Laboratorium

projektowania skupionych i rozproszonych systemów pomiarowych

Ćwiczenie

Obsługa autonomicznych przyrządów pomiarowych

Instrukcje do ćwiczenia i dodatkowe materiały przygotowano i zmodernizowano przy wykorzystaniu środków otrzymanych w ramach Zadania 36 Programu Rozwojowego Politechniki Warszawskiej

1. Cel ćwiczenia

Celem ćwiczenia jest praktyczne wprowadzenie i zapoznanie studentów z możliwościami jakie oferują obecnie środowiska programistyczne przeznaczone do tworzenia systemów pomiarowych, w zakresie obsługi przyrządów autonomicznych. W ramach ćwiczenia przygotowane zostały zadania, w trakcie realizacji których studenci zdobywają wiedzę dotyczącą sposobów łączenia przyrządów autonomicznych z komputerem, konfiguracji przyrządów w środowisku programistycznym, możliwości podłączenia wielu przyrządów tworzących system pomiarowy. Zasadniczym elementem ćwiczenia jest programowanie przyrządu pomiarowego z poziomu środowiska programistycznego. Dodatkowo studenci zdobywają umiejętności w podstawowym zakresie obejmującym budowę, modyfikacje, a nawet tworzenie własnego sterownika przyrządu pomiarowego.

2. Wprowadzenie: Język SCPI

Historia języka SCPI sięga roku 1975, kiedy to IEEE (*ang. Institute of Electrical and Electronic Engineers*) opracował i opublikował normę IEEE 488-1975. Dokument ten dotyczył zasad łączenia i wymiany informacji w systemach cyfrowych, w których wykorzystywano magistralę HPIB (firmy Hewlett-Packard). W roku 1987 powyższy standard został zaktualizowany o dokumentację w zakresie procedur wymiany i sterowania przepływem danych pomiędzy przyrządami w systemie. W ten sposób powstał dokument IEEE 488.1-1987. Aspekty dotyczące programowania w zakresie języka rozkazów i odpowiedzi przyrządów pomiarowych zdefiniowano w IEEE 488.2-1987. Jednakże dopiero w roku 1990 opracowany został jednolity język dla sterowania urządzeniami oraz przesyłania i przetwarzania danych - język SCPI (*ang. Standard Commands for Programmable Instruments*). Język ten określa format rozkazów oraz definiuje ich składnię, ale również dokonuje ogólnej klasyfikacji przyrządów pomiarowych (woltomierz, generator, oscyloskop licznik-czasomierz czy zasilacz) z przyporządkowaniem zestawu standardowych komend przesyłanych pomiędzy przyrządem a kontrolerem w systemie pomiarowym (określanym też jako automatyczny *ang. Automatic Test Equipment*). Podstawową cechą modelu rozkazów wg SCPI jest jednolity zestaw i składnia komend oraz odpowiedzi przesyłanych pomiędzy kontrolerem i przyrządami zgodnymi ze specyfikacją SCPI w systemie pomiarowym. Dzięki tej własności przyrządy pomiarowe należące do określonej grupy np: woltomierze charakteryzują się podobnym zestawem instrukcji pomiarowych. Wpływa to istotnie na łatwość przyswojenia zestawu komend i czyni prostszym zabieg wymiany przyrządu w systemie pomiarowym. Dodatkowym ułatwieniem w wykorzystaniu języka SCPI jest zestaw komend wysokiego poziomu, które przeznaczone są do uproszczonego, ale i „szybkiego" sterowania przyrządem, bez konieczności definiowania zbioru parametrów komendy (przyjmowanie są wartości domyślne bądź aktualne).

Tłumaczeniem rozkazów SCPI w systemach pomiarowych zajmują się tzw. parsery. Budowanie i analiza komunikatów SCPI może być realizowane bezpośrednio w kodzie programu użytkownika lub też wykonywane za pomocą dedykowanych sterowników programowych. Rozkazy są wysyłane i odbieranie w formacie SCPI, z zachowaniem jednoznaczności odpowiedzi na dany rozkaz.

Kolejną cechą języka komunikatów SCPI jest jego niezależność od fizycznego łącza komunikacyjnego (IEEE-488, GPIB, HPIB, IEC-625, RS-232, RS-485, VXIbus, PXI), za pomocą którego są one przesyłane. Co więcej standard SCPI nie jest zamknięty co oznacza możliwość dołączania kolejnych komunikatów.

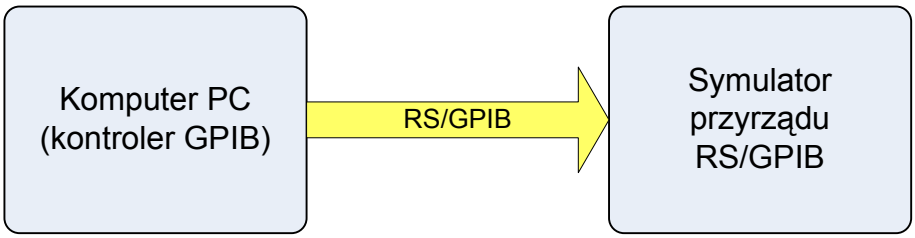
W strukturze standardu SCPI wyróżnić można następujące grupy funkcjonalne komend:

- Input Signal Routing - operacje związane z podłączeniem sygnałów wejściowych,
- Output Signal Routing - operacje związane z wystawieniem sygnałów wyjściowych
- Część pomiarowa - przetwarzanie i analiza sygnałów pomiarowych,
- Część generacyjna - generacja sygnałów wyjściowych zgodnie z zadanymi parametrami
- TRIGger - wyzwalanie wejściowych układów pomiarowych lub wyjściowych generacyjnych (synchronizacja z funkcjami wewnętrznymi, zdarzeniami zewnętrznymi czy innymi przyrządami)
- FORMat - konwersja danych do ustalonej postaci
- MEMory - pamięć danych
- Internal Routing

Rozkazy języka SCPI charakteryzuje hierarchiczność tzn. korzenie drzew rozkazów stanowią rozkazy poziomów głównych (*ang. Root Commands*), które rozwijają się w ramach tzw. podsystemów (*ang. subsystems*) grupujących rozkazy przyporządkowane do określonej funkcji. Rozwinięcie podsystemu na kolejne niższe poziomy związane jest z precyzją realizowanych operacji. Hierarchia rozkazów wymaga, w przypadku wykonania rozkazu z poziomu niższego, podania pełniej ścieżki położenia tego rozkazu, rozpoczynającej się rozkazem głównym danego podsystemu. Słowa kluczowe zapisane w konwencji języka SCPI mają charakterystyczną postać. Pierwsze cztery litery (trzy jeżeli ostatnią jest samogłoska) pisane są wielkimi literami i stanowią minimum mnemoniki komendy. Pozostałe litery dopełniające do pełnej postaci komendy, pisane są małymi literami np: MEASure. Wielkie litery są w formatowanych rozkazach obowiązkowe. Wariant ten pozwala na tworzenie zwartych i krótkich poleceń. Stosowanie pełnych nazw poleceń pozwala natomiast na zwiększenie czytelności pisanych dla systemów pomiarowych, programów. W składni języka SCPI wyróżnia się następujące znaki specjalne:

- w celu rozdzielania następujących po sobie rozkazów w języku SCPI możliwe jest zastosowanie następujących znaków-separatorów:
  - dwukropek - rozdziela następujące po sobie rozkazy, przemieszczając się na niższe poziomy hierarchii poleceń SCPI. Jeżeli występuje przed pierwszym rozkazem powoduje skok na najwyższy (główny) poziom drzewa rozkazów,
  - średnik - oddziela następujące po sobie rozkazy w ramach pojedynczego łańcucha. Nie zmienia aktualnego poziomu rozkazu,
  - przecinek - rozdziela argumenty rozkazów (jeżeli jest ich wiece niż jeden),
  - spacja (tabulacja) - oddziela argumenty od rozkazu. Jest ignorowana w polu argumentów,
- znak zapytania (?) - oznacza rozkaz zapytania (*ang. Query Command*), na który urządzenie zwraca odpowiedź do kontrolera,
- nawiasy kwadratowe ([]) - w nawiasach tych zamieszane są opcjonalne słowa kluczowe bądź argumenty,
- klamry ({}) - ograniczają zbiór argumentów w pojedynczym łańcuchu rozkazowym,
- nawiasy ostre (<>) - oznaczają nazwę argumentu, którą należy zastąpić wartością liczbową,
- symbol logicznej operacji OR (|) - ze zbioru argumentów rozdzielonych ww. znakiem należy wybrać tylko jeden,
- znak gwiazdki (*) - występuje na początku (trzyliterowych) mnemonik rozkazów wspólnych (*ang. Common Commands*). Rozkazy należące do tej grupy zdefiniowano i wykorzystywano przed sformułowaniem języka SCPI i bez modyfikacji włączono je do nowego standardu.

3. Układ pomiarowy



Rysunek 1. Schemat układu pomiarowego

Układ pomiarowy składa się z komputera klasy PC pełniącego funkcję kontrolera i umożliwiającego uruchamianie aplikacji sterujących oraz symulatora przyrządów pomiarowych (National Instruments). Dzięki temu symulatorowi możliwa jest emulacja rzeczywistego przyrządu pomiarowego poprzez interfejs GPIB albo RS. W obydwu trybach możliwa jest symulacja takich urządzeń jak woltomierz czy oscyloskop (lub generator).

Symulator obsługuje następujące zestawy komend (szczegółowy opis każdej komendy zawiera [2]):

- komenda ustawiająca adres

  SADDRess primary[,secondary]
- komendy formatujące ciąg danych z przebiegiem czasowym

  FORMat:DATA {ASCii|INTeger,8|INTeger,16} - ze względu na to, że dane z przyrządu odbierane są jako ciągi znaków ASCII należy dokonać rzutowania na odpowiedni typ U8 albo I16. W LabVIEW służy do tego funkcja *Type Cast*.

  FORMat:DATA?
  FORMat:BORDer {NORMal|SWAPped}
  FORMat:BORDer?
- komendy generujące przebiegi czasowe

  SOURce:FUNCtion {SINusoid|SQUare|NOISe|RANDom|PCHirp}
  SOURce:FUNCtion?
- komendy do odczytu przebiegów czasowych (Query, zapytania)

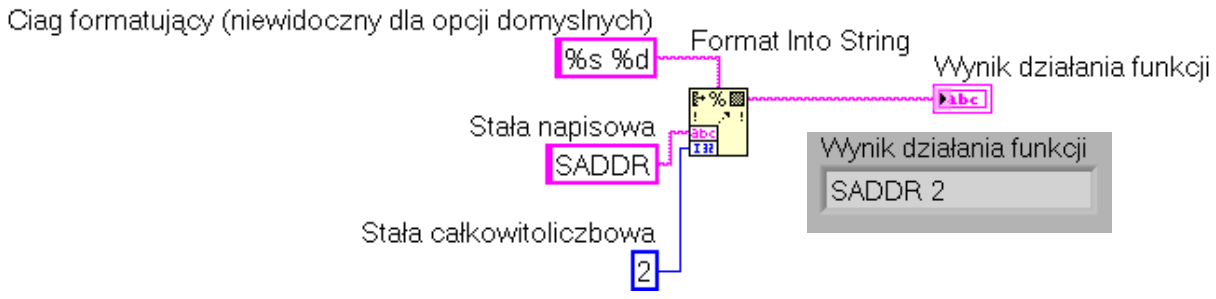
  SENSe:DATA?
  SENSe:VOLTage:RANGe?
  SENSe:VOLTage:RANGe:OFFSet?
  SENSe:VOLTage:HEADer?
  SENSe:SWEep:HEADer?
- komendy do konfiguracji i odpytywania woltomierza

  CONFigure:DC {DEFault|MIN|MAX}
  MEASure:DC?
  CONFigure:DC?
- komendy identyfikacji, statusu, konfiguracji i pomocy (wybrane)

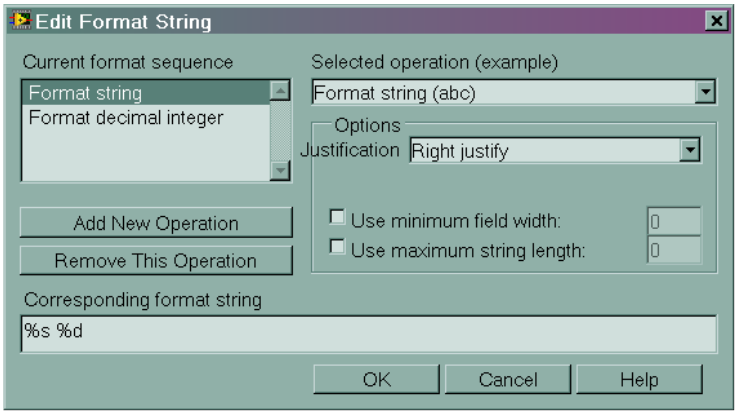
  ***IDN?**
  ***TST**
  ***RST?**
  SYStem:HELP?

4. Realizacja ćwiczenia

Podstawowym celem ćwiczenia jest „napisanie” w środowisku LabVIEW programu/sterownika obsługującego część bądź wszystkie komendy symulatora. Konfigurację symulatora do pracy poprzez łącze szeregowe albo interfejs GPIB należy przeprowadzić zgodnie z rozdziałem 1 pozycji [2]. Formowanie złożonych komend można zrealizować wykorzystując funkcję *Format Into String*, która pozwala tworzenie ciągów znakowych (komend SCPI w przypadku projektowanego programu) ze składowych, które mogą być stałymi lub zmiennymi łańcuchami znakowymi, stałymi lub zmiennymi wartościami liczbowymi (całkowitoliczbowymi i zmiennoprzecinkowymi). Przykładowe zastosowanie funkcji Format Into String przedstawiono na rysunku 2 i 3, na przykładzie formowania komendy do ustawiania adresu. W ogólnym przypadku liczba składowych wejściowych może być większa. Typy tych składowych ustala się niezależnie.

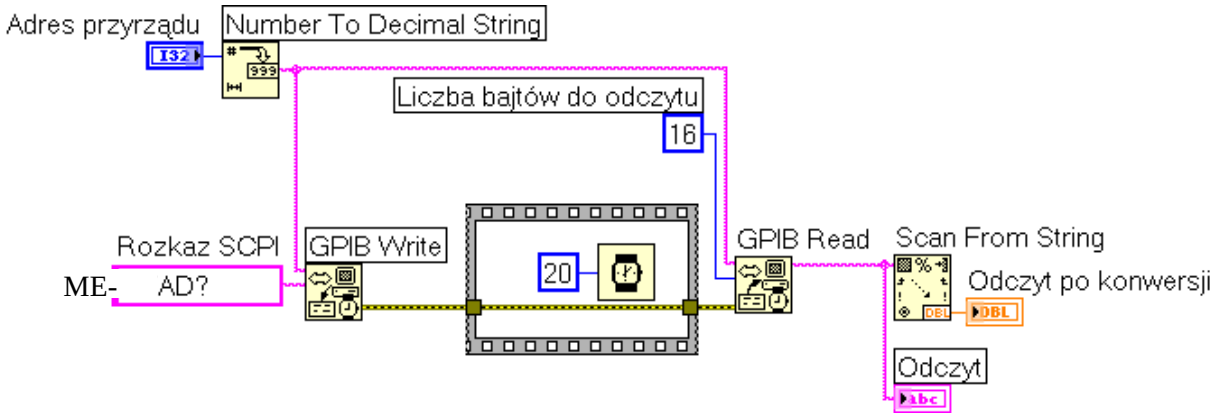


Rysunek 2. Przykładowe wywołanie funkcji *Format Into String*.



Rysunek 5. Okno konfiguracyjne funkcji *Format Into String*.

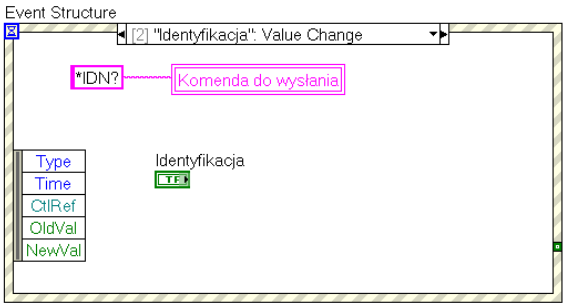
Wysyłanie i odczyt komend z przyrządu można zrealizować poprzez łącze RS lub GPIB. Wysyłane rozkazy pozostają bez zmian, różnica polega jedynie na wykorzystaniu różnych funkcji obsługi łącza (komunikacja RS lub GPIB). Choć i ten aspekt można wyeliminować wykorzystując rozwiązania służące unifikacji funkcji do komunikacji w systemach pomiarowych (standard VISA). Poniżej przedstawiony został przykładowy program do wysyłania rozkazów i odbierania odpowiedzi poprzez interfejs GPIB. Funkcja *Number To Decimal String* zamienia liczbową wartość adresu przyrządu na postać ciągu znakowego, natomiast funkcja *Scan From String* pozwala dokonać konwersji wyniku z ciągu znakowego na wartość liczbową. W przypadku złożonych ciągów wartości (wieloliczbowych i/lub zapisanych w postaci inżynierskiej np; `+1.05E-03`) warto wykorzystać funkcję LabIEW o nazwie *Extract Numbers*.



Rysunek 3. Przykładowy fragment programu do komunikacji z przyrządem pomiarowym

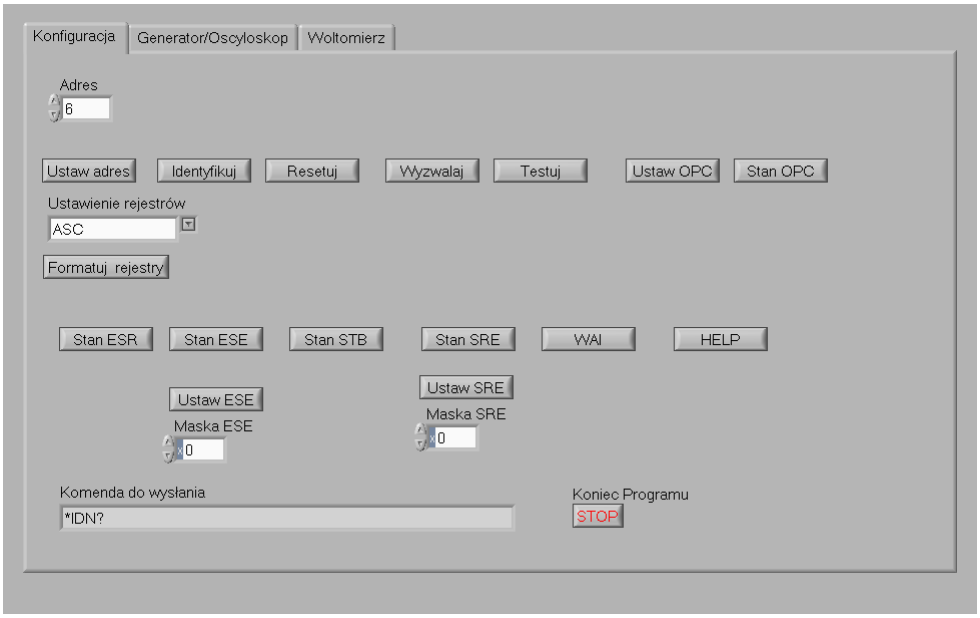
.

Do obsługi formowania i wysyłania rozkazów można wykorzystać konstrukcję *Event Structure*. Na rysunku 4 przedstawiono prosty przykład wykorzystania tej konstrukcji, która po naciśnięciu przycisku Identyfikacja komendę **`*IDN?`**.



Rysunek 4. Przykład użycia konstrukcji *Event Structure*.

Przykładowy panel czołowy aplikacji do sterowania symulatorem przyrządów przedstawiono na rysunku 5. Poszczególne grupy rozkazów rozmieszczono w trzech zakładkach (Tab Control): Konfiguracja, Oscyloskop/Generator, Woltomierz.



Rysunek 5. Przykładowe okno aplikacji (sterownika), po przygotowaniu komendy **`*IDN?`**.

5. Harmonogram ćwiczenia

a) podłączenie przyrządu do komputera i skonfigurowanie go do pracy w systemie pomiarowym,
b) nawiązanie komunikacji z przyrządem poprzez interfejs RS, USB lub GPIB,
c) zaprojektowanie wirtualnego przyrządu pomiarowego z zaimplementowanymi funkcjami do komunikacji z przyrządem autonomicznym,
d) wykorzystanie do komunikacji z przyrządem dedykowanego sterownika i alternatywnie własnych funkcji,
e) weryfikacja i porównanie przyrządu autonomicznego z wirtualnym.

6. Literatura

[1] „Urządzenia pomiarowe i systemy kompatybilne ze standardem SCPI”, Wojciech Mielczarek, Wydawnictwo Helion 1999.
[2] NI Instrument Simulator User Manual
[3] NI LabVIEW User Manual