prof. dr hab. inż. Włodzimierz Gąsowski
dr inż. Stanisław Bocian
mgr inż. Jerzy Frączek
Instytut Pojazdów Szynowych "TABOR"

Zastosowanie programu MATLAB do zadań pomiarowych i sterujących

W artykule przedstawiono sytuację wykorzystania programu Matlab do jednoczesnego wykonywania pomiarów lub generowania sygnałów i prowadzenia związanych z nimi złożonych niekiedy obliczeń. Dotyczy to zarówno prac związanych z projektowaniem i uruchamianiem urządzeń jak i prac związanych z ich badaniem i testowaniem.
Artykuł powstał w wyniku realizacji Projektu Badawczego MN i SzW nr N N509 398236 „Mikrosystemy cyfrowe do inteligentnego, rozproszonego i współbieżnego sterowania pojazdami szynowymi”.

1. Wprowadzenie

Zaledwie 20 lat temu procesy projektowania i badania pojazdów szynowych wyglądały zupełnie inaczej [2, 3]. Nie było wówczas tak doskonałych narzędzi, którymi obecnie dysponują badacze [1, 4]. Zamysłem pracy jest zachęcenie projektantów i badaczy do szerszego korzystania z programu MATLAB.

W praktyce laboratoryjnej i konstruktorskiej często spotykaną sytuacją jest potrzeba jednoczesnego wykonywania pomiarów lub generowania sygnałów i wykonywania związanych z nimi złożonych niekiedy obliczeń. Dotyczy to zarówno projektowania i uruchamiania urządzeń jak i prac związanych z badaniem, testowaniem lub kontrolą urządzeń i układów.

Do prac obliczeniowych często wykorzystywany jest program Matlab firmy MathWorks. Program ten posiada również rzadziej wykorzystywane możliwości prowadzenia prac związanych z pomiarami i sterowaniem przy użyciu zewnętrznych urządzeń pomiarowo – sterujących.
Możliwości te obejmują wszystkie typowe w takiej sytuacji zadania. W zakresie pomiarów Matlab umożliwia:
- konfigurację wybranego urządzenia pomiarowego
- sterowanie przebiegiem pomiarów
- obserwację sygnałów podczas wykonywania pomiarów w postaci różnego typu wykresów lub chwilowych wartości liczbowych
- prezentację wykonanych zapisów po ich zakończeniu
- zapis zebranych wyników w pliku dyskowym.

W zakresie sterowania Matlab dodatkowo umożliwia generowanie sygnałów analogowych i binarnych.

Najnowsze wersje programu Matlab zawierają bardzo zaawansowane wyspecjalizowane możliwości w omawianym zakresie. Ich wykorzystanie wymaga jednak odpowiednio zaawansowanego przygotowania.

W niniejszym tekście przedstawiono podstawowe własności programu Matlab zawarte głównie w pakiecie rozszerzającym: Data Acquisition Toolbox. Korzystanie z funkcji tego pakietu możliwe jest przy niewielkim nakładzie pracy szczególnie, gdy korzysta się z przygotowanych wcześniej szablonów. Materiał przedstawiony w tekście zakłada podstawową wiedzę o korzystaniu z programu Matlab.

Możliwe są różne podejścia do zagadnienia wykonania zadania pomiarowego lub pomiarowo – sterującego. Najprostsze podejście to korzystanie bezpośrednio z komend programu Matlab wpisywanych w linii poleceń lub w skrypcie Matlaba czyli w pliku tekstowym zwanym m–plikiem. Inne podejście polega na przygotowaniu typowego dla systemu operacyjnego Windows okna zawierającego pola wykresu, przyciski, pola tekstowe i podobne elementy. Do przygotowania takiego okna służy zestaw funkcji programu Matlab o nazwie GUIDE.
Pierwsze podejście:
– umożliwia bezpośredni dostęp do wszystkich funkcji związanych z zagadnieniem pomiarów
– umożliwia maksymalnie elastyczne korzystanie z funkcji
– wymaga pewnej znajomości pakietu Data Acquisition Toolbox i zagadnień pomiarowych.
Drugie podejście:
– pozwala przygotować program Matlab do wykorzystania przez osobę nie znającą pakietu Data Acquisition Toolbox
– jest wygodne przy wykonywaniu wielu powtarzających się, podobnych działań.

W pracy omówiono podstawowe elementy umożliwiające szybkie przystosowanie programu Matlab do zadań pomiarowych i sterujących w ramach pierwszego podejścia. W punktach 2 ÷ 4 przedstawiono podstawy korzystania z pakietu Data Acquisition Toolbox między innymi na przykładzie pomiarów z wykorzystaniem wejścia mikrofonowego karty dźwiękowej.

W kolejnych punktach pokazano obsługę analogowych wyjść sterujących, przedstawiono przykładowy program Matlaba realizujący funkcje oscyloskopu oraz omówiono podstawy obsługi urządzeń przez łącze RS-232 przy użyciu Matlaba .

2. Elementy związane z pomiarami

Aby wykorzystać Matlaba do zadań pomiarowych potrzebne są następujące elementy:

– sprzęt pomiarowy, przystosowany do współpracy z komputerem PC oraz od strony

programowej przystosowany do współpracy z programem Matlab

– w ramach Matlaba musi być zainstalowany pakiet rozszerzający

'Data Acquisition Toolbox'. Obecność pakietu można sprawdzić poleceniem 'ver'.

Nazwa pakietu musi znajdować się na wyświetlonej liście pakietów dodatkowych.

– w systemie operacyjnym muszą być zainstalowane sterowniki urządzenia pomiarowo –

sterującego dostarczone przez producenta sprzętu.

W ramach Matlaba musi być zainstalowany element programowy zwany 'Driver Adaptor' (adapter). Element ten pośredniczy między Matlabem a wspomnianym wcześniej sterownikiem producenta sprzętu. Pakiet 'Data Acquisition Toolbox' zawiera pewną liczbę (zależną od wersji) adapterów dla kilku producentów (np. Advantech, Measurement Computing, National Instruments). Ich listę można znaleźć w systemie pomocy pakietu lub korzystając z funkcji daqhwinfo() (p.2.1).

W innych przypadkach (np. firmy IOtech) producent sprzętu dostarcza środki umożliwiające zainstalowanie odpowiedniego adaptera.

Szczególnym przypadkiem jest karta dźwiękowa. Jest ona zwykle obecna na każdym komputerze wraz ze swoim sterownikiem. Adapter karty jest również zawsze elementem pakietu 'Data Acquisition Toolbox'. Wejście mikrofonowe tej karty jest urządzeniem pomiarowym, które można łatwo wykorzystać np. do nauki obsługi pakietu. Wyjście słuchawkowe karty można wykorzystać jako źródło sygnału napięciowego (wyjście analogowe).

Przykłady w dalszej części tekstu są wykonane przy użyciu karty dźwiękowej komputera PC, modułu pomiarowo – sterującego USB–1408FS firmy Measurement Computing oraz modułu pomiarowo – sterującego USB–NI 6221 firmy National Instruments. W funkcjach pakietu 'Data Acquisition Toolbox' karta dźwiękowa jest identyfikowana przez nazwę 'winsound', moduły firmy Measurement Computing przez nazwę 'mcc' a moduły firmy National Instruments przez nazwę 'nidaq'.

2.1. Uzyskiwanie informacji o sprzęcie

Do uzyskiwania informacji o sprzęcie, który wraz ze sterownikiem i adapterem jest dostępny w Matlabie służy funkcja daqhwinfo. Funkcja ta zwraca informacje, które dla danego sprzętu mają stałe wartości – w odróżnieniu od właściwości (properties), omówionych w p.4, których wartości mogą być modyfikowane przez użytkownika.

Przykład kilku sposobów użycia funkcji omówiono poniżej i pokazano na rys. 1 i 2:

– odczyt ogólnych informacji o pakiecie

>> out=daqhwinfo

W takiej formie należy rozpocząć używanie funkcji daqhwinfo. Wyświetlone zostaną podstawowe, ogólne informacje o pakiecie 'Data Acquisition Toolbox' oraz utworzona zostanie zmienna out. Zmienna będzie wykorzystywana przy kolejnych odwołaniach do funkcji daqhwinfo w sposób pokazany poniżej i pokazano na rys. 1 i 2:

– lista zainstalowanych adapterów

>> out.InstalledAdaptors

– informacje dotyczące konkretnego adaptera (tutaj: firmy Measurement Computing)

>> out=daqhwinfo('mcc')

Wyniki działania powyższych wywołań pokazano na rys.1.

Ważną pozycją, potrzebną w dalszej pracy są identyfikatory urządzeń związanych z adapterem. Wartości podane są w wierszu: InstalledBoardIds. W przykładzie na rys.1 jest to wartość 0. Jeżeli w systemie będzie obsługiwanych kilka urządzeń tej samej firmy, to każde z nich będzie miało przydzielony inny identyfikator. Wartość tego identyfikatora może być liczbą lub tekstem. Na przykład dla urządzeń firmy National Instruments identyfikatory mają postać: 'Dev1', 'Dev2', itd.

W celu uzyskania informacji na temat poszczególnych podsystemów urządzenia należy najpierw utworzyć obiekt reprezentujący ten podsystem. Na przykład dla podsystemu wejść analogowych urządzenia firmy Measurement Computing tworzymy obiekt o nazwie np. ai (p. 3.1):

>> ai=analoginput('mcc',0)

Liczba '0' jest omówionym wyżej identyfikatorem urządzenia.

– lista wszystkich właściwości podsystemu uzyskamy wywołując funkcję

daqhwinfo w poniższy sposób:

>> out=daqhwinfo(ai)

– na uzyskanej liście właściwości (rys. 2) znajduje się między innymi pozycja

InputRanges (czyli lista dostępnych zakresów

wejściowych wejść analogowych).

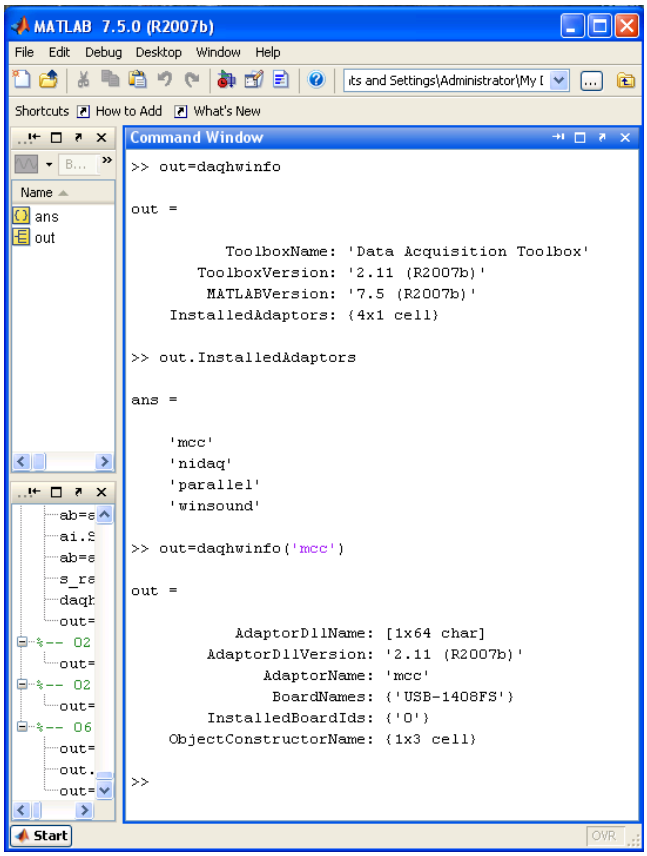
Rozszerzone informacje na temat tej właściwości, czyli wartości liczbowe określające poszczególne zakresy uzyskamy wywołując funkcję daqhwinfo w postaci:

>> out=daqhwinfo(ai,'InputRanges')

Wyniki działania powyższych wywołań pokazano na rys. 2. Dalsze sposoby użycia funkcji daqhwinfo znajdują się w systemie pomocy.

2.2. Informacje nietypowe

Przy korzystaniu ze sprzętu pomiarowo – sterującego trzeba uwzględnić sytuacje nietypowe, nieopisane w dokumentacji. Przykładem jest wykorzystywany w tej pracy moduł USB – 1408FS firmy Measurement Computing. Podczas pracy okazało się, że nie można jednocześnie korzystać z wejść i wyjść analogowych modułu. Ograniczenie dotyczy tylko niektórych typów i serii sprzętu. Do odpowiednich informacji najłatwiej dotrzeć korzystając z wyszukiwarki internetowej.

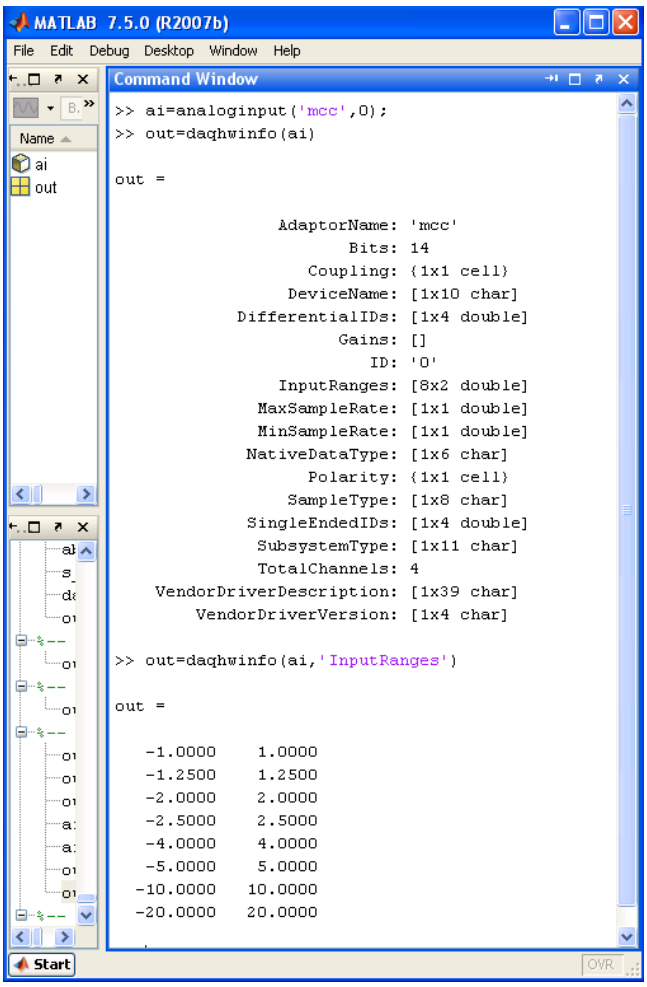


Rys. 1. Przykłady wykorzystania funkcji daqhwinfo dla uzyskania ogólnych informacji o pakiecie 'Data Acquisition Toolbox'.

3. Programowanie z linii poleceń (okno Command Windows)

3.1. Minimalny zestaw poleceń

Minimalny zestaw poleceń programu Matlab pozwalający wykonać pomiar określonej liczby próbek obejmuje:



Rys. 2. Przykłady wykorzystania funkcji daqhwinfo dla uzyskania informacji na temat podsystemu wejść analogowych modułu pomiarowego firmy Measurement Computing

– utworzenie analogowego obiektu wejściowego związanego ze sprzętem pomiarowym (na przykładzie wejścia mikrofonowego):

>> ai = analoginput('winsound',0)

liczba 0 jest identyfikatorem urządzenia. Omówienie sposobu ustalania identyfikatora jest omówione w p. 2.1

– dodanie jednego toru pomiarowego do obiektu urządzenia:

>> addchannel(ai,1)

– Obie powyższe operacje są wykonywane jednorazowo przed rozpoczęciem pomiarów.

– ustalenie częstotliwości próbkowania: 10000 próbek / sekundę:

>> set(ai,'SampleRate',10000)

– ustalenie liczby próbek, które mają być zebrane.

Z podanej wcześniej częstotliwości próbkowania i liczby zbieranych próbek wynika czas wykonywania pomiarów. Tutaj ustalenie liczby 50000 próbek oznacza czas 5s (t=50000/10000):

>> set(ai,'SamplesPerTrigger',50000)

– rozpoczęcie pomiarów:

>> start(ai)

– odebranie wyników wykonanych pomiarów z bufora sterownika do pamięci programu,

czyli tutaj do zmiennej 'zapis', która może być dalej dowolnie przetwarzana w Matlabie.

Ponieważ wcześniej ustalono pomiar w jednym torze pomiarowym addchannel(ai,1), to zmienna 'zapis' będzie wektorem zawierającym 50000 wierszy.

>> zapis = getdata(ai)

– prezentacja zebranych próbek w postaci wykresu i wyświetlenie siatki:

>> plot(zapis)

>> grid on

– zakończenie pracy obiektu 'ai' oraz usunięcie go z pamięci (wykonywane jednorazowo po zakończeniu realizacji zadań pomiarowych):

>> delete(ai)

>> clear ai

Powyższe polecenia (napisane pogrubioną czcionką) należy zapisać w pliku tekstowym z rozszerzeniem .m, np. 'test01.m'. Nieznacznie rozszerzoną treść tego pliku zamieszczono w wydruku nr. 1 na końcu tekstu.

Plik należy uruchomić w typowy dla Matlaba sposób, to znaczy:

– katalog, w którym znajduje się plik musi znajdować się na liście ścieżek Matlaba – polecenie:

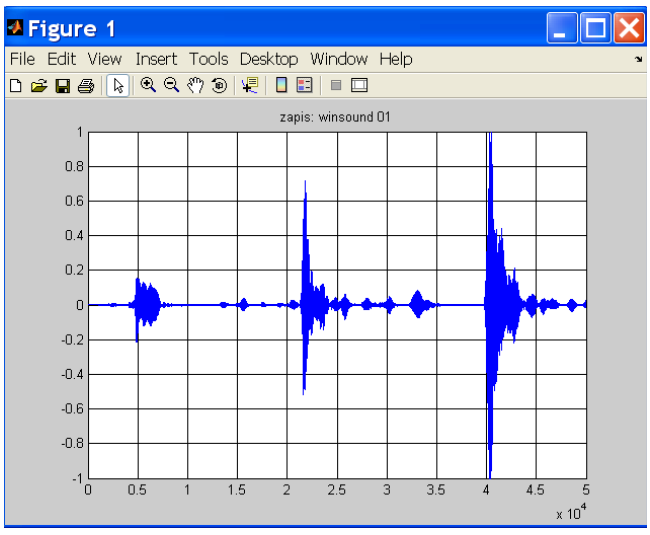
menu File | Set Path

– w oknie Command Window, w linii poleceń należy wpisać nazwę pliku bez rozszerzenia:

>> test01

i nacisnąć klawisz Enter

Gdy minie czas zbierania zadanej liczby próbek (tutaj: 5 s.) na ekranie pojawi się okno pokazane na rys. 3.



Rys. 3. Przykładowy 5 – sekundowy zapis z wejścia mikrofonowego karty dźwiękowej. Wartości sygnału są podane bez przeliczania w [V].

3.2. Narzędzia wspomagające pracę z uzyskanym przebiegiem

Uzyskany w poprzednim punkcie przebieg jest zapisany w zmiennej: 'zapis', która automatycznie została umieszczona w przestrzeni roboczej Matlaba (Workspace). Zmienna ta jest wektorem i można jej używać w dowolnych operacjach obliczeniowych, zapisywać do plików dyskowych lub prezentować w dowolny, inny sposób.

W ramach wyświetlonego okna można przeprowadzać szereg operacji przy pomocy narzędzi zgrupowanych w trzech dodatkowych panelach (otwieranych z menu View):

– Figure Palette

Panel jest podzielony na trzy obszary:

– New Subplots: służy do tworzenia kolejnych wykresów dwu – i trój – wymiarowych

– Variables: zawiera zmienne obszaru roboczego Matlaba, które można umieszczać na wykresach (np. przeciągając je myszką)

– Annotations: zestaw narzędzi do umieszczania na wykresach opisów, strzałek i innych elementów graficznych

– Plot Browser

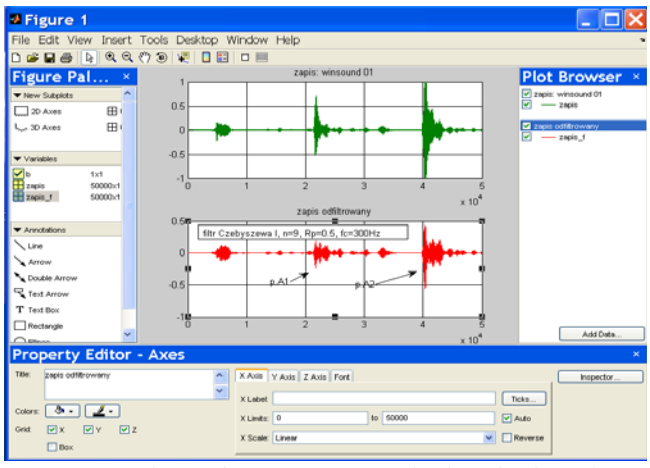
Zawiera listę wszystkich pól wykresów i linii przebiegów. Kliknięcie na dany element powoduje jego zaznaczenie i umożliwia modyfikację jego parametrów.

– Property Editor

Służy do modyfikacji parametrów wybranego elementu. Bezpośrednio w panelu znajdują się wybrane parametry (np. siatki wykresów). Wszystkie parametry są dostępne po wybraniu klawisza 'More properties....'.

Na rys. 4 pokazano prosty przykład wykorzystania tych narzędzi. Najpierw w środowisku Matlaba wykonano filtrację sygnału zapisanego w zmiennej 'zapis'. Funkcje związane z filtrami cyfrowymi znajdują się w pakiecie rozszerzającym Matlaba o nazwie 'Signal Processing Toolbox'.

Zastosowano filtr cyfrowy Czebyszewa typu I. Parametry filtra i zestaw poleceń Matlaba znajdują się w wydruku 2, na końcu artykułu. Wynik filtracji przypisano zmiennej 'zapis_f'. Zmienna jest widoczna w obszarze 'Variables' panelu 'Figure Palette'. Przebieg odfiltrowany został pokazany na dodatkowym polu wykresu.



Rys. 4. Okno wykresów wraz z panelami zawierającymi narzędzia wspomagające.

4. Zaawansowane możliwości pakietu

W p. 3 pokazano najprostszy przypadek wykorzystania programu Matlab do zadań pomiarowych.

Matlab pozwala jednak na wykorzystanie praktycznie wszystkich możliwości zastosowanego urządzenia.

4.1. Konfiguracja sprzętu

Ogólnie sprzęt pomiarowy umożliwia wykorzystywanie go w różnego typu zadaniach pomiarowych. Różne sposoby działania uzyskuje się przez odpowiednie ustawienie parametrów urządzenia. Matlab umożliwia zarówno odczyt możliwych do zastosowania wartości parametrów jak i ustawienie wymaganych wartości.

Jak pokazano w p. 3.1. praca z urządzeniem pomiarowym w ramach Matlaba zaczyna się od utworzenia obiektu programowego. Obiekt ten do momentu jego usunięcia reprezentuje odpowiedni podsystem urządzenia. W przypadku pomiarów jest to podsystem wejść analogowych. Tutaj obiektowi nadano nazwę 'ai'. Obiekt został utworzony poleceniem: ai = analoginput('winsound',0).

Do obiektu trzeba dodać tory pomiarowe, które mają być wykorzystywane. Tutaj dodano jeden tor poleceniem: addchannel(ai,1).

Wszystkie fizyczne parametry urządzenia są dostępne jako właściwości tego obiektu (w Matlabie nazywane Properties). Właściwości dzielą się na:

– właściwości urządzenia wspólne dla wszystkich torów pomiarowych (common properties)

– właściwości poszczególnych torów pomiarowych (channel properties).

Właściwości dzielą się również na:

– bazowe (base properties) – zdefiniowane w Matlabie dla wszystkich typów sprzętu pomiarowego

– specyficzne (device-specific properties) – dodatkowe właściwości określone przez producenta sprzętu.

Listę wszystkich właściwości można znaleźć w systemie pomocy w rozdziale 'Data Acquisition Toolbox', w podrozdziałach odpowiednio:

– Base Properties – Categorical List

– Device-Specific Properties – By Vendor.

Do nadawania wartości właściwościom oraz do ich odczytywania służą polecenia odpowiednio 'set' i 'get'. Jeżeli parametrem tych funkcji będzie tylko nazwa obiektu: set(ai), get(ai), to wynikiem ich działania będzie pełna lista właściwości wspólnych urządzenia. Przy czym funkcja set wyświetli dodatkowo możliwe do zastosowania zakresy wartości, a funkcja get wyświetli wartości aktualne właściwości.

Zmiana wartości pojedynczej właściwości realizowana jest poleceniem 'set' w poniższej postaci:

– dla właściwości wspólnych:

set (nazwa_obiektu, 'nazwa_właściwości', nowa_wartość)

np. nadanie częstotliwości próbkowania wartości 10000 Hz:

>> set(ai, 'SampleRate', 10000)

lub używając pomocniczej zmiennej np. 's_rate'

>>_rate=10000

>> set(ai, 'SampleRate', s_rate)

– dla właściwości toru pomiarowego trzeba dodatkowo wskazać numer tego toru:

set(nazwa_obiektu.Channel(nr_toru), 'nazwa_właściwości', nowa_wartość)

np. ustawienie zakresu wejściowego toru nr 1 na wartość +/–2V

>> set(ai.Channel(1), 'InputRange', [-2 2])

Alternatywnie oba powyższe polecenia można zapisać w postaci:

>> ai.SampleRate=10000

>> ai.Channel(1).InputRange = [-2 2]

W podobny sposób, używając polecenia 'get' można odczytywać aktualne ustawienia poszczególnych właściwości, np:

>> s_rate=get(ai, 'SampleRate')

lub alternatywnie

>> s_rate= ai.SampleRate.

Uwaga:

Niektóre właściwości mogą w nietypowy sposób zależeć od producenta sprzętu. Przykładem jest typ wejścia analogowego w urządzeniach pomiarowych firm Advantech i Measurement Computing. Właściwość ta (wspólna dla torów pomiarowych) w Matalabie nazywa się InputType i oznacza typ wejścia ze wspólną masą lub różnicowe. Dla urządzeń tych producentów Matlabie właściwość ta może być tylko odczytywana. Do jej zmiany należy użyć np. programu fabrycznego producenta.

Należy zaznaczyć, że powyżej wskazano tylko podstawowe sposoby użycia poleceń set i get. Szersze możliwości, na przykład operowanie zakresem torów pomiarowych, są opisane w systemie pomocy.

4.2. Trigger – start pomiarów

Jednym z zagadnień związanych z pomiarami jest ustalenie momentu ich rozpoczęcia. W przykładzie z p. 3.1. tym momentem jest wykonanie polecenia **start(ai)**. Jest jednak szereg innych możliwości. Ogólnie pomiary rozpoczynają się w momencie wystąpienia zdarzenia zwanego trigger. Zdarzenie to jest zdefiniowane za pomocą szeregu właściwości urządzenia, z których najważniejsze są omówione niżej (ustawianie i odczytywanie właściwości – p. 4.1.).

– TriggerType (rodzaj triggera):

Właściwość ta może przyjąć jedną z trzech wartości: Immediate, Manual lub Software.

Producenci sprzętu mogą definiować dodatkowe wartości rozszerzając możliwości swojego sprzętu w tym zakresie.

– Immediate (natychmiast): pomiary są rozpoczynane bezpośrednio po wykonaniu polecenia

start(). Jest to wartość domyślna właściwości Trigger Type i dlatego w przykładzie z p. 3.1.

Przypadek ten wystąpił bez wcześniejszego ustawiania tej wartości. W przypadku tego ustawienia nie można zastosować zjawiska zwanego pretrigger, czyli zapisu pewnej liczby próbek zebranych bezpośrednio przed wystąpieniem triggera (p. 4.2.1.).

– Manual (ręcznie): w tym przypadku wykonanie polecenia start() jest tylko przygotowaniem do rozpoczęcia pomiarów. Pomiary rozpoczną się po wykonaniu dodatkowego polecenia: trigger(). Pomiędzy jednym a drugim poleceniem może być realizowana funkcja pretriggera.

– Software (programowo): Pomiary właściwe rozpoczną się gdy wystąpią zdefiniowane dodatkowo warunki. Warunki te są określone przez kolejne właściwości:

– TriggerChannel: oznacza nr toru pomiarowego

– TriggerCondition: oznacza rodzaj zmiany napięcia na powyższym torze wymagany do wystąpienia zdarzenia trigger.

– TriggerConditionValue: oznacza wartość powyższego napięcia.

Na przykład wykonanie poleceń:

>> set(ai,'TriggerType','Software')
>> set(ai,'TriggerChannel',2)
>> set(ai,'TriggerCondition','Falling')
>> set(ai,'TriggerConditionValue',4.5)

oznacza, że pomiary rozpoczną się, gdy napięcie na torze nr. 2 (TriggerChannel) spadnie poniżej (TriggerCondition=Falling) wartości 4,5V 'przecinając ją' od wartości wyższej do niższej (warunek nie jest spełniony, gdy w momencie wykonania polecenia start() napięcie na torze nr. 2 już jest poniżej 4,5V).

Wspomniane wcześniej zjawisko pretriggera (oraz postriggera – zbierania próbek po zakończeniu pomiarów właściwych) jest konfigurowane za pomocą właściwości: TriggerDelay i TriggerDelayUnits.

W przykładzie z p. 3.1. dla ustalenia liczby próbek, które mają być zebrane ustawiana jest właściwość SamplesPerTrigger: set(ai,'Samples PerTrigger',50000). Oznacza ona, że na jedno wystąpienie triggera zbieranych jest 50000 próbek z każdego toru pomiarowego (w tym przykładzie jest 1 tor). W omawianym przykładzie trigger wystąpi jeden raz – po wykonaniu polecenia start(). Wynika to z domyślnego ustawienia właściwości TriggerType =Immediate. W innych przypadkach po poleceniu start() trigger może wystąpić wielokrotnie. Przy każdym wystąpieniu zostanie zebranych 50000 próbek z każdego toru.

4.3. Charakterystyki przeliczeniowe

Typowo zmierzone próbki są odbierane ze sterownika urządzenia w jednostkach napięcia [V]. W ramach pakietu Data Acquisition Toolbox' można ustalić natychmiastowe przeliczanie na jednostki tej wielkości, która faktycznie jest mierzona (np. ciśnienie, siła, temperatura,...) według charakterystyki liniowej. Do tego celu służą poniższe właściwości toru pomiarowego:

– SensorRange: zakres napięcia wyjściowego zastosowanego przetwornika pomiarowego [V]

– UnitsRange: zakres pomiarowy przetwornika w jednostkach docelowych odpowiadający jego zakresowi napięcia wyjściowego

– Units: tekst opisujący jednostki docelowe, np. 'MPa', 'km/h', ...

Jeżeli zmierzoną wartość próbki [V] oznaczymy przez Up, to wartość przeliczona W będzie:
W=Up * (UnitsRange / SensorRange).

Przykład 1.

Do toru pomiarowego nr. 1 przyłączono przetwornik ciśnienia o zakresie pomiarowym 0..1 MPa.
Sygnał wyjściowy tego przetwornika odpowiadający powyższemu zakresowi, to 0÷10 V.
Wartości zakresów podaje się jako dwupozycyjne wektory:

>> set(ai.Channel(1),'SensorRange',[0 10]);
>> set(ai.Channel(1),'UnitsRange' ,[0 1]);
>> set(ai.Channel(1),'Units' ,'MPa');

Zmierzone napięcie np. 2V zostanie przeliczone na wartość ciśnienia p:
p= 2 * (1 / 10) = 0,2 Mpa.

4.4. Zapis pomiarów do pliku

Wyniki pomiarów można zapisać do pliku dyskowego wraz z informacjami dotyczącymi sesji pomiarowej w formacie wewnętrznym Matlaba lub w postaci tekstowej.
Zapis w formacie wewnętrznym jest wygodny, jeżeli zamierzamy dalej opracowywać te wyniki przy pomocy Matlaba. Postać tekstowa umożliwia np. archiwizację wyników lub wykorzystanie ich w innych programach np. arkuszach kalkulacyjnych.

4.4.1. Zapis w formacie wewnętrznym

Podczas trwania pomiarów uzyskiwane wyniki są zapisywane albo w pliku dyskowym albo w pamięci wewnętrznej komputera albo jednocześnie w obu tych miejscach. Wybór zależy od ustawienia właściwości LoggingMode. Odpowiednio może ona mieć wartość: Disc, Memory lub Disk&Memory.

Domyślnym ustawieniem jest: Memory. Taka sytuacja jest w przykładzie z p. 3.1. Bezpośrednio po wykonaniu pomiaru wyniki są gromadzone w specjalnym obszarze pamięci, skąd do obszaru roboczego Matlaba są wczytywane za pomocą polecenia getdata(). Polecenie getdata() jest poleceniem blokującym co oznacza, że po wywołaniu oczekuje na zebranie zleconej liczby próbek. W tym przypadku jest to 50000 próbek z jednego toru pomiarowego. Zatem po wykonaniu getdata() zmienna 'zapis' jest wektorem zawierającym 50000 zmierzonych próbek i może podlegać wszystkim

dostępnym w Matlabie operacjom na zmiennych.

Jeżeli właściwość LoggingMode='Disk', to wyniki są automatycznie zapisywane na dysku w pliku, którego nazwa jest ustalona przez kolejną właściwość o nazwie LogFileName. Plik jest zapisywany do aktualnego katalogu roboczego Matlaba.

Przykład 2.

>> LoggingMode='Disk'

>> LogFileName='log_abc.dat'

Ponowne wczytanie wyników zapisanych w pliku 'log_abc.dat' do Matlaba jest realizowane poleceniem daqread(...). Polecenie to może być zastosowane w kilku postaciach opisanych szczegółowo w systemie pomocy. Najprostsze z nich ma postać:

zapis2 = daqread('nazwa_pliku'). W tym przykładzie: zapis2 = daqread('log_abc.dat')

i powoduje wczytanie do zmiennej 'zapis2' próbek zapisanych wcześniej w pliku o nazwie: 'log_abc.dat'.

4.4.2. Zapis w postaci tekstowej

4.4.2.1. Niskopoziomowe funkcje dostępu do plików

Najbardziej elastycznym sposobem zapisu zebranych wyników do pliku tekstowego jest użycie niskopoziomowych funkcji dostępu do plików. Funkcje te są oparte na funkcjach znanych z języka C. Sposób ten pozwala na tworzenie plików w dowolnej postaci. Kluczową funkcją, przy pomocy której określana jest postać poszczególnych wierszy pliku jest funkcja fprintf().

Cała procedura zapisu obejmuje:

– otwarcie pliku

– zapis wyników

– zamknięcie pliku.

Otwarcie pliku – funkcja fopen()

>> fid=fopen('ścieżka_do_pliku','tryb_otwarcia_pliku');

– 'ścieżka_do_pliku'

– należy wpisać bezpośrednio lub za pośrednictwem utworzonej wcześniej zmiennej typowo zapisywaną ścieżkę, np.:

S='C:\Katalog1\Katalog1\WynikiA.txt';

– jeżeli podamy tylko nazwę pliku, to zostanie on utworzony w aktualnym katalogu roboczym Matlaba.

– 'tryb_otwarcia_pliku'

– w przypadku zapisu do plików tekstowych parametr ten może mieć jedną z dwóch wartości:

– 'w' : spowoduje utworzenie i otwarcie nowego, pustego pliku o podanej nazwie (oraz ewentualnie usunięcie już istniejącego pliku o takiej nazwie i lokalizacji)

– 'a' : spowoduje otwarcie już istniejącego pliku a kolejne wpisy będą dodawane do już istniejących (jeżeli plik nie istnieje, to zostanie utworzony)

– fid

Jeżeli operacja otwarcia pliku nie powiedzie się, to zmienna fid będzie miała wartość –1.

W przeciwnym przypadku fid będzie większe od 0 i zmienna ta w kolejnych operacjach będzie identyfikować otwarty do zapisu plik.

Przykład 3:

>> S='C:\Katalog1\Katalog2\WynikiA.txt';

>> fid=fopen(S,'w');

Zapis wyników – funkcja fprintf()

Przykładowe użycie funkcji może wyglądać jak niżej:

– jeżeli mamy zdefiniowane

– dwie zmienne zawierające dane:

A=24

B=15.34

– zmienną oznaczającą szerokość kolumny teksu:

sz=7

– zmienną oznaczającą liczbę miejsc po przecinku

mpp=3

to po wykonaniu funkcji:

>> fprintf(fid, 'Wyniki pomiarów nr 27\r\n');

>> fprintf(fid, 'A=%*d\tB=%*.*f\r\n', sz,A,sz, mpp,B);

otrzymamy zapis w pliku w postaci:

Wyniki pomiarów nr 27

A= 24 B= 15.340

Pełny opis funkcji fprintf() znajduje się w systemie pomocy.

W powyższym przykładzie argumenty funkcji podawane w nawiasach (...) oznaczają:

– fid: identyfikator pliku otrzymany po wykonaniu funkcji fopen()

– ciąg ograniczony apostrofami '...' jest ciągiem formatującym tekst i może zawierać:

– zwykły tekst, np.: Wyniki pomiarów nr 27

– znaki sterujące dla edytora tekstu, np.:

– \r: powrót kursora do pierwszej kolumny

– \n: nowa linia

– razem powyższa para znaków odpowiada naciśnięciu klawisza Enter w edytorze tekstu

– \t: wstawienie znaku tabulacji

– elementy formatujące sposób zapisu liczb, zaczynają się od znaku % i kończą literą np. tutaj d i f:

– %*d – w miejscu tego elementu zapisana będzie liczba całkowita, znak * oznacza, że dalej podana będzie szerokość kolumny tekstu

– %*.*f – w miejscu tego elementu zapisana będzie liczba rzeczywista, znak *.* oznaczają, że dalej podana będzie szerokość kolumny tekstu i liczba miejsc po przecinku

– lista zmiennych rozdzielonych przecinkami znajduje się za ciągiem formatującym po przecinku. Liczba i kolejność zmiennych na liście odpowiada elementom formatującym liczby w ciągu formatującym.

W wydruku nr. 3 zamieszczono treść m – pliku zapisującego do pliku tekstowego wyniki pomiarów z przykładu z p. 3.1. Zapisywane są dwie kolumny liczb: kolumna czasu i kolumna wyników pomiarów. Pozycje kolumny czasu są wyliczane na podstawie zadanej częstotliwości próbkowania.

Zamknięcie pliku – funkcja fclose().

Po zakończeniu zapisów do pliku konieczne jest jego zamknięcie przy pomocy poniższej funkcji

>> status=fclose(fid).

Przy poprawnym wykonaniu funkcji zmienna status ma wartość 0, w przeciwnym wypadku –1.

4.4.2.2. Inne metody zapisu do plików tekstowych

W Matlabie jest dostępnych szereg funkcji specjalizowanych do zapisu do plików tekstowych. Funkcje te są opisane w systemie pomocy w rozdziale: MATLAB\Programming \ Data Import and Export \ Exporting Text Data. Funkcje te są przystosowane do realizacji różnego typu zadań, ale nie dają takiej elastyczności zastosowania jak opisana w poprzednim rozdziale funkcje fprintf().

W ramach Matlaba można też korzystać z bibliotek łączonych dynamicznie (.dll) tworzonych w innych systemach. Matlab zawiera również mechanizmy umożliwiające korzystanie z funkcji napisanych bezpośrednio w językach programowania C, Fortran i Java.

5. Obsługa sygnałów wyjściowych

5.1. Minimalny zestaw poleceń

Minimalny zestaw poleceń programu Matlab pozwalający ustawić na wybranym wyjściu analogowym określoną wartość napięcia jest podobny do opisanego w p. 3.1. zestawu poleceń pomiarowych. Poniższy przykład dotyczy modułu pomiarowo – sterującego USB – 1408FS firmy Measurement Computing:

– utworzenie analogowego obiektu wyjściowego związanego ze sprzętem pomiarowym

>> ao= analoginput('mcc',0)

– dodanie jednego toru pomiarowego do obiektu urządzenia:

>> addchannel(ao,1)

– obie powyższe operacje są wykonywane jednorazowo

– ustalenie częstotliwości uaktualniania wartości napięcia na wyjściu: 100 próbek / sekundę:

>> set(ao,'SampleRate',100)

– ustalenie wartości napięcia U[V]=2.5V

>> **U_V=2.5**

– umieszczenie danych w pamięci związanej z wyjściem

>> putdata(ao,U_V)

– uruchomienie obiektu ao:

>> start(ao)

– woltomierz przyłączony do wyjścia analogowego obsługiwanego urządzenia powinien wskazać napięcie 2.5V.

– zakończenie pracy obiektu 'ao' oraz usunięcie go z pamięci:

>> delete(ao)

>> clear ao

5.2. Generowanie złożonego przebiegu

Na wyjściach analogowych można generować złożone, zmienne w czasie przebiegi napięciowe.

W wydruku 4 zamieszczono przykład praktycznego wykorzystania generowanego przebiegu do testowania działania funkcji cyfrowej obróbki sygnału. Tutaj jest to prosta funkcja (o nazwie FFT_01) obliczająca szybką transformatę Fouriera FFT (Fast Fourier Tranform) sygnału zmierzonego na wejściu analogowym. W celu przetestowania funkcji na wejście należy podać sygnał o znanym przebiegu. Sygnał taki zostanie wygenerowany na wyjściu analogowym (które jest fizycznie połączone z wejściem). Przebieg wyjściowy należy przygotować w postaci matrycy. Matryca może zawierać przebiegi dla wielu wyjść. Pojedynczy przebieg jest umieszczony w osobnej kolumnie. W naszym przypadku będzie to matryca jednokolumnowa. Przygotowany zostanie przebieg będący sumą czterech przebiegów sinusoidalnych. Sprawdzana funkcja FFT_01 powinna poprawnie odtworzyć skład przebiegu wynikowego.

Elementy związane z generowanym przebiegiem, to:

– czas trwania przebiegu – 5s:

>> **czas_ao=5**

– częstotliwość uaktualniania wyjścia – 100000 Hz

>> **f_ao=100000**

– wybrana częstotliwość musi zawierać się w dopuszczalnym zakresie. Dodatkowo niektóre urządzenia pozwalają na ustawienie tylko wybranych wartości częstotliwości. Jeżeli nie mamy pewności, czy wybrana częstotliwość znajduje się na liście, to należy odczytać częstotliwość faktycznie ustawioną:

– ustawienie częstotliwości:

>> set(ao,'SampleRate',f_ao)

– odczyt częstotliwości faktycznie ustawionej

>> f_ao_ust = get(ao,'SampleRate')

– wykonanie wektora zawierającego punkty czasowe przebiegu:

>> **czas=0:1/f_ao_ust:czas_ao;**

– ustalenie częstotliwości przebiegów składowych:

>> **f_harm1=1700;** %częstotliwość harmonicznej 1 [Hz]

>> **f_harm2=3250;** %częstotliwość harmonicznej 2 [Hz]

>> **f_harm3=4440;** %częstotliwość harmonicznej 3 [Hz]

>> **f_harm4=7400;** %częstotliwość harmonicznej 4 [Hz]

– wykonanie wektorów poszczególnych składowych:

>> harm1=sin(2*pi*f_harm1*czas);

>> harm2=sin(2*pi*f_harm2*czas);
>> harm3=sin(2*pi*f_harm3*czas);
>> harm4=sin(2*pi*f_harm4*czas);

– wykonanie wektora przebiegu wynikowego:

>> przebieg = harm1 + harm2 + harm3 + harm4;

– zamiana na postać kolumnową

>> przebieg=przebieg';

– tak przygotowany przebieg należy umieścić w pamięci związanej z wyjściem ao

>> putdata(ao,przebieg)

– po uruchomieniu obiektu ao:

>> start(ao)

przebieg zostanie wyprowadzony na wyjście analogowe

W wydruku 4 zamieszczono treść dwóch m – plików:

– 4. 1. skrypt główny

– 4. 2. sprawdzana funkcja FFT_01

Do realizacji przykładu zastosowano moduł pomiarowo – sterujący NI USB–6221 firmy National Instruments:

– ustalony powyżej przebieg został wygenerowany na wyjściu analogowym modułu

– pomiar przebiegu został doprowadzony do jego wejścia analogowego.

Na rys. 5 pokazano początkowy fragment przebiegu oraz wynik działania sprawdzanej funkcji FFT_01, czyli widmo amplitudowe przygotowanego sygnału.

W przypadku cyfrowej obróbki sygnałów ważne jest odpowiednie dobranie częstotliwości próbkowania na wejściu analogowym. Dla poprawnego wykonania analizy szybkiej transformaty Fouriera (FFT) częstotliwość ta musi być większa niż podwojona wartość częstotliwości najwyższej składowej przebiegu. Dla urozmaicenia przykładu wybrano częstotliwość 10000 Hz, która uniemożliwia poprawne zidentyfikowanie składowej 4 o częstotliwości 7400 Hz. Istnienie tej składowej w badanym przebiegu powinno spowodować wystąpienie zjawiska aliasingu. W tym przypadku będzie to pojawienie się w widmie sygnału składowej o częstotliwości:

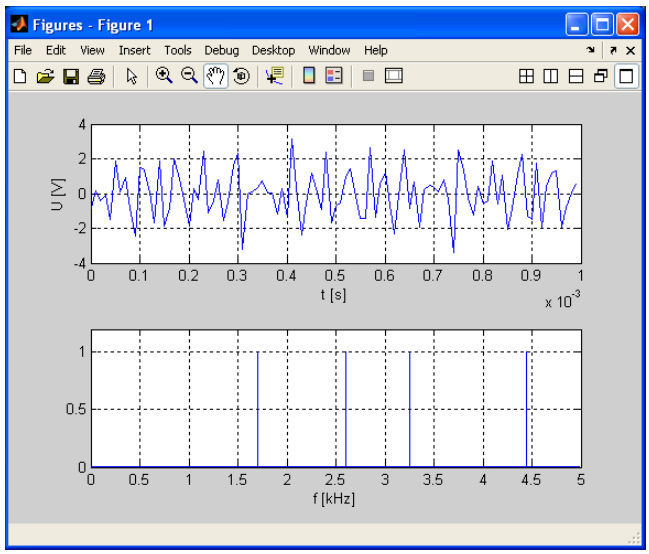
1*10000 – 7400 = 2600 Hz.

6. Program – Oscyloskop

Jak wspomniano w p. 1. W Matlabie można tworzyć charakterystyczne dla systemu Windows okna zawierające elementy służące do komunikacji z użytkownikiem. Do przygotowania takiego okna służy zestaw funkcji o nazwie GUIDE. Funkcje te należą do podstawowej części programu Matlab.

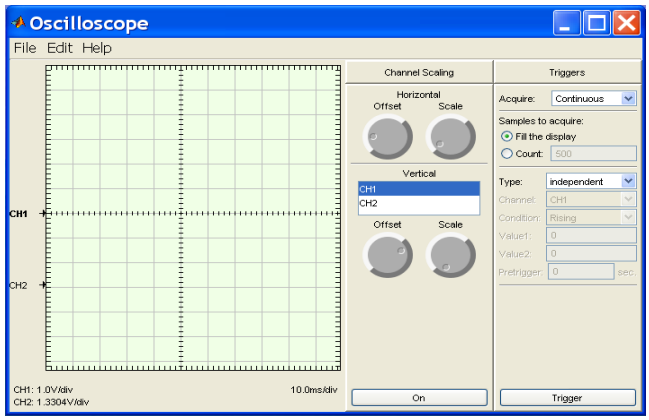
W połączeniu z funkcjonalnością pakietu 'Data Acquisition Toolbox' można tworzyć złożone programy pomiarowo – sterujące.

W pakiecie zawarty jest przykładowy program tego typu o nazwie 'Softscope'. Program ten można



Rys. 5. Wynik działania przykładu z p. 5.2. Górny wykres zawiera początkowy fragment przebiegu. Dolny wykres zawiera jego widmo amplitudowe.

wykorzystać do obsługi dowolnego sprzętu pomiarowego obsługiwanego przez Matlaba. Sposób pracy programu jest charakterystyczny dla obsługi oscyloskopu. Obsługa programu jest szczegółowo opisana w systemie pomocy Matlaba i samego programu 'Softscope'. Na rys. 5. pokazano główne okno programu.



Rys. 5. Okno główne programu 'Softscope'.

Uruchomienie programu dla wybranego sprzętu (tu: urządzenie firmy National Instruments o identyfikatorze 'Dev1') i zakresu torów pomiarowych (tu: tory 0 .. 1) wymaga poniższej sekwencji poleceń:

>> ai=analoginput('nidaq','Dev1')
>> adchannels(ai,0:1)
>> softscope(ai)

7. Obsługa urządzeń przez złącze RS – 232

Szereg urządzeń pomiarowo – sterujących jest obsługiwanych przez złącze RS-232.

Istnieje również szereg konwerterów RS – 232 do innych typów sieci (np. GPIB, CAN). Korzystając z takich konwerterów można zatem wymieniać informacje z urządzeniami różnego typu. W programie Matlab można obsługiwać urządzenia

przez złącze RS – 232. Obsługa RS – 232 nie jest elementem omawianego wcześniej pakietu rozszerzającego Matlaba: 'Data Acquisition Toolbox'. Funkcjonalność ta zawarta jest w podstawowej części programu. Pełny opis jest zawarty w systemie pomocy w punkcie: Matlab \ External Interfaces \ Serial Port I/O.

Poniżej pokazano podstawowe elementy związane z obsługą złącza RS – 232.

– utworzenie obiektu portu RS–232 (COM1)

>> s = serial('COM1')

– ustawienie parametrów portu, np:

– prędkość transmisji – 4800 bitów/s

– liczba bitów danych – 8

– liczba bitów stopu – 1

>> set(s,'BaudRate',4800)

>> set(s,'DataBits',8)

>> set(s,'StopBits',1)

– nawiązanie połączenia z portem

>> fopen(s)

– zapis do portu komunikatu tekstowego: '*IDN?'

>> fprintf(s,'*IDN?')

– odczyt z portu odebranego komunikatu tekstowego:

>> out = fscanf(s)

– odłączenie od portu i usunięcie obiektu z pamięci:

>> fclose(s)

>> delete(s)

>> clear s

8. Wydruki m – plików

1. Skrypt zapisu przebiegu z karty dźwiękowej – p. 3.1.

```
    %parametry ustalane przez użytkownika ----------
    f_probkowania=10000;  %liczba próbek/sek/tor
    czas_zapisu=10;       %czas zapisu[s]
    %-----------------------------------------------
samples_per_trigger=f_probkowania*czas_zapisu;
ai = analoginput('winsound',0);
addchannel(ai,1);
set(ai,'SampleRate',f_probkowania);
set(ai,'SamplesPerTrigger',samples_per_trigger);
start(ai);
zapis = getdata(ai);
plot(zapis);
grid on;
delete(ai);
clear ai;
```

2. Funkcja filtrowania zadanego przebiegu x filtrem cyfrowym Czebyszewa typu I.

```
    function [b,y] = filtr_Czebyszewa_I(x)
    % --Parametry filtracji ----------------------
    % n: rząd filtra
    % Rp: zakres wahań w paśmie przepustowym [dB]
    % fp: częstotliwość próbkowania [Hz]
    % fc: częstotliwość progowa filtra [Hz]
    n=9;
    Rp=0.5;
    fp=10000;
    fc=400;
    % -----------------------------------------
%Częstotliwość względna
Wn=fc/(fp/2)
%Współczynniki filtra
[b,a] = cheby1(n,Rp,Wn);
%utworzenie obiektu filtra dla struktury
%bezpośredniej typu II, transponowanej
%i sprawdzenie stabilności filtra
Hd = dfilt.df2t(b,a);
b=isstable(Hd);
if b==0 %filtr jest niestabilny
  y=[];
else
  y=filter(Hd,x);
end;
```

3. Skrypt zapisu przebiegu do pliku tekstowego.

```
    %Zapis wyników pomiaru do pliku tekstowego 'pomiar.txt'
    %Wyniki znajdują się w zmiennej 'zapis' w przestrzeni roboczej Matlaba.
    %nazwa pliku; lokalizacja: aktualny katalog roboczy Matlaba
    S='pomiar.txt';
    f_probkowania=10000;    %liczba probek/sek/tor
    czas_zapisu=5;          %[s]
    T=1/f_probkowania;      %okres próbkowania [s]
    sz_t =6;   %liczba znaków kolumny czasu
    mpp_t=4;   %liczba miejsc po przecinku czasu
    sz_w =8;     %liczba znaków kolumny wartości próbki
    mpp_w=3;   %liczba miejsc po przecinku wartości próbki
    N=length(zapis);        %długość wektora zapis – liczba zapisanych próbek
    fid=fopen(S,'w');   %otwarcie pliku
    %zapis nagłówka pliku
    fprintf(fid,'Wyniki przykładowych pomiarów.\r\n');
    fprintf(fid,'Pomiary w jednym torze karty dźwiękowej.\r\n');
    fprintf(fid,'\r\n');
    fprintf(fid,'%*s\t%*s\r\n',sz_t,'t[s]',sz_w,'w[V]');
    %zapis kolumn czasu i wartości próbek
    for i = 1:N;
      t=(i–1)*T;          %czas próbki [s]
     w=zapis(i,1)*1000;  %wartość próbki przeliczone z [V] na [mV]

fprintf(fid,'%*.*f\t%*.*f\r\n',sz_t,mpp_t,t,sz_w,mpp_w,w);
    end;
    status=fclose(fid); %zamknięcie pliku
```

4.1. Przykład generowania złożonego przebiegu na wyjściu analogowym (p. 5.2)

```
     %Parametry ustalane przez użytkownika ----------
     f_probkowania_ai=10000;   %liczba probek / sek
```

```
/ tor – wejście analogowe
f_probkowania_ao=100000;  %liczba probek / sek / tor – wyjście analogowe
czas_ao=5;  %czas trwania przebiegu na wyjściu analogowym [s]
%-----------------------------------------------
global ai_zapis
global wynik_FFT
ao = analogoutput('nidaq','Dev1');
ai = analoginput ('nidaq','Dev1');
%konfiguracja wejść analogowych jako wejść ze wspólną masą
set(ai,'InputType','SingleEnded');
addchannel(ao,0);
addchannel(ai,0);
set(ao,'TriggerType','Manual');
set(ai,'TriggerType','Manual');
set(ao,'SampleRate',f_probkowania_ao);
set(ai,'SampleRate',f_probkowania_ai);
samples_per_trigger=f_probkowania_ai*czas_ai;
set(ai,'SamplesPerTrigger',samples_per_trigger);
f_ao_ust = get(ao,'SampleRate');  %faktycznie ustawiona częstotliwość %próbkowania wyjścia
czas=0:1/f_ao_ust:czas_ao;
f_harm1=1700; %częstotliwość harmonicznej 1 [Hz]
f_harm2=3250; %częstotliwość harmonicznej 2 [Hz]
f_harm3=4440; %częstotliwość harmonicznej 3 [Hz]
f_harm4=7400; %częstotliwość harmonicznej 4 [Hz] – niepoprawna –> alias
harm1=sin(2*pi*f_harm1*czas);
harm2=sin(2*pi*f_harm2*czas);
harm3=sin(2*pi*f_harm3*czas);
harm4=sin(2*pi*f_harm4*czas);
ao_przebieg = harm1 + harm2 + harm3 + harm4;
ao_przebieg = ao_przebieg'; %putdata wymaga danych w kolumnach
putdata(ao,ao_przebieg);
start(ai);
start(ao);
trigger(ai);
trigger(ao);
%Czekaj do zakończenia realizacji przebiegu na wyjściu.
%Max. czas oczekiwania: zadany czas przebiegu (czas_ao) + 5s.
wait(ao,czas_ao+5);
ai_zapis = getdata(ai);
delete(ai);
delete(ao);
clear ai;
clear ao;
FFT_01();  %sprawdzana funkcja
mag_FFT=1/(N/2)*abs(wynik_FFT); %amplituda
subplot(2,1,1);
N=length(wynik_FFT);
plot(czas,ai_zapis); grid on;
subplot(2,1,2);
f=(1:N/2)/N*f_probkowania_ai/1000;
plot(f,mag_FFT(1:N/2)); grid on;
```

4.2. Funkcja FFT_01() – sprawdzana w ramach przykładu z wydruku 4. 1.

```
function FFT_01()
global ai_zapis
global wynik_FFT
wynik_FFT = fft(zapis);
```

Truizmem jest stwierdzenie – ale warto je przytoczyć że stosowanie nowoczesnych narzędzi takich jak MATLAB przyczynia się do doskonalenia metod działalności zawodowej konstruktorów i badaczy.

Literatura

[1] Bocian S., Frączek J.: Zastosowanie programu MATLAB do zadań pomiarowych.
TRANSCOMP – XV INTERNATIONAL CONFERENCE COMPUTER SYSTEMS AIDED SCIENCE, INDUSTRY AND TRANSPORT (Logistyka 6/2011), Zakopane2011.
[2] Gąsowski W., Nowak R.: Badania wagonów kolejowych. Wyd. Politechniki Poznańskiej. Poznań, 1989.
[3] Gąsowski W.: Aerodynamika pociągu. Wyd. ITE, Radom, 1998.
[4] Frączek J.: Zastosowanie programu MATAB do zadań pomiarowych – wprowadzenie. Opracowanie OR – 10048. Archiwum, IPS „TABOR". Poznań wrzesień 20011.
[5] Zalewski A., Cegieła R.: Matlab – obliczenia numeryczna i ich zastosowanie. Nakom, Poznań 1996.
[6] MATLAB Reference Guide; High-Performance Numeric Computation and Visualization Software. The Math Works Inc.
[7] MATLAB User's Guide for Microsoft Windows. The Math Works Inc.
[8] USB-1408FS, USB-based Analog and Digital I/O Module. User's Guide. Measurement Computing.