Boundary Scan / IEEE 1149.x

ein strukturelles Verfahren zum Prüfen von Baugruppen und Systemen

Teil 2

Doc. Vers. 2019-02-19

Inh. Dipl. Ing. Mario Blunk

Buchfinkenweg 3
99097 Erfurt / Deutschland

Telefon +49 (0)361 6022 5184

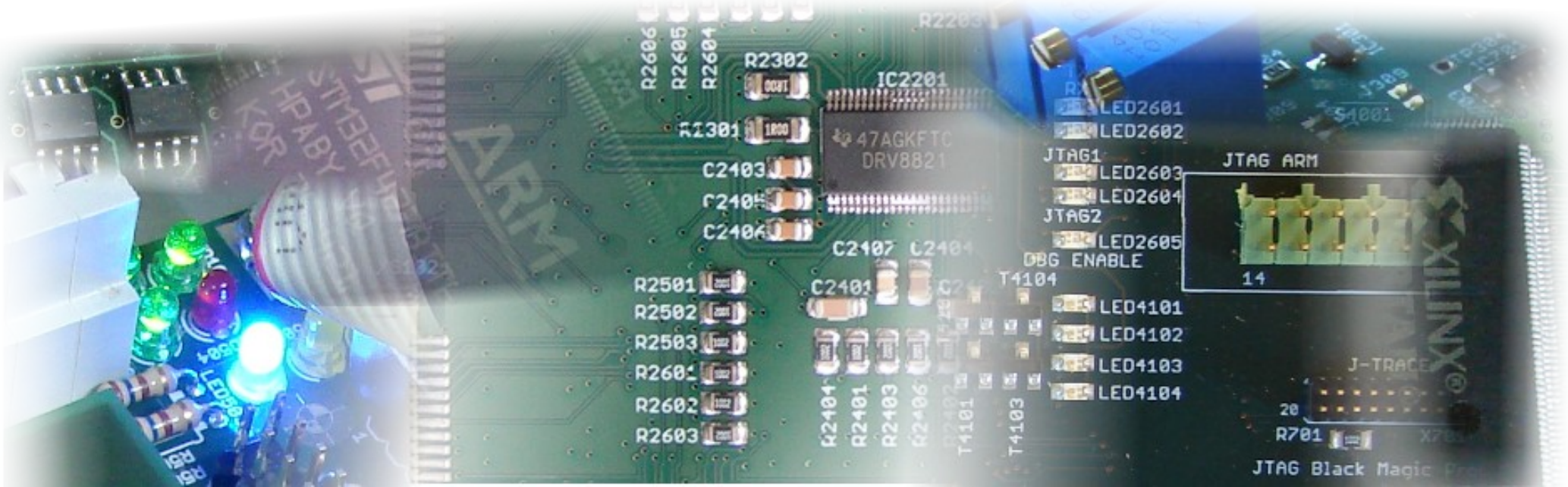
Email info@blunk-electronic.de

Internet www.blunk-electronic.de

BLUNK ELECTRONIC

Design Reviews Gutachten Beratung

HW/SW-Entwicklung Boundary Scan / IEEE 1149.x

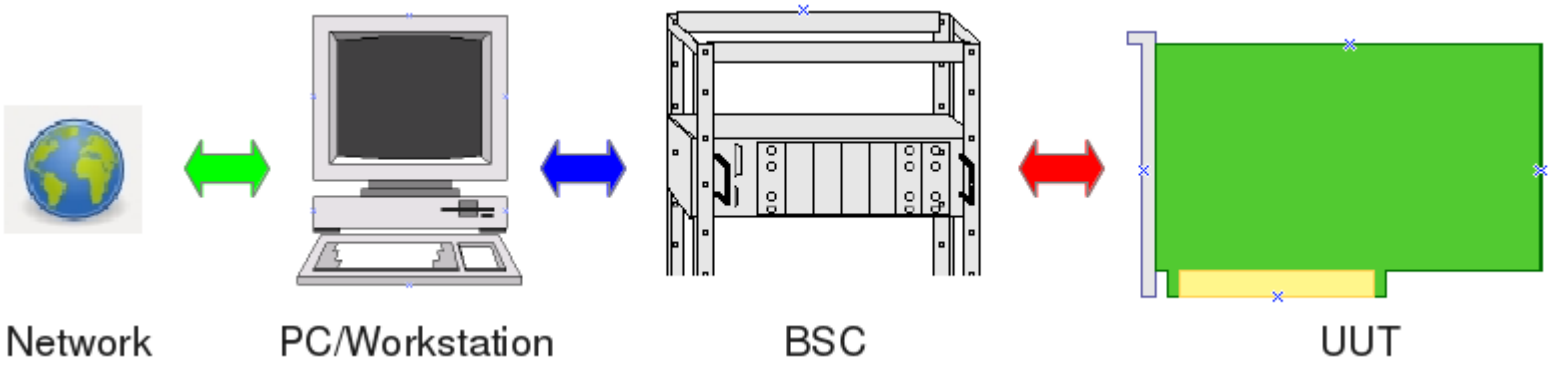


Überblick

- benötige Ausrüstung
- Testarten / Profile
- Boundary Scan Zellen
- Register
- DR/IR-Scan
- TAP Controller
- Erstellung und Ausführung von Tests
- Standards und Literatur

Überblick BST System

Überblick Testsystem:

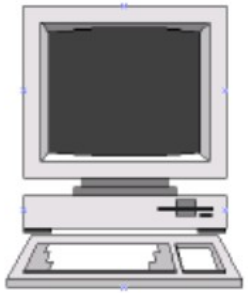


Wie funktioniert es ?

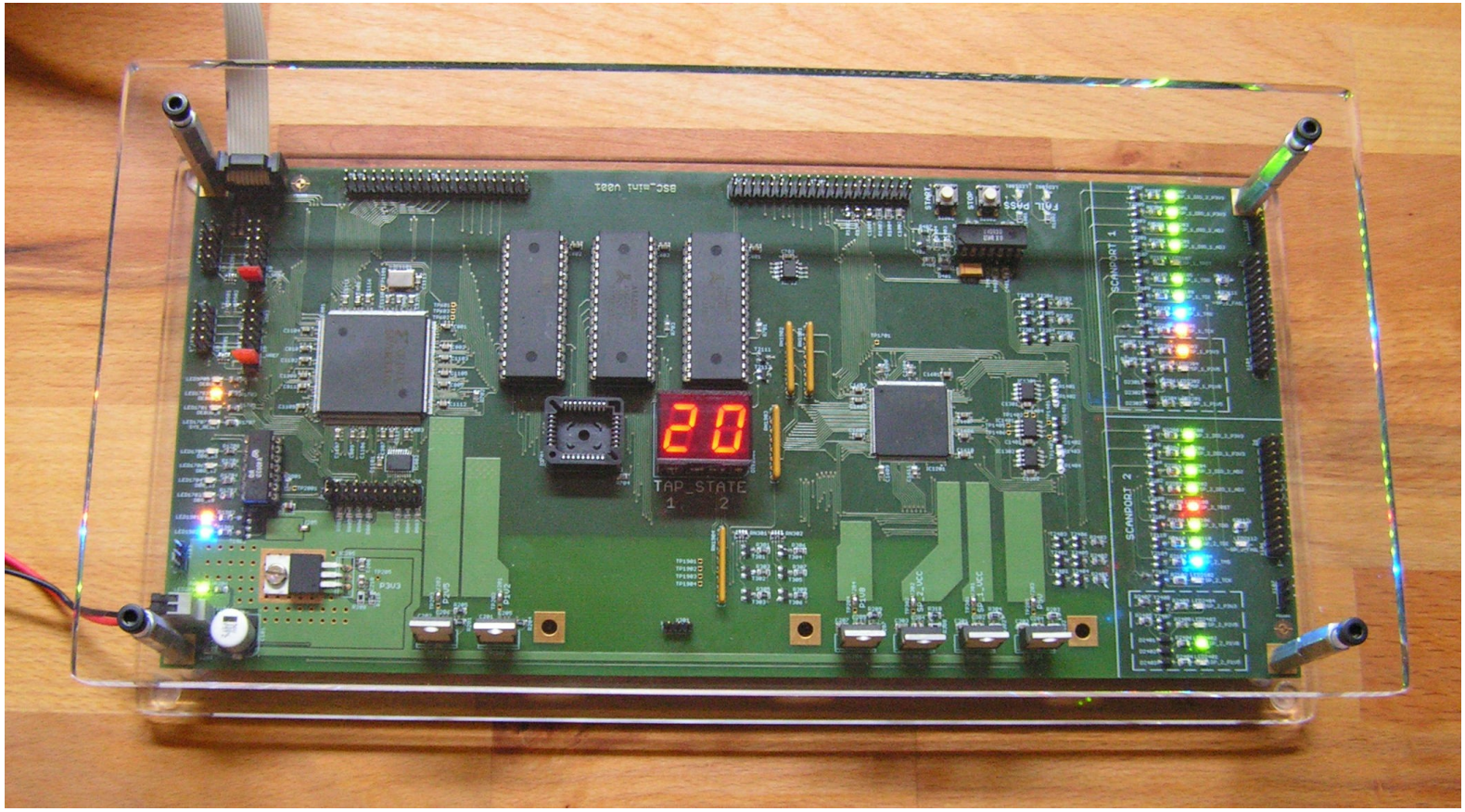
1. Mit PC werden CAD-Daten importiert und Testprogramme generiert (automtisch, halbautomatisch, manuell)
2. Boundary Scan Controller (BSC) führt Testprogramme auf UUT aus.

Ausrüstung #1

1. PC mit BST-Software
2. **Boundary Scan Controller** (BSC) für Training, Labor und Entwicklung
3. Verbindungskabel



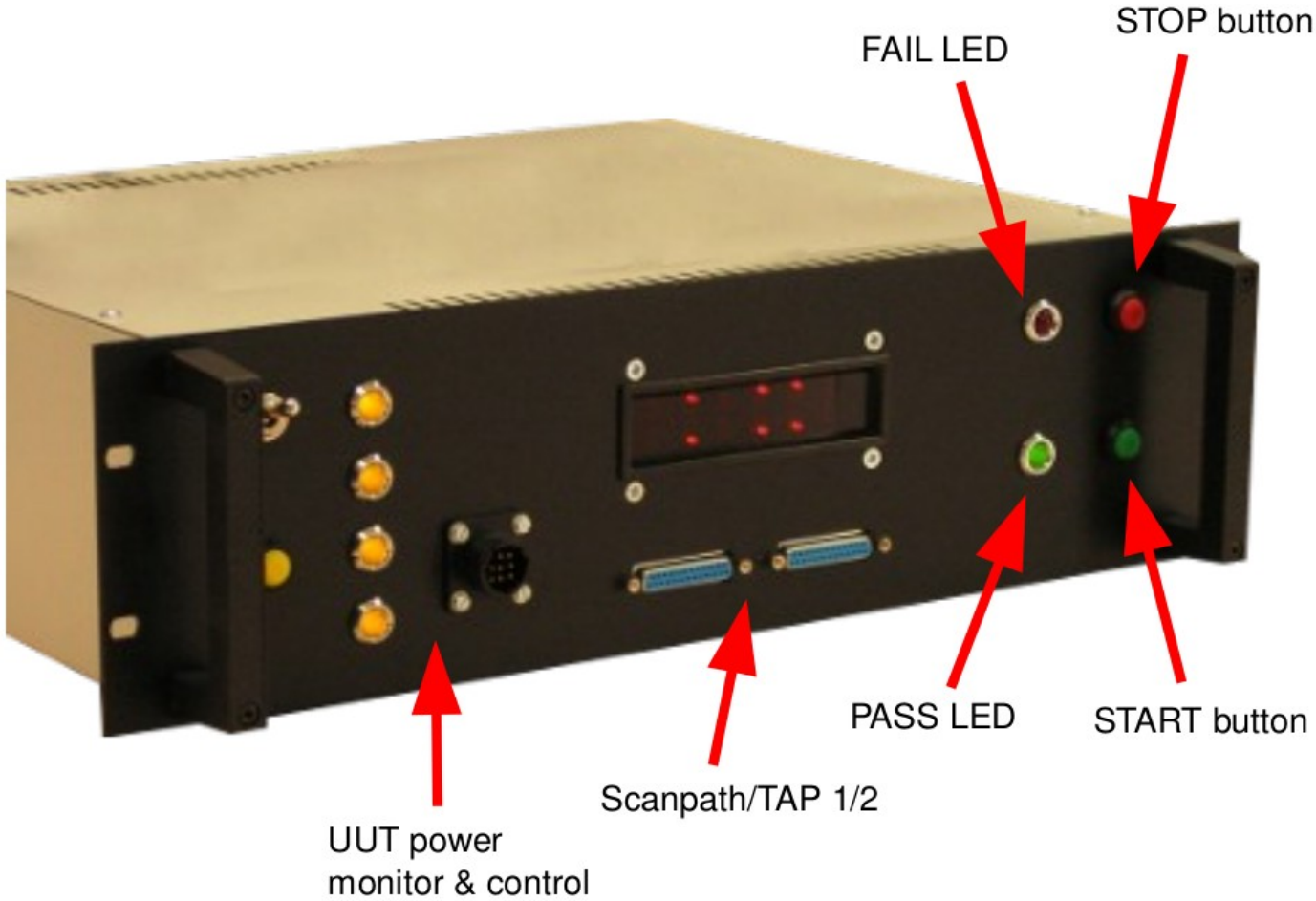
PC/Workstation



Ausrüstung #2

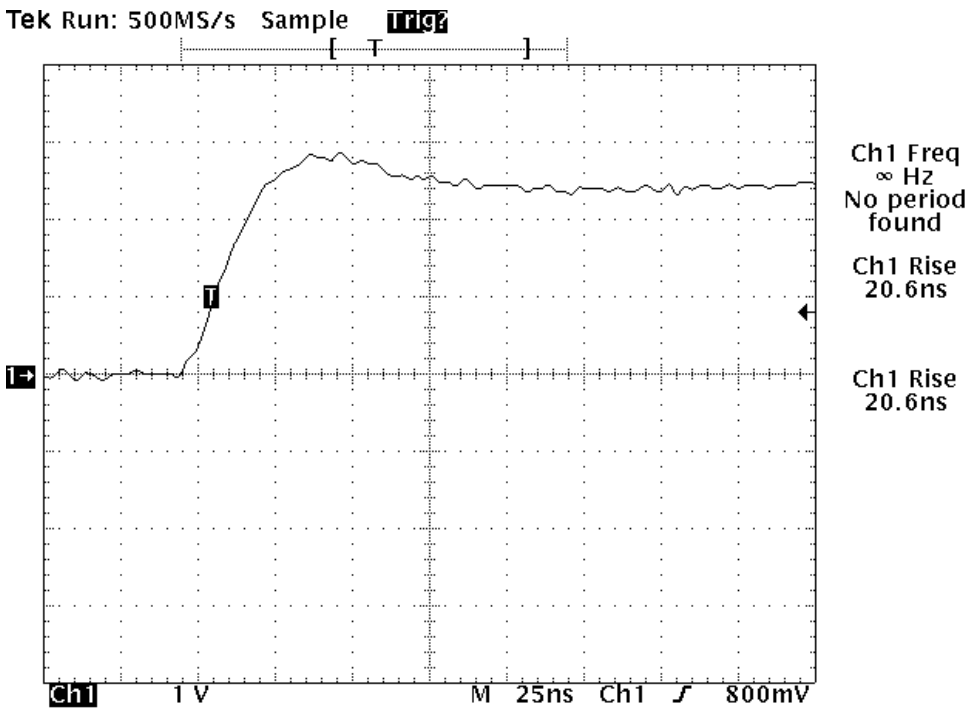
1. PC mit BST-Software
2. **Boundary Scan Controller** (BSC) für industrielle Fertigung
3. Verbindungskabel

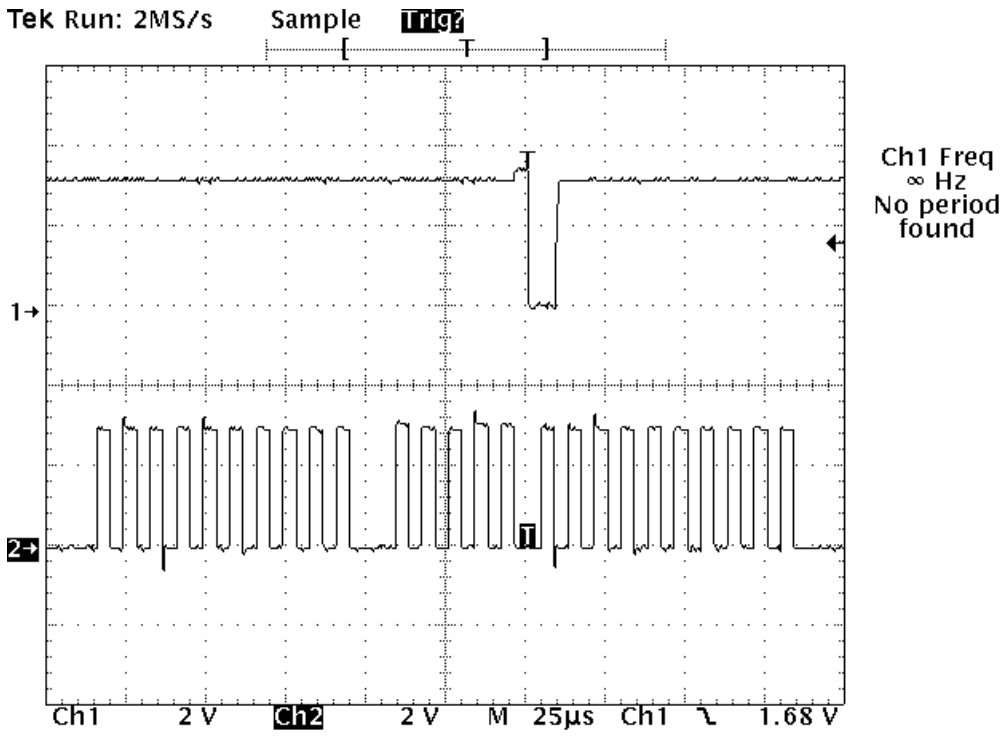
PC/Workstation



Ausrüstung #3

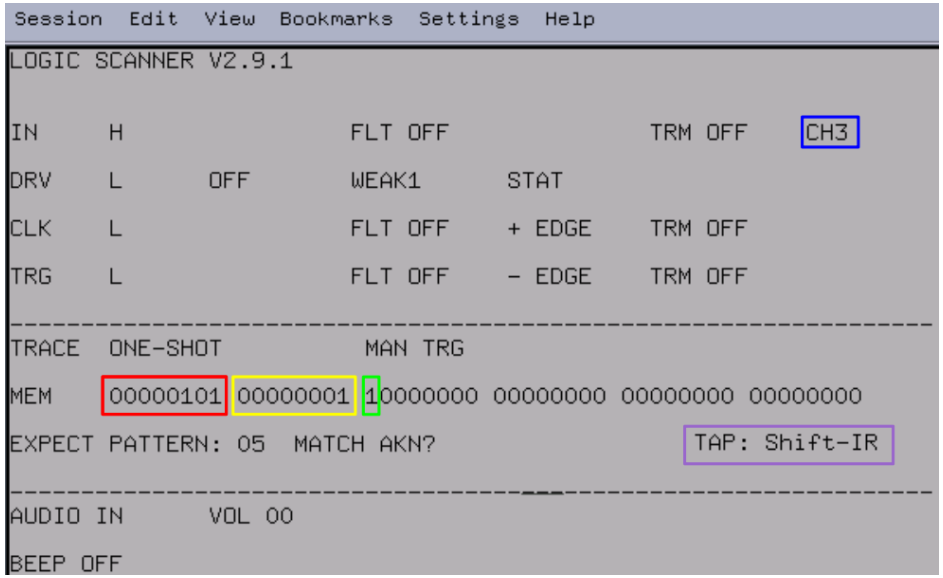
1. Spannungsversorung (Strombegrenzung !)
2. Multimeter
3. Oszilloskop
4. optional Logiktester / Bus-Scanner

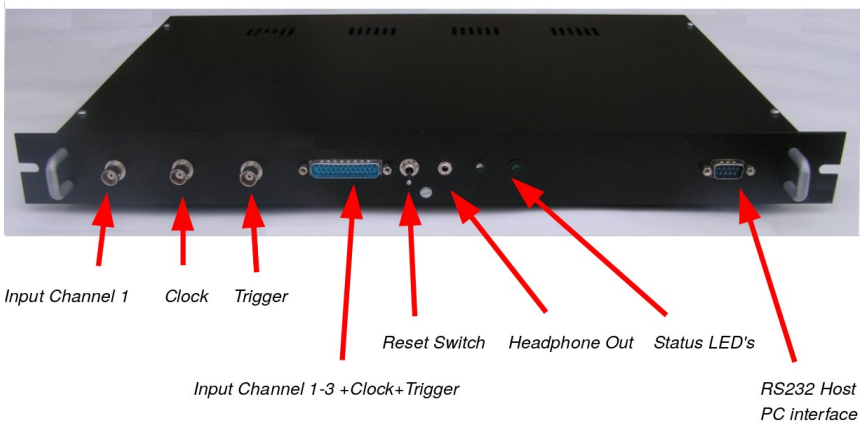


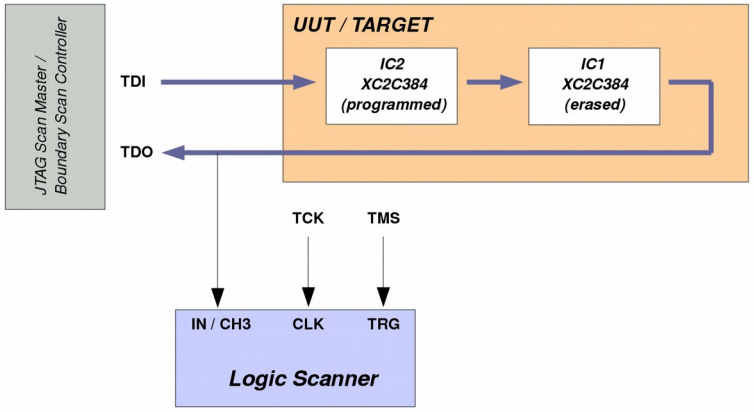


Ausrüstung #4

1. Spannungsversorung (Strombegrenzung !)
2. Multimeter
3. Oszilloskop
4. optional Logiktester / Bus-Scanner



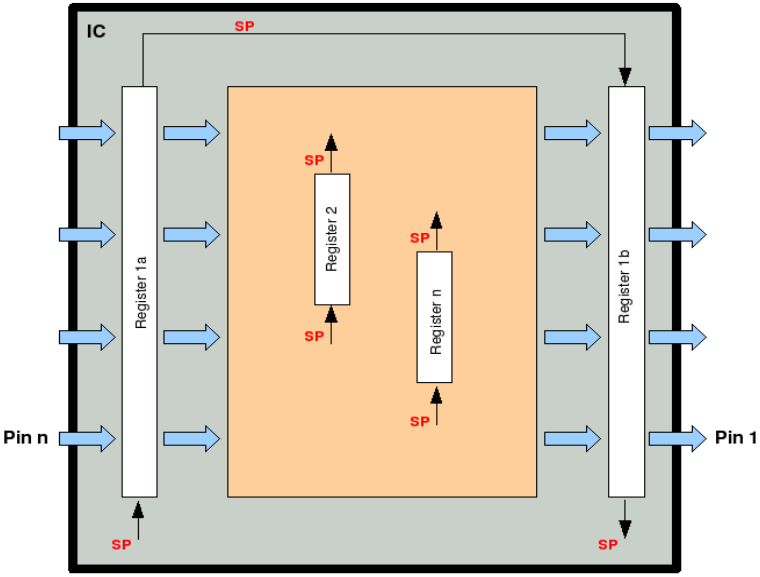




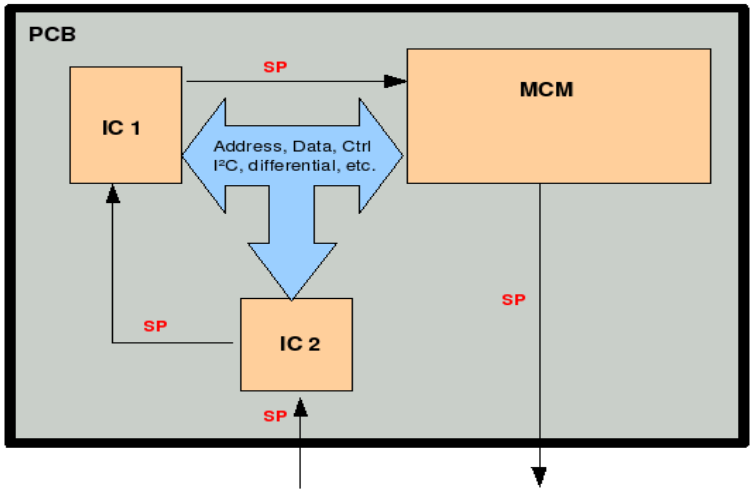
Wie funktioniert BST ? #1

1. Um den eigentlichen Logikkern eines IC (an dessen Grenze/ engl. Boundary) befindet sich ein Register aus Testpunkten. Mit diesen kann ein Pin/Pad gemessen oder angesteuert werden.
2. Zwischen ICs, SOC-Modulen und PCBA werden Testdaten ausgetauscht.

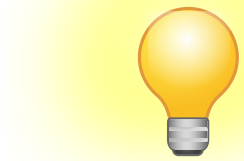
Struktur im IC:



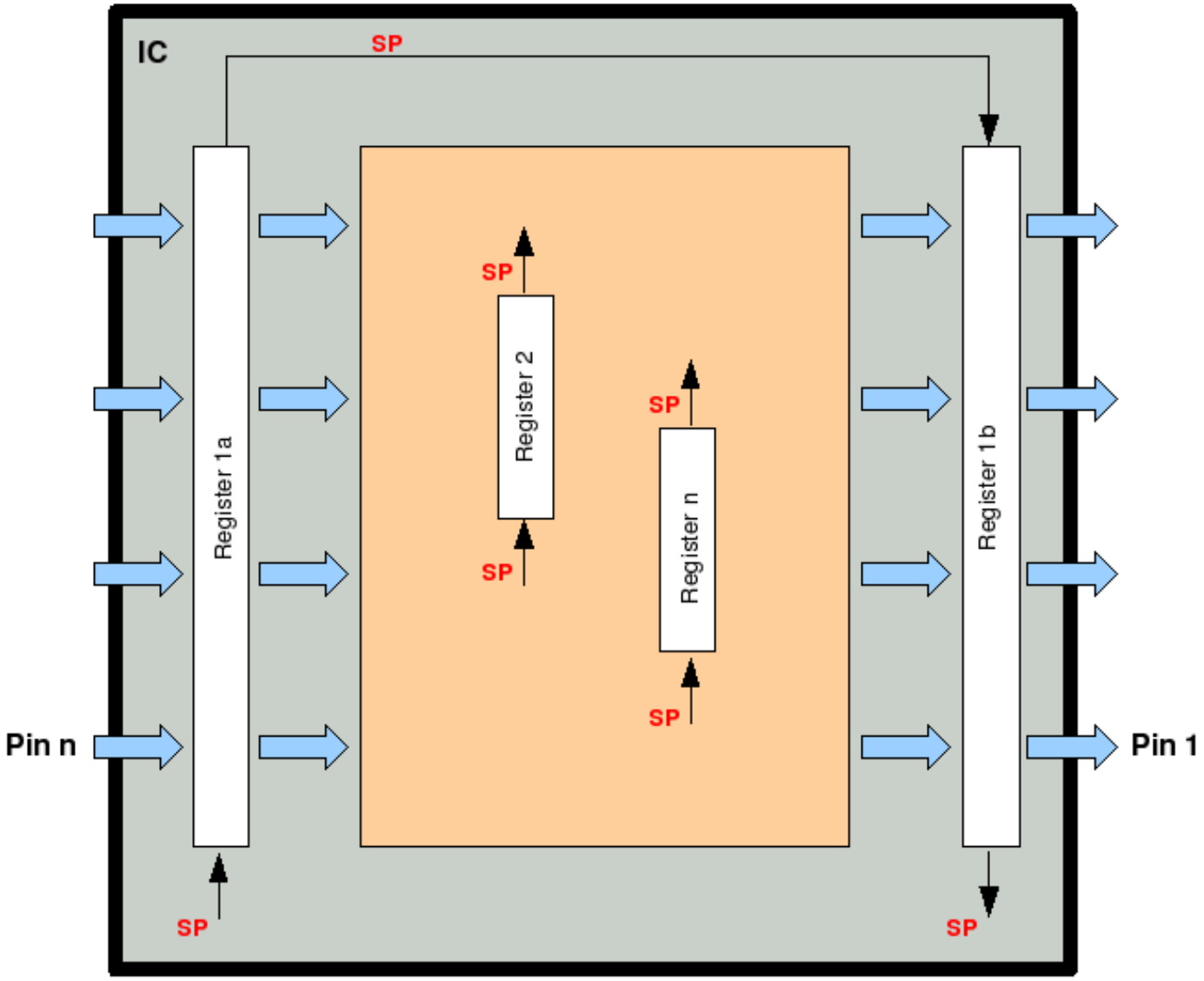
Struktur auf PCBA:



Wie funktioniert BST ? #2

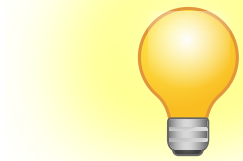


im IC:

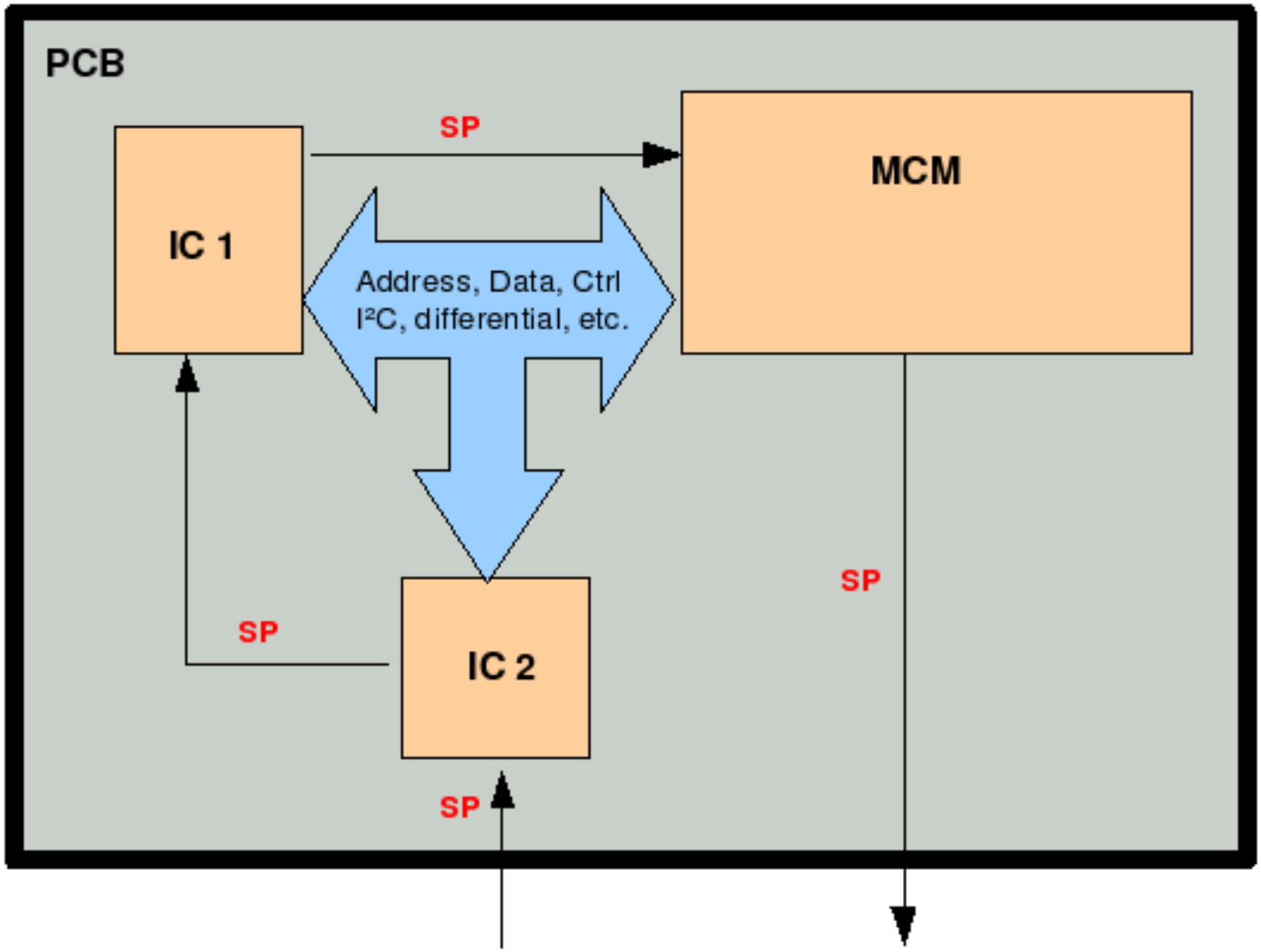


boundary scan fähiger IC → BIC

Wie funktioniert BST ? #3

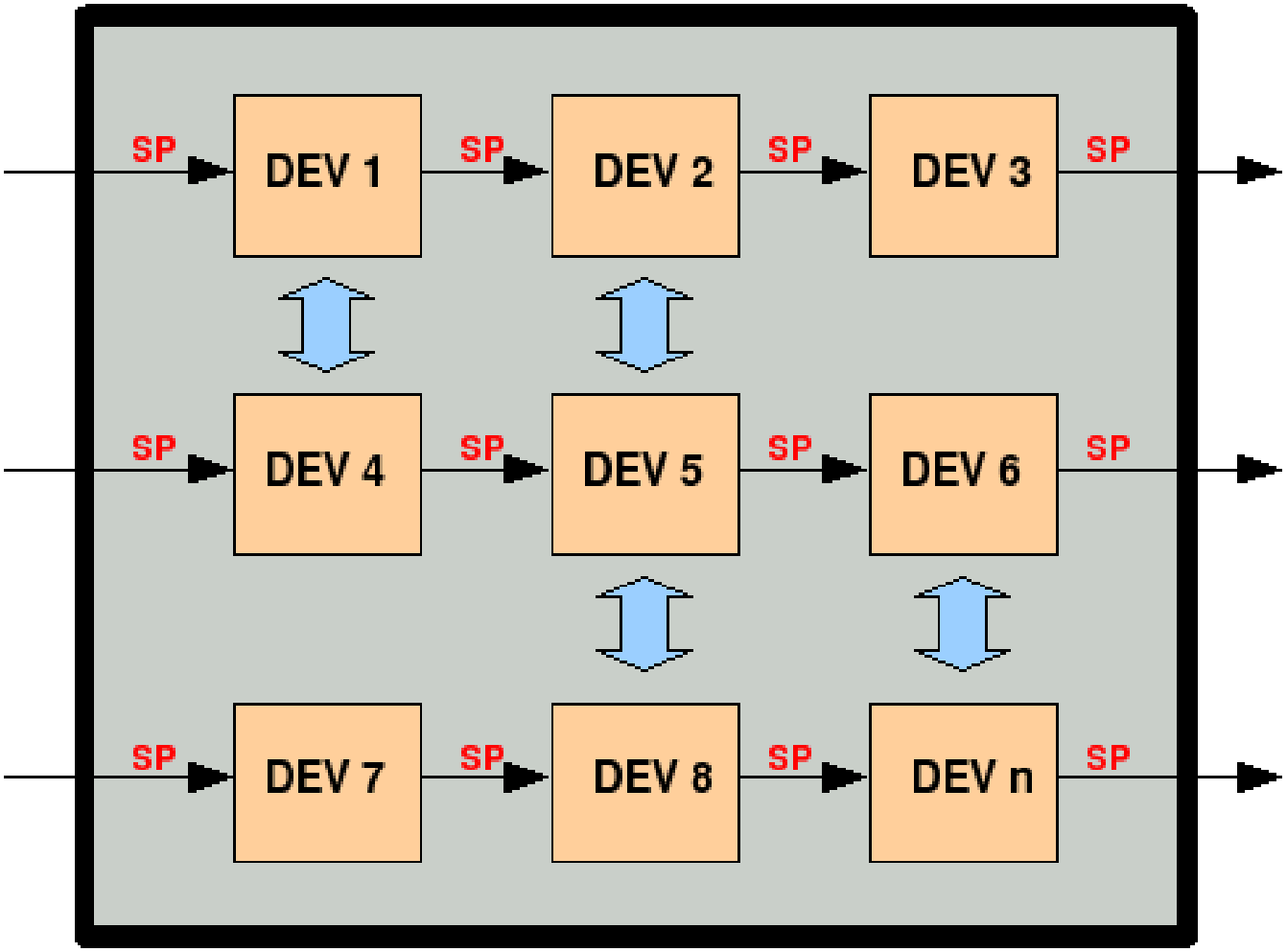


PCBA:



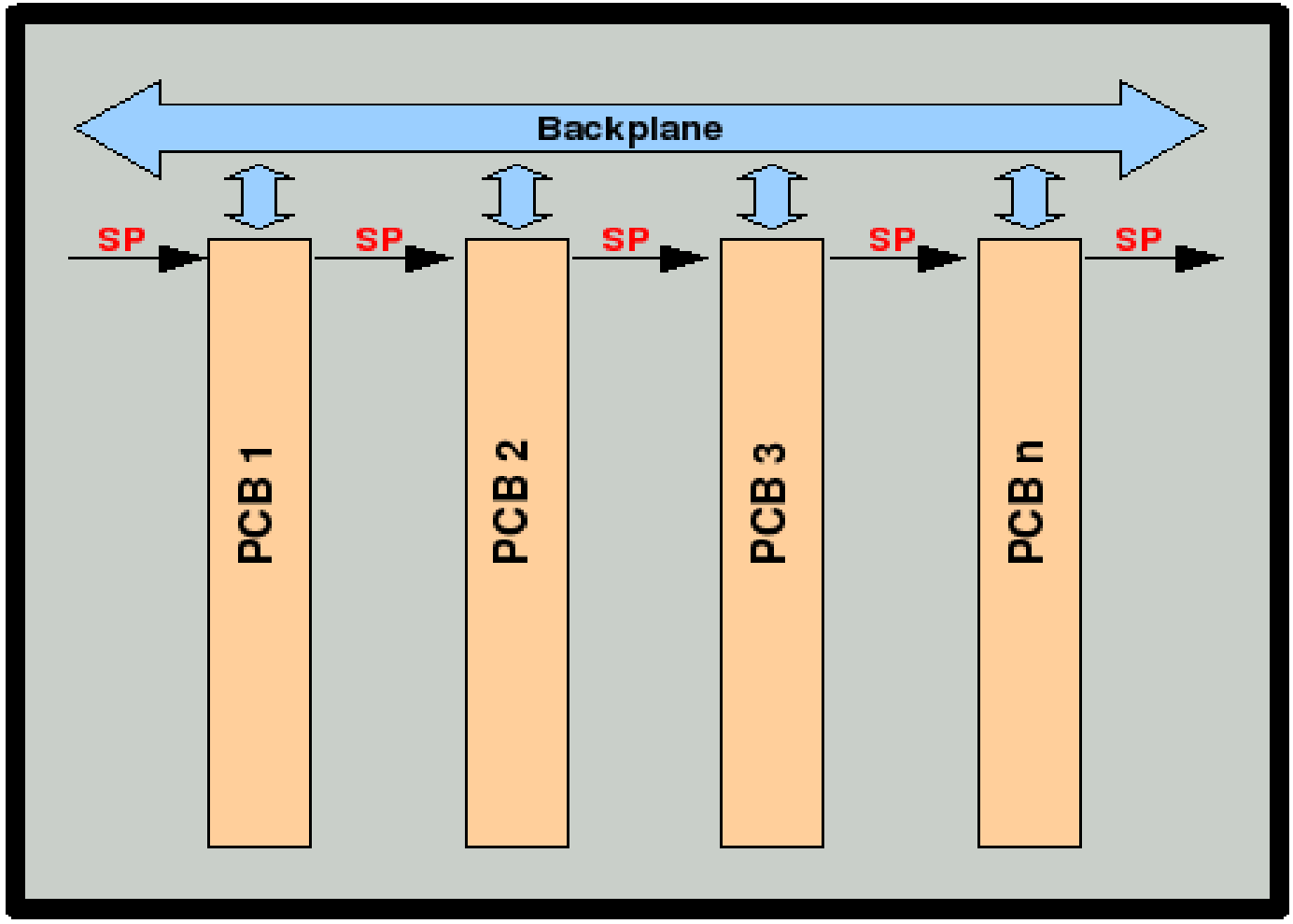
Wie funktioniert BST ? #4

PCBA mit mehreren Scanpfaden:



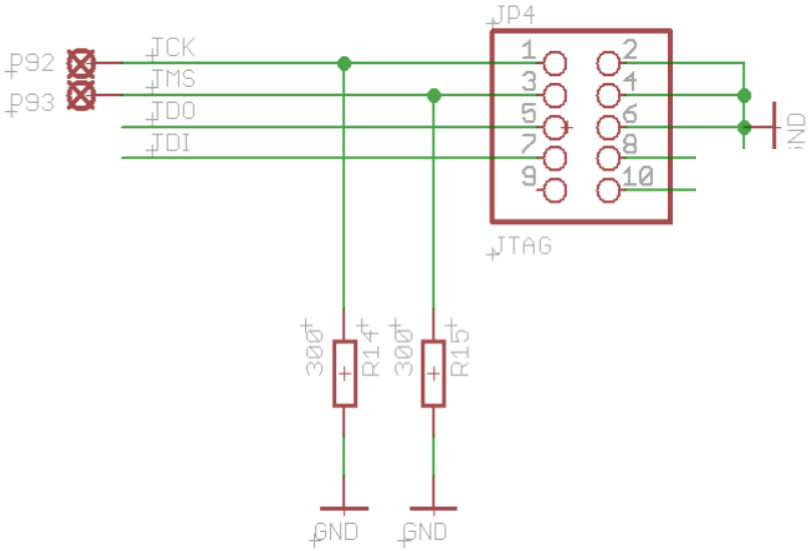
Wie funktioniert BST ? #5

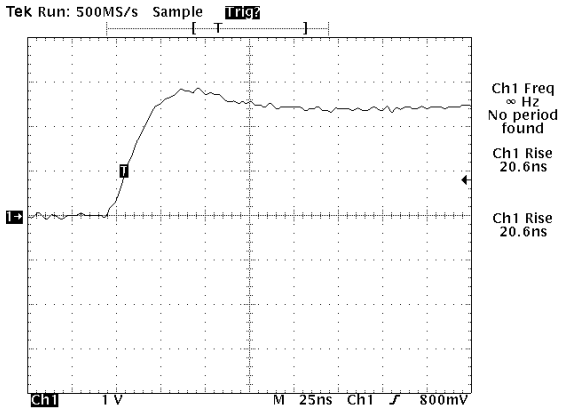
Geräte- und Systemtest:

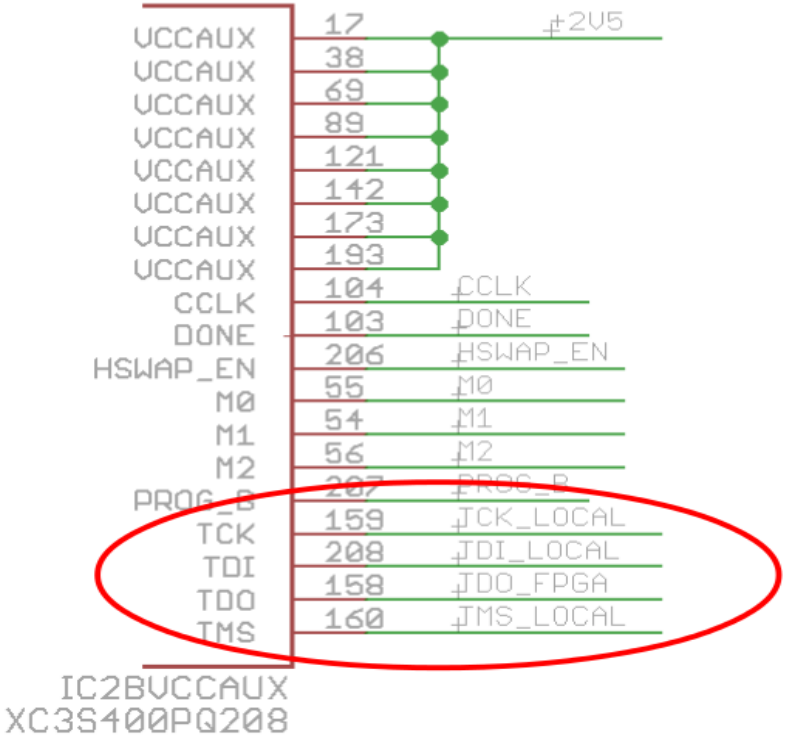


Wie funktioniert BST ? #6

Schaltungstechnik:



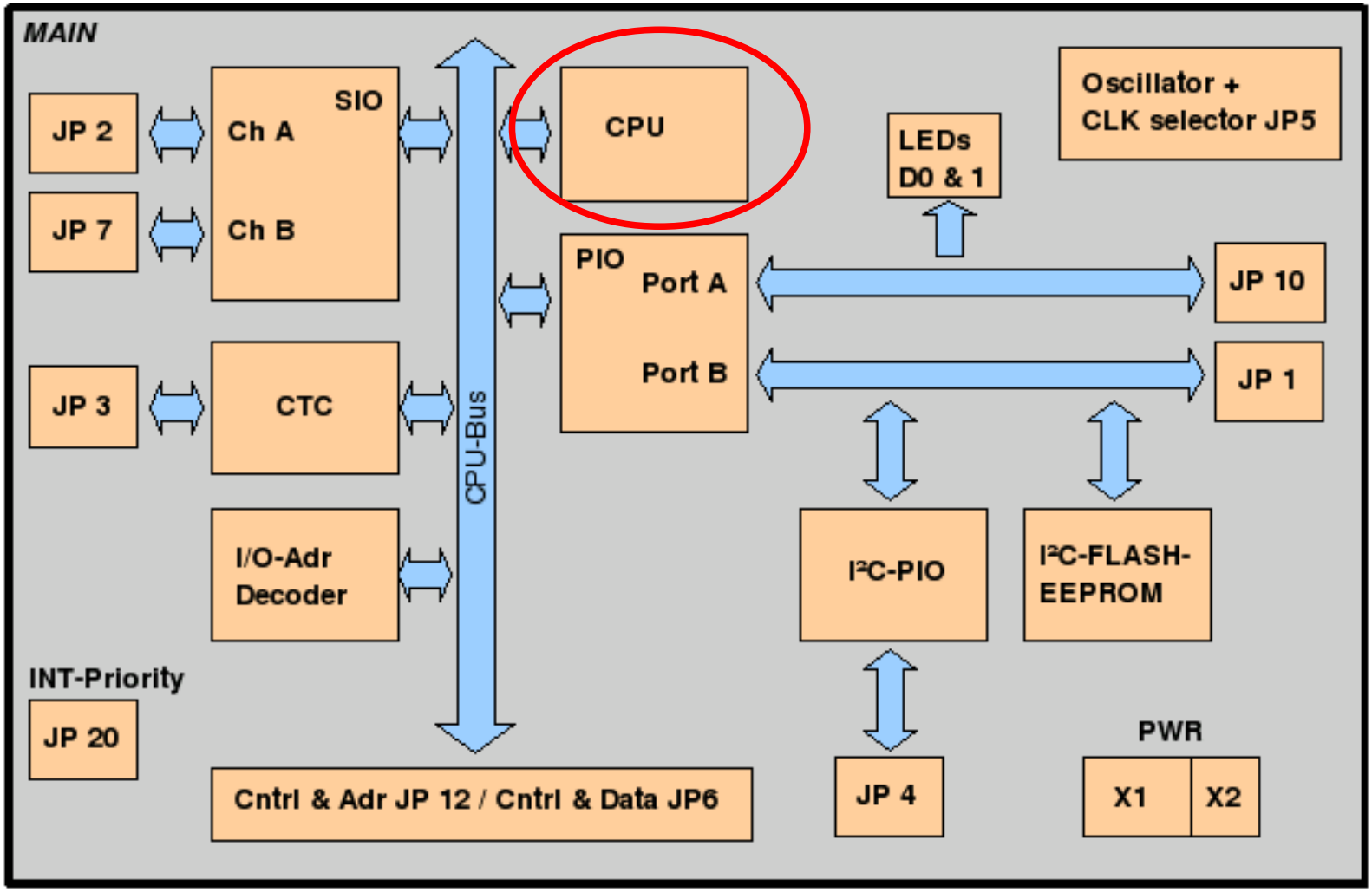




Signalintegrität !!!

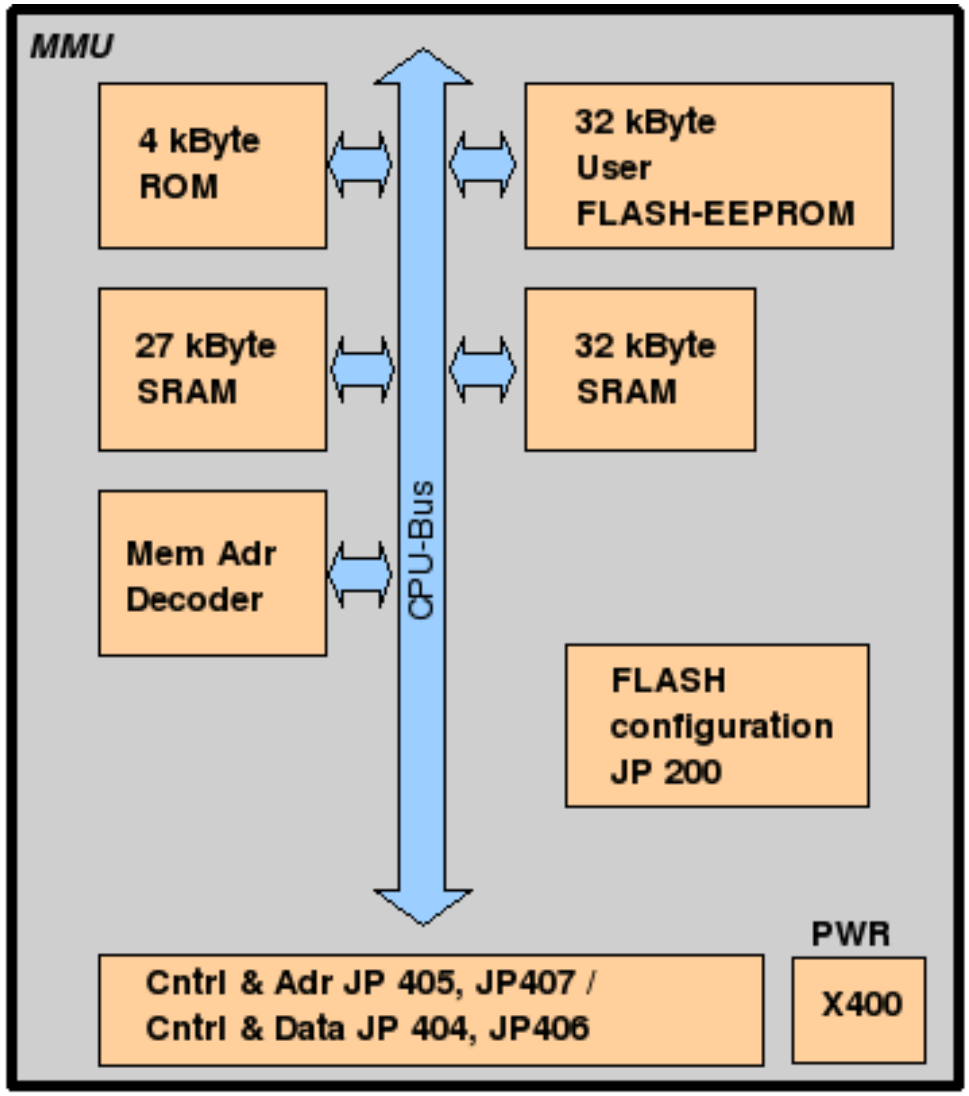
Wie funktioniert BST ? #7

Nicht BST-fähige Schaltungsteile:



Wie funktioniert BST ? #8

Nicht BST-fähige Schaltungsteile:



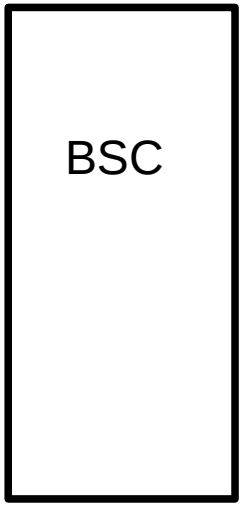
Test – Profile / Arten

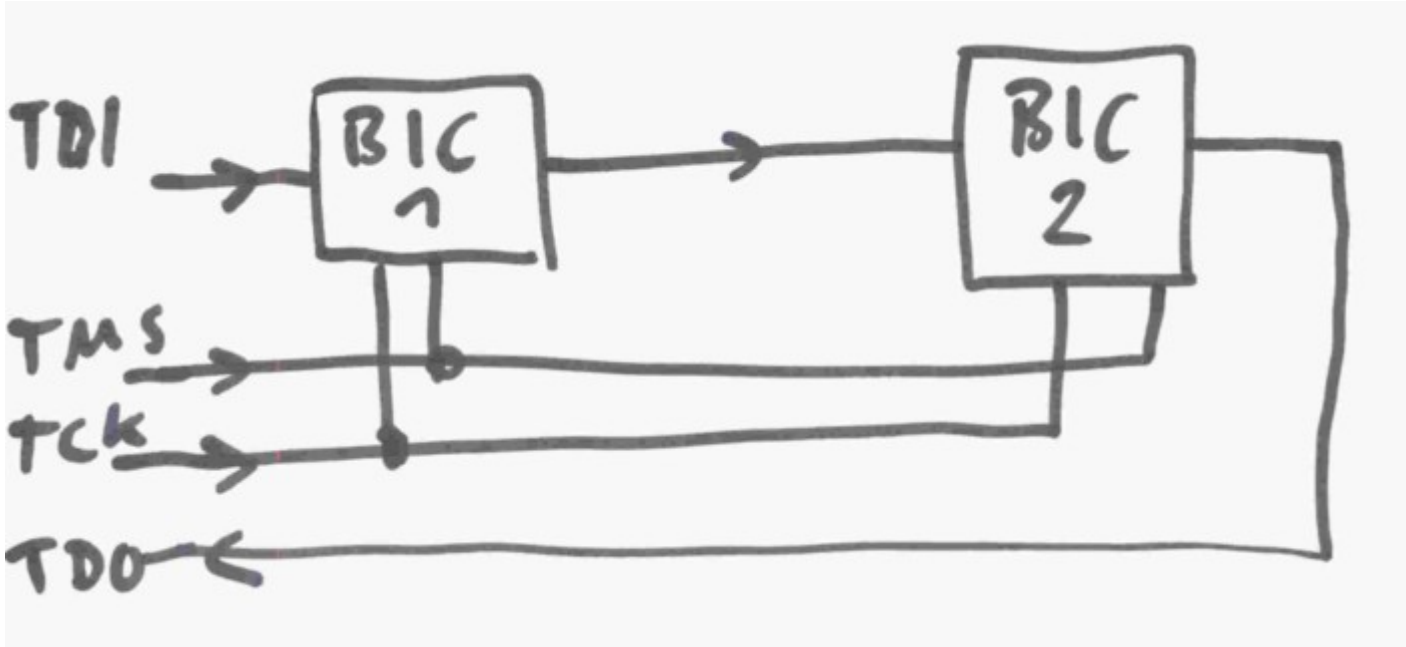
1. Infrastructure-test / scanpath-test / scanchain-test
2. Interconnect-test / Verbindungstest (verschieden Algorithmen)
3. logic cluster test
4. Memory-connect-test / Speicherverbindungstest
5. In-System-Programming (FLASH, SVF, STAPL, IEEE1532, I²C, SPI, …)
6. Peripherietests (LEDs, Displays, Tasten, Relais, Motoren, …)
7. Mixed Signal (ADC, DAC, ...)

8. Verdrahtungstest
9. Modul-test
10. System-test

Infrastrukturtest

1. IEEE1149.x Testbus
2. Verbindung zum Controller (BSC)
3. Längen von Registern → BSDL Modelle
4. ID-Codes
5. nicht-intrusiv



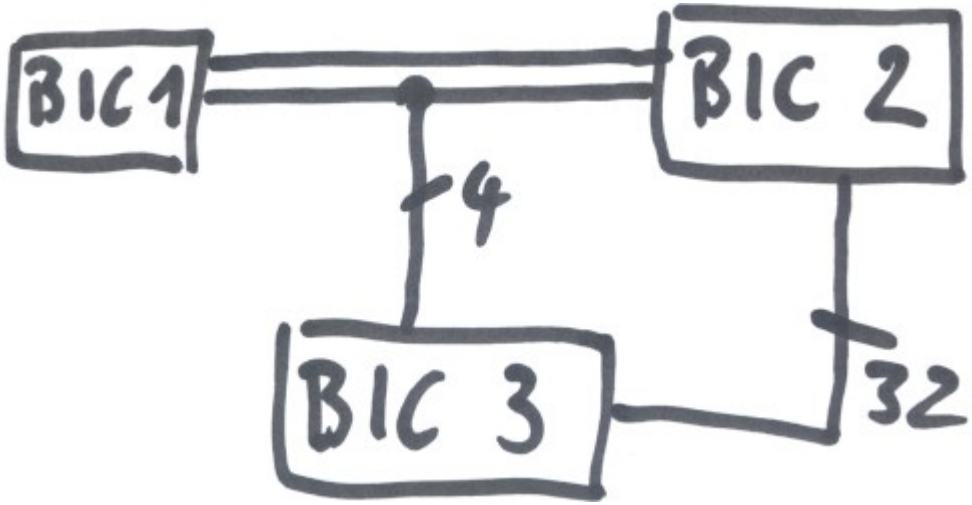


boundary scan fähiger IC → BIC

Verbindungstest #1

1. Interconnections Test
2. Boundary Scan fähige IC (BIC) untereinander

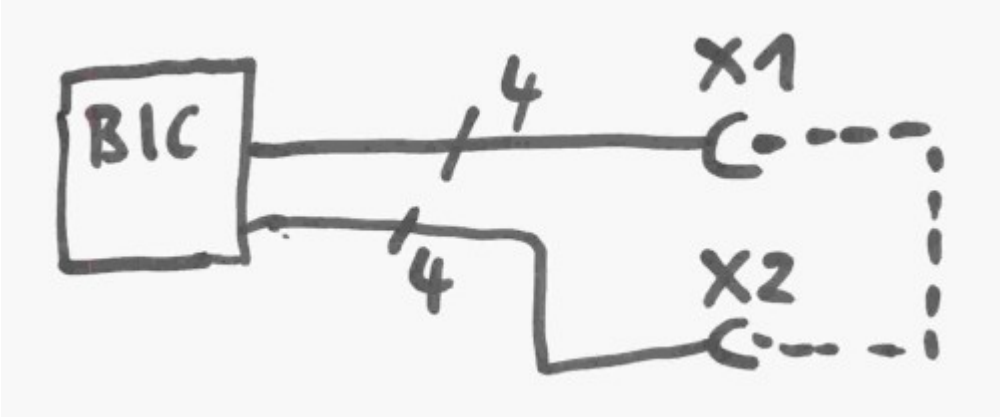
   Serienwiderstände, Treiber, Drähte, Kabel, …
3. Pull-Widerstände
4. Steckverbinder (Systemtest !)
5. Programmerstellung automatisch (ATG) → BSDL Modelle
6. intrusiv



boundary scan fähiger IC → BIC

Verbindungstest #2

1. Steckverbindungen
2. Leitungstreiber, Empfänger -> Loop-Back
3. Pegelwandler (z.B. MAX232, …)
4. ATG
5. intrusiv

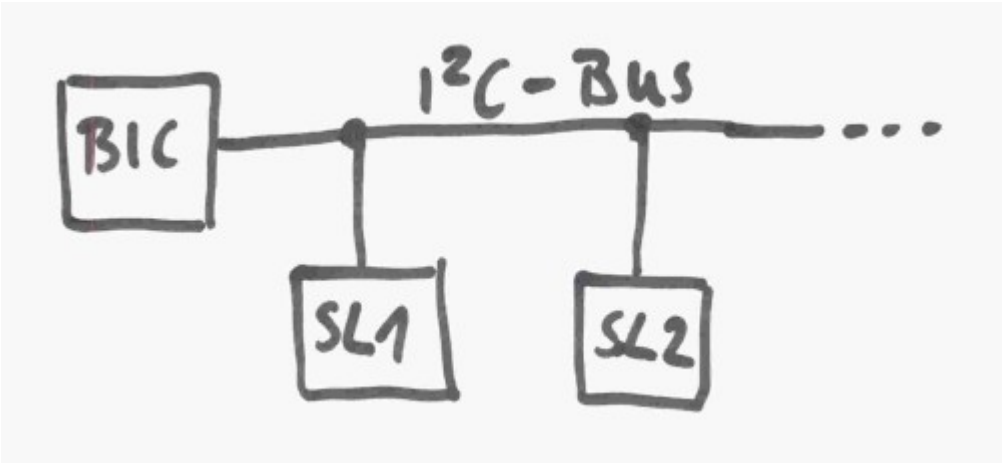


Verbindungstest #3

1. Oszillatoren, Taktgeber, ...
2. Feststellung von Pegelwechseln
3. keine Frequenzmessung, keine Messung von Pulsweiten
4. ATG
5. intrusiv, nicht-intrusiv

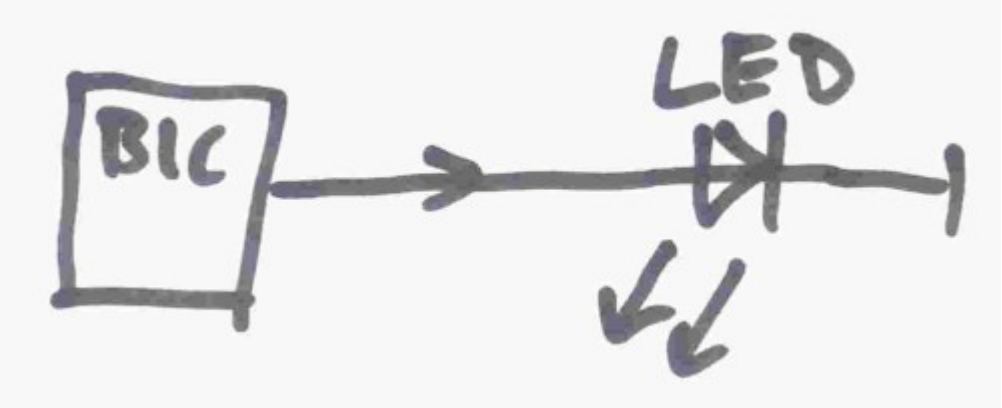
Verbindungstest #4

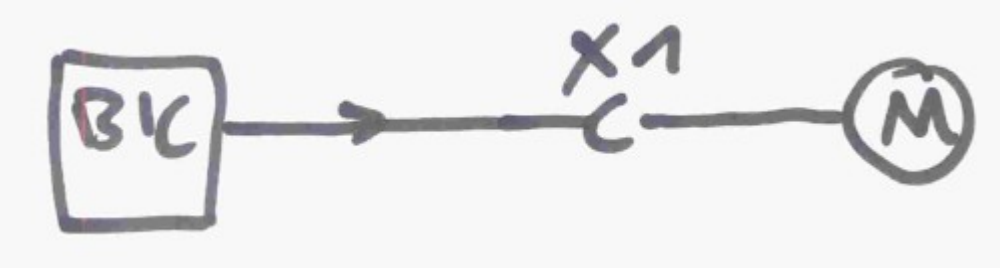
1. Indirekt via Protokolle
2. I²C, Microwire, SPI, …
3. ATG → Modellierung Slave nötig
4. intrusiv



Verbindungstest #5

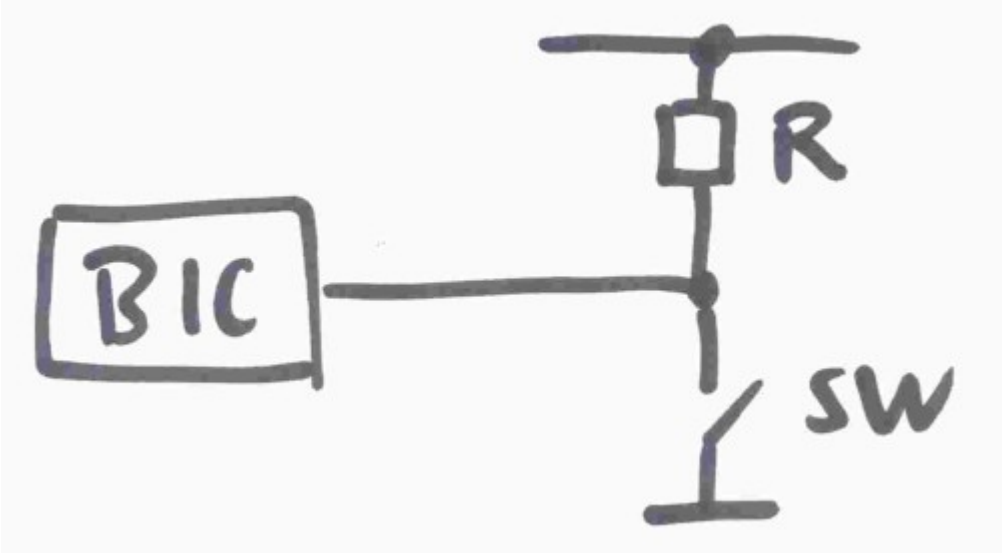
1. Komponenten ohne „Rückmeldung“
2. LEDs, Motoren, …
3. Bediener bestätigt Funktion
4. ATG eingeschränkt
5. intrusiv





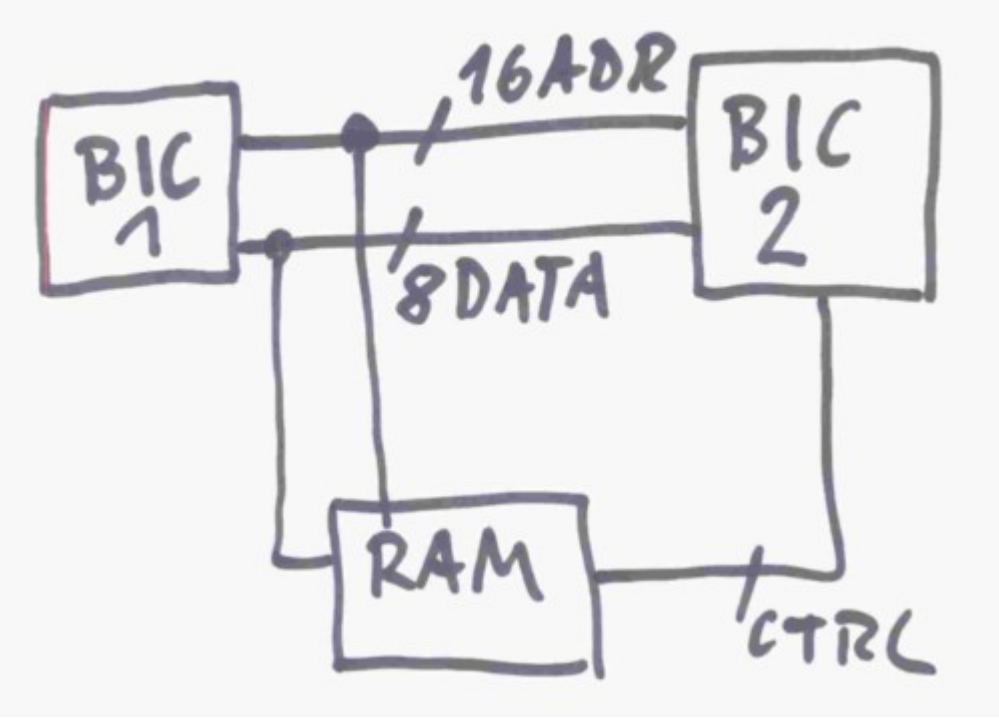
Verbindungstest #6

1. Schalter, Taster, Jumper
2. Betätigung durch Operator
3. ATG eingeschränkt
4. intrusiv, nicht-intrusiv



Speicher-Verbindungstest #1

1. RAM, ROM, FLASH
2. Ziel der Prüfung: Verbindungen !
3. kein Speichertest wie z.B. *MemTest86*
4. Programmerstellung automatisch (ATG)
5. Fehlersuche in Kombination mit „normalem“ Verbindungstest
6. RAM-Dump
7. intrusiv



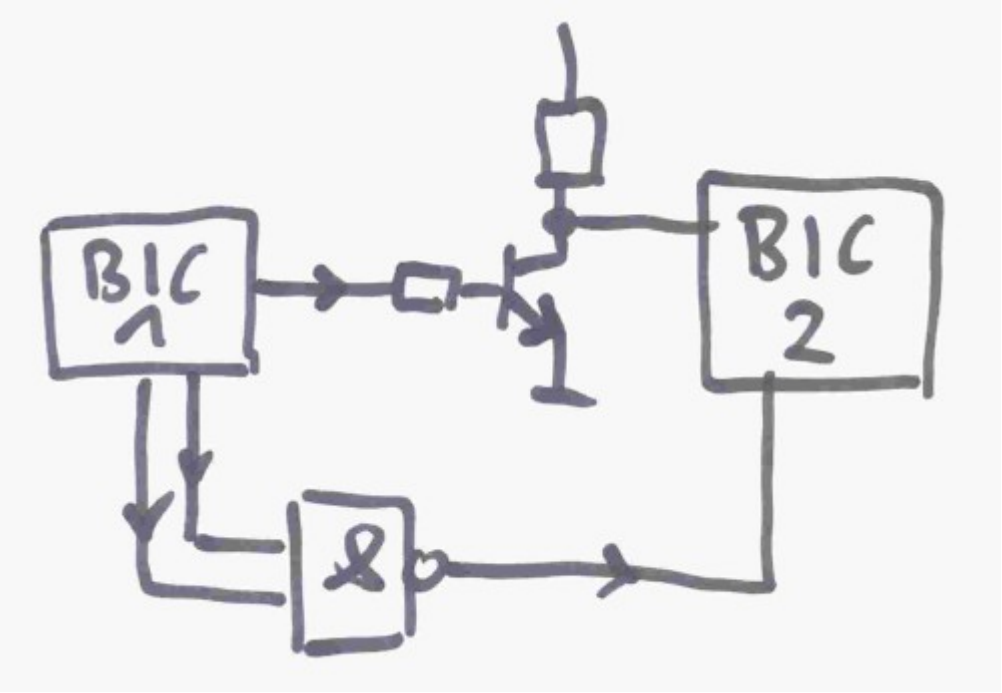
Speicher-Verbindungstest #2

Speicher-
modell:

```
Section port_pin_map NDIP32
»   »   data inout D[7:0] 13 14 15 17 18 19 20 21
»   »   address in A[18:0] 1 30 2 31 3 28 4 25 23 26 27 5 6 7 8 9 10 11 12
--» »   option address min 1000h
»   »   control in CE»  22
»   »   control in OE » 24
»   »   control in WE » 29
EndSection

-------------------------------------------------------------------------------

Section prog
»   SubSection init
»   »   -- step port value
»   »   step 1» ADDR»   drive 7FFFFh »  DATA drive 00h» »   CTRL drive 111b
»   EndSubSection

»   SubSection write
»   »   --Option wordcount 1
»   »   --Option
»   »   -- step port value
»   »   step 2» ADDR»   drive ATG»  »   DATA drive ATG» »   CTRL drive 010b
»   »   step 3» ADDR»   drive ATG»  »   DATA drive ATG» »   CTRL drive 011b
»   EndSubSection

»   SubSection read
»   »   -- step port value
»   »   step 4» ADDR»   drive ATG»  »   DATA expect ATG»»   CTRL drive 001b
»   EndSubSection

»   SubSection disable
»   »   -- step port value
»   »   step 5» ADDR»   drive 7FFFFh »  DATA drive FFh» »   CTRL drive 111b
»   EndSubSection
»
EndSection
```

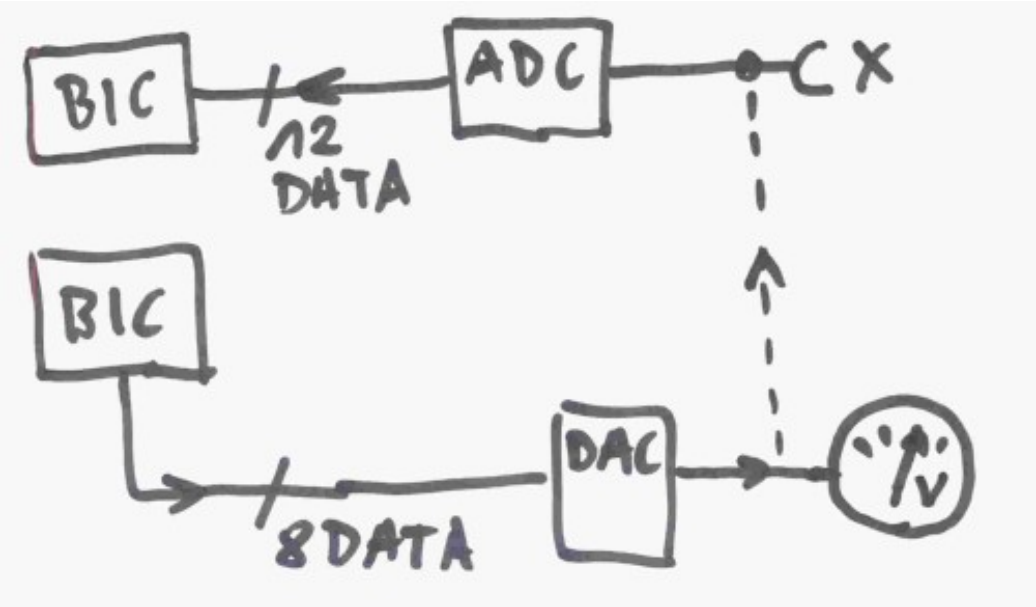
Clustertest

1. digitale **nicht** BST fähiger Logik (Gatter, Multiplexer, Zähler, …)
2. Programmerstellung ohne ATG (Schaltplan, Datenblätter, ...)
3. intrusiv



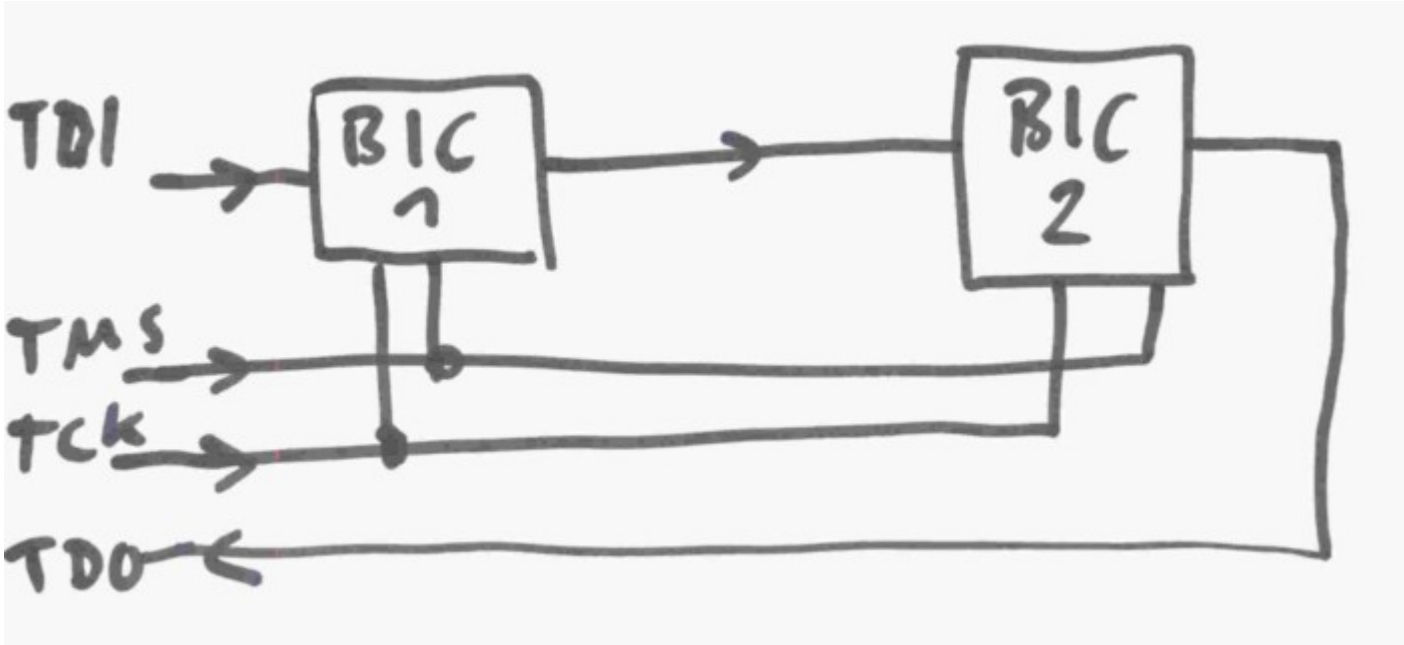
Mixed Signal

1. Programmerstellung ohne ATG (Schaltplan, Datenblätter, …)
2. Zusatzausrüstung: Multimeter, Referenzquellen, ...
3. intrusiv
4. IEEE 1149.4



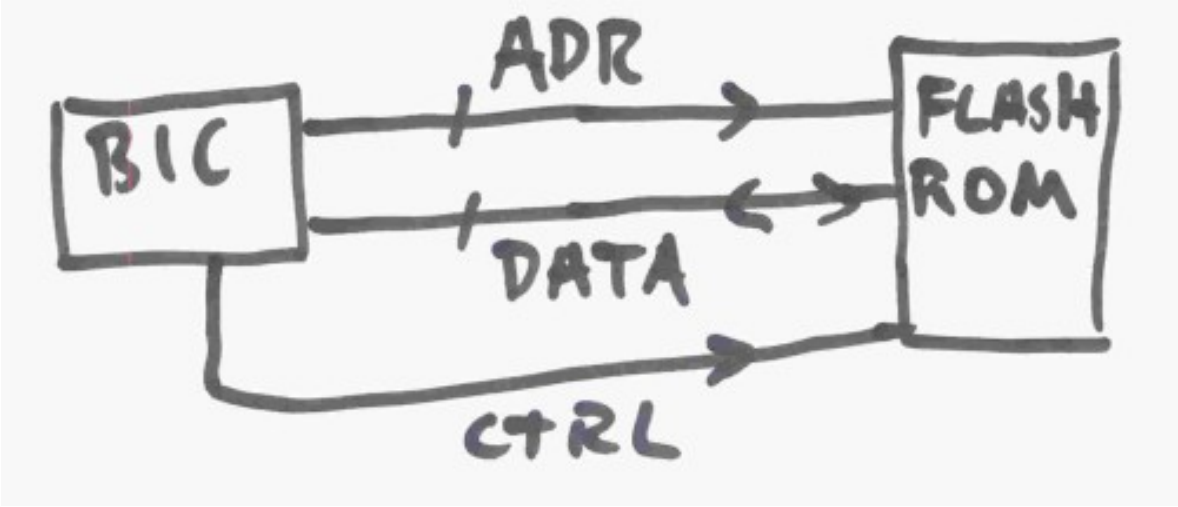
In-System-Programmierung (ISP) #1

1. CPLD, FPGA, Boot-ROMs, ...
2. kein Test, sondern ISP !
3. Firmware (*.svf, JAM, STAPL, IEEE 1532, …)



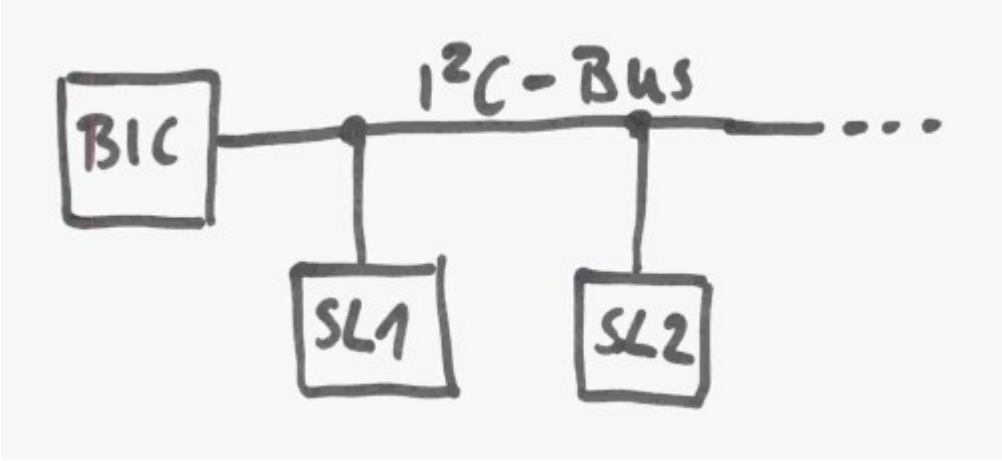
In-System-Programmierung (ISP) #2

1. ROM/FLASH-Speicher
2. kein Test, sondern ISP !
3. Firmware (*.bin, *.hex, *.s3, …)
4. Verifikation
5. Programmierzeiten (Länge Scanpfad, TCK Frequenz, Algorithmus, …)
6. separate Einspeisung WE-Signal via BSC



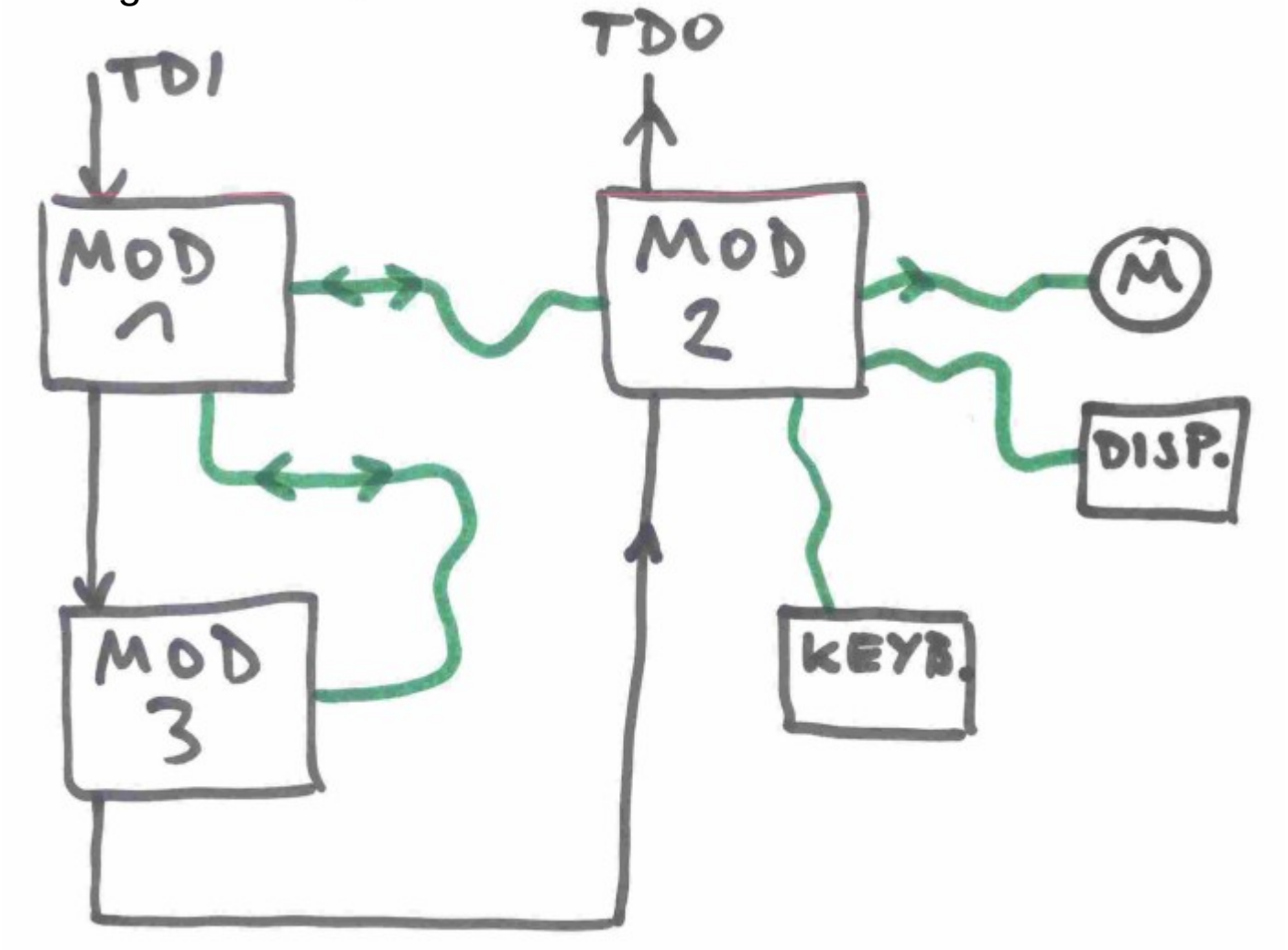
In-System-Programmierung (ISP) #3

1. serielle ROM/FLASH-Speicher
2. kein Test, sondern ISP !
3. Firmware
4. Microwire, I²C, SPI, ...



System-Test

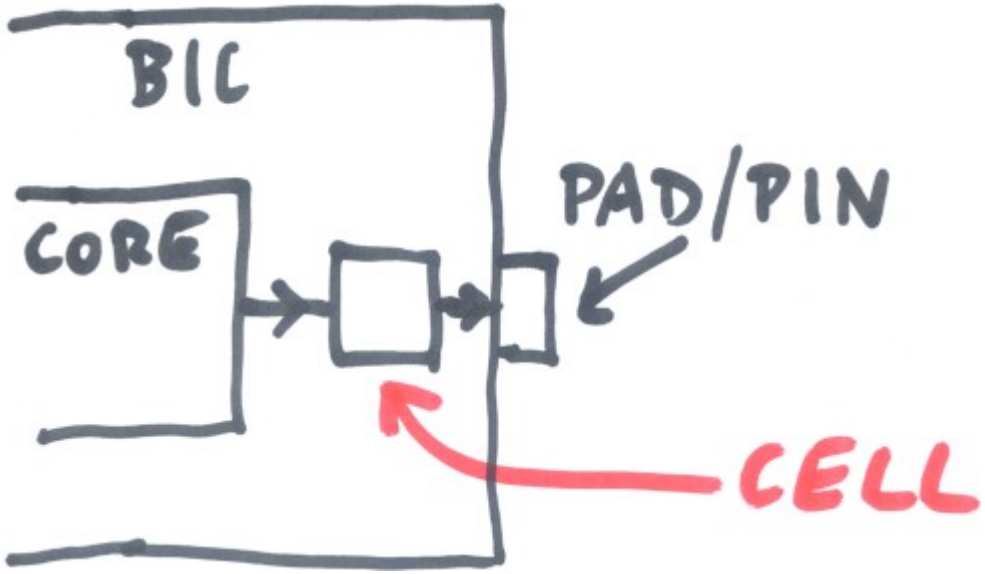
1. Verbindungen von Modulen/PCBA untereinander
2. Peripherie (Displays, Motoren, Tasten, Schalter, …)
3. Anwendung im Feld möglich



BS-Zellen #1

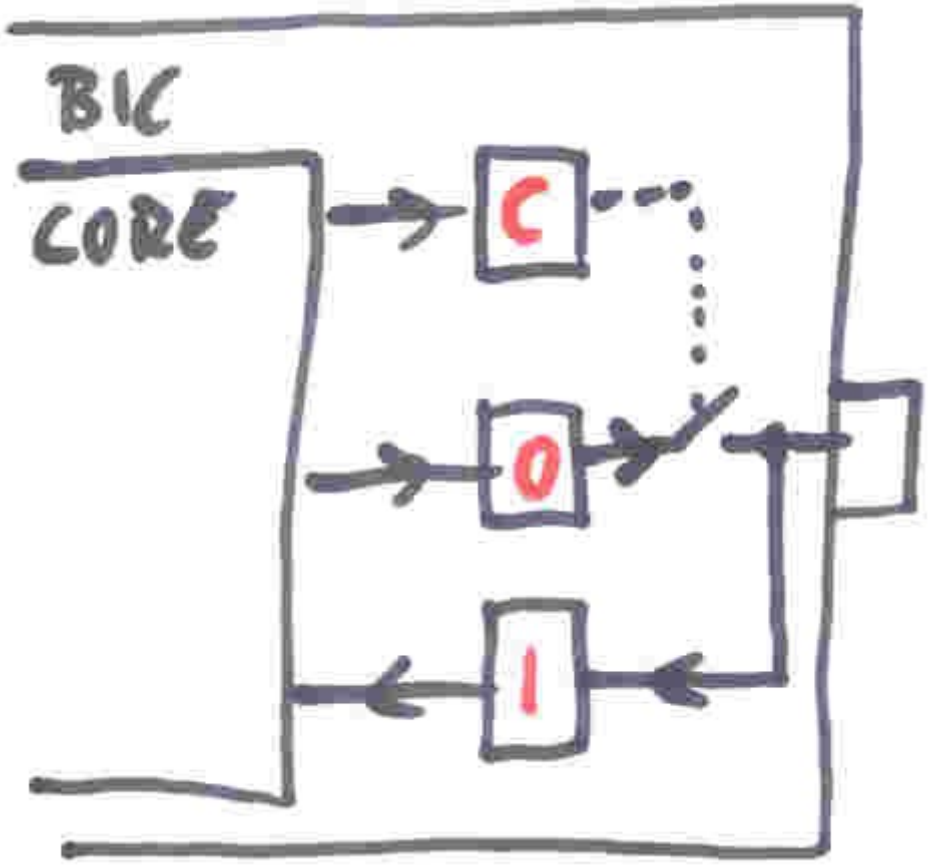
Zugriff auf Pad/Pin via sg. Boundary Scan Zelle

- Ausnahmen: VCC, GND, …
- verschiedene Typen von Zellen: BC_1 .. BC_7, ...



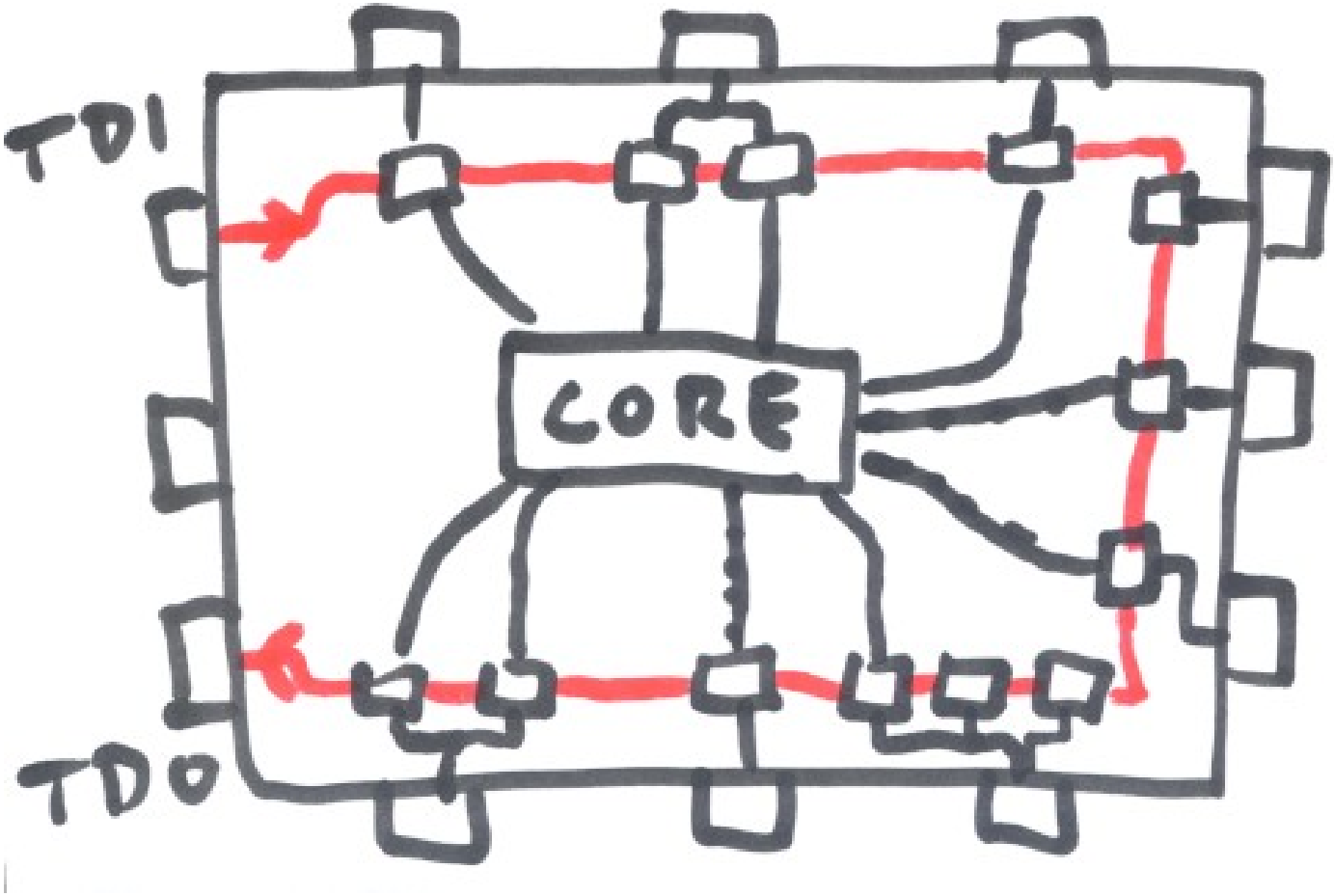
BS-Zellen #2

Mehrere Zellen in Kombination erlauben Bidir, Tristate, Weak0, Weak1 ...



BS-Zellen #3

Zellen virtuell am Rand des IC → Boundary Scan !



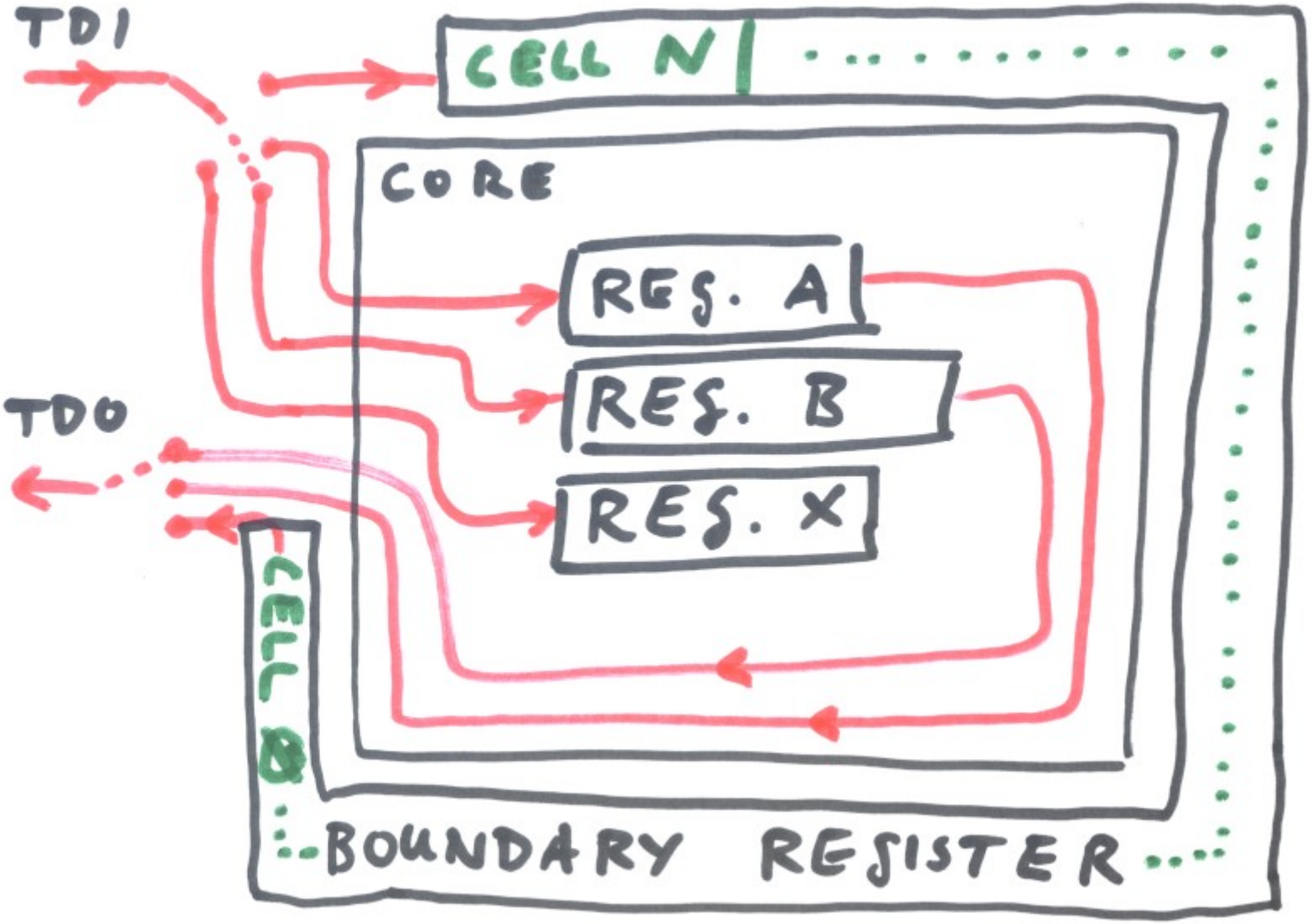
BS-Zellen #4

BSDL:

```
attribute REGISTER_ACCESS of sn74bct8240a : entity is
        "BOUNDARY (EXTEST, SAMPLE, INTEST, READBN, READBT, CELLTST),"  &
        "BYPASS   (BYPASS, HIGHZ, CLAMP, RUNT, TOPHIP)," &
        "BCR[2]   (SCANCN, SCANCT)" ;

attribute BOUNDARY_CELLS    of sn74bct8240a : entity is "BC_1";
attribute BOUNDARY_LENGTH   of sn74bct8240a : entity is 18;
attribute BOUNDARY_REGISTER of sn74bct8240a : entity is

"0  (BC_1, Y2(4), output3, X, 16, 1, Z)," & -- these outputs controlled
"1  (BC_1, Y2(3), output3, x, 16, 1, Z)," & -- by cell 16
"2  (BC_1, Y2(2), output3, X, 16, 1, Z)," &
"3  (BC_1, Y2(1), output3, X, 16, 1, Z)," &
"4  (BC_1, Y1(4), output3, X, 17, 1, Z)," & -- these outputs controlled
"5  (BC_1, Y1(3), output3, X, 17, 1, Z)," & -- by cell 17
"6  (BC_1, Y1(2), output3, X, 17, 1, Z)," &
"7  (BC_1, Y1(1), output3, X, 17, 1, Z)," &
"8  (BC_1, A2(4), input, X)," &
"9  (BC_1, A2(3), input, X)," &
"10 (BC_1, A2(2), input, X)," &
"11 (BC_1, A2(1), input, X)," &
"12 (BC_1, A1(4), input, X)," &
"13 (BC_1, A1(3), input, X)," &
"14 (BC_1, A1(2), input, X)," &
"15 (BC_1, A1(1), input, X)," &
--  "16 (BC_1, OE_NEG2, input, X)," &
"16 (BC_1, OE_NEG2, input, 1)," & -- safe value changed from x to 1 / MBL
"16 (BC_1, *, control, 1)," &
"17 (BC_1, OE_NEG1, input, 1)," &
--  "17 (BC_1, *, control, x)" ;
"17 (BC_1, *, control, 1)" ; -- safe value changed from x to 1 / MBL
```

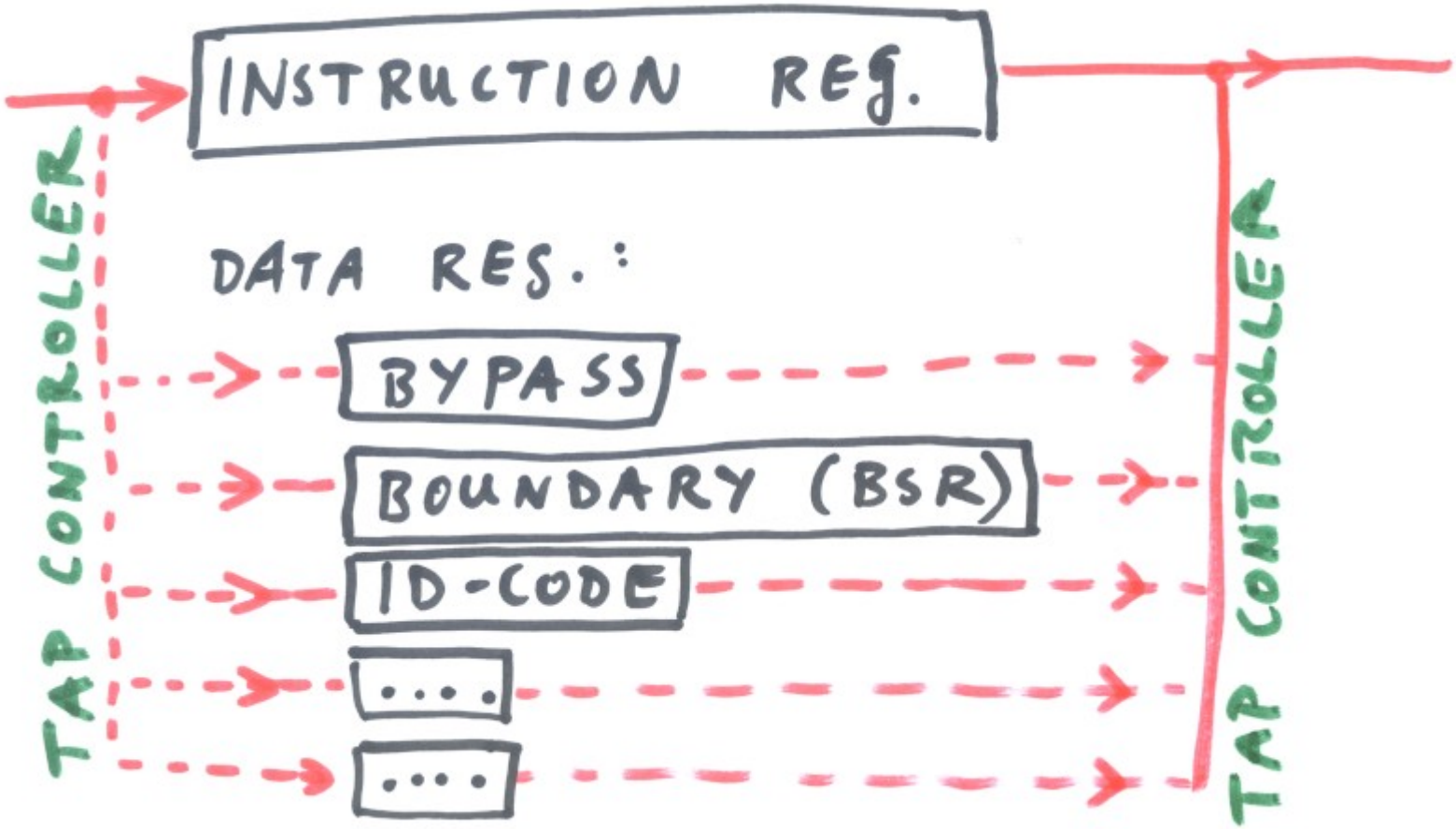
Register #1

In Reihe geschaltete Zellen formen Register:



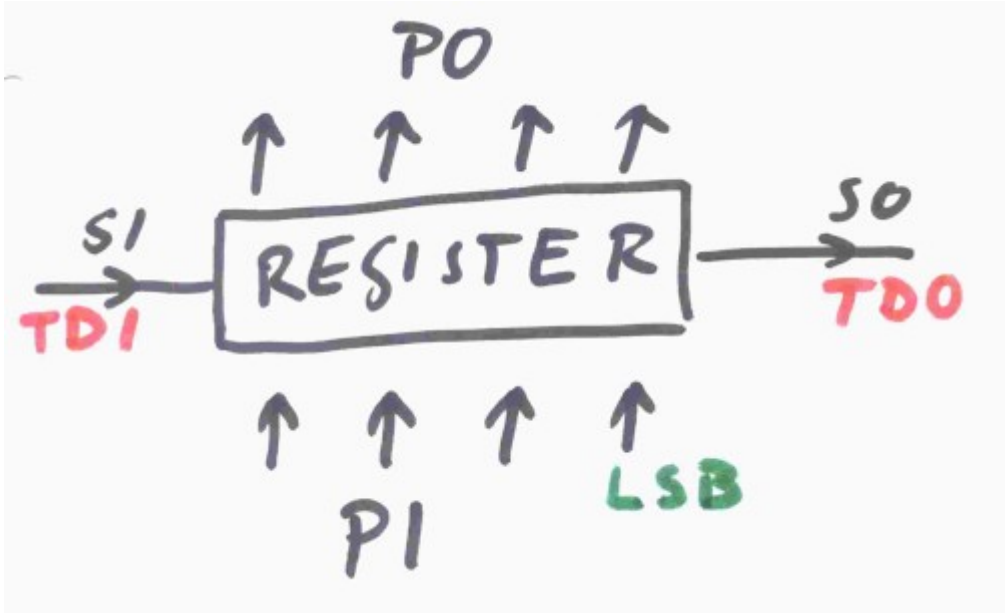
Register #2

Es gibt **ein** Befehlsregister und **mehrere** Datenregister:



Register #3

Serieller **und** paralleler Datentransfer:



PO / PI → parallel Output/Input

im BSR sind das Verbindungen zu Pins/Pads

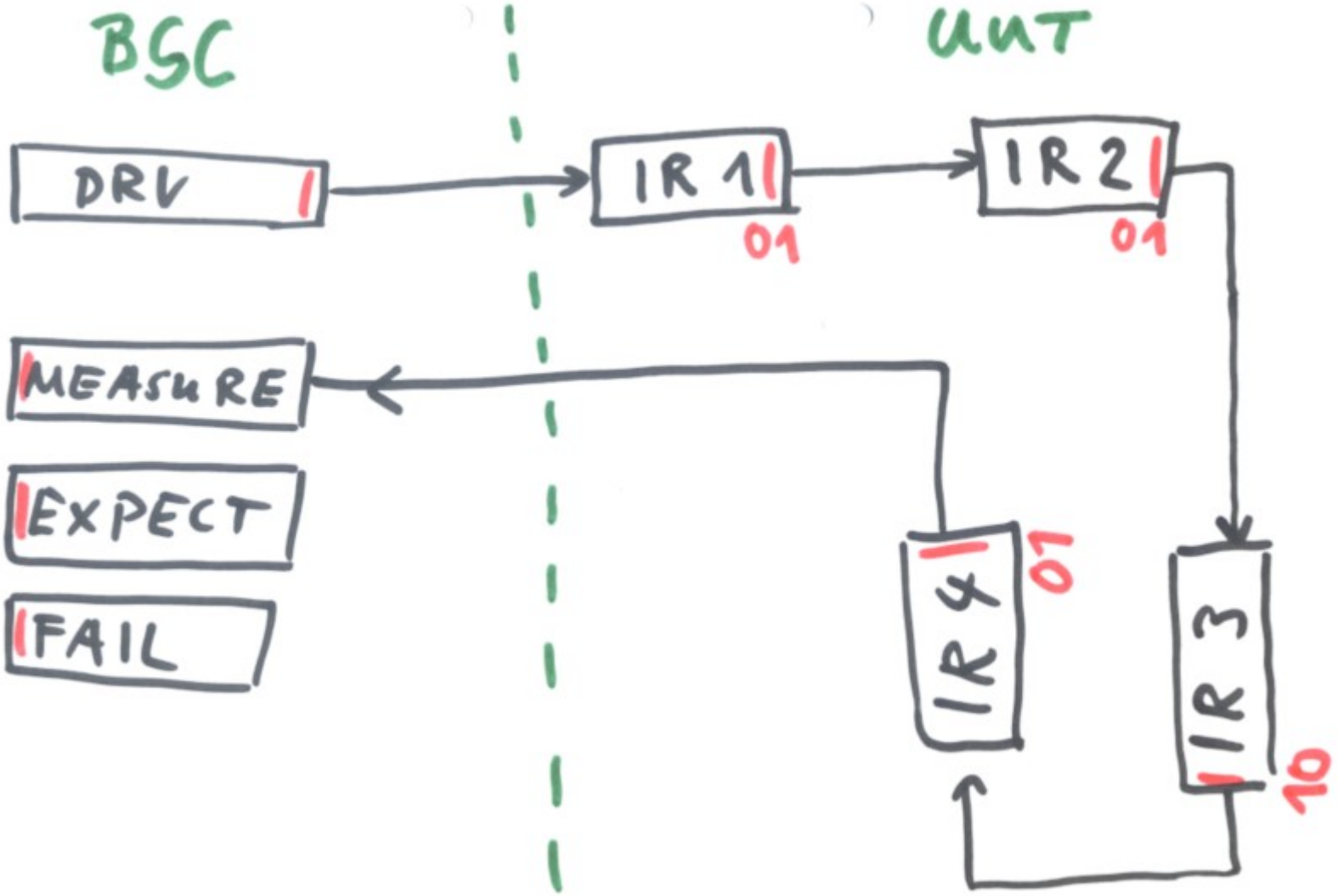
Register #4

BSDL:

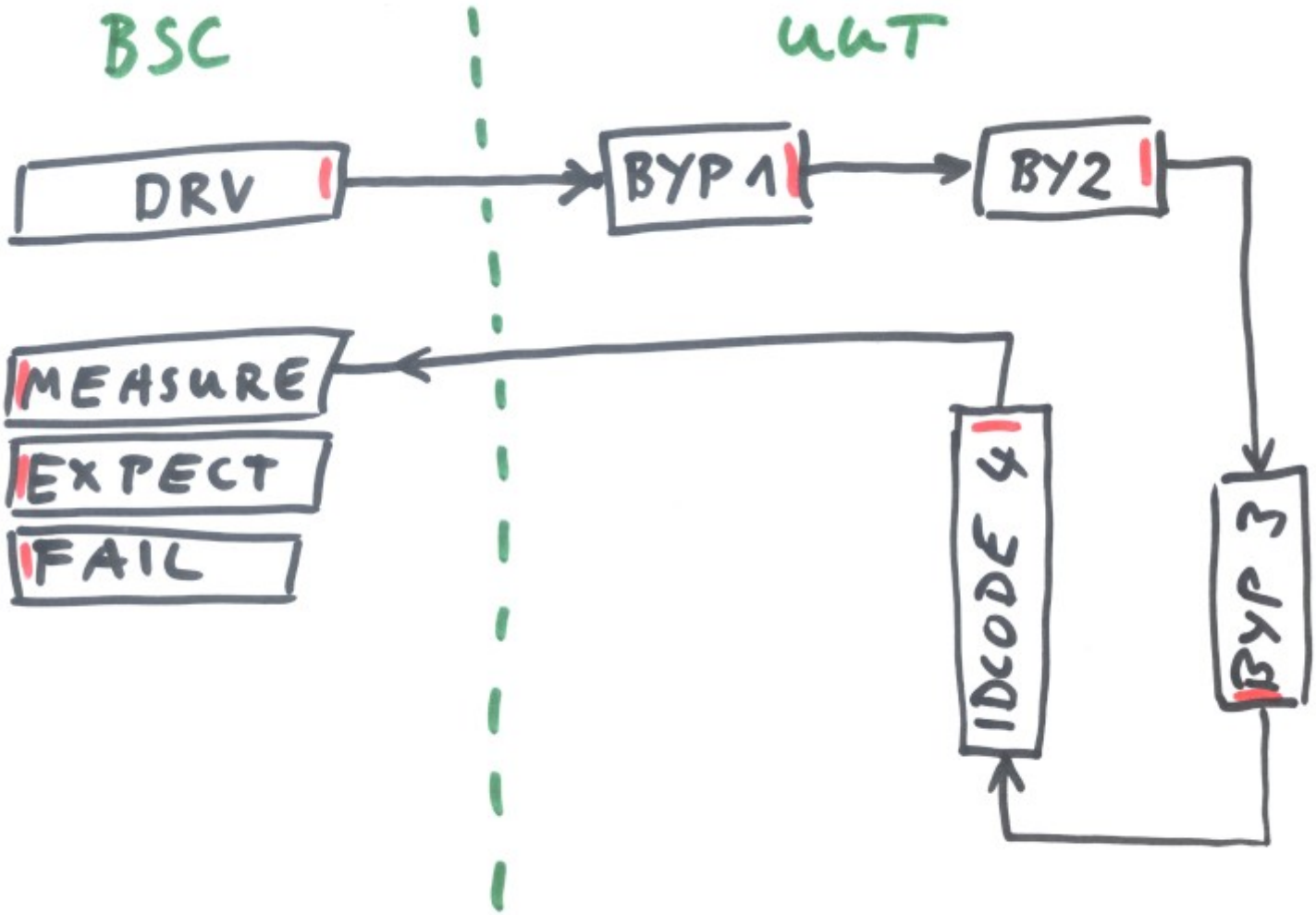
```
attribute INSTRUCTION_LENGTH of sn74bct8240a : entity is 8;
attribute INSTRUCTION_OPCODE of sn74bct8240a : entity is
        "EXTEST (00000000, 10000000), " &
        "BYPASS (11111111, 10000100, 00000101, 10001000, 00000001), " &
        "SAMPLE (00000010, 10000010), " &
        "INTEST (00000011, 10000011), " &
        "HIGHZ  (00000110, 10000110), " &  -- Bypass with outputs high-z
        "CLAMP  (00000111, 10000111), " &  -- Bypass with bs value
        "RUNT   (00001001, 10001001), " &  -- Boundary run test
        "READBN (00001010, 10001010), " &  -- Boundary read normal mode
        "READBT (00001011, 10001011), " &  -- Boundary read test mode
        "CELLTST(00001100, 10001100), " &  -- Boundary selftest normal mode
        "TOPHIP (00001101, 10001101), " &  -- Boundary toggle out test mode
        "SCANCN (00001110, 10001110), " &  -- BCR scan normal mode
        "SCANCT (00001111, 10001111)  " ;  -- BCR scan test mode

attribute INSTRUCTION_CAPTURE of sn74bct8240a : entity is "10000001";
attribute INSTRUCTION_DISABLE of sn74bct8240a : entity is "HIGHZ";
attribute INSTRUCTION_GUARD   of sn74bct8240a : entity is "CLAMP";

attribute REGISTER_ACCESS of sn74bct8240a : entity is
        "BOUNDARY (EXTEST, SAMPLE, INTEST, READBN, READBT, CELLTST)," &
        "BYPASS   (BYPASS, HIGHZ, CLAMP, RUNT, TOPHIP)," &
        "BCR[2]   (SCANCN, SCANCT)" ;

attribute BOUNDARY_CELLS    of sn74bct8240a : entity is "BC_1";
attribute BOUNDARY_LENGTH   of sn74bct8240a : entity is 18;
attribute BOUNDARY_REGISTER of sn74bct8240a : entity is

"0  (BC_1, Y2(4), output3, X, 16, 1, Z)," & -- these outputs controlled
"1  (BC_1, Y2(3), output3, x, 16, 1, Z)," & -- by cell 16
"2  (BC_1, Y2(2), output3, X, 16, 1, Z)," &
```

IR-Scan

Test der IR-Capture Werte und laden **aller** BICs mit Befehlen:



DR-Scan - IDCODE

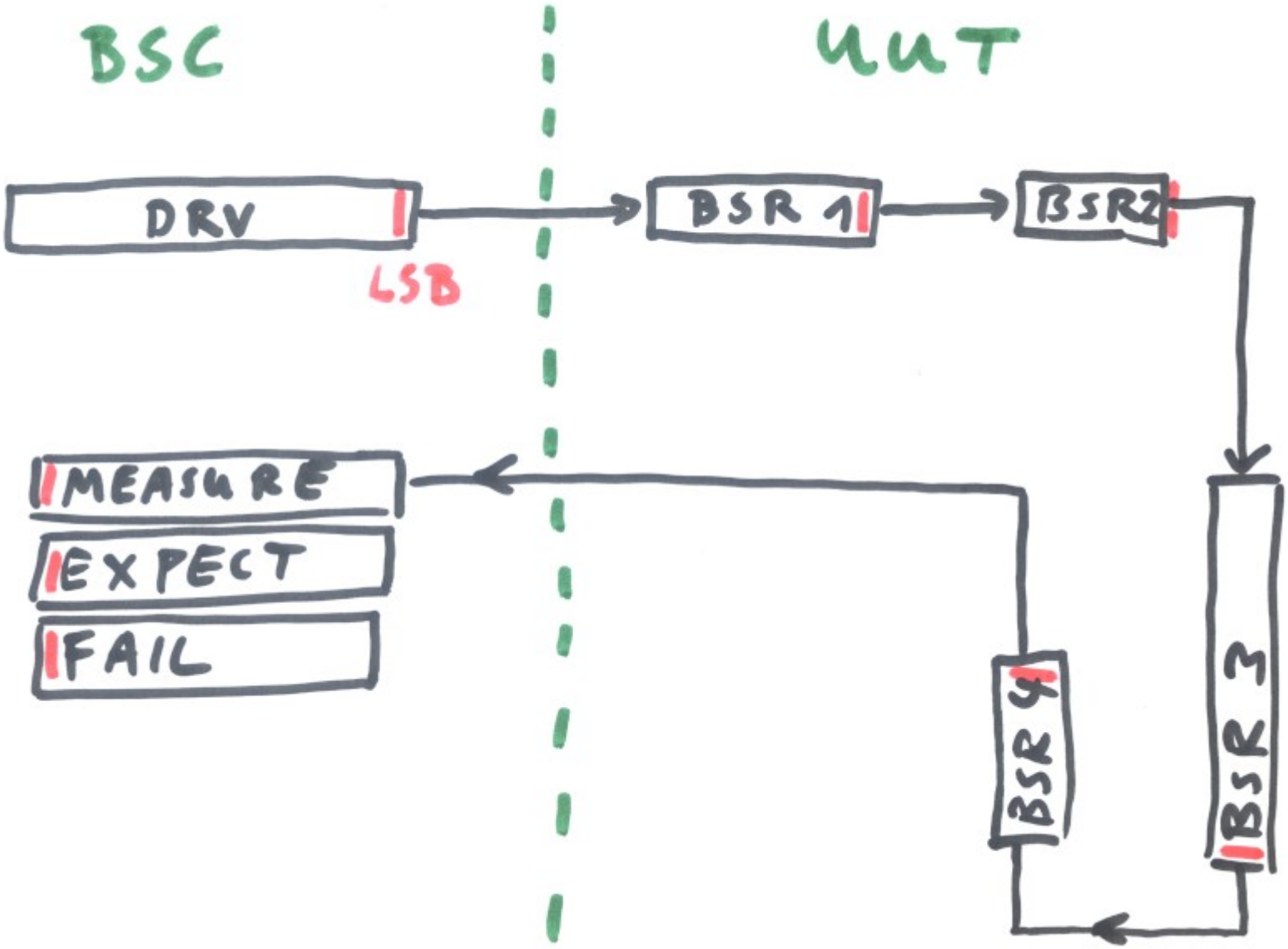


Boundary Scan Register

```
  attribute REGISTER_ACCESS of sn74bct8240a : entity is
          "BOUNDARY (EXTEST, SAMPLE, INTEST, READBN, READBT, CELLTST),"  &
          "BYPASS   (BYPASS, HIGHZ, CLAMP, RUNT, TOPHIP)," &
          "BCR[2]   (SCANCN, SCANCT)" ;

  attribute BOUNDARY_CELLS    of sn74bct8240a : entity is "BC_1";
  attribute BOUNDARY_LENGTH   of sn74bct8240a : entity is 18;
  attribute BOUNDARY_REGISTER of sn74bct8240a : entity is

  "0  (BC_1, Y2(4), output3, X, 16, 1, Z),"  & -- these outputs controlled
  "1  (BC_1, Y2(3), output3, x, 16, 1, Z),"  & -- by cell 16
  "2  (BC_1, Y2(2), output3, X, 16, 1, Z),"  &
  "3  (BC_1, Y2(1), output3, X, 16, 1, Z),"  &
  "4  (BC_1, Y1(4), output3, X, 17, 1, Z),"  & -- these outputs controlled
  "5  (BC_1, Y1(3), output3, X, 17, 1, Z),"  & -- by cell 17
  "6  (BC_1, Y1(2), output3, X, 17, 1, Z),"  &
  "7  (BC_1, Y1(1), output3, X, 17, 1, Z),"  &
  "8  (BC_1, A2(4), input, X),"   &
  "9  (BC_1, A2(3), input, X),"   &
  "10 (BC_1, A2(2), input, X),"   &
  "11 (BC_1, A2(1), input, X),"   &
  "12 (BC_1, A1(4), input, X),"   &
  "13 (BC_1, A1(3), input, X),"   &
  "14 (BC_1, A1(2), input, X),"   &
  "15 (BC_1, A1(1), input, X),"   &
--  "16 (BC_1, OE_NEG2, input, X)," &
  "16 (BC_1, OE_NEG2, input, 1)," & -- safe value changed from x to 1 / MBL
  "16 (BC_1, *, control, 1),"     &
  "17 (BC_1, OE_NEG1, input, 1)," &
--  "17 (BC_1, *, control, x)" ;
  "17 (BC_1, *, control, 1)" ; -- safe value changed from x to 1 / MBL
```

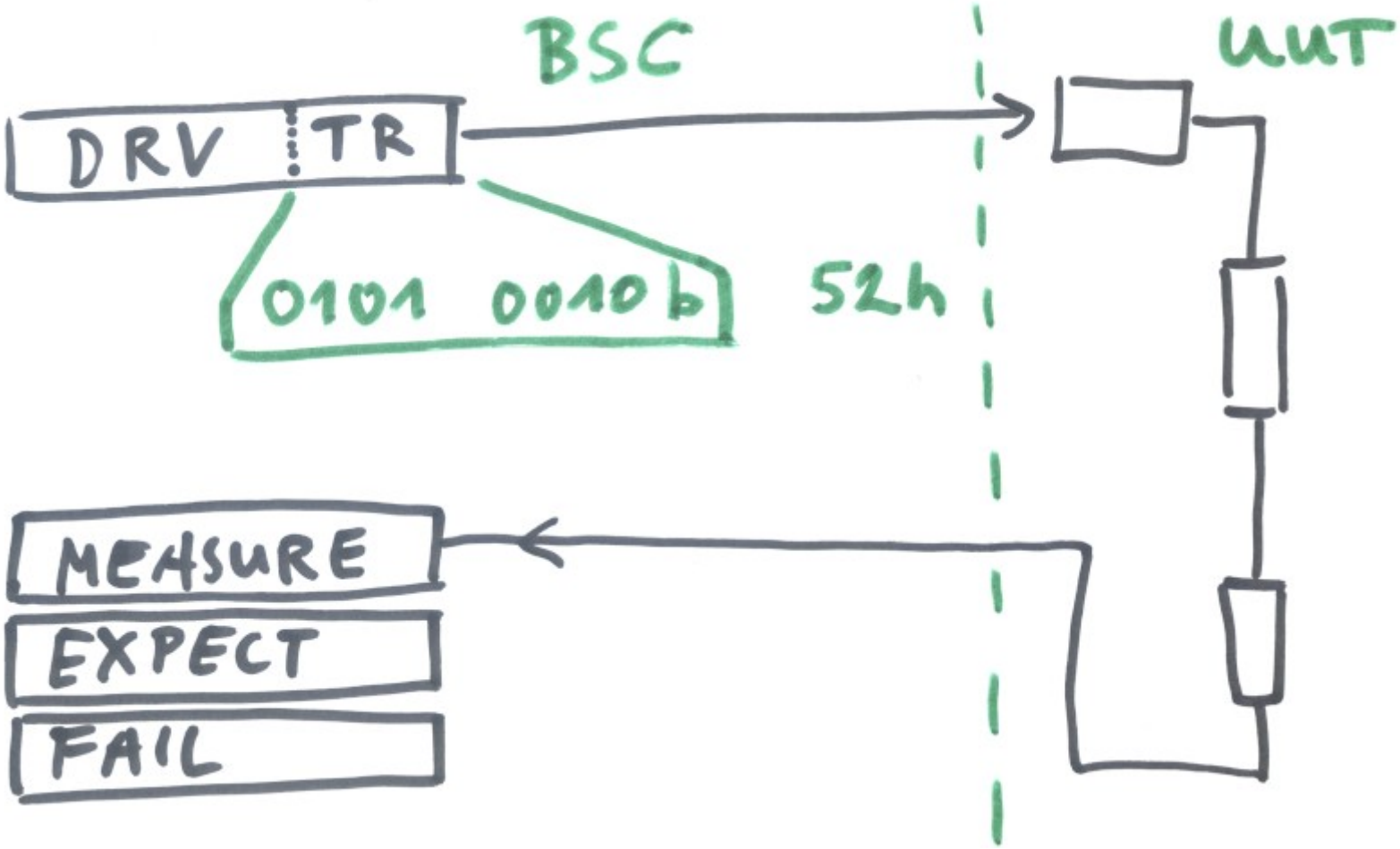
DR-Scan – Boundary Scan Register

Grundlage für alle intrusiven Tests.

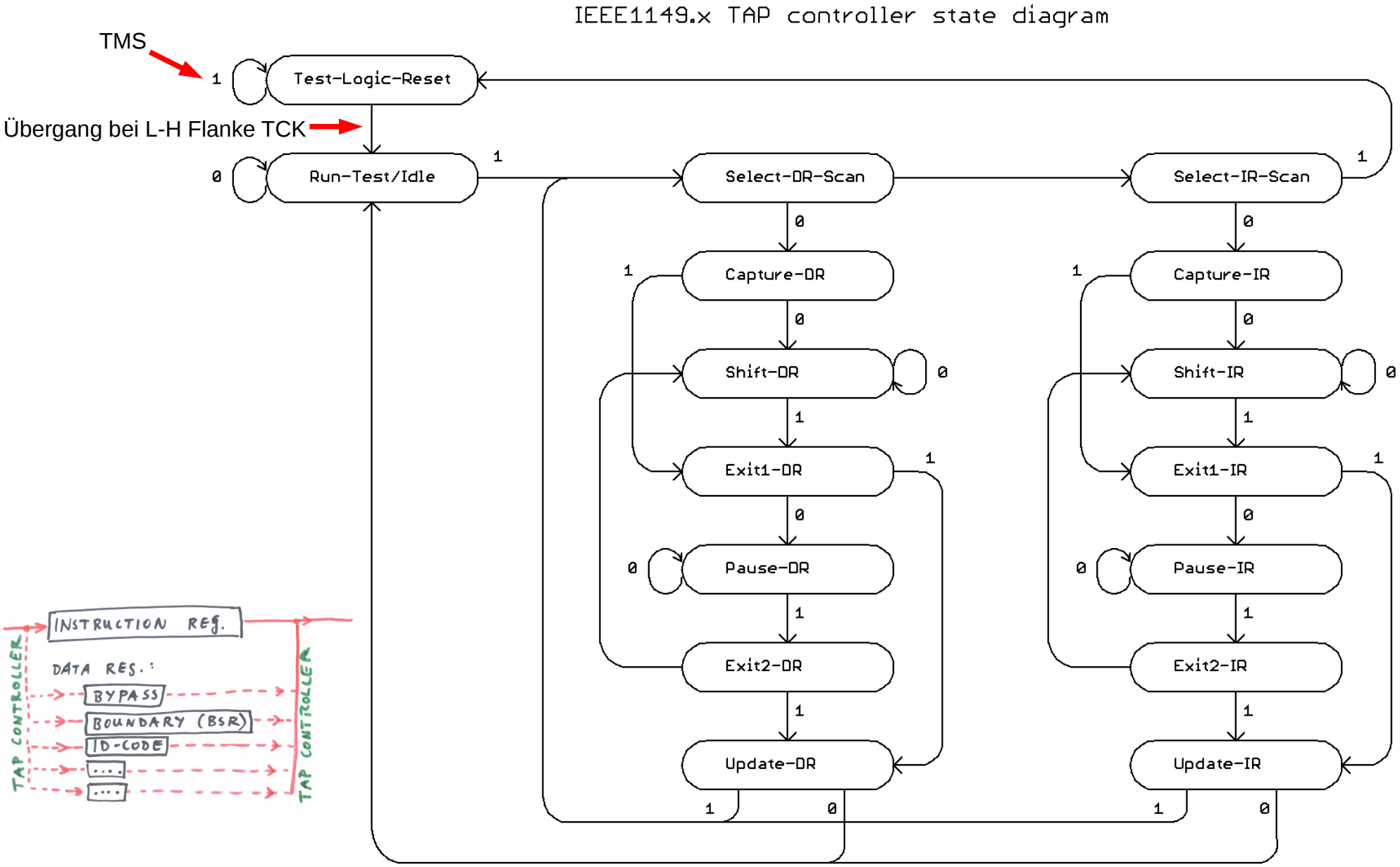


Trailer

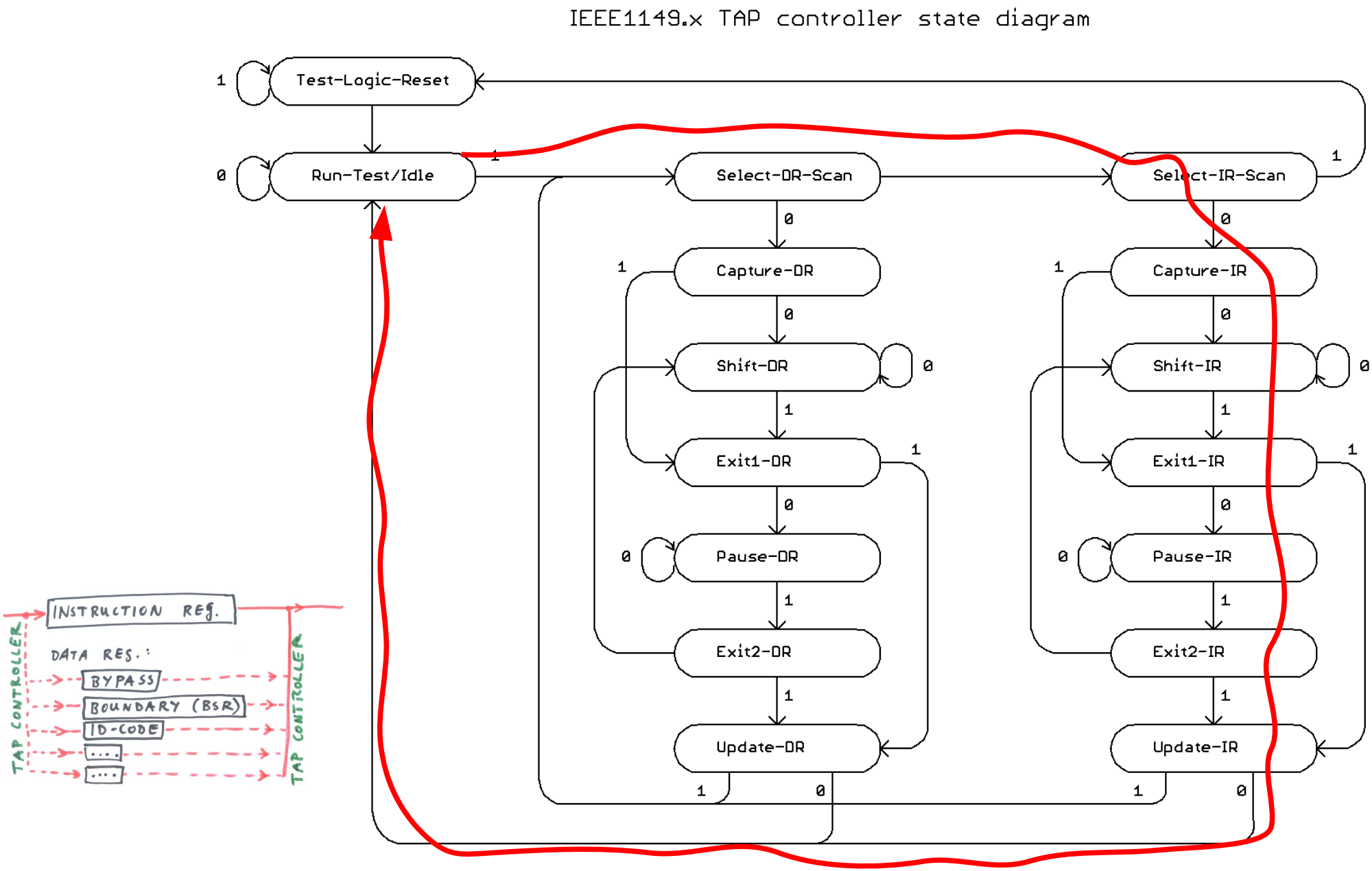
Trailer (TR) oder Checkbyte (CB) unterstützt Diagnose bei Fehlern im Scanpfad:



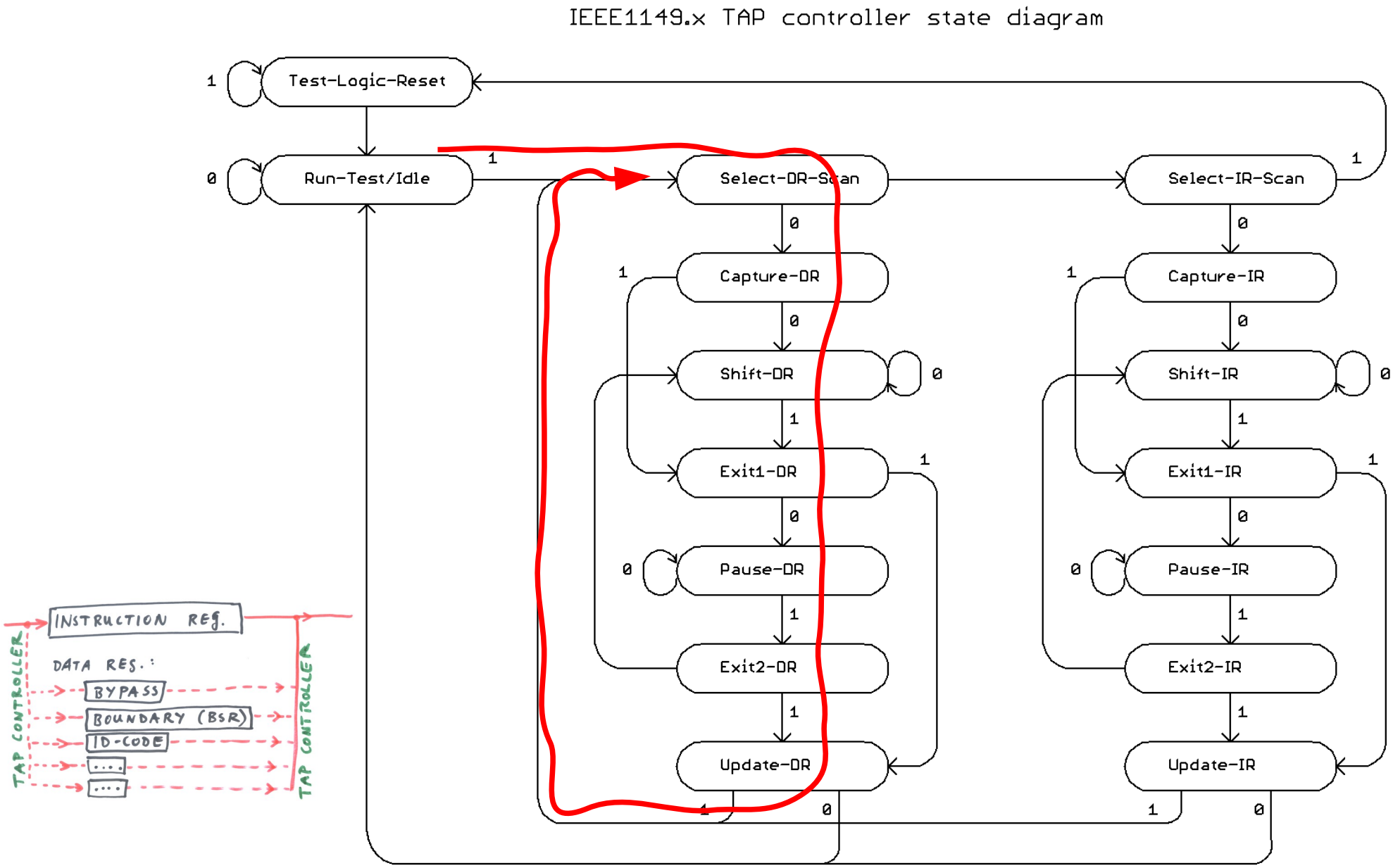
TAP Controller



TAP Controller: IR-Scan

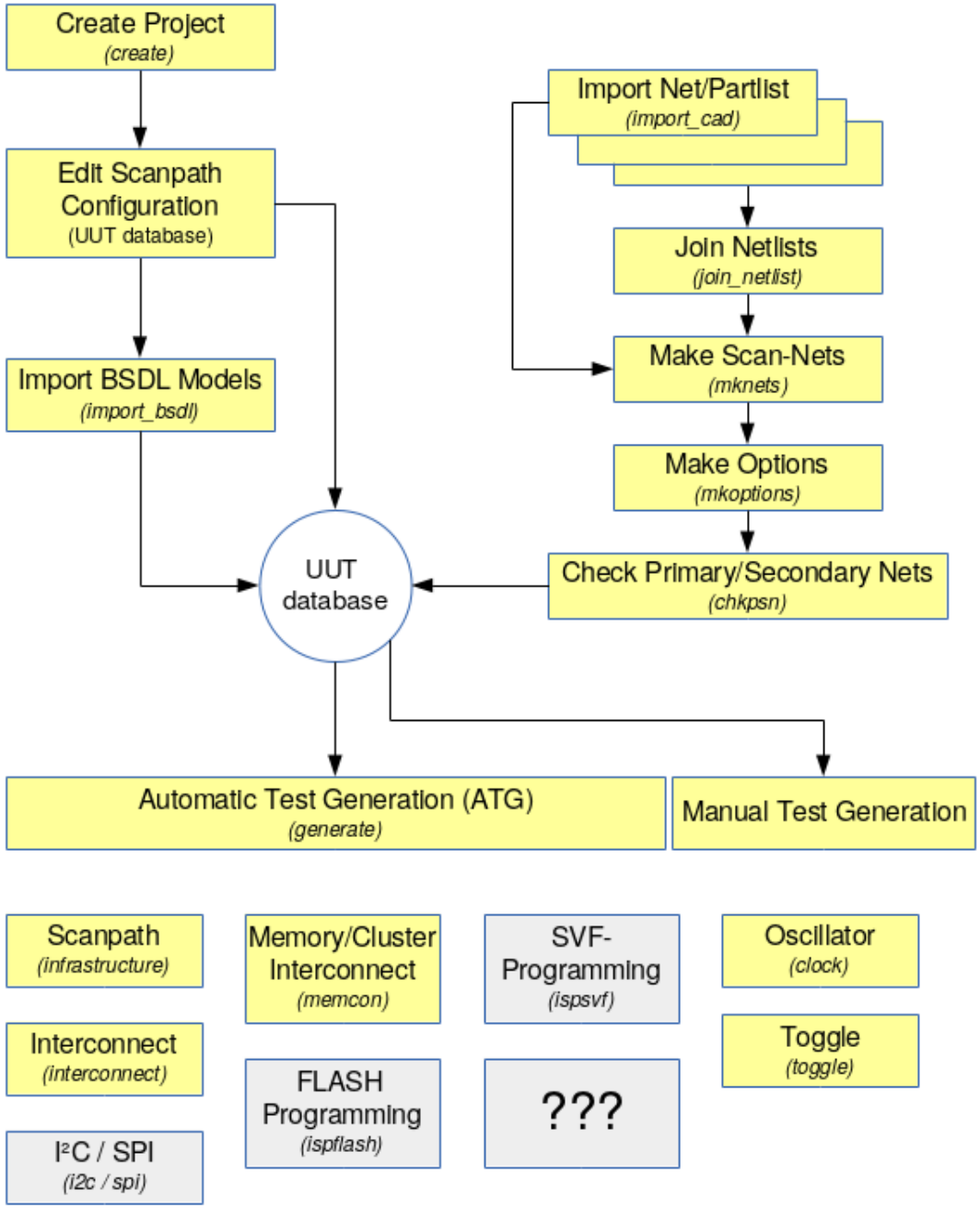


TAP Controller: DR-Scan



Erstellung von Tests #1

allgemeiner Ablauf für *System M-1*:



Erstellung von Tests #2

Projekt anlegen:

```
$ bsmcl create my_project
```

Scanpfade editieren in UUT-Datenbank (*.udb)

Erstellung von Tests #3

Scanpfade editieren in UUT-Datenbank (*.udb):

```
Section scanpath_configuration

 SubSection options
  on_fail power_down -- default is power_down. NOTE: similar commands: hstrst (hard+soft scanpath reset)
  frequency 0.1 -- unit is Mhz, use numbers like 0.1, 1, 4.5
  trailer_ir 01010010 -- 8 bit binary value -- equals 52h
  trailer_dr 01010010
  voltage_out_port_1 2.5 -- unit is V. supported values: 1.8  2.5  3.3
  voltage_out_port_2 3.3
  threshold_tdi_port_1 0.8  -- unit is V. range: 0..3.3
  threshold_tdi_port_2 0.8
 EndSubSection

 SubSection chain 1
    --UUT_TDI_1 / BSC_TDO_1
    --device, package, path to bsdl model, option [ remove_pin_prefix p ]
    --NOTE 1: use lower case letters for package names
    --NOTE 2: position 1 is closest to BSC TDO !

    IC801 pq208 models/xc3s400_pq208.bsd option remove_pin_prefix P
    IC802 vo20  models/xcf04s_vo20.bsd

    --UUT_TDO_1 / BSC_TDI_1
 EndSubSection

 SubSection chain 2
    IC1201 tq144 models/xc2c384_tq144.bsd
 EndSubSection

EndSection
```

Versionskontrolle !

Erstellung von Tests #4

BSDL Modelle importieren:

```
$ bsmcl import_bsdl my_project.udb
```

UUT Datenbank

Netz- und Partliste importieren:

```
$ bsmcl import_cad eagle6 cad/netlist.txt cad/partlist.txt main
```

scanfähige Netze erzeugen:

```
$ bsmcl mknets my_project.udb
```

Editieren Datei mkoptions.conf

```
Section connectors
 -- device A , device B
 --module_a_X1 module_b_X3
 --abc def
EndSection

Section bridges
 RN302 array 1-2 3-4 5-6 7-8
 RN303 array 1-2 3-4 5-6 7-8
EndSection
```

Versionskontrolle !

Optionen für scanfähige Netze generieren:

```
$ bsmcl mkoptions my_project.udb
```

Erstellung von Tests #5

Optionen editieren: my_project.opt

```
Section LED0 class NR -- bscan cluster
  -- D401 NA none LED5MM K
  -- IC303 NA SN74BCT8240ADWR SOIC24 10 Y2(4) | 0 BC_1 OUTPUT3 X 16 1 Z
  -- JP402 NA MON1 2X20 24
  -- RN302 NA 2k7 SIL8 1
 SubSection secondary_nets
  Net LED0_R
  -- IC301 NA XC9536 PLCC-S44 2 PB00_00 | 107 BC_1 INPUT X | 106 BC_1 OUTPUT3 X 105 0 Z
  -- JP402 NA MON1 2X20 26
  -- RN302 NA 2k7 SIL8 2
 EndSubSection
EndSection
```

default

```
Section RST class NA -- single bscan net
  -- IC300 NA XC9536 PLCC-S44 25 PB01_16 | 5 BC_1 INPUT X | 4 BC_1 OUTPUT3 X 3 0 Z
  -- JP403 NA MON2 2X20 40
  -- R402 NA 2k7 0207/10 2
  -- RN301 NA 8x220 SIL9 4
EndSection

Section SCL class PU -- single bscan net
  -- IC301 NA XC9536 PLCC-S44 29 PB01_12 | 17 BC_1 INPUT X | 16 BC_1 OUTPUT3 X 15 0 Z
  -- JP403 NA MON2 2X20 8
  -- R401 NA 160 0207/10 2
  -- RN300 NA 8x1k8 SIL9 4
EndSection
```

Versionskontrolle !

Erstellung von Tests #6

`Optionen editieren: my_project.opt`

Netzklassen:

NA (not assigned) → Netz vollständig von der Testgenerierung ausgeschlossen (default)

NR (no restriction) → Netz kann ohne Einschränkung getestet werden

PU (pull-up) → Netz hat Pull-Up-Verhalten

PD (pull-down) → Netz hat Pull-Down-Verhalten

DH (drive-high) → Netz soll permanent H-getrieben werden (sofern scanfähiger Treiber vorhanden)

DL (drive-low) → Netz soll permanent L-getrieben werden (sofern scanfähiger Treiber vorhanden)

EH (expect high) → Auf dem Netz soll permanent H erwartet werden (unabhängig von der Signalquelle)

EL (expect low) → Auf dem Netz soll permanent L erwartet werden (unabhängig von der Signalquelle)

Erstellung von Tests #7

Optionen von primären/sekundären Netzen prüfen

```
$ bsmcl chkpsn my_project.udb
```

Test generieren:

```
$ bsmcl generate my_project.udb infrastructure my_scanpath_test
$ bsmcl generate my_project.udb interconnect my_intercon
$ bsmcl generate my_project.udb memconnect ram_1 IC202 ...
```

Test compilieren, laden, ausführen:

```
$ bsmcl compile my_project.udb my_scanpath_test
$ bsmcl load my_scanpath_test
$ bsmcl run my_scanpath_test
```

Erstellung von Tests #8

Automatisierung via UNIX/Linux Shell-Skripte:

```
# infrastructure and interconnect tests
bsmcl generate $database_default infrastructure infra
bsmcl compile $database_default infra

bsmcl generate $database_default interconnect intercon
bsmcl compile $database_default intercon

bsmcl generate $database_ram_out_1 memconnect ram_out_1 IC602 models/AS6C4008.txt NDIP32
bsmcl compile $database_ram_out_1 ram_out_1

bsmcl generate $database_ram_out_2 memconnect ram_out_2 IC603 models/AS6C4008.txt NDIP32
bsmcl compile $database_ram_out_2 ram_out_2

bsmcl generate $database_ram_in memconnect ram_in IC601 models/AS6C4008.txt NDIP32
bsmcl compile $database_ram_in ram_in

#bsmcl generate $database_clock_test clock clk_slow IC801 154 10 1
#manually modified for intrusive mode
bsmcl compile $database_clock_test clk_slow

#bsmcl generate $database_clock_test clock clk_master IC801 181 10 0.1
#manually modified for intrusive mode
bsmcl compile $database_clock_test clk_master

#cluster tests
bsmcl compile $database_default cluster_digital
bsmcl compile $database_default cluster_analog

#vccio
bsmcl compile $database_default vccio

#LED - requires vccio set to 3V3 and a very low tck frequency
#bsmcl generate $database_vccio_3v3 interconnect intercon_slow
bsmcl compile $database_vccio_3v3 intercon_slow

bsmcl load infra
bsmcl load intercon
bsmcl load ram_out_1
bsmcl load ram_out_2
bsmcl load ram_in
bsmcl load cluster_digital
bsmcl load cluster_analog
bsmcl load clk_slow
bsmcl load clk_master
bsmcl load vccio
bsmcl load intercon_slow

bsmcl run infra
bsmcl run intercon
```

Versionskontrolle !

Ausführung von Tests #1

Einsatz in Entwicklung / Reparatur:

```
Test FAILED! Diagnosis:
failed scanpath    : 1
step id (dec)      : 7
sxr length (dec)   : 242 (one-based)
sxr fail pos (dec): 9 (one-based)
scan type          : SDR
device position    : 3
device name        : IC303
register           : BOUNDARY
failed bit pos.    : 8 (zero-based)
expected           : HIGH
net class          : PU

secondary net      : CT_D0

JP402 pin 2
IC303 pin 15
IC302 pin 3

primary net        : D0

RN401 pin 2
JP406 pin 2
IC203 pin 11
IC202 pin 11
IC201 pin 13
IC200 pin 13

stuck at LOW or Pull-Up resistor missing !
```

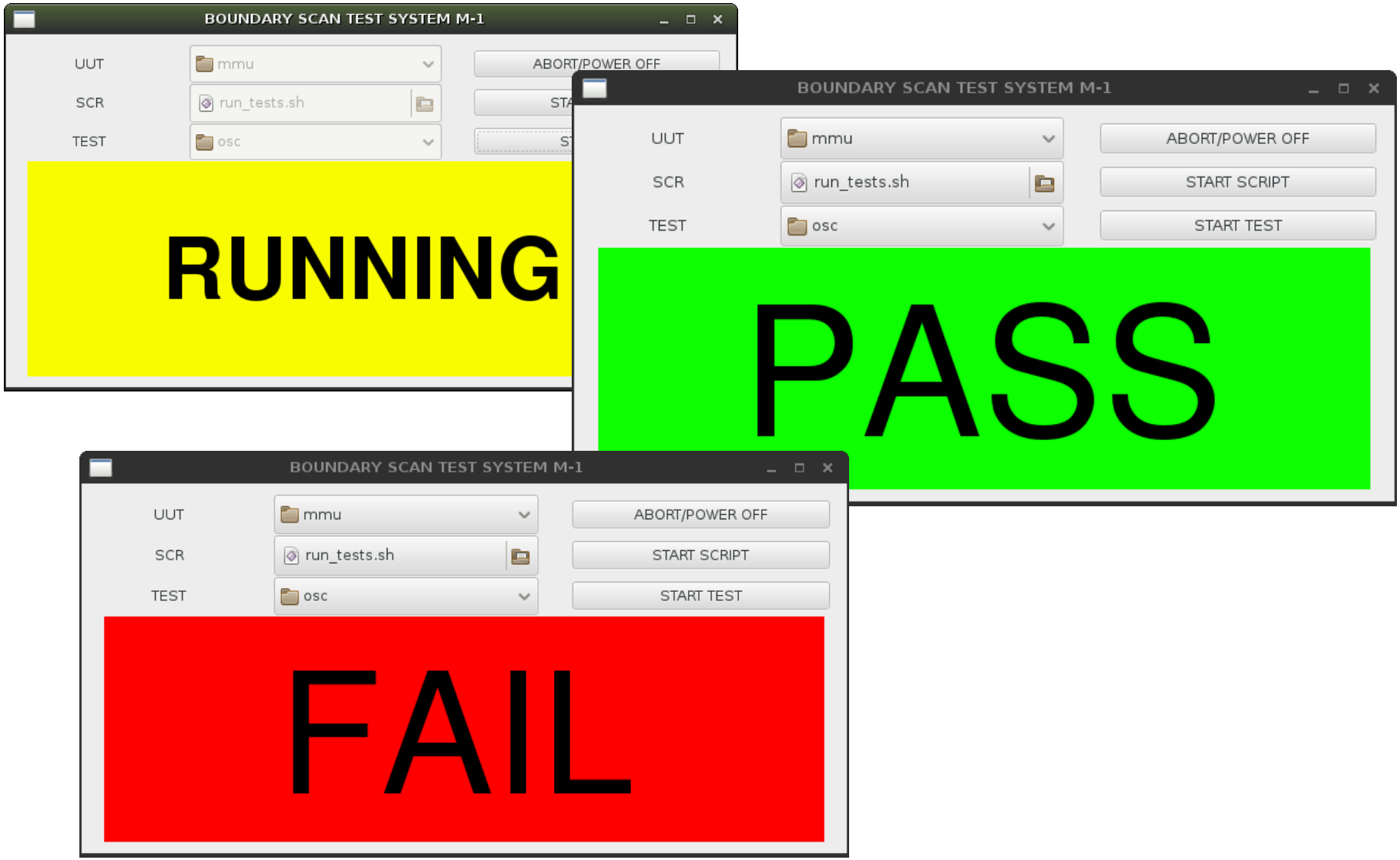
weitere Möglichkeiten:

- Einzelschritt TCK-Mode
- Einzelschritt IR/DR-Scan
- Breakpoints
- Batch/Script

siehe http://www.blunk-electronic.de/bsm/System_M-1_Manual_de.pdf

Ausführung von Tests #2

Einsatz in Fertigung: GO/NOGO - Test



http://www.blunk-electronic.de/bsm/System_M-1_Manual_de.pdf

Literatur & Quellen

1. The Institute of Electrical and Electronics Engineers, Inc. , 3 Park Avenue, New York, NY 100165997, USA; IEEE Std 1149.12001
   “IEEE Standard Test Access Port and Boundary Scan Architecture”
2. Parker, 2015, The Boundary-Scan Handbook

3. http://www.blunk-electronic.de/bsm/Boundary_Scan_Basics.pdf
4. http://www.blunk-electronic.de/bsm/how_to_test.pdf
5. http://www.blunk-electronic.de/bsm/System_M-1_Manual_en.pdf

https://www.youtube.com/watch?v=Y_dfg8h_yEY

Links

PCB Herstellung:

Jenaer Leiterplatten

www.jlp.de

Entwicklung und Bestückung von Kleinserien:

www.blunk-electronic.de

BLUNK ELECTRONIC

Danke für Ihre Aufmerksamkeit !