Samouczek Wprowadzenie do projektowania i prototypowania IoT z przykładami

Wstęp

Ewolucja Internetu Rzeczy (IoT) spowodowała ogromne transformacje cyfrowe w wielu różnych sektorach działalności człowieka, w tym między innymi w opiece zdrowotnej, transporcie, sieciach energetycznych, produkcji i logistyce. Wynika to częściowo z faktu, że technologie komunikacyjne, technologie czujników oraz pamięć komputerowa i moc obliczeniowa stały się łatwo dostępne i niedrogie, tak że producenci produktów konsumenckich mogą sobie pozwolić na umieszczanie ich w szerokiej gamie produktów. Z biegiem lat wszechobecność wspomnianych technologii przekształciła branżę półprzewodników, tworząc sprzyjające środowiska, które ułatwiają opracowywanie i produkcję chipów. Dziś, pomimo zmierzchu prawa Moore′a, producenci chipów nadal ścigają się, aby opracować mniejsze, inteligentniejsze, tańsze i bardziej energooszczędne chipy, a to z kolei znacznie przyspieszyło rozwój różnych platform rozwoju sprzętu IoT. Platformy rozwoju sprzętu IoT to zasadniczo zestawy lub gotowe płyty rozwojowe, które łączą mikrokontrolery i procesory z układami komunikacji bezprzewodowej i innymi komponentami w gotowym do zbudowania i zaprogramowanym pakiecie. Występują w różnych konfiguracjach i zawierają różnorodne urządzenia peryferyjne do podłączania czujników i komunikowania się z innymi komponentami sprzętowymi lub urządzeniami do projektowania i prototypowania urządzeń IoT. Platformy rozwoju sprzętu IoT są generalnie podzielone na dwie kategorie: płyty oparte na mikrokontrolerach (np. płyty Arduino, Particle Photon i Adafruit Feather Bluefruit) oraz komputery jednopłytkowe (SBC) (np. Raspberry Pi, BeagleBone Black i C.H.I.P). Pojawienie się Internetu Rzeczy wraz z utworzeniem kilku innowacyjnych aplikacji i usług zorientowanych na konsumenta w różnych dziedzinach doprowadziło do eksplozji szerokiej gamy potężnych, ale niewielkich platform programistycznych dla sprzętu IoT. Fascynującym trendem jest dziś fakt, że wielu dostawców i firm konkuruje o to, aby ich platformy sprzętowe wyglądały o wiele bardziej atrakcyjnie i atrakcyjnie dla projektantów, programistów i entuzjastów elektroniki IoT. To dążenie do przewagi konkurencyjnej szybko zmienia krajobraz platform rozwoju sprzętu IoT. W rezultacie wiele nowych platform rozwoju sprzętu IoT jest teraz tańszych, wymaga minimalnej konfiguracji systemu, ma więcej pamięci i mocy obliczeniowej, obsługuje wiele języków programowania i jest wyposażonych w inne funkcje, które mogą obejmować wbudowane zabezpieczenia sprzętowe i funkcje komunikacji pudełkowej, takie jak opcje łączności bezprzewodowej. W związku z tym platformy rozwoju sprzętu IoT stają się teraz bardziej przyjazne dla użytkownika do tego stopnia, że mogą z nich korzystać nawet osoby o mniejszych umiejętnościach technicznych. Jednak brak podstawowej wiedzy na temat prawidłowego projektowania i rozwoju może potencjalnie prowadzić do słabego projektowania i wdrażania urządzeń IoT, co może mieć poważne konsekwencje dla wydajności i bezpieczeństwa. Ta część ma na celu zaproponowanie rozwiązania problemu wskazanego powyżej. Celem jest opracowanie kompleksowych wytycznych dotyczących projektowania i prototypowania dla IoT, które będą łatwe do zrozumienia i efektywne dla ucznia. W związku z tym część jest prezentowana w formie samouczka, który może służyć jako przewodnik dla początkujących, którzy są zainteresowani nauką tworzenia prototypów IoT. Biorąc pod uwagę wyzwania związane z projektowaniem i rozwojem systemów wbudowanych, takich jak urządzenia IoT, zarówno w zakresie oprogramowania, jak i części sprzętowych, część ta może również służyć poszerzeniu doświadczenia i wiedzy praktykujących projektantów i programistów.

Główne cechy platform rozwoju sprzętu IoT

Platformy programistyczne sprzętu IoT mają szereg funkcji, które sprawiają, że nadają się do projektowania i prototypowania urządzeń IoT. Jednak w tej sekcji omówiono tylko niektóre z najważniejszych cech tych platform, a mianowicie przetwarzanie i pamięć/pojemność pamięci; zużycie energii, rozmiar i koszt; Systemy operacyjne (OS) i języki programowania; łączność i

elastyczność/możliwość dostosowania urządzeń peryferyjnych platform sprzętowych; oraz wbudowane czujniki i zabezpieczenia sprzętowe. Główny nacisk kładziony jest na płyty Arduino i SBC Raspberry Pi.

Kluczowe cechy platform programistycznych Arduino

Arduino jest w zasadzie platformą do prototypowania elektroniki typu open source, która składa się z dwóch zasadniczych części: sprzętu, którym jest płyta Arduino, oraz oprogramowania, zintegrowanego środowiska programistycznego (IDE). Chociaż istnieją różne typy płyt Arduino, ta sekcja nie ma na celu omówienia wszystkich płyt Arduino, które powstały od 2007 roku. Przedstawia raczej wyczerpujący opis wyżej wymienionych funkcji dla kilku płyt Arduino, a mianowicie Uno (wydanej w 2010), Mega 2560 (wydany w 2010), Due (wydany w 2012), Yún (wydany w 2013), Arduino/Genuino 101 (wydany w 2015) i Arduino/Genuino MKR1000 (wydany w 2016) . Płyty są dobierane w oparciu o ich popularność, funkcjonalność, nowość na rynku oraz możliwość zastosowania w projektach IoT.

Przetwarzanie i pamięć/pojemność pamięci

Sercem każdej płyty Arduino jest jednostka mikrokontrolera (MCU), rodzaj układu scalonego (IC) z procesorem, wbudowaną pamięcią i programowalnymi urządzeniami peryferyjnymi I/O. Niezależnie od tego, czy MCU 8, 16 czy 32-bitowy, częstotliwość zegara procesora lub szybkość zegara, mierzona w megahercach (MHz) lub gigahercach (GHz), określa, ile instrukcji na sekundę może wykonać MCU. 8, 16 lub 32 bity odnoszą się do rozmiaru rejestrów, które są obszarami pamięci o dużej szybkości w procesorze, które przechowują instrukcje i dane aktualnie przetwarzane. Większa liczba rejestrów umożliwia procesorowi jednoczesną obsługę większych obszarów pamięci. Zatem im większy rozmiar rejestrów, tym szybciej MCU może przetworzyć zestaw danych. Wbudowana pamięć w MCU składa się z pamięci o dostępie swobodnym (RAM) do przechowywania danych ulotnych, pamięci flash do przechowywania kodu programu i stałych oraz pamięci programowalnej tylko do odczytu (EEPROM) do przechowywania trwałych danych, takich jak konfiguracje systemu lub inne dane, które zapewniają normalne działanie po ponownym uruchomieniu. Uno bazuje na układzie ATmega328, który jest mikrokontrolerem produkowanym przez Atmel. Układ posiada 8-bitową jednostkę centralną (CPU) o częstotliwości taktowania 16 MHz, 2 kB statycznej pamięci RAM (SRAM), 1 kB EEPROM i 32 kB pamięci flash (z czego 0,5 kB jest wykorzystywane przez program rozruchowy). Mega 2560 jest oparty na MCU ATmega2560. 8-bitowy procesor w układzie ATmega2560 jest taktowany 16 MHz. Płyta Mega 2560 ma SRAM 8 kB, 4 kB EEPROM i 250 kB pamięci flash (z czego 8 kB jest używane przez program ładujący). Due to płyta Arduino oparta na mikrokontrolerze AT91SAM3X8E, a procesor to 32-bitowy SAM3X8E ARM Cortex-M3 o taktowaniu 84 MHz. Posiada 96 KB pamięci SRAM, która jest podzielona na dwa banki: 64 KB i 32 KB. Pamięć flash 512 KB na płycie jest całkowicie dostępna dla użytkownika. Yún jest unikalną płytą Arduino, ponieważ ma dwa różne 8-bitowe procesory, MCU ATmega32U4 i mikroprocesor Atheros AR9331, wszystkie taktowane z częstotliwością 16 MHz. MCU ATmega32U4 ma 2,5 kB pamięci SRAM, 1 kB pamięci EEPROM i 32 kB pamięci flash (z czego 4 kB jest używane przez program ładujący). Procesor Atheros AR9331 ma 64 MB pamięci synchronicznej Double Data Rate 2 (DDR2) (SDRAM), 2,5 KB pamięci SRAM, 1 KB pamięci EEPROM i 16 KB pamięci flash. Yún ma również czytnik kart micro Secure Digital (SD), który jest połączony z procesorem Atheros AR9331. Arduino/Genuino 101 wykorzystuje moduł Intel Curie oparty na układzie Intel Quark SE System on Chip (SoC). Moduł ma dwa maleńkie rdzenie, x86 (Intel Quark) i 32-bitową architekturę rdzenia ARC EM o taktowaniu 32 MHz. Maleńkie rdzenie działają jednocześnie i dzielą tę samą pamięć: 24 KB pamięci SRAM i 196 KB pamięci flash. Płytka Arduino/Genuino MKR1000 oparta jest na układzie Atmel ATSAMW25SoC, który składa się z trzech głównych bloków. Podczas gdy pierwszy blok to 32-bitowy MCU ARM o małej mocy SAMD21 Cortex-M0+ o częstotliwości taktowania 48 MHz, pozostałe dwa główne bloki zostaną wyjaśnione w dalszej części. Płyta ma również wbudowany zegar czasu

rzeczywistego (RTC) używany do koordynowania automatycznego wybudzania z trybu zatrzymania/wstrzymania, który jest taktowany z częstotliwością 32,768 kHz. MKR1000 ma 32 kB SRAM i 256 kB pamięci flash.

Pobór mocy, rozmiar i koszt płyt Arduino

Kluczowym wymogiem dla urządzeń IoT jest zdolność do zużywania bardzo małej mocy przy zachowaniu akceptowalnego poziomu wydajności, co może umożliwić im działanie na bateriach przez długi czas. Strategie osiągnięcia ultraniskiej mocy w urządzeniach IoT obejmują m.in. zastosowanie technologii komunikacyjnych o niskim poborze mocy oraz wdrożenie operacji o niskim cyklu pracy (Maksimović i in., 2014). Niemniej jednak zużycie energii lub żywotność baterii urządzeń IoT nadal pozostaje trudnym tematem badawczym. Żywotność baterii (w godzinach) urządzenia IoT można obliczyć, dzieląc pojemność baterii (w miliamperogodzinach (mAh)) przez średnie zużycie prądu (w miliamperach (mA)) przez urządzenie. Zużycie energii lub żywotność baterii płyty rozwojowej sprzętu zależy w dużej mierze od napięcia i prądu roboczego płyty, a także od komponentów, które są do niej podłączone. W tej sekcji omówiono napięcie robocze, pobór prądu czynnego i pobór mocy rozważanych płyt Arduino. Omówiono również rozmiar i koszt płyt rozwojowych sprzętu. Pamiętaj, że koszt odnosi się do cen tablic w momencie pisania tego rozdziału. Nawiasem mówiąc, specyfikacje napięcia i prądu Uno i Mega 2560 są takie same: ich napięcie robocze wynosi 5 V, prąd stały na pin I/O wynosi 20 mA, a prąd stały dla pinu 3,3 V wynosi 50 mA. Zużycie energii może być obliczone za pomocą równania 1.

$P_{(W)} = I_{(A)} \times V_{(V)}$ (1)

gdzie P to moc w watach, I to prąd roboczy w amperach, a V to napięcie robocze w woltach. W związku z tym użycie 50 mA (0,05 A) i 5 V jako odpowiednio prądu i napięcia roboczego i zastąpienie wartości równaniem 1 daje

$P = 0,05 \times 5 = 0,25$ W = 250mW (2)

Dlatego średni pobór mocy zarówno Uno, jak i Mega 2560 wynosi 250mW. Due ma różne specyfikacje, jego napięcie robocze wynosi 3,3 V, prąd stały dla pinu 3,3 V to 800 mA, a prąd stały dla pinu 5 V to 800 mA, stąd jego średnie zużycie energii wynosi 2,64 W. Specyfikacje napięcia i prądu ATmega32U4MCU na Yún to 5V, prąd stały na pin I/O to 40mA, prąd stały na pin 3.3V to 50mA, a napięcie robocze na procesorze Atheros AR9331 to 3.3V; ponieważ Yún ma wbudowaną obsługę Ethernet i Wi-Fi, prąd roboczy może przekraczać 200 mA, dlatego średni pobór mocy może wynosić 1 W. Chociaż Arduino/Genuino 101 wykorzystuje energooszczędny procesor Intel Curie SoC o prądzie stałym 20 mA na pin we/wy, prąd stały dla pinu 3,3 V (tj. prąd roboczy) nie jest określony. Biorąc pod uwagę, że płyta posiada funkcjonalność Bluetooth LE, pobór prądu przez płytę nie może wynosić 20 mA, stąd jej średniego poboru mocy nie da się poprawnie oszacować na podstawie podanych specyfikacji. Możliwość niskiego poboru mocy MCU Arduino/Genuino MKR1000 sprawia, że prąd DC na pin I/O wynosi 7 mA przy napięciu roboczym 3,3 V. Podczas gdy pobór prądu przez MCU wynosi około 20 mA, samo Wi-Fi może pobierać około 100 mA lub więcej podczas pracy.2 Tak więc w przypadku zastosowań IoT całkowity średni pobór prądu MKR1000 może wynosić 120 mA lub więcej, co oznacza średni pobór mocy większy niż 396mW. Długość i szerokość Uno wynoszą odpowiednio 68,6 i 53,4 mm, a cena Uno-R3 to 22,19 USD. Wymiary Mega 2560 pod względem długości i szerokości wynoszą odpowiednio 101,52 i 53,3 mm, a Mega 2560-R3 obecnie sprzedaje za 38,83 USD. Due ma takie same wymiary jak Mega 2560, ale jego cena to 39,93 USD. Długość i szerokość Yúna wynoszą odpowiednio 73 i 53 mm, a jego cena wynosi 64,90 USD. Długość i szerokość Arduino/Genuino 101 to odpowiednio 68,6 i 53,4 mm, a jego cena to 30,00 USD. Wymiary Arduino/Genuino MKR1000 pod

względem długości, szerokości i wysokości wynoszą odpowiednio 65, 25 i 6 mm, a jego sprzedaż kosztuje 34,38 USD.

Systemy operacyjne i języki programowania dla płyt Arduino

W porównaniu ze standardowymi komputerami, tabletami i smartfonami wiele urządzeń IoT ma ograniczone zasoby, zwłaszcza pod względem zużycia pamięci, a zatem nie mogą one obsługiwać systemów operacyjnych wysokiego poziomu (HLOS), takich jak Windows i Linux. Ponadto różnorodność rodzin i architektur MCU (np. procesory 8-bitowe, 16-bitowe i 32-bitowe) jest jednym z największych wąskich gardeł w rozwoju ogólnych systemów operacyjnych dla tych urządzeń. Wspomniana powyżej różnorodność pogłębia również ograniczenia w zapewnianiu bardziej ogólnego wsparcia systemu operacyjnego dla heterogenicznego sprzętu IoT). W tej sekcji omówiono systemy operacyjne i języki programowania dla platform sprzętowych Arduino. Ze względu na rygorystyczne ograniczenia zasobów, szczególnie małą ilość pamięci RAM, Yún jest jedyną rozważaną płytą Arduino, która obsługuje system operacyjny. Wbudowany procesor Atheros AR9331 obsługuje dystrybucję OpenWrt Linux. W programie zarządza się procesami/wątkami w pozostałych modelach Arduino. Jednak wielu programistów/osób w społeczności Arduino badało możliwości tworzenia przenośnych systemów operacyjnych dla wielu płyt Arduino. Język programowania Arduino lub Arduino Language (AL) jest oparty na C/C++. AL składa się z zestawu funkcji C/C++, które użytkownicy mogą łatwo wywoływać ze swojego kodu (znanego również jako szkic). Podczas gdy praktycznie wszystkie biblioteki wsparcia są podzbiorem standardowej biblioteki C, a nie standardowej biblioteki C++, Arduino IDE zasadniczo używa uproszczonej wersji języka C++. Do pisania i wgrywania programów na płytkę Arduino służy IDE, które można pobrać ze strony Arduino. Inną alternatywą jest skorzystanie z internetowego IDE (Arduino Web Editor), które pozwala użytkownikom zapisywać swoje szkice w chmurze. Szkic automatycznie generuje prototypy funkcji, po czym jest bezpośrednio przekazywany do kompilatora C/C++ (avr-g++). Chociaż płyt Arduino nie można programować bezpośrednio w JavaScript (JS), możliwe jest wykorzystanie funkcjonalności skryptów JS dla klienta WWW przy użyciu zarówno bibliotek Firmata, jak i Johnny-Five. Jest to szczególnie ważne w przypadku aplikacji IoT, zwłaszcza biorąc pod uwagę, jak JS zyskuje ostatnio coraz większą popularność wśród programistów i projektantów IoT w wyniku pojawienia się Node.JS

Łączność i elastyczność/możliwość dostosowania urządzeń peryferyjnych płyt Arduino

Możliwość komunikacji z innymi urządzeniami, zwłaszcza przez Internet, jest bardzo istotną cechą platformy rozwoju sprzętu IoT. Podobnie, dostępność i dostępność urządzeń peryferyjnych zarówno niskiego poziomu (piny I/O), jak i wysokiego poziomu (interfejsy komunikacji sprzętowej) są ważnymi czynnikami, które określają typy projektów, które można zbudować przy użyciu określonej platformy sprzętowej. Wśród rozważanych płyt Arduino tylko Yún i MKR1000 mogą łączyć się bezpośrednio z Internetem bez użycia osłony (osłona Arduino to kompatybilna płytka, którą można podłączyć do płyty Arduino w celu rozszerzenia jej możliwości). Yún obsługuje zarówno Ethernet, jak i Wi-Fi, podczas gdy MKR1000 może łączyć się z Internetem tylko za pomocą WINC1500, niskiej mocy Wi-Fi 2,4 GHz IEEE 802.11b/g/n, która jest drugim blokiem SoC ATSAMW25 na MKR1000. Pozostałe karty mogą łączyć się z Internetem za pomocą nakładek Ethernet, Wi-Fi lub GSM. Arduino/Genuino 101 obsługuje technologię Bluetooth Low Energy (BLE). Pomimo swojej wszechstronności i elastyczności, płyty Arduino nie mogą być używane do obliczeń ogólnego przeznaczenia. Elastyczność płyty Arduino polega na dostępności urządzeń peryferyjnych na MCU, a także ich dostępności dla użytkowników za pośrednictwem programu. Praktycznie wszystkie płyty Arduino mają interfejsy komunikacji szeregowej, takie jak port Universal Serial Bus (USB). Port USB może być używany do zasilania płytki Arduino z komputera; służy również do wgrywania programów z IDE na płytkę. Wszystkie płyty Arduino są wyposażone w szereg cyfrowych pinów I/O, które są używane

jako urządzenia peryferyjne niskiego poziomu, znane również jako piny General Purpose I/O (GPIO). Piny analogowe na płytkach Arduino mają również wszystkie funkcjonalności pinów GPIO. Oprócz cyfrowych pinów we/wy, płyty Arduino obsługują inne sprzętowe interfejsy komunikacyjne, takie jak komunikacja Serial Peripheral Interface (SPI) przy użyciu biblioteki SPI, a także komunikacja Universal Asynchronous Receiver/Transmitter (UART). Obsługują one również komunikację między zintegrowanym obwodem/dwuprzewodowym interfejsem (I2C/TWI) przy użyciu biblioteki przewodów, a oprócz MKR1000 wszystkie inne płyty mają nagłówek ICSP (In-Circuit Serial Programming). Uno posiada port USB, 14 cyfrowych pinów GPIO (z których 6 to wyjście modulacji szerokości impulsu (PWM)) i 6 analogowych pinów wejściowych. Uno ma również nagłówek ICSP i obsługuje komunikację SPI, UART i I2C/TWI. Mega 2560 ma również port USB, 54 cyfrowe piny GPIO (z czego 15 to wyjście WM) i 16 analogowych pinów wejściowych. Posiada również 4 sprzętowe porty szeregowe UART i nagłówek ICSP, a także obsługuje komunikację SPI i I2C/TWI. Due posiada 2 porty USB, 54 cyfrowe piny GPIO (z których 12 to wyjście PWM), 12 analogowych pinów wejściowych i 2 analogowe piny wyjściowe konwertera cyfrowo-analogowego (DAC). Ponadto Due ma 4 porty UART, 1 nagłówek ICSP, 1 nagłówek SPI, 1 nagłówek I2C i 2 nagłówki TWI. Yún ma 2 porty USB, port Ethernet, 20 cyfrowych pinów GPIO (z czego 7 to wyjście PWM) i 12 analogowych pinów wejściowych. Dodatkowo posiada 1 port UART, 1 nagłówek ICSP i obsługuje komunikację SPI i I2C/TWI. Arduino/Genuino 101 posiada port USB, 14 cyfrowych pinów GPIO (z czego 4 to wyjście PWM) i 6 analogowych pinów wejściowych. Jest również wyposażony w nagłówek ICSP i obsługuje komunikację SPI, UART i I2C/TWI. MKR1000 ma port USB, 8 cyfrowych pinów GPIO, 12 pinów wyjściowych PWM, 1 UART, 1 SPI i 1 I2C. Pozostałe piny obejmują 7 pinów wejścia analogowego (przetwornik analogowo-cyfrowy (ADC) 8/10/12 bit) i 1 pin wyjścia analogowego (DAC 10 bit).

Wbudowane czujniki i funkcje zabezpieczeń sprzętowych płyt Arduino

Podczas gdy czujnik jest przeznaczony do odbierania i mierzenia pewnego rodzaju danych wejściowych (np. temperatury, ciśnienia, ruchu, światła, ciepła lub innych zjawisk środowiskowych) ze środowiska fizycznego, czujnik pokładowy jest urządzeniem, które wykrywa i mierzy nie tylko to, co jest dzieje się w jego otoczeniu, ale także w obrębie tablicy. Arduino/Genuino 101 jest jedyną płytą spośród rozważanych płyt Arduino, która posiada wbudowane czujniki. Płytka wyposażona jest w sześcioosiowy akcelerometr i żyroskop. Oba czujniki mogą być używane razem, tworząc jednostkę monitorowania bezwładności (IMU), która może być wykorzystana do dokładnej identyfikacji orientacji płyty Arduino. Ograniczenia obliczeniowe i pamięciowe większości płyt Arduino nałożyły ograniczenia na ich możliwości bezpieczeństwa, a zatem bardzo trudno będzie im uruchomić stosy dojrzałych technologii. Stosy używane do zabezpieczania komunikacji Hypertext Transfer Protocol (HTTP), Constrained Application Protocol (CoAP) lub Message Queuing Telemetry Transport (MQTT) to: Internet Protocol Security (IPsec), Secure Sockets Layer/Transport Layer Security (SSL/TLS) lub Datagram Transport Layer Security (DTLS). Oprócz Arduino/Genuino MKR1000, wszystkie pozostałe płyty Arduino omówione w tym rozdziale nie mają silnika kryptograficznego ani sprzętowego modułu kryptograficznego. Sprzętowy silnik kryptograficzny to samodzielny moduł kryptograficzny z dedykowanym procesorem, co utrudnia hakerom dostęp do cennych danych podczas operacji kryptograficznych. Został zaprojektowany do integracji z urządzeniami jako alternatywa dla implementacji zabezpieczeń opartych na oprogramowaniu. Moduł wykonuje obliczenia szyfrujące i deszyfrujące znacznie szybciej niż programowa implementacja tych samych operacji. MKR1000 ma wbudowany układ kryptograficzny; zasadniczo trzeci blok w ATSAMW25 to ECC508, który zapewnia uwierzytelnianie kryptograficzne. Dodatkowo wbudowany moduł Wi-Fi obsługuje certyfikaty SHA-256 dla zapewnienia bezpiecznej komunikacji

Główne cechy platform sprzętowych Raspberry Pi

Raspberry Pi to SBC wielkości karty kredytowej, który zapewnia prawie pełne możliwości komputera stacjonarnego lub laptopa, pozostając jednocześnie małym, lekkim i niedrogim. Minikomputer Raspberry Pi może być również używany do prototypowania projektów elektronicznych. Po raz pierwszy stworzony w Wielkiej Brytanii w lutym 2012 r. przez Fundację Raspberry Pi, SBC został opracowany w celu promowania nauczania umiejętności programowania i sprzętu komputerowego w szkołach i kraje rozwijające się. W odniesieniu do wyżej wymienionych cech platform rozwoju sprzętu IoT, w tej sekcji opisano następujące modele Raspberry Pi: Raspberry Pi Zero (wydana w listopadzie 2015 r.), Raspberry Pi Zero Wireless (W) (wydana w lutym 2017 r.), Raspberry Pi 1 model B+ (wydany w lipcu 2014), Raspberry Pi 2 model B (wydany w lutym 2015) i Raspberry Pi 3 model B (wydany w lutym 2016). W celu opisania aktualnych funkcji SBC Raspberry Pi, wybrane zostały tylko najnowsze modele.

Przetwarzanie i pamięć/pojemność pamięci

W porównaniu do platform programistycznych Arduino, Raspberry Pi SBC mają szybsze procesory, większą pamięć i większą pojemność. Wszystkie modele Raspberry Pi są oparte na Broadcom SoC, który obejmuje procesor ARM i wbudowany procesor graficzny (GPU), energooszczędny mobilny procesor multimedialny VideoCore IV, który obsługuje rozdzielczość do 1920 × 1200. Na przykład zarówno Raspberry Pi Zero, jak i Pi Zero W są wyposażone w Broadcom BCM2835 SoC z rdzeniem procesora 1 GHz ARM1176JZF-S, członka rodziny ARM11 (która jest o 40% szybsza niż Raspberry Pi 1) i ma 512 MB niski -Zasilanie DDR2 (LPDDR2) SDRAM. Raspberry Pi 1 B+ jest oparty na tym samym SoC Broadcom BCM2835, ale z procesorem 700 MHz ARM1176JZF-S Core i 512 MB pamięci RAM (Bell, 2014). Raspberry Pi 2 B wykorzystuje układ SoC Broadcom BCM2836 z 32-bitowym czterordzeniowym procesorem ARM Cortex-A7 900 MHz i 1 GB pamięci RAM. Podobnie Raspberry Pi 3 B wykorzystuje Broadcom BCM2837 SoC z 64-bitowym czterordzeniowym procesorem ARM Cortex-A53 1,2 GHz i 1 GBRAM.8 Wszystkie Raspberry Pi SBC są wyposażone w gniazdo kart micro SD.

Pobór mocy, rozmiar i koszt platform sprzętowych Raspberry Pi

Jako minikomputer o małej mocy, Raspberry Pi działa na zasilaczu 5 V DC o natężeniu 1–2,5 A, zasilanym przez port micro-USB. Chociaż Raspberry Pi zużywa różne ilości energii w czterech różnych trybach zasilania, a mianowicie trybach pracy, gotowości, wyłączania i uśpienia, aktualny pobór każdego modelu prezentowanego w oficjalnym magazynie Raspberry Pi jest w dwóch trybach: bezczynny (czuwanie) i tryby ładowania (uruchamiania). Pobór prądu w dwóch trybach zasilania dla każdego modelu wynosi: 0,1 i 0,25 A dla Raspberry Pi Zero; 0,25 i 0,31A dla Raspberry Pi 1 B+; 0,26 i 0,42A dla Raspberry Pi 2; oraz 0,31 i 0,58A dla Raspberry Pi 3. Warunki, w jakich te pomiary zostały przeprowadzone, nie są wyraźnie podane w magazynie. Jednak inne źródła, w tym Fundacja Ra spberry Pi i społeczność żywiołów, pokazują bardziej realistyczne zużycie prądu. Na przykład średni pobór prądu Raspberry Pi Zero podczas pracy wynosi około 160 mA, co pokazuje, że średni pobór mocy wynosi około 0,8 W. Według Klosowskiego i RasPi.TV , nowy Raspberry Pi ZeroWrequires około 20mA więcej niż Pi Zero. Oznacza to, że jego średni pobór prądu podczas pracy wynosi około 180 mA, a zatem średni pobór mocy wynosi około 0,9 W. Podczas gdy Raspberry Pi 1 B+ zużywa około 600 mA, co oznacza, że średni pobór mocy wynosi około 3,0 W, średni pobór prądu Raspberry Pi 2 B wynosi około 800 mA, a co za tym idzie średni pobór mocy to około 4,0W. Według Shabaz (2016), pobór prądu Raspberry Pi 3, który obsługuje Rasbian, ale nic nie robi, wynosi 266 mA, a pobór prądu podczas uruchamiania cpuburn-a53 (tj. Test warunków skrajnych, który całkowicie obciąża procesor Pi) wynosi 1,45 A. Wyniki testów pokazują, że pobór mocy Raspberry Pi 3 wynosi od 1,33 do 7,25 W. Podczas gdy pobór mocy Raspberry Pi faktycznie zależy od zastosowania oraz modelu urządzenia, kilka najlepszych technik, które mogą być stosowane podczas pracy w celu zmniejszenia zużycia energii obejmują:

1) Odłącz wszystkie nieużywane urządzenia peryferyjne

2) Wyłącz łączność z Internetem, gdy nie jest potrzebna

3) Wyłącz urządzenie lub przełącz je w tryb uśpienia, gdy nie jest używane;

4) Podczas korzystania z urządzenia w trybie bezgłowym (tj. Dostęp do niego za pośrednictwem połączeń sieciowych bez klawiatury lub wyświetlacza), interfejs multimedialny High-Definition (HDMI) może zostać wyłączony

5) Unikaj uruchamiania kilku demonów naraz i uruchamiaj tylko energooszczędne aplikacje

Wymiary Raspberry Pi Zero i Pi Zero W, które są najmniejsze ze wszystkich, to 65mm×30mm×5,4mm. Ale Raspberry Pi 1 B+, Raspberry Pi 2 B i Raspberry Pi 3 B mają te same wymiary: 85,60 mm × 56 mm × 21 mm. Podczas gdy cena Raspberry Pi Zero wynosi zaledwie 5 USD, Zero W kosztuje 10 USD. Raspberry Pi 1 B+ kosztuje 25 USD. Jednak zarówno Raspberry Pi 2 B, jak i Raspberry Pi 3 B są sprzedawane za 35 USD.

Systemy operacyjne i języki programowania dla Raspberry Pi

Raspberry Pi nie jest dostarczane z systemem operacyjnym. W związku z tym, na podstawie swoich projektów, użytkownicy mogą wybrać typ systemu operacyjnego, który najlepiej odpowiada ich potrzebom. W tej sekcji omówiono systemy operacyjne Raspberry Pi i języki programowania. Raspbian to system operacyjny oparty na Debianie, udostępniany bezpłatnie przez Fundację Raspberry Pi dla sprzętu Raspberry Pi. Z biegiem lat system Raspbian OS znacznie zyskał popularność wśród użytkowników Raspberry Pi ze względu na wysoką wydajność . Podobnie jak pełnoprawny system operacyjny dla tradycyjnych komputerów, zawiera wszystkie podstawowe narzędzia programowe i ponad 35 000 pakietów. Pomimo faktu, że Raspbian OS jest oficjalnie wspierany przez Raspberry Pi Foundation, Raspberry Pi SBC jest w stanie obsługiwać szeroka gama systemów operacyjnych. Systemy operacyjne oparte na Linuksie obsługiwane przez Raspberry Pi to m.in. Ubuntu MATE, Snappy Ubuntu, Pidora (Fedora OS dla Raspberry Pi), Linutop, SARPi (tj. Slackware ARM na Raspberry Pi), Arch Linux ARM, Gentoo Linux, FreeBSD, i Kali Linux. Raspberry Pi 2 i 3 mogą również uruchamiać systemy operacyjne Windows, takie jak Windows 10 IoT Core. Najłatwiejszym sposobem zainstalowania Raspbian na Raspberry Pi jest użycie nowego Out Of Box Software (NOOBS), który jest łatwym w użyciu menedżerem instalacji systemu operacyjnego Raspberry Pi. Użytkownicy mogą kupować karty SD z preinstalowanym NOOBS od dystrybutorów Raspberry Pi. Alternatywnym sposobem uzyskania NOOBS jest pobranie pliku zip za darmo ze strony Raspberry Pi. Do załadowania najnowszej wersji potrzebna jest karta SD o pojemności 8 GB, Jessie. NOOBS zawiera sporo systemów operacyjnych, z których użytkownicy mogą wybierać. Dwa języki programowania, które domyślnie są dostarczane z Raspberry Pi, to Scratch i Python, mimo że Python jest bardziej popularny. Jednak z biegiem lat dla Raspberry Pi zaadaptowano kilka języków programowania. Ponadto wykwalifikowane osoby ze społeczności użytkowników, które chciały zobaczyć swoje ulubione języki na Raspberry Pi, odegrały znaczącą rolę w zapewnieniu, że wybrane przez nich języki zostały dostosowane do Raspberry Pi. Niektóre języki programowania dostępne teraz dla użytkowników do programowania na Raspberry Pi przy użyciu różnych IDE obejmują Java, C, C++, Objective C, JS i Ruby

Łączność i elastyczność/możliwość dostosowania urządzeń peryferyjnych Raspberry Pi

Podobnie jak standardowe komputery stacjonarne i laptopy, prawie wszystkie rozważane SBC Raspberry Pi można podłączyć bezpośrednio do Internetu. Ale w przeciwieństwie do standardowych komputerów, piny I/O i interfejsy komunikacji sprzętowej na wszystkich Raspberry Pis są dostępne dla użytkownika. W tej sekcji omówione zostaną łączność, elastyczność, możliwość dostosowania pinów

we/wy i interfejsy komunikacji sprzętowej Raspberry Pi. Raspberry Pi Zero nie ma wbudowanego portu Ethernet i funkcji Wi-Fi. Można go jednak połączyć z Internetem na różne sposoby. Na przykład można go podłączyć do Internetu za pomocą kabla USB On The Go (OTG) wraz z konwerterem USB RJ45 lub za pomocą klucza USB OTG i Wi-Fi. Z drugiej strony, Pi ZeroW posiada funkcję łączności, która obejmuje bezprzewodową sieć LAN IEEE 802.11 b/g/n, Bluetooth 4.1 i BLE.14 Zarówno Raspberry Pi 1 B+, jak i Pi 2 B mają port Ethernet, ale nie mają wbudowanego Wi- Układ Fi. Raspberry Pi 3 B jest jednak wyposażony zarówno w Ethernet, jak i Wi-Fi (IEEE 802.11n Wireless LAN). Raspberry Pi 3 B obsługuje również Bluetooth 4.1 i BLE. Elastyczność Raspberry Pi pozwala użytkownikom na wykorzystanie urządzenia do obliczeń ogólnego przeznaczenia, a także do projektów elektronicznych. Wszystkie cztery Raspberry Pi są wyposażone w 40 pinów GPIO, interfejs szeregowy kamery (CSI) i HDMI, chociaż HDMI w Raspberry Pi Zero i ZeroW to wersja mini. Z wyjątkiem Raspberry Pi Zero i Pi Zero W, wszystkie inne Pis mają interfejs Display Serial Interface (DSI). Pi Zero i Pi ZeroW nie mają połączonego gniazda audio i kompozytowego wideo 3,5 mm (tj. 4-biegunowego gniazda, które przenosi zarówno sygnał audio, jak i wideo), które znajduje się na drugim Raspberry Pis. Oprócz Pi Zero i ZeroW, które mają port mini USBOTG, wszystkie inne Pis mają 4 porty USB 2.0. Oprócz 40 pinów GPIO, inne interfejsy sprzętowe, które można znaleźć w Raspberry Pi, to SPI, UART i I2C. Na przykład ARM BCM2835 (tj. SoC w Pi Zero, Pi Zero W i Pi 1 B+) ma trzy pomocnicze urządzenia peryferyjne, mini UART, dwa mastery SPI i I2C.16 Podobnie piny GPIO w Pi 2 B i Pi 3 B można skonfigurować jako I2C, SPI i UART.

Wbudowane czujniki i funkcje zabezpieczeń sprzętowych Raspberry Pi

Dodatkowe funkcje, takie jak wbudowane czujniki i zabezpieczenia sprzętowe na platformach rozwoju sprzętu IoT, zwiększają akceptację użytkowników, zwłaszcza teraz, gdy istnieją poważne obawy dotyczące bezpieczeństwa urządzeń IoT. W tej sekcji omówiono wbudowane czujniki i funkcje zabezpieczeń sprzętowych SBC Raspberry Pi. Zasadniczo żaden członek rodziny Raspberry Pi nie ma żadnych wbudowanych czujników; jednak typowy komputer z systemem Linux prawdopodobnie byłby wyposażony we wbudowane czujniki monitorujące. W ten sposób Broadcom SoC zawiera czujniki temperatury i napięcia na chipie, które można odpytywać za pomocą narzędzia vcgencmd w pakiecie oprogramowania układowego Raspberry Pi. Wbudowany czujnik temperatury może służyć do pomiaru temperatury procesora i karty graficznej. Podobnie czujnik napięcia na chipie może być używany do pomiaru różnych napięć, w tym napięcia rdzenia, napięcia wejścia/wyjścia SDRAM i napięcia pamięci fizycznej SDRAM.18 Wartości temperatury lub napięcia można wyświetlić, wpisując odpowiednie polecenia w okno terminala Raspberry Pi. Na przykład polecenie vcgencmdmeasure_volts zwraca wartości napięcia dla niektórych ważnych komponentów Raspberry Pi. Aby zapobiec zwracaniu przez system wartości napięcia dla rdzenia za każdym razem, gdy wykonywane jest polecenie, każdy z następujących składników musi zostać przekazany do polecenia vcgencmd Measure_volts jako opcja:

a) sdram_c (zwraca wartość napięcia dla kontrolera SDRAM)

b) sdram_i (zwraca napięcie wejścia/wyjścia SDRAM-u)

c) sdram_p (zwraca napięcie pamięci fizycznej SDRAM)

d) rdzeń (zwraca napięcie rdzenia procesora GPU)

Pomimo tego, że Raspberry Pi jest wszechstronną platformą programistyczną opartą na Linuksie, która zapewnia nieograniczone możliwości dla różnych aplikacji, brakuje mu sprzętowych mechanizmów bezpieczeństwa. Dodatkowo zabezpieczenie kluczy prywatnych do kryptografii klucza publicznego lub kluczy współdzielonych do kryptografii klucza symetrycznego jest bardzo dużym problemem. Dzieje się tak, ponieważ zarówno dane, jak i kod użytkownika znajdują się na tej samej karcie SD, która zwykle jest widoczna

Projektowanie i prototypowanie aplikacji IoT

Ta sekcja koncentruje się na projektowaniu i prototypowaniu aplikacji obsługujących IoT przy użyciu platform programistycznych Arduino i Raspberry Pi.

Projektowanie i prototypowanie IoT przy użyciu płytek Arduino

Projektując aplikacje IoT dla płytek rozwojowych Arduino, programista zwykle zaczyna od sprawdzenia niezbędnych komponentów dla projektu, które mogą obejmować moduły sprzętowe (i odpowiednie arkusze danych), diagramy, osłony Arduino, biblioteki programistyczne i/lub dodatkowe oprogramowanie. Osłony Arduino to dodatkowe płytki drukowane lub dodatkowy sprzęt, który jest mocowany na górze urządzeń Arduino, aby zapewnić użytkownikom rozszerzone możliwości lub dodatkowe funkcje. Tarcze są specjalnie przeznaczone dla początkujących projektantów i programistów, aby przezwyciężyć złożoność lutowania elementów i dołączania dodatkowych zasobów sprzętowych do swoich projektów. W przypadku, gdy projektant zdecyduje się na użycie alternatywnych modułów elektronicznych zamiast osłon, arkusze danych z tych komponentów umożliwiają projektantowi zrozumienie ich wewnętrznej pracy poprzez opisanie danych i charakterystyk technicznych, takich jak specyfikacje mocy lub prądu. Informacje te pomagają również projektantowi dowiedzieć się, w jaki sposób platforma programistyczna Arduino zareaguje na sygnały otrzymywane z czujników lub innych modułów sprzętowych. Dodatkowo dostarcza informacji, czy regulator napięcia jest potrzebny, czy nie, biorąc pod uwagę istniejące źródło zasilania. Zwykle, aby ułatwić pracę projektantowi, firmy sprzętowe lub programiści zewnętrzni opracowują biblioteki oprogramowania do wysokopoziomowego wykorzystania modułów dla różnych języków programowania i platform, takich jak moduły i biblioteki Adafruit. Istnieje kilka narzędzi, które mogą ułatwić rozwój projektów IoT. Oprogramowanie do projektowania elektronicznego i prototypowania, takie jak Fritzing i Oscad, może być używane do rysowania różnego rodzaju schematów, które mogą pomóc w rozwoju projektów IoT. Fritzing to oprogramowanie typu open source do prototypowania elektronicznego, napisane w C++ i dostępne dla kilku systemów operacyjnych. Został opracowany przez Wyższą Szkołę Zawodową w Poczdamie. Oprogramowanie umożliwia użytkownikom projektowanie projektów w kilku różnych widokach, takich jak widok płytki prototypowej (graficzna reprezentacja płytki prototypowej i odpowiednich obwodów), widok obwodu elektronicznego (schematyczna reprezentacja połączeń elektronicznych) i widok schematu płytki drukowanej (schematyczne przedstawienie obwodu drukowanego). Oznacza to, że projektanci są w stanie zaprojektować prototyp w trybie widoku płytki prototypowej, a następnie przekształcić go w ostateczny projekt, korzystając z widoku płytki drukowanej. Inną alternatywną aplikacją jest Oscad , narzędzie typu open source do projektowania, symulacji i analizy obwodów drukowanych (PCB). Oprogramowanie pomaga projektantom planować, testować i badać ich obwody. Obsługuje Python, KiCad, Ngspice i Scilab. Jedną z głównych cech narzędzia Oscad jest możliwość symulacji obwodów opracowanych w tym narzędziu. Arduino IDE, jak wcześniej wspomniano, jest oficjalnym oprogramowaniem open-source Arduino opracowanym w języku Java, które umożliwia tworzenie i przesyłanie kodu na płytkę rozwojową. To oprogramowanie jest dostępne na oficjalnej stronie Arduino. IDE jest dostarczane z wbudowanymi bibliotekami, które ułatwiają operacje we/wy. Typowy program Arduino jest zbudowany z co najmniej dwóch głównych funkcji, a mianowicie setup() i loop(). Funkcja setup() służy do ustawiania inicjalizacji, która może obejmować inicjalizację zmiennych, ustawianie zmiennych bibliotecznych i komunikację szeregową. Funkcja loop() jest główną funkcją, która będzie działać iteracyjnie do momentu wyłączenia zasilania. Operacje na danych i manipulacje we/wy są zwykle wykonywane w ramach tych funkcji. Urządzenia IoT są zazwyczaj zaprojektowane do komunikowania się z innymi urządzeniami w sieci przy użyciu określonych protokołów przesyłania wiadomości, takich jak CoAP, MQTT oraz Extensible Messaging and Presence Protocol (XMPP). Chociaż

istnieje kilka możliwych protokołów, które można wykorzystać, jak wcześniej wspomniano, MQTT zostanie tutaj pokrótce opisany, ponieważ zostanie wykorzystany w projekcie Arduino. MQTT to lekki protokół przesyłania wiadomości zaprojektowany dla urządzeń z ograniczonym sprzętem, które są zwykle połączone z zawodnymi lub stratnymi sieciami o ograniczonej i nieprzewidywalnej przepustowości oraz dużym opóźnieniu. W chwili pisania tego tekstu najnowsza wersja protokołu to 3.1.1, a ostatnia recenzja odbyła się w październiku 2014 r. MQTT jest zbudowany na protokole kontroli transmisji (TCP). Protokół ma na celu łączenie wbudowanych urządzeń i sieci za pomocą aplikacji i oprogramowania pośredniczącego przy użyciu wzorca publikowania/subskrybowania, co pozwala na prostą implementację. MQTT składa się z trzech elementów: (1) brokera, (2) subskrybenta i (3) wydawcy. Zarówno subskrybent, jak i wydawca są klientami brokera, jak wyjaśniono:

1) Broker. Broker działa jako brama; odbiera komunikaty od wydawcy (klienta) i dostarcza komunikaty do subskrybenta (innego klienta). Brokerzy są czasami określani jako serwery.

2) Abonent. Subskrybent deklaruje brokerowi interesujące go tematy, a broker wysyła do tych tematów wiadomości publikowane.

3) Wydawca. Wydawca wysyła komunikaty do brokera przy użyciu przestrzeni nazw lub nazwy tematu, a broker przekazuje komunikaty do odpowiednich subskrybentów.

W MQTT klienci mogą subskrybować wiele tematów, a subskrybent może wypisać się z dowolnego tematu. Ponadto MQTT ma funkcję dostarczania wiadomości typu jeden do wielu. Protokół ma również poziomy jakości usług (QoS), które zapewniają, że wiadomość jest dostarczana lub przesyłana od nadawcy do odbiorcy. W MQTT istnieją trzy poziomy QoS: co najwyżej raz, co najmniej raz i dokładnie raz. Przykładem ilustrującym projektowanie i prototypowanie projektów IoT z wykorzystaniem opisanych powyżej narzędzi i protokołu MQTT jest rejestrator temperatury, w którym aktualna temperatura jest publikowana za pośrednictwem wydawcy MQTT do brokera MQTT.

Projektowanie i prototypowanie IoT przy użyciu platform Raspberry Pi

Jak wspomniano, jedną z głównych różnic między Raspberry Pi a płytą Arduino jest to, że ta pierwsza jest komputerem opartym na procesorze System on Chip, podczas gdy ta druga jest programowalnym mikroprocesorem. Oznacza to, że system operacyjny jest niezbędny do korzystania z Raspberry Pi do projektowania i prototypowania projektów IoT. Wybór systemu operacyjnego jest zwykle motywowany jego funkcjami, typem i wymaganiami aplikacji, którą programista zamierza zaprojektować. Lista oficjalnie zgodnych systemów operacyjnych dla tego typu urządzeń jest dostępna na stronie internetowej Fundacji Raspberry Pi. Projektowanie i prototypowanie projektów IoT z wykorzystaniem obu platform jest w pewnym sensie podobne. Zwykle projektant zaczyna od sprawdzenia, jaki sprzęt i oprogramowanie jest niezbędne do pożądanego projektu IoT. Może to obejmować użycie arkuszy danych sprzętowych, modułów i bibliotek modułów, osłon Raspberry Pi i/lub dodatkowego oprogramowania. Moduły czujnikowe i wykonawcze można również podłączyć do platformy za pomocą nagłówka GPIO urządzenia. Raspbian zawiera aplikację kontrolera GPIO, która umożliwia użytkownikom odczytywanie i zapisywanie stanu pinów GPIO. Aby ułatwić integrację modułu w projekcie, który jest projektowany i prototypowany, dostępne są dla projektanta biblioteki modułów dla różnych języków programowania, które zazwyczaj są pisane przez społeczność lub przez producenta modułu. Nakładki dla Raspberry Pi to płytki rozszerzające umożliwiające integrację modułów z dodatkowymi funkcjonalnościami, takimi jak moduły LCD, moduły odbiornika GPS oraz moduły połączenia internetowego (GPRS/HSDPA). Główna różnica między rozwojem aplikacji IoT dla Raspberry Pi i dla płyt Arduino leży w językach programowania. Podczas gdy C/C++ jest używany do programowania płyt Arduino, Python jest często używany do programowania Raspberry Pi ze względu na jego prostotę i czytelność, co pozwala początkującym projektantom szybko nadrobić zaległości.

Jedną z metod projektowania części sprzętowej projektu IoT jest użycie aplikacji do projektowania wspomaganego komputerowo (CAD). CAD może pomóc w rysowaniu fizycznych i schematycznych diagramów projektu. Prototypowanie aplikacji IoT dla platformy Raspberry Pi można rozwijać w samym urządzeniu za pomocą edytorów tekstu, takich jak vi, nano, gedit i leafpad. Inne opcje obejmują stosowanie IDE, takich jak NINJA-IDE, Adafruit i WebIDE, a także używanie kompilatorów dla danego używanego języka programowania. Projekt IoT zwykle korzysta z protokołu komunikacyjnego w celu zapewnienia zgodności i zgodności. Pozwala to np. projektowi łączyć się z innymi urządzeniami, które w sieci działają według tego samego protokołu. Protokoły komunikacji aplikacji, takie jak MQTT, CoAP, XMPP i aplikacja internetowa Interfejsy programistyczne (API) są zwykle wdrażane podczas projektowania i prototypowania rozwiązań IoT. Przykładem projektowania i prototypowania projektu Raspberry Pi IoTapplication, który można wykorzystać do zilustrowania prototypowania za pomocą wspomnianych powyżej narzędzi, jest sterowany przez Internet projekt diody elektroluminescencyjnej (LED). W tym projekcie dioda LED podłączona do Internetu jest włączana i wyłączana poprzez dostęp do serwera WWW. Chociaż w tym przykładzie używana jest dioda LED, do zastąpienia diody LED można użyć dowolnego siłownika, takiego jak silnik elektryczny.

Projekty dotyczące aplikacji IoT

Aby zrozumieć, w jaki sposób można używać platform Arduino i Raspberry Pi, w tej sekcji opisano podstawowe wykorzystanie środowiska Arduino IDE i sposób pisania kodu w języku C, który może manipulować cyfrowymi pinami, a także wyjaśnia, jak przesłać kod na płytkę Arduino . Obejmuje również podstawowe zastosowanie platformy Raspberry Pi, w tym sposób konfiguracji pinów GPIO za pomocą skryptów Bash Shell i języka programowania Python.

Projekt Arduino dla aplikacji IoT

Celem tego projektu jest przekazanie odczytów temperatury zebranych przez czujnik do brokera MQTT. W tym przykładzie wykorzystano router z połączeniem Ethernet i Wi-Fi, rezystor 4,7 kΩ, czujnik temperatury Maxim Integrated DS18B20, moduł Ethernet ENC28J60 oraz dwie aplikacje: Arduino IDE21, gdzie zostanie opracowany program mikroprocesorowy, oraz Mosquitto, broker MQTT o otwartym kodzie źródłowym. Zakłada się, że komputer, na którym będzie rozwijany projekt, jest podłączony do routera za pośrednictwem sieci Ethernet lub Wi-Fi. W celach demonstracyjnych komputerowi przypisywany jest adres IP 192.168.1.1.

Instalowanie Arduino IDE i oprogramowania Mosquitto

W systemie Linux za pomocą terminala Linux można zainstalować Arduino IDE (arduino i arduino-core) oraz pakiety Mosquitto (klienci mosquitto i mosquitto), a także ich zależności. Pamiętaj, że wymagane pakiety zostaną zainstalowane tylko za pomocą następującego polecenia, jeśli używasz menedżera pakietów oprogramowania Debiana:

$> sudo apt-get update

$> sudo apt-get install arduino arduino-core mosquitto

mosquitto-clients

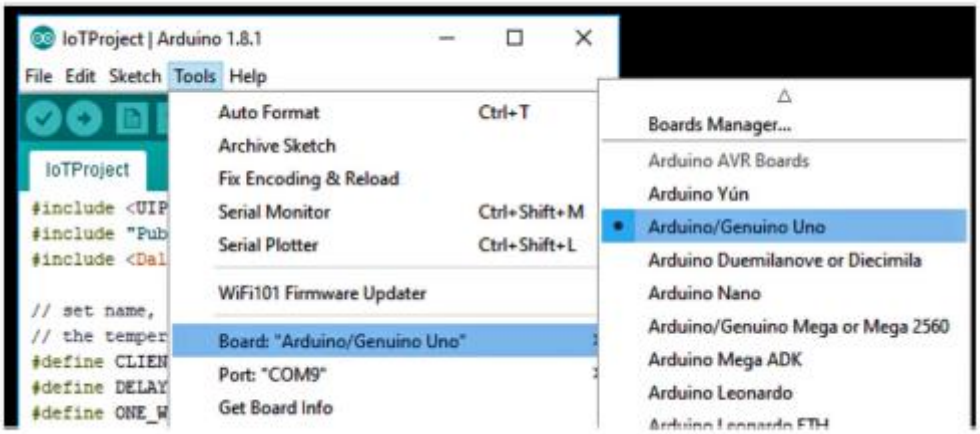
W systemie Windows obie aplikacje można pobrać z ich oficjalnych stron internetowych. Po zakończeniu pobierania oprogramowanie można zainstalować, wykonując pobrane pliki. Należy pamiętać, że wersja Mosquitto dla systemu Windows wymaga OpenSSL i biblioteki pthreads dla systemu Windows. Instrukcje i oba linki do pobrania są podawane użytkownikowi podczas instalacji Mosquitto.

Korzystanie z Arduino IDE i pobieranie bibliotek

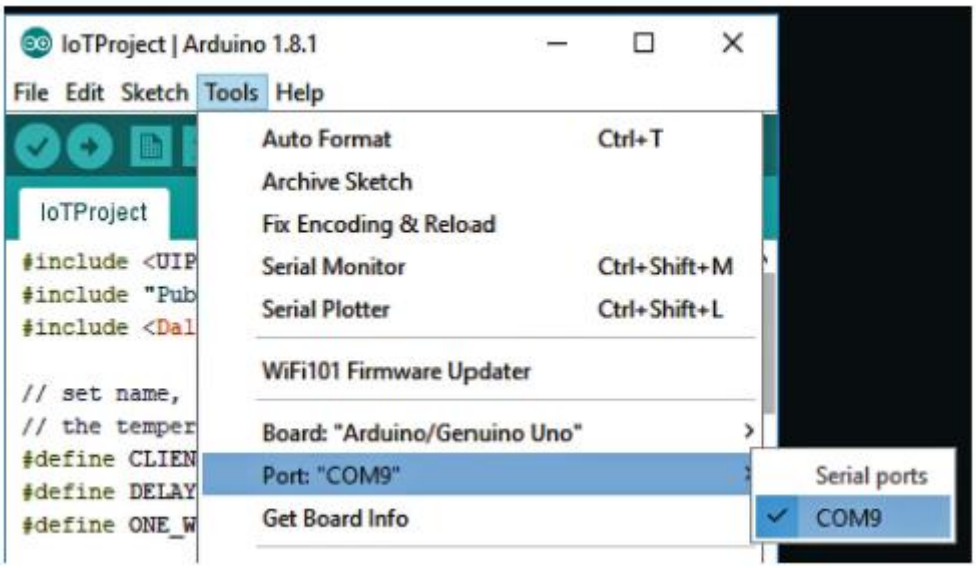
Arduino IDE należy uruchomić z uprawnieniami administratora, co można zrobić, otwierając terminal Linux i wykonując następujące polecenie:

$> sudo arduino &

Po otwarciu IDE kroki dla obu systemów operacyjnych (Linux i Windows) są podobne. W tym momencie płyta rozwojowa musi być podłączona do komputera, aby można było skonfigurować IDE. Odbywa się to poprzez wybranie płytki rozwojowej i określenie portu szeregowego, który ma być użyty do przesłania kodu przez menu płytki i portu w Narzędziach. Po otwarciu menu Płytka, w którym pojawi się lista kompatybilnych płyt rozwojowych z IDE, użytkownik może wybrać żądaną płytkę, którą w tym przykładzie jest Arduino/Genuino Uno, jak pokazano na rysunku.

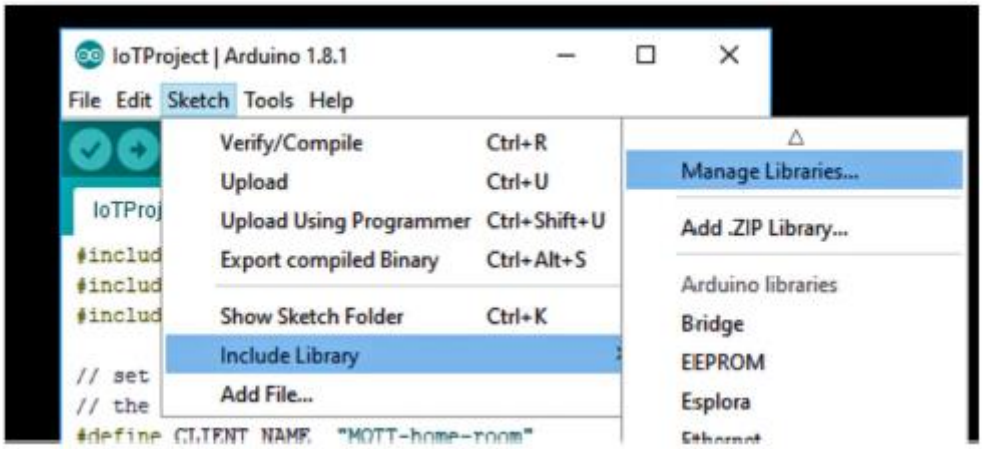


Port, który będzie używany, można skonfigurować, przechodząc do menu Narzędzia, a następnie wybierając menu Port. W tym menu, zakładając, że do komputera nie ma już podłączonych urządzeń szeregowych, użytkownik musi wybrać jedyną dostępną pozycję, np./dev/ttyUSB∗ lub COM∗, gdzie ∗ oznacza numer odpowiadający portowi szeregowemu płyty rozwojowej , jak pokazano na rysunku.



Kolejnym krokiem jest pobranie niezbędnych bibliotek do rozwoju aplikacji Arduino, które zostaną wgrane na płytkę rozwojową. Od wersji 1.6.2 Arduino IDE zawiera funkcjonalność, która pozwala

programiście zarządzać bibliotekami bez pobierania ich za pomocą przeglądarki. Dostęp do tej funkcji można uzyskać poprzez menu Szkic, Dołącz bibliotekę i wybierając opcję Zarządzaj bibliotekami, jak pokazano na rysunku.



Korzystając z paska wyszukiwania, użytkownik może pobrać wymagane biblioteki, które są następujące: UIPEthernet, OneWire, DallasTemperature i PubSubClient.

Kod źródłowy projektu i schemat połączeń Arduino

Listing 1 przedstawia typową implementację aplikacji (w języku C), która wykorzystuje protokół MQTT do publikowania pomiarów temperatury do brokera MQTT (proszę zwrócić uwagę na komentarze w tekście):

```
#include < UIPEthernet.h >

#include "PubSubClient.h"v

#include < DallasTemperature.h >

// set name, interval between readings,

// and the temperature module PIN

#define CLIENT_NAME "MQTT-home-room"

#define DELAY 5000

#define ONE_WIRE_PIN 7

// Set MAC address and IP. Just make sure you don’t have

// the same MAC / IP addresses set in different devices

uint8_t mac[6] = {0x00, 0x54, 0x45, 0x4d, 0x50, 0x01};

IPAddress ip(192, 168, 1, 2);

// Declare the modules variables

EthernetClient ecEth;

PubSubClient pscMQTTClient;
```

```
OneWire owOneWire(ONE_WIRE_PIN);
DallasTemperature dtSensor(&owOneWire);
// Declare variables
long lPrevMillis;
float fTemp;
void setup(){
// Setup serial connection and modules
Serial.begin(9600);
dtSensor.begin();
Ethernet.begin(mac, ip);
// Set the ethernet module variable to PubSub library
// and set the gateway / broker IP and Port.
// Assume the broker is running on
// ip-address: 192.168.1.1
// port-number: 1883
(this is the default port of the MQTT Broker)
pscMQTTClient.setClient(ecEth);
pscMQTTClient.setServer("<ip&hyphen;address>",
< port-number >);
}
void loop() {
// If the defined delay is exceeded.
if(millis() - lPrevMillis >= DELAY) {
dtSensor.requestTemperatures(); //
Request temperature to the module
fTemp = dtSensor.getTempCByIndex(0); //
and set it to a variable again
publishMessage(fTemp);
lPrevMillis = millis();
}
pscMQTTClient.loop(); //
```

Maintain the connection with the broker

}

void publishMessage(float fTemp) {

char caMsgBuf[10];

if(pscMQTTClient.connect(CLIENT_NAME)) {

pscMQTTClient.publish("home/room/temp", dtostrf

(fTemp, 6, 2, caMsgBuf));

// Note the MQTT topic where the values will be published

(home/room/temp)

// The topic may describe, with few words, what the

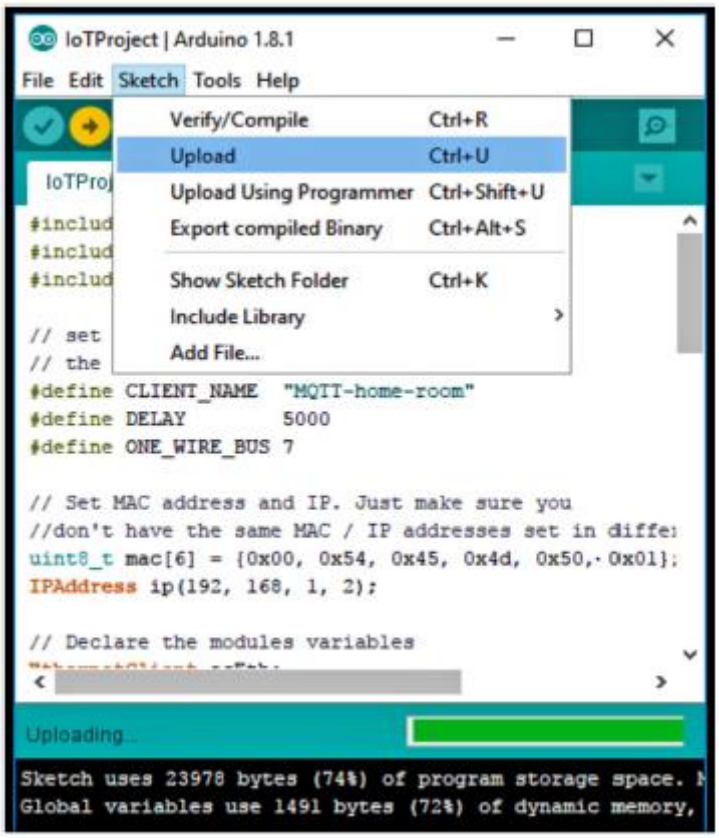
code will publish: in this example, we are

publishing temperature data from a room within a home.

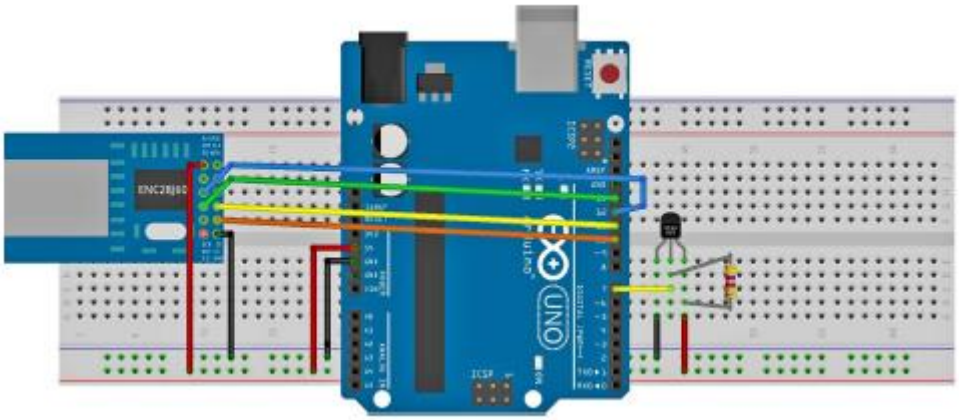
}

}

Po napisaniu kodu można go skompilować i wgrać na płytkę rozwojową Arduino. Można to zrobić wchodząc do menu Szkic i wybierając Prześlij, jak pokazano na rysunku.



Użytkownikowi zostanie wyświetlony pasek ukończenia. Ponieważ komponenty nie są połączone z płytką, kod można wykonać, ale nic się nie stanie. W tym momencie płytkę można odłączyć od komputera w celu bezpiecznego połączenia komponentów z płytką Arduino. Rysunek 6.7 pokazuje, w jaki sposób różne komponenty są połączone na płytce stykowej za pomocą przewodów połączeniowych. Po pełnym podłączeniu płyty Arduino i innych komponentów, jak pokazano na rysunku, użytkownik może podłączyć kabel Ethernet do modułu ENC28J60 i routera.



Po podłączeniu kabla Ethernet, zarówno broker MQTT jak i abonent MQTT mogą być wykonane z domyślnymi ustawieniami w celu przetestowania aplikacji Arduino.

Uruchamianie Mosquitto i testowanie aplikacji

W przypadku obu systemów operacyjnych sugeruje się, aby instancja brokera Mosquitto była otwarta w jednym oknie, a instancja subskrybenta Mosquitto w innym. W systemie Linux obie aplikacje można uruchomić, otwierając dwa różne okna terminala. Następujące polecenia mogą być wykonywane w tej samej kolejności, jedna w pierwszym oknie, a druga w drugim oknie:

$> sudo mosquito

$> sudo mosquito_sub -h localhost -p 1883 -t “/home/room/temp”

W przypadku systemu Windows użytkownik może skorzystać z funkcji Run systemu, naciskając klawisze CTRL + R, a następnie wpisując ścieżkę bezwzględną do brokera Mosquitto, którą w tym przykładzie przyjmuje się C:/Program Files (x86)/mosquitto/ mosquitto.exe. Aplikację subskrybenta Mosquitto należy zainstalować w tej samej ścieżce co broker Mosquitto, co można wykonać otwierając funkcjonalność Uruchom i wpisując:

“C:/Program Files (x86)/mosquitto/mosquitto_sub.exe”

-h localhost -p 1883 -t ”/home/room/temp” v

Flaga -t służy do określenia, który temat MQTT będzie subskrybowany. Należy zauważyć, że zarówno broker Mosquitto, jak i subskrybent Mosquitto nie będą przedstawiać żadnych komunikatów, dopóki aplikacja Arduino nie opublikuje ich w brokerze. Mając komputer przygotowany do odbioru danych z Arduino, można go włączyć i zobaczyć przychodzące wiadomości na subskrybenta Mosquitto. Broker Mosquitto przedstawia komunikaty dziennika, w których ogłaszane są połączenia subskrybenta i wydawcy. Podczas gdy dziennik jest prezentowany w brokerze Mosquitto, wiadomości zaczynają być wyświetlane w subskrybencie Mosquitto, jak widać na zrzucie ekranu PuTTY (bezpłatny telnet, serwer

SSH i oprogramowanie terminala portów szeregowych, które umożliwia użytkownikom zdalny dostęp do urządzeń przez Internet). Jest to prosta aplikacja IoT, która działa jako publikator temperatury w środowisku MQTT. Inną interesującą aplikacją, którą można opracować w celu uzupełnienia opisanej tutaj, jest subskrybent MQTT, który można skonfigurować tak, aby wysyłał wiadomość alarmową, gdy temperatura wzrośnie powyżej pewnego progu.

Projekt Raspberry Pi dla aplikacji IoT

Podobnie jak starsze wersje, nowsze wersje SBC Raspberry Pi zawierają również 40-pinowe złącza GPIO lub możliwość ich lutowania. Oznacza to, że opisany tutaj projekt można powielić na wszystkich dostępnych na rynku wersjach Raspberry Pi. Projekt oparty jest na Raspberry Pi Model B Rev. 2 i wykorzystywany jest system operacyjny Raspbian. System operacyjny jest preinstalowany z niektórymi niezbędnymi dla projektu oprogramowaniem, takimi jak interpreter Pythona i narzędzie do manipulacji GPIO.

Instalowanie systemu operacyjnego

W przypadku użytkownika systemu Linux obraz Raspbian można skopiować na czystą kartę micro SD za pomocą narzędzia dd lub dowolnego odpowiedniego narzędzia. Przed zapisaniem obrazu na kartę pamięci należy podłączyć ją do komputera, zidentyfikować literę urządzenia i upewnić się, że jest odmontowana. Po odmontowaniu woluminu następujące polecenie pozwoli użytkownikowi zapisać obraz systemu operacyjnego na karcie pamięci, gdzie ∗ jest literą identyfikatora karty micro SD:

$> dd if=~/2017-01-11-raspbian-jessie.img of=/dev/sd* bs=4M

W przypadku użytkownika systemu Windows obraz Raspbian można skopiować na kartę micro SD za pomocą narzędzia Win32DiskImager, które jest dostępne na stronie internetowej projektu Sourceforge. Przed zapisaniem obrazu na karcie pamięci należy uruchomić narzędzie i wybrać plik obrazu za pomocą opcji selektora pliku. Po wybraniu pliku użytkownik może wybrać opcję Zapisz, a obraz zostanie zapisany na karcie micro SD. Po zapisaniu obrazu na karcie micro SD użytkownik może włożyć kartę do gniazda karty micro SD w urządzeniu Raspberry Pi. Użytkownik może teraz podłączyć wszystkie urządzenia peryferyjne potrzebne do korzystania z urządzenia (mysz, klawiatura, kabel Ethernet i monitor). Wreszcie urządzenie można włączyć, podłączając kabel USB do zasilania. W tym przykładzie Ethernet będzie używany do celów połączenia. Po uruchomieniu Raspberry Pi uruchomi skrypt powłoki Bash, prosząc użytkownika o zalogowanie. Domyślnym loginem dystrybucji Raspbian jest nazwa użytkownika Pi i hasło raspberry.

Pobieranie i instalowanie wymaganych pakietów

Pierwszym krokiem po zalogowaniu jest upewnienie się, że urządzenie jest połączone z Internetem poprzez pingowanie strony internetowej. Polecenie ping służy do testowania łączności między dwoma urządzeniami poprzez pomiar czasu między pakietem żądania a odpowiednim pakietem odpowiedzi. Następujące polecenie spowoduje, że urządzenie wyśle dokładnie cztery pakiety ping (-c 4) na jeden z serwerów Google:

$> ping www.google.com -c 4

Chociaż niektóre oprogramowanie jest preinstalowane, dla projektu potrzebne są dodatkowe pakiety Linux, takie jak pakiet wirePi (wiringpi), menedżer pakietów Python (python-pip) i pliki nagłówkowe dla języka Python (pythondev):

$> sudo apt-get install python-dev python-pip wiringpi

Po pomyślnym zainstalowaniu Menedżera pakietów języka Python do tego projektu będą potrzebne dwa moduły języka Python. Pierwszym z nich jest moduł wirePi, który pozwala użytkownikowi manipulować pinami GPIO za pomocą Pythona. Drugi to Flask, mikro framework do tworzenia stron internetowych dla Pythona. Oba moduły można zainstalować za pomocą następującego polecenia:

$> sudo pip install flask wiringPi

Ponieważ wszystkie zależności są zainstalowane w systemie, należy być w stanie rozpocząć tworzenie projektu. Przed rozpoczęciem kodowania sugeruje się uruchomienie polecenia gpio readall. Da to użytkownikowi przegląd stanów pinów GPIO, a także numerację pinów, która zostanie wyświetlona w tabeli na ekranie. Istnieje inna numeracja ze względu na różne implementacje biblioteki GPIO. Polecenie wygląda następująco:

pi@raspberrypi:~ $ gpio readall

Polecenie gpio pozwala użytkownikom kontrolować piny GPIO na urządzeniu (tj. ustawiać tryb, zmieniać ich wartości itp.). Użytkownik może teraz skonstruować układ elektroniczny na płytce stykowej. W tym momencie zaleca się wyłączenie urządzenia, wpisując w terminalu następujące polecenie:

$> sudo shutdown now

Konstruowanie i testowanie obwodu

Gdy urządzenie jest wyłączone, diodę LED i rezystor 4,7 kΩ można podłączyć na płytce stykowej za pomocą dwóch zworek. Dodatni pin (biegun) diody LED jest zwykle dłuższy niż ujemny. W związku z tym należy upewnić się, że dodatni pin diody LED jest podłączony do pinu GPIO 15 nagłówka GPIO na Raspberry Pi, a ujemny pin diody LED jest podłączony do dowolnego z pinów rezystora. Drugi pin rezystora należy podłączyć do pinu GND (masy) złącza GPIO Raspberry Pi. Po podłączeniu komponentów można włączyć Raspberry Pi. Gdy urządzenie poprosi o wprowadzenie danych logowania, można wprowadzić domyślną nazwę użytkownika i hasło Raspbian OS. Zaraz po kroku logowania można manipulować pinami GPIO za pomocą polecenia gpio. Używany pin (GPIO 15) musi zostać skonfigurowany w tryb wyjścia, zanim będzie można manipulować jego stanem, co można wykonać za pomocą następującego polecenia:

$> sudo gpio mode 15 out

Po prawidłowym skonfigurowaniu pinu w tryb wyjścia, może on mieć dwa różne stany, WYSOKI i NISKI, które są reprezentowane odpowiednio przez 1 i 0. Piny GPIO wyślą 3,3 V, gdy są w stanie wysokim (z wyjątkiem pinów uziemienia i 5 V, co można zaobserwować na liście 2) oraz 0 V, gdy piny GPIO są w stanie LOW. Poniższe polecenie umożliwia zmianę stanu określonego pinu GPIO na WYSOKI:

$> sudo gpio write 15 1

Po wykonaniu tego polecenia w urządzeniu można zauważyć zaświecenie się diody LED. Alternatywnie, jeśli cyfra 1 w poleceniu zostanie zmieniona na 0, pin zostanie ustawiony na LOW, a dioda LED zgaśnie. Do włączania i wyłączania diody LED zostaną użyte dwa pliki skryptów Bash. Poniższe polecenia utworzą dwa pliki skryptów Bash o nazwach turnOn.sh i turnOff.sh. Każdy z tych plików pozwoli na włączanie i wyłączanie diody LED za pomocą wcześniej przetestowanych poleceń:

$> echo -e '#!\\bin\\bash\nsudo gpio mode 15 out\nsudo

gpio write 1' > turnOn.sh

$> cp turnOn.sh turnOff.sh && truncate -s-2 turnOff.sh

$> echo "0" >> turnOff.sh && chmod +x turn*.sh

Te pliki skryptowe pozwolą użytkownikowi manipulować pinem GPIO za pośrednictwem powłoki Bash i można je wykonać, po prostu wpisując ./turnOn.sh lub ./turnOff.sh. W związku z tym sterowanie może być wykonane tylko wtedy, gdy użytkownik ma bezpośredni dostęp do Raspberry Pi (biorąc pod uwagę połączenia SSH i Telnet są wyłączone).

Tworzenie i testowanie aplikacji internetowych w Pythonie

Kolejny krok to stworzenie małej aplikacji webowej, która będzie pełniła rolę przełącznika, pozwalającego użytkownikowi kontrolować stan diody LED za pomocą prostej przeglądarki. Poniższego polecenia można użyć do utworzenia pliku, który będzie zawierał skrypt aplikacji internetowej i zmianę uprawnień do pliku:

$> touch ledServer.py && chmod +x ledServer.py

Listing zawiera kod, który należy wpisać w pliku, a także objaśnienie każdego wiersza kodu w komentarzach wbudowanych. Plik można edytować i zapisywać za pomocą preferowanego przez użytkownika edytora tekstu (np. nano, vi lub gedit).

```
#!/usr/bin/python

import wiringpi # Import the wiringPi module for Python

from flask import Flask # Import the Flask

app = Flask(__name__)

pin, state = 15, 0

@app.route("/switch") # When someone access the http server endpoint

def switchLED(): # the switchLED() function is executed.

global state # It starts by using the global variable state,

if state == 0 : # and if the variable is LOW (0)

state = 1 # change it to HIGH (1)

else: # otherwise

state = 0 # change it to LOW

wiringpi.digitalWrite(pin, state)

# Write the state to the specified pin

return "{’pin’:"+str(pin)+", ’state’:"+str

(state)+"}" # Return a response with the pin and state

if __name__ == "__main__":

wiringpi.wiringPiSetup() # Initialization of the module

wiringpi.pinMode(pin, 1) # Input = 0; output = 1;
```

app.run(host='0.0.0.0', port=8080)

Run the application on all interfaces using port 8080

Giving the variable host the value 0.0.0.0, will allow

to run the application in all interfaces the device has available

Przed uruchomieniem serwera w urządzeniu konieczna jest znajomość adresu IP interfejsu Ethernet (eth0) Raspberry Pi, aby podczas pracy serwera można było uzyskać do niego dostęp przez przeglądarkę. Adres IP można poznać, uruchamiając polecenie ipconfig. Po przygotowaniu skryptu użytkownik może go wykonać, umożliwiając sterowanie diodą LED z poziomu przeglądarki poprzez wykonanie następującego polecenia:

$> ./ledServer.py

Teraz, gdy inne urządzenie jest podłączone do tej samej sieci, można otworzyć przeglądarkę i przejść do serwera za pomocą adresu IP Raspberry Pi. Ponieważ serwer został skonfigurowany do korzystania z portu 8080 urządzenia, można przejść do http://<ip-address>:8080/switch i sprawdzić, czy dioda LED włącza się lub wyłącza zgodnie ze stanami HIGH lub LOW.

Wniosek

IoT szybko staje się wszechobecny w życiu codziennym i daje początek niezliczonym zastosowaniom. Prawdopodobnie postęp i konwergencja sprzętu i oprogramowania w jednym pakiecie, znanym również jako zestaw rozwojowy, bez wątpienia pozostaną centralnym szkieletem zapewniającym znaczny wzrost i wsparcie dla wdrożeń IoT. W tej części przedstawiono podstawowe, ale kompleksowe wprowadzenie do projektowania i prototypowania IoT. Szczegółowo opisał główne cechy różnych platform programistycznych IoT, skupiając się na szeregu płyt Arduino i niektórych modelach z rodziny Raspberry Pi. Omówiono również projektowanie i prototypowanie w kontekście IoT, gdzie przedstawiono kilka podstawowych przykładów projektowania i prototypowania IoT, pokazujących krok po kroku procedury wdrażania projektów IoT zarówno w oprogramowaniu, jak i sprzęcie. Projekty te mają na celu nauczenie początkujących podstawowych zasad projektowania i prototypowania, a także pokazanie, jak łatwo jest rozpocząć projektowanie i wdrażanie IoT.