Internet Rzeczy w przykładach (6)
Serwer Web

W artykule omówimy sposób skonfigurowania serwera Web. Przy tej okazji zaprezentujemy projekt lampki sterowanej za pomocą interfejsu na stronie internetowej. Do budowy urządzenia zastosujemy moduł startowy CC3200 LaunchPad. Na zakończenie zamieścimy krótkie podsumowanie kursu.

W pamięci ROM mikrokontrolera CC3200 zaimplementowano aplikację serwera Web ze wsparciem dla protokołu http w wersji 1.0. Parametry pracy serwera są konfigurowane za pomocą procedur zawartych w bibliotece Simplelink. Na serwerze umieszczono stronę WWW (źródło w pamięci ROM mikrokontrolera CC3200). Wbudowana strona ma mechanizmy służące do konfigurowania, testowania oraz przeprowadzania diagnostyki mikrokontrolera CC3200 (odczyt stanu, konfigurowanie parametrów pracy *Access Point* i profilów Wi-Fi, test *ping* i inne). Dostęp do strony można zablokować (domyślnie jest włączony).

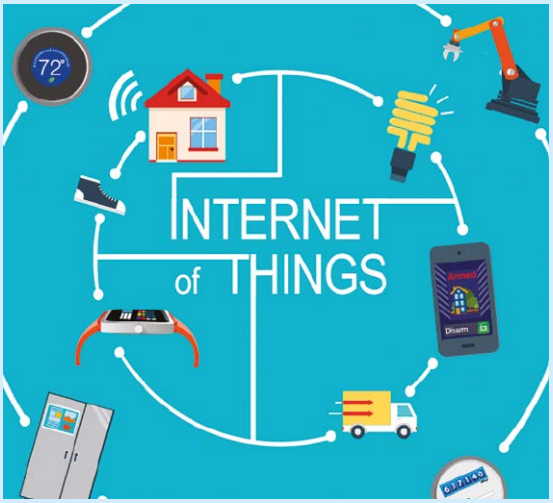
Kod własnej strony WWW można wgrać do zewnętrznej pamięci S-FLASH. Obsługiwane formaty plików to: html, htm, css, xml, png, jpg. Serwer obsługuje zapytania GET oraz POST (komunikacja za pomocą znaczników systemowych oraz użytkownika). Zaimplementowano również obsługę formularzy (przesyłanie danych metodą POST). Serwer jest obsługiwany w trybach *Access Point* oraz *Station* (obsługa DNS, DHCP, trybów chronionych, autoryzacja dostępu do strony WWW). Szczegółowy opis funkcji serwera Web jest dostępny w dokumentacji zamieszczonej w materiałach dodatkowych dołączonych do artykułu.

Budowa

Podstawowym elementem sterownika lampki internetowej jest moduł startowy CC3200 LaunchPad. W pamięci mikrokontrolera CC3200 umieszczono aplikację serwera Web oraz źródło strony WWW – interfejsu służącego do sterowania lampką (włączanie i wyłączanie światła). Moduł startowy CC3200 LaunchPad umieszczono w obudowie Z-34 o wymiarach 129 mm×67 mm×28 mm. Na obudowie zamontowano zewnętrzną antenę Wi-Fi (TRF1002 – 2,4 GHz, 5 dBi). Przekaźnik do sterowania zasilaniem lampki wykonano z elementów dyskretnych. Płytkę drukowalną przekaźnika umieszczono w obudowie lampki.

Zasilanie modułu LaunchPad jest dostarczane przez zasilacz napięcia 3,3 V i obciążalności 0,5 A. Przewody zasilające modułu LaunchPad oraz linia sterująca zasilaniem lampki wyprowadzono na zewnętrz obudowy sterownika. Wygląd sterownika z dołączoną lampką pokazano na **fotografii 1**.

Przekaźnik do sterowania zasilaniem lampki wykonano z wykorzystaniem optotriaka, triaka, tranzystora NPN oraz rezystorów. Załączanie zasilania jest sterowane poziomem logicznym na wyjściu P02 mikrokontrolera CC3200. Poziom wysoki (3,3 V) włącza przepływ prądu, załącza przekaźnik i zasilanie lampki, poziom niski (ok. 0 V) wyłącza przepływ prądu i zasilanie lampki. Schemat ideowy płytki wykonawczej załączającej zasilanie lampki pokazano na **rysunku 2**.

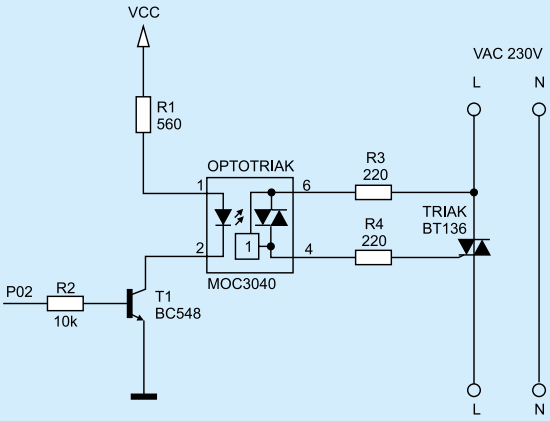


Budowę sterownika rozpoczynamy od zmian w module CC3200 LaunchPad. Aby dołączyć zewnętrzną antenę Wi-Fi usuwamy rezystor R111 oraz wykonujemy zworkę (np. w postaci kleksa z cyny) w miejscu rezystora R110. Antenę dołączamy do złącza J18. Następnie, w układzie przekaźnika łączamy przewód fazowy L oraz przewód zerowy N zasilania sieciowego lampki. Linię sterującą pracą przekaźnika doprowadzamy do wyjścia P02 mikrokontrolera CC3200. Linie zasilania VCC oraz masy GND przekaźnika wyprowadzamy na zewnątrz obudowy lampki. Moduł przekaźnika montujemy w obudowie lampki.

Moduł startowy CC3200 LaunchPad będziemy zasilali korzystając z zasilacza. Żeby skonfigurować moduł LaunchPad do pracy z zewnętrznym zasilaniem, usuwamy zworkę ze złącza J13. Następnie, końcówki zasilacza przyłączamy do pinów VCC i GND złącza J20. Do zasilacza doprowadzamy również VCC i GND wyprowadzone z przekaźnika zamontowanego w lampce. Na zakończenie

Fotografia 1. Sterownik lampki internetowej

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 07641, pass: yus9jv2r



Rysunek 2. Schemat ideowy płytki załączającej zasilanie lampki

Do komunikacji pomiędzy stroną WWW, której kod pokazano na **listingu 1**, a mikrokontrolerem CC3200 wykorzystywane są znaczniki GET i POST. W oprogramowaniu serwera Web zdefiniowano znaczniki systemowe. Opis niektórych z nich umieszczono w **tabeli 1**. Pełen wykaz znaczników systemowych zamieszczono w dokumentacji dołączonej do artykułu. Nazwy znaczników systemowych są tworzone zgodnie ze schematem __SL_G_XXX (dla GET) oraz __SL_P_XXX (dla POST) gdzie XXX to trzy dowolne znaki bądź cyfry. Użytkownik może definiować własne znaczniki GET i POST. Żeby nie powielać nazw znaczników systemowych zaleca się stosowanie nazewnictwa zgodnego ze schematem __SL_G_UXX (dla GET) oraz __SL_P_UXX (dla POST) gdzie XX to dwa dowolne znaki bądź cyfry.

wykonujemy w obudowie otwór do montażu anteny Wi-Fi, montujemy antenę oraz moduł CC3200 LaunchPad.

Uwaga! W module przekaźnika występują napięcia niebezpieczne dla życia i zdrowia człowieka. Nie zaleca się konstruowania urządzenia przez osoby niemające doświadczenia i świadomości zagrożeń. Montaż przekaźnika w lampce należy wykonać przy odłączonym napięciu sieciowym lampki.

Funkcjonalność

Zaprezentowany w artykule sterownik umożliwia sterowanie wyjściem mikrokontrolera CC3200 za pomocą strony WWW. W omawianym przykładzie, do kontrolowanego w ten sposób wyjścia dołączono układ przekaźnika elektromechanicznego sterującego zasilaniem lampki oświetleniowej. W praktyce sterownik może kontrolować załączanie/wyłączanie również innych odbiorników energii elektrycznej.

W pamięci mikrokontrolera umieszczono kod strony WWW. Podczas jej tworzenia wykorzystano pliki źródłowe umieszczone w pamięci ROM mikrokontrolera CC3200. Strona ma funkcje przeznaczone do odczytu stanu urządzenia (zakładka *Status*), jego konfigurowania (zakładka *Device Config*), zmiany parametrów IP w trybie *Access Point* i *Station* (zakładka *IP Config*), konfigurowania profilów Wi-Fi (zakładka *Profiles*), narzędzi do testowania sieci (zakładka *Tools*, test *ping*). Konfigurowanie wyjścia sterującego lampką odbywa się za pomocą zakładki *Control*. Komunikacja pomiędzy stroną a mikrokontrolerem CC3200 jest realizowana za pomocą znaczników POST i GET (systemowych oraz definiowanych przez użytkownika).

Tabela. 1. Przykładowe znaczniki systemowe GET i POST

Nazwa	Znaczenie	Typ
__SL_G_S.C	Odczyt nazwy domeny.	GET
__SL_G_N.D	Odczyt adresu MAC.	GET
__SL_G_W.C	Odczyt zabezpieczeń sieci (brak, WEP, WPA).	GET
__SL_P_S.C	Ustaw nazwę domeny.	POST
__SL_P_N.P	Ustaw adres IP w trybie Access Point.	POST
__SL_P_W.B	Ustaw SSID.	POST

Oprogramowanie

Oprogramowania sterownika lampki internetowej wykonano w środowisku CCSv6 – przykładowy projekt zawarty w materiałach dodatkowych nosi nazwę *iot_control*. Użyto w nim systemu czasu rzeczywistego freeRTOS, framework simplelink, drivery dla CC3200. Dodatkowo, wykorzystano przygotowane przez Texas Instruments procedury obsługi urządzeń peryferyjnych i sieci (uart, udma, network, gpio, smartconfig, przykładowy projekt serwera HTTP z folderu *examples*).

Projekt sterownika lampki internetowej skonfigurowano zgodnie z opisem publikowanym w poprzednich częściach kursu. Pliki źródłowe projektu dostępne są w materiałach dodatkowych dołączonych do artykułu. W katalogu *source* umieszczono pliki z konfiguracją linii wejścia-wyjścia mikrokontrolera CC3200 wygenerowaną przy zastosowaniu oprogramowania *Pin Mux Tool*. W katalogach *hardware* i *system* zostały umieszczone pliki źródłowe oprogramowania. W katalogu *hardware* pliki do obsługi modułów sprzętowych mikrokontrolera CC3200 (uart, gpio). W katalogu *system* pliki do obsługi logiki pracy sterownika (obsługa sieci, konfiguracja serwera i inne). W katalogu *html* zostały umieszczone pliki strony WWW. Aplikacja pracuje pod kontrolą systemu czasu rzeczywistego freeRTOS. Utworzony został wątek o nazwie *system* (plik *system.c*). W wątku jest wywoływana procedura *HTTPServerTask*.

W treści strony sterującej pracą lampki (*light.html*) zostały umieszczone dwa znaczniki GET (*__SL_G_U.I* , *__SL_G_U.V*) oraz jeden znacznik POST (*__SL_P_U.N*). Kod źródłowy strony *light.html* pokazano na listingu 1.

W momencie wywołania strony znaczniki GET są przesyłane do mikrokontrolera CC3200. W procedurze *SimpleLinkHttpServerCallback* znaczniki są przechwytywane i jest ustawiana ich wartość (dynamiczne generowanie treści strony). W zależności od stanu wyjścia sterującego pracą lampki, jest wybierana grafika do wyświetlenia na stronie (znacznik *__SL_G_U.I* – żarówka włączona albo wyłączona) oraz jest zmieniana

Listing 1. Kod źródłowy strony light.html

```
<html>
  <head>
    <title> IoT Light Control</title>
  </head>
  <body style="background-color: white;">
    <center>
      <img  src="__SL_G_U.I"/>
      <br><br><br>
      <form method="POST" name="" action="light.html">
        <tr>
          <input type="hidden"   name="post"     value="__SL_P_U.N">
          <input type="submit"   name="activate" value="__SL_G_U.V">
        </tr>
      </form>
    </center>
  </body>
</html>
```

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 07641, pass: yus9jv2r

nazwa przycisku sterującego pracą lampki (znacznik *__SL_G_U.V* – nazwa przycisku *Turn on* albo *Turn off*). Kod źródłowy procedury *SimpleLinkHttpServerCallback* pokazano na **listingu 2**. Znacznik POST (*__SL_P_U.N*) zdefiniowano w formularzu umieszczonym w treści strony *light.html*. Dane z formularza są przesyłane po naciśnięciu przycisku sterującego pracą lampki. W procedurze *SimpleLinkHttpServerCallback* jest przechwytywany znacznik POST (*__SL_P_U.N*) i stan wyjścia sterujące praca lampki jest zmieniany na przeciwny (włączenie bądź wyłączenie lampki). Po przesłaniu formularza strona *light.html* jest ponownie uruchamiana (pole *action* w parametrach formularza). Wczytanie strony powoduje wywołanie znaczników GET i zaktualizowanie treści strony.

Uruchomienie

Projekt sterownika lampki internetowej dostępny jest w materiałach dodatkowych dołączonych do artykułu (folder *iot_control*). Kopiujemy katalog z projektem do lokalizacji c:/ti/ep/. Następnie uruchamiamy oprogramowanie *Code Composer Studio* i importujemy projekt (*Project → Import CCS Projects*). Aby uruchomić urządzenie do pamięci S-FLASH modułu LaunchPad należy wgrać wsad z oprogramowaniem sterownika oraz dodatkowo pliki tworzące stronę WWW (pliki strony WWW są dostępne w folderze *html* projektu). Montujemy zworkę w złączu J13, ustawiamy zworkę w polu numer 2 złącza SOP, podłączamy moduł LauchPad do portu USB komputera PC. Następnie uruchamiamy oprogramowanie *CCS UniFlash*. Z dostępnych opcji wybieramy *Open Target*

Listing 2. Obsługa znaczników GET i POST

```
void SimpleLinkHttpServerCallback(SlHttpServerEvent_t *pSlHttpServerEvent, SlHttpServerResponse_t
*pSlHttpServerResponse)
{
    unsigned char state;
    unsigned char strLenVal = 0;
    if(!pSlHttpServerEvent || !pSlHttpServerResponse)
    {
        return;
    }
    switch (pSlHttpServerEvent->Event)
    {
        case SL_NETAPP_HTTPGETTOKENVALUE_EVENT:
        {
          unsigned char  *ptr;
          ptr = pSlHttpServerResponse->ResponseData.token_value.data;
          pSlHttpServerResponse->ResponseData.token_value.len = 0;
          if(memcmp(pSlHttpServerEvent->EventData.httpTokenName.data, „__SL_G_U.I”,
                    strlen((const char *)”__SL_G_U.I”)) == 0)
          {
            state = RelayState();
            if(state & 0x01)
            {
              strLenVal = strlen(„images/on.jpg”);
              memcpy(ptr, „images/on.jpg”, strLenVal);
              ptr += strLenVal;
              pSlHttpServerResponse->ResponseData.token_value.len += strLenVal;
            }
            else
            {
              strLenVal = strlen(„images/off.jpg”);
              memcpy(ptr, „images/off.jpg”, strLenVal);
              ptr += strLenVal;
              pSlHttpServerResponse->ResponseData.token_value.len += strLenVal;
            }
            *ptr = ,\0’;
          }
          if(memcmp(pSlHttpServerEvent->EventData.httpTokenName.data, „__SL_G_U.V”,
                    strlen((const char *)”__SL_G_U.V”)) == 0)
          {
             state = RelayState();
             if(state & 0x01)
             {
                 strLenVal = strlen(„Turn off”);
                 memcpy(ptr, „Turn off”, strLenVal);
                 ptr += strLenVal;
                 pSlHttpServerResponse->ResponseData.token_value.len += strLenVal;
              }
              else
              {
                 strLenVal = strlen(„Turn on”);
                 memcpy(ptr, „Turn on”, strLenVal);
                 ptr += strLenVal;
                 pSlHttpServerResponse->ResponseData.token_value.len += strLenVal;
               }
               *ptr = ,\0’;
           }
        }
        break;
        case SL_NETAPP_HTTPPOSTTOKENVALUE_EVENT:
        {
          unsigned char *ptr;

          ptr = pSlHttpServerEvent ->EventData.httpPostData.token_name.data;

          if(memcmp(ptr, „__SL_P_U.N”, strlen((const char *)”__SL_P_U.N”)) == 0)
          {
            state = RelayState();
            if(state & 0x01) RelayOff();
            else RelayOn();
        }
          break;
        }

        default: break;
    }
}
```

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 07641, pass: yus9jv2r

W kursie „Internet Rzeczy w Przykładach" używaliśmy pakietu oprogramowania SDK w wersji 1.0.0. Aktualnie dostępna wersja pakietu SDK to 1.1.0 (pierwsza, oficjalna, produkcyjna wersja SDK). Zmiany wprowadzone w stosunku do poprzedniej wersji są niewielkie. Mimo to firma Texas Instruments zaleca korzystanie z najnowszej wersji pakietu. Pakiet SDK w wersji 1.1.0 jest dostępny pod adresem *www.ti.com/tool/cc3200sdk*. Należy pobrać pakiet *SDK* oraz dodatek *servicepack*. Następnie należy zainstalować pakiet *SDK* oraz wgrać do pamięci S-FLASH modułu LaunchPad dodatek *servicepack*. Pobieranie, instalowanie i wgrywanie oprogramowania zostały omówione w drugiej części kursu (EP02/15).

Configuration i wczytujemy plik konfiguracyjny *httpserver.ucf* dostępny w folderze html projektu. Wprowadzamy numer portu COM, pod którym w systemie operacyjnym został zainstalowany sterownik modułu LaunchPad i wybieramy przycisk *Program*. Do pamięci S-FLASH modułu LaunchPad są wgrywane pliki strony WWW (*main.html*, *light.html*, *on.jpg*, *off.jpg*) oraz oprogramowanie mikrokontrolera (*iot_control.bin*). Po zakończeniu programowania usuwamy zworki z złącza J13 i z złącza SOP oraz odłączamy moduł LaunchPad od portu USB komputera PC. Następnie konfigurujemy moduł LaunchPad do pracy w trybie *Access Point* (domyślny tryb pracy to *Station*). W tym celu montujemy w złączu P1 zworkę łącząc pin P58 z sygnałem VCC (żeby sterownik pracował w trybie *Access Point* na wejściu P58 mikrokontrolera CC3200 musi być poziom wysoki). Następnie dołączamy lampkę do modułu LaunchPad (wejście płytki przekaźnika do wyjścia P02 mikrokontrolera CC3200). Na zakończenie dołączamy zasilacz do GND i VCC złącza J20 modułu LaunchPad oraz do GND, VCC wyprowadzonych z przekaźnika zamontowanego w lampce. Po dokonaniu zmian włączamy zasilacz oraz włączamy zasilanie sieciowe lampki.

Uruchamiamy urządzenie wyposażone w kartę sieciową Wi-Fi 2.4 GHz standard IEEE 802.11 b/g/n (telefon komórkowy, laptop, tablet, komputer PC) i odnajdujemy dostępne sieci Wi-Fi. Punkt dostępu do sieci *Access Point* sterownika lampki internetowej jest rozgłaszany pod nazwą *mysimplelink-2B2246* (konfiguracja SSID w zakładce *Device Config*). Łączymy się z *Access Point* sterownika (brak hasła dostępu, konfiguracja w zakładce *Device Config*). Po połączeniu z *Access Point* uruchamiamy przeglądarkę internetową i wpisujemy adres naszej strony: *mysimplelink.net* (konfiguracja DNS w zakładce *Device Config*). Zostanie wyświetlona strona WWW. Aby rozpocząć sterowanie lampką internetową, przechodzimy do zakładki *Control*. Treść strony jest generowana dynamicznie i odzwierciedla aktualny stan lampki (dynamiczna zmiana grafiki strony oraz opisu przycisku sterującego stanem lampki – obsługa znaczników GET). Korzystając z zamieszczonego na stronie przycisku zmieniamy stan lampki (z wyłączonego na włączony lub z włączonego na wyłączony – obsługa formularza metody POST). Przykład pracy sterownika lampki internetowej pokazano na **fotografii 3**.

Moduł CC3200MOD LaunchPad

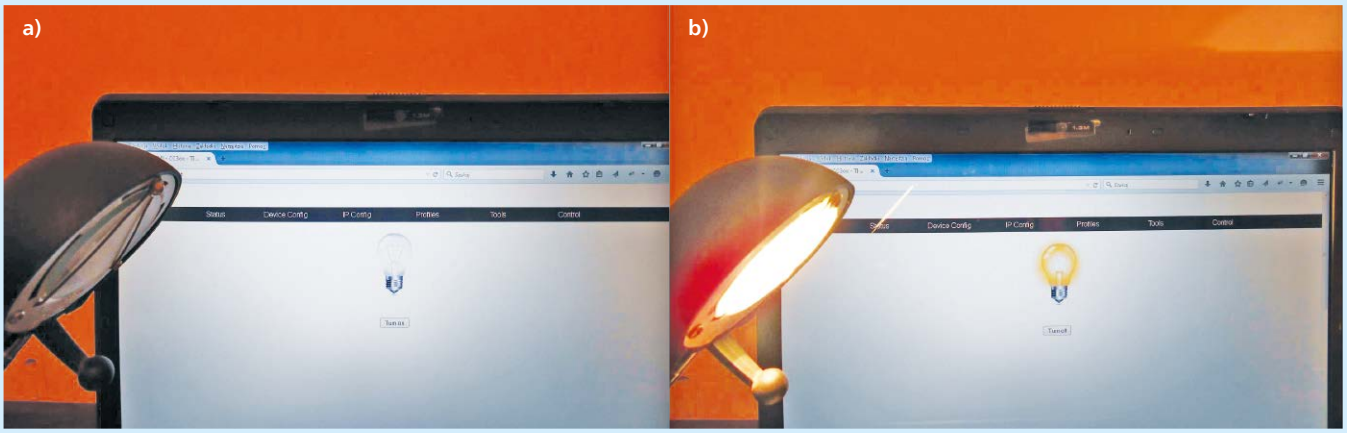
Dostępna jest nowa wersja mikrokontrolera CC3200 z oznaczeniem CC3200MOD. Mikrokontroler CC3200MOD funkcjonalnie jest identyczny jak jego poprzednik CC3200, jednak został przystosowany do pracy w warunkach przemysłowych. Ma certyfikaty zgodności FCC, IC, CE i TELEC. Zakres temperatury pracy wynosi –20…+70°C. Mikrokontroler jest oferowany w obudowie metalowej (LGA 20,5 mm×17,5 mm). Dla promocji mikrokontrolera CC3200MOD zaprojektowano moduł LaunchPad. Nazwa modułu to CC3200MOD-LAUNCHXL.

Podsumowanie

Ten artykuł kończymy cykl „Internet Rzeczy w Przykładach". W jego trakcie omówiliśmy mikrokontroler CC3200 oraz moduł CC3200 LaunchPad (EP01/15). Pokazaliśmy jak zainstalować pakiet oprogramowania narzędziowego (EP02/15) i zaprezentowaliśmy cztery praktyczne projekty: sterownik inteligentnej szafy na ubrania (EP03/15), urządzenie pomiarowe z prezentacją wyników w „chmurze" (EP04/15), sterownik inteligentnej skrzynki na listy (EP05/15) oraz sterownik lampki internetowej (EP06/15). W projektach min.: pokazaliśmy, w jaki sposób dołączyć moduł CC3200 LaunchPad do Internetu, jak komunikować się z serwisami WWW i wymieniać dane, jak wysyłać wiadomości e-mail oraz w jaki sposób pracować z serwerem Web. Omówione zostały tryby oszczędzania energii, komunikacja poprzez UART i SPI oraz pomiary wartości analogowych.

Technologia Internet Rzeczy staje się coraz bardziej popularna, a jej rozwój jest niesamowicie dynamiczny (przez 5 miesięcy trwania kursu do Internetu przyłączono ponad miliard nowych urządzeń!). Producenci prześcigają się w dostarczaniu nowych rozwiązań. Praktycznie każdy nowy produkt opatrzony jest znacznikiem *IoT*. Technologia *Internet of Things* zaczyna wyznaczać trend rozwoju współczesnej elektroniki.

Łukasz Krysiewicz, EP



Fotografia 3. Sterownik lampki internetowej podczas pracy a) lampka wyłączona b) lampka włączona

Poprzednie części kursu i dodatkowe materiały dostępne są na FTP: ftp://ep.com.pl, user: 07641, pass: yus9jv2r