

基于公共网关接口的智能交通信号机控制系统

胡媛^{1†}, 胡涛²

(1. 湖南力唯中天科技发展有限公司, 湖南 长沙 410000; 2. 长沙智能驾驶研究院有限公司, 湖南 长沙 410000)

摘要:为提高智能交通信号机的远程控制效率,设计了一个基于公共网关接口(CGI, Common Gateway Interface)的信号机控制系统。系统硬件设计上实现了云平台层、中央控制系统层、扩展单元层的多层系统架构,软件设计上以 CGI 技术为基础来形成多个功能模块的控制体系,云平台通过以太网直接调用中央控制系统上的 CGI 模块完成控制数据的处理与转发,通信协议则以 JavaScript 对象表示法(JSON, JavaScript Object Notation)定义。然后建立共享内存主题模型来解决系统内存与 CGI 模块的数据交互,共享内存空间出现数据变化将会唤醒订阅该主题的系统进程及时更新内存数据,最后系统通过 CAN 通信对扩展单元的灯色进行控制。

关键词:公共网关接口;JavaScript 对象表示法;共享内存主题模型;远程控制
中图分类号:TP273.5 **文献标识码:**A

Intelligent Traffic Signal Machine Control System Based on Common Gateway Interface

HU Yuan^{1†}, HU Tao²

(1. Hunan Livesky Science and Technology Development Co., Ltd., Changsha, Hunan 410000, China;
2. Changsha Intelligent Driving Institute Co., Ltd., Changsha, Hunan 410000, China)

Abstract: In order to improve the remote control efficiency of intelligent traffic signal machine, a novel control system based on common gateway interface(CGI) was presented. The multi-layer system architecture of cloud platform layer, central control system layer, expansion unit layer for the system hardware was designed. In the software design, different functional modules formed the control framework based on CGI technology that could be called by the cloud platform for data processing and forwarding, while the communication protocol was defined by JavaScript object notation(JSON). Then, to address the data exchange challenge between the CGI module and system memory, a shared memory topic model was established, any data changes in the shared memory space would wake up the processes that subscribe to the topic to update memory data, finally the system controlled the expansion light unit by CAN communication.

Key words: common gateway interface(CGI); JavaScript object notation(JSON); shared memory topic model; remote control

近年来,汽车等交通工具呈爆炸式增长,为了解决交通拥堵问题,智能交通系统应运而生,而远程控制又是智能交通系统的核心内容。目前拥有较高智能化的智能交通信号机^[1]正在替代老旧的

只能执行固定单一方案的单点控制信号机^[2],与之相对应的各大城市正在或者已经建成了交通管理中心来统一控制各个厂家的智能交通信号机,实现了简单的远程控制功能和部分高级运行策略,例如

绿波控制^[3]、区域协调控制^[4]、诱导控制^[5]、自适应控制等^[6]。这类信号机系统远程控制的响应速度问题也将会逐渐凸显,目前国内的交通远程控制系统在实际应用中为了兼顾各个厂家的交通信号机,交通统一信号控制(以下简称信控)平台被研发出来对接各厂家的信控云平台进行统一协调控制,导致了控制响应延时较为明显,这必然会降低系统的性能,因此远程控制效率成了亟须需解决的关键问题。Opoku 等^[7]采用 FPGA 控制器与交通管理中心进行通信,以及基于云的应用程序进行远程控制,其应用程序的维护与开发成本较高,并且最终仍然采用传统的 TCP/IP 协议与设备进行通信,仍然存在响应延时问题。Zhang 等^[8]提出使用移动端来进行远程控制具有很高的灵活性,但是移动通信由于受到信号衰减、干扰和安全性的影响,传输距离通常较短,不适用大城市的交通管理中心统一控制。Yang 等^[9]提出的信号机系统利用边缘计算的方式统计流量实时数据,远程控制则是基于传统的 UDP 应用协议,实际仍然存在控制延时较高的问题。与国内的统一交通远程控制技术不同,国外普遍的交通信号控制技术多为研究的自适应控制,如 Jamil 等^[10]提出的深度强化学习方法解决信号灯运行策略的配时问题,Boskany^[11]提出的机器学习技术与图像识别技术判断特殊车辆的方法优化红绿灯控制,Suneetha 等^[12]提出的射频传感器处理特种车辆同行优化红绿灯等,其原因主要为受限于城市没有建设交通管理中心来对信号机进行统一管理,也与欧美国家不能架设摄像头监控实时交通状况有关。

CGI 技术是一种重要的互联网技术,可以让一个客户端从网页浏览器向执行在网络服务器上的程序请求数据,具有适用性强,兼容性好,扩展性强,人机交互效果优异的特点。基于上述优势,本文将 CGI 技术与嵌入式系统相结合,主要采用 ARM9 系列嵌入式处理器、云平台服务器、Web Server Boa^[13]实现一种智能交通信号机控制系统。结合系统需求确立多层整体架构模式,利用 CGI 技术确立多模块控制架构,将数据处理进行边缘部署,实现采集数据、处理数据、上报数据的一体化,增强控制效率。

1 系统原理设计

以大城市的交通管理中心的远程控制为前提,信号机控制系统总体架构由远程信控云服务器、信

号机中央控制系统、信号机外设灯控单元、外设数据采集单元组成,每层单元均有各自的职责,其看似独立实则协同工作,同时也方便后期功能扩展与功能更改。在多层硬件架构模式下,下层的外设灯控单元与采集单元主要负责灯色开关量的直接执行与数据量、模拟量采集,中层的中央处理器主要负责逻辑控制算法与数据处理,上层云平台主要负责对接交通管理中心作为数据中转站和数据汇总处理。软件框架设计上利用 CGI 技术根据交通控制参数的控制类型进行分模块编程,模块相互独立,同样方便后期功能扩展与更新,且上层与中层都可进行前端展示。与 Huang^[14]提出通过客户端调用 CGI 在浏览器上实现简单的控制功能不同,本文可以使用信控云平台直接调用 CGI 来形成信控平台端对信号机端的通信机制,结合了 CGI 技术与应用层协议,采用了平台处理与传输均高效的 JSON^[15]格式作为信息交互媒介。而在信号机控制系统与 CGI 各个模块之间,建立了高效的通信机制,根据本系统实际情况改进了 Chen 等^[16]提出的共享内存主题模型,共享内存空间数据出现变化时将会唤醒订阅该主题的进程更新控制系统内存数据。最终,数据的执行与采集通过中央处理器传递到外设控制单元、外设数据采集单元完成,实现信号机的远程控制功能。如图 1 所示的是远程控制系统框架,省略了与远程控制无关的部分硬件设备。

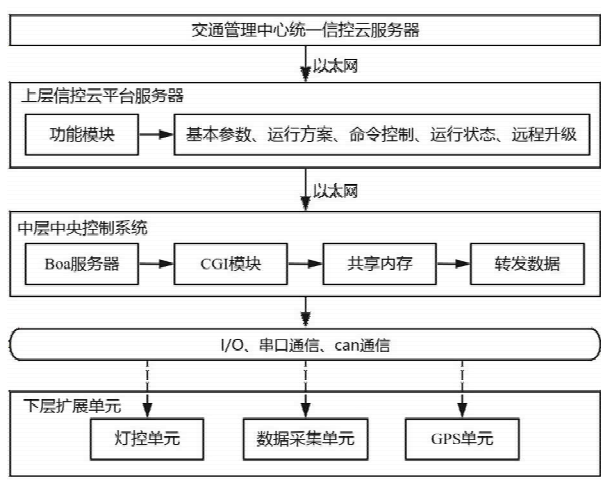


图 1 远程控制系统框架

2 系统硬件框架

2.1 中央控制器

系统选择了 AM335x 作为主控单元,该主控单元是 ARM 公司的 TI ARM Cortex-A8 处理器

核,其具有高性能低功耗的优点,并配备了丰富的外设接口,引出双路千兆网口、LCD、HDMI、GPMC、CAN 等接口,非常适合应用在智能交通信号机系统上,核心电路控制器的构成框图如图 2 所示。

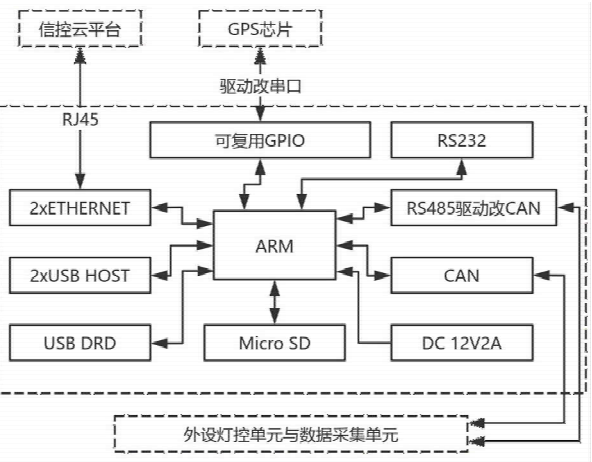


图 2 核心控制电路的框图

2.2 以太网接口模块

AM335x 拥有 2 路以太网接口电路,主要用于交通信号机的远程控制通信,系统使用的是 Realtek 的 RTL8211F 芯片,支持 10Base-T、100Base-TX 和 1000Base-T IEEE802.3 标准,拥有电源控制模块、时钟控制模块、PHY 模块、MAC 模块、接口模块,其 PHY 模块的唤醒功能很大程度上降低了功耗,在休眠状态下可以通过唤醒引脚 WOL (Wake-On-LAN)接收到外部唤醒信号。接口模块则通过隔离变压器与 RJ45 以太网接口进行连接,实现了外部电路与内部电路的电气隔离,使系统的抗干扰能力提高。

2.3 串口模块与外设 GPS 单元

AM335x 拥有四路独立的串行接口,每个串口都可以独立地以终端模式或者 DMA 模式完成 CPU 与 UART 之间的数据传输。其中 UART0 作为默认调试口,为了完成交通信号机与 GPS 通信,UART1 被用作与 GPS 芯片进行数据交互的接口,功能主要包含校时、秒脉冲、定位。GPS 芯片则采用 CASIC 的 ATGM332D 北斗定位模块,具有高灵敏度、低功耗、小尺寸等特点,十分适用于交通信号机系统功能。

2.4 CAN 模块与外设灯控单元、数据采集单元

传统的信号机采用多片 8255 扩展 I/O 口,使得系统较为庞大、成本也相应增加,控制逻辑也非常复杂。AM335x 拥有一路 CAN 接口电路和一路 RS485 电路,本文设计的交通信号机系统采用嵌入式 STM32F103 作为灯控单元与数据采集单

元的核心,灯控单元扩展控制了 128 路 I\O 口,而只占用了 AM335x 的一路 CAN 通信,另一路将 RS485 功能更改为 CAN 功能,用作与数据采集单元的数据交换。因此使用两路 CAN 通信就完成了下层大部分的数据交换,用嵌入式的方式可靠性强,控制逻辑简单,同时也能很好地监测扩展单元的实时数据,保障路口交通安全。

3 系统软件框架

3.1 信号机服务器平台构建

Boa 服务器是一款微型高效的开源 Web 服务器,与其他传统的 Web 服务器不同的是当有连接请求到来时,它并不为每个连接单独创建进程,也不通过复制自身进程来处理多链接,而是通过建立 HTTP 请求列表来处理 HTTP 连接请求,同时它只为公共网关接口程序创建新的进程,这样就在最大程度上节省了系统资源,因此该服务器在不占用信号机性能的情况下也具有极快的响应速度,非常适合在信号机这类嵌入式设备上使用。移植步骤如下:

步骤 1:优化 Boa 服务器配置。

根据本系统的特点,对 Boa 服务器作适当的修改,来提高系统性能。优化内容主要包括:

1)增大输入输出缓存 BUFFER_SIZE 的值,本文使用经验值 10 k,减少大数据循环发送与接收次数;

2)优化服务器可以处理的 HTTP 连接池 KeepAliveMax 参数调整为 10,对于本系统访问量较小的局域网信控服务器是足够的,减轻服务器负载压力;

3)更改服务器地址 SERVER_ROOT 为交通信号机系统指定目录/usr/system/elf/boa,方便统一管理与执行。

4)优化服务器连接超时时间 KeepAliveTimeOut 参数调整为 60 s,对于本系统该时间内足够能完成基本命令操作,不会新增重复连接消耗性能。

步骤 2:交叉编译与移植。

根据核心设备为 ARM9 系列处理器,更改 makefile 的 gcc 编译链接为交叉编译:arm-linux-gnueabi-hf-gcc,

通过执行:

```
./configure--host = arm-linux CC = arm-linux-gnueabi-hf-gcc
make
```

便可生成 elf 格式的可执行文件 boab,然后移植到设备目录/usr/system/elf/boab。

Boa 服务器的配置文件为 boab.conf,更改 ServerName 为本机 IP 即可,同样移植到设备目录/usr/system/elf/boab。

3.2 公共网关接口 CGI 编程算法与扩展

公共网关接口 CGI 是一个为 Web 服务器提供信息服务的标准接口。由于智能交通信号远程控制系 统拥有多个平台的特殊性,无法通过客户端调用 ARM 系统上的 CGI 来实现控制功能,本文使用信控平台直接调用 CGI 获取信号机数据,然后将数据再传递给统一信控平台,最后客户端通过统一信控平台来实现命令控制功能。CGI 程序主要通过环境变量和标准输入输出与信控平台进行通信,当 HTTP 请求模式采用 POST 方式时,CGI 程序通过标准输入流和有关环境变量来获取信控平台传输数据;采用 GET 方式时,CGI 程序直接通过标准输出流将该结果数据传递给信控平台。本文定义道路交通信号的数据流格式为 JSON 格式,该格式有着最为高效的传输效率并且能够使用内置的 JSON 解析器来快速的转换成数组和对象,从而提高平台性能。以控制通道为例,其数据流格式描述如下:

"channelList": [{ "channelNo": x, "channel-Type": x, " lampBoardNo": x, " direction": x, " channelTurn": x, " countdownMode": x, " control-Oper": x, " remark": "东向南左转"}]。括号内对应内容为:通道号、通道类型、所属灯控板序号、进口方向、通道转向、倒计时模式、控制类型、通道描述,其中 x 表示数值。

因为 Boa 服务器本身不具备鉴权功能,本文为了数据安全性,在每一段数据中增加动态校验密码,信控平台端与交通信号机端均采用带时间参数的随机密码,每次进行数据访问时先校验密码再进行后续处理。在数据完整性校验后,数据最终都以 JSON 格式存储。算法步骤如下:

步骤 1:首行声明,选择数据类型为 JSON 格式以及打开跨域权限。

```
fprintf ( stdout, " Content-Type: application/
json\r\n");
```

```
fprintf ( stdout, " Access-Control-Allow-Ori-
gin: * \r\n\r\n");
```

步骤 2:获取环境变量,类型变量存储在 method 中,检查其 HTTP 请求类型为 GET 或者 POST 还是其他类型等等。

```
method=getenv("REQUEST_METHOD");
if(! memcmp(method,"POST",4))
else if(! memcmp(method,"GET",3))
```

步骤 3:数据的密钥校验,以当前时间为动态密码的输入参数,校验失败即反馈给信控云平台并退出。

步骤 4:数据的完整性校验,校验失败即反馈给信控云平台并退出。

步骤 5:数据的存储与读取,最终各项参数都以 JSON 格式存储与传递。

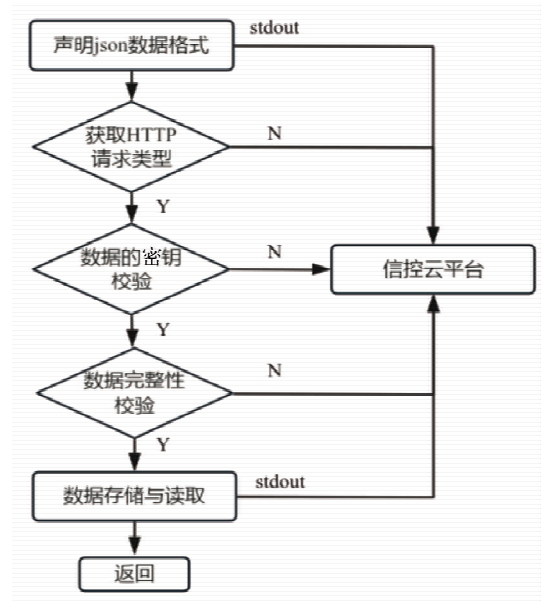


图 3 CGI 编程算法流程图

3.3 共享内存主题模型

3.3.1 共享内存空间的组织结构

CGI 程序与本地控制系统分属不同进程,且由多个模块构成,为解决多进程之间的数据交换问题,对属于 CGI 模块的控制数据建立主题模型,关键词代表的是控制数据的具体类型,一系列词的集合构成控制模块的主题。假设在多进程间通信机制中,系统根据模块数量申请 T 块共享内存空间,共享内存空间又由多个共享内存块组成,在每个共享内存块会保存特定关键词消息的数据,如式(1)所示,则共享内存空间 D_i 被建模为 T 个主题的集合:

$$D_i = \{Z_1, Z_2, \cdots, Z_i, \cdots, Z_T\},$$
$$Z_i \in \{1, 2, \cdots, T\}$$
(1)

式中, Z_i 表示共享内存空间的主题之一。而共享内存空间中的共享内存块都有一个关键词,每个主题 Z_i 又包括 N 个关键词:

$$Z_i = \{w_1, w_2, \cdots, w_n, \cdots, w_N\},$$

$w_n \in \{1,2,\cdots,N\}$ (2)

式中, w_n 是主题 Z_i 中关键词之一。

传递数据步骤:

步骤 1:由发送方找到需要传递的数据所包含的主题 $data_i \sim Z_i$ 以及数据所包含的关键词 $data_n \sim w_n$ 。

步骤 2:根据已知的主题集合 D_i 找到对应主题 Z_i 的共享内存空间。

步骤 3:将 $data_n$ 在主题 Z_i 的共享内存空间中按照关键词 w_n 依次拷贝共享内存块。

步骤 4:本地控制系统监测进程查询到共享内存空间的数据变化。

步骤 5:唤醒所有订阅该消息主题的进程,拷贝对应共享内存空间数据到本地控制进程内存中。

多次重复上述步骤即可完成数据传递。图 4 是共享内存主题模型的表现形式。

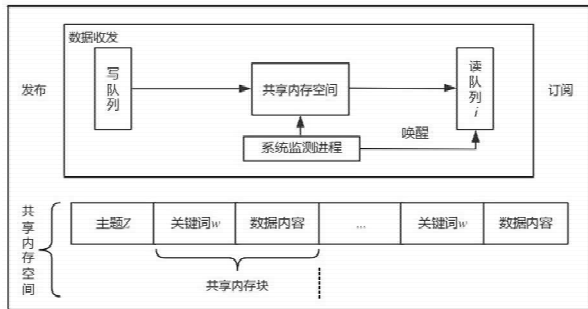


图 4 共享内存主题模型

3.3.2 共享内存块的读写机制

本文采用 Fifo(First Input First Output)环形队列管理共享内存块,在数据传输过程中,发送方每次将数据拷贝到空闲的内存块中,而接收方每次从保存数据的内存块中获取数据。本机制中收发双方无需建立同步机制,发送方发送数据时,从写环形队列中获取一个写指针,根据该指针,发送方将要发送的数据拷贝到剩余内存块中,发送方指针将进行偏移,而接收方同步此写指针偏移。每个接收方依次遍历自己的读环形队列,根据每个队列项访问特定的内存块中的数据。当接收方获取一个内存块内的数据后,通过读写指针的偏移计算判断是否将该内存块的数据读出,如果是,则将读指针更新;反之,则继续访问下一个队列项。Fifo 环形队列结构体包含输入、输出 Buff,内存块大小 DataSize,读写标签 WriteFlag 与 ReadFlag,主题 Topic 与关键词 Word。通过读写环形队列的方式,从 Buff 输入到 MirrorBuff 输出。如图 5 所示,通过对环形队列的交叉操作,屏蔽了共享内存空间

的物理结构,实现了收发双方数据的异步传输,降低了模块间的耦合。

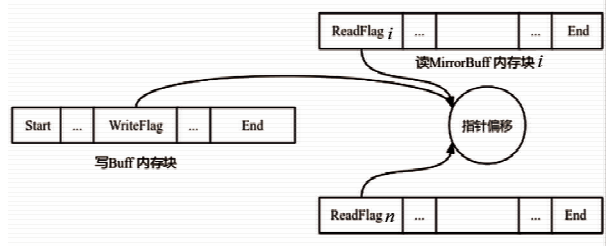


图 5 共享内存块数据读写队列模型

3.4 控制系统主程序设计

根据 CGI 程序编码的相对独立特性,本系统将控制逻辑分为五个模块,分别是基本参数模块、运行方案参数模块、命令控制模块、运行状态模块、远程升级模块。基本参数模块负责信号机设备的必要参数,如 IP、灯色控制板等配置信息;运行方案参数模块负责信号灯的运行逻辑,如使用不同的配时方案分时段来引导交通流动,比较典型的是在主干道多个路口的信号灯控制中,通过时段表设定相同的时段方案执行无缆线协调控制,以使主干道上的车流依次到达前方各交叉口时,均会遇上绿灯,实现绿波交通,也可以提前设定某些特殊的时段方案应对重大活动等特殊情况;命令控制模块负责在出现紧急情况需要远程实现路口控制时进行临时控制,如步进控制、全红控制等;运行状态模块负责监测设备运行数据以及故障数据;远程升级模块负责信号机的迭代更新,包含 CGI 模块更新与主程序更新。在数据的交互上,这五个模块相互独立,信控云平台可以根据实际控制逻辑单独调用,因此各个模块也可以单独进行升级。图 6 为软件总体设计主流程图,主程序远程控制流程如下:

步骤 1:交通管理中心下发命令。

步骤 2:统一信控云平台接收命令,解析、处理并转发。

步骤 3:信控云平台接收命令,解析、处理并转发调用对应信号机的 CGI 模块。

步骤 4:CGI 模块接收命令,解析、处理后,存放入对应的共享内存空间中。

步骤 5:信号机常驻进程检查到共享内存空间的数据变化后,将其存放入控制系统的内存中,通过内部算法将其转化成需要执行的实时灯色控制流。

步骤 6:核心控制系统与外设灯色控制板则通过 CAN 通信进行实时数据交互完成信号灯控制。

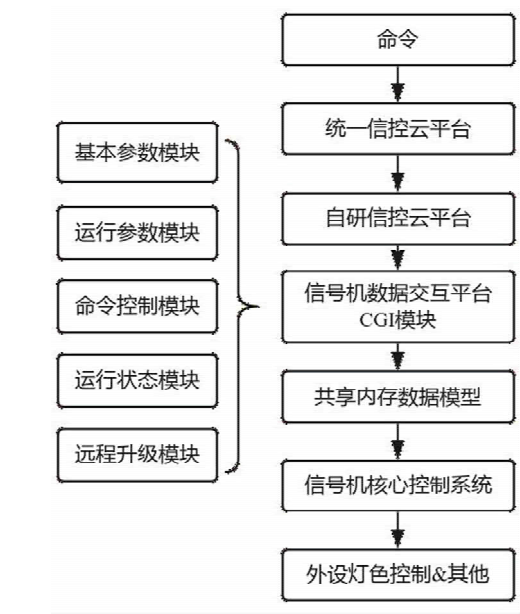


图 6 软件总体设计主流程图

4 结 论

数据的快速与可靠的传递机制,是智能交通远程控制最基本的要求,由于信控平台需要转发处理信号机系统和统一信控平台数据的特殊性,本文主要以减轻信控平台性能压力与优化信控平台与信号机之间的数据交互时间来达到命令的可靠与快速响应。根据远程控制是间断性控制的特点,提出了基于公共网关接口的智能交通远程控制方法。通过在 ARM 系统移植的 Boa 服务器构建了高效的信号机数据交互平台,结合了 CGI 技术与应用层协议,远程信控平台可以直接调用 CGI 模块完成数据的交互,通信内容采用便于平台解析与传输高效的 JSON 格式,降低了信控平台的性能压力,共享内存主题模型则解决了信号机系统内部的数据交互问题,通过测试证明该设计能有效提高远程控制命令的响应速度。本系统已在信控平台得到应用,并且它在新增控制命令功能方面也具有很好的扩展性,也可在信号机运行状态下分模块升级,还能够兼容开发嵌入式 web 来方便运维人员在本地调试信号机。需要指出的是,本文方法的有效性依赖于移植的服务器类型与 CGI 编程算法,通信协议定义的内容也具有特殊性,在应对双向实时数据交互时具有一定的局限性,因此仍有很大的改进空间,同时在更大量数据的状况下,该方法可以到 V2R(Vehicle to Road)上再进行研究。

参考文献

[1] AUTILI M, CHEN L, ENGLUND C, et al. Cooperative intelligent transport systems: choreography based urban traffic coordination[J]. IEEE Transactions on Intelligent Transportation Systems, 2021(99):1-12.

[2] LIU X, YUAN H, HU J, et al. Single-point adaptive control method for urban mixed traffic flow[J]. Journal of Advanced Transportation, 2020, 2020:8827824.

[3] PHAN C T, PHAM D D, Tran H V, et al. Applying the IoT platform and green wave theory to control intelligent traffic lights system for urban areas in Vietnam[J]. KSII Transactions on Internet and Information Systems, 2019, 13 (1):34-52.

[4] LIANG Z, XIAO Y, FLTTERD Y P. An overlapping phase approach to optimize bus signal priority control under two-way signal coordination on urban arterials[J]. Journal of Advanced Transportation, 2021, 2021: 6624130

[5] LI Z W, DU R Y, HE Z Z. Multiple dynamic traffic-flow information fusion and traffic dynamic information real-time induction release system[J]. Western China Communications Science & Technology, 2018, 6:144-147.

[6] DU R. A review of research on intersection control based on connected vehicles and data-driven intelligent approaches[J]. Electronics, 2020, 9, 885.

[7] OPOKU D, KOMMEY B. FPGA-based intelligent traffic controller with remote operation mode[J]. 2020, 3(3):490-500.

[8] ZHANG J J, ZHANG J, LIU B X, et al. Remote management software design for urban road traffic equipment based on android mobile terminal[J]. Applied Mechanics and Materials, 2014, 513-517:518-521.

[9] YANG J, SUO H. Intelligent traffic signal control system based on edge computing and big data[J]. Journal of Physics Conference Series, 2020, 1624:032039.

[10] JAMIL A R M, GANGULY K K, NOWER N. Adaptive traffic signal control system using composite reward architecture based deep reinforcement learning[J]. IET Intelligent Transport Systems, 2021, 14(2015):1-12.

[11] BOSKANY M A F N W. Intelligent traffic congestion control system using machine learning and wireless network[J]. UHD Journal of Science and Technology, 2020, 4(2):123-131.

[12] SUNEETHA K, SREEKNTH M, SANKARA K. An approach for intelligent traffic signal control system for ambulance using IoT[J]. Social Science Electronic Publishing, 2018.

[13] KUNDALA D, BAMMID R. Remote monitoring and control by embedded database design and web server implementation using SQLite database and Boa web server[C]//International Conference on Electronics & Communication Systems. 2017: 143-147.

[14] HUANG X. Application of embedded web server in the design of ship network monitoring system[J]. Journal of Physics: Conference Series, 2021, 1952:042005.

[15] GAO J, DUAN H C. Research on data transmission efficiency of JSON[J]. Computer Engineering and Design, 2011, 32 (7):2267-2271.

[16] CHEN C T, ZHAO J Q, YE C, et al. Inter-process asynchronous communication in autonomous vehicle based on shared memory[J]. Journal of Software, 2017, 28(5):1315-1325.