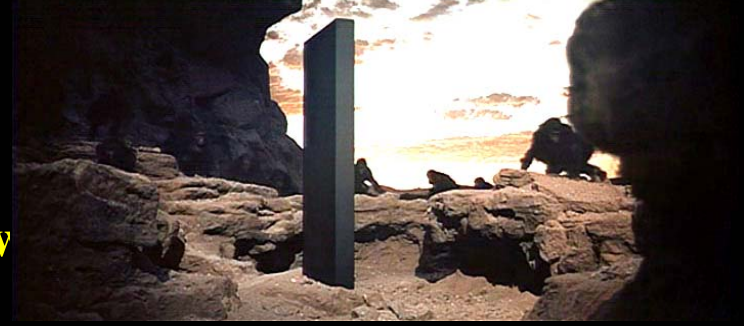
# 2001(+5): A Space/Time Odyssey

## Timing and Electrical Measurement of DDR2 Memory with IBIS 4.1 and AMS

**Gary L. Pratt, P.E.**
**Product Manager**
gary_pratt@mentor.com

Mentor Graphics®

# In the beginning …



- There was SPICE … and it was sssslllooowv
  - but nobody cared, because hey … time didn't exist yet
- Then, there was time …
  - But, never quite enough
- Then, there was IBIS
  - and, it was good, and simple to use, and saved much precious time
- Then, came new technology
  - which brought a plague of new keywords across the land
- Then, IBIS grew large and unwieldy
  - and slow to respond to new technologies
- Then, the IBIS wise-men … errrr …wise-people said
  - let there be AMS
- And, then there was AMS
  - and it was very good
- Then, the masses rejoiced & created multitudes of innovation
  - some of which not even the wise-people had foretold …

# Behold …

## AMS

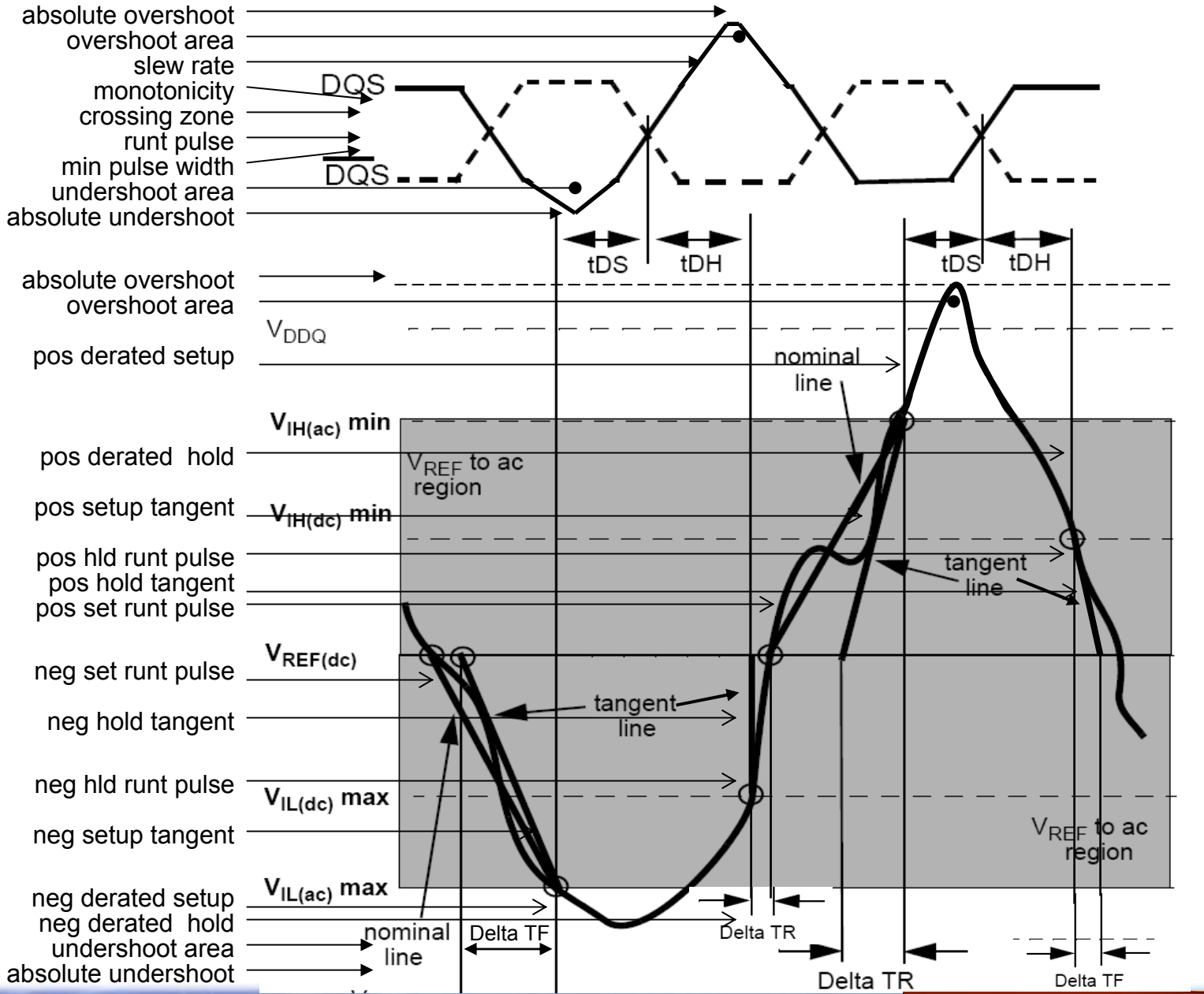
for

## DDR2 Electrical

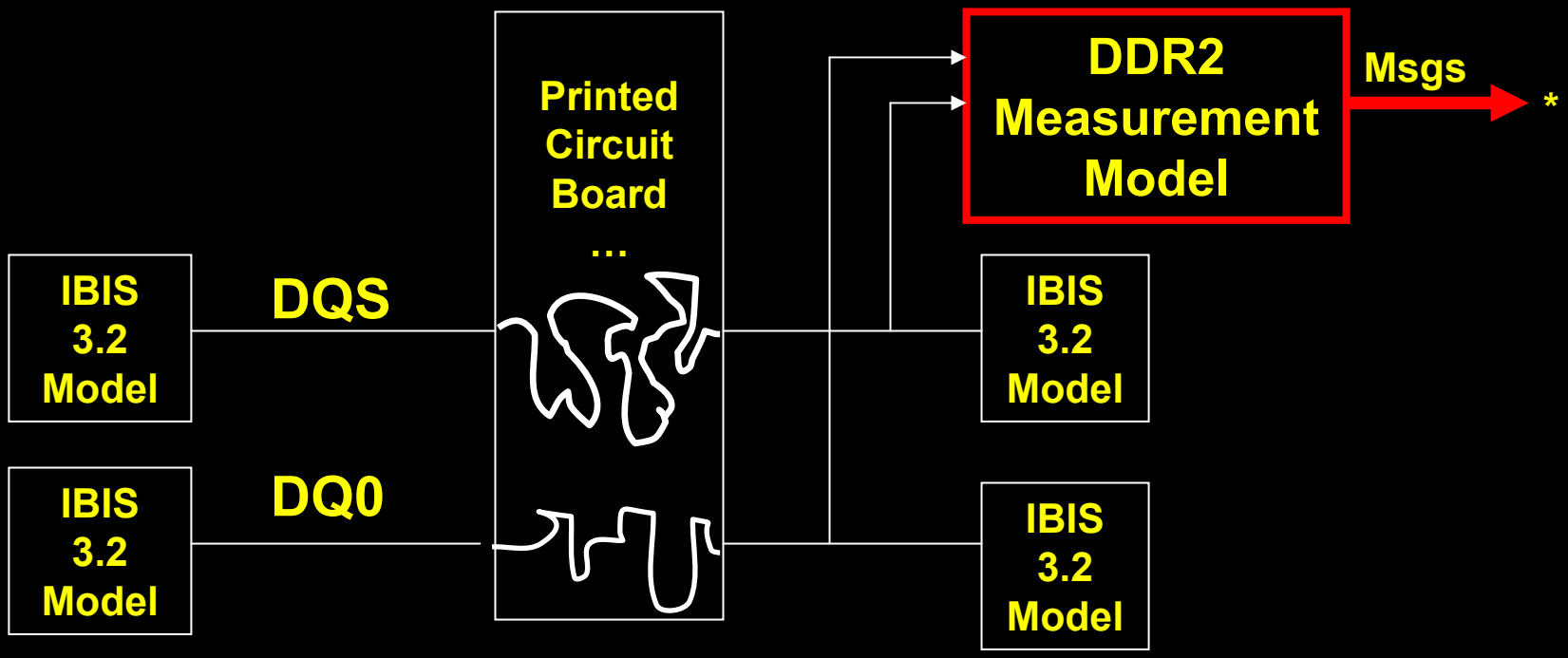
and

## Timing Measurement

# DDR2 Electrical and Timing Measurement Requirements

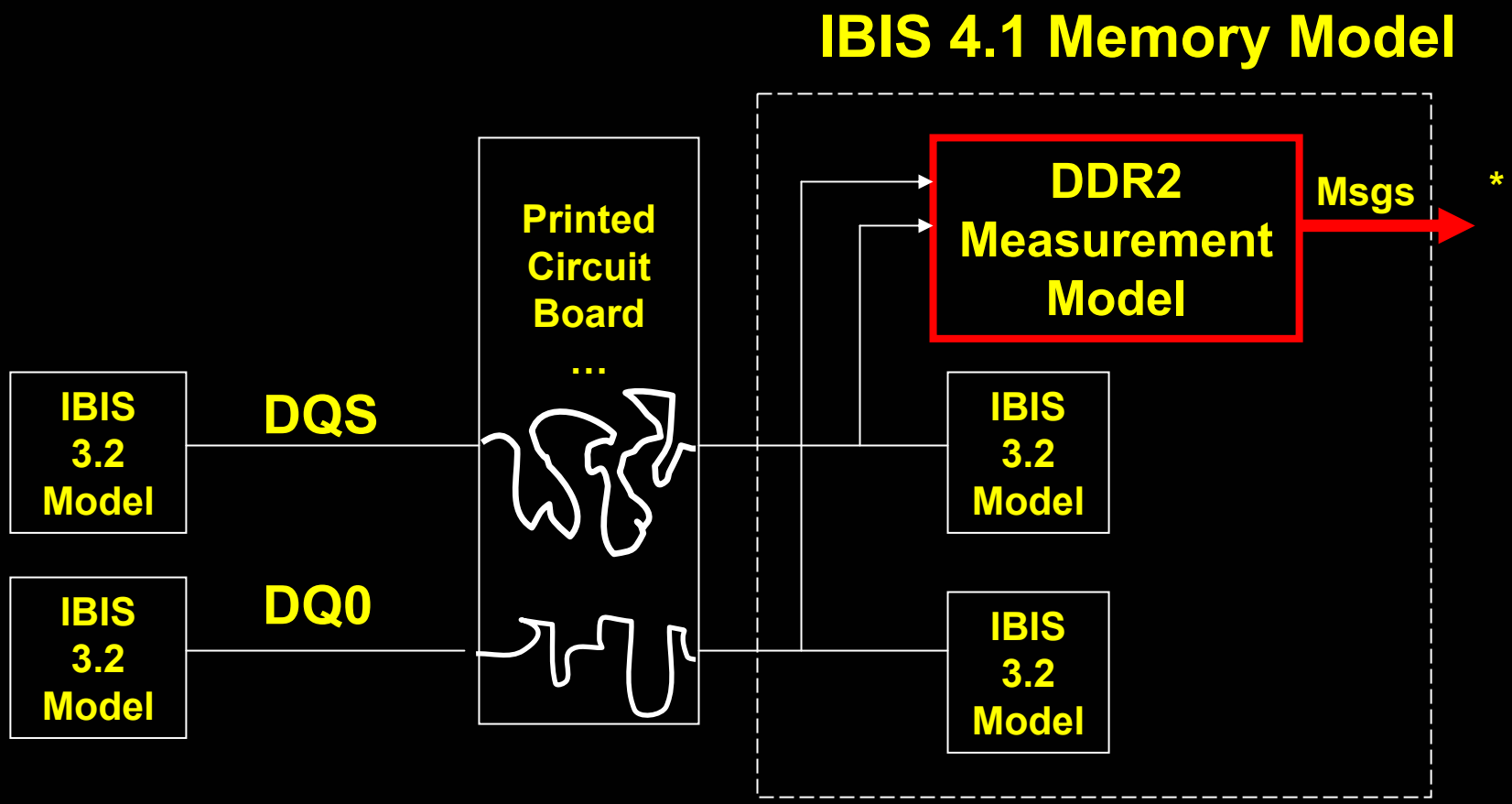


# IBIS 4.1 Measurement Model



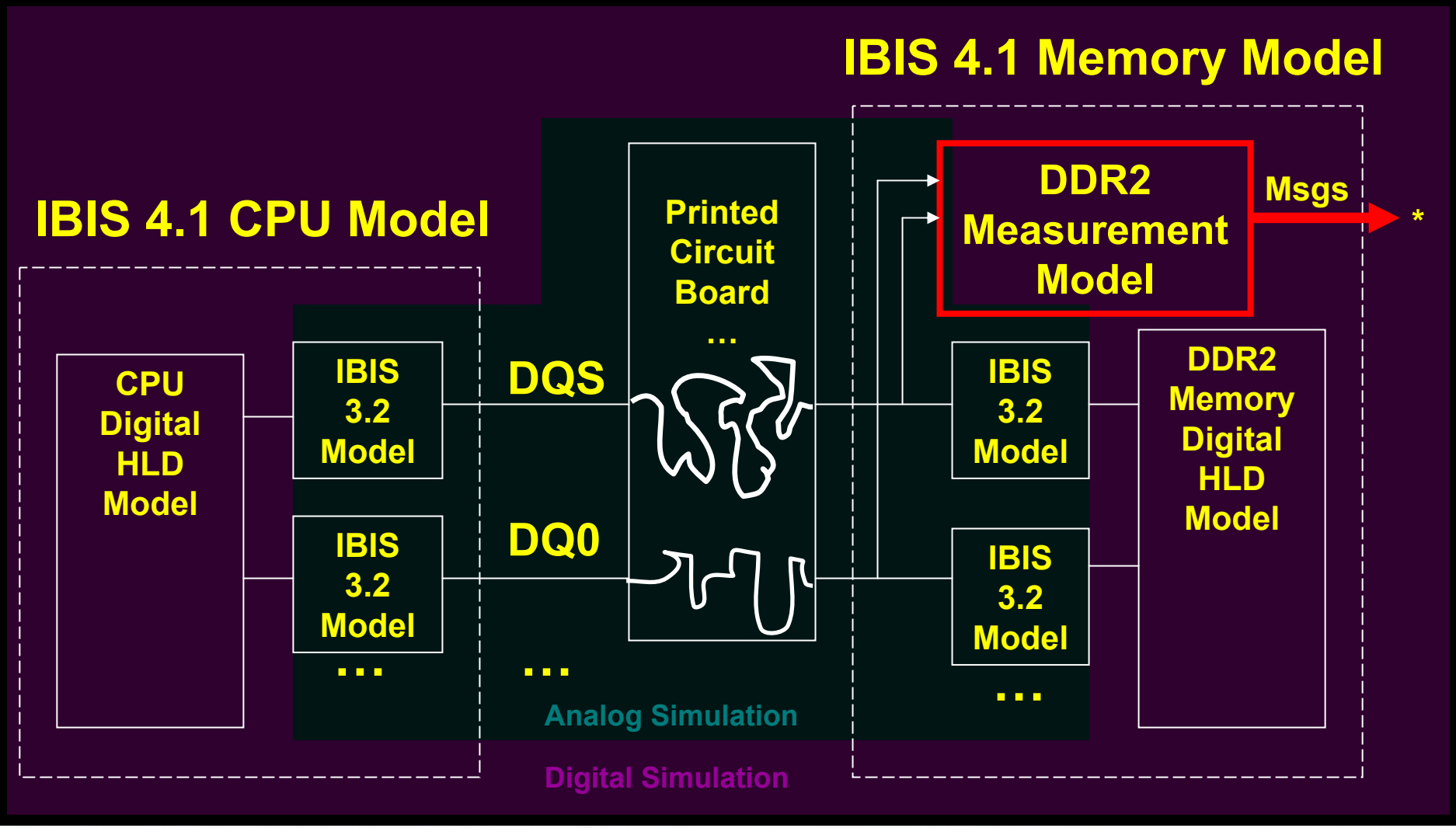

*But, where do violation messages go?

# IBIS 4.1 Measurement Model




*But, where do violation messages go?

# Measurement Model in IBIS 4.1 Functional SI




*But, where do violation messages go?

# IBIS 4.1 Memory Model

```
[Pin]   signal_name   model_name   R_pin      L_pin         C_pin
|
1       DQ0           NC           0.043      5.22nH        1.26pF |dqbuff
2       CK            NC           0.030      2.78nH        0.91pF |clk_input
3       CK#           NC           0.029      2.73nH        0.86pF |clk_input

[Diff_pin]      inv_pin      vdiff     tdelay_typ     tdelay_min    tdelay_max
    2             3          .360V       0ns              NA            NA
|
|************************EXTERNAL MODEL*************************************
[node declarations]                    |Instantiate the DQ input buffer
DQ0 CLK CLK_N GND VDD                  [circuit call]     dq_full_input
[end node declarations]                Port_map A_signal   1
                                       Port_map D_receive DQ0
|Instantiate the clock                 ...
|IBIS input buffer          *         [end circuit call]
[circuit call]      clk_input          |
Port_map A_signal    2                 |Instantiate the DDR2 Measurement Model
Port_map D_receive   CLK               [circuit call]          DQS_DQ_CHECK
...                                    Port_map DQ             1
[end circuit call]                     Port_map DQS_p          2
|                                      Port_map DQS_n          3
[circuit call]      clk_input          [end circuit call]
Port_map A_signal    3                 |
Port_map D_receive   CLK_N             |declare the DDR2 Measurement Model
...                                    [external circuit] DQS_DQ_CHECK
[end circuit call]                     language vhdl-ams
                                       corner typ DQS_DQ.vhd DQS_DQ(behav)
                                       ports DQ DQS_p DQS_n
                                       [end external circuit]
|
|************************END EXTERNAL MODEL*********************************
|
[Model]          dq_full_input
Model_type       Input
|
Vinl = 940.000mV
Vinh = 1.560V
```

*How to deal with single-clock, multiple data?

# Memory Data Sheet

```
PACKAGE Data_Sheet_DDR2_400 IS *
   constant VDDQ    : real := 1.8;
   constant VIXACMIN  : real := 0.50 * VDDQ - 0.175;
   constant VIXACMAX  : real := 0.50 * VDDQ + 0.175;
   constant VREFDC    : real := 0.9;
   constant VIHACMIN  : real := VREFDC + 0.250;
   constant VIHDCMIN  : real := VREFDC + 0.125;
   constant VILDCMAX  : real := VREFDC - 0.125;
   constant VILACMAX  : real := VREFDC - 0.250;
   constant VINDCMAX  : real := VDDQ + 0.300;
   constant VINDCMIN  : real := -0.300;
   constant VIDACMIN  : real := 0.5;
   constant TCK    : time := 2.0 ns;
   constant tDQSH : time := 0.35*TCK;
   constant tDQSL : time := 0.35*TCK;
   constant tDS_base : time := 150*ps;
   constant tDH_base : time := 275*ps;
   constant MAXOVERSHOOT : real := 0.5;
   constant MAXUNDERSHOOT : real := 0.5;
   constant MAXOVERSHOOTAREA : real := 1.33e-9; --vsec
   constant MAXUNDERSHOOTAREA : real := 1.33e-9; --vsec
   -- units of values of horizontal axis of table
   constant dqs_slew_units : real := 1.0e9; -- volts/ns
   -- values of Horizontal axis of table (must be descending!)
   constant dqs_slew_values : dqs_slew_value_type :=
   (4.0,3.0,2.0,1.8,1.6,1.4,1.2,1.0,0.8);
   -- units of values of vertical axis of table
   constant dq_slew_units : real := 1.0e9; -- volts/ns
   -- values of vertical axis of table (must be descending!)
   constant dq_slew_values : dqs_slew_value_type :=
   (2.0,1.5,1.0,0.9,0.8,0.7,0.6,0.5,0.4);
   -- units of table entries
   constant derating_table_units : time := 1.0*ps;
   -- table entries
   constant derating_table_data : derating_table_type :=
   --dqs=4.0,   3.0,       2.0,       1.8,       1.6,       1.4,       1.2,       1.0,       0.8        --dq
   (0125,0045, 0125,0045, 0125,0045, dash,dash, dash,dash, dash,dash, dash,dash, dash,dash, dash,dash, --2.0
    0083,0021, 0083,0021, 0083,0021, 0095,0033, dash,dash, dash,dash, dash,dash, dash,dash, dash,dash, --1.5
    0000,0000, 0000,0000, 0000,0000, 0012,0012, 0024,0024, dash,dash, dash,dash, dash,dash, dash,dash, --1.0
    dash,dash, -011,-014, -011,-014, 0001,-002, 0013,0010, 0025,0022, dash,dash, dash,dash, dash,dash, --0.9
    dash,dash, dash,dash, -025,-031, -013,-019, -001,-007, 0011,0005, 0023,0017, dash,dash, dash,dash, --0.8
    dash,dash, dash,dash, dash,dash, -031,-042, -019,-030, -007,-018, 0005,-006, 0017,0006, dash,dash, --0.7
    dash,dash, dash,dash, dash,dash, dash,dash, -043,-059, -031,-047, -019,-035, -007,-023, 0005,-011, --0.6
    dash,dash, dash,dash, dash,dash, dash,dash, dash,dash, -074,-089, -062,-077, -050,-065, -038,-053, --0.5
    dash,dash, dash,dash, dash,dash, dash,dash, dash,dash, dash,dash, -127,-140, -115,-128, -103,-116);--0.4
   end package;
```

*How to deal with multiple speed grades?

# TANGENT MEASUREMENT

- Wait for vref crossing
- Store data points
- Wait for vix_ac cross
- Calculate the slope from each point to the vix_ac crossing point
- Return the maximum slope
- Wait for vix_dc crossing
- Calculate the slope from each subsequent point back to the vix_dc crossing
- Wait for vref crossing
- Return the max slope

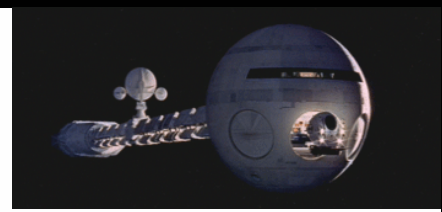
```
begin
   ------------------------------------------------------------------------
   -- measure the setup time tangent
   ------------------------------------------------------------------------
   wait until VREFDC; -- wait for a crossing of correct direction
   max_slope:=0.0; data_point_cntr:=0;  setup_crossing <= 0.0*sec;
   while not vix_ac'event loop -- store all the data points until vix_ac crossing
      data_point_v(data_point_cntr) := Vin'reference;
      data_point_t(data_point_cntr) := now;
      wait on vix_ac, ASP;*-- wait for next event
      data_point_cntr := data_point_cntr + 1;
   end loop;  -- go on to find the maximum slope
   setup_crossing <= now;
   for i in min_slope to data_point_cntr-1 loop
      slope := (crossing_point_v - data_point_v(i)) /
               (crossing_point_t - data_point_t(i));
      if slope > max_slope then max_slope:=slope; end if;
   end loop;
   setup_slope <= max_slope;
   ------------------------------------------------------------------------
   -- measure the hold tangent
   ------------------------------------------------------------------------
   wait until not vix_dc;  -- wait for opposite crossing of vix_dc
   max_slope := 0.0;
   crossing_point_v := Vin'reference; crossing_point_t:=now;
   -- calculate slope of each point until vix_dc, or max_points
   while not VREFDC'event loop
      wait on VREFDC, ASP ;
      slope := -(Vin'reference - crossing_point_v) /
                (now - crossing_point_t);
      if slope > max_slope then max_slope := slope; end
   end loop;
   hold_slope <= max_slope; -- in v/s
end process;
```

*But, “ASP” isn’t an IEEE-1076.1 standard.

(error and exception handling removed for clarity)

# Absolute and Area Over/Undershoot

- Create events on crossings
- Wait for a crossing into the overshoot or undershoot area
- While in the over/undershoot area, integrate the area under the curve
- At any time, if the maximum area is exceeded, generate an violation message
- Notice: “REF” is included in each message *
- Error if absolute limit is exceeded



```
begin
   vdd_cross <= vin'above(VDDQ);
   vss_cross <= vin'above(0.0);
   max_cross <= vin'above(VDDQ+MAXOVERSHOOT);
   min_cross <= vin'above(0.0-MAXUNDERSHOOT);
   process
      variable area : real := 0.0;
      variable last_asp : real;
   begin
      wait until vdd_cross or not vss_cross;
      area := 0.0; last_asp := now;
      while vdd_cross or not vss_cross loop
         wait on ASP;
         if vin < 0.0 then area := area + (0.0-vin)*(now-last_ASP); end if;
         if vin > VDDQ then area := area + (vin-VDDQ)*(now-last_ASP); end if;
         if area > MAXOVERSHOOTAREA or area < -MAXUNDERSHOOTAREA then
            REPORT REF&"Violation - Exceeded Overshoot or Undershoot Area";
            exit;
         end if;
         last_asp := now;
      end loop;
   end process;
   process (max_cross, min_cross) begin
      if max_cross or not min_cross then
        REPORT REF&"Violation - MAXOVERSHOOT or MAXUNDERSHOOT Level Exceeded";
      end if;
   end process;
end architecture;
```

*But, how to get “REF” into the model?

# Setup Violation Check

```
-- Check the setup timing
check_setup: process (dqs_slope'transaction)
   variable derating : time := 0*sec;
   variable expected, actual : line;
begin
   derating := interp_derating_table(dqs_slope, setup_slope, SETUP, REF);
   if setup_time = 0.0*sec then
      REPORT REF&"Setup Violation -- DQ transistion in progress during DQS event";
   elsif derating /= time'left then
      if (now-dqs'last_event) - setup_time < tDS_base+derating then
         write(expected,tDS_base+derating,left); write(actual,(now-dqs'last_event)-setup_time,left);
         REPORT REF&"Setup Violation, Expected: "&expected.all&". Actual: "&actual.all;
      end if;
   end if;
end process;
```

Trigger the process after each DQS slope is determined (VIXACMAX-0.50*VDDQ)

Read and interpolate the derating chart according to DQ and DQS slew rate

If DQ is in transition (setup_time=0), report a violation

If derating is valid, check the setup time

If actual setup time is less than derated setup specification, report violation

(error and exception handling removed for clarity)

# Timing Analysis with IBIS 3.2 Models

## (adjusting pin-to-pin specs to die-to-die timing)

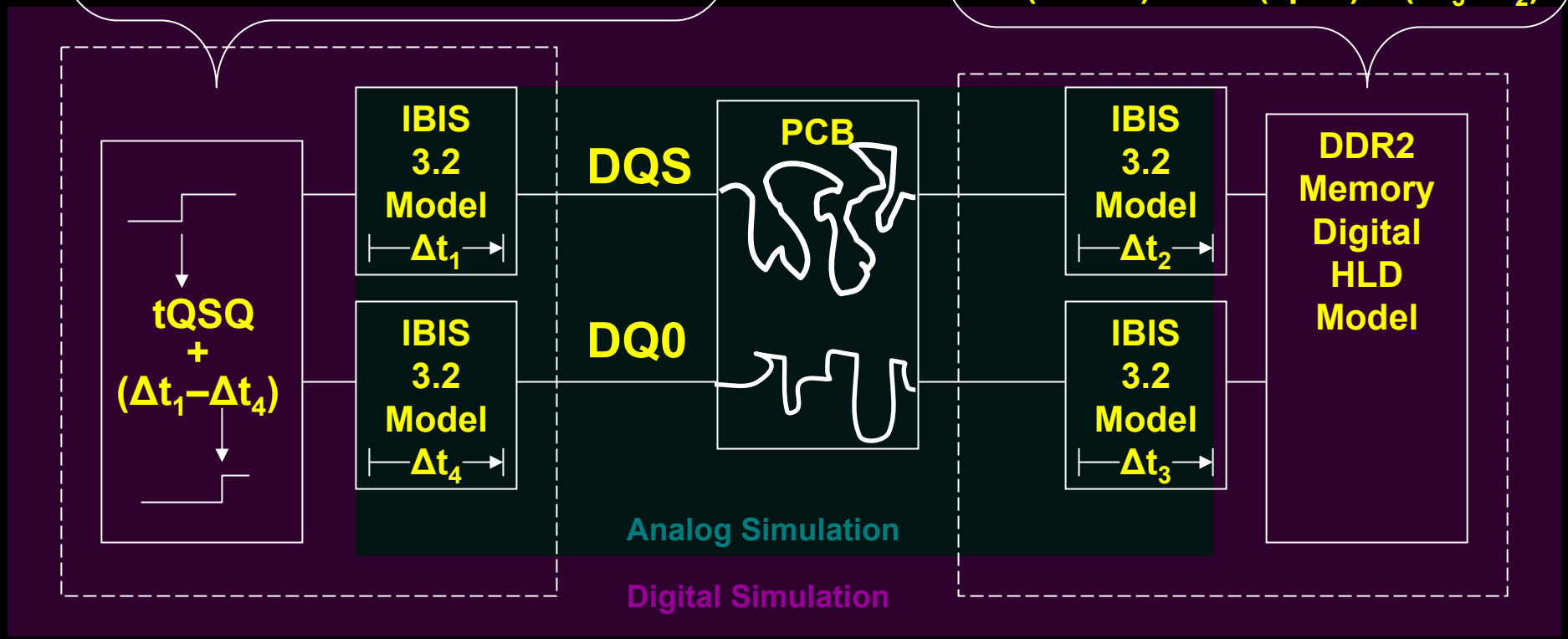
**Adjustments to Digital Output Timing:**

$$tQSQ(actual) = tQSQ(spec) + (\Delta t_1 - \Delta t_4)$$

**Adjustments to Digital Input Timing:**

$$tDS(actual) = tDS(spec) + (\Delta t_2 - \Delta t_3)$$

$$tDH(actual) = tDH(spec) + (\Delta t_3 - \Delta t_2)$$




Rising and falling Δt Measured with databook load for each process corner.

*Can Δt be made standard between models and tool vendors?

# Timing Analysis with IBIS 3.2 Models

## (compensating for delay in IBIS 3.2-models)

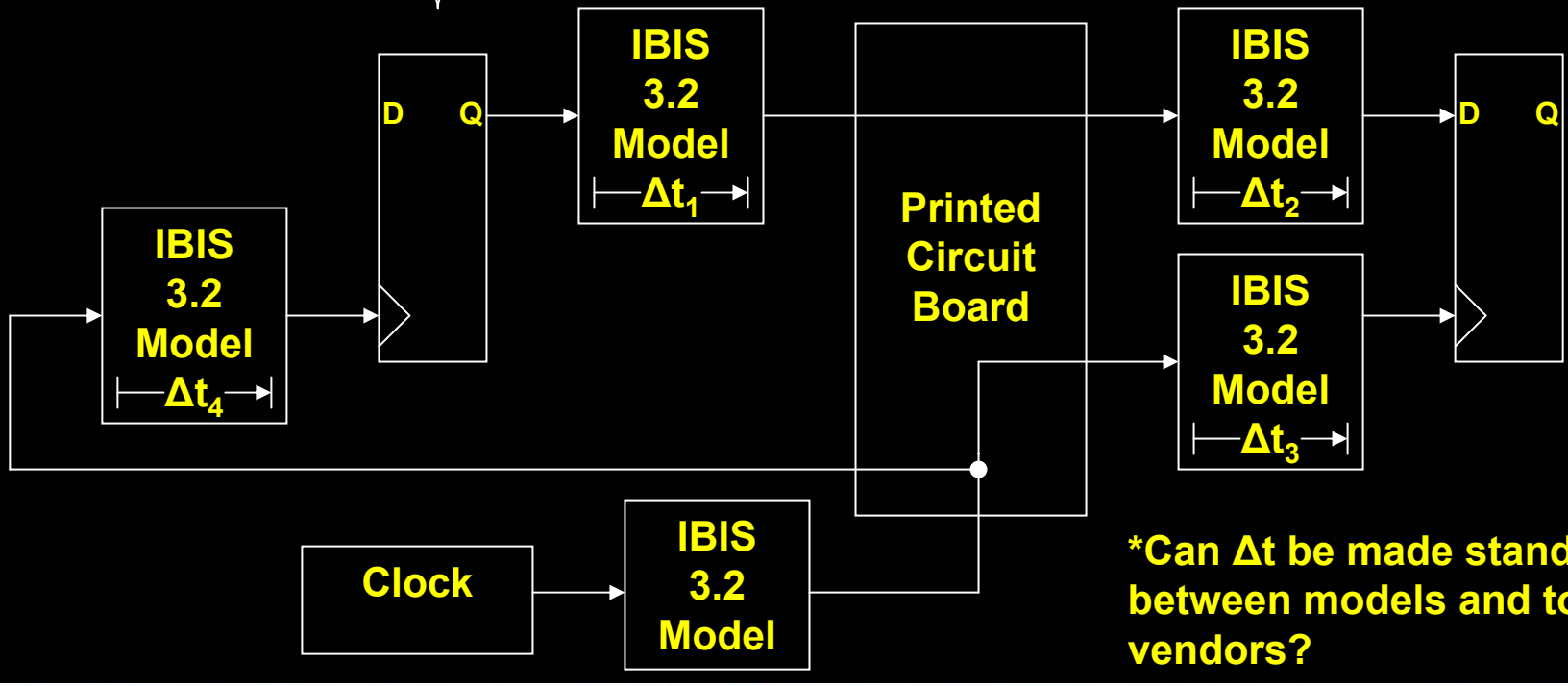
**Adjustments to Propagation Delay:**

$tPCQ\ (actual) = tPCQ(spec) - (\Delta t_1 + \Delta t_4)$

**Adjustments to Setup and Hold:**

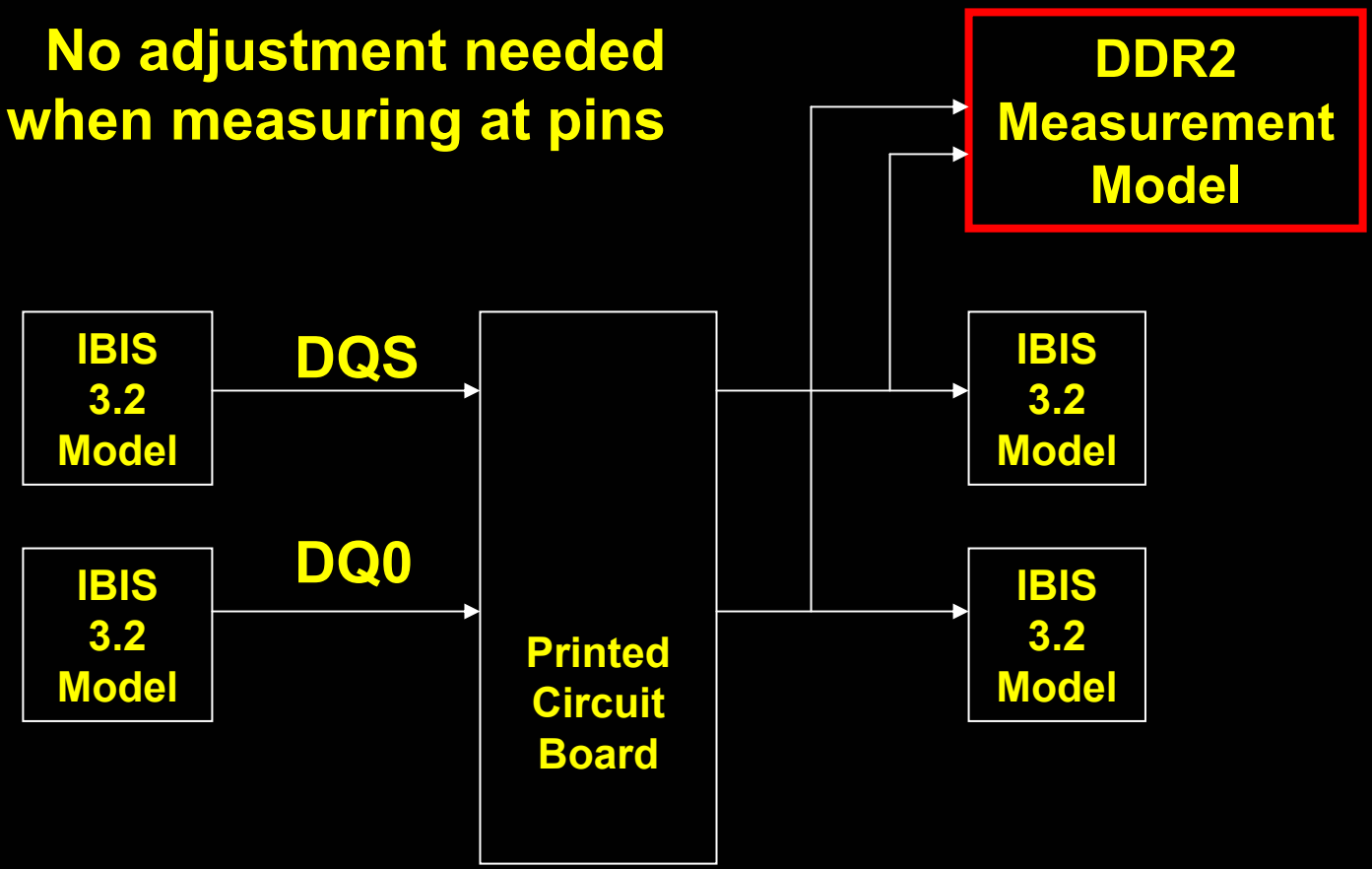
$tSU(actual) = tSU(spec) + (\Delta t_3 - \Delta t_2)$

$tH(actual) = tH(spec) + (\Delta t_2 - \Delta t_3)$



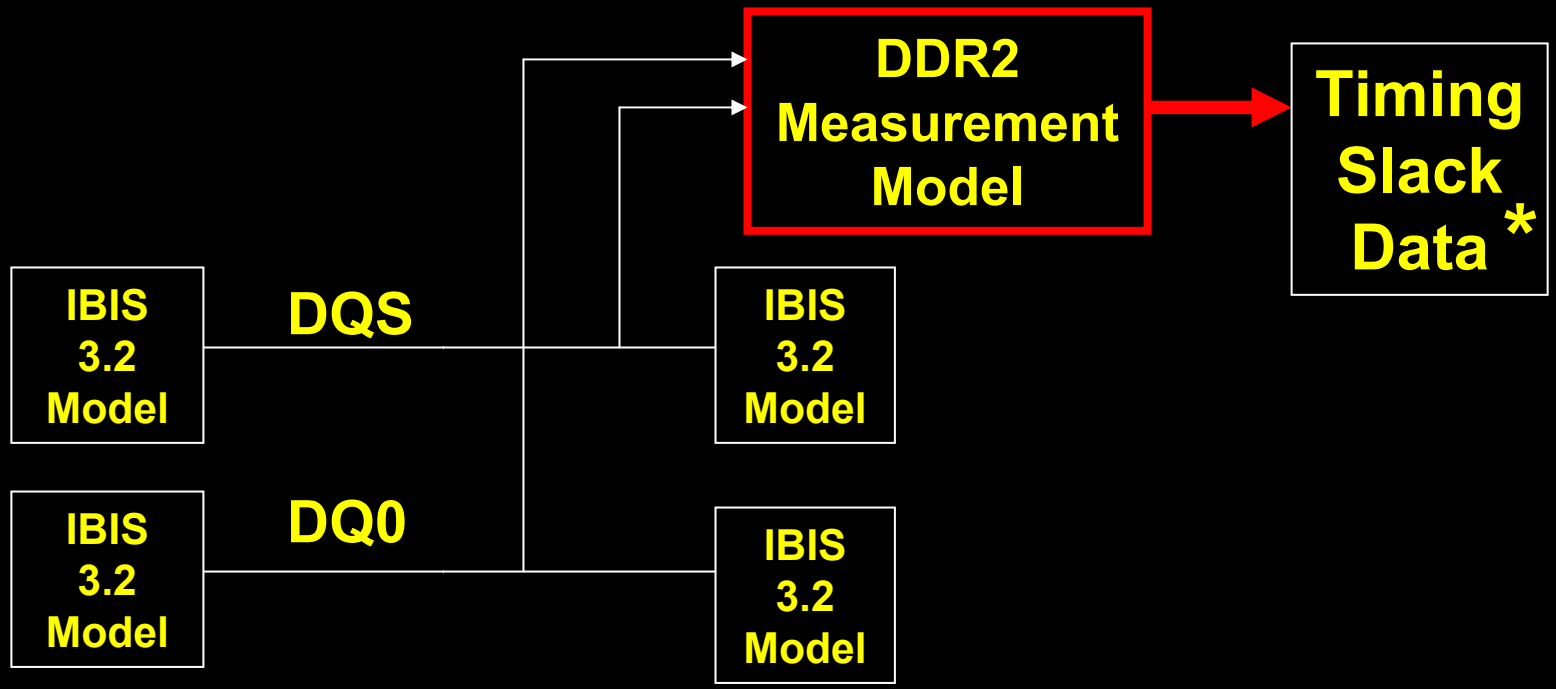

*Can Δt be made standard between models and tool vendors?

# Timing Analysis with IBIS 3.2 Models



# Trace Routing Budget

## IBIS 4.1 Memory Model




*But, how to get slack data into layout tool?

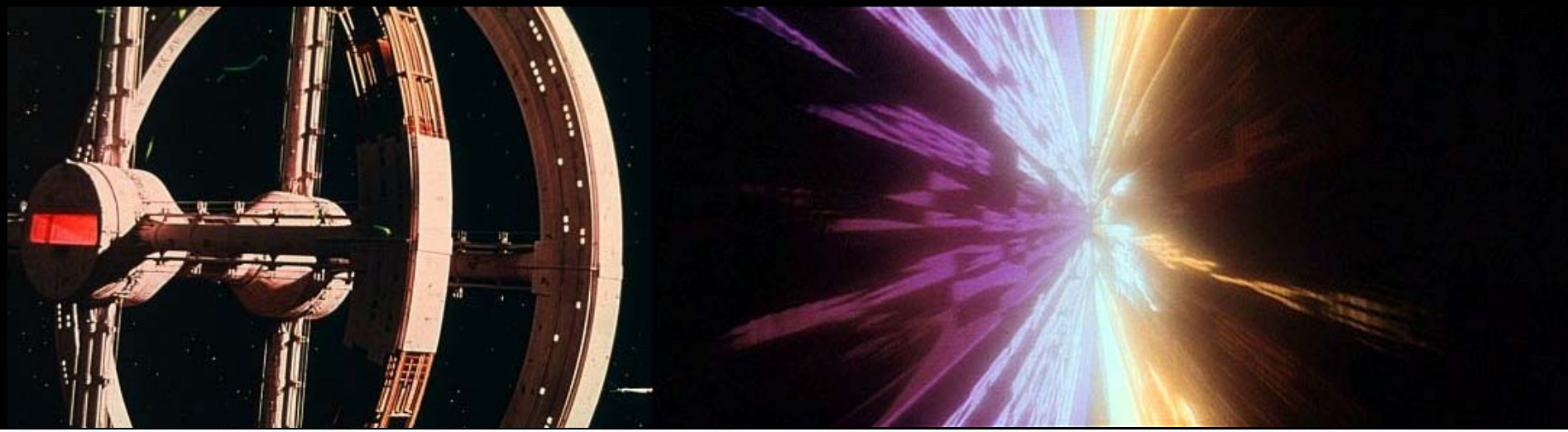
*What slew value to use?

# IBIS 4.1 and IEEE 1076.1

- Global -- Open Industry Standard
  - Portable between IBIS4.1-1076.1 simulators
- Complete Automation
  - No user intervention required
  - violation messages pinpoint problems
- Tests all connections in one simulation
  - All 180 cpu→mem and 128 mem→cpu nets in one step
- Modeling Engineer is in complete control
  - No need to involve the IBIS committee or EDA vendors to add new functions or features

# Future Applications

- Automated PCI Express Test Suite?
- ????



# Next Steps

- **EDA Vendors**
  - Support IBIS 4.1 AMS
- **Silicon Vendors**
  - Develop and Release IBIS 4.1 AMS models
- **PCB Engineers**
  - Demand the benefits of IBIS 4.1 AMS
    - Accept no substitutes
- **IBIS Committee**
  - Stay tuned for Ian's presentation

# Thanks to

- Randy Wolfe and his associates at Micron for advice on interpreting the JEDEC DDR2 specification and moral support in developing this measurement model.
- Todd Westerhoff for help using IBIS 3.2 models in timing analysis.