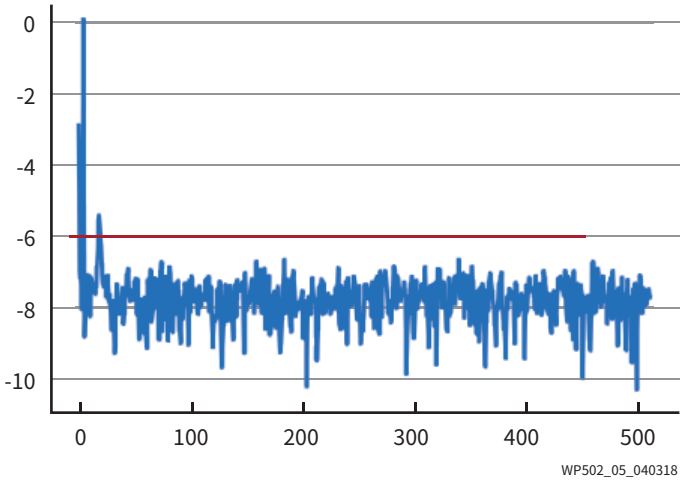

图 6： **使用电流信号特征的振动分析**

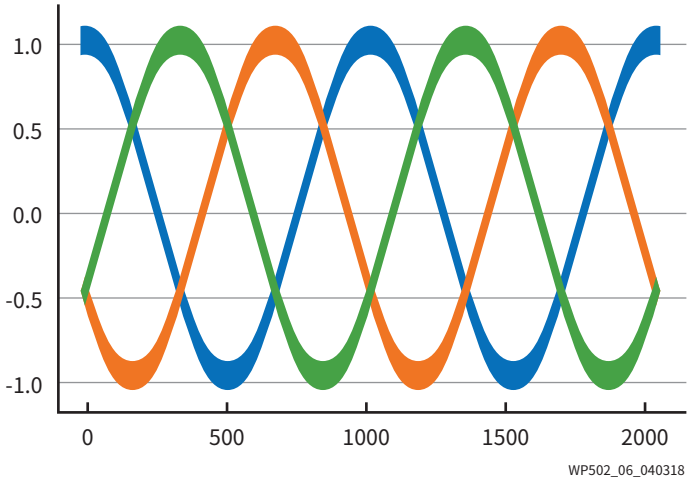

图 7： **三相电动机的波形（滚珠轴承信号特征被噪声淹没）**

上述示例证明，只要正确理解了问题和 Zynq 产品系列特性，就能开展切实的高级诊断，为具备机器学习功能的工业物联网系统敞开通向更智能方法的大门。

结论

本白皮书仅涵盖 PYNQ 框架的一些基础知识，该框架将 Python 与 Zynq 组合结合为一体。PYNQ 框架提供了新的切实可能性，有助于打造能够充分利用工业物联网全部功能，以及机器学习提供的新兴功能的系统。

边缘计算机器学习的发展尚处于萌芽阶段，而赛灵思致力于交付一流的产品和设计框架，以加快和简化智能和自适应资产的设计、部署与维护。这仅仅是动人旅程的开端，本白皮书中提及的许多话题将在后续的白皮书、应用说明和示例中详细介绍。其中一部分已经在 PYNQ GitHub 库里提供。

入门

- **PYNQ 框架**见 http://www.pynq.io。
- 本白皮书里介绍的**Arty Z7 板**系由 Digilent 公司提供，见：store.digilentinc.com/arty-z7-apsoc-zynq-7000-development-board-for-makers-and-hobbyists/
- 含 Arty Z7 板、用于电动机控制的电源模块以及 BLDC 电动机的套件见：https://shop.trenz-electronic.de/en/TEC0053-04-K1-EDDP-Motor-Control-Kit- with-Motor-Power-Supplies

修订历史

下表列出了本文档的修订历史：

日期	版本	修订描述
06/15/2018	1.0	赛灵思初始版本。

免责声明

关于与汽车相关用途的免责声明