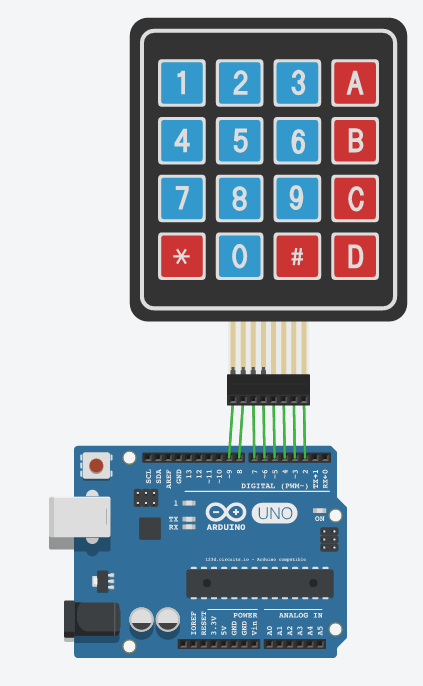
|  |  |  |
| --- | --- | --- |
|  | INSTYTUT FIZYKI ***WYDZIAŁ INŻYNIERII PROCESOWEJ, MATERIAŁOWEJ I FIZYKI STOSOWANEJ***  ***POLITECHNIKA CZĘSTOCHOWSKA*** | Instfiz |
| HARDWAROWA PRACOWNIA APARATURY MEDYCZNEJĆ W I C Z E N I E N R 10 Temat: Model „zamka szyfrowego” w oparciu o klawiaturę matrycową | | |

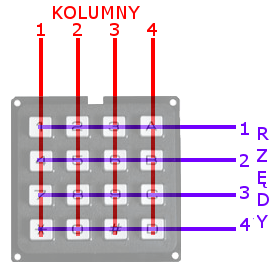
**1. Cel ćwiczenia:**

Celem tego ćwiczenia będzie podsumowanie dotychczasowo nabytej wiedzy i zapoznanie się z działaniem klawiatury matrycowej w połączeniu z innymi elementami elektronicznymi oraz stworzenie prototypu działającego zamka na szyfr, który porównuje wprowadzoną przez użytkownika kombinację z szyfrem zapisanym w programie i podejmuje powiązane z tym akcje.

**2. Wstęp teoretyczny.**

Każdy układ ARDUINO wyposażony został w przynajmniej jeden port szeregowy, który umożliwia proste przesyłanie danych do i ze sterownika. ARDUINO UNO posiada jeden sprzętowy port szeregowy podłączony do portu USB. Z pozycji ArduinoIDE możliwe jest uruchomienie terminala, za pomocą którego można odczytać dane wysyłane poprzez port szeregowy ARDUINO, jak również wysłać informacje do układu.  
Podstawowymi komendami wykorzystywanymi do współpracy z terminalem są: serial.begin(X) – służy do ustawienia połączenia szeregowego z parametrem prędkości X (zazwyczaj przyjmującym wartość 9600) oraz serial.print, służący do wyświetlania w terminalu tekstu.  
Terminal pozwala na „porozumiewanie się” z Arduino, czyli uzyskiwania wyników naszych operacji.  
  
Dodatkowym elementem, który nie został jeszcze przedstawiony, a będzie wprowadzony w dzisiejszym ćwiczeniu, jest klawiatura matrycowa. Zasada jej działania jest prosta – jej przyciski są połączone w rzędy i kolumny, co pozwala na odczytanie aktualnie wciśniętego przycisku.





Rys. 11 Schemat połączeń wewnątrz klawiatury

Rys. 2 Schemat podłączenia klawiatury do Arduino

**3. Przebieg ćwiczenia.**

1. Pierwszym układem, który powinien zostać wykonany w związku z tym ćwiczeniem, jest układ składający się z Arduino oraz klawiatuy membranowej. Pozwoli on na zrozumienie podstaw działania tego układu. Należy złożyć układ zgodnie ze schematem zamieszczonym w Rysunku 2, po czym skompilować w ArduinoIDE następujący kod:

1. #include <Keypad.h>                  //dodaj bibliotekę obsługującą klawiaturę
3. const byte wiersze = 4;                      //ilość wierszy klawiatury
4. const byte kolumny = 4;                      //ilość kolumn klawiatury

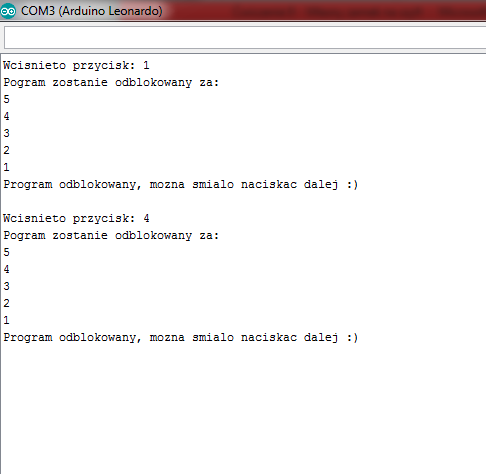
7. char klawiatura[wiersze][kolumny] = {                 //ustaw rozkład klawiszy
8. {'1','2','3','A'},
9. {'4','5','6','B'},
10. {'7','8','9','C'},
11. {'\*','0','#','D'}
12. };
14. byte wierszePiny[wiersze] = {9,8,7,6};            //wiersze podłączone do pinów (numeracja od lewej strony do prawej)
15. byte kolumnyPiny[kolumny] = {5,4,3,2};            //kolumny podłączone do pinów (numeracja od lewej strony do prawej)

18. //utwórz klawiaturę na postawienie ustawienia klawiszy, pinów Arduino oraz ilości kolumn i wierszy
19. Keypad klaw1 = Keypad( makeKeymap(klawiatura), wierszePiny, kolumnyPiny, wiersze, kolumny );
21. void setup()
22. {
23. Serial.begin(9600);                      //rozpocznij transmisję UART
24. }
26. void loop()
27. {
28. char przycisk = klaw1.getKey();              //zmiennej char przypisujemy zwrócony znak z klawiatury
30. if (przycisk)                                 //jeżeli wykryto wciśnięty przycisk to:
31. {
32. Serial.print("Wcisnieto przycisk: "); //wyświetl wciśnięto przycisk:
33. Serial.println(przycisk);                  // wyświetl klawisz
34. delay(50);
35. }}

W kodzie zostały pozostawione komentarze pozwalające na zapoznanie się ze schematem działania kodu. Można się z nim zapoznać, uruchamiając w ArduinoIDE Monitor Szeregowy i nasłuchując efektu wciskania poszczególnych klawiszy. Polecane jest także zapoznanie się z dokumentacją biblioteki Keypad.h dla lepszego zrozumienia dalszej części ćwiczenia. Wymagane jest również dodanie osobnej biblioteki, którą wgrywamy poprzez opcję *Szkic->Dodaj bibliotekę->Dodaj bibliotekę Zip* i wskazaniu biblioteki *Keypad.h.zip* z folderu macierzystego.

Po sprawdzeniu praktycznego działania kodu, wprowadzić do projektu modyfikacje, których efektem będą następujące scenariusze:

1. Po wciśnięciu dowolnego przycisku zapalona zostanie zewnętrzna dioda, podłączona pod pin 12
2. Po naciśnięciu przycisku wprowadź okres oczekiwania na następną możliwość komunikacji z Arduino, odliczając w Monitorze Szeregowym czas 5 sekund. Wynik wyjściowy powinien wyglądać następująco:



2. Posiadając już podstawową wiedzę na temat działania klawiatury membranowej, następnym krokiem będzie stworzenie z jej użyciem czegoś bardziej praktycznego – zamka szyfrowego. Podstawowy schemat podłączenia klawiatury do Arduino dla uproszczenia pozostaje ten sam, co na Rysunku 2; podłączenie dodatkowych elementy, które będą potrzebne, zostało omówione w poprzednich ćwiczeniach. W środowisku ArduinoIDE należy wprowadzić następujący kod programu:

1. #include <Keypad.h>
3. const byte wiersze= 4; //ilość wierszy
4. const byte kolumny= 4; //ilość kolumn
6. char mapaklawiszy[wiersze][kolumny]=
7. {
8. {'1', '2', '3', 'A'},
9. {'4', '5', '6', 'B'},
10. {'7', '8', '9', 'C'},
11. {'\*', '0', '#', 'D'}
12. };
14. byte wiersze\_piny[wiersze] = {9,8,7,6}; // wiersze 0-3
15. byte kolumny\_piny[kolumny]= {5,4,3,2}; // kolumny 0-3
16. volatile int l=0; // zmienna pomocnicza, licznik znaków
17. int mykey[5]; // taklica wciśnietych znaków
18. Keypad myKeypad= Keypad(makeKeymap(mapaklawiszy), wiersze\_piny, kolumny\_piny,
19. wiersze, kolumny);
21. void warning\_bad()
22. {
23. Serial.println(" ---> Kod NIEPRAWIDLOWY");
24. l=0;
25. }
27. void setup()
28. {
29. pinMode(10,OUTPUT);
30. pinMode(11,OUTPUT);
31. digitalWrite(10,0);
32. digitalWrite(11,0);
33. Serial.begin(9600);
34. Serial.flush();
35. }
36. //--------------------------------------------------------
37. void loop()
38. {
39. Serial.print("Wprowadz KOD dostepu: ");
40. while(l<5)
41. {
42. char keypressed = myKeypad.getKey();
43. if (keypressed != NO\_KEY)
44. {
45. mykey[l]=keypressed;l++;Serial.print(keypressed);
46. }
47. }
48. if( mykey[0]=='1' && mykey[1]=='1' && mykey[2]=='1'
49. && mykey[3]=='1' && mykey[4]=='1' )
50. {
51. Serial.println(" OK !!!");
52. digitalWrite(11,HIGH);delay(3000);digitalWrite(11,LOW);
53. l=0;
54. }
55. else
56. {warning\_bad();delay(500);}}

Po sprawdzeniu praktycznego działania kodu, wykonać następujące polecenia

1. Opisać praktyczne działanie zamieszczonego kodu, wykorzystane w programie instrukcje warunkowe oraz działanie funkcji *void warning\_bad()*
2. Odpowiedzieć na poniższe pytania:

- jaką funkcję pełni w kodzie *int mykey[5];* oraz opisać zastosowanie tego typu zabiegu  
- jaką funkcję pełni w programie *zmienna l* oraz jaka jest rola dyrektywy volatile  
- jaki jest pin startowy w programie

- jakie jest praktyczne zastosowanie pisania własnych funkcji poza obszarem funkcji void loop()

- czym jest inkrementacja i gdzie oraz w jakim celu została użyta w tym programie

- wymienić min 3 możliwe do zrealizowania z wykorzystaniem klawiatury membranowej projekty.

1. Wprowadzić do programu następujące modyfikacje:  
   - przepisać kod programu tak, by nie definiować w nim funkcji *void warning\_bad()*- przepisać kod programu tak, by funkcja void loop() zawierała tylko jedną linię kodu

- dodać do projektu czerwoną i zieloną diodę LED by uzyskać następujący efekt: podczas błędnego wpisania PIN dioda czerwona będzie migotała w dowolnym tempie przez 3 sekundy, natomiast po podaniu prawidłowego PIN zielona dioda zapali się na 5 sekund

- zmienić pin startowy na 3424A

Sprawozdanie powinno zawierać każdy zmodyfikowany kod programu z wymaganymi zmianami. Wszelkie kody oraz biblioteki załączone do ćwiczenia dostępne są w katalogu macierzystym.