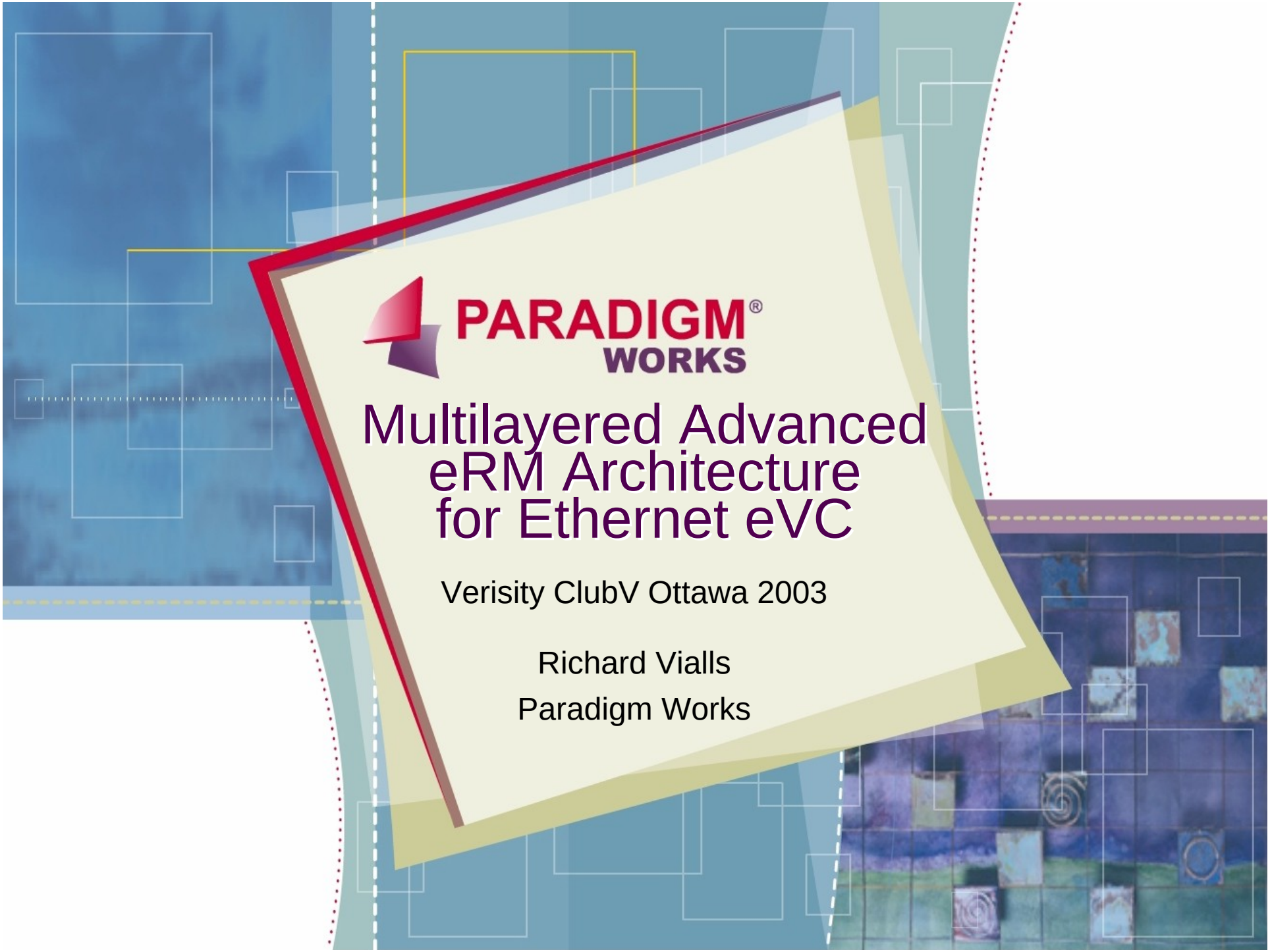


Multilayered Advanced eRM Architecture for Ethernet eVC

Verisity ClubV Ottawa 2003

Richard Vialls

Paradigm Works

PARADIGM® WORKS

Introduction

- Layering is the new ‘buzz word’ in ***e***VC methodology
- This presentation aims to explain why layering is important
- Will use 10G Ethernet as an example

PARADIGM® WORKS

Traditional approach to data generation

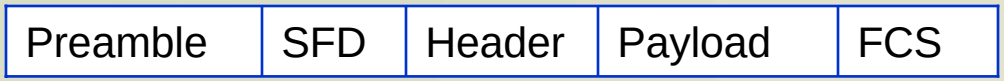
- High level data structure with control fields for low level behaviour:

```
struct my_packet {
    header : my_header_s;
    payload : my_payload_s;
    parity_error_positions : list of uint;
};
```

PARADIGM® WORKS

A complex protocol: Ethernet over XSBI

Ethernet packet:



XGMII columns:

/S/	data	data	data
data	data	data	data
data	data	data	data

...
...

XSBI blocks:

10	0x78	data	data	data	data	data	data	data
01	data	data	data	data	data	data	data	data
01	data	data	data	data	data	data	data	data

...
...

PARADIGM® WORKS

Single layer approach breaks down

- Difficult to control lower layer behaviour from high layer data structures
- Often want to concentrate on lower-layer testing
- Often need control of behaviour between ‘packets’
- Often need to co-ordinate low and high level behaviour

PARADIGM® WORKS

Enter layering

- Layering allows separation of control and observability
- Should break layers at natural boundaries for protocol
- Layering has only become viable as a result of introduction of ***e***RM
- Use of ***e***RM very important to get full advantages of layered approach

PARADIGM® WORKS

Layering: where and why?

- Most appropriate where protocol has layers
  - e.g. ATM/Utopia
- Allows for independent control/monitor of behaviour at each layer
  - Sequence driver(s) control each layer
  - Monitor(s) track behaviour of each layer

PARADIGM® WORKS

Typical Applications

- Individual layer *e*VC
  - e.g.: Utopia *e*VC, ATM *e*VC
    - Handles single layer per *e*VC
    - Provides hooks as appropriate to higher/lower layers
- Multiple layer *e*VC
  - e.g. Paradigm Works, Ethernet *e*VC
    - All layers handled within one *e*VC

PARADIGM® WORKS

Requirements for lowest layer

- Sequence driver
  - higher layers provide connector sequence(s)
- Monitor (with scoreboard hooks)
  - higher layers extend scoreboard hooks
- In other words...
  - Just a normal *e*RM compatible *e*VC

PARADIGM® WORKS

Requirements for higher layers

- Connector sequence
  - Sequence kind of lower layer *e*VC
    - E.g. PW_ATM_ATM pw_utop_sequence
  - Usually extends **`pre_do()`** method of sequence to grab data from higher layer
  - Constrain default lower layer sequence
- Extension to lower layer scoreboard hook(s)
  - Used to extract data from lower layer *e*VC

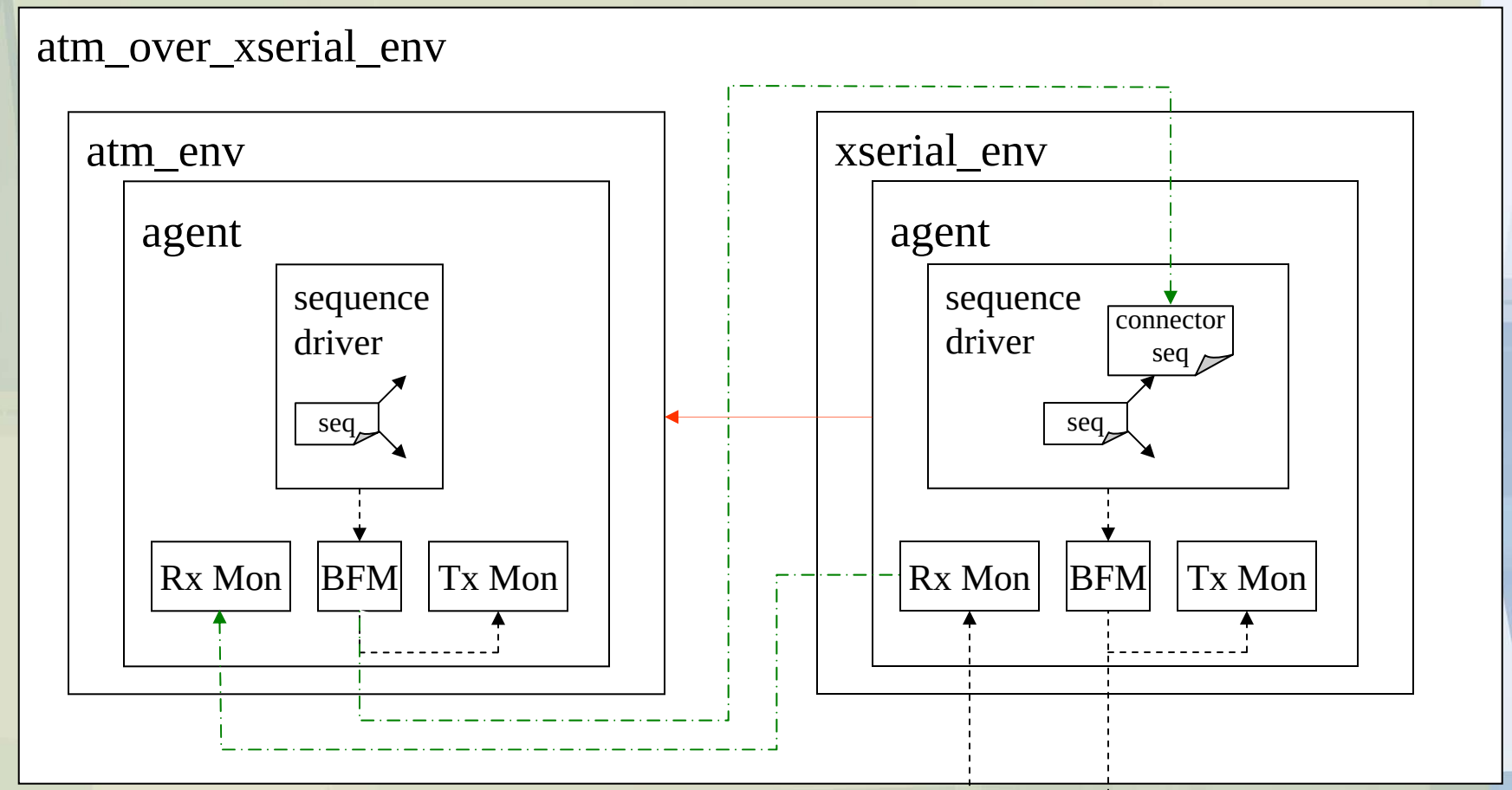
PARADIGM® WORKS

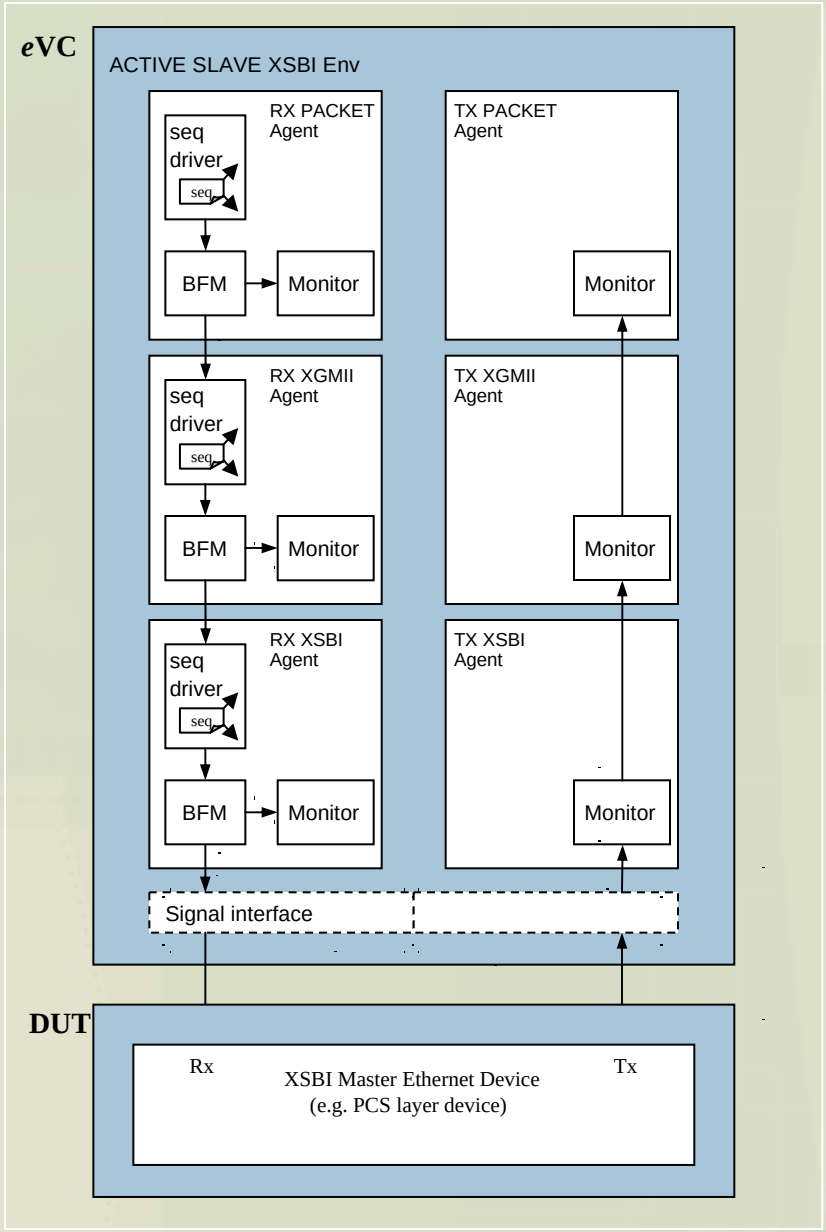
More on higher layers

- Important distinction
  - Higher layer *e*VCs usually don't have a physical (signal based) interface
  - Instead, they provide a method based interface
- Possible to have optional physical interface

PARADIGM® WORKS

Simple layering example



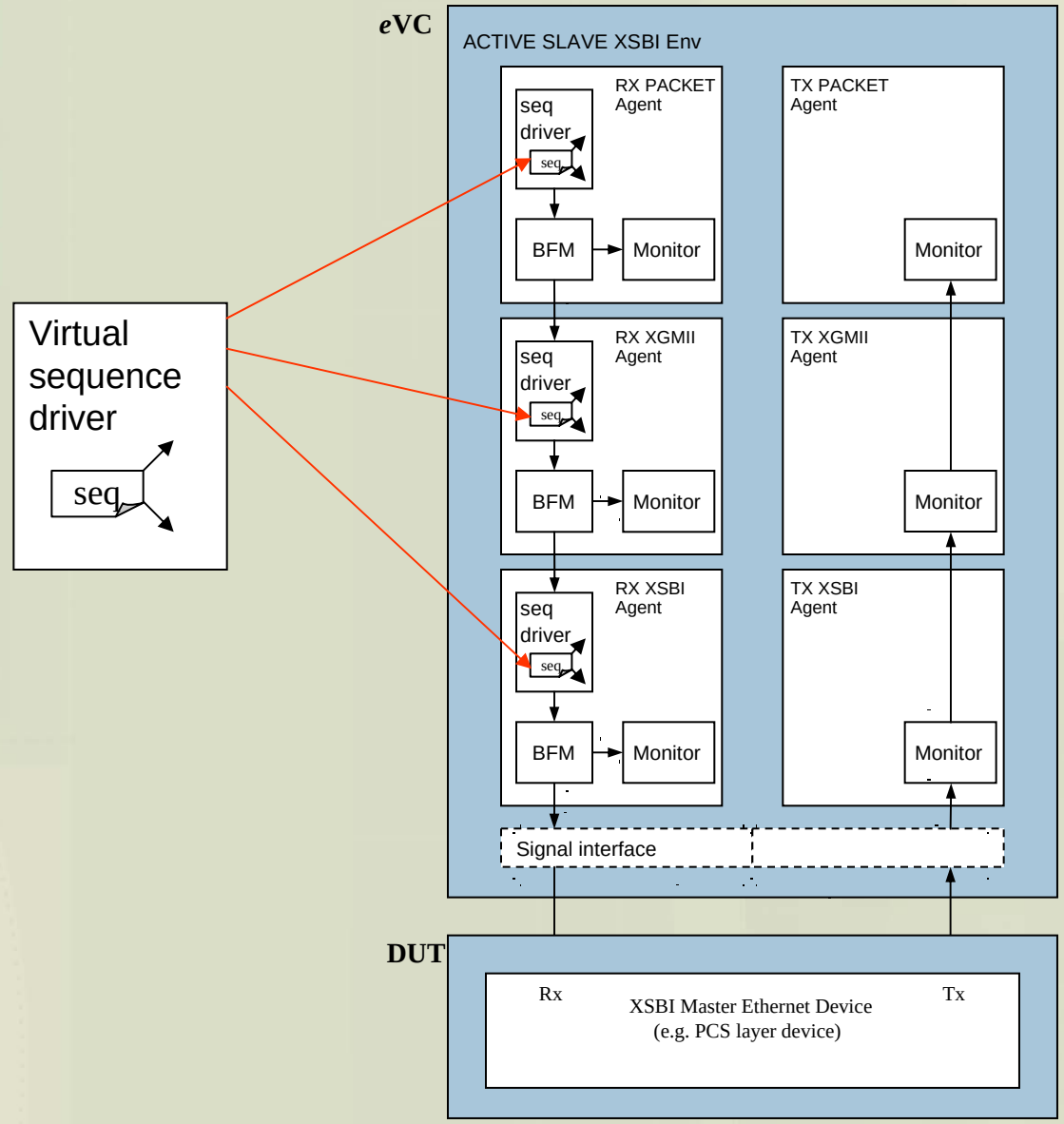


Full Example: Ethernet *e*VC

PARADIGM® WORKS

Co-ordinated testing – Virtual sequences

- Can build virtual sequences to control simultaneous behaviour across multiple layers.
- E.g. 1: Ethernet packet with XSBI block error on last block.
- E.g. 2: Disconnection of XAUI signal during packet.



PARADIGM® WORKS

More on Virtual Sequences

- Can also co-ordinate behaviour across multiple layers AND multiple ***e***VCs.
- E.g.: Set up DMA transfer on PCI interface to receive Ethernet packet that has error in last XSBI block.

PARADIGM® WORKS

Summary

- Layering solves complex test scenario problems
- Layering will be necessary to solve tomorrows verification problems.
- Paradigm Works Ethernet ***e***VC is state-of-the-art layered ***e***VC
- For more info on layering see Verisity's ***e***RM documentation.