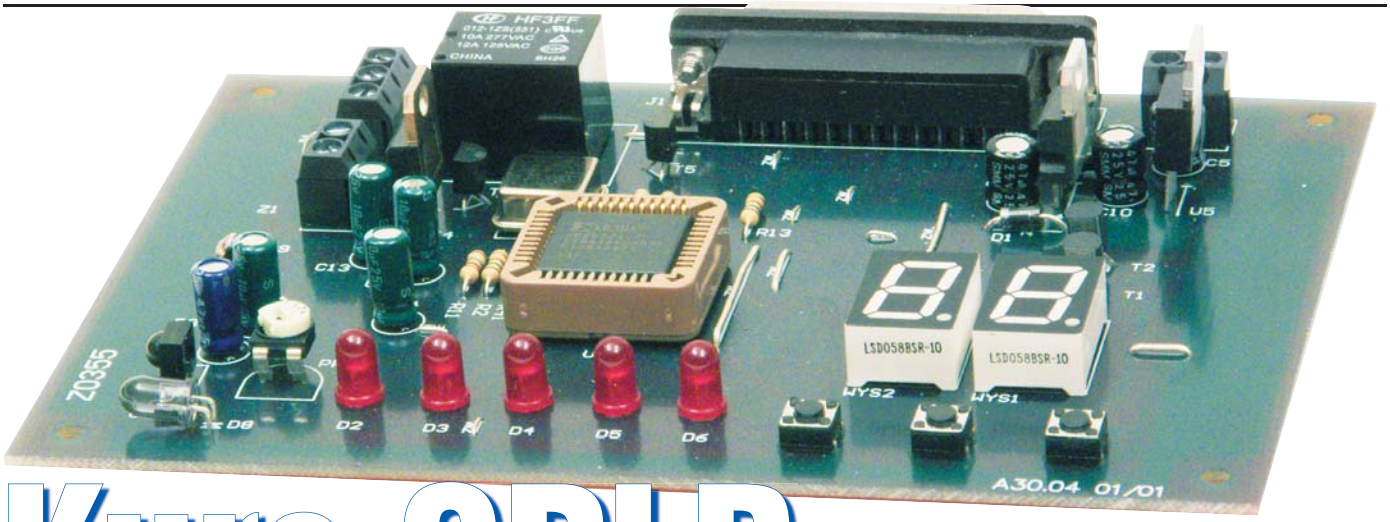


Kurs CPLD

część 1

Wstęp

Korzystając z możliwości, jaką udostępniła mi Redakcja, chciałbym zaprosić Czytelników do udziału w kursie poświęconym układom programowalnym, a konkretnie układom CPLD. Niniejszy cykl ma w przystępny sposób zapoznać Czytelnika z podstawami tworzenia projektów dla układu programowalnego, oprogramowaniem wykorzystywanym do tego celu oraz sposobem jego obsługi.

Internet pełen jest artykułów poświęconych temu tematowi, jednakże niniejsze opracowanie ma tę zaletę, iż jest kursem praktycznie od zera, opartym na zaprojektowanej w tym celu, dobrze udokumentowanej płytce testowej, której opis zawarty był w sierpniowym numerze EdW. Płytka ta umożliwi zbadanie w praktyce wszystkich poruszonych zagadnień, co oprócz wiedzy teoretycznej, pozwoli samodzielnie sprawdzić swoje umiejętności. Płytka testowa posiada podstawowe bloki funkcjonalne umożliwiające wykonanie szeregu różnorodnych ćwiczeń, a oprócz tego zawiera również programator! Jest to dodatkowe ułatwienie, które zaoszczędzi Czytelnikom sporo czasu i nerwów. Do rozpoczęcia zabawy potrzebna będzie wspomniana płytka (dostępna w sklepie AVT), przewód do podłączenia jej do komputera przez port LPT oraz komputer ze stosownym (bezpłatnym) oprogramowaniem.

Kurs jest przeznaczony dla wszystkich pasjonatów techniki cyfrowej, w tym dla zagorzałych zwolenników mikrokontrolerów, którzy zyskają możliwość rozszerzenia swojej wiedzy, oraz dla Czytelników mniej obeznanych z zagadnieniami techniki cyfrowej. Niezbędna będzie jednak podstawowa wiedza z zakresu układów cyfrowych (bramki, liczniki, rejestry, etc.), którą można uzupełnić ze stosownej książki (np. „Układy cyfrowe – pierwsze kroki" P. Góreckiego). Znacznie ułatwi to zrozumienie problemów poruszanych w kolejnych odcinkach. Część tego materiału będzie przypomniana w następnej części kursu, ale z naciskiem na aspekty praktyczne. Pokazane zostanie funkcjonowanie wszystkich podstawowych „bloczków" tak, aby Czytelnik mógł samodzielnie sprawdzić jak one pracują.

Kurs będzie prowadzony przede wszystkim w formie ćwiczeń. Przez najbliższe kilka miesięcy postaramy się zrealizować szereg projektów, zrozumieć ich funkcjonowanie oraz w jaki sposób przystąpić do ich implementacji. Realizacja projektów będzie opierać się głównie na edytorze schematów, za pomocą którego można rysować połączenia pomiędzy elementami bibliotecznymi, takimi jak bramki, liczniki, rejestry... Jest to narzędzie bardzo proste w obsłudze, intuicyjne i przejrzyste. Widząc schematy połączeń w książkach, będzie je można bezpośrednio przenieść do układu programowalnego i wypróbować, co daje nowy wymiar nauczania – praktyczne sprawdzenie tego, co opisuje autor danej publikacji.

Omówione zostaną również typowe techniki syntezy układów cyfrowych (tablice Karnaugha, układy kombinacyjne, automaty) oraz poruszony będzie temat języka opisu sprzętu, jakim jest VHDL, aby pokazać alternatywną metodę tworzenia projektów. Dość wygodną funkcją środowiska WebPACK ISE jest tworzenia własnych elementów bibliotecznych, co także zostanie przybliżone w jednym z najbliższych odcinków. Zagadnienia te mogą wydawać się skomplikowane i trudne, jednakże postaram się pokazać, że tak nie jest. Pomimo wymaganej podstawowej wiedzy z zakresu układów cyfrowych, poruszane zagadnienia zostaną przedstawione w skrótowej formie w najbliższym odcinku. Nie poprzestaniemy oczywiście na teoretycznych rozważaniach, tylko skorzystamy z poznanych narzędzi i będziemy w oparciu o nie konstruować różnorodne urządzenia.

Pozostaje mi tylko zaprosić do dalszej lektury i mieć nadzieję, iż kurs znajdzie uznanie w oczach Czytelnika.

Plan kursu

Zanim rozpoczniemy na poważnie zgłębiać tajniki układów CPLD, chciałbym przedstawić ogólny plan, według którego będą przygotowywane następne odcinki. W miarę możliwości postaram się uwzględnić sugestie Czytelników i wprowadzać zmiany w kolejnych częściach – szczególnie rozwinąć kwestie wymagającej dokładniejszego wyjaśnienia lub dodawać materiały, który będą interesujące dla szerszego grona.

Pierwsza część kursu, czyli ta, którą właśnie czytasz, poświęcona jest sprawom organizacyjnym.

Znajduje się tu przede wszystkim instrukcja opisująca krok po kroku, jak pobrać i zainstalować środowisko WebPACK ISE 6.2i, z którym będziemy pracować w najbliższej przyszłości. Bardzo szczegółowe przedstawienie tego zagadnienia, poparte dużą liczbą rysunków, pozwoli pomyślnie przeprowadzić proces instalacji nawet najmłodszym i mało zaawansowanym Czytelnikom. Nauczymy się przy okazji korzystać z programatora i przesyłać program do układu CPLD. Jest to potrzebne również podczas wykonywania płytki testowej – po jej zmontowaniu warto byłoby przetestować poprawność lutowania. Do tego celu przewidziano specjalny test dostępny na stronie Elportalu, który można pobrać ze strony i uruchomić na płytce. Po szczegóły testowania odsyłam do artykułu poświęconego płytce testowej.

Część druga porusza zagadnienia związane ze środowiskiem WebPACK ISE 6.2i. W tej części nauczymy się rysować schematy i konfigurować układ programowalny. Część

kursu będzie poświęcona na pokazanie, jak pracują elementy opisywane w książkach, tzn. sprawdzimy, czy bramka AND rzeczywiście daje jedynkę dla dwóch jedynek na wejściu, czy liczniki rzeczywiście liczą i w jaki sposób, jak zbudować niestandardowy dzielnik (np. przez 7) czy też wykorzystać przerzutnik typu D.

	XC9536XL	XC9572XL	XC95144XL	XC95288XL
Macrocells	36	72	144	288
Usable Gates	800	1,600	3,200	6,400
Registers	36	72	144	288
T_{PD} (ns)	5	5	5	6
T_{SU} (ns)	3.7	3.7	3.7	4.0
T_{CO} (ns)	3.5	3.5	3.5	3.8
f_{SYSTEM} (MHz)	178	178	178	208

Rys. 1

W **części trzeciej** przedstawiona zostanie tablica Karnaugha i sposób jej wykorzystania przy budowie prostych układów kombinacyjnych. Przyjrzymy się również narzędziom wspomagającymi pracę z tymi tablicami. Zaczniemy też wykonywać bardziej praktyczne rzeczy, takie jak obsługa wyświetlacza 7-segmentowego, budowa własnych symboli bibliotecznych, etc.

Część czwarta zostanie przeznaczona na doskonalenie umiejętności syntezy układów cyfrowych – wykonamy proste projekty takie jak minutnik do jajek oraz kostkę do gry.

W **części piątej** przyjrzymy się bliżej językowi opisu sprzętu jakim jest VHDL. Poznamy w ten sposób alternatywną metodę implementacji – opisywanie zachowania układu logicznego zamiast rysowania jego schematu. Po zapoznaniu się z podstawami VHDL-a wykonamy w ramach tej części kursu regulator mocy oparty o PWM oraz prostą barierę podczerwieni.

W **części szóstej** będziemy zmuszać układ programowalny do interakcji z otoczeniem. Zobaczymy, jak za pomocą bramek i rejestrów zbudować dekoder kodu RC5 i dzięki temu zyskamy możliwość zdalnego sterowania urządzeniem przyłączonym do płytki testowej.

Przedostatnia, czyli **siódma część** kursu, będzie poświęcona automatom synchronicznym. Przeznaczymy trochę czasu na omówienie tego „tworu", przyjrzymy się grafom przejść takiego automatu, będziemy konstruować własne grafy i następnie „implementować je w krzemie". Na tę okazję został oczywiście przewidziany stosowny projekt – automat oświetleniowy.

Ostatnia, **ósma część**, poświęcona jest również automatom i innemu projektowi, jakim jest zamek szyfrowy. Nastąpi tutaj podsumowanie kursu i być może, co będzie zależne od Czytelników i stanowiska Redakcji, wytyczenie planu zdobycia kolejnego stopnia wtajemniczenia, jakim są układy FPGA.

Czym jest układ programowalny?

Zanim zaczniemy na poważnie zagłębiać tajniki układów programowalnych, warto byłoby odpowiedzieć sobie na pytanie: co to jest? Wierzę, że każdy Czytelnik bez problemu znajdzie w Internecie stosowną definicję (choćby na Wikipedii), więc nie będę jej tu przytaczał. Rozwinięciem tego tajemniczego CPLD skrótu jest: *Complex Programmable Logic Device*, co oznacza po polsku złożony, programowalny układ logiczny. W uproszczeniu możemy przyjąć, że układ programowalny jest zbiorem dużej liczby bramek oraz rejestrów zgromadzonych wewnątrz pojedynczego układu scalonego. Co więcej, układ taki jest wyposażony w pamięć nieulotną (np. EEPROM), którą można wielokrotnie przeprogramować. Zawiera ona informację o połączeniach pomiędzy zasobami układu CPLD, co pozwala je dowolnie konfigurować i tworzyć określone bloki spełniające zadaną funkcję. Od strony użytkownika wygląda to trochę jak fabryka układów scalonych, gdzie Czytelnik przygotowuje projekt, zleca jego opracowanie odpowiednim ludziom (środowisko WebPACK ISE), a na końcu jest on wykonywany przez fabrykę (konfiguracja za pomocą programatora). W przeciwieństwie jednak do fizycznej produkcji układu scalonego, możesz się, Czytelniku, wielokrotnie mylić lub zmieniać zdanie – bez żadnych konsekwencji. Ceną tej elastyczności jest jednak mniejsza wydajność oraz mniejsze zasoby logiczne (liczba dostępnych elementów) oferowane przez układ CPLD w stosunku do produkowanych na zamówienie układów scalonych. W naszym przypadku nie będzie miało to dużego znaczenia. Gdyby jednak w przyszłości było potrzebne zaprojektowania bardziej złożonego projektu, to pozostają układy FPGA wyposażone w kilka milionów bramek.

Użyty układ XC9572 pozostawia do dyspozycji użytkownika 34 linie I/O (dla obudowy PLCC), które mogą pełnić rolę wejścia lub wyjścia. Wejścia pozwalają wprowadzić dowolny sygnał logiczny, a na wyjściu pojawia się sygnał zależny od konfiguracji układu i stanu wejść, podobnie jak w typowym układzie z bramkami. Różnica polega na tym, iż porty te mogą być dowolnie zamieniane i określenie, który port gdzie ma być dołączony, leży już w gestii Czytelnika – projektanta. Służy do tego specjalne narzędzie zintegrowane ze środowiskiem WebPack ISE, któremu również się przyjrzymy.

W porównaniu do mikrokontrolerów, układy programowalne są znacznie szybsze, gdyż przy odpowiednim zaprojektowaniu potrafią wykonywać nawet najbardziej skomplikowane zadania w jednym takcie zegara. Poza tym umożliwiają równoległe wykonywanie kilku zadań. Układ XC9536XL może być taktowany częstotliwością sięgającą od 100MHz do 178MHz (zależnie od wersji układu).

Filozofia projektowania jest oczywiście odmienna niż dla mikrokontrolerów. Spór o to, czy lepszy jest mikrokontroler, czy układ programowalny, jest bezcelowy, gdyż odpowiedź na to pytanie jest zależna od urządzenia, jakie chcemy zbudować. W przypadku bardzo dużych projektów z niektórymi zadaniami lepiej radzi sobie jednak procesor, ale nic nie szkodzi na przeszkodzie, aby... zbudować własny! Jest to rozwiązanie pracochłonne i wymagające dużych zasobów logicznych (dostępnych w układach FPGA), dlatego nie będziemy poruszać tego zagadnienia.

Architektura rodziny XC9500

Przed rozpoczęciem właściwego kursu warto byłoby zapoznać się z architekturą układu XC9572XL. Materiał zawarty w niniejszym paragrafie ma charakter uzupełniający wiedzę i jego zrozumienie nie jest wymagane do prowadzenia własnych eksperymentów. Pozwoli jednak spojrzeć trochę głębiej i zapoznać się z zasadami rządzącymi pracą CPLD.

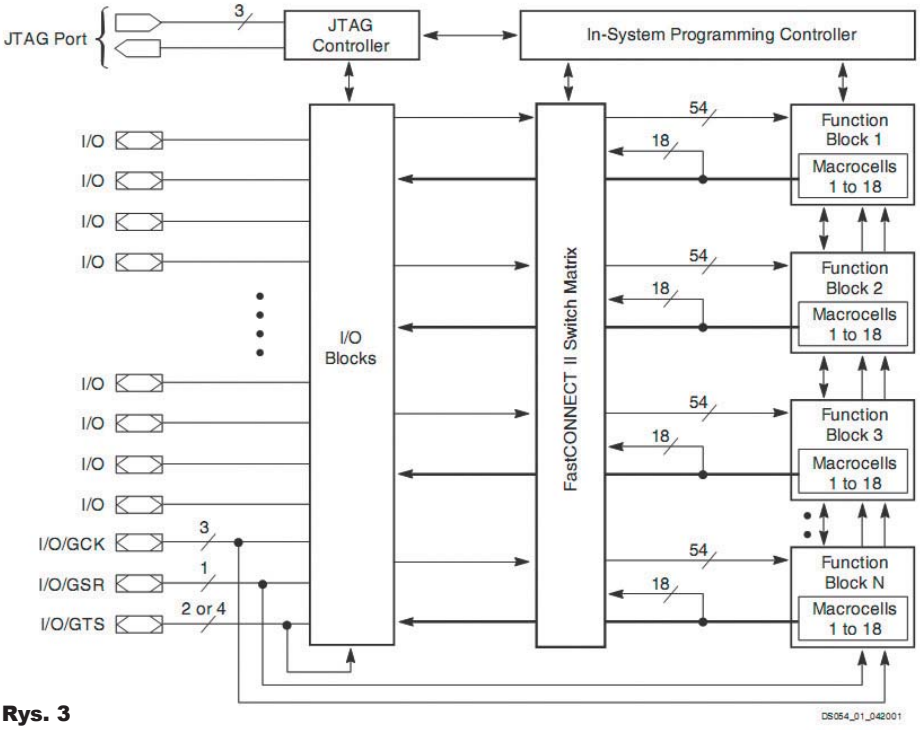
Rodzina układów XC9500XL wymaga zasilania 3,3V i jest dedykowana do wysokowydajnych, niskonapięciowych aplikacji, w których wymaga się dużej niezawodności oraz niskiego zużycia energii. Każdy układ z tej rodziny zapewnia wsparcie dla interfejsu JTAG (IEEE 1149.1) oraz programowania w systemie (ISP). Pozwala to na lepsze wyszukiwanie błędów w programowanie, a pozostawienie dedykowanego złącza dla JTAG na płytce drukowanej umożliwia bezstresową pracę nawet z układami SMD w małych obudowach. Rodzina XC9500XL została stworzona głównie z myślą o współpracy z większymi układami logicznymi, takimi jak FPGA, pozwalając na optymalny podział zadań pomiędzy szybki układ przetwarzający (CPLD) oraz bardziej złożone, ale wolniejsze układy FPGA. Jak pokazano na **rysunku 1**, złożoność układów z serii XC9500XL zawiera się w granicach od 800 do 6400 bramek logicznych z liczbą rejestrów w przedziale od 36 do 288. Dostępne obudowy oraz liczba

portów I/O zostały wymienione w tabeli na **rysunku 2**. Dużą zaletą jest zachowanie zgodności wyprowadzeń pomiędzy poszczególnymi przedstawicielami tej rodziny, pozwalając łatwo zwiększyć dostępne zasoby w razie potrzeby. W płytce prototypowej do niniejszego kursu bez przeszkód można zastosować również układ XC9536XL, który charakteryzuje się jednak mniejszą ilością zasobów.

Jak wspomniałem wcześniej, architektura XC9500XL zapewnia obsługę programowania w systemie (ISP) w pełnym zakresie temperatur. Podwyższona żywotność pamięci i możliwość przypisywania sygnałów do dowolnego portu I/O zapewniają bezstresowe projektowanie. Ma to szczególne znaczenie na etapie projektowania płytki drukowanej, gdyż nie trzeba się przejmować rozmieszczeniem portów – można je dowolnie podłączyć i uwzględnić to na etapie tworzenia oprogramowania. Wydłużony czas podtrzymania danych w pamięci Flash zapewnia bardziej niezawodną pracę urządzenia przez dłuższy czas.

Zaawansowane funkcje systemu dają możliwość kontrolowania czasu narastania zboczy sygnałów wyjściowych oraz programowego dołączania portów do masy, co przekłada się na zmniejszenie zakłóceń w urządzeniu. Wszystkie wejścia są kompatybilne ze standardami napięć 5V, 3,3V oraz 2,5V, natomiast wyjścia mogą pracować w standardzie 3,3V lub 2,5V.

Każdy układ CPLD z serii XC9500XL jest systemem złożonym z wielu bloków funkcyjnych (FB – Functions Blocks) oraz bloków wejścia-wyjścia (IOB – I/O Blocks) połączonych między sobą za pomocą specjalnej matrycy przełączników: FastCONNECT II switch matrix. Bloki IO buforują wejścia i wyjścia układu programowalnego. Każdy z bloków funkcjonalnych zapewnia możliwość programowania 54 wejść i 18 wyjść. Wszystkie sygnały wejściowe i wyjściowe FB mogą być łączone ze sobą za pomocą matrycy przełączników FastCONNECT. Każdy blok funkcyjny posiada do 18 wejść (zależnie od obudowy) oraz przypisane wyjścia umożliwiające bezpośrednie doprowadzenie sygnałów do bloków IO – **rysunek 3**.



Rys. 3

W bloku funkcyjnym, jak pokazano to na **rysunku 4**, zawartych jest 18 niezależnych makrokomórek. Doprowadzone są tu również globalne sygnały zegarowe, wejścia aktywujące oraz sygnały resetujące i ustawiające. Blok funkcyjny generuje 18 sygnałów wyjściowych, które są dołączone do matrycy przełączników. Sygnały te oraz powiązane z nimi sygnały aktywujące mogą być także dołączone do bloków I/O. Funkcje logiczne są implementowane w oparciu o reprezentację sum wynikowych. Wejścia, w liczbie 54, dostarczają 108 sygnałów komplementarnych, które zostają doprowadzone do matrycy bramek AND i tworzą w ten sposób 90 linii product term, które mogą być alokowane do makrokomórek poprzez układ PTA (Product Term Allocator).

Układ CPLD zawiera makrokomórki, które można indywidualnie konfigurować. Pełnią one funkcje kombinacyjne lub rejestrowe. Na **rysunku 5** przedstawiono makrokomórkę i stowarzyszony z nią blok funkcyjny.

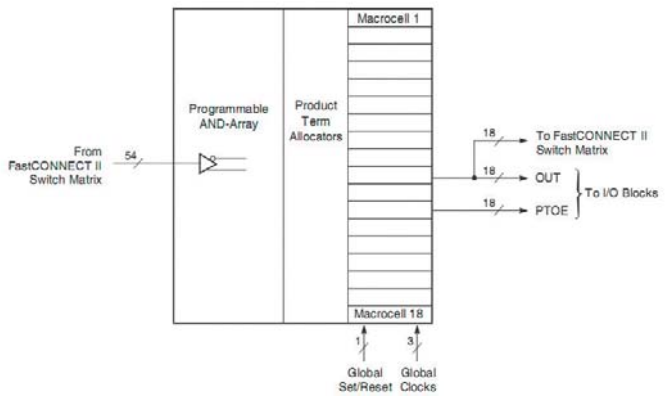
Pięć elementów wynikowych (linie product terms) wychodzących z matrycy AND można wykorzystać jako pierwotne wejścia danych dla bramek OR lub XOR. Umożliwia to implementowanie funkcji kombinacyjnych lub kontrolowanie wejść w tym sygnały zegarowe, ustawiające, kasujące, bramkowanie itp. Elementy te są powiązane z każdą makrokomórką, pozwalając określić sposób wykorzystania pięciu linii product term.

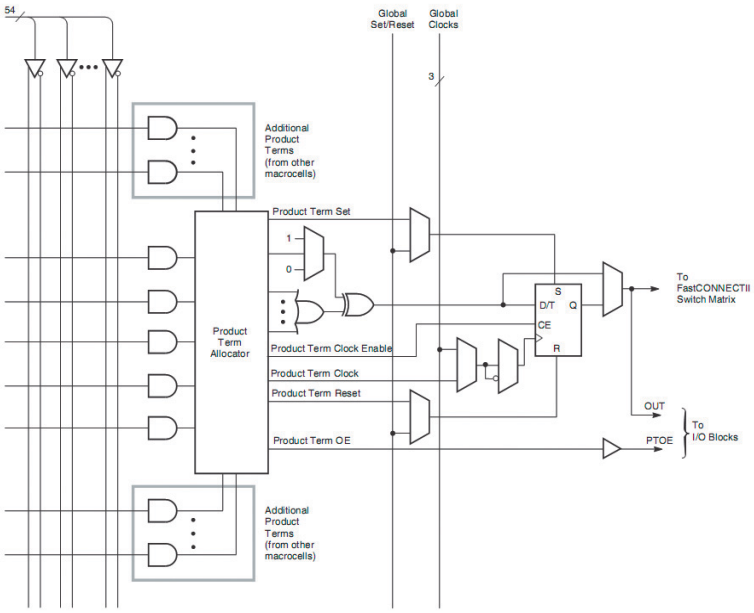
Rejestr makrokomórki może być skonfigurowany jako przerzutnik typu D lub T lub zostać wykorzystany do pełnienia funkcji kombinacyjnych. Każdy z tych rejestrów może być asynchronicznie ustawiany lub kasowany. Po włączeniu zasilania wszystkie

Rys. 2

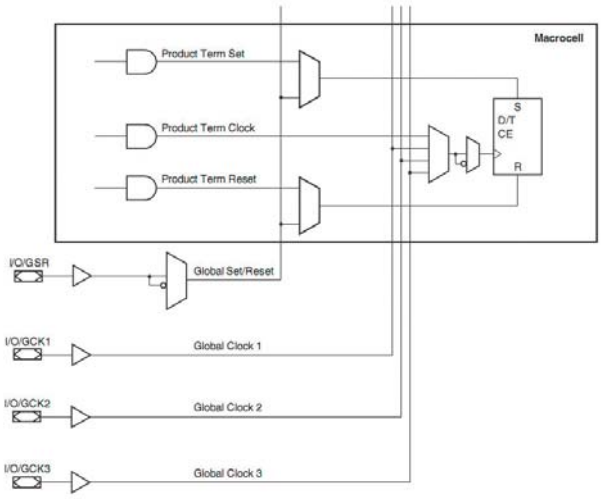
Package(1)	XC9536XL	XC9572XL	XC95144XL	XC95288XL
PC44	34	34	-	-
PCG44	34	34		
VQ44	34	34	-	-
VQG44	34	34		
CS48	36	38	-	-
CSG48	36	38		
VQ64	36	52	-	-
VQG64	36	52		
TQ100	-	72	81	-
TQG100		72	81	
CS144	-	-	117	-
CSG144			117	
TQ144	-	-	117	117
TQG144			117	117
PQ208	-	-	-	168
PQG208				168
BG256	-	-	-	192
BGG256				192
FG256	-	-	-	192
FGG256				192
CS280	-	-	-	192
CSG280				192

Rys. 4



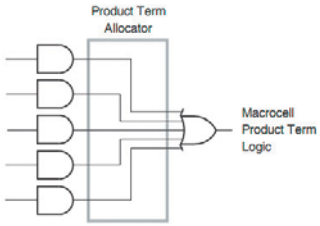


Rys. 5



Rys. 6

Rys. 7

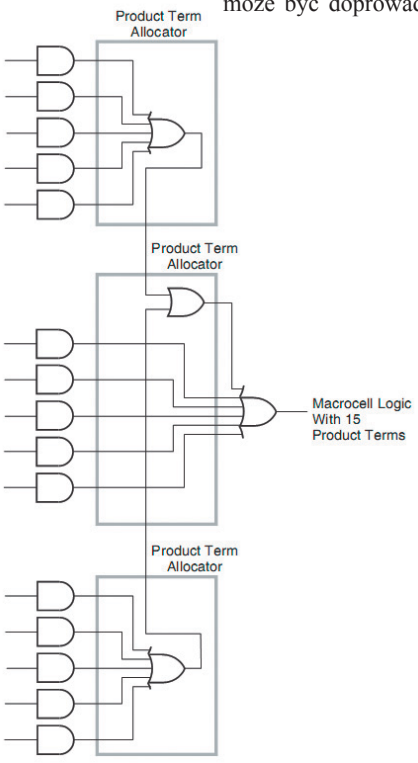


rejestry są inicjowane do stanu określonego przez użytkownika (domyślnie jest to zero).

Wszystkie globalne sygnały kontrolne dostępne są w makrokomórkach (wejścia zegarowe, ustawiające, kasujące oraz aktywujące). Jak pokazano na **rysunku 6** rejestry makrokomórek mogą być taktowane za pomocą jednego z wejść zegarowych lub linii product term. Możliwe jest również wybranie aktywnego zbocza: opadające lub narastające. Wejście GSR umożliwia ustawienie rejestrów w stan zadany przez użytkownika.

Układ Product Term Allocator (PTA) określa w jaki sposób pięć linii product term jest przypisywanych do każdej makrokomórki. Przykładowo wszystkich pięć linii może być wyprowadzonych na wejście bramki OR, jak pokazano to na **rysunku 7**. Układ PTA może dokonać reorganizacji innych linii product term wchodzących w skład bloku funkcyjnego, aby zwiększyć zasoby logiczne makrokomórki. Każda z makrokomórek wymagająca dodatkowych linii product term może wykorzystać wolne linie dostępne w danym bloku funkcyjnym. Jak pokazano na **rysunku 8**, makrokomórka może wykorzystać do 15 linii product term kosztem nieznacznego zwiększenia czasu propagacji sygnału.

Rys. 8



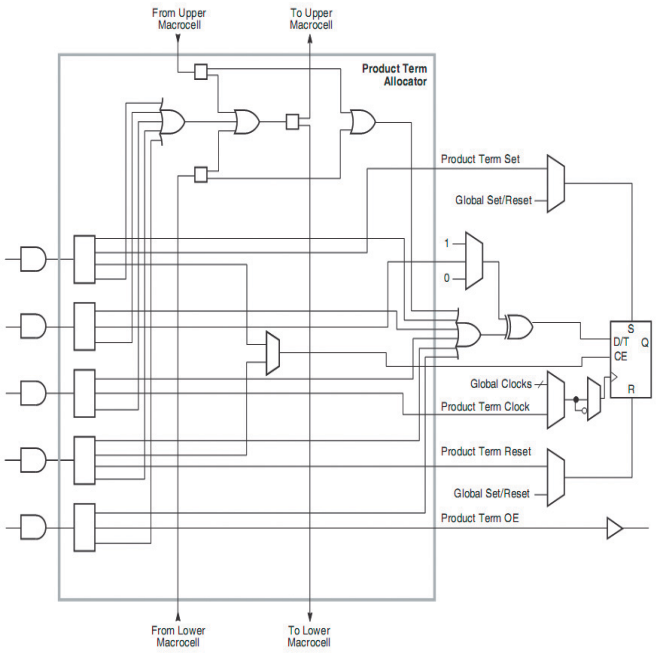
Warto zauważyć, że dotyczy to tylko linii korzystających z zasobów innej makrokomórki.

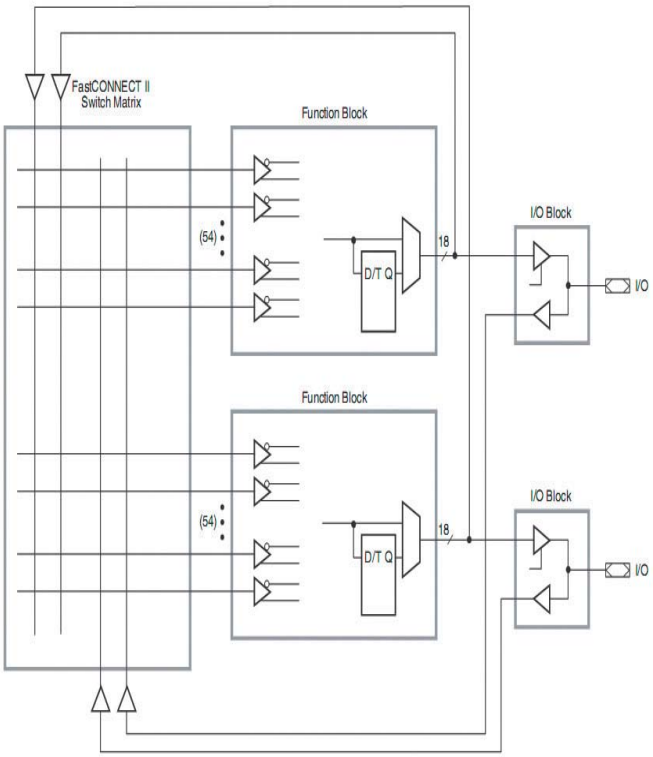
Wewnętrzna logika układu PTA została przedstawiona na **rysunku 9**.

Szybka matryca przełączników (FastCONNECT II Switch Matrix) podłącza sygnały do wejść bloków funkcyjnych, jak pokazano na **rysunku 10**. Wszystkie wyjścia bloku I/O (odpowiadające portom wejściowym) oraz wszystkie wyjścia bloków funkcyjnych stanowią wejścia matrycy FastCONNECT II. Każde z tych wejść może być doprowadzone do wejść bloków funkcyjnych z jednakowym opóźnieniem.

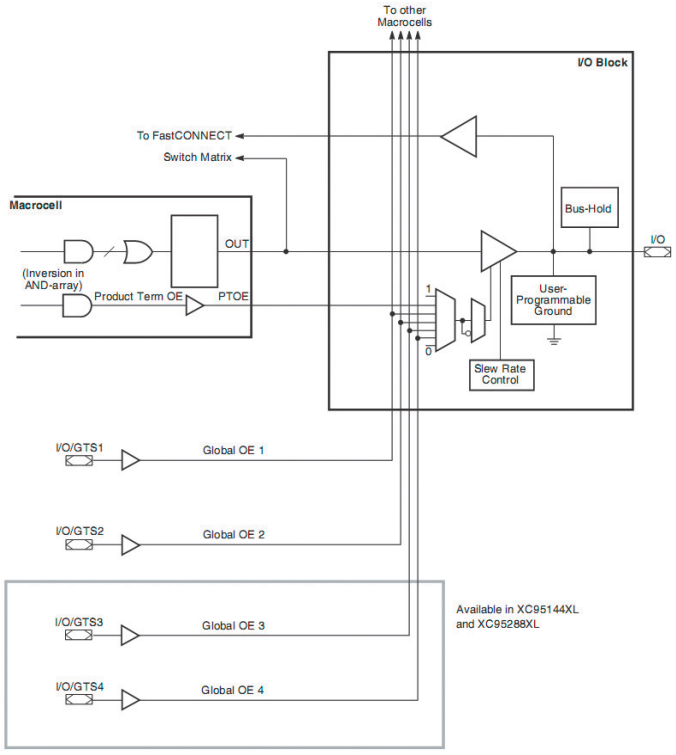
Bloki I/O pośredniczą pomiędzy wewnętrznymi zasobami logicznymi oraz fizycznymi portami I/O. Każdy z nich posiada bufor wejściowy, stopień wyjściowy, multiplekser służący do wyboru aktywnego wyjścia oraz programowalne podciąganie do masy. Szczegóły pokazano na **rysunku 11**. Bufor wejściowy jest kompatybilny z sygnałami w standardzie 5V CMOS, 5V TTL, 3,3V CMOS oraz 2,5V CMOS. Bufor ten wykorzystuje wewnętrzne zasilanie napięciem 3,3V (V_{CCINT}), aby zapewnić stabilny i niezmienny próg przełączenia, który nie będzie zmieniał w przypadku zakłóceń napięcia V_{CCIO}. Każdy z buforów wejściowych zapewnia histerezę przełączenia

Rys. 9



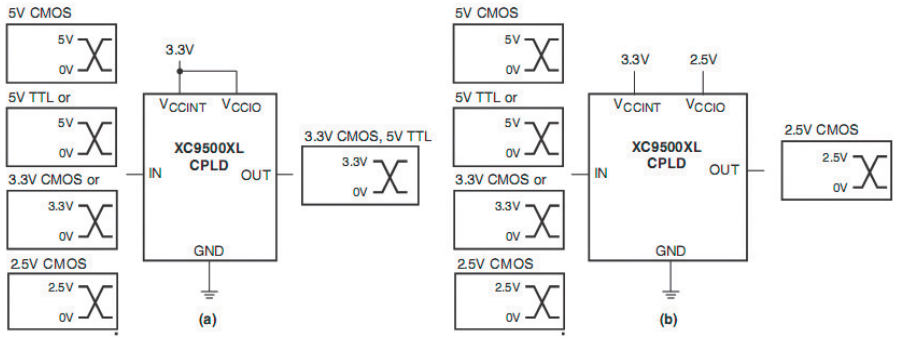


Rys. 10 *Figure 9:* **FastCONNECT II Switch Matrix**



Rys. 11

Rys. 12



na poziomie 50mV, aby pomóc zredukować wpływ szumów w układach z wolno narastającymi lub opadającymi zboczami.

Stopnie wyjściowe zostały zaprojektowane pod kątem szybkiego przełączania przy minimalnym zużyciu energii. Każdy z nich może zostać skonfigurowany do pracy z poziomami napięć 3,3V CMOS (kompatybilnymi również ze standardem 5V TTL) lub 2,5V CMOS. Wybór pomiędzy tymi dwoma poziomami dokonuje się poprzez dołączenie do wejścia V_{CCIO} napięcia 3,3V lub 2,5V. Na **rysunku 12** przedstawiono przykłady współpracy układu CPLD z różnymi standardami napięć.

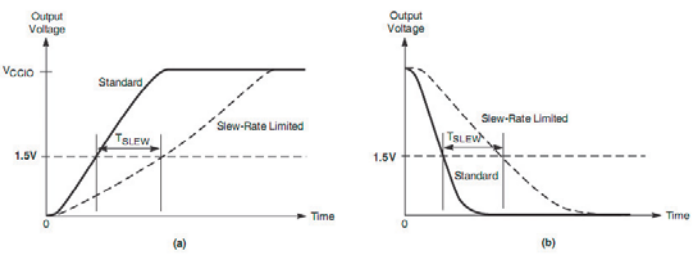
Porty wyjściowe posiadają funkcję zmniejszania stromości zboczy sygnałów – możliwe jest pogorszenie ich jakości kosztem mniejszych zakłóceń generowanych przez układ. Szybkozmienne sygnały charakteryzują się dużą zawartością harmonicznych, co zwiększa zakłócenia radiowe generowane przez pracujące urządzenia. Obowiązujące regulacje prawne nakazują przeciwdziałanie nadmiernym zakłóceniom i spowolnienie zboczy jest jedną z możliwości. Podczas pracy na płytce uniwersalnej nie będzie miało to jednak znaczenia. Rezultat działania spowalniania pokazano na **rysunku 13**.

Sygnał wyjściowy na porcie I/O może pochodzić z linii product term, linii GTS lub można na stałe wymusić stan logiczny 1 lub 0. Te dwie ostatnie opcje będą użyteczne przy pierwszym kontakcie z płytką prototypową, gdyż pozwolą w prosty sposób włączyć np. diodę i tym samym określić, czy układ CPLD pracuje prawidłowo i daje się programować. Dla układów wyposażonych w 72 lub mniej makrokomórek przewidziano dwie linie GTS, a dla zawierających 144 lub więcej makrokomórek – cztery. Każdy z tych sygnałów może być lokalnie zanegowany, co prowadzi do zwiększenia elastyczności projektu.

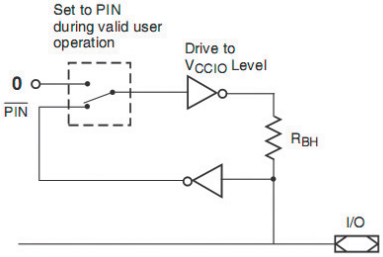
Każdy z bloków I/O umożliwia programowe dołączenie wybranego pinu do masy. Pozwala to ustawić porty w stan niski, co może zapewnić lepszą pracę układu. Podciąganie do masy jest realizowane przez wewnętrzną logikę, która wymusza stan niski niezależnie od sygnału pochodzącego z makrokomórki. Logika wewnętrzna makrokomórki jest w tym wypadku ignorowana.

Układy z rodziny XC9500XL zostały wyposażone w obwody podtrzymujące stan wyjściowy portów. Funkcja ta elimi-

Rys. 13



Rys. 14



nuje konieczność definiowania stanu nieużywanych pinów, gdyż na wyjściu jest podtrzymywany ostatni znany stan do momentu pojawienia się nowego. Układ ten zapewnia sprzężenie stanu wejściowego poprzez rezystor o wartości 50kΩ, pokazano to na **rysunku 14**. Warto zauważyć, że napięcie wyjściowe na porcie nie przekracza wartości V_{CCIO}, co zapobiega uszkodzeniu układów pracujących z napięciem 2,5V.

		Read Security	
		Default	Set
Write Security	Default	Read Allowed Program/Erase Allowed	Read Inhibited Program Inhibited Erase Allowed
	Set	Read Allowed Program/Erase Allowed	Read Inhibited Program/Erase Inhibited

Rys. 15

W przypadku, gdy stan portu nie został określony, obwód podtrzymujący zapewnia funkcję podciągania portu przez rezystor 50kΩ, tak aby zapewnić znany stan. Sytuacja taka zachodzi podczas programowania, skasowania pamięci układu CPLD, wprowadzenia w tryb INTEST przez interfejs JTAG lub podczas inicjacji po włączeniu zasilania. Można dołączyć również rezystor 1k podciągający port do masy, aby ominąć działania domyślnego rezystora podciągającego lub wymusić stan niski podczas inicjacji układu lub w którymś z wymienionych trybów.

Ciekawą cechą układów XC9500XL jest możliwość dowolnego przypisywania i późniejszej zmiany przyporządkowania sygnałów do portów, co pozwala w prosty sposób rozwiązać problem modyfikacji projektów. W architekturze omawianego układu CPLD zaimplementowano mechanizmy pozwalające na dowolne zarządzanie takimi zmianami, przy zachowaniu jednakowego rozkładu wyprowadzeń.

Warto zdawać sobie sprawę, iż układy XC9500XL zostały wyposażone w mechanizmy zapobiegające nieautoryzowanemu odczytowi oprogramowania znajdującego się w pamięci oraz możliwość kasowania i ponownego zapisu. Dostępne tryby ochrony zostały przedstawione w tabeli na **rysunku 15**. Użytkownik może ustawić bity zabezpieczające odczyt, aby uniemożliwić skopiowanie oprogramowania z układu CPLD. Uniemożliwia to przy okazji przeprogramowanie układu, jednakże pozostawia możliwość jego skasowania, które jest jedyną drogą do wyzerowania bitu zabezpieczającego odczyt.

Drugi z bitów, zabezpieczający układ przed zapisem, wprowadza przy okazji ochronę przez przypadkowym skasowaniem pamięci np. na skutek zakłóceń pojawiających się na wejściach portu JTAG podczas włączania zasilania. Po ustawieniu tego bitu można go zdeaktywować jedynie odpowiednią sekwencją instrukcji przesłanych przez port JTAG.

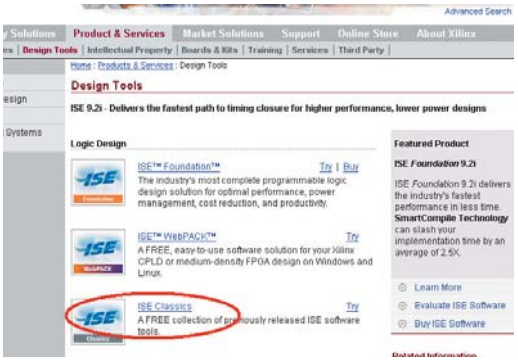
Instalacja środowisa WebPack ISE 6.2

Jak wspomniano wcześniej, środowisko, którym będziemy się posługiwać, jest bezpłatne i gotowe do pobrania ze strony producenta, czyli www.xilinx.com. Aktualnie na stronie dostępna jest wersja 9.2i, z której nie będziemy jednak korzystać, zadowalając się wcześniejszym produktem z numerem 6.2i. Ta wersja oprogramowania będzie w zupełności wystarczająca do nauki, a co istotne, ma pewną poważną zaletę: instalator wcześniejszej wersji ma rozmiar 186MB, a nowej 1,7GB.

Chcąc ułatwić rozpoczęcie kursu również młodym Czytelnikom, którzy nie znają angielskiego, przedstawię krok po kroku sposób rejestracji na stronie Xilinksa, pobrania potrzebnego oprogramowania oraz sposób instalacji.

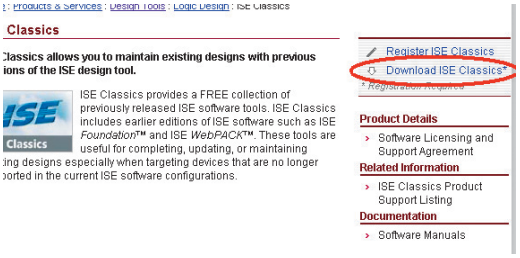
Po załadowaniu strony www.xilinx.com należy odnaleźć link *Design Tools* i kliknąć go. Po załadowaniu kolejnej strony wybieramy link *ISE Classic* (**rysunek 16**), a następnie *Download ISE Classic* (**rysunek 17**). Po załadowaniu kolejnej strony zostaniemy poproszeni o zalogowanie się. Konieczne jest założenie konta, więc klikamy *Create Account* (**rysunek 18**). Naszym oczom ukaże się formularz, który należy wypełnić według wzoru na **rysunku 19**. Uwaga! Pole *Password* musi zawierać hasło składające się z minimum 7 znaków, w tym jednej cyfry. Po wypełnieniu formularza klikamy *Create Account*. Po kilku chwilach na podany adres e-mail zostanie wysłany link, który należy kliknąć, celem dokończenia tworzenia konta.

Teraz można powrócić do strony pokazanej na rysunku 18 i wprowadzić login oraz hasło. Niestety nie jest to koniec, gdyż uzyskanie dostępu do oprogramowania wymaga rejestracji. W tym celu klikamy na *Register to gain access to content* (**rysunek 20**) i ukazuje się kolejny formularz



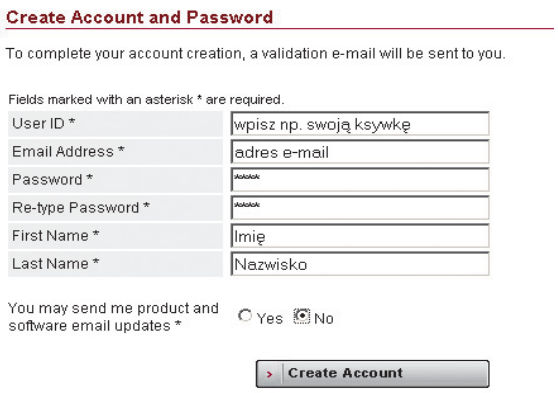
Rys. 16

Rys. 17



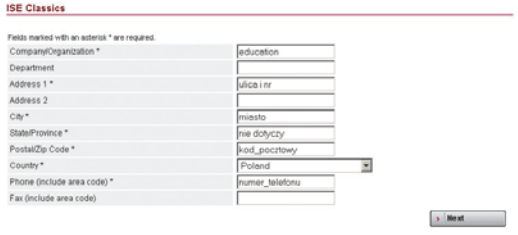
Rys. 18

Rys. 19

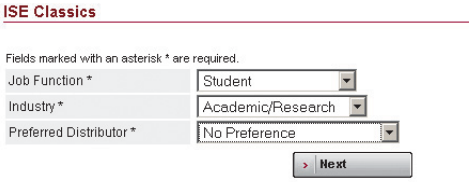


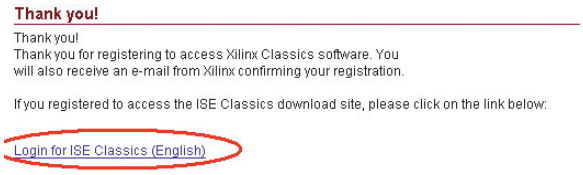
Rys. 20

Rys. 21



Rys. 22

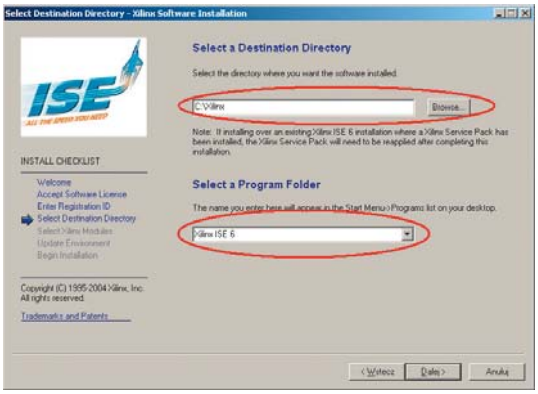




Rys. 23

Rys. 24

Rys. 25



Rys. 26

(**rysunek 21**), który wypełniamy według wzoru. Po wypełnieniu wybieramy *Next*. Kolejny formularz i wzór jego wypełnienia zamieszczono na **rysunku 22**. Ukazuje się informacja z **rysunku 23** i klikając na link *Login for ISE Classic*, zostajemy przeniesieni do strony z oprogramowaniem. Na **rysunku 24** zaznaczono pole, które należy kliknąć, aby pobrać wymagane oprogramowanie. Na następnej stronie wyświetli się lista wszystkich dostępnych wersji, należy na niej odnaleźć link z wersją 6.2i (**rysunek 25**) i kliknąć go, co zainicjuje ściąganie instalatora.

Po pobraniu ze strony firmy Xilinx oprogramowania, nadeszła pora, by je zainstalować. W tym celu uruchamiamy instalator (*WebPACK_62_fcfull_i.exe*) i czekamy kilka chwil na jego uruchomienie. Pierwsze, co należy zrobić, to zaakceptować warunki licencji i kliknąć przycisk *Dalej*. W kolejnym okienku pokazanym na **rysunku 26** wybieramy folder (pierwsza czerwona obwódka), w którym ma być zainstalowane oprogramowanie. Należy pamiętać o bardzo ważnej rzeczy: NIE WOLNO STOSOWAĆ SPACJI! Przeoczenie tego szczegółu uniemożliwi pracę środowiska WebPack ISE. Druga czerwona obwódka wskazuje miejsce, w którym określa się, jaki wpis ma się pojawić w menu Start. Po ustawieniu tych atrybutów przechodzimy do dalszego okienka i zostawiamy zaznaczone obie opcje. Kolejne okienko jest podsumowaniem dokonanych wcześniej wyborów i wymagane jest tu jedynie kliknięcie przycisku *Instaluj*.

Rozpocznie się proces instalacji, którego postęp można śledzić za pomocą paska postępu. W tym momencie można udać się do kuchni i zrobić herbatę, gdyż instalacja zajmie kilka minut.

Po zakończeniu instalacji instalator zapyta, czy chcemy zainstalować sterowniki dostarczane przez firmę Xilinx (**rysunek 27**). Wybieramy opcje NIE, gdyż nie będą one potrzebne podczas pracy, a w czasie instalacji może dojść do uszkodzenia już istniejących sterowników systemowych i wymagana może być ich reinstalacja.

Pierwszy kontakt z nowo zainstalowanym oprogramowaniem będzie sprowadzał się do zaprogramowania układu CPLD celem sprawdzenia poprawności montażu płytki prototypowej. Szczegóły z tym związane zostały przedstawione w artykule opisującym budowę płytki.

Rys. 27

Podsumowanie

Niniejsza część kursu jest jedynie długim wstępem, który miał za zadanie nakreślić kierunek, w jakim będzie on zmierzał, przedstawić ogólny rys architektury układu CPLD oraz przygotować narzędzia do dalszej pracy. Za miesiąc zaczniemy budować pierwsze układy i testować ich działanie. Czas ten warto wykorzystać na samodzielne zbudowanie płytki testowej lub jej zakup w sklepie AVT i spokojne uruchomienie, aby z pojawieniem się następnego numeru EdW można było od razu rozpocząć pierwsze próby.

Najbardziej niecierpliwi Czytelnicy mogą spróbować wprowadzić własne modyfikacje do udostępnionego testu lub rozpocząć pierwsze próby we własnym zakresie. Uszkodzenie płytki uniwersalnej jest praktycznie niemożliwe, a możliwość programowania jej 10 000 razy daje komfort psychiczny podczas prowadzenia prób. Zapraszam za miesiąc do dalszego zagłębiania tajemnic układów CPLD.

Jakub Borzdyński
jakub.borzdynski@elportal.pl