Life Cycle Testing for SSD Production

Marilyn Kushnick

R&D Engineer

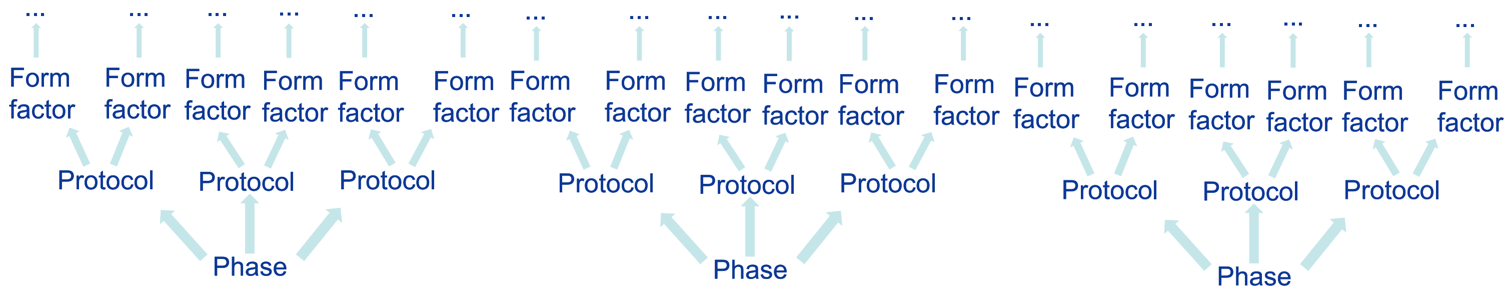
Advantest

Introduction

- Current challenges in SSD test
- Concept of a Test Primitive
- The life cycle of SSD testing
- How the Primitive concept can be applied to the life cycle test phases

Current Challenges in SSD Test

- Need to test each phase of SSD development
- Each phase has unique test focus



... ...

Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor Form factor

Protocol Protocol Protocol Protocol Protocol Protocol Protocol Protocol Protocol

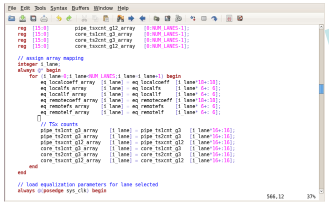
Phase Phase Phase

- Is there a way to combine this?

Concept of a Test Primitive

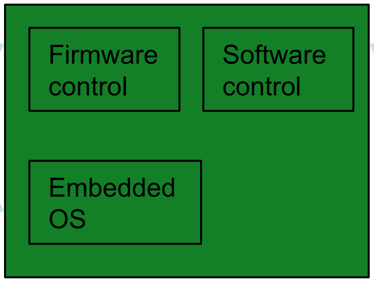
- Combination of:
  - Firmware/software board
  - Power supply board
- The Primitive insert into different tester frameworks to address various testing needs

Test Primitive is the Heart of Testing



```
    reg  [15:0]         pipe_tsxcnt_g12_array   [0:NUM_LANES-1];
    reg  [15:0]         core_ts1cnt_g3_array    [0:NUM_LANES-1];
    reg  [15:0]         core_ts2cnt_g3_array    [0:NUM_LANES-1];
    reg  [15:0]         core_tsxcnt_g12_array   [0:NUM_LANES-1];

    // assign array mapping
    integer i_lane;
    always @* begin
        for (i_lane=0;i_lane<NUM_LANES;i_lane=i_lane+1) begin
            eq_localcoeff_array  [i_lane] = eq_localcoeff  [i_lane*18+:18];
            eq_localfs_array     [i_lane] = eq_localfs     [i_lane* 6+: 6];
            eq_locallf_array     [i_lane] = eq_locallf     [i_lane* 6+: 6];
            eq_remotecoeff_array [i_lane] = eq_remotecoeff [i_lane*18+:18];
            eq_remotefs_array    [i_lane] = eq_remotefs    [i_lane* 6+: 6];
            eq_remotelf_array    [i_lane] = eq_remotelf    [i_lane* 6+: 6];

            // TSx counts
            pipe_ts1cnt_g3_array    [i_lane] = pipe_ts1cnt_g3   [i_lane*16+:16];
            pipe_ts2cnt_g3_array    [i_lane] = pipe_ts2cnt_g3   [i_lane*16+:16];
            pipe_tsxcnt_g12_array   [i_lane] = pipe_tsxcnt_g12  [i_lane*16+:16];
            core_ts1cnt_g3_array    [i_lane] = core_ts1cnt_g3   [i_lane*16+:16];
            core_ts2cnt_g3_array    [i_lane] = core_ts2cnt_g3   [i_lane*16+:16];
            core_tsxcnt_g12_array   [i_lane] = core_tsxcnt_g12  [i_lane*16+:16];
        end
    end

    // load equalization parameters for lane selected
    always @(posedge sys_clk) begin
                                                                566,12        37%
```

FPGA

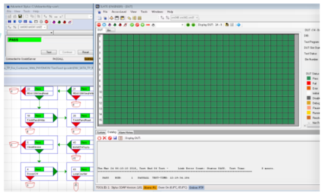


Firmware control

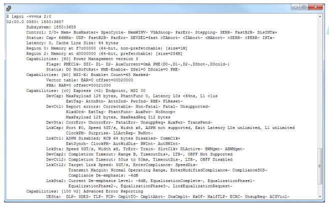
Software control

Embedded OS

Example of software/ firmware board

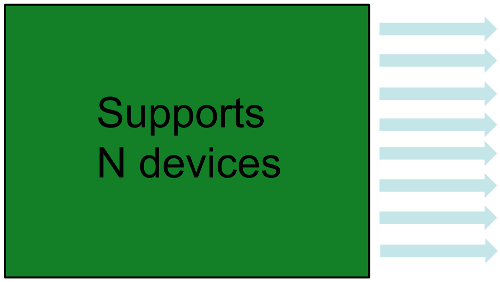


UL Software



```
$ lspci -vvvns 2:0
02:00.0 0580: 1850:3657
        Subsystem: 1850:3658
        Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx+
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0, Cache Line Size: 64 bytes
        Region 0: Memory at f7c00000 (64-bit, non-prefetchable) [size=1M]
        Region 2: Memory at d0000000 (64-bit, prefetchable) [size=256M]
        Capabilities: [80] Power Management version 3
                Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
                Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-
        Capabilities: [b0] MSI-X: Enable+ Count=65 Masked-
                Vector table: BAR=0 offset=00020000
                PBA: BAR=0 offset=00021000
        Capabilities: [c0] Express (v2) Endpoint, MSI 00
                DevCap: MaxPayload 128 bytes, PhantFunc 0, Latency L0s <64ns, L1 <1us
                        ExtTag- AttnBtn- AttnInd- PwrInd- RBE+ FLReset-
                DevCtl: Report errors: Correctable- Non-Fatal- Fatal- Unsupported-
                        RlxdOrd- ExtTag- PhantFunc- AuxPwr- NoSnoop+
                        MaxPayload 128 bytes, MaxReadReq 512 bytes
                DevSta: CorrErr+ UncorrErr- FatalErr- UnsuppReq+ AuxPwr- TransPend-
                LnkCap: Port #0, Speed 5GT/s, Width x8, ASPM not supported, Exit Latency L0s unlimited, L1 unlimited
                        ClockPM- Surprise- LLActRep- BwNot-
                LnkCtl: ASPM Disabled; RCB 64 bytes Disabled- CommClk+
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
                LnkSta: Speed 5GT/s, Width x8, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
                DevCap2: Completion Timeout: Range B, TimeoutDis+, LTR-, OBFF Not Supported
                DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled
                LnkCtl2: Target Link Speed: 5GT/s, EnterCompliance- SpeedDis-
                         Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
                         Compliance De-emphasis: -6dB
                LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete-, EqualizationPhase1-
                         EqualizationPhase2-, EqualizationPhase3-, LinkEqualizationRequest-
        Capabilities: [100 v2] Advanced Error Reporting
                UESta:  DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq- ACSViol-
```

Embedded Linux



Supports
N devices

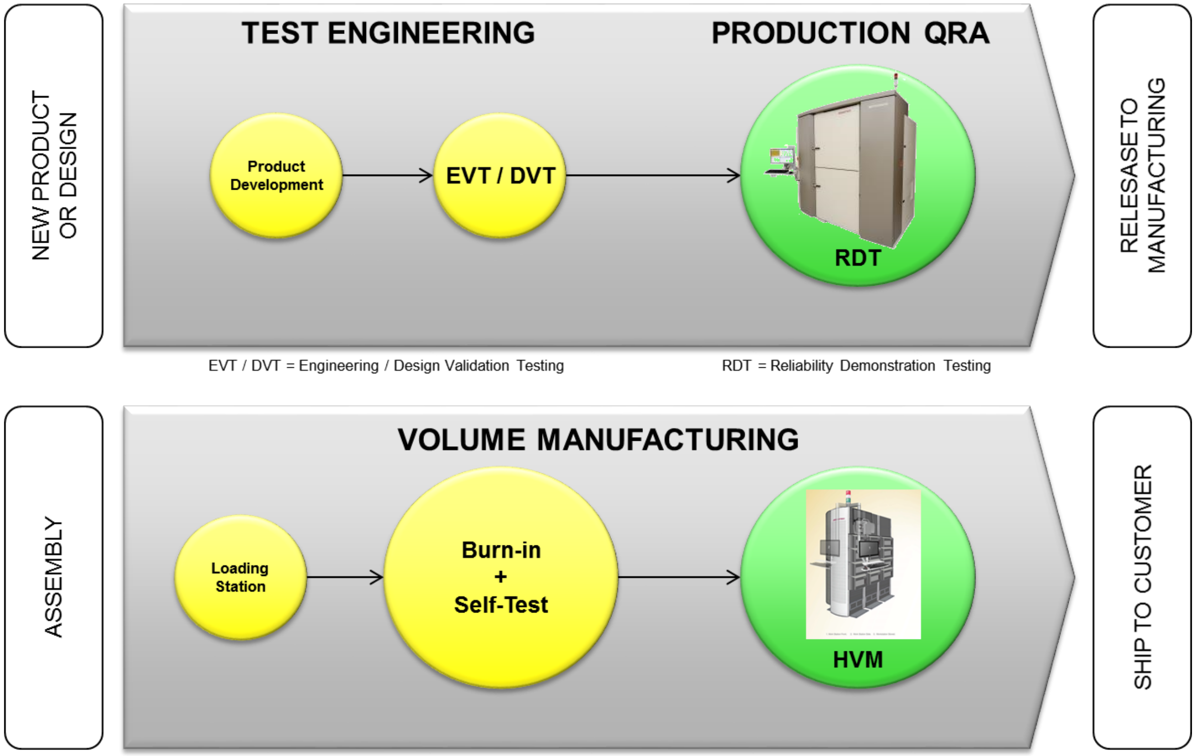
Example of power supply board

Power control for each device individually

Primitive Benefits

- Can use the same:
  - User-level software
  - System-level control
- Across different tester types of the SSD life cycle
- Scalable to configurations across life cycle

The Life Cycle of SSD Test



NEW PRODUCT OR DESIGN

TEST ENGINEERING

Product Development → EVT / DVT →

PRODUCTION QRA

RDT

RELESASE TO MANUFACTURING

EVT / DVT = Engineering / Design Validation Testing

RDT = Reliability Demonstration Testing

ASSEMBLY

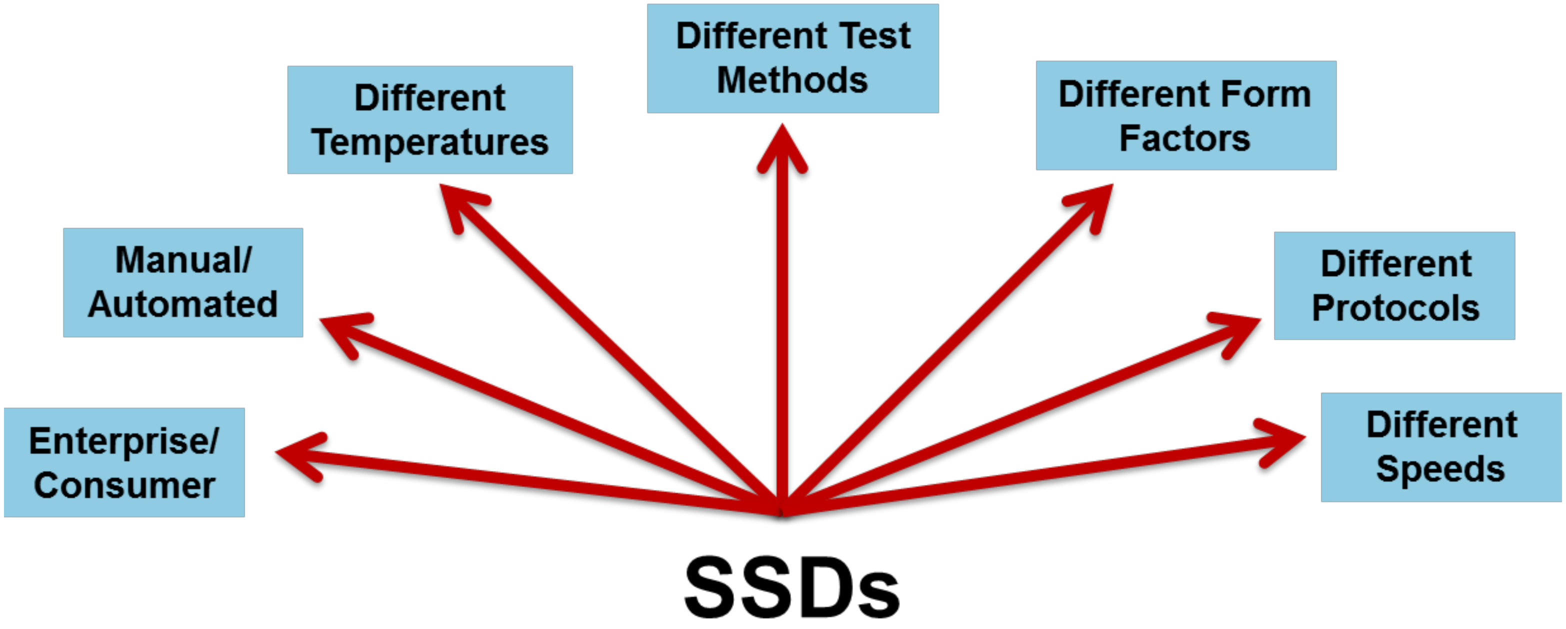
VOLUME MANUFACTURING

Loading Station → Burn-in + Self-Test → HVM

SHIP TO CUSTOMER

FT = Functional Testing

Test Requirements



Different Test Methods

Different Temperatures

Different Form Factors

Manual/ Automated

Different Protocols

Enterprise/ Consumer

Different Speeds

SSDs

The Primitive for Development

- Primitive in small tester
  - Develop test programs
  - Doesn’t burn power like large systems
  - Focused on basic functionality verification

Would use one primitive

Example of a development type of tester and its loadboard

The Primitive for Reliability Testing

- Primitive in environmental chamber
  - Use same software
  - Use development test programs
- Test end of life of devices and corner case exposure of the controller/NAND interaction

Scalability of Primitive for RDT

- Same primitive as development tester can be multiplied for an RDT tester
  - Could use up to 16 primitives for this type of system

Would use two primitives per loadboard

Example of an RDT type of tester and its loadboard

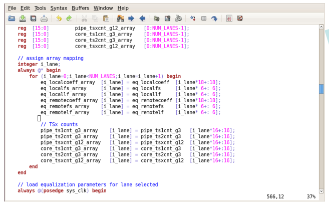
The Primitive for High Volume Manufacturing

- Primitive in rack system
  - Use high-density power supply and firmware boards for the primitive
  - Use same software
  - Use same development test programs
- Shorter tests for device confirmation for manufacturing quality

High-Density Primitive

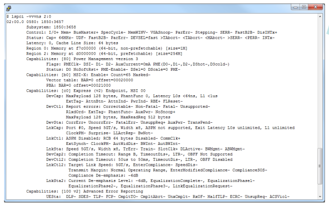
- Same embedded Linux and FPGA architecture
  - No need to change how tests are run
  - Same method, just larger parallelism

High-Density Primitive Examples



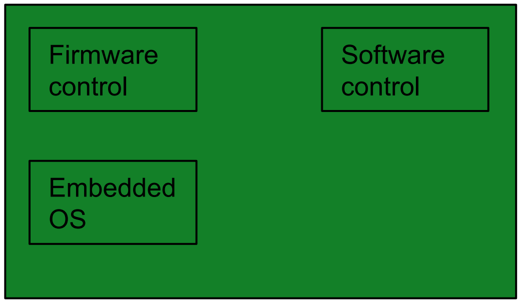
```
reg  [15:0]        pipe_tsxcnt_g12_array   [0:NUM_LANES-1];
reg  [15:0]        core_ts1cnt_g3_array    [0:NUM_LANES-1];
reg  [15:0]        core_ts2cnt_g3_array    [0:NUM_LANES-1];
reg  [15:0]        core_tsxcnt_g12_array   [0:NUM_LANES-1];

// assign array mapping
integer i_lane;
always @* begin
    for (i_lane=0;i_lane<NUM_LANES;i_lane=i_lane+1) begin
        eq_localcoeff_array  [i_lane] = eq_localcoeff  [i_lane*18+:18];
        eq_localfs_array     [i_lane] = eq_localfs     [i_lane* 6+: 6];
        eq_locallf_array     [i_lane] = eq_locallf     [i_lane* 6+: 6];
        eq_remotecoeff_array [i_lane] = eq_remotecoeff [i_lane*18+:18];
        eq_remotefs_array    [i_lane] = eq_remotefs    [i_lane* 6+: 6];
        eq_remotelf_array    [i_lane] = eq_remotelf    [i_lane* 6+: 6];

        // TSx counts
        pipe_ts1cnt_g3_array    [i_lane] = pipe_ts1cnt_g3   [i_lane*16+:16];
        pipe_ts2cnt_g3_array    [i_lane] = pipe_ts2cnt_g3   [i_lane*16+:16];
        pipe_tsxcnt_g12_array   [i_lane] = pipe_tsxcnt_g12  [i_lane*16+:16];
        core_ts1cnt_g3_array    [i_lane] = core_ts1cnt_g3   [i_lane*16+:16];
        core_ts2cnt_g3_array    [i_lane] = core_ts2cnt_g3   [i_lane*16+:16];
        core_tsxcnt_g12_array   [i_lane] = core_tsxcnt_g12  [i_lane*16+:16];
    end
end

// load equalization parameters for lane selected
always @(posedge sys_clk) begin
                                                          566,12        37%
```

FPGA

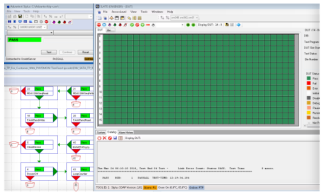


```
$ lspci -vvvns 2:0
02:00.0 0580: 1850:3657
        Subsystem: 1850:3658
        Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx+
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0, Cache Line Size: 64 bytes
        Region 0: Memory at f7c00000 (64-bit, non-prefetchable) [size=1M]
        Region 2: Memory at d0000000 (64-bit, prefetchable) [size=256M]
        Capabilities: [80] Power Management version 3
                Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
                Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-
        Capabilities: [b0] MSI-X: Enable+ Count=65 Masked-
                Vector table: BAR=0 offset=00020000
                PBA: BAR=0 offset=00021000
        Capabilities: [c0] Express (v2) Endpoint, MSI 00
                DevCap: MaxPayload 128 bytes, PhantFunc 0, Latency L0s <64ns, L1 <1us
                        ExtTag- AttnBtn- AttnInd- PwrInd- RBE+ FLReset-
                DevCtl: Report errors: Correctable- Non-Fatal- Fatal- Unsupported-
                        RlxdOrd- ExtTag- PhantFunc- AuxPwr- NoSnoop+
                        MaxPayload 128 bytes, MaxReadReq 512 bytes
                DevSta: CorrErr+ UncorrErr- FatalErr- UnsuppReq+ AuxPwr- TransPend-
                LnkCap: Port #0, Speed 5GT/s, Width x8, ASPM not supported, Exit Latency L0s unlimited, L1 unlimited
                        ClockPM- Surprise- LLActRep- BwNot-
                LnkCtl: ASPM Disabled; RCB 64 bytes Disabled- CommClk+
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
                LnkSta: Speed 5GT/s, Width x8, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
                DevCap2: Completion Timeout: Range B, TimeoutDis+, LTR-, OBFF Not Supported
                DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-, LTR-, OBFF Disabled
                LnkCtl2: Target Link Speed: 5GT/s, EnterCompliance- SpeedDis-
                         Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
                         Compliance De-emphasis: -6dB
                LnkSta2: Current De-emphasis Level: -6dB, EqualizationComplete-, EqualizationPhase1-
                         EqualizationPhase2-, EqualizationPhase3-, LinkEqualizationRequest-
        Capabilities: [100 v2] Advanced Error Reporting
                UESta:  DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq- ACSViol-
```

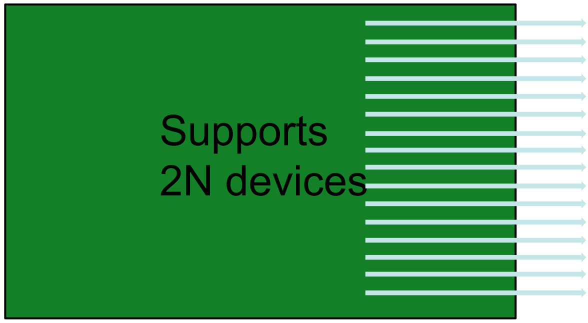
Embedded Linux



Example of HD firmware/software board



UL Software



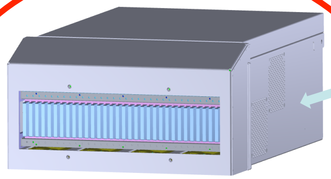
Example of HD power supply board

Still could have power control for each device individually

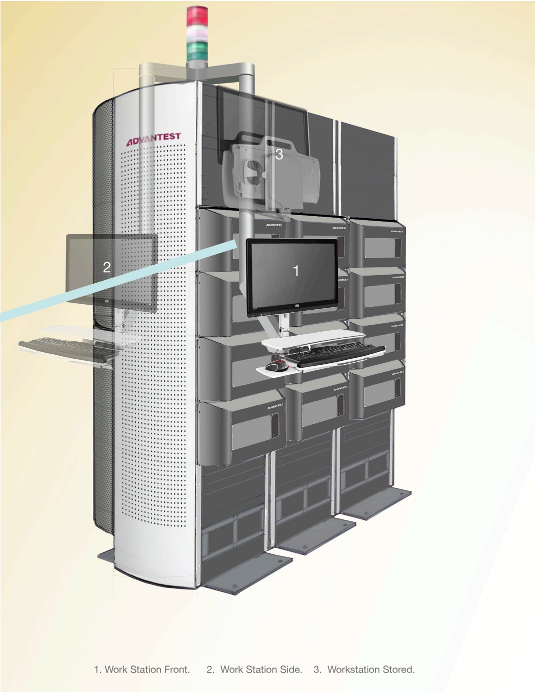
Flash Memory Summit 2016
Santa Clara, CA

Primitive for HVM Design

Would use one high-density primitive per shelf



One shelf of the rack



Example of an HVM type of tester.
Three racks with four shelves each.

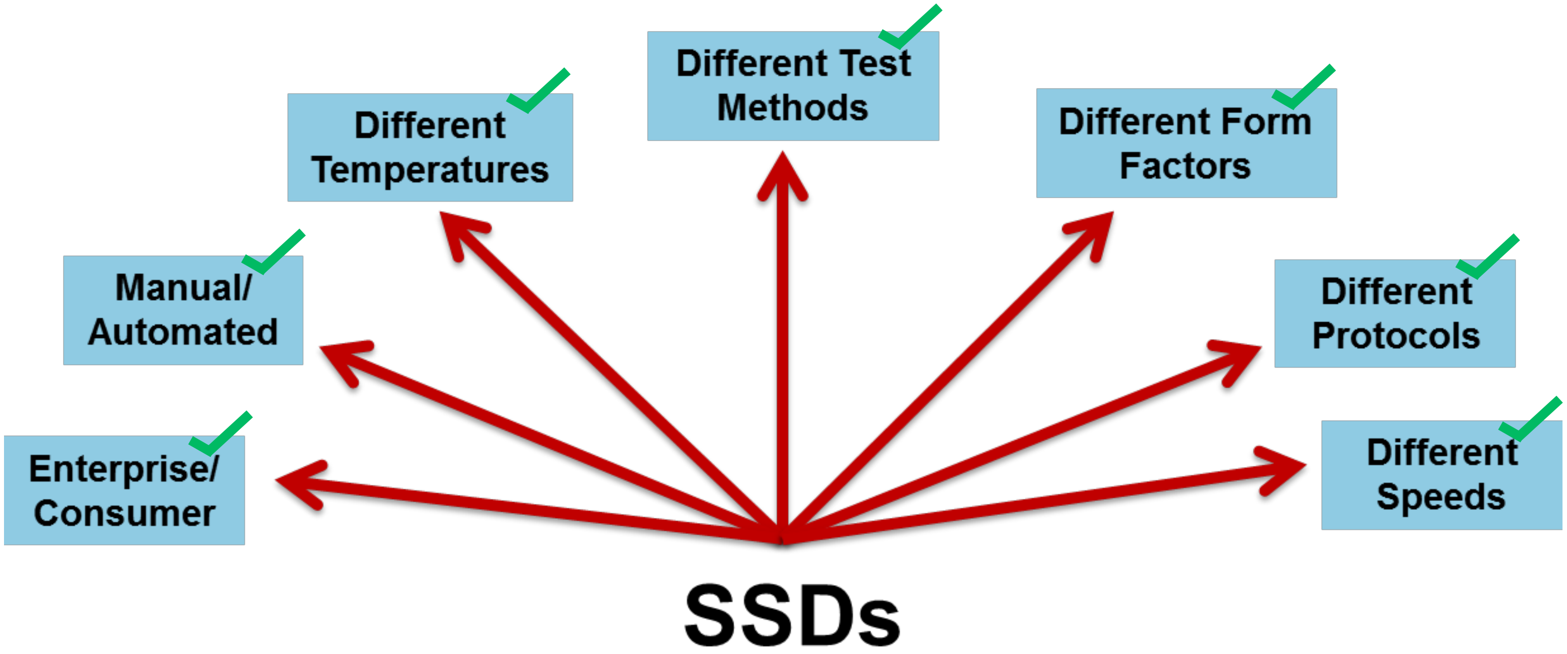
The Primitive with Automation

- HVM tester with primitive
  - Can add automation to drive insertion and removal
  - No change to primitive required

The Primitive with BIST

- Simple firmware and power supply boards for primitive
  - Not as much control or performance needed
- Can be combined with burn-in testing
- Concept of Primitive scalability remains

Revisit Test Requirements



Different Temperatures ✓

Different Test Methods ✓

Different Form Factors ✓

Manual/ Automated ✓

Different Protocols ✓

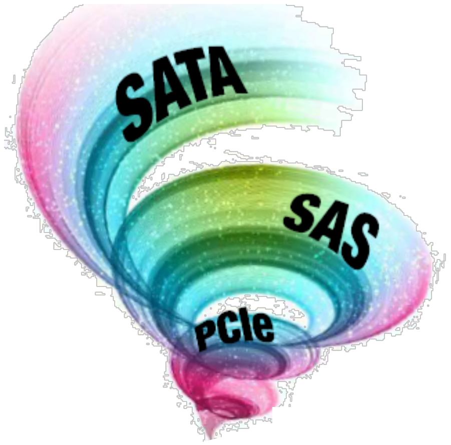
Enterprise/ Consumer ✓

Different Speeds ✓

SSDs

Concept of the Primitive as Applied to Life Cycle Testing

- Same primitive concept used across many tester types
  - Reuse software, firmware, embedded OS across:
    - Test phases
    - Tester types
    - Form factors
    - Protocols



Implications of this Solution

- Meet cost targets of each individual phase of life cycle testing
- Make the heart of tester portable and scalable to address challenge of testing the life cycle of an SSD