Dział "Projekty Czytelników" zawiera opisy projektów nadesłanych do redakcji EP przez Czytelników. Redakcja nie bierze odpowiedzialności za prawidłowe działanie opisywanych układów, gdyż nie testujemy ich laboratoryjnie, chociaż sprawdzamy poprawność konstrukcji.

Prosimy o nadsyłanie własnych projektów z modelami (do zwrotu). Do artykułu należy dołączyć podpisane ***oświadczenie, że artykuł jest własnym opracowaniem autora i nie był dotychczas nigdzie publikowany****. Honorarium za publikację w tym dziale wynosi 200,- zł (brutto) za 1 stronę w EP. Przysyłanych tekstów nie zwracamy. Redakcja zastrzega sobie prawo do dokonywania skrótów.*

Prosta centralka alarmowa z procesorem PIC16C84

Projekt 055

Zabezpieczenie mienia było od zawsze domeną mechaniki. Za pomocą solidnych drzwi, krat, sejfów i coraz to wymyślniejszych zamków próbowano powstrzymać amatorów cudzej własności. Od pewnego czasu burzliwy rozwój elektroniki, a szczególnie techniki cyfrowej i mikroprocesorowej spowodował włączenie się tej dziedziny techniki do walki z włamywaczami. Ci ostatni oprócz łomów, pilników i dynamitu teraz są często wyposażeni w solidną wiedzę i najnowocześniejsze urządzenia techniczne.

Urządzenia elektroniczne nie utrudniają fizycznego dostępu do chronionego obiektu. Dają coś, co, jak się okazało, jest równie istotne: informację o naruszeniu chronionej strefy. Elektroniczne systemy alarmowe, a szczególnie obwody, których naruszenie powoduje wywołanie alarmu przeszły długą drogę ewolucji. Początkowo były to pętle obwodu elektrycznego, których przerwanie uruchamiało alarm. Później pojawiły się bariery optoelektroniczne w zakresie światła widzialnego, a potem podczerwonego. Obecnie stosuje się bardzo sprawne i niezawodne czujki ruchu przestrzenne i kurtynowe (płaszczyznowe).

Obwody alarmowe mają coraz częściej charakter parametryczny. Oznacza to, że centrala alarmowa na swoim wejściu sprawdza czy jest zachowany prawidłowy parametr linii np.: napięcie, prąd lub rezystancja. Na przykład rezystancja linii powinna wynosić 1kΩ ±20%. Zwarcie lub rozwarcie linii powoduje wywołanie alarmu. Centrale alarmowe to często skomplikowane systemy mikroprocesorowe mogące nadzorować wiele linii, a informacja o alarmie może być wysyłana do komputera onitorującego. Możliwe jest też wysyłanie informacji słownej o włamaniu przez linię telefoniczną pod zadany numer.

W wielu przypadkach jednak potrzebne są proste systemy alarmowe do nadzorowania mieszkań, garaży lub piwnic. Duże systemy byłyby zbyt drogie, a poza tym ich możliwości wykorzystane byłyby tylko w niewielkiej części.

Przedstawiona tutaj centralka ma wszystkie niezbędne funkcje do skutecznej ochrony małych obiektów i posiada:

- Wejście włączające centralkę w stan czuwania. Wejście to może być sterowane przez ukryty wyłącznik, cyfrowy szyfrator lub odbiornik zdalnego sterowania (radiowego albo na podczerwień).
- Linię antysabotażową, która zabezpiecza przed uszkodzeniem (przecięciem) kabli systemu i próbami dostania się do syren alarmowych, czujek ruchu lub centralki alarmowej.
- Linię szybką. Naruszenie tej linii wywołuje natychmiastowy alarm kiedy centralka jest w stanie czuwania.
- Linię opóźnioną. Naruszenie tej linii, gdy centralka jest w stanie czuwania, powoduje rozpoczęcie odliczania czasu na wyjście. Po tym czasie ponownie sprawdzane jest naruszenie tej linii. Jeżeli to nastąpi, to rozpoczyna się odliczanie czasu na wejście. W tym czasie należy wyłączyć centralę ze stanu czuwania. Gdy to nie nastąpi, to wywoływany jest alarm.
- Wyjście syreny alarmowej.
- Wyjście sygnalizacji uzbrojenia alarmu.
- Sygnalizację (zapamiętanie) naruszenia linii.
- Niezawodny, ciągły system zasilania.

Opis urządzenia

Schemat centralki przedstawiono na **rys. 1**. Głównym elementem jest mikrokontroler PIC16C84 firmy Microchip. Małe wymiary (obudowa DIL18), mały pobór prądu, obciążalność wyjść do 20mA, dwukierunkowe linie portów I/O oraz programowana szeregowo pamięć programu typu EEPROM, to tylko niektóre cechy PIC16C84 predestynujące go do tego zastosowania. Zastosowanie tego elementu spowodowało, że liczba elementów potrzebnych do budowy urządzenia została ograniczona do minimum.

Obwody wszystkich linii alarmowych zamykają się od 12V poprzez styki czujek, rezystory ograniczające, diody transoptorów do masy. Wartości rezystorów ograniczających R1, R2 i R3 są tak dobrane, że prąd płynący w obwodzie wynosi ok. 10mA dla napięcia zasilającego o wartości 10V. Dzięki temu, że obwody te mają charakter prądowy, są odporne na zakłócenia. Ma to duże znaczenie dla pewności działania systemu przez zmniejszenie ryzyka fałszywych alarmów wywołanych właś-

nie zakłóceniami na liniach. Wszystkie wejścia alarmowe są typu NC, a to oznacza, że kiedy obwód jest zwarty, to linie alarmowe nie są naruszone. Prąd płynący przez diodę transoptora powoduje włączenie w stan nasycenia fototranzystora tego transoptora. Na odpowiednim wejściu PIC16C84 pojawia się wtedy stan niski. Naruszenie linii powoduje rozwarcie styków czujek i przerwanie przepływu prądu. Tranzystor transoptora przełącza się wtedy w stan odcięcia i na wejściu PICa pojawia się stan wysoki, wymuszony przez odpowiedni rezystor R4, R5 lub R6.

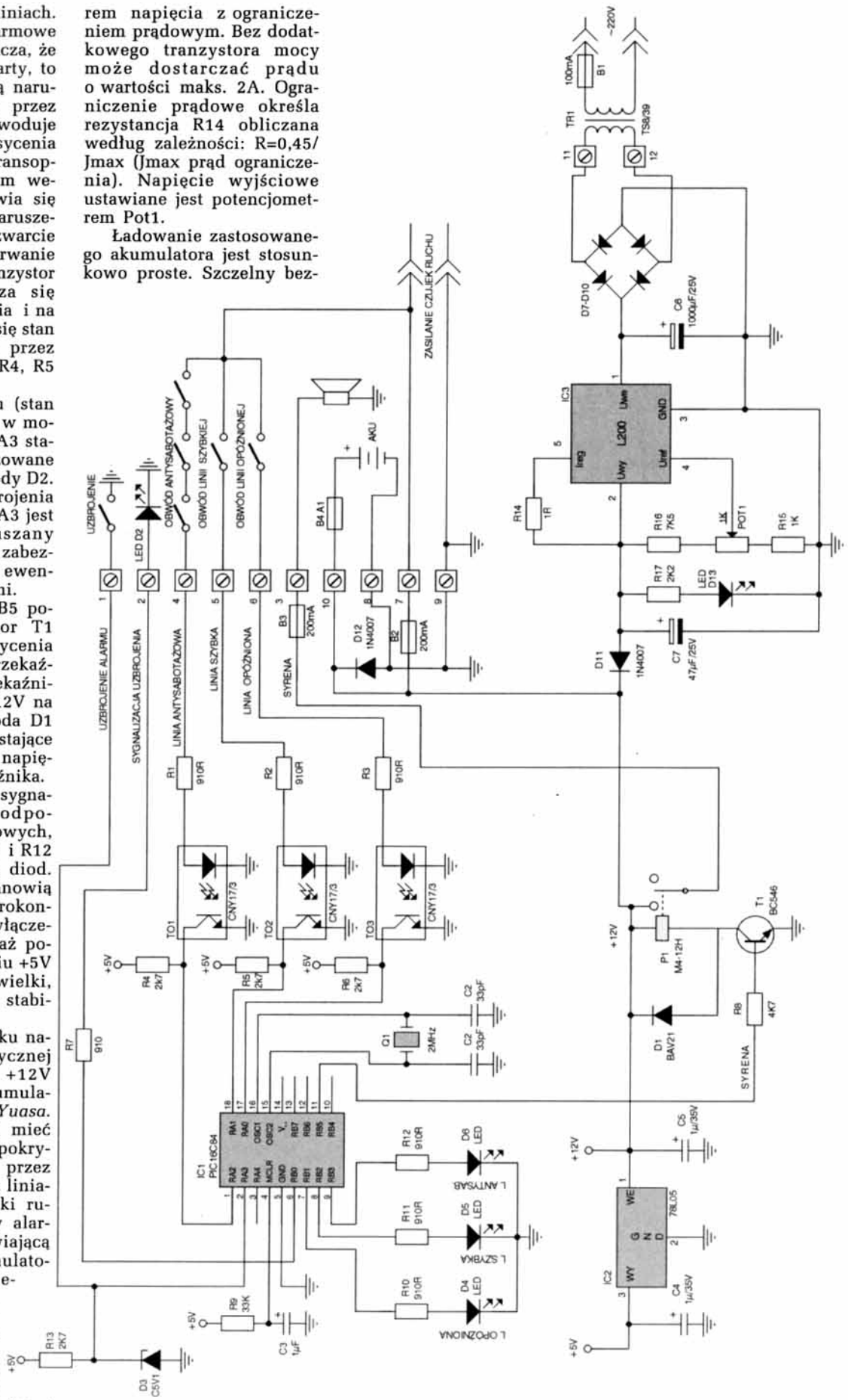
Uzbrojenie alarmu (stan czuwania) następuje w momencie podania na RA3 stanu niskiego. Sygnalizowane jest to migotaniem diody D2. Kiedy wyłącznik uzbrojenia jest rozwarty, to na RA3 jest stan wysoki wymuszany przez R13. Dioda D3 zabezpiecza wejście przed ewentualnymi przepięciami.

Stan wysoki na RB5 powoduje, że tranzystor T1 wchodzi w stan nasycenia i zaczyna działać przekaźnik P1. Styki tego przekaźnika podają napięcie +12V na syrenę alarmową. Dioda D1 tłumi przepięcia powstające w momencie zaniku napięcia na cewce przekaźnika.

Diody D4, D5 i D6 sygnalizują naruszenie odpowiednich linii alarmowych, a rezystory R10, R11 i R12 ograniczają prąd tych diod. Elementy R9 i C3 stanowią obwód zerowania mikrokontrolera w momencie włączenia zasilania. Ponieważ pobór prądu przy napięciu +5V jest stosunkowo niewielki, to można zastosować stabilizator 78L05.

W przypadku zaniku napięcia sieci energetycznej napięcie zasilające +12V jest dostarczane z akumulatora 12V/1,2Ah firmy *Yuasa*. Układ zasilania musi mieć wydajność prądową pokrywającą pobór prądu przez układ centralki wraz z liniami alarmowymi, czujki ruchu, działanie syreny alarmowej oraz umożliwiającą doładowywanie akumulatora. Napięcie z uzwojenia wtórnego transformatora jest prostowane na mostku z diod D7..D10, a następnie filtrowane przez kondensator C6. Układ L200 jest stabilizatorem napięcia z ograniczeniem prądowym. Bez dodatkowego tranzystora mocy może dostarczać prądu o wartości maks. 2A. Ograniczenie prądowe określa rezystancja R14 obliczana według zależności: R=0,45/Jmax (Jmax prąd ograniczenia). Napięcie wyjściowe ustawiane jest potencjometrem Pot1.

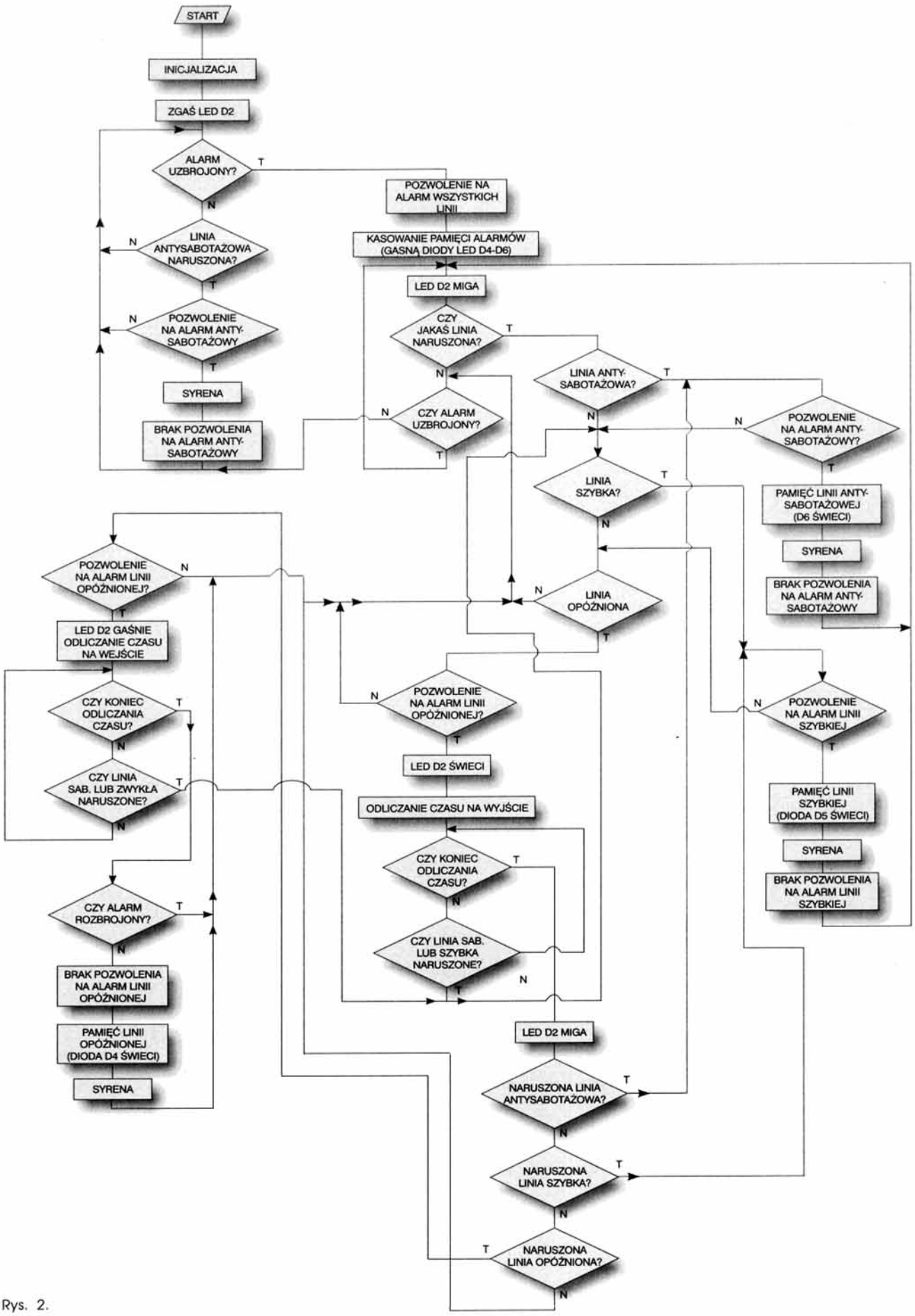
Ładowanie zastosowanego akumulatora jest stosunkowo proste. Szczelny bez-



Rys. 1.

Listing 1.

```
#include<16c84.h>

#define led          0 //sterowanie led
#define alm          5 //sterowanie syrena
#define pa_lo        1 //pamiec linii opoznionej
#define pa_ls        2 //pamiec linii szybkiej
#define pa_sab       3 //pamiec linii sabotaz.

#define lo           0 //linia opozniona
#define ls           1 //linia szybka
#define sab          2 //linia antysabotazowa
#define w_alm        3 //wylaczenie alarmu

#define w_led        0 //wskaznik led
#define p_cz         1 //wskaznik poczatku odliczania czasu
#define k_cz         2 //wskaznik konca odliczania czasu
#define p_sab        3 //wskaznik alarmu sabotazowego
#define p_ls         4 //wskaznik alarmu linii szybkiej
#define p_lo         5 //wskaznik alarmu linii opoznionej
#define nrs          6 //wskaznik nierownosci stanow

#define o_wyj        4  //opoznienie na wyjscie w sek.
#define o_wej        6  //opoznienie na wejscie w sek.
#define alarm        10 //czas trwania alarmu w sek.

#pragma portrw  zn @0x0c; //znacznik bitowy
#pragma portrw  stan @0x0d;//znacznik stanu wejsc

unsigned char czas;     //czas trwania opoznienia
unsigned char msek;     //licznik 10msek
unsigned char cz_alm;   //czas trwania alarmu

void __INT()
{
      INTCON.T0IF=0;    //zeruj flage przerwania
      TMR0=0xd9;        //10msek dla 2MHz
      --msek;           //mod. licznika 10msek
      if(msek!=0)
        goto koniec;
      else
        msek=100;
      if(zn.p_cz==1)    //odliczaj opoznienie
      {
        --czas;
        if(czas==0)
        {
          zn.k_cz=1;    //koniec opoznienia
          zn.p_cz=0;    //nie odliczaj opoznienia
        }
        else
          goto koniec;
      } // if(zn.p_cz==1)
      else;
        if(zn.w_led==0)//dioda nie miga
          goto koniec;
        else
          if(PORTB.led==0)
            PORTB.led=1;
          else
            if(PORTB.led==1)
              PORTB.led=0;
            else
koniec:               ;
}

void sprstan()          //czytanie portu a
{
      unsigned char pom;
      unsigned int i;

      INTCON.T0IE=0;
      pom=PORTA;        //port a do pom
      pom=pom&0x0f;
      if(pom!=stan)
      {
        for(i=0;i<=2000;i++)
          stan=pom;

        pom=PORTA;
        pom=pom&0x0f;
        stan=pom;
      }
      else;
      INTCON.T0IE=1
}

void ssyrena ()         //alarm sabotazowy
{
      czas=alarm;
      zn.p_cz=1;        //start odliczania czasu
      PORTB.alm=1;      //uruchomienie syreny

      while(zn.k_cz==0)
        ;               //nic nie rob
      zn.k_cz=0;
      PORTB.alm=0;      //syrena stop
}


void syrena()           //alarm z linii
{
      czas=alarm;
      zn.p_cz=1;
      PORTB.alm=1;

      while(zn.k_cz==0&&PORTA.w_alm==0)
        ;               //nic nie rob
      zn.k_cz=0;
      PORTB.alm=0;
}

void spr()
{
      stan=0;
      while(stan==0)    //w petli gdy na wszystkich liniach 0
        sprstan();

      if(zn.p_sab==1)   //pozwolenie na alarm sabotazowy
      {
        if(stan.sab==1)//linia sab naruszona
        {
          PORTB.pa_sab=1;          //pamiec linii sabotazowej
          ssyrena();
          zn.p_sab=0;
        }
        else;
      }
      else;

      if(zn.p_ls==1)    //pozwolenie na alarm linii szybkiej
      {
        if(stan.ls==1) //linia szybka naruszona
        {
          PORTB.pa_ls=1;           //pamiec linii szybkiej
          syrena();
          zn.p_ls=0;
        }
        else;
      }
      else;
}


void main(void)
{
      OPTION=0xd6;      //ustawienie option
      TRISA=0xff;       //port a jako wejsciowy
      TRISB=0x00;       //b0-b7 portu b jako wyjscia
      PORTB=0x00;
      TMR0=0xd9;        //ladowanie timera
      INTCON.GIE=1;     //odblokowanie przerwan
      zn=0x38;          //wartosc poczatkowa znacznika
      stan=0x0f;        //wartosc poczatkowa stan
      msek=100;
      INTCON.T0IE=1;    //odblokuj przerwanie od t0

pocz: zn.w_led=0;
      PORTB.led=0;      //zgas led
      while(stan.w_alm==1)
      {
        sprstan();      //sprawdz linie
        if(stan.sab==1)//naruszenie linii sabotaz.
        {
          if(zn.p_sab==1)          //pozwolenie na ssyrene
          {
            PORTB.pa_sab=1;        //pamoec linii sab.
            ssyrena();             //syrena sabotazowa
            zn.p_sab=0;            //nie uruchamiaj syreny
          }
          else
            ;
        }
        else
          ;
        zn.p_ls=1;      //pozwolenie na alarm linii szybkiej
        zn.p_lo=1;      //pozwolenie na alarm linii opoznionej
      }
      zn.p_sab=1;       // pozwolenie na alarm linii antysabotazowej
      PORTB=0;          // kasuj pamiec linii


poczl: zn.w_led=1;      //led miga
      stan=0;
      zn.p_cz=0;
      zn.k_cz=0;
      spr();            //czekaj na nar.linii
      if(stan.w_alm==1)
        goto pocz;
      else
        if(zn.p_lo==1) //pozwolenie na alarm linii opoznionej
        {
          if(stan.lo==1)          //linia opozniona naruszona
          {
            czas=o_wyj;           //czas na wyjscie
            zn.w_led=0;
            PORTB.led=1;          //led swieci
            zn.p_cz=1;            //start odliczania czasu na wyjscie
            while(zn.k_cz==0)
            {
              sprstan();
              if((stan.sab==1&&zn.p_sab==1)||(stan.ls==1&&zn.p_ls==1))
                goto poczl;
              else;
            }
            zn.k_cz=0;
            zn.w_led=1;           //led miga
            spr();                // spr naruszenia linii

            if(stan.w_alm==1)
              goto pocz;
            else
              if(zn.p_lo==1)      // pozwolenie na alarm linii opoznionej
              {
                if(stan.lo==1)    // linia opzniona naruszona
                {
                  czas=o_wej;     //czas na wyjscie
                  zn.w_led=0;
                  PORTB.led=0;
                  zn.p_cz=1;      //start odliczania czasu na wejscie

                  while(zn.k_cz==0)
                  {
                    sprstan();
                    if((stan.sab==1&&zn.p_sab==1)||(stan.ls==1&&zn.p_ls==1))
                                  //sprawdz.naruszenia linii sab i ls
                      goto poczl;
                    else;
                  } // while...
                  zn.k_cz=0;

                  if(stan.w_alm==0)           //sprawdz wylaczenie alarmu
                  {
                    zn.p_lo=0;
                    PORTB.pa_lo=1;
                    syrena();                 //syrena
                  }
                  else;
                }
                else;
              }
              else;
        }
        else;
      goto poczl;
}
```


START
INICJALIZACJA
ZGAŚ LED D2
ALARM UZBROJONY?
T
N
POZWOLENIE NA ALARM WSZYSTKICH LINII
KASOWANIE PAMIĘCI ALARMÓW (GASNĄ DIODY LED D4-D6)
LED D2 MIGA
CZY JAKAŚ LINIA NARUSZONA?
LINIA ANTY-SABOTAŻOWA?
CZY ALARM UZBROJONY?
LINIA ANTYSABOTAŻOWA NARUSZONA?
POZWOLENIE NA ALARM ANTY-SABOTAŻOWY
SYRENA
BRAK POZWOLENIA NA ALARM ANTY-SABOTAŻOWY
POZWOLENIE NA ALARM ANTY-SABOTAŻOWY?
PAMIĘĆ LINII ANTY-SABOTAŻOWEJ (D6 ŚWIECI)
SYRENA
BRAK POZWOLENIA NA ALARM ANTY-SABOTAŻOWY
LINIA SZYBKA?
LINIA OPÓŹNIONA
POZWOLENIE NA ALARM LINII OPÓŹNIONEJ?
LED D2 GAŚNIE ODLICZANIE CZASU NA WEJŚCIE
CZY KONIEC ODLICZANIA CZASU?
CZY LINIA SAB. LUB ZWYKŁA NARUSZONE?
POZWOLENIE NA ALARM LINII OPÓŹNIONEJ?
LED D2 ŚWIECI
ODLICZANIE CZASU NA WYJŚCIE
CZY KONIEC ODLICZANIA CZASU?
CZY LINIA SAB. LUB SZYBKA NARUSZONE?
POZWOLENIE NA ALARM LINII SZYBKIEJ
PAMIĘĆ LINII SZYBKIEJ (DIODA D5 ŚWIECI)
SYRENA
BRAK POZWOLENIA NA ALARM LINII SZYBKIEJ
CZY ALARM ROZBROJONY?
BRAK POZWOLENIA NA ALARM LINII OPÓŹNIONEJ
PAMIĘĆ LINII OPÓŹNIONEJ (DIODA D4 ŚWIECI)
SYRENA
LED D2 MIGA
NARUSZONA LINIA ANTYSABOTAŻOWA?
NARUSZONA LINIA SZYBKA?
NARUSZONA LINIA OPÓŹNIONA?

Rys. 2.

obsługowy żelowy akumulator wymaga stabilizowanego napięcia o wartości zależnej od temperatury otoczenia. Ponieważ urządzenie będzie pracować w prawie stałej temperaturze pokojowej, to można przyjąć dla tego typu akumulatora napięcia ładowania o wartości 13,6V. Napięcie na wyjściu stabilizatora (nóżka 2 L200) musi mieć więc wartość 13,6V plus spadek napięcia na diodzie D11. Dioda D11 odcina przepływ prądu w kierunku stabilizatora w czasie zaniku napięcia sieci energetycznej. Dioda D12 zabezpiecza układ przed odwrotnym podłączeniem akumulatora. Dioda D13 sygnalizuje obecność napięcia ładowania. Prąd ładowania jest ograniczany przez samo ogniwo i nie ma potrzeby go kontrolować lub regulować. Dla naszej centralki, przy założeniu, że podłączone będą maksymalnie 2 czujki ruchu i 1 syrena alarmowa, ograniczenie prądowe można ustalić na ok. 450 mA. Aby zwiększyć wydajność prądową należy zmniejszyć wartość R14. Należy wówczas pamiętać o zwiększeniu powierzchni radiatora oraz ewentualnej zmianie transformatora i diod D7..D10.

Program

Program sterujący pracą centralki został napisany w języku C. Wydruk wersji źródłowej jest przedstawiony na **list. 1.**

Procedura *void_INT( )* obsługuje przerwanie od licznika/czasomierza T0. Źródłem impulsów dla T0 jest częstotliwość zegara mikrokontrolera podzielona przez 4. Preskaler jest przypisany do T0 i dzieli wstępnie częstotliwość przez 128. Do licznika jest wpisywana wartość D9hex tak, że łącznie z preskalerem generuje on przerwanie co 10 ms. *Void_INT( )* odmierza czas z dokładnością 1 s, oraz steruje migotaniem diody D2 sygnalizującej uzbrojenie alarmu.

Procedura *sprstan( )* realizuje czytanie linii portu A. Jeżeli stan linii nie zmienił się od ostatniego wywołania *sprstan( )* to następuje wyjście z procedury. Jeżeli nastąpiła zmiana (zmienna *stan* nie ma wartości równej zmiennej *pom*), to następuje odliczanie programowego opóźnienia i wpisanie do zmiennej *stan* wartości *pom*. Ma to na celu likwidację skutków drgań styków czujek w momencie ich przełączania oraz dodatkowo zabezpiecza przed krótkotrwałymi zakłóceniami na liniach alarmowych.

Procedura *syrena( )* uaktywnia syrenę alarmową na czas określony zmienną *alarm*. Przerwanie działania syreny następuje gdy *w_alm*=1 (alarm rozbrojony). Procedura *syrena( )* działa jak syrena, ale nie można jej przerwać stanem linii *w_alm*.

Procedura *spr( )* sprawdza naruszenie linii antysabotażowej i szybkiej oraz uruchamia odpowiedni alarm. Algorytm działania programu centralki przedstawiony jest na **rys. 2.**

W Internecie na stronach Microchipa pod adresem: *http://www.microchip.com/10/Tools/mTools/MPLAB/index.htm* można znaleźć pakiet MPLAB, który zawiera między innymi asembler, symulator programowy i kompilator języka C dla procesorów PIC.

Kompilator C jest w wersji demo i umożliwia napisanie programów, których kod wynikowy ma rozmiar nie większy niż 256 słów. Program centralki zajmuje pamięć do adresu F7hex, a więc wykorzystuje prawie wszystkie możliwości wersji demo.

Po ściągnięciu plików pakietu MPLAB i jego rozpakowaniu (*format.zip*) oraz zainstalowaniu (wersja Windows 95) należy zainstalować kompilator C. Po rozwinięciu ikony „Project" trzeba wybrać „Install Language Tool". W polu „Language" zaznaczamy Microchipa, a w polu „Tool Name" MPLAB-C V1.10. Teraz trzeba znaleźć zbiór *labcdemo.exe*.

Jeżeli pakiet MPLAB był instalowany zgodnie z poleceniami programu instalacyjnego to powinien się znajdować w katalogu *c:\progra~1\mplab*. Jeżeli w trakcie instalacji zostały wybrane inne ścieżki dostępu to należy kliknąć na „browse" i odnaleźć *labcdemo.exe* w innym katalogu. Po stworzeniu własnego projektu i przepisaniu zbioru źródłowego trzeba go skompilować (klawisz F10). Jeżeli wszystko jest w porządku, to pojawi się „build completed successfully" i zostanie utworzony zbiór z rozszerzeniem **.hex*.

I tutaj dochodzimy do momentu, kiedy układ trzeba zaprogramować. Nie potrzeba drogich programatorów i oprogramowania. Wszystko można mieć prawie za darmo z Internetu. Układ PIC16C84 można zaprogramować za pomocą prostego programatora opisanego w polskiej wersji Elektora 7/97. Programator taki składa się z kilku elementów i jest zasilany i sterowany z łącza COM PCta. Możliwe to jest dlatego, że pamięć programu mikrokontrolera jest szeregowo programowaną pamięcią EEPROM. Z programatorem opisanym w Elektorze współpracuje program PIP02. I tutaj zaczynają się pewne problemy. Starsze wersje tego programu nie chcą współpracować z komputerami z zegarem szybszym niż 40..50 MHz.

Pod adresem *http://www.d2mac.com/picprog.html* można znaleźć program *PIP02* i zbiór *slow.zip*. Ten ostatni po rozpakowaniu i uruchomieniu programu *cachectl.exe* zwalnia działanie procesora i programowanie przebiega prawidłowo.

Jest jednak pewne ograniczenie: nie wolno pod Windows 95 uruchamiać *cachectl.exe*, ponieważ blokuje on cache procesora i jest zgłaszany błąd systemu operacyjnego. Należy więc zamknąć WINDOWS 95 i uruchomić komputer tylko pod DOS-em. Aby ułatwić sobie pracę z programowaniem można napisać krótki plik wsadowy:

```
com84 com1
;instaluj driver programatora
;na COM1
cachectl d
;cache procesora zablokowany
pip02   ;program programatora
cachectl e
;cache procesora odblokowany
com84 remove
;usuń driver programatora
```

Przed każdym zaprogramowaniem należy ustawić bezpieczniki: *OSC=XT, WDT=OFF, PWR=OFF, CODE PROTECT=ON*. Pod adresem *http://www.sistudio.com/sistudio/download/html* można znaleźć najnowszą wersję programu PIP02, która poprawnie pracuje z szybszymi procesorami. Jednak autorowi nie udało się prawidłowo zaprogramować bezpieczników, pomimo ich ustawienia w programie źródłowym.

Jeżeli po uwzględnieniu wszystkich powyższych uwag i po prawidłowym zmontowaniu programatora, mikrokontroler dalej nie chce się programować, to należy sprawdzić, czy port szeregowy komputera pracuje w pełnym standardzie RS232, to znaczy, że napięcia na liniach tego portu zmieniają się w granicach ±12V.

Dla napięć niższych należy zmodyfikować układ programatora tak, jak to zostało opisane w polskiej wersji Elektora 2/98. Tak zmodyfikowany programator pracuje poprawnie z portami o napięciu na liniach od ±7V do ±12V. Mikrokontroler opisanej centralki został prawidłowo zaprogramowany za pomocą laptopa OPTIMUS z procesorem

WYKAZ ELEMENTÓW

Rezystory
R1, R2, R3, R7, R10, R11, R12, R17: 910Ω
R4, R5, R6, R13: 2,7kΩ
R9: 33kΩ
R8: 4,7kΩ
R14: 1Ω
R15: 1,5kΩ
R16: 7,5kΩ
R17: 2,2kΩ
POT1: 1kΩ

Kondensatory
C1, C2: 33pF
C3, C4, C5: 1µF/35V tantal
C6: 1000µF/25V
C7: 47µF/25V

Półprzewodniki
IC1: PIC16C84/04
IC2: 78L05
IC3: L200
TO1, TO2, TO3: CNY17/3
T1: BC546
D1: BAV21
D2, D4, D5, D6, D13: LED 5mm czerwona (o dużej jasności)
D3: C5V6
D7..D12: 1N4007

Różne
B1: 100mA
B2, B3: 200mA
B4: 1A
Q1: 2MHz
TR1: transformator TS8/39
Listwa zaciskowa śrubowa do druku, zaciski bezpiecznikowe do druku 4 szt., oprawa bezpiecznikowa na kabel, oprawa bezpiecznikowa, obudowa plastykowa, kabel sieciowy, akumulator Yuasa 12V/1,2Ah

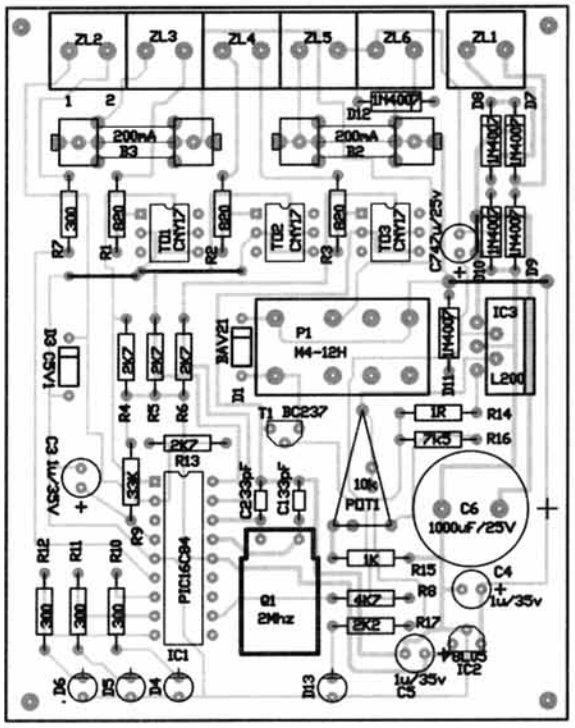
486DX4/100MHz i portem szeregowym z napięciami ±8V (z układem MAX213).

Centralka została zmontowana na jednostronnej płytce drukowanej. Rozmieszczenie elementów jest pokazane na **rys. 3.** Radiator układu L200 przymocowany jest bezpośrednio do płytki. Płytka drukowana została zaprojektowana za pomocą programu Autotrax i wykonana z wykorzystaniem emulsji pozytywowej POSITIV 20.

Pierwszą czynnością uruchomieniową po zmontowaniu całości jest sprawdzenie zasilacza. Potencjometrem Pot1 należy ustawić napięcie wyjściowe na wartość 14,4V (13,6V napięcia ładowania akumulatora plus 0,8V spadku na diodzie D11). Następnie sprawdzamy ograniczenie prądowe. Dla R14=1Ω prąd ograniczenia powinien wynosić ok. 450 mA.

Po tych czynnościach można dołączyć akumulator (plus do plusa, minus do minusa). Prąd ładowania może mieć początkowo wartość ograniczenia prądowego zasilacza, ale powinien z czasem maleć. Naładowany akumulator nie powinien pobierać więcej niż 10..15mA. Po naładowaniu korygujemy napięcie na akumulatorze do wartości 13,6V. Następnie należy sprawdzić poprawność napięcia +5V i można włożyć do podstawki zaprogramowany mikrokontroler. Na wszystkie wejścia alarmowe podajemy napięcie +12V, a wyłącznik uzbrojenia ma być rozwarty. Żadna dioda LED nie powinna się świecić, a przekaźnik nie powinien zadziałać.

Rozwarcie linii antysabotażowej powoduje zadziałanie przekaźnika P1 na czas określony zmienną alarm i zaświeci się dioda LED D6. Następnie należy zewrzeć linię antysabotażową i zewrzeć wyłącznik uzbrojenia alarmu. Powinna zgasnąć D6 i zacząć migać dioda LED D2 (co 1 s). Rozwieramy linię antysabotażową i powinien zadziałać P1 oraz zapalić się dioda D6.



Rys. 3.

Rozwieramy teraz linię szybką. Powinien zadziałać P1 i zapalić się dioda LED D5. Rozwarcie linii opóźnionej powoduje zapalenie się diody D2 na czas określony zmienną *o_wyj*. Jeśli po tym czasie linia opóźniona nie jest naruszona, to D2 zaczyna migotać. Ponowne naruszenie teraz tej linii powoduje rozpoczęcie odliczania czasu na wejście (zmienna *o_wej*) i zgaszenie diody D2. Jeżeli po tym czasie centralka jest nadal w stanie uzbrojenia to wywoływany jest alarm i świeci się dioda D4. Trwałe naruszenie linii nie powoduje ciągłego alarmu.

Syrena uaktywnia się od naruszenia każdej z linii tylko raz w jednym cyklu uzbrojenia. Wyłączenie czuwania nie powoduje gaszenia diod D4..D6. Można więc stwierdzić, czy w czasie naszej nieobecności jakaś linia nie została naruszona. Ponowne uzbrojenie centralki powoduje zgaszenie diod D4..D6. Podłączenie czujek i syren alarmowych w czasie instalacji systemu alarmowego najlepiej jest wykonać za pomocą kabla zawierającego min. 3 pary przewodów. Jedna para do podłączenia zasilania, druga to obwód antysabotażowy a trzecia obwód linii alarmowej.

Modelowe urządzenie zostało zamontowane w plastykowej obudowie o wymiarach 180x150x70 mm. Jeżeli centralka ma być rozbrajana ukrytym wyłącznikiem, a czasy na wejście i wyjście wynoszą kilkadziesiąt sekund, to należy ją ukryć. Umieszczona w widocznym miejscu może być łatwo zniszczona przed upływem czasu potrzebnego na wejście. Można też całe urządzenie umieścić w typowej metalowej obudowie, zamykanej na kluczyk, dostępnej w sklepach z urządzeniami alarmowymi. **Należy wtedy pamiętać o ochronie przeciwporażeniowej (zerowanie lub uziemienie).**

Przedstawiony tutaj opis pozwala na całkowicie samodzielne wykonanie małego systemu alarmowego. W założeniu miało to być urządzenie proste, ale może być bazowym do dalszych rozszerzeń i modyfikacji. Wzorując się na tym rozwiązaniu opracowano prosty alarm samochodowy z sygnalizacją włączonych świateł.

Tomasz Jabłoński

Literatura:
1. PIC16C84 materiały firmowe Microchip Technology Inc. DS30445C
2. B. W. Kernighan, D. M. Ritchie, „Język C", PWN, 1988.
3. YUASA Co. „Application manual".