Eliminating or minimising Embedded Software Tests.

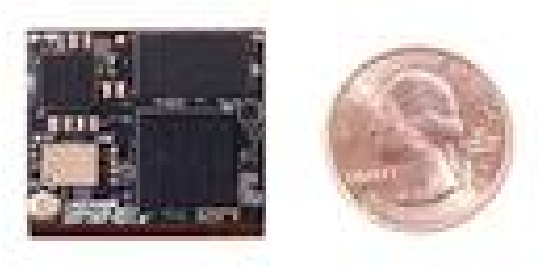
Mick Austin JTAG Technologies,
Jussi Mustola Etteplan. TestForum 2017

Agenda

- Introduction
- Drivers
- Generic Boundary Scan and Restrictions
- Using the JTAG Port for Functional Tests
- Experiences to date
- Conclusions and Benefits
- Questions.

Introduction

- Todays Control Electronics becomes simpler to design but more complex to test. (SOC, SOM)



- iMX processor, DDR, eMMC,
- PCIe, ADC, +++

- Traditional Structural Test methods are not possible.

- Typical Test solutions rely on Embedded special Test software

Advantages of using ESW

- Specialised Embedded Test Software is developed by SW R&D
- Reuse the tests by simply executing the commands.

Disadvantages of using ESW

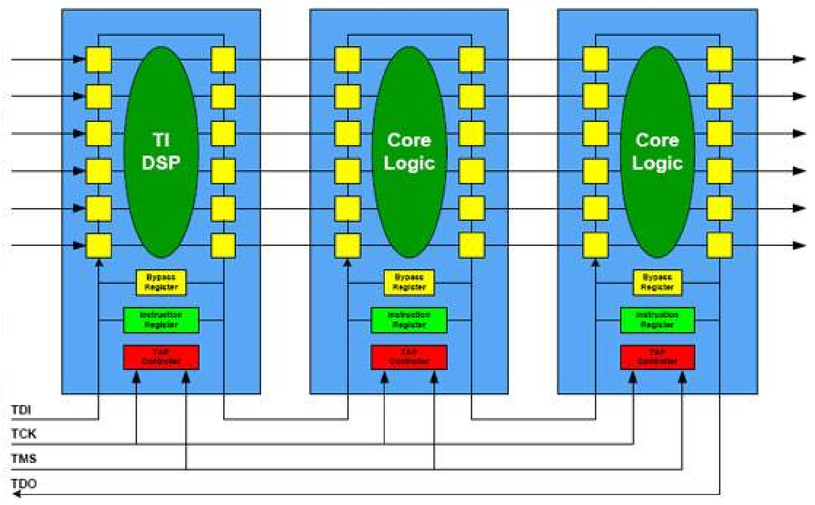
- Tedious to explain to ESW Engineer and to communicate Test Specification. Diiferent competance areas.
- Poor documentation. Typically only comments are written in SW.
- Resources are difficult to find. Focus is on real target SW not Test SW
- Test SW is done as afterthought so probably have to rely on the Product SW to carry out production tests.
- After initial R&D project resource moves to next one. Not available for debugging current tests or future ones.
- Future SW update may change the commands used in the Production tester. Resulting in failed production tests.

Traditional Boundary Scan

- Tradititional Boundary Scan tests

  interconnections between components
- Tests are generated automatically
- Tests executed in standard environment

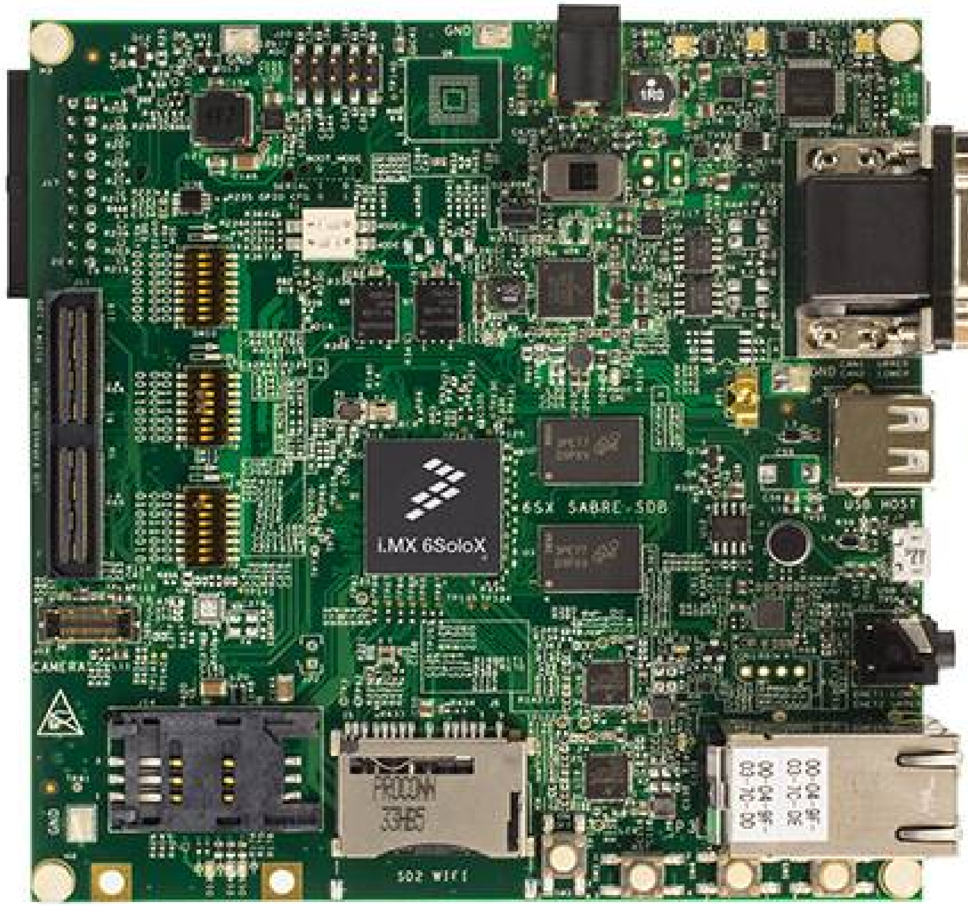
  Teststand or own Test Executive



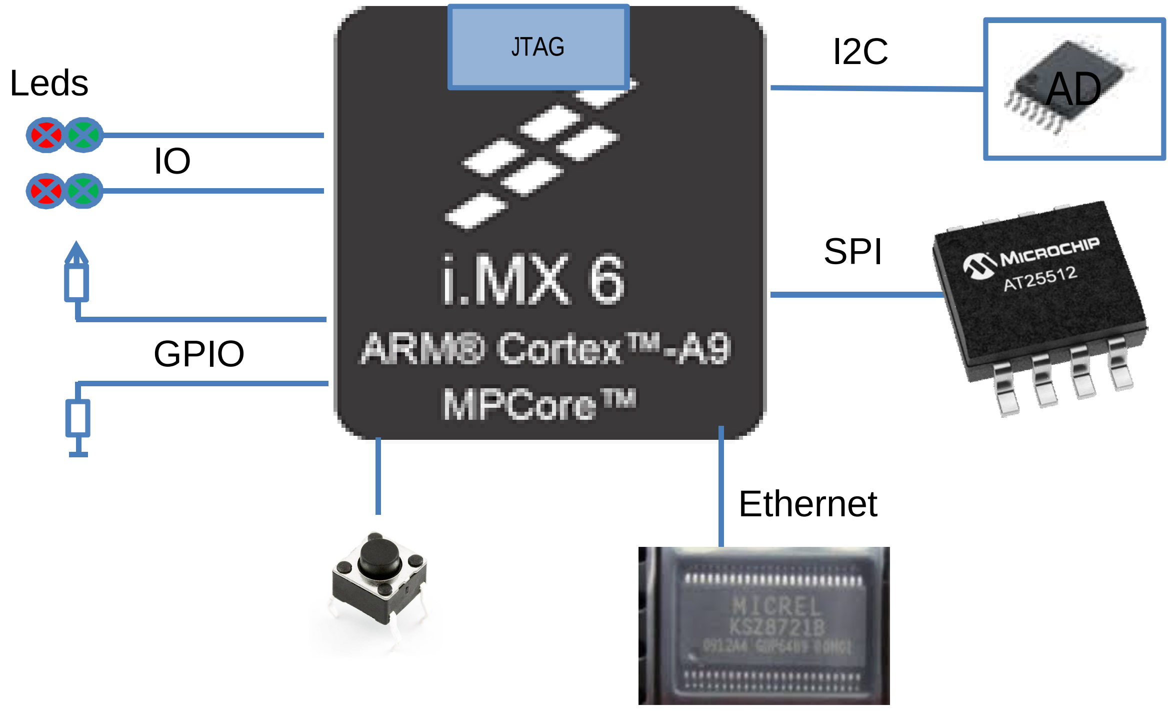
- Possibility to add diagnostics.

- Traditional Boundary Scan testing is more suitable for the rather complex boards, containing several components with JTAG port and/or a lot of I/O between those components. The method is also useful if there is a lot of I/O going out from the board

A typical control board today

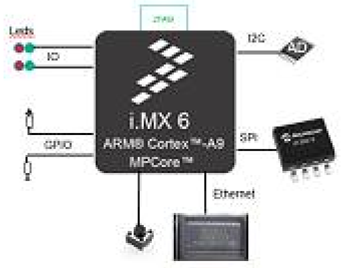


A typical control board today



How is this tested with ESW?

- Write Communication protocols.
- Set up peripheral controllers.
- Write and read information to the CPU
- Take measurements or visually verify
- Process results

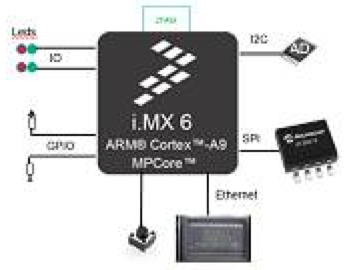


How to test using the JTAG Port with JFT

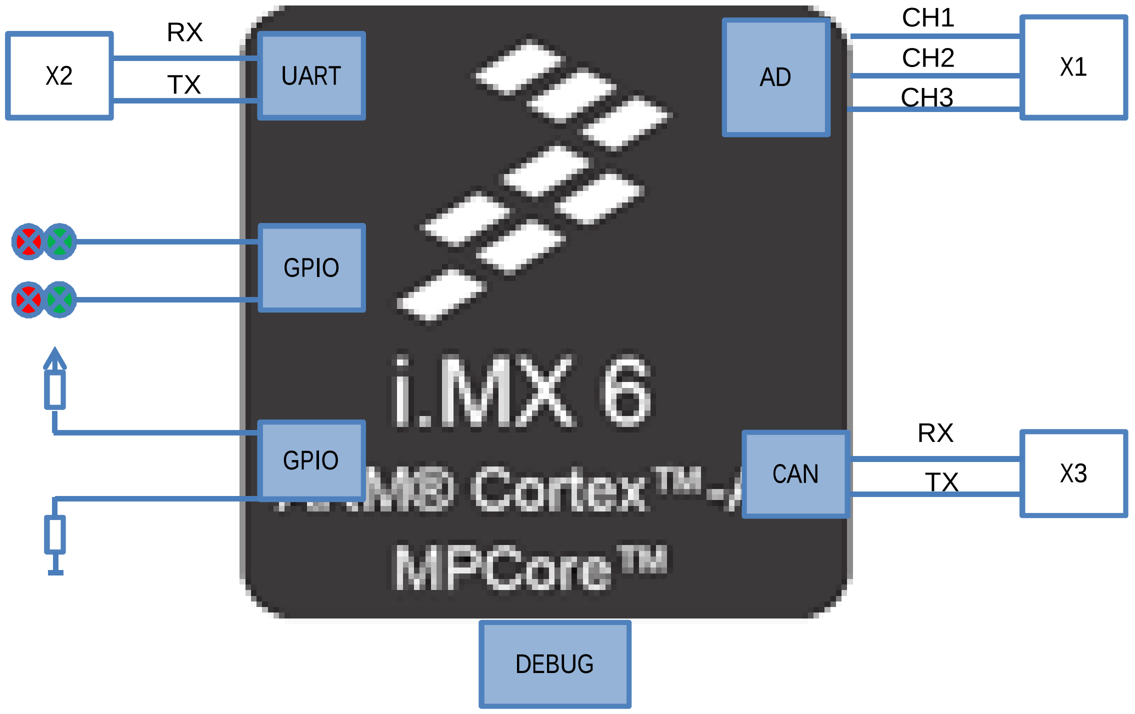
- Read and write to the Bscan pins directly.
- Read digital inputs drive Digital outputs.
- Emulate I2C and SPI (Bit bashing)
- Drive digital outputs and take functional measurement.
- Process results.

- Available in LabVIEW and other languages. LW/VB/DLL/Python.

- Library of serial protocols (I2C SPI).

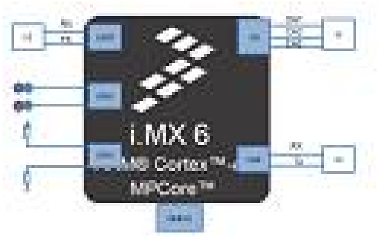


SOC internal core

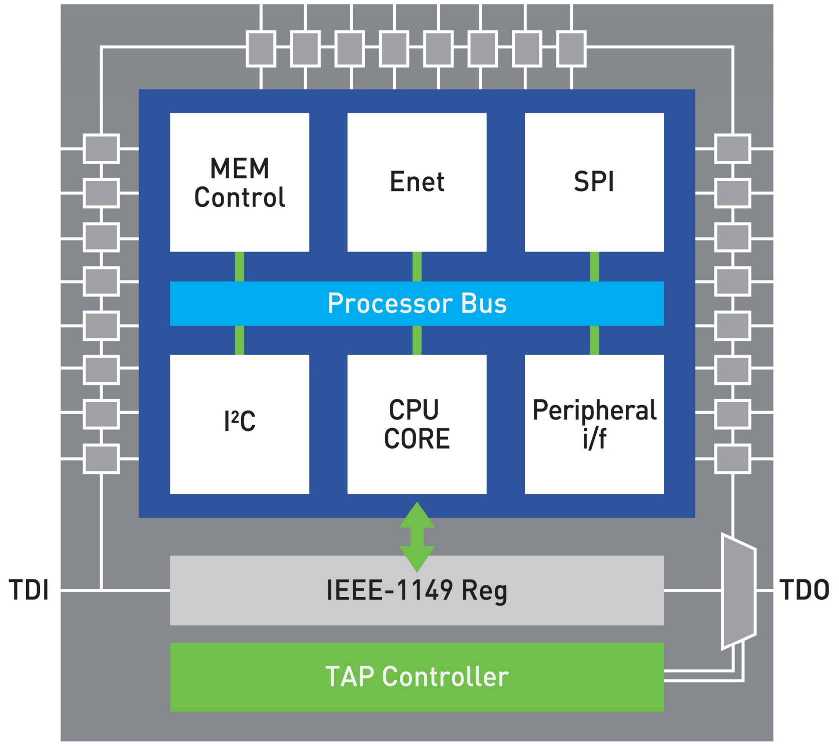


Using the JTAG Port Debug mode

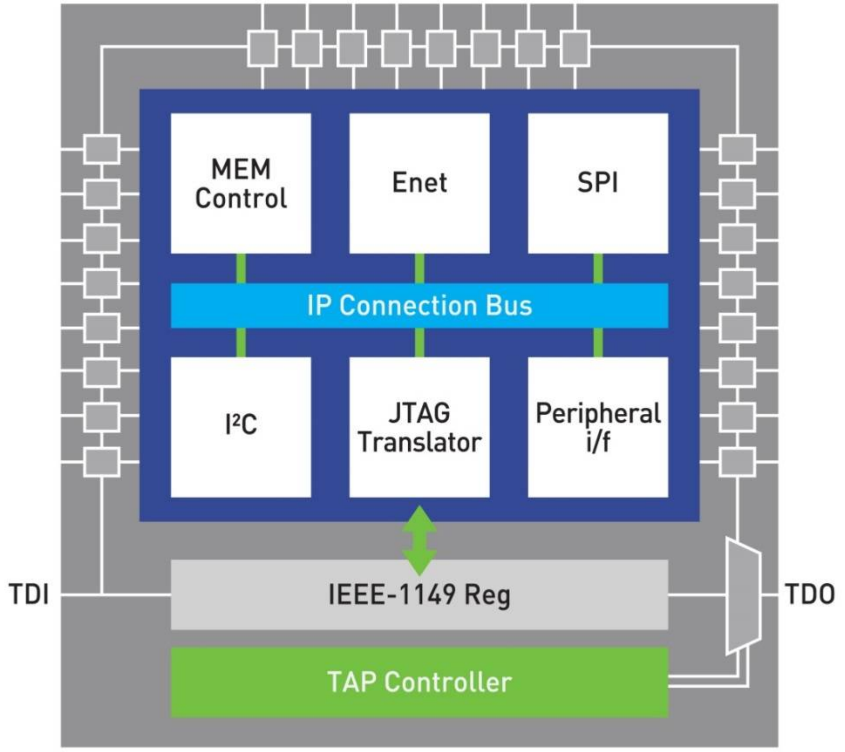
- JTAG CoreCommander gives access to SOC core
- Access is through standard JTAG port
- Read and write to memory locations.
- Each peripheral controller is memory mapped.
- Simple instructions needed to set up peripheral controller
- Read and write directly to the peripheral controller.



CoreCommander for uC's and FPGA's

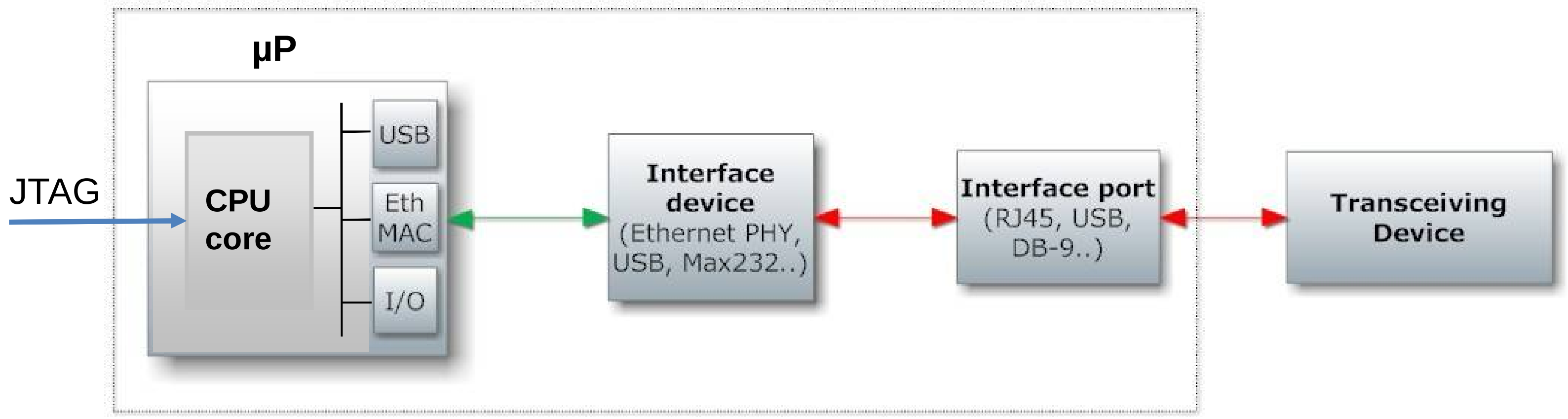


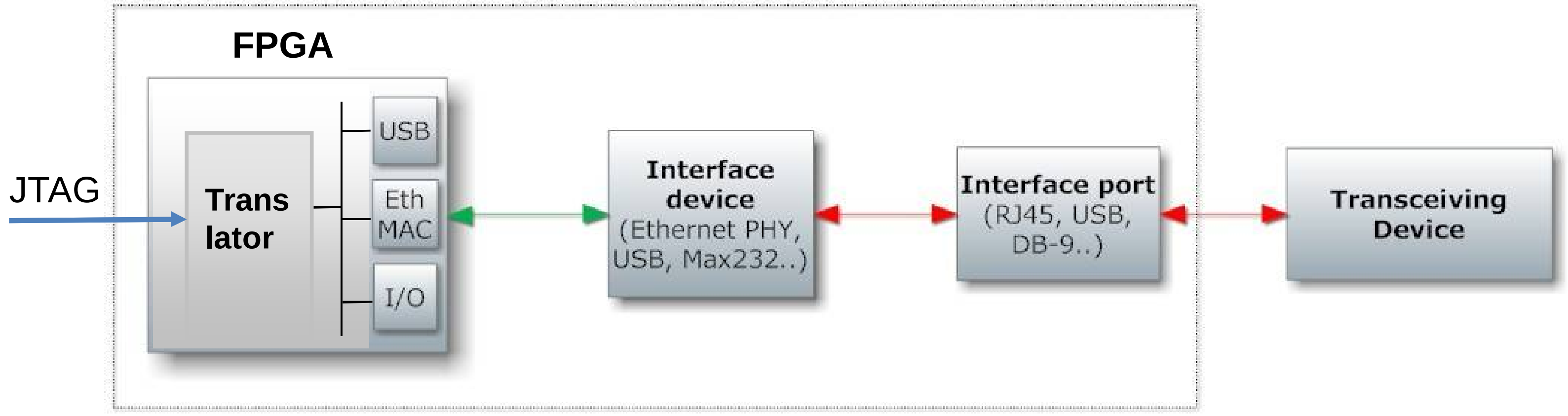
CPU



FPGA

Core Commander Principle


µP
JTAG
CPU core
USB
Eth MAC
I/O
Interface device (Ethernet PHY, USB, Max232..)
Interface port (RJ45, USB, DB-9..)
Transceiving Device


FPGA
JTAG
Trans lator
USB
Eth MAC
I/O
Interface device (Ethernet PHY, USB, Max232..)
Interface port (RJ45, USB, DB-9..)
Transceiving Device

Test connections between µP / peripheral controllers and connected peripheral logic (at-speed)

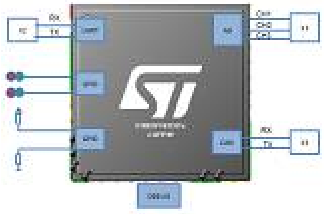
Example read value of external ADC (1)

Project specific parameters

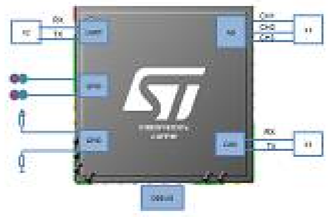
```
board_name = "STM32F103_1"
CORTEX_M4_REF = "CORTEXM4"                #this is the ARM- core
jftsettings.SetConfig("BYPASS")
jftuproc.Init("cortexm4",board_name,CORTEX_M4_REF)


jftuproc.EnterDebug()
print("#(( Starting ADC Example ))#")


print("Enable ADC Clocks...")
STM32F103_RCC.EnableAPB2Clocks ()
print("Enable ADC Clocks...ok")
```



Example read value of external ADC (2)



```
print("Initializing ADC-peripheral..")
STM32F103_ADC.ADC_Init()
print("Initializing ADC-peripheral..ok")

print("Measuring internal temperature(C)...")
temperature = STM32F4_ADC.TM_ADC_MeasureTemperature()
print("Measuring internal temperature(C)...ok")
print("Measured temperature: ",temperature,"degrees Celsius")

print("Measuring VREFINT...")
VREFINT = STM32F4_ADC.TM_ADC_ReadVREFINT()
print("Measuring VREFINT...ok")
print("Measured VREFINT:",VREFINT,"mV")
```

Development Flow

- Try to achieve as much of replacing ESW with JFT only.

  Using only JFT functions the programming is just like using another DIO Board in the Test system.

  Achieve as much test coverage as possible
- Next consider using the Debug port for at speed and functional tests for devices or transmission formats not supported. Asynchronous transmission (RS485) Internal ADC's etc.
- If design has multiple Boundary Scan devices consider normal Boundary Scan for connections and testing peripheral devices.

Experiences to date

Time savings

- Development time to create JFT or CC applications is 50 – 80% of time to develop Embedded Software Tests.
- This time does not include the Platform development needed.

- JFT+CC makes the maintenance of test system easier
  - The whole functionality is in “one place” (Test Platform)
  - Own Test Engineer or 3rd party able to maintain the system without the need of embedded SW developers.
  - To Test Engineer the JFT tests are just another DIO Board.

Conclusions and Benefits (1)

- JFT and CC used with other Functional Test tools gives good coverage
- JFT and CC testing is good enough for mass production test purposes and no need for proprietary tools or knowledge.
- Testing with JFT and CC can be done without firmware
- Test software can be done by Test Engineers or third party. Engineer working on the Test solution does not need special ESW tools.
- Tools can be purchased from the vendor with latest update and support.

Conclusions and Benefits (2)

- JFT and CC test routines are fast to do → save development time.
- Test development can be separate and independent process, which doesn't need R&D software resources
- Testing development and sequencing can be done as table top setup before fixturing is ready
- No Need for additional special Test software download in production, results in shorter test time.

Etteplan

JTAG TECHNOLOGIES®

JTAG TECHNOLOGIES™

We *are* boundary-scan.™