POZNAN UNIVERSITY OF TECHNOLOGY ACADEMIC JOURNALS
No 102 Electrical Engineering 2020

DOI 10.21008/j.1897-0737.2020.102.0005

Paweł SZULCZYŃSKI*, Piotr SAUER*

STANOWISKO LABORATORYJNE SYSTEMÓW AUTOMATYKI DOMOWEJ

W pracy przedstawiono zaprojektowane i wykonane stanowisko laboratoryjne z zakresu systemów automatyki domowej. Do budowy stanowiska wykorzystano moduły ESP8266 oraz mikrokomputer Raspberry Pi. Głównym celem powstałego stanowiska jest jego wykorzystanie w działaniach dydaktycznych, jako elementów ćwiczeń laboratoryjnych. Przygotowane ćwiczenia użyteczne będą zarówno na studiach pierwszego i drugiego stopnia. W pracy opisano projekt oraz wykonaną konstrukcje stanowiska badawczego, sposoby jego wykonania na zajęciach oraz zastosowane technologie.

SŁOWA KLUCZOWE: Automatyka budynków, IoT, stanowisko dydaktyczne, MicroPython, JavaScript, Arduino, Lua, Raspberry Pi, NodeMCU, MQTT, Modbus.

1. WPROWADZENIE

Automatyka domowa stała się produktem powszechnie dostępnym. Rosnące zainteresowanie tego typu rozwiązaniami sprawiło, że dla wielu jest ono istotnym elementem przy kupnie własnej nieruchomości. Jednak dla osób już posiadających mieszkanie wdrożenie takiego systemu wydaje się zbyt pracochłonne i drogie. Optymalnym rozwiązanie wydaje się Internet Rzeczy (IoT). Dzięki coraz większej ilości urządzeń wyposażonych w IoT zmienia się architektura systemów Automatyki Domowej. Połączenia kablowe przestają mieć znaczenie a automatyzację domu można zacząć od jednej żarówki IoT.

W celu przybliżeniu studentom nowych technologii postanowiono wykonać stanowisko dydaktyczne, na którym poznają zagadnienia związane z Internetem rzeczy oraz automatyką je wykorzystującą. Do budowy stanowiska użyto względnie tanich i powszechnie dostępnych komponentów tak by zachęcić studentów do samodzielnego eksperymentowania. System, jaki można wykonać w oparciu o przygotowany projekt może kosztować tyle, co jednostka centralna systemu automatyki budynków w ofercie komercyjnego producenta.

Kolejnym celem wykonanego stanowiska laboratoryjnego jest poznanie przez studentów sposobów programowania mikrokontrolerów w różnych językach, połączenia ich do lokalnej sieci WiFi oraz przesyłania informacji między tymi komponentami za pomocą popularny obecnie protokołów. Uwypuklenie możliwości

* Politechnika Poznańska

zaprogramowania tego samego urządzenia, aby wykonywało to samo zadanie w różnych językach i za pomocą różnych środowisk pozwala zrozumieć jak szerokim wachlarzem rozwiązań dysponuje rynek elektroniki. Wartością edukacyjną jest też możliwość porównania wydajności języków programowania [2].

Dodatkowo wszystkie wykorzystane programy są typu open-source, przez co nie zwiększa to kosztów poniesionych przez studenta gdyby chciał system ten wykorzystać w swoim domu.

2. STANOWISKO

2.1. Założenia projektowe

W pracy przyjęto, że zostanie wykonane stanowisko przedstawiające system automatyki domowej dla małego domu. Model ma składać się z typowych elementów i układów stosowanych w budownictwie mieszkaniowym. W związku z powyższym, w domu powinny znajdować się podstawowe pomieszczenia takie jak pokój, kuchnia i łazienka. Dodatkowo w modelu uwzględnić należy sterowanie urządzeniami znajdującymi się w większej odległości. Założono, że w każdym z pomieszczeń znajdować się będzie niezależne oświetlenie wraz z włącznikami oraz czujniki monitorujące komfort cieplny. Dodatkowo wymaga się by można było zmieniać i sprawdzać stan okien i drzwi (otwarte, zamknięte). Całość powinna pozwalać na modelowanie różnych scenariuszy zarządzania automatyką budynku.

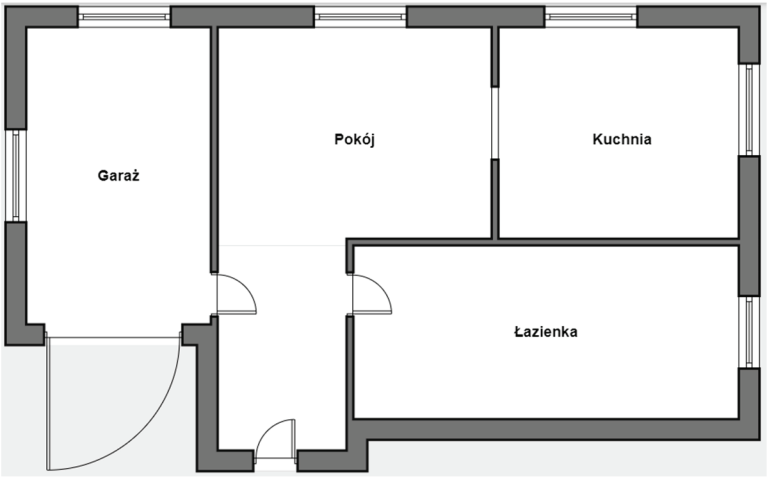
Stanowisko nie powinno być zbyt duże i ciężkie tak by swobodnie mieściło się na stole. Wymaga się też by wszystkie połączenia kablowe były opisane i poprowadzone zgodnie ze sztuką. Układy zasilane napięciem przemiennym powinny być odpowiednio zabezpieczone. Duże urządzenia wykonawcze należy zamodelować za pomocą prostszych urządzeń.

Elementy sterujące powinny być swobodnie programowalne a przeznaczone dla nich oprogramowanie typu open-source. Spełniać powinny podstawowe wymagania stawiane urządzeniom IoT, czyli gromadzić i przetwarzać dane oraz ich wymianę poprzez różne protokoły komunikacyjne w szczególności bezprzewodowe. Ponad to powinny cechować się niski poziom zużycia energii, być tanie i powszechnie dostępne.

2.2. Wykonanie stanowiska

Na potrzeby zobrazowania wyglądu budynku, przygotowano plan rozmieszczenia pomieszczeń, który przedstawiono na rysunku 1. Uwzględniono w nim typowe pomieszczeń takie jak pokój, kuchnia i łazienka. Dodatkowo obok mieszkania umieszczono garaż. Planszę z planem budynku umieszczono na mlecznej płycie poliwęglanowej przykręconej do konstrukcji z profili ze stali nierdzewnej.

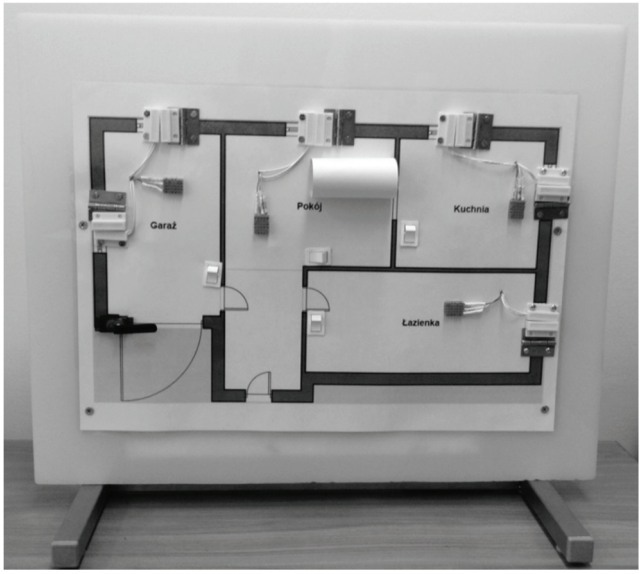
W celu jak najbardziej wiernego odzwierciedlenia pomieszczeń mieszkalnych przyjęto, że układ oświetlenia zasilany będzie napięciem sieciowym 230 V. Oprawy żarówek i żarówki umieszczono, dla bezpieczeństwa, za płytą na środku każdego z pomieszczeń. Od frontu natomiast umieszczono przełączniki kołyskowe 2-pozycyjne. We wszystkich pomieszczeniach znajdują się czujniki DTH11 mierzące temperaturę i wilgotność powietrza. Informujące o tym czy dane okno jest otwarte czy zamknięte symulowane jest za pomocą kontaktronu umieszczonego na zawiasie taśmowym. W celu symulowania działania układu wentylacji w pokoju umieszczono wentylator napędzany sinikiem prądu stałego natomiast w garażu wstawiono serwomechanizm modelarski, który symuluje napęd bramy. Gotowe stanowisko pokazano na rysunku 2.



Rys. 1. Plan rozmieszczenia pomieszczeń

Od tylnej strony stanowiska umieszczono elementy związane ze sterowaniem oraz zasilaniem wszystkich układów. W skład obwodu prądu przemiennego wchodzą elementy umieszczone na szynie DIN takie jak wyłącznik nadmiarowo-prądowy, przekaźniki elektromagnetyczne z nominalnym napięciem cewki 230 VAC. Pozostałe elementy to przekaźniki elektromagnetyczne z nominalnym napięciem cewki 12 VDC, przełączniki kołyskowe 2-pozycyjne oraz żarówki. Do zasilania elementów elektronicznych wykorzystano zasilacz komputerowy z liniami 3,3 V, 5 V i 12 V.

Do akwizycji danych z czujników oraz sterowania układami wykonawczymi wykorzystano mikrokontrolery ESP8266 umieszczone w modułach NodeMCU v3 oraz urządzenie dostępne komercyjnie Sonoff Basic. Za nadzór nad modułami sterującymi oraz wizualizację procesów odpowiedzialny jest mikrokomputer Raspberry Pi Zero W.



Rys. 2. Stanowisko laboratoryjne systemów automatyki domowej

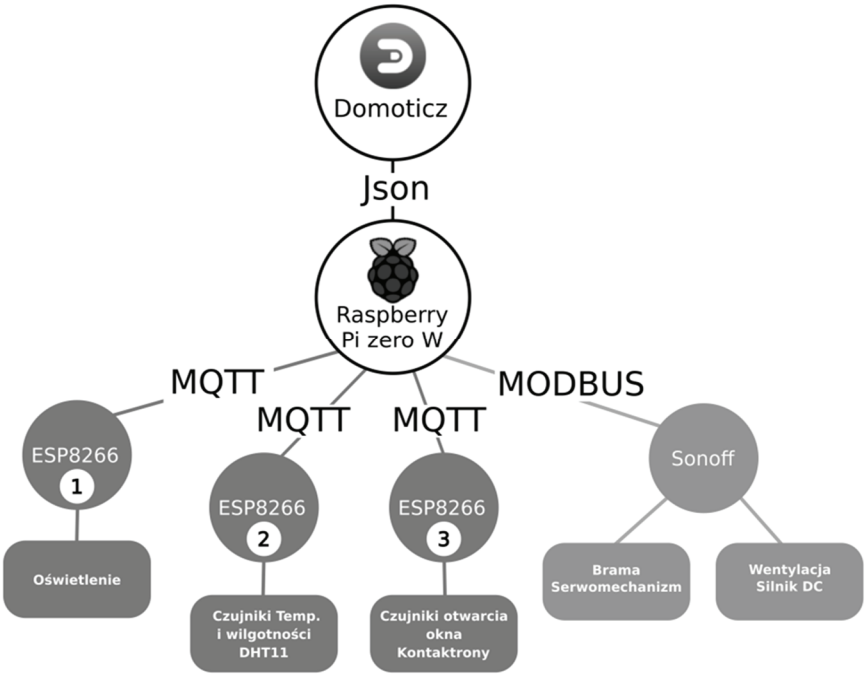
2.3. Konfiguracja stanowiska

Ponieważ głównym celem powstałego stanowiska jest pokazanie, że można w stosunkowo łatwy sposób zintegrować różne niezależnie działające urządzenia, dlatego każdy z mikrokontrolerów został przygotowany do pracy w innym środowisku. Komercyjne urządzenie Sonoff Basic to gotowy moduł umożliwiający zdalne załączenie lub wyłączenie urządzeń domowych zasilanymi z sieci 230 V o maksymalnej mocy do 2200 W. Sterowanie urządzaniem odbywa się poprzez sieć WiFi z wykorzystaniem aplikacji na smartfona. Urządzenie przygotowane jest do wysyłania danych do chmury na serwer Amazon AWS za pośrednictwem dedykowanej aplikacji. Za komunikację z siecią odpowiada moduł ESP8266. Na płytce wewnątrz urządzenia wyprowadzone są piny UART, dzięki czemu możliwe jest programowanie urządzenia poprzez odpowiedni konwerter. Dla wygody programowania wyprowadzono te złącza na zewnątrz obudowy. Oryginalny program został zastąpiony programem napisanym w środowisku Arduino IDE.

Moduły NodeMCU również składają się z moduł WiFi ESP8266 oraz oprogramowania NodeMCU stąd też popularnie tak się je nazywa. Ponadto na płytce umieszczono konwerter USB-UART (CH340), który umożliwia programowanie bezpośrednio poprzez port USB. Urządzenie zostało zaprojektowane do pracy z językiem skryptowym Lua [5], możliwe jest jednak programowanie za pomocą Arduino IDE, JavaScript [4] oraz MicroPython [1]. Ponieważ oprogramowanie Arduino IDE wykorzystano już do programowania zmodyfikowanego urządzeniu

Sonoff Basic, do programowania w pozostałych językach wykorzystano 3 moduły NodeMCU.

Na mikrokomputerze Raspberry Pi Zero W zainstalowano system operacyjny Raspbian. Ponieważ wszystkie mikrokontrolery mogą komunikować się poprzez sieć Wi-Fi postanowiono wykorzystać moduł Raspberry Pi, jako access point [7]. W ten sposób uniknięto korzystania z zewnętrznej sieci zwiększając tym samym bezpieczeństwo pracy stanowiska. Dodatkowo w celu uniknięcia niepożądanych podłączeń do sieci nałożono filtr MAC.



Rys. 3. Schemat połączeń komunikacyjnych

Jako podstawowy protokół transmisji na stanowisku wybrano MQTT (Message Queue Telemetry Transport). Protokół MQTT został opracowany w 1999 roku przez firmy IBM oraz Circus Link a w 2013 roku został oficjalnie uznany, jako standard ISO. Architektura MQTT oparta jest o wzorzec publish-subscribe. Wiadomości wysyłane przez nadawców (publisher) trafiają do serwera pośredniczącego (broker), a następnie odczytywana przez odbiorców (subscriber). W tym celu na Raspberry został zainstalowany broker Mosquitto [9]. Jest to open-source'owy broker implementujący protokół MQTT w wersji 3.1. Publikowane wiadomości mają określony temat (topic). Klienci, którzy subskrybują dany temat

otrzymują opublikowane na nim wiadomości. Dane publikowane przy użyciu protokołu MQTT są danymi tekstowymi. Schemat połączeń komunikacyjnych przedstawiono na rysunku 3.

Do wizualizacji oraz zarządzania stanowiskiem wykorzystano Open source'owy system automatyki domowej Domoticz [6]. Dzięki gotowym modułom można z jego pomocą dowolnie konfigurować i monitorować odczyty z różnych czujników takich jak czujniki temperatury, wilgotności, deszczu, prędkości wiatru, zużycia energii elektrycznej oraz sterowanie urządzeniami domowymi takimi jak przełączniki światła, rolety, alarmy itp. Informacje z systemu mogą być przesyłane i wysyłane bezpośrednio na urządzenia mobilne. Sterowanie systemem Domoticz możliwe jest z poziomu większości przeglądarek internetowych. Wiele urządzeń producentów komercyjnych (Xiaomi, Logitech, Philips, BLEBOX) również bezpośrednio współpracuje z tym systemem. Modułowa budowa strony pozwala na dodawanie oraz pozycjonowanie nowych odczytów i urządzeń na każdym etapie projektowania systemu oraz rozwijanie projektu w miarę potrzeb. Wiadomości wysyłane i odbierane przez Domoticz mają formę obiektów JSON (JavaScript Object Notation)[3]. Jest to lekki format wymiany danych w formie tekstu. Pomimo swej nazwy, jest obsługiwany przez wiele języków. Składa się z dwóch struktur: tablicy asocjacyjnej i uporządkowanej listy wartości. Możliwe jest tworzenie pojedynczych danych lub złożonych struktur. W projekcie przyjęto, że obiekty JSON będą statyczne i znane przed przystąpieniem do programowania mikrokontrolerów.

Do komunikacji poprzez protokół Modbus wykorzystano plugin Modbus Read/Write Plugins for Domoticz [8]. Protokół ten jest często używany w sterowaniu klimatyzacją czy też falownikiem. Jest to też przykład jak można rozwiązać problem komunikacji na większe odległości poprzez medium przewodowe. W połączeniu za pomocą protokołu Modbus Domoticz pełni rolę urządzenia Mastera. Jako urządzenia Slave posłużył jeden z modułów NodeMCU.

2.4. Wybrane języki programowania

W pracy automatyka często zdarza się, że można znaleźć równoważne rozwiązania tego samego problemu. Jak już wspomniano wcześniej na stanowisku przygotowano cztery mikrokontrolery, które można programować w różnych językach.

Środowisko Arduino IDE zaprojektowane zostało do współdziałania z układami z rodziny Arduino jednak pozwala programować inne różnego rodzaju mikrokontrolery. Skorzystano z rozszerzenia środowiska do obsługi płytek i modułów z serii ESP. Językiem programowania jest C/C++. Standardowo plik z programem zawiera trzy sekcje. W pierwszej dołączane są biblioteki, deklaracje funkcji i zmienne globalne oraz tworzone są obiekty klas. Kolejna sekcja to void

setup(), w której znajdują się instrukcje wykonywane tylko raz. Trzecią część stanowi pętla główna programu void loop() gdzie instrukcje wykonywane są cyklicznie.

Kolejnym językiem jest język skryptowy Lua [5]. Bazuje on na standardzie ANSI C, jednak od klasycznego języka C różni się brakiem deklaracji typów zmiennych czy składnią funkcji. Język ten nie jest skomplikowany i posiada szerokie wsparcie w postaci bibliotek i podręczników. W celu łatwego pisania i wgrywania programu na płytkę NodeMCU wykorzystano środowisko ESPlorer.

W ostatnich latach dużą popularnością cieszy się język Python. Jest to interpretowany, interaktywny, skryptowy język programowania stworzony przez Guido van Rossuma w 1990. Z kolei MicroPython to pełny kompilator i środowisko wykonawcze Python 3 działające na mikrokontrolerach. W odróżnieniu od innych języków posiada REPL, czyli read evaluate print loop. Oznacza to, że w konsoli na bieżąco prezentowany jest efekt działań kontrolera. Wraz z jego pojawieniem się w roku 2014 wydano płytkę pyboard obsługującą ten język [1]. Obecnie zwiększono ilość wspieranych urządzeń o ESP8266, WiPy, Teensy 3 oraz BBC micro:bit. MicroPython podobnie jak Arduino pozwala między innymi na sterowanie wyjściami, komunikację poprzez I2C, SPI czy nawiązywanie połączeń z WiFi. W projekcie wykorzystano środowisko uPyCraft, które pozwala na swobodne wgrywanie kodu na płytkę oraz sprawdzenie poprawności składni programu.

Ostatnim wykorzystywanym językiem jest Javascript. Opracowany został w 1995 roku przez firmę Netscape [4]. Powszechnie stosuje się go do pisania stron internetowych oraz aplikacji. Pozwala na tworzenie rozbudowanych animacji, wyświetlanie interaktywnych elementów i dynamiczne przetwarzanie danych. Wykorzystane środowisko to Espruino Web IDE [10]. Pozwala ono na równoczesne programowanie i obserwacje wykonanego kodu w konsoli. Język wspierany jest przez liczne biblioteki. Szczegółowa dokumentacja znajduje się na stronie Espruino [11].

3. PROPOZYCJA ĆWICZEŃ

Przygotowane stanowisko można wykorzystać do dwóch rodzajów ćwiczeń. Pierwsze związane są ze strategią sterowania automatyką budynku natomiast drugie dotyczą programowania mikrokontrolerów.

3.1. Sterowanie automatyką domową

Podczas wykonywania ćwiczeń z zakresu sterowania automatyką domową studenci poznają w praktyce jak podłączyć czujniki i elementy wykonawcze do interfejsu HMI oraz uzależnić ich wzajemne działanie.

Do wizualizacji działania automatyki wykorzystano program Domoticz. W pierwszej kolejności studenci zaznajamiają się ze sposobem dodawania urządzeń zarówno fizycznych jak i wirtualnych. Następnie przechodzą do ustawienia harmonogramów np. dla oświetlenia. Można w nich ustawić wiele automatycznych akcji jak włączenie lub wyłączenie w określone dni o określonej godzinie czy o wschodzie lub zachodzie słońca. Sterowanie wentylatorem można oprzeć o zdarzenia. Programowanie zdarzeń odbywa się w wizualnym języku Blockly podobnym do edukacyjnego języka obiektowego Scratch. Jest to część, w której można zawrzeć całą logikę łącznie z wysyłaniem powiadomień. Dostępne są też inne języki jak Lua, Bash, Perl, PHP czy Python. Kolejnym elementem są sceny. Dzięki nim do jednego zdarzenia można przypisać zmianę stanu wielu elementów budynku. Przykłady popularnych scen i reakcji to:

Scena: dobranoc – wyłącz wszystkie światła.

Scena: poranek – wyłącz światło, włącz wentylator, gdy jest za ciepło.

Scena: wyjście – wyłącz wszystkie urządzenia, sprawdź czy okna zamknięte, otwórz bramę.

Część związaną z wizualizacją można ustawić na stronie Dashboard. Umieszcza się tam najczęściej używane elementy. Ich wygląd można dowolnie zmieniać poprzez modyfikację plików stylu CSS.

Zaproponowane ćwiczenie idealne jest, jako wprowadzenie do automatyki budynku. Można też w nim poznać podstawy języków programowania użytych w mikrokontrolerach.

3.2. Programowanie mikrokontrolerów

Ze względu na rozbudowaną tematykę dotyczącą języków programowania przewiduje się wykonanie przez studentów serii kilku ćwiczeń. W pierwszej kolejności przedstawiany jest sposób tworzenia obiektów JSON. Następnie sposób komunikacji za pomocą protokołu MQTT. Studenci powinni prawidłowo skonfigurować komunikat tak by opublikować i odebrać wiadomość w konsoli. Dokładne zrozumienie sposobu działania MQTT pozwoli na łatwiejsze rozpoczęcie dalszej pracy na stanowisku.

Kolejnym krokiem jest już nauka wybranego języka programowania. Wydaje się, że program na NodeMCU studenci powinni wykonać kompleksowo. W praktyce rzadko, kiedy natrafia się na gotowe programy napisane zgodnie z przyjętymi przez nas założeniami. Mając jednak na uwadze, że nie wszyscy są biegli w programowaniu, dlatego proponuje się by w pierwszym kroku skorzystać z przygotowanego wzorca. Ponieważ w każdym programie można wyróżnić sekcje odpowiedzialną za inne zadanie:

– nawiązywanie połączenia z siecią WiFi,
– wysłanie do serwera MQTT wiadomości,

– podłączenie czujnika (np.: DTH11) do mikrokontrolera i odczytanie mierzonych wartości,
– przygotowanie wiadomości do wysłania w formacie JSON.

Bardziej złożonym projektem jest przygotowanie komunikacji za pomocą protokołu Modbus. W pierwszej kolejności studenci muszą przygotować w programie Domoticz moduł typu Modbus, wybrać wersję protokołu, ustawić parametry i rozkazy. Następnie należy skonfigurować urządzenie Slave, które będzie odbierać rozkazy. Zadania postawione urządzeniu Slave to:
– odczyt ramki danych Modbus,
– zaimplementować funkcje pozwalające na zapis i odczyt danych z rejestrów,
– realizacja zadania związanego z wybranymi rejestrami.

4. PODSUMOWANIE

W pracy przedstawiono projekt i realizację stanowiska laboratoryjnego. Przedstawiono zastosowane rozwiązania bazujące na najnowszych trendach w dziedzinie automatyki. Zaproponowano również dwa rodzaje ćwiczeń, które można wykonać ze studentami. Ćwiczenia te różnią się zasadniczo poziomem przekazywanej treści. Pierwsze ćwiczenie może być wykorzystane nie tylko na kierunku Automatyka ale też na mu pokrewnych a nawet na przedmiotach technicznych Kierunków Logistyki i Zarządzania. Drugi zestaw ćwiczeń wymaga od studenta więcej pracy. Zastosowanie jednak dość popularnych rozwiązań jak Arduino IDE czy phyton, z którymi studenci spotkali się na innych zajęciach, powinno znacząco ułatwić im pracę.

Warto podkreślić, że tematyka związana z automatyką budynków jest popularna wśród studentów. Chętnie taki system zbudowaliby dla siebie, jednak ogranicza ich skala potencjalnie poniesionych kosztów. Dlatego ważne wydaje się, że w projekcie wykorzystano popularne, łatwe w obsłudze a przede wszystkim tanie podzespoły. Celem takiego zabiegu było udowodnienie, że małym kosztem można zbudować obszerne oraz wielozadaniowe narzędzie automatyki domowej.

LITERATURA

[1] Bell C. MicroPython for the Internet of Things: A Beginner's Guide to Programming with Python on Microcontrollers, ISBN: 978-1-4842-3122-7, 2017.
[2] Choroszucho A., Golonko P., Sumorek M., Żukowski J., Porównanie wydajności wysokopoziomowych języków programowania w systemach mikroprocesorowych, Poznan University of Technology Academic Journals. Electrical Engineering, DOI 10.21008/j.1897-0737.2019.99.0014, 2019.
[3] Crockford D., The application/json Media Type for JavaScript Object Notation (JSON), RFC 4627, IETF, lipiec 2006, DOI: 10.17487/RFC4627.
[4] Flanagan, D., JavaScript: the definitive guide, O'Reilly Media, Inc., ISBN: 9781491952016, 2019.

[5] Ierusalimschy R, Programming in Lua, Fourth Edition, Lua.Org, ISBN: 978-85-903798-6-7, 2016.
[6] Peters R.E., Domoticz Open Source Home Automation System, 2015, https://www.domoticz.com/DomoticzManual.pdf.
[7] Wykorzystanie Raspberry Pi jako access point. https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md.
[8] Plugin Domoticz-Modbus. https://github.com/DomoticX/domoticz-modbus.
[9] Pakiet mosquitto. https://packages.debian.org/pl/jessie/mosquitto.
[10] Środowisko Espruino Web IDE: https://www.espruino.com/ide/, 2019.
[11] Dokumentacja JavaScript: http://www.espruino.com/Reference#software, 2019.

LABORATORY STATION FOR HOME AUTOMATION SYSTEMS

The paper presents a designed and constructed laboratory stand in the field of home automation systems. ESP8266 modules and Raspberry Pi microcomputer were used to build the station. The main purpose of the resulting position is its use in teaching activities as part of the laboratory exercises. The prepared exercises will be useful in both first and second cycle studies. The paper describes the design and construction of the test stand, the methods of its implementation in class and the technologies used.

(Received: 03.12.2019, revised:15.12.2019)