2018

Układy scalone

CZĘŚĆ II: O PROJEKTOWANIU UKŁADÓW SCALONYCH

WIESŁAW KUŹMICZ

METODY PROJEKTOWANIA, STYLE PROJEKTOWANIA, WERYFIKACJA, SYMULACJA, EDYTOR TOPOGRAFII, FULL CUSTOM, KOMÓRKA STANDARDOWA, MATRYCA BRAMKOWA, FPGA

W TEJ CZĘŚCI OMAWIANE SĄ RÓŻNE ASPEKTY PROJEKTOWANIA UKŁADÓW SCALONYCH. PRZEDSTAWIONY JEST TYPOWY PRZEBIEG PROCESU PROJEKTOWANIA, OMAWIANE SĄ PROBLEMY PROJEKTOWANIA (PRACOCHŁONNOŚĆ I PODATNOŚĆ NA BŁĘDY) I SPOSOBY PRZEZWYCIĘŻANIA TYCH PROBELMÓW. OMAWIANE SĄ KOMPUTEROWE NARZĘDZIA WSPOMAGANIA PROJEKTOWANIA ORAZ RÓŻNE SPOSOBY PROJEKTOWANIA UPROSZCZONEGO I ZAUTOMATYZOWANEGO, ZWANE STYLAMI PROJEKTOWANIA. CĘŚĆ TA KOŃCZY SIĘ ĆWICZENIEM PRAKTYCZNYM W PROJEKTOWANIU W STYLU FULL CUSTOM.

Spis treści

1 Wstęp: o czym tu będzie mowa

Droga Czytelniczko, Drogi Czytelniku: teraz będzie mowa o projektowaniu układów scalonych. Najpierw o tym, jakie mogą być uzasadnienia dla zaprojektowania własnego układu specjalizowanego, potem o tym, jak się do tego zabierać, a w końcu o tym, jakie są metody, komputerowe narzędzia wspomagające i style projektowania. Na koniec samemu spróbujesz zaprojektować pojedyncze tranzystory.

2 Układy specjalizowane – po co i jakie

A co to jest specjalizowany układ scalony? Zdefiniujemy to w sposób ścisły.

DEFINICJA

Specjalizowany układ scalony (Application-Specific Integrated Circuit – w skrócie ASIC) jest to układ zaprojektowany do konkretnego urządzenia lub klasy urządzeń elektronicznych, dla konkretnego odbiorcy, nie przeznaczony do sprzedaży na rynku dowolnemu odbiorcy.

2.1 Po co układy specjalizowane

Specjalizowane układy scalone znajdziemy dziś praktycznie w każdym wyrobie elektronicznym. Dlaczego? Dwa główne powody wymienione są niżej.

2.1.1 Indywidualizacja wyrobu

W początkowym okresie rozwoju mikroelektroniki układy scalone były projektowane wyłącznie przez ich producentów – firmy półprzewodnikowe. Konstruktorzy sprzętu mogli jedynie wybierać z katalogów firm półprzewodnikowych najbardziej odpowiadające im układy. Szybko okazało się, że z punktu widzenia zastosowań jest to duża niedogodność, ograniczająca kreatywność konstruktorów sprzętu. Nieco upraszczając, można dać taki przykład: jeśli kilku różnych producentów telewizorów skonstruowało swoje telewizory przy użyciu tego samego zestawu układów scalonych firmy XXX, to telewizory te z punktu widzenia funkcji i parametrów technicznych były takie same lub prawie takie same. Funkcje i parametry techniczne były bowiem określone przez taką, a nie inną konstrukcję użytych układów scalonych. Fakt ten jest dobrze znany tym, którzy interesują się komputerami PC. Jeśli wiadomo, że w płycie głównej firmy AAA użyto zestawu układów (zwanego z angielska *chipset*) BBB, to mniej więcej wiadomo, jakie są możliwości i parametry tej płyty.

A przecież na konkurencyjnym rynku producenci starają się, by ich wyroby wyraźnie odróżniały się od wyrobów konkurentów, by oferowały użytkownikowi nowe funkcje i lepsze parametry użytkowe! Odpowiedzią na to dążenie są właśnie specjalizowane układy scalone, powszechnie znane jako układy ASIC. Układów ASIC nie znajdziemy w handlowych katalogach. Producenci sprzętu elektronicznego dzięki układom zaprojektowanym wyłącznie do produkowanych przez nich wyrobów mogą uzyskać przewagę konkurencyjną oferując wyroby odróżniające się na plus pod względem funkcji i/lub parametrów od wyrobów konkurentów.

2.1.2 Ochrona własności intelektualnej

Bardzo ważnym argumentem na rzecz stosowania układów ASIC jest potrzeba ochrony własności intelektualnej. Urządzenie zbudowane z elementów katalogowych, które każdy może kupić, jest łatwe do skopiowania przez konkurenta (w Polsce w ostatnich latach było sporo takich przypadków). Ochrona patentowa nie zawsze jest możliwa. Urządzenie z układem specjalizowanym jest praktycznie nie do skopiowania. Sam układ można wprawdzie „rozszyfrować” i zaprojektować podobny, ale rozszyfrowywanie układu scalonego jest tak pracochłonne i kosztowne, że w praktyce nieopłacalne. Ponadto projekt układu specjalizowanego można zarejestrować w urzędzie patentowym (procedura jest bardzo prosta i niewiele kosztuje). Podlega on wówczas

ochronie przed bezprawnym skopiowaniem. Ochronie podlega projekt topografii układu scalonego. Topografia jest to – mówiąc w skrócie – projekt masek produkcyjnych układu (będzie jeszcze o tym mowa dalej). Ochronie nie podlega schemat elektryczny ani logiczny układu (chyba że są tam rozwiązania oryginalne mające zdolność patentową – ale patentowanie to inny rodzaj ochrony).

Dlaczego topografia? Ponieważ jest to końcowy rezultat procesu projektowania układu. Skopiowanie topografii umożliwia produkcję kopii układu bez ponoszenia kosztów projektowania, a te mogą być dla dużych, skomplikowanych układów bardzo wysokie. Ochrona topografii jest więc ochroną przed nieuczciwą konkurencją. Całkowicie legalnie można kopiować topografię (na przykład wykonać mikrofotografię gotowego układu), jeśli jedynym celem wykonania kopii jest analiza układu, odtworzenie jego schematu czy algorytmu działania. Na podstawie wyników takiej analizy całkowicie legalne jest wykonanie własnego projektu układu o takim samym działaniu, pod warunkiem, że topografia tego układu będzie istotnie różna od topografii oryginalnej. Tak pomyślana ochrona zabezpiecza przed nieuczciwą konkurencją, a równocześnie nie blokuje prac badawczych i rozwoju, który często przecież polega na uczeniu się na wcześniejszych dobrych rozwiązaniach technicznych.

2.2 Droga do układu specjalizowanego

2.2.1 Czy zawsze trzeba zaprojektować własny układ od początku?

Są dwa sposoby uzyskania układu ASIC o potrzebnej nam funkcji. Pierwszy z nich polega na użyciu układu programowalnego. Istnieją dziś układy scalone zawierające w sobie zbiór bramek i bardziej złożonych bloków logicznych oraz programowalną sieć połączeń. Układy te, znane najczęściej pod ogólną nazwą FPGA (skrót od angielskiej nazwy „Field Programmable Logic Array"), bezpośrednio po wyprodukowaniu nie wykonują żadnej konkretnej funkcji. Funkcję tę definiuje użytkownik poprzez zaprogramowanie układu. Zaprogramowanie układu określa jego schemat logiczny, a tym samym wykonywaną funkcję. Istnieją także układy programowalne zawierające oprócz bramek i bloków cyfrowych także bloki analogowe oraz przetworniki analogowo-cyfrowe i cyfrowo-analogowe. Takie układy znane są pod nazwą PSoC (skrót od angielskiej nazwy „Programmable System on Chip" – skrót zastrzeżony przez firmę Cypress).

Drugi sposób uzyskania układu ASIC polega na zaprojektowaniu go od początku i zleceniu produkcji firmie półprzewodnikowej. Istnieje wielu producentów specjalizujących się w produkcji układów scalonych na zamówienie, według projektu klienta. Takie firmy są na Dalekim Wschodzie, w USA, a także w Europie, m.in. w Austrii, Belgii, Francji, Niemczech. Wśród producentów najłatwiej dostępnych dla polskiej firmy można wymienić: TSMC (Tajwan – największy na świecie producent typu „*silicon foundry*"), UMC (Tajwan), STMicroelectronics (Francja), AMS (Austria), AMIS (Belgia), XFAB (Niemcy), IHP (Niemcy), GlobalFoundries (firma światowa, w Europie fabryka w Dreźnie). Do wszystkich tych firm projekty można kierować za pośrednictwem instytucji pośredniczącej, jaką jest EUROPRACTICE (http://www.europractice-ic.com). EUROPRACTICE jest konsorcjum powołanym i częściowo finansowanym przez Unię Europejską. Szczegółowe informacje najprościej uzyskać przez internet. Można tam znaleźć wszystkie potrzebne informacje o zakresie dostępnych usług, oferowanych technologiach, cenach, procedurze uzyskiwania dostępu do danych technicznych, sposobie przesyłania projektów itp. EUROPRACTICE nie tylko pośredniczy między klientami, a producentami układów, ale także dostarcza (dla instytucji akademickich i badawczych) oprogramowanie do projektowania układów po bardzo niskich cenach i świadczy szereg innych usług. Inną europejską instytucją pośredniczącą jest CMP Service przy Narodowym Instytucie Politechnicznym w Grenoble (https://mycmp.fr). Zakres usług i ich koszt jest podobny jak w EUROPRACTICE. CMP Service nastawia się głównie na obsługę

klientów francuskich (dla których, dzięki finansowaniu przez rząd Francji, ma szczególnie korzystne warunki), ale jest to również, jak EUROPRACTICE, instytucja dostępna dla wszystkich. W roku 2019 CMP Service weszło w skład konsorcjum EUROPRACTICE, ale świadczy usługi również samodzielnie.

Każdy ze sposobów uzyskiwania układów ASIC ma swoje wady i zalety i swój zakres przydatności, o czym będzie mowa dalej. Tutaj skoncentrujemy się na drugim sposobie tworzenia specjalizowanych układów scalonych, tj. na samodzielnym ich projektowaniu.

2.2.2 Kolejne etapy – od pomysłu do produkcji

Gdy potrzebny jest specjalizowany układ scalony, postępujemy następująco:

1. Określamy sposób realizacji układu

Jeżeli układ jest cyfrowy, można rozważyć użycie układu programowalnego (FPGA). Może to być najlepsze rozwiązanie, jeśli potrzeba niewiele egzemplarzy układu, lub jeżeli zależy nam na bardzo szybkim uzyskaniu działającego układu. W polskich warunkach argumentem na rzecz układu programowalnego może być też łatwa dostępność oprogramowania służącego do projektowania i sprzętu służącego do programowania takich układów. Jednak układy programowalne są rozwiązaniem niekonkurencyjnym cenowo w przypadku sprzętu produkowanego w dużych seriach, a ich parametry techniczne (przede wszystkim szybkość) nie dorównują układom specjalizowanym zaprojektowanym do bezpośredniej realizacji w krzemie, o jakich tu opowiadamy.

2. Wybieramy technologię dla projektowanego układu

Najpierw ze wszystkich dostępnych technologii wybieramy te, które pozwalają osiągnąć cel techniczny. Jeśli takich technologii jest kilka, kierujemy się względami ekonomicznymi. Będzie o tym mowa dalej, w punkcie 4.1. Aby móc wybierać technologie, trzeba oczywiście wiedzieć, do jakich technologii możemy mieć dostęp i znać ich dane techniczne oraz ceny. Te informacje uzyskujemy bądź bezpośrednio od producenta, bądź (najczęściej) od instytucji pośredniczącej. Uzyskanie dostępu do danych technicznych charakteryzujących daną technologię wymaga podpisania umowy o poufności (w języku ang. Non-Disclosure Agreement), ponieważ zdecydowana większość producentów traktuje tego rodzaju dane jako poufne. Gdy korzystamy z instytucji pośredniczącej, umowę taką zwykle podpisujemy z tą instytucją. Dla polskich instytucji i firm nie ma tu żadnych barier ani ograniczeń, chociaż oczywiście każdy producent może zastrzec sobie prawo odmowy przekazania informacji poufnych.

3. Decydujemy, kto będzie projektował układ

Jeżeli dysponujemy odpowiednimi umiejętnościami i oprogramowaniem komputerowym, możemy to robić sami. Na świecie bardzo wiele firm produkujących sprzęt elektroniczny projektuje samodzielnie układy specjalizowane do tego sprzętu. W polskich warunkach pewną barierą może być koszt potrzebnego do tego oprogramowania. Projekt można zlecić wyspecjalizowanej firmie projektowej (w Polsce istnieje sporo takich firm, a oprócz nich projekty na zamówienie wykonuje także Instytut Technologii Elektronowej w Warszawie). Jeżeli nie planujemy częstego projektowania nowych układów, zlecenie projektu wyspecjalizowanej firmie może być najlepszym rozwiązaniem (ale i w tym przypadku potrzebna jest elementarna znajomość zagadnień projektowania, by mieć wspólny język z projektantami z firmy projektowej).

Dalsze kroki dotyczą przypadku, gdy projektujemy układ samodzielnie.

4. Wybieramy sposób, w jaki zostanie zaprojektowany układ

Istnieje szereg sposobów uproszczonego i zautomatyzowanego projektowania układów. Będą one omawiane dalej. Każdy z tych sposobów (zwanych także stylami projektowania) ma swoje zalety i wady. Wybór może zależeć także od tego, jakim oprogramowaniem komputerowym dysponujemy.

5. Zamawiamy wykonanie prototypu układu

W tym celu gotowy projekt wysyłamy do instytucji pośredniczącej lub bezpośrednio do producenta. Projekt wysyła się w postaci elektronicznej przez Internet. Równocześnie wysyła się zamówienie. W tym momencie określony jest przybliżony koszt prototypu. Jako prototyp otrzymuje się kilkadziesiąt (zwykle od 10 do 50) egzemplarzy układu w obudowach wybranego rodzaju, nie testowanych. Za dodatkową opłatą można zamówić testowanie - trzeba wówczas równocześnie z projektem układu przesłać program testowania. Można także za dodatkową opłatą otrzymać większą liczbę egzemplarzy układu, zamówić montaż w nietypowych obudowach itp. Instytucja lub firma, do której wysłany został projekt, weryfikuje jego formalną poprawność. Oznacza to zbadanie, czy projekt sporządzono zgodnie z wymaganiami producenta, przede wszystkim z t.zw. geometrycznymi i elektrycznymi regułami projektowania (w języku ang. Design Rules - będzie o nich mowa dalej). Nie jest weryfikowana funkcjonalna poprawność projektu. Za to, czy schemat układu jest poprawny i czy układ będzie poprawnie wykonywał swą funkcję, odpowiada wyłącznie projektant. W razie stwierdzenia, że wymagania producenta nie są spełnione, odsyłana jest lista stwierdzonych błędów (naruszeń reguł) w projekcie. Istnieje możliwość dostarczenia poprawionej wersji projektu, a specjaliści producenta lub instytucji pośredniczącej zwykle pomagają w poprawieniu błędów. Na wyraźne życzenie projektanta może być przyjęty do produkcji układ naruszający niektóre reguły projektowania, jednak wówczas producent ostrzega, że układ najprawdopodobniej nie będzie działał prawidłowo.

6. Otrzymujemy prototypowy układ i badamy jego działanie

Prototypowe egzemplarze układu otrzymujemy zwykle po 3 - 5 miesiącach od wysłania projektu. W praktyce najniższa opłata za wykonanie prototypowych egzemplarzy niezbyt złożonego układu w taniej, mniej zaawansowanej technologii może wynosić kilka tysięcy EURO (za całość zamówienia, czyli kilkadziesiąt egzemplarzy), ale układy duże, wykonane w zaawansowanych, kosztownych technologiach mogą być oczywiście wielokrotnie droższe.

7. Jeśli układ prototypowy spełnia oczekiwania, a potrzeba więcej egzemplarzy, zamawiamy krótką serię lub produkcję masową. Można złożyć takie zamówienie bezpośrednio u producenta (zwykle tylko wtedy, gdy dotyczy ono produkcji w wielkiej serii) lub w instytucji pośredniczącej. Warunki (ceny, terminy) są w każdym przypadku indywidualnie negocjowane. Można uzyskać wcześniej (nawet jeszcze przed zamówieniem prototypu) wstępne oszacowanie kosztu układu w produkcji seryjnej, jeśli znane są podstawowe dane: technologia produkcji, rodzaj układu i jego wielkość, typ obudowy, wielkość serii itp. Pozwala to oszacować ekonomiczną celowość całego przedsięwzięcia. Cena jednego egzemplarza układu w produkcji seryjnej może się wahać w bardzo szerokich granicach, od ułamka EURO do kilkudziesięciu EURO, zależnie od rodzaju układu, a przede wszystkim od długości serii produkcyjnej.

W opisanym wyżej sposobie postępowania przy projektowaniu i zamawianiu specjalizowanych układów scalonych role pełnione przez projektanta i przez producenta są ściśle rozdzielone. Projektanta nie interesuje, jak realizowane są operacje technologiczne, w wyniku których powstaje struktura scalona (choć oczywiście nie zaszkodzi, by miał pewną ogólną wiedzę na ten temat). Zadaniem projektanta jest dostarczenie producentowi projektu topografii układu, czyli mówiąc w uproszczeniu - rysunku wszystkich masek fotolitograficznych, które

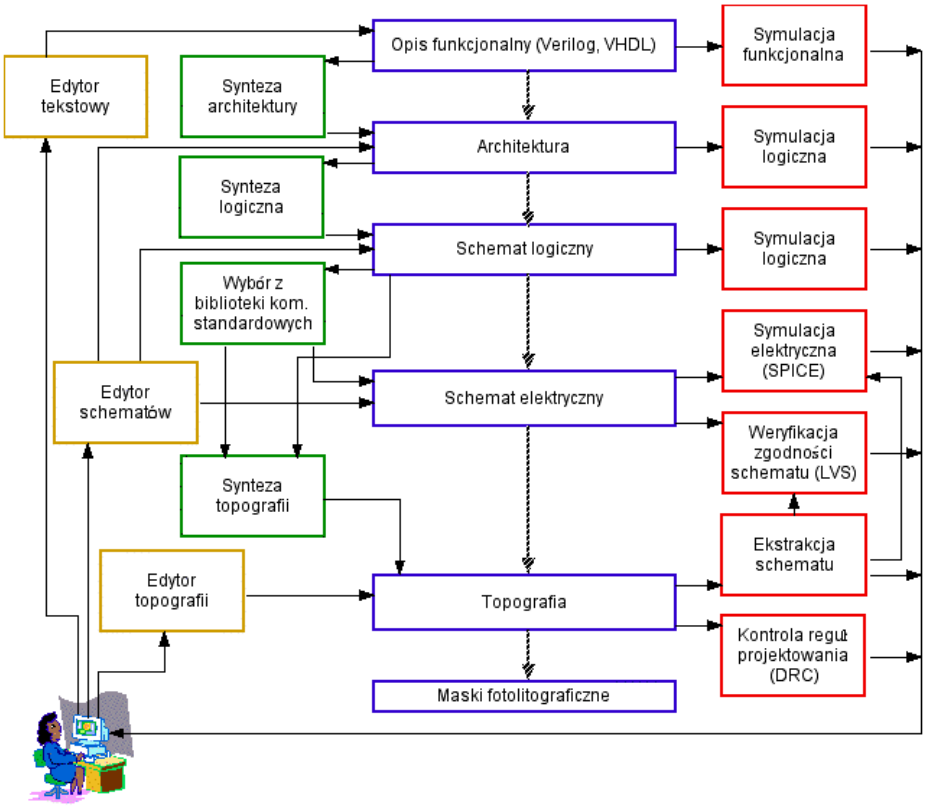
określają strukturę każdego elementu, jego położenie oraz sieć połączeń między elementami. Projekt topografii musi spełniać wymagania producenta zwane regułami projektowania. Producent sprawdza dostarczony projekt wyłącznie pod kątem zgodności z tymi regułami. Nie interesuje go ani funkcja układu, ani jego schemat logiczny i elektryczny. Innymi słowy:

- projektant odpowiada za prawidłowość zdefiniowania funkcji układu, zaprojektowania jego schematu logicznego i elektrycznego oraz struktury fizycznej (czyli topografii), natomiast nie ma wpływu na proces wytwarzania układu (nie może na przykład zażądać wykonania tranzystorów o innych niż typowe parametrach),
- producent odpowiada za prawidłowość wykonania wszystkich operacji technologicznych i wytworzenie struktur półprzewodnikowych, które spełniają normy przewidziane dla danego procesu technologicznego, natomiast nie bierze na siebie żadnej odpowiedzialności za to, czy wytworzone według dostarczonych z zewnątrz projektów układy będą działały zgodnie z intencjami ich projektantów,
- testowanie układu należy do projektanta (lub użytkownika), a nie producenta.

Ten sposób organizacji projektowania i produkcji układów specjalizowanych („fabless design") umożliwił powstanie firm wykonujących na zamówienie projekty układów, a nie dysponujących żadną własną bazą produkcyjną. Firmy takie istnieją także w Polsce.

3 Układy specjalizowane – zarys metod projektowania

Spójrzmy na rysunek 3-1. Ilustruje on w pewnym uproszczeniu ogólny schemat procesu projektowania oraz oprogramowania tworzącego typowy system projektowania. W niebieskich ramkach opisy układu na kolejnych poziomach abstrakcji i etapach projektowania. W czerwonych ramkach programy komputerowe służące do



Rysunek 3-1. Poziomy abstrakcji projektu układu scalonego, narzędzia do projektowania i weryfikacji projektu

weryfikacji projektu (będą one omawiane dalej). W zielonych ramkach programy wykonujące kolejne etapy automatycznej syntezy projektu (przy wykorzystaniu biblioteki komórek standardowych - ten sposób projektowania będzie omawiany dalej). W żółtych ramkach programy komputerowe służące projektantowi do wprowadzania informacji (one także będą omawiane dalej). Cienkie linie pokazują kierunki przepływu informacji.

3.1 Problemy projektowania układów scalonych

Omówimy teraz trzy główne problemy projektowania układów scalonych. Te problemy powodują, że nie można sobie dziś wyobrazić projektowania układów scalonych przy użyciu kartki i ołówka, bez wspomagania odpowiednim oprogramowaniem komputerowym.

3.1.1 Problem pracochłonności

Oszacujmy najpierw ilość wytwarzanej i przetwarzanej w projekcie informacji. Zrobimy to w dość mechaniczny, uproszczony sposób, ale wynik da nam pojęcie o rozmiarach problemu. Załóżmy, że na początku mamy do czynienia z opisem funkcji układu w języku opisu sprzętu (jest to język podobny do klasycznych języków programowania, w którym możemy opisać funkcję i strukturę układu – wspomnimy o tym dalej). Taki opis nawet dla bardzo złożonego układu to kilkaset do kilku tysięcy linii kodu. Traktując ten kod jako zwykły tekst możemy oszacować jego objętość na kilkanaście do kilkudziesięciu kilobajtów. Ostatecznym wynikiem procesu projektowania jest opis masek układu scalonego. Załóżmy, że układ liczy milion tranzystorów MOS (największe projektowane obecnie układy liczą nawet miliard elementów). Aby opisać strukturę tranzystora MOS w układzie CMOS w najprostszy sposób, trzeba zdefiniować położenie i wymiary 6 prostokątów na 4 różnych maskach. Kompletny opis prostokąta (o bokach równoległych do osi układu współrzędnych) wymaga podania 4 liczb (np. współrzędnych lewego dolnego i prawego górnego wierzchołka). A więc jeden tranzystor wymaga opisu złożonego z 24 liczb. Milion tranzystorów to $24*10^6$ liczb. Jeśli są to liczby 32-bitowe, otrzymujemy objętość opisu rzędu 100 MB (w rzeczywistości dużo więcej, bo nie są uwzględnione w naszym rachunku połączenia). Ten sposób szacowania objętości informacyjnej projektu, choć naiwny, daje pojęcie o tym, jaką ogromną ilość informacji trzeba wygenerować i przetwarzać w procesie projektowania. Wiąże się to bezpośrednio z pracochłonnością procesu projektowania. Bez metod wspomagania komputerowego oraz uproszczonych metod projektowania (o których będzie mowa dalej), projekty mające więcej niż kilkaset tranzystorów nie byłyby możliwe do wykonania w rozsądnym czasie i przy rozsądnym koszcie.

3.1.2 Problem błędów w projekcie

Z olbrzymią ilością przetwarzanej informacji wiąże się bezpośrednio problem omyłek i błędów. Statystyki zebrane w różnych obszarach działalności człowieka pokazują, że w swoich działaniach człowiek przy starannej pracy popełnia średnio około 2% pomyłek. Oznacza to, że gdyby człowiek „ręcznie” zaprojektował układ złożony z 1000 tranzystorów, to w przypadku około 20 z nich mielibyśmy do czynienia z omyłkami, np. błędnie doprowadzonymi połączeniami, pomyłkami w wymiarach itp. Zwykle nawet jedna taka omyłka prowadzi do układu, który nie działa lub działa wadliwie. Doświadczenie pokazuje, że nawet w najprostszych układach liczących kilkadziesiąt elementów człowiek nie jest w stanie dostrzec wszystkich popełnionych błędów. Toteż użycie komputerowych narzędzi do weryfikacji poprawności projektu jest konieczne nawet dla bardzo prostych układów.

W procesie weryfikacji poprawności projektu układu scalonego możemy wyróżnić weryfikację formalną i weryfikację funkcjonalną.

Weryfikacja formalna projektu układu polega między innymi na sprawdzeniu, czy:

- projekt masek nie narusza geometrycznych reguł projektowania określonych przez producenta, jak np. minimalne dopuszczalne wymiary obszarów, odstępy między nimi itp.,
- projekt masek układu, opisujący jego topografię, odpowiada zadanemu schematowi elektrycznemu układu.

Pierwszy rodzaj weryfikacji określany jest skrótem DRC (od angielskiego „Design Rule Checking"), drugi - skrótem LVS (od angielskiego „Layout versus Schematic").

Weryfikacja formalna nie zapewnia, że zaprojektowany układ będzie poprawnie działał, bo przecież można sobie wyobrazić, że projekt spełnia wszystkie reguły projektowania, a odczytany z masek schemat jest zgodny z założonym, tylko że ten schemat był od początku błędny. Konieczna jest więc weryfikacja funkcjonalna, której istotą jest zbadanie, jak będzie działał zaprojektowany układ.

Weryfikacja funkcjonalna wykonywana jest metodami symulacyjnymi, ponieważ nie da się zbudować prototypu układu z poszczególnych elementów i poddać go pomiarom. Symulacja działania układu może być wykonywana na kilku poziomach abstrakcji. Wyróżniamy następujące rodzaje symulacji:

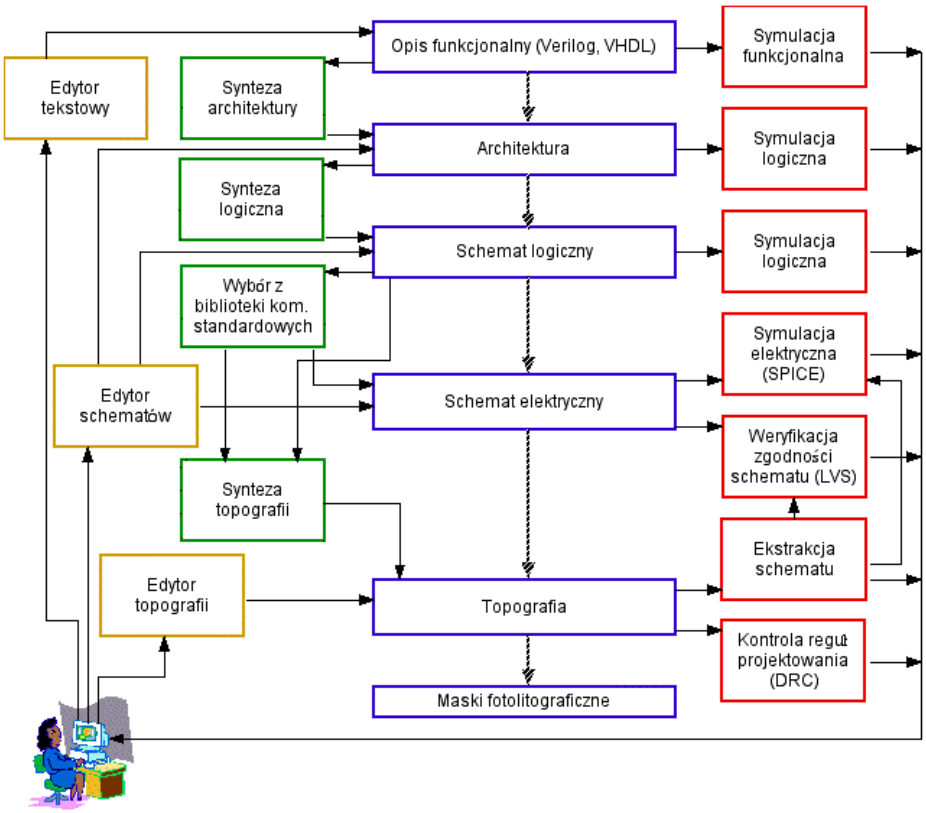
- symulacja elektryczna,
- symulacja logiczna,
- symulacja funkcjonalna.

Jednym ze skutecznych sposobów unikania błędów w projektowaniu jest wykonywanie projektu w sposób zautomatyzowany, przez odpowiednie programy komputerowe (co jest określane angielskim terminem „correctness by construction"). Będzie o tym mowa dalej. Nawet tak wykonany projekt poddaje się jednak wszystkim weryfikacjom. Podkreślmy: każdy projekt, niezależnie od tego, w jaki sposób i przy pomocy jakich narzędzi wspomagających został wykonany, musi być poddany zarówno weryfikacji formalnej, jak i weryfikacji funkcjonalnej. Bardziej szczegółowo jest to omawiane dalej.

3.2 Projektowanie układów scalonych - metody i narzędzia

3.2.1 Etapy procesu projektowania

Dla wygody poniżej powtórzony jest rysunek 3-1 ilustrujący kolejne etapy projektowania.



Rysunek 3-2. Poziomy abstrakcji projektu układu scalonego, narzędzia do projektowania i weryfikacji projektu oraz przepływ informacji w procesie projektowania (powtórzony rysunek 3-1)

Dla układu cyfrowego można wyróżnić następujące etapy projektowania układu

- określenie funkcji układu,
- projekt architektury układu (czyli schematu złożonego z bloków funkcjonalnych),
- projekt logiczny (czyli schematu złożonego z bramek logicznych),
- projekt elektryczny (czyli schematu złożonego z tranzystorów),
- projekt struktury fizycznej układu (topografii układu, czyli projektu elementów, ich rozmieszczenia oraz połączeń między nimi).

W przypadku układów analogowych nie ma oczywiście etapu tworzenia projektu logicznego. Projektowanie rozpoczyna się od zdefiniowania projektu elektrycznego.

Na każdym etapie projektowania dostępne są programy komputerowe wspomagające proces projektowania oraz programy służące do weryfikacji projektu na danym etapie. Wszystkie te programy tworzą zintegrowany system i są powiązane ze sobą poprzez wspólne formaty danych. Istotną częścią składową systemu projektowania są specjalizowane programy do wprowadzania danych: edytory graficzne do graficznego definiowania schematów elektrycznych i logicznych oraz do „rysowania" topografii układu. Potrzebny jest też edytor tekstów do wprowadzania opisów w językach opisu sprzętu i ewentualnie innych informacji tekstowych.

Omówimy teraz krótko poszczególne etapy powstawania projektu.

Definicja funkcji układu

Na tym etapie określa się funkcję wykonywaną przez układ. Jak uczy wieloletnie doświadczenie, na tym, pozornie oczywistym, etapie projektowania popełniane są bardzo często błędy prowadzące do niepowodzenia całego procesu projektowania. Funkcja układu może być zdefiniowana w sposób niekompletny lub wewnętrznie sprzeczny. Zdarza się też, że definicja funkcji układu jest kompletna i spójna, ale nie oddaje dokładnie intencji projektanta lub użytkownika układu, który zamawia projekt i będzie potem wykorzystywać układ. Aby uniknąć takich sytuacji, opracowano specjalne języki, zwane językami opisu sprzętu. Miały one początkowo jedynie uporządkować i sformalizować proces definiowania funkcji układu, pomóc w sprawdzeniu jego kompletności i wewnętrznej spójności. Obecnie pełnią bardzo ważną rolę jako języki wejściowe w systemach zautomatyzowanego projektowania. Będzie o nich mowa dalej.

Projekt architektury układu

Na tym etapie tworzy się projekt układu w postaci powiązanych ze sobą bloków funkcjonalnych o zdefiniowanych funkcjach oraz definiuje się powiązania między blokami i przepływy danych między nimi. Projekt ten można wykonać „ręcznie” i wprowadzić w postaci graficznej lub w postaci opisu tekstowego w języku opisu sprzętu (ta druga możliwość nie jest pokazana na rysunku 3-2). Projekt architektury układu w przypadku układów dużych o złożonych funkcjach ma z reguły budowę hierarchiczną. Cały układ zbudowany jest z niewielkiej liczby dużych bloków funkcjonalnych, każdy z tych bloków ma z kolei swoją wewnętrzną architekturę złożoną z mniejszych bloków o prostszych funkcjach itp. Takich poziomów hierarchii może być wiele. Hierarchiczna budowa układu umożliwia zapanowanie nad całym projektem przez podzielenie go na niewielkie fragmenty, które można projektować mniej lub bardziej niezależnie. Bez takiej strategii projektowania wykonanie dużego i złożonego projektu byłoby praktycznie niemożliwe - projektowanie układu mającego tysiące, a tym bardziej miliony elementów jako jednolitej całości jest niewykonalne.

Projekt schematu logicznego układu

Projekt schematu logicznego polega na wypełnieniu bloków funkcjonalnych schematami zbudowanymi z bramek logicznych lub prostych typowych bloków. Przejście od architektury do schematu logicznego nie jest jednak jednoznaczne, każdą funkcję logiczną można zrealizować na wiele różnych sposobów. Tę fazę projektu można wykonać „ręcznie” i opracowany schemat wprowadzić w postaci graficznej lub w postaci tekstowego opisu w języku opisu sprzętu (ta druga możliwość nie jest pokazana na rysunku 3-2). Projekt schematu logicznego może być też wykonany automatycznie na podstawie projektu architektury układu.

Projekt schematu elektrycznego układu

Ten etap polega na określeniu schematów elektrycznych bramek i bloków występujących w schemacie logicznym. Jeżeli wykorzystuje się typowe, standardowe sposoby realizacji bramek kombinacyjnych, przerzutników itp., to ich schematy elektryczne są znane, a przejście od schematu logicznego do elektrycznego jest niemal automatyczne (i może być zautomatyzowane). W tym etapie określa się jednak także parametry elementów (w przypadku układu CMOS oznacza to określenie wymiarów kanałów tranzystorów MOS). Jeżeli układ ma spełniać wysokie wymagania techniczne (np. szybkość działania), to określenie najlepszych wymiarów tranzystorów jest niełatwe (i nie poddaje się łatwo automatyzacji). W większości przypadków korzysta się jednak z biblioteki standardowych bramek wcześniej zaprojektowanych, sprawdzonych i scharakteryzowanych dla danej technologii produkcji układów. Taką bibliotekę dostarcza producent układów.

Niekiedy wyśrubowane wymagania techniczne zmuszają do użycia niestandardowych rozwiązań bramek logicznych (np. układy zwane logiką dynamiczną; będzie o nich mowa dalej). Wtedy przekształcanie schematu logicznego w elektryczny jest twórczym i często niełatwym zadaniem dla projektanta.

W przypadku projektowania układu analogowego projekt schematu elektrycznego rozpoczyna cały proces projektowania. Nie ma, jak dotąd, ogólnych metod syntezy układów analogowych na podstawie ich opisów funkcjonalnych. Schemat układu analogowego musi być więc opracowany przez projektanta. Istnieją dziś rozszerzenia języków opisu sprzętu pozwalające opisywać nie tylko układy cyfrowe, ale i analogowe.

Projekt topografii układu
To jest etap, w którym powstaje projekt fizycznej struktury układu. Przy projektowaniu „ręcznym" wszystkie elementy są „rysowane" przy pomocy specjalizowanego edytora graficznego przez projektanta, który również określa ich położenie i „rysuje" schemat połączeń (jest to sposób projektowania topografii wspomniany w wcześniej jako projektowanie w stylu *full custom*). Ten sposób projektowania daje największe możliwości optymalizacji topografii układu pod względem zajmowanej powierzchni, długości połączeń i innych kryteriów. Jest jednak nadzwyczaj pracochłonny i podatny na omyłki. W praktyce jest stosowany w projektowaniu układów analogowych, a w przypadku układów cyfrowych wykorzystuje się go do projektowania bramek i małych bloków, które następnie są wykorzystywane jako komórki standardowe (będzie o tym mowa dalej). Projektowanie w taki sposób nie poddaje się automatyzacji, istnieją natomiast inne metody projektowania struktury fizycznej układu umożliwiające automatyczną syntezę topografii – o nich dalej.

Końcowym rezultatem procesu projektowania jest opis masek fotolitograficznych definiujących strukturę fizyczną układu. Opis ten otrzymuje się w jednym z dwóch standardowych języków: CIF lub GDSII. Opis w języku CIF ma postać pliku tekstowego możliwego do analizy i interpretacji przez człowieka, natomiast opis w języku GDSII jest plikiem binarnym. Język GDSII jest powszechnie stosowany w przemyśle. Język CIF wychodzi obecnie z praktycznego użycia, bowiem wszystkie obiekty geometryczne są w nim opisywane na siatce o kroku równym 100 nanometrów, co nie wystarcza do opisu kształtów, jakie występują w układach o minimalnych wymiarach rzędu 100 nanometrów i poniżej.

3.2.2 Narzędzia komputerowego wspomagania projektowania

Minimalny zestaw programów umożliwiający zaprojektowanie układu scalonego składa się z następujących programów:

Programy do wprowadzania danych i tworzenia projektu:

- Edytor tekstowy
- Edytor topografii

Programy do weryfikacji:

- Program do kontroli reguł projektowania (DRC)
- Ekstraktor schematu elektrycznego
- Symulator układów elektronicznych (typu SPICE)

Funkcje tych programów mogą być połączone, na przykład edytor topografii ma często wbudowany moduł umożliwiający kontrolę reguł projektowania, co daje możliwość kontroli na bieżąco, w trakcie tworzenia topografii. Dzięki temu unika się pracochłonnych rozległych przeróbek projektu, jakie mogłyby być konieczne, gdyby naruszenia reguł projektowania zostały wykryte dopiero po zakończeniu projektowania topografii.

Wymienione wyżej programy umożliwiają wprowadzenie opisu schematu elektrycznego w postaci tekstowej (np. w języku wejściowym programu SPICE), wykonanie symulacji, zaprojektowanie topografii (tj. „narysowanie" jej na ekranie komputera w edytorze topografii), skontrolowanie, czy topografia ta spełnia reguły projektowania, odtworzenie (wyekstrahowanie) z topografii schematu elektrycznego, oraz poddanie tego schematu symulacji dla sprawdzenia, czy układ wyprodukowany na podstawie zaprojektowanej topografii będzie działał zgodnie z intencją projektanta. Może to być zarówno układ analogowy, jak i cyfrowy. Dodatkowym ułatwieniem przy projektowaniu może być graficzny edytor schematów, czyli program pozwalający „narysować" na ekranie komputera schemat układu zamiast opisywać go tekstowo. Dużą pomocą przy weryfikacji jest program do sprawdzenia zgodności schematów elektrycznych (Layout versus Schematic - LVS). Pozwala on skontrolować, czy schemat układu opisany tekstowo lub wprowadzony graficznie zgadza się ze schematem odczytanym z topografii przez ekstraktor.

Taki zestaw programów umożliwia projektowanie w stylu *full custom*. W tym przypadku programy komputerowe wspomagają pracę człowieka przy projektowaniu i umożliwiają weryfikację projektu, ale tej pracy nie automatyzują. Jak już wiemy, taki sposób projektowania jest niezwykle pracochłonny. W praktyce możliwe jest projektowanie w ten sposób układów mających do kilkuset elementów. Ten sposób projektowania nie ogranicza jednak w niczym swobody projektanta, dając mu największe możliwości optymalizacji projektu pod każdym względem. Od projektanta wymaga zarówno znajomości zagadnień układowych, jak i zrozumienia struktury fizycznej układu i działania jego elementów.

Warto dodać, że wymieniony wyżej zestaw oprogramowania można skompletować z bezpłatnych programów publicznie dostępnych, co umożliwia naukę projektowania w stylu *full custom* bez kosztownych inwestycji w oprogramowanie komercyjne. Niestety producenci układów nie dostarczają do tych programów plików konfigurujących te programy dla konkretnych technologii, np. plików z modelami tranzystorów czy też plików z regułami projektowania. Zatem w praktyce do projektów układów, które mają być wyprodukowane, trzeba użyć oprogramowania komercyjnego.

W przypadku układów cyfrowych mamy jednak możliwość daleko idącej automatyzacji procesu projektowania. Mamy do dyspozycji następujące programy:

Programy automatyzujące projektowanie

- Program syntezy logicznej
- Programy automatycznej syntezy struktury fizycznej układu

Programy do weryfikacji

- Symulator funkcjonalny
- Symulatory logiczne

Układ cyfrowy może być opisany w języku opisu sprzętu (o tych językach będzie mowa dalej) i poddany symulacji funkcjonalnej w celu sprawdzenia, czy opis jest poprawny, a układ działa zgodnie z intencją projektanta. Służą do tego programy umożliwiające symulację funkcjonalną. Prawidłowy opis funkcjonalny może być poddany syntezie logicznej, której ostatecznym wynikiem jest schemat logiczny układu złożony z bramek oraz ewentualnie większych bloków takich, jak sumatory, układy mnożące czy bloki pamięci. Taki schemat może być w mniejszym lub większym stopniu ulepszany lub uzupełniany przez projektanta, na przykład przy wykorzystaniu edytora schematów. Gotowy do dalszych prac projektowych schemat logiczny poddaje się

weryfikacji przy użyciu symulatora logicznego. Jest kilka rodzajów takich symulatorów. Schemat logiczny może być reprezentowany przy pomocy typowych bramek cyfrowych, które wykonują określoną funkcję logiczną i dodatkowo mogą być scharakteryzowane pod względem parametrów elektrycznych, np. czasów propagacji sygnału (będzie o nich mowa dalej) czy też poboru mocy. Te dane mają oczywiście charakter przybliżony, bo na etapie symulacji logicznej, gdy fizyczna realizacja układu nie jest jeszcze znana, dokładne wartości parametrów elektrycznych nie są określone. Inny rodzaj symulatora logicznego to symulator, w którym elementami są pojedyncze tranzystory, ale traktowane są one jako idealne przełączniki, które przewodzą prąd lub nie. W wielu symulatorach mogą występować zarówno bramki, jak i pojedyncze tranzystory. Istnieją także symulatory umożliwiające symulację mieszaną - część układu może być symulowana logicznie, a część elektrycznie. Są one przydatne przy projektowaniu układów mieszanych, analogowo-cyfrowych.

Gdy schemat logiczny jest gotowy i zweryfikowany przy pomocy symulatorów, można projektować fizyczną strukturę układu. Wiele profesjonalnych systemów projektowania daje tu różne możliwości. Można wygenerować plik danych do zaprogramowania układu programowalnego (FPGA). Możliwe jest wykonanie automatycznej syntezy fizycznej struktury układu przy wykorzystaniu matryc bramkowych lub komórek standardowych. Wkrótce będzie to omawiane dokładniej.

W skład profesjonalnych systemów projektowania wchodzą też zwykle programy do rozwiązywania pewnych specjalnych problemów występujących zwłaszcza przy projektowaniu układów przeznaczonych do wytwarzania w najbardziej zaawansowanych technologiach lub układów, którym stawia się szczególnie wysokie wymagania. Przykłady takich programów to:

- Programy do szacowania poboru mocy
- Programy do projektowania i analizy sieci połączeń zasilania bloków układu
- Programy do projektowania sieci połączeń rozprowadzających sygnały zegarowe
- Programy do generacji testów dla układów cyfrowych

Te programy nie będą dalej omawiane, ale poruszone będą problemy, do rozwiązywania których te programy służą.

Odrębną kategorię systemów oprogramowania do projektowania układów scalonych stanowią systemy projektowania układów mikrofalowych. Krzemowe układy scalone CMOS z powodzeniem mogą pracować do częstotliwości rzędu kilkudziesięciu GHz, a układy bipolarne są przydatne nawet przy częstotliwościach rzędu setek GHz. Te układy i sposoby ich projektowania wykraczają poza zakres naszego wykładu.

Większość programów do projektowania i weryfikacji układów scalonych wymaga danych dotyczących konkretnej technologii, w której układy będą produkowane. Przykładowo, program do kontroli reguł projektowania musi mieć dany zestaw tych reguł, program do symulacji elektrycznej wymaga parametrów modeli elementów, itp. Tego rodzaju dane dostarczane są przez producentów w postaci zestawów plików zwanych plikami technologicznymi (w języku angielskim zwanych „design kit”, co można przetłumaczyć jako „pakiet projektowy”).

3.2.3 Języki opisu sprzętu

Języki opisu sprzętu były początkowo pomyślane jako narzędzie do definiowania projektów w sformalizowany sposób, aby ułatwić uniknięcie luk, omyłek i niespójności. Obecnie dają one znacznie większe możliwości. Projekt układu opisany w jednym z tych języków może być poddany symulacji w celu sprawdzenia, czy układ będzie działał zgodnie z intencją projektanta. Możliwa jest także (choć nie w każdym przypadku) automatyczna

synteza układu - do poziomu projektu logicznego, a nawet do poziomu topografii. W ten sposób języki opisu sprzętu stały się językami wejściowymi systemów automatycznego projektowania układów scalonych. Dwa standardowe, powszechnie używane języki opisu sprzętu to Verilog i VHDL. Każdy z nich może być używany w powiązaniu z oprogramowaniem do projektowania układów scalonych pochodzącym z wielu różnych firm.

Języki opisu sprzętu pozwalają opisywać układy na kilku poziomach abstrakcji. Możliwy jest opis funkcjonalny (zwany też behawioralnym); taki opis pozwala zdefiniować funkcję układu (opisuje algorytm realizowany przez układ) bez określania, w jaki sposób ta funkcja ma być zrealizowana. Możliwy jest opis architektury, gdzie definiuje się bloki, z jakich składa się układ, określa ich działanie, a w tym sygnały wejściowe i wyjściowe przesyłane między blokami, ale nie określa się, jak bloki mają być zrealizowane. Możliwy jest także opis strukturalny, w którym definiowany jest konkretny schemat logiczny bloku lub całego układu. Opisy te można łączyć w jednym projekcie i najczęściej w opisach bardziej złożonych układów znajdziemy fragmenty opisane na każdym z tych poziomów abstrakcji. Przykładowo, układ może być opisany przez zdefiniowanie jego architektury, a w tym opisie jedne bloki mogą być opisane przez zdefiniowanie ich funkcji (czyli opis funkcjonalny), a inne przez podanie konkretnego schematu logicznego (czyli opis strukturalny).

Te trzy poziomy abstrakcji potrzebne są po pierwsze dlatego, że nie każdy opis funkcjonalny może służyć do automatycznej syntezy układu. Mówimy, że opis funkcjonalny jest syntezowalny, jeśli automatyczna synteza jest możliwa. Jednak w pewnych przypadkach automatyczna synteza nie jest możliwa bez znajomości intencji projektanta. Przykładowo, instrukcja zsumowania dwóch liczb może być jednoznacznie przekształcona na schemat logiczny sumatora tylko wtedy, gdy znany jest sposób reprezentacji tych liczb oraz rodzaj sumatora, jaki ma być zastosowany. Jak widzimy, czysty opis funkcjonalny zwykle nie wystarcza, konieczne jest zdefiniowanie pewnych szczegółów pożądanej implementacji. Niemniej, opis funkcjonalny jest bardzo wygodny jako punkt startowy projektu, ponieważ można poddać go symulacji funkcjonalnej nawet, jeśli nie jest syntezowalny. W ten sposób można uniknąć błędów na etapie definiowania funkcji układu. Ponadto opis funkcjonalny jest znacznie krótszy, prostszy i możliwy do interpretacji przez człowieka, podczas gdy opis architektury, a tym bardziej opis strukturalny jest dla człowieka bardzo trudny do analizy i interpretacji. Nawet nie znając żadnego z wymienionych języków nietrudno wyobrazić sobie, że gdy widzimy instrukcję postaci Z = X+Y, to intuicyjnie rozumiemy, o co chodzi, zaś gdy widzimy opis schematu logicznego sumatora złożonego z kilkuset bramek logicznych (czyli kilkaset linii kodu opisu strukturalnego), to bez dodatkowych wyjaśnień nie domyślimy się, co ów układ robi (chyba że sami go zaprojektowaliśmy), i nie sprawdzimy również bez użycia symulatora, czy jest to schemat poprawny.

Opis architektury oraz opis strukturalny są potrzebne nie tylko dlatego, że nie każdy opis funkcjonalny jest syntezowalny, ale także dlatego, że wyniki automatycznej syntezy układu mogą być dalekie od optymalnych. Dlatego najczęściej spotyka się opisy, w których projektant definiuje architekturę układu, część bloków opisuje funkcjonalnie, a niektóre opisuje strukturalnie, ponieważ wie, jaki schemat logiczny będzie dla realizacji funkcji tych bloków najlepszy.

Języki opisu sprzętu były początkowo przeznaczone tylko do opisywania układów cyfrowych. Obecnie istnieją rozszerzenia (Verilog-A, VHDL-AMS) umożliwiające także opisywanie układów analogowych. Opisy układów analogowych nie są syntezowalne, ponieważ nie udało się, jak dotąd, zautomatyzować projektowania układów analogowych. Możliwa jest natomiast ich symulacja. Możliwa jest też symulacja układów mieszanych, analogowo-cyfrowych.

Opis funkcjonalny układu w języku opisu sprzętu jest na pierwszy rzut oka podobny do opisu algorytmu w tradycyjnym języku programowania (zwłaszcza że składnia języków opisu sprzętu pod wieloma względami wywodzi się z języków programowania, np. dla języka Verilog wzorem składni był język C, dla VHDL - język ADA). Istnieje jednak kilka zasadniczych różnic. Po pierwsze, operacje arytmetyczne i logiczne w językach programowania mają sens operacji na abstrakcyjnych obiektach, jakimi są liczby, zmienne czy też wartości logiczne. Podobnie zapisane operacje w językach opisu sprzętu oznaczają działanie konkretnych fizycznych obiektów - bramek, przerzutników, rejestrów itp. Dlatego instrukcja postaci Z = X+Y w języku opisu sprzętu nie jest wystarczająca. Aby opis był kompletny, muszą być znane takie informacje, jak format liczb, liczba bitów, ich kolejność (bit najmniej znaczący na początku czy na końcu?), a także co się ma stać po wykonaniu operacji - czy wynik ma być zapamiętany, czy gdzieś przesłany, jeśli tak, to gdzie? Po drugie, języki opisu sprzętu umożliwiają zdefiniowanie zależności czasowych między operacjami, w tym także operacji współbieżnych, tj. wykonywanych równocześnie i niezależnie.

A oto przykład: jak wygląda opis prostego bloku logicznego w języku Verilog.

PRZYKŁAD

```
module ARP (A, PM, Q);
input [3:0] A;
input PM;
output [3:0] Q;
reg [3:0] Q;
  always @(A or PM)
  if (PM && A < 4'b1111)
    Q = A + 1;
  else
    Q = A;
endmodule
```

Opisywany blok cyfrowy o nazwie „arp” ma realizować następujący algorytm. Na wejściu dane jest czterobitowe słowo A oraz jeden bit oznaczony PM. Po każdej zmianie wartości A lub PM należy sprawdzić, czy nowa wartość PM jest jedynką i czy nowa wartość A nie jest równa maksymalnej możliwej, tj. czterem jedynkom. Jeśli oba warunki są spełnione, to wyjściowe czterobitowe słowo Q otrzymuje wartość o 1 większą od A, w przeciwnym razie Q otrzymuje wartość równą A. Wartość Q jest zapamiętywana w czterobitowym rejestrze.

Na koniec warto dodać, że języki opisu sprzętu mogą być używane do opisu układów cyfrowych, analogowych i mieszanych realizowanych w dowolny sposób, nie tylko jako układy specjalizowane (ASIC).

3.3 Weryfikacja projektów układów scalonych

3.3.1 Symulacja elektryczna

Symulacja elektryczna polega na rozwiązywaniu układów równań opisujących sieć elektryczną układu. Napięcia i prądy w sieci opisane są równaniami teorii obwodów. Elementy są reprezentowane przez modele matematyczne. Taki model to równanie lub układ równań opisujących, jakie są zależności między napięciami na zewnętrznych wyprowadzeniach elementu, a prądami płynącymi przez element. Najprostszy model to model rezystora o stałej rezystancji - jest nim po prostu prawo Ohma: $I = U/R$. R jest w tym modelu parametrem

modelu elementu, czyli wielkością zmienną o wartości specyficznej dla danego konkretnego rezystora. Modele elementów półprzewodnikowych (diod, tranzystorów MOS, tranzystorów bipolarnych) są dużo bardziej skomplikowane. Najprostsze modele tranzystora MOS i tranzystora bipolarnego były omówione poprzednio (część I, punkty 3.1.5 i 3.1.6). Historycznie najstarszym symulatorem układów elektronicznych, który został powszechnie zaakceptowany jako użyteczne narzędzie w projektowaniu układów elektronicznych, jest symulator o nazwie „Spice" opracowany na Uniwersytecie Kalifornijskim w Berkeley. Od niego wywodzi się bezpośrednio lub pośrednio olbrzymia liczba istniejących dziś symulatorów, samą symulację elektryczną nazywa się często „symulacją typu Spice", a symulatory określa się nazwą „symulator klasy Spice" (nawet jeśli jest to symulator nie wykorzystujący kodu źródłowego oryginalnego symulatora Spice).

Symulacja elektryczna jest niezastąpionym narzędziem weryfikacji przy projektowaniu układów scalonych. Umożliwia zbadanie prądów i napięć w układzie w stanie ustalonym, poznanie charakterystyk amplitudowych i fazowych układu w funkcji częstotliwości dla sygnałów zmiennych o małej amplitudzie, określenie przebiegów napięć i prądów w funkcji czasu dla sygnałów o dowolnej amplitudzie i zmienności w czasie, a także zbadanie bardziej subtelnych właściwości układu takich, jak poziom szumów własnych lub zniekształceń nieliniowych. Elegancka grafika współczesnych symulatorów sprawia, że na ekranie komputera oglądamy wyniki symulacji w postaci przebiegów do złudzenia przypominających przebiegi na ekranie oscyloskopu podłączonego do realnego układu. Może to powodować nadmierne zaufanie do wiarygodności wyników symulacji. Tymczasem trzeba pamiętać, że symulacja elektryczna układu przy użyciu symulatora klasy Spice jest jedynie numerycznym rozwiązywaniem równań algebraicznych i różniczkowych, i niczym więcej. Jeśli równania nie opisują dobrze symulowanego układu lub jego elementów, to wyniki symulacji mogą daleko odbiegać od działania rzeczywistego układu. Jeżeli wyniki symulacji nie są zgodne z oczekiwanymi, należy je traktować z wielką ostrożnością i koniecznie ustalić przyczynę. Ślepa wiara w wyniki symulacji elektrycznej doprowadziła do niejednej klęski projektowej.

UWAGA

Ważne: symulacja nie zastępuje zrozumienia działania układu!

Typowe przyczyny rozbieżności między wynikami symulacji, a działaniem rzeczywistego układu są następujące:
- omyłki w symulowanym schemacie elektrycznym,
- modele elementów (zwłaszcza tranzystorów) nie oddające dostatecznie dokładnie ich rzeczywistych właściwości,
- źle określone parametry elementów układu,
- brak odwzorowania w schemacie niektórych zjawisk występujących w rzeczywistym układzie (np. sprzężeń między elementami poprzez podłoże układu scalonego).

Omyłki w symulowanym schemacie to sprawa trywialna, a jednak zdarzają się one dość często i nie zawsze są łatwe do zauważenia. Dlatego, jeśli wyniki symulacji nie są zgodne z oczekiwanymi, należy przede wszystkim sprawdzić, czy poprawnie zdefiniowano symulowany schemat (a w tym także wartości parametrów elementów, jednostki, wartości i znaki napięć zasilania itp.).

Modele elementów (w postaci odpowiednich układów równań) są zwykle wbudowane w symulator. W przypadku tranzystorów, a zwłaszcza tranzystorów MOS, mamy zwykle do wyboru kilka różnych modeli.

Producent układów scalonych w swej dokumentacji określa, jakich modeli należy używać i podaje dla nich wartości parametrów. Niekiedy ten sam element może być opisany kilkoma różnymi modelami do różnych zastosowań, należy to sprawdzać w dokumentacji producenta. Nawet bardzo dokładny model da złe wyniki przy niewłaściwych wartościach parametrów.

Brak odwzorowania w układzie niektórych zjawisk dotyczy przede wszystkim oddziaływań zwanych pasożytniczymi, takich jak pojemności między ścieżkami połączeń w układzie, rezystancje rozproszone obszarów elementów, rezystancje i indukcyjności ścieżek połączeń, działanie pasożytniczych elementów aktywnych. Dopóki dysponujemy jedynie schematem układu, a jego fizyczna struktura nie jest jeszcze zaprojektowana, nie da się dobrze przewidzieć tych wszystkich oddziaływań. Zależą one bowiem od konkretnych kształtów, wymiarów i rozmieszczenia elementów i połączeń między nimi. Dlatego w projektowaniu układów scalonych symulację elektryczną wykonuje się zwykle dwukrotnie. Najpierw symuluje się projektowany układ przed zaprojektowaniem jego struktury fizycznej, dla sprawdzenia poprawności projektu i zgrubnego oszacowania parametrów układu. Po zaprojektowaniu struktury fizycznej układu (kształtów i wymiarów elementów, ich rozmieszczenia i połączeń, czyli topografii) projekt tej struktury poddaje się ekstrakcji - specjalny program komputerowy zwany ekstraktorem odtwarza schemat elektryczny układu na podstawie projektu jego struktury fizycznej (w praktyce na podstawie projektu masek produkcyjnych). Dobre ekstraktory znajdują i umieszczają w schemacie także wiele rodzajów elementów pasożytniczych i określają ich parametry. Można teraz powtórzyć symulację elektryczną, a jej wyniki będą bliższe rzeczywistemu działaniu projektowanego układu.

Istotną wadą symulacji elektrycznej jest jej duża złożoność obliczeniowa. Nawet przy użyciu bardzo wydajnych komputerów symulację elektryczną można w praktyce wykonywać dla układów mających co najwyżej kilkaset do kilku tysięcy elementów czynnych. Zatem jest ona w pełni przydatna (a zarazem niezbędna) dla układów analogowych, które są zwykle niewielkie. W przypadku układów cyfrowych nie do pomyślenia jest symulacja elektryczna całego układu liczącego setki tysięcy lub miliony tranzystorów. Dlatego symulację elektryczną wykonuje się dla bramek lub stosunkowo niewielkich bloków funkcjonalnych. Mamy jednak w przypadku układów cyfrowych inne możliwości symulacji: symulację logiczną i symulację funkcjonalną.

3.3.2 Symulacja logiczna

W symulacji logicznej układ jest reprezentowany przez sieć połączonych ze sobą abstrakcyjnych obiektów - bramek logicznych lub większych bloków. W sieci tej nie występują napięcia i prądy, lecz abstrakcyjne sygnały - stany logiczne „0” i „1”, a także inne stany, np. stan nieokreślony oznaczany zwykle symbolem „X”. W symulacji logicznej także występują modele elementów (w tym przypadku bramek i bloków). Takimi modelami są na przykład tablice prawdy dla bramek kombinacyjnych lub opis przejść między stanami dla przerzutników. Symulator logiczny określa zmiany stanów w układzie logicznym w funkcji czasu dla zadanej sekwencji zmian stanów wejść układu. Taka symulacja daje pojęcie o tym, czy od strony logicznej układ jest zaprojektowany poprawnie. W symulacji mogą być uwzględniane zależności czasowe wynikające z opóźnień wnoszonych przez bramki, tj. skończonego czasu reakcji bramek na zmianę stanów wejść. Pozwala to zaobserwować takie zjawiska, jak hazardy i wyścigi. Jednak modelowanie zjawisk opóźnień czasowych w symulacji logicznej jest bardzo uproszczone, dalekie od realizmu fizycznego. Dlatego wyniki symulacji logicznej nie mogą być podstawą do wyciągania wniosków na przykład o maksymalnej częstotliwości pracy układu. Tę pozwala oszacować tylko symulacja elektryczna.

3.3.3 Symulacja funkcjonalna

Symulacja funkcjonalna występuje na jeszcze wyższym poziomie abstrakcji. Schemat logiczny układu nie jest jeszcze znany, natomiast zdefiniowana jest funkcja układu (czyli opis jego działania, zwany także opisem behawioralnym). Opis taki formułuje się w jednym z języków opisu sprzętu (ang. „Hardware Description Language", w skrócie HDL). Dwa najbardziej popularne języki tego typu to Verilog i VHDL. Języki te początkowo służyły wyłącznie do opisu układów cyfrowych, ale obecnie mają rozszerzenia pozwalające także na opisywanie układów analogowych. Opis funkcji układu w takim języku jest formalnie podobny do opisu algorytmu w typowym języku programowania, np. w języku C (zresztą składnia języka Verilog jest wzorowana na składni języka C). Dysponując odpowiednim symulatorem można opis behawioralny układu w języku VHDL lub Verilog poddać symulacji. Jej celem jest sprawdzenie czy funkcja układu jest zdefiniowana w sposób prawidłowy, kompletny i zgodny z intencją projektanta. Istnieją obecnie komputerowe systemy projektowania, które umożliwiają dokonanie syntezy układu cyfrowego na podstawie jego opisu behawioralnego. Jednak nawet jeśli nie dysponujemy takim systemem lub nie zamierzamy go użyć, opisanie funkcji układu w Verilogu lub VHDL i wykonanie symulacji funkcjonalnej jest bardzo pożądane, bo pozwala uniknąć projektowania układu o funkcji zdefiniowanej błędnie, w sposób niekompletny lub wewnętrznie sprzeczny. Doświadczenie pokazuje, że takie przypadki zdarzają się częściej, niż można by się spodziewać.

4 Projektowanie topografii układu scalonego

Ten temat będzie omówiony bardziej szczegółowo. Projektowanie topografii jest końcowym, a zarazem najbardziej pracochłonnym i podatnym na błędy etapem projektowania. Omówimy tu między innymi kilka sposobów projektowania uproszczonego i zautomatyzowanego, określanych czasem terminem „style projektowania". Sposób zaprojektowania topografii ma wpływ zarówno na poprawność działania układu i jego parametry, jak też na pracochłonność projektu, koszt projektu i koszt układu.

4.1 Wybór technologii i rozplanowanie topografii

Bez względu na to, w jaki sposób i jakimi narzędziami będziemy projektować układ, po określeniu jego funkcji i wymaganych parametrów musimy zdecydować, jaka będzie jego technologia produkcji. Wybór ma istotny wpływ nie tylko na to, jak układ będzie działał, ale i na koszt układu.

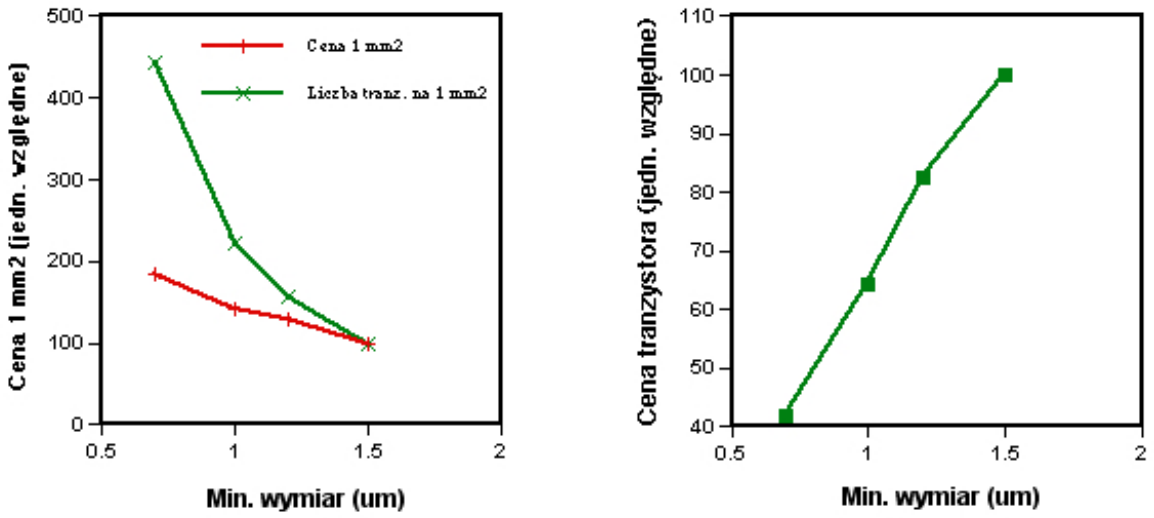
A więc mamy do zaprojektowania i wykonania układ specjalizowany określonego rodzaju. Do dyspozycji mamy kilka różnych technologii wytwarzania układów jednego lub różnych producentów. Którą z tych technologii wybrać? Zakładamy przy tym, że wszystkie rozważane technologie zapewniają spełnienie wymagań technicznych dla naszego układu. Wyboru należy dokonać na podstawie analizy ekonomicznej - która z dostępnych technologii zapewni nam najniższy koszt układu? Jak już wiemy, na koszt jednego egzemplarza układu składa się koszt wytworzenia struktur oraz koszt montażu. Gdy zamawiamy układy specjalizowane, oba składniki liczone są odrębnie. Cena za wyprodukowanie struktur obliczana jest proporcjonalnie do powierzchni układu. Dla przykładu, jeden z europejskich producentów przy zamawianiu prototypów układów CMOS stosuje następujące ceny (podane są ceny dla trzech generacji technologii, z malejącą minimalną długością kanałów tranzystorów, od 0,7 mikrometra do 0,35 mikrometra):

Tabela 1. Przykładowe ceny za 1 mm^2 układu CMOS w EURO

Technologia	Cena za 1 mm^2 układu w EURO
CMOS 0.7	280
CMOS 0.5	390
CMOS 0.35	590

Za tę cenę otrzymuje się 20 struktur nieobudowanych i nie testowanych.

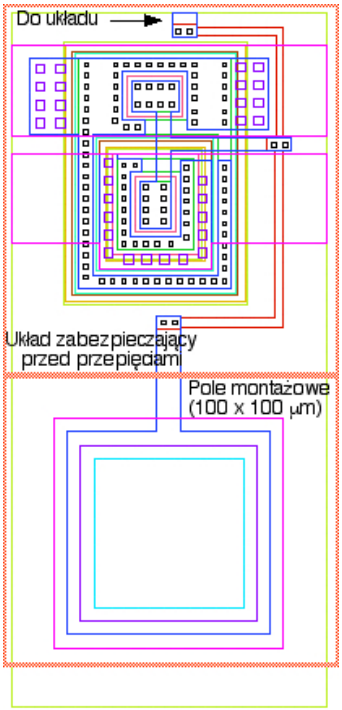
Jeżeli każda z tych trzech technologii zapewnia wymagane parametry potrzebnego nam układu, to na pierwszy rzut oka wydawałoby się, że należy wybrać technologię najtańszą (CMOS 0.7). Może się jednak okazać, że w rzeczywistości układ wykonany w tej technologii będzie najdroższy. Rzecz w tym, że liczba elementów, jakie można zmieścić na 1 mm^2 układu, ze zmniejszaniem się minimalnego wymiaru rośnie szybciej, niż cena 1 mm^2 (była o tym mowa w pierwszej części). Innymi słowy, cena jednego milimetra kwadratowego jest najniższa dla technologii CMOS 0.7, ale cena za jeden tranzystor w układzie może być najniższa dla technologii CMOS 0.35. Ilustruje to przykład dla czterech generacji technologii innego producenta:



Rysunek 4-1. Cena jednego mm^2 układu scalonego i średnia liczba tranzystorów na 1 mm^2 dla czterech generacji technologii CMOS oraz wynikający z tego koszt jednego tranzystora

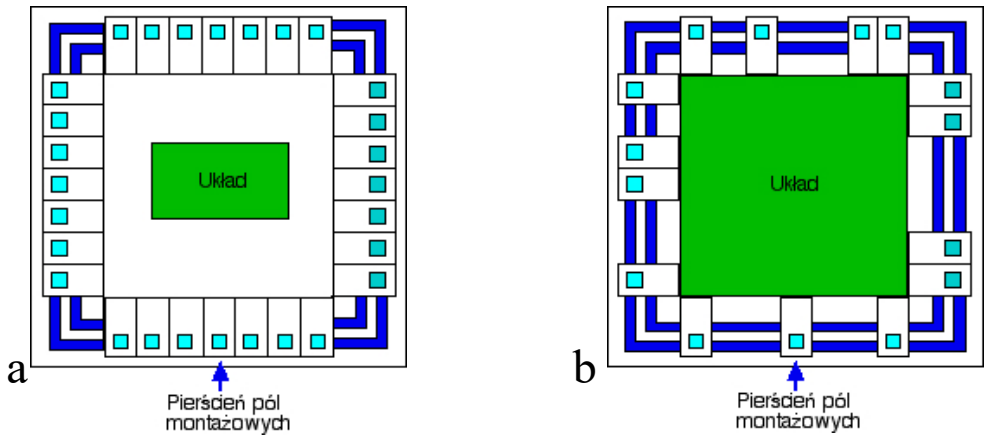
Należy więc ustalić, jaką powierzchnię zajmie układ zaprojektowany w każdej z dostępnych technologii i dopiero wtedy można zdecydować, która technologia da nam układ najtańszy. Trzeba rozważać układ kompletny, tj. wnętrze wraz z pierścieniem pól montażowych, do których dołączane są zewnętrzne wyprowadzenia (obejrzyj układ zmontowany w obudowie – rysunek 4-19 w części I).

Pola montażowe muszą mieć określone wymiary (zwykle 100 x 100 mikrometrów, w nowszych technologiach trochę mniej) niezależnie od technologii i muszą znajdować się w określonych odstępach, aby dołączenie wyprowadzeń było możliwe. Z polem montażowym zawsze związane jest kilka elementów zabezpieczających wnętrze układu przed nadmiernym napięciem z zewnątrz, które mogłoby uszkodzić układ. Pole montażowe jest więc częścią komórki, którą dostarcza producent układów. Rysunek 4-2 pokazuje topografię takiej komórki.



Rysunek 4-2. Topografia pola montażowego z elementami zabezpieczającymi przed przepięciami

Jeśli więc układ jest prosty i ma niewiele elementów, a równocześnie ma dużą liczbę zewnętrznych wyprowadzeń, to całkowita powierzchnia może być określona przez pierścień pól montażowych, a wewnątrz tego pierścienia pozostaje wolna, nie wykorzystana powierzchnia. Taką sytuację ilustruje rysunek 4-3a. Układ o dużej liczbie elementów i małej liczbie wyprowadzeń jest pokazany na rys. 4-3b. W tym przypadku wnętrze pierścienia pól montażowych jest całkowicie wypełnione. O całkowitej powierzchni decyduje powierzchnia zajęta przez układ.



Rysunek 4-3. Mały układ z dużą liczbą wyprowadzeń (a) i duży układ z małą liczbą wyprowadzeń (b)

Oto wnioski:

- Im bardziej zaawansowana technologia, tym więcej kosztuje jednostka powierzchni układu.
- Im bardziej zaawansowana technologia, tym mniej kosztuje jeden element układu.
- Układy duże i złożone z reguły opłaca się wytwarzać w najbardziej zaawansowanych technologiach.
- Układy proste i małe mogą być tańsze, gdy są wytwarzane w mniej zaawansowanych technologiach (czy tak jest, zależy od liczby wyprowadzeń).

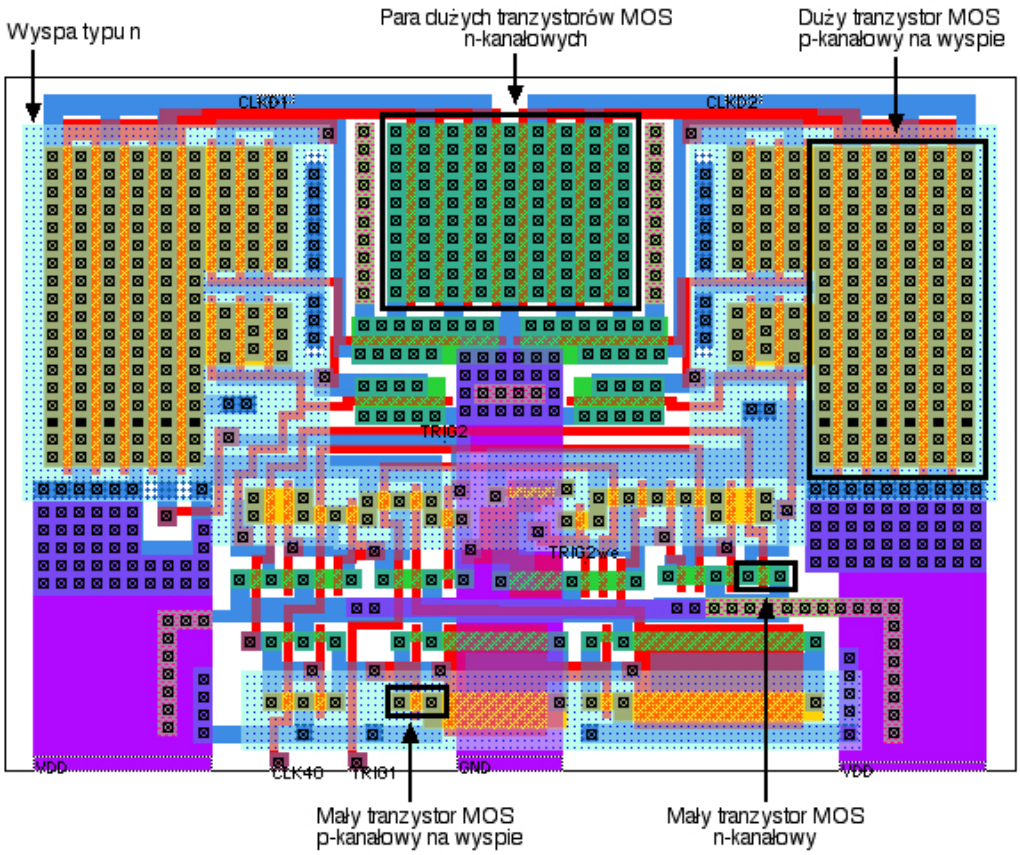
Szacowanie kosztu układu wymaga umiejętności przewidywania, jaką zajmie on powierzchnię, zanim jeszcze został zaprojektowany. Można takie oszacowanie zrobić jedynie w grubym przybliżeniu. Mogą się tu przydać dane statystyczne o przeciętnej powierzchni potrzebnej na jeden tranzystor w danej technologii. Powierzchnia ta zależy jednak silnie od rodzaju projektowanego układu (np. w układach analogowych tranzystory są znacznie większe, niż w cyfrowych) oraz od sposobu (stylu) projektowania. Toteż danych takich nie podają producenci układów. Średnią powierzchnię przypadającą na jeden tranzystor można oszacować samemu po próbnym zaprojektowaniu kilku małych fragmentów układu.

4.2 Projektowanie w stylu „*full custom*”

4.2.1 Rysowanie topografii

Zanim zaczniesz czytać, przypomnij sobie jak powstają układy scalone CMOS (część I, punkt 4.2)!

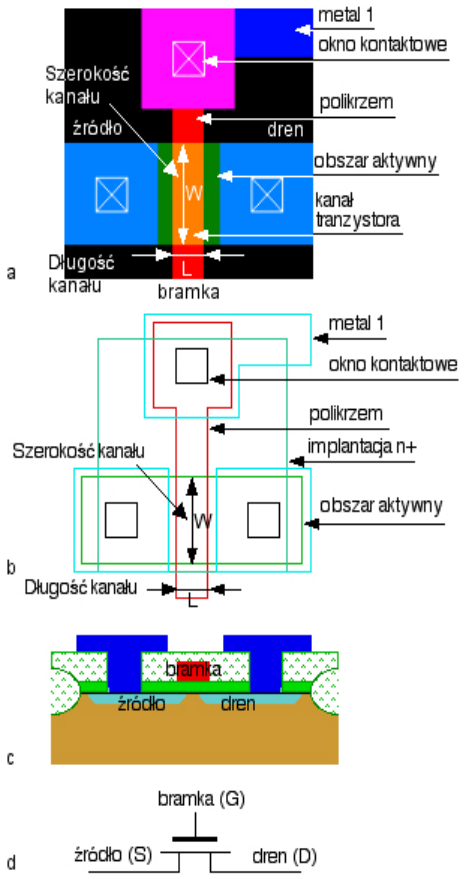
Projektowanie w stylu *full custom* polega na „narysowaniu” przez projektanta całej topografii układu. Używany jest do tego specjalizowany edytor graficzny zwany edytorem topografii. Taki edytor jest niezbędnym składnikiem każdego komputerowego systemu projektowania układów scalonych. Przykład fragmentu topografii układu zaprojektowanej w stylu *full custom* pokazuje rysunek 4-4.



Rysunek 4-4. Mały blok funkcjonalny (generator dwóch przesuniętych w czasie sygnałów zegarowych) zaprojektowany w stylu full custom

Zasadniczo rysowane są wszystkie maski fotolitograficzne, które posłużą do produkcji układu, ale możliwe są tu różne uproszczenia. Omówimy je na najprostszym przykładzie: projektu tranzystora MOS. Rysunek 4-5 pokazuje symbol tranzystora nMOS, przekrój przez jego strukturę oraz maski fotolitograficzne określające tę strukturę. Widoczny jest też obraz topografii tego tranzystora w programie „Microwind”, który posłuży do ćwiczeń w

projektowaniu – o czym dalej. „Microwind” jest to program, w którym zintegrowane są trzy funkcje: edytor topografii, ekstraktor schematu i symulator układów elektronicznych.



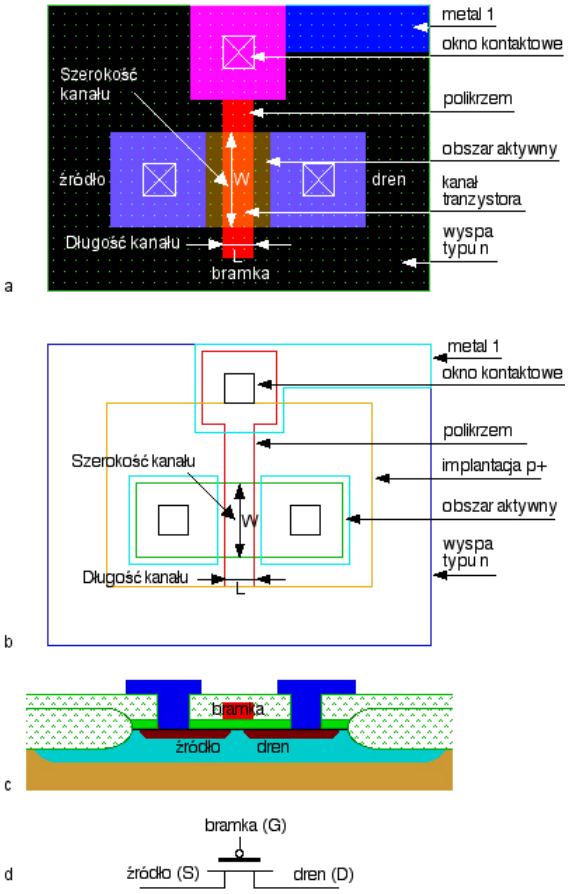
Rysunek 4-5. Widok topografii tranzystora nMOS w programie "Microwind" (a), kontury masek fotolitograficznych (b), przekrój przez strukturę tranzystora (c) oraz symbol w schematach elektrycznych (d)

Na rysunku 4-5 widoczne są następujące maski:

- Maska obszaru aktywnego (zwana czasami z przyczyn historycznych maską dyfuzji) - kolor jasnozielony. Ta maska określa szerokość kanału tranzystora.
- Maska polikrzemu - kolor czerwony. Ta maska określa długość kanału tranzystora.
- Maska implantacji typu *n* - kolor ciemnozielony.
- Maska kontaktów - kolor czarny (trzy obszary).
- Maska metalu 1 - kolor jasnobłękitny (trzy obszary).

W przypadku tranzystora p-kanałowego wygląd topografii i zestaw masek jest podobny – rysunek 4-6.
Na rysunku 4-6 widoczne są następujące maski:

- Maska wyspy typu n - kolor ciemnoniebieski (część krawędzi tej maski jest zasłonięta przez krawędź maski metalu 1).
- Maska obszaru aktywnego (zwana czasami z przyczyn historycznych maską dyfuzji) - kolor jasnozielony. Ta maska określa szerokość kanału tranzystora.
- Maska polikrzemu - kolor czerwony. Ta maska określa długość kanału tranzystora.
- Maska implantacji typu *p* - kolor ciemnożółty.
- Maska kontaktów - kolor czarny (trzy obszary).
- Maska metalu 1 - kolor jasnobłękitny (trzy obszary).



Rysunek 4-6. Widok topografii tranzystora pMOS w programie "Microwind" (a), kontury masek fotolitograficznych (b), przekrój przez strukturę tranzystora (c) oraz symbol w schematach elektrycznych (d)

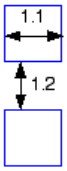
Dla wszystkich masek producenci określają geometryczne reguły projektowania. Dotyczą one minimalnych szerokości i odstępów obszarów na tej samej masce, minimalnych odstępów obszarów na różnych maskach, minimalnych zakładek, gdy obszary powinny się częściowo lub całkowicie nakładać itp. Dla najbardziej zaawansowanych technologii pełny zestaw reguł może zawierać nawet kilkaset reguł. Rysowanie topografii układu z zachowaniem tych wszystkich reguł jest bardzo pracochłonne i uciążliwe. Dlatego powstały uproszczone sposoby reprezentacji topografii. Ale zanim je poznamy, najpierw zapoznamy się z przykładowym zestawem reguł projektowania.

4.2.2 Przykładowe reguły projektowania

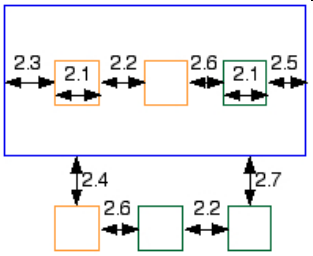
Oto przykładowy zestaw reguł projektowania dla prostej technologii CMOS z minimalną długością bramki 0,8 μm. Wszystkie reguły są wyrażone w umownych jednostkach λ (lambda) (powierzchnie w lambda^2). Zbiór reguł projektowania może być też stosowany do innych technologii CMOS pod warunkiem właściwego określenia wymiaru jednostki lambda w mikrometrach. Zazwyczaj $\lambda = 0{,}5 * L_{min}$, gdzie L_{min} jest minimalną długością bramki tranzystora MOS w danej technologii. Wartość odstępu równa 0 oznacza, że krawędzie obszarów mogą się stykać, ale nie przecinać.

Przytoczone niżej reguły mogą być stosowane w projektach wykonywanych przy użyciu programu „Microwind”, który posłuży nam do ćwiczeń w projektowaniu. Niektóre reguły nieistotne z punktu widzenia tego wykładu pominięto.

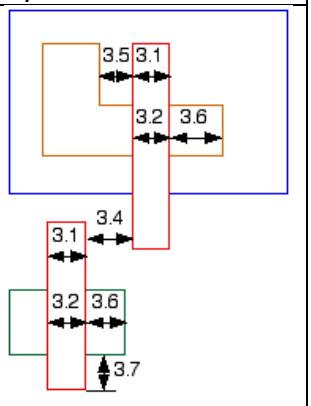
1. Reguły dla wyspy typu *n*

Nr reguły	Opis	Wartość	Rysunek
1.1	Min. szerokość	13	
1.2	Min. odstęp	18	
1.3	Min. powierzchnia	144	

2. Reguły dla obszarów aktywnych (zwanych także obszarami dyfuzji) typu *n* (oznaczonych N+) i typu *p* (oznaczonych P+)

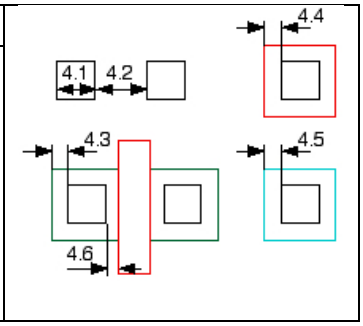
Nr reguły	Opis	Wartość	Rysunek
2.1	Min. szerokość	5	
2.2	Min. odstęp	6	
2.3	Min. odstęp między obszarem aktywnym P+ na wyspie, a krawędzią wyspy	8	
2.4	Min. odstęp między obszarem aktywnym N+ poza wyspą, a krawędzią wyspy	6	
2.5	Min. odstęp między obszarem aktywnym N+ na wyspie, a krawędzią wyspy	2	
2.6	Min. odstęp między obszarem aktywnym N+, a obszarem aktywnym P+	0	
2.7	Min. odstęp między obszarem aktywnym P+ poza wyspą, a krawędzią wyspy	6	
2.10	Min. powierzchnia	24	

3. Reguły dla obszarów polikrzemu

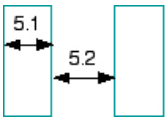
Nr reguły	Opis	Wartość	Rysunek
3.1	Min. szerokość	2	
3.2	Min. szerokość nad obszarem aktywnym (długość bramki tranzystora)	2	
3.4	Min. odstęp	3	
3.5	Min. odstęp polikrzemu od obszaru aktywnego	2	
3.6	Min. zakładka obszaru aktywnego w stosunku do bramki tranzystora	4	
3.7	Min. zakładka obszaru polikrzemu w stosunku do bramki tranzystora	3	
3.10	Min. powierzchnia	8	

4. Reguły dla kontaktów

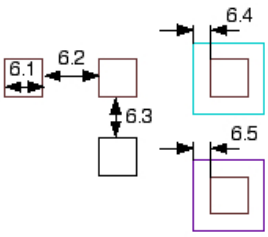
Nr reguły	Opis	Wartość	Rysunek
4.1	Wymagany wymiar (kontakt musi być kwadratem o podanej długości boku, inne kształty i wymiary nie są dozwolone)	2	
4.2	Min. odstęp	3	
4.3	Zakładka obszaru aktywnego wokół kontaktu	2	
4.4	Zakładka polikrzemu wokół kontaktu	2	

4.5	Zakładka metalu 1 wokół kontaktu	2	
4.6	Min. odstęp od bramki tranzystora	3	

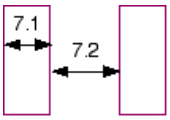
5. Reguły dla pierwszej warstwy metalu (metal 1)

Nr reguły	Opis	Wartość	Rysunek
5.1	Min. szerokość	3	
5.2	Min. odstęp	3	
5.3	Min. powierzchnia	16	

6. Reguły dla kontaktów metal 1 – metal 2 (zwanych via)

Nr reguły	Opis	Wartość	Rysunek
6.1	Wymagany wymiar (via musi być kwadratem o podanej długości boku, inne kształty i wymiary nie są dozwolone)	3	
6.2	Min. odstęp	3	
6.3	Min. odstęp od kontaktu (kontakt i via mogą się nakładać)	0	
6.4	Min. zakładka metalu 1 wokół via	2	
6.5	Min. zakładka metalu 2 wokół via	2	

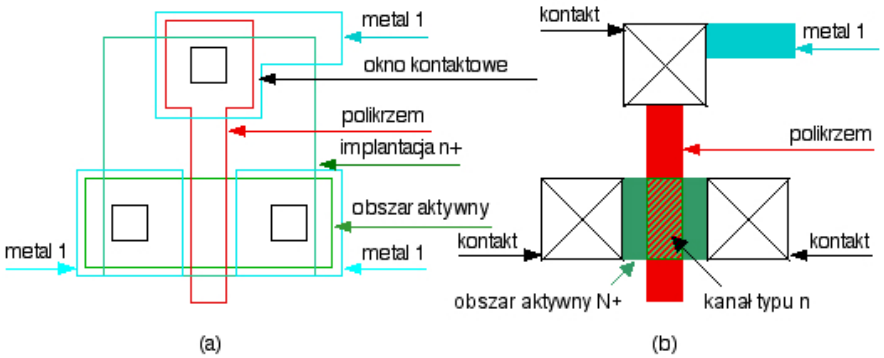
7. Reguły dla metalu 2

Nr reguły	Opis	Wartość	Rysunek
7.1	Min. szerokość	3	
7.2	Min. odstęp	3	
7.3	Min. powierzchnia	16	

Pokazane wyżej reguły dotyczą technologii dość prostej. Im bardziej zaawansowana technologia, tym więcej coraz bardziej skomplikowanych reguł. Obok reguł geometrycznych pojawiają się reguły, które można nazwać elektrycznymi, na przykład reguły służące uniknięciu niektórych rodzajów oddziaływań pasożytniczych. Tych reguł nie będziemy tu jednak omawiać.

4.2.3 Uproszczenia rysowania topografii „*full custom*”

Dla uproszczenia i zmniejszenia pracochłonności stosowane bywają dwa uproszczone sposoby reprezentacji topografii: topografia wyrażona w postaci zwanej półsymboliczną przy pomocy warstw abstrakcyjnych oraz topografia w postaci symbolicznej wyrażona poprzez schemat kreskowy. Reprezentacją półsymboliczną topografii nazywamy taką reprezentację, w której obok obiektów geometrycznych reprezentujących wprost



Rysunek 4-7. Maski tranzystora nMOS (a) i odpowiadająca im topografia zbudowana z warstw abstrakcyjnych (b)

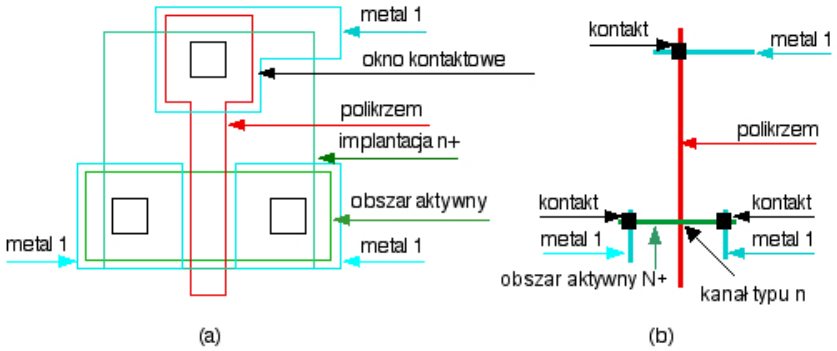
warstwy fizyczne występujące w strukturze układu (na przykład polikrzem, metal) występują także obiekty o złożonej strukturze, takie jak obszar aktywny N+, obszar aktywny P+, kanał tranzystora. Obiekty te występują na warstwach abstrakcyjnych. Warstwami tymi przy projektowaniu posługujemy się tak samo, jak maskami, jednak warstwy abstrakcyjne nie są równoważne maskom. Obiekty geometryczne na warstwach abstrakcyjnych odpowiadają pewnym kombinacjom obiektów na dwóch lub więcej maskach. Przykładowo: warstwa „obszar aktywny N+” oznacza obszar aktywny, do którego wykonano implantację donorów. Każdy obiekt geometryczny na tej warstwie oznacza więc obszar wspólny obiektów na dwóch maskach: na masce obszaru aktywnego i na masce implantacji typu *n*. Z kolei kanał tranzystora nMOS powstaje tam, gdzie obszar polikrzemu przecina obszar aktywny typu N+. Zatem obiekt geometryczny na warstwie abstrakcyjnej „kanał tranzystora nMOS” oznacza obszar wspólny obiektów na trzech maskach: obszaru aktywnego, implantacji typu *n* i polikrzemu. Gdy topografia układu jest już zaprojektowana, edytor topografii przekształca obiekty geometryczne na warstwach abstrakcyjnych na odpowiednie obiekty na maskach. Rysunek 4-7 ilustruje ideę warstw abstrakcyjnych.

Zastosowanie umiejętnie zdefiniowanych warstw abstrakcyjnych przynosi kilka korzyści:

- Topografia układu staje się bardziej przejrzysta i czytelna.
- Interesujące dla konstruktora obiekty (przede wszystkim tranzystory) rysuje się wprost, a nie jako kombinacje kształtów na kilku maskach.
- Znaczną część reguł projektowania można ukryć w algorytmach przekształcania warstw abstrakcyjnych na maski, przez co zbiór reguł projektowania ulega dużemu uproszczeniu.

Uproszczone skalowalne reguły projektowania zdefiniowane w umownych jednostkach lambda odnoszą się często do warstw abstrakcyjnych, chociaż mogą też być zdefiniowane dla masek, tak jak te pokazane w poprzednim punkcie. Są one tak obmyślone, że mogą być stosowane do wielu różnych technologii, pod warunkiem ustalenia właściwego dla danej technologii przeliczenia lambdy na mikrometry. Reguły skalowalne są proste i jest ich niewiele. Jednak po to, aby można je było dostosowywać do różnych technologii bez zmian, a jedynie przez dobór wielkości lambdy, mają one zwykle minimalne wymiary, odstępy itp. określone z pewnym zapasem. Zatem stosując tego rodzaju reguły otrzymujemy topografię układu nieco mniej gęsto „upakowaną”, zajmującą nieco większą powierzchnię niż w przypadku stosowania reguł ściśle dostosowanych do jednej konkretnej technologii. Nie ma to jednak w praktyce na ogół większego znaczenia.

W typowych systemach projektowania (również w przypadku programu „Microwind”) reguły projektowania są definiowane w pliku technologicznym, a nie w samym programie. Umożliwia to stosowanie systemu projektowania do różnych technologii. Plik technologiczny definiuje także warstwy abstrakcyjne (ich nazwy, wygląd – kolor, deseń – na ekranie komputera) oraz algorytmy przekształcania obiektów na warstwach abstrakcyjnych na obiekty na maskach fotolitograficznych.



Rysunek 4-8. Maski tranzystora nMOS (a) i odpowiadająca im topografia w postaci schematu kreskowego (b)

Reprezentacją symboliczną topografii w postaci schematu kreskowego nazywamy taką reprezentację, w której wszystkie obiekty takie, jak obszary aktywne, obszary polikrzemu i ścieżki połączeń są przedstawione w postaci linii prostych. Taka reprezentacja abstrahuje od rzeczywistych wymiarów obiektów i określa jedynie ich wzajemne położenia oraz połączenia elektryczne między nimi, do czego służą symbolicznie przedstawione kontakty. Ideę schematu kreskowego ilustruje rysunek 4-8.

Istnieją edytory topografii, które pozwalają narysować topografię w postaci schematu kreskowego, a następnie potrafią przetworzyć ten schemat na rzeczywistą topografię (tj. komplet masek) przy zastosowaniu reguł projektowania danej technologii. Jednak otrzymane w ten sposób topografie są dalekie od doskonałości. Metoda schematów kreskowych miała przed laty okres pewnej popularności, ale utraciła tę popularność, gdy pojawiły się systemy projektowania umożliwiające automatyczną syntezę topografii przy zastosowaniu komórek standardowych (będzie o tym mowa dalej). Dziś można polecić metodę schematów kreskowych jako sposób wstępnego rozplanowania rozmieszczenia elementów. Nawet jeśli nie mamy edytora topografii przystosowanego do rysowania schematów kreskowych, naszkicowanie takiego schematu na papierze pozwala wstępnie ocenić, jak wzajemnie rozmieścić elementy i połączenia w projektowanym fragmencie układu.

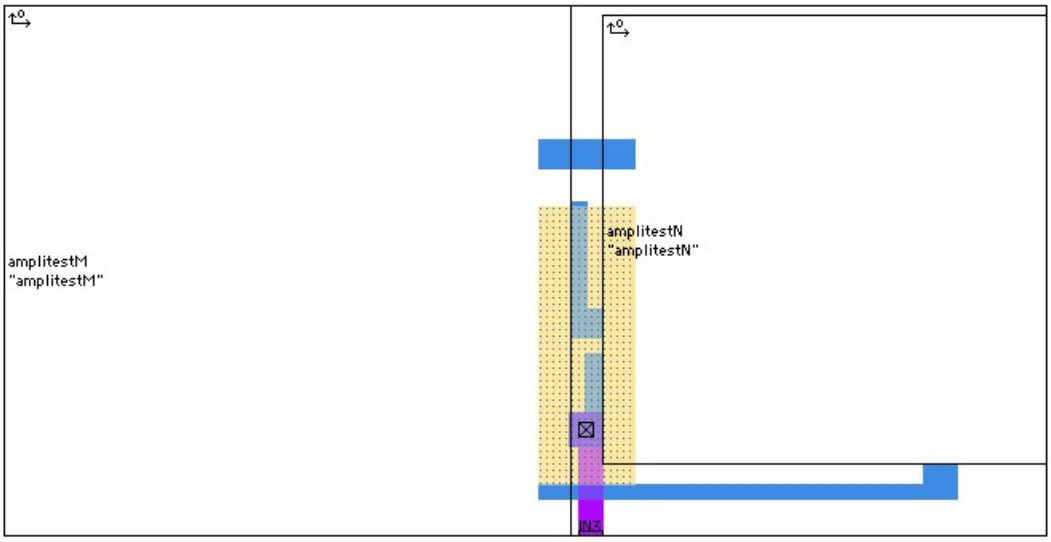
Mimo wielu lat intensywnych prac projektowania topografii w stylu *full custom* nie udało się jak dotąd skutecznie zautomatyzować, Dlatego tym sposobem projektowania posługujemy się tylko wtedy, gdy jest to niezbędne. Jest kilka takich przypadków:

- projektowanie układów analogowych,
- projektowanie topografii komórek standardowych (będzie o nich mowa dalej),
- prowadzenie połączeń między wnętrzem układu, a pierścieniem pól montażowych.

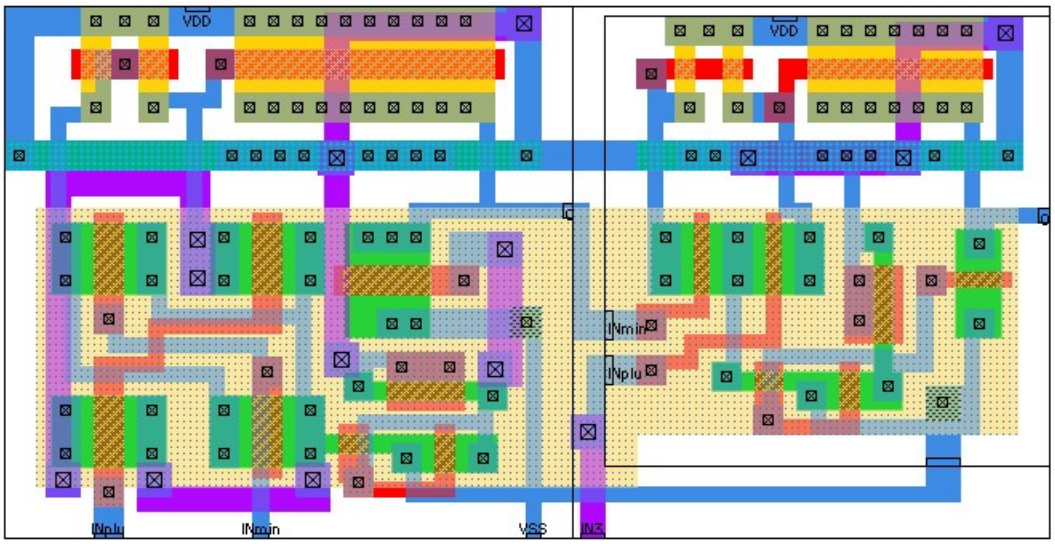
Zasadą powszechnie stosowaną w projektowaniu dużych układów jest projektowanie hierarchiczne. Z reguły każdy większy układ, a w tym praktycznie wszystkie układy cyfrowe, składa się z pewnej liczby bloków funkcjonalnych, każdy z tych bloków z kolei można podzielić na mniejsze i prostsze bloki funkcjonalne itd., aż dojdziemy do poziomu najprostszych bramek. Ta naturalna hierarchia jest widoczna nie tylko na poziomie

architektury i schematu logicznego układu, ale znajduje także bezpośrednie odbicie w projekcie topografii. Każdy profesjonalny edytor topografii umożliwia projektowanie hierarchiczne, które polega na tym, że w projekcie topografii można używać wcześniej zaprojektowanych bloków jako obiektów wstawianych w całości i reprezentowanych przez prostokątne obrysy. W projekcie zorganizowanym hierarchicznie dopiero po zakończeniu dokonuje się spłaszczenia hierarchii, tj. topografię zorganizowaną hierarchicznie przekształca się na topografię, w której nie ma już hierarchicznie zorganizowanych bloków, lecz jedynie tranzystory i połączenia. Taka topografia jest potrzebna do wykonania końcowej kontroli reguł projektowania oraz do wytworzenia masek produkcyjnych.

Hierarchiczna organizacja projektu topografii nie tylko pozwala uzyskać bardziej przejrzysty i łatwiejszy do opanowania projekt dużego układu, ale także znacznie zmniejszyć jego pracochłonność, jeśli ten sam blok powtarza się w projekcie w wielu miejscach. Projektuje się go wówczas tylko raz, a następnie wstawia wszędzie, gdzie trzeba. Jest to regułą w układach cyfrowych. Rysunki 4-9 i 4-10 przedstawiają prosty przykład topografii hierarchicznej. W jej skład wchodzą dwa kompletne bloki oraz połączenia między nimi.



Rysunek 4-10. Topografia hierarchiczna z dwoma blokami oraz połączeniami



Rysunek 4-9. Topografia hierarchiczna z rys. 4-9 po spłaszczeniu

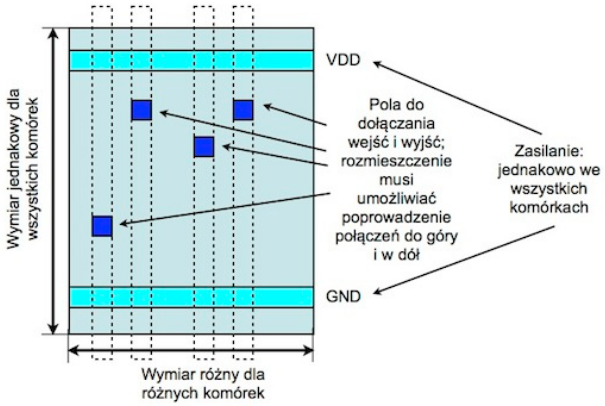
Nieco dalej znajdziesz ćwiczenia, które nauczą Cię podstaw projektowania topografii układu w stylu *full custom* przy zastosowaniu reprezentacji półsymbolicznej.

4.3 Projektowanie uproszczone i zautomatyzowane

Najbardziej pracochłonnym etapem projektowania układów scalonych jest projektowanie ich topografii. Ten etap jest też najtrudniejszy do zautomatyzowania. Podejmowane od wielu lat próby automatyzacji sposobu projektowania *full custom* nie dały jak dotąd wyników przydatnych dla praktyki inżynierskiej. Gdy zarówno rozmieszczenie elementów, jak i trasy połączeń między nimi są zupełnie dowolne, to żaden znany algorytm komputerowy – choć opracowano ich wiele – nie potrafi zastąpić intuicji i doświadczenia projektanta. Automatyzacja staje się możliwa, jeśli zrezygnujemy z całkowitej dowolności w rozmieszczeniu elementów i połączeń między nimi, i narzucimy pewne sztywne ograniczenia. Umożliwia to uproszczenie procesu projektowania i pozwala ten proces zautomatyzować. Omówimy teraz najważniejsze metody projektowania uproszczonego i zautomatyzowanego. Pozwalają one poważnie zmniejszyć pracochłonność projektu oraz ryzyko popełnienia błędów. Często w odniesieniu do nich używa się terminu „style projektowania”.

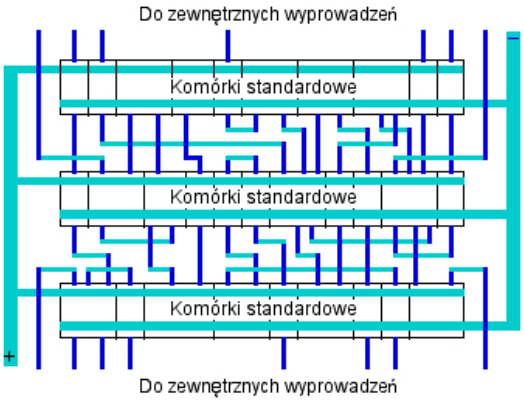
4.3.1 Projektowanie w stylu komórek standardowych

Oczywistym sposobem zmniejszenia pracochłonności projektowania jest budowanie topografii układu nie z pojedynczych tranzystorów, lecz z wcześniej zaprojektowanych bloków o sprawdzonym działaniu, które można używać wielokrotnie w różnych projektach. Ta koncepcja doprowadziła do powstania stylu projektowania zwanego stylem komórek standardowych (ang. *standard cells*). Komórka standardowa jest to blok funkcjonalny (cyfrowy lub analogowy) do wielokrotnego wykorzystania, którego topografia została zaprojektowana w specjalny sposób. Topografia każdej komórki mieści się w prostokącie, którego jeden wymiar (wysokość) jest standardowy – dla wszystkich komórek taki sam. Drugi wymiar jest dla różnych komórek różny, zależny od schematu elektrycznego komórki, liczby i wielkości elementów. Standardowe jest również rozmieszczenie doprowadzeń zasilania i masy. Są one realizowane jako szyny przechodzące przez całą szerokość komórki w jej dolnej i górnej części – rysunek 4-11.



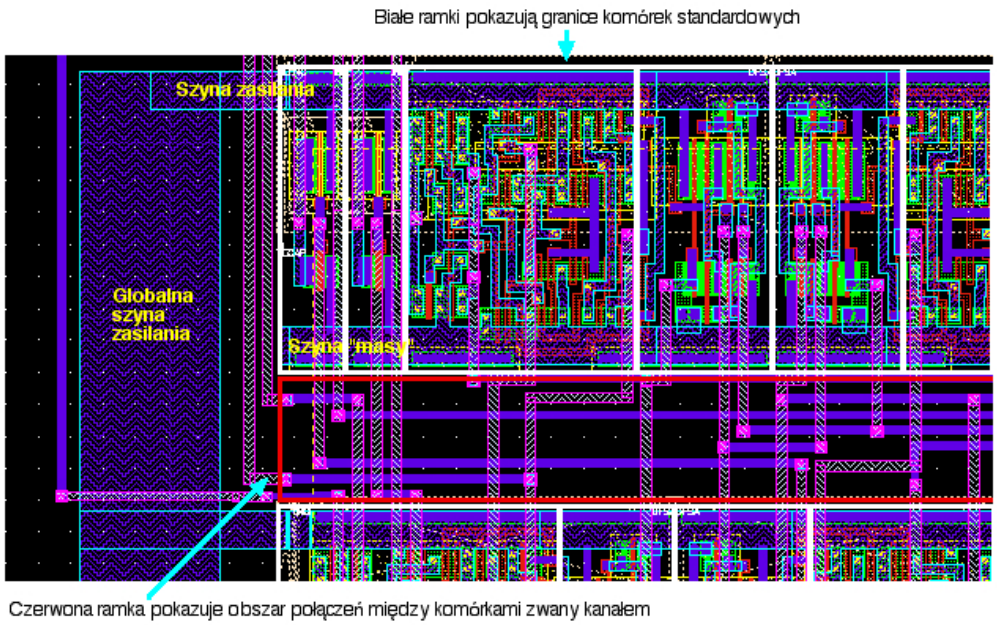
Rysunek 4-11. Schemat budowy komórki standardowej

Jeśli ustawić dwie komórki obok siebie w taki sposób, by ich pionowe krawędzie zetknęły się, szyny zasilania i masy zetkną się i połączą elektrycznie. Dzięki temu długi rząd komórek stykających się pionowymi krawędziami (które mają jednakową długość) ma gotowe połączenia masy i zasilania. Wejścia i wyjścia wyprowadzone są w kierunku poziomych krawędzi. Połączenia sygnałowe między komórkami ustawionymi w rzędy prowadzone są w obszarach pomiędzy rzędami, zwanych kanałami. Zazwyczaj w kanałach wszystkie odcinki pionowe połączeń prowadzone są w jednej warstwie metalu, a poziome - w drugiej. Jeśli dana technologia dysponuje większą liczbą warstw połączeń (co w technologiach najbardziej zaawansowanych jest regułą), to połączenia mogą być także prowadzone ponad obszarami komórek, co znacznie zmniejsza powierzchnię układu. Rysunek 4-12



Rysunek 4-12. Zasada budowy topografii układu z komórek standardowych

pokazuje schematycznie topografię układu zbudowanego z komórek standardowych, a rysunek 4-13 przedstawia fragment projektu topografii rzeczywistego układu.



Rysunek 4-13. Fragment topografii układu zbudowanego z komórek standardowych

Projektowanie topografii zbudowanej z komórek standardowych może odbywać się całkowicie automatycznie, i to jest wielka zaleta tego stylu projektowania. Projekt powstaje w trzech etapach:

- przydział komórek do rzędów,
- ustawienie kolejności komórek w rzędach,
- zaprojektowanie połączeń.

Każdy z tych trzech etapów można dość łatwo zalgorytmizować. Istnieje wiele systemów projektowania umożliwiających automatyczne zaprojektowanie topografii w stylu komórek standardowych, jeśli dany jest schemat logiczny układu. Schemat taki nie może być oczywiście zupełnie dowolny. Musi on być zbudowany wyłącznie z bramek logicznych, dla których istnieją odpowiednie komórki standardowe. Biblioteki komórek standardowych dostarcza każdy producent układów cyfrowych CMOS. Można oczywiście także zaprojektować samemu takie komórki, ale jest to pracochłonne, ponieważ odbywać się musi w stylu *full custom*. Ponadto komórki muszą być scharakteryzowane, tzn. każda z nich musi mieć znane parametry elektryczne, takie jak czasy opóźnienia sygnału, pojemności wejściowe, pobór prądu itd.

Automatyczna synteza topografii układu w stylu komórek standardowych przy równoczesnym wykorzystaniu istniejących dziś metod automatycznej syntezy układów logicznych umożliwia zautomatyzowanie całego procesu projektowania. Najpierw na podstawie opisu behawioralnego układu wyrażonego w języku opisu sprzętu (Verilog, VHDL) wykonywana jest (w kilku etapach) automatyczna synteza schematu logicznego. Następnie schemat ten służy do automatycznego zaprojektowania topografii w stylu komórek standardowych. W ten sposób można uzyskać projekt układu przy minimalnym udziale człowieka. Oznacza to nie tylko olbrzymie zmniejszenie nakładów pracy, ale i wyeliminowanie możliwości popełnienia błędów, zgodnie z zasadą „*correctness by construction*”. Dzięki możliwości pełnej automatyzacji projektowania większość cyfrowych układów scalonych jest obecnie projektowania w całości lub w znacznej części w stylu komórek standardowych. Przy automatycznej syntezie układu punkt ciężkości pracy projektanta przenosi się na sporządzenie możliwie najlepszego opisu behawioralnego w języku opisu sprzętu. Tę samą funkcję układu można opisać na wiele

sposobów. Jak już wiemy, nie każdy opis nadaje się do automatycznej syntezy układu, nawet jeśli jest całkowicie prawidłowy. Ponadto jakość zaprojektowanego automatycznie układu w bardzo dużym stopniu zależy od umiejętnego sformułowania opisu jego funkcji. Jest tu pewna analogia z tradycyjnym programowaniem. Wiadomo, że jeden i ten sam algorytm można opisać w języku programowania na wiele sposobów, dających w rezultacie programy niekiedy bardzo różniące się pod względem sprawności obliczeniowej, dokładności itp. Niestety, nie ma miejsca tutaj na rozwijanie tego skądinąd bardzo ciekawego i ważnego tematu.

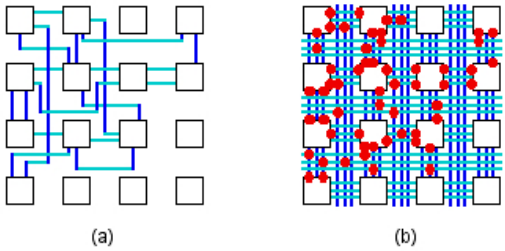
W bibliotekach komórek standardowych istnieją też komórki analogowe, toteż ten styl projektowania umożliwia zaprojektowanie niektórych układów analogowych lub mieszanych (analogowo-cyfrowych). Jednak układy analogowe są tak dalece zróżnicowane pod względem funkcji i wymagań technicznych, że w bardzo wielu przypadkach nowy układ analogowy trzeba projektować całkowicie od początku, bowiem istniejące komórki nie nadają się do wykorzystania.

W porównaniu ze stylem *full custom* styl komórek standardowych ma też pewne ograniczenia i wady. Skończony zbiór komórek w bibliotece nie zawsze umożliwia pełną optymalizację projektu logicznego. Komórki projektuje się w taki sposób, aby były możliwie uniwersalne, tj. aby można było z nich zestawiać jak z klocków układy o dowolnych schematach logicznych. Takie uniwersalne komórki nie są zaprojektowane optymalnie z punktu widzenia szybkości działania, powierzchni czy też poboru mocy. Toteż spotyka się projekty, w których niewielkie, a krytyczne dla parametrów układu fragmenty są starannie zoptymalizowane i zaprojektowane w stylu *full custom*. Projekt w stylu komórek standardowych ma też z reguły większą powierzchnię niż dobrze zrobiony projekt w stylu *full custom*. Doświadczenie pokazuje, że typowy projekt zbudowany z komórek standardowych da nam układ działający wolniej o kilkadziesiąt procent oraz zajmujący o kilkadziesiąt procent większą powierzchnię, niż ten sam układ zaprojektowany w stylu *full custom*. Jeśli jednak pamiętamy o olbrzymiej pracochłonności projektów w stylu *full custom*, to staje się jasne, dlaczego styl komórek standardowych stał się dominujący w projektowaniu układów cyfrowych.

4.3.2 Projektowanie w stylu matryc bramkowych

Jeszcze dalej idącym uproszczeniem jest zastosowanie matryc bramkowych (ang. *gate arrays*). Matrycą bramkową nazywamy układ w postaci matrycy regularnie rozmieszczonych komórek (jednakowych lub różnych), połączonych siecią połączeń w taki sposób, by uzyskać zadany schemat. Zaletą matryc bramkowych jest to, że projektuje się tylko połączenia. Nie projektuje się rozmieszczenia komórek w matrycy. Oznacza to dalsze zmniejszenie pracochłonności i ryzyka popełnienia omyłek. Co więcej, układy matryc bramkowych można produkować bez połączeń i takie półfabrykaty przechowywać. Gdy powstanie projekt konkretnego układu, wykonuje się odpowiednie do tego projektu maski ścieżek metalu i kontaktów, i wykonuje operacje wytwarzania połączeń na wcześniej wyprodukowanych matrycach. Dzięki temu bardzo skraca się nie tylko czas potrzebny na zaprojektowanie układu, ale i czas oczekiwania na prototypowe układy oraz koszt prototypów. Podobnie jak w przypadku komórek standardowych, układ przeznaczony do wykonania przy użyciu matrycy bramkowej może być zaprojektowany całkowicie automatycznie na podstawie opisu behawioralnego.

Matryce bramkowe miały okres sporej popularności, ale zostały w znacznym stopniu wyparte przez układy programowalne (FPGA), które były już wspominane wcześniej. Układ programowalny to taki rodzaj matrycy bramkowej, w której sieć połączeń określa użytkownik przez zaprogramowanie układu. Matryca zawiera nie



Rysunek 4-12.Zasada budowy matryc bramkowych: (a) zwykłej, (b) prostej matrycy programowalnej. Czerwone punkty oznaczają kontakty, które zostały zaprogramowane w matrycy programowalnej. Tam, gdzie nie ma takiego punktu, nie ma połączenia elektrycznego.

tylko komórki, ale i wstępnie poprowadzoną sieć połączeń, która jednak wymaga wykonania kontaktów. Kontakty te służą do połączenia wejść i wyjść komórek ze ścieżkami oraz ścieżek między sobą. W ten sposób powstaje gotowy układ. Kontakty są programowalne, tj. powstają w drodze elektrycznego zaprogramowania układu. Istnieje kilka sposobów wytworzenia programowalnego kontaktu.

Programowalne matryce bramkowe FPGA zdobyły dużą popularność, ponieważ przy ich użyciu działające układy otrzymuje się niemal natychmiast. Opis behawioralny układu poddawany jest syntezie logicznej, po czym generowane są dane dla programatora, który programuje połączenia w matrycy podając odpowiedni ciąg sygnałów na specjalne wejścia programujące układu FPGA.

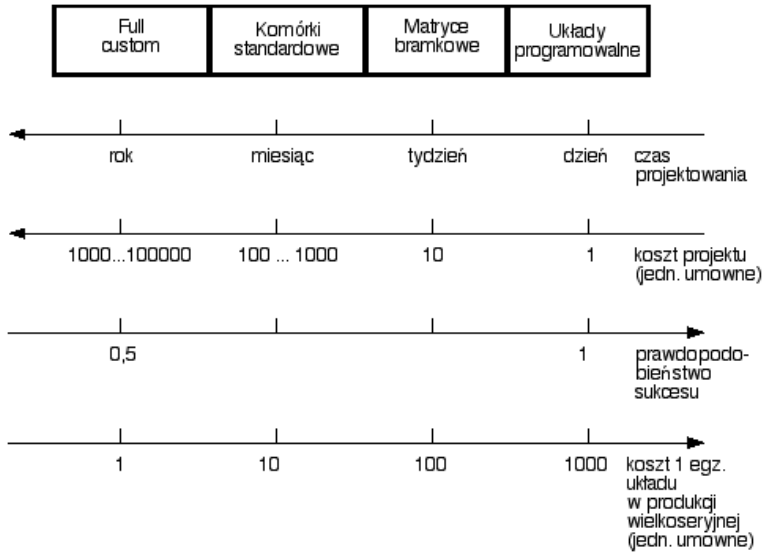
W nowoczesnych układach FPGA programowalne mogą być nie tylko połączenia, ale i funkcje logiczne samych komórek w matrycy, co daje tym układom bardzo dużą elastyczność. Oprócz matryc prostych komórek najbardziej zaawansowane układy FPGA zawierają także większe bloki funkcjonalne, aż do rdzeni mikroprocesorów i bloków pamięci włącznie. Jak wspominaliśmy wcześniej, produkowane są także programowalne układy analogowo-cyfrowe, zawierające matryce bramek cyfrowych i bardziej złożone bloki, a równocześnie typowe bloki analogowe, jak np. wzmacniacze operacyjne, oraz przetworniki analogowo-cyfrowe i cyfrowo-analogowe. Umożliwia to łatwe realizacje układów typu „System on chip", stąd ten rodzaj układów programowalnych znany jest pod nazwą „PSoC" („*Programmable system on chip*").

4.3.3 Wybór stylu projektowania

Możemy więc zrealizować układ specjalizowany różnymi sposobami: w stylu *full custom*, w stylu komórek standardowych, w stylu matryc bramkowych lub jako układ programowalny (FPGA lub PSoC). Każdy z tych sposobów ma swoje zalety i wady. Można dobrać najwłaściwszy styl projektowania do każdego zadania technicznego. Rysunek 4-15 stanowi podsumowanie – klasyfikuje omówione wyżej style projektowania z kilku punktów widzenia: czasu projektowania (rozumianego jako czas, jaki musi upłynąć od rozpoczęcia projektowania do otrzymania pierwszego egzemplarza układu), kosztu projektu (proporcjonalnego do pracochłonności), prawdopodobieństwa sukcesu (rozumianego jako prawdopodobieństwo, że pierwsze wyprodukowane egzemplarze układu będą spełniać wszystkie wymagania techniczne) oraz kosztu jednego egzemplarza układu przy produkcji wielkoseryjnej. Wartości liczbowe na osiach należy oczywiście traktować jedynie jako czysto symboliczne, służące do porównań z dokładnością do rzędu wielkości.

Biorąc pod uwagę zalety i wady wszystkich omówionych stylów projektowania można wskazać następujące ich główne obszary zastosowań:

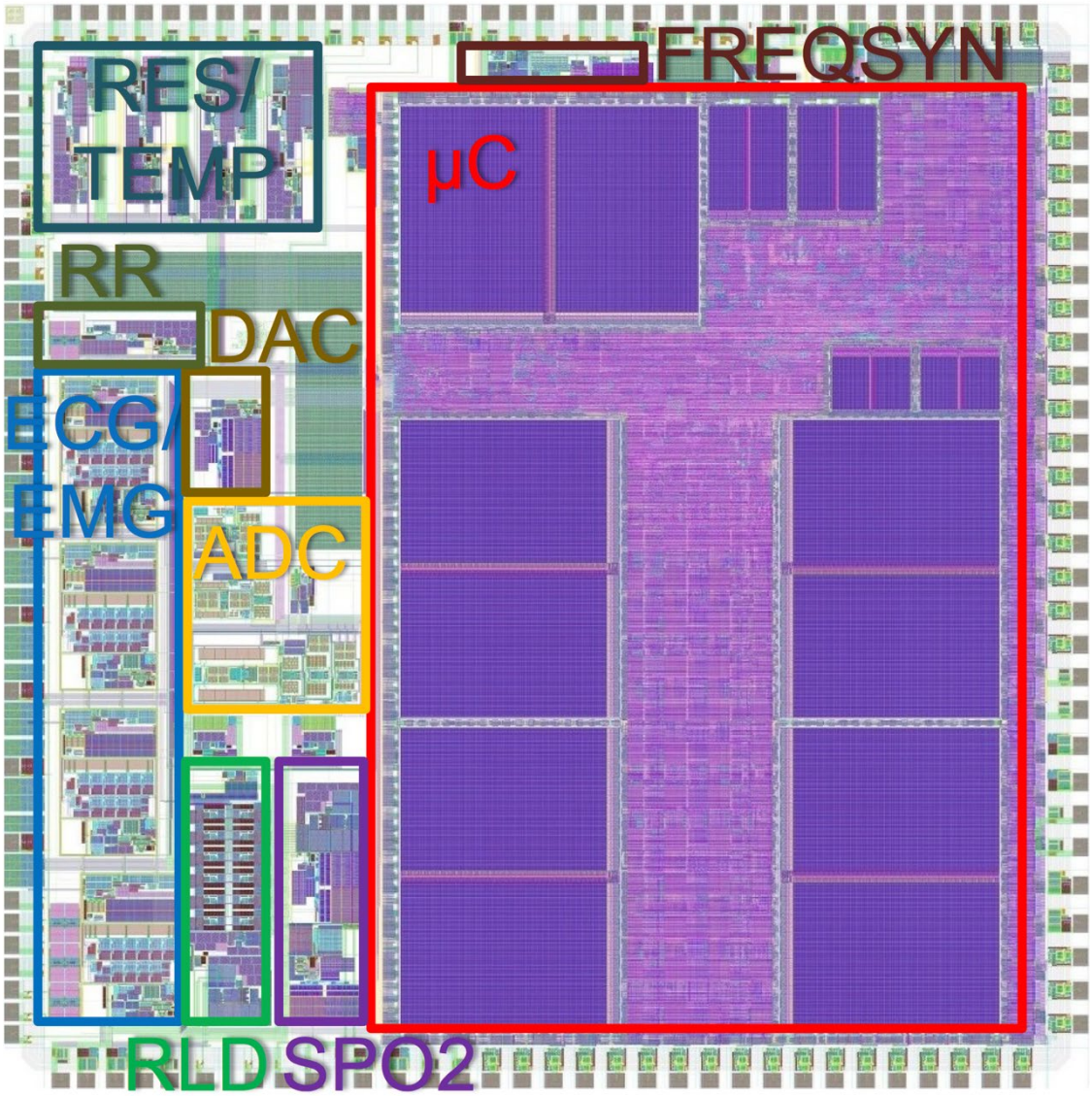
- styl *full custom*: układy analogowe i bloki analogowe układów analogowo-cyfrowych, topografie komórek standardowych, krytyczne z punktu widzenia parametrów fragmenty układów cyfrowych, niezbyt wielkie układy przeznaczone do produkcji masowej,
- styl komórek standardowych: większość układów cyfrowych przeznaczonych do produkcji w małych, średnich i długich seriach, niektóre układy analogowo-cyfrowe,
- styl matryc bramkowych: dziś coraz rzadziej stosowany,
- układy programowalne (FPGA): układy cyfrowe potrzebne w niewielkiej liczbie egzemplarzy, prototypy do szybkiego sprawdzenia konstrukcji urządzenia, układy cyfrowe potrzebne w dłuższej serii tylko wtedy, gdy ich koszt jest bez znaczenia.



Rysunek 4-15. Porównanie stylów projektowania

Postępy technologii umożliwiają wytwarzanie coraz większych i bardziej złożonych układów, i nawet wykorzystanie wymienionych wyżej sposobów projektowania uproszczonego i zautomatyzowanego nie wystarcza, gdy układ ma wiele milionów elementów. Jednym z rozwiązań problemu jest budowa wielkich systemów scalonych z wykorzystaniem gotowych dużych i złożonych bloków funkcjonalnych takich, jak rdzenie mikroprocesorów, standardowe układy peryferyjne i komunikacyjne itp. Rozwinął się rynek projektów takich bloków, można takie projekty zamawiać, kupować i sprzedawać. Taki produkt może występować w dwóch postaciach: syntezowalnego kodu w języku opisu sprzętu (można go wtedy użyć w układach wytwarzanych w różnych technologiach) lub gotowego projektu topografii dla konkretnej technologii. Jest to rynek myśli technicznej w czystej postaci, a produkty występujące na tym rynku noszą ogólną nazwę bloków IP, od angielskiego terminu „*intellectual property*" - własność intelektualna. Istnieją firmy (również w Polsce), których jedynym produktem są bloki IP. Duże układy typu „system on chip", w których są zarówno bloki analogowe, jak i cyfrowe, często są składane w większym lub mniejszym stopniu z bloków IP.

Przykład układu typu „system on chip" pokazuje rysunek 4-16. Układ zawiera analogowe bloki odczytu danych (sygnały ECG, EMG, temperatura i in.), przetworniki analogowo-cyfrowe i cyfrowo-analogowe oraz mikrokontroler.



Rysunek 4-16. Układ typu "system on chip", służy do pomiaru parametrów zdrowotnych człowieka. Projekt wykonany w Zakładzie Metod Projektowania w Mikroelektronice PW.

5 Ćwiczymy projektowanie

W tym ćwiczeniu zapoznasz się z programem „Microwind" służącym do nauki projektowania układów scalonych oraz zaczniesz opanowywać podstawy projektowania w stylu *full custom* projektując pojedynczy tranzystor.

Przed rozpoczęciem ćwiczenia musisz zainstalować program „Microwind" na swoim komputerze. Program działa w systemie operacyjnym Windows. Instalacja jest bardzo prosta - skopiuj cały katalog „Microwind2" do katalogu „Program Files". W katalogu „Microwind2" znajdziesz plik „Microwind2.exe" - to jest właśnie program. Możesz dla wygody zrobić skrót i umieścić go na pulpicie lub dodać program do menu startowego. W katalogu „Microwind2" znajdziesz także kilkaset plików. Są to m.in. pliki technologiczne definiujące różne technologie dla programu „Microwind2" (rozszerzenie „.rul"), pliki z przykładowymi projektami (rozszerzenie „.msk") i inne, oraz katalog „Html.

W kolejnych krokach otrzymasz szczegółowe wskazówki, jak wykonywać poszczególne czynności. Nie będą jednak omówione wszystkie możliwości i opcje programu. W katalogu „Microwind2\Html" znajdziesz zwięzłą

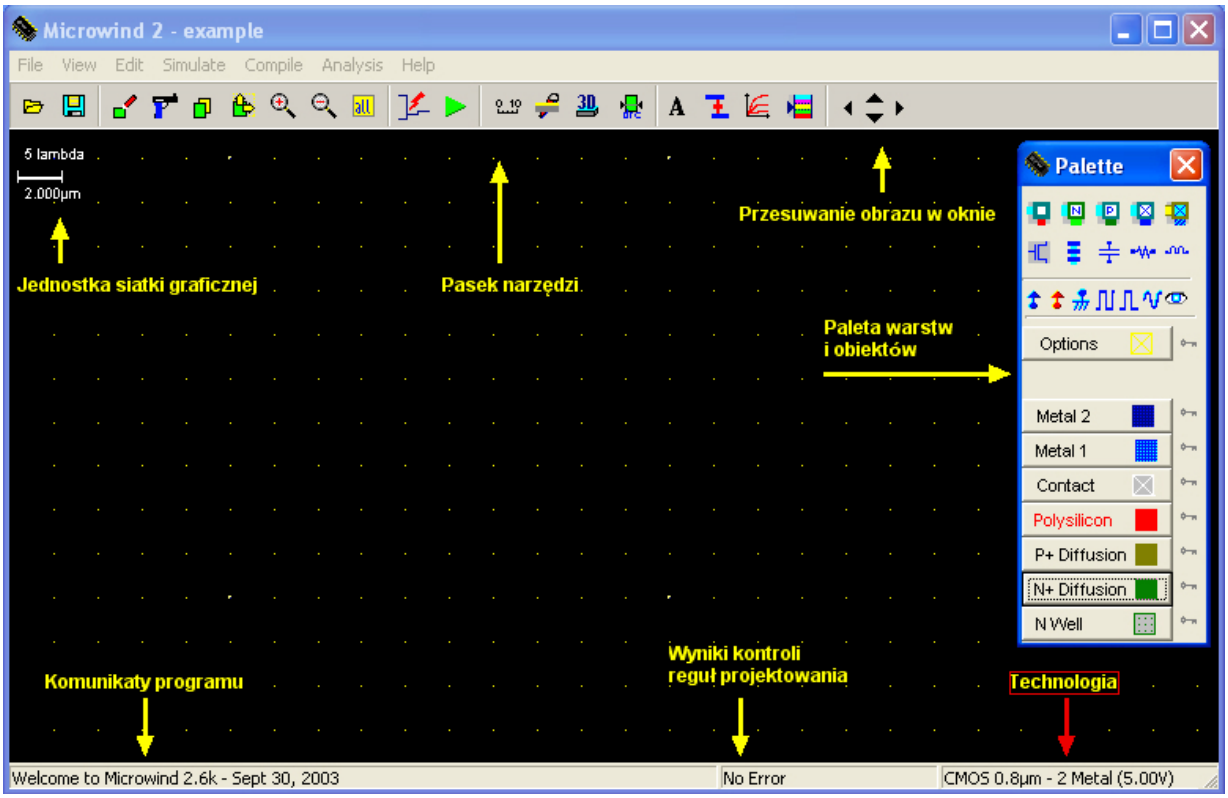
instrukcję obsługi programu (w języku angielskim) w postaci stron w języku HTML do przeglądania dowolną przeglądarką. Aby je przejrzeć, otwórz plik „index.htm”.

W katalogu „Microwind2” znajdziesz też książkę: Etienne Sicard, „Microwind and Dsch User's Manual” w postaci pliku PDF. Jeśli dostatecznie dobrze znasz język angielski, przejrzyj ją choćby pobieżnie. W książce opisane są nie tylko programy „Microwind” i „Dsch” (ten ostatni poznasz nieco później), ale i zawarta jest wielka liczba dobrze opisanych przykładów stanowiących w sumie znakomite wprowadzenie do mikroelektroniki.

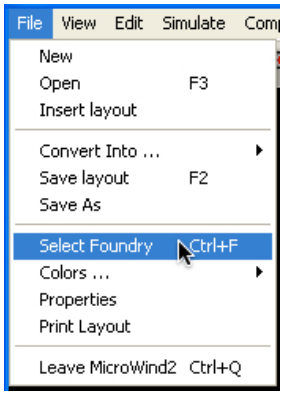
Możesz również zapoznać się z wprowadzeniem do ćwiczenia w wersji wideo, gdzie wykładowca opowie Ci i pokaże na ekranie, jak wykonywać poszczególne czynności.

5.1 Projektujemy tranzystory MOS

Uruchom program "Microwind2". Zobaczysz okno jak niżej. Na ilustracji kolorem żółtym opisano najważniejsze widoczne obiekty.

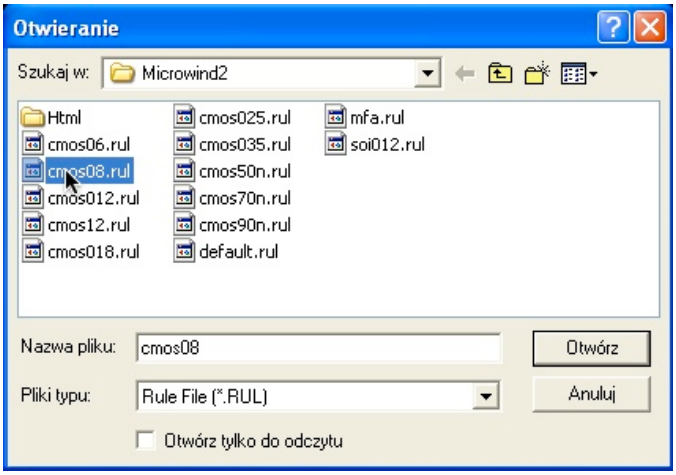


Będziemy wykorzystywać prostą technologię CMOS z minimalną długością bramki 0,8 mikrometra. W prawym dolnym rogu widnieje nazwa wczytanej przez program technologii. Przed rozpoczęciem projektowania zawsze

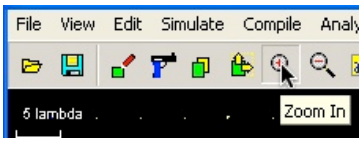


sprawdź, czy jest to technologia CMOS 0.8 µm. Jeśli nie, trzeba najpierw wczytać odpowiedni plik technologiczny.

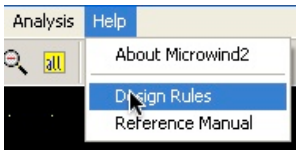
Z menu "File" wybierz "Select Foundry". Otrzymasz na ekranie typowe okno wyboru plików. Otwórz plik „cmos08.rul”.



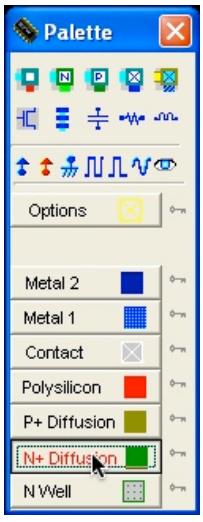
Dla wygody rysowania zmień skalę w oknie wybierając z paska narzędzi ikonę powiększenia.



Zapoznaj się z najważniejszymi regułami projektowania - wybierz „Design Rules” z menu „Help”. Otrzymasz na ekranie tabelę z podstawowymi regułami projektowania. Są w niej też inne dane, na razie nieistotne.



Zaczynasz teraz rysowanie n-kanałowego tranzystora MOS. Wybierz z palety „N+ diffusion”. Jest to warstwa abstrakcyjna oznaczająca obszar aktywny typu *n*. Nazwa warstwy, która jest w danej chwili wybrana, jest w

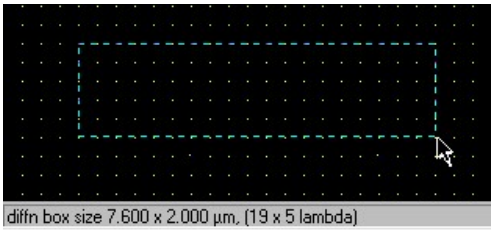


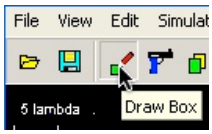
palecie oznaczona kolorem czerwonym. Warstwa nazywa się „diffusion” z przyczyn historycznych – niegdyś domieszkowanie obszarów aktywnych odbywało się przy zastosowaniu dyfuzji, a nie implantacji jonów (nie pamiętasz, co to – zajrzyj do części I, punkt 4.1).

Narysujesz teraz prostokąt na warstwie „N+ diffusion”. Wybierz ikonę rysowania prostokąta z paska narzędzi.

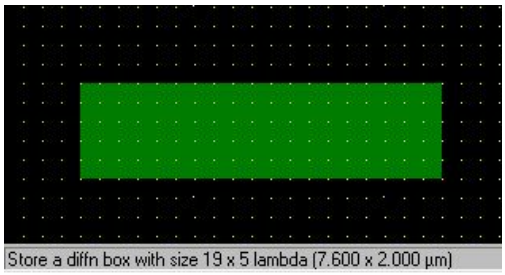
Ustaw kursor w pobliżu wybranego węzła siatki, naciśnij lewy klawisz myszki i ciągnij aż do otrzymania prostokąta o potrzebnych wymiarach, następnie puść klawisz.

Prostokąt o krawędziach zaznaczonych przerywaną kreską, który pokazuje prostokąt do narysowania, będziemy nazywali selektorem.





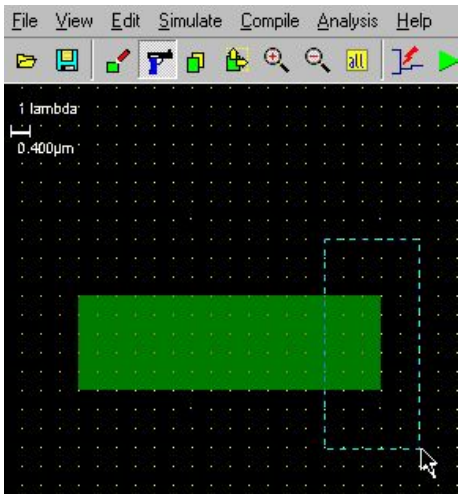
Zauważ, że nawet jeśli nie trafiasz dokładnie w węzły siatki, narysowany będzie prostokąt o wymiarach będących



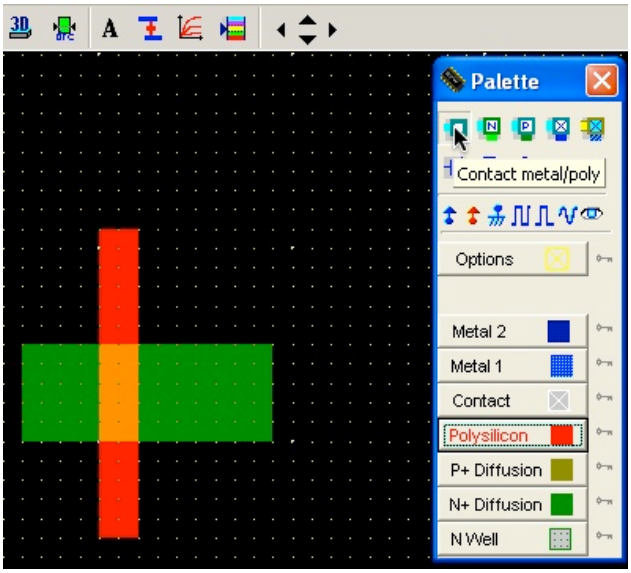
całkowitą wielokrotnością jednostki lambda. Staraj się narysować prostokąt o szerokości 5 lambda i długości zbliżonej do pokazanej na ilustracji. Jeśli Ci się nie uda, możesz wybrać „Undo” z menu „Edit” i zacząć jeszcze raz. Po puszczeniu klawisza myszki selektor „wypełni się” obszarem na warstwie „N+ diffusion”.

Jeśli narysujesz prostokąt zbyt długi lub szeroki, możesz usunąć jego fragment. Wybierz ikonę usuwania (pistolet) z paska narzędzi.

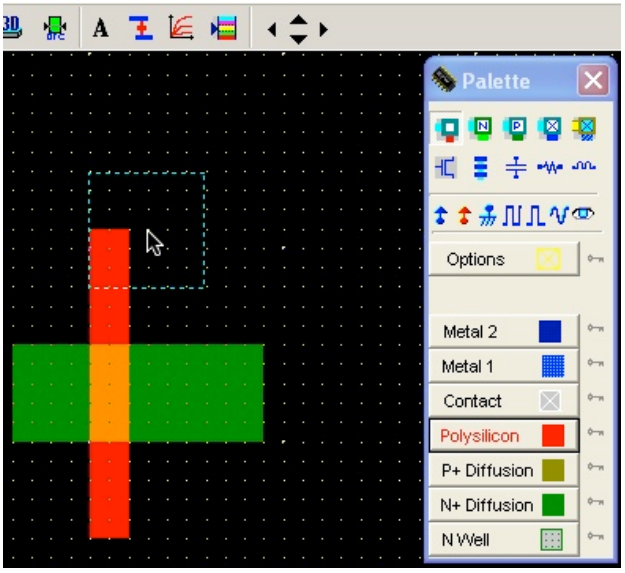
Następnie postępuj tak jak przy rysowaniu. Gdy puścisz klawisz myszki, wnętrze selektora zostanie wymazane.



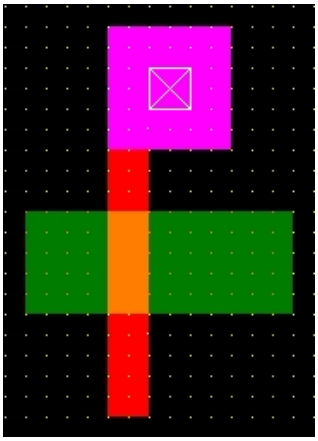
Narysujesz teraz prostokąt na warstwie „Polysilicon” (polikrzem), a potem dorysujesz do niego kontakt do warstwy „Metal 1”. Wybierz ikonę rysowania prostokąta z paska narzędzi. Następnie wybierz „Polysilicon” z palety i narysuj pionowy pasek polikrzemu o szerokości 2 lambda przecinający obszar aktywny. Następnie wybierz z palety obiekt „Contact metal/poly”.



Kontakt jest obiektem zdefiniowanym w pliku technologicznym, ma wymagany kształt kwadratu i zawiera wszystkie potrzebne warstwy: polikrzem, okno kontaktowe i metal 1. Ciągnąc myszką kwadrat selektora umieść kontakt tak jak na ilustracji i puść klawisz myszki.

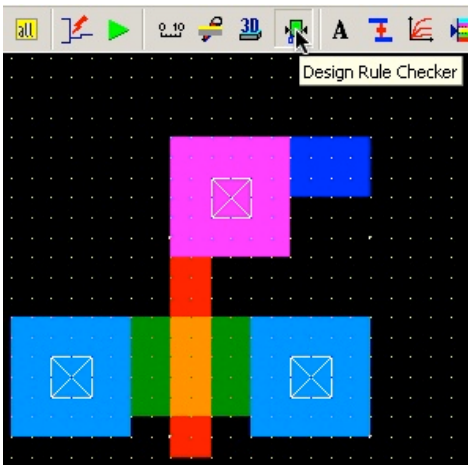


Oto wynik tej operacji:



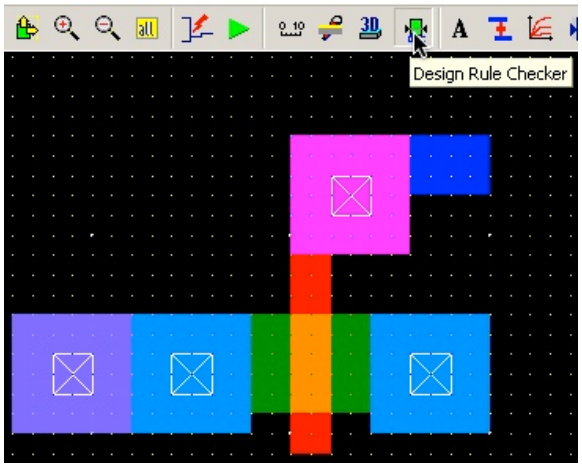
Projektuj dalej tranzystor.

Postępując tak, jak poprzednio, dodaj kontakty do obszaru aktywnego (czyli źródła i drenu tranzystora) oraz pasek metalu 1 do kontaktu bramki. Pasek polikrzemu możesz skrócić. Wystarczy, że wystaje o 2 lambda poza obszar kanału tranzystora. Następnie wybierz z paska narzędzi ikonę kontroli reguł projektowania. Jeśli wykonany projekt topografii wygląda tak, jak poniżej, otrzymasz komunikat o braku błędów.



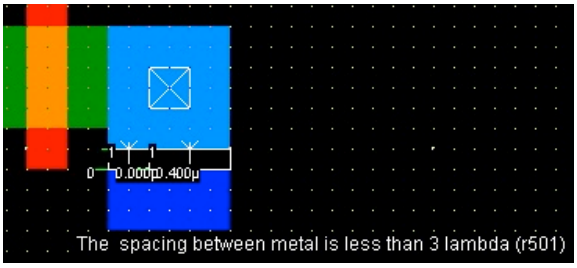
Twój tranzystor jest w zasadzie gotowy, ale trzeba jeszcze coś dodać. W układach CMOS podłoże musi być dokładnie uziemione (dlaczego? - to było opisane w części I). Aby móc uziemić podłoże, trzeba wykonać do niego kontakt. Podłoże jest półprzewodnikiem typu *p*, należy użyć kontaktu między metalem, a obszarem typu *p*. Wybierz z palety obiekt „Contact P+diff/Metal1" i postępując tak, jak przy umieszczaniu poprzednich kontaktów, umieść kontakt tak, by sąsiadował z obszarem źródła tranzystora.

Obiekt „Kontakt" zawiera wszystkie potrzebne warstwy, w tym warstwę metalu 1. Umieszczenie go tak, by sąsiadował z obszarem źródła tranzystora, oznacza że w gotowym układzie źródło będzie elektrycznie połączone z kontaktem, a ponieważ kontakt będzie uziemiony, tj. połączony z „minusem" zasilania, to uziemione będzie też źródło tranzystora. Oczywiście nie zawsze tak musi być, w układzie zawierającym wiele tranzystorów nMOS tylko niektóre będą miały źródła połączone z minusem zasilania. Ale w każdym układzie musi być przynajmniej jeden uziemiony kontakt do podłoża; w każdym większym układzie takich kontaktów musi być wiele.



Jeśli wszystko zostało wykonane poprawnie, twój tranzystor wraz z kontaktem do podłoża powinien wyglądać jak powyżej. Wykonaj jeszcze raz kontrolę reguł projektowania. Powinien ukazać się kontakt o braku błędów.

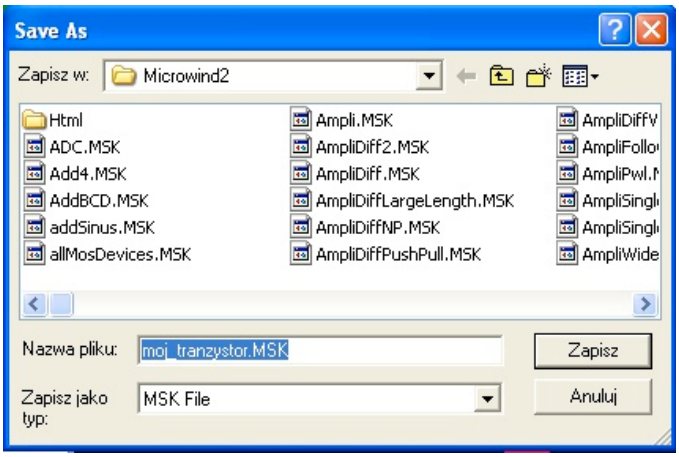
Teraz zobacz, co się stanie, jeśli w projekcie będzie błąd. Dorysuj pasek metalu 1 w odległości 1 lambda od innego obszaru metalu 1, a następnie wybierz z paska narzędzi ikonę kontroli reguł projektowania. Poniżej widzisz wynik: komunikat o błędzie.



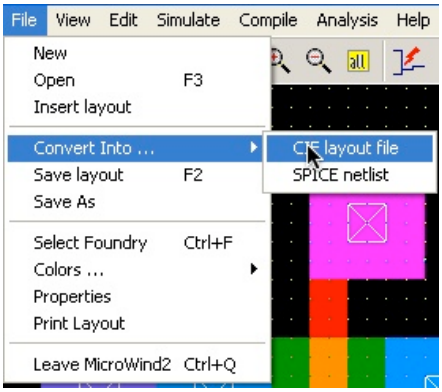
Przed dalszymi czynnościami usuń dorysowany pasek tak, aby pozostał prawidłowy projekt.

Jeśli wszystko jest w porządku, zapisz swój pierwszy projekt na dysku. Może się jeszcze przydać!

Wybierz „Save As” z menu „File” i zapisz projekt pod nową nazwą, np. "moj_tranzystor".

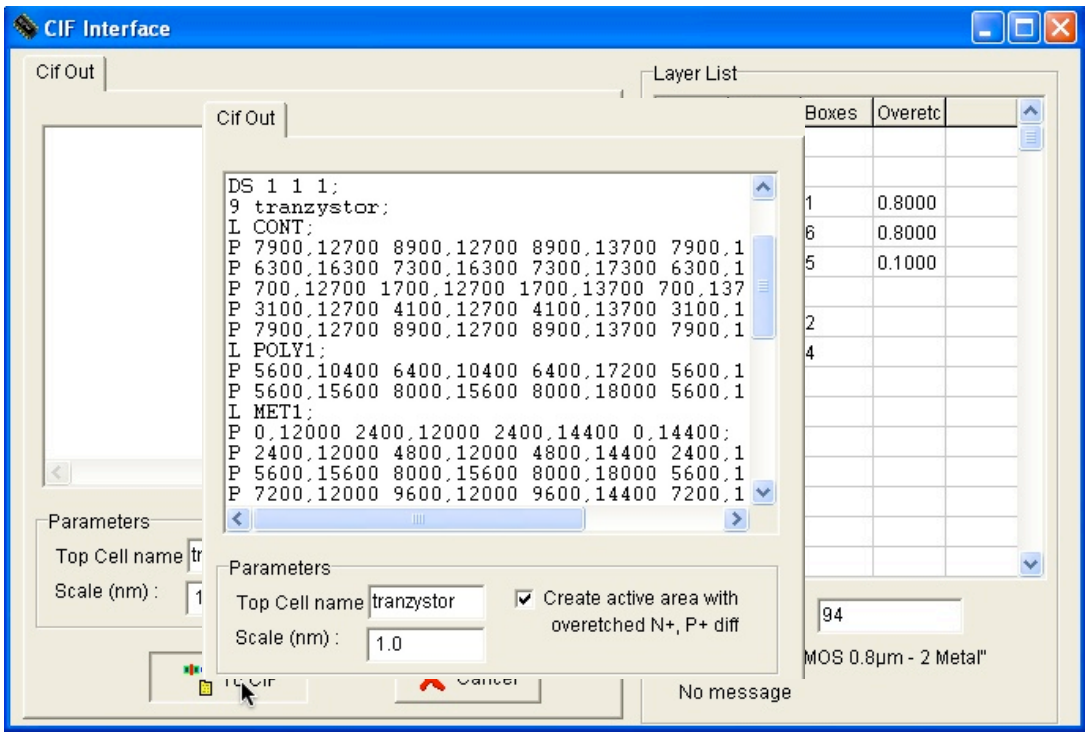


Zapisany uprzednio plik (z rozszerzeniem „.msk”) zawiera opis Twojego projektu w wewnętrznym formacie programu „Microwind”. Teraz możesz jeszcze zapisać projekt w standardowym formacie języka CIF i zobaczyć, jak taki zapis wygląda.



Wybierz „Convert Into...” -> „CIF layout file” z menu „File”. Otrzymasz na ekranie tablicę, w której nie musisz zmieniać niczego poza dopisaniem nazwy zaprojektowanej komórki (*topcell*). Tablica pokazuje jakie maski zostaną zapisane w pliku w formacie CIF. Podczas zapisu dokonywana będzie konwersja z warstw abstrakcyjnych połączona w przypadku niektórych masek ze zmianami wymiarów. Są to nieistotne dla nas w tym momencie szczegóły technologiczne.

Wpisz nazwę komórki (np. "tranzystor") w polu "Top Cell name" i kliknij "To CIF".



Projektowanie zakończone! Możesz teraz jeszcze obejrzeć zawartość przykładowego pliku CIF (Twój może w szczegółach wyglądać nieco inaczej).

Teraz pora na samodzielny trening. Narysuj topografię tranzystora MOS p-kanałowego analogiczną do narysowanej w ćwiczeniu 1 topografii tranzystora MOS n-kanałowego. Zachowaj te same wymiary kanału tranzystora. Różnice będą następujące:

- Tranzystor p-kanałowy musi być na wyspie typu n („N Well”).
- Zamiast obszaru aktywnego typu *n* należy użyć obszaru aktywnego typu *p* („P+ diffusion”).
- Wyspa oprócz tranzystora musi zawierać kontakt, który w gotowym układzie będzie podłączony do „plusa” zasilania. Wyspa jest obszarem typu *n*, więc użyj obiektu „Contact N+diff/metal1”.

Nie zapomnij po zakończeniu rysowania sprawdzić, czy spełnione są reguły projektowania!

Zapisz Twój pierwszy samodzielny projekt na dysk. Może się dalej przydać!

W tym miejscu kończy się druga część materiałów opowiadających o układach scalonych. Czy zaprojektowanie tranzystorów nMOS i pMOS było trudne? Chyba nie! W następnych częściach będzie mowa o całych układach!