EMBEDDED & IT NEWS **ISSN 1733-036X**
Nieperiodyczny biuletyn nowości technicznych dla partnerów handlowych QTTC

Wstęp do testu obwodów drukowanych systemem XJTAG Boundary Scan

Spis treści

Cześć teoretyczna

Cześć praktyczna

Projektowanie płytek drukowanych z uwzględnieniem systemu testującego bazującego na technice JTAG Boundary Scan - wskazówki dla projektantów (DFT- Design For Testability)

Dział Marketingu, KTT Quantum

Cześć teoretyczna

1. Wstęp

Wielu inżynierów zdaje sobie sprawę jak trudne jest wykonanie testów wysoko zaawansowanych, współczesnych systemów elektronicznych. Szukając najbardziej odpowiedniej metody, nie raz spotkali się z techniką JTAG Boundary Scan.

Technika te jest znana już od początków lat 90 ale mimo tego, jest wciąż mało znana, szczególnie na rynku polskim.
Możliwości JTAG Boundary-Scan wykraczają poza zakres obecnie stosowanych technik, dla których testowanie płytek obustronnie montowanych z wykorzystaniem montażu powierzchniowego, jest prawie niemożliwe. Wymagania dzisiejszej elektroniki podążają w kierunku miniaturyzacji, obniżenia cen, jak również skrócenia czasu wprowadzania nowych produktów na rynek i poprawy ich jakości. Często dynamiczne wprowadzanie nowych produktów, nie idzie w parze z wymaganą jakością. W przypadku systemów elektronicznych problem tkwi w przeprowadzeniu odpowiednich procedur testujących w całym cyklu powstawania produktu. Testy te powinny obejmować jakość montażu w tym poprawność połączeń oraz sprawdzać niezawodność pracy elementów elektronicznych.

Część teoretyczna tej publikacja wprowadza nas w techniczne zagadnienia JTAG Boundary Scan oraz przedstawia nieocenione możliwości testowania układów na tle innych technik.

2. Zrozumieć JTAG Boundary Scan

JTAG jest techniką istniejącą od ponad dekady ale dopiero teraz jest możliwości jako narzędzia do testowania oraz programowania zostały dopiero teraz w pełni docenione. Testowanie oraz programowanie układów (ISP) to dwa zastosowania najbardziej powiązane z interfejsem JTAG. Obecnie technika JTAG oferuje nam znacznie więcej.

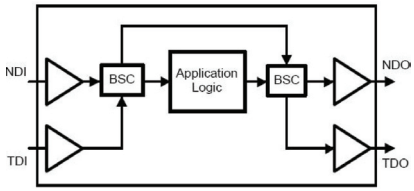
W tej części zostanie przedstawiona dokładna budowa architektury JTAG Boundary Scan, zostaną opisane zachodzące w niej procesy oraz jakie ma to odniesienie do rzeczywistego wykorzystania w procesie testowania.

2.1 Boundary – Scan (IEEE Std 1149.1)

IEEE1149.1 jest protokołem określającym budowę struktury JTAG Boundary-Scan, opracowanym przez współpracujących ze sobą ponad 200 grup inżynierskich z różnych firm.
Pierwszymi współpracownikami w rozwoju standardu IEEE1149.1 byli **AT&T, DEC, Ericsson, IBM, Nixdorf, Philiphs, Siemens** oraz **TI**. Korporacje te uznały, że jedynie opracowanie architektury bez tytułu własności, zachęci pozostałe korporacje do wykorzystania tej techniki w implementacjach układowych, narzędziach testujących, narzędziach typu CAD oraz przyczyni się obniżenia kosztów procedur testowych.
Technika JTAG Boundary-Scan opiera się na specjalnym rodzaju ścieżki testującej, usytuowanej na brzegach układu z wbudowanymi rejestrami połaczonymi do każdego wyprowadzenia w układzie.
Największą korzyścią zastosowania techniki boundary-scan jest możliwość wyszukiwania błędów na najniższym poziomie abstrakcji technicznej a mianowicie na poziomie pinów układu. Głównym motywem opracowania techniki boundary-scan, były trudności testowania płytek drukowanych, do budowy których stosowano techniki montażu powierzchniowego. Implementacja ścieżki boundary-scan w układach elektronicznych dawało jedyną możliwość przeprowadzenia testów połączeń pomiędzy elementami.
Idea tej techniki jest prosta. Polega ona na wysłaniu z układu, za pośrednictwem ścieżki boundary-scan, sygnałów o znanych wartościach a następnie, również za pośrednictwem ścieżki boundary-scan innego układu, odebrać te sygnały.
Jeżeli wartości są sobie równe oznacza to poprawność połączenia pomiędzy tymi układami. Jest to najprostszy przykład jaki może zostać zainicjowany.
Osiągnięcia tej techniki pozwalają na wykonanie skomplikowanych algorytmów testowych, które umożliwiają prze-rowadzenie testów układów bez wbudowanej sieci JTAG Boundary-Scan. Metoda ta może wskazać typowe błędy montażu takie jak, **przerwania, zwarcia, zimne luty lub elektryczne naładowanie (ESD).**
Obecnie technika boundary-scan jest dynamicznie rozwijana a narzędzia bazujące na tej technice umożliwiają, oprócz przeprowadzenia testów połączeń, to również wykonać zaawansowane testy dowolnych pamięci półprzewodnikowych, kontrolerów Ethernet, kontrolerów wideo, diod LED, buforów, układów z wbudowaną szyną IIC i wiele inych.

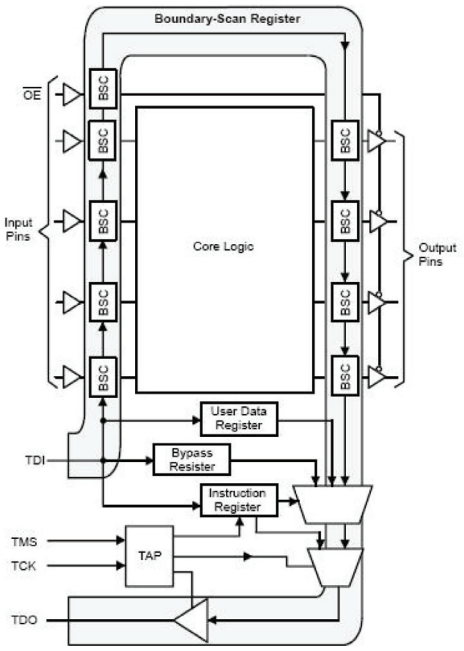
2.2 Architektura Boundary – Scan

Podstawą architektury Boundary-Scan jest ścieżka boundary-scan, umiejscowiona na krawędziach układu, przed wyprowadzeniami. Ścieżka boundary-scan złożona jest z komórek BSC (Boundary Scan Cells) połączonych szeregowo z równoległymi wejściami i wyjściami. Komórki BSC stanowią wewnętrzne połączenie pomiędzy ścieżką testującą boundary-scan a rdzeniem układu. Podczas normalnej pracy układu, komórki są przezroczyste dla sygnałów przechodzących przez układ. Natomiast gdy układ jest w trybie testowania, wartości sygnałów przechwytywane są przez komórki BSC. Praca sieci boundary-scan może pracować w kilku trybach. Jednym z trybów jest praca w trybie testowym, w którym sygnały wprowadzane są przez wejście TDI, przesuwane szeregowo do odpowiedniej komórki BSC, zatrzaskiwane a następnie wysyłane przez wyjście NDO. W przypadku gdy układ jest w trybie analizowania (próbkowania) sygnały docierające do wejścia NDI, są przechwytywane przez komórki BSC a następnie przesuwane szeregowo do wyjścia ścieżki testującej TDO (rys.1).



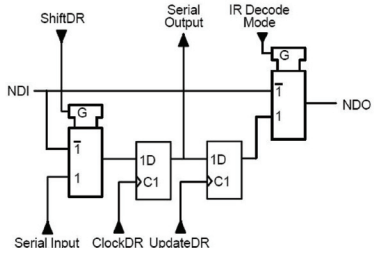
Rys. 1. Uproszczony schemat architektury Boundary Scan

W ten sposób można symulować normalne działanie układu, wysłać dowolne komendy do układów peryferyjnych a następnie odczytać otrzymane odpowiedzi. Sterowanie przepływem sygnałów wewnątrz struktury odpowiedzialne jest główny kontroler TAP (Test Access Port). To on decyduje w jakim trybie pracy znajduję się JTAG Boundary-Scan oraz przekierowuje dane do odpowiednich rejestrów. Kontroler TAP do dyspozycji ma kilka rejestrów, które podzielić można na rejestry instrukcji oraz danych. Jednym z rejestrów danych to BSR (Boundary-Scan Register). W skład, którego wchodzą ścieżka z komórkami BSC, połączone ze sobą szeregowo z równoległym



Rys. 2. Schemat pełnej architektury łańcucha boundary-scan

wejściem i wyjściem, łączącymi wyprowadzenia układu z głównym rdzeniem. Podstawowa wewnętrzna architektura pojedynczej komórki BSC zbudowana jest z przerzutników i multiplekserów. Całość stanowi rejestr szeregowy z równoległym wejściami i wyjściami (rys. 3). Kolejnym ważnym rejestrem jest rejestr instrukcji Register Instructions (IR). Rejestr IR przechowuje rozkazy (instrukcje), zawierające informacje o trybie pracy boundary-scan, wykonywanych operacjach oraz miejscu przekazywania danych.Pozostałe rejestry zaliczają się do rejestrów danych, należą do nich Bypass Register oraz User Data Register.



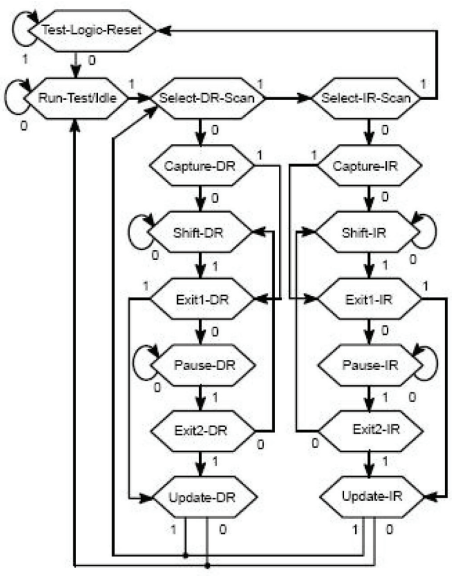
Rys. 3. Wewnętrzny schemat struktury pojedynczej komórki BSC

Bypass Register jest to rejestrem jedno bitowym, wprowadzenie do niego wartości '1' powoduje aktywacje i całkowite odseparowanie linii TDI - TDO od pozostałych rejestrów tworząc w ten sposób połączenie TDI-Bypass-Register-TDO.

User Data Register może przechowywać rejestry identyfikujące układu (ang. Device Identification Register). Zawierają w sobie informacje o numerze identyfikacyjnym lub seryjnym układu, wersji oraz ewentualnie dodatkowe oznaczenia.

2.3 Przepływ danych

Praca łańcucha boundary-scan zarządzana jest przez interfejs TAP, sterowanego sygnałem zegarowym TCK (Tesc Clock) oraz sygnałem TMS (Test Mode Selection). Wewnętrzna budowa TAP złożona jest z mniejszych kontrolerów, który każdy z nich ma przypisany jeden rodzaj wykonywanej funkcji. Pracę kontrolera TAP można przedstawić w postaci diagramów stanu (rys. 4). Diagram odzwierciedla poszczególne stany w jakich znajduje się sieć Boundary-Scan oraz fazy przejść pomiędzy nimi. Każde przejście z jednego stanu na drugi sterowane jest sygnałem TMS, próbkowanym zboczem narastającym zegara TCK. W momencie uruchomienia układu lub podczas normalnej pracy, w kontrolerze TAP wymuszony jest stan Test-Logic-Reset.



Rys. 4. Schemat diagramów stanu interfejsu TAP

W tym stanie układ jest w trybie normalnej pracy a kontroler TAP nie blokuje działania rdzenia układu. Stan ten trwa do momentu przejścia sygnału TMS z wysokiego na niski, wówczas układ przechodzi w tryb testowy i osiąga stan Run- Test/Idle. Następnie w zależności od tego czy wprowadzane dane dotyczą instrukcji czy danychosiągany jest stan Select-DR-Scan (dane) lub stanSelect-IR-Scan (instrukcje). Następujące po sobie instrukcję stanów Select-DR-Scan oraz Select-IR-Scan są odbiciem lustrzanym ale wykonują nieco inna pracę. Najczęściej pierwszym stanem po rozpoczęciu testów jest stan instrukcji a dopiero następuję przejście do stanu wprowadzania danych do miejsca określonego przez instrukcję.

Dane mogą zostać przekierowane do rejestrów BSR, Bypass Register lub User Register.

Rodzaje instrukcji

BYPASS: Instrukcja aktywuje jedno bitowy rejestr *Bypass Register* i podłącza go do linii TDI-TDO. Dane wprowadzane przez wejście TDI będą kierowane do *Bypass Register*. Instrukcja ta powoduje całkowite wyłączenie ścieżki boundary-scan oraz pozostałych rejestrów od wprowadzanych danych.

SAMPLE/PRELOAD: Instrukcja ta aktywuje rejestr BSR (komórki BSC), tworząc połączenie TDI-BSR-TDO. W tym trybie układ pracuje normalnie natomiast sieć boundary-scan pracuje w trybie próbkowania (analizowania). Przechodzące sygnały przez układ przechwytywane są przez BSC i szeregowo przesyłane do wyjścia TDO. Tryb pracy w jaki jest włączona sieć boundary-scan analogiczna jest to pracy analizatora sygnałów.

EXTEST: Instrukcja ta wymusza nadrzędność łańcucha boundary-scan nad głównym rdzeniem układu. Następuje przejście w stan testowy. W tym trybie, również wprowadzany rejestr BSR w stan aktywny. Operacja testowania polega na wprowadzeniu przez wejście TDI danych (sygnałów) do komórek BSC następnie wysyłane są na zewnątrz układu np. do pamięci Flash. Możliwe jest również w tym trybie przechwytywać wracające odpowiedzi i kierować do wyjścia TDO. Stan ten można porównać do współpracy analizatora i generatora sygnałów.

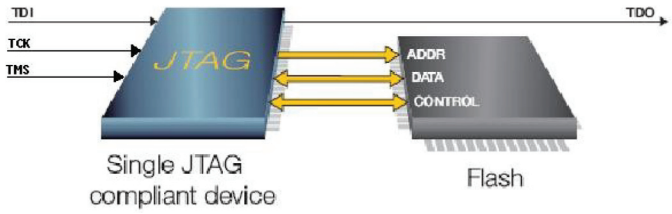
Przepływ stanów:

Instrukcje Select-IR-Scan

- ***Capture-IR*** - następuję podłączenie do wejścia TDI rejestru instrukcji IR. Powstaje połączenie TDI-IR-TDO
- ***Shift-IR*** - następuję przesuwanie bitu po bicie wartości instrukcji od wejścia TDI do rejestru IR.
- ***Exit1-IR*** - osiągnięcie tego stanu oznacza zakończenie wprowadzania instrukcji
- ***Update-IR*** - następuję zatrzaśniecie wprowadzonej instrukcji w rejestrze *Shadow Latch* a następnie jej wykonanie
- ***Pause-IR*** – aktywacja tego stanu następuje w momencie wystąpienia przerwania (np.: uzupełnienie bufora danych)
- ***Exit2-IR*** – w momencie wyjścia z przerwania następuje powrót do stanu Shift-IR lub przejście do stanu Update-IR
  Instrukcje Select-DR-Scan (na przykładzie trybu EXTEST)
- ***Capture-DR*** - następuję podłączenie do wejścia TDI rejestru BSR (komórek BSC).
- ***Shift-IR*** - następuję przesuwanie bitu po bicie wartości danych do rejestrów komórek BSC od wejścia TDI w kierunku wyjścia TDO
- ***Exit1-IR*** - zakończenie wprowadzania danych
- ***Update-IR*** - następuję równoległe zatrzaśniecie wprowadzonych danych w rejestrze *Shadow Latch* (rejestry I/O) oraz wyprowadzenie na zewnątrz układu.

2.4 Jak wygląda to w praktyce

Podstawowym warunkiem zastosowania techniki JTAG Boundary Scan jest implementacja łańcucha JTAG Boundary-Scan w co najmniej jednym układzie. Im więcej układów z implementacją JTAG tym łatwiej i szybciej przygotować procedurę testową a obszar testowania płytki zwiększa się. Zagadnienia te wymagają uwzględnienia w projektowaniu płytki, tak zwane podejście „Design For Test".
Technika ta jest mało powszechna dlatego też projekty płytki nie uwzględniają tej techniki jako metody późniejszych testów.
Najczęściej spotykaną koncepcją jest zastosowania, jako głównego procesora, układu z implementacją JTAG Boundary Scan, który będzie testował pozostałe układy peryferyjne pozbawione łańcucha boundary-scan.
Przykładem może być połączenie głównego układu JTAG z pamięcią Flash (rys. 5). W sytuacji tej w pełni można wykorzystać możliwości techniki JTAG Boundary-Scan i wykonać test poprawności połączenia pomiędzy tymi układami jak i sprawdzić poprawność działania Flash.



Rys. 5. Przykład połączenia układu JATG z pamięcią Flash

Najprostszym sposobem na sprawdzenie poprawności współpracy z pamięcią Flash jest zapisanie do niej pewnej znanej wartość a następnie odczytanie. Jeżeli dane są identyczne, test można uznać za zakończony pomyślnie. Komunikacja pomiędzy pamięcią odbywa się poprzez wygenerowanie odpowiednich sekwencji sygnałów z wyprowadzeń procesora za pośrednictwem komórek BSC.
Układ musi być wówczas w trybie testowania (EXTEST).
W taki sposób można komunikować się prawie z każdym elementem znajdującym się na płytce. Bardziej optymalną sytuacją jest gdy na płytce znajduje się więcej układów z implementacją JTAG Boundary-Scan, wówczas przeprowadzone testy są łatwiejsze w opracowaniu.
Do przeprowadzenia tego rodzaju testy należy wykorzystać w tym celu sprzęt wspierający technikę JTAG Boundary-Scan.

Na rynku polskim dostępny jest system XJTAG należący do brytyjskiej firmy o tej samej nazwie XJTAG. Jest to zaawansowane narzędzie wspierające pracę projektanta, testera oraz producenta płytek.
Firma XJTAG została założona przez grupę naukowców z Cambridge i obecnie jest bardzo dynamicznie rozwijana. Współpracuje z największymi korporacjami takimi jak **ARM, Aeroflex, Thales** i wiele innych.

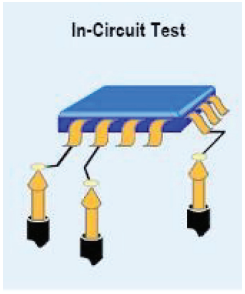
Podsumowanie

Obecny postęp technologiczny w elektronice przyczynił się do powstawania zaawansowanych systemów elektronicznych ale również zapoczątkował rozwój technik testowania. Dodatkowo poprzeczka podwyższana jest przez coraz wyższe wymagania związane z niezawodnością produktów i obniżenia ich cen.
W tym właśnie celu rozwijana jest technika JTAG Boundary-Scan, która dzięki swoim możliwością spełnia powyższe wymagania.
Koszty wprowadzenia tej techniki w sektorze firmy są niewielkie w porównaniu z obecnie dostępnymi technikami takimi jak X-Ray, AOI (Automated Optical Inspection) lub ICT (In-Circuit-Test) a przynosi ona niewspółmierne korzyści finansowe i czasowe podnosząc konkurencyjność końcowych produktów.

3. Alternatywa testowania współczesnych zintegrowanych układów elektronicznych

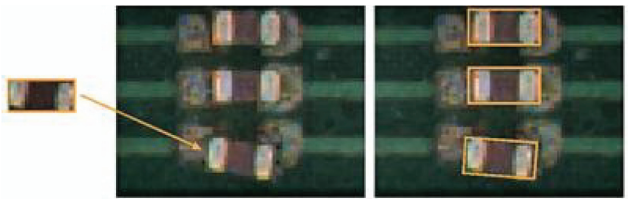
Współczesne systemy elektroniczne zbudowane są układów charakteryzującymi się wysoką skalą integracji z setkami wyprowadzeń przypadając na jeden układ. Idea ta wynika z tendencji do miniaturyzacji w dziedzinie technologii elektronicznych. Trend ten pociąga również za sobą wykorzystywanie montażu powierzchniowego, dwustronnego, układów zamkniętych w obudowach typu BGA (Ball Grid Array). Z pewnością, zaletą układów BGA jest mniejsza powierzchnia zajmowana na płytce, natomiast mają też kilka wad, które zniechęcają projektantów do ich wykorzystania. Do wad tych należy zaliczyć zjawisko pękania kulek podczas lutowania na skutek naprężeń termicznych jak również trudności w ich testowaniu.
Ze względu na to, że obecne techniki testujące nie spełniają stawianych im wymagań, są kosztowne i skomplikowane, testy systemów elektronicznych są traktowane lekceważąco.

3.1 Dostępne techniki testowania



Rys. 6. Schemat metody ICT

Obecnie stosowane techniki można podzielić na trzy grupy metod: kontaktowe, bezdotykowej inspekcji optycznej lub wbudowane .
W obliczu dzisiejszych trendów elektronicznych najmniej pożyteczną wydaja się metody kontaktowe takie jak ICT (in-circuit-test). Metoda ICT należy do metod bardzo skomplikowanych i drogich a przygotowanie procedury testowej trwa od czterech do sześciu tygodni. Narzędzia tej metody wykorzystują matrycę szpilek, które się dociska do wyprowadzeń układów na płytce (rys. 1).
Metoda ta już przy starciu z płytką dwustronnie montowana lub z układami BGA , przegrywa.
Do technik kontaktowych można jeszcze zaliczyć metody manualne z wykorzystaniem generatora i oscyloskopu.



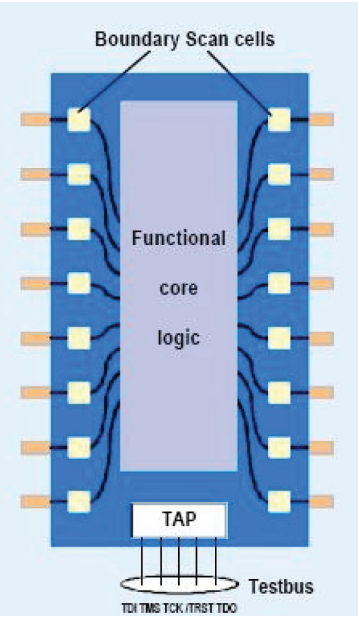
Rys. 7. Zdjęcie wadliwego montażu wykonanego za pomocą systemu AOI

Metody są znacznie tańsze, nie wymagają skomplikowanego sprzętu i są łatwe w obsłudze ale nie podlegają one procesowi automatyzacji, są trudno powtarzalne, czasochłonne i wymagają dużego nakładu pracy. Ręczne metody mogą jedynie stanowić uzupełnienie pozostałych metod a nie tworzyć trzon procesu testowania.

Pewną alternatywę stanowią techniki bezkontaktowe tzw. optyczne, typu AOI (Automated Optical Inspection) lub X-ray. Stosując te techniki możemy otrzymać odpowiedzi na jakość połączeń, błędy montażowe, pozostałości po pastach lutowniczych lub pęknięcia (rys. 2).
Należy pamiętać, że powyższe rezultaty można otrzymać poprzez dokładną analizę zdjęć otrzymanych ze specjalnych narzędzi. Zdjęcia te analizowane są manualnie przez wyspecjalizowane osoby lub automatycznie przez inteligentne programy. Analiza manualna nie sprawdza się przy dużej liczbie elementów znajdujących się na płytce, jak również na linii produkcyjnej. Ratunkiem są inteligentne programy analizujące ale ceny ich są bardzo wysokie oraz wymagają aktualizacji wraz z naniesionymi zmianami na płytce.
Często firmy produkujące systemy elektroniczne stosują testy wbudowane. Trzonem tej metody jest kod źródłowy wykonywany przez mikrokontroler lub procesor, sprawdzający poszczególne układy peryferyjne a następnie dostarcza odpowiedzi, które elementy zawiodły. Jest to tania i szybka metoda ale za jej pomocą nie jesteśmy wstanie zweryfikować, które linie sygnałów są wadliwe.
Może również zdarzyć się tak, że program w ogóle nie wystartuje, wówczas metoda ta nie zwróci żadnej odpowiedzi, następnie taka płytka trafia do kosza bez głębszej analizy błędów.

3.2 Jaka alternatywa?

Obecnie stosowane techniki testowania układów elektronicznych nie potrafią sprostać nowym wymaganiom jakie są stawiane przez postęp technologiczny. Należało znaleźć nowy sposób testowania układów, bazujący na zupełnie innej technice niż dotychczasowe.
Rozwiązanie przyniosła technika JTAG Boundary Scan, której innowacyjność polega na testowaniu układów wprowadzając szeregowo sygnały do łańcucha boundary scan poprzez interfejs JTAG a następnie wyprowadzając szeregowo otrzymane odpowiedzi.
JTAG Boundary Scan został opracowany w latach 80 ubiegłego wieku przez grupę inżynierów z Europy i Ameryki Północnej zwanej Joint Test Action Group (JTAG). Następnie w 1990 roku został zestandaryzowany przez instytut IEEE i oznaczony sygnaturą 1149.1
Wprowadzenie tego standardu miało na celu wprowadzenie nowego sposobu sprawdzania połączenia układu z podłożem płytki, programowanie oraz debugowania układów.



Rys. 8. Schemat struktury łańcucha JTAG boundary scan

Pierwotnie technika JTAG Boundary Scan nie znalazła szerszego zainteresowania wśród inżynierów i czekała ponad dziesięć lat na ponowne jej odkrycie i uznanie. Ostatecznie standard ten został zaktualizowany, już jako IEEE1149.1a.
Technika JTAG Boundary Scan nie opiera się na fizycznym kontakcie z każdym wyprowadzeniem testowanego elementu elektronicznego, lecz wykorzystuje w tym celu złącze JTAG zaimplementowanego w co najmniej jednym układzie na płytce.
Łańcuch Boundary Scan zbudowany jest z komórek logicznych, połączonych ze sobą szeregowo z równoległym wejściem i wyjściem (rys. 3). Łańcuch sterowany jest za pomocą kontrolera TAP, natomiast kontroler TAP sterowany jest czterema liniami TDI, TDO, TCK, TMS (standard JTAG).

Ogólna idea techniki polega na wprowadzeniu sygnałów sterujących przez złącze JTAG, które będą generowały lub analizowały zmiany wartości pinów. Analiza i generacja stanów pinów umożliwia sprawdzenie poprawności połączeń lub komunikację z prawie dowolnym elementem na płytce.
Przykładem może być komunikacja z pamięcią półprzewodnikową typu Flash, Static RAM, itp. do której, dzięki odpowiedniemu wysterowaniu pinów, możemy zapisać lub odczytać dowolne dane.
Narzędzia bazujące na technologi JTAG Boundary Scan, które obecnie się wyłaniają, znacznie ułatwiają pracę projektantom, testerom oraz serwisantom. Wymagania jakie zostały postawione przed tą techniką to kompatybilność z innymi technikami testującymi, brak wpływu typu montażu na proces testowania, krótki czas przygotowania procedury testującej oraz jak największy zakres testowania.
Technika JTAG może współpracować z technikami typu ICT lub ze środowiskami typu „LabWiev". Obecnie prowadzone są intensywne badania nad techniką JTAG Boundary Scan i dotychczasowe osiągnięcia tej techniki pozwalają na szersze jaj możliwości i samodzielną pracę bez wspomagania pozostałych technik.

3.3 Oferta na rynku polskim

Jednym z dostępnych na rynku polskim komercyjnych systemów bazujących na technice JTAG Boundary Scan jest system XJTAG, należący do brytyjskiej firmy XJTAG.
System ten został stworzony i rozwijany przy współpracy z naukowcami z Cambridge Technology Group. Rdzeniem działania systemu XJTAG jest współpraca trzech elementów:

- pliki BSDL (Boundary Scan Description Language),
- schemat połączenia płytki w formie „netlisty"
- skrypty testowe.

Pliki BSDL zawierają opis struktury łańcucha boundary scan, natomiast skrypty testowe napisane w języku wysokiego poziomu XJEase Language, zawierają kod opisujący procedury testowe. Wykorzystanie testów funkcjonalnych, znacznie zwiększa obszar testowalności płytki, obejmując również układy bez implementacji JTAG, takich jak:

- kontrolery Ethernet,
- kontrolery wideo,
- diody LED,
- bufory,
- układy z wbudowaną szyną IIC
  i wiele innych.

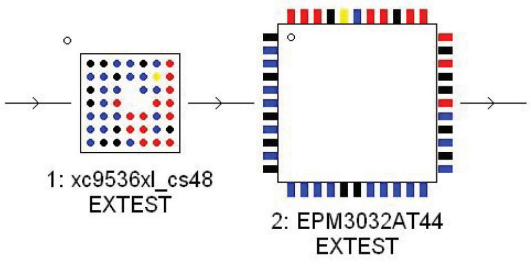
Dla systemu XJTAG nie stoją na przeszkodzie pamięci:

- moduły SDRAM,
- pamięci Flash,
- Static RAM,
- FRAM,
- Compact Flash.

Wszystkie te elementy są testowane przez system XJTAG bez konieczności fizycznego podłączania się do każdego z nich oraz bez konieczności programowania głównego mikrokntrolera w celu przeprowadzenia wbudowanych testów.

Jeżeli w trakcie projektowania, płytka zostanie zmodyfikowana, wówczas należy jedynie importować do projektu nową „netlistę" oraz nanieść ewentualne małe poprawki do procedury, bez konieczności tworzenia całej procedury testowej od nowa. Jest to znacznym ułatwieniem dla projektantów i producentów, którzy tworzą różne wersje układów elektronicznych, bazujących na tych samych układach elektronicznych.
Dodatkowo, jeżeli projekt płytki będzie zoptymalizowany z uwzględnieniem testowania za pomocą techniki JTAG Boundary Scan (ang. Design For Test, DFT), wówczas czas potrzebny na przygotowanie testów jeszcze bardziej ulegnie skróceniu, obniżą się koszty, przyspieszy proces wprowadzenia nowego produktu na rynek, zapewni najwyższą jakość produktu.

- XJAnalyser jest wizualnym narzędziem służącym do analizowania i debugowania urządzeń w łańcuchu JTAG. Umożliwia on natychmiastową weryfikację łańcucha po czym dostarcza interaktywny, graficzny wgląd na piny urządzeń JTAG (rys. 4).
- XJEase jest kompletnym systemem testowym dostarczającym pełnej kontroli oraz elastyczności opartym o technologię JTAG. Umożliwia wykrycie zwarć i przerwań pomiędzy liniami sygnałowymi używając wbudowanej funkcji do testowania połączeń oraz informacji pobranych bezpośrednio z pliku netlisty.
- XJRunner jest wyspecjalizowanym środowiskiem uruchomieniowym dla testów opracowanych za pomocą pakietu XJEase. Szereg specjalnych właściwości sprawia, iż jest on skierowany głównie na potrzeby producentów płytek oraz testowania w warunkach mobilnych.
- XJDeveloper jest graficzną aplikacją umożliwiającą generowanie dla XJEase opisu obwodu,który ma być przetestowany. Prosty interfejs typu przeciągnij i upuść pozwala na łatwy i szybki sposób skonfigurowania łańcucha JTAG oraz sklasyfikować wszystkie urządzenia pozbawione interfejsy JTAG.



Rys. 9. Analiza graficzna układów JTAG za pomocą XJAnalyser.

Podsumowanie

Ogromne możliwości JTAG Boundary Scan na tle dzisiejszego postępu w dziedzinie technologii elektronicznych i montażu elektronicznego, powoduje że technika ta stanie się przez długi czas jedyną alternatywą w technice pomiarowej systemów elektronicznych. Obniżenie kosztów, skrócenie czasu oczekiwania oraz poprawa jakości produktu to zalety, które skłaniają projektantów i producentów układów do zmiany dotychczas stosowanych technik testowania i rozpoczną budowę profesjonalnych stanowisk

Cześć Praktyczna

1. Wstęp

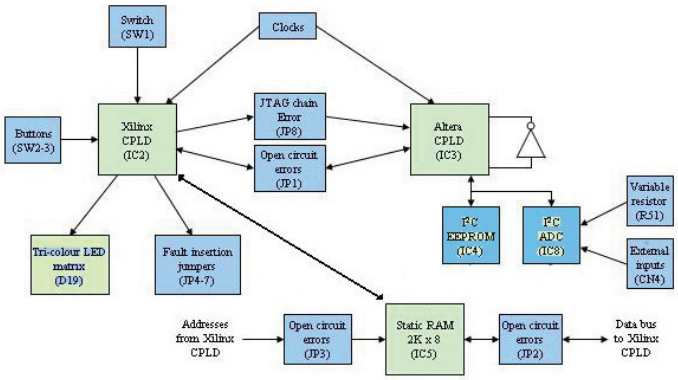
Przedmiotem testu za pomocą systemu XJTAG Boundary Scan będzie zintegrowana płytka drukowana XJTAG Demo Board V2.0.

Przed rozpoczęciem procedur testowych dobrą metodą na sprawdzenie poprawnego połączenia sieci JTAG Bounday Scan jest wykorzystanie programu XJAnalyser. Jest to jedno z narzędzi systemu XJTAG.
Następnie rozpoczniemy pracę nad tworzeniem projektu za pomocą XJDeveloper, który poprowadzi nas krok po kroku przez pełną konfigurację procedury testowej.
Po zakończonej konfiguracji przeprowadzone zostaną testy obejmujące następujące operacje:test połączeń, symulacje błędów w połączeniu, test pamięci SRAM1 oraz test diod LED.

Opis płytki

Głównym rdzeniem płytki są dwa programowalne układy CPLD: Xilinx XC9536XL (IC2) oraz Altera EPM3032A (IC3).
Układy te maja wbudowany interfejs JTAG wraz zaimplementowanym łańcuchem Boundary Scan. Układy te tworzą sieć w ten sposób, że wyjście TDO układu Xilinx podłączone jest do wejścia TDI układu Altera. Pomiędzy tymi sygnałami dodatkowo znajduję się zworka JP8, której usunięcie symuluję zerwanie sieci JTAG Boundary Scan. Natomiast pomiędzy zewnętrznym złączem JTAG a układami CPLD znajduję się zworka JP1, która posłuży nam do symulowania niemożliwości wykrycia łańcucha JTAG Boundary Scan.

Jednym z elementów, który będzie testowany to BS62LV256 Static RAM. Linie adresowe i danych podłączone są do układu Xilinx. Linie adresowe A7-A10 poprowadzone są przez rezystory oraz zworkę JP3, natomiast linie danych D0-D3 przez zworkę JP2. Usunięcie zworki JP3 oraz JP2 będzie symulowało przerwanie połączenia. Pozostałe elementy to pamięć EEPROM Microchip 24LC32A podłączona do układu Altera przez szynę I^2C oraz przetwornik analogowocyfrowy ADS7830, również podłączony do układu Altera przez szynę I^2C.
Schemat blokowy płytki znajduję się na rysunku poniżej.



Podłączenie płytki

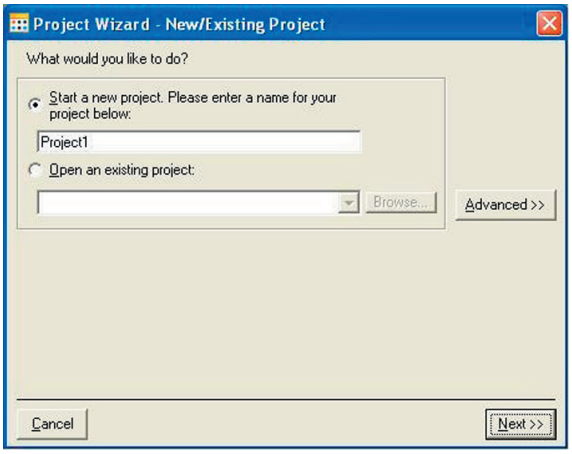
Przed podłączeniem płytki należy zainstalować oprogramowanie znajdujące się na płycie CD. Po zainstalowaniu podłączamy XJLink do złącza JTAG znajdującego się na płytce XJDemo przez płaski 20 żyłowy przewód. Następnie należy podłączyć XJLink do komputera PC przez kabel USB.

2. Uruchomienie płytki za pomocą XJAnalyser

2.1 Konfiguracja projektu XJAnalyser

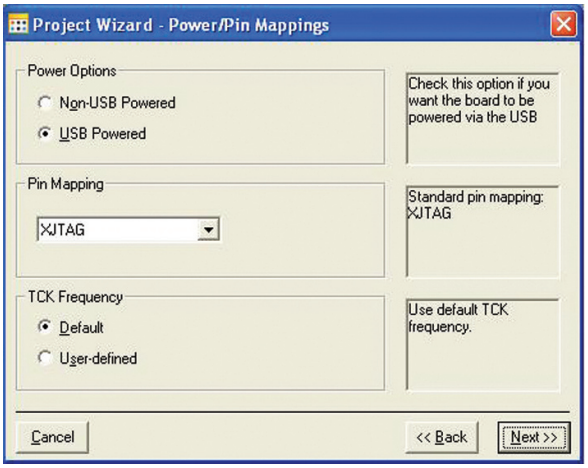
Nowy projekt

Po uruchomieniu XJAnalyser, tworzymy nowy projekt wpisując jego nazwę lub otwieramy już istniejący. W naszym przypadku tworzymy nowy projekt o nazwie Project1, następnie zatwierdzamy klikając Next



Konfiguracja zasilania, złącza JTAG oraz częstotliwości taktowania sygnału TCK

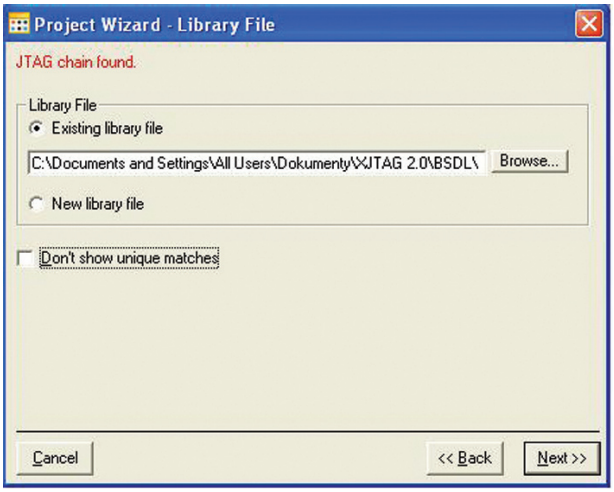
W następnej kolejności należy wybrać opcję zasilania płytki, odpowiednią konfigurację pinów JTAG oraz określić częstotliwość taktowania sygnału TCK.



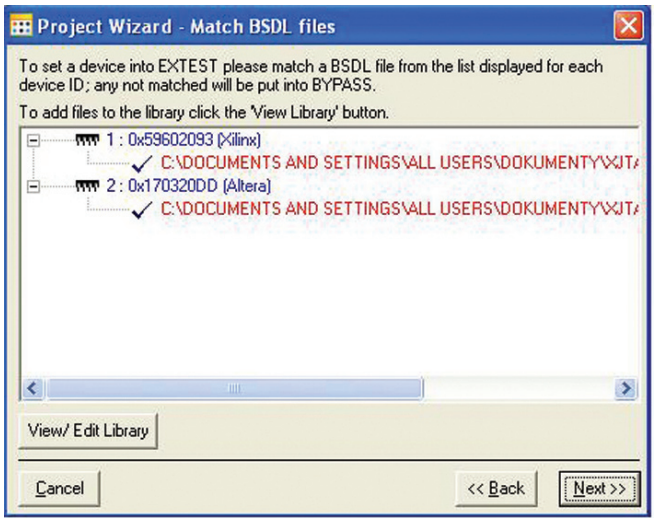
System XJTAG umożliwia zasilanie płytki przez złącze USB jak i wyłączenie tego zasilania w przypadku gdy płytka zasilana jest z zewnętrznego źródła. Płytka XJDemo zasilana jest z wykorzystaniem kabla USB. W konfiguracji tej wybieramy opcje **USB powered**. Następny krok to wybór konfiguracji pinów interfejsu JTAG. Płyta XJDemo ma zaimplementowaną standardową konfigurację, tak więc pozostawiamy opcję **XJTAG**. Ostatnim krokiem w tej części jest określenie częstotliwości pracy sygnału TCK. Częstotliwość ta zależy od możliwości układu do którego podłączony jest XJTAG oraz od możliwości samego systemu XJTAG. Częstotliwość ta nie powinna przekraczać częstotliwości maksymalnej jaką podają producenci. W naszym przypadku wybór opcji **Default** zdefiniowane jest jako częstotliwość 25MHz. Po zakończonej konfiguracji zatwierdzamy klikając na **Next.**

Importowanie plików BSDL

Trzecim i ostatnim krokiem przed rozpoczęciem pracy z XJAnalyser jest importowanie odpowiednich plików BSDL.
W momencie zatwierdzenia poprzedniej konfiguracji system XJAnalyser skanuję płytkę w celu identyfikacji numerów ID układów z implementowanym łańcuchem JTAG Boundary Scan. W ten sposób wykrywa układy podłączone do sieci JTAG Boundary Scan.
Następnie należy zaimportować pliki BSDL, które opisują strukturę łańcucha Boundary Scantych układów.Pierwszą możliwą opcją jest wybór już istniejącej biblioteki plików BSDL.
W przypadku płytki XJDemo biblioteka ta znajduję się w domyślnym katalogu XJTAG2.0/BSDL i nosi nazwę „Library.xjb". Pozostałe opcję dotyczą przypadku utworzenia nowej biblioteki.
Jeżeli opcję **Don't show unique matches** pozostawimy pustą, wówczas przejdziemy do opcji umieszczania nowych plików BSDL do biblioteki.



Klikamy **Next** i przechodzimy do następnej opcji z możliwością edytowania biblioteki plików BSDL. W oknie dialogowym wyświetlają się numery ID oraz odpowiednio przyporządkowane pliki BSDL.
W przypadku zmiany plików BSDL należy wybrać opcję **View/Edit Library**. Jeżeli chcemy stworzyć nową biblioteke narzędzi wybieramy opcję **New Library File**.

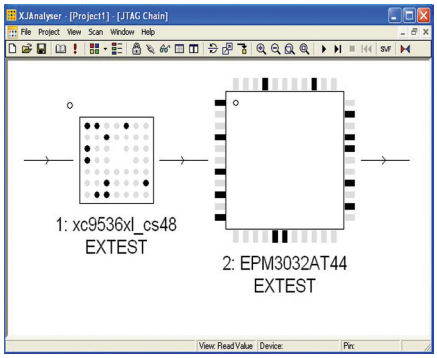


To wszystkie wymagane kroki konfiguracyjne.
Teraz aby rozpocząć prace z XJAnalyser wystarczy kliknąćna **Next**.

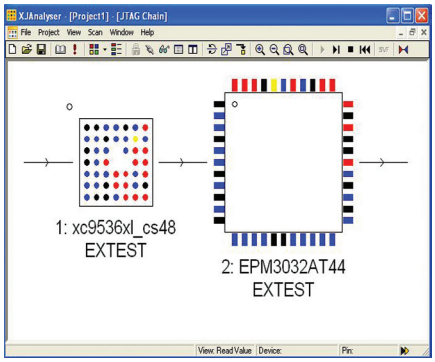
2.2 XJAnalyser

Okno dialogowe

Jeżeli konfiguracja została poprawnie przeprowadzona i pliki BSDL zgadzają się z układami to powinno wyświetlić się główny panel programu XJAnalyser. Wyświetlone zostały układy tworzące sieć JTAG Boundary Scan: Xilinx oraz Altera. Graficzne przedstawienie tych układów odzwierciedla dokładnie typ obudowy. W przypadku Xilinx jest to typ BGA, natomiast w przypadku Altera jest to obudowa standardowa. Potwierdza się tu jedna z najważniejszych zalet systemu XJTAG, mówiąca o braku zależności typu obudowy układów na ich sposób testowania.



W następnej kolejności przechodzimy do trybu analizy poziomów pinów układów. W tym celu wybieramy z menu **Scan**, a następnie **Start JTAG Scan** lub klikamy na skrót ▶



Aplikacja XJAnalyser oprócz analizowania poziomów pinów umożliwia również ich zmianę. Można to zrobić na kilka sposobów jeden z nich to podwójne kliknięcie lewym przyciskiem myszy na wybrany pin, wówczas stan zmienia się na przeciwny lub kliknąć prawym przyciskiem myszy na wybrany pin i po rozwinięciu menu wybrać jedną z możliwych opcji.
Pomocnym narzędziem jest wyszukanie pinów.

W tym celu z menu wybieramy **View** a następnie **Goto**.
W oknie, które się pojawiło wybieramy układ a następnie szukany pin. Klikamy na **Goto**, szukany pin zacznie migać.
Obecnie układy pracują w trybie EXTEST, który umożliwia zarówno analizę pinów procesora jak i ich aktywację. Układy w tym trybie wyłączone są z normalnej pracy i nie wykonują zaprogramowanego kodu. Tryb pracy układu możemy zmienić klikając na wybrany układ prawym przyciskiem myszy, wybierając mode a następnie tryb pracy. Przykładowo możemy przejść w tryb sample. Wówczas układ będzie pracował normalnie, wówczas jednocześnie możemy śledzić wartości jego pinów. Jeżeli układ CPLD został zaprogramowany to po naciśnięciu przycisku SW2lub SW3 rozpocznie się miganie diod LED jednocześnie efekt ten będzie widoczny na ekranie XJAnalysera. Programowanie układów CPLD z wykorzystaniem plików SVF lub STAMPLzostanie omówione w rozdziale 6. Przykład dotyczący wyboru pracy zostanie również powtórzony.

Przykład:
Ponownie wybierzmy opcję wyszukiwania pinów **Goto**, wybierzmy układ Xilinx, pin E2:PB01_15, klikamy na **Goto**.
Gdy pin zostanie odnaleziony kliknijmy na niego prawym przyciskiem myszy i wybierzmy opcje **Set toggle slow**. Jedna z diod na płytce powinna migać na czerwono. Aby powrócić do poprzedniego stanu klikamy prawym przyciskiem myszy i wybieramy opcję **Disable**.

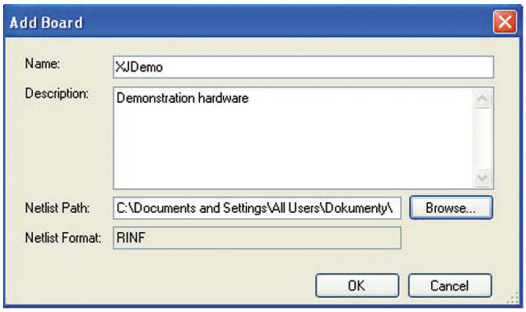
XJAnalyser posiada jeszcze wiele dodatkowych, bardziej zaawansowanych opcji, które znacznie ułatwiają pracę projektantom oraz testerom. W prezentacji tej nie będą one szerzej omawiane.

Uruchomienie XJAnalyser miało na celu sprawdzenia poprawności działania sieci JTAG Boundary Scan. Wstępne testy zakończone zostały pomyślnie.
Kolejny etap to konfiguracja projektu, który będzie stanowił trzon procedury testowej. W tym celu wykorzystamy aplikację XJDeveloper.

3. Konfiguracja procedury testowej za pomocą XJDeveloper

3.1 Nowy projekt

Program XJDeveloper w prosty i przejrzysty sposób poprowadzi nas przez pełną konfigurację skryptu testującego (procedury testowej).
W pierwszej kolejności uruchamiamy aplikację XJDeveloper.
Zaraz po uruchomieniu pojawi się główny panel oraz okno dialogowe, w którym należy wpisać nazwę testowanej płytki, ewentualny opis oraz zlokalizować netlistę. Nazwa może być dowolna, ale dla naszego projektu nazwiemy ją **tutorial**.
Netlista jest to plik tekstowy wygenerowany przez program typu CAD do projektowania płytek. Zawiera ona informację o schemacie połączeń.
System XJTAG przyjmuje kilka formatów netlisty, jedną z nich jest format RINF.
Netlistę wyszukujemy i importujemy klikając na **Browse**. Po tych czynnościach klikamy **OK** oraz w głównym oknie dialogowym na **save** aby zapisać cały projekt.



3.2 Identyfikacja linii zasilających

Identyfikacja linii zasilającej i uziemiającej przekazuje ważną informacją dla systemu i określi jego sposób postępowania względem tych linii:
- nie będzie zmieniał ich stanów podczas przeprowadzania testów
- automatycznie wykryje poprawność połączenia do zasilania sprawdzając ich stan
- może zidentyfikować rezystory podciągające

Na płytce XJDemo linie zasilające mają nazwy 3.3V dla zasilania i GND dla uziemienia. Nazwy te są zawarte w netliście.

Wykonujemy następujące kroki:
- klikamy na ikonę **Power/Ground Nets** w głównym panelu XJDeveloper
- klikamy na **Suggested Ground Nets** i przeciągamy linię GND w obszar **Ground Nets**
- takie same czynności powtarzamy z linią 3.3V przeciągając ją do **Power Nets**
- zapisujemy klikając na **save** w głównym panelu

3.3 Identyfikacja łańcucha JTAG

Tylko niektóre układy elektroniczne w zintegrowanych płytkach drukowanych są kompatybilne z interfejsem JTAG, dlatego też następny krok będzie dotyczył identyfikacji układów z wbudowanych interfejsem JTAG.
W tej części, również importujemy pliki BSDL. Pliki te można pobrać ze stron producenta.
Dla naszych układów pliki te są już dostępne w katalogu BSDL/Demo.

Aby być pewnym poprawnie zidentyfikowanego łańcucha z netlistą należy określić sygnały TDI do którego XJLink wprowadza dane do sieci JTAG Boundary Scan oraz sygnał TDO, którym wyprowadza dane. Na płytce drukowanej XJDemo złącze JTAG wyprowadzone jest przez konektor CN1. Posługując się schematem płytki możemy zidentyfikować, które piny tego konektora są odpowiedzialne za sygnały TDI i TDO.

Identyfikacja pinów TDI i TDO
- klikamy na ikonę **JTAG Chain** w głównym panelu XJDeveloper
- klikamy na **Set TDI** następnie pojawi się okno **Select Device and Pin**
- rozwijamy opcję **Suggested Connectors** i wybieramy **CN1**
- poniżej wyświetli się lista dostępnych pinów, wybieramy pin **5** jako sygnał TDI (patrz schemat płytki)
- klikamy **OK**
- czynności te powtarzamy dla sygnału TDO wybierając **Set TDO i dla CN1** wybieramy pin **13**
- zapisujemy klikając na **save**

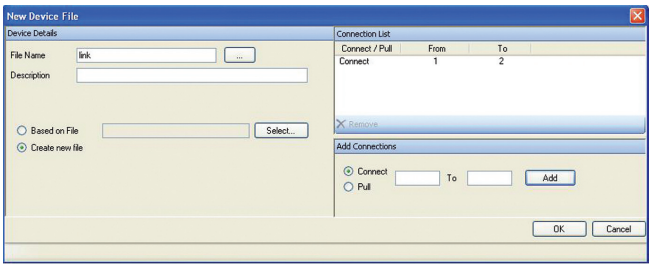
Po wykonaniu tych czynności system wyświetlił nam w oknie **Select Next Pin** nazwę pinu IC2.B3 który został napotkany na linii TDI – TDO. Należy określić do jakiego elementu jest podłączony i jaką spełnia ten element funkcję.
- przeciągamy pin IC2.B3 na obszar okna **JTAG Devices** lub klikamy na niego podwójnie lewym przyciskiem myszy
- w oknie **Edit JTAG Chain** określamy jaki to typ układu wybierając **Assign IC2 as JTAG device**
- lokalizujemy plik BSDL klikając na **Browse**
- wybieramy plik **xc9536xl_cs48.bsd** z katalogu BSDL/Demo
- klikamy **OK**

W oknie dialogowym **Select Next Pin** pojawił się kolejny, napotkany na drodze TDI – TDO, element **JP8.1**. Jak popatrzymy na schemat układu to widzimy, że **JP8** jest zworką pomiędzy pinem TDO układu Xilinx a pinem TDI układu Altera. System nie wie dokładnie co to za typ elementu, należy więc go określić.

W tym przypadku stworzymy oddzielny plik, który będzie określał jego funkcję w systemie.

Należy kolejno:
- kliknąć podwójnie na element **JP8.1**
- określić typ jako **Assign JP8 as Connect device**
- kliknąć na **Create File** w oknie **Edit JTAG Chain**
- w nowym oknie dialogowym wpisać nazwę **link** (nazwa może być dowolna)
- klikamy **OK**
- w obszarze **Add Connections** wybrać opcję **Connect** wpisać **1 To 2** i kliknąć na **Add**
- klikamy **OK**

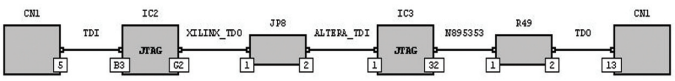
W tym momencie stworzyliśmy plik **link.pdd**, który informuje system, że jest to element, którego wyprowadzenia są bezpośrednio ze sobą połączone



Kolejnym elementem, który się pojawił w obszarze **Select Next Pin** jest IC3.1. Jest to wejście sygnału TDI układu Altera.
Jest to element sieci JTAG, dlatego też postępujemy w tym przypadku tak samo jak z układem Xilinx IC2 z tą różnicą, że plik BSDL nosi nazwę **3032at44.bsd**.

Kolejnym element na liście **Select Next Pin** to R49.1.
Jest to rezystor, który też może być potraktowany jako element łączący, jak zworka JP8. Postępujemy tak samo jak z elementem JP8, podając nazwę tworzonego pliku jako „resistor".

W ten oto sposób doszliśmy do końca konfiguracji sieci JTAG Boundary Scan, począwszy od wejścia sygnału TDI do wyjścia TDO. W głównym oknie dialogowym widzimy graficzną prezentację sieci JTAG. Struktura ta jest zgodna ze schematem płytki.



Jeżeli już zdefiniowaliśmy sieć JTAG Boundary Scan możemy zająć się pozostałymi elementami znajdującymi się na płytce. Zanim jednak do tego przystąpimy, należy opisać w jaki sposób XJLink łączy się z płytką.

Należy przypisać odpowiednie funkcję pinom konektora JTAG.
W tym celu:
- wybieramy ikonę **Pin Mapping** w oknie głównym
- klikamy na pin 1 (Power) i w oknie, które się pojawiło, ustaw pin jako aktywny wybierając **Power On** (ustawienie to zależy od rodzaju płytki testowanej)
- klikamy na **Apply** (konfiguracja ta zależy od rozmieszczenia pinów konektorze JTAG. Dla innych płytek może być inna konfiguracja).

3.4 Klasyfikacja układów bez implementacji JTAG (non JTAG devices)

Następnym etapem będzie sklasyfikowanie pozostałych elementów zamontowanych na płytce.
System XJTAG wymaga tych informacji w celu przeprowadzenia bezpiecznego testu. Jeżeli system nie będzie znał typu układu nie będzie mógł aktywować pinów do nieznanych elementów.
W ten sposób system zabezpiecza układy przed ewentualnym zniszczeniem. Klasyfikowanie elementów odbywa się w oknie dialogowym **Categorise Devices.**

3.4.1 Dodawanie połączeń jako elementy pasywne

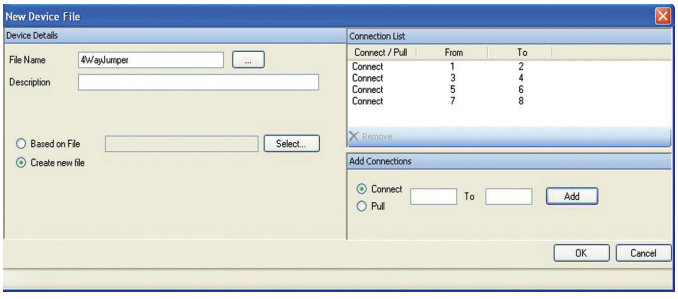
W pierwszej kolejności zajmiemy się prostymi elementami, które pełnią jedynie funkcję połączenia. Do tych elementów należą zworki lub rezystory.

W oknie dialogowym **Categorise Devices** mamy listę wszystkich elementów, którym system nadał pewne sugestię ich przeznaczenia. Rozpoczniemy dokładną klasyfikację od rezystorów seryjnych (Suggested Series Resistors) a następnie rezystory podciągające (Suggested Pull Resistors).

- wybieramy z lisy **Suggested Series Resistors** i przenosimy je na obszar **Passive Devices**
- w oknie dialogowym, które się pojawiło wybieramy już istniejący plik **resistor.pdd.** Plik ten został stworzony na podstawie rezystora R49, więc możemy utożsamić pozostałe rezystory z tym plikiem.
- klikamy **OK**

W następnej kolejności określimy takie elementy jak:
JP1, JP2 oraz JP3. Są to cztero-pinowe zworki.

- rozwijamy listę wszystkich elementów klikając na symbol **+ All Components**
- wybieramy **JP1, JP2, JP3** (przytrzymując klawisz Ctrl) i przeciągamy na obszar **Passive Devices**
- wybieramy **Create File** w oknie dialogowym **Add Connection Devices**
- w oknie dialogowym **New Device File** wpisujemy nazwę **4WayJumper**
- wykonujemy połączenia pomiędzy wyprowadzeniami tej zworki łącząc 1 To 2; 3 To 4; 5 To 6; 7 To 8, klikając za każdym razem **Add**
- klikamy **OK**



Do prostych elementów łączących należą też rezystory podciągające. Postępujemy z nimi podobnie jak z rezystorami seryjnymi.

- wybieramy z lisy **Suggested Series Resistors** i przenosimy je na obszar **Passive Devices**
- wybieramy **Create File** w oknie dialogowym **Add Connection Devices**
- w oknie dialogowym **New Device File** wpisujemy nazwę **pull-resistor**
- w obszarze **Add Connections** wybieramy opcję **Pull** zamiast **Connect** i wpisujemy **1 To 2** i klikamy na **Add**
- klikamy **OK**

3.4.2 Dodawanie elementów do listy elementów ignorowanych

Istnieje możliwość zignorowania niektórych elementów znajdujących się na płytce drukowanej. Dotyczy to elementów, których zmiana stanu linii nie daje odpowiedzi o poprawnym połączeniu lub elementów, których testowanie nie jest wymagane.
Elementy, które nie będą testowane można dodać do listy elementów ignorowanych. W naszym układzie takimi elementami są to konektory CN1, CN3 oraz zworki JP5 i JP6. Konektor CN1 jest to złącze JTAG, które nie wymaga testowania samego siebie. Element CN3 jest pustym, nieużywanym złączem. JP5 i JP6 są to nieużywane zworki.

Dodawanie do listy elementów ignorowanych:
- rozwijamy listę wszystkich elementów klikając na symbol **+ All Components**
- wybieramy **CN1, CN3, JP5, JP6** (przytrzymując klawisz Ctrl) i przenosimy na obszar **Ignore Devices**

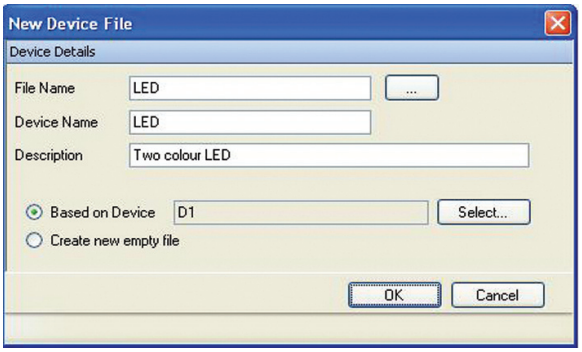
3.4.3 Tworzenie plików testowych

Podczas przeprowadzania testu połączeń wymagane jest aby wszystkie elementy napotkane na drodze sieci JTAG Boundary Scan, przez które będą przebiegać sygnały testujące, należy odpowiednio skonfigurować aby system miał informacje o danych elementach i wiedział jakich odpowiedzi może się spodziewać.

Tworzenie plików testowych poszczególnych elementów:

.

Przykład tworzenia pliku diody LED:
- rozwijamy listę diod **Suggested Diodes**
- wybieramy diody od D1-D9 i przeciągamy na obszar **Test Devices**
- w dialogu **Add Test Devices** klikamy na **Create File**
- w nowym dialogu wpisujemy nazwę pliku jako **LED,** nazwę układu jako **LED,** jako opis: **Two colour LED**
- wybieramy opcję **Based on Device,** natomiast jako układ bazowy diodę **D1** klikając na **Select**
- zatwierdzamy klikając **OK**



Tak samo postępujemy z elementami:
IC1 - inverter.
IC4 - 24LC32A.
IC6 - ADS7830.
SW1 - switch.
SW2, SW3 - pushbutton.

Plik testowy dla pamięci SRAM nie będzie tworzony jako plik na bazie elementu IC5 tak jak to ma miejsce z poprzednimi elementami ale jako plik pusty. Nie ma to większego znaczenia dla testowania. Wprowadzone jest to w celach dydaktycznych.

Tworzymy plik testujący dla pamięci SRAM:
- rozwijamy listę elementów **Suggested Devices**
- wybieramy układ IC5 i przenosimy na obszar **Test Devices**
- w dialogu **Add Test Devices** klikamy na **Create File**
- w nowym dialogu wpisujemy nazwę pliku jako **BS62LV256,** nazwę układu **(Device Name)**: **BS62LV256,** jako opis: **SRAM**
- wybieramy opcję **Create new empty file**
- zatwierdzamy klikając **OK**

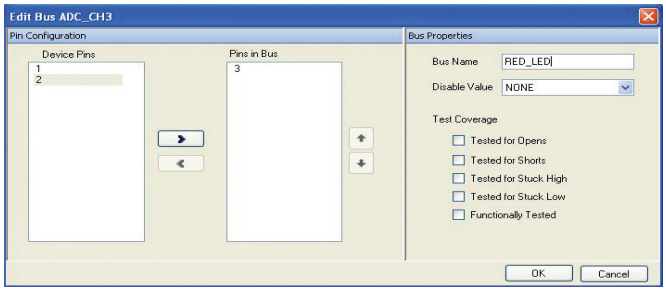
W tym momencie system sygnalizuje, że podstawowa konfiguracja jest zakończona.

Następnym krokiem będzie dokładne przyjrzenie się testowanym elementom i sprawdzenie które spośród pinów danego układu należy zabezpieczyć przed jego zmianą stanu.

3.4.4 Dodawanie połączeń jako elementy pasywne

W tym przypadku stworzymy oddzielny plik, który będzie określał jego funkcję w systemie.

Należy kolejno:
- klikamy na ikonę **Device Files** w głównym panelu XJDEveloper
- wybieramy **LED.xje** z okna dialogowego **Device Files**
- kliknąć na **Create File** w oknie **Edit JTAG Chain**
- wybieramy linie **ADC_CH3** z obszaru **Busses**
- klikamy na **Edit Bus**
- wpisujemy nazwę **RED_LED** (**Bus Name** w obszarze **Bus Properties**)
- klikamy **OK**
- powtarzamy tą operację dla linii **N10506307** wpisując nazwę **GREEN_LED**
- klikamy **Save Device File**



Inwerter

- wybieramy **inverter.xje** w oknie dialogowym **Device Files**
- wśród wyświetlonych pinów wybieramy pin 20 o nazwie **NALT1** następnie klikamy na **Edit Bus**
- ustawiamy w opcji **Disable Value** na stan **INPUT** (W ten sposób określamy, że sygnał ten jest jednokierunkowy. W przeciwnym wypadku podczas testu połączeń pojawiłby się błąd o niemożliwości poprowadzenia sygnału testującego w dwóch kierunkach).
- klikamy **OK**
- czynność ta powtarzamy dla sygnałów **CLOCKS** oraz **CLOCKF** ustawiąc jako sygnały wejściowe **INPUT**
- zapisujemy plik klikając na **Save Device File**

Pamięć EEPROM

- wybieramy **24LC32A.xje** w oknie dialogowym **Device Files** wśród wyświetlonych linii wybieramy **SDA**, następnie klikamy na **Edit Bus**
- ustawiamy w opcji **Disable Value** na stan **INPUT**
- klikamy **OK**
- zapisujemy plik klikając na **Save Device File**

Pamięć Static RAM :

Plik testowy, który był utworzony dla pamięci SRAM nie powstał na bazie żadnego elementu, tak więc edytując plik nie będzie tam żadnej zdefiniowanych pinów. Wymagane będą linie adresowe, linie danych oraz sygnały sterujące:

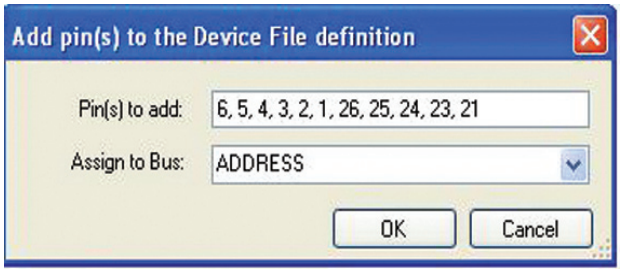
ADDRESS - 6, 5, 4, 3, 2, 1, 26, 25, 24, 23, 21
DATA - 19, 18, 17, 16, 15, 13, 12, 11
nWE - 27
nOE - 22
nCS - 20

Konfiguracja pamięci SRAM

- wybieramy BS62LV256.xje z okna dialogowego **Device Files**
- klikamy na **New** w obszarze **Pins**
- wpisujemy nazwę linii ADDRESS oraz numery pinów od 6-21

Uwaga! *Ważna jest kolejność wpisywania pinów.*
Należy przestrzegać adnotacji big endian lub little endian.
Nie można wpisywać numery pinów dowolnie.
Kwestia ta zostanie jeszcze poruszona przy testowaniu zapisu i odczytu do pamięci SRAM.

- klikamy **OK**
- powtarzamy tą czynność dla pozostałych linii wpisując nazwę **DATA, nWE, nOE, nCS** i kolejno numery pinów w tej samej adnotacji co linie adresowe
- zapisujemy plik **Save Device File**

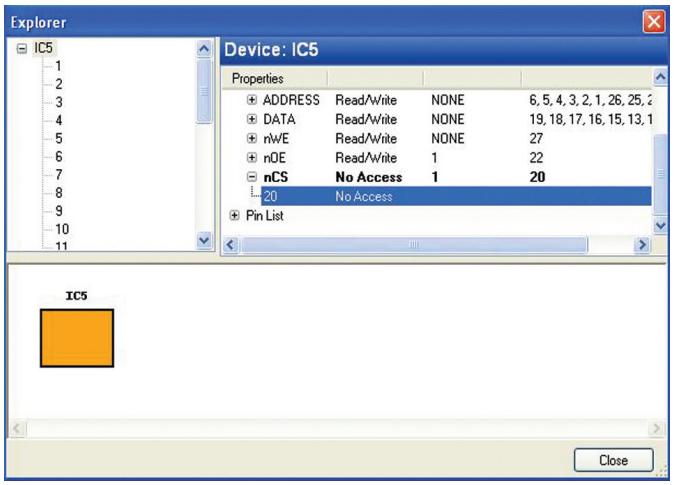


W celu zabezpieczenia pamięci przed przypadkowym pobudzeniem podczas testów należy określić, w jaki sposób układ ten może być chroniony.
W tym celu możemy posłużyć się pinami nCS oraz nOE.
Jeżeli stan tych pinów jest wysoki, wówczas pamięć SRAM jest nieaktywna.

- wybieramy linie **nCS** z obszaru **Busses** i klikamy **Edit Bus**
- w oknie dialogowym ustawiamy **Disable Value** na **HIGH**
- klikamy **OK** i zapisujemy **Save Device File**

Po dokonaniu tych zmian w dolnej części głównego okna dialogowego pojawiło się ostrzeżenie o niemożliwości ustawienia pinu nCS na stan wysoki, ponieważ linia ta podpięta jest do masy. Aby dokładnie to zweryfikować, możemy edytować wszystkie wyprowadzenia układu IC5 i sprawdzić do jakich linii należą

- klikamy na ikonę **Categorise Devices** na głównym panelu XJDeveloper
- klikamy prawym przyciskiem myszy na **IC5** w panelu **Test Devices**
- z menu, które się rozwinęło wybieramy **Explore**
- wyświetliło się menu z informacją o wszystkich pinach (liniach) podłączonych do układu IC5
- klikamy na symbol **+Busses** i rozwija się szyny sygnałów
- rozwijamy szynę nCS i wyświetla nam się informacje, że pin 20 jest podłączony do masy **GND**
- wychodzimy z okna **Explore** klikając na **Close**



Jeżeli znalezione źródło błędu nie stanowi zagrożenia na niepowodzenie testów, wówczas możemy ustawić aby ostrzeżenie to nie pojawiało się

- pozostając w tym samym panelu XJDeveloper w oknie **Test Devices** klikamy lewym przyciskiem na układ **IC5** a następnie na **Configure**
- pojawiło się okno **Configure Test Device** i odznaczamy opcję **Show Warnings** aby okienko zostało puste
- klikamy **OK** i zapisujemy projekt **save**

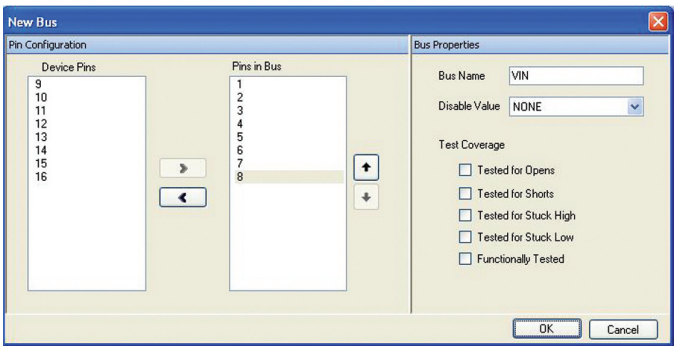
Przetwornik analogowo – cyfrowy

Układ ten jest przetwornikiem analogowo – cyfrowym, komunikującym się za pomocą szyny I2C.

Oznaczenia pinów:

SCL - 14
SDA - 15
VIN - 1, 2, 3, 4, 5, 6, 7, 8

- wybieramy **ADS7830.xje** w oknie dialogowym **Device Files**
- klikamy na **Create Bus**
- wpisujemy nazwę **VIN**
- wybieramy numery pinów od 1-8 z okna **Device Pins**
- klikamy na przycisk **>** aby przenieść wybrane piny
- tworzymy szynę klikając **OK**
- zapisujemy plik **Save Device File**



Teraz możemy usunąć niepotrzebne linie pozostawiając jedynie SDA, SCL oraz VIN

- wybierz przykładowo linie _3_3V, następnie klikamy na **Delete**
- powtarzamy tą czynność pozostając linie SDA, SCL, VIN
- zapisujemy plik **Save Device File**
- dodatkowo ustawiamy linie **SDA** w stan I**NPUT** w opcji **Disable Value**

Przełączniki i przyciski

Na płytce zamontowane są dwa rodzaje przycisków:

- suwak (SW1)
- przyciski guzikowe (SW2, SW3).

Każdy rodzaj przycisku ma inny pin, który reprezentuje wartość wyjściową.
Dla SW1 jest to pin 2, natomiast dla SW2 i SW3 jest to pin 3.
Jeżeli nie zostanie naciśnięty przycisk guzikowy, wówczas nie nastąpi zmiana stanu pinu wyjściowego. Nie ma potrzeby aby zmieniać ustawienia tego pinu. Inna sytuacja jest dla przycisku suwakowego.
Dla pinu 2 należy przypisać stan wyjściowy **INPUT** ponieważ wartość jego jest zawsze wysoka lub niska.
Dodatkowo dla łatwiejszego manipulowania sygnałami można zmienić nazwy linii **BUT0** (SW2, SW3) oraz **GP1** (SW1) na **VALUE**

4. Procedury testowe

W trakcie tworzenia projektu za pomocą XJDeveloper system ten automatycznie generował wersję tekstową tego projektu. Wersja tekstowa jest ważną częścią projektu ponieważ to ona będzie wywoływana i uruchamiana przez aplikację XJEase.

XJDeveloper doskonale posłużył nam do podstawowej konfiguracji projektu, natomiast do tworzenia procedur testowych najlepiej użyć programu TextPad. Jest to edytor tekstowy dostosowany do pisania skryptów testowych systemu XJTAG.

Jak otworzymy katalog, w którym zapisywaliśmy nasz projekt w XJDeveloper to odnajdziemy plik o takiej samejnazwie jak nasz projekt z rozszerzeniem **.xje.**
Jak go otworzymy to powinniśmy zobaczyć tekst:

```
CIRCUIT NAME := „New Project”
COMPAT_VERSION := 1;
BOARD NAME := „XJDemo”
NETLIST := „..\\Demo Board\\TestCode\\DEMO.NET”, RINF;

POWER LIST
  GROUND := „GND”;
  POWER := „3.3V”;
END;

JTAG LIST
  IC2 := „Demo\\xc9536xl_cs48.bsd”;
  IC3 := „Demo\\3032at44.bsd”;
END;

DEVICE LIST
  D1, D2, D3, D4, D5, D6, D7, D8, D9 := „LED.xje”;
  IC5 := „BS62LV256.xje”, NOWARN;
END;

IGNORE LIST
  CN3, CN1, JP6, JP5, IC1, IC4, IC6, SW1, SW2, SW3;
END;

CONNECTION LIST
  JP8 := „link.pdd”;
  R49, R10, R11, R14, R15, R16, R17, R18, R19, R20, R21, R22, R23, R24,  R25,R26, R27,R28,
  R5, R6, R7, R8, R9 := „resistor.pdd”;  JP1, JP2, JP3 := „4WayJumper.pdd”;
  R12, R13, R29, R3, R30, R31, R32, R33, R34, R35, R36, R37, R38, R39, R40, R41, R42, R43,
  R44, R45, R46, R47, R48 := „pull-resistor.pdd”;
END;

END; // End of board XJDemo

// JTAG interface description
JTAG CHAIN

CONNECTOR „JTAG Header”
  TDI := XJDemo.CN1.5;
  TDO := XJDemo.CN1.13;

MAPPING
  XJTAG;
  POWER := ON;
END;
END;
END;
// Additional files
FILES
„circuittest.xje”;
END;
END;
```

Powyższy kod odzwierciedla projekt jaki tworzyliśmy w programie XJDeveloper. Teraz możemy docenić jak pożytecznym narzędziem jest XJDeveloper. Dzięki niemu zaoszczędziliśmy pisanie trudnej konfiguracji procedury testującej. W dalszej części będziemy posługiwać się już programem TexPad i ręcznie pisać programy testujące.
Procedury pisane są w języku opracowanym przez producenta XJTAG i nazywa się XJEase Language. Język oraz struktura skryptów podobna jest do aplikacji pisanych w języku ANSI C.

4.1 Test połączeń

Podczas tworzenia projektu, test połączeń został automatycznie załączony do projektu w oddzielnym pliku circuittest.xje.
W głównym pliku została zawarta ta informacja dodając nagłówek:

```
FILES
  „circuittest.xje";
END
```

Jak otworzymy plik circuittest.xje. to zobaczymy definicję pustej funkcji bez dodatkowych komend. Dlatego też należy uzupełnić tą funkcję kodem aby funkcja ta wyglądała następująco:

```
CircuitTest()()
  INT result;
  INT debugLevel := 3;
  CONNTEST(debugLevel)(result);
     IF result != 0 THEN
     PRINT(result, „ error(s) found in connection test.\n");
     EXIT;
  ELSE
  PRINT(„Connection test completed successfully.\n");
  END;
END;
```

Należy pamiętać aby plik po każdej zmianie zapisywać.
Aby uruchomić test należy użyć środowiska XJTAG Command Prompt klikając:
Start -> Programs -> XJTAG <*version*> -> "XJTAG Command Prompt"
Test uruchamiamy wywołując komendę XJEase nazwa.xje np.:

```
XJEase tutorial.xje
```

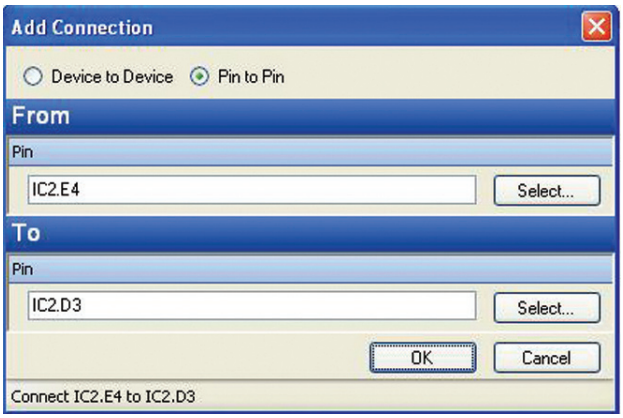
Test powinien się uruchomić automatycznie i po zakończeniu powinien pojawić się na ekranie komunikat:

```
Error: Short found between nets: __Created__IC2.E4, __Created__IC2.D3.
Run CONNTEST at level 0 or 1 for more information.
1 error(s) found in connection test.
ERROR: Exited
End of code reached.
```

Oznacza to, że system podczas testu odnalazł błąd w połączeniu.
W tym przypadku jest to zwarcie na linii pomiędzy pinami IC2.E4 – IC2.D3.

Jak popatrzymy na schemat układu to odnajdziemy, że zwarcie to rzeczywiście tam jest ale nie zostało ono opisane w netliście.
Miało to na celu symulowanie zwarcia dwóch wyprowadzeń układu.
Czynności te były czysto dydaktyczne.
Powyższy błąd można ominąć odpowiednio podając informacje dla systemu o tym, że jest tam zwarcie.
W tym celu wracamy do naszego projektu w XJDeveloper.

- klikamy na ikonę **Connections** w panelu głównym XJDeveloper
- klikamy na Add w oknie **Connections** w oknie **Add Connection** przechodzimy na opcję **Pin to Pin** wpisujemy lub wyszukujemy pinu IC2.E4 w sekcji **From**, natomiast IC2.D3 w sekcji **To**
- zatwierdzamy klikając **OK** a następnie **save**



W oknie **CONNECTION LIST** powinno wyświetla się potwierdzenie połączenia:

```
CONNECT PINS IC2.E4 TO IC2.D3;
```

Uruchamiamy ponownie test. Tym razem zakończył się sukcesem:

```
Connection test completed successfully.
```

Należy pamiętać, że jest to jedynie symulacja przerwania, wykonana celowo przez producenta.
W rzeczywistości błąd taki należy dokładnie przeanalizować porównując ze schematami.

4.2 Test diod LED

Oprócz testu połączeń system XJTAG umożliwia, również przeprowadzanie testów elementów bez wbudowanego interfejsu JTAG. System działanie swoje opiera na skanowaniu oraz ustawianiu stanów pinów układów z interfejsem JTAG. Jeżeli dowolny układ podłączony jest do układu JTAG to wywołując odpowiednie zmiany na pinach możemy komunikować się z prawie dowolnym urządzeniem.
Jeżeli więc wystawimy wysoką wartość na pin do którego podłączona jest dioda LED, wówczas możemy wywołać świecenie tej diody.

Test diod LED
Rozpoczynamy od edytowania pliku LED.xje. Najlepiej za pomocą programu TexPad.
Skrypt zawiera jedynie wstępną konfigurację, którą wcześniej wykonaliśmy. Należy dodać odpowiedni kod źródłowy na koniec pliku.

```
//-------------------------------------------------------------
// LED colours constants
//-------------------------------------------------------------
CONST INT OFF := 0;
CONST INT RED := 1;
CONST INT GREEN := 2;
CONST INT ORANGE := 3;
//-------------------------------------------------------------
SetLED(INT value)()
//-------------------------------------------------------------
 SET RED_LED := value[0], GREEN_LED := value[1];
END;
//-------------------------------------------------------------
Test()(INT result)
//-------------------------------------------------------------
 INT key;
 PRINT("Is LED ", DEVICE_REF, " toggling red and green?\nPress the Spacebar to pass or any
 other key to fail.\n");

 DO
   SetLED(RED)();
   SLEEP(100);
   SetLED(GREEN)();
   SLEEP(100);
   key := GETKEY();
 WHILE key = 0
 END;

 IF key = ASC(" ") THEN
   result := RESULT_PASS;
 ELSE
 result := RESULT_FAIL;
 END;
 //Make sure both the LEDs are switched off at the end.
 SetLED(OFF)();
END;
```

Głównym sercem kodu jest funkcja SetLED().
Wywoływana jest ona z trzema parametrami RED, GREEN, ORANGE. Parametry te odpowiadają wartością: 1, 2, 3. Funkcja SetLED składa się jedynie z komendy **SET**. Odpowiada ona za ustawienie, pinu (RED_LED lub GREEN_LED wartością wysoką niską lub w stan wysokiej impedancji. Wartość jaka ma być wystawiona określona jest przez zmienną value[numer bitu]. W nawiasach kwadratowych jest określony numer bitu.
Jeżeli wyślemy wartość RED = 0x01 (zapis bitowy) to bit zerowy wynosi 1, czyli wartość wysoka. Wówczas na pin RED_LED zostanie wysłana wartość wysoka. Powinna zapalić się dioda czerwona.
Aby uruchomić test należy jeszcze wywołać funkcję „**Test"** oraz zdefiniować stałe:

Stałe definiuje się przed wywołaniem funkcji. W naszym przypadku możemy dodać następujący kod przed funkcję CircuitTest w pliku **circuittest.xje**

```
//-------------------------------------------------------------
// Constants
//-------------------------------------------------------------
// Global constants
CONST INT RESULT_PASS := FALSE;
CONST INT RESULT_FAIL := TRUE;
CONST INT EXIT_ON_ERROR := TRUE;
```

Poniższy kod wywołuje test (funkcję „**Test"**) wpisujemy do pliku circuittest.xje w miejscu gdzie kończy się kod wywołujący test połączeń.

```
PRINT("\n\n---- Running LED D1 Test ----\n\n");
CALL("D1", "Test")()(result);
IF (result != RESULT_PASS) THEN
  PRINT("LED test (D1) failed.\n");
  IF EXIT_ON_ERROR THEN EXIT; END;
ELSE
  PRINT("LED test (D1) passed.\n");
END;
```

Zapisujemy zmiany i wywołujemy ponownie test w XJTAG Command Prompt komendą: XJEase nazwa.xje.
Po uruchomieniu testu w pierwszej kolejności przeprowadzony jest test połączeń a w następnej test diody D1. Dioda miga w trzech kolorach natomiast na ekranie wyświetlił się komunikat do użytkownika z pytaniem o potwierdzenie działania diody.
Był to przykład prostego testu diody. Pozostałe elementy, również mogą zostać przetestowane pisząc dla nich odpowiedni kod, który będzie sterować pinami podłączonymi do tych układów. Wiele kodów źródłowych jest dostępnych na stronie producenta www.xjtag.com. Kody te można w łatwy sposób modyfikować na potrzeby własnych testów i układów.

4.3 Test pamięci Static RAM

Na płytce Demo Board znajduje się również układ StaticRAM BS62LV256SC oznaczony jako IC5. Najprostszy test pamięci jaki można wykonać to proces zapisu i odczytu danych a następnie porównanie ich ze sobą. Jeżeli dane są identyczne to można założyć prawidłowe połączenie z pamięcią. W przypadku wystąpienia błędów, test ten nie zweryfikuje na której linii sygnałowej wystąpił.
Należy w tym celu wykonać kolejne bardziej zaawansowane testy, które to zweryfikują.

4.3.1 Prosty test typu zapis/odczyt

Test odczyt/zapis polega na zapisaniu dowolnych danych do pamięci SRAM o określonym adresie a następnie odczytaniu z tego samego adresu danych i porównaniu ze sobą.

W tym celu należy zdefiniować dwa cykle:
WriteCycle (zapis) oraz **ReadCycle** (odczyt).

WriteCycle
Cykl zapisu składa się z sekwencji (algorytmu) kodu niskiego poziomu, który wymagany jest do komunikacji z pamięcią SRAM. Algorytm ten musi być zgodny z dokumentacją techniczną danego układu, w tym przypadku z układem BS62LV256.

- wystawienie na liniach danych (D0 - D7) wartości, która ma być zapisana, natomiast a liniach adresowych (A0 - A10) wartość adresu
- ustawić pin nCE (Chip Enable) oraz pin nWE (Write Enable) na stan niski (aktywny)
- ustawić piny nCE oraz nWE na stan wysoki

W kodzie XJEase Language będzie wyglądało to następująco:

```
WriteCycle(INT address, INT data)()
SET ADDRESS := address[10..0], DATA := data[7..0];
SET nCS :=0, nWE := 0;
SET nCS :=1, nWE := 1;
END;
```

Jak widzimy funkcja WriteCycle(INT address, INT dada) posiada dwa argumenty wejściowe odpowiadające wartością adresu i danych.
Powyższy kod dodajemy do pliku BS62LV256.xje.
Następnie definiujemy funkcję odczytu **ReadCycle**

ReadCycle
Cykl odczytu składa się z następujących po sobie sekwencji:

- ustawienie na liniach adresu (A0-A10) wartości adresu, spod którego będzie wykonany odczyt, natomiast linie danych należy wyłączyć, ustawiając w stan wysokiej impedancji
- ustawić piny nCE (Chip Enable) oraz nOE (Output Enable) a stan niski (aktywny)
- ustawić piny nCE oraz nOE na stan wysoki i przechwycić dane wystawione na liniach danych.

W kodzie XJEase Language będzie wyglądało to następująco:

```
ReadCycle(INT address)(INT data)
SET ADDRESS := address[10..0], DATA := I; // This will set the
// data bits to input.
SET nCS := 0, nOE := 0;
SET nCS := 1, nOE := 1, data := DATA;
END;
```

Funkcja **ReadCycle** przyjmuję jeden parametr wejściowy **INT address** (wartość adresu) , natomiast jako parametr wyjściowy zwraca **INT data** (wartość danych odczytanych). Powyższy kod należy dodać do pliku BS62LV256.xje.
Powyższe sekwencję zostały opracowane na podstawie tabeli prawdy układu BS62LV256 (tab. 1).

MODE	$\overline{CE}$	$\overline{WE}$	$\overline{OE}$	I/O OPERATION	V_{CC} CURRENT
Not selected (Power Down)	H	X	X	High Z	I_{CCSB}, I_{CCSB1}
Output Disabled	L	H	H	High Z	I_{CC}
Read	L	H	L	D_{OUT}	I_{CC}
Write	L	L	X	D_{IN}	I_{CC}

Kolejnym krokiem jest opracowanie krótkiej procedury, która wywoła funkcje zapisu i odczytu oraz zweryfikuje poprawność zapisu.
Przykładowy kod:

```
TestRAM()(INT result)
  INT val1, val2;
  WriteCycle(0, 0xaa)();
  WriteCycle(1, 0x55)();
  ReadCycle(0)(val1);
  ReadCycle(1)(val2);
  IF (val1 = 0xaa && val2 = 0x55) THEN
    result := RESULT_PASS;
  ELSE
    result := RESULT_FAIL;
  END;
END;
```

Powyższy kod definiuje funkcję TestRAM()(INT result), w której następuje wywołanie funkcji zapisu i odczytu danych.Następnie porównuje się dane ze sobą.
W przypadku poprawnego zweryfikowania funkcja zwraca result := RESULT_PAS (wartość TRUE) w przeciwnym wypadku funkcja zwraca RESULT_FAIL (wartość FALSE).
Powyższy kod należy dodać do pliku BS62LV256.xje.
Brakuje jeszcze wywołania tej funkcji z pliku circuittest.xje.

Można zrobić to w ten sposób:
Przykładowy kod:

```
PRINT("\n\n---- Running SRAM Tests ----\n\n");
CALL("IC5","TestRAM")()(result);
IF (result != RESULT_PASS) THEN
PRINT(„Static RAM (IC5) test failed.\n");;
IF EXIT_ON_ERROR THEN EXIT; END;
ELSE
PRINT(„Static RAM (IC5) test passed.\n");;
END;
```

Uwaga! *Kod należy dodać do pliku circuittest.xje ale przed ostatnim słowem kluczowym **END**. W przeciwnym wypadku kompilator zwróci błąd. Słowo **END** zamieszczone w pliku circuittest.xje kończy całą funkcję **CircuitTest**.*

Teraz wywołujemy naszą funkcję jeszcze raz w aplikacji Command Prompt wpisując komendę **„XJEase tutorial.xje”.**
Po uruchomieniu, XJTAG wykona testy zdefiniowane poprzednio oraz nowy test układu IC5.

4.3.2 Zaawansowany test linii adresowych, danych i sterujących.

Poprzedni test był jedynie namiastką profesjonalnego algorytmu testującego poprawność połączenia z układem SRAM.
Jednym ze standardowych algorytmów testujących pamięci jest algorytm wędrującego ‘zera’ i wędrującej ‘jedynki’. Algorytm ten bazuje również na cyklach odczytu i zapisu unikatowej wartości, wykonanych sekwencyjnie, wielokrotnie zmieniając adres zapisanej wartości. Adres zmieniany jest w ten sposób, że następuje przesuwanie zera (wartości niskiej) linia po linii od A0 do A10, przy zachowaniu na pozostałych liniach jedynki (wartości wysokiej).
Następnie następuje inwersja wartości i przesuwana jest wartość wysoka (wędrująca ‘jedynka). Algorytm ten potrafi wykryć zwarcia lini, przerwania, zwarcia do masy lub zasilania zarówno linii adresowych jak i linii danych.

Algorytm jest dostępny na stronie producenta ***www.xjtag.com.***

Algorytm zawarty jest w pliku **memtestSRAM.xje**, wywołany jest natomiast z pliku **SRAM_SOP28.xje**. Plik **SRAM_SOP28.xje** jest podobnie jak plik BS62LV256, plikiem konfiguracyjnym i testowym pamięci SRAM (IC5).

Najprostszym sposobem aplikowania testu będzie dokonanie zmiany w głównym pliku konfiguracyjnym płytki. Można to zrobić na dwa sposoby.
Pierwszy sposób to edytować plik konfiguracyjny i w miejsce pliku BC62LV256.xje wpisać SRAM_SOP28.xje. Należy jeszcze pamiętać aby plik SRAM_SOP28.xje oraz memtestSRAM.xje skopiować do katalogu projektu.

```
DEVICE LIST
D1, D2, D3, D4, D5, D6, D7, D8, D9 := „LED.xje”;
// IC5 := „BS62LV256.xje”;
IC5 := „SRAM_SOP28.xje”;
IC1 := „inverter.xje”;
IC4 := „24LC32A.xje”;
IC6 := „ADS7830.xje”;
SW1 := „switch.xje”;
SW2, SW3 := „pushbutton.xje”;

END;
```

Drugi sposób jest z wykorzystaniem XJDeveloper'a.
Narzędzie to służy do tworzenia plików konfiguracyjnych, więc posłużyć nam może do ich modyfikacji.

Uruchamiamy XJDeveloper'a klikając podwójnie na plik **Tutorial.xjd,** przechodzimy do zakładki **Categorise Devices**, z okna **Test Devices** wybieramy IC5, klikamy na **Configure**.
W nowym oknie, które się pojawiło wybieramy **Other Device File** i wybieramy nowy plik **SRAM_SOP28.xje.** Po dokonaniu zmian zapisujemy projekt.

Należy dokonać jeszcze jednej drobnej korekty w pliku **circuittest.xje**. W pliku tym wywoływana jest funkcja testująca pamięć RAM o nazwie **„TestRAM"**, natomiast funkcja wywołująca test linii danych i adresu nosi nazwę **„Test"**. Dokonujemy zmian:
Zamieniamy linie kodu:

```
CALL("IC5", "TestRAM")()(result);
```

na poniższą linie kodu:

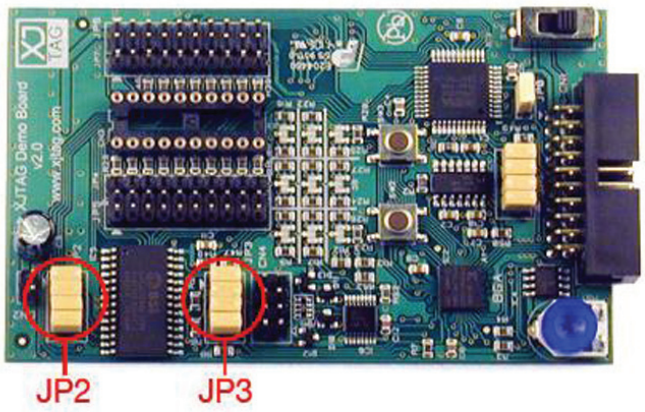
```
CALL(„IC5", „Test")()(result);
```

Wywołujemy ponownie test w Command Prompt komendą **„XJEase tutorial.xje".**

Po uruchomieniu na ekranie wyświetlają się ostrzeżenia informujące, że nie można wystawić na pin nCE wartości wysokie i niskiej. Ostrzeżenie to wynika z faktu, że linia ta jest zwarta do masy. Widoczne jest na schemacie układu.Ostrzeżenia te możemy zignorować lub ewentualnie uwzględnić to w skrypcie testującym, w którym możemy w ogóle nie brać pod uwagę sygnału nCE w sekwencjach zapisu i odczytu.

Aby się upewnić, że ten algorytm działa poprawnie, przeprowadźmy kolejną symulację.

Na płytce Demo Board pamięć SRAM jest obudowana zworkami, usunięcie jednej ze zworek spowoduje zerwanie linii adresowych lub danych.
Na rysunku poniżej przedstawione są jako JP2 i JP3.
Gdy wyciągniemy jedną z nich i uruchomimy test, wyświetli się komunikat o błędzie i test zostanie „niezaliczony".



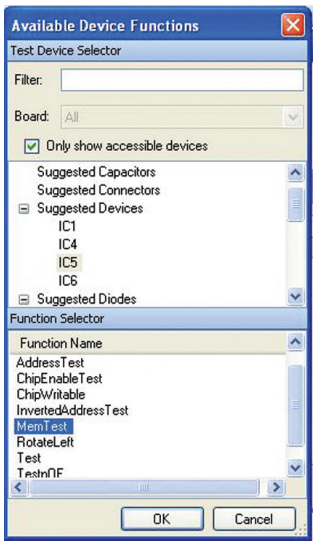
5. XJRunner

XJRunner jest wyspecjalizowanym środowiskiem uruchomieniowym dla testów opracowywanych za pomocą pakietu XJEase. XJRunner wykorzystuje spakowany plik, zawierający wszystkie wymagane komponenty oraz pliki testowe wykorzystywane w projekcie.
Szereg specjalnych właściwości sprawia, iż jest on skierowany głównie na potrzeby producentów płytek oraz testowania w warunkach mobilnych. Można określić XJRunnera jako końcową aplikację w cyklu życiowym testowania płytki.

5.1 Przygotowanie pliku źródłowego

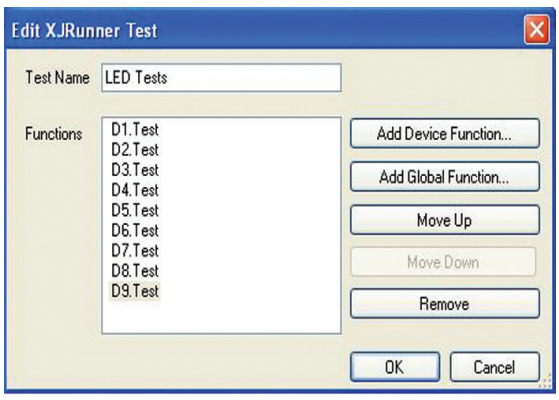
Przed rozpoczęciem testów za pomocą XJRunnera należy odpowiednio przygotować plik, poprzez sporządzenie odpowiedniej listy testów. W tym celu wykorzystujemy ponownie XJDeveloper.

Po uruchomieniu wykonujemy następujące czynności:
- wchodzimy do zakładki **XJRunner Test List**
- w oknie X**JRunner Tests** klikamy na **New**
- w oknie które się pojawiło w pierwszej kolejności wybieramy **Add Global Function**
- z listy funkcji, które się pojawiły wybieramy **CONNTEST** i klikamy **OK**
- następnie wpisujemy nazwę testu np.: **Connection Test** klikamy **OK** i zapisujemy



W tym momencie zadeklarowaliśmy funkcję globalną.
Następnie zadeklarujemy funkcje testujące układy. Czynności są bardzo podobne do poprzednich. Funkcje będą testować układ IC5 oraz diod LED D1-D9

- w oknie **XJRunner Tests** klikamy na **New**
- w oknie które się pojawiło w pierwszej kolejności wybieramy **Add Device Function**
- w oknie **Available Device Functions** wybieramy układ, który chcemy przetestować. W pierwszej kolejności wybieramy układ IC5 z listy testów **Function Selector** wybieramy **MemTest** i zatwierdzamy **OK**
- wpisujemy nazwę np.: **StaticRam Test**, klikamy **OK** i zapisujemy
- następne dodajemy funkcje testujące diody LED, wybieramy z listy układów diody LED, zaznaczając od D1 do D9 i klikamy **OK**
- wpisujemy nazwę testu np.: LED tests
- zapisujemy cały projekt

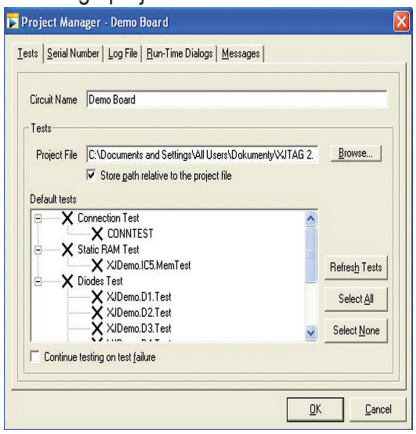


Następnym krokiem w przygotowywaniu pliku źródłowego będzie spakowanie wszystkich wymaganych komponentów do jednego pliku z rozszerzeniem ***xjp.**
Procedurę tą wykonujemy w aplikacji Command Prompt wpisując polecenie: **„XJPack tutorial.xje"**.
Następnie wyświetlą się wszystkie spakowane komponenty.
W tym momencie zakończyliśmy proces tworzenia pliku źródłowego, wymaganego przez XJRunner.

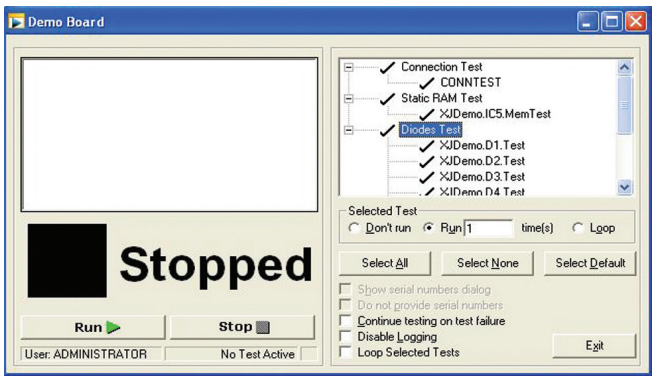
5.2 Wykonanie testów za pomocą XJRunner

W pierwszej kolejności uruchamiamy XJRunner.
Po uruchomieniu wyświetli się okno wymagające podania loginu i hasła. Jeżeli hasło i login nie zostały zmienione to domyślnie ustawienie jest **ADMINISTRATOR** jako hasło i login. Po przejściu przez logowanie do programu wyświetli się główne okno dialogowe XJRunner'a.
W celu rozpoczęcia testów wykonujemy następujące czynności:

- w głównym oknie wybieramy **New Project**
- w oknie **Project Manager** wpisujemy nazwę układu np.: **Demo Board** oraz należy wybrać (zlokalizować) nasz plik źródłowy. Klikamy na **Browse** i wybieramy nasz plik **tutorial.xjp**. Plik znajduje się w głównym katalogu projektu.
- klikamy **OK**
- w tym momencie załadowane zostały wybrane przez nas testy
- klikamy **OK**, po zatwierdzeniu wyświetla się pytanie o zapisaniu projektu
- zapisujemy projekt o dowolnej nazwie. Program przypisał nazwie projektu rozszerzenie ***XJRP**- po zapisaniu nastąpił powrót do głównego okna XJRunner'a

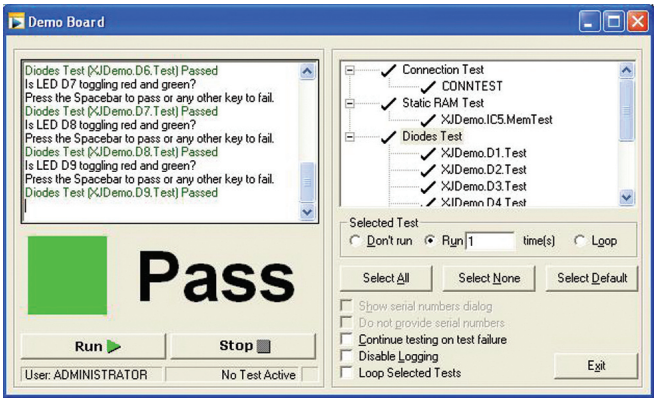


- klikamy na **Tools -> Run Tests** lub wchodzimy przez skrót znajdujący się na pasku narzędzi
- po uruchomieniu wyświetliło się okno dialogowe służące do zarządzania testami
- na obecną chwile wszystkie testy są wyłączone. Należy je aktywować, klikając podwójnie na wybrany test (rysunek poniżej)
- oprócz włączenia i wyłączenia poszczególnych testów XJRunner posiada wiele dodatkowych możliwości, takich jak wielokrotne wykonanie testów, kontynuowanie mimo błędów itp. Opcji tych nie będziemy teraz potrzebować



- aby uruchomić testy należy kliknąć na **Run**
- po uruchomieniu testów w oknie dialogowym wyświetlają się te same komunikaty jake wyświetlały się podczas uruchamiania za pomocą komendy **„XJEase"** w **Command Prompt**.

Po wykonaniu pozytywnie wszystkich testów wyświetli się komunikat informujący zaliczenie testów płytki **„Pass"**. Jeżeli test nie zakończył się sukcesem to wyświetli się komunikat **„Fail"**. Można to przetestować usuwając jedną ze zworek przy pamięci StaticRAM



6. XJTAG jako programator układów CPLD

System XJTAG, jak się już przekonaliśmy, posiada wiele możliwości. Dodatkową zaletą tego systemu jest programowanie układów CPLD. W tym celu wykorzystuje on odpowiednio przygotowane pliki SVF/STAMPL a następnie za pomocą interfejsu JTAG odpowiednio konfiguruje układ CPLD.

W niniejszym rozdziale przedstawiona zostanie kompletna procedura programowania układów CPLD, obejmująca przygotowanie przykładowego projektu dedykowanego pod układy CPLD, przygotowanie plików SFV i STAMPL a następnie programowanie z wykorzystaniem XJTAG.

Projekt zostanie wykonany na przykładzie układu Xilinx XC9536XL, znajdującego się na płytce Demo Board. Przygotowanie plików SFV/STAMPL odbędzie się za pomocą darmowych narzędzi oferowanych przez firmę Xilinx: ISE WebPACK.

6.1 Przygotowanie plików SVF/STAMPL

Program

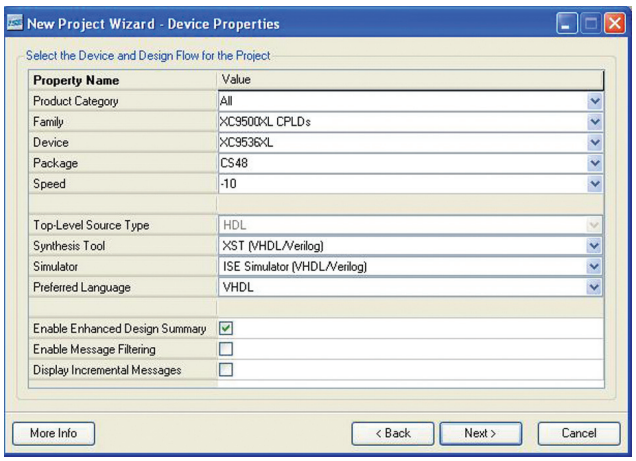
W pierwszej kolejności należy napisać odpowiedni program w języku VHDL dedykowany pod układ XC9536XL. Dla naszych potrzeb stworzony został program, symulujący działanie pamięci ROM, w której zostały zapisane wartości w momencie konfigurowania. Zapisane wartości wyświetlane są na diodach LED. Odczyt poszczególnych wartości odbywa się po przyciśnięciu przycisku SW3, który inkrementuje wartość adresu. Przyciśnięcie przycisku SW2 powoduje wyzerowanie adresu.

Tworzenie projektu

Napisany program należy poddać odpowiednim procedurom charakterystycznym dla układów programowalnych (synteza, implementacja, generowanie plików programujących). W tym celu wykorzystamy środowisko Xilinx ISE. W tym celu wykonujemy następujące kroki:

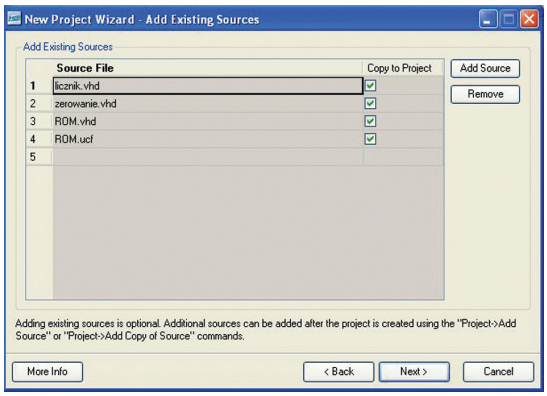
- Tworzymy nowy projekt i wybieramy odpowiedni układ. (Dokładne dane dotyczące układu XC9536XL odczytujemy z dokumentacji technicznej).

Poniżej znajduje się zrzut ekranu przedstawiającego poprawną konfigurację układu.

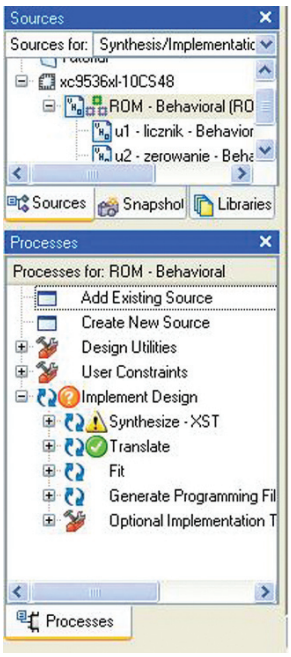


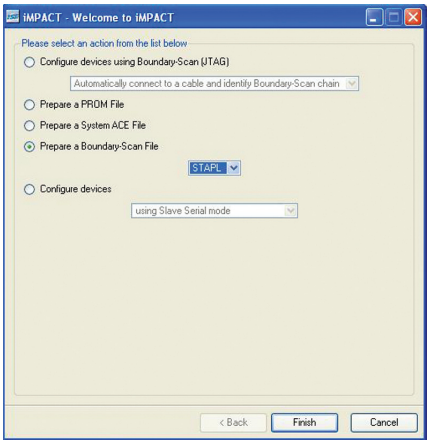
- następnie zatwierdzamy klikając **Next**
- W następnym oknie dialogowym mamy możliwość utworzenia nowego źródła, pomijamy to klikając **Next**. W naszym projekcie nie będziemy tworzyć nowych źródeł lecz dodamy już gotowe pobrane ze strony **www.quantum.com.pl**.
- Następne okno dialogowe dotyczy dodania gotowych źródeł. W tym przypadku klikamy na **Add Source**.

Dodajemy do naszego projektu cztery pliki źródłowe: **licznik.vhd, zerowanie.vhd, ROM.vhd oraz ROM.ucf**. Pliki z rozszerzeniem ***.vhd** to pliki z kodem źródłowym w języku VHDL, natomiast plik *.ucf to plik zawierający informację o konfiguracji wyprowadzeń układu. Plik ten został stworzony za pomocą oprogramowania **PACE** dostępnego w pakiecie ISE WebPACK. Po dodaniu plików, klikamy na **Next**. a następnie na **Finish**.



- Po zakończeniu, pliki zostaną dodane do projektu oraz automatycznie syntezowane.
- Pliki te możemy edytować klikając podwójnie na wybrany plik w górnym lewym oknie **Sources**.
- Następnie należy przeprowadzić proces implementacji projektu, wykonujemy to klikając podwójnie na Implement Design w oknie Processes, wówczas nastąpi wykonanie kolejno wszystkich wymaganych procedur: **Synthesise, Translate, Fit, Generate Programming File**. Można również procedury te wykonać indywidualnie (zalecane). Jeżeli procedura nie powiedzie się należy ją powtórzyć.
- Po zakończeniu implementacji należy wygenerować pliki SVF lub STAMPL, W tym celu rozwijamy opcję **Optional Implementation Tools** w oknie **Processes**. Podwójnie klikamy na **Generate SVF/XSVF/STAMPL**. Nastąpi uruchomienie programu **iMPACT**.

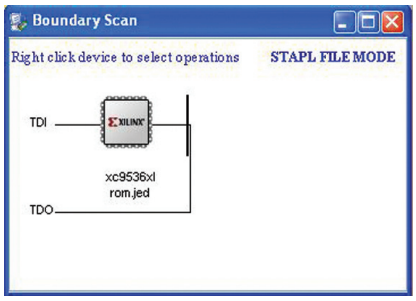




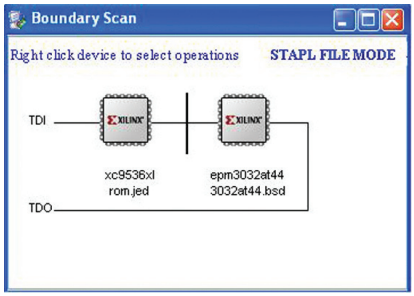
● Po uruchomieniu należy wstępnie skonfigurować projekt ustawiając opcję **Prepare a Boundary-Scan File** i wybrać rodzaj generowanego pliku.
W naszym przypadku klikamy na **STAMPL**. Klikamy na **Finish.**

● Po zatwierdzeniu należy wpisać nazwę pliku STAMPL i wybrać miejsce, w którym będzie zapisany.

● Następnie należy dodać odpowiedni plik utworzony w procesie Implementacji.
W tym celu automatycznie wyświetli się okno **Add Device** lub w przeciwnym wypadku należy kliknąć prawym przyciskiem myszy i wybrać opcję **Add Device.**



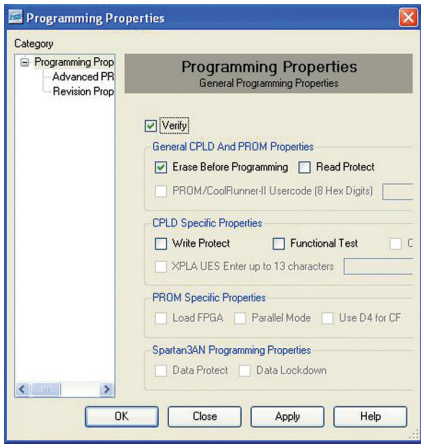
● Wybieramy plik **ROM.jed**, powinien pojawić się w głównym oknie nasz układ z zaznaczonym wejściem TDI i wyjściem TDO.



● Należy pamiętać, że nasz płytka składa się dodatkowo z drugiego programowalnego układu firmy Altera. Układy te połączone są w łańcuch. Należy to dodatkowo uwzględnić w projekcie.
W tym celu dodajemy kolejny układ klikając prawym przyciskiem myszy na wolne pole i wybieramy **Add non-Xilinx Device.**
Po wyświetleniu pytania czy chcemy dodać plik BSDL, klikamy **Yes.**

● Wyszukujemy pliku BSDL naszego drugiego układu.
Dostępny on jest w katalogu BSDL

Po dodaniu pliku w oknie wyświetli się poprawna konfiguracja naszej płytki.



● W następnej kolejności wykonujemy proces zapisu wektorów programowania do pliku STAMPL. Klikamy prawym przyciskiem myszy na układ XC9536XL a następnie wybieramy opcję Program.

● W nowym oknie wybieramy opcję procesu programowania. Ważne jest aby zaznaczona była opcja Erase Before Programming oraz opcja Verif. Wówczas podczas programowania nastąpi wykasowanie poprzedniej konfiguracji układu CPLD oraz zweryfikowana poprawność konfiguracji.

● Zatwierdzamy klikając Apply i OK.

● W następnej kolejności należy zablokować plik STAMPL przed zapisaniem kolejnych strumieni danych.
Klikamy prawym przyciskiem myszy na wolne pole i wybieramy Output File Type -> STAPL File -> Stop Writting to File. Plik został zamknięty.

● W obecnej chwili plik STAMPL jest już gotowy do wykorzystania przez XJTAG.

6.2 Programowanie układu XC9536XL za pomocą XJTAG Boundary Scan

Do programowania układów CPLD za pomocą systemu XJTAG, najłatwiej wykorzystać XJAnalysera, który ma wbudowaną opcję importowania plików SVF/ STAMPL.

● Uruchomiamy XJAnalysera i tworzymy nowy projekt, tak jak to miało miejsce w rozdziale drugim.

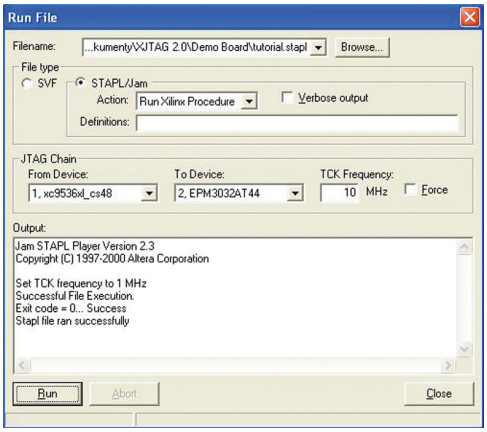
● Praca łańcucha JTAG musi być wyłączona aby aktywna była opcja importowania STAMPL/SFV.

● Klikamy na przycisk SVF, widoczny w pasku narzędzi głównego okna

● W nowym oknie Run File ustawiamy opcję programowania z pliku STAMPL oraz wyszukujemy go na dysku klikając na Browse.

● Następnie klikamy na Run. XJTAG po chwili zaprogramuje układ XC9536XL

● Po zaprogramowaniu zamykamy programator klikając na **Close**.



Podgląd działania naszego programu możemy zobaczyć na żywo klikając na przycisk SW3 i SW2.
Widzimy jak kolejno są wyświetlane wartości zaprogramowane w układzie CPLD.

Dodatkowo działanie programu możemy zobaczyć w oknie XJAnalysera jak ustawimy układ w stanie SAMPLE:

● W pierwszej kolejności uruchamiamy łańcuch JTAG klikając na start JTAG Scan (F5)

● prawym przyciskiem myszy klikamy na układ XC9536XL, wybieramy opcję Mode ->Sample. Układ będzie w stanie analizowania sygnałów przychodzących z zewnątrz.
Jeżeli jesteśmy zainteresowani zaprogramować układ wykorzystując plik SVF należy na wybrać rodzaj pliku SVF przy pierwszym uruchomieniu IMPACT a następnie powtórzyć te same czynności jak dla pliku STAMPL.

7. Przypadki wykorzystania oraz kierunki rozwoju systemu XJTAG Boundary Scan

Obecny szybki rozwój systemów elektronicznych przyczynia się również do szybkiego rozwoju technik testowania. Zintegrowane płytki elektroniczne, charakteryzujące się wysoka skalą upakowania elementów na powierzchni płytki, montażem obustronnym z wykorzystaniem elementów BGA. Jak w poprzednim rozdziale wspomniano, że obecne techniki testowania zarówno metody dotykowe jak i optyczne nie spełniają podstawowych wymagań. Metodami optycznymi nie można wykonać analiz elektrycznych, jedynie strukturalne. Dodatkową wadą metod optycznych jest wysoki koszt. Metodami dotykowymi nie jest możliwe testowanie układów w obudowach BGA oraz płytek obustronnie montowanych. Dlatego też producenci oraz projektanci płytek coraz częściej sięgają po technikę JTAG Boundary Scan. Rozwój tej techniki przyczynił się również do dynamicznego rozwoju firmy XJTAG, oferującej w tym zakresie kompletne niezbędne narzędzia, wsparcie techniczne oraz sieć sprzedaży

7.1 Konkretne możliwości XJTAG

XJTAG jest narzędziem testującym zintegrowane płytki drukowane lub gotowe już systemy elektroniczne w całym cyklu życia produktu od momentu tworzenia prototypu, do gotowych finalnych produktów a nawet w serwisie gwarancyjnym. Podstawowym zadaniem do którego została opracowana technika JTAG Boundary Scan to sprawdzanie poprawności montażu układu JTAG do powierzchni płytki. Następnie testy poszerzono do sprawdzenia poprawności połączeń z sąsiednimi elementami połączonymi w sieć JTAG.
Kolejnym krokiem na przód było implementacja odpowiednich algorytmów testowych, napisanych w języku wysokiego poziomu, do testowania układów typu non-JTAG. Możliwe to było dzięki integracji odpowiednich narzędzi (sprzętu z oprogramowaniem).
Wykorzystanie języka wysokiego poziomu owoduję zwiększenie elastyczności, uniwersalności oraz odseparowanie inżyniera od dokładnej wiedzy na temat techniki JTAG Boundary Scan. System XJTAG wprowadził do swojej platformy język XJEase Language. Jest to język mający kilka cech wspólnych z językiem ANSI C ale znacznie bardziej przystosowany do „rozmawiania" ze sprzętem. Posiada on instrukcje sterujące, funkcję operujące na plikach, operatory i wiele innych możliwości charakterystycznych dla języka wysokiego poziomu.
Kolejną cechą języka XJEase jest modułowość czyli łatwość rozbijania projektów na pliki. Projekty stają się przejrzyste i łatwe w modyfikacji lub rozszerzaniu na kolejne moduły.
XJTAG umożliwia testowanie takich elementów na płytce jak:

- Pamięci półprzewodnikowe
  - EEPROM
  - Flash
  - Karty pamięci:
    Compact Flash Card, SD Card
  - SDRAM/DDR
  - SSRAN/ZBTRAM
  - Static RAM
  - Algorytm: wędrujące „zero", wędrująca „jedynka"
- Ethernet
- Konwerter Analogowo-Cyfrowy
- Zegar czasu rzeczywistego
- Układy wideo
- Przełączniki
- Watchdog
- Protokoły
  - IIC, PCI, SPI

7.2 Kierunki rozwoju

XJTAG idąc za potrzebami rynku, szczególnie w branży przemysłowej, wprowadził nową kartę **3U PXI XJTAG**.
Jest to karta rozszerzeń dla platformy PXI LabVIEW. Nowa karta PXI XJTAG umożliwi użytkownikom platformy PXI na poszerzenie możliwości wyszukiwania błędów (debugowania) i testowania zintegrowanych płytek drukowanych zawierających układy typu BGA.
Dodatkowo, wszystkie zalety aktualnego oprogramowania zawierającego zestaw narzędzi wirtualnych (Vis), zostały dodane do interfejsu systemu XJTAG.

Karty PXI XJTAG

PXI łączy właściwości elektryczne szyny PCI z odpornością na warunki zewnętrzne, modułowością, oraz możliwość zabudowy typu Eurocard.. PXI jest wysoko wydajną, o niskich kosztach wdrożenia, platformą znajdującą zastosowania w technice pomiarowej i systemach automatyki.

System PXI znajduje zastosowanie w technice wojskowej i lotniczej, monitoringu pracy maszyn, motoryzacji oraz testach przemysłowe i produkcyjnych.

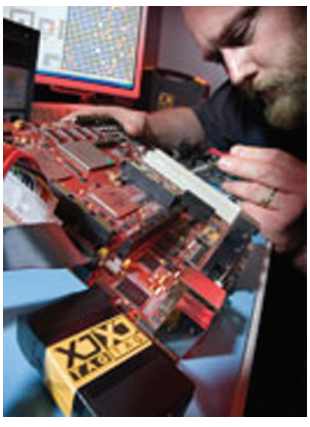
7.3 Firmy, które już postawiły na system XJTAG

Liczba firm, które wykorzystują system XJTAG stale rośnie.
Są to firmy, która kładą nacisk na jakość swojego produktu, dla których szybkie wprowadzenie produktów na rynek ma ogromne znaczenie oraz mimo dużego kapitału, liczą się z kosztami.

Oto kilka przykładów:

Korporacja ARM

Jest światowym liderem projektującym i udzielającym licencji na produkcję procesorów opartych o rdzenie ARM. System XJTAG został wykorzystany do budowy najnowszej wersji płytki uruchomieniowej środowiska „RealView".
W trakcie projektowania płytki uwzględniono rodzaj technologii testującej (design-for-test, DTF) w tym przypadku uwzględniono technikę JTAG Boundary Scan.
Pozwoliło to na testowanie i poprawianie błędów już w początkowym stadium tworzenia płytki.
Projektowanie płytki z uwzględnieniem technologii XJTAG przyniosło niesamowite korzyści firmie ARM. Zapewniło oszczędność czasu i przyspieszyło procedurę wprowadzania nowego produktu na rynek, dzięki przyspieszonemu procesowi uruchamiania płytki oraz szybkim i dokładnym korekcjom błędów.



Curtiss-Wright Controls Embedded Computing

Firma jest czołowym projektantem i producentem systemów oraz urządzeń elektronicznych określanych jako Commercial-Off-The-Shelf (COTS), a więc cywilnych (komercyjnych), zaadaptowanych na potrzeby wojskowe.
Produkcja tych systemów prowadzona jest od poziomu płytki drukowanej aż do osiągnięcia kompletnych zintegrowanych podsystemów elektronicznych na potrzeby wojska, lotnictwa lub zastosowań komercyjnych.
System XJTAG został zastosowany przez grupę inżynierów z Curtiss-Wright w Letchworth (UK) zajmującymi się systemami graficznymi, wideo oraz alokacji radarowej.

Wykorzystali oni system XJTAG do wyszukiwania błędów, usterek oraz do testowania poszczególnych elementów elektronicznych w ostatnich, najnowszych wersjach zintegrowanych płytek drukowanych. W układach tych zintegrowana jest znaczna ilość programowalnych układów FPGA, zamkniętych w obudowach typu BGA (Ball Grid Array).

Alan McCormick, dyrektor, menadżer grupy inżynierów graficznych i wideo w Curtiss-Wright, powiedział:

"***Zdecydowaliśmy się wybrać system XJTAG ze względu na jego cenę, szybkość i dokładność sprawdzania błędów oraz ze względu na wielokrotne wykorzystanie skryptów testowych, które mogą być przenoszone i adoptowane z jednego projektu na drugi oraz migrować wraz z prototypem aż do osiągnięcia produktu finalnego.***

„Wykorzystując system XJTAG możemy bardzo szybko przeprowadzić testy układów z implementowanym łańcuchem Boundary Scan jak i u układy bez tej implementacji. Takiej możliwości nie dawały dotychczas stosowane tradycyjne metody diagnostyczne wykorzystujące analizatory sygnałów, oscyloskopy oraz systemy X-ray."

TVonics Solution

Firma specjalizuję się w projektowaniu i produkowaniu telewizyjnych cyfrowych systemów elektronicznych w tym nadajników, odbiorników, przetworników itp.
Firma TVonics Solution jest również partnerem Sony Manufacturing, który jest czołowym dostawcą układów scalonych dla TVonics. Mike Jones, jeden z menadżerów wyjaśnia:

„Profesjonalny system XJTAG jest niezwykle opłacalnym i elastycznym rozwiązaniem. Zastosowanie tego systemu pomoże nam na dalsze skrócenie czasu rozwoju projektu, co ma bardzo duże znacznie przy szybko zmieniającym się rynku konsumenckim w branży TV."

Inżynierowie z działu projektowania w Bridgend, South Wales (UK), wykorzystują system XJTAG do szybkiej weryfikacji i wyszukiwania błędów oraz testowania płytek, zbudowanych z układów BGA. Płytki te zawierają wiele typów układów elektronicznych, między innymi: dekodery MPEG, pamięci Flash, kontrolery Ethernet, układy I2C i Tvonic dodatkowo zintegrował system XJTAG z linia produkcyjną w fabryce Sony Manufacturing.

Mike Jones wyjaśnia:

„***Wykorzystujemy XJTAG w całym cyklu życia produktu. Nasi inżynierowie używają go do wstępnego uruchamiania płytek prototypowych, natomiast inżynierowie odpowiedzialni za produkcję, zintegrowali system XJTAG z automatyczną linią produkcyjną na której są testowane układy DVR (Digital Video Recorder) oraz pamięci NOR Flash. Przyczyniło się to do wzrostu szybkość i poprawy jakości procesu testowania na końcowego produktu".***

Powyższe przykłady pokazują tylko niewielką część firm, które zdecydowały się na wybór systemu XJTAG. Relacje menadżerów oraz inżynierów, określają jasno, że system XJTAG jest profesjonalnym narzędziem, o niskiej cenie, przynoszącym niezmierne korzyści jakościowe, czasowe oraz finansowe.

Podsumowanie

Firma XJTAG umożliwia 30 dniowe bezpłatne wypróbowanie systemu XJTAG, zapewnia obszerne wsparcie techniczne, posiada programy dla uczelni, którym zapewniają wysokie upusty cenowe.

Projektowanie płytek drukowanych z uwzględnieniem systemu testującego, bazującego na technice JTAG Boundary Scan

- wskazówki dla projektantów (DFT- Design For Testability)

Wstęp

Poniższy rozdział zawiera pewne wskazówki, dla projektantów płytek drukowanych, dotyczących poprawienia testowalności płytek drukowanych z wykorzystaniem systemu XJTAG. Poniższe zalecenia nie powinny stanowić sztywnych i twardych wskazań ale powinny być traktowane jako sugestie.
Projektowanie dla poprawienia testowalności nie powinno być jedynym wyznacznikiem w tworzeniu produktu ale stanowić jako jeden z pozostałych warunków, które są brane pod uwagę podczas projektowania. Są to: funkcjonalność, koszty elementów, wielkość płytki oraz poziom skomplikowania.

Wykorzystanie układów JTAG

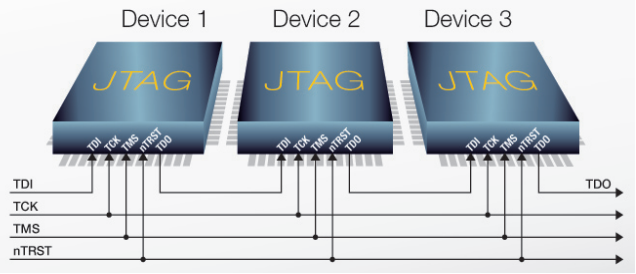
System XJTAG umożliwia testowanie układów typu non-JTAG, ale pomimo tej możliwości, najlepiej jest wykorzystywać układy typu JTAG, gdzie tylko jest to możliwe. Im większa ilość zastosowanych układów JTAG, tym większa ilość węzłów sieci JTAG znajduję się na płytce. Wówczas obszar testowania płytki zwiększa się a czas potrzebny na przygotowanie procedury testowej znacznie się skraca.

Sprawdzanie plików BSDL dla układów JTAG

Układy JTAG wymagają plików BSDL (Boundary Scan Description Language) i powinny być dostępne dla każdego układu JTAG. Pliki te zawierają opis struktury łańcucha JTAG Boundary Scan. Należy wcześniej się upewnić się czy pliki BSDL są dostępne, najczęściej na stronie producenta oraz czy plik BSDL jest aktualny i zgodny ze standardem. Składnia oraz poprawność pliku BSDL jest sprawdzana przez system XJTAG.

Upewnić się czy łańcuch JTAG jest poprawnie zaprojektowany i wyprowadzony na zewnątrz

Należy się upewnić czy wszystkie układy JTAG mają poprawnie wyprowadzone sygnały z wewnętrznego kontrolera TAP. Preferuję się aby sygnały JTAG poprowadzone zostały z dala od innych aktywnych sygnałów oraz odseparowane za pomocą linii zasilających (VCC lub GND). Należy również stosować odpowiednio dopasowane (standardowe) konektory aby umożliwić łatwy dostęp do sygnałów JTAG.



Zabezpieczenie przez niebezpieczeństwem wystąpienia przesłuchu pomiędzy liniami JTAG

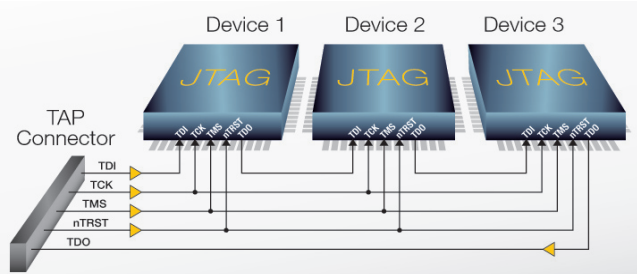
Bardzo ważne jest aby przeprowadzone operacje odbyły się poprawnie, bez przekłamań na liniach JTAG. Szczególnie dotyczy to linii TCK oraz TMS, ponieważ to one są odpowiedzialne za kontrolowanie łańcucha. Dodatkowo długość linii oraz pojemność pasożytnicza powinna być równa dla każdej z linii.

Odpowiednie zakończenie sygnałów JTAG

Sygnał TCK powinien być zakończony rezystorem 68K podpiętym szeregowo oraz zwarty do masy kondensatorem 100pF. Sygnały TDI oraz TMS powinny być podciągnięte do linii zasilających za pomocą rezystora 10K. Sygnał TDO również podciąga się do zasilania rezystorem 10K oraz dodatkowo podłącza się szeregowo rezystory 22R przy wyprowadzeniu z ostatniego układu w łańcuchu. Również linia nTRST wymaga podpięcia do masy rezystorem aby zapobiec wejściowym upływom. Wartość rezystora powinna być indywidualnie dobrana.

Buforowanie sygnałów JTAG

Jeżeli to możliwe, należy buforować sygnały JTAG aby minimalizować szum, niedopasowanie impedancji oraz aby poprawić obciążalność wyjściową



Zapasowe wyprowadzenia pinów układów JTAG

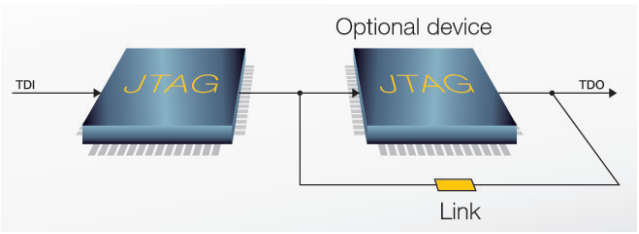
Dodatkowe wyprowadzenia sygnałów z nieużywanych pinów układów JTAG umożliwiają w niektórych sytuacjach, monitorowanie dodatkowych obszarów na płytce pod warunkiem, że nie będą wchodziły w konflikt z normalnymi operacjami układów. Procedura ta może znacznie poszerzyć obszar testowalności.

W pełni wykorzystać wyprowadzenia I/O układów JTAG

W przypadku gdy układ JTAG połączony jest z układem non-JTAG, należy wykorzystać piny układu JTAG aby przeprowadzić komunikację pomiędzy tymi układami. W tym celu wysyłamy odpowiednią sekwencję danych do układu non-JTAG a następnie odczytujemy odpowiedź.

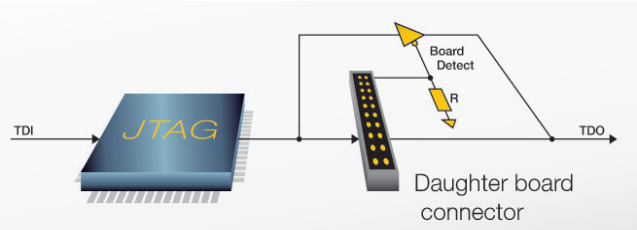
Obejście układu opcjonalnego

Jeżeli na płytce znajduję się układ JTAG, którego obecność jego jest opcjonalna, należy zapewnić. odpowiednio obejście linii danych. W momencie gdy opcjonalny układ będzie nie wykorzystany (nie zamontowany) wówczas należy odpowiednio połączyć linie danych.



Dodatkowa opcjonalna płyta zewnętrzna

Jeżeli do płyty głównej dołączona jest dodatkowa płyta zewnętrzna, wówczas możemy do niej również poprowadzić łańcuch JTAG. Należy jedynie pamiętać, aby wykonać odpowiedni układ logiczny do detekcji płytki aby zapewnić odpowiednie poprowadzenie linii danych w momencie gdy płytka nie będzie podłączona.

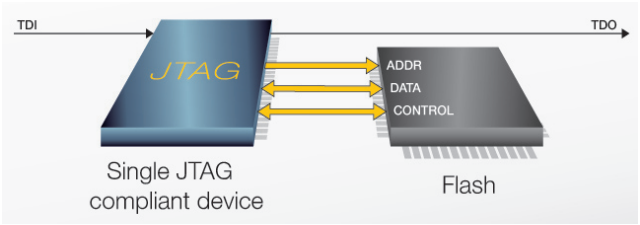


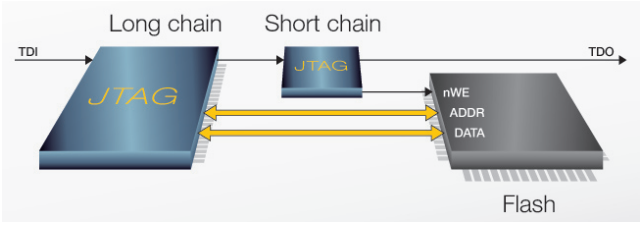
Układy typu ASIC

Jeżeli na płytce będą znajdować się układy ASIC, projektant powinien wziąć pod uwagę połączenia go z układem JTAG.

Dostęp do programowalnych układów

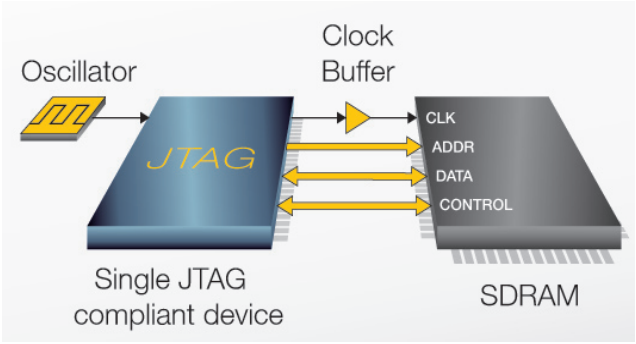
Jeżeli płytka wymaga zastosowania układów programowalnych (EEPROM, Flash, FPGA, CPLD) a układy te nie mają implementacji JTAG, wówczas dostęp do wszystkich pinów jest możliwy za pośrednictwem pinów układu JTAG. Czasami wykorzystuje się krótkie rozgałęzienie łańcucha JTAG aby kontrolować wyprowadzenie układów, którego stan często zmienia się (sygnał nWE układu pamięci Flash). Takie rozwiązanie skraca długość łańcucha oraz poprawia oraz zapewnia lepszą kontrolę.





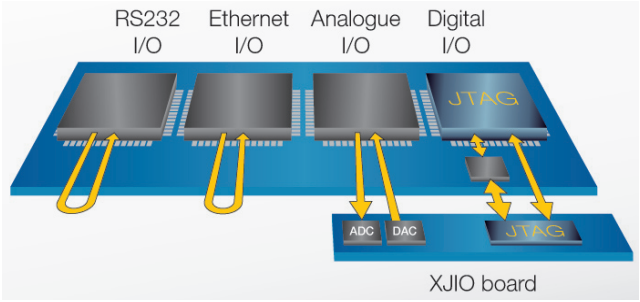
Testowanie układów synchronicznych

Układy synchroniczne, również mogą być testowane przez system XJTAG, pod warunkiem, że możliwa jest kontrola nad sygnałem zegarowym. Na przykład: pamięć SDRAM podłączona do układu FPGA może być testowana w momencie kiedy sygnał zegarowy będzie kontrolowany przez układ JTAG. Jeżeli natomiast pamięć SDRAM będzie taktowana samodzielnym zegarem, wówczas XJTAG nie będzie mógł synchronizować wektory testujące wysyłane do SDRAM w czasie rzeczywistym. Jeżeli możliwe jest aktywowanie łańcucha boundary scan w układzie CPLD lub FPGA, który będzie sterował pamięcią SDRAM, należy samodzielny sygnał zegarowy poprowadzić przez ten układ sterujący. Dodatkowo można zastosować bufor.



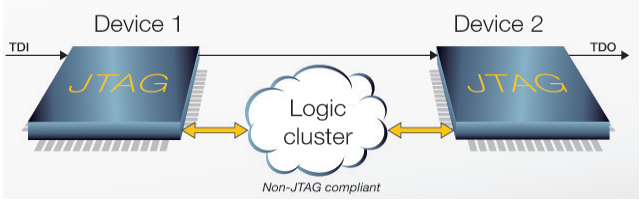
Testowanie układów peryferyjnych I/O

Testowanie wyjściowych układów peryferyjnych oraz złączy możliwe jest kiedy połączone są z układem JTAG. Poszerzenie testowalności układów zewnętrznych najlepiej wykonać za pomocą zewnętrznej płytki XJIO. Jeżeli testowanie zewnętrznych układów nie jest możliwe tym sposobem można zastosować pętle zwrotną.



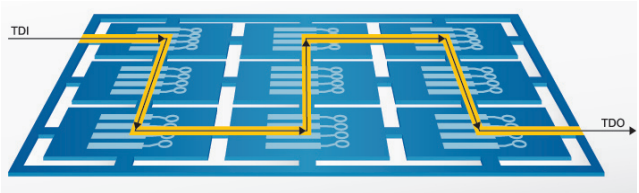
Pogrupowane układy logiczne

Jeżeli płytka będzie zawierała kilka układów logicznych typu non-JTAG to należy je pogrupować w taki sposób aby tworzyły klaster zakleszczony pomiedzy układami JTAG. Wówczas można kontrolować sygnały wejściowe oraz monitorować sygnały wyjściowe. Należy się upewnić, że ilość węzłów łańcucha JTAG jest wystarczająca aby przeprowadzić pełne testy. Jeżeli liczba węzłów jest niewystarczająca oraz jeżeli to możliwe należy dodać kolejny układ JTAG.



Testowanie wielu płyt jednocześnie

Jeżeli projekt składa się z kilku płytek to należy rozważyć przeprowadzenie testów całego złożonego układu. Wówczas należy pamiętać aby poprowadzić sygnały zasilania oraz JTAG przez wszystkie testowane płytki.



Uwaga na Watchdog'a

Jeżeli projekt zawiera watchdog, należy upewnić się że układ ten może być wyłączony podczas przeprowadzania testów. Watchdog'a można wyłączyć wykorzystując obejście sygnałów watchdog'a. Drugie rozwiązanie to sterowanie układem watchdog za pomocą JTAG'a. Nieokreślone skutki mogą się pojawić gdy układ watchdog'a wyśle sygnały resetujące podczas testów. Alternatywnym rozwiązaniem jest upewnienie się, że czas programowy watchdog'a jest dłuższy od czasu przeprowadzenia testów.

Wykorzystanie nieulotnej pamięci do przechowywania konfiguracji

Można użyć nieulotnej pamięci znajdującej się na płytce aby przechować dane potrzebne do przeprowadzenia operacji testowania np.: opcje testowania, numery seryjne, numery MAC itp.

Układy analogowe

Technika JTAG Boundary Scan najlepiej sprawdza się testując układy cyfrowe, natomiast w przypadku układów analogowych należy znacznie dokładniej zaprojektować procedurę testowania.
Można do układów JTAG doprowadzić sygnały wejściowe lub wyjściowe i odpowiednio przetworzyć. Jednym z dobrych rozwiązań jest dołączenie przetworników ADC oraz DAC. Dodatkowe przetworniki umożliwiają komunikację z układami analogowymi za pomocą interfejsów cyfrowych.

Ostrożnie z układami peryferyjnymi podłączonymi do układów programowalnych

Przykładowo, jeżeli do układu programowalnego podłączony jest inny układ na szynie IIC, wówczas trudne a czasami niemożliwe jest wykonać testy tego układu za pomocą XJTAG. Należy wówczas wyprowadzenia szyny IIC z tego układu wyprowadzić na zewnątrz płytki. Wówczas wykonanie testów będzie możliwe.

Wykorzystać w pełni inteligentne układy na płytce oraz zaawansowane właściwości XJTAG

Jeżeli na płytce znajdują się układy, których nie można przetestować należy umiejętnie wykorzystać możliwości XJTAG jakie daje tworzenie procedur testowych z wykorzystaniem języka wysokiego poziomu XJEase Language. Jeżeli wykonanie testu bezpośrednio jest nie możliwe, można wykorzystać inteligentne układy. Na przykład: mały program może być załadowany do pamięci Flash na płytce lub do układu CPLD, który będzie aktywował układ FPGA, CPLD lub mikrokontroler, a następnie wykona testy niedostępnych układów non-JTAG

Podsumowanie

Jak już wcześniej wspomniano powyższe wskazania powinny być traktowane jako sugestię, które znacznie ułatwią pracę testerom, powiększą zasięg testowalności a tym samym podniosą jakość produktu. Należy odpowiednio wykalkulować jakie procedury projektowania się bardziej opłacają, jaki jest różnica kosztów projektowanie z uwzględnieniem testów XJTAG a projektowania bez uwzględnienia. Procedury te wymagają ścisłej współpracy pionu projektowego z pinem testowym oraz serwisem. Istotne również jest aby klienci wiedzieli, że firma wykorzystuje zaawansowane techniki testujące. Znaczne podniesienie jakości produktu przy nieznacznym wzroście kosztów, podnosi prestiż firmy i zwiększa ilość zadowolonych klientów.

Firma QUANTUM jest wyłącznym dystrybutorem narzędzi XJTAG na Europe Środkowo Wschodnią.
Jeżeli jesteś zainteresowany systemem XJTAG skontaktuj się z nami:

QUANTUM Sp. z o. o.
Korporacja Transferu Technologii
ul. Wystawowa 1 | 51-618 Wrocław | Poland
Tel. (+48 71) 3626356
Fax (+48 71) 3626357
email: *info@quantum.com.pl*
web: *www.quantum.com.pl*

XJTAG
The Irwin Centre
Scotland Road | Dry Drayton | Cambridge | UK | CB23 8AR
Tel: +44 (0)1954 213888
Fax: +44 (0)1954 211565
email: *enquiries@xjtag.com*
web: *www.xjtag.com*