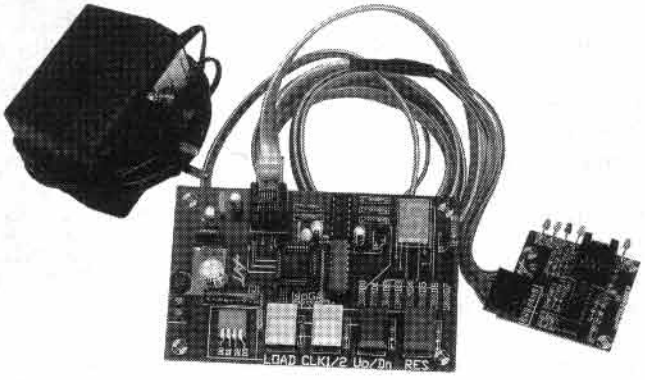
Starter Kit dla układów ispGAL, część 2

KITY EDUKACYJNE



W pierwszej części artykułu przedstawiliśmy Czytelnikom układ ispGAL22V10 *oraz konstrukcję opracowanego w laboratorium AVT zestawu dydaktycznego. Niezbędnym dodatkiem do zestawu AVT-300 jest programator układów* isp *- to właśnie jemu poświęcimy drugą część artykułu. Przedstawiony przez nas programator wraz z dołączonym oprogramowaniem sterującym można wykorzystać także jako samodzielne urządzenie do programowania innych układów isp. Dzięki zastosowaniu tego prostego urządzenia znacznie łatwiej (i taniej...) jest zacząć niezwykłą przygodę z układami PLD.*

Zaczynamy od krótkiego opisu programu „zaszytego" w dostarczanym wraz zestawem AVT-300 układzie ispGAL22V10.

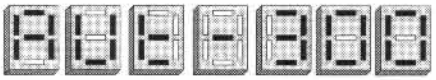
Program demonstracyjny

W celu ułatwienia uruchomienia zestawu każdy z dostarczonych wraz z kitem AVT-300 układów ispGAL22V10 jest wstępnie zaprogramowany.

Układ ispGAL22V10 skonfigurowano jako licznik z możliwością wyboru kierunku zliczania (góra/dół), zmiany częstotliwości zliczania impulsów i asynchronicznego kasowania. Wyjścia licznika sterują bezpośrednio segmentami wyświetlacza Stany wyjściowe układu liczącego dobrano tak, że na wyświetlaczu 7-segmentowym pojawiają się kolejno stany jak na **rys.1**. Program demonstracyjny przedstawiono na **list.1**. Napisany został w języku CUPL.

Przycisk UP/DN powoduje zmianę kierunku zliczania układu (napis „przewijany" jest do przodu lub do tyłu), przycisk RES powoduje asynchroniczne kasowanie licznika (wyświetlacz gaśnie), a przy pomocy przycisku CLK1/2 można zmienić szybkość zliczania licznika. Przełącznik LOAD oraz DIP-switch nie mają żadnego wpływu na pracę układu.

Wykorzystanie wstępnie zaprogramowanego układu ispGAL22V10 znacznie ułatwi uruchomienie układu i znalezienie ewentualnych błędów montażu.



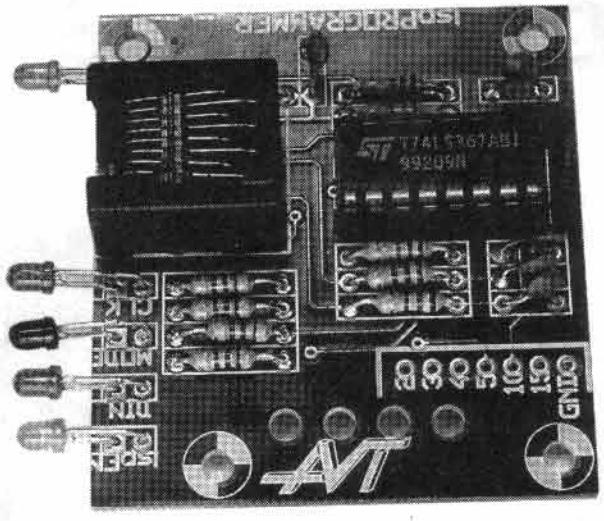
Rys. 1. Efekt działania programu demonstracyjnego.

```
NAME demo_isp;
REV 1.01;
DATE 29/03/95;
DESIGNER Piotr Zbysinski;
DEVICE G22V10LCC;

/* Deklaracja sygnalow wejsciowych */
pin 2 = CLK;
pin 3 = CLK1;
pin 4 = CLK2;
pin 12 = !UD;
pin 13 = CLK_SEL;
pin 16 = RES;

/* Deklaracja sygnalow wyjsciowych */
pin 26 = CLK_OUT;
PIN 27 = SEL;
PIN [25,24,23,21,20,19,18,17] =
[DP,G,F,E,D,C,B,A];

/* Opis logiczny ukladu              */

CLK_OUT = !CLK_SEL & CLK1
        # CLK_SEL & CLK2;
SEL = 'b'1;
[DP,G,F,E,D,C,B,A].ar = RES;
[DP,G,F,E,D,C,B,A].sp = 'b'0;
field COUNT = [DP,G,F,E,D,C,B,A];

$define S0 'b'00000000
$define S1 'b'11110111
$define S2 'b'00111110
$define S3 'b'11111000
$define S4 'b'01000000
$define S5 'b'11001111
$define S6 'b'00111111
$define S7 'b'10111111

sequence COUNT {
present S0 if UD      next S1;
                      if !UD     next S7;
present S1 if UD      next S2;
                      if !UD     next S0;
present S2 if UD      next S3;
                      if !UD     next S1;
present S3 if UD      next S4;
                      if !UD     next S2;
present S4 if UD      next S5;
                      if !UD     next S3;
present S5 if UD      next S6;
                      if !UD     next S4;
present S6 if UD      next S7;
                      if !UD     next S5;
present S7 if UD      next S0;
                      if !UD     next S6;
}
```

Listing 1. Program demonstracyjny napisany w języku CUPL.

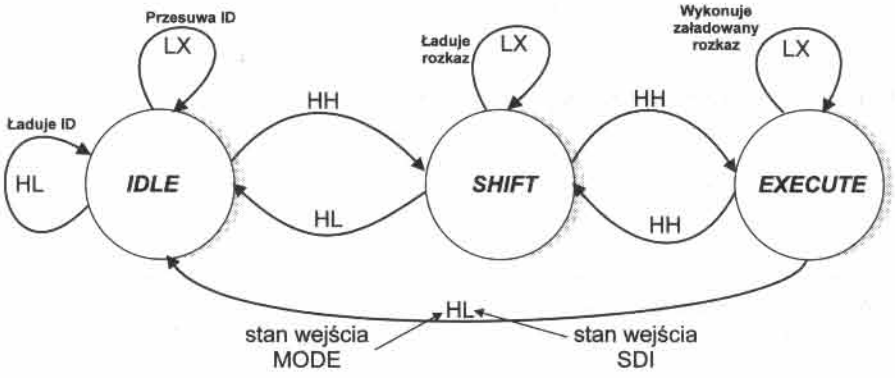
Programator układów *isp*

kit AVT-300P



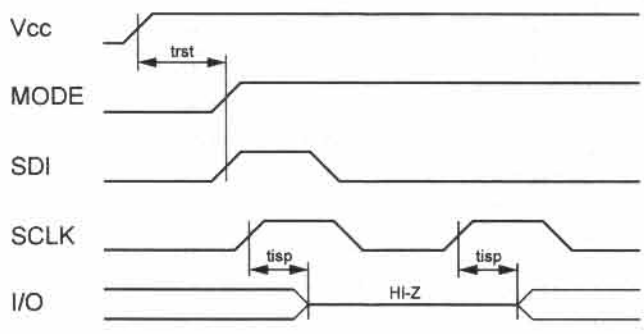
Jak zostało już wcześniej wspomniane, układy serii *isp* (ang. In System Programmability) opracowane przez Lattice'a, wyposażone są w interfejs szeregowy służący do ich programowania bezpośrednio w uruchamianym systemie. Jest to interfejs prosty w obsłudze, ale niestety nie w pełni zgodny z dotychczas stosowanymi standardami. Większość układów programowalnych lub rekonfigurowalnych w systemie wykorzystuje interfejs JTAG. Ta wada jest jednocześnie zaletą - ominięcie standardu umożliwiło opracowanie interfejsu mało wymagającego od otoczenia sprzętowego i programowego.

Praca interfejsu opiera się na wbudowanym w układ prostym dwubitowym automacie o trzech stanach, który pracuje zgodnie z algorytmem z **rys.2**. Przejścia pomiędzy kolejnymi stanami są wyznaczane przez sygnał zegarowy SCLK (dokładniej - przez jego narastające zbocze), a ich kolejność zależy od stanu sygnałów na wejściach MODE i SDI. Tak więc wejście SDI spełnia dwie funkcje - jest wejściem danych wpisywanych szeregowo do wewnętrznych rejestrów układu *isp* oraz wejściem sterującym pracą automatu. Funkcję spełnianą przez wejście SDI określa poziom logiczny na wejściu MODE.

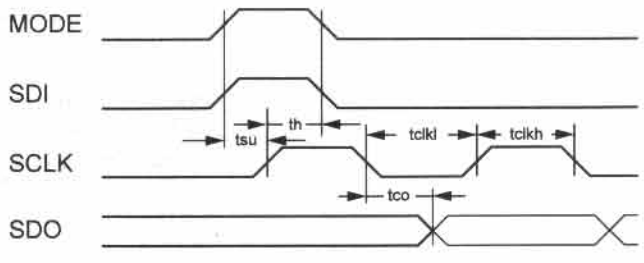


Rys. 2. Algorytm pracy automatu sterującego interfejsem isp.

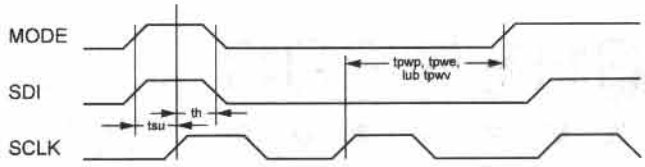
Automat sterujący interfejsem ma trzy stany pracy:
- IDLE, domyślny stan „spoczynkowy", w którym układ *isp* pracuje zgodnie wpisaną w strukturę konfiguracją,
- SHIFT, podczas którego dane z wejścia SDI wpisywane są do 5-bitowego rejestru rozkazów, dzięki czemu możliwe jest elastyczne sterowanie pracą układu. Rozkazy dopuszczalne dla układów ispGAL22V10 zestawiono



Rys. 3. Przebiegi charakterystyczne dla trybu programowania.



Rys. 4. Przebiegi charakterystyczne dla trybu SHIFT.



Rys. 5. Czasy trwania impulsów programowania, weryfikacji i kasowania.

Tabela 1.

Instruction	Operation	Description
00000	NOP	Bez operacji.
00010	SHIFT_DATA	Przesuwanie zawartości rejestru danych.
00011	BULK_ERASE	Kasowanie wewnętrznej pamięci.
00101	ERASE_ARRAY	Kasowanie pamięci konfiguracji matrycy logicznej.
00110	ERASE_ARCH	Kasowanie rejestru opisującego architekturę OIMC.
00111	PROGRAM	Programowanie matrycy pamięci.
01010	VERIFY	Przepisanie rejestru danych zawartością wybranego wiersza pamięci.
01101	IOPRLD	Załadowanie rejestrów I/O danymi.
01110	FLOWTHRU	Ominięcie rejestru przesuwczego (SDI=SDO).
10100	ARCH SHIFT	Przesuwanie zawartości rejestru konfiguracji architektury.

w **tab.1** wraz z odpowiadającymi im kodami i krótkim opisem,
- EXECUTE, który umożliwia wykonanie kroków przypisanych uprzednio wprowadzonemu do układu rozkazowi.

Poprawne funkcjonowanie wbudowanego w układ interfejsu szeregowego wymaga sterowania sygnałami o odpowiednio dobranych parametrach czasowych. W przypadku oferowanego przez nas rozwiązania za poprawne dobranie czasu trwania impulsów sterujących odpowiada program obsługujący układ programatora. Na **rys.3..5** przedstawione zostały wykresy definiujące podstawowe parametry czasowe sygnałów sterujących. Wartości dopuszczalnych czasów trwania poszczególnych impulsów zawarto w tab.2.

We wnętrzu układu ispGAL22V10 wbudowano cztery rejestry przesuwne, których zadaniem jest sprzętowa obsługa interfejsu programującego (**rys.6**). Każdy z tych rejestrów ma za zadanie konwertować informacje podawane na wejście SDI do postaci równoległej. Długość rejestrów zależy od przeznaczenia (od 5 do 138 bitów).

Rejestr oznaczony na rysunku jako ID jest rejestrem tylko do odczytu i zawiera kod typu układu, tzw. elektroniczną sygnaturę. Jest ona wykorzystywana do automatycznej identyfikacji programowanego układu. Sygnaturę można odczytać tylko w trybie IDLE.

Rejestr rozkazu ma długość pięciu bitów i służy do odebrania kodu rozkazu z programatora (zgodnie z tab.1). Jest on dostępny tylko w trybie SHIFT.

Rejestr adres/dane ma długość 138 bitów. Dzięki niemu możliwe jest wprowadzenie danych przeznaczonych do wpisania w strukturę układu oraz odebranie danych z jego wnętrza. Najstarszych sześć bitów zawiera adres wiersza, którego dotyczy informacja zapisana na pozostałych 132 bitach. Rejestr adres/dane dostępny jest w trybie EXECUTE.

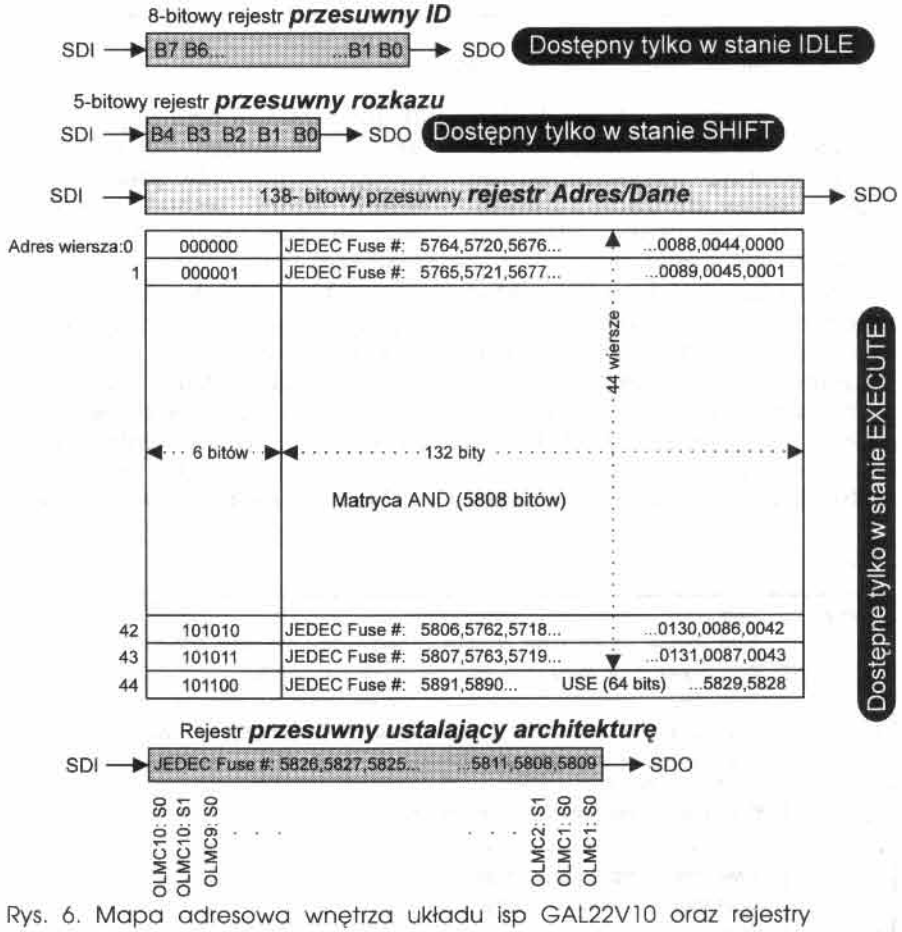
Ostatnim wewnętrznym rejestrem układu ispGAL22V10 jest 20-bitowy rejestr konfiguracyjny, który odpowiada za ustalenie trybu pracy makrocel OLMC (po dwa bity - S0 i S1 - na każdą z dziesięciu OLMC). Ten rejestr jest także dostępny w trybie EXECUTE.

Tak więc dzięki wbudowaniu we wnętrze układu ispGAL specjalizowanego automatu wykonującego podawane z zewnątrz rozkazy, możliwe okazało się wykonanie prostego w obsłudze interfejsu. Ogromną zaletą wszystkich układów *isp* oferowanych przez Lattice'a jest także nowoczesna technologia produkcji struktur, która pozwala na wielokrotne (powyżej 10000 cykli) programowanie układu zasilanego napięciem 5V, bez konieczności stosowania przetwornicy podwyższającej napięcie.

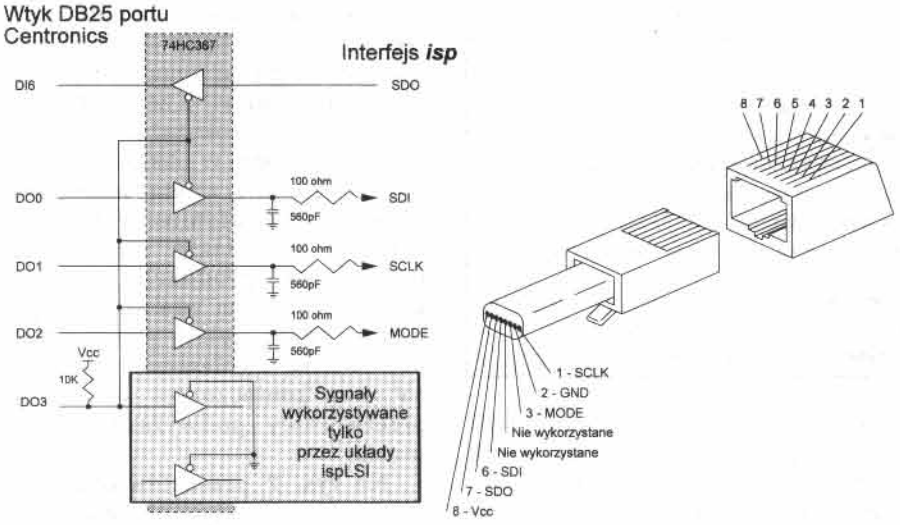
Na **rys. 7** przedstawiono zalecane przez producenta rozwiązanie interfejsu - programatora oraz kolejność wyprowadzeń w złączu interfejsu (gniazdo RJ45).

Opis układu

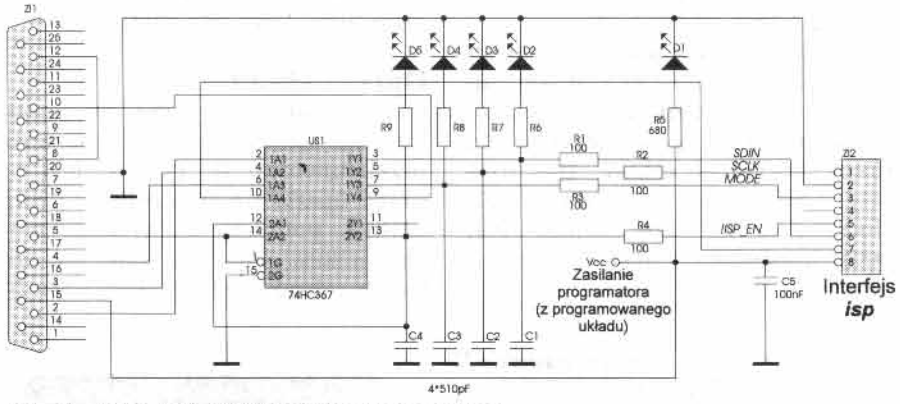
Schemat elektryczny proponowanego przez nas rozwiązania przedstawiono na **rys. 8**. Jest to układ oparty na zaleceniach firmy Lattice, poddany niewielkim modyfikacjom. Polegają one na dodaniu diod świecących D1..5,



Rys. 6. Mapa adresowa wnętrza układu isp GAL22V10 oraz rejestry umożliwiające transmisję danych.



Rys. 7. Zalecany przez producenta układ programatora dla układów isp.



Rys. 8. Schemat elektryczny programatora.

dzięki którym możliwe będzie analizowanie stanu interfejsu. Diody monitorują wszystkie istotne sygnały szyny sterującej i danych. Dioda D1 sygnalizuje włączenie zasilania programatora.

Jak widać na schemacie, układ elektryczny programatora jest bardzo prosty. Za konwersję plików przygotowanych przez dowolny kompilator logiczny (CUPL, PALASM, ABEL, TANGO PLD, ORCAD PLD, itp.) do postaci *isp*STREAM (akceptowanej przez układy ispGAL) oraz odpowiednie sterowanie pinami portu drukarkowego odpowiada program sterujący (który wchodzi w skład zestawu AVT-300P). Rola przedstawionej przez nas konstrukcji sprowadza się do buforowania sygnału przekazywanego do i z komputera.

Układ montujemy na płytce drukowanej przedstawionej na wkładce, a rozmieszczenie elementów przedstawia **rys. 9**.

Montaż układu nie powinien sprawić żadnej trudności, pewnej uwagi wymagać będzie przylutowanie przewodów do złącza DB-25 (na schemacie rys. 8 podane zostały numery wyprowadzeń tego złącza!) i do punktów lutowniczych na płytce drukowanej. Na powierzchni płytki wykonane zostały cztery dodatkowe otwory o średnicy ok. 3..4mm, które są przeznaczone do wykonania przeplotu przewodu wychodzącego z płytki do złącza DB-25. Wykonanie tego przeplotu zapobiega możliwości wyrwania lub ułamania przewodu podczas eksploatacji urządzenia. Pomocą podczas montażu może być **rys.10**.

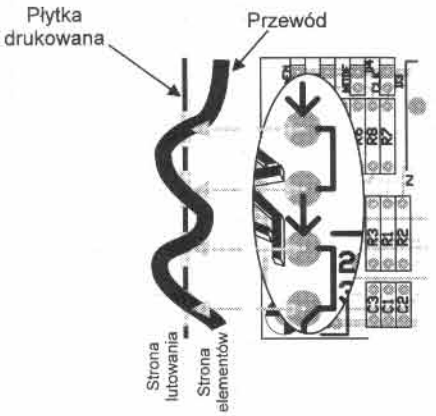
Konstrukcja płytki została opracowana w taki sposób, aby jak najbardziej ułatwić montaż mechaniczny układu. Cztery otwory wykonane blisko rogów płytki można wykorzystać do przymocowania płytki do dna dowolnej obudowy plastikowej lub metalowej. Jedynym trudnym do samodzielnego wykonania elementem obudowy mogą być otwory na diody świecące i gniazdo RJ45.

Długości przewodów łączących programator ze złączem drukarkowym oraz pomiędzy programatorem i programowanym układem powinny być jak najmniejsze. Sprawdzono, że układ pracuje poprawnie z przewodami o długości nawet 1 metra (każdy), co w zupełności wystarcza w większości typowych aplikacji.

Przewód łączący programator z programowanym układem powinien być 8-żyłowy z zaciśniętymi na obu końcach wtykami telefonicznymi. Wtyki te powinny być zaciśnięte w taki sposób, aby przewód dołączony po jednej stronie do pinu 1 wtyku łączył się także z pinem 1 drugiego wtyku.

Tabela 2.

Nazwa	Opis	Min.	Max.	Jedn.
t_{rst}	Czas odstępu od włączenia zasilania do rozpoczęcia procedury programowania.	1	-	µs
t_{isp}	Czas przejścia ze stanu IDLE do uaktywnienia portów I/O.	-	10	µs
t_{su}	Czas odstępu pomiędzy uaktywnieniem SDI lub MODE do zbocz SCL.	100	-	ns
t_h	Czas nakładania się impulsu SCLK ze stanami aktywnymi SDI lub MODE.	100		ns
t_{co}	Czas opóźnienia pojawienia się informacji na wyjściu SDD w stosunku do opadającego zbocza SCLK.	150	nsi	
t_{clkh}	Czas trwania poziomu "1" sygnału SCLK.	0,5		µs
t_{clkl}	Czas trwania poziomu "0" sygnału SCLK.	0,5		µs
t_{pwp}	Czas trwania impulsu programującego.	40	100	ms
t_{pwe}	Czas trwania impulsu kasującego.	200		ms
t_{pwv}	Czas trwania impulsu weryfikującego	5		µs

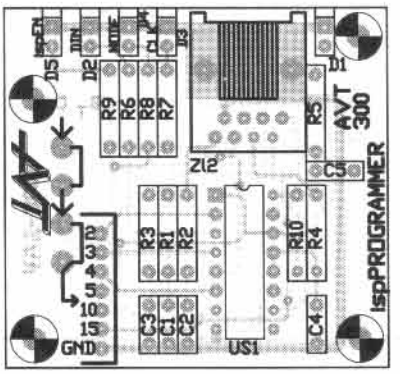


Rys. 10. Sposób umocowania przewodu na płytce drukowanej.

Inne możliwości układu

Firma Lattice produkuje następujące rodziny układów *isp*:

- ispLSI1000/2000/3000 - są to układy o średniej i dużej skali integracji, przeznaczone do stosowania w zaawansowanych aplikacjach. Są one stosunkowo drogie, wymagają także specjalistycznego oprogramowania projektowego,
- ispGAL22V10 - ścisły odpowiednik funkcjonalny standardowego układu GAL22V10 w obudowie PLCC28. Niezwykle atrakcyjna alternatywa dla popularnych struktur o znacznie większej elastyczności niż GAL16V8 i GAL20V8,
- ispGDS (ang. isp Generic Digital Switch) - elektroniczne przełączniki *isp*, coraz częściej stosowane w miejsce mechanicznych DIP-switchy m.in. w kartach komputerowych Plug&Play, zdalnie programowanych układach pomiarowych, interfejsach SCSI, itp.



Rys. 9. Rozmieszczenie elementów na płytce drukowanej.

Opisany przez nas programator wraz z oprogramowaniem obsługuje dwie ostatnie rodziny układów. Wybrano je ze względu na niewielką cenę i dostępność oprogramowania projektowego. Układy ispGDS są wyposażone w identyczny jak ispGAL22V10 interfejs, dzięki czemu programowanie tych dwóch rodzin układów nie wymaga przebudowy oprogramowania ani sprzętu.

Piotr Zbysiński, AVT

W programie sterującym pracą programatora wykorzystano procedury ispCODE firmy Lattice.

WYKAZ ELEMENTÓW:

Rezystory
R1, R2, R3, R4: 100Ω
R5, R6, R7, R8, R9: 680Ω

Kondensatory
C1, C2, C3, C4: 510pF
C5: 100nF

Półprzewodniki
D1, D2, D3, D4, D5: Diody LED 3mm
US1: 74HC367

Różne
ZI1: Wtyk męski DB-25 + obudowa
ZI2: Gniazdo RJ-45 (8 styków)
Wtyki RJ45 (8 stykowe) 2szt.
Kabel płaski 8-żyłowy 1m
Przewód 7-żyłowy 40cm