IEEE Std P1149.1-2012
VTS Maui, Hawaii

CJ Clark, Intellitech

(Note: P1149.1-2012 became ratified as 1149.1-2013)

Intellitech

Call for participation & WG formed at ITC 2009

Elections 12/2010

- CJ Clark, Intellitech, Chair
- Carol Pyron, Freescale, Vice Chair
- Carl Barnhart, SiliconAid, Editor
- Bill Tuthill , Raytheon, Secretary

Great group of dedicated users/IC/DFT/ICT
Approx 19-23 people on weekly conferences
Tuesday call - 10:30-12PM EST
Friday - 11:30-1PM EST

IEEE 1149.1WG website

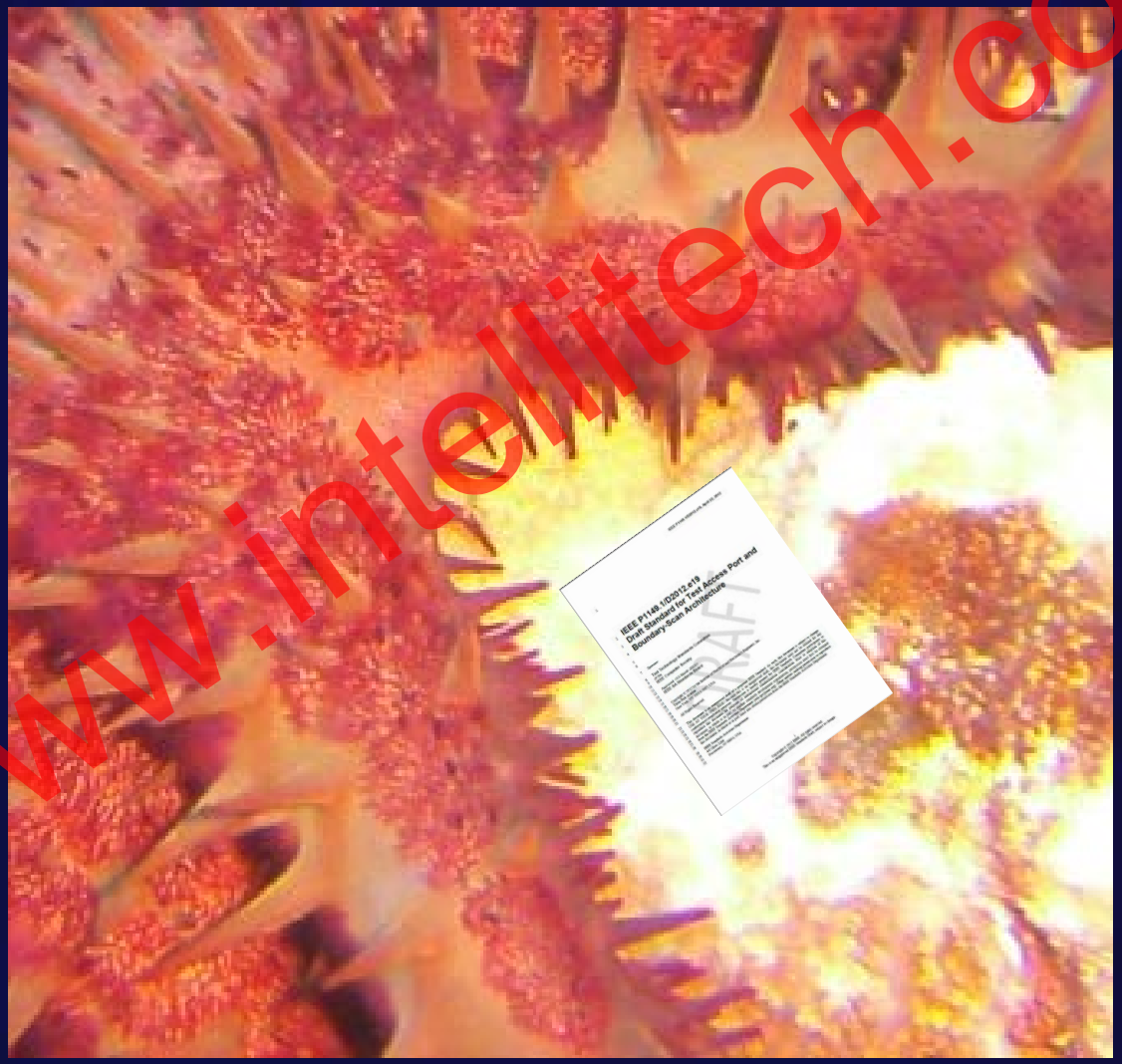
http://grouper.ieee.org/groups/1149/1/

Presentations on material used

For development of standard

Why talk about 1149.1 in Hawaii?

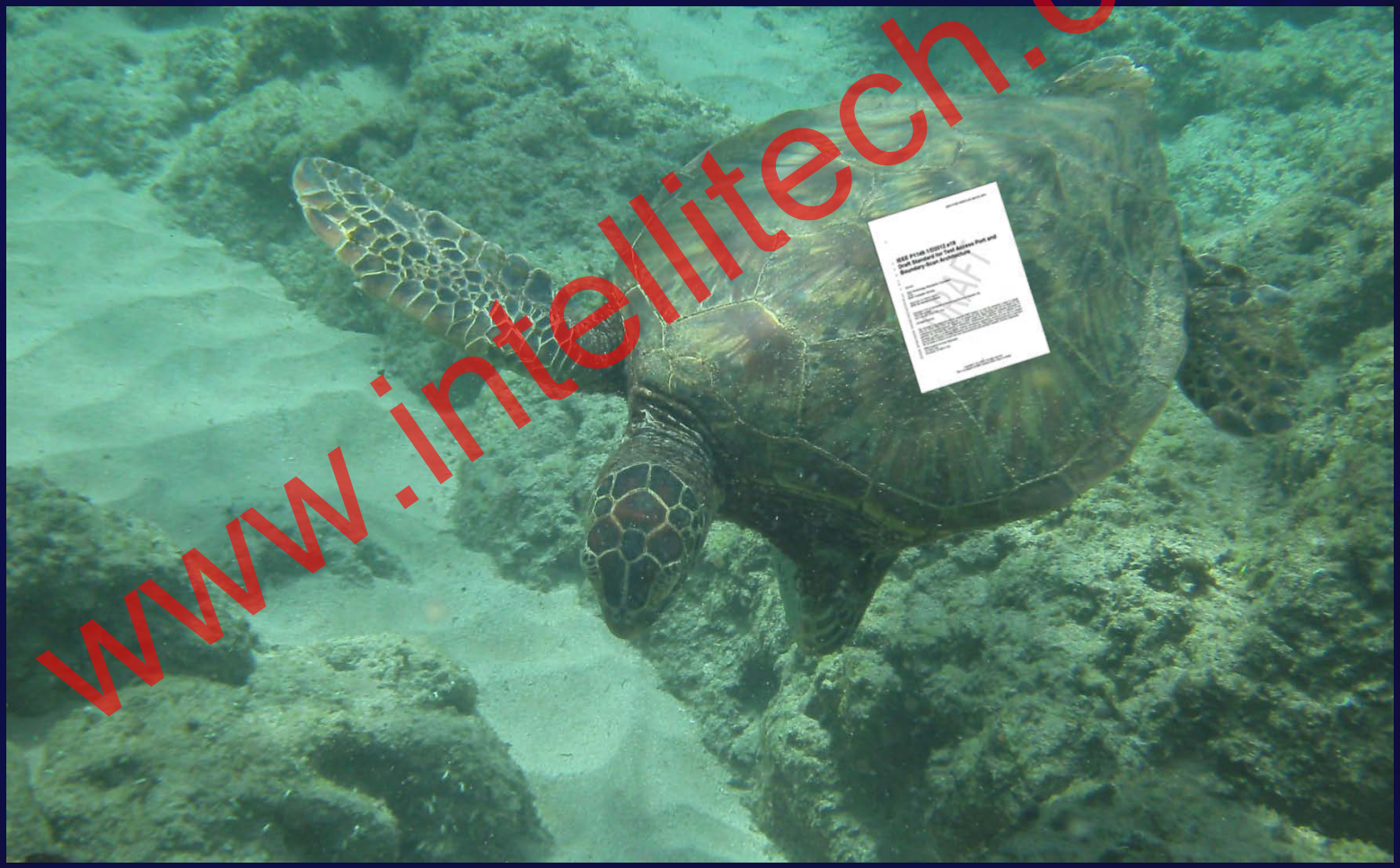
Test ocean readiness of 1149.1 standard ☺



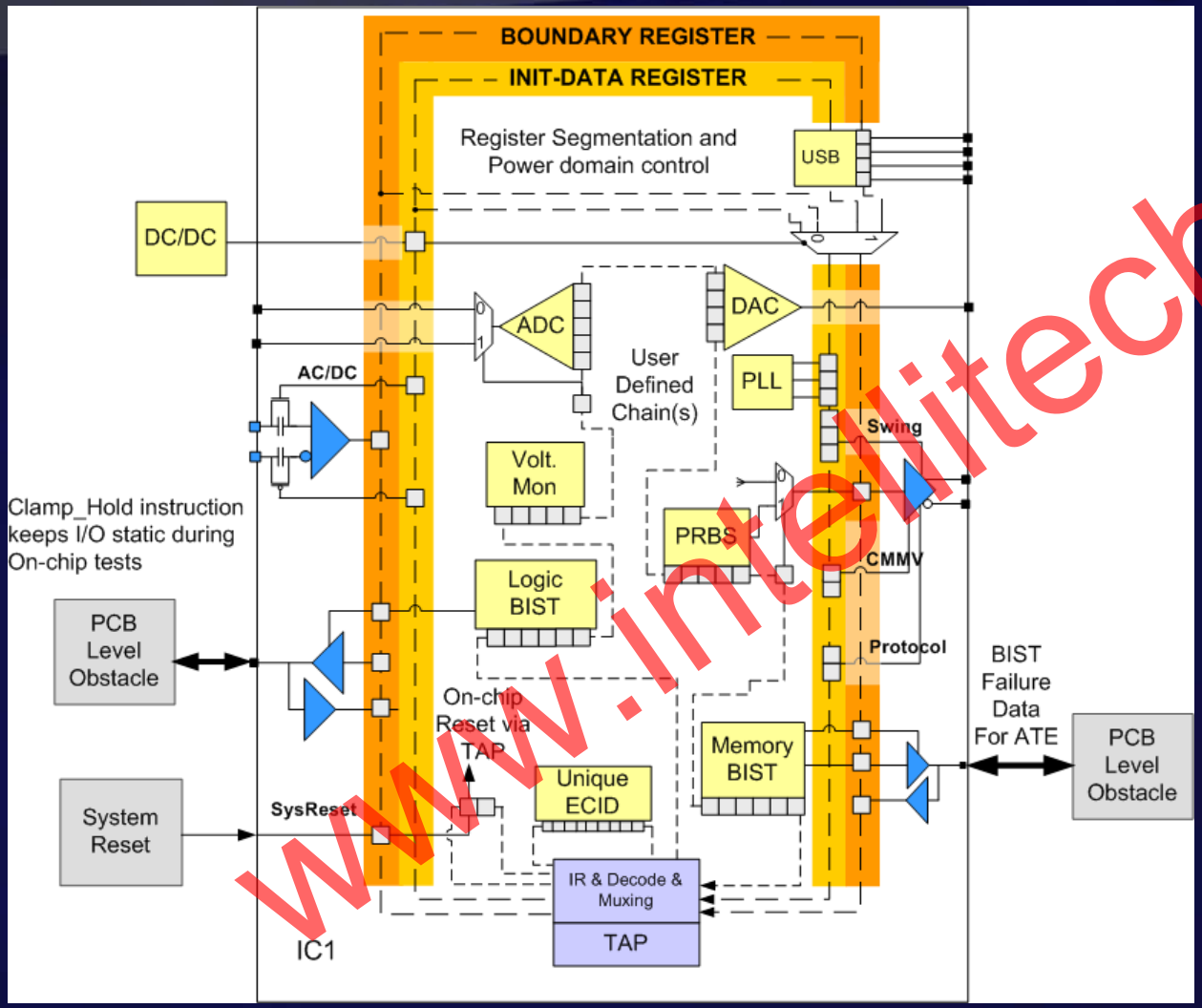
Spotted Moray bite test ☺

Green Sea Turtle (Kame)

Scientific Test Ride ☺



P1149.1-2012 at a glance



Six new instructions:

Clamp_hold/release

Init_setup/Init_run

IC_Reset

ECIDCode

New BSDL attributes

IP register description files

Procedure Description Language

IP Vendor

I make silicon instruments (IP) how do I?

a) Document how to use it

b) Document diagnostics

c) Demonstrate it was verified?

d) Design a re-usable JTAG interface?

Ex. SERDES PRBS

Verilog is not enough

IC Designer

I need silicon instruments but don't know how to operate the IP via JTAG

a) how can I test and characterize?

How do I exonerate my chip as not the problem for board test?

How do I reduce support

Enable my customers to do on-board tests (ex test DDR3 at-speed)?

System Designer

I need on-chip and ecosystem tests

I want to characterize board/FR4 early before software has functional tests

Datasheet isn't enough info

How do I run this on-chip thing?

How do I trouble shoot it?

How do I get EMS/Board test up to speed quickly?

Design Specific Test Data Registers exist in IEEE 1149.1 standard since 1990 but...

BSDL lacked definition language for the register fields below chip level

2001 BSDL lacked hierarchy & re-use

Since 1990s many Vendor/Proprietary BSDL extensions developed

BSDL extensions and languages supported high level operation (vendor specific) Classic tower of Babel problem

```
attribute REGISTER_NAME of exinit : entity is
    "mbist-csr[1]     (INIT_DATA[101,101]),"&
    "alg[5]       (INIT_DATA[205,201]),"&
    "done[1]        (INIT_DATA[101,101]),"&
    "Status[2]      (INIT_DATA[101,100]),"&
    "fail_row[8]    (INIT_DATA[117,110]),"&
    "fail_col[8]    (INIT_DATA[127,120])";
```

```
-- format register/bus:MNEMONIC (bit pattern,
      data, data)
```

```
attribute TDI_MNEMONIC of exinit : entity is
    "mbist-csr:Start    (1)," &
    "mbist-csr:Stop     (0)" &
```

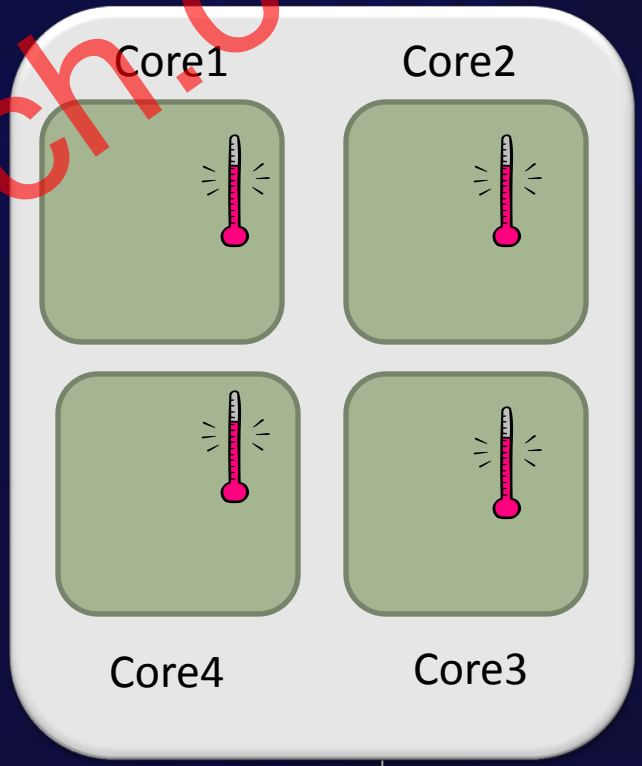
Dotted hierarchy

```
setTDI  u1.mbist-csr start
setTDI  u1.alg          walk1
drscan
runtest 10000
set result [getTDO status]
If  {$result != pass}
   puts "memorybist failed"
```

Re-usable Script to operate on Mbist registers

Unified Procedural Description Language

```
# vendor supplied reg to temp conversion
proc  Reg2Temp { $regval $CorF }  {
...
}
# this proc returns a temperature  and
# high level warnings could be specified
iProc  -export init-setup-temp-check { } {
iRead tempreg
iApply
set val  [iGet tempreg]
# convert reg value to temperature in Celsius
set  temp [Reg2Temp $val  CEL]
if  {$temp > 70} {
puts "Temperature is excessive $temp"
}
return $temp
}
```



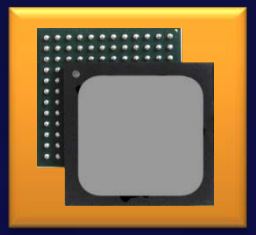
POD

Where the big money is

IP Domain Expertise

HDL

"I know my infrastructure IP" "I don't want To know the entire system to make it work"



"I know my IC" "I don't want to support the infrastructure IP or the entire system"

"I know my system design" "How Does this Infrastructure IP work?"

EMS

"I know board test and assembly" "How does this Infrastrucure IP work?"

Fast forward to today

Critical Elements for Successful use of JTAG Assisted Functional Tests

1) Standardized languages and BSDL constructs
2) Ease of use/Re-use
3) Minimal ecosystem requirements
4) Minimize false failure mechanisms (see 2 & 3)

Reduce Cost of IC for the System Integrator

- IC is good, price is right, cost to deploy system with IC is high

Correlation

IC Tester

- Stable temperature
- 50ohm Z_T DUT card design, dedicated
- Low noise Power, DC/DC converters
- Perfect Low jitter, 50/50 duty clocks
- BIST/Compression vectors, delay test

IC in System

- Changing temp
- Std. FR4, multi-IC signals
- Commodity LDOs, DC/DC
- Tin Can Osc, System origin clocks
- JTAG assisted Functional/BIST

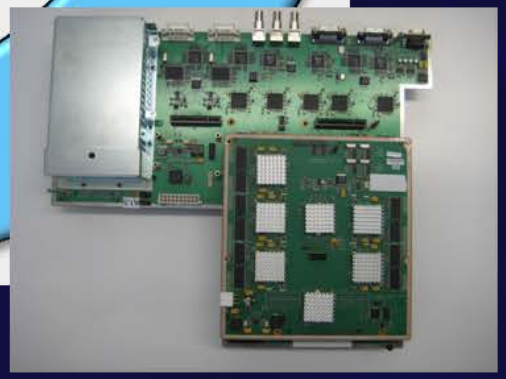
On-Chip test via IEEE 1149.1 - the lowest common denominator

IC Tester

System Test

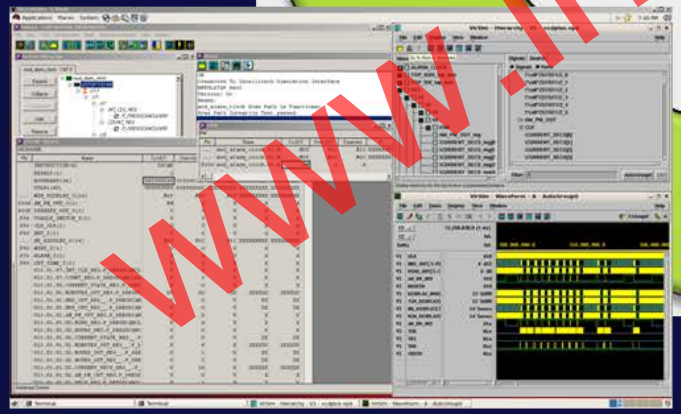
Test

Field



TEST
DEBUG
CONFIGURATION
With 1149.1/JTAG

Simulation



IEEE P1149.1-2012 Test Data Register interface

An IP block has a TDR segment directly attached to it.
-Purpose built IP block (PLL, SERDES, etc)

Broad industry applications
- microcontrollers, FPGAs, ASICs

IP blocks which don't have a TDR segment currently, will
In the future as the ecosystem around the standard develops

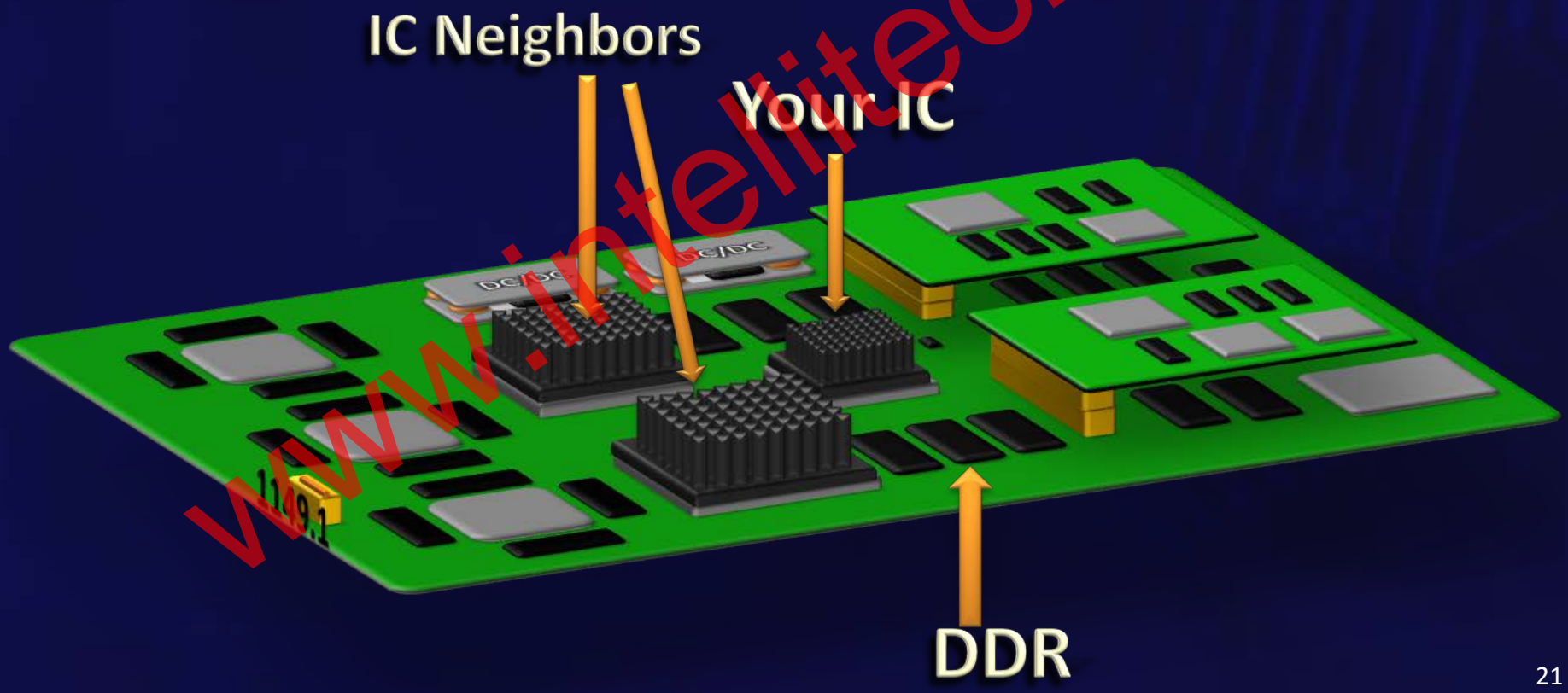
CLAMP_HOLD / CLAMP_RELEASE / IC_RESET needed for guaranteed operation of IP based tests

Focus on ecosystem to isolate IP block or IC such that no system or ecosystem faults can prevent IP from working

Document IC power/ground and system clock pins
use REGISTER_PIN_ASSOCIATION attribute

Board Test Complexity

Need help in testing Ecosystem around IC : DDRs, SERDES I/O
May require powering down/up system between BIST tests
What system knowledge is required to understand
how to operate on-chip Infrastructure IP?

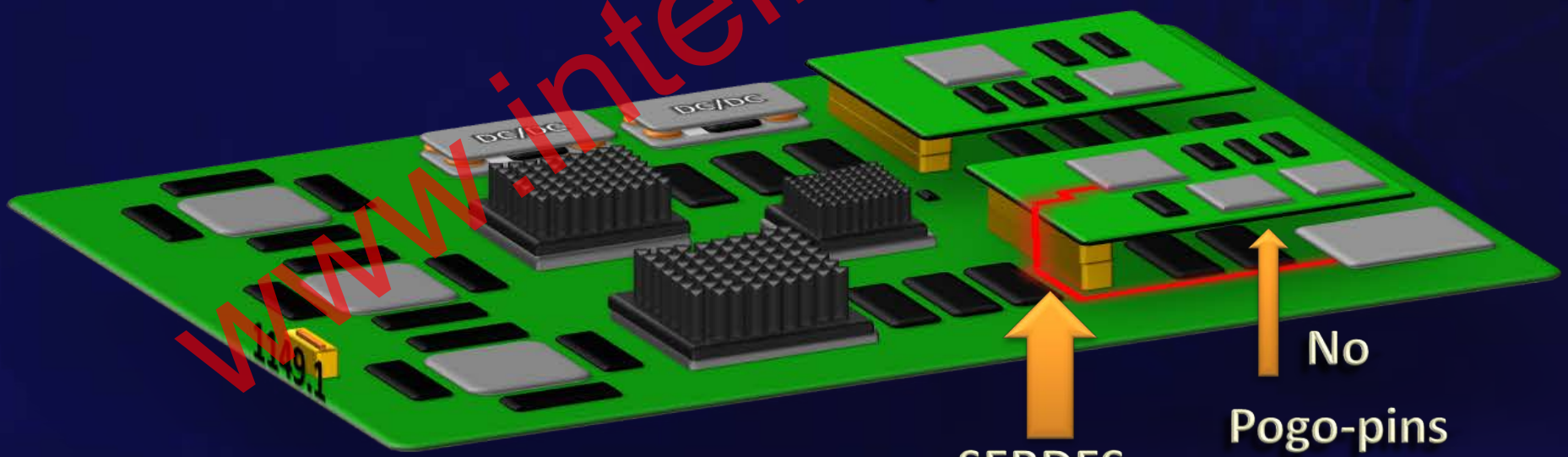


Need simple JTAG assisted functional tests

- small bite-sized at-speed test
- ex.: using on-chip PRBS/BER test

Remove need for entire system to boot

Enable in-field structural test (no external tester)



SERDES

No Pogo-pins On mezzanine

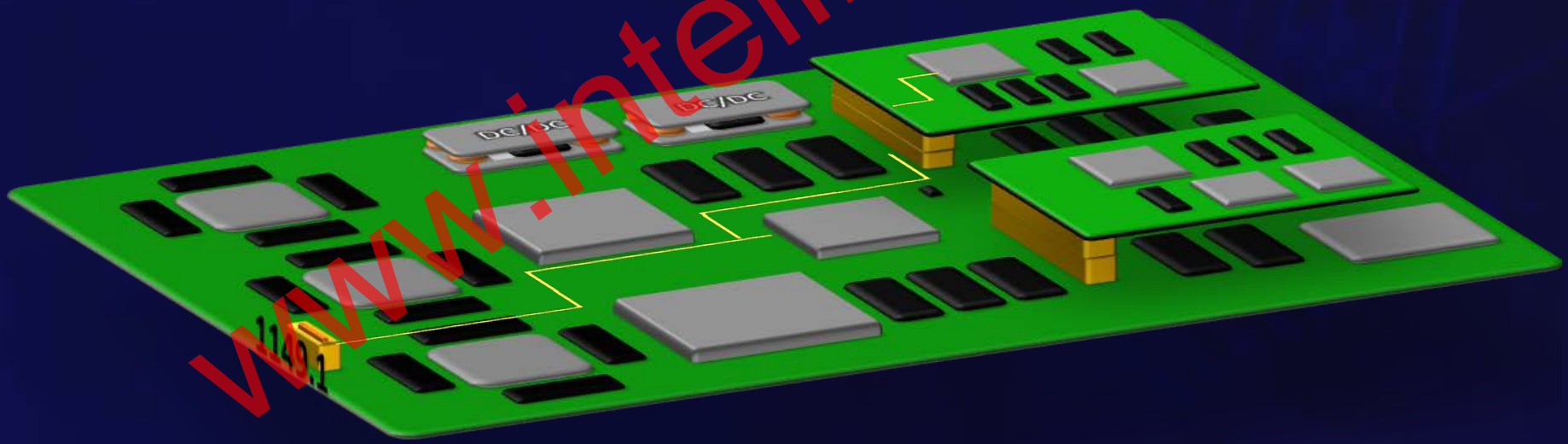
Board Test Complexity

Heat Sinks may not be present at board test

On-chip PLLs need to be controlled (safe and cool)

Access to system reset of IC critical

- power up/down is costly in terms of time
- many resets (and not always routed for JTAG test)



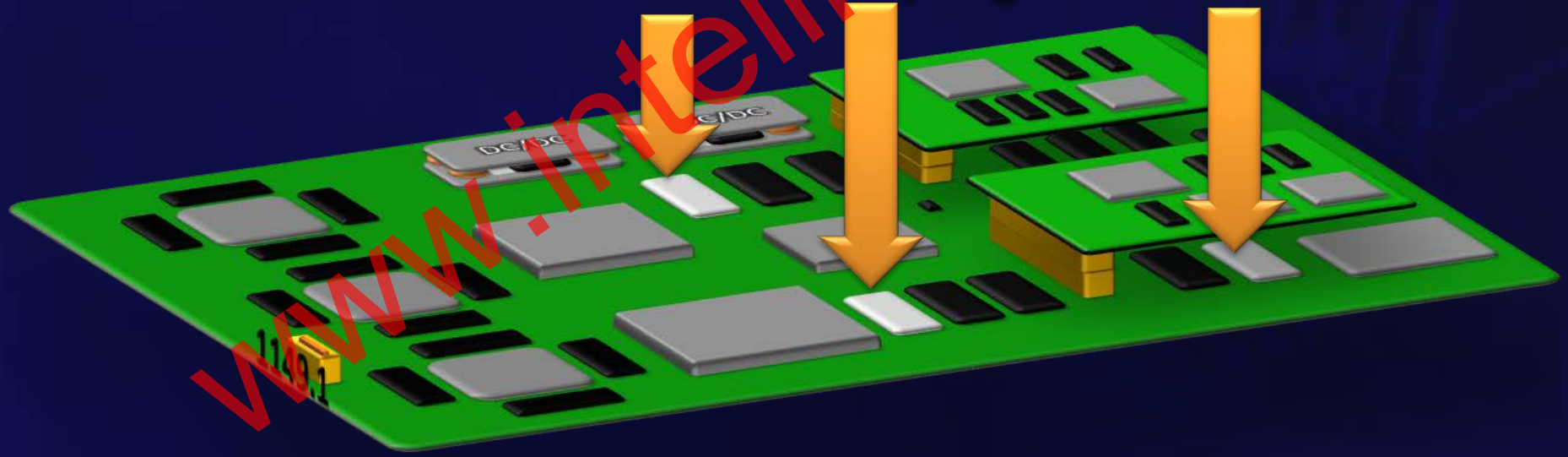
CPU may not have ecosystem to initialize I/O

-Clocks, code, resets, watchdogs

PLDs unprogrammed

-Inputs to ASICs perhaps undriven

Flash un-programmed

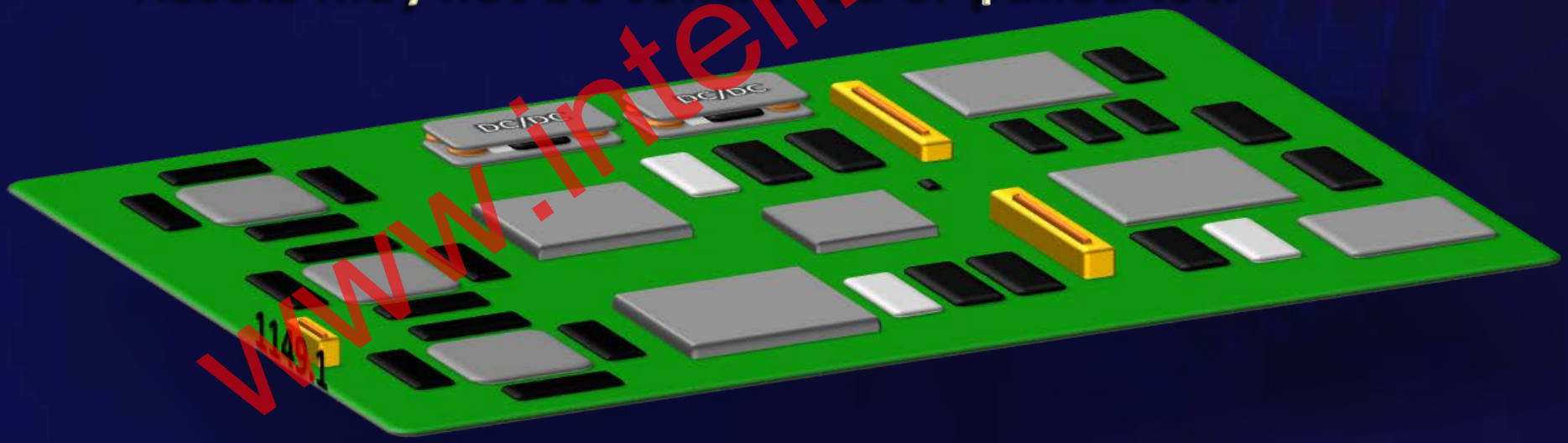


Board Test Complexity

COTs CPU modules missing

IC I/O may be uninitialized

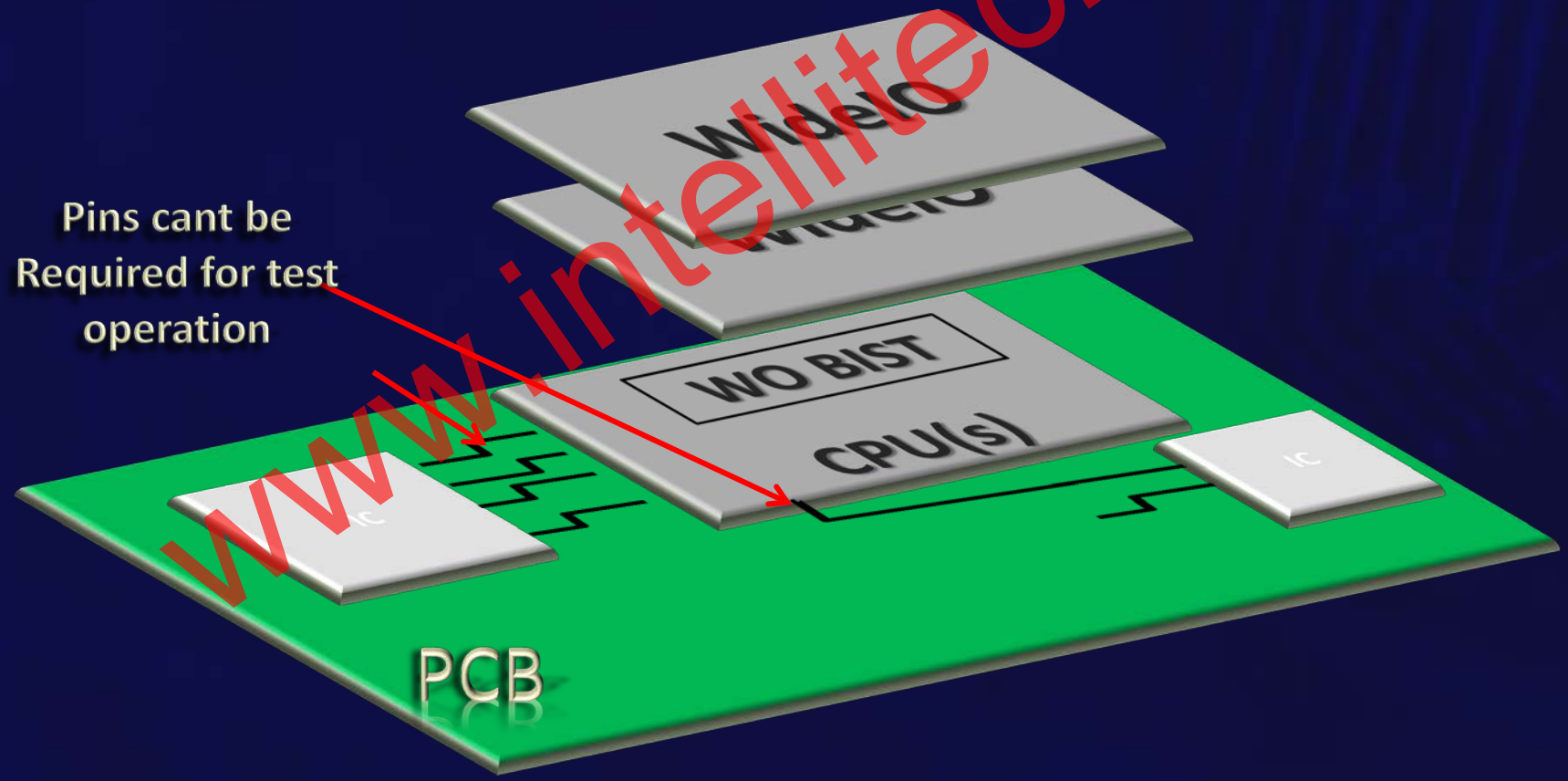
Resets may not be controlled or pulled-low



Isolation critical for reliable Test & F/A

1149.1 accessible SDRAM BIST in 3D package

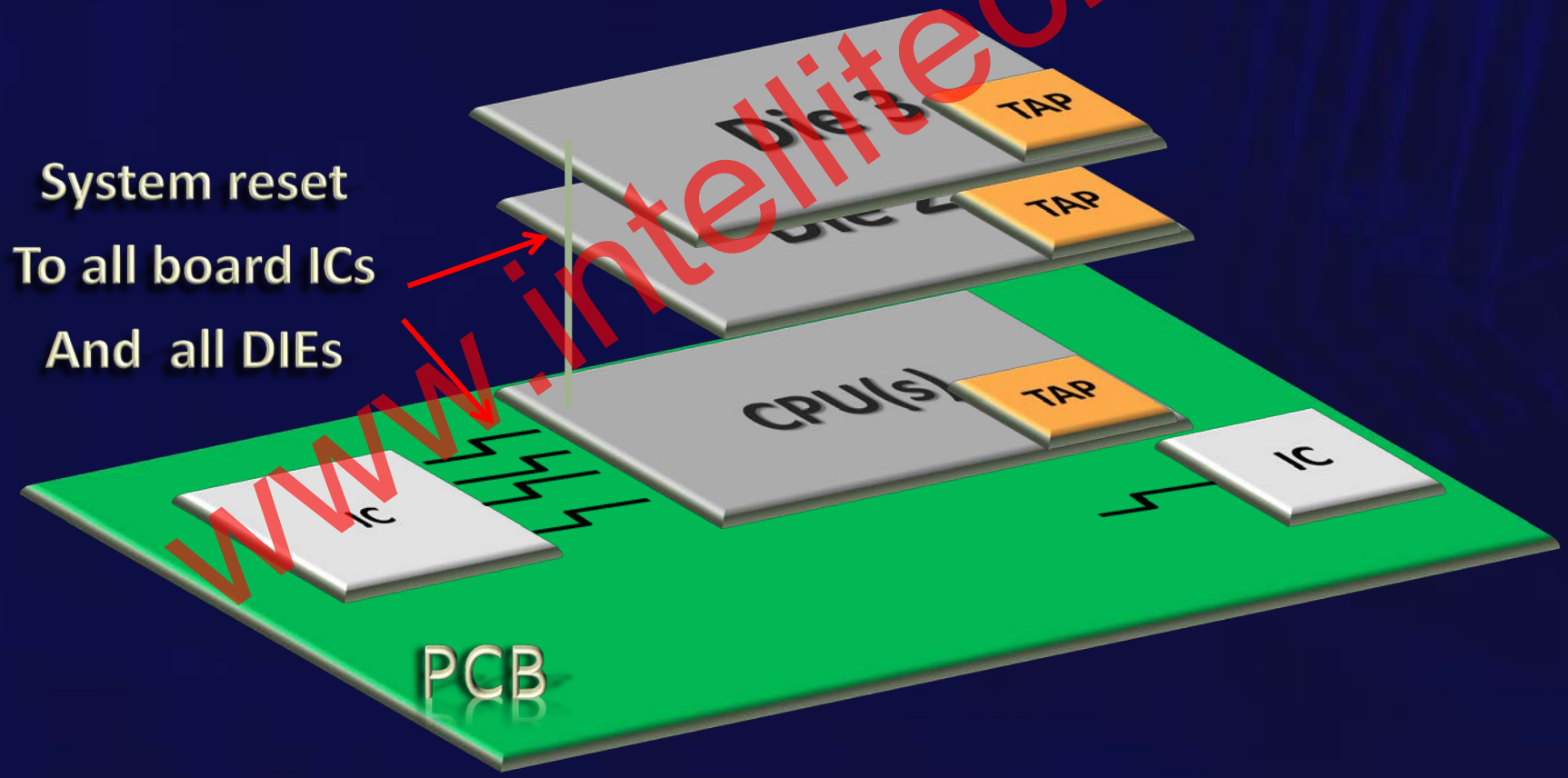
- can't fail good SDRAM due to system design/fault
- need to hold/isolate package pins during SDRAM BIST



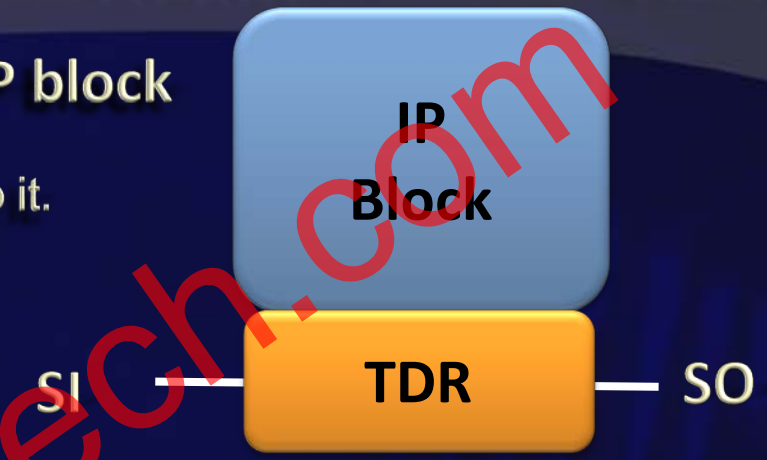
JTAG controlled IC RESET needed

- LogicBIST, memoryBIST, etc - all need IC reset after execution
- Need to isolate the reset from all the other ICs
  - Save time -hold reset internally to IC/DIE so IP based test can run

after reset each IC has to potentially be re-INIT, re-configged for next test



IEEE 1149.1 IP block



An IP block has a TDR segment directly attached to it.
-Purpose built IP block (PLL, SERDES, etc)

Broad industry applications
- microcontrollers, FPGAs, ASICs

IP blocks which don't have a TDR segment currently, will
In the future as the ecosystem around the standard develops

CLAMP_HOLD / CLAMP_RELEASE / IC_RESET needed for guaranteed operation of IP based tests

Focus on ecosystem to isolate IP block or IC such that no system or ecosystem faults can prevent IP from working

Document IC power/ground and system clock pins
use REGISTER_PIN_ASSOCIATION attribute

Ideally less reliance on external digital I/O to prepare IC for tests or execute on-chip IP for board test

Small features

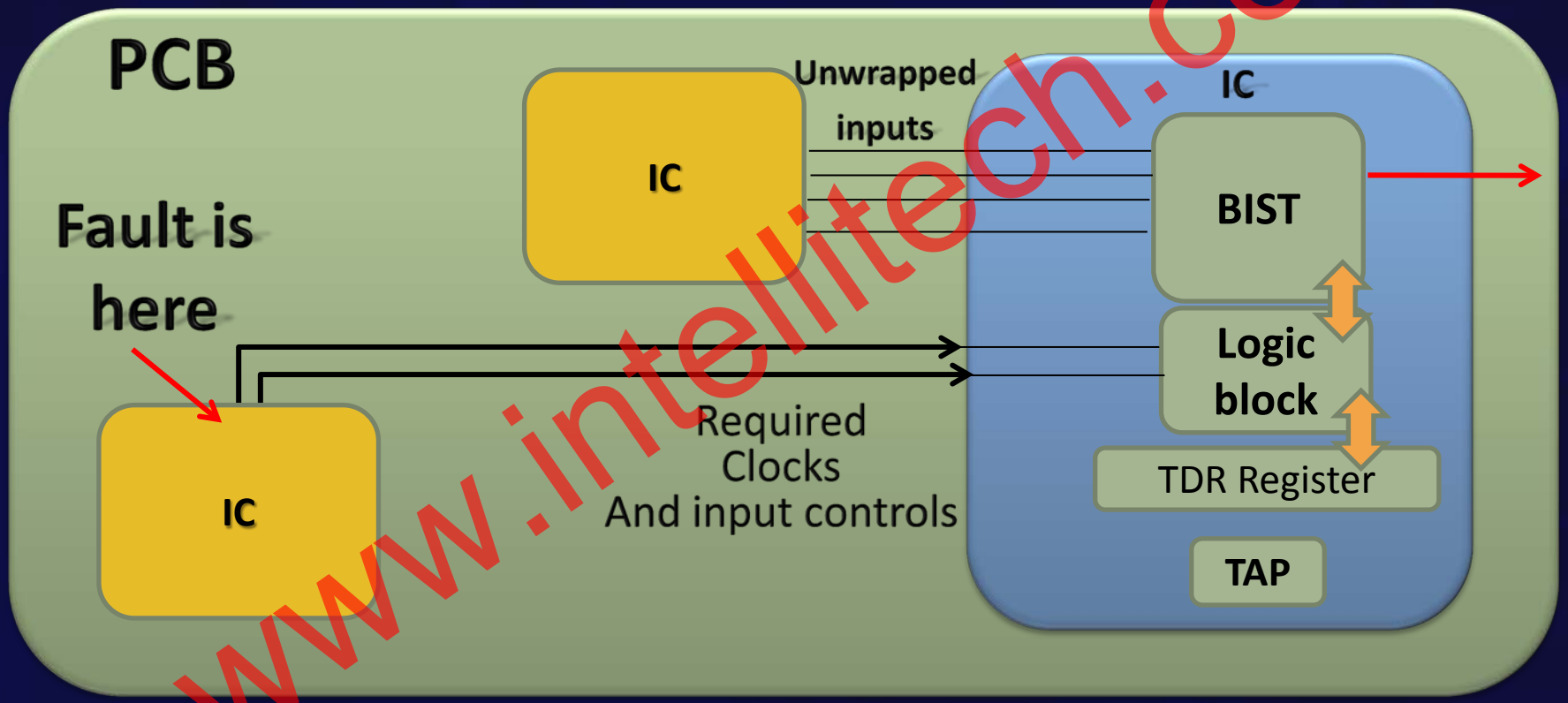
- High Density

-Difficult to probe

Typical .035”

.017”

Prior to IEEE 1149.1-2012

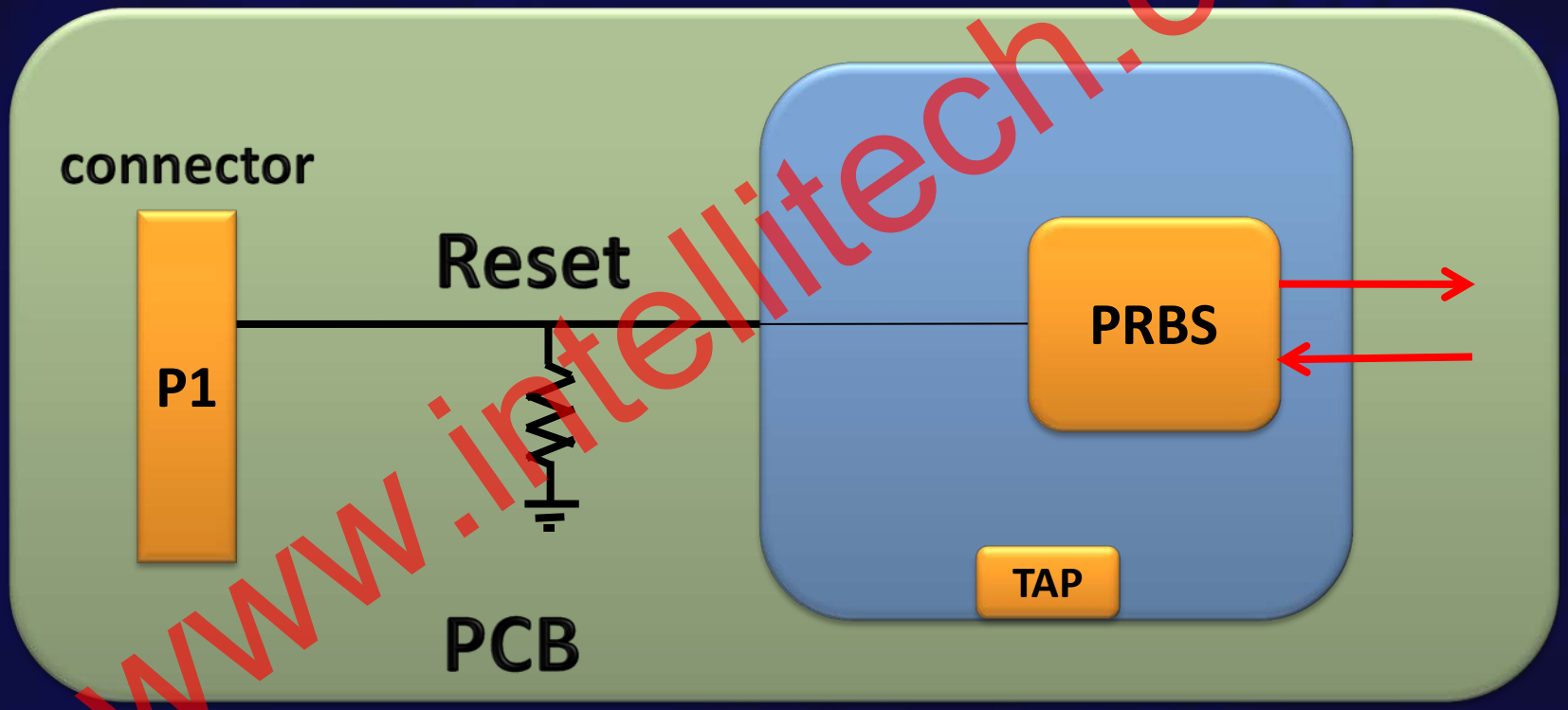


If you can control your ASIC design flow as the System design is being done – perhaps possible to manage And avoid – but at what cost/risk. How is it diagnosed?

On-chip JTAG based tests require stable system resets

Without System reset control – IP block tests will fail mysteriously

(open, toggling, non-driven RESET inputs) - Difficulty for CM to find root cause



This is managed in 1149.1 with the IC_RESET instruction

IP cannot count on external ecosystem pins

Need EXTEST to Hold/Toggle pins

-3000 cells causes large test time increase

-(compared to direct access)

-IC2 no longer in functional mode

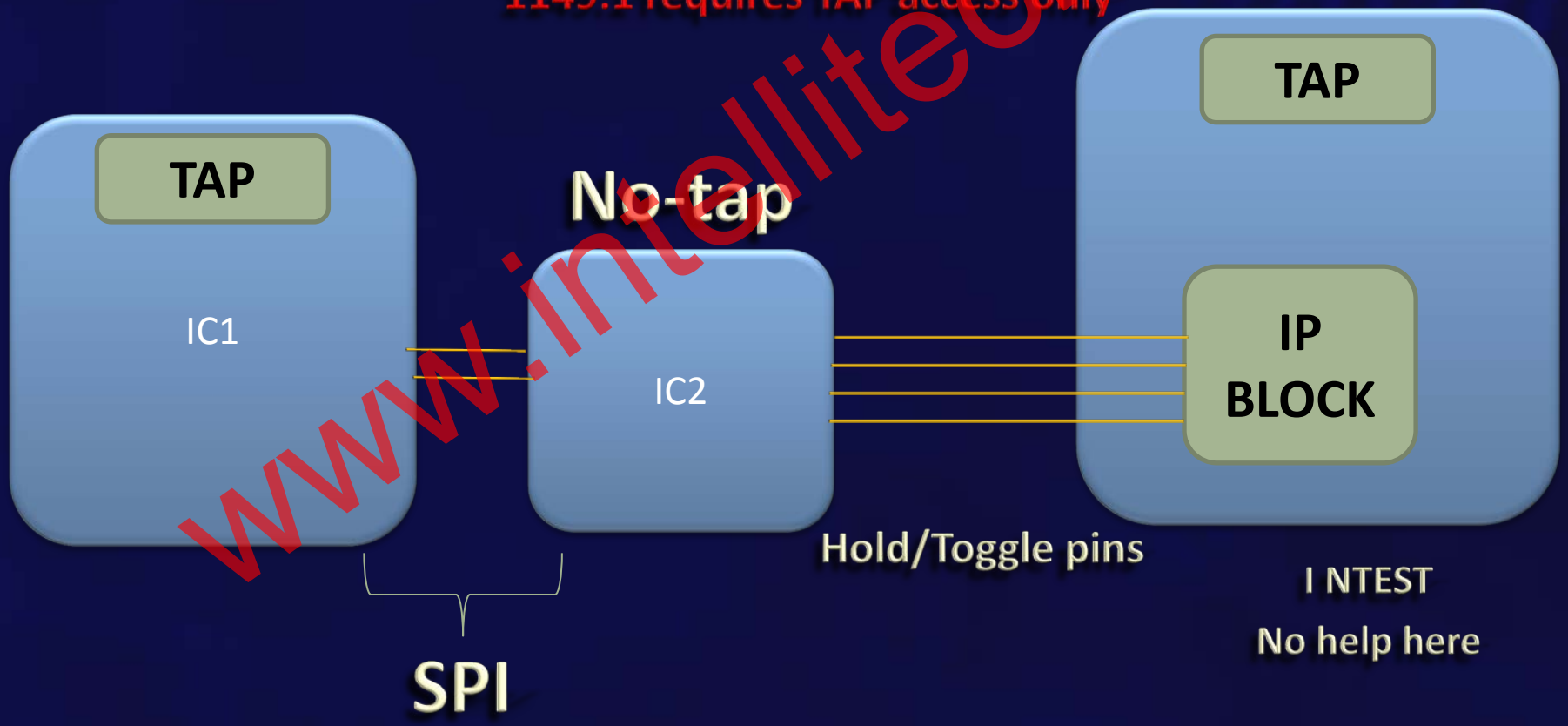
Need to reduce dependencies on ecosystem

Ex: control IP block through I/O pins

Talk to IC1 - communicate SPI protocol

Toggle I/O pins to operate IP Block

1149.1 requires TAP access only

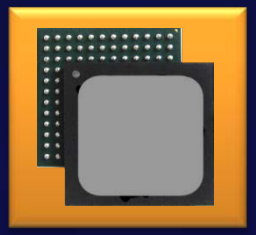


Where the big money is

IP Domain Expertise

HDL

"I know my infrastructure IP" "I don't want To know the entire system to make it work"



"I know my IC" "I don't want to support the infrastructure IP or the entire system"

"I know my system design" "How Does this Infrastructure IP work?"

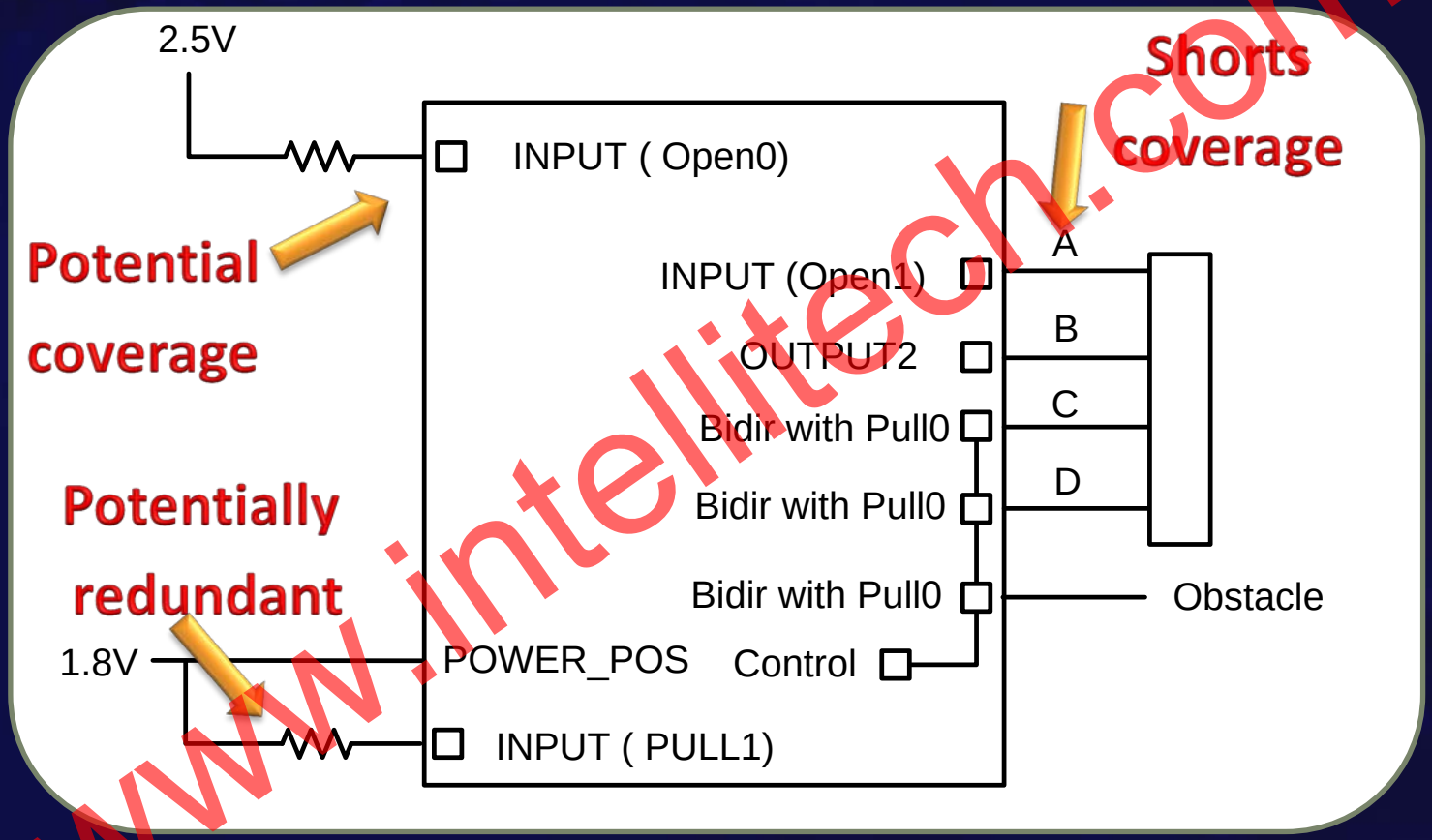
EMS

"I know board test and assembly" "How does this Infrastrucure IP work?"

What IEEE 1149.1-2012 is proposing

(still unapproved/unballoted)

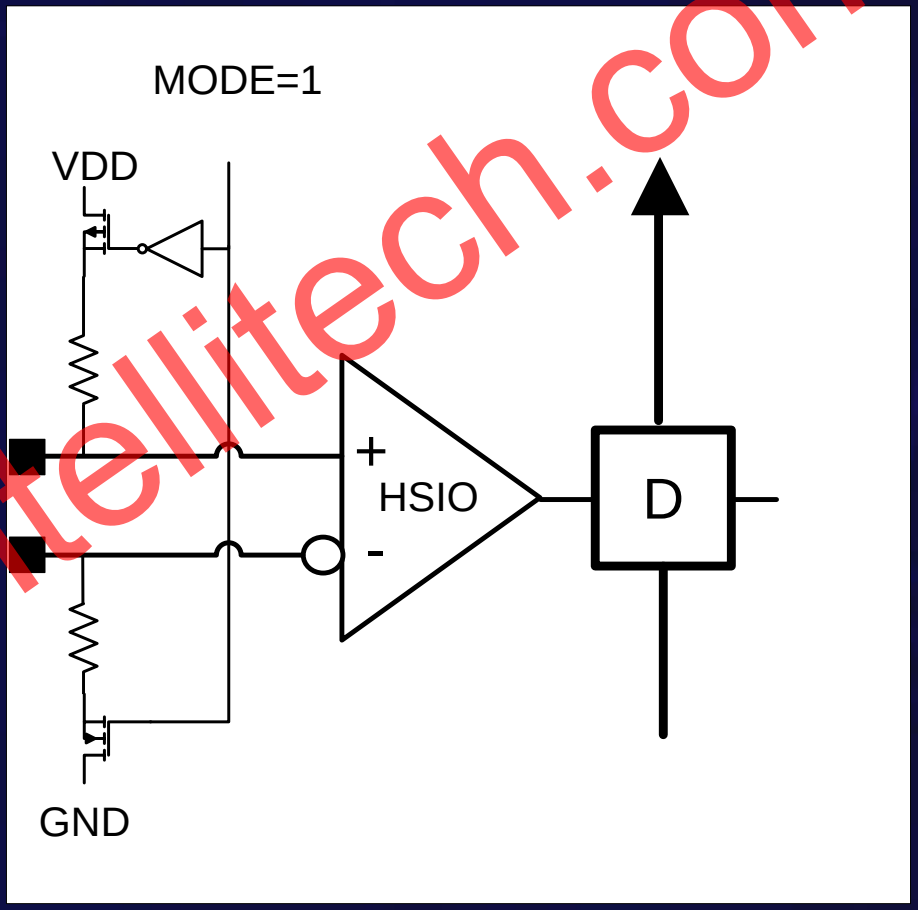
New <input spec> = Open1/0/X, Pull1/0, Keeper



"8 (BC_1, in1, input, open0)," &
"7 (BC_1, out2, output3, X, 16, 1, PULL1)," &

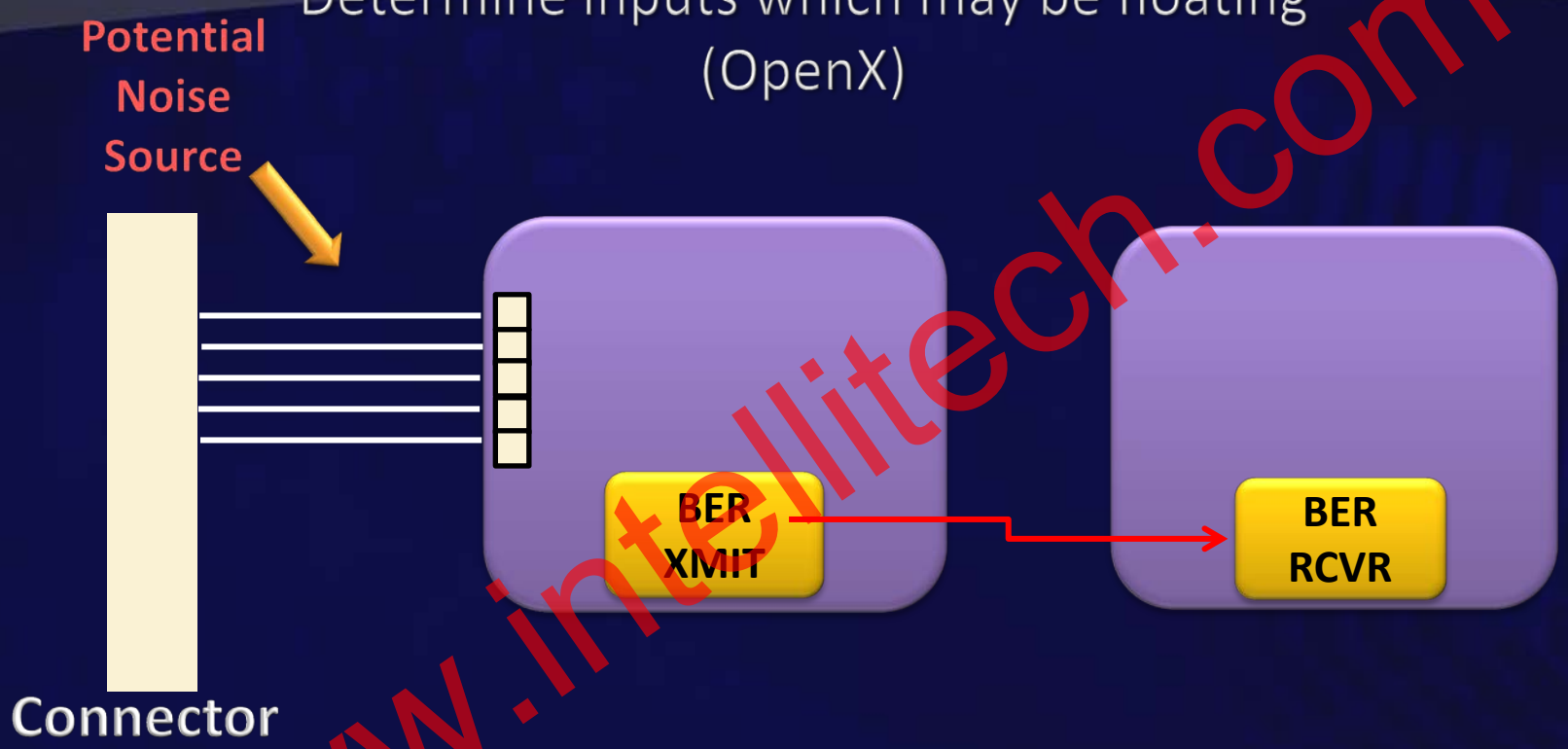
PULL effect may occur only in test modes

LVDS
And other
Standards
Require
Differential
Receivers to
produce '1' on open
connection



"8 (BC_1, in1, input, PULL1)," &

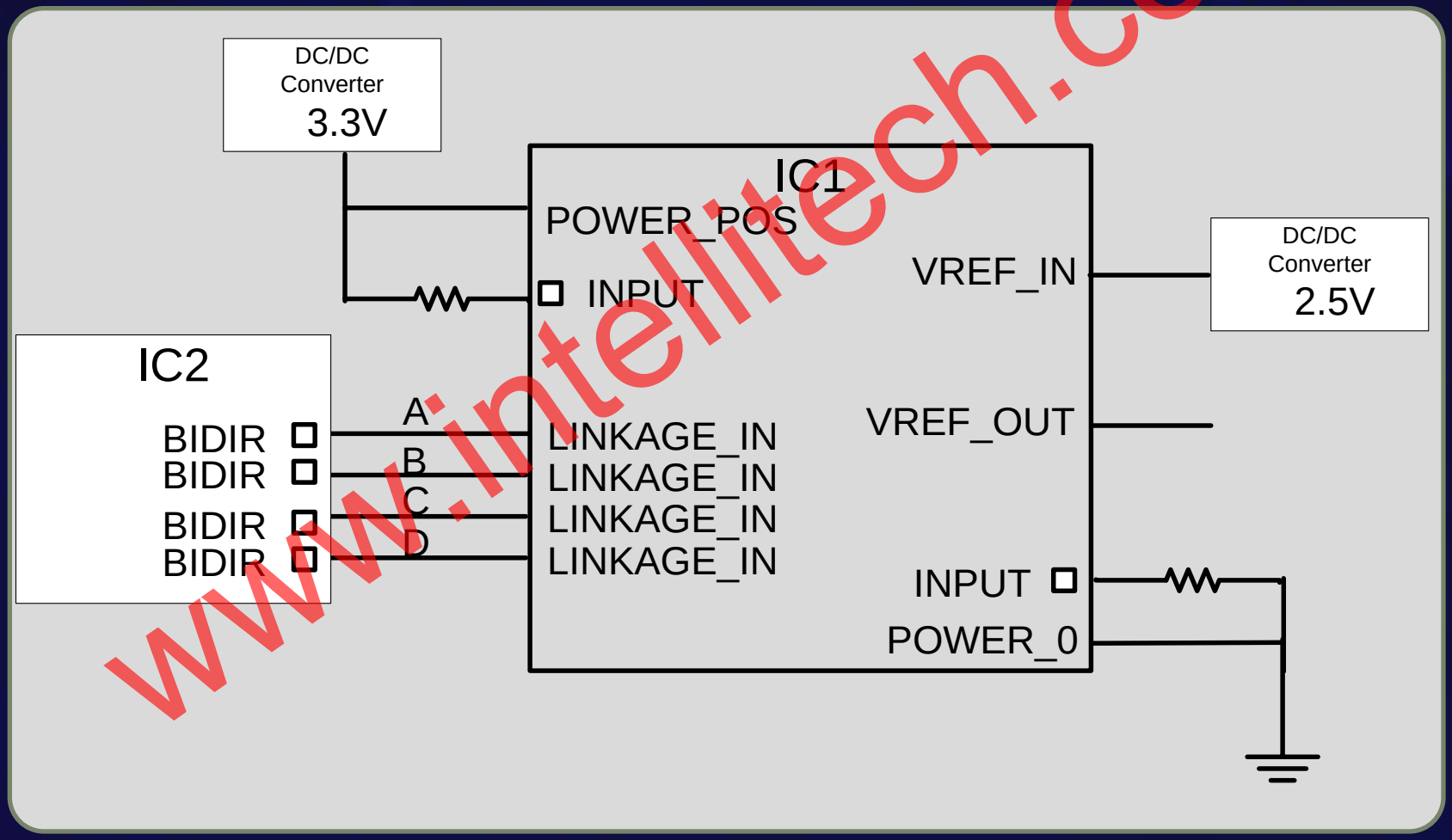
Determine inputs which may be floating (OpenX)



During board test, input s may not be driven due to connectors (Mezzanine cards not present, CPUs not socketed)

Floating inputs are noise sources that need to be identified for Reliable BERT

New descriptive port types for linkage pins



New descriptive port types for linkage

linkage_out	A non-boundary scan analog port capable of sourcing/sinking significant current that has a disable method.
linkage_in	A non-boundary scan analog input that does not source or sink significant current
linkage_inout	A non-boundary scan analog bidirectional
Linkage_buffer	A non-boundary scan analog port capable of sourcing/sinking significant current, but does not have a disable method.
linkage_mechanical	A non-electrical port used for positioning, heat sinks or other non-electrical use. There is generally no connection to the chip silicon.
vref_in	A non-boundary scan input reference voltage port
vref_out	A non-boundary scan output reference voltage port
power_0	Zero volt Ports. These are ports which are normally associated with GROUND. Keyword GROUND or GND is not used here in order to leave these words for signal names.
power_pos	Power supply ports which receive a constant potential with respect to power_0 that is greater than zero volts.
power_neg	Power supply ports which receive a constant potential with respect to power_0 that is less than zero volts.

Six new Instructions:

IC_RESET

- reset IC and power domains through JTAG

INIT_SETUP/INIT_RUN

- configure I/O on-chip resources for

CLAMP_HOLD/CLAMP_RELEASE

- hold pins for in-situ on-chip tests

ECIDCODE

- read unique die/TAP ID value

New Documentation Capabilities

BSDL for Internal JTAG TDR registers
- for BIST/PLLs/SERDES IP blocks

MNEMONICS for JTAG registers
- Easy to remember words

Package files for on-chip Infrastructure IP blocks
- self-contained definitions for IIP

PDL Script files for device initialization and IIP access
- operates on registers, packages, Mnemonics

IC_RESET

IC_RESET Objectives

Cause an on-chip reset to occur via TAP

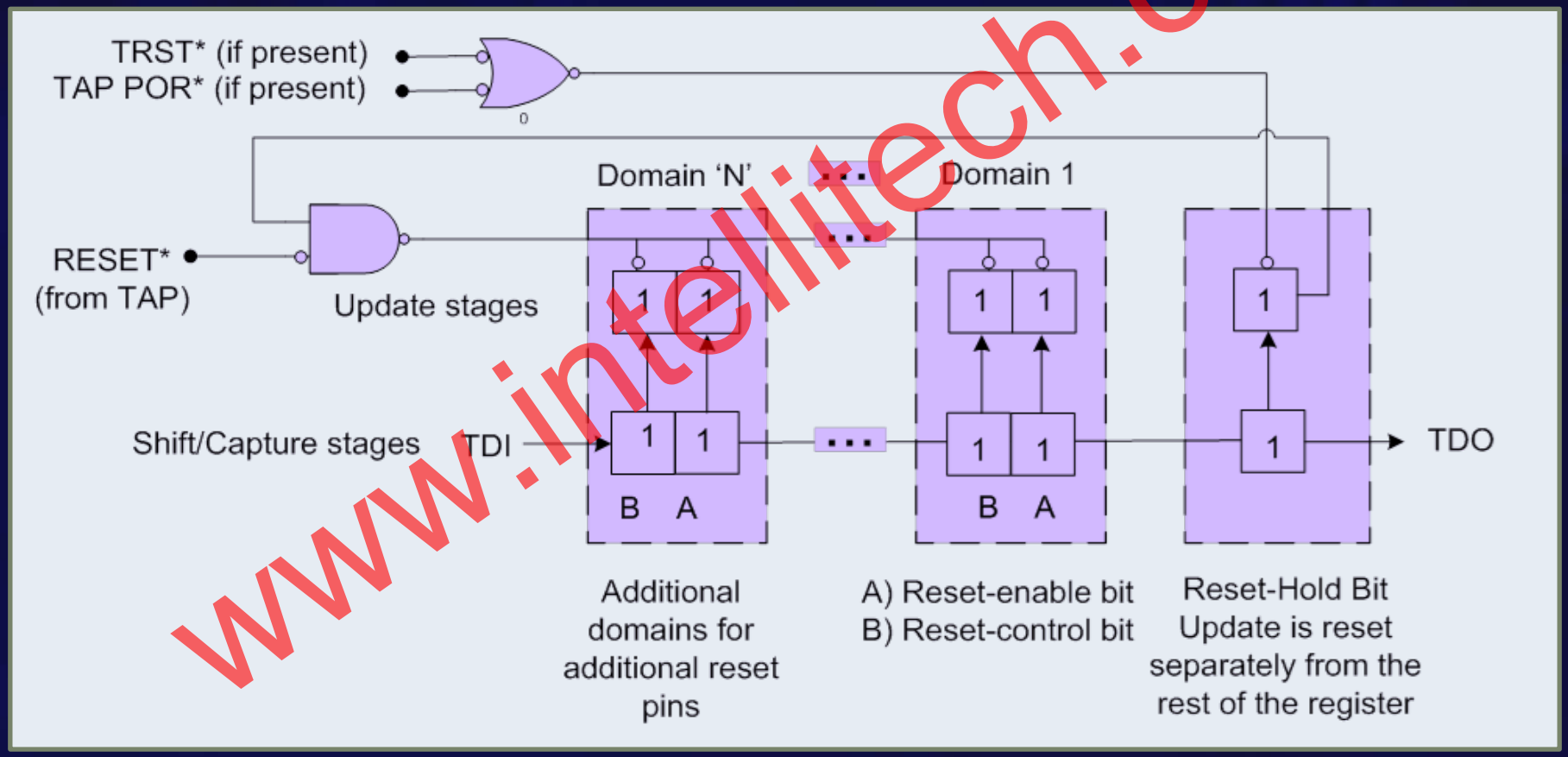
- emulate functionality of system reset pin

Isolate on-chip logic from external system reset affects

Enable control of on-chip POR resets in non-I/O domains

Prevent loss of reset isolation/control when TAP enters Test-Logic-Reset State

Reset-select register design



Clamp_Hold/Clamp_Release

Clamp_Hold/Clamp_Release

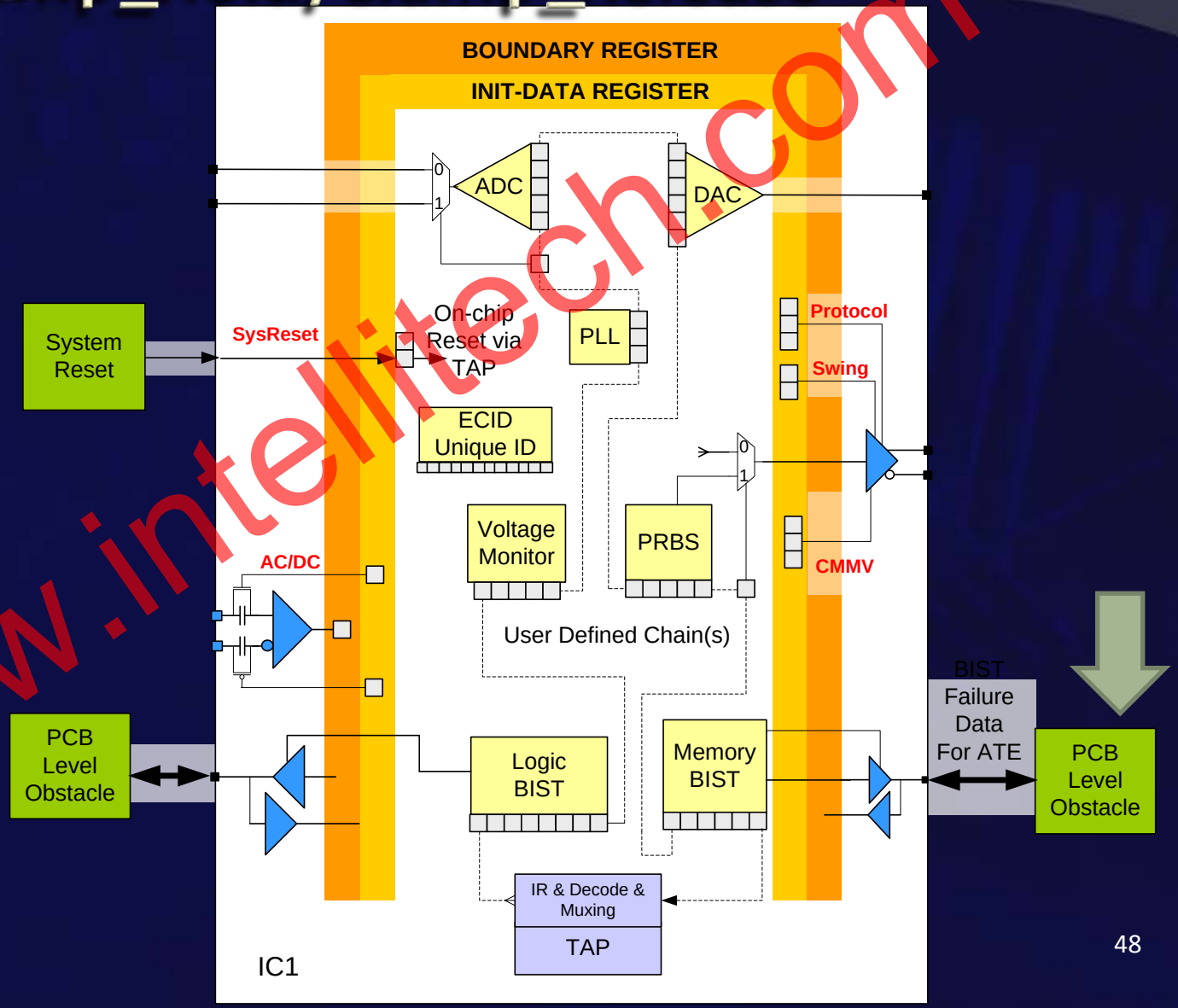
Objectives of CLAMP_HOLD/CLAMP_RELEASE

1) Set I/O to a persistent state while other instructions Are loaded into instruction register
2) Enable in-situ on-chip test while I/O are isolated
3) Keep I/O analog parameters persistent
4) Prevent TAP RESET* from changing b-s cells
   a) controlr - reset cells
   b) SEGSEL - TDR length control collapses at T-L-R

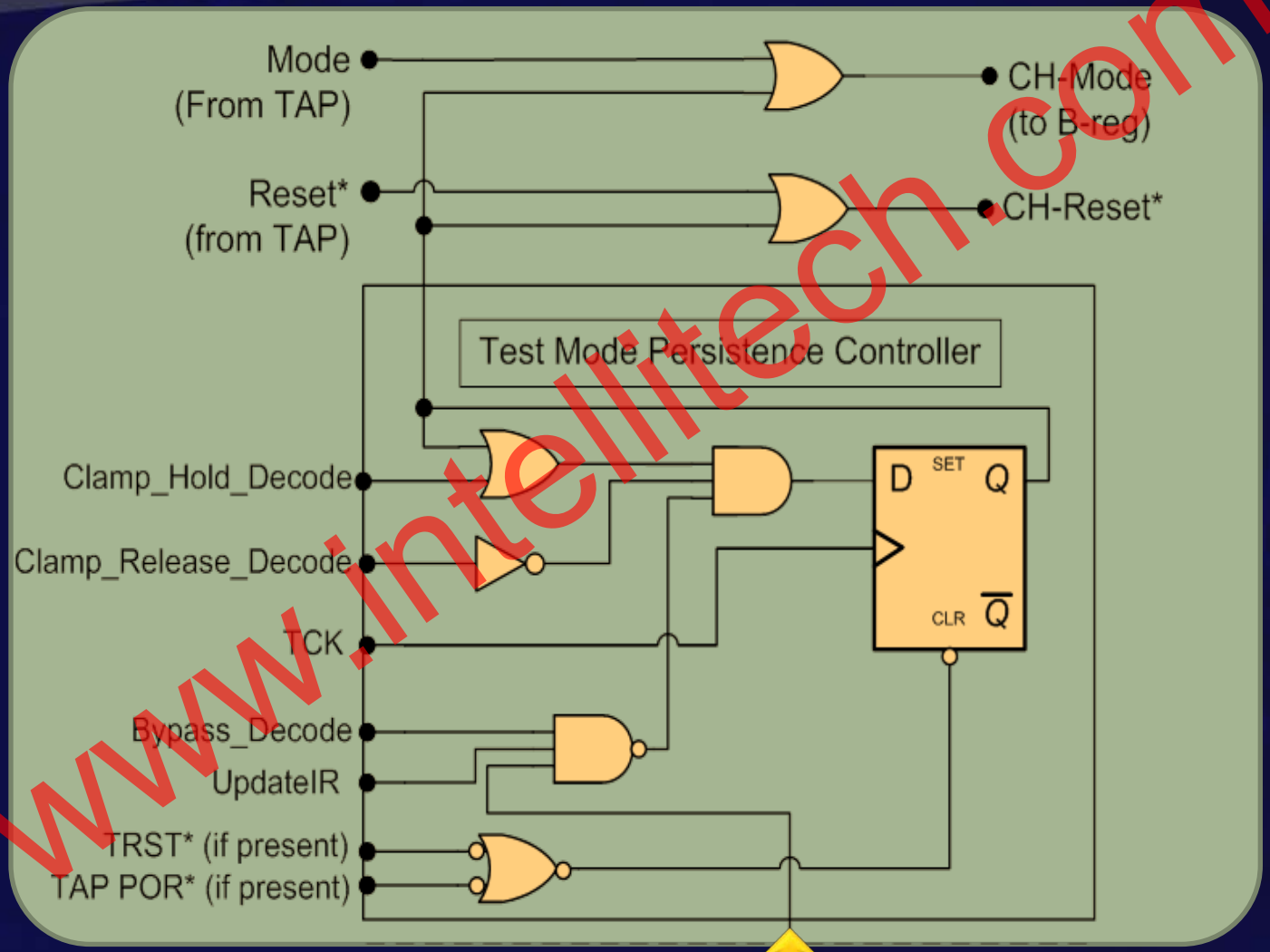
Clamp_Hold/Clamp_Release

Protect Against PCB Obstacles

(or die in a 3D Stack)



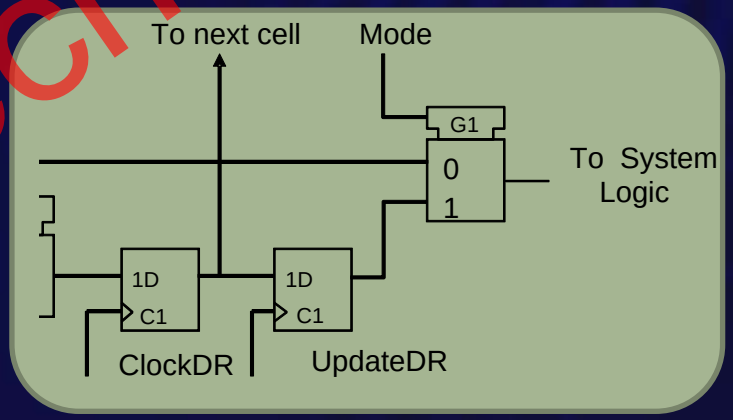
CLAMP_HOLD (intercept mode/reset to TDRs)



From Bypass-escape register

CLAMP_HOLD (intercept mode/reset to TDRs)

Mode (From TAP)
Reset* (from TAP)
CH-Mode (to B-reg)
CH-Reset*
Test Mode Persistence Controller
Clamp_Hold_Decode
Clamp_Release_Decode
TCK
Bypass_Decode
UpdateIR
TRST* (if present)
TAP POR* (if present)
D SET Q CLR $\overline{Q}$
PI
Capture_ByEsc
TDI
Shift_ByEsc
TCK
ByEsc_tdo
Bypass-Escape Register



INIT_SETUP/INIT_RUN

INIT_SETUP & INIT_RUN

Objective:

Configure analog parameters of I/O prior to going into EXTEST

Configure PLLs to enable this (above) and shut PLLs off

Provide path to allow user defined I/O analog parameters while IC In mission mode

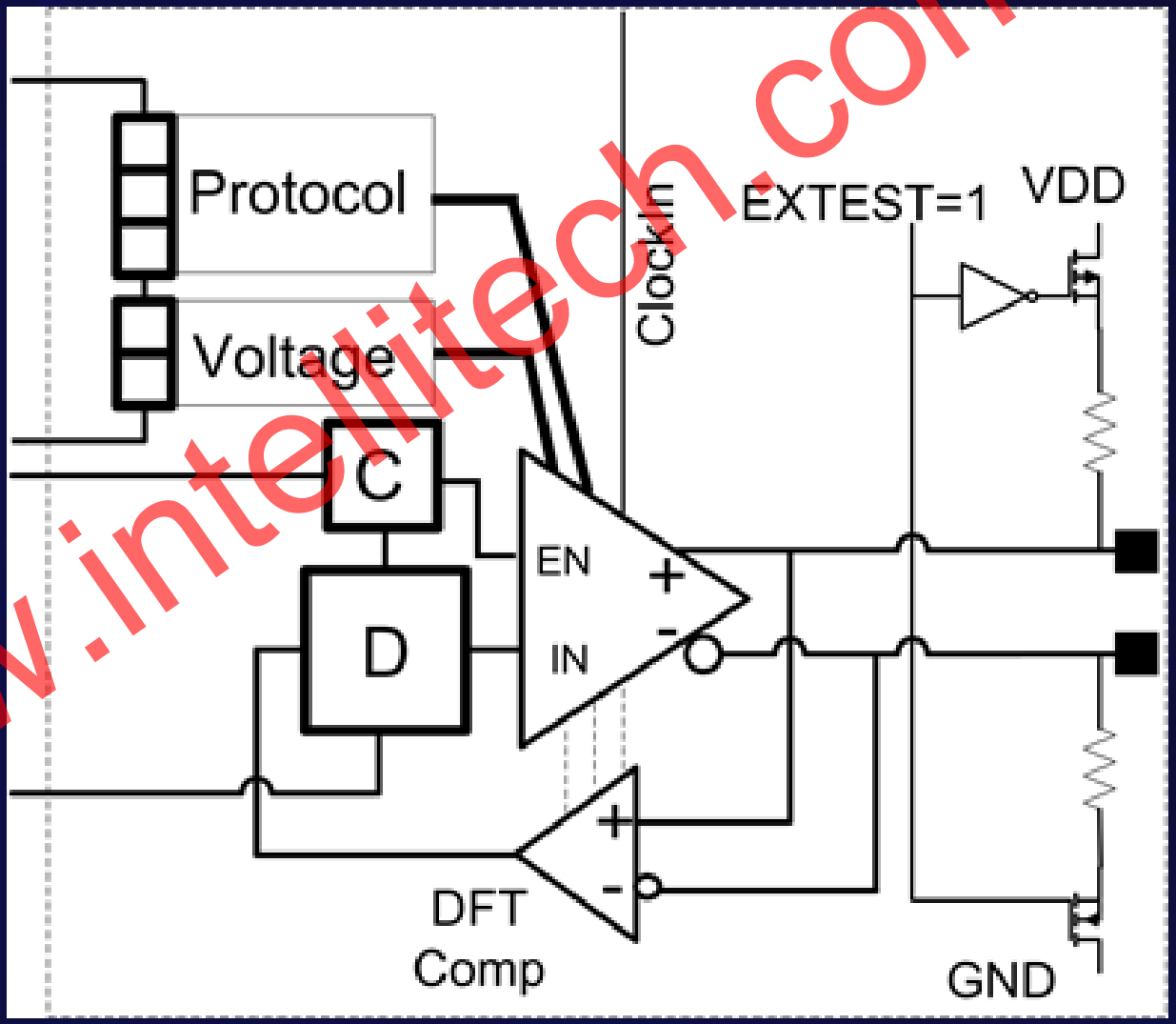
Use minimal resources: TAP, Compliance_Enable, Power, Sysclock

Use INIT_RUN to sequence I/O state machines & check status

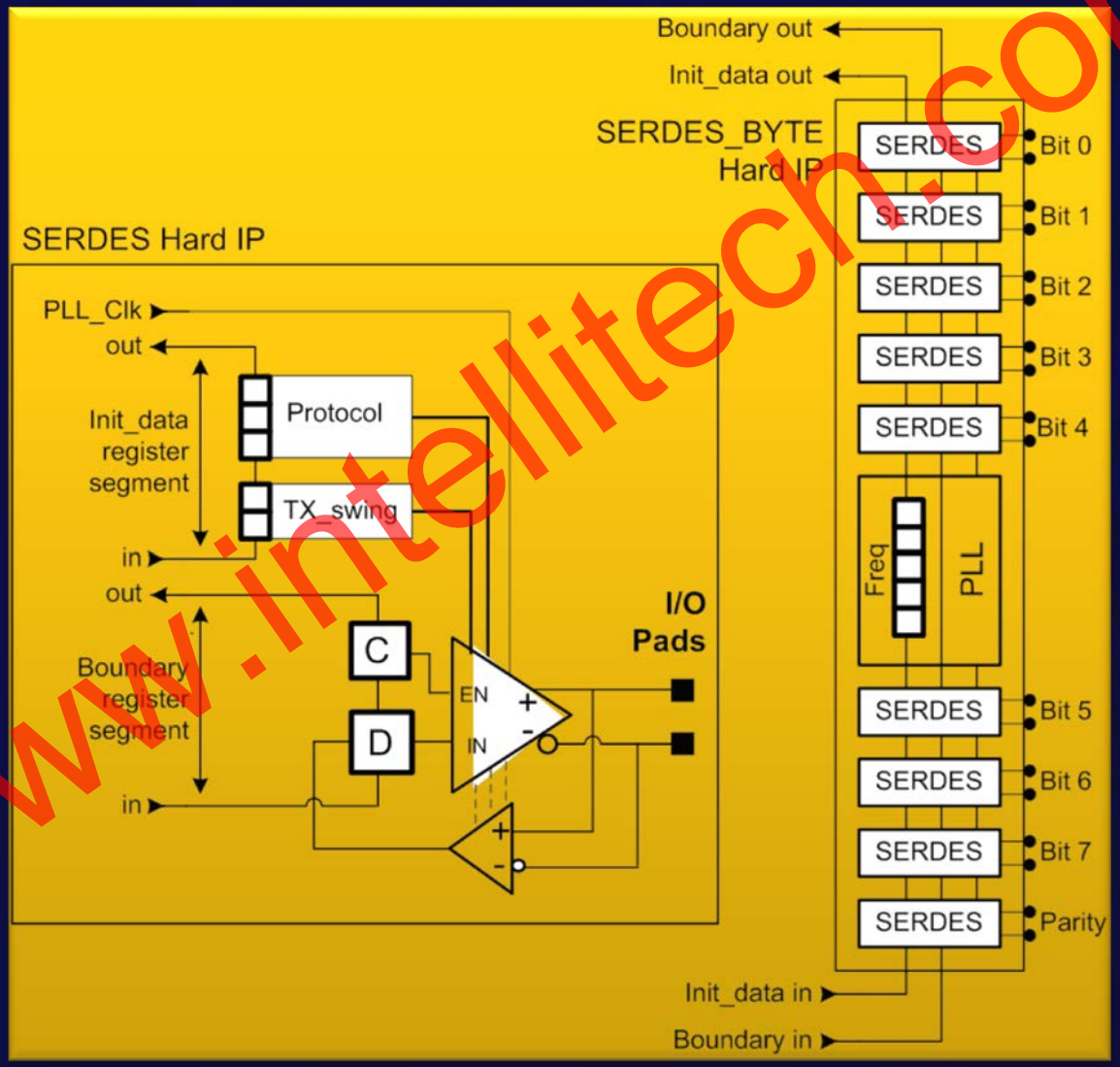
New standard INIT_DATA

Init_data Register

Boundary Register



Descriptions of I/O can be built into hierarchical blocks



Why can't I/O settings be delivered in BSDL?

Same IC
Could have different
I/O settings on one PCB

U3.PDL

```
iProc init_setup {} {
iWrite IO1  PCIe
iApply
}
```

U4.PDL

```
iProc init_setup {} {
iWrite IO1  SRIO
iApply
}
```

Board Test Engineer Developed via Software or from Templates from IC Vendor

Board.PDL

```
iCall U3.init_setup
iCall U4.init_setup
```

New PDL language
Allows description

Board Test Engineer Developed via Board Test Software, Automatically, assisted or manually

User Defined Test Data Registers

Recommendations for user TDRs
- Enable IP blocks TDRs to plug together

IP block interface

Scan In

Scan Out

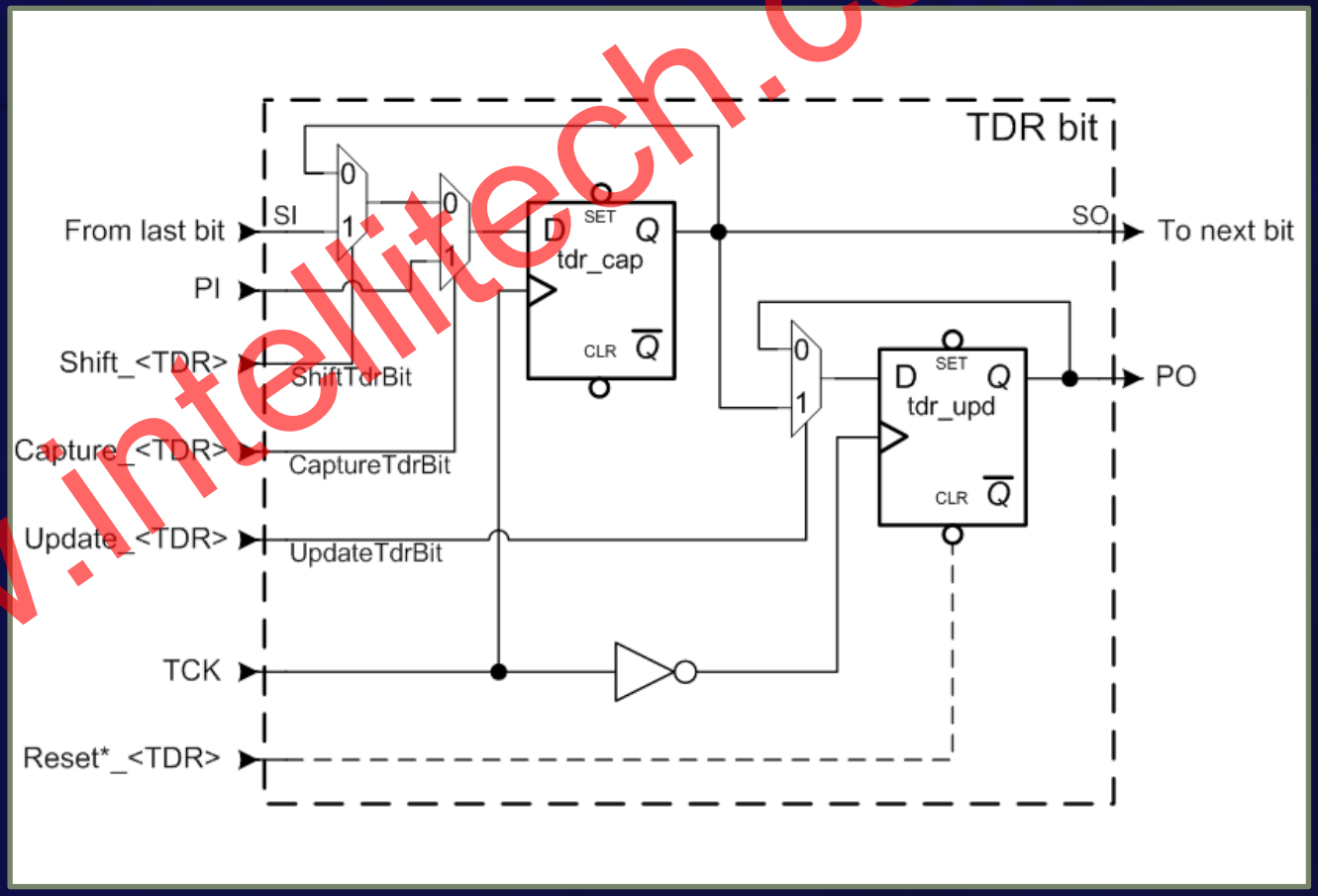
Shift_<TDR>

Capture_<TDR>

Update_<TDR>

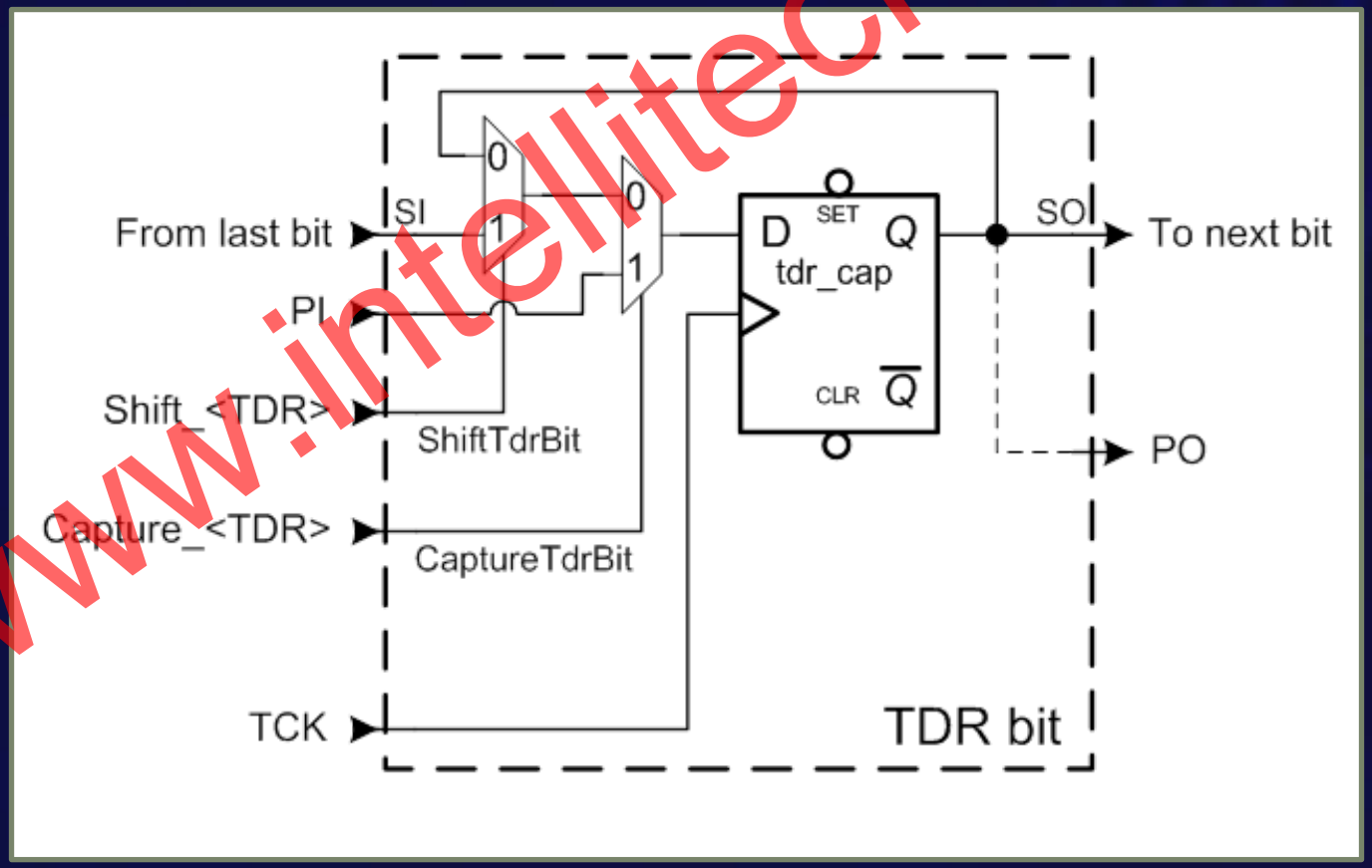
TCK

RESET*_<TDR>

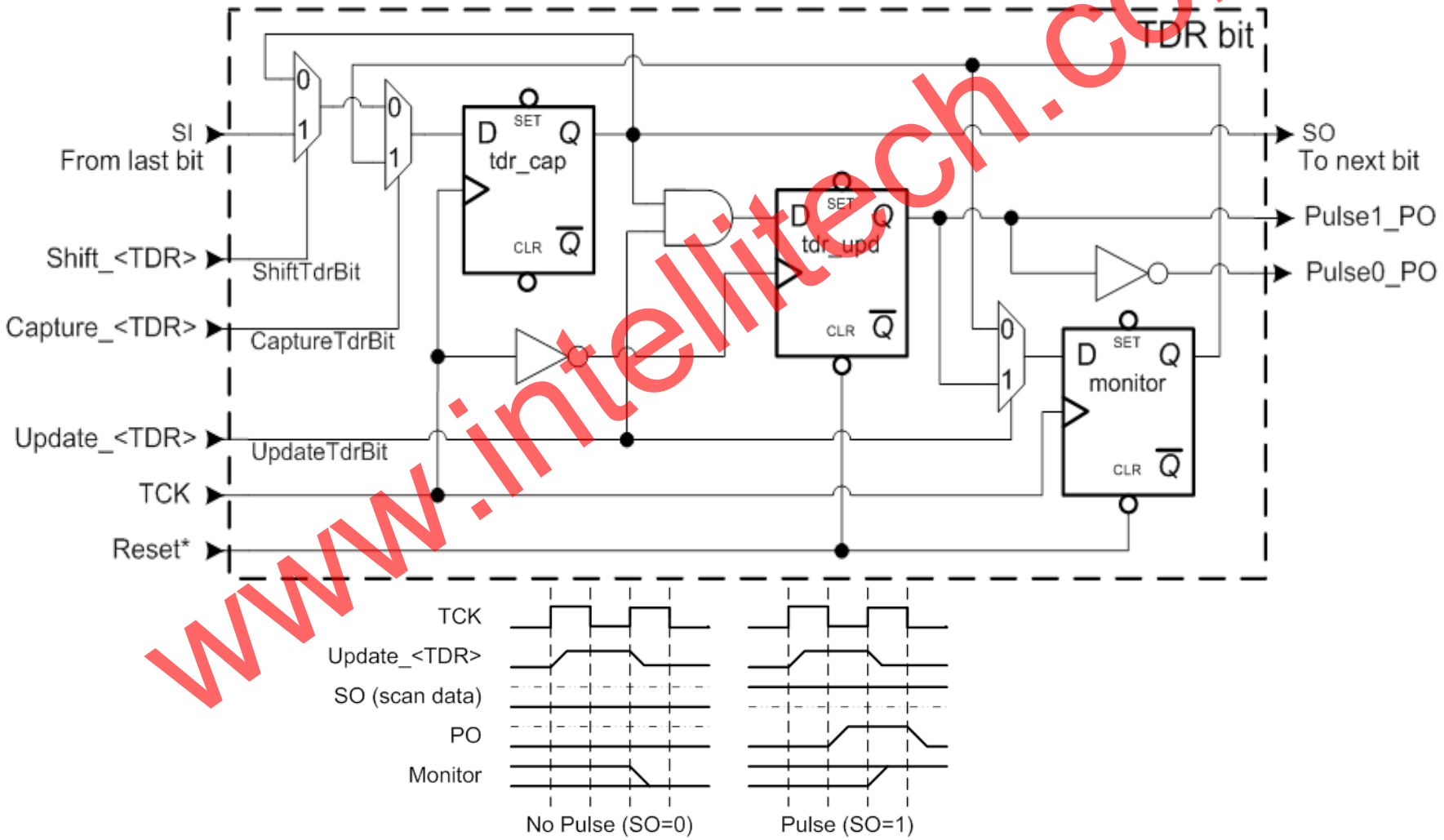


Recommendations for user TDRs
(six example User Defined Cells with verilog/VHDL)

capture/scan flop (can be shared with mission mode logic)



Recommendations for user TDRs
Self-resetting and Self-monitoring User Defined Cell



BSDL attributes allow description of User Defined Registers

Values associated with register bit

CAPTURES | DEFAULT | SAFE | RESETVAL

Type of cell:

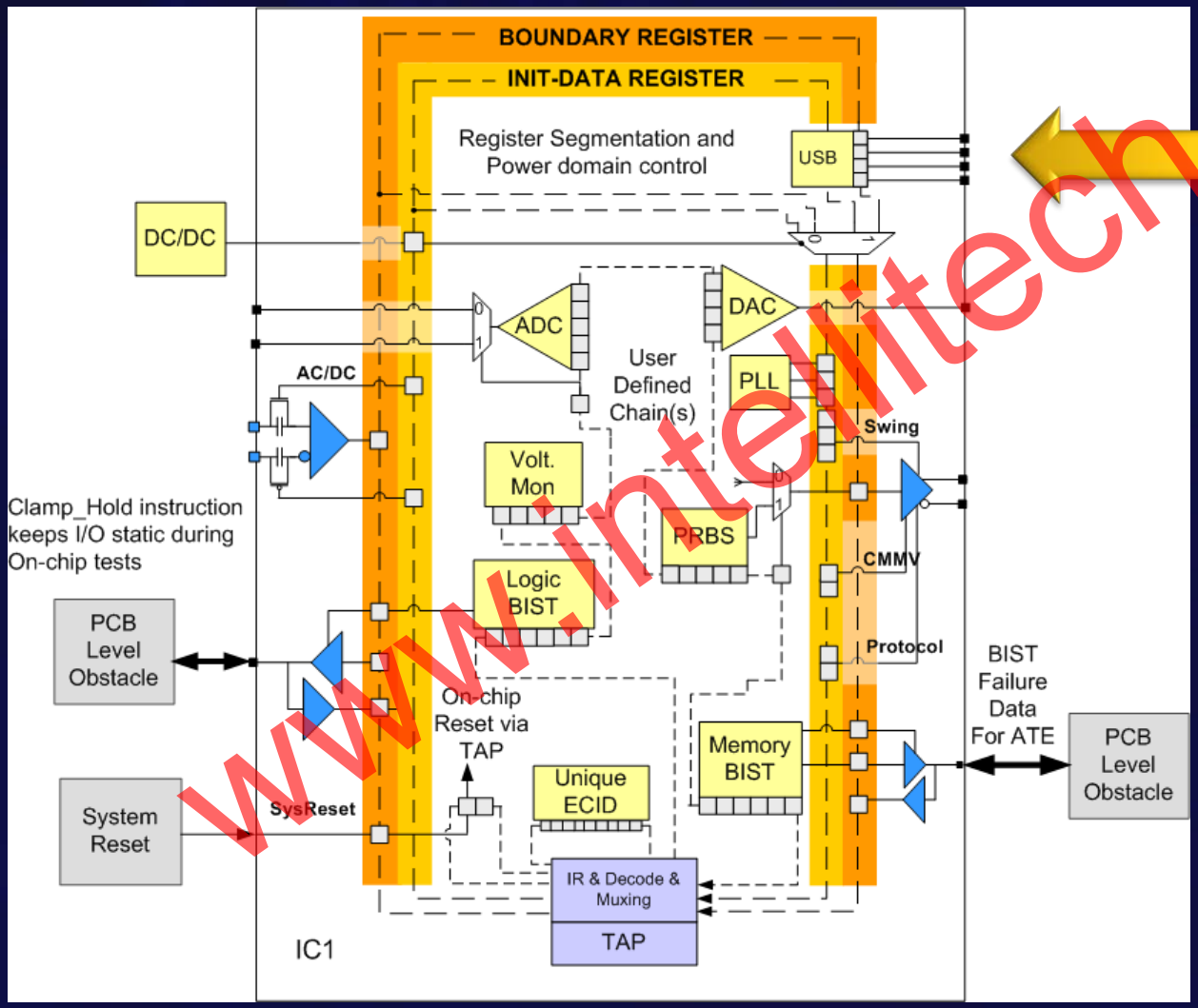
NOPI | NOPO | NOUPD | MON

| PULSE0 | PULSE1 | SHARED

Reset (if present) :

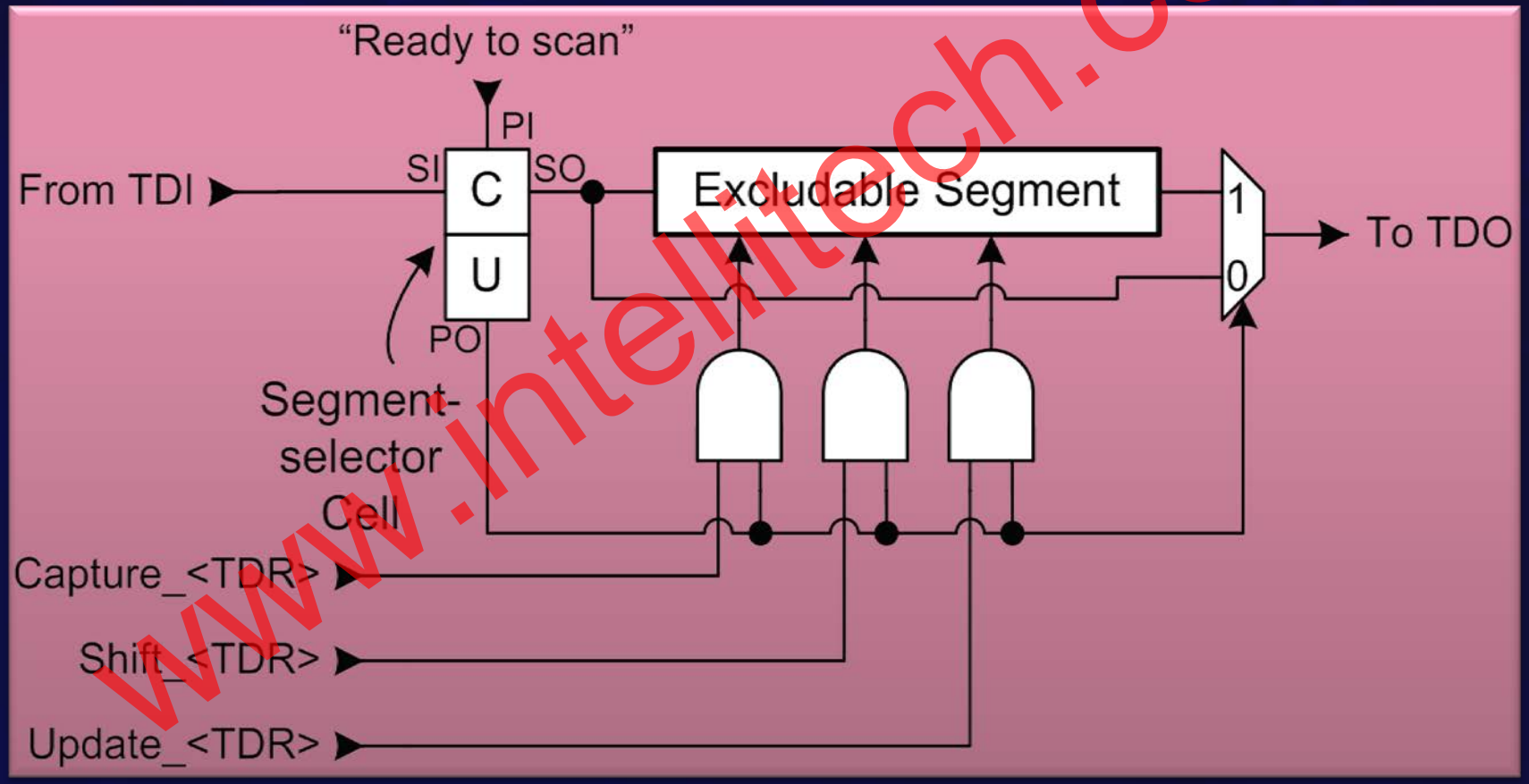
PORRESET | TRSTRESET | TAPRESET | CHRESET

Domain Segmentation of TDRs

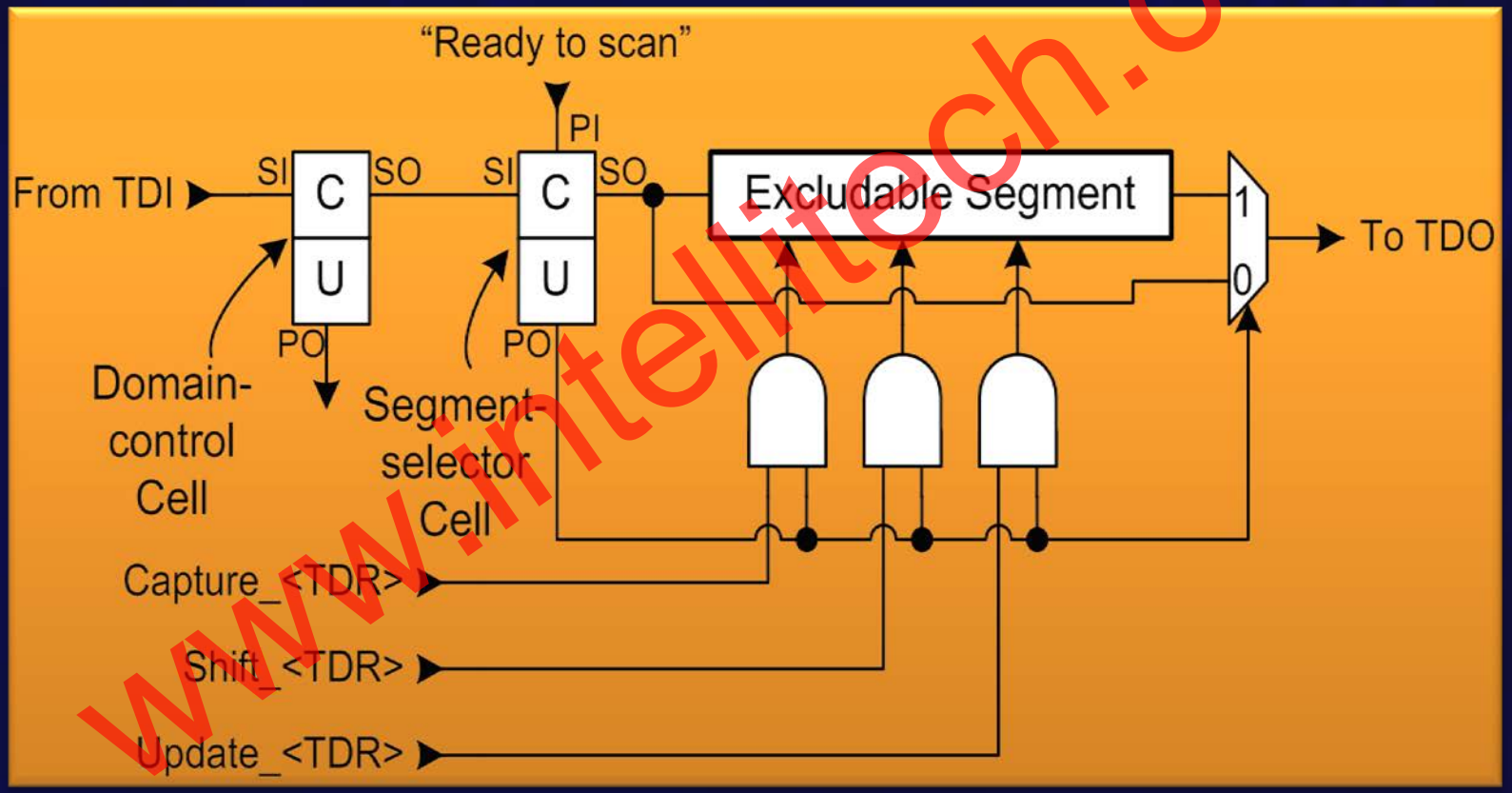


Power is off On USB

Boundary or user defined



Control Domain Bit

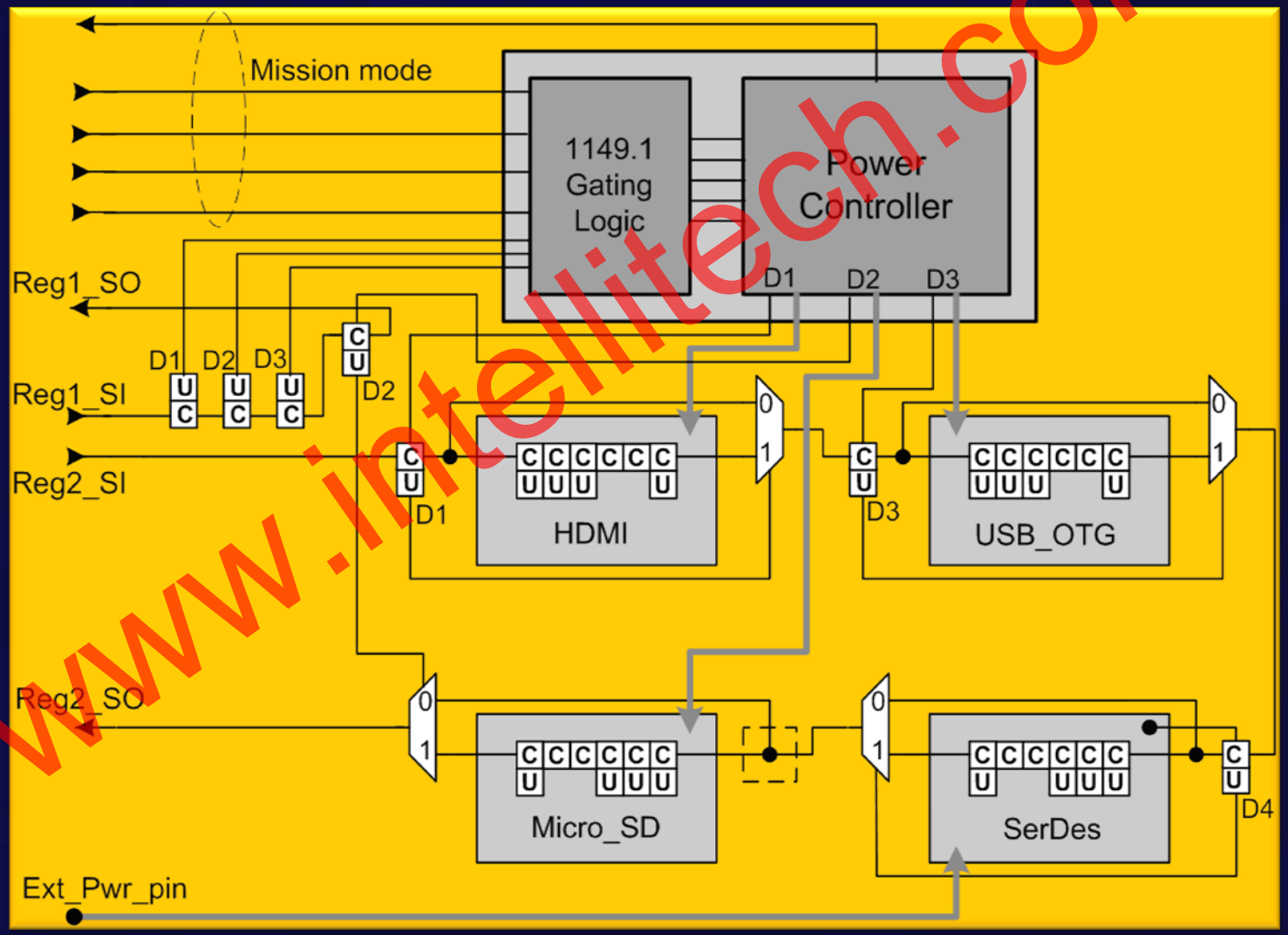


Domain Segmentation

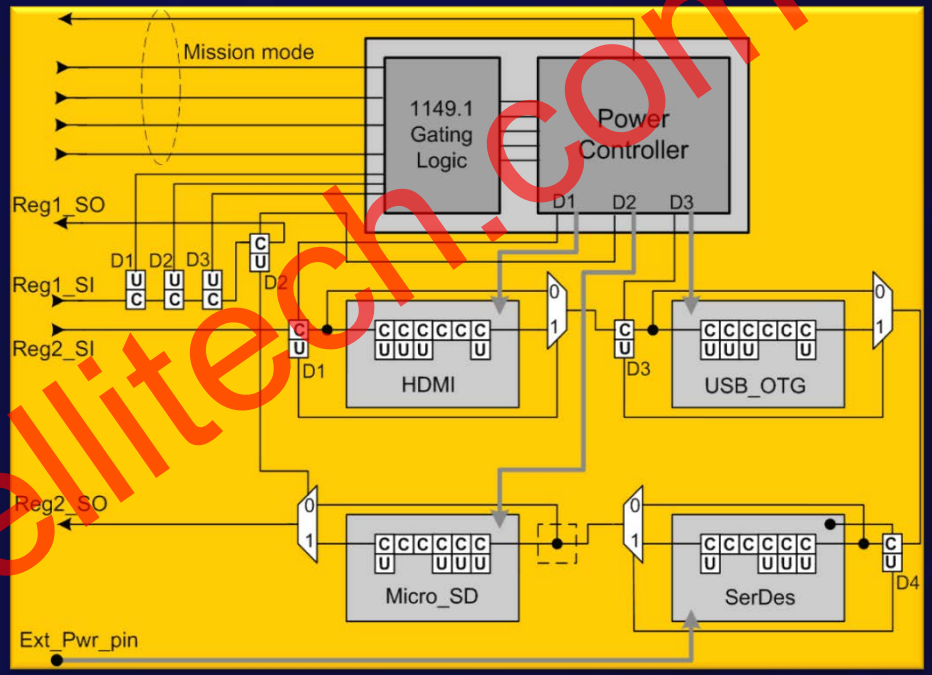
BSDL keywords allow one to describe DOMAIN, or if externally powered, DOMAIN_EXTERNAL and SEGSEL (SEGSTART) and SEGMUX (SEGEND)

<association type>::= **DOMAIN** | **DOMAIN_EXTERNAL** | **SEGSEL | SEGMUX**
<association name>::= <VHDL identifier>

Domain Segmentation

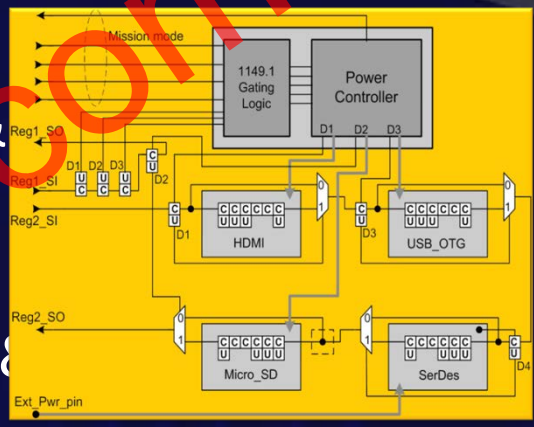


Getting ahead of ourselves with new BSDL constructs



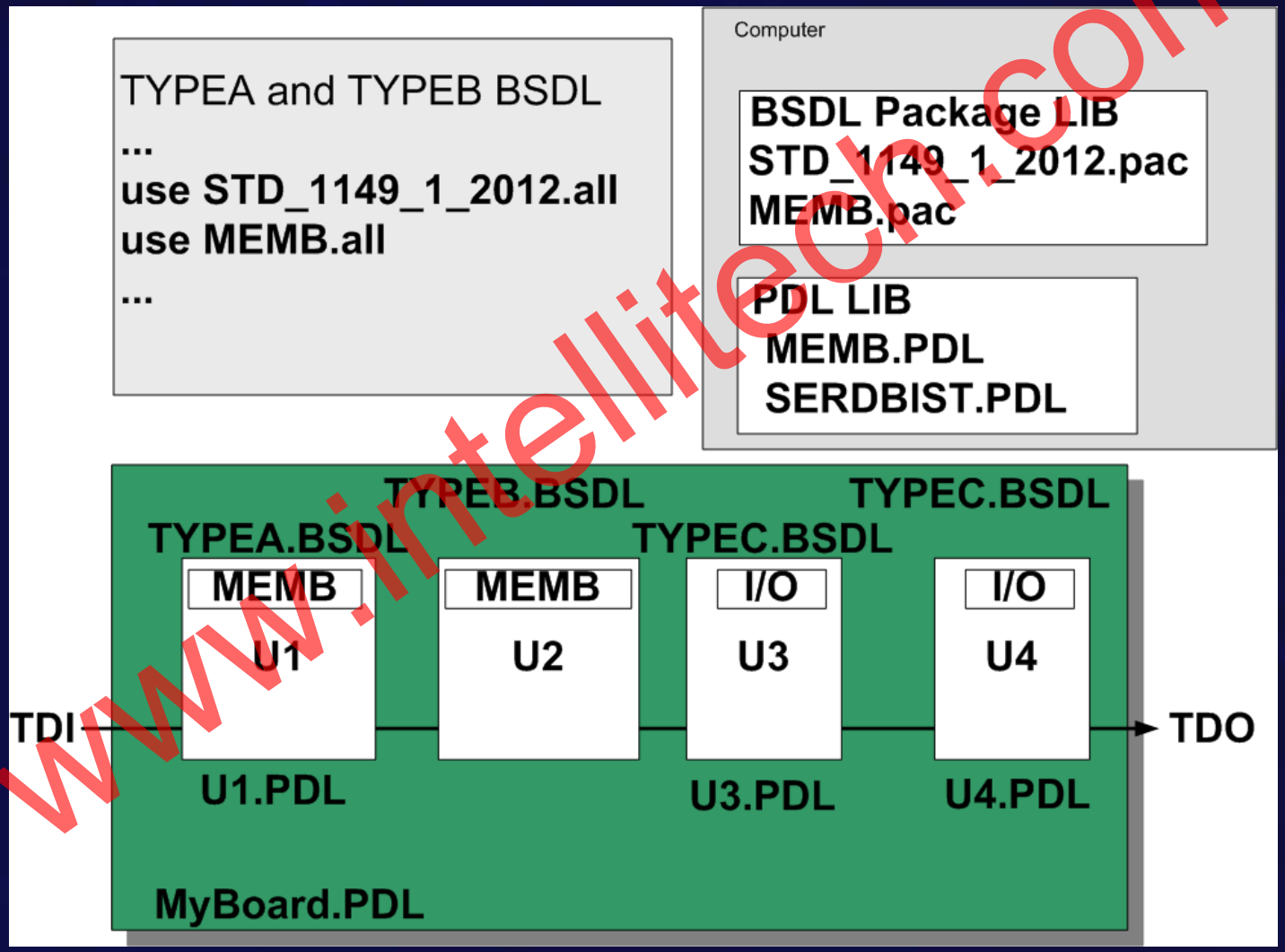

```
attribute REGISTER_ASSEMBLY of PwrDomStruc : entity IS
  "Reg1 ( "&
    "(hdmi_pwr     IS DomCtrl Domain(D1) CHReset), "&
    "(micro_sd_pwr IS DomCtrl Domain(D2) CHReset), "&
    "(usbotg_pwr   IS DomCtrl Domain(D3) CHReset), "&
    "(micro_sd_sel IS SegSel  Domain(D2) Segment(S2) CHReset)), "&
```

Getting ahead of ourselves with new BSDL constructs




```
"Reg2 ( "&
    "(hdmi_sel     IS SegSel  Domain(D1) CHReset), "&
    "(hdmi         IS hdmi_seg), "&
    "(hdmi_mux     IS SegMux), "&
    "(usb_otg_sel  IS SegSel  Domain(D3) CHReset), "&
    "(usb_otg      IS usb_otg_seg), "&
    "(usb_otg_mux  IS SegMux), "&
    "(SerDes_sel   IS SegSel  Domain_External(D4) CHReset), "&
    "(SerDes       IS SerDes_seg), "&
    "(SerDes_mux   IS SegMux), "&
    "(micro_sd_start IS SegStart Segment(S2)), "&
    "(micro_sd     IS microsd_seg), "&
    "(micro_sd_mux IS SegMux) )";
Attribute Register_Port_Association of PwrDomStruc : entity is
  "SerDes_sel : (Ext_Pwr_pin) ";  -- See next clause.
```

Hierarchy supported through package files



Documenting with BSDL & PDL

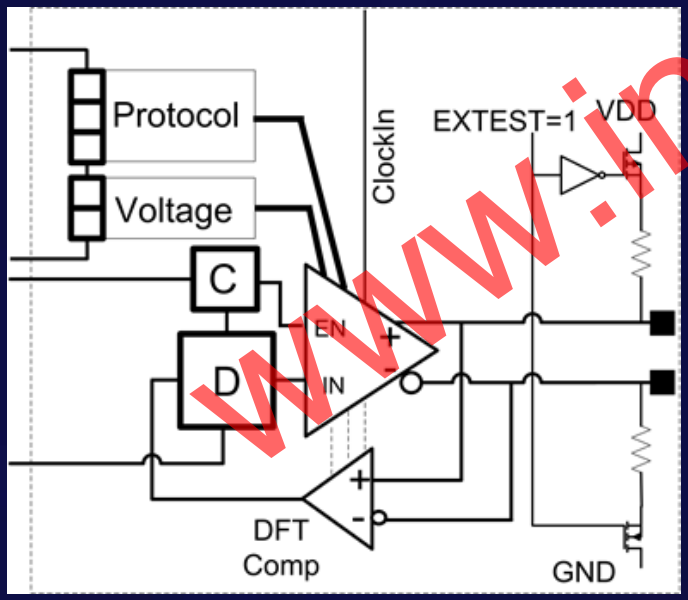
BSDL - Boundary Scan Description Language

PDL - Procedural Description Language

Basic Register Fields

TDR Name

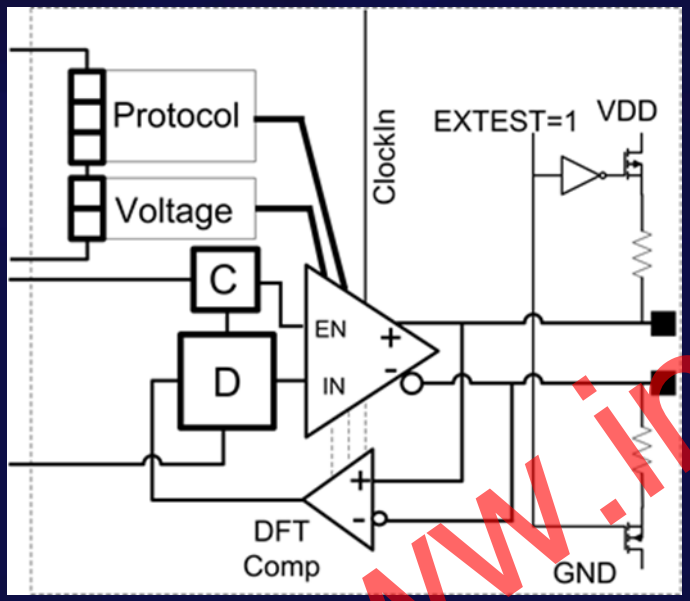
```
attribute REGISTER_FIELDS of INIT_Example :
entity is
"init_data ( "&
"(Clock[5]        IS (504 DOWNTO 500) ), "&
"(Voltage[2]      IS ( 101 DOWNTO 100) ) "&
");"
```



BSDL syntax for “INIT_DATA” and

For Clause 9 user defined TDRs

REGISTER_MNEMONICS




```
attribute REGISTER_MNEMONICS of SERDES :
package is
" Protocol ( “ &
"  OFF (0b000) <I/Os powered down>, "&
" PCIe (0b001) <PCI Express>, "&
" SATA (0b010) <SATA>, "&
" SRIO (0b011) <Serial RapidIO>, "&
" XAUI (0b100) <XAUI>, "&
" Rsvd1 (0b101) <Undefined, do not use>"&
)," &
"Voltage ( " &
" 500MV (b00), "&
" 800MV (b01), "&
"1400MV (b10) <Do not use!>)";
```

Basic Register Fields with Mnemonics

```
attribute REGISTER_FIELDS of INIT_Example : entity is
"init_data ( "&
"(Clock[5]   IS (504 DOWNTO 500)  Default(Clockset(100Mhz) ), "&
"(Protocol[3] IS (302 DOWNTO 300)  Default(Protocol (off) ), "&
"(Voltage[2]   IS ( 101 DOWNTO 100)  RESETVAL(11) ), "&
"(Reserved [20] IS ( 19 DOWNTO 0))"&
")" &
“myTDR ( "&
"(Addr[64] IS (163 DOWNTO 100) ), "&
“(Data[64]  IS (227 DOWNTO 164) ), "&
“(WE[1]      IS (228) RESETVAL(1)   ), "&
“(TempMON[7]  IS (236 DOWNTO 229)) "&
);”
```

User Defined TDR

Sparse definitions compared to register length

Software reads BSDL

Possible to record PDL commands

PROTOCOL1(10)	OFF	0000000000	0000000000
PROTOCOL2(10)	OFF SRIO PCIE XAUI	0000100000	0000100000
SWING(2)		00	00
PLL(2)		10	10
CAMBIST(2)	STOP	00	00
CAMSTATUS(2)	00	10	10
LBIST(2)	RUN	00	00
LBISTSTATUS(1)	0	PASS	PASS
MODESTATUS(1)	0	0	X
STATUS1(1)	0	PASS	PASS

Device PDL (Procedure Definition Language) - Board specific

```
Proc  init_setup {} {

iWrite  Clock       F125Mhz       # use of mnemonics
iWrite  Voltage     800MV
iWrite  Protocol    PCIe
iApply
}

Proc  init_status {} {

iRead  Status(1)   Pass           # use of mnemonics
iApply
}
```

Some PDL Commands

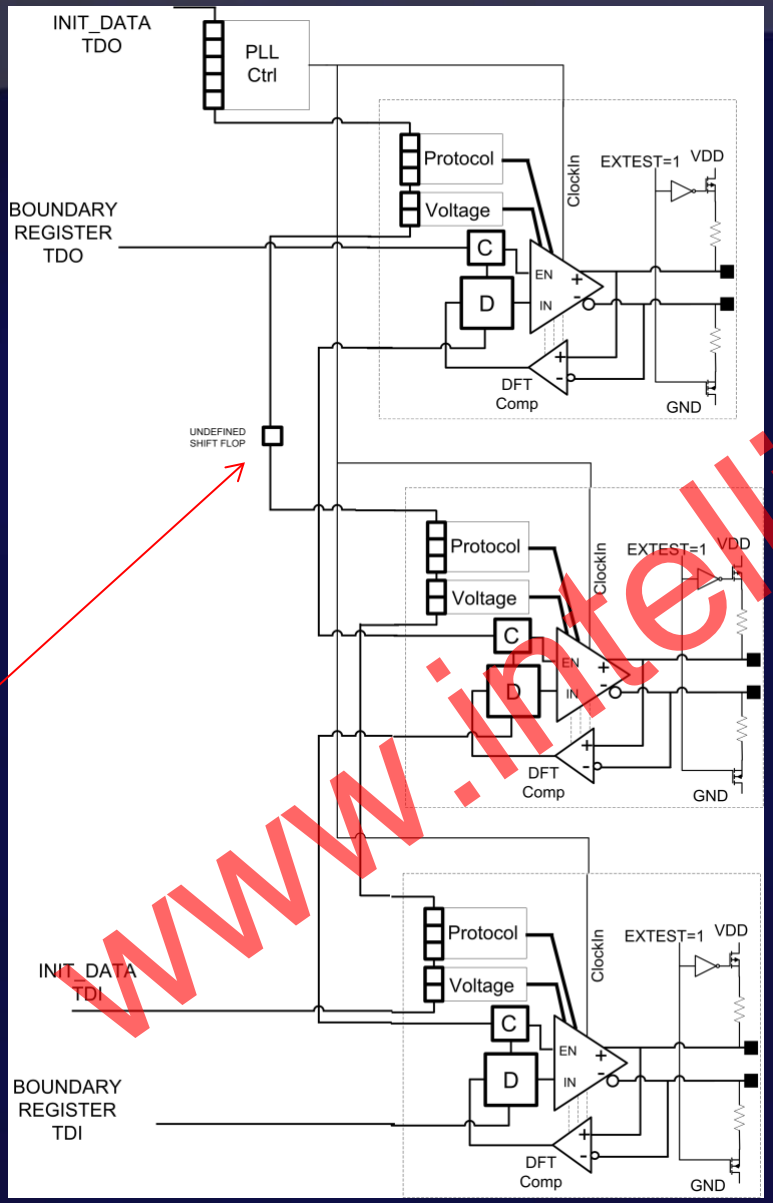
```
iWrite   <reg>  <value>       | mnemonic
iRead    <reg>  <expected> | mnemonic

iApply                                  # perform DR scan RTI-RTI
iPrefix  <dotted path>                  #  iPrefix bank0.serdes
iReset                                  #   Test Logic Reset
iEndState   RTI | PDR                   #   set end state
iRunloop    <TCK-Count>                 #   Loop in RTI
iCall    <iproc name>
```

```
iPrefix U1              #  U1.LBIST

# run some basic tests on registers
iWrite LBIST RUN      # bit-position independent regs
iApply
iRunLoop 300000
iRead  LBISTSTATUS PASS  # check that LBIST passed
iApply
iWrite SWING S400MV      # set differential Swing to
400mv
iWrite PROTOCOL1 SRIO    # set protocol to SRIO
iApply
iWrite CAMBIST RUN       # execute CAM BIST
iApply
iRead  CAMSTATUS DONE
```

Support for IP blocks



Extra Bit

3 SERDES with init_data Registers

Common PLL

BSDL with package files allows Hierachical access to pre-defined Registers

Package File enables Re-usable IP Descriptions to Incorporate Into BSDL

```
PACKAGE XYZ_IO IS

USE Std_1149_1_2012.all;

attribute REGISTER_MNEMONICS of XYZ_IO : package IS

"SerDes_Protocol  (off  (000) <Powered down>, "&
"                  PCIe (001) <PCIExpress>, "&
"                  SATA (010) <SATA>, "&
"                  SRIO (011) <Serial RapidIO>, "&
"                  XAUI (101) <XAUI>, "&
"                 Resvd1 (11X) <Undefined - Do Not Use>), "&

"SerDes_TX_Outputs (off (00)   <Powered down>,  "&
                    -- Output driver swing level
"            Full_Swing (01)    <100% Swing>, "&
"             Swing_p75 (10)    <75% Swing>, "&
"            Swing_p527 (11)    <52.7% Swing
                          Not legal if XAUI is protocol>), "&
```

Local Mnemonics

Package File Cont'd

Local Register Segment Names

```
   attribute REGISTER_FIELDS of XYZ_IO : package IS
      "Channel [5] ( "&
      "Protocol[3] (2, 0, 1) IS DEFAULT (SerDes_Protocol (PCIe)) "&
      "                          RESETVAL(SerDes_Protocol (off)), "&
      "TX_Swing [2] (3, 4) IS DEFAULT (SerDes_TX_Outputs (off)) "&
      "), "&

END XYZ_IO;

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

PACKAGE BODY XYZ_IO IS

   USE Std_1149_1_2012.all;

END XYZ_IO;
```

Local default/reset values

Register assembly – bits predefined defined – length calculated by BSDL reader

REGISTER_ASSEMBLY

3 serdes in blocks – 1 PLL block

```
Use XYZ_IO.all;
Use XYZ_PLL.all;

--  stuff removed for brevity
```

attribute REGISTER_ASSEMBLY of INIT_Example : entity is

"init_data ("&
" (USING XYZ_PLL), " &
" (P1 is Settings), " &
" (USING XYZ_IO), " &
"(Array SerDes(1 TO 2) is Channel), " &
"(dummy[1]), " &
"(SerDes(0) is Channel), " &
"(reserved[105])" &
");"

TDR NAME FROM XYZ_PLL

Array

Association of ports (pins) to registers for diagnostics

attribute REGISTER_PORT_ASSOCIATION ("&

"SerDes00_PRBS (SD_RX(0), SD_RX_B(0), SD_TX(0), SD_TX_B(0)),"&
"SerDes01 (SD_RX(1), SD_RX_B(1), SD_TX(1), SD_TX_B(1)) “;

Register Field

Pins associated with Register Field

Diagnostic and automated fault coverage reports

Software

Free BSDL parser supporting IEEE 1149.1-2012 BSDL constructs

http://www.intellitech.com/bsdl

Free NEBULA software tool

1) Supports PDL and BSDL register fields/mnemonics
2) Interfaces to ModelSIM, VCS and Xilinx JTAG Pod

http://www.intellitech.com/ijtag

Contact me if you're interested in being part of the P1149.1-2012 Ballot Group

cclark@intellitech.com

Conclusion

Infrastructure IP Providers:

Encouraged to use recommended TDR interface

For I/O related IP (SERDES etc) – provide INIT_DATA TDR interface

Encourage customer to design IC with

- CLAMP_HOLD/CLAMP_RELEASE, IC_RESET

IC Designer:

Provide INIT_SETUP/INIT_RUN

-Enables I/O initialization by non-system means

-Control PLLs via INIT_SETUP

-Use IC_RESET

-Enable customer to use in-situ JTAG based tests via CH/CR