Uniwersytet Wrocławski
Wydział Fizyki i Astronomii
Instytut Fizyki Doświadczalnej
pl. M. Borna 9
50-204 Wrocław
tel. +48 71 375 93 02, +48 71 328 73 65
fax +48 71 328 73 65
e-mail: sekr@ifd.uni.wroc.pl
www.ifd.uni.wroc.pl

(Opracował: F. Gołek)

Ćwiczenie nr 43 Mikrokontroler
Mikrokontroler

Cele:

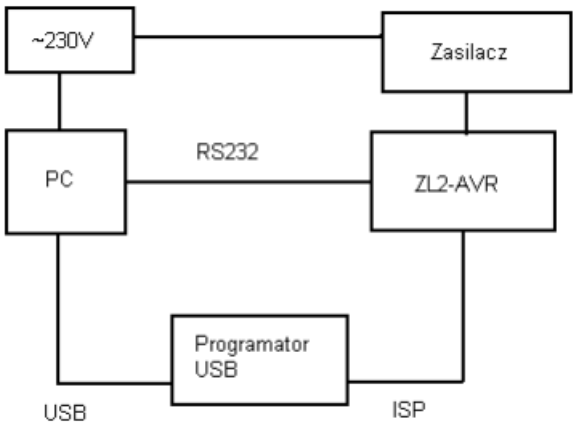
Poznanie programowania i zastosowań układów sterujących (mikrokontrolerów). Poznanie sensora przyspieszenia oraz jego współpracy z mikrokontrolerem.
Przepisanie programów zapisanych w dokumentacji, poprawnej ich kompilacji i wgraniu do układu mikrosterownika. Z uzyskanych wyników należy odpowiednio obliczyć kąt nachylenia powierzonych klinów oraz wyznaczenia masy kuleczki opuszczanej z danego klina.

1 Wykonanie ćwiczenia

1.1 Sprawdzić schemat połączeń początkowych (rys.1)
1.2 Wprowadzić programy zamieszczone w dokumentacji, dokonać kompilacji.
1.3 Przesłać wybrany program do mikrokontrolera.
1.4 Połączyć obwód elektryczny zgodnie z zamieszczonym schematem (do wybranego programu).
1.5 Przy realizacji programu nr 3 należy uzyskane wyniki (wartości przyspieszeń) przeliczyć na kąt nachylenia klina.
1.6 Przy realizacji programu nr 4 należy (po zapoznaniu się z programem) puścić powierzoną kulkę z klina na zabudowany w metalowej odbudowę moduł akcelerometru. Eksperyment powtórzyć wielokrotnie, naszkicować rozkład wyników i ocenić odtwarzalność eksperymentu.

2 Wymagane zagadnienia

2.1 Budowa i działanie akcelerometrów elektronicznych.
2.2 Budowa i działanie mikrokontrolerów.
2.3 Przetworniki A/D i D/A.
2.4 Przykładowe sensory (czujniki).



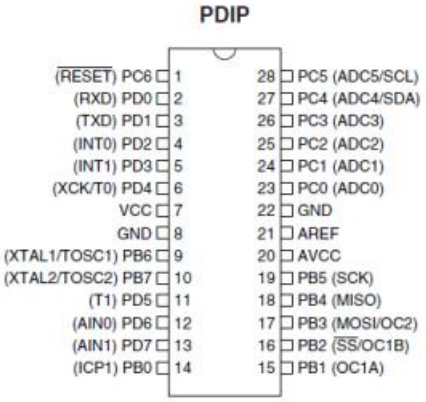
Rysunek 1. Schemat połączeń modułu ZL2AVR z komputerem PC.

1.Wstęp Użyty w zastawie ćwiczeniowym układ scalony ATmega8 należy do rodziny wydajnych mikrsterowników wyprodukowanych przez firmę Atmel. W przeciwieństwie do

układów produkowanych w latach wcześniejszych (rodziny 8051, 6800), nowe układy AVR zostały zaprojektowane pod kątem użycia języków wysokiego poziomu - głownie C. We wczesnych konstrukcjach, stosowanie języków programowania innych niż asembler wiązało się z dużym spadkiem mocy obliczeniowej rdzenia oraz narzucało konieczność posiadania przez procesor dużej ilości pamięci dla programu i danych, co przyczyniało się do podnoszeniem kosztów. Jednakże pisanie bardziej rozbudowanych programów w asemblerze skutkowało wzrostem nakładów pracy potrzebnej na wykonania projektu. Wychodząc naprzeciw oczekiwaniom rynku, firma Atmel w 1993 r. wprowadziła do sprzedaży 8-bitowe procesory RISC wyposażonych w stosukowo dużo pamięci. Rodzinę procesorów AVR możemy podzielić na dwie grupy:

- układy ATiny- układy z mniejszą ilością wbudowanych urządzeń peryferyjnych, zamknięte w małych obudowach - 8pinowych
-układy ATmega- układy z dużą liczbą portów zewnętrznych i układów peryferyjnych.

Duże wyspecjalizowanie poszczególnych modeli pozwala ograniczyć do minimum obecność dodatkowych elementów i optymalnie dopasować układ do danego projektu. W przypadku ATmega8 mamy do czynienia z mikrosterownikiem wyposażonym w 8kb pamięci typu flash na program oraz 1kb pamięci SRAM pełniącej funkcje pamięci operacyjnej. Układ ten dysponuje ponadto 23 liniami I/O ogólnego użytku, zgrupowanymi w 3 porty 8bitowe. Jak łatwo możemy zauważyć na rysunku nr.1. niektóre wyprowadzenia posiadają alternatywne funkcje. Obecność tych dodatkowych funkcji jest następstwem posiadania przez mikrosterownik wbudowanych dodatkowych układów peryferyjnych. Należy zauważyć, że niektóre z wyprowadzeń posiadają więcej niż dwie funkcje na jednej linii.



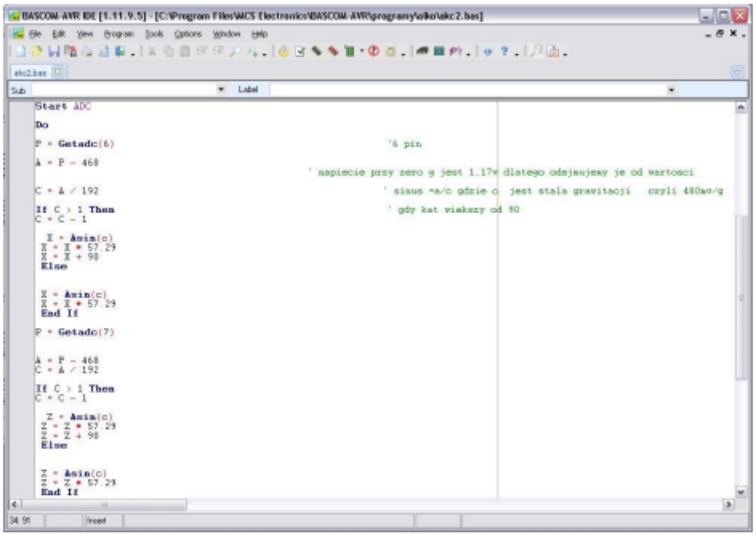
Rysunek 2. Wyprowadzenia mikrosterownika Atmega8.

Alternatywnymi portu B poza zwykłymi wejściami I/O są: obsługa transmisji SPI, za pomocą której jest programowany mikrosterownik (SCK,MISO,MOSI) lub wejścia dla układów liczników PWM (OC1B,OC1A,TOSCx). W przypadku portu C dostępnymi dodatkowymi funkcjami są wejścia na przetwornik cyfrowo analogowy (ADCx) lub obsługa komunikacji transmisji zgodnej z I^2C (SCL,SDA). Obsługa komunikacji w standardzie USART(RS232) zrealizowana może być poprzez sprzętowe wyprowadzenia znajdujące się w porcie D. Port ten oferuje nam również wejścia dla „timerów" jak i zewnętrznych źródeł przerwań(INTx).

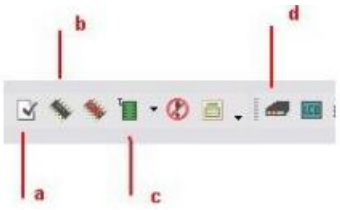
2.Krótki opis środowiska Bascom

Jak wcześniej wspomniano, procesory AVR powstały z myślą programowania ich w językach wysokiego poziomu. Jednym z takich języków jest odmiana języka BASIC dla AVRBASCOM. Program ten dostępny jest w na stronie producenta (http://www.mcselec.com/) w wersji DEMO z ograniczeniem wielkości kodu wynikowego. Środowisko IDE[1] języka BASCO oferuje nam wszystkie potrzebne do programowania

funkcje w jednym programie. Poniżej zostanie przedstawione tylko kilka z nich, potrzebnych do zrealizowania ćwiczenia. Poza oczywistymi funkcjami takimi jak odczyt/zapis plików oraz edycja kodu programu; w środowisku tym możemy odnaleźć program emulacji protokołu terminala VT52 (do komunikacji przez RS)[2] oraz narzędzia wgrywania programu do mikrostrownika.



Rys. 3 - Okno edytora kodu środowiska BASCOM.



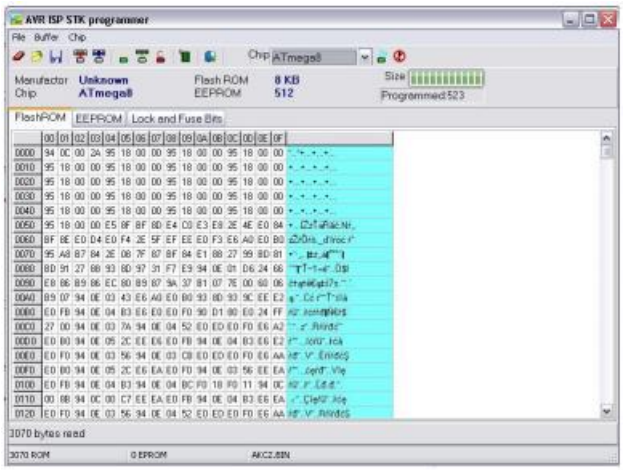
Rys. 4 Pasek zadań a - sprawdzenie poprawności kodu, b - kompilacja kodu, c - uruchomienie modułu programatora, d - emulator VT52 do komunikacji przez RS.

Na rysunku 3 zaznaczono 4 najważniejsze dla tego ćwiczenia ikony. Pierwsza z ikon służy do uruchomienia procesu sprawdzania poprawności kodu pod względem składni, struktury oraz użytych typów zmiennych. W celu ułatwienia pracy programisty, środowisko edycyjne rozpoznaje polecenia języka BASCOM i słowa kluczowe tych poleceń wyświetla w kolorze niebieskim. W dolnej części ekranu edytora, w razie problemów związanych z napisanym kodem, mogą pojawiać się informacje wskazujące rodzaj i miejsce wystąpienia błędu. W przypadku braku zgłoszeń błędów można przystąpić do kolejnej ikony. Pod ikoną oznaczaną litera b kryje się kompilator. Kompilator przed wygenerowaniem kodu maszynowego dla procesora AVR niezależnie sprawdzi napisany program pod względem składni. Dopiero po pomyślnym przejściu procesu kompilacji można przystąpić do użycia ikony oznaczonej literą c - narzędzia programatora. W przeciwnym wypadku bufor aplikacji kompilatora będzie pusty i konieczne będzie wskazanie pliku z poprawnie skompilowanym kodem.

---

[1] Integrated Development Environment

[2] Standard terminala 80 kolumn 24 rzędy.

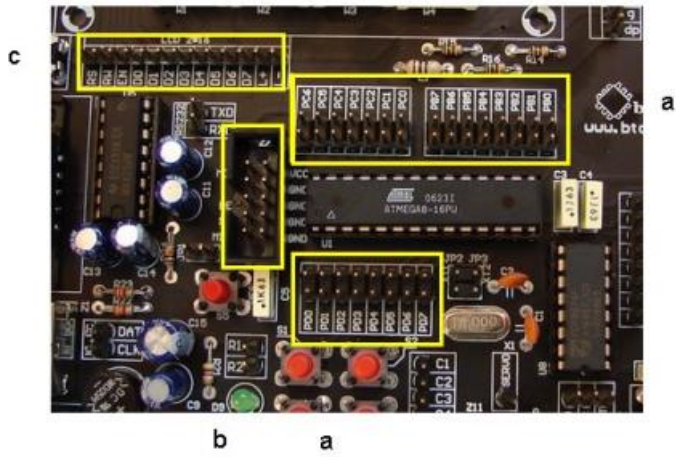


Rysunek 5. Okno narzędzi programatora.

Jeżeli nasz układ został poprawnie podłączony do programatora, w pasku opcji obok napisu chip powinien się znaleźć rodzaj rozpoznanego układu AVR. W przeciwnym wypadku pojawienie się okna programatora poprzedzone będzie informacja o braku możliwości odczytania identyfikatora układu. Programator oprócz podstawowych funkcji odczytywania, kasowania oraz wgrywania programu do sterownika, oferuje opcje związane z programowaniem dodatkowej pamięci EEPROM oraz możliwości ustawiania Fuse Bit. Ustawianie fuse bits pozwala nam na wybór rodzaju jak i częstotliwości zegara procesora atmega, zabezpieczenia programu przed odczytem oraz w niektórych układach, wykorzystać linie reset jako port we/wy. W powyższym ćwiczeniu niewskazana jest jakakolwiek manipulacja fuse bits - nie umiejętne ustawienie bitów powoduje zablokowanie sterownika.

3.Zestaw ZL2AVR[3]

Zestawy uruchomieniowe służą przede wszystkim nauce programowania mikrosterowników jak i przetestowaniu własnych układów zanim zostaną zaprojektowane płytki drukowane. Zestaw ZL2AVR wyposażony jest w szereg dodatkowych układów takich jak konwerter magistrali IC, wyświetlacz LCD, LED czy odbiornik IR. Wszystkie te układy można bez problemu połączyć z układem Atmega8 bez konieczności lutowania wykorzystując tylko przewody. Na rysunku 6 literą a zaznaczono porty wyjściowe mikrosterownika do których podłącza się urządzenia peryferyjne.



Rys. 7 Oznaczenie wyprowadzeń. Litera a oznacza (zakreślone na czerwono) - porty Atmega8, b - port programatora, c - port wyświetlacza LCD.

Literą b oznaczono gniazdo programatora podłączane do komputera PC. Na płytce można odnaleźć również gniazdo zasilania i masy oraz gniazdo wyświetlacza LCD (c).

4.Programowanie w BASCOM

Ćwiczenie nie ma na celu nauki programowania procesorów AVR a jedynie przybliża ich potencjalne możliwości. Do poprawnego wykonania ćwiczenia wystarczą proste modyfikacje programów przykładowych które są przedstawione poniżej. Programy należy poprawnie przepisać do komputera oraz dokonać kompilacji kodu (F7) i uruchomienia ich na sterowniku.(znak ' oznacza początek komentarza)

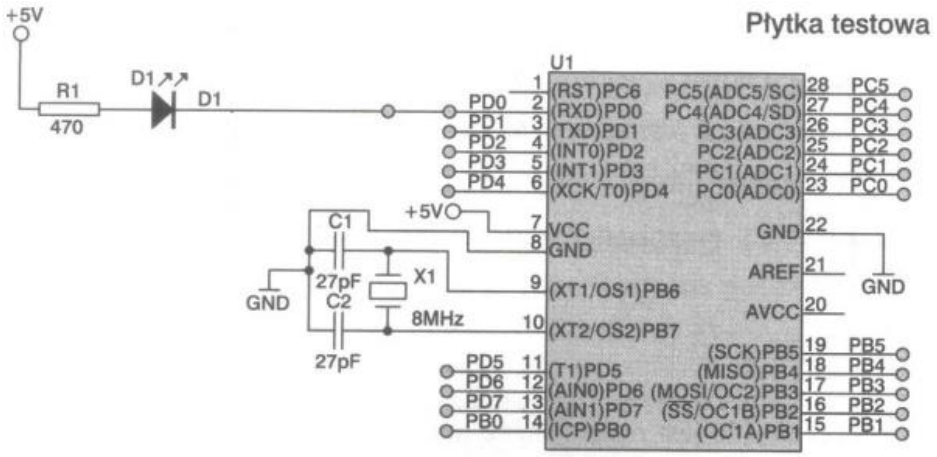
4.1 Przykład pierwszy

```
$regfile = "m8def.dat".dat "   ' wskazanie procesora AVR, na który piszemy program
$crystal = 8000000             ' określenie częstotliwości pracy procesora

Config Portd = Output          'ustawienie portu D jako wyjście.
```

[3] Szczegółowy opis zestawu można znaleźć na stronie producenta pod adresem www.kamami.pl/dl/zl2avr.pdf.

```
Do                 ' początek pętli
Toggle portd.0     'zmiana stanu po na przeciwny (negacja logiczna)
Wait 1             ' odczekanie 1s LOOP ' etykieta końca pętli do-loop
End ' identyfikator zakończenia programu.
```

Do poprawnego działania programu należy podłączyć pin o numerze 0 z portu D z wyprowadzeniem na diodę LED d1 (rys.8). Jeśli wszystko wpisaliśmy poprawnie, to po kompilacji i wgraniu programu do mikro sterownika (F4 a następnie ikona Auto program chip) powinniśmy zaobserwować mruganie diody. Zmieniając wielkość parametru wait zmieniamy czas pomiędzy kolejnymi zapaleniami diody (zademonstrować to prowadzącemu).



Rys. 8 Schemat do pierwszego programu [1.158].

4.2 Przykład 2

W tym przykładzie wykorzystamy wbudowany na płytce rozwojowej wyświetlacz LCD służący do wykonywania komunikacji tekstowej ze światem zewnętrznym. Przed przystąpieniem do realizacji tego przykładu należy rozłączyć uprzednio wykonany układ. Dla poprawnego działania wyświetlacza należy połączyć kabelkami piny z portu D sterownika z

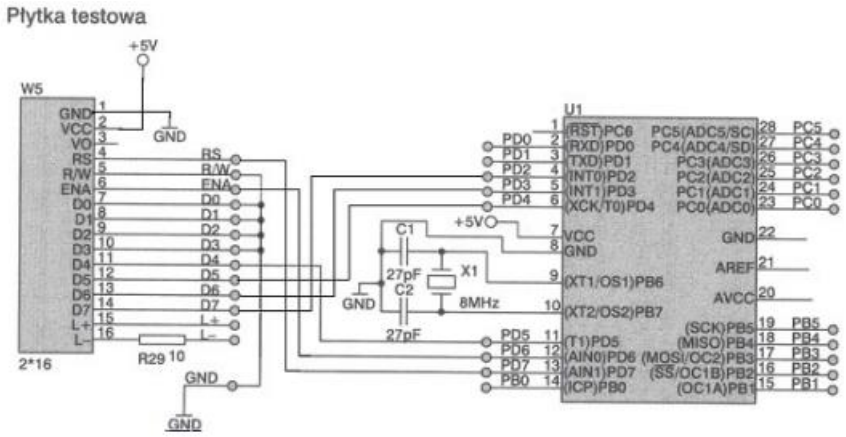
portem wyświetlacza w sposób opisany w instrukcji cofig lcdpin (patrz również rys. 8) : Instrukcja „Config Lcdpin” wygląda następująco:
Config Lcdpin = Pin, Db4 = Portd.5, Db5 = Portd.4, Db6 = Portd.3, Db7 = Portd.2, E = Portd.6, Rs = Portd.7.

Czyli, zgodnie z powyższą instrukcją „Db4 = Portd.5” linię 5 portu D (pin oznaczony PD5) podłączyć należy do linii określającej bit 4 na wyświetlaczu (pin z oznaczeniem D4), linię 4 portu D z linią D5 na wejściu wyświetlacza. Linie 3 z portu D do linii 6 wyświetlacza. Linie 2 portu D do linii oznaczonej cyfrą 7 na wyświetlaczu. Linie 6 z portu D łączymy z linią oznaczoną symbolem E. Linię 7 z portu D łączymy z pinem RS na porcie wyświetlacz. Dodatkowo musimy pin RW na wyświetlaczu podłączyć do punktu masy (jeden z pinów oznaczonych napisem GND). Wyżej opisane linie stanowią 4biotwy kanał danych oraz 2 sygnały sterujące potrzebne do komunikacji z wyświetlaczem (rys.9).

Jeżeli poprawnie wszystko zostanie wykonane po wpisaniu poniższego kodu powinniśmy zobaczyć na wyświetlaczu tekst znajdujący się miedzy znakami " " (cudzysłowami) polecenia LCD.

```
$regfile = "m8def.dat"          'deklaracja typu procesora
$crystal = 8000000              ' deklaracja częstotliwości zegara procesora.

Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portd.5 , Db5 = Portd.4 , Db6 = Portd.3 , Db7 = Portd.2 , E = Portd.6 , Rs = Portd.7 ' konfiguracja wyświetlacza na porcie d.

cls

 Do Lcd " tekst"          ‘wypisanie tekstu na LCD
Wait 1                    ‘pausa na 1s
Lowerline                 ‘przejście do dolnej linii wyświetlacza
LCD „tekst”
Wait 3
Cls                       ‘czyszczenie zawartości ekranu
wait 1
Loop

End
```

Jak łatwo zauważyć programy w Bascomie składają się z kilku głównych elementów. Pierwszym z nich jest deklaracja procesora dla którego jest pisany kod ($regfile = "m8def.dat"), w naszym przypadku jest to Atmega8. Następnym ważnym elementem jest określenie częstotliwości pracy układu (konieczne dla komunikacji zewnętrznej oraz gdy używamy polecenia oczekiwania). Ostatnim elektem wymaganym przez kompilator jest polecanie końca programu *End*.



Rys. 9 Schemat do programu drugiego.

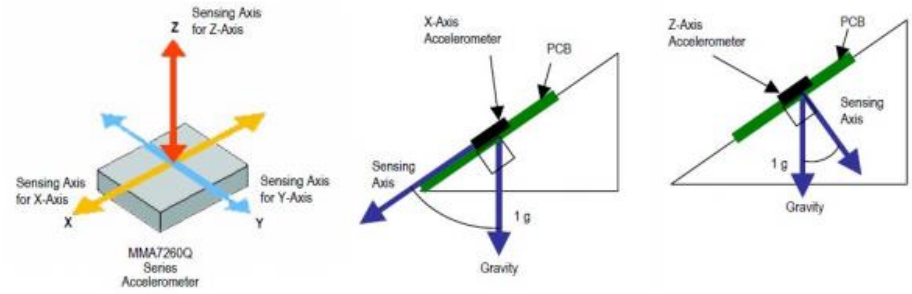
4.3 Program akcelerometru

Do wykona tego ćwiczenia niezbędne będzie podłączenie dodatkowo do układu modułu akceleormetra XYZ. Wyjście sygnału z powyższego moduł jest wyjściem analogowym, to znaczy że w celu wykorzystania układu musimy skorzystać z wbudowanego do atmegi przetwornika cyfrowo analogowego. Jak można zobaczyć na rysunku nr1. alternatywne wejścia przetwornika znajdują się tylko na porcie C, dlatego jesteśmy zmuszeni podpiąć akcelerometr do powyższego portu. układ należy podłączyć zgodnie z rysunkiem nr10, czyli sygnały akcelerometru pod 3 pierwsze linie portu C (od PC0 do PC2). Zasilanie do modułu można pobrać z płyty rozwojowej. Powyższy układ jest 3 osiowym akcelerometrem o napięciowym sygnale proporcjonalnym do wartości przyspieszenia: 0.8V na 1G. W stanie spoczynku na wyjściu układu zaobserwować można napięcie rzędu 1,6, które w zależności od kierunku przyspieszenia (ciążenia) będzie zmieniało się od 1,6 do 0 lub do 3V. Wielkości te zależą bezpośrednio od nachylenia układu, co za tym idzie rzutu wektora siły grawitacji na daną oś sensora (rys.9). Widzimy zatem że potrzebny zakres pomiarowy przetwornika dla przyspieszenia ziemskiego wystarczy 2,5V. Dlatego też wewnętrzny przetwornik zostaje ustawiony w tryb pracy z zakresem 2,56V. Ponieważ przetwornik ma rozdzielczość 10 bitową uzyskana waga jednego bitu kwantyzacji to 2,56V/1024 = 0,0025 V/kwant. Wartość ta jest potrzebna do poprawnego przeliczenia danych z przetwornika na napięcie a w drugiej kolejności na przyspieszenie ziemskie.

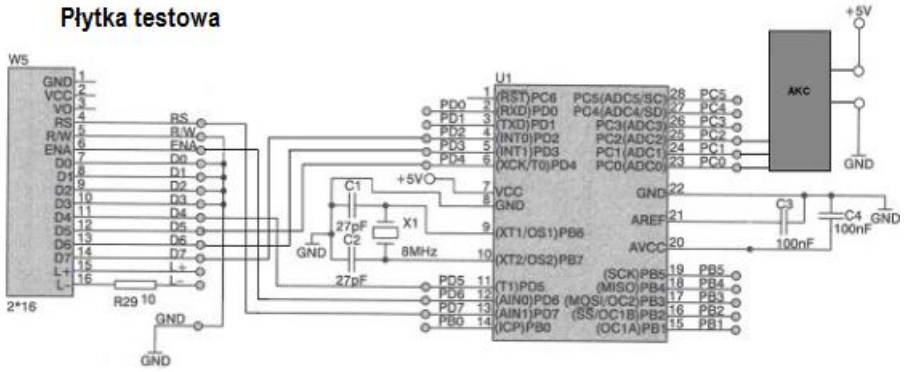
```
$regfile = "m8def.dat"          'deklaracja typu procesora
$crystal = 8000000              'deklaracja częstotliwości zegara procesora
Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portd.3 , Db5 = Portd.2 , Db6 = Portd.1 , Db7 = Portd.0 , E =
Portd.4 , Rs = Portd.5          'konfiguracja wyświetlacza

Config Adc = Single , Prescaler = Auto , Reference = Internal
                                'konfiguracja AD, stalla kwantowania przy ref2,56 jest 0,0025.
```

```
Dim P2 As Word
Dim A As Single
Dim B As Single
Dim C As Single
Dim A1 As Single
Dim B1 As Single
Dim C1 As Single

Dim Z As String * 3
Dim W As String * 3

Start Adc                          ' aktywacja przetwornika

Do                                 'pętla główna programu

P = Getadc(0)                      '0 pin portu c pobranie danych z przetwornika
P1 = Getadc(1)                      '1 pin portu c
P2 = Getadc(2)                      '2 pin portu c

                                   'przeliczanie wyniku na volty
A = P * 0.0025                 'wynik z przetwornika AD razy stała kwantowania  0.0025(10bitowy przetwornik)
B = P1 * 0.0025
C = P2 * 0.0025

' przeliczenie napięcia na g

A = A - 1.65                       ' odejmowanie pozycji zerowej  (1,65 V bez przyspieszeń i grawitacji).
B = B - 1.65
C = C - 1.65

A1 = A / .8                         'stała czujnika to 800 mV/g
B1 = B / .8
C1 = C / .8

W = Fusing(a1 , "#.&&")                 ' formatowanie danych wyjściowych do postaci x,yy
Lcd "b " ; Z ; "g "

            ' przejście do dolnej linii wyświetlacza
Lowerline
 Z = Fusing(c1 , "#.&&")

Lcd "c " ; Z ; "g"

Waitms 500                         ' odczekanie 500 ms

 Loop
End
```

Wtyczka z sygnałami akcelerometru nie posiada oznaczeń numerów kanałów, dlatego też umownie oznaczamy je literami A,B,C. W sposób eksperymentalny należy przypisać kanałom odpowiadające im osie układu współrzędnych. W przypadku bascom, podobnie jak w innych językach występuje konieczność definiowania używanych zmiennych. Do dyspozycji mamy kilka możliwych zmiennych: Bit - zmienna 1bit Byte – zmienna 8bit Single - zmienna zmienno-przecinkowa Word - zmienna 16bit String - zmienna tekstowa Niestety użyty język wprowadza ograniczenia co do wpisywania operacji matematycznych - możliwe jest zapisanie tylko jednej operacji w jednej linijce - dlatego też wszystkie wzory zmuszeni jesteśmy rozbić na pojedyncze równania. Powyższy program nie pokazuje wyniku przechylenia w stopniach, a jedynie wartości rzutu przyspieszenia ziemskiego na jedną z osi układu (rys. 10). W celu uzyskania wyniku w stopniach można dopisać fragment kodu zamieniającego odpowiednie składowe na kąt (odpowiednia funkcja arcsin występuje w bascom) lub samemu przeliczyć uzyskane wyniki.

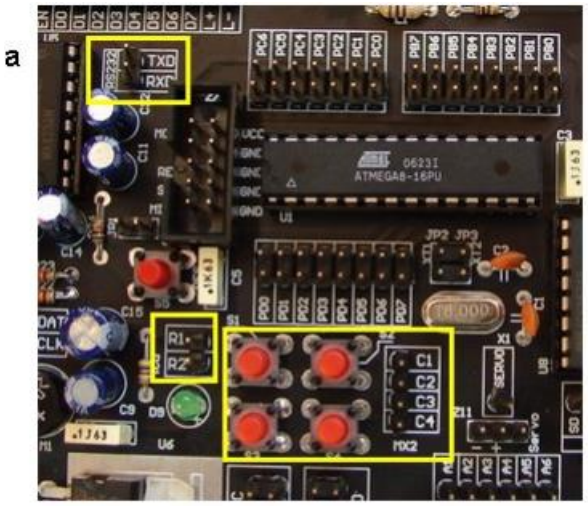


Rys. 10 Rzut wektora grawitacji na oś układu [2,2]



Rysunek 11 Sposób podłączenia układu z sensorem przyspieszenia

4.4 Program pomiaru udaru



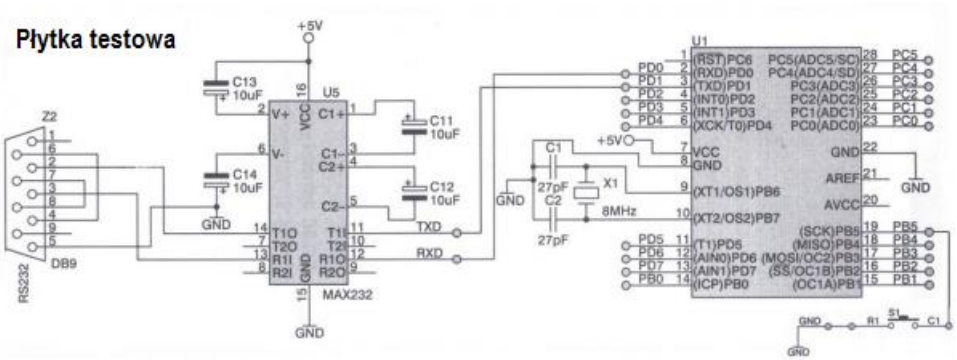
Rys. 12 Płytka rozwojowa ZL2AVR. a-piny do układu max232, b- przyciski wraz z pinami wyjściowymi, c- piny do podłączenia masy dla przycisków.

W przypadku tego programu (nie rozłączając połączeń wyświetlacza) użyjemy dodatkowego modułu mikrosterownika, a mianowicie układu USART. Układ ten jest modułem komunikacyjnym, który pozwala na transmisje danych w kodzie zgodnym z RS232. Niestety poziomy napięć wychodzących z mikrosterownika nie pozwalają na bezpośrednią komunikację, dlatego sygnał zanim zostanie podany do komputera musi przejeść przez układ zmieniający poziomy napięć (układ scalony max232 znajdujący się na płytce ZL2AVR). Dlatego też musimy podłączyć kabelkami odpowiednie porty zgodnie ze schematem (rys. 13).

Dodatkowo w przedstawionym programie użyty zostanie przycisk podłączony do portu B (należy wybrać jeden z przycisków zaznaczonych literką b i c na ryc. 12). Wciśnięcie tego przycisku spowoduje przesłanie paczki zarejestrowanych danych do komputera. W programie w głównej pętli znajduje się program porównujący stan zmiennych tablicy jednowymiarowej z danymi z przetwornika. W przypadku wystąpienia większej wartości na przetworniku niż w pamięci, wielkość ta zostanie zapisana jako aktualna wartość zmiennej. Program ten ma na celu zapamiętanie największych wyników które przyjdą z przetwornika. Przed wystąpieniem instrukcji LOOP znajduje się instrukcja „Debounce Pinb.5 , 0 , Prg1 , Sub", której działanie polega na sprawdzeniu stanu klawisza i ewentualny skok do podprogramu PRG1. Podprogram ten służy do przesłania tablicy zimnych na komputer oraz dodatkowo wyświetla na wyświetlaczu jeden z najwyższych zarejestrowanych wyników

```
$regfile = "m8def.dat"                    'deklaracja typu procesora

$crystal = 8000000                        ' deklaracja częstotliwości zegara procesora


Config Lcd = 16 * 2
Config Lcdpin = Pin , Db4 = Portd.5 , Db5 = Portd.4 , Db6 = Portd.3 , Db7 = Portd.2 , E = Portd.6 , Rs = Portd.7
 ' konfiguracja wyświetlacza     na porcie b

Config Adc = Single , Prescaler = Auto , Reference = Avcc   'konfiguracja AD, stala kwantowania przy ref.5v
jest 0,0048



Config Pinb.5 = Input                     'linia 5c jako wejście
Set Portb.5                               'ustawienie podciągnięcie linii do 5v

$baud = 9600                              ' ustawienie prędkości transmisji rs


Dim P As Word                             ' definiowanie zmiennych
Dim D(10) As Word
Dim A As Single
Dim A1 As Single
Dim W As String * 3
Dim K As Byte
Dim L As Byte


For K = 1 To 10                           ' czyszczenie zawartości zmiennych
 D(k) = 0
 Next K

Lcd " przyspieszenie"

Start Adc                                 ' aktywacja przetwornika

Do                                        'pętla główna programu
                                         'wyszukuje największe wartości i przepisuje do tablicy


P = Getadc(0)                            '0 pin portu c pobranie danych z przetwornika
If P > D(1) Then D(1) = P                    'jeżeli większe przepisanie do zmienne

P = Getadc(0)
If P > D(2) Then D(2) = P
   P = Getadc(0)
 If P > D(3) Then D(3) = P

P = Getadc(0)
If P > D(4) Then D(4) = P
```

```
P = Getadc(0)
If P > D(5) Then D(5) = P

P = Getadc(0)
If P > D(6) Then D(6) = P

P = Getadc(0)
If P > D(7) Then D(7) = P

P = Getadc(0)
If P > D(8) Then D(8) = P

P = Getadc(0)
If P > D(9) Then D(9) = P
 P = Getadc(0)
If P > D(10) Then D(10) = P

Debounce Pinb.5 , 0 , Prg1 , Sub              'sprawdzanie stanu przycisku


Loop


                                              ' podprogram
Prg1:
Print "pomiar"
L = 1
For K = 1 To 10                               'przeliczanie wartości  na przyspieszenie

A = D(k) * 0.0048
A = A - 1.65
A1 = A / .8
Print K ; "  " ; A1 ; " G"                    ' przesłanie wyników przez RS232


If D(k) > D(l) Then
                        'program wyświetlenia największej wartości na wyświetlaczu
Cls
Lcd A1 ; " g"
End If

  If K > 2 Then L = L + 1

Next K

For K = 1 To 10                               ' czyszczenie zmiennych
D(k) = 0
Next K

Return                                        ' powracanie do programu głównego

            End
```



Rys. 13 Dodatkowe połączenia do programu nr 4.

Do obserwacji wyników pomiaru na komputerze, należy uruchomić terminal z środowiska BASCOM (Ctrl +T). Szczegółowe opisy instrukcji języka BASCOM znaleźć można w dokumentacji dostępnej na stronie producenta http://www.mcselec.com/ oraz w wersji polskojęzycznej dostępnej na serwerze PE.

Literatura

[1] Programowanie mikrokontrolerów AVR w języku BASCOM, Marcin Wiązania, BTC2004.
[2] Measuring Tilt with Low-g Accelerometers, Michelle Clifford and Leticia GomezSensor Products, Tempe, AZ, Freescale Semiconductor 2005.
[3] Nota katalogowa układu scalonego A7260.

Wnioski:……
……
…….