

基于异步非阻塞框架的电力物联网通信技术

吴振冲¹, 徐成司², 董树锋¹

- (1. 浙江大学电气工程学院, 浙江省 杭州市 310027;
2. 国网浙江省电力有限公司杭州供电公司, 浙江省 杭州市 310016)

Communication Technology for Power Internet of Things Based on Asynchronous Non-blocking Framework

WU Zhenchong¹, XU Chengsi², DONG Shufeng¹

- (1. College of Electrical Engineering, Zhejiang University, Hangzhou 310027, Zhejiang Province, China;
2. Hangzhou Power Supply Company of State Grid Zhejiang Electricity Power Co., Ltd., Hangzhou 310016, Zhejiang Province, China)

摘要: 针对传统阻塞式通信技术在电力物联网硬件资源受限的条件下难以实现可靠高效通信的问题, 文章提出基于异步非阻塞框架的电力物联网通信技术。首先, 根据异步非阻塞技术原理, 基于 Rust 语言异步编程方法和 Tokio 异步框架, 设计异步非阻塞通信框架, 包括客户端连接、服务端连接与监听以及非阻塞的网络数据读写方式等。然后以当前电力远动通信领域中应用最为广泛的 IEC 104 规约为例, 设计其异步非阻塞通信实现方式。最后, 对基于异步非阻塞框架实现的通信软件进行一致性测试和性能测试, 将测试结果与传统阻塞式通信实现方式对比分析, 并通过实际工业园区电力物联网工程应用验证所提技术的有效性和高效性。

关键词: 电力物联网; 异步非阻塞通信; Rust 语言; Tokio 框架; 数据读写; IEC 104 规约

ABSTRACT: To solve the problem that traditional blocking communication technology is difficult to achieve reliable and efficient communication under the condition of limited hardware resources of the power Internet of things, a communication technology for power Internet of things based on asynchronous non-blocking framework is proposed. Firstly, according to the principle of asynchronous non-blocking technology, based on the asynchronous programming method of Rust language and Tokio asynchronous framework, the asynchronous non-blocking communication framework is designed, including client connection, server connection and monitoring, and the non-blocking network data reading and writing methods. Then, taking the IEC 104 protocol, the most widely used protocol in the field of power telecontrol communication, as an example, the asynchronous non-blocking communication implementation is designed. Finally, the consistency test and performance test of the communication software based on the asynchronous non-blocking framework are carried out. The test results are compared with the traditional blocking communication method, and the software is applied to power Internet of things of actual industrial parks, which verifies the effectiveness and efficiency of the proposed technology.

KEY WORDS: power Internet of things; asynchronous non-blocking communication; Rust language; Tokio framework; data read and write; IEC 104 protocol

0 引言

在“双碳”战略背景下电力系统接入的新能源日益增长, 为保障电网安全可靠运行, 用户侧主动参与电网调节成为必然要求, 目前国内外对于园区微电网、虚拟电厂、综合需求响应等技术已逐步开展试点工程实施^[1], 提升用户侧资源配置效率和电网电力电量平衡能力。构建支持可靠快速数据通信的电力物联网是实现用户侧能量管理和参与电网

友好互动的基础。然而在用户侧主体主要考虑经济性的条件下, 在电力物联网中配置电网自动化系统采用大规模、高性能硬件设备在实际工程难以实施^[2]。为此有必要研究高性能的电力物联网网关通信技术, 在用户侧硬件资源受限的条件下实现高效率的数据通信。

目前, 国内外对于电力物联网通信问题的研究主要集中于通信架构和功能方面^[3-4], 对于具体通信技术仍需进一步研究。在电力通信领域已经形成了 IEC 60870、IEC 61850 等标准规约, 但当前在规约实现方面主要采用阻塞式通信技术^[5]。随着近年来

计算机网络和通信技术的快速发展,非阻塞通信技术得到越来越多的关注和应用^[6]。与非阻塞通信相比,阻塞式通信效率较低,在连接数较多或连续发送报文量较大的情况下易造成数据丢失或延迟,且对计算资源的占用和浪费较大^[7]。因此,在电力物联网硬件资源受限而通信数据量不断增大的趋势下,迫切需要研究非阻塞式的电力通信协议实现方法,提高通信速度、性能和系统运行稳定性。

国内外学者运用非阻塞输入/输出(I/O)技术来提高服务器并发性能、构建高性能网络系统等方面已取得一定成果。非阻塞框架的实现方式在近年来也有了进一步发展,如何选取合适的编程语言和非阻塞框架提升通信性能是工程应用中需要重点考虑的内容^[8]。目前基于 Java 的开源框架 Netty 作为异步框架,成为非阻塞 I/O 实现的主流^[9],但 Java 与新兴的 Rust、Go 等编程语言相比在空闲运行时的内存占用和浪费较大。Rust 语言在开发设计之初就考虑了异步、高并发、运行速度等问题^[10],相比 Java 效率更高、资源占用小,且相比 C/C++ 语言具有更高的内存安全性,基于 Rust 语言的 Tokio 框架提供了异步技术实现^[11],在通信密集型程序中有利于高效完成非阻塞 I/O 操作。因此,基于 Rust 和 Tokio 框架研发电力物联网网关的异步非阻塞通信技术是具备良好应用前景的方案。

在上述背景下,本文提出基于异步非阻塞框架的电力物联网通信技术。首先介绍异步非阻塞技术原理和基于 Rust 的 Tokio 异步框架,在此基础上设计异步非阻塞通信框架,包括非阻塞客户端连接、服务端连接与监听、网络读写方式等。进而针对电力远动通信中应用最为广泛的 IEC 104 规约^[12-13],设计其异步非阻塞通信实现方案。最后,对基于异步非阻塞框架实现的 IEC 104 规约通信进行一致性测试和性能测试,与传统阻塞式实现方式对比验证所提技术的高效性。

1 异步非阻塞技术实现方法

1.1 异步非阻塞通信技术原理

计算机系统中阻塞的概念是指在执行如获取网络数据、读写文件和访问数据库等会占用 I/O 资源的任务时,所需数据未准备好情况下的处理器等待过程^[14]。如图 1 左半部分所示,一个线程先执行存在阻塞的任务 1,然后执行任务 2 和任务 3,在整个通信任务中,任务 1 中的阻塞让处理器处于空闲状态从而浪费了其宝贵资源。

对于多核计算机,采用多线程方式是通信阻塞

问题的一种解决方案^[15]。然而在多线程方式下,线程切换调度开销和上下文切换代价会给系统带来额外负担。此外,开辟线程需要为其分配一定的栈内存和虚拟内存,限制了操作系统可建立的总线程数,还有可能带来死锁、竞争冒险等问题。因此,目前使用多线程方式尚不能很好地解决阻塞通信问题。

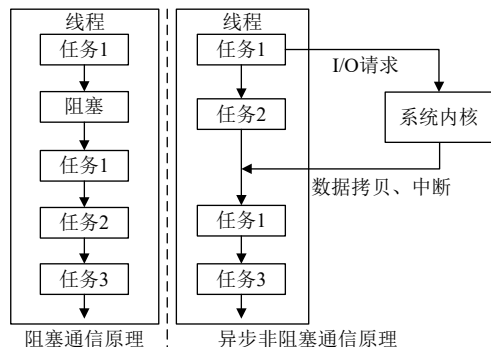


图 1 阻塞式与异步非阻塞式通信原理

Fig. 1 Blocking and asynchronous non-blocking communication principle

非阻塞 I/O 技术的提出旨在解决阻塞 I/O 的资源浪费和数据处理延迟问题,在非阻塞 I/O 中对阻塞处的返回参数进行判断,若返回参数是非法值,则线程将转而执行其他任务,待阻塞处有合法返回参数后,再继续处理该任务。非阻塞 I/O 的主要实现方式有同步和异步之分,同步非阻塞采用轮询方式查询阻塞处的返回结果,会带来额外的资源开销^[16]。

异步非阻塞技术通过异步运行时对用户设定的任务进行调度,如果某个任务发生了阻塞,则安排其他任务在线程上执行,当 I/O 数据准备好时,操作系统内核直接将数据传递到线程,并引发中断,提醒线程数据已经准备好,可以进行后续操作^[17]。如图 1 的右半部分所示,线程在执行任务 1 中遇到阻塞的 I/O 请求时,将其交给系统内核后就转而执行任务 2。I/O 得到数据后,将数据拷贝到该线程并引发中断,继续执行任务 1 的剩余部分以及后续的任务 3。

通过上述实现原理对比可看出,异步非阻塞技术的实现难度较大、需要更为复杂的异步程序编写,但其是充分利用处理器资源并支持高性能并发通信的最佳方案,能够利用等待 I/O 数据的空闲时间执行其他任务,也能通过异步运行时的调度实现多个数据请求的处理并避免额外开销。因此,本文将基于异步非阻塞 I/O 技术实现高效的电力通信。

1.2 基于 Rust 的 Tokio 异步框架

Rust 是新兴的系统级编程语言,具有运行速度快、内存利用率高、高并发和高安全性等特点,常

用于高性能的 Web 服务、嵌入式设备、系统编程等开发场景^[18]。Rust 异步编程依赖于 Future 接口的设计。如图 2 所示, Future 可理解为一块供将来调度执行的代码,其中需要实现一个 Poll 方法,用于推动 Future 完成其内部的一系列操作。如果得到了数据结果, Poll 方法返回值为 Ready(T),可以继续处理该任务,其中 T 指代返回的数据;如果未得到数据结果,则返回 Pending 线程挂起该任务。系统中断可由 Waker 唤醒挂起的 Future,重新调用 Poll 方法,此时的调用能够确保任务取得更多进展。

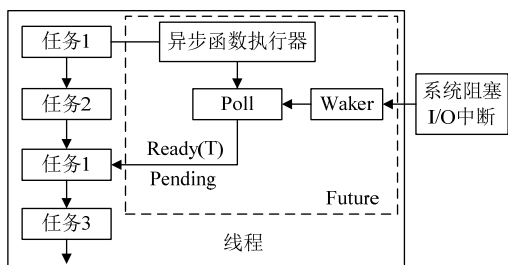


图 2 Rust 的异步非阻塞原理

Fig. 2 Asynchronous non-blocking principle of Rust

Tokio 是基于 Rust 的异步框架,其实现了一个异步运行时,对任务进行合理的调度实现异步并发操作。Tokio 框架的组织架构如图 3 所示,其中 Mio 的作用是把不同操作系统的 I/O 中断提供给 Tokio,实现跨平台的抽象。Future 和 Mio 分别提供了异步和系统中断,使 Tokio 得以实现非阻塞通信。Tokio 框架实现了非阻塞的异步任务、非阻塞的监听(TcpListener)和非阻塞的流(TcpStream),TcpStream 实现了 TCP/IP 连接的非阻塞读写。



图 3 Tokio 框架的组织架构

Fig. 3 Architecture of Tokio framework

基于 Rust 编程语言和 Tokio 异步框架开发通信程序可满足非阻塞通信的异步和系统中断 2 个条件,可用于实现电力物联网的非阻塞通信技术。

2 电力物联网异步非阻塞通信框架设计

2.1 客户端通信框架设计

基于异步非阻塞技术原理和 Tokio 框架,本节设计电力物联网通信客户端(主站)和服务端(子站)^[19]的实现框架。客户端是在通信中主动发起连接的一方,对其构造结构体 EventLoop,包含: network,由形成连接的套接字和读取缓冲区组成; cancel_rx,取消连接指令的发送端; cancel_tx,取

消连接指令的接收端。

其中, cancel_rx 和 cancel_tx 是一对异步的收发通道,可以在不同的异步任务之间传递取消连接的指令信息。构造 EventLoop 结构体的 Poll 函数,流程如图 4 所示,首先判断是否已经存在 network,若存在 network,说明已经完成连接,继续执行业务逻辑即可;若不存在 network,则需要主动发出连接请求。在连接前,可利用 Tokio 库提供的非阻塞的延迟函数 sleep 延迟一定时间,如 0.1 s,防止重连的频率过快导致处理器资源使用占比过高。

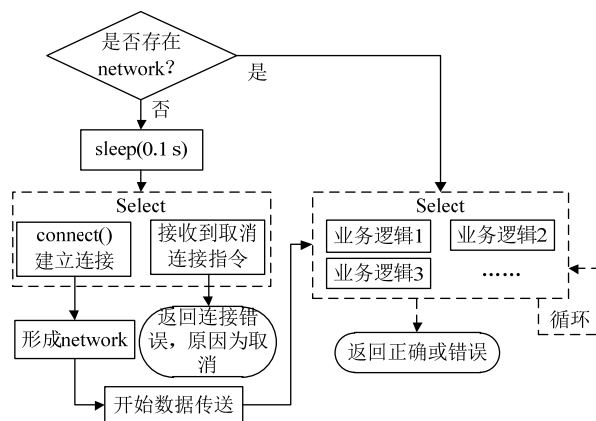


图 4 客户端 Poll 函数流程

Fig. 4 Flow chart of Poll function on client

图 4 所示 Select 宏可以同时运行多个 Future,并且对最先返回 Ready(T)的 Future 做出响应,设计利用 Select 处理连接时可能出现的两种情况:

1) 成功建立连接,这里使用的是 Tokio 框架的 TcpStream::connect()方法,是非阻塞的连接操作,连接成功后,形成 network 并开始数据传送;

2) 接收到取消连接的指令,直接返回连接错误,错误原因为取消连接。

无论是已经连接,还是重新建立了连接,后续运行的程序逻辑都是相同的。在存在连接的情况下,使用 Select 选择执行不同的业务逻辑,执行结果可能为正确、错误或执行下一次 Select。Select 的业务逻辑与应用层有关,具体实现是由 EventLoop 的 Select 函数完成,该函数将在第 3 节的应用层设计方案中进一步说明。

设计客户端 Poll 的返回值类型为 Result<Event, ConnectionError>,其中 Result<T, E>是 Rust 中的枚举类,包括 Ok(T)和 Err(E)两个成员,T 表示合法的返回数据,E 表示错误类型。在执行了 Select 函数后,若得到了正确的事件,则 Poll 函数返回该事件 Ok(Event),通知调用者此事件发生,若发生连接错误或业务逻辑报错, Poll 函数返回 ConnectionError,错误原因可分为以下几类:收到取消连接指令;连

接过程出错,包括收发数据出错、连接断开等情况,需要断开重连;应用层的断开重连。

客户端 Poll 函数及其中的 Select 函数是非阻塞的异步函数,设计为可循环调用,根据 1.2 节所述的异步编程原理,当上一次 Poll 函数返回后,才会进行下一次调用,无需担心大量循环占用处理器资源。因此,断开重连操作只需要将 EventLoop 中的 network 清除,等待下一次循环调用 Poll 函数即可。

2.2 服务端通信框架设计

通信服务端可以形成多个通道连接,为多个客户端提供服务。在收到客户端的连接请求后,服务端将验证客户端的 ip 和端口,若合法,则建立相应的通道连接^[20]。服务端 Listener 与 Poll 函数流程如图 5 所示,首先建立一个 TcpListener,其规定了服务端监听的端口。初始化时,通道状态都是断开状态。利用 Select 选择程序分支,若连接状态发生了变化,某个通道被关闭,或某个通道已经建立连接,则更新通道的状态;若使用 TcpListener::accept()方法接收到某个客户端的连接请求,则将客户端 ip 和端口与服务端通道的配置进行对比,若合法,则发送 TcpStream,否则忽略该请求,并返回连接错误。

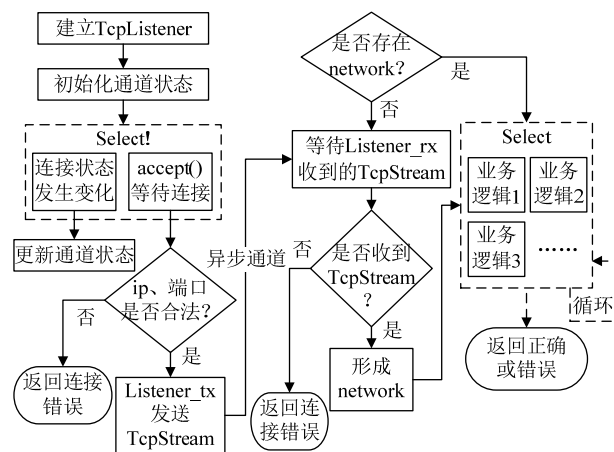


图5 服务端 Listener 与 Poll 函数流程

Fig. 5 Flow of Listener and Poll functions on server

客户端通道对应 EventLoop 的 Poll 函数执行时,依然要先判断连接是否存在。若连接不存在,则查看 Listener_rx 是否接收到 TcpStream,并形成 network,后续动作与客户端一样,执行应用层规定的业务逻辑,如读写操作、断开连接等。

2.3 非阻塞的网络数据读写

上述通信客户端和服务端框架设计实现了非阻塞的连接,进一步需要将数据读写操作也设计成非阻塞式。基于 Tokio 的 socket.read_buf()方法可以非阻塞地将数据读取至 network 的读取缓冲区。设计非阻塞的数据读取流程如图 6 所示,该图左侧为

读取一条完整报文的流程,首先检验缓冲区中的数据,尝试形成一条完整报文,若已经可以组成完整报文,则返回该报文;若经检查报文出错,则返回错误,错误类型为数据读取出错;若还缺少一定长度的报文,则继续读取该长度的报文。

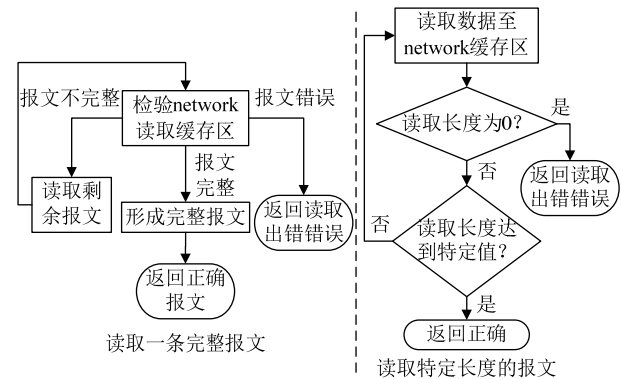


图6 非阻塞数据读取流程

Fig. 6 Flow chart of non-blocking data reading

读取特定长度报文的流程如图 6 右侧所示,只有当读取到数据时,socket.read_buf()方法才会返回,因此若读取的长度是 0,说明连接出现问题。当读取的长度大于或等于需要长度时,返回新读取的报文长度,否则需要继续读取。因读取至缓冲区的底层实现是非阻塞的,其外部的异步函数均具有了非阻塞的性质,基于此可实现非阻塞的读操作。

非阻塞的写操作实现较为简单,基于 Tokio 的 socket.write_all()方法即可非阻塞地发送数据。以上基于 Rust 和 Tokio 提供的 TcpStream、TcpListener 和异步通道所设计的客户端与服务端通信框架,实现了客户端与服务端的非阻塞连接和读写操作。

3 异步非阻塞的 IEC 104 规约实现

3.1 IEC 104 规约应用层实现方案

在第 2 节所述异步非阻塞通信框架基础上,本节以目前电力运动通信中广泛应用的 IEC 104 规约为例,重点描述协议应用层以及应用层与上层用户进程的交互设计。

以异步非阻塞实现电力通信规约,可将其结构分为 TCP/IP 层、应用层和用户进程。TCP/IP 层是实现异步非阻塞通信的关键,基于第 2 节所述的异步非阻塞通信框架建立。在具体的通信过程中,客户端首先按照 2.1 节的客户端通信框架和 2.2 节的服务端通信框架建立底层的 TCP/IP 连接,再通过 2.3 节的非阻塞网络数据读写方法完成应用层以下的数据交互。

在通信进程中,用户进程主要完成维护测点数据和运行控制逻辑两个任务。IEC 104 应用层负责

接收 TCP/IP 层的报文并解析, 以及组装报文并交由 TCP/IP 层发送^[21], 根据 IEC 104 链路传输规则, 实现四遥、总召和对时等应用功能。发送和接收报文需要在应用层和用户进程间建立异步通信通道。异步通信通道能够在线程中以异步的方式完成应用层和用户进程之间的交互。按照 2.3 节的非阻塞方式从 TCP/IP 层接收遥信、遥测等上行报文后, 经过应用层的解析得到测点信息, 通过异步通道交由用户进程进行测点数据维护。需要下达控制指令时, 用户进程通过异步通道下发指令, 经应用层转换成下行报文, 通过 2.3 节的非阻塞方式由 TCP/IP 层下发遥控、遥调等指令。

在应用层内部, 除了需要处理来自用户进程的报文和 TCP/IP 层的网络数据, 还需要运行规约的其他内容, 如超时机制的实现等。利用 EventLoop 结构的 Select 函数实现多个任务的并行, 如图 7 所示。

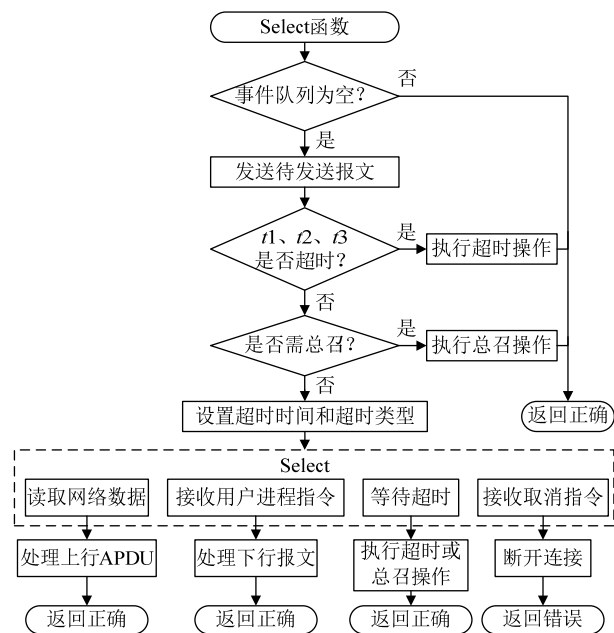


图 7 应用层任务执行流程

Fig. 7 Task execution process of application layer

如图 7 所示, Select 函数的返回值包括正确和错误两类事件。首先检查事件队列是否为空, 若事件队列为空, 则先将队列中的所有事件返回, 然后检查报文发送缓冲区是否为空, 如果仍有未发送的报文, 则立即发送。发送报文后检查 IEC 104 协议中的 $t1$ 、 $t2$ 、 $t3$ 超时时间, 若发生超时则执行对应的超时操作。判断 $t1$ 超时需要记录上一次发送 U 格式测试激活报文的时间, 计算其与当前时间的间隔, 若超时则需要关闭连接, 并返回 $t1$ 超时错误。判断 $t2$ 超时需要记录最先收到且没有确认的报文的时间, 计算其与当前时间的间隔, 若超时则需要发送 S 格式进行确认。判断 $t3$ 超时需要记录上一条

发送报文和接收报文的时间, 找出其与当前时间的间隔更短的一条进行判断, 若超时则发送 U 格式测试激活报文。

基于 Select 函数将应用层与上层用户进程交互、下层网络交互以及应用层自身的超时处理相统一, 用分支的方式进行表达, 该设计能够省去多个任务之间信息共享的麻烦, 提高程序运行效率。

3.2 IEC 104 规约链路传输规则实现

IEC 104 规约采用启/停的传输控制, 需要对此时的状态进行维护, 根据 U 格式报文进行状态的转移^[22]。发送和接收计数器是 IEC 104 规约防止报文丢失和重复的关键。在客户端与服务端建立连接后, 2 个计数器被初始化为 0。每发送一个 I 格式报文, 需要对发送计数器加 1。发送 I 格式报文流程如图 8 所示, 其中参数 K 为规约中未被确认的 I 帧格式 ASDU 的最大数目^[22]。发送计数器的变化发生在报文发送之后, 报文发送前需要判断当前传输状态是否为 Started, 若不处于 Started 状态, 则服务端不能发送遥信和遥测, 不能主动上传数据变化信息。发送报文后还需要记录报文序号和发送时间, 若长时间未被确认, 则有可能产生 $t3$ 超时。

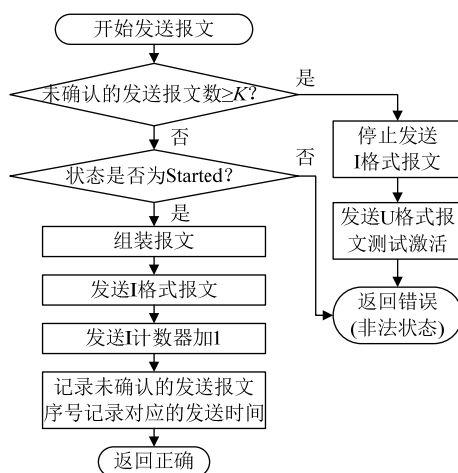


图 8 发送 I 格式报文流程

Fig. 8 Process of sending I format message

每接收到一个 I 格式报文, 需要判断报文的发送序号是否等于接收计数器的值, 如果二者相同说明报文顺序正常; 如果发送序号大于接收计数器, 说明发生了报文丢失; 如果发送序号小于接收计数器, 则是同一个报文被重复发送, 需要丢弃该报文。在报文发送序号正常情况下, 需要累加接受计数器, 并且记录最早一个没有被确认的报文的接收时间, 以供判断 $t2$ 超时。接收到报文后, 还需要根据报文的接收序号确认己方发送报文已被对方接收, 报文序号小于该接收序号的本地发送报文将被确认。处理接收的报文前, 需要判断当

前状态,若未处于 Started 状态,服务端只能被动接收遥控和遥调指令,不能响应总召等命令。

4 测试与应用

4.1 一致性测试

本节对基于电力物联网异步非阻塞通信框架开发的 IEC 104 规约应用软件进行一致性测试^[23]和性能测试,并介绍其在实际工业园区微电网工程中的应用,验证所提技术的有效性。

一致性测试是通过将所设计的应用软件与第三方软件建立连接,进行数据传输,对比测试是否能够实现 IEC 104 规约规定的各项功能。采用 elseta 公司开发的 Vinci 软件中的 IEC 104 仿真功能与所开发应用软件进行 IEC 104 通信模拟,进行一致性测试。

4.1.1 客户端四遥与总召测试

在所开发软件中运行客户端,Vinci 中运行服务端。为便于观测,设定每隔 10 s 进行一次总召。在该次测试中,地址 1—3 为单点遥信,11—13 为浮点数遥测,6001—6003 为单点遥控,分别对应遥信点 1—3,6011—6013 为浮点数遥调,分别对应遥测点 11—13。

首先验证客户端遥信和遥测过程。图 9 展示了在监视过程测试中所开发客户端的运行日志,可以看出,客户端每隔 10 s 进行一次总召,服务端发送所有的测点数据信息。客户端的总召发送间隔符合配置值,日志中显示了客户端发出的总召和接收的遥信和遥测等信息。客户端能够接收并正确解析所有的数据信息,实时性较强。

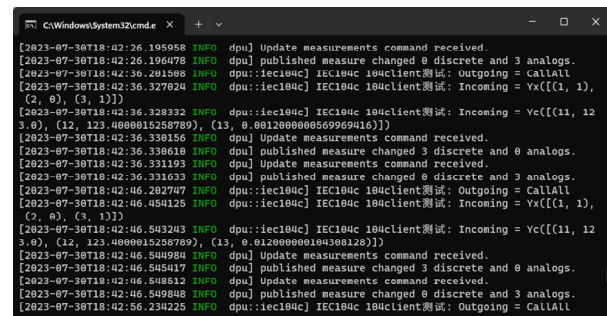


图 9 监视过程中所开发客户端的日志

Fig. 9 Developed client logs of monitoring process

然后验证客户端遥控和遥调过程。客户端对遥控点 6001、遥调点 6011 进行控制,数据变化反映在地址 1 的遥信点和地址 11 的遥测点上。图 10 展示了 Vinci 服务端的报文,所示控制结果表明,所开发客户端能够完成遥控和遥调的选择、执行过程,成功控制单点和浮点数的值。如果没有设置控制点地址,在客户端日志中将给出告警提示。

Time	Source	Ti	Cause	IOA	Value
13:09:13:181		127.0.0.1:65511	C_SC_NA_1 (45)	Pos. Act (6) (T=0 ... 6001	OH[0]
13:09:13:181	Vinci		C_SC_NA_1 (45)	Pos. ActCon (7) (... 6001	OH[0]
13:09:13:182		127.0.0.1:65511	C_SC_NA_1 (45)	Pos. Act (6) (T=0 ... 6001	OH[0]
13:09:13:182	Vinci		C_SC_NA_1 (45)	Pos. ActCon (7) (... 6001	OH[0]
13:09:13:182	Vinci		C_SC_NA_1 (45)	Pos. ActTerm (10... 6001	OH[0]
13:09:14:127	Vinci		M_SP_NA_1 (1)	Spontan (3)	1
13:09:49:600		127.0.0.1:65511	C_SE_NC_1 (50)	Pos. Act (6) (T=0 ... 6011	99.9
13:09:49:600	Vinci		C_SE_NC_1 (50)	Pos. ActCon (7) (... 6011	99.9
13:09:49:601		127.0.0.1:65511	C_SE_NC_1 (50)	Pos. Act (6) (T=0 ... 6011	99.9
13:09:49:601	Vinci		C_SE_NC_1 (50)	Pos. ActCon (7) (... 6011	99.9
13:09:49:601	Vinci		C_SE_NC_1 (50)	Pos. ActTerm (10... 6011	99.9
13:09:50:492	Vinci		M_ME_NC_1 (13)	Spontan (3)	11

图 10 控制过程中的 Vinci 服务端报文

Fig. 10 Vinci server message of control process

4.1.2 服务端四遥与总召测试

在所开发软件中运行服务端,Vinci 运行客户端。利用客户端发送命令,测试服务端是否能够正确响应。服务端需要设定不同的类型标识,在客户端查看服务端发送的是否是正确的数据。在该测试中测点地址设置与客户端测试时相同。图 11 展示了 Vinci 客户端的报文,可以看出,客户端发出的总召、遥控和遥调指令,服务端均能够正确响应。

Time	Source	Ti	Cause	IOA	Value
12:52:48:956	Vinci		C_IC_NA_1 (100)	Pos. Act (6) (T=0 ... 0	
12:52:48:969		127.0.0.1:2404	C_IC_NA_1 (100)	Pos. ActCon (7) (... 0	
12:52:48:969		127.0.0.1:2404	M_SP_NA_1 (1)	Inrogen (20)	1
12:52:48:969		127.0.0.1:2404	M_SP_NA_1 (1)	Inrogen (20)	2
12:52:48:969		127.0.0.1:2404	M_SP_NA_1 (1)	Inrogen (20)	3
12:52:48:969		127.0.0.1:2404	M_ME_NC_1 (13)	Inrogen (20)	11
12:52:48:969		127.0.0.1:2404	M_ME_NC_1 (13)	Inrogen (20)	12
12:52:48:969		127.0.0.1:2404	M_ME_NC_1 (13)	Inrogen (20)	13
12:52:48:969		127.0.0.1:2404	C_IC_NA_1 (100)	Pos. ActTerm (10) ... 0	
12:52:59:543	Vinci		C_SC_NA_1 (45)	Pos. Act (6) (T=0 ... 6001	OH[0]
12:52:59:543		127.0.0.1:2404	C_SC_NA_1 (45)	Pos. ActCon (7) (... 6001	OH[0]
12:53:01:161	Vinci		C_SC_NA_1 (45)	Pos. Act (6) (T=0 ... 6001	OH[0]
12:53:01:161		127.0.0.1:2404	C_SC_NA_1 (45)	Pos. ActCon (7) (... 6001	OH[0]
12:53:01:176		127.0.0.1:2404	C_SC_NA_1 (45)	Pos. ActTerm (10) ... 6001	OH[0]
12:53:01:178		127.0.0.1:2404	M_SP_NA_1 (1)	Spontan (3)	1
12:53:10:995	Vinci		C_SE_NC_1 (50)	Pos. Act (6) (T=0 ... 6011	99.9
12:53:10:995		127.0.0.1:2404	C_SE_NC_1 (50)	Pos. ActCon (7) (... 6011	99.9
12:53:12:309	Vinci		C_SE_NC_1 (50)	Pos. Act (6) (T=0 ... 6011	99.9
12:53:12:309		127.0.0.1:2404	C_SE_NC_1 (50)	Pos. ActCon (7) (... 6011	99.9
12:53:12:317		127.0.0.1:2404	C_SE_NC_1 (50)	Pos. ActTerm (10) ... 6011	99.9
12:53:12:318		127.0.0.1:2404	M_ME_NC_1 (13)	Spontan (3)	11
12:53:46:798	Vinci		C_IC_NA_1 (100)	Pos. Act (6) (T=0 ... 0	

图 11 Vinci 客户端报文

Fig. 11 Vinci client message

图 12 展示了所开发服务端的日志,其中显示了地址 1 和 11 对应测点的遥控和遥调结果,与图 11 中同时刻的报文对应,日志中清楚地写明了上行和下行各个事件。在客户端的控制下,服务端的测点数据能够实时变化。

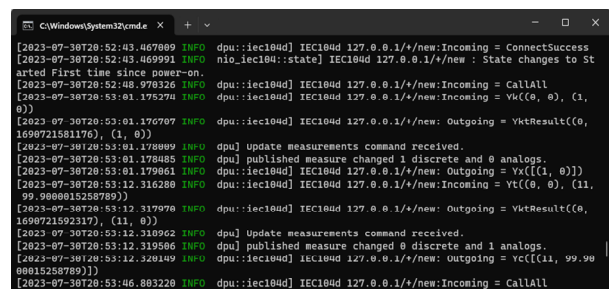


图 12 被控过程中所开发服务端的日志

Fig. 12 Developed server logs of controlled process

4.2 性能测试

性能测试是在大量连接和大量传输的情况下，测试通信的延迟、处理器和内存占用等性能，并与传统的实现方案进行对比。采用 openMUC 开发的 IEC 104 应用软件作为对比的对象，该软件在性能和稳定性上都有一定保证，其基于 Java 语言和传统阻塞式 I/O 技术开发。压力测试在 2 台计算机之间进行，其中 1 台为客户端，另外 1 台为服务端，处理器型号均为 Intel(R) Core(TM) i5-8250U，基准速度 1.80 GHz，内存总量为 23.6 GB。在局域网内进行通信，内部带宽 1000 M，采用 CAT6 类网线。服务端连接数设置为 100，客户端主动与服务端连接并进行总召。服务端设定数据量为 1000 个单点遥信，1000 个浮点数遥测，保证每次总召的报文发送量较大。改变总召的发送间隔，发送间隔越小，通道可能出现的延迟就越大，通信的压力就越大，越能接近极限性能。

在不同总召发送间隔下，测量本文所开发服务端和 openMUC 的软件服务端运行时的处理器和内存占用。总召发送间隔分别为 500、300、200、150 和 100 ms，测试结果如表 1 所示。

表 1 阻塞与非阻塞通信实现方式的 CPU 和内存占用
Table 1 CPU and memory occupation of blocking and non-blocking communication

总召发送间隔/ms	CPU 利用率(以 500 ms 发送间隔结果为基准)		内存占用/MB	
	非阻塞	阻塞	非阻塞	阻塞
500	1	1	22.1	423.7
300	1.72	1.62	22.1	423.7
200	2.32	2.31	22.1	811.1
150	3.05	2.76	23.1	821.5
100	3.31	2.99	42.6	826.4

由表 1 可看出，随着客户端总召发送间隔减小，2 种实现方式下的 CPU 利用率和内存占用不断增加，其中 Java 阻塞式通信的性能极限约为 200 ms 以内，而本文异步非阻塞式通信的性能极限可达 150 ms 以内。在没有到达通信性能极限时，随着发送数据间隔缩小，CPU 利用率保持线性增长趋势，内存占用基本保持不变，且本文异步非阻塞式通信的内存占用远小于传统阻塞通信方式，仅为其 5.2%。CPU 利用率增加是由于单位时间内处理的数据变多，内存占用基本不变的原因是没有到达通信极限时数据没有形成堆积。在到达通信性能极限之后，随着发送间隔缩小，CPU 利用率增长速度放缓，而内存占用增长迅速。CPU 利用率增长速度放缓是因为性能逐渐接近极限，单位时间内能够处理的数据量无法持续线性增长，此时本文异步非阻塞式通信下的 CPU 利用率高于传统通信方式，在较高通信压力下能够充分发挥 CPU 的性能；内存占用增加迅速是由于出现大量未能及时处理的数据，在内存中形成数据堆积，本文异步非阻塞式通信的极限性能相比传统通信方式更高，因此出现内存数据堆积的数据发送间隔更短，且对内存的占用远小于传统阻塞通信方式。

进一步测量不同数据发送间隔下的通信延迟^[24]。在客户端发送一定次数的总召后立即停止，并利用发送计数器和接收计数器记录此时 100 个客户端与服务端收发的报文总量。若客户端收到的报文少于服务端发出的报文，说明出现了通信延迟。分别设置总召次数为 200、400、600、800，进行多次测试，结果如表 2 和表 3 所示。

表 2 非阻塞通信实现方式的通信延迟
Table 2 Communication delay of non-blocking communication implementation

总召发送间隔/ms	报文数量				延迟率/%	单通道吞吐量/(数据量/s)
	总召次数 200	总召次数 400	总召次数 600	总召次数 800		
500	70 000	1 400 000	2 100 000	2 800 000	0	20 000
300	70 000	1 400 000	2 100 000	2 800 000	0	20 000
200	70 000	1 400 000	2 100 000	2 800 000	0	20 000
150	699 857	1 399 689	2 096 511	2 796 974	0.108 07	19 978.4
100	651 800	1 278 506	1 939 873	2 621 071	6.390 32	18 721.9

表 3 阻塞通信实现方式的通信延迟
Table 3 Communication delay of blocking communication implementation

总召发送间隔/ms	报文数量				延迟率/%	单通道吞吐量/(数据量/s)
	总召次数 200	总召次数 400	总召次数 600	总召次数 800		
500	70 000	1 400 000	2 100 000	2 800 000	0	20 000
300	699 492	1 399 980	2 099 794	2 799 789	0.007 54	19 998.5
200	699 302	1 399 261	2 099 407	2 799 490	0.018 21	19 996.4
150	653 625	1 309 484	1 970 652	2 630 362	6.058 5	18 788.3
100	430 212	870 387	1 291 111	1 734 474	38.054 5	12 389.1

表 2 和表 3 中展示了总召次数为 800 次时的通信延迟率和单通道吞吐量测试结果, 通过对比可看出, 在 500 ms 的总召发送间隔下, 2 种实现方式都能完成所有报文的传输, 随着发送间隔缩短, 本文非阻塞通信能够成功发送的报文量显著大于传统阻塞式通信, 同时通信延迟率更小。在 100 ms 的发送间隔下, 本文异步非阻塞式通信的延迟率为 6.39%, 远低于传统阻塞式通信的 38.05%, 且非阻塞通信的吞吐量约为阻塞式通信的 1.5 倍。

综上所述, 本文非阻塞通信实现方式在压力测试中相比传统阻塞式通信具有更好的性能, 能够充分利用 CPU 资源, 且占用内存远小于传统实现方式, 通信延迟率更低、吞吐量更高, 能更好适用于电力物联网硬件资源受限条件下的可靠通信。

4.3 实际工程应用

本文基于异步非阻塞框架的电力物联网通信技术已在杭州萧山欣美电气公司工业园区微电网中得到实际工程应用。欣美电气作为率先打造浙江省内首个“零碳”智慧工厂的企业^[25], 其园区内目前已建成 3.15 MW 分布式光伏发电、150 kW/425 kW·h 储能装置, 形成了光充储一体化的园区微电网系统, 实现了园区清洁能源渗透率达 80%, 2022 年获得浙江省首张企业“碳中和”证书。

基于本文技术开发的 IEC 104 应用软件已部署在欣美电气的林洋光伏平台电力物联系统上, 实现了光伏并网逆变器和电能表的数据采集, 为搭建园区电力物联网和综合能源管理平台提供了技术支撑。对光伏并网逆变器的并网状态、有功功率、发电量等遥信和遥测信息进行监视, 在所开发软件中运行客户端, 能够实现与光伏服务端之间的实时通信。通过灵活设定总召时间间隔, 实现了对 12 个光伏并网逆变器和电能表数据的实时观测, 并且能够对异常数据进行告警。图 13 展示了客户端后台日志, 图 14 展示了客户端前端界面的实时数据显示情况。目前, 园区电力物联系统稳定运行时长已超过 15 个月, 采集量测数据量达到 200 万次以上, 验证了所提电力物联网通信技术的有效性。

```
2022-04-27T02:12:12Z INFO dpu: published measure changed 0 discrete and 1 analogs.
2022-04-27T02:12:12Z INFO dpu::iec104c IEC104c 光伏一期逆变器:Incoming = Yc{[(16397, 33.0899983215332)]}
2022-04-27T02:12:12Z INFO dpu: Update measurements command received.
2022-04-27T02:12:12Z INFO dpu::iec104c IEC104c 光伏一期逆变器:Incoming = Yc{[(16395, 33.52999877926685)]}
2022-04-27T02:12:12Z INFO dpu: Update measurements command received.
2022-04-27T02:12:12Z INFO dpu::iec104c IEC104c 光伏一期逆变器:Incoming = Yc{[(16399, 34.29999922706855)]}
2022-04-27T02:12:12Z INFO dpu: Update measurements command received.
2022-04-27T02:12:12Z INFO dpu::iec104c IEC104c 光伏一期逆变器:Incoming = Yc{[(16385, 33.9900016784668)]}
2022-04-27T02:12:12Z INFO dpu: Update measurements command received.
2022-04-27T02:12:12Z INFO dpu::iec104c IEC104c 光伏一期逆变器:Incoming = Yc{[(16401, 31.950000762939453)]}
```

图 13 实地运行的客户端运行日志

Fig. 13 Client operation log of field operation

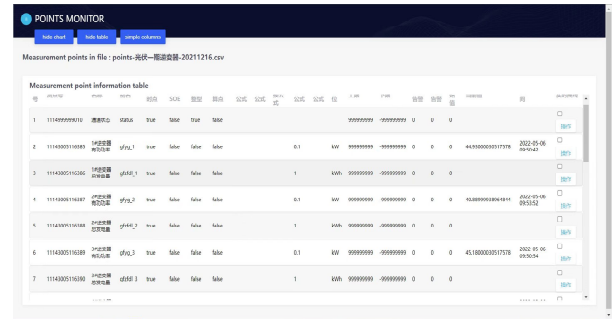


图 14 客户端前端数据界面

Fig. 14 Front-end data interface of the client

5 结语

本文根据异步非阻塞技术原理提出一种基于异步非阻塞框架的电力物联网通信技术, 并将其用于 IEC 104 规约实现, 通过与传统阻塞式电力通信技术的性能对比和实际工程应用可得到如下结论。

1) 基于所提技术开发的 IEC 104 应用软件能够完成与第三方软件的连接并进行四遥和总召等数据交互, 能够完整实现通信规约规定功能。

2) 与传统电力通信技术相比, 所提异步非阻塞的电力物联网通信技术具有内存占用更小、硬件计算资源利用更充分、通信延迟更低、数据吞吐量更大的优势, 能够更好地适用于电力物联网中硬件资源受限条件下的数据通信。

3) 基于所提技术开发的 IEC 104 应用软件能够在实际工业园区电力物联网中长期稳定运行, 表明所提技术具有良好的可靠性和稳定性。

电力物联网中多样化的终端设备接入将涉及多种类型工业通信协议, 进一步将所提通信技术用于高效地实现 IEC 61850、Modbus、DL/T 645、MQTT 等典型协议, 并将各类协议集成于电力物联网网关软件, 是本文的后续研究内容。

参考文献

- [1] WANG Siyuan, WU Wenchuan. Aggregate flexibility of virtual power plants with temporal coupling constraints[J]. IEEE Transactions on Smart Grid, 2021, 12(6): 5043-5051.
- [2] 孙宏斌, 郭庆来, 吴文传, 等. 面向能源互联网的多能流综合能量管理系统: 设计与应用[J]. 电力系统自动化, 2019, 43(12): 122-128, 171.
SUN Hongbin, GUO Qinglai, WU Wenchuan, et al. Integrated energy management system with multi-energy flow for energy internet: design and application[J]. Automation of Electric Power Systems, 2019, 43(12): 122-128, 171(in Chinese).
- [3] 刘林, 祁兵, 李彬, 等. 面向电力物联网新业务的电力通信网需求及发展趋势[J]. 电网技术, 2020, 44(8): 3114-3130.
LIU Lin, QI Bing, LI Bin, et al. Requirements and developing trends of electric power communication network for new services in electric internet of things[J]. Power System Technology, 2020, 44(8): 3114-3130(in Chinese).

- [4] 吕继伟. 基于泛在电力物联网的换流站在线监测系统优化综述[J]. 电力工程技术, 2019, 38(6): 9-15.
LÜ Jiwei. Optimization survey of online monitoring system for converter station based on ubiquitous power IoT[J]. Electric Power Engineering Technology, 2019, 38(6): 9-15(in Chinese).
- [5] 叶键民, 何光宇, 梅生伟, 等. 动态 SCADA 原型系统构建及其通信系统的实现方法[J]. 电工电能新技术, 2014, 33(1): 16-20.
YE Jianmin, HE Guangyu, MEI Shengwei, et al. Dynamic SCADA prototype system construction and implementation of communication system[J]. Advanced Technology of Electrical Engineering and Energy, 2014, 33(1): 16-20(in Chinese).
- [6] 段楠. 异步非阻塞网络通讯技术研究[J]. 现代计算机, 2019(17): 79-82.
DUAN Nan. Research on asynchronous non-blocking network communication technology[J]. Modern Computer, 2019(17): 79-82(in Chinese).
- [7] GUO Junxia, LI Zheng, SHI Cunfeng, et al. Thread scheduling sequence generation based on all synchronization pair coverage criteria[J]. International Journal of Software Engineering and Knowledge Engineering, 2020, 30(1): 97-118.
- [8] GRIFFIN L, RYAN K, DE LEASTAR E, et al. Scaling instant messaging communication services: a comparison of blocking and non-blocking techniques[C]//2011 IEEE Symposium on Computers and Communications (ISCC). Kerkira, Greece: IEEE, 2011: 550-557.
- [9] ARTHO C, HAGIYA M, POTTER R, et al. Software model checking for distributed systems with selector-based, non-blocking communication[C]//2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE). Silicon Valley, USA: IEEE, 2013: 169-179.
- [10] 牛聚川. 融合模糊测试和形式化验证的 RUST 测试工具研究[D]. 长春: 吉林大学, 2022.
- [11] SYDOW S, NABELSEE M, GLESNER S, et al. Towards profile-guided optimization for safe and efficient parallel stream processing in rust[C]//2020 IEEE 32nd International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD). Porto, Portugal: IEEE, 2020: 289-296.
- [12] MUSIL P, MLYNEK P. Overview of communication scenarios for IEC 60870-5-104 substation model[C]//2020 21st International Scientific Conference on Electric Power Engineering (EPE). Prague, Czech Republic: IEEE, 2020: 1-4.
- [13] 郭创新, 董树锋, 张金江. 电力信息技术[M]. 北京: 科学出版社, 2015: 140-143.
- [14] 黄海军, 汤嘉立, 冯新翎. 数字化变电站数据通信不确定性阻塞仿真分析[J]. 计算机仿真, 2019, 36(9): 194-197, 287.
HUANG Haijun, TANG Jiali, FENG Xinling. Simulation analysis of data communication uncertainty blocking in digital substation[J]. Computer Simulation, 2019, 36(9): 194-197, 287(in Chinese).
- [15] 齐四清, 刘世民. 多线程通信互连网络阻塞疏导模型构建[J]. 舰船科学技术, 2020, 42(22): 133-135.
QI Siqing, LIU Shimin. Construction of congestion grooming model for multi thread communication interconnection network[J]. Ship Science and Technology, 2020, 42(22): 133-135(in Chinese).
- [16] 杨兰兰, 郭建. 非阻塞算法的形式化建模与分析[J]. 计算机应用, 2020, 40(S2): 106-111.
YANG Lanlan, GUO Jian. Formal modeling and analysis of non-blocking algorithm[J]. Journal of Computer Applications, 2020, 40(S2): 106-111(in Chinese).
- [17] 万立超. 高并发环境下微服务网关的研究与设计[D]. 南昌: 南昌大学, 2022.
- [18] MILANESI C. Creative projects for rust programmers: build exciting projects on domains such as web apps, webAssembly, games, and parsing[M]. Birmingham, UK: Packt Publishing Limited, 2020: 1-6.
- [19] 吴路明, 裘愉涛, 陈琦. 基于 SDN 的电力通信网络关键技术综述[J]. 电力工程技术, 2018, 37(3): 134-144.
WU Luming, QIU Yutao, CHEN Qi. The critical technology of the electric power telecommunication based on SDN[J]. Electric Power Engineering Technology, 2018, 37(3): 134-144(in Chinese).
- [20] 阮黎翔, 沈奕菲, 王自成, 等. 基于角色的访问控制在变电站网络安全中的应用[J]. 浙江电力, 2022, 41(7): 86-93.
RUAN Lixiang, SHEN Yifei, WANG Zhicheng, et al. Application of role-based access control in cyber security of substation[J]. Zhejiang Electric Power, 2022, 41(7): 86-93(in Chinese).
- [21] 朱海兵, 吴奕, 陈宁, 等. 提高远程遥控安全性的 IEC 60870-5-104 规约扩展研究[J]. 江苏电机工程, 2014, 33(6): 51-54.
ZHU Haibing, WU Yi, CHEN Ning, et al. Extension research on IEC 60870-5-104 to improve the security of remote control[J]. Jiangsu Electrical Engineering, 2014, 33(6): 51-54(in Chinese).
- [22] 国家能源局. 远动设备及系统 第 5-104 部分: 传输规约 采用标准传输协议集的 IEC60870-5-101 网络访问: DL/T 634.5104—2009[S]. 北京: 中国标准出版社, 2009.
- [23] 郑明忠, 卜强生, 高磊, 等. 面向服务端的变电站自主可控站控层服务协议一致性测试技术[J]. 电力自动化设备, 2022, 42(11): 211-217.
ZHENG Mingzhong, BU Qiangsheng, GAO Lei, et al. Consistency testing technology of autonomous and controllable substation control level service protocol for service-oriented substations[J]. Electric Power Automation Equipment, 2022, 42(11): 211-217(in Chinese).
- [24] 赵瑞玉, 李文娟. 移动通信延迟对信息并行传输效率影响仿真[J]. 计算机仿真, 2020, 37(4): 192-195.
ZHAO Ruiyu, LI Wenjuan. Simulation of the influence of mobile communication delay on information parallel transmission efficiency[J]. Computer Simulation, 2020, 37(4): 192-195(in Chinese).
- [25] 张梦月. “双碳大脑” 双控有招[N]. 浙江日报, 2021-10-12(01).



吴振冲

收稿日期: 2023-05-14.

作者简介:

吴振冲(2000), 男, 博士研究生, 研究方向为电力系统通信与优化控制技术;

徐成司(1995), 男, 工程师, 研究方向为电力系统通信、控制与仿真技术;

董树锋(1982), 男, 博士, 副教授, 通信作者, 研究方向为电力信息技术、高性能计算, dongshufeng@zju.edu.cn.

(责任编辑 张京娜)