|  |  |
| --- | --- |
|  | Warszawa, dnia 16 września 2020 r. |
|  |  |

Pan

Wiktor Książek, Bartłomiej Wach

Stażysta, Praktykant

Symulacyjne badania algorytmu regulacji dla napędu silnika prądu stałego z mostkiem H

Z wyrazami szacunku

Opiekun

dr Dariusz Świerczyński

Główny Specjalista

Spis treści

[1. Instalowanie niezbędnego oprogramowania 3](#_Toc51148720)

[2. Model napędu silnika prądu stałego z mostkiem H w programie PLECS 9](#_Toc51148721)

[3. Wykonanie układu regulacji napędu DC w dziedzinie ciągłej s. 14](#_Toc51148722)

[4. Wykonanie układu regulacji napędu DC w dziedzinie dyskretnej z. 17](#_Toc51148723)

[5. Wykonanie układu regulacji napędu w C i uruchomienie w SIL. 19](#_Toc51148724)

[6. Komunikacja z mikrokontrolerem F28379D 24](#_Toc51148725)

[7. Technologia PIL 29](#_Toc51148726)

[8. Utworzenie programu regulującego w C w CCS oraz uruchomienie w PIL 33](#_Toc51148727)

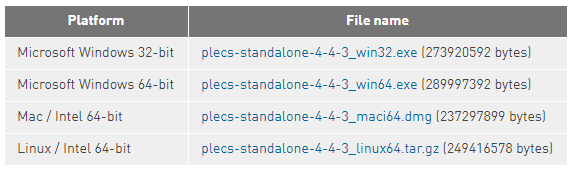
# 1. Instalowanie niezbędnego oprogramowania

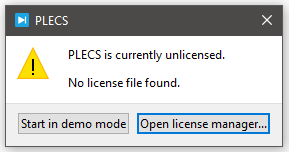
Do wykonania projektu wymagana jest instalacja i konfiguracja następującego oprogramowania:

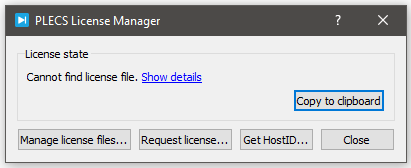
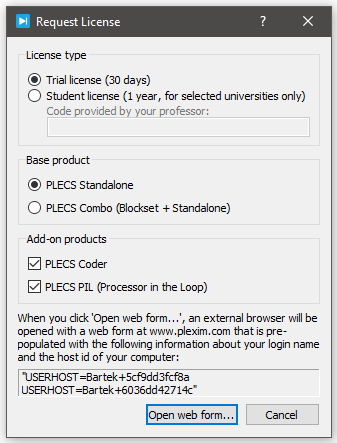
-Program PLECS 4.4.3;

-Program Code Composer Studio 6.2.0;

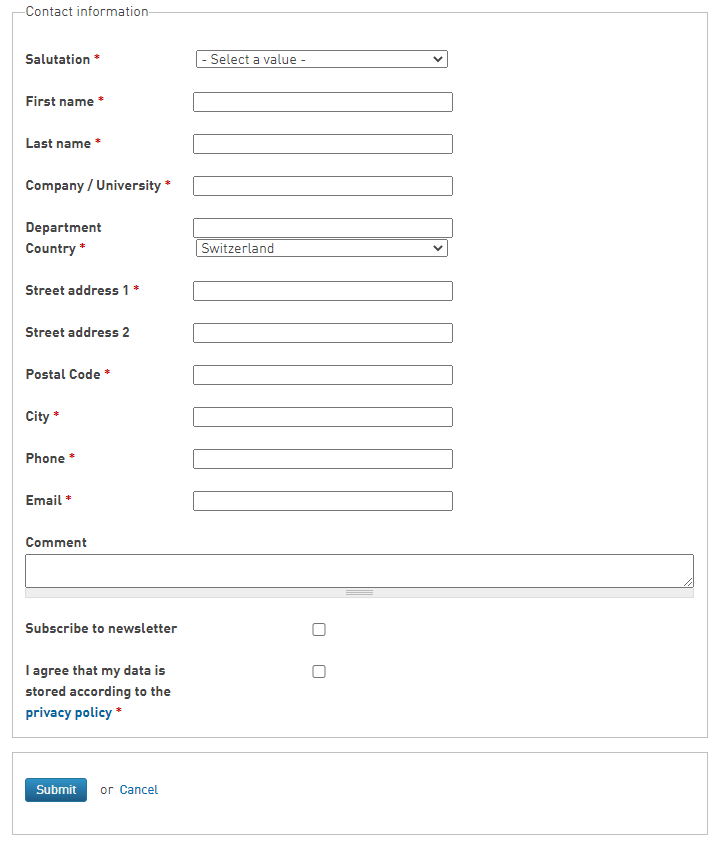
-Program UniFlash.

W celu instalacji programu PLECS 4.4.3, należy pobrać instalator, który można znaleźć pod linkiem: <https://www.plexim.com/download/standalone>.

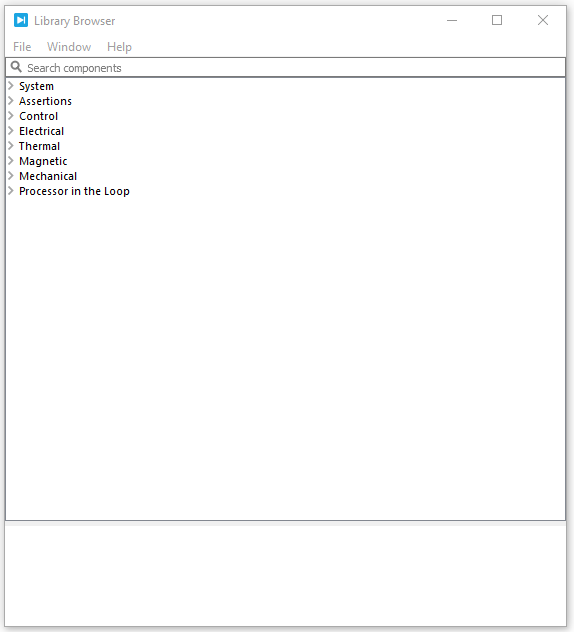
Po wybraniu wersji odpowiedniej do swojego systemu operacyjnego i pobraniu pliku instalacyjnego, należy go uruchomić i zainstalować program w domyślnej lokalizacji. Lokalizacja może być zmieniona, aczkolwiek może to przeszkodzić w kolejnych krokach wykonywania projektu. Po zakończonej instalacji, należy uruchomić program. Ukaże się okno informujące o braku licencji, z dostępnych opcji należy wybrać „Open license manager…”.

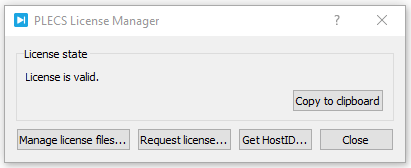
Po pojawieniu się okna „PLECS License Manager” należy wybrać opcję „Reguest license…”. Po ukazaniu się nowego okna należy zaznaczyć opcje „Trial License (30 days)”, „PLECS Standalone”, „PLECS CODER” i „PLECS PIL (Processor in the loop)”, a następnie kliknąć w przycisk „Open web form…”.

W nowo otwartym oknie przeglądarki internetowej, należy wypełnić formularz swoimi danymi, wystarczą jedynie wymagane dane zaznaczone odpowiednio czerwoną gwiazdką. Następnie należy kliknąć na przycisk „Submit”.

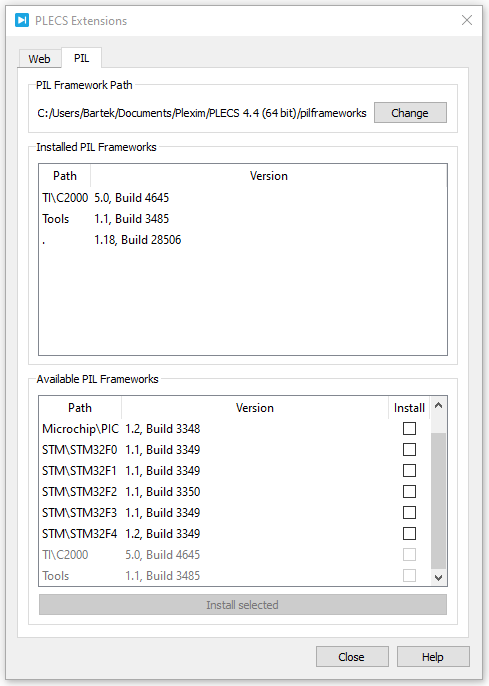


Na podany w formularzu adres e-mail, zostanie wysłany plik licencyjny. Czas oczekiwania na odpowiedź to zazwyczaj mniej niż 24 godziny. Po otrzymaniu pliku licencyjnego należy go pobrać i otworzyć ponownie program PLECS i w taki sam sposób jak poprzednio wejść w „PLECS license manager” i wybrać opcję „Manage license files” i następnie w nowo ukazanym oknie wybrać opcję „Install…”. W otwartym menadżerze plików wybrać uprzednio pobrany plik licencyjny z rozszerzeniem .lic i następnie kliknąć przycisk „Otwórz”. PLECS następnie powiadomi o zainstalowaniu pliku z licencją. Następnie należy zamknąć i otworzyć na nowo program.

Po uruchomieniu okno „Library Browser” powinno być widoczne.

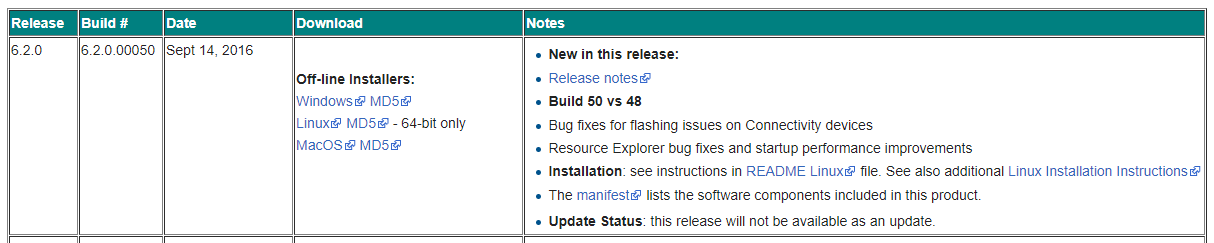
W celu sprawdzenia pliku licencyjnego należy kliknąć przycisk „Help” i następnie „PLECS License Manager”. Widoczny poniżej komunikat powinien być widoczny.

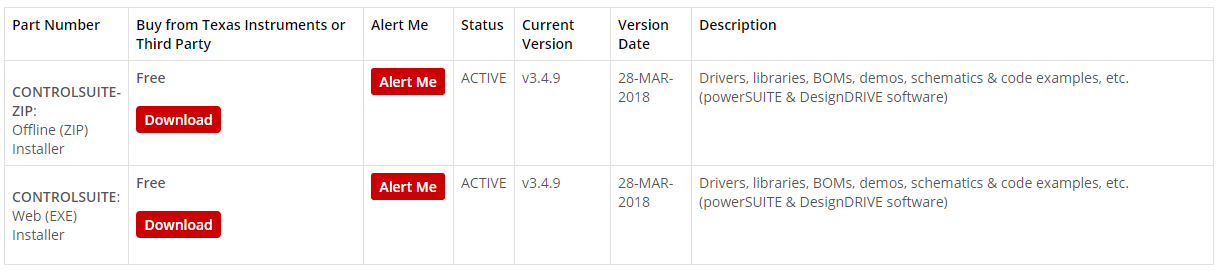
Następnie należy kliknąć w „File” w górnym menu i następnie „PLECS Extensions”. W podstronie PIL, paczki TI/C200 i Tools. Jeśli nie widnieją one w menu „Installed PIL Frameworks”, należy je zainstalować. Dokonuje się tego poprzez wybranie ich z menu „Available PIL Frameworks” i kliknięciu na przycisk „Install selected”.



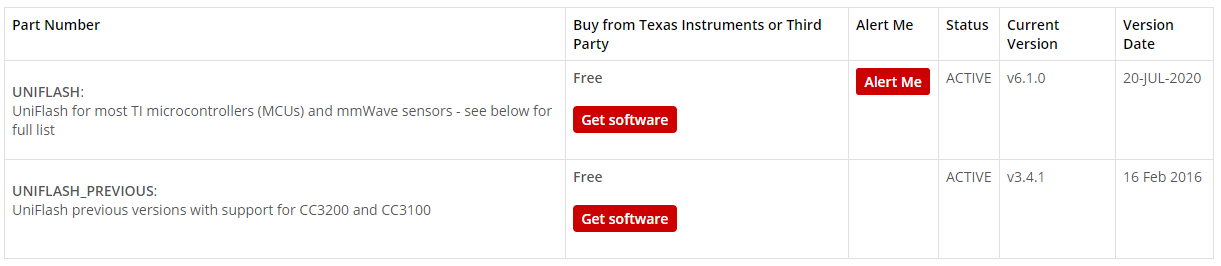
Po zainstalowaniu wybranych paczek sterowników należy zresetować program PLECS i ponownie wejść w „PLECS Extensions”, jeśli to okno wygląda tak jak na obrazie powyżej, program PLECS jest gotowy do pracy w nim.

W celu zainstalowania Code Composer Studio 6.2.0 należy udać się pod adres internetowy:<https://processors.wiki.ti.com/index.php/Download_CCS%20?DCMP=slulplaunch&HQS=ep-con-lprf-slulplaunch-pr-sw-ccs-en>.

Następnie znaleźć na liście Code Composer Studio 6.2.0 i w kolumnie „Download” wybrać instalator dla swojego systemu operacyjnego.

Do pobrania pliku instalacyjnego wymagane jest założone konto na stronie producenta. Przy próbie pobrania należy również wypełnić formularz „U.S. Government export approval” swoimi danymi, wybrać opcję „Civil” i „Yes”, następnie kliknąć na przycisk „Submit” i po przekierowaniu na kolejną stronę należy kliknąć przycisk „Download”. Po pobraniu pliku instalacyjnego należy go uruchomić i zainstalować program w domyślnej lokalizacji. Lokalizacja może być zmieniona, aczkolwiek może to przeszkodzić w kolejnych krokach wykonywania projektu. Po zakończonej instalacji, należy pobrać dodatkową paczkę ze sterownikami o nazwie ControlSuite, odpowiednimi dla naszego mikrokontrolera. Można ją znaleźć pod adresem internetowym: <https://www.ti.com/tool/CONTROLSUITE>. Należy wybrać drugą opcję z listy – instalator w formie pliku z rozszerzeniem .exe, również mieć na uwadze to że do pobrania tego pliku również wymagane jest konto na stronie producenta i wypełnienie formularza „U.S. Government export approval”.

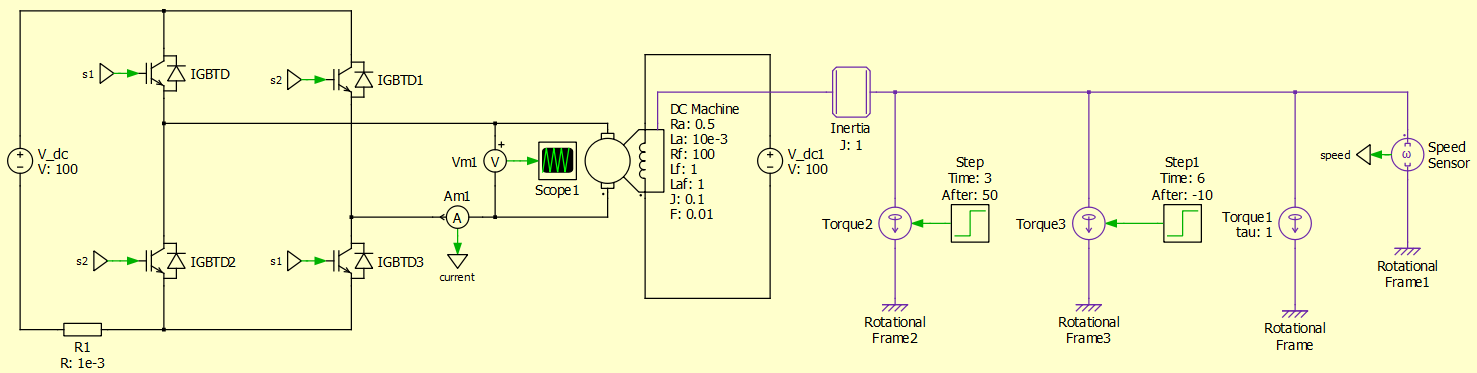
Po pobraniu pliku instalacyjnego, należy uruchomić go i zainstalować paczkę w domyślnej lokalizacji, chyba że program Code Composer Studio zainstalowany jest w innej lokalizacji niż domyślna. Finalnie ControlSuite powinien znajdować się w tym samym folderze co Code Composer Studio.

W celu zainstalowania programu UniFlash należy udać się pod adres internetowy: <https://www.ti.com/tool/UNIFLASH> i pobrać wersję 6.1.0 – pierwszą na liście.

W celu pobrania pliku instalacyjnego należy posiadać konto na stronie producenta jak i wypełnić formularz, tak jak przy instalacji Code Composer Studio 6.2.0.

Po zainstalowaniu wszystkich opisanych uprzednio programów można rozpocząć pracę nad komunikacją z mikrokontrolerem.

# 2. Model napędu silnika prądu stałego z mostkiem H w programie PLECS

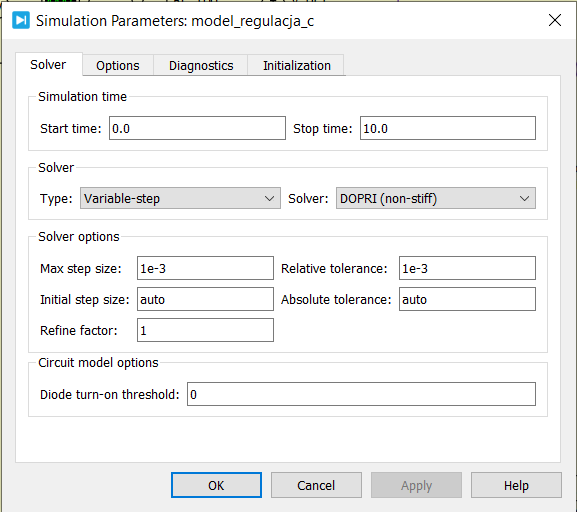


Do rozpoczęcia prac nad stworzeniem regulatora do napędu silnika DC konieczne jest zamodelowanie układu. Topologia układu napędowego wraz z najważniejszymi jego parametrami jest pokazana na rysunku wyżej. Jak widać układ zasilany jest z źródła napięcia stałego o wartości 100V i rezystancji wewnętrznej o wartości 1e-3 Ohm. Regulacja napięcia na zaciskach twornika maszyny DC jest dokonywana na mostku H zbudowanym na idealnych łącznikach (co dyktuje nam oprogramowanie PLECS). Mostek H jest to układ energoelektroniczny pozwalający na zmianę napięcia na swoim wyjściu w zależności od współczynnika wypełniania sygnału PWM sterującego naprzemiennym załączaniem i rozłączaniem par łączników S1 i S2 według wzoru:

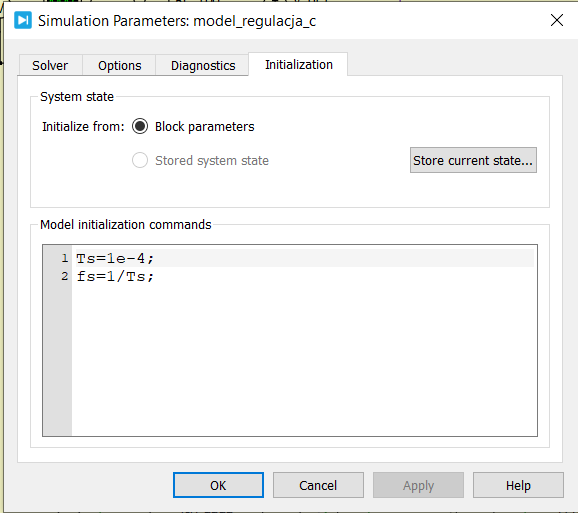
,gdzie D <0,1> nazywamy współczynnikiem wypełnienia, który jest stosunkiem czasu załączenia danego łącznika do sumy czasu załączenia i rozłączenia.

Parametry obwodu twornika ustalono na Ra =0.5 ohm oraz La = 10mH, a obwodu wzbudzenia na Rf= 100ohm, Lf = 1H oraz zasilania z idealnego źródła napięciowego o wartości 100V. Przyjmuje się idealne sprzężenie indukcyjności obydwu obwodów. Parametry mechaniczne maszyny przyjęto na J = 0.1 kg\*m2 (moment bezwładności) oraz F=0.01 (współczynnik tarcia).

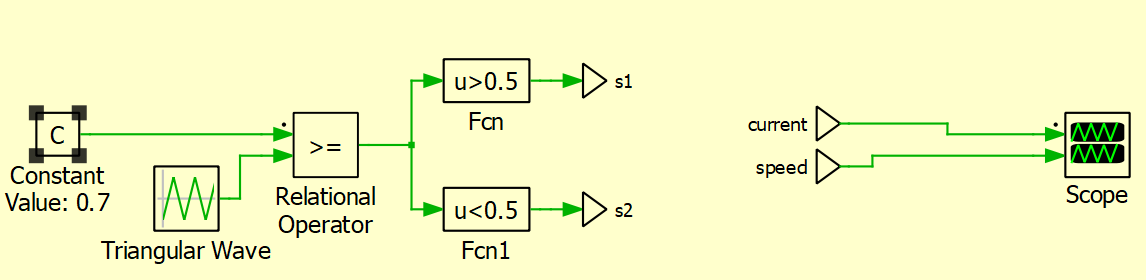
W trakcje trwania dziesięciosekundowej symulacji obciążenie maszyny zmienia się trzykrotnie. W ciągu pierwszych 3 sekund symulacji maszyna pracuje w stanie jałowym (obciążenie stanowi wyłącznie przyłączony wał o momencie bezwładności J=1kg\*m2 oraz moment o wartości M = 1N\*m). W czasie od 3 do 6 sekund po rozpoczęciu symulacji dodane do obciążenia jest 50 N\*m, a po 6 sekundzie odjęte jest 10 N\*m obciążenia. Ważne jest, aby pamiętać o tym, żeby nastawić czas regulacji na odpowiednią długość w oknie „simulation parameters”:



Korzystając z okazji należy nastawić również częstotliwość sygnału nośnego w modulacji PWM. Wykonujemy to deklarując wartości zmiennych w języku C w zakładce „Initialization”. Częstotliwość (lub okres) tą należy potem także wpisać w parametrach regulatorów dyskretnych.

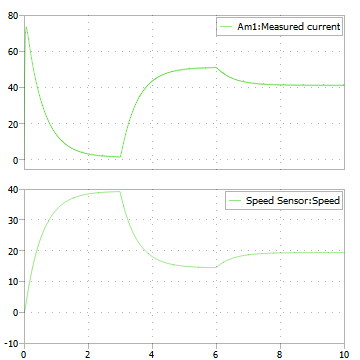


Po wykonaniu modelu układu napędowego konieczne jest wykonanie układu PWM do wysterowania bramek mostka H. Po wykonaniu odpowiedniego układu przedstawionego na poniższym rysunku uzyskujemy w pełni funkcjonalny nieregulowany układ napędowy.



Należy pamiętać o tym, żeby pary S1 oraz S2 pod żadnym pozorem nie były w tym samym stanie logicznym. Taka sytuacja doprowadziłaby do zwarcia na mostku H.

Zgodnie z założeniem na wyjściu mostka H po zadaniu wypełnienia D=0.7 pojawia się w uśrednieniu za okres napięcie o wartości 40V. Brak regulacji prądu oraz prędkości obrotowej maszyny elektrycznej powodują, że po zmianie obciążenia te dwie wielkości zmieniają się w przestawiony poniżej sposób.

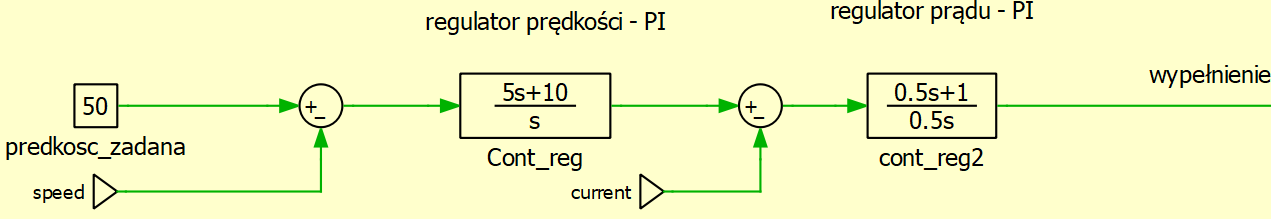


Jeżeli chcemy, aby napęd utrzymywał zadaną prędkość obrotową na wale niezależnie od obciążenia konieczne jest zastosowanie układu regulacji. Układ ten będzie zmieniał wypełnienie sygnału PWM (a w konsekwencji napięcie na wyjściu mostka H), w taki sposób, aby wymusić na tworniku maszyny taki prąd, który odpowiadać będzie założonej prędkości obrotowej.

# 3. Wykonanie układu regulacji napędu DC w dziedzinie ciągłej s.

Sterowanie z wykorzystaniem układu regulacji automatycznej z definicji wykorzystuje sprzężenie zwrotne. W układzie z takim sterowaniem dokonuje się pomiaru wartości wielkości regulowanej, a następnie ten pomiar wykorzystuje się do odpowiedniego wysterowania aktuatora, który wpływa na wielkość regulowaną. W naszym przypadku aktuatorem będzie mostek H, a czujnikami będzie amperomierz (pracujący na tworniku maszyny DC) oraz prędkościomierz (mierzący prędkość obrotową wału maszyny DC).

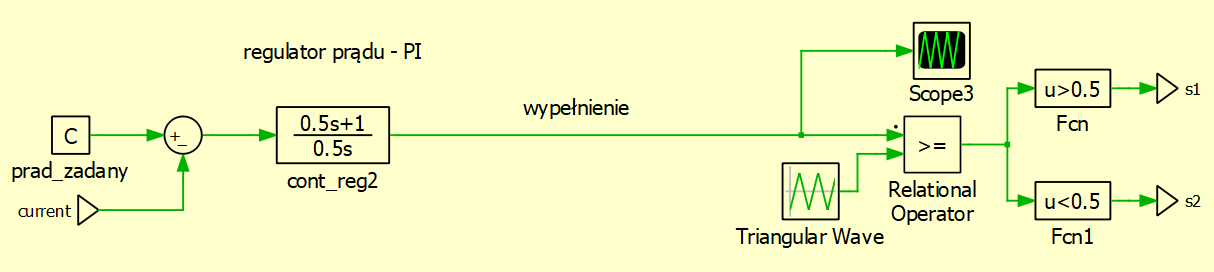
Do regulacji napędu DC wykorzystany zostaną wykorzystane szeregowo połączone regulatory prędkości i prądu. Obydwa regulatory mają formę równoległemo regulatora PI. Topologia układu sterowania przedstawiona została na poniższym rysunku:



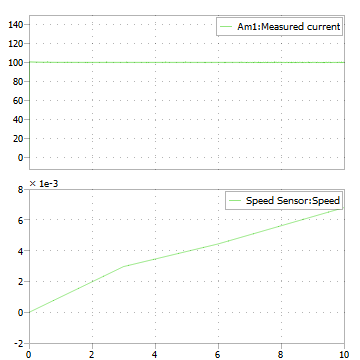
Zastosowanie regulatora PI powoduje, że uchyb ustalony prędkości oraz prądu jest praktycznie zerowy.

W celu zbudowania odpowiednio działającego regulatora konieczny jest odpowiedni dobór nastaw PI.

Na początku wykonano nastawy regulatora prądu. Po zadaniu prądu referencyjnego prąd płynący przez twornik nie powinien ulegać zmianom w stanie ustalonym. W celu wykonania odpowiedniego badania należy zablokować wał silnika. Można tego dokonać zwiększając do bardzo dużej wielkości moment bezwładności (np. 1e6).



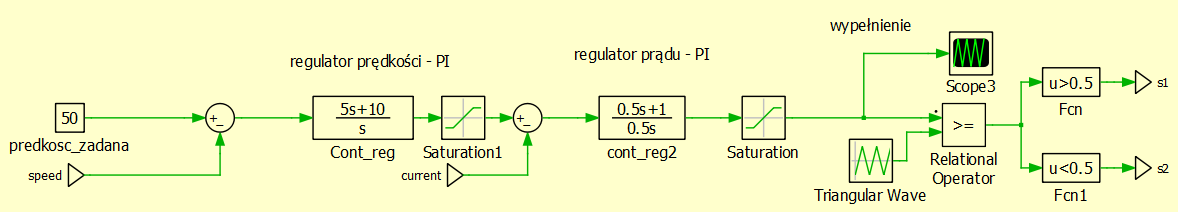
Dobrze działający regulator powinien nie mieć uchybu ustalonego oraz przesadnych tętnień prądu. Dla przedstawionych wyżej nastaw poprawny przebieg prędkości i prądu przedstawiono poniżej:



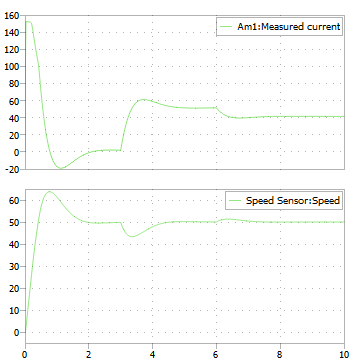
Warto zwrócić uwagę na to, że układ ten układ regulacji zawiera nasycenie. Wartość wypełnienia poniżej 0 lub powyżej 1 nie wpływa już na zmianę napięcia na wyjściu mostka H. Jest to więc układ nieliniowy.

W przypadku zbyt dużych tętnień prądu wyjściowego należy zmniejszyć wzmocnienie członu proporcjonalnego. Poprawa dynamiki następuje po zwiększeniu wzmocnienia na członie całkującym, ale może prowadzić ono również do zmniejszenia stabilności układu oraz zwiększenia wartości przewartościowania.

Po odpowiednim nastawieniu regulatora prądu można do tego regulatora szeregowo dołączyć regulator prędkości. Jako, że ten regulator nie powinien wymuszać przepływu prądu większego niż maksymalny chwilowy prąd w tworniku to dodano w tym regulatorze nasycenie. Wartość maksymalną prądu przyjęto na 150A.



Do wykonanie nastaw tego regulatora konieczne jest odblokowanie wału maszyny elektrycznej (wartość J wraca do 1). Wzmocnienia członu całkującego oraz różniczkującego należy dobrać z identycznymi kryteriami co regulator prądu. Odpowiednio wyregulowany układ sterowania powinien oddawać odpowiedź podobną do przedstawionej poniżej:

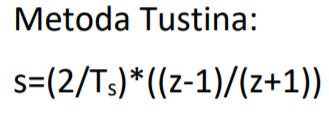


Na wykresach widać, że układ utrzymuje zadaną wartość prędkości obrotowej niezależnie od obciążenia. Oznacza to, że układ regulacji został odpowiednio wykonany w dziedzinie s (regulacja ciągła). Warto zwrócić uwagę, że wartość prądu nie wzrasta powyżej 150A, co oznacza, że nasycenie działa poprawnie.

# 4. Wykonanie układu regulacji napędu DC w dziedzinie dyskretnej z.

Wykonany dotychczas układ sterowania pomimo swojej poprawności nie jest praktycznie implementowany. Przeważająca większość regulatorów stosowanych współcześnie realizowane są nie na analogowych, ale cyfrowych układach. W przeciwieństwie do zbioru rezystorów, cewek, kondensatorów oraz wzmacniaczy operacyjnych stanowiących regulator analogowy, regulator cyfrowy oparty na mikrokontrolerze, może przyjąć wartości oraz zwrócić wartość wyjściową regulatora tylko pewnych chwilach czasowych.

Konieczne jest więc przekształcenie transmitancji regulatorów z postaci ciągłej na postać dyskretną. Z paru dostępnych metod dyskretyzacji (metoda Eulera w przód/tył, metoda Tustina) wybrano metodę Tustina. Metoda ta polega na podstawieniu operatora wyrażenia z operatorem z w miejsce operatów z zgodnie ze wzorem:



gdzie Ts to okres odpowiadający częstotliwości fs pracy regulatora.

Przekształcenie transmitancji ciągłych na dyskretne mogą zostać wykonane „na papierze”, ale wymaga to pewnej ilości czasu i skupienia. Alternatywną metodą rozwiązania problemu dyskretyzacji jest wykorzystanie poniższego skryptu w języku Matlab:

clc

clear

s = tf('s');

G1 = (10/s)+5;

D1 = c2d(G1,0.0001,"tustin");

G2 = (1/(0.5\*s))+1;

D2 = c2d(G2,0.0001,"tustin");

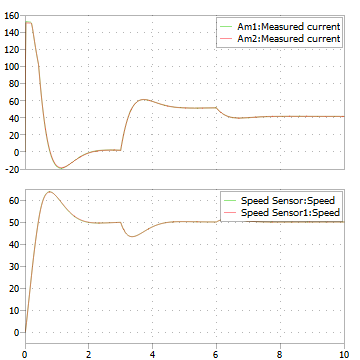
G1

D1

G2

D2

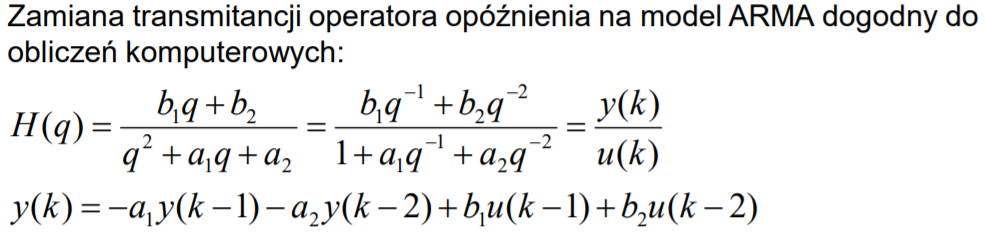
Otrzymane transmitancje dyskretne należy zamienić z transmitancjami ciągłymi. Odpowiedź układu powinna niemal idealnie pokrywać się z odpowiedziami uzyskiwanymi dzięki regulatorom czynnym (oczywiście w momentach odległych od siebie o Ts), co pokazano na wykresach poniżej:



# 5. Wykonanie układu regulacji napędu w C i uruchomienie w SIL.

Po uzyskaniu odpowiednio działających regulatorów w dziedzinie z można już łatwo przekształcić układ regulacji z postaci blokowej do kodu C. W tym celu należy przekształcić transmitancję dyskretną w dziedzinie z na równanie opisane operatorem wyprzedzenia k(który w realizowalnym układzie zamieniamy na operator opóźnienia q), korzystając z modelu autoregresyjnego z ruchomą średnią (ARMA). Pozwoli nam to w prosty sposób wykonać potrzebny algorytm.

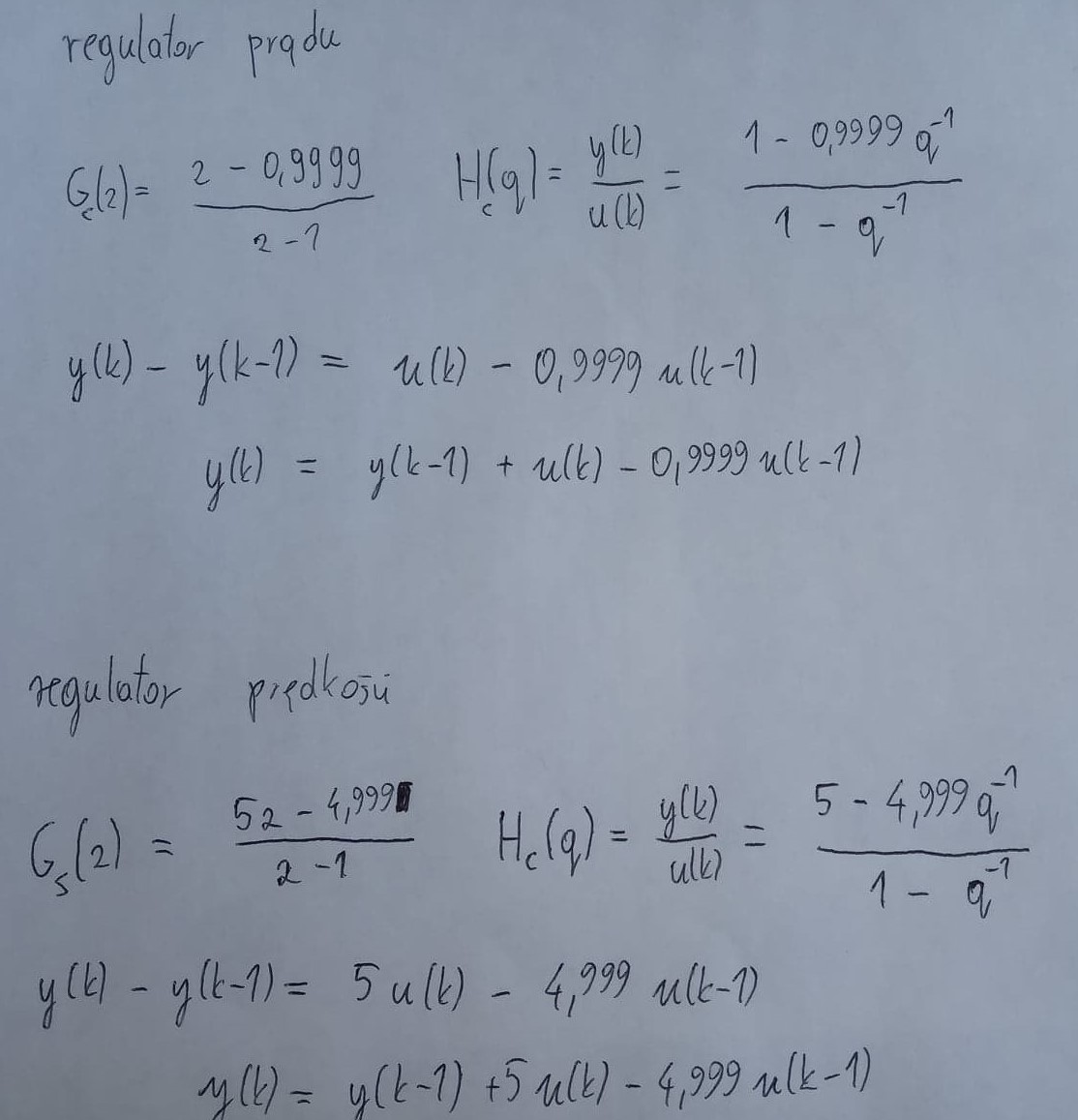
W celu przekształcenia transmitancji regulatora zamieniamy operator z na q (przez podstawienie) oraz wykonujemy przekształcenia zgodnie z poniższym przykładem:



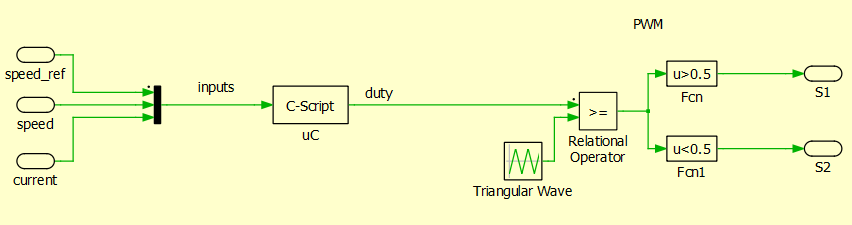
Przykład pochodzi z wykładu Dr inż. Wojciecha Kozińskiego o nazwie „Komputerowe sterowanie obiektami/Regulacja cyfrowa”

Należy pamiętać, że pomnożenie wyjścia/wejścia(y(k)/u(k)) przez operator opóźnienia o potędze -n powoduje, że wejście/wyjście to w równaniu będzie pobierane jako opóźnione o n próbek.

Przekształcenia wykonane dla regulatora prędkości i prądu użytych w przykładowym układzie napędowym przedstawiono poniżej:



Tak wykonane równania wyjściowe można już bezpośrednio użyć w kodzie. Aby dodatkowo uprościć schemat blokowy obwodu regulacji zintegrowano w kodzie C nasycenia na wyjściu (patrz przykład). Tak wykonany układ regulacji zawarto w pojedynczym bloku C-script. Układ regulacji w języku C:



Blok C należy ustawić na 3 wejścia i 1 wyjście. W odpowiednich zakładkach w kategorii „Code” napisano odpowiedni program:

- „Code declarations”:

struct EXAMPLE\_STRUCT{

float s\_ref;

float s;

float diff1;

float diff1\_1;

float reg\_s;

float reg\_s\_1;

float c;

float diff2;

float diff2\_1;

float out;

float out\_1;

};

void Calculate(struct EXAMPLE\_STRUCT \*data){

data->diff1 = data->s\_ref - data->s; //wyliczanie uchybu predkosci

data->reg\_s = data->reg\_s\_1+5 \* data->diff1 - 4.999 \* data->diff1\_1; //regulator prędkości

data->diff1\_1 = data->diff1; //zapisywanie ostatniej próbki

data->reg\_s\_1 = data->reg\_s; //zapisywanie ostatniej próbki

if(data->reg\_s > 150) data->reg\_s = 150; //wykonanie nasycenia prądu 150A

else if(data->reg\_s < -150) data->reg\_s = -150; /wykonanie nasycenia prądu -150A

data->diff2 = data->reg\_s - data->c; //wyliczanie uchybu pradu

data->out=data->out\_1+data->diff2-0.9998\*data->diff2\_1; //regulator prądu

data->diff2\_1=data->diff2; //zapisywanie ostatniej próbki

data->out\_1=data->out; //zapisywanie ostatniej próbki

if(data->out > 1) data->out = 1; //nasycenie wypełnienia

else if(data->out < 0) data->out = 0; //nasycenie wypełnienia

}

struct EXAMPLE\_STRUCT reg1;

- „Start function code”:

reg1.s\_ref=0;

reg1.s=0;

reg1.diff1=0;

reg1.diff1\_1=0;

reg1.reg\_s=0;

reg1.reg\_s\_1=0;

reg1.c=0;

reg1.diff2=0;

reg1.diff2\_1=0;

reg1.out=0;

reg1.out\_1=0;

- „Output function code”:

Output(0)=reg1.out; //wyprowadzenie zmiennej out na wyjście

- „Update function code”:

//sczytanie wejść

reg1.s\_ref=Input(0);

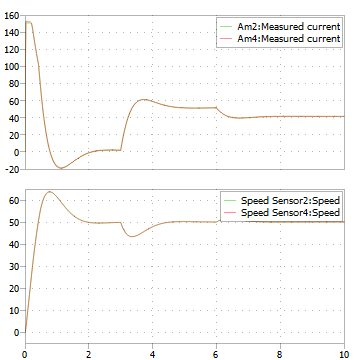
reg1.s=Input(1);

reg1.c=Input(2);

//obliczanie wyjścia

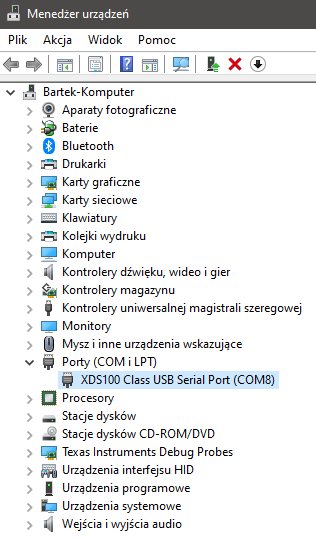
Calculate(&reg1);

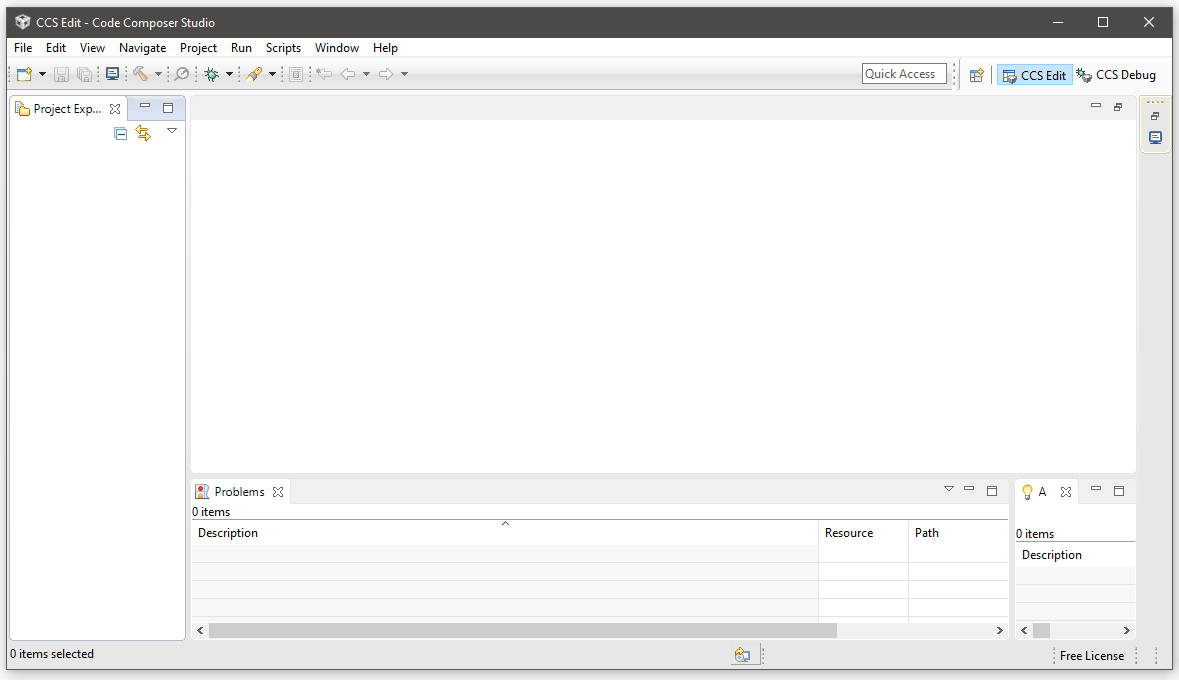
W celu sprawdzenia poprawności działania bloku C-Script wraz z programem regulacyjnym (oraz poprawności algorytmu ARMA) porównano przebiegi prądu i prędkości tego układ z układem wykorzystującym regulator dyskretny w postaci blokowej. Wyniki porównania przedstawiono poniżej:

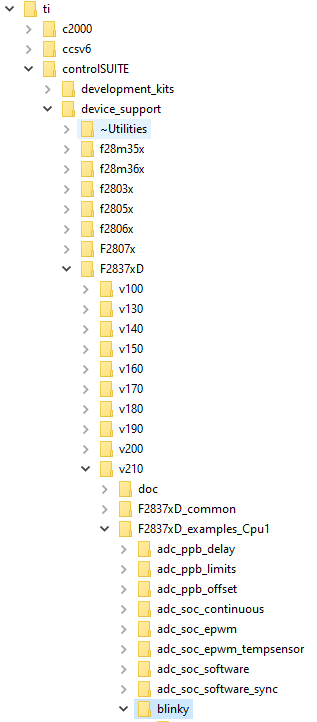


Jak widać układ regulacyjny w postaci blokowej jest praktycznie równoważny do układu zapisanego w kodzie C. Jedyną różnicą między dwoma przebiegami jest to, że układ regulacyjny w kodzie C wolniej wchodzi w nasycenia (w praktyce oznacza to, że maksymalny prąd jest o około 2A większy niż dla układu blokowego). Wynika to najpewniej z tego, że w kodzie C nasycenie jest sprawdzane raz na okres głównej pętli, co może być zbyt rzadkie w porównaniu do układu blokowego.

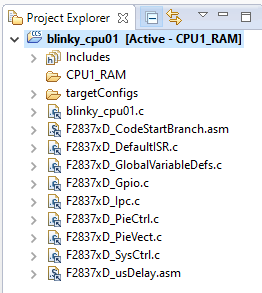
# 6. Komunikacja z mikrokontrolerem F28379D

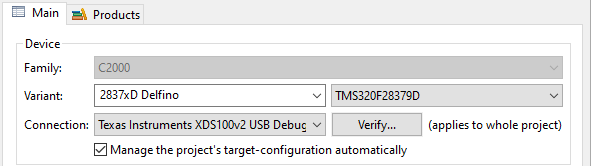
W celu uzyskania i sprawdzenia komunikacji z mikrokontrolerem, należy go (mikrokontroler) podłączyć do naszego komputera poprzez kabel USB-Mini USB. Po podłączeniu, mikrokontroler zainstaluje sterowniki potrzebne do komunikacji automatycznie. Po ukończonej instalacji, powinien on być widoczny w menadżerze urządzeń jako XDS100 Class USB Serial Port (COMX) – X to liczba przypisana przez komputer.

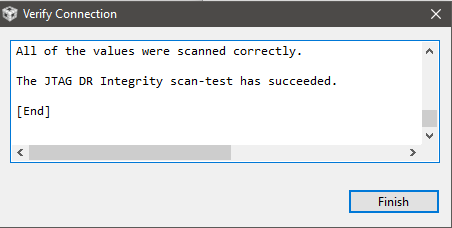
Następnie należy uruchomić program Code Composer Studio 6.2.0, przy pierwszym uruchomieniu program zapyta o zainstalowanie dodatkowych paczek – należy zezwolić na instalację. Po uruchomieniu program zapyta o ścieżkę do „Workspace” czyli miejsca w którym będziemy przechowywać projekty, najlepiej wybrać domyślną opcję. Później poniższe okno powinno być widoczne.

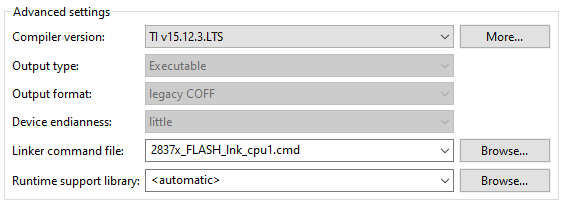
W górnym menu wybieramy „Project”, następnie „Import CCS Project”. W nowo ukazanym oknie należy wybrać opcję „Browse…”. Po otworzeniu się menadżera plików wybieramy dysk na którym zainstalowaliśmy Code Composer Studio i ControlSuite, następnie folder „ti”, „controlSUITE”, „device\_support”, „F2837xD”, „v210”, „F28379xD\_exapmles\_Cpu1”, „blinky”.

Po wybraniu katalogu klikamy przycisk „OK”, zaznaczamy opcję „Copy Projects into workspace”, a następnie „Finish”.

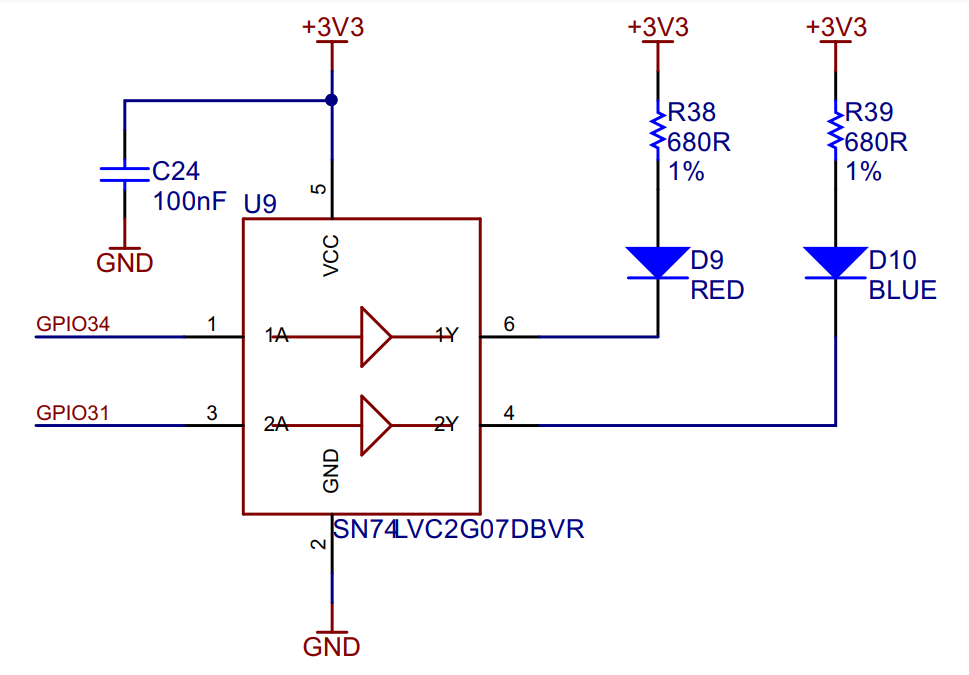
Projekt „Blinky” jest teraz widoczny w oknie „Project Explorer” po lewej stronie. W przypadku braku tego okna klikamy w górnym menu na „View” i potem „Project Explorer”. Strzałką obok nazwy projektu można rozwinąć projekt, należy to zrobić.

Następnym krokiem jest wejście we właściwości projektu, w tym celu należy kliknąć prawym przyciskiem myszy na nazwę projektu – „blinky\_cpu01 [Active – CPU1\_RAM]” i następnie „Properties”. W nowo otworzonym oknie w zakładce „Resource”, „General”, a następnie „Main” należy wybrać mikrokontroler, którego używamy, w tym przypadku jako wariant należy wybrać „2837xD Delfino”, „TMS320F28379D”. Jako Connection wybrać należy „Texas Instruments XDS100v2 USB Debug Probe”.

Następnie klikamy przycisk „Verify…”. Pojawi się okno „Verify Connection”, po zjechaniu na dół okna powinno być widać wiadomość o poprawnej weryfikacji połączenia. Jeśli okno nie wygląda jak to na obrazie poniżej, należy powtórzyć poprzednie kroki.

Po udanej próbie komunikacji klikamy przycisk „Finish”. Następnie należy wybrać wersję kompilatora w „Advanced Settings”, „Compiler Version” ustawiamy na TI v15.12.3.LTS. Jeśli nie można wybrać tej wersji należy ją pobrać i zainstalować w ścieżce, gdzie zainstalowane jest Code Composer Studio, w folderze „Tools” i następnie „Compiler”.

Następnie należy kliknąć „OK”, zapisując tym samym zmiany.

Kolejnym krokiem jest zbudowanie projektu, należy zrobić to poprzez naciśnięcie strzałki rozwijającej opcje, znajdującej się przy rysunku młotka w górnym menu i następnie wybraniu LAUNCHPAD\_FLASH lub LAUNCHPAD\_RAM. Podczas testowania kodu zalecane jest używanie pamięci RAM. Gdy kod się zbuduje można przejść do debugowania. Jeśli program zwraca błędy podczas budowania go, należy powtórzyć poprzednie kroki. W celu debugowania programu, należy kliknąć ikonę robaka w górnym menu. Po kliknięciu Code Composer Studio wejdzie w tryb debuggera, można to zauważyć w prawym górnym rogu. Gdy Code Composer Studio zakończy debugować program należy go uruchomić klikając zielony trójkąt z żółtym prostokątem w górnym menu. Po naciśnięciu można zauważyć mrugającą niebieską diodę na naszym mikrokontrolerze. By wyjść z sesji debugera należy nacisnąć czerwony kwadrat w górnym menu. Warto mieć na uwadze, że różne mikrokontrolery mogą mieć różne ustawienia pinów GPIO, co oznacza, że jeśli nasz program po zbudowaniu i debugowaniu nie sprawia, że dioda led zapala się i gaśnie, to należy zmodyfikować kod. Można to zrobić, również przy działającym programem w celu testu. Dokładne oznaczenia pinów GPIO można znaleźć w poradniku użytkowania mikrokontrolera, dostępnej pod adresem: <https://www.ti.com/lit/ug/sprui77c/sprui77c.pdf?ts=1600153012360&ref_url=https%253A%252F%252Fwww.google.com%252F>.

Jak widać na obrazie powyżej, diody w mikrokontrolerze F2837xD mają wyjścia GPIO34 i GPIO31. Chcąc zmienić, by inna dioda mrugała po wgraniu programu, należy zmodyfikować następującą linijkę kodu:

#define BLINKY\_LED\_GPIO 31

GPIO31 które należy do diody niebieskiej, zamieniamy na GPIO34, które należy do diody czerwonej. Powstały kod wygląda następująco:

#define BLINKY\_LED\_GPIO 34

Chcąc wgrać zmodyfikowany kod, należy zbudować program, a następnie go zdebugować, tak jak w poprzednich krokach. Po wykonaniu tych kroków dioda czerwona powinna zapalać się i gasnąć.

Posiadając sprawną komunikację z mikrokontrolerem, można przejść do kolejnego kroku projektu, którym jest połączenie mikrokontrolera z programem PLECS używając technologii Processor in the Loop (PIL).

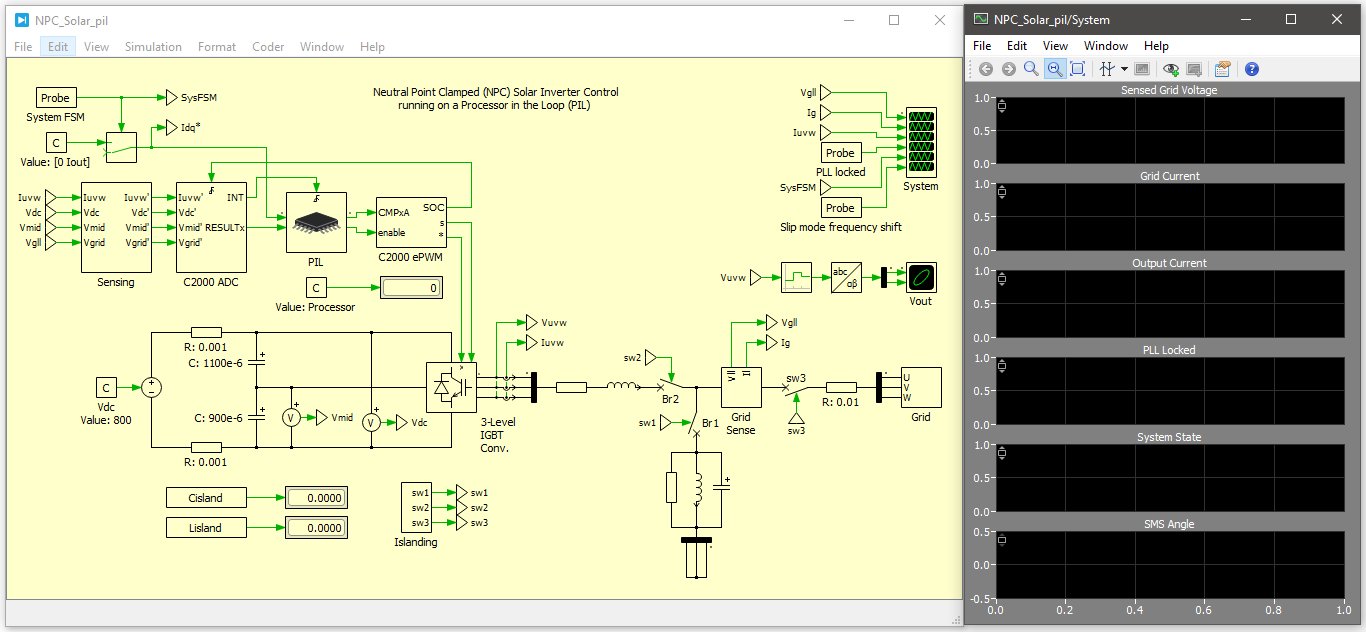
# 7. Technologia PIL

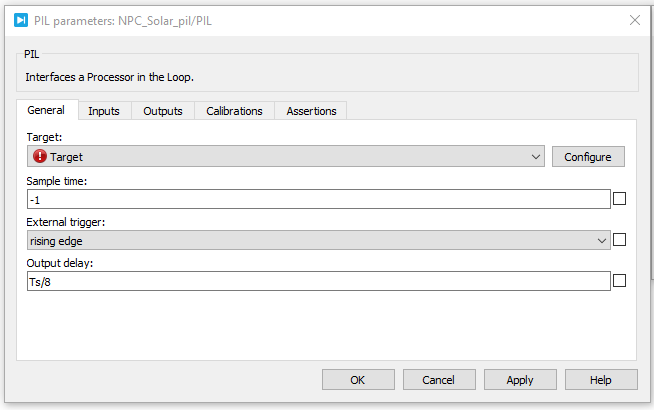
PIL, czyli Processor in the Loop to technologia, która pozwala na komunikację między mikrokontrolerem, a modelem na komputerze. Dzięki niej model kontrolera może być sprawdzony na platformie symulacyjnej, takiej jak na przykład program PLECS.

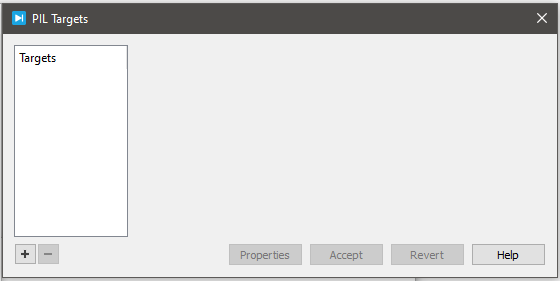
By rozpocząć działanie z technologią PIL potrzebne są wszystkie programy i dodatkowe paczki sterowników zawarte w rozdziale pierwszym. Bez nich praca z tą technologią jest niemożliwa.

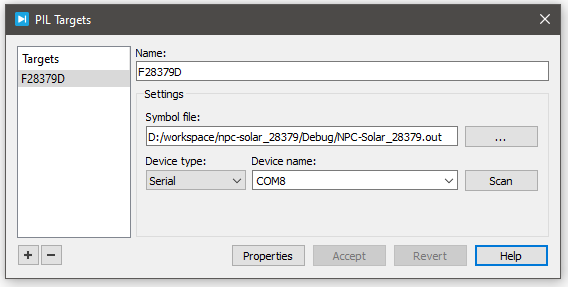
Należy uruchomić program Code Composer Studio i importować do niego projekt „npc-solar”. Tak jak w poprzednim rozdziale, należy kliknąć w górnym menu „Project”, „Import CCS Projects…”, „Browse..” i znaleźć w eksploratorze plików folder instalacyjny programu PLECS, dalej „pilframework”, „”TI”, „C2000”, „5.0”, „npc-solar” i w nim wybrać folder 28379, a następnie kliknąć przycisk „OK”, zaznaczyć opcję „Copy projects into workspace” i dalej „Finish”. Tym sposobem nasz projekt został importowany do obszaru roboczego „Workspace” jako kopia.

W tym projekcie konfiguracja komunikacji z mikrokontrolerem nie jest wymagana, jest ona ustawiona domyślnie. WAŻNE! Podczas zmiany ustawień komunikacji we właściwościach projektu, pliki w których domyślnie ustawiona jest komunikacja ulegają zmianie i nie jest możliwe powrócenie do początkowego stanu! Jeśli konfiguracja komunikacji została zmieniona, należy usunąć projekt z obszaru roboczego (kliknąć prawym przyciskiem myszy na nazwę projektu i następnie wybrać opcję „Delete”, w nowo otworzonym oknie zaznaczyć opcję „Delete project contents on disk”) i następnie importować projekt ponownie. Następnym krokiem jest zbudowanie projektu poprzez kliknięcie ikony młotka w górnym menu programu. Następnie należy kliknąć w ikonę robaka w celu debugowania programu i wgrania go na mikrokontroler. Po zakończonym debugowaniu należy uruchomić program, naciskając zieloną strzałkę z żółtym prostokątem w górnym menu. Mrugająca niebieska dioda powinna być widoczna.

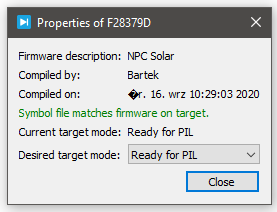
Następnym krokiem jest otwarcie projektu w programie PLECS w tym celu należy otworzyć folder „npc-solar” w tej samej lokalizacji z której importowany był projekt do programu w Code Composer Studio, następnie należy wejść folder „demo” i uruchomić plik NPC\_Solar\_pil.plecs, poniższe okna powinny być widoczne.

Następnie należy wejść w blok „PIL”, poprzez podwójne kliknięcie na niego lewym przyciskiem myszy. Ukaże nam się okno PIL Parameters. W zakładce „General” należy kliknąć przycisk „Configure”, pozwoli to na skonfigurowanie połączenia z mikrokontrolerem.

W nowo otwartym oknie należy kliknąć przycisk „+”,

dalej w polu „Name” należy wybrać nazwę naszej konfiguracji, przykładowo „F28379D”. W polu „Symbol file” kliknąć przycisk „…”, otworzy się eksplorator plików, należy przejść do folderu obszaru roboczego „Wrokspace”, który był podawany przy uruchamianiu programu Code Composer Studio, dalej otworzyć folder „npc-solar\_28379”, dalej folder „Debug”, wybrać plik „NPC-Solar\_28379.out” i kliknąć przycisk „Otwórz”. Kolejnym krokiem jest pole „Device type” tutaj wybieramy opcję „serial”, a następnie w polu „Device name” wybieramy odpowiedni port z mikrokontrolerem, przypisany mu podczas pierwszej próby podłączenia do komputera. Ustawienia w oknie „PIL Targets” i na obrazie poniżej powinny być takie same lub różniące się jedynie numerem portu COM.

Następnie klikamy przycisk „Accept”, i potem „Properties”. Poniższe okno powinno być widoczne.



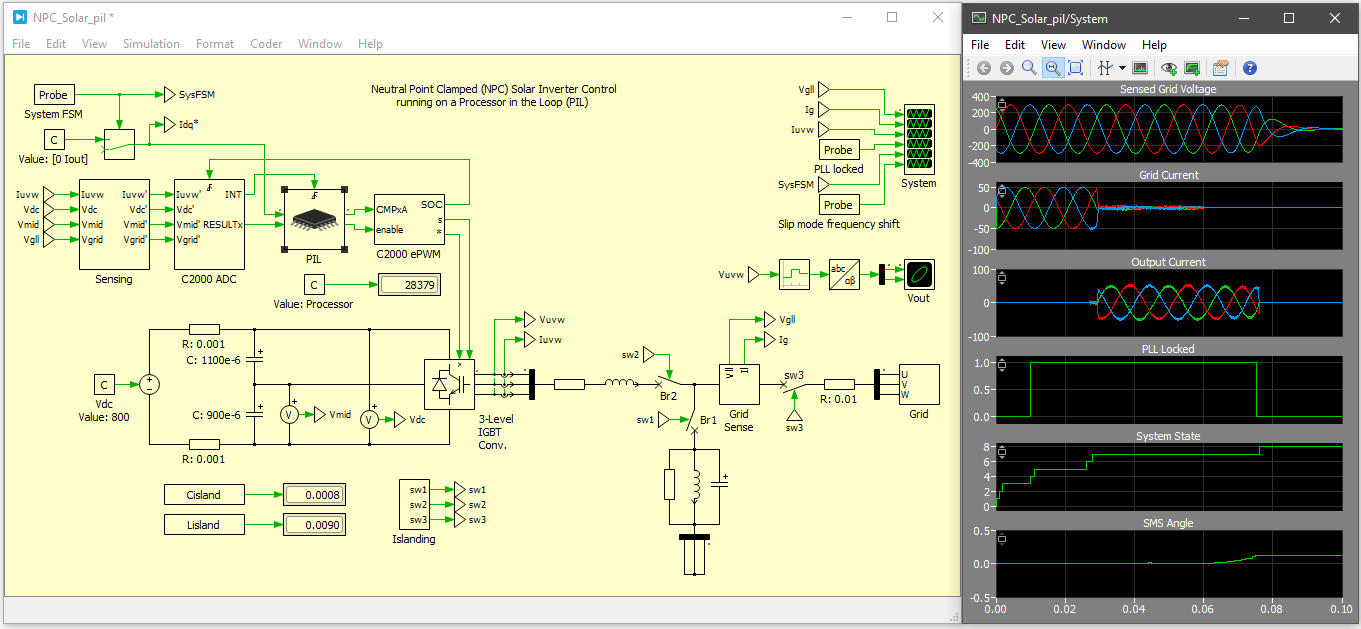
Ważnym jest by w polu „Desired target mode” była wybrana opcja „Ready for PIL”.

Widoczna jest również wiadomość podświetlona na zielono „Symbol file matches firmware on target”. Podczas budowania projektu w Code Composer Studio, tworzone są pliki „pil\_symbols\_c.inc” i „pil\_symbols\_p.inc”, są one w późniejszym etapie budowania projektu wykorzystywane do utworzenia pliku „NPC-Solar\_28379.out”. Plik z rozszerzeniem .out zawiera wszystkie zmienne zawarte w kodzie projektu. Podczas debugowania plik ten jest wgrywany do pamięci FLASH mikroprocesora. Podczas pracy w oknie „PIL Targets” wybrany został ten sam plik. Wiadomość podświetlona na zielono informuje o tym, że plik symbolów na mikrokontrolerze jest tym samym co wybranym w oknie „PIL Targets”. Oznacza to, że od teraz możliwa jest komunikacja pomiędzy modelem w programie PLECS a mikrokontrolerem.

Okno „Properties of F28379D” i “PIL Targets” można zamknąć. Następnie w oknie PIL parameters” w polu target należy wybrać z listy nowo utworzoną konfigurację, kliknąć „Apply” a następnie „OK”.

Kolejnym krokiem jest uruchomienie symulacji, w oknie z modelem w programie PLECS w górnym meny wybieramy „Simulation” a następnie „Start”.

Jeśli okno w wynikami oscyloskopu nie jest widoczne, należy kliknąć na niego dwa razy lewym przyciskiem myszy.

Po ukończonej symulacji poniższe wykresy powinny być widoczne.

# 8. Utworzenie programu regulującego w C w CCS oraz uruchomienie w PIL

W celu testowania praktycznej implementacji algorytmu sterowania napisano program realizujący układ regulacji na rzeczywistym mikrokontrolerze w środowisku CCS. Projekt wykonany w CCS został wykonany specjalnie z myślą o możliwości symulacji pracy mikrokontrolera wraz z symulacją napędu w programie PLECS. Taki sposób realizacji testowania algorytmu sterowania nazywany jest PIL (Processor in a Loop) i stanowi on następny krok w stosunku do wcześniej pokazanych symulacji SIL (Software in a Loop), które polegają na realizacji algorytmu sterowania wyłącznie na komputerze realizującym symulację całego napędu. Proces rozwoju algorytmu sterowania mikrokontrolerowego przebiega przeważnie (szczególnie w większych projektach) według przedstawionego poniżej schematu:

