Itasdi

Co-funded by the
Erasmus+ Programme
of the European Union

ITASDI PROJECT

Innovative Teaching Approaches in development of Software Designed Instrumentation and its application in real-time systems

Erasmus+ KA2 2018-1-RS01-KA203-000432

Modifcation of course „Principles of the Virtual Instruments Design“

Instrukcja do laboratorium 13: „Współpraca LabVIEW z Arduino”

Leader Partner: [1]Warsaw University of Technology
Contribution: [2]University of Novi Sad, [3]University of Belgrade
Authors: Dariusz Tefelski [1], Angelika Tefelska [1], Boris Jakovljevic[2], Marko Barjaktarovic[3]

Circulation: Public
Version: 01
Stage: Final
Date: 11.01.2019

Funding Disclaimer
This project has been funded with support from the European Commission. This communication reflects

1 Arduino

Arduino to otwarto-źródłowa elektroniczna platforma, której serce stanowi 8-bitowy mikrokontroler Atmel AVR. Powstał jako proste narzędzie do prototypowania dla studentów bez doświadczenia w zakresie elektroniki i programowania. Przez swoją prostotę zyskał zwolenników na całym świecie w szczególności w środowiskach tzw. *makers*. Ze względu na rosnące zapotrzebowania Arduino zaczęło rozwijać się dodając nowe wersje płytek dedykowanym różnym celom np. *Internet of Things* (IoT), medycznym pomiarom, budowy drukarek 3D czy systemów wbudowanych.

Płytka Arduino Mega 2560 została zaprojektowana do złożonych projektów i wyposażona w 54 wejść/wyjść cyfrowych (w czym 15 z nich może być użyte jako wyjścia PWM) oraz 16 wejść analogowych. Oprócz znacznie bogatszej ilości wejść/wyjść cyfrowych i analogowych zasadniczą zaletą Arduino Mega 2560 nad myDAQ jest posiadanie wyjść PWM, magistrali I2C oraz SPI.

Główne środowisko programistyczne dedykowane dla Arduino jest oparte na Wiring i zasadniczo C++. Jednak istnieje biblioteka LINX stworzona przez LabVIEW MakerHub, organizację wspierającą środowisko *makers* przy użyciu LabVIEW. Więcej na stronie https://www.labviewmakerhub.com/. Bibliotekę LINX można zainstalować ze strony: http://sine.ni.com/nips/cds/view/p/lang/pl/nid/212478 a także wykorzystując narzędzie VIPM (VI Package Manager).

Aby wgrać firmware do Arduino należy po uruchomieniu LabVIEW wybrać: **Tools** → **MakerHub** → **LINX** → **LINX firmware Wizard...** a następnie wybrać odpowiedni typ płytki Arduino oraz port szeregowy.

2 Biblioteka LINX

Biblioteka LINX zawiera podstawowe funkcji takie jak:

- **Open Serial** - funkcja, która rozpoczyna połączenie z urządzeniem na wybranym porcie szeregowym.
- **Close** - funkcja, która kończy połączenie z danym urządzeniem.

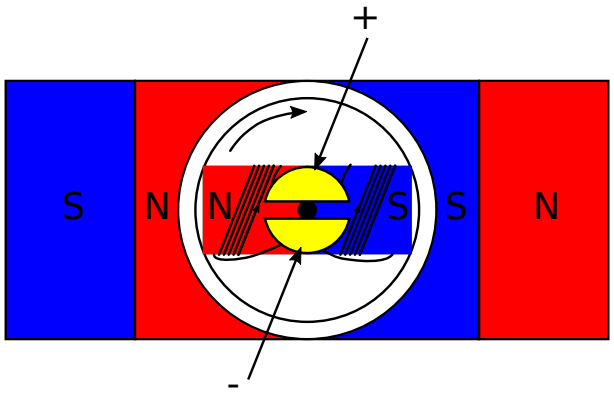
oraz funkcje zgrupowane w katalogach:

- *Peripherals* → *Analog* - do obsługi wejść analogowych.
- *Peripherals* → *Digital* - do obsługi wejść/wyjść cyfrowych.
- *Peripherals* → *PWM* - do obsługi wyjść generujących sygnał PWM.
- *Peripherals* → *I2C* - do obsługi magistrali I2C.
- *Peripherals* → *SPI* - do obsługi magistrali SPI.

3 Silniki DC

3.1 Zasada działania silnika DC

Silniki DC, czyli prądu stałego (Direct Current) mogą być zbudowane w różny sposób, ale generalnie łatwo zrozumieć je jako urządzenia, które zapewniają ruch obrotowy wału w wyniku odpychania jednoimiennych biegunów magnetycznych bądź przyciągania różnoimiennych (N-S). Silnik DC zbudowany jest zazwyczaj z magnesu stałego w statorze (części nie poruszającej się) oraz elektromagnesów w rotorze (części poruszającej się). Elektromagnes (-y) w rotorze w zależności od kierunku przepływu prądu przez uzwojenie wytwarzają po "zewnętrznej stronie" albo biegun S albo N. Aby zapewnić ruch obrotowy, kierunek prądu musi być w odpowiednim miejscu zmieniany. Zapewnia to komutator, czyli element, który w wyniku obrotu zmienia polaryzację doprowadzeń do uzwojeń cewek w elektromagnesach. Komutator w małych silniczkach zbudowany jest najczęściej ze sprężynujących blaszek, które stykają się z polami na części obracającej się. W większych silnikach stosuje się grafitowe "szczotki" popychane sprężynkami. Po długotrwałej eksploatacji ulegają one zużyciu i konieczna jest ich wymiana. Obecnie coraz częściej do użycia wchodzą silniki bezszczotkowe, które mają stały magnes w rotorze, a elektromagnesy w statorze. Osobne uzwojenia statora wyprowadzone są na zewnątrz a za ich załączanie odpowiada układ elektroniczny.



Rysunek 1: Silnik DC w uproszczeniu. Szczotki (strzałki) doprowadzają zasilanie do komutatora, który zaznaczony jest na żółto. Rotor obraca się zgodnie ze wskazówkami zegara.

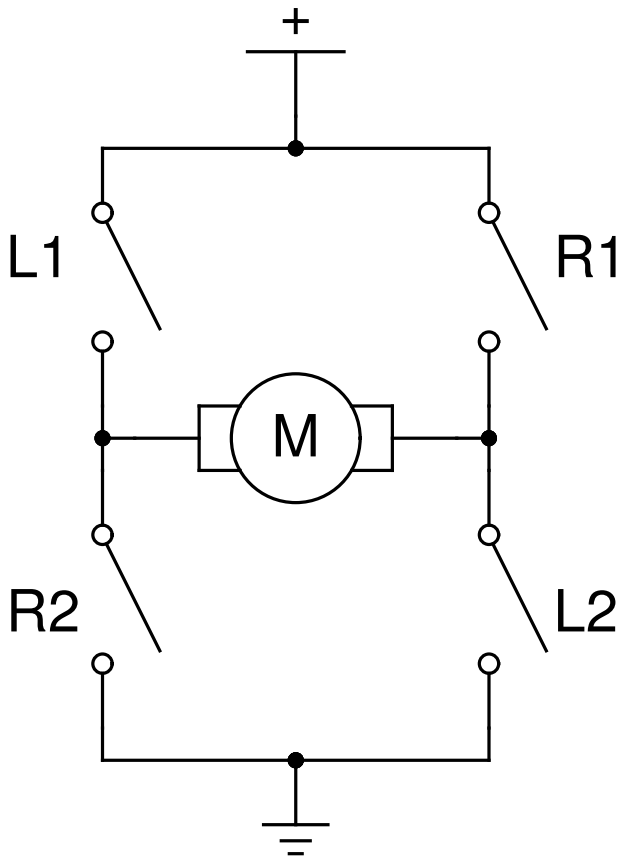
3.2 Sterowanie silnikiem DC

Obroty silnika DC często kontroluje się stosując modulację PWM. Natomiast kierunek obrotu silnika zmienia się poprzez zmianę polaryzacji napięcia podawanego na wyprowadzenia silnika. Do realizacji tego zadania służy mostek typu "H". Na rysunku nr 2 przedstawiono zasadę działania mostka H. Normalne stany to:

- wszystkie przełączniki wyłączone - silnik nie jest zasilany,
- L1 oraz L2 włączone, R1 oraz R2 wyłączone - silnik kręci się w lewo,

- R1 oraz R2 włączone, L1 oraz R2 wyłączone - silnik kręci się w prawo,
- L2 oraz R2 włączone - silnik hamuje.

Natomiast zabronione jest włączenie równocześnie L1 i R2 albo R1 i L2, gdyż powoduje to zwarcie zasilania do masy. W praktyce zamiast przełączników stosuje się tranzystory, natomiast cały układ może być zrealizowany w postaci układu scalonego tak jak np. sterownik DRV8835. Zapewnia on bezpieczne sterowanie kierunkiem i obrotami silnika DC.



Rysunek 2: Schemat przedstawiający zasadę działania mostka typu H.

4 Magistrala I2C

Magistrala I2C to szeregowa, dwukierunkowa magistrala służąca do przesyłania danych z różnych urządzeń elektrycznych np. czujników. Składa się z linii danych (SDA) oraz linii zegara taktującego transmisję (SCL). Każdy układ elektroniczny podłączony do magistrali I2C jest rozpoznawalny poprzez swój adres. Wyróżniamy układ master (nadrzędny), który inicjuje transmisję i generuje sygnał zegarowy oraz układy slave (podrzędne) np. podłączone czujniki. Każda transmisja poprzez magistralę I2C odbywa się według sekwencji: wysłanie bitu START → wysłanie adresu urządzenia 7bitowego oraz bitu wybierającego

tryb pracy: odczyt/zapis → wysłanie bitu STOP → wysłanie bitu START → wysłanie/odebranie informacji → wysłanie bitu STOP. Więcej informacji o zasadzie działania magistrali I2C będzie przekazanych na przedmiocie *Podstawy Systemów Mikroprocesorowych*.

Przydatne funkcje do obsługi I2C korzystając z biblioteki LINX to:

- **I2C Open** - otwiera kanał I2C.
- **I2C Write** - wysyła podane bajty danych do urządzenia zidentyfikowanego na podstawie podanego adresu. **Wysyłane dane należy przekazać jako tablicę, która jako pierwszy element tablicy zawiera wartość do wysłania w formie dziesiętnej lub szesnastkowej. Jako parametr EOF wybierz restart! Po wybraniu opcji restart można od razu odczytać wartość przy pomocy funkcji I2C read**.
- **I2C read** - funkcja odczytuje zadaną liczbę bajtów z magistrali I2C od urządzenia pod podanym adresem. Odczytana wartość jest w formie tablicy, gdzie poszczególne bajty po kolejne elementy tablicy.
- **I2C close** - funkcja zamykająca dany kanał magistrali I2C.

5 Czujnik ciśnienia LPS331

Czujnik ciśnienia LPS331 jest czujnikiem o dużej rozdzielczości (0.020 mbar), który działa w zakresie (260-1260) mbar. Obsługiwany jest przez magistralę I2C lub SPI.

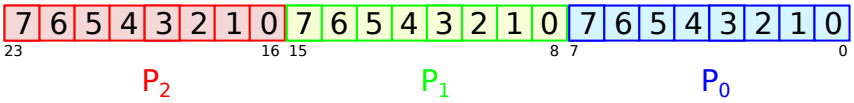
Schemat odczytu ciśnienia wygląda następująco:

1. Inicjalizacja:
   - Wysłanie zapytania o wartość rejestru 0x0F (dziesiętnie 15). I2C Slave adres czujnika ciśnienia jest równy 93.
   - Odczytanie 1 bajta danych.
   - Jeśli zwrócona wartość jest równa 187 to inicjalizacja przebiegła poprawnie.
2. Odczyt ciśnienia:
   - Wysłanie zapytania o wartość rejestru 0xA8 (dziesiętnie 168).
   - Odczytanie 3 bajtów danych (P_0, P_1, P_2).
   - Konwersja odczytanych wartości na ciśnienia w mbar/hPa według wzoru 1.

Konwersja odczytanych bajtów danych na ciśnienie odbywa się następująco:

$$p = \frac{p_{raw}}{4096} + 260 \tag{1}$$

gdzie p_{raw} został przedstawiony na rysunku nr 3.



Rysunek 3: 24-bitowa wartość p_{raw} składająca się z 3 odczytanych bajtów P2,P1,P0.

6 Zadania do wykonania

1. (1pkt) Wgraj firmware do Arduino i podłącz diodę LED do dowolnego pinu cyfrowego analogicznie jak na rysunku 4. Napisz program, który korzystając z kontrolki typu *boolean* będzie umożliwiał zapalenie i zgaszenie diody.

2. (2pkt) Podłącz silnik DC do Arduino zgodnie z rysunkiem nr 5 oraz napisz program, który umożliwi zmianę kierunku obrotu silnika (1pkt) oraz jego prędkość (1pkt).

3. (4pkt) Stwórz program do akwizycji temperatury. W tym celu:

   a) Podłącz czujnik temperatury LM35 do Arduino do wejścia analogowego zgodnie z rysunkiem nr 6. Napisz program, który odczyta napięcie z czujnika temperatury, wyświetli je w indykatorze i zamieni otrzymaną wartość na temperaturę w stopniach Celsiusza (10mV na 1^oC).(1pkt)

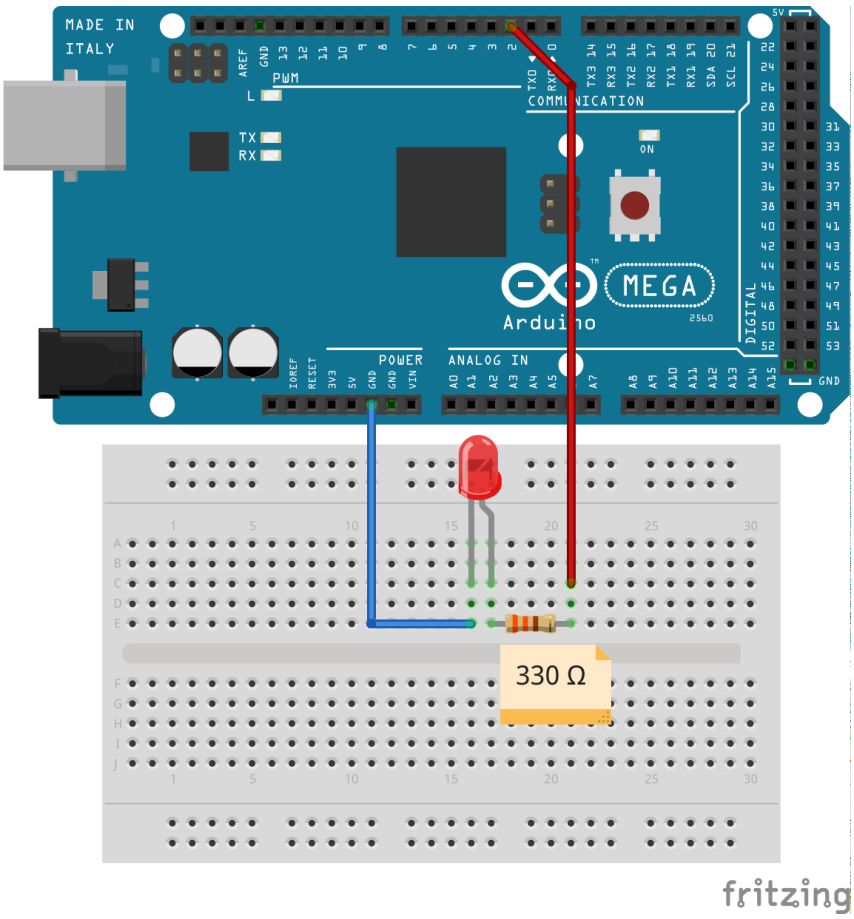
   b) Wyświetl otrzymaną wartość temperatury na wykresie typu *Waveform Chart* oraz stwórz kontrolkę, do której będzie można wpisać maksymalną temperaturę. Gdy temperatura z czujnika LM35 będzie większa od maksymalnej temperatury to wykres ma zmienić kolor na czerwony. W przeciwnym wypadku linia na wykresie ma mieć kolor zielony. (1pkt)

   c) Podłącz diodę LED do Arduino i zapal ją w momencie gdy temperatura z czujnika LM35 będzie większa od maksymalnej temperatury wpisanej do kontrolki. (0,5pkt)

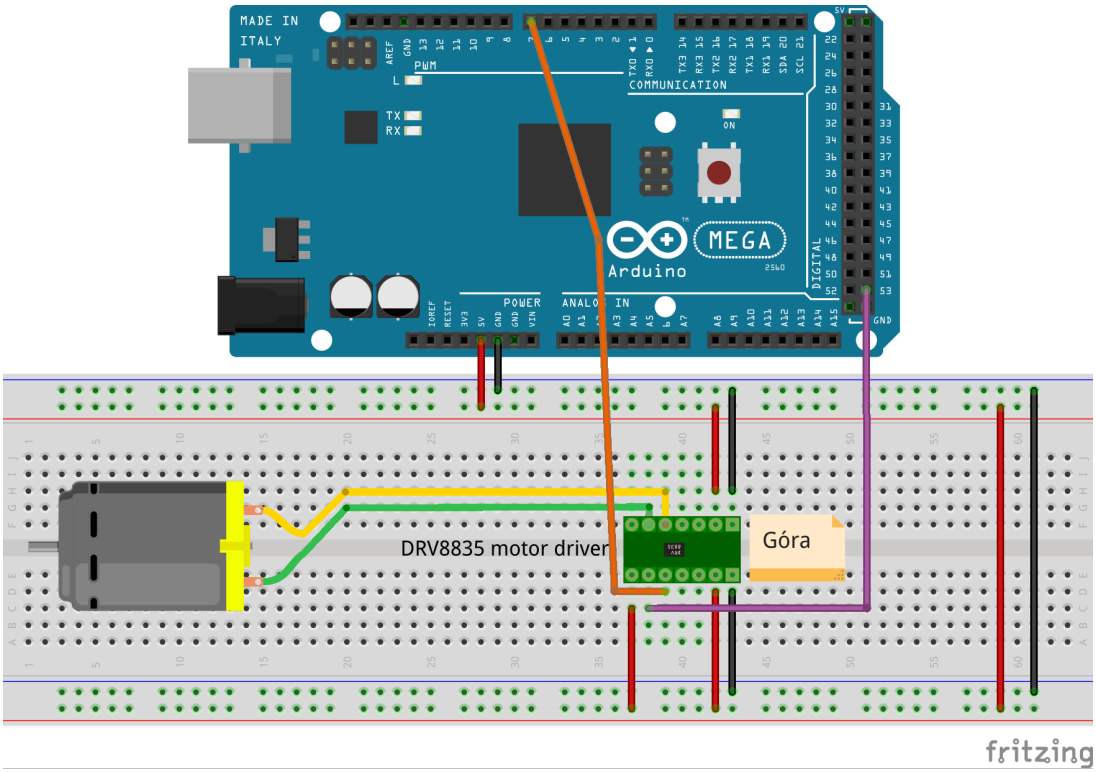
   d) Stwórz przycisk "Zapisz do pliku". Gdy użytkownik go wybierze to do pliku zostaną zapisane wszystkie wartości temperatury odczytane z czujnika LM35. (0,5pkt)

   e) Przy każdym uruchomieniu programu indykator z napięciem odczytanym z czujnika LM35 powinien wracać do wartości domyślnej. W tym celu wykorzystaj *Invoke Node Method*. Dodatkowo przy każdym uruchomieniu programu, wykres powinien być czyszczony. Wykorzystaj do tego *Property nodes*. (1pkt)

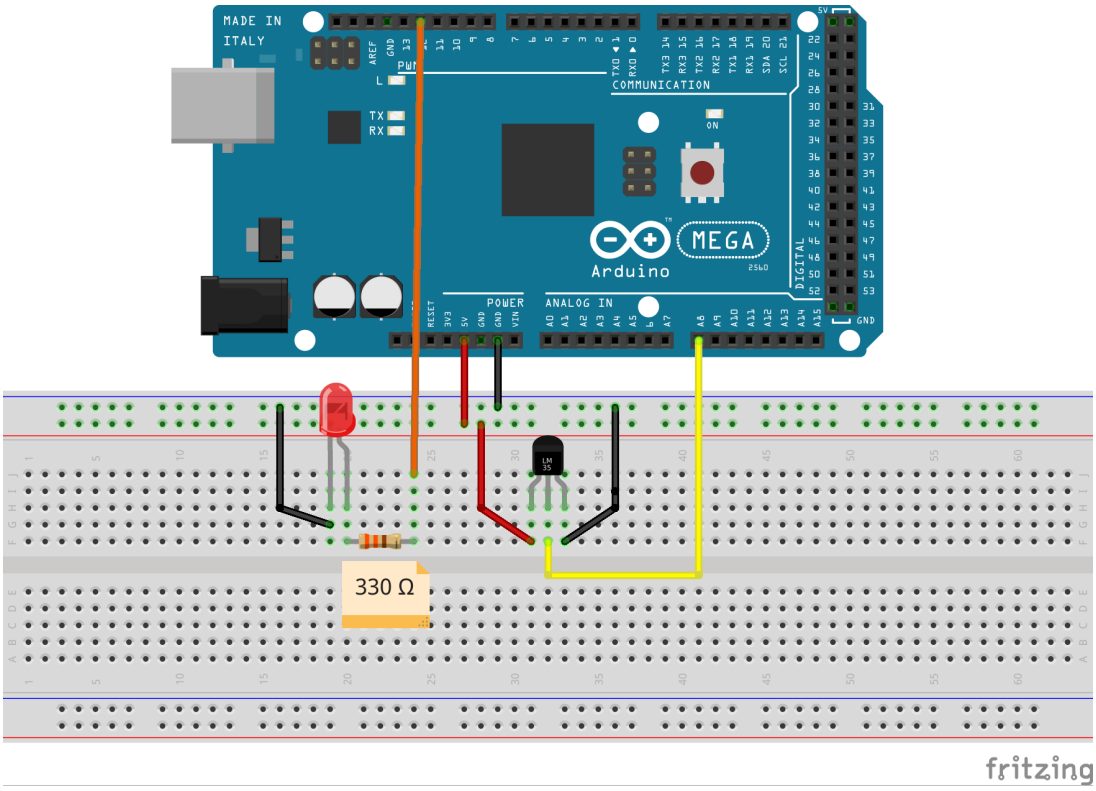
4. (3pkt) Podłącz *Nucleo Weather Shield* do Arduino i napisz program, który odczyta wartość ciśnienia w [hPa] z czujnika ciśnienia LPS331. Całe zadanie wykonaj przy użyciu maszyny stanów (nazwy stanów: "init", "readPressure").



Rysunek 4: Schemat podłączenia układu do zadania 1.



Rysunek 5: Schemat podłączenia układu do zadania 2.



Rysunek 6: Schemat podłączenia układu do zadania 3.