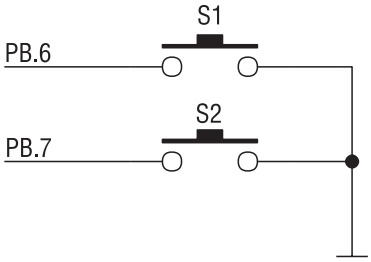
Przykłady zastosowań TCP/IP w mikrokontrolerach, część 2

Jak mówi stare przysłowie: „co dwie głowy to nie jedna". A gdy głów tych będą tysiące, a nawet miliony? Aż strach pomyśleć. Tymczasem taką globalną siłę intelektu mamy dziś w zasięgu ręki, ba – nawet z niej korzystamy. Wszystko za pośrednictwem terabajtów informacji, które w każdej sekundzie przesyłane są olbrzymią siecią informatyczną oplatającą całą kulę ziemską.

Po przedstawieniu wstępnych informacji na temat protokołu TCP/IP i skonfigurowaniu modułu IIM7000A do pracy w naszej sieci możemy przystąpić do konkretnych prób.

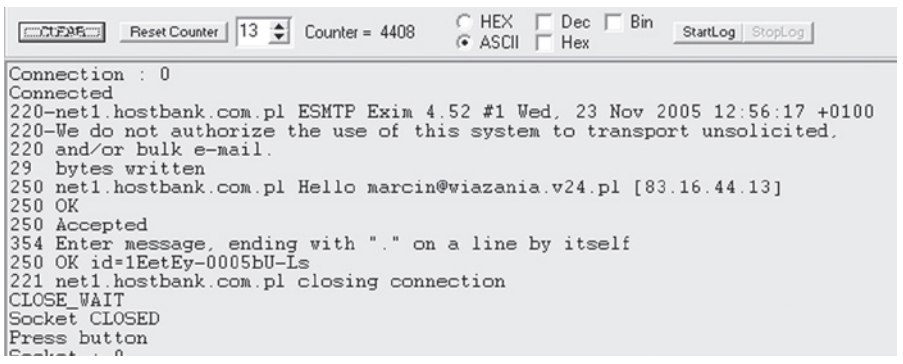
Aplikacja wysyłająca wiadomości e–mail bez autoryzacji po naciśnięciu przycisku (protokół SMTP)

Poczta elektroniczna należy chyba do najczęściej wykorzystywanych aplikacji pracujących w TCP/IP. Adres w poczcie elektronicznej zawiera nazwę konta użytkownika oraz domenę usługodawcy internetowego. Do wysyłania wiadomości e–mail wykorzystywany jest protokół SMTP (*Simple Mail Transfer Protocol*) zapewniający przesyłanie listów pomiędzy dwiema stacjami. Protokół SMTP wykorzystuje numer portu 25. Określa on zarówno format wiadomości jak i metody przesyłania poczty. Przesyłając pocztę do adresata, aplikacja wysyłająca posługuje się poleceniami protokołu. Po zakończeniu transmisji połączenie pomiędzy aplikacjami jest przerywane. W przykładzie przedstawionym niżej zostanie pokazany sposób wysłania wiadomości e–mail po naciśnięciu jednego z przycisków S1 lub S2, które występują w zestawie Easy TCP/IP. Treścią wysyłanej wiadomości będzie informacja, który przycisk został naciśnięty. Wysyłanie wiadomości przebiega bez autoryzacji, czyli użycia loginu i hasła. Przyciski S1, S2 zostały dołączone do zestawu w sposób pokazany na **rys. 2**. Na **list. 2** przedstawiono program wysyłający wiadomość e–mail. Przebieg działania programu można śledzić przez interfejs RS232 z uruchomionym programem terminala. Prędkość transmisji wynosi 19200 bodów. Do terminala wysyłana jest większość danych otrzymanych z modułu TCP/IP. Początek programu jak i konfiguracja modułu TCP/IP będzie identyczna we wszystkich prezentowanych aplikacjach. W tym przypadku linie portów, do których dołączono przyciski zostały skonfigurowane jako wejścia z rezystorami podciągającymi. W pętli głównej programu następuje oczekiwanie na naciśnięcie przycisku S1 lub S2. Po naciśnięciu któregoś z nich, następuje zapisanie numeru naciśniętego przycisku do zmiennej *P_s*, po czym są wykonywane instrukcje odpowiedzialne za wysłanie wiadomości e–mail. W pierwszej kolejności są to instrukcje pętli *do–loop*, która jest opuszczana, gdy *j=0*. Instrukcja *getsocket* otwiera gniazdo do komunikacji TCP/IP. Pierwszy parametr to numer gniazda. W tym przypadku jest to gniazdo numer 0. Kolejny parametr to tryb pracy gniazda. Dla komunikacji TCP/IP wykorzystywany jest tryb *Sock_stream*. W przypadku UDP będzie wykorzystywany tryb *Sock_sdgrm*. Następny parametr to port używany przez otwierane

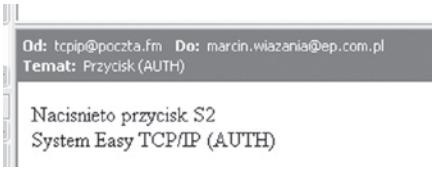


Rys. 2. Sposób dołączenia przycisków S1 i S2

Rys. 3. Widok odebranej wiadomości, która została wysłana przez moduł Easy TCP/IP



Rys. 4. Informacje wysłane przez mikrokontroler do terminala podczas transmitowania wiadomości e–mail

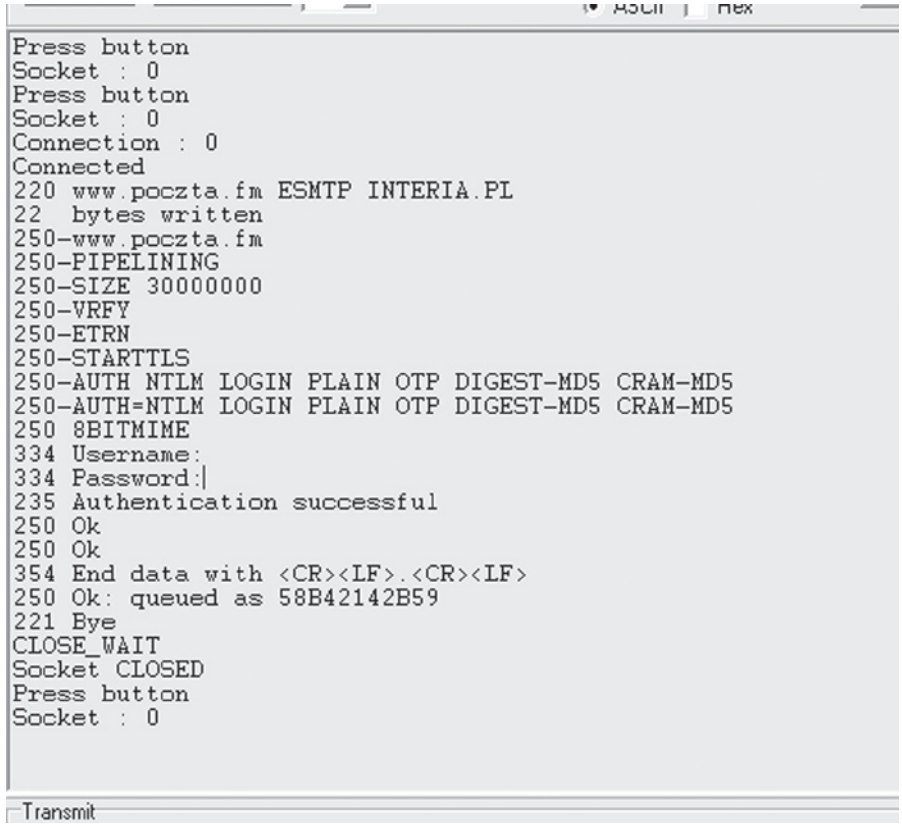


Rys. 5. Widok odebranej wiadomości e–mail wysłanej z autoryzacją przez Easy TCP/IP

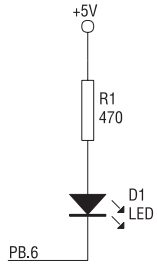
gniazdo. Ostatni parametr określa dodatkowe opcje. Wartość 0 oznacza, że brak jest dodatkowych opcji. Instrukcja ta w przypadku otwarcia gniazda zwraca jego numer. W przeciwnym przypadku zwracana jest wartość 255. Jeśli gniazdo zostanie otwarte, wykonywana jest instrukcja *socketconnect*, która służy do nawiązania połączenia z serwerem TCP/IP. Pierwszy jej parametr to numer otwartego gniazda. Następnym parametrem jest adres IP serwera, z którym nawiązywane jest połączenie oraz numer portu. Dla protokołu SMTP port wynosi 25. Adres IP będzie adresem serwera pocztowego. Adres IP serwera można sprawdzić wykonując polecenie:

```
ping.exe  serwer_p
```

gdzie *serwer_p* to nazwa serwera pocztowego. W przypadku nawiązania połączenia instrukcja zwraca wartość 0, w przeciwnym razie zwróci 1. W przypadku uzyskania połączenia wykonywana jest instrukcja *socketstat* zwracająca informacje o stanie danego gniazda. Pierwszym jej parametrem jest numer otwartego gniazda, a drugim parametr określający, co ma zwrócić instrukcja. Przy wartości 0 drugiego parametru zwracana jest wartość rejestru statusu, czyli stan gniazda. Zwracana wartość przez te instrukcje sprawdzana jest w instrukcji wyboru, która będzie identyczna dla wielu protokołów TCP/IP. Sprawdzane są w niej trzy wartości: *Sock_established* informuje o ustabilizowaniu się połączenia, *Sock_close_wait* i *Sock_closed* informują o zakończeniu połączenia. Przy wartości drugiego parametru równej 1 zwracana będzie informacja, ile bajtów można umieścić w buforze, a przy wartości równej 2 zwracana będzie liczba danych nieodebranych z bufora. Po odczytaniu rejestru z informacją *Sock_established* zgłasza się serwer, po czym jest już możliwe wysłanie wiadomości e–mail. Kolejna instrukcja *socketstat* z drugim parametrem o wartości 2 sprawdza, czy są jakieś dane do odebrania z bufora. Do odbierania danych z bufora służy instrukcja *tcpread*, w której pierwszy parametr to numer otwartego gniazda. Drugim jest zmienna, do której będą trafiać odebrane dane. Możliwe jest także zastosowanie trzeciego parametru informującego o liczbie odbieranych danych, mającego znaczenie tylko przy zmiennych nie będących ciągami znaków. Odczytywane dane z modułu TCP/IP są nie tylko przetwarzane w programie, ale i wysyłane do komputerowego terminala. Typowe połączenie (wysłanie e–maila) składa się z sześciu etapów, które zostaną przedstawione na podstawie kolejnych instrukcji w programie. Po poprawnym połączeniu serwer poczty wysyła komunikat o kodzie 220. Po rozpoznaniu tego komunikatu w programie, do serwera wysyłane jest polecenie *HELO*, które rozpoczyna sesję. Polecenie to zawiera także nazwę konta e–mail, z którego będzie wysyłana wiadomość. Znaki 013 i 010 w kodzie ASCII są znakami potwierdzenia. Wysyłanie danych do TCP/IP umożliwia instrukcja *tcpwrite*, której pierwszy parametr to numer otwartego gniazda. Drugim parametrem jest zmienna, z której będą wysyłane dane. Instrukcja ta zwraca liczbę wysłanych bajtów. Po wysłaniu do serwera polecenia rozpoczęcia sesji, serwer potwierdza jej otrzymanie komunikatem o kodzie 250. Następnie wysyłane jest polecenie *MAIL FROM:* z adresem e–mail nadawcy wiadomości. Serwer także i to polecenie potwierdza komunikatem o kodzie 250. Kolejnym poleceniem wysyłanym do serwera jest *RCPT TO:* w którym należy podać adresata wiadomości. I to polecenie serwer potwierdza komunikatem o kodzie 250. Kolejne polecenie *DATA* umożliwia zapisanie treści wiadomości e–mail tj. nadawcy, odbiorcy, tytułu oraz treści. Po wysłaniu tego polecenia serwer zwraca komunikat o kodzie 354. W tym przypadku w treści wiadomości zapisywana jest informacja o naciśniętym przycisku. Zapis treści wiadomości e–mail kończy się umieszczeniem znaku kropki (.). Po zapisaniu treści wiadomości serwer zwraca komunikat o kodzie 250. Po zapisaniu wiadomości e–mail wysyłane jest do serwera polecenie *QUIT* kończące sesję. Serwer to polecenie potwierdza komunikatem o kodzie 221. Po otrzymaniu z rejestru kontrolnego modułu TCP/IP komunikatu *Sock_close_wait*, należy zamknąć gniazdo instrukcją *closesocket*, której parametrem jest numer otwartego gniazda. Komunikat *Sock_close_wait* jest otrzymywany, gdy serwer zamyka połączenie. Po zamknięciu gniazda program przechodzi do oczekiwania na naciśnię-



Rys. 6. Informacje wysłane przez mikrokontroler do terminala podczas wysyłania wiadomości e–mail z autoryzacją



Rys. 7. Sposób dołączenia diody LED

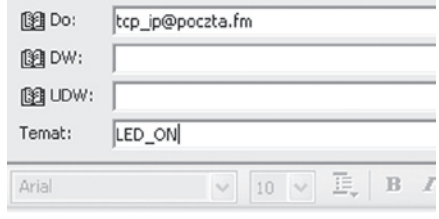
cie przycisku S1 lub S2. Na **rys. 3**. przedstawiono widok odebranej wiadomości e–mail wysłanej przez Easy TCP/IP po naciśnięciu przycisku S2. Na **rys. 4** przedstawiono informacje wysłane przez mikrokontroler do terminala podczas transmitowania wiadomości e–mail.

Aplikacja wysyłająca wiadomości e–mail z autoryzacją po naciśnięciu przycisku (protokół SMTP)

Coraz więcej serwerów poczty podczas wysyłania wiadomości wymaga autoryzacji, czyli podania nazwy użytkownika i hasła do konta e–mail. Przedstawiony na **list. 3** program realizuje to samo, co program z list. 2, z tą różnicą, że wiadomość e–mail jest wysyłana z autoryzacją. Opisane zostaną tylko różnice pomiędzy programami z list. 2 i list. 3. Zamiast polecenia *HELO* wysyłane jest polecenie *EHLO*, będące poleceniem rozszerzonym. Także w tym przypadku serwer potwierdza otrzymane polecenie komunikatem o kodzie 250. Następnie do serwera jest wysyłane polecenie autoryzacji. Występują różne jego wersje, jak: *auth login*, *auth plan...*, w których login oraz hasło są kodowane w różny sposób. W przykładzie wybrano polecenie *auth login*, w którym login (nazwa użytkownika) oraz hasło są zapisywane algorytmem *Base64*. Algorytm *Base64* służy do konwersji ciągu bajtów binarnych do postaci znaków kodu ASCII i jest często wykorzystywany w TCP/IP do przesyłania nazw użytkowników oraz haseł. Base64 pozwala zapisać dane 8–bitowe w formie 6–bitowych liczb. Dane przeznaczone do zakodowania dzielone są najpierw na 3–bajtowe grupy, po czym każda z takich trójek jest dzielona na cztery 6–bitowe liczby. Każda liczba otrzymuje następnie odpowiednik w postaci jednego znaku. Przydział znaków odbywa się na podstawie ściśle określonego, 64–znakowego alfabetu, będącego podzbiorem kodów ASCII. W wyniku zakodowania danych powstaje całkowicie nieczytelny dla człowieka ciąg znaków. W programie po wysłaniu do serwera polecenia *auth login*, otrzymywany jest komunikat o kodzie 334 z zakodowaną w Base64 nazwą *Username:*. Do dekodowania informacji zapisanej w *Base-64* służy funkcja *base64dec*. Wysyłana nazwa użytkownika powinna zostać zakodowana do formatu *Base64*. Umożliwia to funkcja *base64enc*. Po wysłaniu nazwy użytkownika serwer zwraca komunikat o kodzie 334 z zakodowaną nazwą *Password:*. Po zakodowaniu hasła i jego wysłaniu, serwer zwraca komunikat o kodzie 235, który informuje o prawidłowo przeprowadzonej autoryzacji. Dalsze etapy wysyłania wiadomości e–mail są identyczne jak dla przypadku bez autoryzacji. Na **rys. 5**. przedstawiono widok odebranej wiadomości e–mail wysłanej z autoryzacją przez Easy TCP/IP po naciśnięciu przycisku S2. Na **rys. 6** przedstawiono informacje wysłane przez mikrokontroler do terminala, podczas wysyłania wiadomości e–mail z autoryzacją. Przykład pokazuje, w jak prosty sposób można wysłać wiadomość e–mail pod dowolny adres po zaistnieniu określonego zdarzenia, np. awarii jakiegoś urządzenia wykrytej przez odpowiedni czujnik.

Aplikacja odbierająca wiadomości e–mail z informacją o włączeniu, bądź wyłączeniu diody LED (POP3)

Protokół POP3 (*Post Office Protocol 3*) jest najpowszechniej stosowany do odbierania wiadomości e–mail. Serwer POP3 przechowuje pocztę do czasu, gdy użytkownik połączy się i odbierze swoje listy. Listy będące na serwerze można pozostawić lub skasować po odebraniu. Protokół POP3 wykorzystuje port 110. W tym przykładzie zostanie pokazany sposób sterowania diodą LED poprzez wysłanie wiadomości (możliwe włączenie, bądź wyłączenie diody). Informacja o stanie diody LED będzie



Rys. 8. Widok wysyłanej wiadomości z informacją w temacie, że dioda LED ma zostać włączona

przekazywana w temacie wiadomości e–mail. Zestaw Easy TCP/IP będzie pobierał wiadomości z serwera i w zależności od informacji podanej w temacie będzie sterował diodą LED dołączoną do niego w sposób pokazany na **rys. 7**. Na wyświetlaczu LCD będzie prezentowana liczba e–maili znajdujących się na serwerze oraz informacja z tematu odbieranej wiadomości. Przebieg działania programu można śledzić także przez interfejs RS232 z uruchomionym programem terminala. Na **list. 4** przedstawiono program odbierający wiadomości z serwera POP3, na podstawie których steruje diodą LED. W pierwszej kolejności w programie konfigurowany jest moduł TCP/IP, wyświetlacz LCD oraz linia portu PB.6 jako wyjście, do którego została dołączona dioda LED. Program realizuje w nieskończonej pętli co 5 sekund połączenie do serwera POP3 sprawdzając czy nie ma wiadomości do odebrania. Nawiązywanie połączenia z serwerem POP3 przebiega podobnie jak w przypadku wysyłania wiadomości, z tym że z wykorzystaniem portu 110. Po połączeniu zgłasza się serwer POP3 wysyłając komunikat zaczynający się od znaków *+OK*. W dalszej kolejności wysyłane jest polecenie *USER* z nazwą użytkownika konta e–mail. Po uzyskaniu potwierdzenia (*+OK*) wysyłane jest polecenie *PASS* z hasłem do konta e–mail. Zarówno nazwa użytkownika, jak i hasło wysyłane są w sposób niekodowany. Po otrzymaniu komunikatu *+OK* można rozpocząć odbieranie wiadomości e–mail. W tym celu wysyłane jest polecenie *STAT*, które pozwala określić liczbę wiadomości przechowywanych na serwerze. Po wysłaniu tego polecenia serwer wysyła potwierdzenie (+OK), liczbę wiadomości na serwerze oraz ilość zajmowanego przez nie miejsca na dysku. Po wyfiltrowaniu z otrzymanych danych informacji o liczbie wiadomości na serwerze (dane te są wyświetlane na wyświetlaczu LCD) następuje odbieranie wiadomości w pętli *For*. Pętla wykonywana jest tyle razy, ile na serwerze jest wiadomości. W przypadku braku wiadomości pętla *For* nie jest wykonywana. W pętli *For* wysyłane jest polecenie *TOP*, które odbiera wiadomość (bez kasowania z serwera). Polecenie to posiada dwa parametry: numer wiadomości oraz liczbę odbieranych wierszy.

Podanie drugiego parametru o wartości 0 umożliwia pobranie całej wiadomości. W dalszej części pętli jest wykrywany tekst *Subject:*, po którym znajduje się informacja, jaki ma być stan diody LED. Jeśli odebrano w temacie tekst *LED_ON*, dioda LED zostanie zapalona, natomiast w przypadku odebrania w temacie tekstu *LED_OFF* lub dowolnego innego, dioda LED zostanie wyłączona. Zawartość tematu jest także wyświetlana w drugiej linii wyświetlacza LCD. Odczyt wiadomości e–mail następuje, aż do wykrycia znaku kropki (.). Po odczytaniu danej wiadomości wysyłane jest polecenie *DELE* z numerem odczytanej wiadomości. Polecenie to kasuje z serwera wiadomość e–mail o podanym numerze. Bez użycia tego polecenia wiadomości e–mail po odczytaniu nie będą usuwane z serwera. Wiadomości e–mail są odczytywane z opóźnieniem 1 sekundy. Po odczytaniu wszystkich wiadomości wysyłane jest do serwera polecenie *QUIT*, które kończy sesję POP3. Po rozłączeniu i zamknięciu gniazda program przechodzi do opóźnienia, po którym znów się połączy z serwerem POP3. Dostępnych jest wiele innych poleceń protokołu POP3 jak: *LIST*, *RETR* czy *NOOP*, które nie zostały wykorzystane. Dokładne informacje na ich temat znajdują się w dokumentach RFC. Na **rys. 8**. przedstawiono widok wysyłanej wiadomości z informacją w temacie, że dioda LED ma zostać włączona. Na **rys. 9** przedstawiono informacje wysłane przez mikrokontroler do terminala podczas odbierania wiadomości z serwera POP3. Zaprezentowany przykład pokazuje prosty sposób sterowania elementami wykonawczymi za pomocą wysyłanej wiadomości e–mail. Łącząc wysyłanie i odbieranie wiadomości w jedną całość można uzyskać potwierdzenie wiadomością e–mail wykonania polecenia wysłanego również wiadomością e–mail.

```
Ready
socket : 0
Connection : 0
Connected
+OK <97508.1132862801@poczta.fm>
+OK
+OK
+OK 1 1679
LED_ON
+OK

CLOSE_WAIT
CLOSED
socket : 0
```

Rys. 9. Informacje wysłane przez mikrokontroler do terminala podczas odbierania wiadomości z serwera POP3

W ostatniej części kursu będzie mowa o serwerze zapytań, komunikacji za pośrednictwem protokołu UDP oraz serwerze HTTP.

Marcin Wiązania, EP
marcin.wiazania@ep.com.pl

Zawody Sumo Robotów i Międzynarodowy Festiwal Robotów CybAiRBot 2006

22 kwietnia na Politechnice Poznańskiej odbyły się Zawody Sumo Robotów i Międzynarodowy Festiwal Robotów CybAiRBot 2006. Roboty startujące w Zawodach Sumo mogą ważyć maksymalnie 3 kg i mieć rozmiary 20x20 cm. Muszą to być pojazdy w pełni autonomiczne i inteligentne, potrafiące samodzielnie odszukać innego robota i wypchnąć go z ringu. Do Poznania zjechało się ponad 30 robotów skonstruowanych głównie przez studentów uczelni technicznych. Udział w imprezie wzięli m.in. znani w środowisku „robociarzy" Demon, Plugawy Oprawca, Bizon, Skorpion, więc konkurencja była bardzo duża.

W ramach CybAiRBot 2006 odbył się Konkurs na Najciekawszą Konstrukcję Robota Amatorskiego. Wziął w niej udział m.in. robot Torquemax, który przybył w asyście swoich konstruktorów do Poznania aż z dalekiej Kolumbii.

Organizatorzy zaprosili na zawody także inne roboty. W holu nowego centrum wykładowego

Politechniki Poznańskiej można było spotkać całą gamę robotów mobilnych zarówno kołowych, jak i kroczących oraz gąsienicowych, praktycznie ze wszystkich uczelni oraz wielu firm działających w tej branży. Publiczność mogła poznać m.in. Inspektora – policyjnego robota do rozbrajania bomb pracującego na co dzień w poznańskiej Policji. Gościom usługiwał robot–hostessa TOKA–3, u którego zawsze znajdzie się trochę łakoci. Roboty LEGO oraz piesek–robot AIBO dostarczyły dużo frajdy dzieciom.

Na CybAiRBot można było zobaczyć m.in. jak dalecy kuzyni robotów – inteligentne systemy wizyjne radzą sobie z rozpoznawaniem ludzkich twarzy. Nie zabrakło także prawdziwych samurajów, którzy pokazali jak kiedyś w Japonii walczyło się na miecze. W przerwach można było wysłuchać wykładów i prezentacji na temat współczesnej robotyki.

Impreza jest adresowana do szerokiego grona odbiorców w wieku od 2 do 102 lat i ma za zadanie popularyzację nowoczesnej techniki oraz robotyki.

Więcej informacji na www.sumo.put.poznan.pl.