Współpraca OrCADa z AutoTraxem

Artykuł ten powstał pod wpływem uwag Czytelników, jakie napłynęły do redakcji po opublikowaniu na łamach EP opisu AutoTraxa. Skarżą się oni na brak informacji na temat możliwości automatyzacji pracy z tym programem. W artykule przedstawiamy jeden z najprostszych sposobów wymiany informacji pomiędzy edytorem schematów OrCAD SDT, a edytorem płytek drukowanych Traxedit.

Zaczynamy od schematu

Zawsze musimy mieć schemat elektryczny urządzenia, do którego przygotujemy płytkę. Niewątpliwą zaletą AutoTraxa, w porównaniu z jego wersją freeware EasyTraxem, jest występowanie w menu głównym polecenia *Netlist*. Polecenie to zostało silnie rozbudowane i zapewnia m.in. wykorzystywanie listy połączeń generowanej przez program Netlist z pakietu OrCAD SDT. Z tej możliwości trzeba obowiązkowo korzystać. Brak ochoty na opanowanie tej sztuki sprowadza nas do poziomu EasyTraxa. Zakupmy więc stosowne numery „Elektroniki dla Wszystkich", gdzie był prowadzony kurs projektowania przy pomocy tego programu.

Schemat elektryczny możemy przygotować przy pomocy dowolnego edytora schematów. Może to być popularny OrCAD SDT, oryginalny edytor „Protela for DOS" lub dowolny inny edytor. Warunkiem jest to, aby edytor schematów miał możliwość wytworzenia listy połączeń w formacie czytelnym dla AutoTraxa. W artykule pokażemy, jak niemal automatycznie otrzymać listę połączeń (netlistę) z OrCADa, która będzie „strawna" dla AutoTraxa.

Format listy połączeń AutoTraxa

Przypomnijmy zatem format netlisty AutoTraxa. Składa się ona z dwóch części (**list. 1**):
- opisu podzespołów,
- opisu węzłów.

Na list. 1 jest przedstawiony przykładowy fragment takiej listy połączeń. Jest to lista połączeń płytki drukowanej zamka szyfrowego.

Opis pojedynczego podzespołu w pierwszej części listy zawiera się pomiędzy nawiasami kwadratowymi [..] i składa się z trzech pól. Pole to ciąg znaków zakończony znakiem końca linii, a więc jest to jedna linijka tekstu.

W pierwszym polu jest zawarta nazwa i numer oznaczenia podzespołu. Nazwa podzespołu jest jego wyróżnikiem w całym projekcie i nie może się już powtórzyć (np. R1, US2 itp.).

Pole drugie zawiera nazwę bibliotecznego prototypu obudowy.

W trzecim polu opisu podzespołu jest podana wartość parametru lub jego typ, jak np. wartość rezystancji, pojemności, typ tranzystora, diody, układu scalonego itp.

Część opisowa węzłów definiuje poszczególne połączenia. Bazuje na pierwszej części netlisty, korzystając z nazw podzespołów. Pod pojęciem węzła należy rozumieć punkty układu o jednakowym potencjale elektrycznym. Oznacza to, że węzeł jest wydzielonym obszarem miedzi, do którego mają być dołączone konkretne wyprowadzenia podzespołów.

Opis pojedynczego węzła jest zawarty pomiędzy nawiasami zwykłymi (...) Tutaj też można wyróżnić pola, jako pojedyncze linie tekstu. Liczba pól jest zmienna, zależna od liczby końcówek podzespołów dołączonych do węzła.

Pierwszym polem opisu węzła jest zawsze jego nazwa, która jest niepowtarzalna w całym projekcie. Nazwa ta jest tworzona przez program usługowy edytora schematów, wytwarzający listę połączeń. Jest ustalana dwojako: narzuca ją program tworzący listę połączeń lub jest wzięta ze schematu.

Pierwszy sposób jest stosowany do tych węzłów projektu, które nie mają swojej nazwy na schemacie. Sposób drugi jest stosowany do węzłów posiadających oryginalną nazwę na schemacie. Nazwą węzła może być węzeł zasilania układów cyfrowych (VCC, VDD), węzeł masy (GND, VSS), linie sygnałowe tworzące szyny sygnałowe (np. AD0, AD1) itp. Jak to może być pomocne, przekonamy się przy omawianiu prowadzenia ścieżek - polecenie *Netlist-Information* jest bardziej czytelne.

Następne pola opisu węzła to zakodowane nazwy końcówek poszczególnych podzespołów. Sposób zakodowania jest banalny: do przecinka (albo znaku myślnika „-") jest to nazwa podzespołu, a po nim jest nazwa końcówki tego podzespołu. Specjalnie napisałem „nazwa", ponieważ wcale nie musi to być jej numer.

AutoTrax dopuszcza czteroznakowe nazwy końcówek, przy czym nie przyjmuje on nazw zaczynających się od znaku myślnika „-". Jest to oczywiste, ponieważ ten znak jest znakiem rozdzielającym nazwę podzespołu od nazwy jego końcówki. Dlatego nazwa -DC będzie zinterpretowana błędnie.

Czteroznakowe nazwy końcówek pozwalają na opis nie tylko liczbowy. Na przykład na list. 1 węzeł N00001 zawiera podzespół o nazwie D1 i końcówce ANOD, co sugeruje

anodę diody. Stosowanie opisowych nazw końcówek niektórych podzespołów znakomicie ułatwia rozpoznawanie końcówek. Gdyby tam były numery, byłoby to trudniejsze.

Liczba pól końcówek podzespołów jest zmienna, zależnie od stopnia rozbudowy węzła, a opis węzła kończy się znakiem zamykającego nawiasu „)".

Generacja listy połączeń

Generacja listy połączeń polega na uruchomieniu programu albo podprogramu usługowego, który w oparciu o poprawnie narysowany schemat przygotuje listę połączeń w żądanych formacie. Okazuje się jednak, że określenie formatu listy jeszcze nie załatwia całej sprawy.

Pokażemy teraz, jak wytworzyć listę połączeń przez pakiet OrCAD SDT. Skorzystamy z nieco starszej wersji OrCADa, wersji 3.2x. Różnica pomiędzy tą wersją, a następnymi polega na bardziej rozbudowanym interfejsie użytkownika w wersjach nowszych.

Po pierwsze, narysujmy przykładowy schemat. Jest on przedstawiony na **rys. 1**. Dokonajmy analizy tego schematu pod kątem przypisania rodzajów obudów poszczególnym podzespołom.

W układzie z rys. 1 mamy rezystory o jednakowych i różnych wartościach. Wszystkie są jednak tej samej mocy: maksymalnie 0,25W, mają więc jednakowe wymiary.

Układ nie jest skomplikowany, wymiary płytki nie są w żaden sposób narzucone, zastosujemy więc montaż poziomy rezystorów. Typowym rozstawem nóżek dla rezystorów 0.25W jest 10mm, czyli 400mils. Niektórzy przyjmują rozstaw 7,5mm (300mils), ale ta wartość zawęża gamę dostępnych rezystorów do mocy 1/8W i 1/6W. Jeśli nie ma takiej potrzeby, warto przyjąć 10mm.

Podobnie możemy sklasyfikować kondensatory, z tą różnicą, że musimy je rozdzielić na kondensatory elektrolityczne i kondensatory z dielektrykiem stałym. Kondensatory z dielektrykiem stałym mają standardowy rozmiar 5mm (200mils) i taki przyjmiemy w naszym projekcie.

Kondensatory elektrolityczne będą stać pionowo i zastosujemy obudowy kondensatorów przeznaczonych do montażu pionowego.

Mikrokontroler PIC16C84 może występować w kilku rodzajach obudów, jedną z nich jest typ DIL (18-nóżkowa) i taką przyjmiemy.

Układ stabilizatora LM7805 jest w typowej obudowie TO220. Ponieważ stabilizator zasila tylko procesor i diody LED, więc pobór prądu jest na tyle mały, że nie ma potrzeby umieszczania stabilizatora na radiatorze. Stabilizator będzie stał pionowo i miejsce na radiator nie będzie przewidziane.

Diody LED będą miały średnicę 5mm, a więc przyjmiemy, że rozstaw nóżek wynosi 2,5mm (100mils). Diody te będą ustawione pionowo.

Przekaźnik PK1 jest przekaźnikiem samochodowym i ma tylko jedną obudowę, nie ma więc kłopotów z jej wybraniem.

Jako złącza JP2 i JP3 przyjmiemy złącza typu ARK2 o rozstawie nóżek 5mm. Złącze JP1 będzie rzędem złoconych pinów, takiego bowiem wymaga taśma klawiatury membranowej.

Tranzystor Q1 ma tylko jeden rodzaj obudowy i nie potrzebuje radiatora.

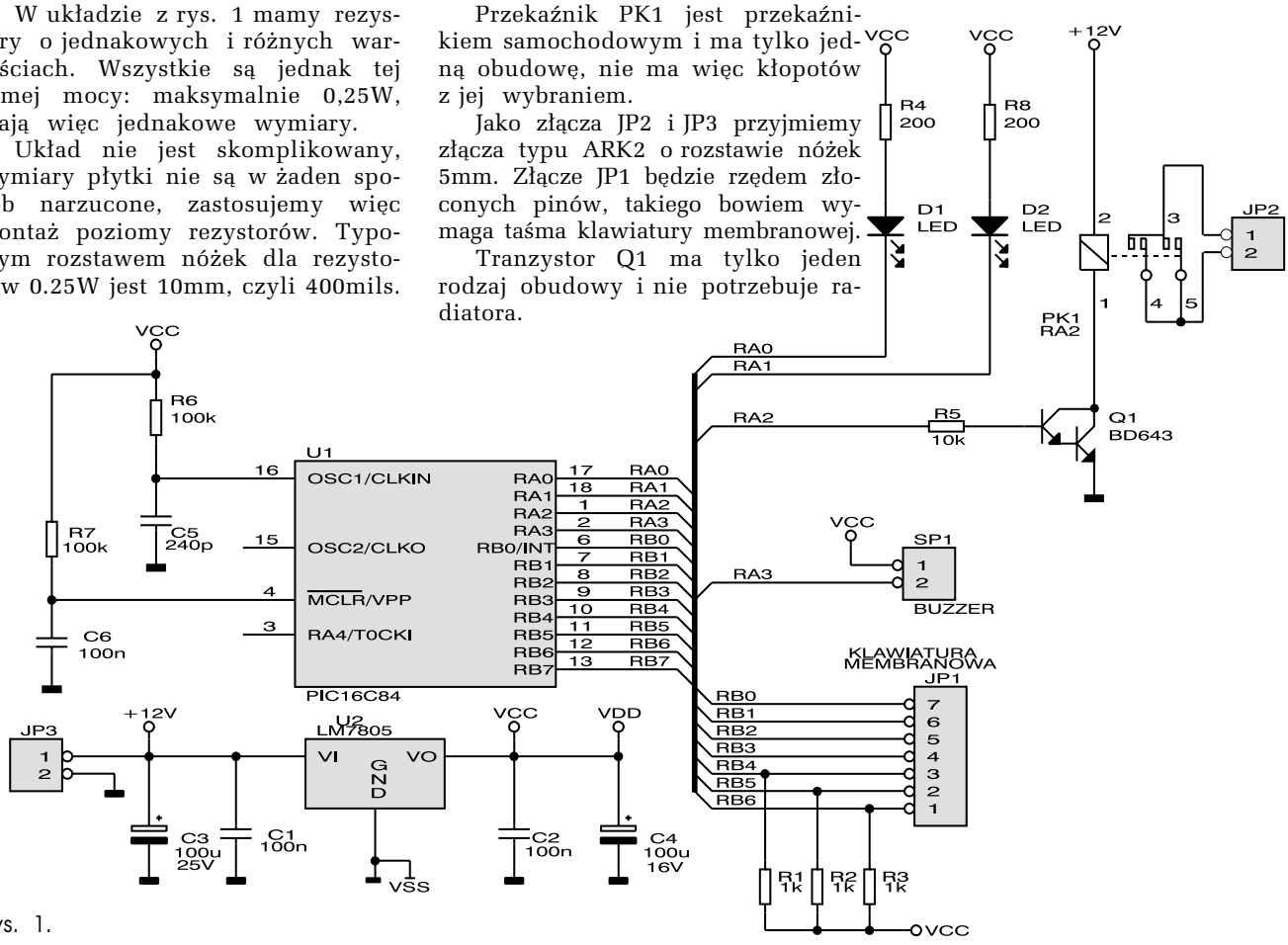
Wyjście buzzera SP1 to dwa punkty lutownicze na płytce, do których może być przylutowana para przewodów buzzera.

Przyglądając się powyższej wyliczance widać, że liczba rodzajów obudów potrzebnych dla naszego projektu jest znacznie mniejsza niż liczba podzespołów.

OrCAD zapewnia półautomatyczne przypisywanie obudów do podzespołu. Do tego celu służą dodatkowe pola opisu podzespołu - *Part Fields*. Autor przyjął, że polem zawierającym nazwę obudowy będzie pierwsze pole, które przemianował w konfiguracji pakietu na Module.

Pola Part Fields służą ponadto do przechowywania dodatkowych informacji o podzespole. I tak:

- pole 2, nazwane *Voltage/Power*, przechowuje informację o napięciu przebicia kondensatorów albo mocy rezystorów;
- pole 3, nazwane *Tolerance*, przechowuje informację o tolerancji wartości podzespołów;
- pole 4, nazwane *Kind of Mount*, przechowuje informację na temat podstawowego sposobu montażu;
- pole 5, nazwane *Standing*, zawiera informację o ustawieniu podzespołu.



Rys. 1.

Listing 1. Fragment przykładowej listy połączeń dla schematu zamka szyfrowego.

```
[                1K               ]                (
R6                                [                RA3-1
R402510                           U2               SP1,2
100K                              LM78XXT          U1,2
                 ]                LM7805           )
                 [                                 (
                 R4                                RB7-1
]                R402510                           U1,13
[                200              ]                )
R7                                [                (
R402510                           U1               RB0-1
100K                              18DIP300         JP1,7
                 ]                PIC16C84         U1,6
                 [                                 )
                 R8                                (
]                R402510                           RB1-1
[                200              ]                JP1,6
C1                                [                U1,7
C203010                           PK1              )
100N                              RA2              (
                 ]                RA2              RB2-1
                 [                                 JP1,5
                 C5                                U1,8
]                C203010                           )
[                240P             ]                (
C2                                (                RB3-1
C203010                           N00001           JP1,4
100N                              R4,2             U1,9
                 ]                D1,ANOD          )
                 [                )                (
                 JP2              (                RB4-1
]                H2X200           N00002           R1,1
[                2X1              R8,2             JP1,3
C6                                D2,ANOD          U1,10
C203010                           )                )
100N                              (                (
                 ]                N00003           RB5-1
                 [                PK1,3            R2,1
                 JP3              JP2,1            JP1,2
]                H2X200           )                U1,11
[                2X1              (                )
C3                                N00004           (
CE2040                            PK1,1            RB6-1
100U                              Q1,COLL          R3,1
                 ]                )                JP1,1
                 [                (                U1,12
                 JP1              N00005           )
]                7SIP100          PK1,4            (
[                7X1              PK1,5            +12V
C4                                JP2,2            JP3,1
CE1530                            )                C3,2
100U                              (                C1,1
                 ]                N00006           U2,VI
                 [                R5,2             PK1,2
                 Q1               Q1,BASE          )
]                CE30V            )                (
[                BD643            (                GND
R5                                N00007           C4,1
R402510                           R6,2             C2,2
10K                               U1,16            U2,GND
                 ]                C5,1             C1,2
                 [                )                C3,1
                 SP1              (                JP3,2
]                2SIP100          RA0-1            C6,2
[                BUZZER           U1,17            C5,2
R1                                D1,CATH          Q1,EMIT
R402510                           )                U1,5
1K               ]                (                )
                 [                RA1-1            (
                 D1               U1,18            VCC
                 CO26             D2,CATH          R1,2
]                LED              )                R2,2
[                                 (                R3,2
R2                                RA2-1            U2,VO
R402510                           U1,1             C2,1
1K               ]                R5,1             C4,2
                 [                )                SP1,1
                 D2               (                R7,1
                 CO26             N00011           R6,1
]                LED              R7,2             R8,1
[                                 U1,4             R4,1
R3                                C6,1             U1,14
R402510                           )                )
```

Pola 2-5 są najczęściej wykorzystywane do opisu rezystorów i kondensatorów, chociaż mogą być także częściowo wypełnione dla innych podzespołów.

Do pola *Module* każdego podzespołu pracowicie wpisuje się nazwę obudowy. Oczywiście, wpisywanie nazwy tej samej obudowy dla np. 50 rezystorów jest zajęciem nudnym, zmuszającym do skorzystania z innego sposobu, dostępnego w OrCA-Dzie. Do automatycznego przypisania obudów podzespołom służy w pakiecie OrCAD program FLDSTUFF. Jego wywołanie jest następujące:

FLDSTUFF <plik ze schematem> <numer pola> <plik przypisań>

Znaczenie poszczególnych parametrów powyższego polecenia jest następujące:

<*plik ze schematem*> - jest to nazwa pliku ze schematem;

<*numer pola*> - pola *PartField* są numerowane cyframi od 1 do 8; pole *Module* będzie więc miało numer 1;

<*plik przypisań*> - jest to nazwa pliku tekstowego przygotowanego pod dowolnym edytorem tekstowym znaków ASCII (np. Norton Editor) o dowolnym rozszerzeniu.

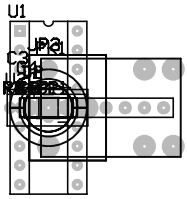
Na **list. 2** przedstawiono format zapisu w takim pliku. Każda linia tego pliku składa się ze wzorca wartości podzespołu, umieszczonego pomiędzy pierwszą parą apostrofów, minimum jednego znaku rozdzielającego, którym jest tutaj spacja albo znak tabulacji, i nazwy prototypu obudowy, umieszczonej między drugą parą apostrofów. Brak tych apostrofów powoduje, że program FLDSTUFF wykryje błąd i nie będzie mógł wykonać przypisania.

To, czy dla danego podzespołu zostanie przypisana nazwa obudowy zależy od konfiguracji pakietu. Otóż w deklaracji *FLDSTUFF Combine for Field 1* możemy zdefiniować złożenie pól opisu podzespołu w taki sposób, aby mogły jednoznacznie wskazywać na obudowy w pliku przypisań. Na przykład, niech na schemacie z rys. 1 w polu *Kind of Mount* będzie wpisane „PN“ oraz w polu Standing również „PN“. Pola te są wygaszone, stąd ich treść nie jest widoczna. Kondensator C3 jest kondensatorem elektrolitycznym. „PN“ w tych polach oznacza za pierwszym razem, że podstawowym sposobem montażu tego kondensatora jest montaż pionowy, zaś w kolejnym polu oznacza to, że ten kondensator będzie zamontowany pionowo. W wyżej wymienionej deklaracji wpiszemy *FLDSTUFF Combine for Field 1 V2345*.

Taki zapis oznacza, że złożenie ciągów znaków z pól *Part Value* (temu właśnie służy pierwsza litera V) oraz pól od drugiego do piątego będzie tworzyć ciąg znaków, który

List. 2. Przykład treści pliku przypisań dla programu FLDSTUFF

```
'100u10VPZPZ'  'CE958540'
'100u16VPNPN'  'CE1530'
'100u25VPNPN'  'CE2040'
'100u40VPNPN'  'CE2053'
'100u63VPNPN'  'CE2053'
'100u63VPNPZ'  '4UM22V40'
'150u16VPNPN'  'CE1530'
'150u35VPNPN'  'CE2040'
'220u10VPNPN'  'CE2053'
'220u16VPNPN'  'CE2053'
'220u16VPZPZ'  'CE1408050'
'220U25VPNPN'  'CE2053'
'220u25VPZPZ'  'CE1408050'
'470u40VPNPN'  'CE2053'
'470u40VPNPZ'  '4UM22V40'
```



Rys. 2.

będzie poszukiwanym wzorcem w pliku przypisań.

W naszym przykładzie wzorcem będzie ciąg znaków *100u25VPNPN*. Program FLDSTUFF zaczyna poszukiwać tego wzorca w pliku przypisań. Zawartość pliku przypisań pokazano na list. 2. Znalezienie wzorca w pliku przypisań powoduje, że jeżeli pole 1 danego podzespołu jest puste, tam zostanie zapisany tekst znajdujący się między drugą parą apostrofów, czyli w naszym przykładzie będzie to *CE2040*.

Oczywiście puste pola wzorca zapisu *V2345* będą potraktowane na zasadach ogólnych, czyli w procesie tworzenia wzorca nie będą brane pod uwagę, a więc np. puste pole 2 w naszym przykładzie spowoduje, że wzorcem będzie *100uPNPN*.

Autor przygotował kilkanaście plików przypisań, dotyczących typowych przypisań, takich jak:
- rezystory o różnej mocy i tolerancji,
- kondensatory bipolarne,
- kondensatory elektrolityczne w różnych wariantach ustawienia,
- podzespoły aktywne dyskretne: tranzystory, diody itp,
- serie układów scalonych: analogowych, cyfrowych różnych serii, mikroprocesory itp.

Pliki te są dostępne na stronie WWW Wydawnictwa AVT. Ponieważ pliki przypisań są nierozerwalnie związane z prototypami obudów, dołączono do nich pliki biblioteczne obudów z AutoTraxa. Prototypy obudów są dla projektanta bardzo cenne, bo przez lata używania zostały „wyczyszczone" z pewnych niedoróbek bibliotek standardowych.

Wiadomo, że nie ma wśród wymienionych plików przypisań takiego, który byłby uniwersalny. Dlatego program FLDSTUFF musi być uruchamiany kilkakrotnie. Jednak zdecydowanie przyspiesza pracę, szczególnie, gdy mamy do czynienia ze schematem układu cyfrowego.

O tym, czy zostały już przypisane wszystkie obudowy możemy dowiedzieć się wykorzystując do tego celu program PARTLIST, służący do wytworzenia listy podzespołów potrzebnych do zmontowania płytki.

W konfiguracji pakietu, w deklaracji *PARTLIST Part Value Combine* możemy wpisać *V2345->1*, co spowoduje, że wartość podzespołu w liście elementów będzie złożeniem *Part Value*, pól od 2 do 5, znaków „->„ i pola 1. Taką listę podzespołów dla naszego przykładu przedstawiono na **list. 3**.

Po każdym wywołaniu programu FLDSTUFF uruchamiamy program PARTLIST, z którego plik wynikowy możemy wylistować i zobaczyć, do jakich podzespołów trzeba jeszcze przypisać obudowy. Oczywiście nazwy obudów podzespołów nietypowych wpisujemy ręcznie, korzystając z edytora schematów DRAFT.

Kiedy program PARTLIST wytworzy raport o podzespołach, w którym wszystkie pola prototypu podzespołu będą wypełnione, możemy przystąpić do generacji listy połączeń.

Polecenie DOS wygląda następująco: *NETLIST <nazwa pliku ze schematem> <nazwa pliku z listą połączeń> TANGO/S*.

W naszym przykładzie to polecenie będzie następujące: *NETLIST CODELOCK.SCH CODELOCK.NET TANGO/S*.

Klucz /S oznacza tu, że lista połączeń będzie wytworzona w formacie specjalnym, za taki bowiem jest uważany format Tango. Czytelnicy w bardzo prosty sposób mogą podejrzeć, ile i jakich formatów wyróżnia program NETLIST, jeżeli zamiast TANGO/s napiszą na przykład AKUKU/S. Jedynym formatem niespecjalnym jest format EDIF, jednak dla nas nieprzydatny.

Najczęściej spotykane błędy popełniane przez początkujących projektantów, a dotyczące generacji listy połączeń, są następujące:

✓ Rysowanie poza rastrem. Jest to błąd poważny, bo trudny do wychwycenia na ekranie monitora. Pomóc może włączenie klucza /G w poleceniu generacji listy połączeń. Dla naszego przykładu polecenie będzie następujące: *NETLIST CODELOCK.SCH CODELOCK.NET TANGO/S/G*. Zostanie wtedy wytworzony plik o nazwie CODELOCK.GRD, który jest raportem zawierającym listę obiektów schematu, które leżą poza rastrem. Dla schematu bez takich przypadków będzie to plik o długości zerowej.

✓ Przeciągnięcie linii połączeniowej i nałożenie jej na wyprowadzenie podzespołu. Program NETLIST wychwytuje ten błąd i wskazuje jego położenie, podając współrzędne.

✓ Zdublowana numeracja podzespołów. Zdublowanie numeracji podzespołów następuje wtedy, kiedy dokonujemy ręcznej renumeracji podzespołów. Na przykład, na schemacie są dwa różne rezystory R12. Innym przypadkiem jest zdublowana numeracja podzespołów wielokrotnych. OrCAD dopuszcza obiekty biblioteczne, jako wielokrotnie powtarzające się fragmenty układu, np. bramki logiczne zamknięte w jednej obudowie. Przykładem niech będzie powtarzająca się nazwa U2A, która oznacza, że ta sama bramka występuje w co najmniej dwóch różnych miejscach układu. Podobnie jest z niekonsekwentnym wartościowaniem podzespołów. Jeśli np. zdarzy się, że bramka jednej kostki raz jest typu 7400 i jest to U1A, zaś w innym miejscu schematu mamy U1B jako 7410, to jest to błąd, który będzie zasygnalizowany przez program NETLIST.

✓ Puste pola *Part Value* i *Reference*. Program nie dopuszcza pustych pól *Part Value* i *Reference*. Zawsze muszą być wypełnione tekstem. Początkujący projektanci, nie znający zbyt dobrze OrCADa, chcąc wygasić te pole, po prostu kasują ich zawartość. Żeby te pola nie były drukowane i nie były widoczne na ekranie, należy odpowiednio ustawić atrybut widoczności każdego pola opisu podzespołu. Wystarczy w tym celu podczas edycji podzespołu wyłączyć atrybut *Visible* i po sprawie.

✓ Błędnie zbudowane opisy podzespołów w bibliotekach. Początkujący projektanci budują biblioteki podzespołów nie wypełniając pól opisu wyprowadzeń podzespołów. Powstaje niejednoznaczność w opisie listy podzespołów i dlatego jest sygnalizowany błąd.

Wprowadzanie listy połączeń do projektu

Przebrnęliśmy przez procedurę tworzenia listy połączeń, która może być odczytana przez program TRAXEDIT.

List. 3. Lista podzespołów z widocznym polem obudowy

```
Revised:       January 3, 1997
Revision:
Bill Of Materials
January 3, 1997     10:57:41
Page    1

Item Quantity  Reference  Part
-----------------------------------------------
  1       3    C1,C2,C6   100n->C203010
  2       1    C3         100u25VPNPN->CE2040
  3       1    C4         100u16VPNPN->CE1530
  4       1    C5         240p->C203010
  5       2    D1,D2      LED->CO26
  6       1    JP1        7x1->7SIP100
  7       2    JP2,JP3    2x1->H2x200
  8       1    PK1        RA2->RA2
  9       1    Q1         BD643->CE30
 10       3    R1,R2,R3   1k->R402510
 11       2    R4,R8      200->R402510
 12       1    R5         10k->R402510
 13       2    R6,R7      100k->R402510
 14       1    U1         PIC16C84->18DIP300
 15       1    U2         LM7805->LM78XXT
```

List. 4. Zawartość przykładowego pliku raportu o błędach wprowadzania listy połączeń.

```
REPORT OF MISSING PARTS FROM NETLIST   C:\DESIGN\MLACH\KOKO\CODELOCK.NET
_____________________

MISSING COMPONENTS FROM NETLIST LOAD : 1
_____________________
Q1
```

Teraz uruchamiamy więc program TRAXEDIT. Jeśli znamy wymiary płytki, zaczynamy od narysowania obrysu płytki za pomocą polecenia *Place|Track* i ewentualnie *Place|Arc*. Obrys płytki wykonujemy na warstwie *Board Layer*. Jest to ważne, bowiem tylko warstwa *Board Layer* może wystąpić na wszystkich kliszach poszczególnych warstw.

Teraz sprawdzamy czy odpowiednia biblioteka jest otwarta. Do tego celu używamy polecenia *Library|File*. Jeśli biblioteka jest otwarta, dostaniemy jej nazwę do ponownego otwarcia. Zmiana biblioteki nastąpi po wprowadzeniu jej nowej nazwy. Jeśli nie pamiętamy nazwy, możemy użyć znaku zapytania „?“. Wtedy w kolejnym okienku pojawi się lista plików bieżącego katalogu o rozszerzeniu *.LIB. Ustawiamy kursor w miejscu, w którym chcemy rozmieścić obudowy. Obudowy zostaną umieszczone we wskazanym miejscu, jak na stosie.

Włączamy polecenie *Netlist|Auto, Place|Load Components From Netlist*. Następuje zapytanie o nazwę pliku z listą połączeń. Obowiązkowym rozszerzeniem tego pliku jest *.NET. Program proponuje nazwę pliku, która była użyta podczas poprzedniej sesji projektowej. Jeśli projektujemy nową płytkę, to jest niemal pewne, że nazwa ta nie będzie nam odpowiadać. Autor wprowadza wtedy znak zapytania ?, co jest w tym programie zachętą do przedstawienia wszystkich plików z rozszerzeniem .NET, jakie znajdują się w bieżącym katalogu.

Po wprowadzeniu nazwy pliku program przejmuje kontrolę nad procesem wprowadzania listy połączeń. Pierwszym etapem jest pobranie prototypów obudów z otwartej biblioteki. Ten etap kończy się raportem w postaci otwartego okienka COMPONENT LOAD, które ma trzy linijki tekstu. Tekst ten zawiera dane statystyczne odnośnie procesu ładowania podzespołów (Ach, ci Amerykanie, oni lubują się w statystykach!) i ma on następujące znaczenie:

- *Component Loaded* - liczba faktycznie nowo wprowadzonych podzespołów;
- *Missing Patterns* - liczba nieistniejących podzespołów w bieżąco otwartej bibliotece;
- *Existing Components* - liczba wcześniej ustawionych podzespołów, przed wywołaniem polecenia *Netlist|Auto Place|Load Components From Netlist*.

Liczba określana jako Missing Patterns jest użyteczna w czasie ładowania podzespołów tylko z jednej biblioteki i tylko do nowego projektu, bowiem daje ona od razu informację o brakach w bibliotece lub o błędach w specyfikacji listy połączeń.

Kolejny, drugi etap jest poświęcony analizie połączeń między nóżkami podzespołów. Tutaj też zachodzi optymalizacja połączeń w węzłach. Określona grupa punktów tworzących węzeł elektryczny, może być połączona na jeden z trzech sposobów:

- *Shortest Path* - według najmniejszej odległości pomiędzy punktami węzła;
- *X Bias* - według najmniejszej odległości ze względu na współrzędną X;
- *Y Bias* - według najmniejszej odległości ze względu na współrzędna Y.

Teraz program się zatrzymuje, oczekując na naciśnięcie klawisza ENTER. Ostatni etap jest podsumowaniem listy połączeń i jest on dla nas najważniejszy.

Kolejne okienko zawiera następujące informacje:

- *Nets Loaded* - liczba wprowadzonych pętli (obwodów) elektrycznych;
- *Missing Patterns* - liczba nie wprowadzonych, brakujących podzespołów z powodu niewystępowania ich w bibliotece;
- *Missing Pins* - liczba błędnie nazwanych punktów lutowniczych we wprowadzonych podzespołach.

Jeżeli jeden z dwóch ostatnich parametrów (*Missing Patterns* lub *Missing Pins*) jest niezerowy, program proponuje zapis raportu o błędach. Rozszerzenie pliku z taką informacją jest *.REP.

W praktyce, niespecjalnie obchodzą nas te statystyczne dane, które pojawiają się w kolejnych okienkach. Naciskamy trzy razy ENTER i czekamy czy nie pojawi się propozycja zapisu komunikatu o błędach. Niezależnie od tego czy w liście połączeń występują błędy czy nie, na ekranie pojawiają się te podzespoły, które udało się dołączyć do projektu, w tym także te, które mają błędnie opisane wyprowadzenia. W przykładzie, dotyczącym zamka szyfrowego, efekt naszych działań widać na **rys. 2**.

Biblioteka, z której były czerpane prototypy podzespołów nie zawierała jednego z nich. Został więc wytworzony plik raportu CODELOCK.REP (**list. 4**). Plik ten składa się z dwóch części. Pierwsza z nich zawiera listę nie znalezionych podzespołów, zaś druga część jest listą tych wyprowadzeń podzespołów, których nazwy nie zgadzają się z nazwami opisanymi w liście połączeń.

Już wprowadzone podzespoły (rys. 2) zapamiętujemy na dysku, najlepiej pod tą samą nazwą co schemat (oczywiście pliki te powinny być różne, o tym decyduje rozszerzenie: *.SCH i *.PCB).

Po przeczytaniu zawartości pliku z raportem o błędach, poprawiamy nazwę na schemacie albo do biblioteki dołączamy brakujący podzespół.

Ponownie uruchamiamy TRAXEDIT

Zanim znowu uruchomimy znane nam polecenie wprowadzania listy połączeń, musimy skasować istniejącą w projekcie listę połączeń. Czynimy to za pomocą polecenia *Netlist|Clear*. Po jego wykonaniu pamięć zajmowana przez dotychczasową listę połączeń zostanie zwolniona. Jeśli nowa lista połączeń będzie wprowadzona bez usuwania starej listy, to pamięć przeznaczona dla starej listy nie zostanie zwolniona, zaś dostęp do niej nie będzie możliwy. Innymi słowy pamięć ta będzie utracona. Po kilku czy kilkunastu takich operacjach, w czasie tej samej sesji projektowej, może się okazać, że raptem zabrakło pamięci na wprowadzenie kolejnej modyfikacji listy połączeń.

Po użyciu *Netlist|Clear* nic już nie stoi na przeszkodzie, aby ponownie użyć polecenia *Netlist|Auto Place|Load Components From Netlist*.

Kiedy już uda się nam wprowadzić wszystkie obudowy na płytkę, zaczynamy rozmieszczanie podzespołów.

Mirosław Lach, AVT

Pliki przypisań są dostępne w Internecie pod adresem:
http://www.avt.com.pl/avt/ep/download.htm.