Computer Science and Application 计算机科学与应用, 2025, 15(2), 110-117
Published Online February 2025 in Hans. https://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2025.152038

Hans汉斯

基于Python的双串口数据采集系统

王晨露，焦慧敏*

北京印刷学院机电工程学院，北京

收稿日期：2025年1月12日；录用日期：2025年2月14日；发布日期：2025年2月25日

摘　要

为了解决工程应用中的多串口通信需求，本文提出了一种基于Python和串口库实现双串口数据采集的技术方案。通过分析基于Python的双串口通信实现原理，结合人体站立平衡反馈实验数据采集系统的设计实例分析系统设计流程，利用实验配套硬件模块搭建并验证了该双串口数据采集系统的可行性与实用性。实验结果表明，系统在满足基本功能需求的同时实现了可靠的双串口通信。该系统的设计和实现方法为解决多串口通信问题提供了一个有效的解决方案。

关键词

数据采集，Python，串口通信，上位机

Python-Based Dual Serial Ports Data Acquisition System

Chenlu Wang, Huimin Jiao*

Department of Mechanical and Electrical Engineering, Beijing Institute of Graphic Communication, Beijing

Received: Jan. 12th, 2025; accepted: Feb. 14th, 2025; published: Feb. 25th, 2025

Abstract

In order to solve the need of multi-serial communication in engineering applications, this paper presents a technical scheme to realize dual-serial data acquisition based on Python and serial library. By analyzing the implementation principle of Python-based dual-serial communication and combining with the design example of human standing balance feedback experimental data acquisition system, the system design process was analyzed, and the feasibility and practicability of the dual-serial data acquisition system was established and verified by the experimental supporting hardware module. The experimental results show that the system can meet the basic functional

*通讯作者。

文章引用: 王晨露, 焦慧敏. 基于 Python 的双串口数据采集系统[J]. 计算机科学与应用, 2025, 15(2): 110-117.
DOI: 10.12677/csa.2025.152038

requirements and realize reliable dual-serial communication. The design and implementation of the system provide an effective solution to the problem of multi-serial communication.

Keywords

Data Collection, Python, Serial Communication, Upper Computer

1. 引言

在数据采集系统中，双串口通信是一种常见的需求，尤其是涉及系统同时采集多个外部设备数据时的场景。双串口通信能够使数据采集系统独立、并行地与多个外部设备进行数据交换，避免了数据延迟、提高了系统的实时性。同时，它解决了数据采集系统中的多设备管理、复杂通讯协议拆分、数据集中分析处理等问题，提高了数据传输的效率和系统的灵活性与可靠性[1]。

当前研究中，实现多串口通信的方案可分为硬件和软件两种。硬件方案包括专用多串口卡[2]、嵌入式系统的多串口控制器[3]-[5]和 FPGA [6]等。专用多串口卡通过增加多个串口来提供高稳定性和实时性，适用于高并发的工业自动化和数据采集系统，但成本较高，扩展性差。嵌入式系统(如 ARM、DSP)通过集成多个串口进行并行通信，功耗低、集成度高，但开发周期长，灵活性较差。FPGA 方案则通过硬件编程实现极高的实时性和并发处理能力，适用于高速数据采集，但其开发难度较大且成本较高。与硬件方案相比，软件方案更具灵活性和成本优势。例如使用操作系统自带的串口应用程序接口(Application Programming Interface, API)开发[7]，该方法操作简单，但在多串口处理时可能存在阻塞问题，无法充分发挥并发性能。使用第三方软件开发如 LabVIEW [8]，其提供了丰富的图形化界面和模块，适用于原型设计，但实时性较差。Python 中的 pyserial 库因其简单易用、功能强大、开源等优点[9]，成为一种常见的串口通信解决方案，可以灵活运用于小型数据采集系统，缩短开发周期，节省成本。

根据上述论点，本文分析了基于 Python 的双串口通信实现原理，并通过人体站立平衡反馈实验数据采集系统的设计案例，详尽阐述了基于 Python 的双串口数据采集系统的设计流程。系统功能验证表明，该数据采集系统不仅有效地执行了实验数据采集、分析、存储和数据可视化等核心任务，而且实现了主机与多模块间的数据交互，从而简化了实验操作流程，并满足了多串口通信的需求。这一设计在实际工程当中具有一定的应用价值。

2. 基于 Python 的双串口通信实现原理

基于 Python 的双串口通信系统软件架构采用分层设计的原则，以保证整个系统的可维护性、可扩展性和模块化。系统的架构由上至下主要由数据访问层、业务逻辑层、表现层构成，如图 1。

数据访问层：在双串口通信系统中，数据访问层负责与串口硬件进行底层数据交互，并实现串口初始化、数据读写、状态监控和错误处理等功能；

业务逻辑层：业务逻辑层是双串口通信系统的核心，它负责处理串口通信的核心业务规则和逻辑，主要实现了数据解析与封装、通信协议处理、数据流转控制、业务规则实施和日志等功能；

表现层：在双串口通信系统中其主要实现了用户端的界面支持、请求响应、反馈与交互以及数据展示等功能。

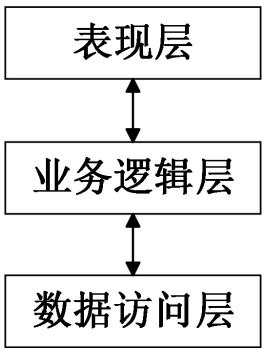

Figure 1. Software layered architecture
图 1. 软件分层架构图

2.1. Pyserial 库调用与数据访问层实现

在 Python 中对串口进行的操作是通过调用 pyserial 库来实现的。pyserial 库是 Python 第三方库，支持 Windows、OS X、Linux 等多种操作系统。通过 pyserial 库的 API 函数可以实现数据访问层所需的串口的初始化、数据读写和状态监控等各项功能。

串口初始化时，通过创建不同的 serial 对象并配置端口号、波特率、数据位、停止位及校验位等参数，确保与实际硬件匹配，实现串口的正确连接。数据读写功能利用 pyserial 封装的 API 函数实现，使用 open()、close()函数控制打开或关闭串口，使用 write()和 read()函数按字节读取串口数据。通过读取 in_waiting 变量获取串口缓冲区的可读数据的字节数，使用 timeout 参数设置数据读取的超时时间。对于可能出现的异常情况，如端口打开失败、数据传输错误、超时等，需要在程序中触发异常类 serial.SerialException 来帮助识别错误类型。使用 try-except 块，先执行打开、关闭等相应的串口操作程序，当程序执行失败时触发 serial.SerialException 来捕获异常。当出现异常时，通过提前设置的重试次数和间隔时间，重新执行相应的串口操作程序。对于某些可恢复的错误，重试机制可以提高数据传输的稳定性。

2.2. 基于 Threading 库与多线程的数据流转控制

多串口在收发数据时通常会遇到数据阻塞、数据并发处理、响应时间要求高以及业务逻辑复杂等诸多问题，此时就需要引入多线程机制来进行数据的流转控制。多线程编程可以确保每个串口的操作都能独立进行，不会相互干扰，从而提高系统的响应速度和可靠性。

在 Python 中，threading 模块提供了一个高级的、基于线程的并发性接口，能够实现系统的多线程处理。同时双串口通信作为 IO 密集型任务，在能够避开 Python 当中全局解释器锁(Global Interpreter Lock, GIL)同一时间只有一个线程可以执行 Python 字节码的限制，使得多线程在 IO 等待期间实现有效的并发处理，从而提高系统的整体性能和响应能力。

为实现并发数据处理，系统通过 threading 库为每个串口创建独立线程来处理数据收发。每个线程使用 thread.start()启动，并通过 thread.join()确保主线程在所有子线程完成后继续执行。为了避免多线程访问共享串口资源时发生冲突，采用 threading.Lock 来保证同一时间只有一个线程能进行读写操作，从而确保数据的正确性与线程安全。

2.3. 基于 Qt 和 Matplotlib 库的用户界面设计

Qt 是一个成熟的跨平台 C++框架，提供了丰富的图形用户界面(Graphical User Interface, GUI)组件、事件管理和布局机制。Python 中通过 PyQt 库可以访问 Qt 的所有功能，PyQt 使得开发者能够在 Python 环

境下开发复杂的 GUI 应用。Matplotlib 则是 Python 中一个功能强大的绘图库，能够生成静态、动态以及交互式图形，广泛应用于数据可视化领域。

在实际的 GUI 开发过程中，常使用 Qt Designer 进行用户界面的设计，Qt Designer 通过可视化操作简化了界面的布局和控件的配置。界面中的控件通过 PyQt 与 Python 脚本进行绑定，用户可以在界面上配置串口参数，并实时查看串口通信状态。为了实现采集数据的动态可视化，界面中将 Matplotlib 绘制的曲线通过 FigureCanvas 类嵌入到 Qt Designer 中的 QWidget 绘图区域，来展示实时数据曲线。

Qt 的信号(signals)和槽(slots)机制是实现界面交互的核心。每当串口接收到数据时，串口读取线程通过发出信号将数据传递给主界面。主界面中的槽函数接收数据，并对其进行解析和更新，最终通过 Qt 的 GUI 组件实时展示数据。此机制确保了数据处理与界面更新的高效同步。

利用 Qt 结合 Matplotlib 的方案不仅能够满足数据交互与实时显示的需求，还通过信号和槽的异步通信机制，避免了界面卡顿，提升了系统的响应速度和用户体验。

3. 双串口数据采集系统设计与实现

为了进一步双串口通信方案的可行性，本文以人体站立平衡反馈实验[10]为例，为该实验系统设计了一个基于 Python 的双串口数据采集系统。

3.1. 系统结构

人体站立平衡反馈实验的数据采集系统结构如图 2 所示。穿戴在人体上的 IMU (惯性测量单元，Inertial Measurement Unit)传感器模块通过串口向计算机实时发送人体站姿数据，计算机将站姿数据分析和处理后，将相应控制信号通过另一串口发送至温度反馈模块，同时温度反馈模块将实时温度信息传回计算机。人体通过感知反馈模块的温度变化，判断自身当前的站立姿态，以便调整站姿并保持站立平衡。

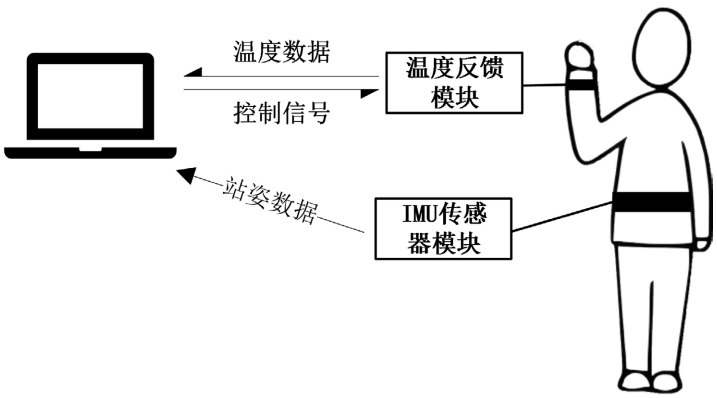

Figure 2. Structure of human standing balance feedback experiment data acquisition system
图 2. 人体站立平衡反馈实验数据采集系统结构图

3.2. 系统硬件

在人体站立平衡反馈的实验中，实验穿戴装置主要使用了 IMU 十轴姿态传感器、TCM 数字温控模块及相关外围硬件，两模块分别用于采集人体姿态数据和提供温度反馈。

TCM 温控器是温控模块的核心器件，它采集外接的温度传感器数据与内部设定温度进行比较，再通过内部控制算法产生电压差作用于帕尔贴上，最终实现对目标的温度控制。温控器采用 RS232 三针(1

COM; 2 RXD; 3 TXD)接口并使用 RS232 协议与电脑进行通讯。图 3 为 TCM 数字温控模块的结构图。

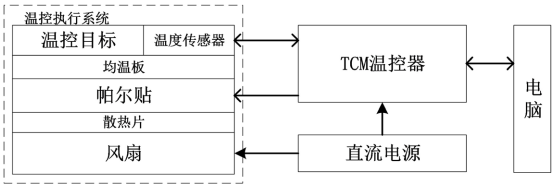

Figure 3. Temperature control module structure diagram
图 3. 温控模块结构图

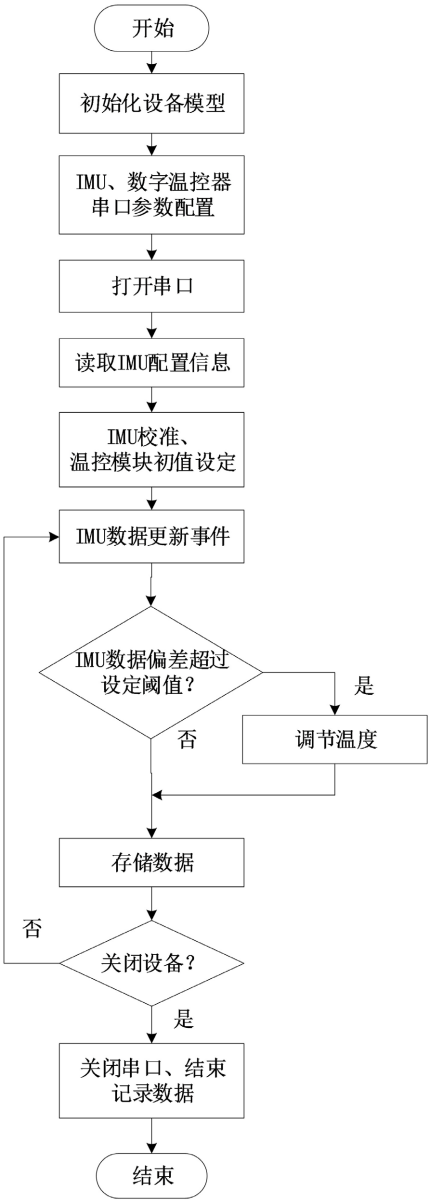

Figure 4. Main program block diagram
图 4. 主程序框图

IMU 传感器集成模块是基于 MEMS (微电子机械系统，Micro-electromechanical Systems)技术的高性

能三维运动姿态测量模块，模块内部自带卡尔曼滤波、高精度校准和标定算法。模块采用串口通信，波特率 4800~921,600 可调。

3.3. 程序设计

根据软件分层架构，将人体站立平衡反馈实验系统的程序进行分层和模块化设计，程序在每一层分别实现了如下功能：

数据访问层：根据温控模块及 IMU 模块的通信参数要求封装了两个串口模型，实现了上位机对两个串口的初始化、串口控制、数据读写、错误处理等功能；

业务逻辑层：根据通信协议对串口数据进行解析与处理，将 IMU 采回数据进行存储，同时该层还实现了整个系统的多线程处理和温度反馈控制逻辑；

表现层：实现了温控模块与 IMU 模块串口参数配置的可视化，以及温度和 IMU 采回的加速度数据曲线的实时显示。

主程序的程序框图如图 4 所示。该框图描述了整个系统的主要工作流程。

系统启动后初始化设备模型并将与 IMU 和温控模块连接的两个串口进行参数配置；打开串口后首先对两个模块进行校准并设置初始工作状态，随后开始循环更新数据并引入判断逻辑和温度调节过程，系统工作过程中的数据会被实时存储，并在接收到关闭设备信号关闭串口后，将数据记录在文件当中。

为了避免在读取 IMU 数据时频繁操作串口导致程序阻塞在程序中使用了多线程编程，本方案中将主程序作为主线程，其负责执行串口资源的配置与调用、分支线程的创建与调用等任务。分支线程作为输入处理线程，从串口读出数据并将数据传送给主线程。多线程的引入可以让耗时的 IO 任务转到分支线程中执行，在此期间主线程可以执行其他任务。

3.4. 上位机界面设计

上位机界面主要实现了对温控模块与 IMU 模块的串口参数可视化配置，以及实时显示 IMU 采集的加速度数据曲线和实时温度信息，方便实验人员的使用。

用户界面具体的设计过程是：使用 Qt Designer 工具完成界面的绘制并生成文件；根据界面功能编辑信号与槽函数，实现界面与主程序的逻辑连接；最终将界面文件生成为可执行的应用程序。

4. 系统功能验证

根据人体站立平衡反馈实验数据采集系统的设计实例，本文结合系统硬件进行了实验测试，对整个系统的功能和双串口通信性能进行了验证，包括：结合用户界面和系统存储功能验证系统是否成功收到 IMU 和温控模块两个串口的数据；在实验条件下对系统数据的传输速率、准确性以及系统的稳定性是否满足实验要求进行验证。

4.1. UI 界面功能验证

将硬件与电脑用串口相连，在电脑上运行 Qt Designer 生成的可执行文件。此时，在打开的 UI 界面上配置串口参数，两个串口的端口号、波特率、数据位、校验位等参数信息分别要与实际的硬件模块相匹配。最终配置完成的串口参数：IMU 连接端口 COM3、波特率为 115,200；温控模块连接端口 COM4、波特率为 9600。

参数配置完成后点击打开串口，此时系统软件正常运行，观察到曲线绘制窗口产生了当前加速度数据生成的图像信息，温度窗口则成功显示当前温控模块传回的温度信息，如图 5 所示。

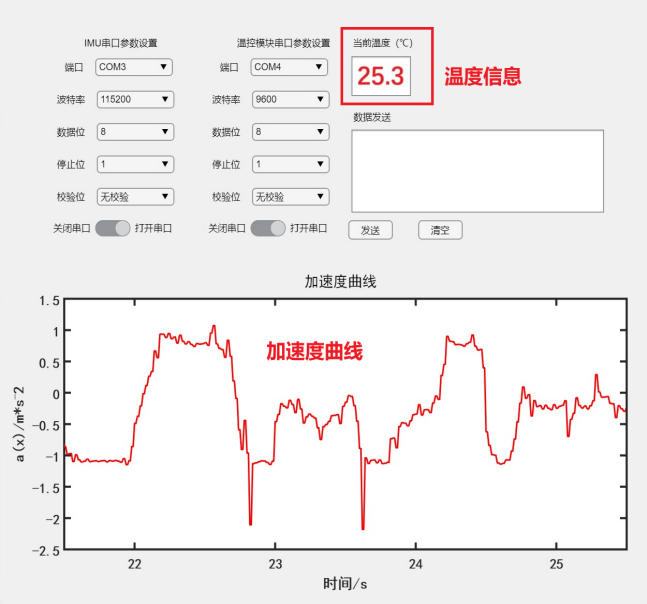

Figure 5. UI interface display
图 5. UI 界面显示

4.2. 数据存储功能验证

在系统运行程序结束后，在程序内设定的存储路径下，可以观察到 IMU 传回的时间、加速度等信息和温控模块传回的温度信息成功保存在文本文件中，如图 6 所示。通过时间数据可得系统每 10 ms 更新一次加速度和温度等参数信息，满足实验要求。

Chiptime 时间	ax(g) x轴加速度)	az(g)	wx(deg/s)	wy(deg/s)	wz(deg/s)	AngleX(deg)	AngleY(deg)	AngleZ温度	T(°)
2023-2-26 10:49:0.849	0.0635	0.2134	0.9678	-47.4854	45.2881	10.0098	11.536	4.713	122.651	31.89
2023-2-26 10:49:0.949	0.0747	0.1572	0.9907	-48.9502	-1.709	18.8599	6.85	6.57	125.046	31.87
2023-2-26 10:49:1.49	0.0352	0.1372	1.0542	28.1982	10.0708	-13.0615	4.724	5.944	126.008	31.9
2023-2-26 10:49:1.149	-0.0356	0.1055	1.2065	80.4443	11.5967	-112.5488	10.701	4.01	119.894	31.86
2023-2-26 10:49:1.249	-0.0786	-0.0488	0.9878	-107.3608	-18.7988	-109.8022	3.79	6.196	107.1	31.85
2023-2-26 10:49:1.349	-0.0142	-0.2935	0.9849	-13.4277	5.4321	-42.2363	-2.483	5.685	97.224	31.87
2023-2-26 10:49:1.449	-0.0254	-0.1201	1.0913	101.9897	-8.606	171.7529	3.999	5.587	103.634	31.86
2023-2-26 10:49:1.549	-0.0732	0.04	1.1035	49.9268	23.1323	104.4312	7.152	3.95	115.461	31.9
2023-2-26 10:49:1.649	0.0864	0.1587	0.9639	12.207	-7.4463	1.2207	10.931	3.433	122.536	31.85
2023-2-26 10:49:1.749	0.0981	0.1274	0.9634	60.791	-33.7524	-16.0522	14.101	3.241	122.173	31.88
2023-2-26 10:49:1.849	0.0601	0.2197	0.9028	-16.4185	-16.9678	-29.1748	14.974	3.801	118.68	31.85
2023-2-26 10:49:1.949	-0.0195	0.1602	0.9121	12.3901	13.3057	-12.9395	15.837	4.021	116.494	31.86
2023-2-26 10:49:2.49	-0.0605	0.1548	0.877	25.7568	15.5029	-10.6201	17.715	6.735	115.582	31.88
2023-2-26 10:49:2.149	-0.0737	0.1543	0.8232	0.3052	2.1362	-3.7842	19.133	7.872	115.093	31.84
2023-2-26 10:49:2.249	-0.0547	0.1523	0.835	-3.479	0.1221	-0.4883	18.166	7.515	115.093	31.86
2023-2-26 10:49:2.349	-0.0645	0.146	0.835	-0.9766	2.5024	-0.5493	17.457	7.328	115.159	31.88
2023-2-26 10:49:2.449	-0.0591	0.1523	0.8374	0.7324	0.5493	-0.9766	16.919	7.092	115.296	31.86
2023-2-26 10:49:2.549	-0.0542	0.1475	0.8315	-0.2441	-0.7324	-0.061	16.49	6.762	115.411	31.89
2023-2-26 10:49:2.649	-0.0396	0.1968	0.835	-12.3291	-0.4272	9.7656	15.557	6.383	115.675	31.85

Figure 6. Data storage function test
图 6. 数据存储功能测试

4.3. 数据传输稳定性与准确性验证

依据[10]中的实验方法，使设备工作在实验条件下，待硬件与采集系统成功连接并开始工作后，连续采集数据并通过适当地摆动改变 IMU 的位姿，单次采集过程持续 5 分钟，并进行多次试验。在程序中加入校验计数功能，当数据未通过 CRC (循环冗余码校验，Cyclic Redundancy Check)校验时计数值加一，

实验结束时输出未通过校验的次数。其中三次试验的计数结果见表 1。

Table 1. Data transmission verification
表 1. 数据传输验证

序号	未通过校验的次数	准确率
1	0	100%
2	0	100%
3	0	100%

实验结果表明，该双串口数据采集系统能够稳定地采集两个串口的数据，同时系统在运行过程中未产生数据报错，数据传输的准确性较高，系统基本能够满足实际数据采集的需求。

5. 结论

为了解决工程应用中数据采集系统的多串口通信需求，本文提出了使用 Python 语言和串口库方式实现双串口通信的技术方案。文章分析了基于 Python 的双串口通信实现原理，提出了一个基于 Python 的人体站立平衡反馈实验系统软件的设计实例，通过将系统与实际硬件相结合进行实验测试，验证了该系统的可行性与实用性。实验表明，该系统软件实现了可靠的双串口通信性能，可以稳定地采集数据并进行实时处理，能够满足系统的功能需求。该方法简化了人体平衡实验的实验流程，方便实验者对数据进行采集分析。同时该系统的设计和实现方法为解决多串口通信问题提供了一个有效的解决方案，具有较高的实用价值和推广潜力。在未来的研究中，将在系统功能的拓展、算法融合和系统安全性等方面进一步完善和优化系统的性能，增强其在高负载和复杂环境下的稳定性和可靠性。

参考文献

[1] 李昊昱, 缪炜涛, 郭玉洁, 等. 一种多串口通信冲突回避与数据同步方法[J]. 通信电源技术, 2020, 37(24): 103-104, 108.

[2] 朱梓铭. 基于 FPGA 的全国产 PCIe 总线隔离多串口卡的设计与实现[J]. 工业控制计算机, 2024, 37(2): 23-24.

[3] 王欣. 嵌入式系统多串口通讯方案的设计与实现[D]: [硕士学位论文]. 武汉: 华中科技大学, 2024.

[4] 单彦虎, 于皓博, 任勇峰, 等. 基于 DSP 的多路串口扩展技术[J]. 电子器件, 2019, 42(6): 1558-1563.

[5] 刘刚. 基于 EM9000 工控板高性能双串口通信模型设计与实现[J]. 现代计算机, 2020(3): 104-108.

[6] Jin, X. and Jin, J. (2022) Design and Implementation of Multi-Serial Monitoring System Based on DSP and FPGA. 2022 *IEEE 5th International Conference on Information Systems and Computer Aided Education* (*ICISCAE*), Dalian, 23-25 September 2022, 79-83. https://doi.org/10.1109/iciscae55891.2022.9927642

[7] 龙灏, 丁旭玲. 基于 Windows API 函数的串口通讯程序设计[J]. 电子世界, 2015(19): 120-121.

[8] 孙小明, 韩彦龙. 基于 LabVIEW 的直流电机性能通用测试系统[J]. 计量与测试技术, 2024, 51(7): 99-101.

[9] 沈阚, 黄凯锋, 陈碧欢, 等. 基于静态分析的 Python 第三方库 API 兼容性问题检测方法[J]. 软件学报, 2024(7): 1-26.

[10] Ballardini, G., Florio, V., Canessa, A., Carlini, G., Morasso, P. and Casadio, M. (2020) Vibrotactile Feedback for Improving Standing Balance. *Frontiers in Bioengineering and Biotechnology*, **8**, Article No. 94. https://doi.org/10.3389/fbioe.2020.00094