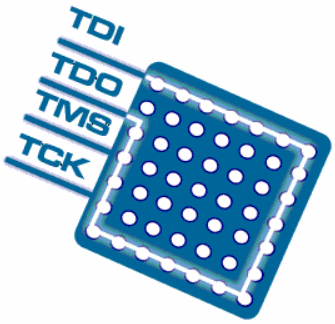
JTAG-Interface

Überblick über Aufbau, Funktion und Nutzung



Stephan Günther, Informationssystemtechnik, TU Dresden

Gliederung

- Einführung
- Aufbau und Funktionsweise
- Nutzung
- Einschätzung

Einführung

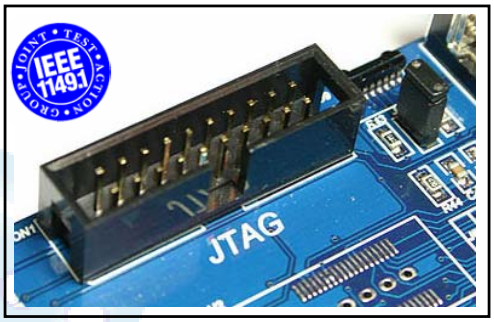
- Begrifsdefinition
- Begriffserklärung
- Motivation zur Einführung von JTAG

Begriffsdefinition

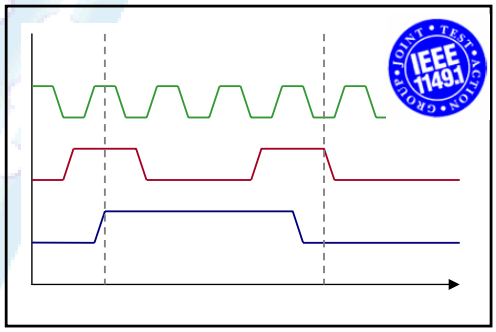
- **JTAG** bezeichnet den Standard IEEE 1149.1
- steht für „**J**oint **T**est **A**ction **G**roup“
- stellt Mittel zum Debugging von Hardware und zu deren Programmierung zur Verfügung
- ist auch unter dem Namen „Boundary Scan Test bekannt“

Begriffserklärung

- JTAG - Schnittstelle (mechanisch)

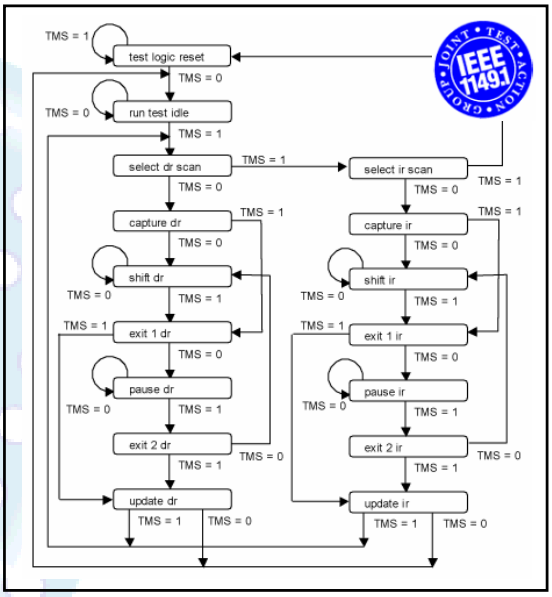


- JTAG - Schnittstelle (elektrisch)



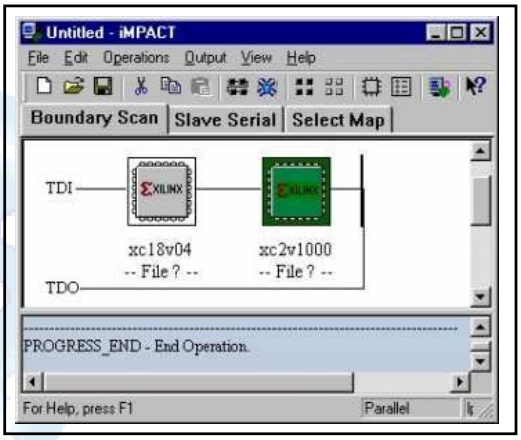
Begriffserklärung

- JTAG - Protokoll

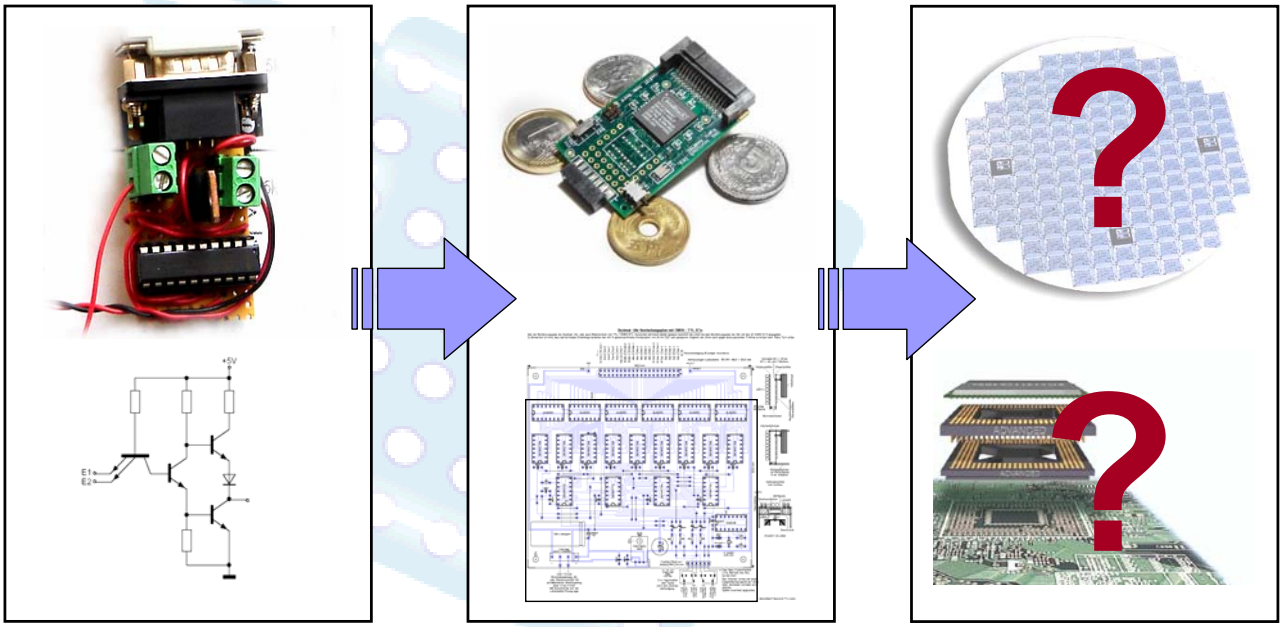


Begriffserklärung

- JTAG - Test
- JTAG - Programmierung

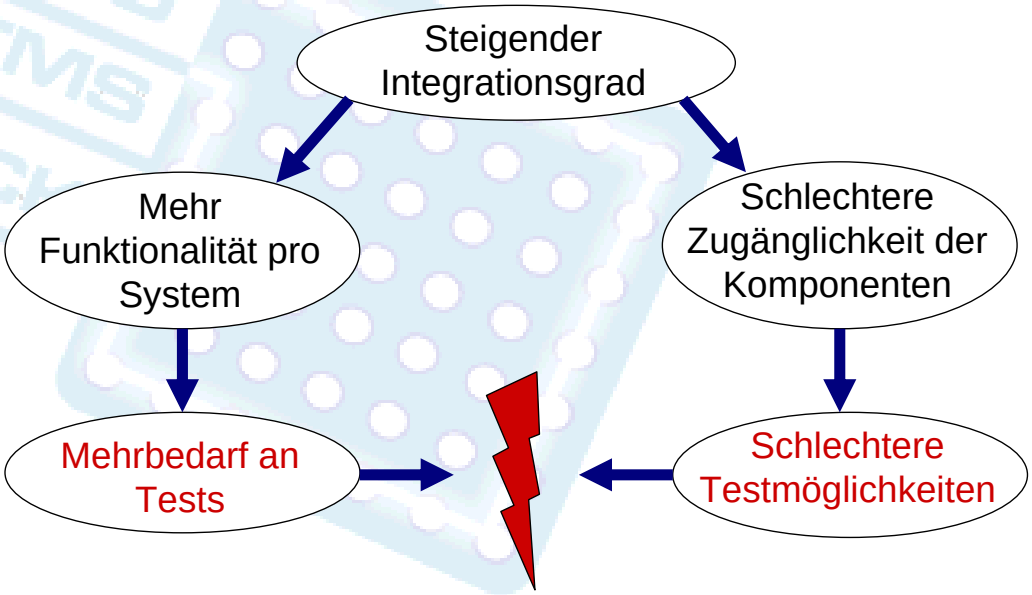


Motivation zur Einführung von JTAG



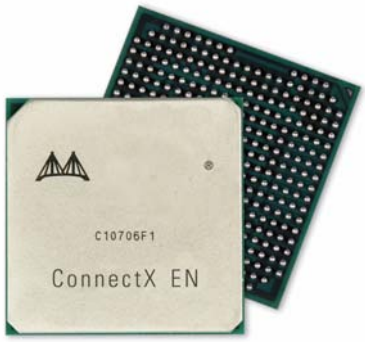
Motivation zur Einführung von JTAG

- Fortschreiten der technischen Entwicklung



Motivation zur Einführung von JTAG

- Beispiel: modernes BGA-Gehäuse mit ca. 400 Anschlüssen:
  - Bietet Platz für mehrere komplette Systeme
  - Kabelaufwand zum Test vergleichbar mit dem in einem Serverschrank
  - Filigrane Anschlussmechanik nötig
  - Gehäuseanschlüsse evtl.sogar unerreichbar
  - Wichtige interne Signale unzugänglich (Register,Schnittstellen, …)
  - Schlechte Austauschbarkeit und Überbrückbarkeit



Motivation zur Einführung von JTAG

- Kostenexplosion bei unzureichendem Test
  - Device Level 1 Kosteneinheit
  - Board level 10 Kosteneinheiten
  - System Level 100 Kosteneinheiten
  - Field Level 1000 Kosteneinheiten

Motivation zur Einführung von JTAG

- Lösungsmöglichkeit für diese Probleme:

  Der Mitte der 80er Jahre durch die Joint Test Action Group entwickelte und 1990 unter IEEE 1149.1 standardisierte Ansatz:
  - Einbau von virtuellen Testpunkten in die Hardware statt Nadelbetten auf Boards
  - Erreichbarkeit dieser Testpunkte über eine einheitliche Schnittstelle statt durch physische Verbindungen
  - Erweiterbarkeit des Protokolls um Nutzer- und Herstellerspezifische Befehle

Aufbau und Funktionsweise

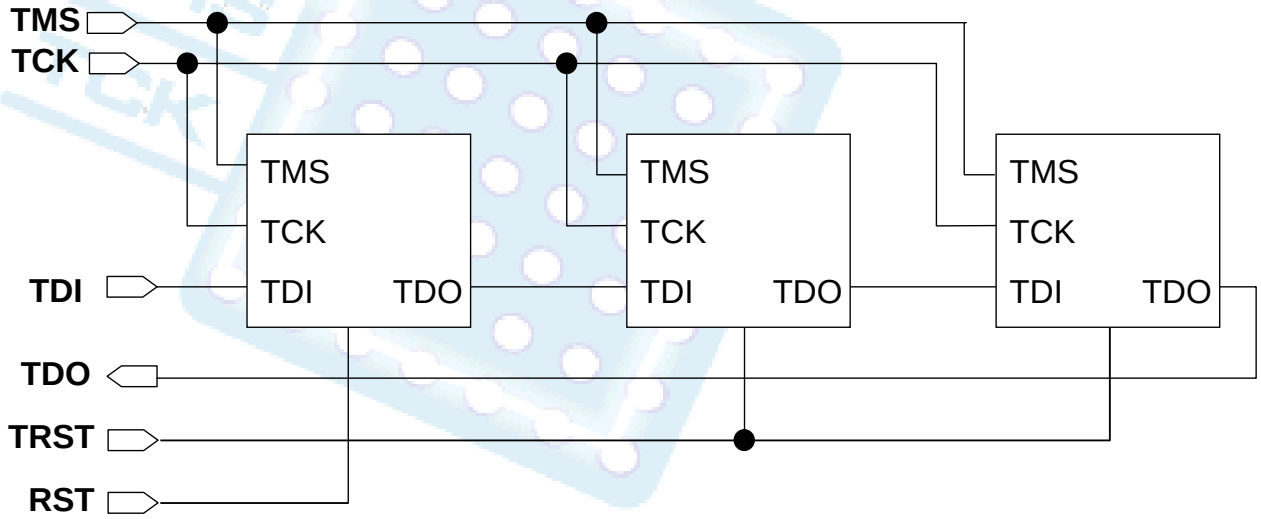
- Die elektrische und mechanische Schnittstelle
- Die Boundary-Scan-Zellen
- Aufbau eines JTAG-fähigen Schaltkreises
- JTAG - Befehle

Die elektrische und mechanische Schnittstelle

- Besteht aus 4 essentiellen Signalen
  - **TCK → Test Clock**
  - **TMS → Test Mode Select (Steuerung)**
  - **TDI → Test Data In**
  - **TDO → Test Data out**
- Weitere Signale
  - V_{ddh} → Versorgungsspannung (3,3V)
  - V_{SS} → Masse (0V)
  - $\overline{TRST}$ → Test Reset (Low Aktiv)
  - $\overline{RST}$ → Systemreset (Low Aktiv)

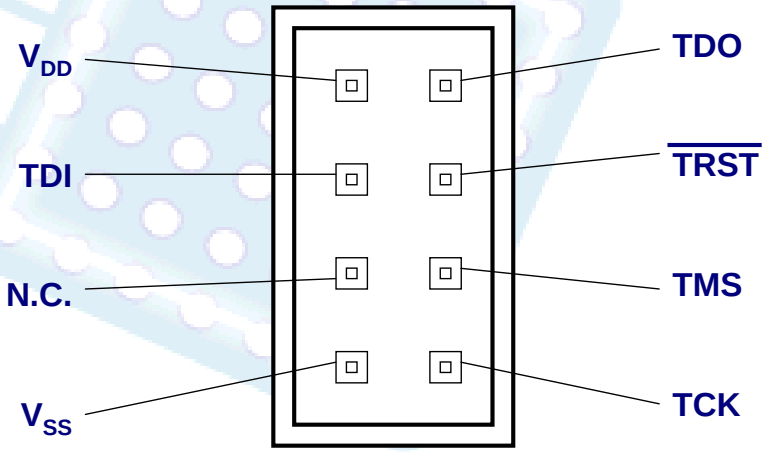
Die elektrische und mechanische Schnittstelle

- Anschluss der zu testenden Komponenten in einer Kette, der sogenannten „JTAG Chain“



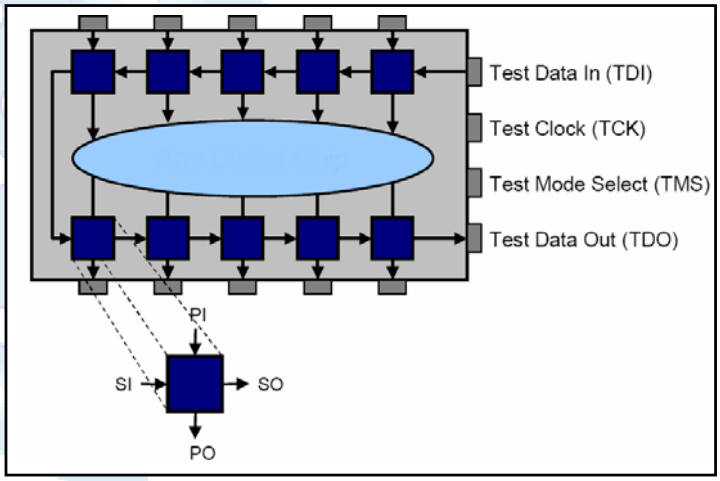
Die elektrische und mechanische Schnittstelle

- Viele verschiedene Steckerbauformen
- Unterschiedlichste Belegungen
- Beispiel (bei einigen PLD Boards benutzt):



Die Boundary-Scan-Zellen

- Zwischen den externen Anschlüssen und dem eigentlichen Schaltkreis werden Boundary-Scan-Zellen eingefügt
- So werden verschiedene Testoperationen möglich

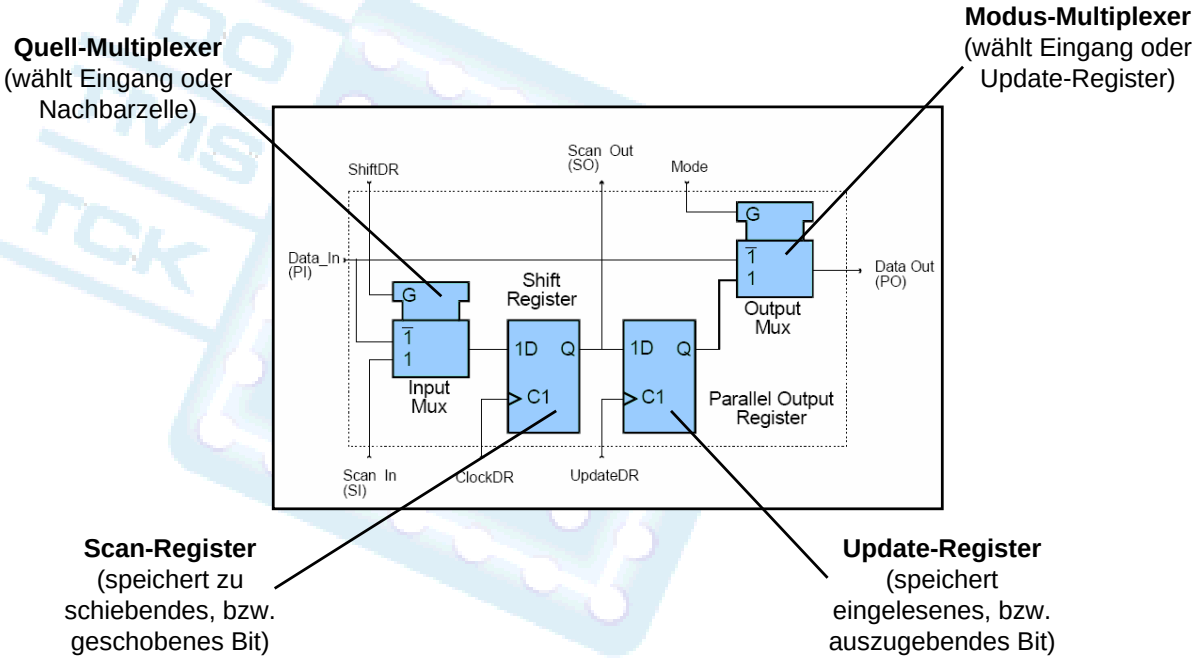


Die Boundary-Scan-Zellen

- Eine Boundary-Scan-Zelle kann - gesteuert durch Befehle von TMS – folgendes ausführen
  - **Capture (Preload)** → **anliegende Daten laden**
  - **Update (Unload)** → **geladene Daten ausgeben**
  - **Shift (Scan)** → **Daten zur nächsten Zelle**
  - **Normal** → **Transparentes Verhalten**

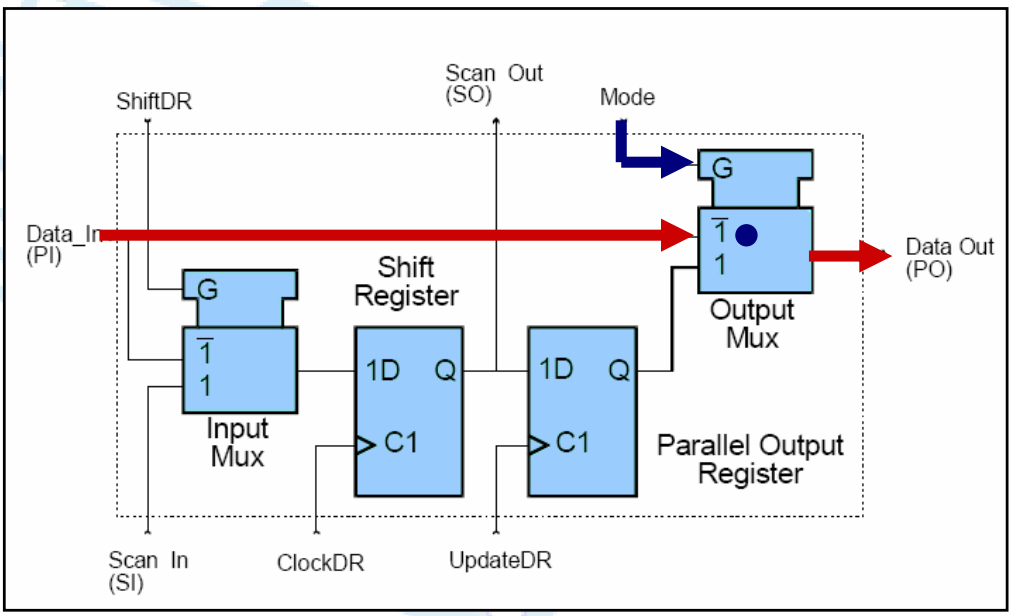
Die Boundary-Scan-Zellen

- **4 wesentliche Bestandteile**



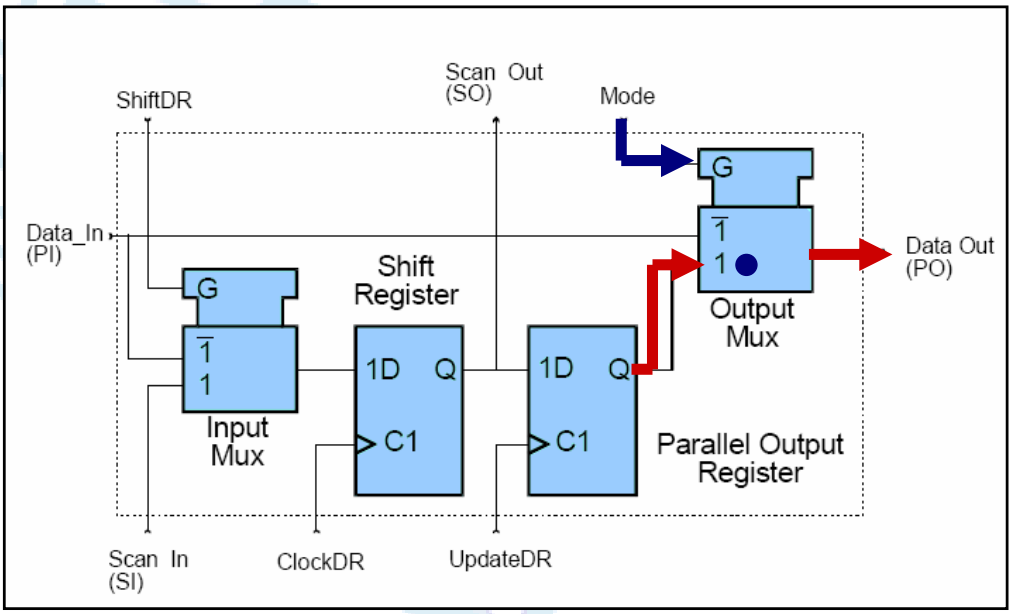
Die Boundary-Scan-Zellen

- **Modus: Normal**



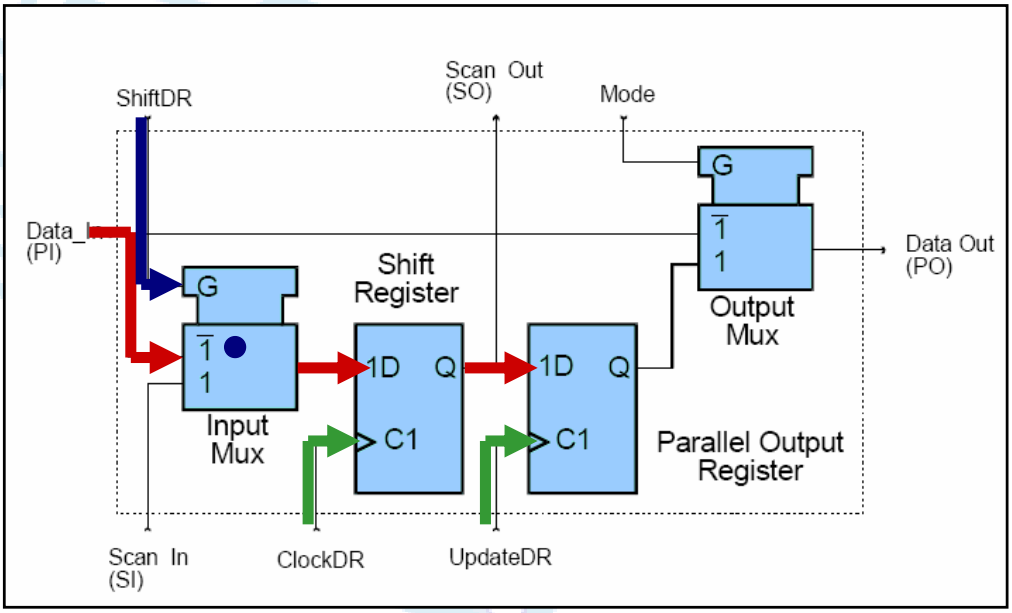
Die Boundary-Scan-Zellen

- **Modus: Update (Unload)**



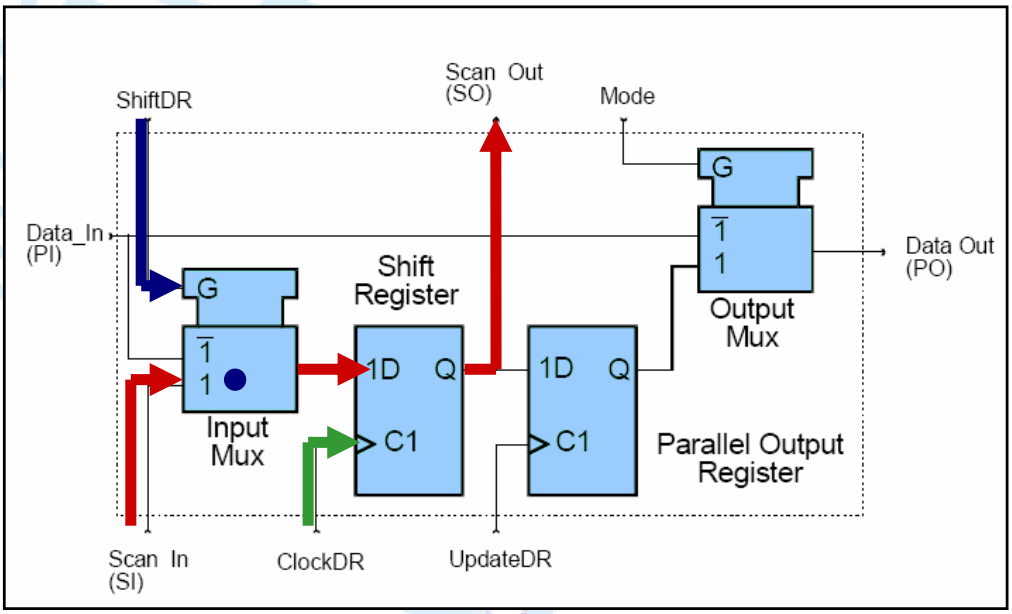
Die Boundary-Scan-Zellen

- **Modus: Capture (Preload)**



Die Boundary-Scan-Zellen

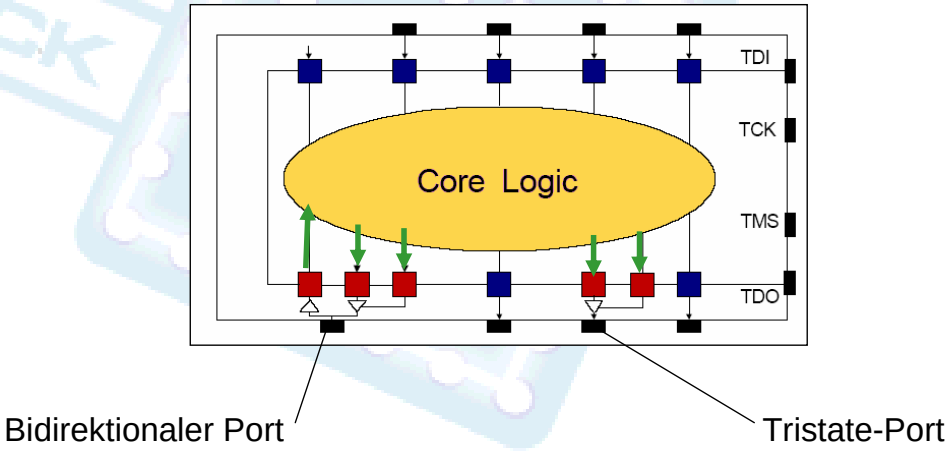
- **Modus: Scan (Shift)**



Die Boundary-Scan-Zellen

- **Tristate- und bidirektionale Ports:**
  - Boundary Scan-Zellen können nur **entweder** in Eingangs- **oder** in Ausgangsrichtung betrieben werden

  ➔ Erweiterung um zusätzliche Boundary Scan-Zellen



Aufbau eines JTAG-fähigen Schaltkreises

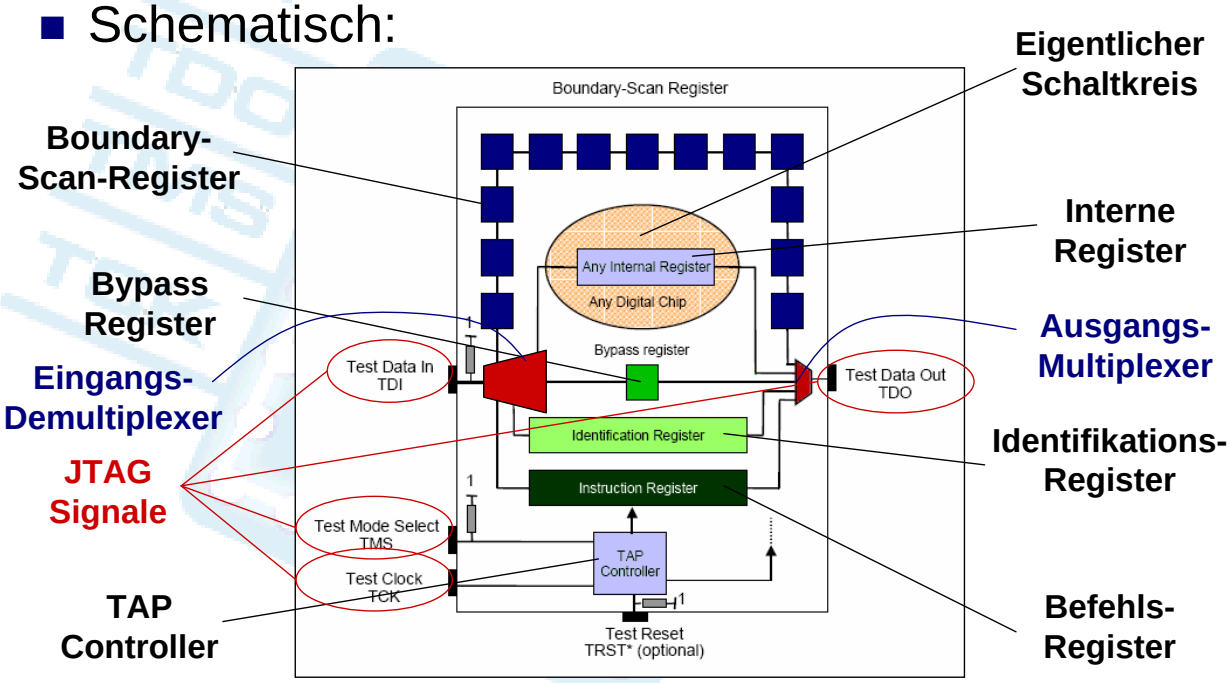
- **Ein Schaltkreis nach IEEE 1149.1 enthält**
  - Das eigentliche IC
  - Boundary-Scan-Zellen für jeden I/O-Pin
  - Ein ByPass Register zur Überbrückung des Bausteins in der JTAG-Kette
  - Eventuell Anschlüsse an interne Register
  - Zusätzlich mindestens 4 (dedizierte) JTAG-Pins
  - Ein Befehlsregister, das die letzte Anweisung sichert
  - Eventuell ein Identifikationsregister (Bausteinkennung)
  - Den sogenannten „**T**est **A**ccess **P**ort“ (**TAP**) Controller, welcher die einlaufenden TMS-Befehle verarbeitet

Aufbau eines JTAG-fähigen Schaltkreises

- Den Ausgangsmultiplexer, der vom dekodierten Befehl aus dem Befehlsregister gesteuert wird
- Den Eingangsdemultiplexer, der ebenfalls vom dekodierten Befehl aus dem Befehlsregister gesteuert wird

Aufbau eines JTAG-fähigen Schaltkreises

- Schematisch:



JTAG - Befehle

- **3 Anweisungen standardmäßig:**
  - Bypass
  - Sample / Preload
  - ExTest
- **Optionale Anweisungen*:**
  - InTest
  - IdCode
  - RunBIST
  - HighZ
  - …

* Müssen nicht implementiert werden, haben aber vorgeschriebenes Verhalten

JTAG - Befehle

- **Hersteller- und Anwender-spezifische Anweisungen*:**
  - **Programmieren eingebauter Speicher**
  - **Auslesen spezieller Register**
  - **...**

* Müssen nicht implementiert werden, haben kein vorgeschriebenes Verhalten

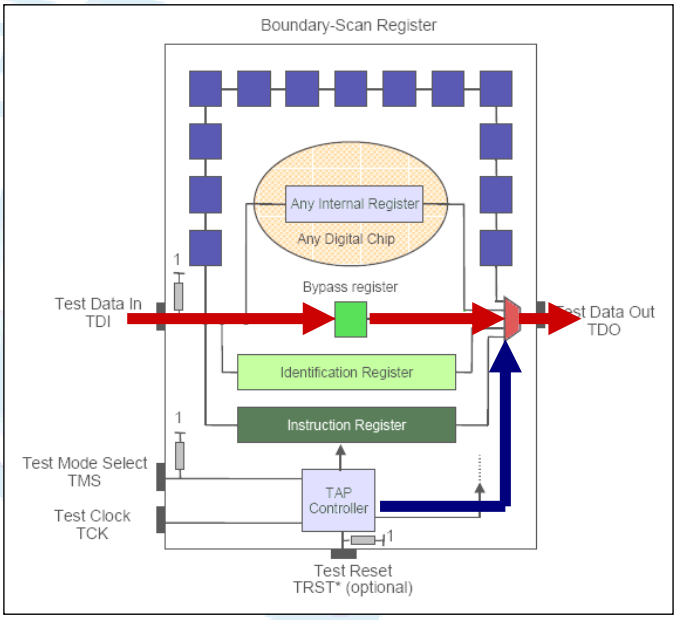
JTAG - Befehle

- **Bypass Anweisung**
  - Darf nur aus einer Folge logischer 1en bestehen (vgl. Folie 28: PullUp Widerstände!)
  - ByPass Register zwischen TDI und TDO
  - Initialer Zustand des Befehlsregisters falls kein Identifikationsregister vorhanden ist (dann IdCode Anweisung als Initialanweisung)

➔**Schaltkreis leitet JTAG-Daten nur weiter (mit Verzögerung um 1 Takt)**

JTAG - Befehle

- **Bypass Anweisung**



JTAG - Befehle

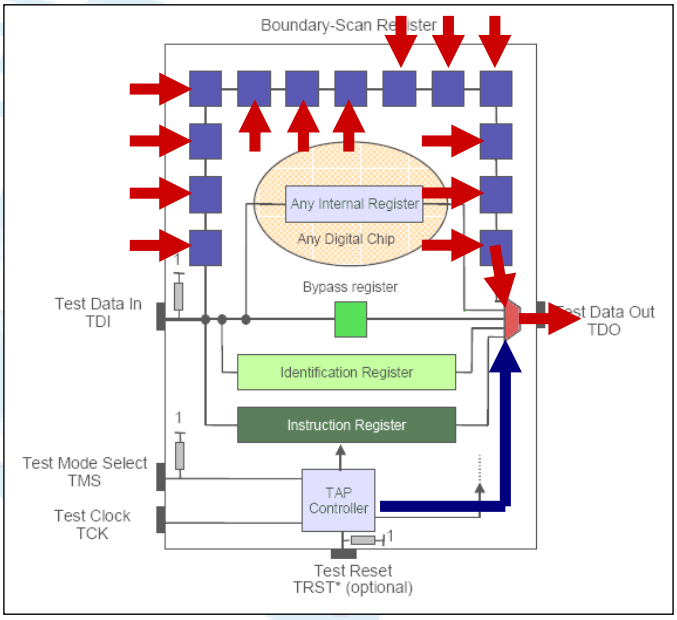
- **Sample / Preload Anweisung**
  - Wählt das Boundary-Scan-Register als Ausgang
  - Aktiviert das Einlesen von Daten in die Boundary-Scan-Zellen
  - Kein Befehlscode definiert

➔ Von Eingängen werden Eingangswerte gelesen

➔ Für Ausgänge werden Werte vorgeladen

JTAG - Befehle

- **Sample / Preload Anweisung**



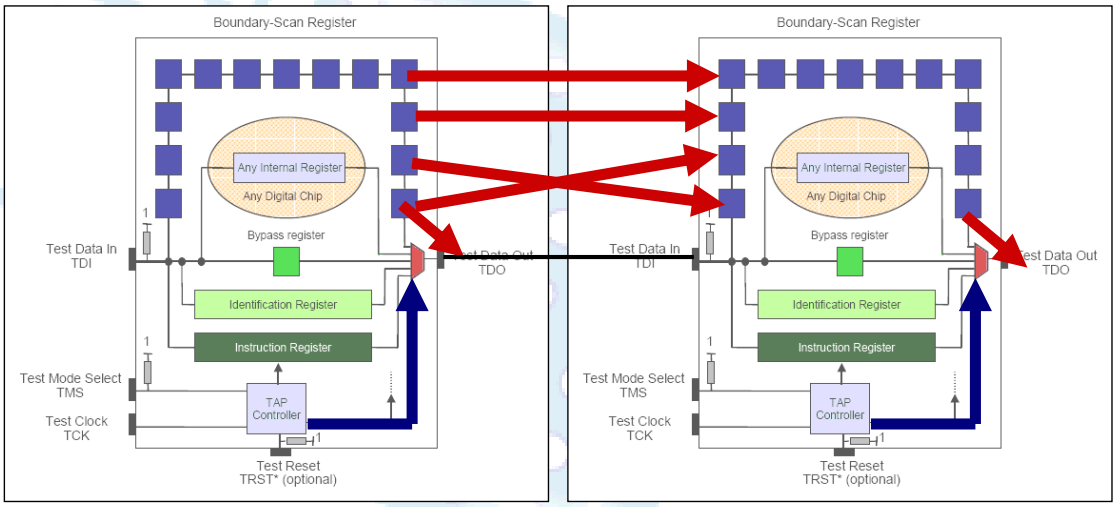
JTAG - Befehle

- **ExTest Anweisung**
  - Wählt das Boundary Scan-Register als Ausgang
  - Bereitet die Zellen auf einen Externen Test vor
  - Befehlscode ausschließlich logische 0en

➔ Prüfung der Verbindungen zwischen den Bausteinen

JTAG - Befehle

- **ExTest Anweisung**

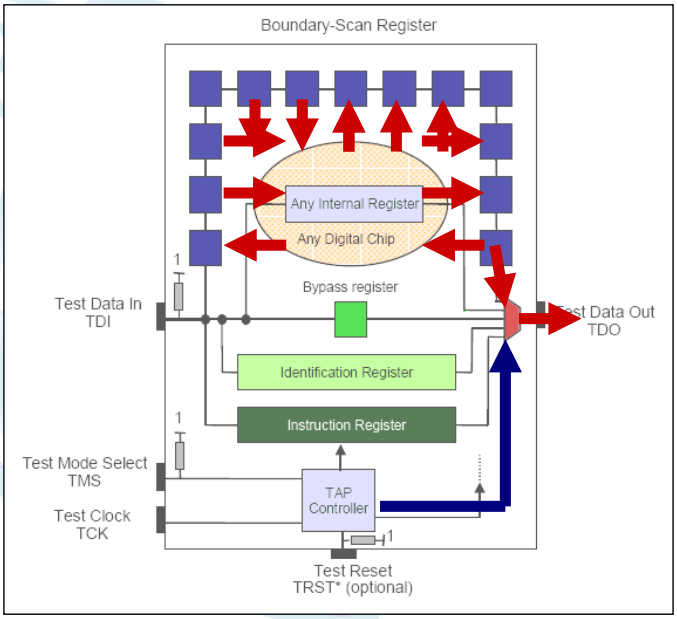


JTAG - Befehle

- **InTest Anweisung**
  - Wählt das Boundary Scan- Register als Ausgang
  - Bereitet die Zellen auf einen Internen Test vor
  - ➔ Test des Verhaltens des Schaltkreises
  - ➔ Zum Beispiel Emulation eines kompletten Boards noch bevor dieses existiert
  - ➔ Oder Prüfen auf Fehler in den Ausgangstreibern

JTAG - Befehle

- **InTest Anweisung**



JTAG - Befehle

- **IdCode Anweisung**
  - Wählt das Identifikations Register als Ausgang

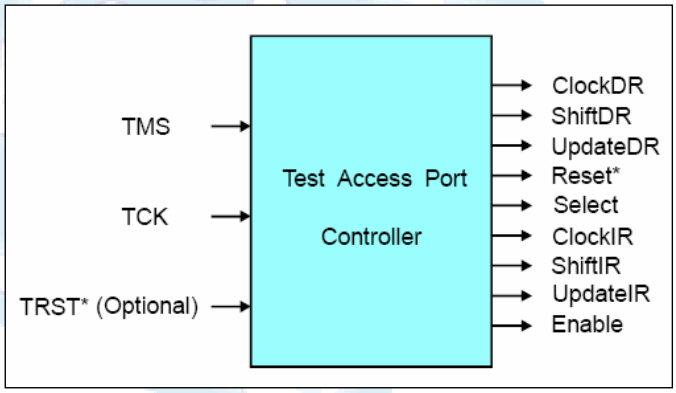
  ➔ Bestimmung der Bausteine in der JTAG Kette
- **RunBIST**
  - Startet Selbsttest und legt Ergebnis an Pass Fail Register an

  ➔ Ausschluss von Fehlern 2. Art
- **HighZ Anweisung**
  - Alle Ausgänge werden hochohmig geschaltet
  - Auswahl des ByPass Registers

  ➔ Absicherung gegen „verbotene“ Verbindungen

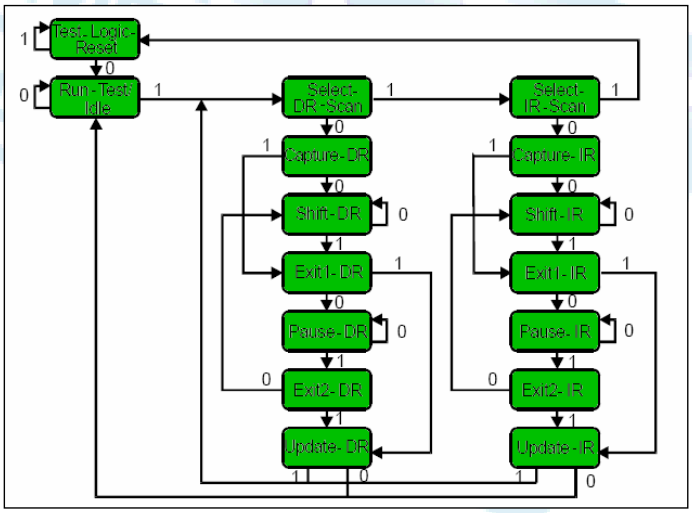
JTAG - Befehle

- **TAP-Controller:**
  - **Steuert das Verhalten der Boundary-Scan-Register und das aller anderen**



JTAG - Befehle

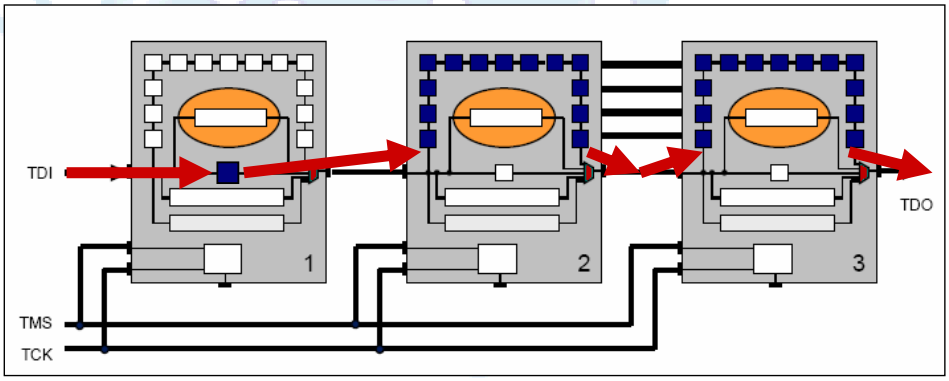
- **TAP – Zustandsautomat:**
  - **Darstellung für das Verhalten das TAP-Controllers in Abhängigkeit der eingehenden TMS-Daten**



- ➔ **Initialzustand, bzw. Zustand nach TRST ist Test-Logic-Reset**
- ➔ **Ab 5 logischen 1en wieder Ausgangs-zustand**

JTAG - Befehle

- **Beispiel 1**
  - 3 Bausteine in der JTAG Kette
  - 1. Baustein soll nur durchleiten
  - 2. und 3. Baustein sollen auf ihre externe Verbindung miteinander überprüft werden

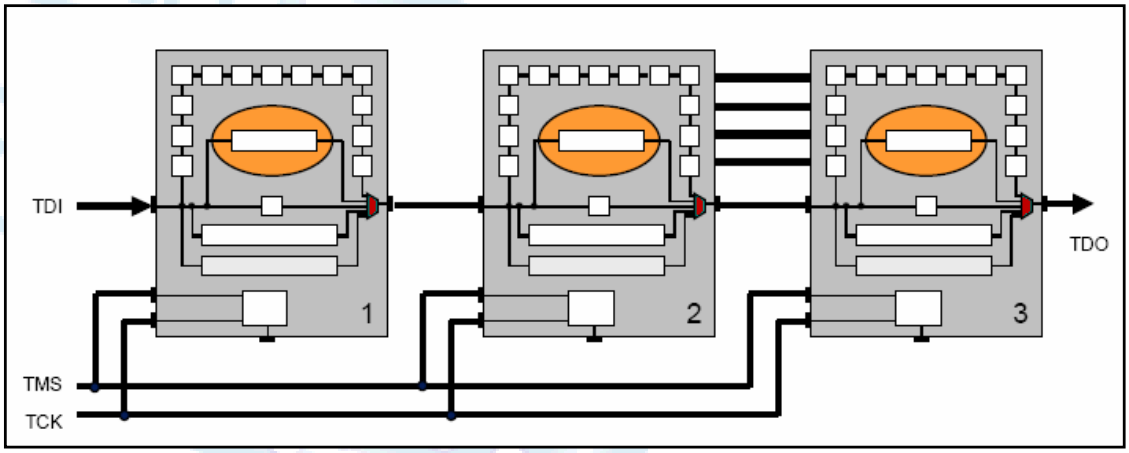


JTAG - Befehle

- **Ablauf:**
  - Befehl über TMS: Befehlsregister mit TDI und TDO verbinden (alle Bausteine!)
  - Befehl über TDI: 2 mal 0000… und 1 mal 1111… (bei 2 Bit Befehlsregister: 11 00 00)
  - Befehl über TMS: Befehl anwenden (Shift to Hold)
    - ➔ Baustein 1 deselektiert das Befehlsregister und schaltet dafür auf das ByPass-Register um
    - ➔ Baustein 2 und 3 deselektieren das Befehlsregister und schalten dafür auf das Boundary-Scan-Register um

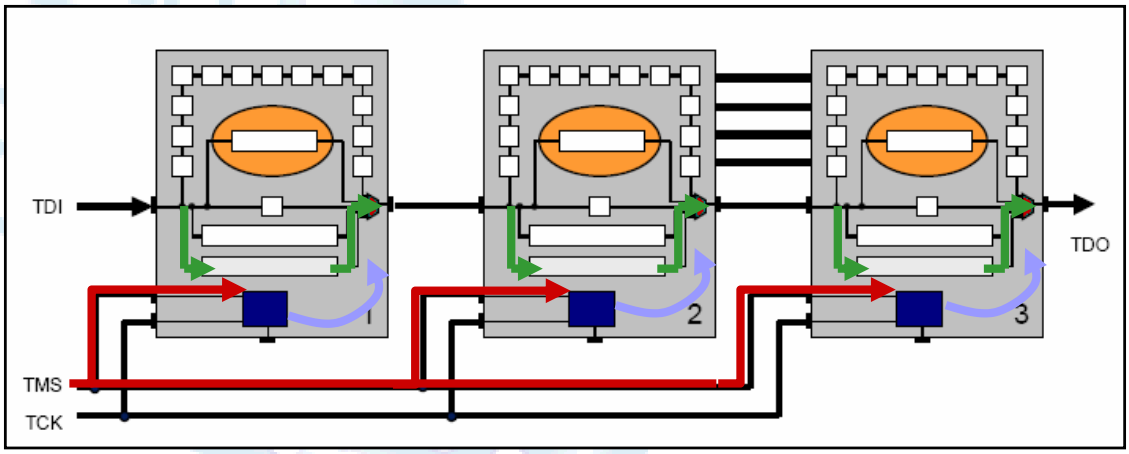
JTAG - Befehle

- **Grundzustand**



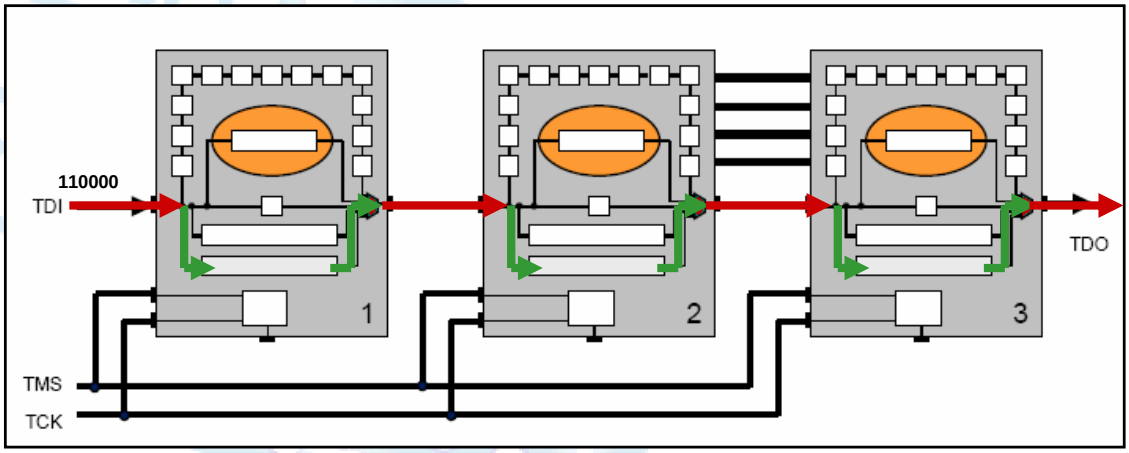
JTAG - Befehle

- **Schritt 1**



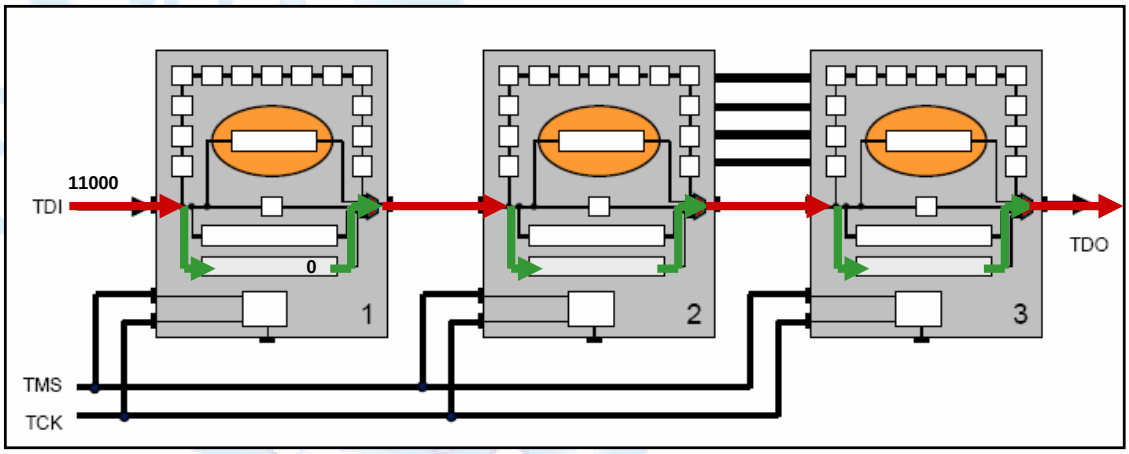
JTAG - Befehle

- Schritt 2



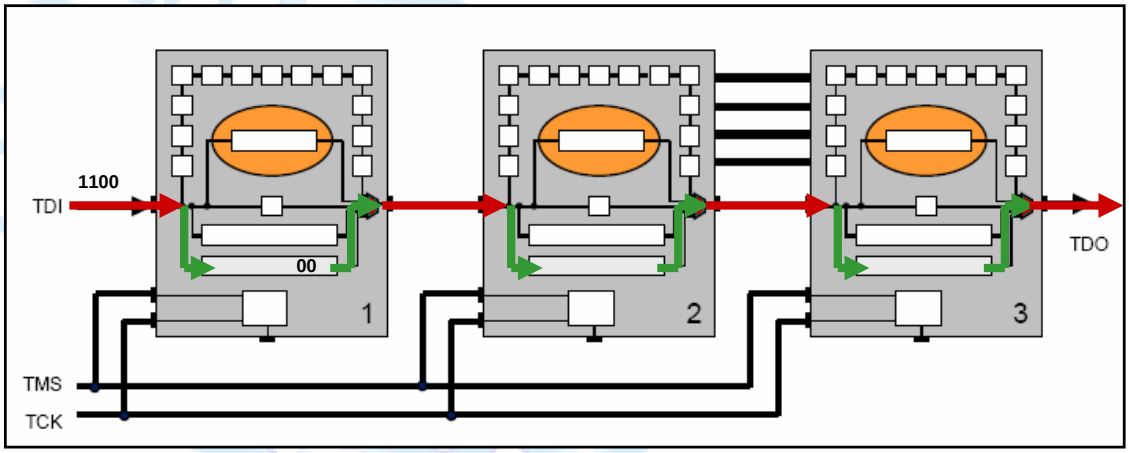
JTAG - Befehle

- Schritt 2



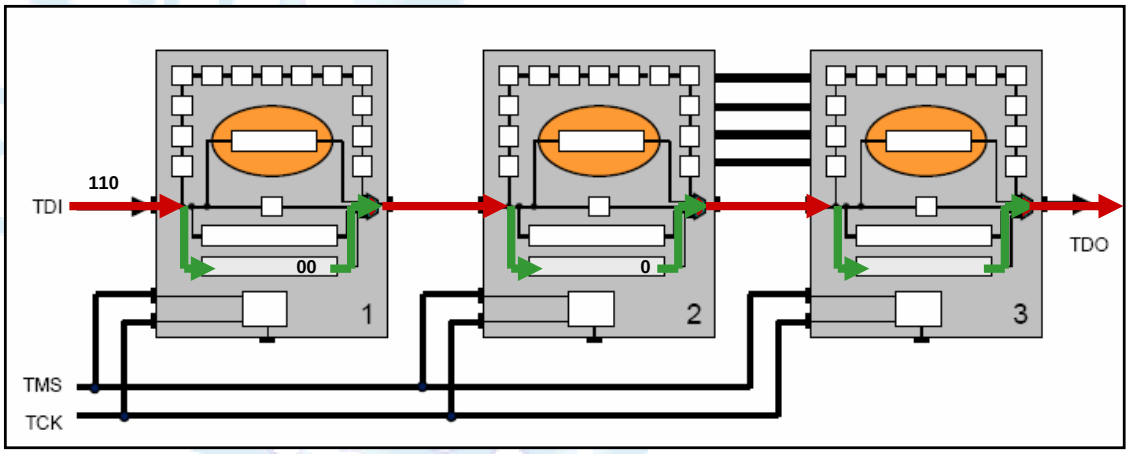
JTAG - Befehle

- Schritt 2



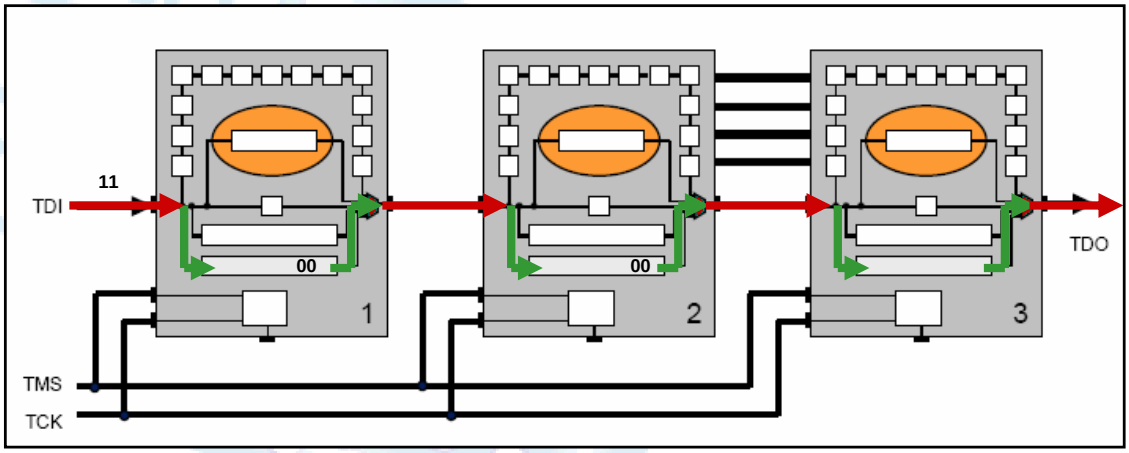
JTAG - Befehle

- Schritt 2



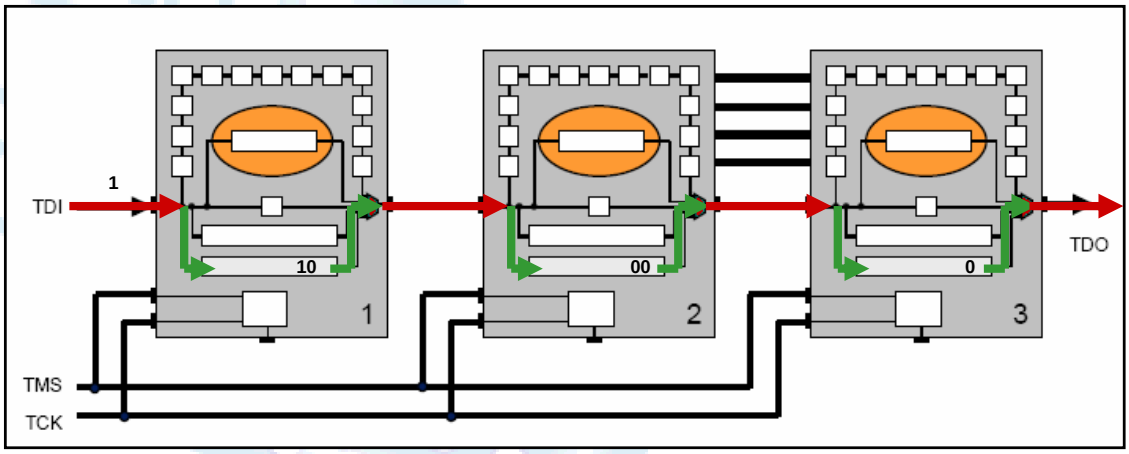
JTAG - Befehle

- Schritt 2



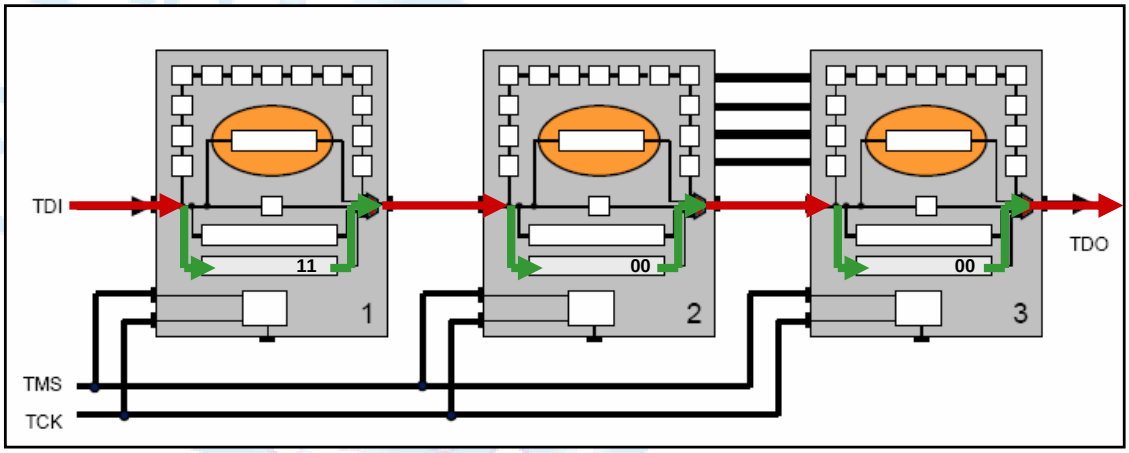
JTAG - Befehle

- Schritt 2



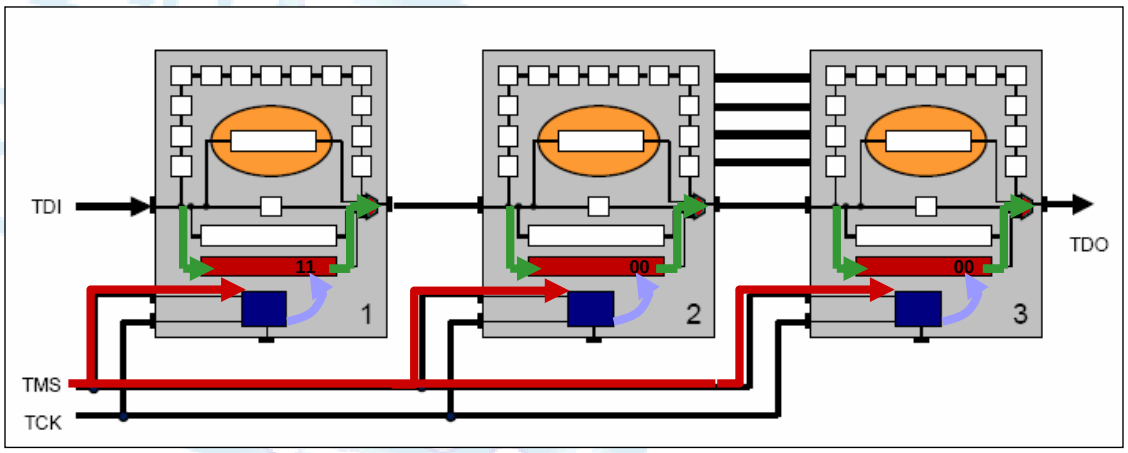
JTAG - Befehle

- Schritt 2



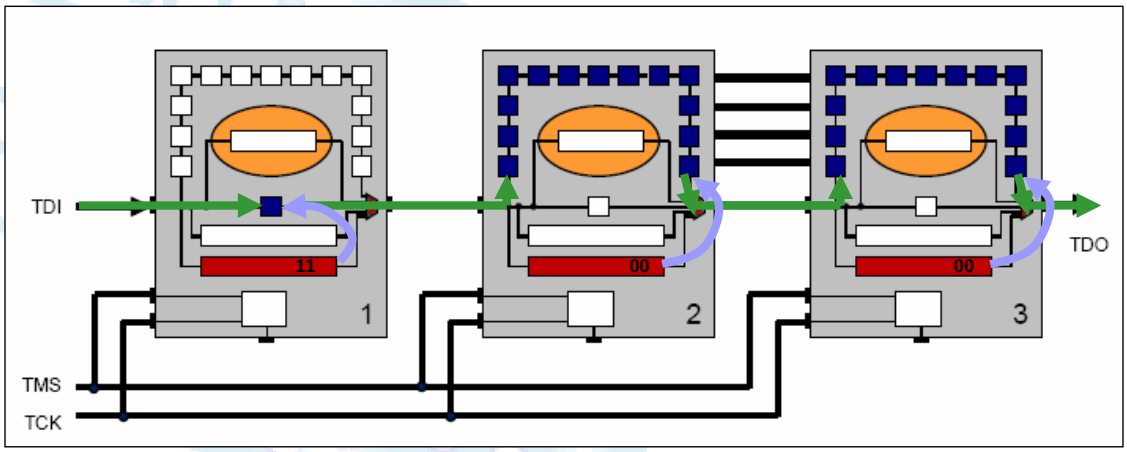
JTAG - Befehle

- Schritt 3



JTAG - Befehle

- Schritt 3



JTAG - Befehle

- Anschließend Prüfung auf Unterbrechung, Kurzschluss, usw. zwischen Bauteil 2 und 3 mit entsprechenden Testmustern

➔ Was passiert aber, wenn das Testinterface nicht ordnungsgemäß funktioniert?

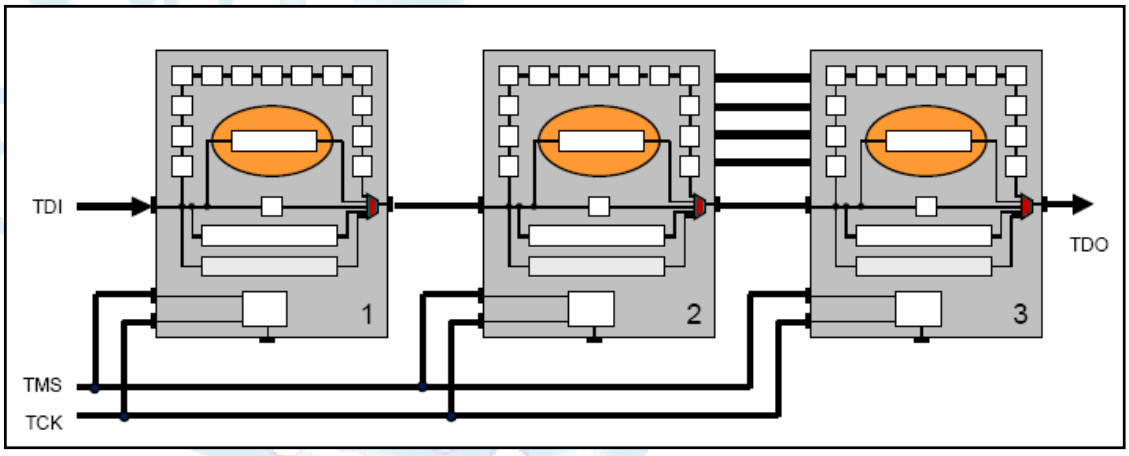
➔ Wie kann man das ausschließen?

JTAG - Befehle

- **Beispiel 2:**
  - 3 Bausteine in der JTAG-Kette
  - Prüfung, ob das Testinterface korrekt verbunden ist und ordnungsgemäß funktioniert
- **Ablauf:**
  - Wahl des Befehls-Registers über TMS
  - Ausführen von Capture-DR über TMS
  - Auslesen über TDO

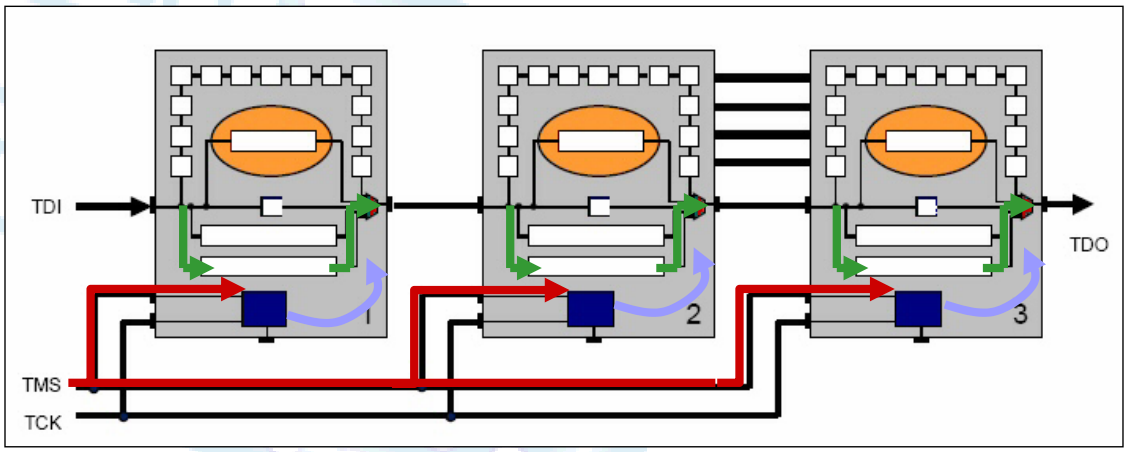
JTAG - Befehle

- **Grundzustand**



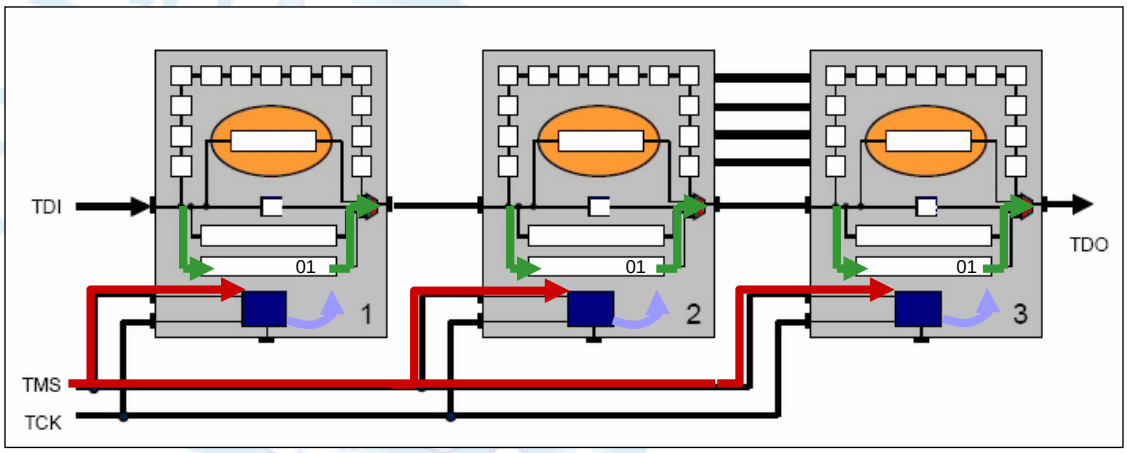
JTAG - Befehle

- Schritt 1



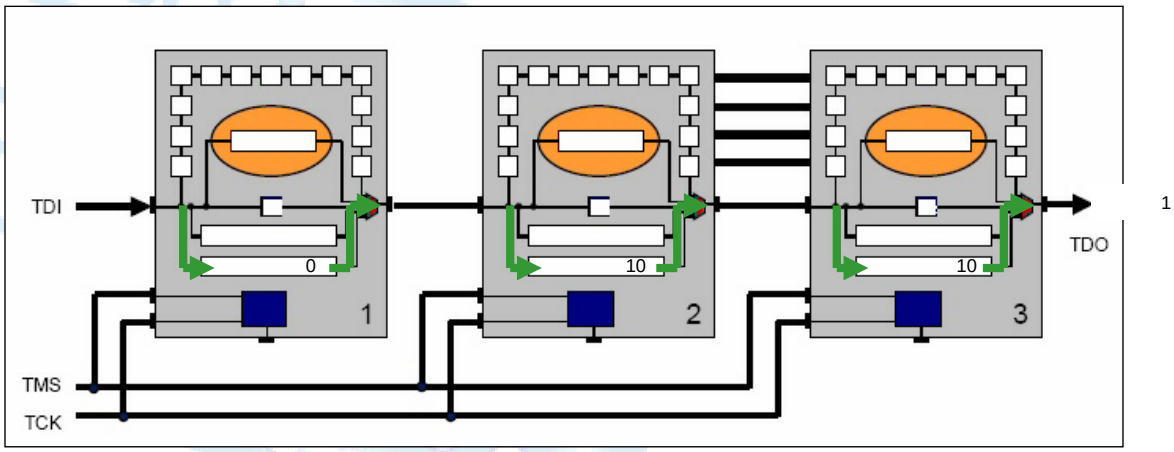
JTAG - Befehle

- Schritt 2



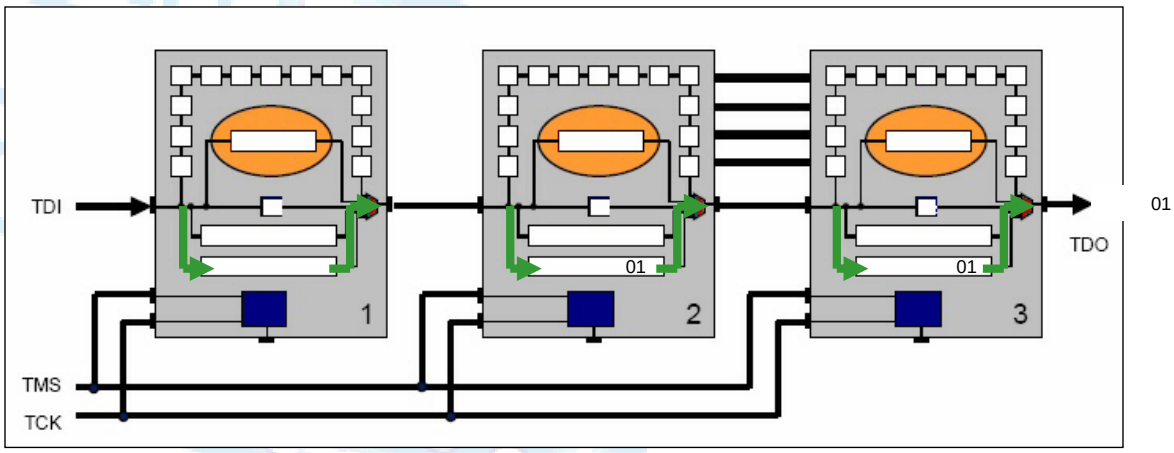
JTAG - Befehle

- Schritt 3



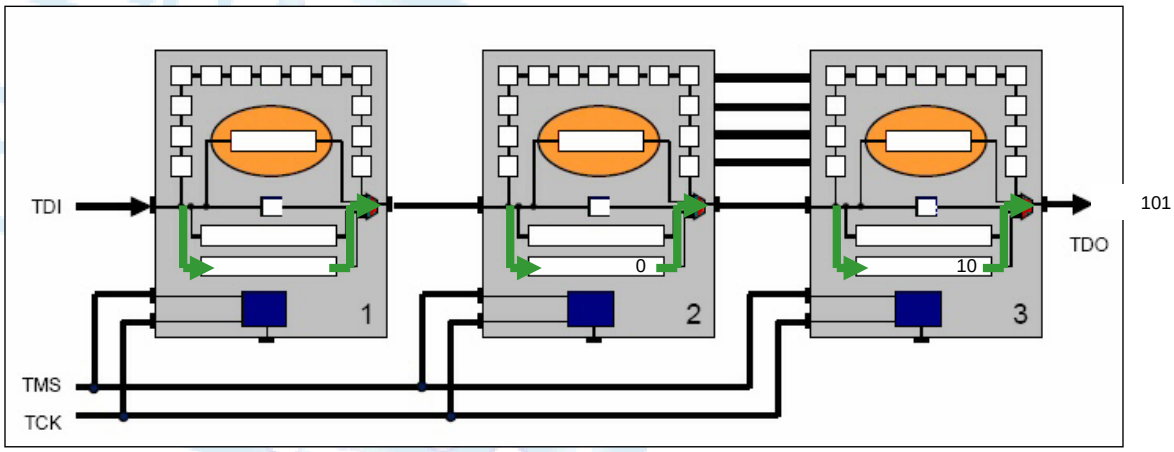
JTAG - Befehle

- Schritt 3



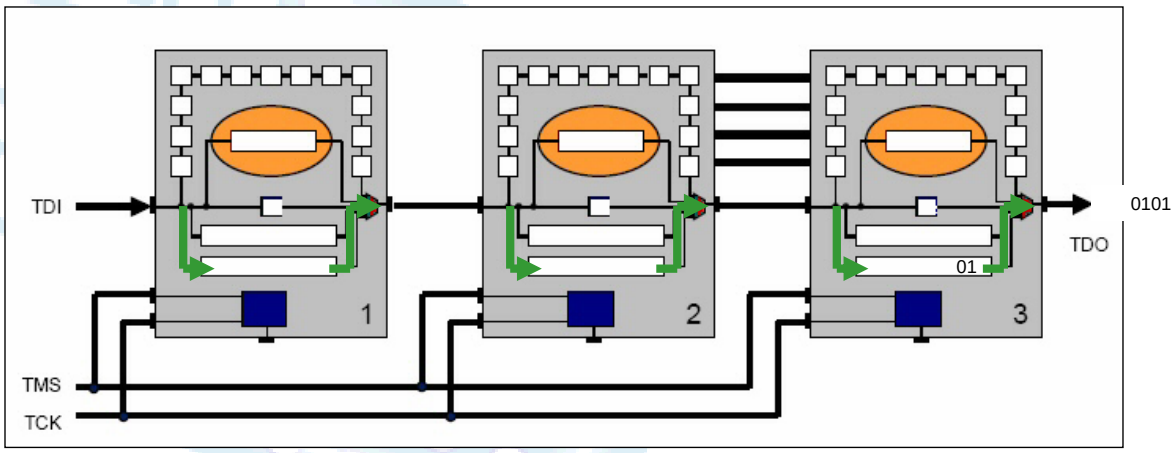
JTAG - Befehle

- Schritt 3



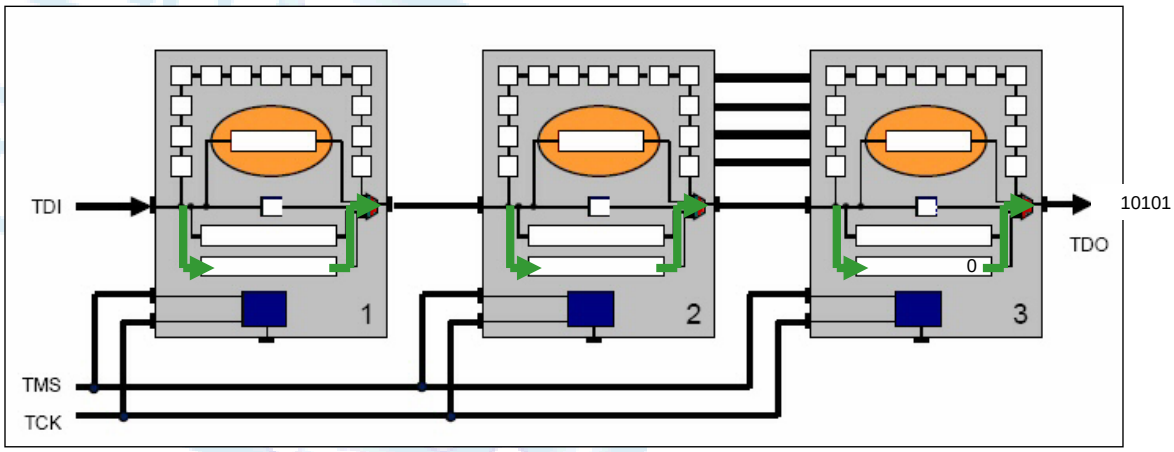
JTAG - Befehle

- Schritt 3



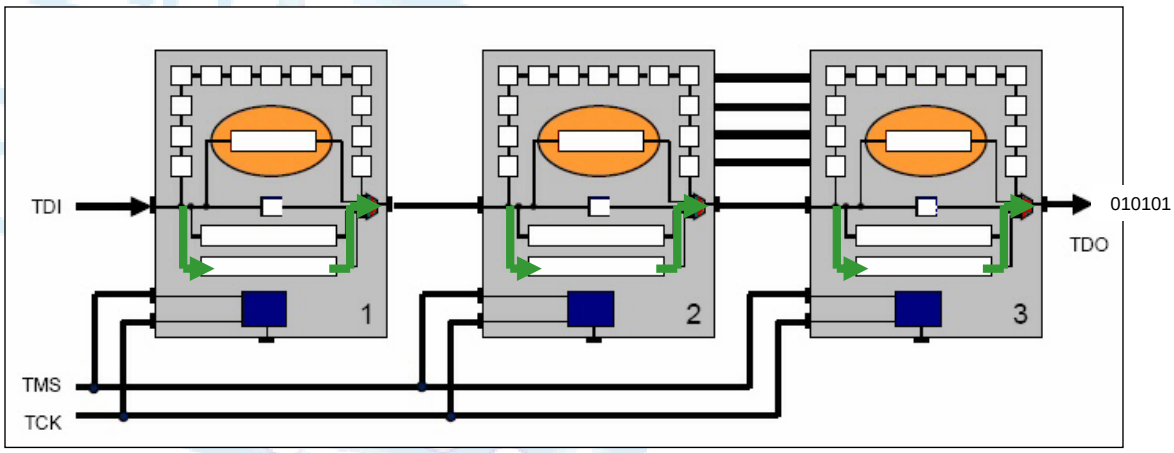
JTAG - Befehle

- Schritt 3



JTAG - Befehle

- Schritt 3



JTAG - Befehle

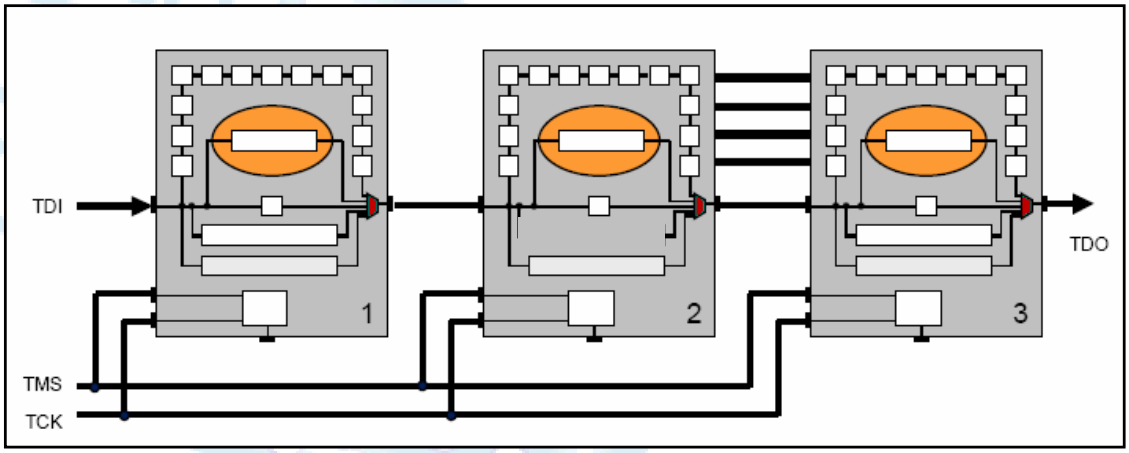
- **Wenn das Muster 010101 an TDO erscheint, ist folgendes sicher:**
  - **TMS ist richtig am Board und an jedem Bauelement angeschlossen**
  - **TCK ist richtig am Board und an jedem Bauelement angeschlossen**
  - **TDO und TDI sind korrekt mit dem Nachbarbau-element, bzw. JTAG-Stecker verbunden**
  - **Jeder interne TAP-Controller reagiert richtig (zumindest auf den CAPTURE-IR-Befehl)**
- **Schickt man noch 10 über TDI nach, weiß man:**
  - **Kein Baustein hat einen TDO-TDI Kurzschluss**

JTAG - Befehle

- **Beispiel 3:**
  - Identifikation einer beliebigen Anzahl unbekannter Systemkomponenten nach einem Fehlerfall
    - Wie viele?
    - Welche haben Identifikations-Code?
    - Wie lautet dieser jeweils?
- **Ablauf:**
  - **Einschalten oder TRST**
  - **TMS: Select-DR-Scan State (01)**
  - **TMS: Capture-DR (0)**
  - **TMS: Shift-DR (000000….)**

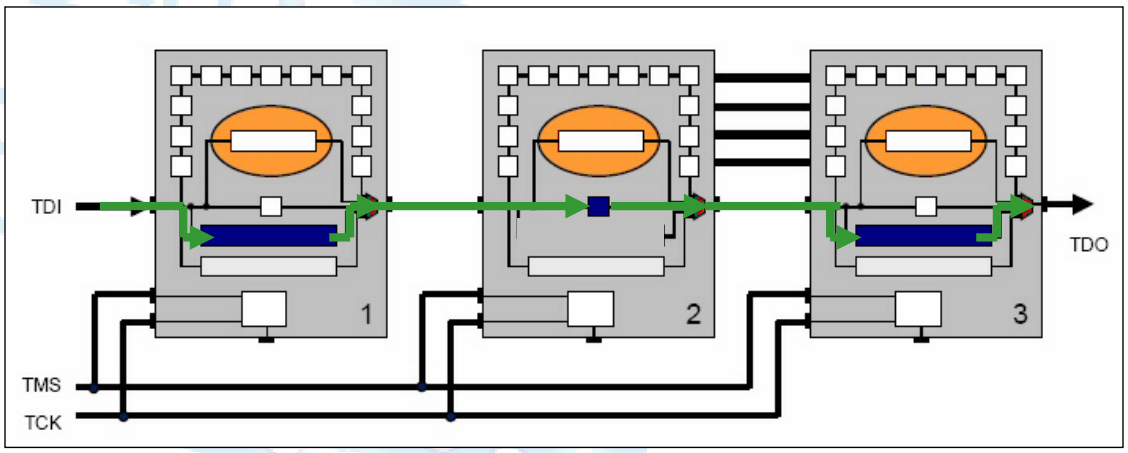
JTAG - Befehle

- **Grundzustand**



JTAG - Befehle

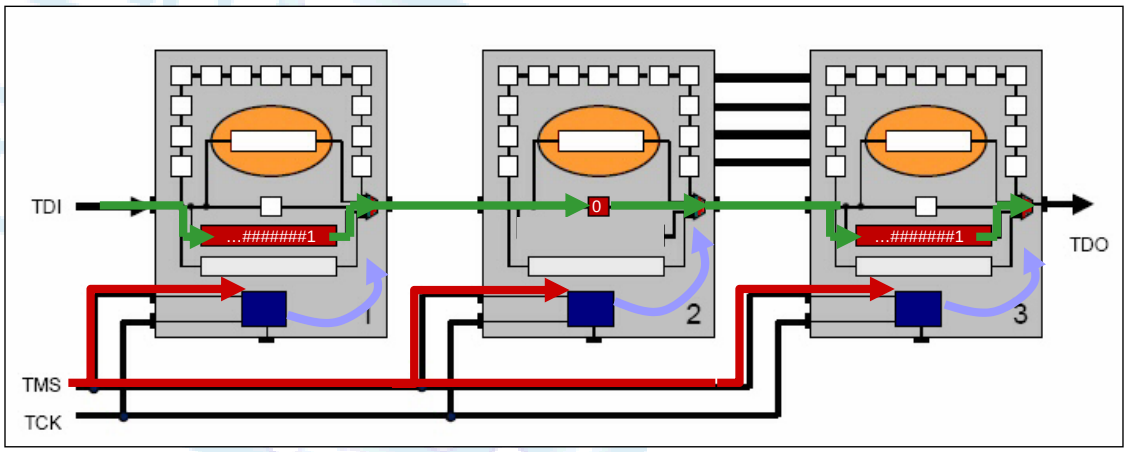
- **Schritt 1**



➔ Komponenten ohne ID starten mit ByPass Register

JTAG - Befehle

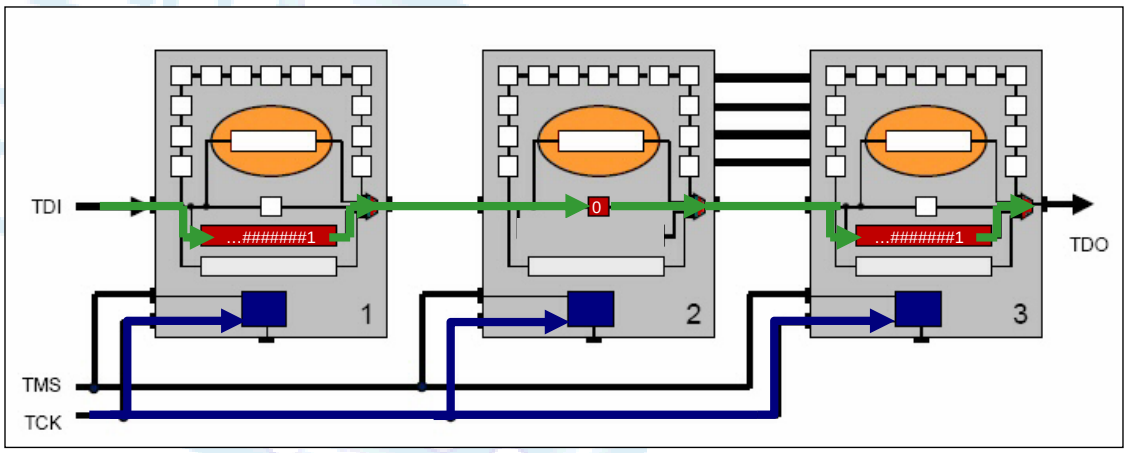
- **Schritt 2**



➔ ByPass Register werden mit 0 geladen, Identifikations Register mit 32 Bit ID, LSB ist 1 (Definition)

JTAG - Befehle

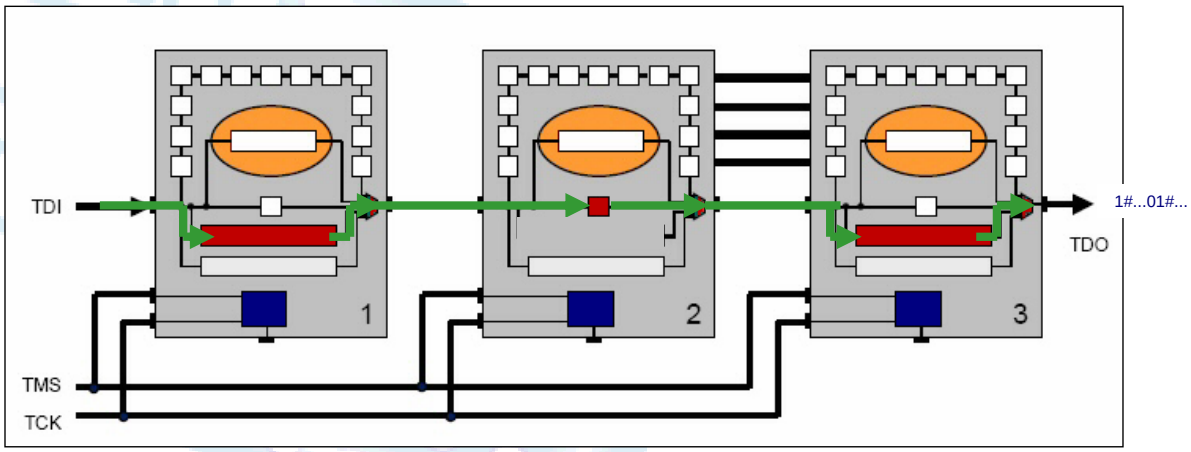
- **Schritt 3**



➔ Herausschieben der Daten aus TDO

JTAG - Befehle

- **Schritt 3**



➔ Herausschieben der Daten aus TDO

JTAG - Befehle

- **Am Ausgang Erscheint folgendes Muster:**
  - 1####.....01####..... (zeitliche Abfolge)
- **Dabei bedeuten während des Schiebens:**
  - **Führende 1: nächste 31 Bit sind ID eines ICs**
  - **Führende 0: IC vorhanden, aber keine ID**
- **Somit wird folgendes erkannt:**

1 0 ### #### #### #### 1

ID Marker (IC 3)
ID des 3. ICs in der Kette
No ID Marker (IC 2)
No ID Marker (IC 1)
ID des 1. ICs in der Kette

JTAG - Befehle

- Mit diesen Informationen können
  - Die Anzahl der Bausteine
  - Der Typ der Bausteine mit ID

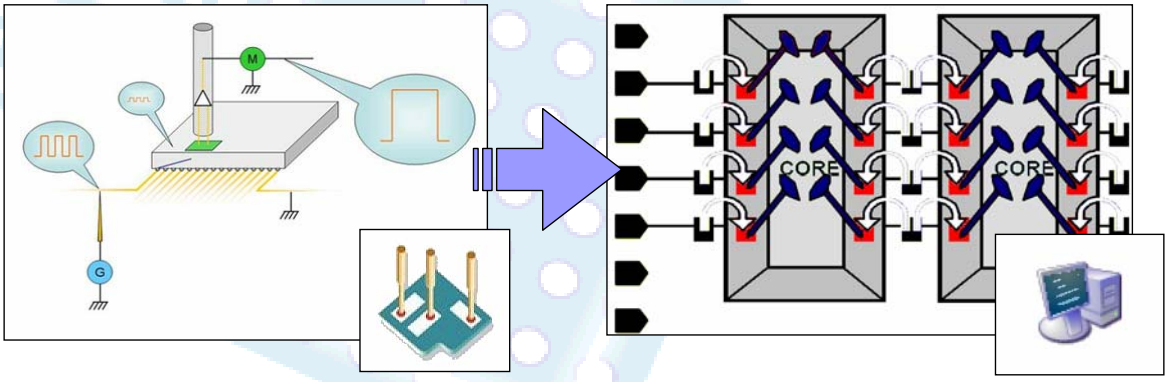
  ermittelt werden

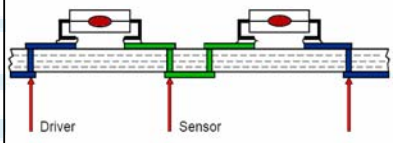
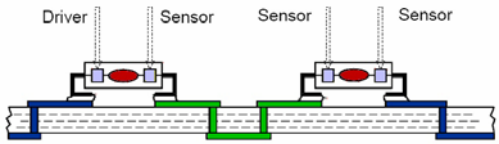
Nutzung

- Ersatz für klassische Testmethoden
- Erweiterung klassischer Testmethoden
- Zusammenschaltung mit nicht JTAG-fähigen Geräten
- Anwendungsbeispiele

Ersatz für klassische Testmethoden

- Ziel war: Umstellung vom „Bed-Of-Nails“ Verfahren auf „virtuelle Nägel“



Fehler	Bed of Nails 	Boundary Scan 
Nagel/Scan-Zelle	OK	OK
Durchkontaktierung	OK	OK
Leiterbahn	OK	OK
Lötstelle	OK	OK
Pin	OK	OK
Bond	OK	OK
Treiber		OK
Schaltkreis	OK	OK

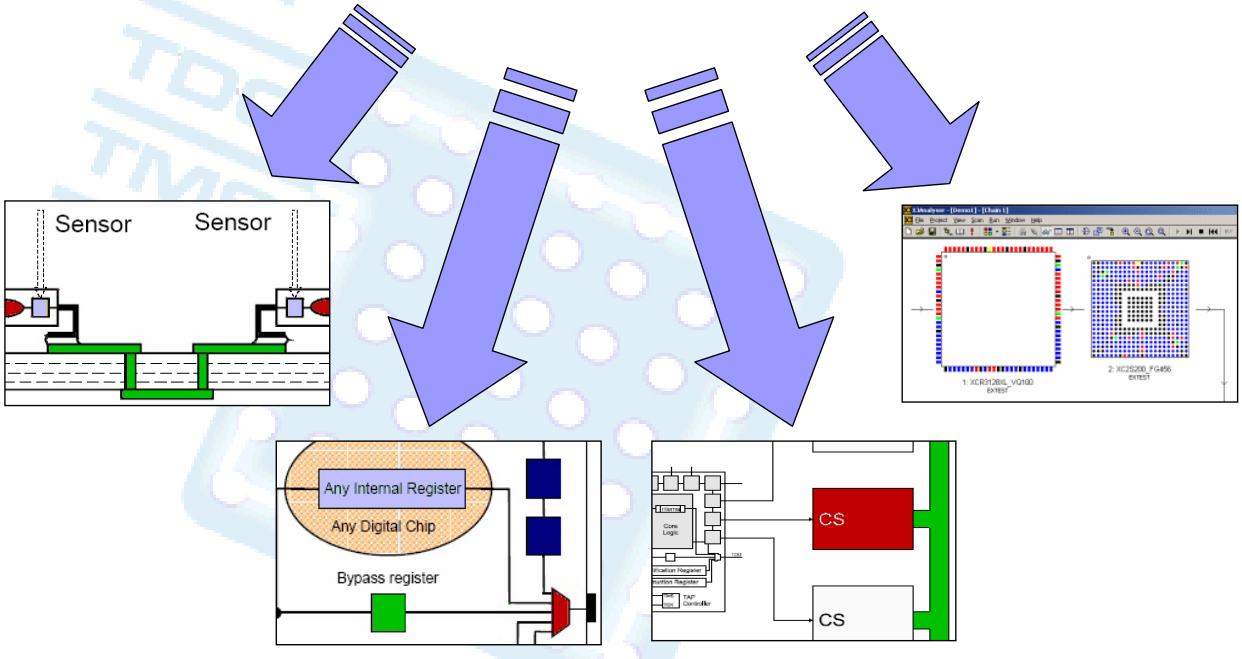
Ersatz für klassische Testmethoden

- Entwicklung
  - Test einzelner Baugruppen
  - Emulation der Umgebung
- Fertigungskontrolle
  - Test von Baugruppen
  - Test von Verbindungen
- Service
  - Erkennung der Systemversion und Systemkonfiguration

Erweiterung klassischer Testmethoden

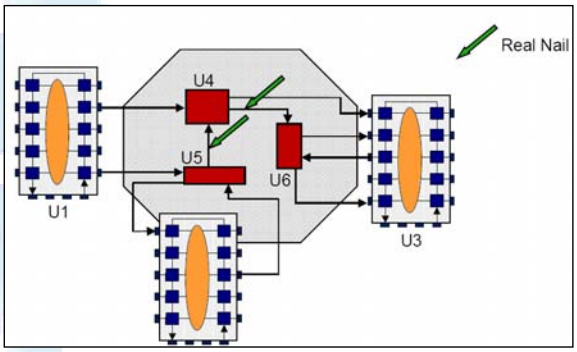
- Durch Integration der Scan-Zellen auf dem Schaltkreis und Freiheitsgrade im JTAG-Standard:
  - Testen der Ausgangstreiber
  - Schreiben und Lesen von Systemregistern
  - Programmierung und Auslesen interner Speicher
  - Unabhängigkeit von der Gehäusebauform

Erweiterung klassischer Testmethoden



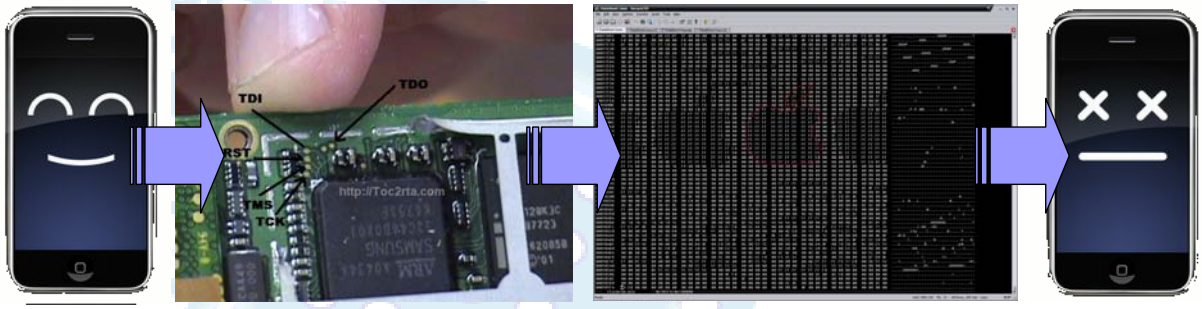
Zusammenschaltung mit nicht JTAG-fähigen Geräten

- Möglichkeit 1
  - Verhalten vollständig über JTAG-Bausteine testbar
- Möglichkeit 2
  - Kombination von JTAG mit dem klassischen Nagelbett-testverfahren



Anwendungsbeispiele

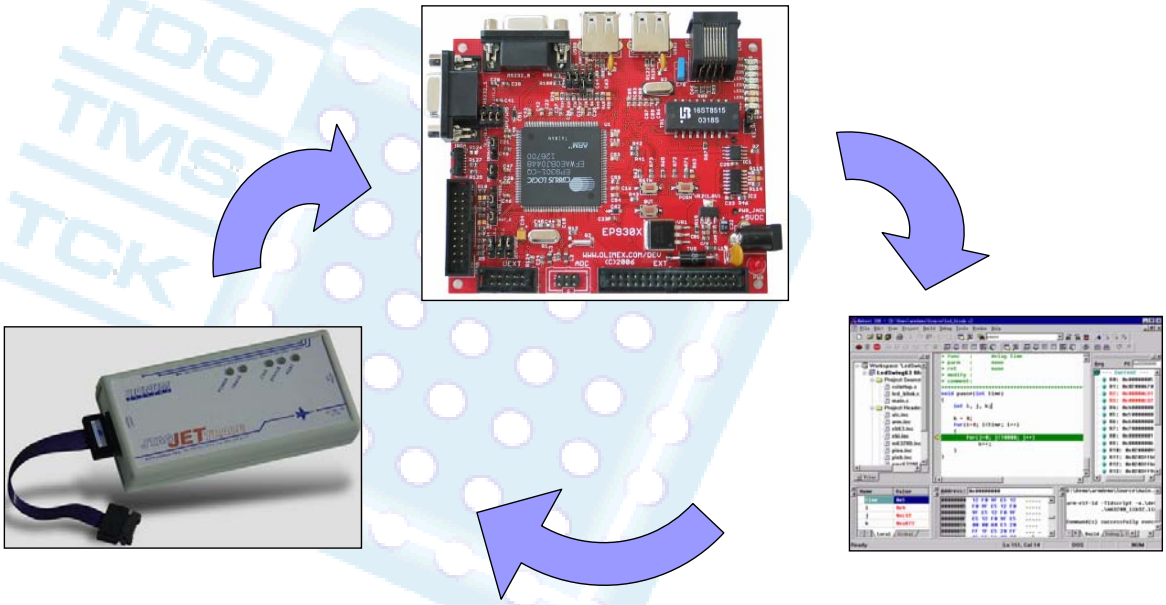
- Upgrade und Anpassung von Firmware



➔Oft leider nicht offiziell unterstützt oder nur schwer / teuer durchführbar

Anwendungsbeispiele

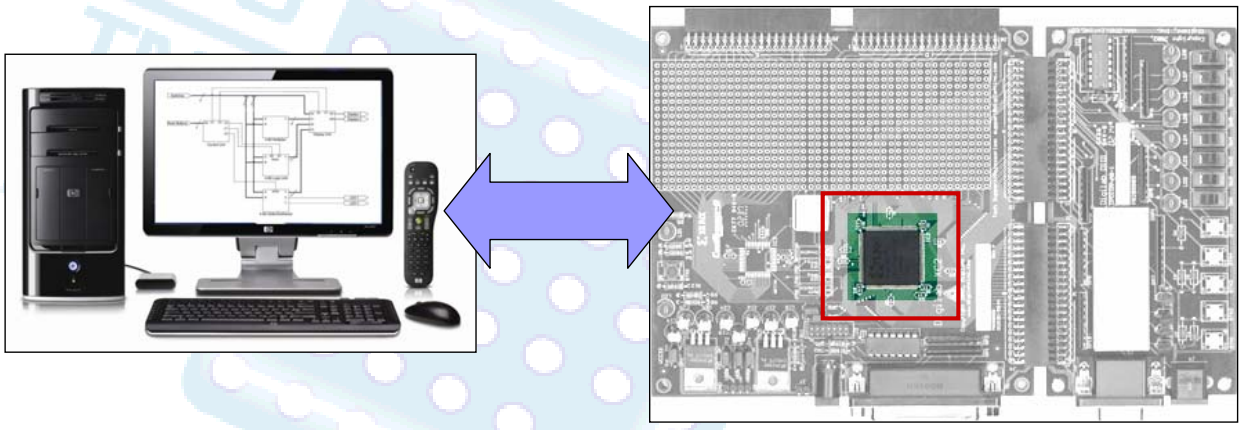
- In-System-Debugging von Mikroprozessoren



➔Datentransferraten begrenzen Geschwindigkeit

Anwendungsbeispiele

- Emulation der Systemumgebung vor der Fertigung

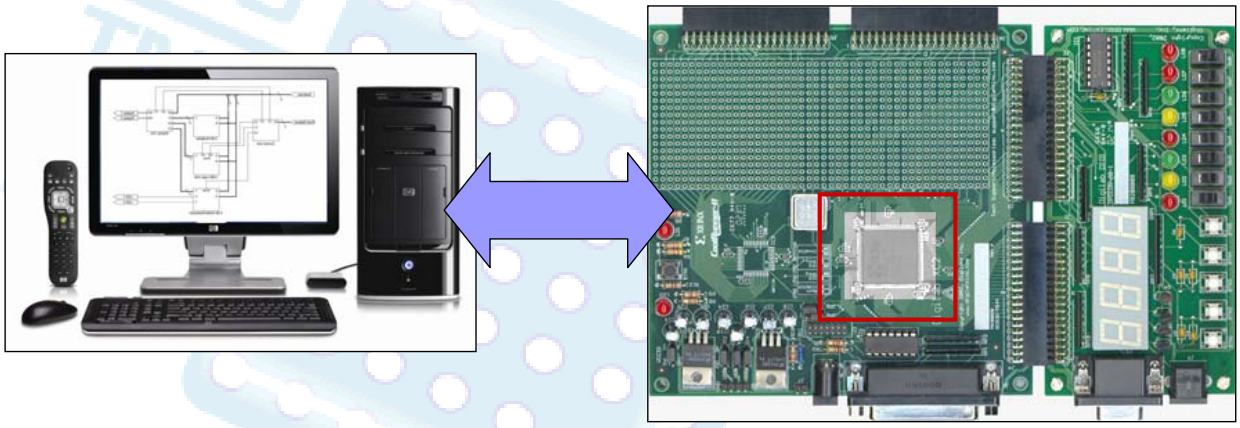


➔Datentransferraten begrenzen Geschwindigkeit

➔Emulation analoger Baugruppen nur in Grenzen

Anwendungsbeispiele

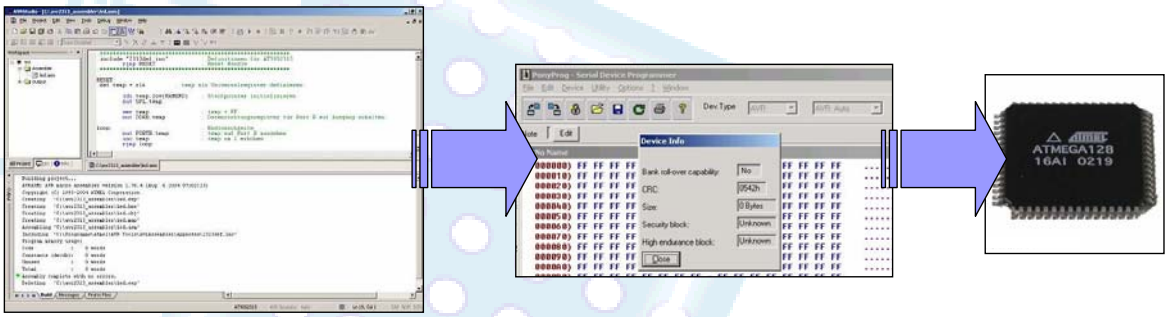
- Emulation des Schaltkreises vor der Fertigung



➔Alle Anschlüsse müssen an JTAG Geräten enden

Anwendungsbeispiele

- Programmierung und Reprogrammierung eines Schaltkreises



➔Gute Unterstützung auch für Feldeinsatz

Einschätzung

- JTAG löst die meisten digitalen Nagelbretttests ab
- JTAG erweitert das Spektrum der Testmöglichkeiten um neue Verfahren
- JTAG ist einfach in Hardware zu integrieren (z.Bsp. TAP: 4+4 FFs und 20-40 Gatter)
- JTAG vereinfacht Tests und fördert diese so
- JTAG ermöglicht frühzeitige Test
- JTAG ist standardisiert und lässt so viele Tests ohne Vorkenntnis der Hardware zu
- Analoge Tests können nur bedingt mit JTAG umgesetzt werden, ebenso Tests mit nicht JTAG-fähiger Hardware (LED)
- Die Verwendung von JTAG schränkt u.U. die Testgeschwindigkeit ein
- Einige Befehle sind herstellerspezifisch, so dass spezielle Software benötigt wird

➔ **JTAG ist Bestandteil fast aller aktuellen Mikroprozessoren, CPLDs, FPGAs, Mobiltelefone, Spielekonsolen, usw. und hat sich offensichtlich seit seiner Einführung in der Elektronikbranche etabliert, da die immensen Vorteile gegenüber den Nachteilen deutlich überwiegen**

Verfügbarkeit

- Dieser Seminarvortrag ist momentan verfügbar unter

  web.inf.tu dresden.de/~s8646344/JTAG/

Abgrenzug

- Dieser Seminarvortrag soll einen Überblick über die beschriebene Thematik geben und erhebt daher keinen Anspruch auf Vollständigkeit, weder in der Tiefe noch in der Breite der Themenabdeckung
- Beschriebene Verfahren wurden zum größten Teil nicht praktisch überprüft, jedoch mit mehreren unterschied lichen Quellen abgeglichen. Dennoch können bisher unbemerkte Fehler enthalten sein, die Darstellung der Problematik erhebt keinen Anspruch auf Fehlerfreiheit
- Einige Aspekte wurden bewusst ausgespart, um den zeitlichen Rahmen des Vortrags nicht zu sprengen, so zum Beispiel die Beschreibungssprache BSDL

Quellen

- www.boundary-scan.co.uk/
- hem.hj.se/~mabe/TestForum2002/
- www.tele.pw.edu.pl/~ptomasze/docs/
- www.eet.bme.hu/~poppe/
- www.ee.ic.ac.uk/pcheung/teaching/ee3_DSD/
- www.inaccessnetworks.com/ian/projects/ianjtag/jtag-intro/
- www.xjtag.com/support-jtag/
- www.intellitech.com/company/
- www.scienceprog.com/wp-content/uploads/2006i/
- www.digitaldawgpound.org/wp-content/uploads/2006/08/
- www.pitts-electronics-home.de/electron/schaltpl
- www.valueforyou.de/Photos/
- www.spea.com/
- www.dataghost.com/ipl/
- www.embedded-tools.de/Produktbilder/Hersteller/Signum/
- www.moli.uwaterloo.ca/ECE223/Images/
- www.pitsch.de/stuff/mmc2iec
- www.robomodules.de/portal/typo3temp/
- www.icbank.com/
- en.wikipedia.org/wiki/Boundary_scan_test/