# Uniwersalny zamek szyfrowy

## kit AVT-311

*Na łamach EP (EP4/93) opis zamka szyfrowego był już prezentowany. Oto jego kolejna wersja, której konstrukcja została oparta na mikrokontrolerze PIC16C84.*

Mikrokontroler PIC16C84 posłużył do konstrukcji prostego zamka cyfrowego. Cechą wyróżniającą go w swej klasie jest reprogramowalność w układzie - pamięć programu i fragment pamięci danych są typu EEPROM. Poza tym ma on wszystkie cechy charakterystyczne dla procesorów PIC16CXX:

- architektura harwardzka, czyli pamięć programu i pamięci danych są rozdzielone;
- wszystkie rozkazy są wykonywane w czterotaktowym cyklu;
- układ watchdoga ma niezależny od systemowego zegar;
- cztery rodzaje oscylatorów zegara systemowego;
- jednopoziomowy układ przerwań o priorytetach źródeł ustawianych programowo;
- w pełni statyczna logika CMOS;
- szeroki zakres napięć zasilania: 2.0V do 6.0V;
- niewielki pobór mocy.

Dodatkowo, oprócz 36 8-bitowych rejestrów, dla programisty dostępna jest również nieulotna 64-bajtowa pamięć EEPROM. Wprawdzie producent dostarcza wersję co najwyżej 10MHz, ale trzeba pamiętać, że taki zegar odpowiada zegarowi 30MHz w architekturze procesora z rodziny '51. Do naszych zastosowań to aż nadto.

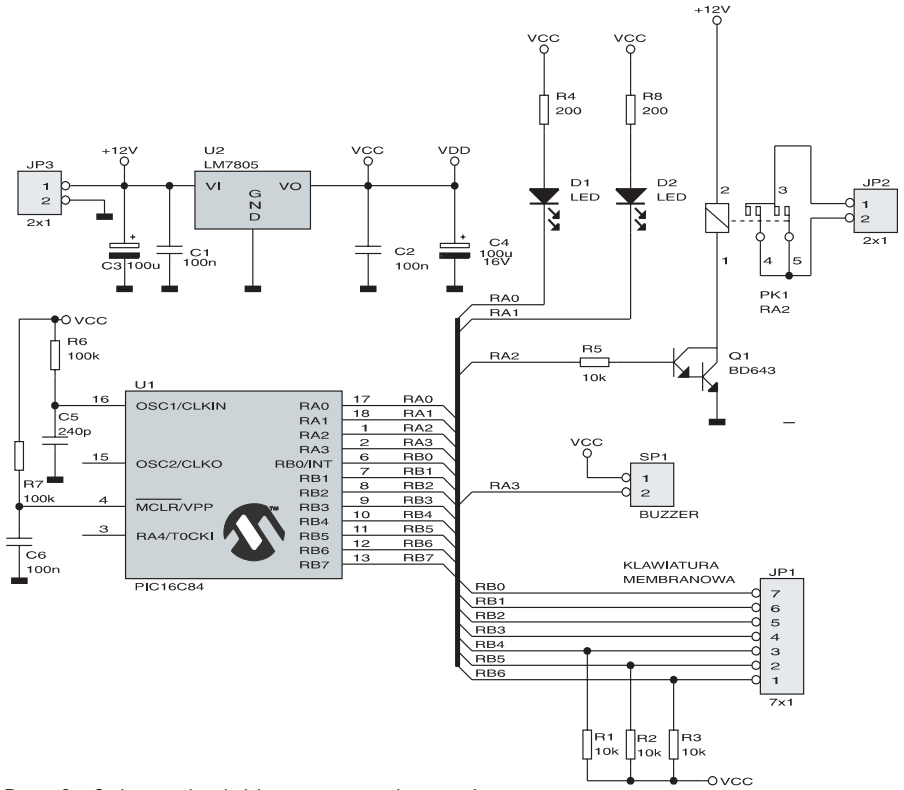
### Opis układu

Nasz zamek posiada następujące walory:

- prostotę układową;
- możliwie dużą liczbę kombinacji kodowych, a wcale nie muszą to być ciągi liczbowe, wynoszącą $11^{63}$;
- odporność na wyłączenie zasilania, polegającą na tym, że kod otwarcia jest pamiętany niezależnie od zasilania układu.

Na **rys.1** pokazano schemat elektryczny zamka szyfrowego. Sercem urządzenia, i zarazem jedynym układem scalonym, jest mikroprocesor PIC16C84. Zapisany w nim program steruje całym urządzeniem. Do linii portów procesora dołączono klawiaturę matrycową 3x4, tranzystor włączający przekaźnik PK1, piszczyk oraz dwie diody LED, służące do sygnalizacji stanu zamka. Zasilanie procesora, diod LED i klawiatury zapewnia stabilizator typu LM7805. Zegar procesora jest typu RC, nie ma bowiem potrzeby stosowania kwarcu, jeśli układ nie wykonuje precyzyjnych pomiarów czasu. Na tym w zasadzie można zakończyć opis budowy zamka. Bardziej interesujący jest program zapisany w pamięci mikrokontrolera.

Zanim omówimy program zamka, skupmy się na chwilę na klawiaturze. Jest to klawiatura membranowa, wykonana w wersji bez opisu klawiszy. Opis klawiszy jest sprawą użytkownika, czyli zamiast standardowych napisów użytkownik może, w formie podkładu, przyjąć własny układ kla-




Rys. 1. Schemat elektryczny zamka szyfrowego.

wiatury. Na **rys.2**, dla potrzeb tego artykułu, przyjęto układ klawiatury telefonicznej. Dalej pokażemy, jak stworzyć inny układ klawiatury.

## Algorytm pracy procesora

Algorytm programu pokazano na **rys.3** i **4**. Rys.3 dotyczy algorytmu programu głównego, zaś rys.4 przedstawia procedurę szyfrującą kod otwarcia.

Po restarcie procesora następuje sprawdzenie, czy pamięć EEPROM nie zawiera kodu. Brak kodu oznacza, że we wszystkich 64 bajtach pamięci jest zapisana liczba 0FFH. Należy więc wywołać procedurę szyfrującą kod otwarcia. O procedurze szyfrującej napiszemy dalej, teraz dokończymy omawianie programu głównego. Po wykryciu zapisu danych w pamięci EEPROM procedura szyfrowania jest omijana i zamek jest zamykany. W ten sposób układ broni się przed próbą rozkodowania poprzez wyłączenie zasilania.

Po zakończeniu kodowania zamka następuje jego zamknięcie. Zrobiono to w celu natychmiastowego sprawdzenia poprawności kodu otwarcia. Przechodzimy zatem do właściwej procedury otwierania zamka.

Zmienna EEADR widoczna na rys.3 jest zmienną systemową oznaczającą bieżący adres pamięci EEPROM mikrokontrolera. Do niej zapisuje się wstępnie wartość odpowiadającą adresowi najstarszej komórki pamięci.

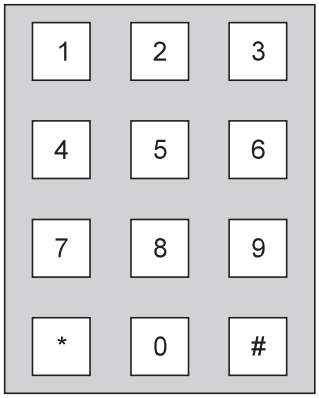
Odczyt klawiatury w tym urządzeniu jest sekwencyjny. Klawiatura jest matrycą włączników 3x4. Co pewien czas, zależny od wewnętrznego układu timera, sprawdzany jest stan klawiszy. Wykrycie naciśnięcia klawisza powoduje zapis jego kodu do bufora klawiatury. Przy zwolnionych przyciskach klawiatury w buforze jest wpisywany kod 0FFH. Procedura odczytu klawiatury najpierw oczekuje na naciśnięcie klawisza, a potem na jego zwolnienie, zapisując kod naciśniętego klawisza do osobnej komórki pamięci danych.

Po odczytaniu klawiatury, w celu potwierdzenia jej naciśnięcia, dioda koloru zielonego (na schemacie z rys.1 jest to dioda D2) błyśnie krótko, a piszczyk równie krótko zapiszczy (przyjmijmy dla uproszczenia opisu, że jeśli dioda D2 błyśnie, to również odzywa się piszczyk). Następuje odczyt komórki pamięci EEPROM o adresie zawartym w EEADR. Jeśli kod odczytanej komórki i kod klawisza są zgodne, zmienna EEADR jest modyfikowana oraz sprawdzane jest, czy jest to kod klawisza # (rys.2), gdyż naciśnięcie klawisza # oznacza zakończenie wprowadzania kodu otwarcia. Dwukrotne błyśnięcie i ciągłe świecenie się diody zielonej oraz włączenie przekaźnika jest sygnałem, że zamek został otwarty.

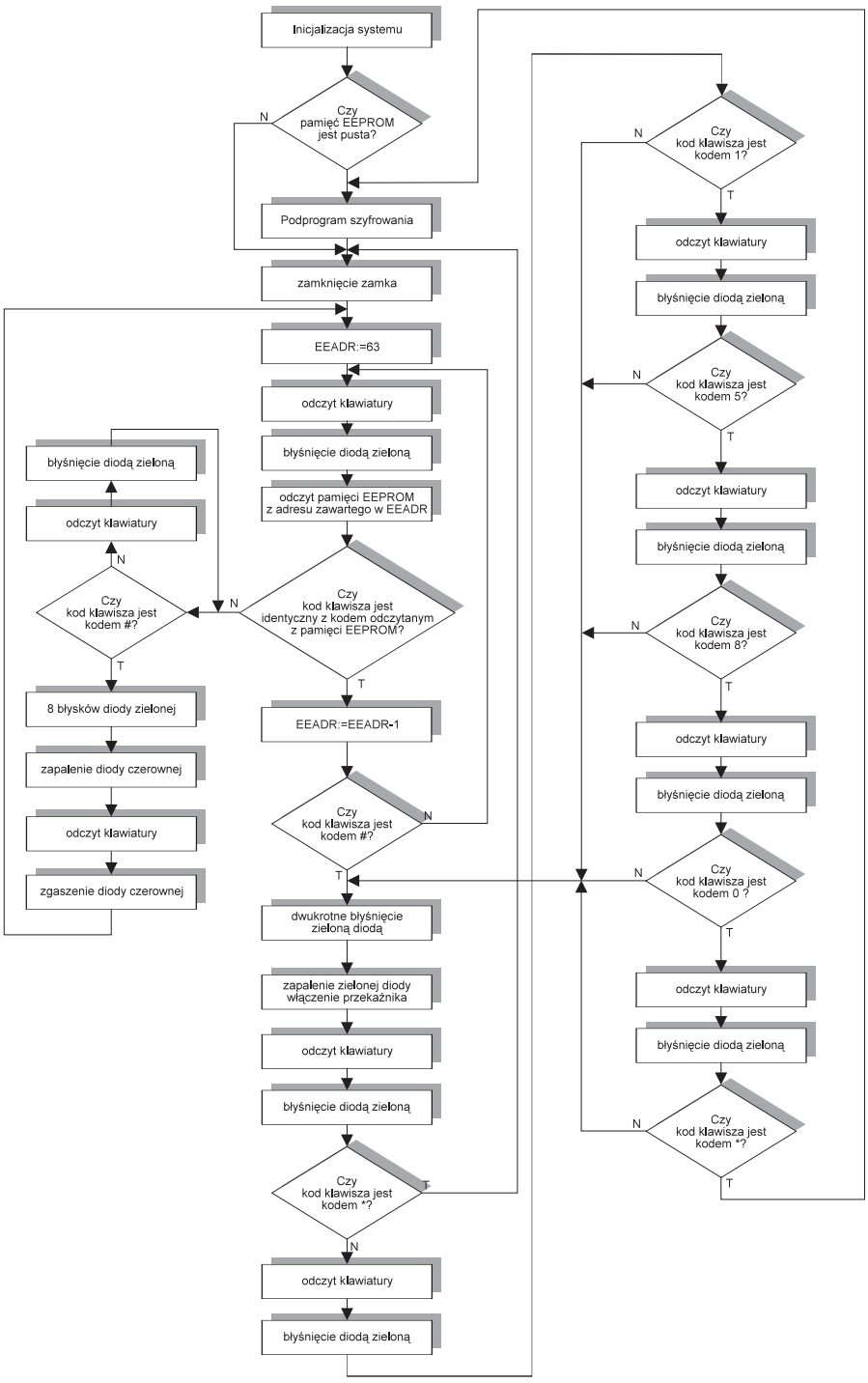
W stanie otwarcia zamka istnieje możliwość zmiany kodu otwarcia. Ażeby utrudnić osobom niepowołanym tego rodzaju manipulacje, zakodowano również dostęp do procedury szyfrującej. Tym razem jest to kod stały : 1-5-8-0-*. Podanie takiej sekwencji znaków spowoduje przeniesienie sterowania na początek programu, do procedury szyfrującej. Pomyłka w jej wprowadzaniu oznacza, że trzeba będzie ją powtórzyć od początku. Sam znak gwiazdki zamyka zamek.

Jeżeli kod odczytanej komórki i kod klawisza różnią się od siebie, program przechodzi do pętli obok i w niej oczekuje na naciśnięcie klawisza #, już bez porównywania wspomnianych wartości. Kiedy już ten klawisz zostanie naciśnięty, zielona dioda LED ośmiokrotnie zabłyśnie i zapali się dioda czerwona D1. Taka jest sygnalizacja podania błędnego kodu otwarcia. W tym stanie mikrokontroler pozostanie aż do naciśnięcia dowolnego klawisza. Wtedy program powróci na początek procedury otwierania zamka, dając kolejną szansę na jego otwarcie.

Opiszemy teraz procedurę szyfrowania kodu otwarcia zamka. Jak to widać na rys.4, wejście do niej jest sygnalizowane zapaleniem się obu diod, czerwonej i zielonej. Ustawiany jest adres początkowy




Rys. 2. Wygląd klawiatury membranowej.




Rys. 3. Algorytm procedury głównej zamka szyfrowego.

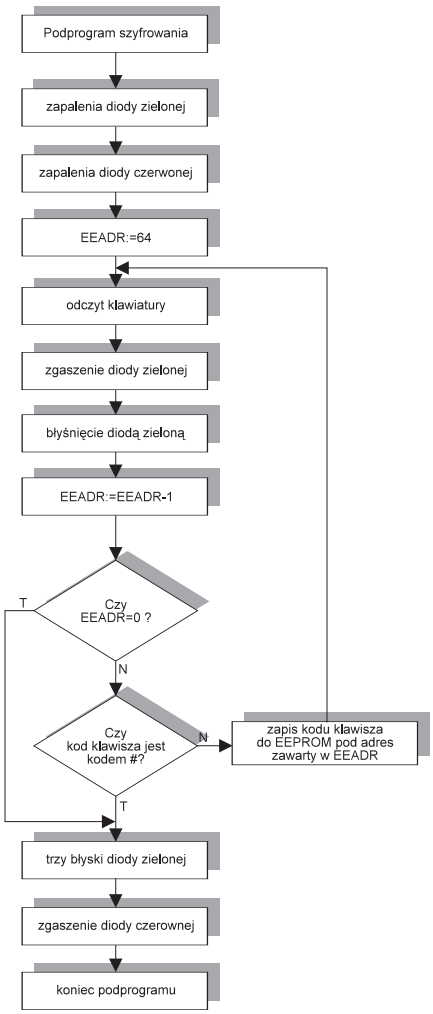
pamięci EEPROM, następuje odczyt klawiatury z jednoczesnym błyśnięciem diody zielonej. Zmienna EEADR może być tak długo dekrementowana (jej wartość jest zmniejszana o 1), aż osiągnie wartość 0, wtedy procedura jest automatycznie zakończona. Procedurę szyfrowania można zakończyć wcześniej poprzez naciśnięcie klawisza # i ten przypadek będzie w praktyce częściej spotykany, ponieważ trudno wyobrazić sobie pracowite „wklepywanie“ tak wielkiej liczby znaków. W razie pomyłki proces wprowadzania kodu otwarcia należy zacząć od początku.

Kilku słów wyjaśnienia wymaga zapis i odczyt pamięci EEPROM. Dostęp do pamięci EEPROM jest możliwy poprzez jednobajtowe okno, obsługiwane przez trzy rejestry - EECON2, EEADR i EEDATA - oraz przez trzy zmienne jednobitowe: WREN, WR, RD, które są bitami rejestru EECON1. Zmienna EEADR, jak już wyżej napisano, pamięta adres komórki pamięci, w zmiennej EEDATA zawarto daną przeznaczoną do zapisu albo będzie tam dana odczytana z pamięci EEPROM. Wskaźnik RD, poprzez jego ustawienie, inicjuje odczyt z komórki pamięci o adresie zawartym w EEADR. Ustawiony wskaźnik WR, po zezwoleniu przez ustawienie wskaźnika WREN, inicjuje zapis do pamięci EEPROM. Zapis do niej jest obwarowany dodatkowymi warunkami. Przed zapisem właściwego bajtu należy:

- do EEADR wpisać właściwy adres komórki pamięci EEPROM;
- do EEDATA wpisać daną do zapisu;
- ustawić bit WREN;
- wyłączyć system przerwań w mikrokontrolerze;
- do rejestru EECON2 wpisać 55H;
- do rejestru EECON2 wpisać 0AAH;
- włączyć system przerwań w mikrokontrolerze;
- ustawić bit WR inicjujący zapis.

Tak skomplikowany sposób zapisu jest podyktowany chęcią zapewnienia maksymalnej ochrony danych przed ich utratą. W momencie zerowania mikrokontrolera nie można przewidzieć, czy przypadkiem nie nastąpi inicjacja zapisu poprzez ustawienie bitu WR. Ustawienie takiej bariery praktycznie uniemożliwia taki przypadek. Jeśli założymy, że taka sekwencja rozkazów, jak wyżej wymieniona jest możliwa, to przy 14 bitach słowa procesora i potrzebnych do zapisu co najmniej 6 słów programu, prawdopodobieństwo samorzutnego pojawienia się właściwej sekwencji zapisującej jest równe $1/(2^{14*6}) = 5.17x10^{-26}$.

Należy wiedzieć, że czas trwania zapisu wynosi ok. 10ms, dlatego w przypadku próby zapisu sekwencji bajtów warto skorzystać z systemu przerwań, który ma możliwość obsługi przerwania od zakończenia zapisu do pamięci EEPROM. W opisywanym urządzeniu taka potrzeba nie występowała ze względu na długie czasy odczytu klawiatury, dochodzące do 1 sekundy. Bit WR jest automatycznie zerowany po zakończeniu procesu zapisu.



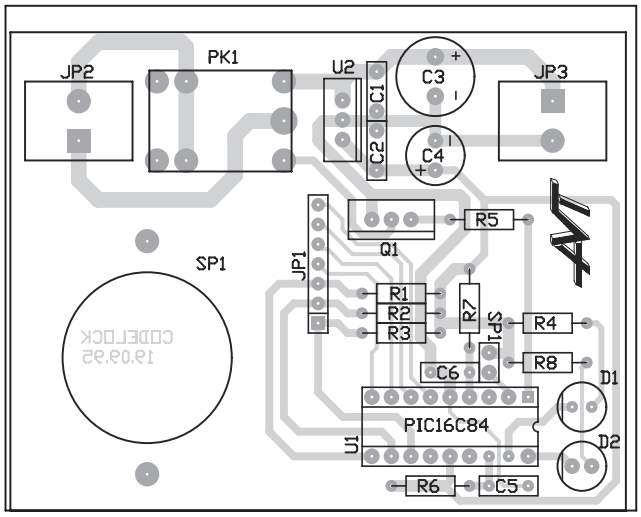

Rys. 4. Algorytm procedury szyfrującej.

Proces odczytu nie wymaga tak skomplikowanej obsługi. Wystarczy ustawić bit RD bajtu kontrolnego EECON1. Po zakończeniu transmisji danej do EEDATA, bit RD zostanie automatycznie wyzerowany.

## Montaż i uruchomienie

**Rys.5** przedstawia rozmieszczenie elementów na płytce. Projekt płytki drukowanej przedstawiono na wkładce wewnątrz numeru. Montaż płytki zaczynamy od wlutowania podzespołów na swoje miejsce na płytce. Pod mikrokontroler proponujemy zastosować podstawkę. Nie zapomnijmy o właściwym wlutowaniu diod o odpowiednich kolorach na swoich miejscach. Podłączamy klawiaturę do złącza JP1. Układ połączeń wewnętrznych klawiatury przedstawiono na **rys.6**. Taśma wychodząca z klawiatury ma siedem ścieżek, taka jest bowiem sumaryczna liczba kolumn i wierszy. Każdy pin taśmy połączeniowej klawiatury został opisany jako ciąg znaków podłączonych do danego pinu. Ażeby uzyskać zwarcie po naciśnięciu konkretnego klawisza, należy na rys.6 znaleźć go na dwóch różnych nóżkach opisu złącza JP1.

W czasie uruchomienia zasilamy układ ze źródła napięcia 12V. Przed włożeniem procesora PIC należy na nóżkach 14 i 5 podstawki sprawdzić obecność napięcia zasilającego procesor, wynoszącego 5V.




Rys. 5. Płytka drukowana zamka szyfrowego.

Teraz należy bardzo uważnie postępować, aby nie doprowadzić do sytuacji, w której nie będzie znany kod otwarcia zamka. Wkładamy procesor w podstawkę, włączamy zasilanie. Po pierwszym włączeniu jest uruchamiana procedura szyfrowania kodu otwarcia, czyli obie diody się świecą. Za pierwszym razem należy wprowadzić szyfr jednoznakowy, kończąc go znakiem #. Potem trzeba go sprawdzić, próbując wpisać ten sam szyfr. Zamek powinien się otworzyć, co będzie sygnalizowane stałym świeceniem diody zielonej.

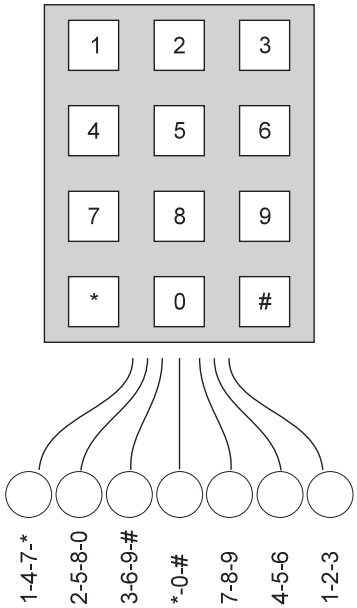
W stanie otwarcia zamka możemy zmienić kod na obrany przez nas. Musimy nacisnąć po kolei klawisze 1-5-8-0-*. Naciśnięcie każdego z tych klawiszy jest sygnalizowane jednym piśnięciem piszczyka i jednym błyśnięciem diody zielonej. Wybranie innego klawisza lub zmiana kolejności spowoduje dwukrotne piśnięcie piszczyka i dwukrotne błyśnięcie diody zielonej. Wtedy, chcąc przekodować zamek, musimy wprowadzać od początku tę sekwencję znaków. Zamknięcie zamka polega na przyciśnięciu klawisza *.

Wybranie błędnego kodu zakończy się ośmiokrotnym błyśnięciem diody zielonej i ciągłym świeceniem diody czerwonej. Diodę czerwoną można zgasić dowolnym klawiszem i zacząć wprowadzanie kodu od początku.

## Inne oznakowanie klawiatury

Autor celowo przewidział stosowanie klawiatury membranowej bez opisu. Wprowadzanie kilkunastocyfrowych ciągów liczb nie jest wygodne, a tylko takie ciągi zapewnią znaczną trudność otworzenia zamka. Autor proponuje dwie zasady kodowania. Kiedyś, w nie tak odległych czasach, gdy komputery były marzeniem fantastów, do pamiętania np. cyfr liczby pi używano łatwych do zapamiętania sentencji. Liczba liter poszczególnych słów określała kolejną cyfrę. Np. „Daj, o pani, o boska Mnemozyno, pi liczbę...“

W naszym zamku, zamiast pamiętać długie ciągi cyfr, zapamiętajmy znany nam zwrot, którego liczba liter poszczególnych słów




Rys. 6. Układ połączeń klawiatury membranowej 3x4.

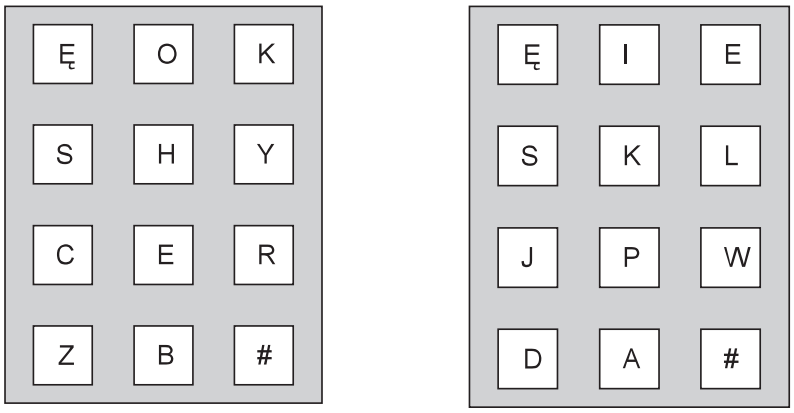
**WYKAZ ELEMENTÓW**

**Rezystory**
R1, R2, R3, R5: 10kΩ (5,6kΩ÷20kΩ)
R4, R8: 200Ω (180Ω÷300Ω)
R6, R7: 100kΩ

**Kondensatory**
C1, C2, C6: 100nF (47nF÷150nF)
C3, C4: 100µF/25V
C5: 240pF

**Półprzewodniki**
D1: Dioda LED ϕ5 koloru czerwonego
D2: Dioda LED ϕ5 koloru zielonego
Q1: BD643, BD645, BD647, BD649
U1: PIC16C84-04/P, PIC16C84-10/P - zaprogramowany
U2: LM7805, LM78M05

**Różne**
JP1: rząd siedmiu złoconych pinów
JP2, JP3: Złącze ARK2
PK1: przekaźnik RA2, RA2P, RA2Z, na napięcie 12V
Klawiatura membranowa 3x4 bez opisu albo z opisem klawiatury telefonicznej
podstawka DIL18




Rys. 7. Wkładki kodowe dla zwrotów CHRZĘSKRZYBOCZEK i APELAJDA SĘKLIWA.

będzie oznaczać cyfry kodu.

Drugim sposobem łatwiejszego kodowania jest zakodowanie zwrotu, przez zastąpienie układu dotychczas omówionej klawiatury telefonicznej układem liter. Budowa klawiatury zapewnia łatwą wymianę wkładki opisującej klawisze.

Zakodowany zwrot powinien być na tyle egzotyczny, aby nie został łatwo rozpoznany. Autor jest miłośnikiem twórczości Stanisława Lema, a jego książki są pełne neologizmów oraz wyrazów zupełnie niezrozumiałych i nie posiadających znaczenia. Poniżej podano dwa przykłady takich zwrotów. Są to:

*CHRZĘSKRZYBOCZEK*
*APELAJDA SĘKLIWA*

Na **rys.7** pokazano układy klawiatury dla obu tych zwrotów. Przyjemnego kodowania!

**Mirosław Lach, AVT**