Internet Rzeczy w przykładach (4)

Urządzenia pomiarowe w technologii Internet Rzeczy

W tym artykule zaprezentowano projekt urządzenia pomiarowego pracującego w technologii Internet of Things. Wyniki pomiarów są przechowywane w „chmurze" na serwerach plot.ly. Korzystając z możliwości serwisu plot.ly wykonamy wykres wartości pomiarowych. Do budowy urządzenia wykorzystamy moduł startowy CC3200 LaunchPad.

Charakterystyczną cechą urządzeń pomiarowych pracujących w technologii Internet Rzeczy jest transmisja danych pomiarowych w obrębie sieci Internet. Dane pomiarowe umieszczane są na serwerach, gdzie poddawane są analizie. Na potrzeby kursu zaprojektujemy urządzenie do pomiaru temperatury otoczenia. Dane pomiarowe prześlemy do serwerów plot.ly. Urządzenie zoptymalizujemy pod kątem poboru mocy.

Budowa

Urządzenie do pomiaru temperatury otoczenia zostało zbudowane w oparciu o moduł startowy CC3200 LaunchPad. Do modułu został dołączony czujnik temperatury MCP9700. Urządzenie jest zasilane z dwóch akumulatorków AA (napięcie 1,2 V, pojemność 2500 mAh każdy). Moduł CC3200 LaunchPad oraz akumulatorki zostały umieszczone w obudowie Z-30A o wymiarach 44 mm×70 mm×120 mm. Sonda do pomiaru temperatury otoczenia została wyprowadzona na zewnątrz obudowy. Dodatkowo, na obudowie zamontowano miniaturowy panel słoneczny (3 V/80 mA), który zostanie wykorzystany do ładowania akumulatorków. Wygląd urządzenia pokazano na **fotografii 1**.

Analogowy czujnik temperatury MCP9700 jest zasilany bezpośrednio z linii wyjścia mikrokontrolera CC3200 (pin 61). Zasilanie MCP9700 jest włączane na czas pomiaru temperatury. Poza pomiarami czujnik temperatury jest wyłączony. Wyjście analogowe czujnika temperatury zostało podłączone do kanału analogowego mikrokontrolera CC3200 (pin 58). Akumulatorki zostały połączone szeregowo, dzięki czemu powstało ogniwo zasilania o napięciu 2.4 Volt i pojemności 2500 mAh. Na wyjściu ogniwa zasilania zostały zamontowane dwa kondensatory o pojemności 470uF. Kondensatory mają za zadanie buforować spadki napięcia zasilania. Panel słoneczny został podłączony bezpośrednio do ogniwa zasilania. Schemat ideowy urządzenia prezentuje **rysunek 2**.

Budowę sterownika rozpoczynamy od zmian w module CC3200 LaunchPad. Żeby zmniejszyć pobór prądu modułu odlutowujemy diody D1 oraz D4. Po usunięciu diod podłączamy czujnik temperatury. Następnie usuwamy zworkę z złącza J13 (zasilanie modułu z zewnętrznego napięcia doprowadzonego do złącza J20) i do pinów Vcc, GND złącza J20 podłączamy zasilanie urządzenia.

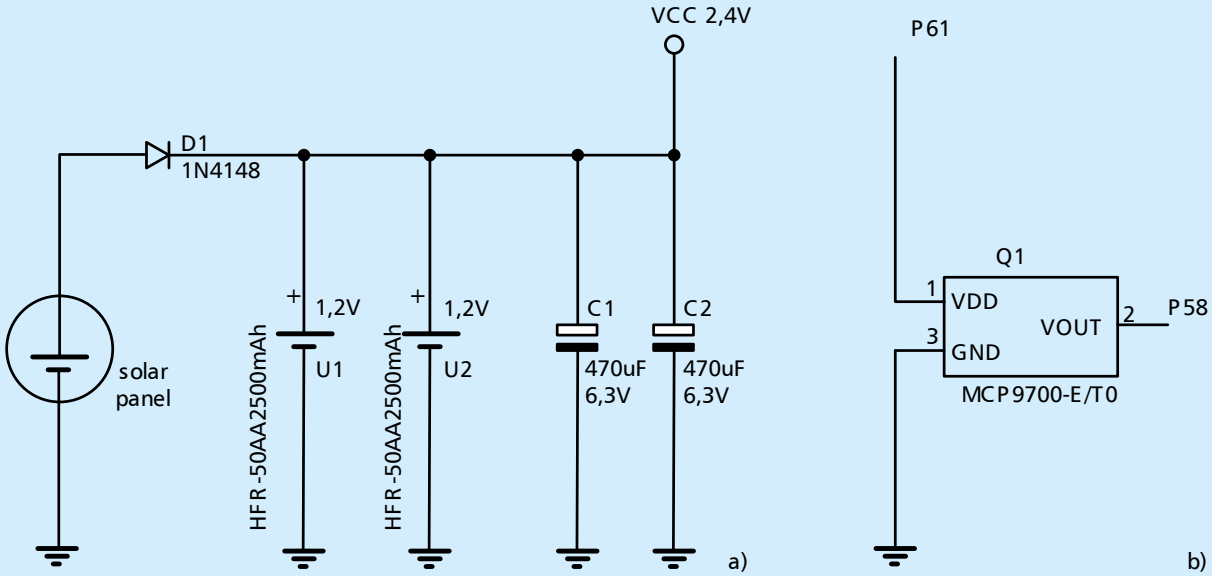
Fotografia 1. Budowa rządzenia pomiarowego

W obudowie urządzenia montujemy moduł CC3200 LaunchPad, akumulatorki i kondensatory buforujące. Sondę do pomiaru temperatury wyprowadzamy na zewnątrz obudowy. Na obudowie montujemy mini panel słoneczny.

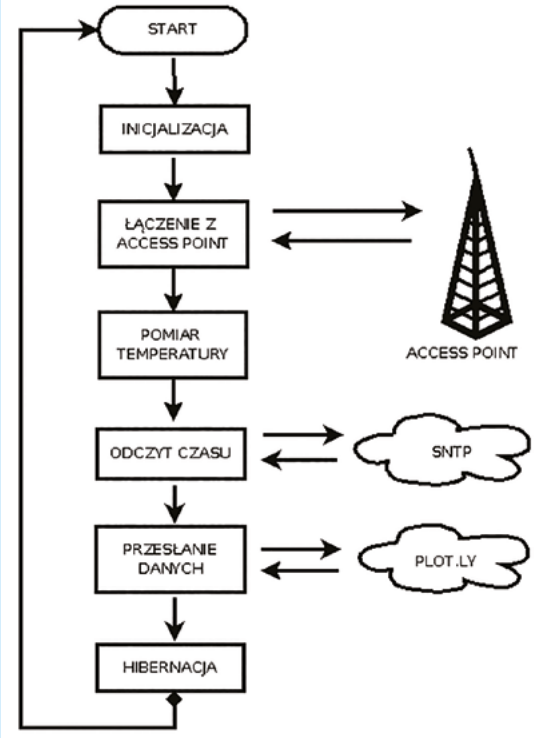
Funkcjonalność

Zaprojektowane urządzenie realizuje pomiar temperatury, a wynik przesyła do serwerów plot.ly. Wraz z próbką pomiarową przesyłany jest czas zakończenia pomiaru. Czas pobierany jest z „sieci" z serwera SNTP. Pomiary wykonywane są cykliczne co 5 minut. Pomiędzy pomiarami mikrokontroler CC3200 wprowadzany jest w tryb

Do poprawnej pracy urządzenie pomiarowe z modułem CC3200 LauchPad potrzebuje dostępu do sieci Wi-Fi (2,4 GHz, standard IEEE 802.11 b/g/n). Należy zapewnić możliwość komunikowania się urządzenia z Access Pointem. Punktem dostępu do sieci może być router, komputer PC, telefon komórkowy. Parametry transmisji (nazwa SSID dla Access Point, szyfrowanie transmisji, ew. hasło dostępu do Access Point) ustawiamy w plikach konfiguracyjnych projektu.



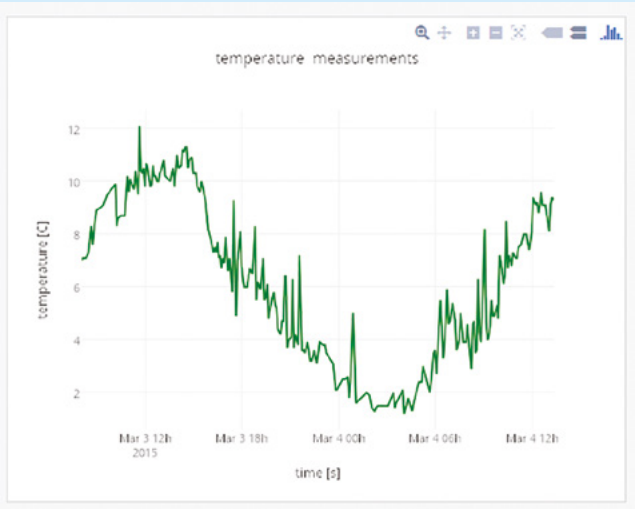
Rysunek 2. Schemat elektryczny urządzenia: a) moduł zasilania b) czujnik temperatury



Rysunek 3. Uproszczony schemat pracy urządzenia pomiarowego

uśpienia. W sposób graficzny zasadę pracy urządzenia pokazano na **rysunku 3**.

Po włączeniu zasilania konfigurowane są parametry pracy modułów peryferyjnych mikrokontrolera CC3200 (inicjalizacja UART, ADC, Watchdog, konfiguracja parametrów pracy w trybie hibernacji). Następnie urządzenie łączy się z Access Point. Po połączeniu z siecią internetową wykonywany jest pomiar temperatury. Następnie urządzenie łączy się do serwera czasu SNTP i odczytywane są dane z aktualnym czasem. W kolejnym kroku urządzenie łączy się z serwerem plot.ly i przesyła zmierzoną wartość temperatury oraz czas wykonania pomiaru. Na zakończenie urządzenie przechodzi w tryb hibernacji. W wypadku braku połączenia z Access Point, błędu połączenia z serwerem czasu SNTP, bądź błędu połączenia z serwerem plot.ly mikrokontroler przerywa opisany powyżej cykl pracy i bezwarunkowo przechodzi w tryb hibernacji. Z hibernacji mikrokontroler budzony jest po 5 minutach (czas liczony od startu programu). Po wyjściu z trybu hibernacji kod programu wykonywany jest od początku. Ciągłość pracy oprogramowania chroniona jest przez układ Watchdog. Domyślnie układ Watchdog jest wyłączony. W momencie aktywacji czas pracy układu Watchdog wynosi 50 sekund. Wykres prezentujący wyniki pomiarów dostępny jest w serwisie plot.ly pod adresem *http://goo.gl/8w5WCX*. Przykładowe dane pomiarowe zarejestrowane przez urządzenie pokazano na **rysunku 4**.

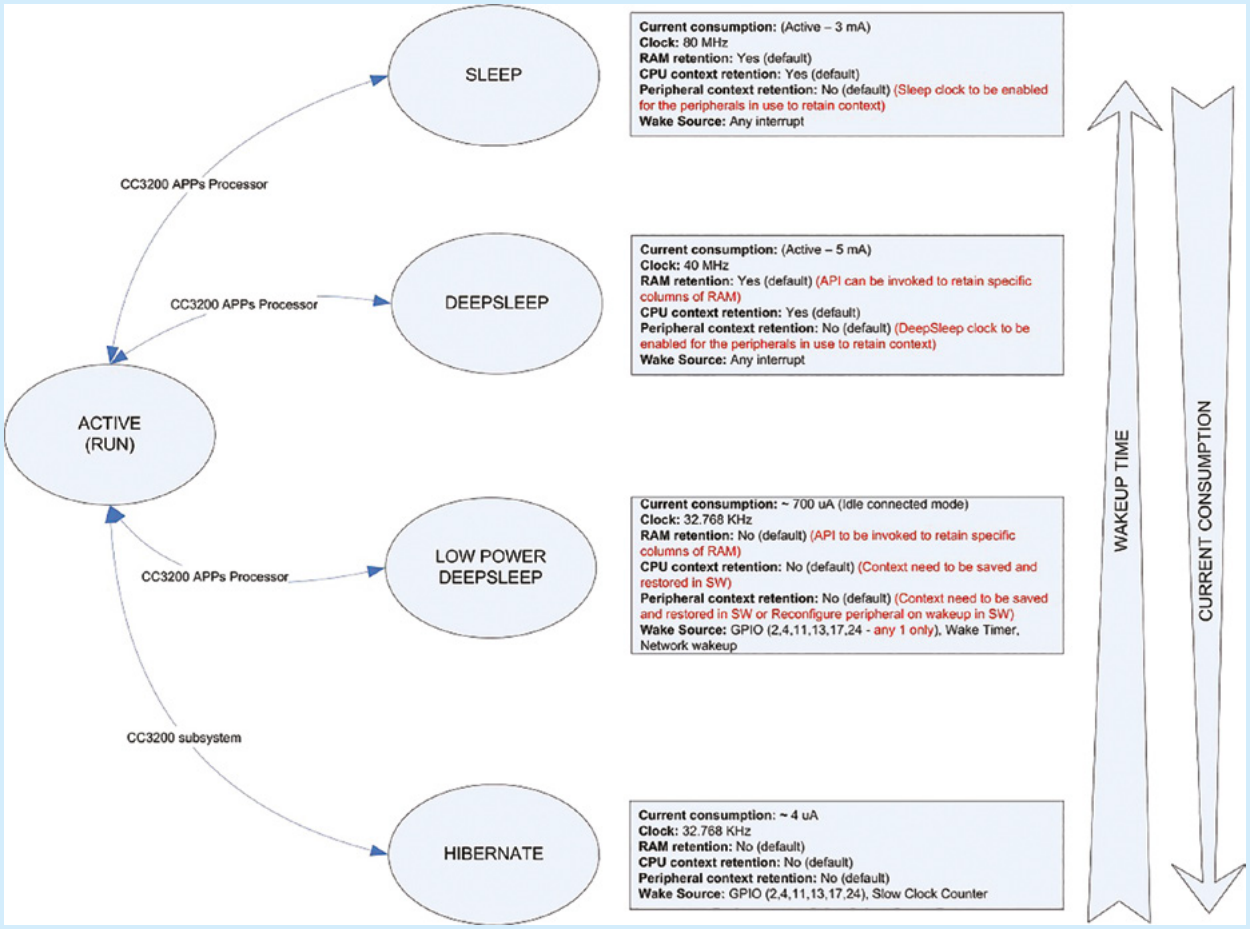


Rysunek 4. Pomiar temperatury. Wykres z serwisu plot.ly

Pomiary

Urządzenie pomiarowe skonstruowane na potrzeby artykułu wykonuje pomiary analogowe. Do wejścia analogowego mikrokontrolera CC3200 podłączony został czujnik temperatury MCP9700. Funkcjonalność urządzenia można rozbudować o obsługę dodatkowych czujników pomiarowych (analogowych, bądź cyfrowych).

Pomiary analogowe wykonywane są przy użyciu wbudowanego w mikrokontroler CC3200 przetwornika



Rysunek 5. Mikrokontroler CC3200. Tryby pracy energooszczędnej

A/C. Zainstalowany w CC3200 przetwornik A/C to układ typu SAR o rozdzielczości 12 bitów. Przetwornik obsługuje 8 kanałów pomiarowych (4 użytkownika, 4 wewnętrzne). Kanały wewnętrzne (CH1/CH3/CH5/CH7) używane są przez wbudowany koprocesor Wi-Fi. Dostęp do kanałów wewnętrznych jest zablokowany. Kanały użytkownika (CH0/CH2/CH4/CH6) pozwalają mierzyć napięcie analogowe podane na piny mikrokontrolera (57…60). Zakres pomiarowy napięcia wynosi od 0 do 1,45 V. Maksymalna prędkość próbkowania to 65,5 kS/s. Pomiary analogowe mogą być obsługiwane w trybie przerwań oraz z wykorzystaniem modułu DMA. Próbki pomiarowe mogą być znakowane czasem pomiaru (wbudowany licznik czasu). Wynik pomiaru dla kanału X jest zapisywany w 32-bitowym rejestrze CHANNELXFIFODATA. Dane pomiarowe zapisywane są na bitach 13:2, stempel czasu na bitach 30:14, pozostałe bity rejestru są nieużywane.

W prezentowanym urządzeniu analogowy czujnik temperatury MCP9700 został podłączony do pinu numer 58 mikrokontrolera CC3200 (kanał pomiarowy CH2). Czujnik jest zasilany bezpośrednio z wyjścia mikrokontrolera (pin numer 61). Zasilanie czujnika jest włączane na czas pomiaru. Po wykonaniu pomiaru czujnik jest wyłączany. Pomiar napięcia wykonywany jest w procedurze Adc12Read(). Realizowana jest seria 16 pomiarów, a wynik pomiaru jest uśredniany. Wartość zmierzonego napięcia obliczamy korzystając ze wzoru 1.1.

$$V_{IN} = \frac{N_{ADC}}{4095} * 1.45[V] \quad (1.1)$$

gdzie:
VIN – wartość mierzonego napięcia [V]
NADC – cyfrowy wynik pomiaru odczytany z rejestru

Parametry pracy czujnika temperatury MCP9700 definiuje wzór 1.2. Zmiana temperatury o 1°C powoduje zmianę napięcia na wyjściu czujnika o 0,01 V. Przy temperaturze 0°C wartość napięcia na wyjściu czujnika wynosi 0,5 V. Po zastosowaniu przekształceń matematycznych otrzymujemy wzór 1.3 pozwalający na obliczenie temperatury.

$$V_{IN} = 0.01.[V/^{O}C] * T[^{O}C] + 0.5[V] \quad (1.2)$$

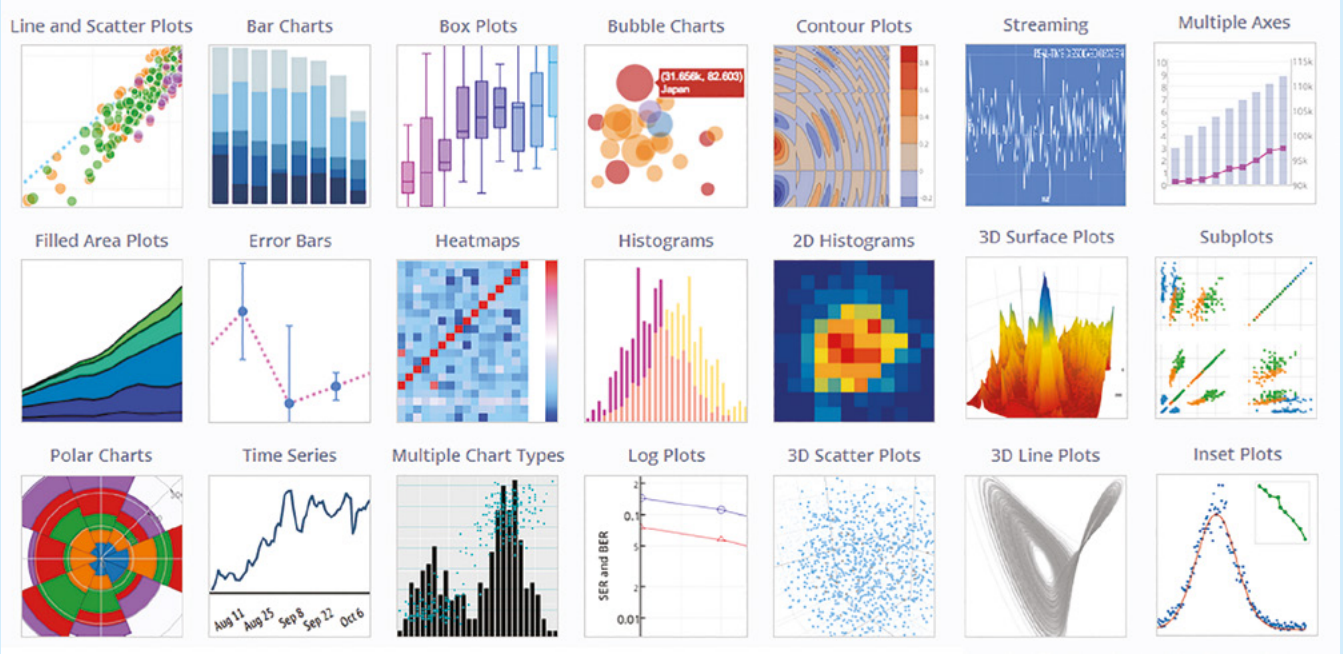
$$T = (V_{IN}[V] - 0.5[V]) * 100[V/^{O}C] \quad (1.3)$$

gdzie:
VIN – wartość napięcia na wyjściu czujnika [V]
T – temperatura otoczenia [°C]

Zakres pomiarowy czujnika MCP9700 wynosi od -40 do +125°C. Ponieważ maksymalna wartość napięcia mierzonego przez wejście analogowe mikrokontrolera CC3200 wynosi 1,45 V, to czujnik może być użyty do pracy w zakresie temperatury od -40 do +95°C.

Hibernacja

Mikrokontroler CC3200 został wyposażony w cztery tryby oszczędzania energii (**rysunek 5**). W urządzeniu pomiarowym prezentowanym w artykule użyty został tryb hibernacji. W trybie hibernacji pobór prądy mikrokontrolera wynosi zaledwie 4 uA. Wyłączony jest rdzeń procesora, nie działają moduły urządzeń peryferyjnych, nie jest odświeżana pamięć RAM (za wyjątkiem dwóch rejestrów o rozmiarze 32 bity każdy). Aktywny jest



Rysunek 6. Przykładowe wykresy dostępne w serwisie plot.ly

jedynie 48-bitowy licznik Slow Clock Couter taktowany za pomocą rezonatora „zegarkowego" o częstotliwości 32768 Hz oraz wybrane wejścia mikrokontrolera (2, 4, 11, 13, 17, 24) tzw. wejścia HibWakeUp. Z hibernacji mikrokontroler może być budzony przez licznik oraz przez zmianę stanu na wejściu HibWakeUp. Po wyjściu z trybu hibernacji mikrokontroler jest ponownie uruchamiany. Kod programu startuje od początku (jak po restarcie).

W oprogramowaniu urządzenia pomiarowego do budzenia mikrokontrolera CC3200 z trybu hibernacji użyty został licznik Slow Clock Couter. Czas pracy licznika ustawiany jest przy starcie programu w procedurze HibernateInit i wynosi 5 minut. Tryb hibernacji aktywowany jest w procedurze HibenateEnter. Obie procedury umieszczone zostały w pliku hibernate.c. Mikrokontroler budzony jest cyklicznie co 5 minut. Wykonuje powierzone mu zadania i ponownie jest wprowadzany w tryb hibernacji.

Serwis plot.ly

Serwis plot.ly umożliwia analizę oraz wizualizację danych pomiarowych. Możliwości serwisu zostały mocno rozbudowane (wykresy liniowe, punktowe, słupkowe, kołowe, płaszczyznowe itd.). Przykłady wykresów z serwisu plot.ly pokazano na **rysunku 6**.

Żeby rozpocząć pracę z serwisem plot.ly należy utworzyć konto użytkownika. Uruchamiamy serwis plot.ly i rejestrujemy nowego użytkownika. Po zakończeniu rejestracji mamy możliwość utworzenia nielimitowanej liczby publicznych wykresów oraz dziesięciu wykresów prywatnych (w wersji darmowej liczba prywatnych wykresów przypisanych do konta została ograniczona do dziesięciu). Wszystkie wykresy użytkownika dostępne są w zakładce Workspace. Istnieje możliwość zarządzania utworzonymi wykresami, tworzenia nowych wykresów, importowania wykresów. W zakładce API Libraries zostały udostępnione biblioteki do obsługi serwisu plot.ly na platformach: Exel, Matlab, Python, R, Julia, Node.js, Arduino. Podczas korzystania z bibliotek wymagane jest podanie klucza API przypisanego do konta użytkownika. Wartość klucza zapisana jest w ustawieniach konta użytkownika w zakładce Settings.

W oprogramowaniu urządzenia pomiarowego połączenie TCP do serwera plot.ly (gniazdko TCP) tworzone jest w procedurze UpdateDataTask. Dane do serwera wysyłane są metodą POST protokołu HTTP. W oprogramowaniu zaimplementowana została uniwersalna procedura SetPlotLyData służąca do przesyłania danych pomiarowych. W parametrach procedury należy podać: indeks gniazdka TCP, wartość próbki pomiarowej, nazwę pliku z danymi pomiarowymi, tytuł wykresu, nazwę osi OX, nazwę osi OY. Dane pomiarowe umieszczane są na osi OY. Na osi OX prezentowany jest czas pomiaru.

W procedurze SetPlotLyData jest budowana treść zapytania do serwera plot.ly (zapytanie HTTP metoda POST), definiowany typ wykresu, wprowadzane są wartości „x" i „y", definiowane są kolor, typ i rozmiar linii wykresu oraz kolor i wielkość markerów do oznaczania próbek pomiarowych. Wprowadzana jest nazwa wykresu, nazwa pliku z danymi pomiarowymi, opisy dla osi OX i OY. Szczegółowy opis parametrów wykresów plo. tly dostępny jest pod adresem plot.ly/learn. Dodatkowo w materiałach dołączonych do artykułu udostępniamy formularz www o nazwie plotly.html. Korzystając z formularza można „ręcznie" przesłać dane do serwera plot. ly. Formularz stanowi proste narzędzie do komunikacji z serwerami plot.ly. Korzystanie z formularza może okazać się pomocne w procesie projektowania wykresów.

Oprogramowanie

Oprogramowania urządzenia pomiarowego zostało stworzone w środowisku CCSv6. Projekt napisany został w języku programowania C. W projekcie użyty został system czasu rzeczywistego freeRTOS, framework SimpleLink, drivery dla CC3200. Dodatkowo wykorzystano przygotowane przez Texas Instruments interfejsy obsługi urządzeń peryferyjnych i sieci (uart, udma, wdt, network). Pliki z kodem źródłowym interfejsów

Oprogramowanie urządzenia pomiarowego zostało stworzone w środowisku programistycznym CCSv6. Utworzony został projekt o nazwie iot_measurements. W projekcie obsługiwane są system czasu rzeczywistego freeRTOS, framework SimpleLink, drivery dla CC3200. Projekt należy skonfigurować zgodnie z opisem publikowanym w poprzednich częściach kursu.

zostały dołączone do projektu (cc3200-sdk\example\common). W katalogu source umieszczone zostały pliki z konfiguracją linii wejścia – wyjścia mikrokontrolera CC3200. Konfiguracja linii wejścia – wyjścia została wygenerowana przy użyciu oprogramowania Pin Mux Tool. Plik projektu Pin Mux Tool (project.pinmux) jest dostępny w materiałach dodatkowych dołączonych do artykułu. W podkatalogach device, hardware, system umieszczone zostały pliki źródłowe oprogramowania. W katalogu device umieszczone zostały pliki do obsługi czujnika temperatury MCP9700. W katalogu hardware pliki do obsługi modułów sprzętowych mikrokontrolera CC3200 (uart, watchdog, adc12). W katalogu system pliki do obsługi logiki pracy urządzenia (konfiguracja urządzenia, obsługa sieci, analiza czasu, obsługa trybu hibernacji itp.).

W oprogramowaniu uruchomiony zostały system czasu rzeczywistego freeRTOS. Utworzony został wątek o nazwie system (patrz plik system.c).

Listing 1. Procedura DataUpdateTask

```
void DataUpdateTask()
{
    int     iSocketDesc;
    sInt16  apConnection;
    float   temperature;
    long ulStatus;
    unsigned long ulDestinationIP;
    DBG_PRINT(„Update Data Begin \n\r”);

    while(1)
    {
        /***************** Connect to specific AP **************/
        apConnection = Connect2AccessPoint();

        if(apConnection < 0)
        {
            DBG_PRINT(„can’t connect to %s AP”,SSID_NAME);
            break;
        }
        else
        {
            DBG_PRINT(„connected to %s AP”,SSID_NAME);
        }
        /****************** GET TEMPERATURE ***************/
        Adc12Enable();
        temperature = Mcp9700Temperature();
        Adc12Disable();
        DBG_PRINT(„temperature %.1f \n\r”,temperature);
        /************** GET SNTP Time **********************/
        // Get the serverhost IP address using the DNS lookup
        ulStatus = Network_IF_GetHostIP(SNTP_SERVER_NAME,&ulDestinationIP);
        if(ulStatus < 0)
        {
            DBG_PRINT(„NTP DNS lookup failed. \n\r”);
            break;
        }
        // Create UDP socket
        iSocketDesc = sl_Socket(AF_INET, SOCK_DGRAM, IPPROTO_UDP);
        if(iSocketDesc < 0)
        {
            UART_PRINT(„Socket create failed\n\r”);
            break;
        }
        GetSNTPTime(ulDestinationIP);
        // Close the socket
        close(iSocketDesc);
        DBG_PRINT(„\n\rSocket SNTP closed\n\r”);
        /***************** SET PLOTLY DATA **********************/
        // Get the serverhost IP address using the DNS lookup
        ulStatus = Network_IF_GetHostIP(PLOTLY_SERVER_NAME,&ulDestinationIP);
        if(ulStatus < 0)
        {
            DBG_PRINT(„PLOT.LY DNS lookup failed. \n\r”);
            break;
        }
        // Create a TCP connection to the server
        iSocketDesc = CreateConnection( ulDestinationIP );
        if(iSocketDesc < 0)
        {
            DBG_PRINT(„Socket creation failed.\n\r”);
            break ;
        }
        // upload temperature value
        SetPlotLyData(iSocketDesc,temperature,
       „iot-temperature”,”temperature  measurements”,”time [s]”,”temperature [C]”);
        // Close the socket
        close(iSocketDesc);
        DBG_PRINT(„\n\rSocket PLOT.LY closed\n\r”);
        break;
    }
    // Stop the driver
    Network_IF_DeInitDriver();
    DBG_PRINT(„Update Data End\n\r”);
}
```

W wątku wywoływane są procedury DataUpdateTask, oraz HibernateEnter. Logika pracy urządzenia została opisana w procedurze DataUpdateTask. W procedurze HibernateEnter włączany jest tryb uśpienia. Kod źródłowy procedury DataUpdateTask pokazano na **listingu 1**.

W procedurze DataUpdateTask jest tworzone połączenie z Access Point (funkcja Connect2AccessPoint). Jest wykonywany pomiar temperatury (funkcja Mcp9700Temperature). Jest pobierany czas z serwera SNTP (funkcja GetSNTPTime). Wysyłane są dane do serwera plot.ly (funkcja SetPlotLyData). W programie można aktywować obsługę układ Watchdog. Czas pracy układu Watchdog zdefiniowano na jedną minutę. Jeśli w zdefiniowanym czasie mikrokontroler nie zdoła wykonać zadań wymienionych w procedurze DataUpdateTask, to układ Watchdog wykona restart mikrokontrolera.

Uruchomienie

Projekt urządzenia pomiarowego dostępny jest w materiałach dodatkowych dołączonych do artykułu (folder iot_measurements). Kopiujemy katalog z projektem do lokalizacji c:/ti/ep/. Następnie uruchamiamy oprogramowanie Code Composer Studio i importujemy projekt (Project → Import CCS Projects). W kolejnym kroku zmieniamy ustawienia oprogramowania. W pliku konfiguracyjnym configure.h ustawiamy nazwę SSID dla Access Point, hasło dostępu do Access Point oraz algorytm szyfrowania transmisji danych. Wprowadzamy nazwę użytkownika w serwisie plot.ly, oraz podajemy hasło dostępu (tzw. API key). Żeby aktywować układ Watchdog w opcjach projektu w zakładce Predefined Symbols z pola Undefine NAME usuwamy wpis WATCHDOG_ON a następnie do pola Pre-define NAME dodajemy wpis WATCHDOG_ON. Po dokonaniu zmian kompilujemy projekt (Project → Build All). W wyniku kompilacji tworzony jest plik binarny o nazwie iot_measurements.bin. Oprogramowanie wgrywamy korzystając z aplikacji CCS UniFlash (opis w poprzednich częściach kursu). Podczas programowania w złączu J13 należy zamontować zworkę.

Podczas pracy sterownik inteligentnej szafy na ubrania wysyła komunikaty serwisowe (procedura DBG_PRINT). Komunikaty wysyłane są za pomocą transmisji UART. Żeby odebrać informacje wysyłane przez sterownik należy podłączyć moduł LaunchPad do portu USB komputera PC, a zworki JP6, JP7 ustawić w pozycji Flash. Wówczas w systemie operacyjnym Windows pod nazwą CC3200LP Dual Port aktywowany zostanie port COM do obsługi modułu LaunchPad. Parametry transmisji UART to: 115200 b/s, 8N1.

Zaprogramowany moduł CC3200 LaunchPad umieszczamy w obudowie. Usuwamy zworkę z złącza J13, montujemy panel słoneczny, instalujemy czujnik temperatury, podłączamy zasilanie.

Podsumowanie

W kolejnym odcinku kursu zostanie opisany projekt sterownika skrzynki na listy. W momencie pojawienia się nowej wiadomości w skrzynce sterownik będzie miał za zadanie wysłać do użytkownika wiadomość e-mail. Sterownik będzie zasilany z akumulatorów ładowanych z paneli słonecznych. Działanie urządzenia zostanie zoptymalizowane pod kątem poboru mocy.

Łukasz Krysiewicz, EP