Test Method for Instantaneous IDD Profiling

Leon Yeow Ting, Stephen Fu, Tan Kheng How and Kar Leong Foong

Agenda

- Introduction
- DC Profiling on the V93000
- Dynamic DC measurements on the V93000
- Test Method for Instantaneous IDD Profiling
- Smoothing Method
- Test Method Parameters
- Test Vector Requirement
- Sample Results of Current Measurement
- External Scope – “Current” Capture
- Test Method vs External Scope
- DC Profiling vs Event Insertion
- New CSDPS Hardware Support
- Limitations and Enhancement Requests
- Summary

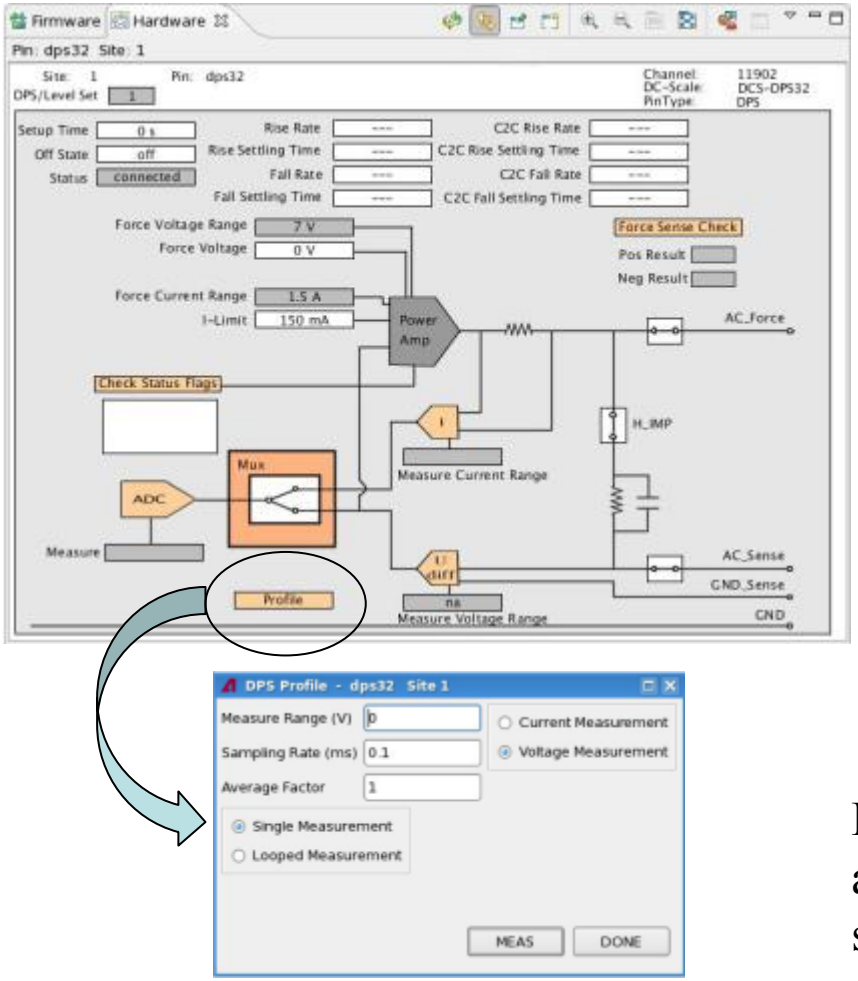
Introduction

- In the history of SOC testing, instantaneous current is examined on bench (long turn-around time). Instantaneous current profiling on ATE using external scope is difficult due to nature of current being measured in serial.

- No established efficient method to examine instantaneous current on ATE. Existing TM measures only the operating current and leakage rather than time domain current profile.

- The capability to be able to measure instantaneous current profile on ATE is critical for both silicon or vector debug, or any power related studies.

DC Profiling on the V93000



Profiling (DC profiling) allows you to measure current and voltage waveforms during pattern execution. Utilizes the DC Profiling APIs. Specifically:

DC_PROFILING_ON (TDC topic 125003)
DC_PROFILING_OFF (TDC topic 125004)

```
// Defines and activates a current profiling setup for pin 1.
DC_PROFILING_ON("pin1", TM::CURRENT,100 uA, 100 ms, 16, TM::RESTORE_RANGE);
....
Execute test.
....
DC_PROFILING_OFF("pin1");  // Disables the defined current profiling.
DC_RESULT_ACCESSOR ra;
ra.uploadResult("pin1", TM::RESULT_INDEX);
ARRAY_D results = ra.getValues("pin1"); // Obtains the measured values.
```

DC_PROFILING_ON API defines and activates a DC profiling profile. It is equivalent to the call sequence:

DC_PROFILING_DEFINE (TDC topic 125001)
DC_PROFILING_ENABLE (TDC topic 125002)

DC Profiling on the V93000

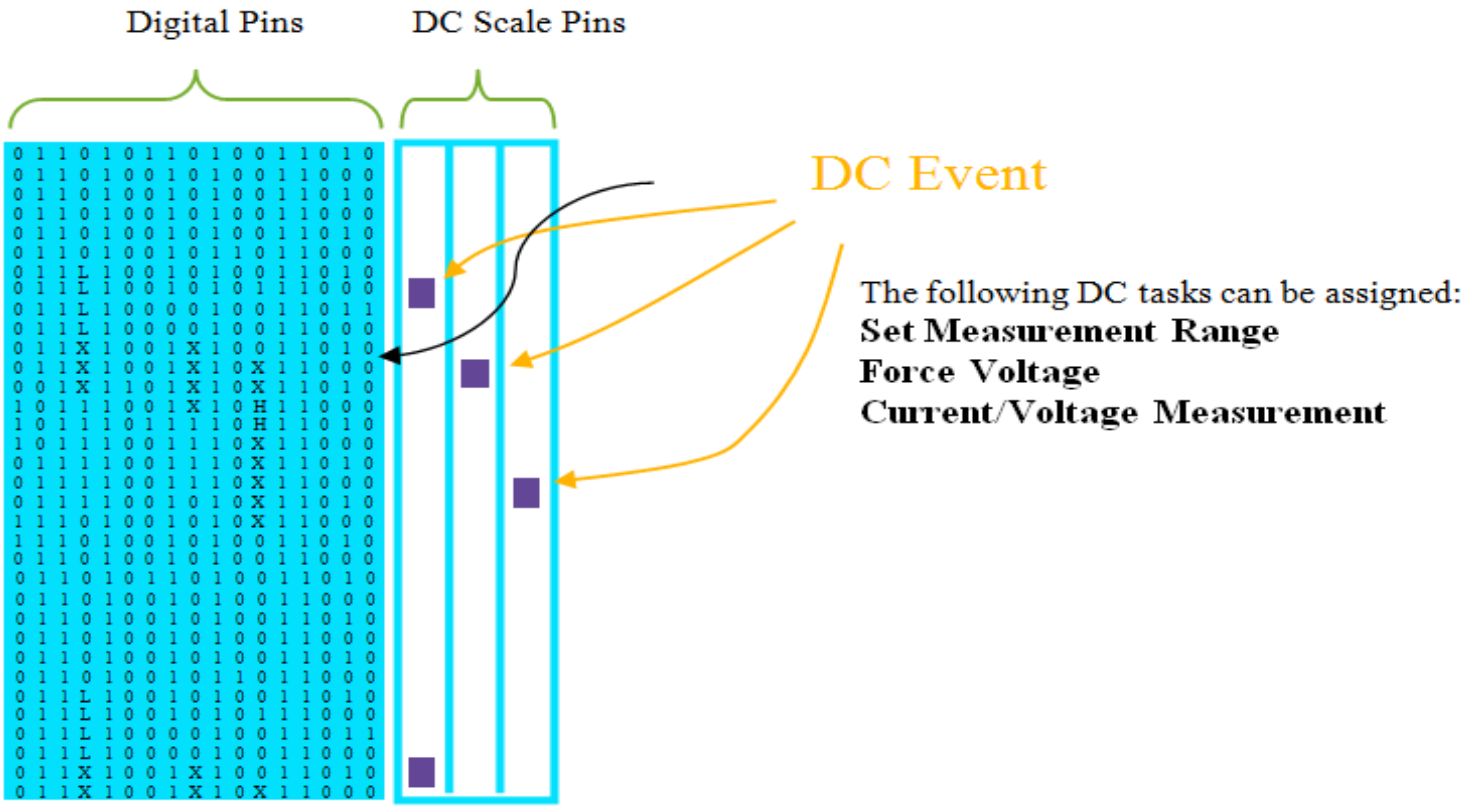
- New TM APIs are introduced to define a DC profiling setup in SmarTest 7.1.1.
  - DC_PROFILING_DEFINE
  - DC_PROFILING_ENABLE
  - DC_PROFILING_ON
  - DC_PROFILING_OFF

- These APIs defines and turned on/off the DC profiling based on user-defined pin lists / pin groups, measurement type, expected measurement range, sampling interval, oversampling--number of measurements per sample, and keep or restore measurement range.

- The minimum sampling interval is 50µs.

- For oversampling, The number of measurements per sample is internally corrected to a power of 2 number (2^n). if a sampling interval of 50µs is defined, the max. number of measurements per sample is 4 to avoid skipped samples.

Dynamic DC measurements on the V93000



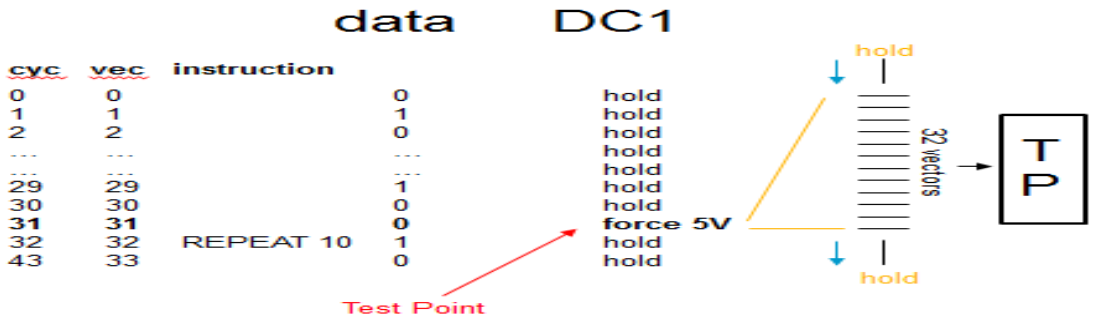
Dynamic DC tests (TDC topics 113758)

DC actions can be executed fully synchronized with digital data, No continuous switching from pattern execution to DC measurements implemented in C++ and back. The instruments are fully controlled by a digital pattern execution (through the digital sequencer). Measurements are embedded into the pattern.

Dynamic DC measurements on the V93000

- Utilizes EVENT_RANGE_IM, EVENT_IM, and PATTERN classes/APIs. Requires user to setup a sequencer controlled DC measurement events at multiple test points in the vector for current profiling.

- 32 vectors will be required to code a Test Data Point into a label.
  - ➢Limits for the placement of the first anchor point and
  - ➢Limits minimum distance between 2 adjacent Test Points (anchor points) in a label.



- No direct restriction on the minimum period that can be programmed for a DC Scale pin. User must ensure to have enough vectors between 2 Test Data Points to avoid violating the respective specs.
  - ➢Max sample rate of Digitizer mode is 500kHz

Test Method for Instantaneous IDD Profiling

Developed a TM utilizing the DPS32 to measure the current profile. Two approaches will be studied for current profiling:

DC Current Profiling using DC Profiling API (denoted as DC-Profiling)

Test Method	Dynamic_IDD_tml.dynamic_IDD_measure
Parameters	DPS1,DPS2, , @, DC-Profiling, 4, 50, 100, 0.001, 0, 19.2, None, 11, 0.01
dps_pins	DPS1,DPS2
Pattern	
Port	@
Measurement Mode	DC-Profiling
Num Meas Per Sample	4
Sampling Interval [us]	50
CurrentRange_Max [mA]	100
CurrentRange_Min [mA]	0.001
ToStopCycle	0
Frequency [MHz]	19.2
Smoothing Mode	None
Num Pts for Moving Avera	11
Smoothing Factor	0.01
Limits	...

Synchronized dynamic DC IDD measurement (denoted as Event-Insertion)

Test Method	Dynamic_IDD_tml.dynamic_IDD_measure
Parameters	DPS1,DPS2, Vector_A, new_define_dps, Event-Insertion, 4, 2.5, 100, 0.1, 10007
dps_pins	DPS1,DPS2
Pattern	Vector_A
Port	new_define_dps
Measurement Mode	Event-Insertion
Num Meas Per Sample	4
Sampling Interval [us]	2.5
CurrentRange_Max [mA]	100
CurrentRange_Min [mA]	0.1
ToStopCycle	1000710
Frequency [MHz]	40
Smoothing Mode	Discrete-Time-LPF
Num Pts for Moving Avera	11
Smoothing Factor	0.05
Limits	...

Accuracy of the measurements are studied and smoothing method is also explored. ATE test method results are also cross-checked with external scope measurement.

Test Method for Instantaneous IDD Profiling

DC-Profiling

```
virtual void run()
{

  //Add your test code here.
    DC_RESULT_ACCESSOR result;
    PATTERN_CONTROLLER patCtrl;
    vector <string> vPins_dps;
    int iNum_dps_pins, i, InsertionInterval;
    //Convert string to vector <string>
    tokenize (sPins_dps, vPins_dps, ",");
    iNum_dps_pins = vPins_dps.size();

    ON_FIRST_INVOCATION_BEGIN();
    CONNECT();
    if(sMeaMode == "DC-Profiling")
    {
      for (i=0; i<iNum_dps_pins; i++)
      {
        DC_PROFILING_ON(vPins_dps[i], TM::CURRENT, mCurrentRange_Max mA, mSampInterval us, mNumMeas, TM::RESTORE_RANGE);
      }

    FUNCTIONAL_TEST();

      for (i=0; i<iNum_dps_pins; i++)
      {
        DC_PROFILING_OFF(vPins_dps[i]);
        result.uploadResult(vPins_dps[i], TM::RESULT_INDEX);
        ARRAY_D currentProfile = result.getValues(vPins_dps[i]);
        cout << "DC-Profiling: Current on pin " << vPins_dps[i] << " = "<< currentProfile << endl;
        PUT_DEBUG(vPins_dps[i],"DC-Profile Current",currentProfile);
      }
    }
```

define and activate current profiling setup for individual DPS

current profiling begins at the beginning of the pattern execution

disable the current profiling

Retrieve measured current

Plotting current curve to Signal Analyzer

Test Method for Instantaneous IDD Profiling

Event-Insertion

the sampling interval and Frequency are used to calculate the cycle interval for event insertion.

```
else if(sMeaMode == "Event-Insertion")
{
    InsertionInterval = INT(ceil(mSampInterval*mFreqValue));
    if (InsertionInterval < 32)
    {
        cout << "Vector cycle Interval must be >= 32. Current vector cycle interval is " << InsertionInterval << "." << endl;
        return;
    }
    EVENT_RANGE_IM CurSetRange("SetCurMeasRange");
    CurSetRange.iRange(mCurrentRange_Min mA,mCurrentRange_Max mA);
    EVENT_IM CurMeasure("eCurMeasure");
    CurMeasure.averages(1);
    PATTERN &patternDynDC(patCtrl.label(mPattern,mPort));

    patternDynDC.vecEvents(sPins_dps).clearAll();
    patternDynDC.vecEvents(sPins_dps).insert(InsertionInterval,CurSetRange);
    patternDynDC.vecEvents(sPins_dps).insert(2*InsertionInterval,CurMeasure);
    for(int anchor = 3*InsertionInterval; anchor < mLength; anchor+=InsertionInterval)
    {
        patternDynDC.vecEvents(sPins_dps).reinsert(anchor,"eCurMeasure");
    }
    patternDynDC.vecEvents(sPins_dps).setup();
```

Create DC events and test points, inserts DC events at the **specified anchor** vector.

```
    FUNCTIONAL_TEST();
```

DC events are executed fully synchronized with digital data.

```
    for (i=0; i<iNum_dps_pins; i++)
    {
        result.uploadResult(vPins_dps[i],TM::RESULT_INDEX);
        ARRAY_D sample = result.getValues(vPins_dps[i]);
        cout << "Event-Insertion: Current on pin " << vPins_dps[i] << " = "<< sample << endl;
        PUT_DEBUG(vPins_dps[i],"Event-Insertion Current",sample);
```

Retrieve measured current

Plotting current curve to Signal Analyzer

Smoothing Method

Data smoothing method: "N-Points-Moving-Average" and " Discrete-Time-LPF"

```
if(sSmoothingMode == "None")
{
    // No smoothing method required
}
else if(sSmoothingMode == "N-Points-Moving-Average")
{
  if (mNumPoints % 2 == 0)
  {
      mNumPoints= mNumPoints + 1;
      cout << "Warning! Num points for moving average is adjusted to be an odd number: " << mNumPoints << endl;
  }
  ARRAY_D sample_temp = result.getValues(vPins_dps[i]);
  INT size = sample.size(), n = (mNumPoints-1)/2;
  DOUBLE sum_temp;
  for (INT t =n; t<size-n;t++)
  {
      sum_temp = 0;
      for (INT j = t - n; j <= t + n; j++ )
      {
          sum_temp = sum_temp + sample_temp[j];
      }
      sample[t]= sum_temp/mNumPoints;
  }
  cout << "Note: No smoothing on the first and last " << n << " points. Original raw data is used." << endl;
  for (INT t = 0; t<n; t++)
  {
      sample[t] = sample_temp[t];
      sample[size-1-t] = sample_temp[size-1-t];
  }
  cout << "Event-Insertion with Smoothing: Current on pin " << vPins_dps[i] << " = "<< sample << endl;
  PUT_DEBUG(vPins_dps[i],"Event-Insertion (Smoothing) Current",sample);
}
else if(sSmoothingMode == "Discrete-Time-LPF")
{
  for (INT t =1; t<sample.size();t++)
  {
      sample[t]=sample[t-1]*(1-mSmoothingFactor)+sample[t]*mSmoothingFactor;
  }
  cout << "Event-Insertion with LPF: Current on pin " << vPins_dps[i] << " = "<< sample << endl;
  PUT_DEBUG(vPins_dps[i],"Event-Insertion (LPF) Current",sample);
}
```

→ the mean is normally taken from an equal number of data on either side of a central value.

→ the algorithm simulates the effect of a low-pass filter on a series of digital samples.

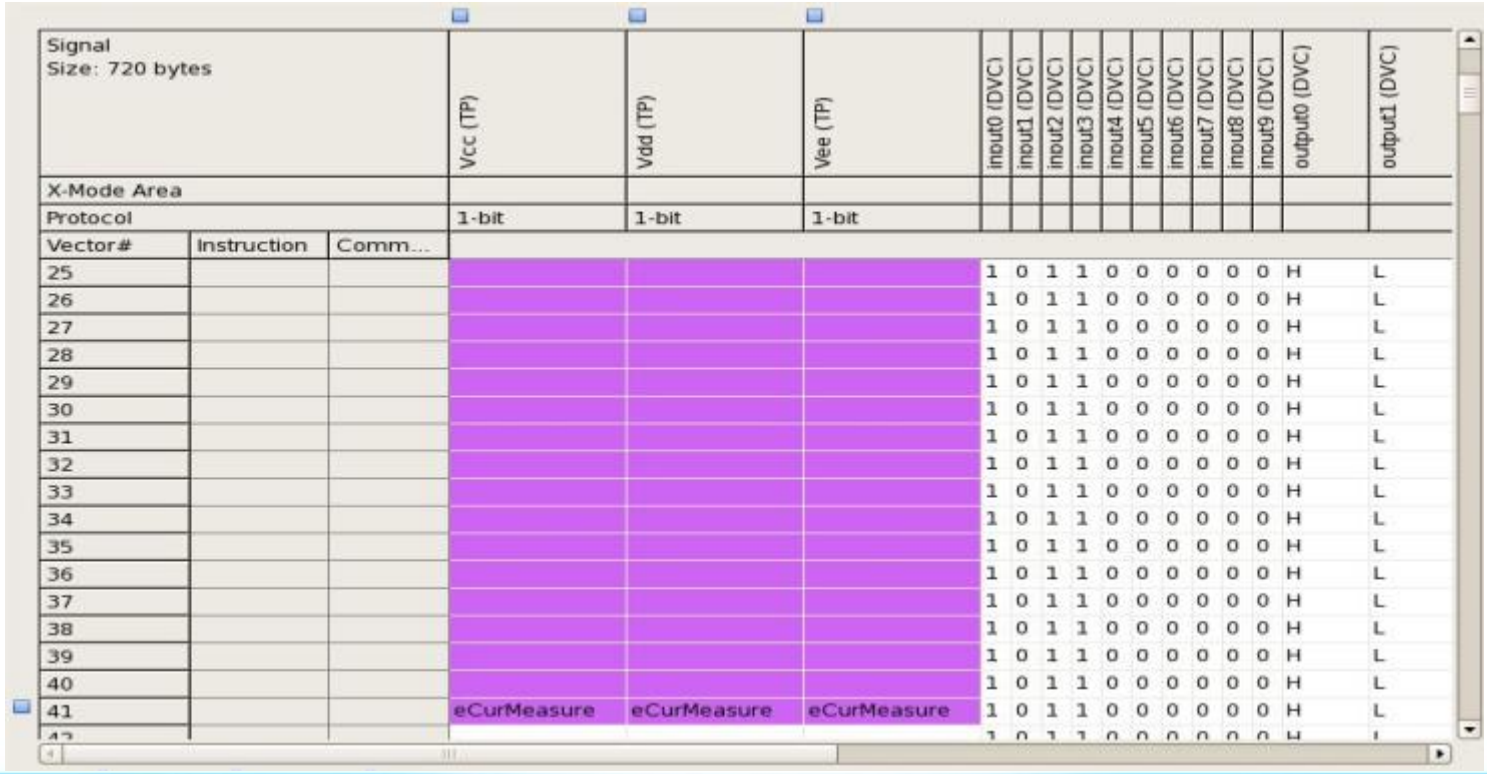
Test Method Parameters

Parameters	Remark	DC-Profiling	Event-Insertion
dps_pins	List of CSDPS separated by comma	√	√
Pattern	Pattern label containing the DPS		√
Port	Port containing the DPS		√
Measurement Mode	Measurement Mode using DC-Profiling or Event-Insertion	√	√
Num Meas Per Sample	Number of measurement per sample point	√	
Sampling Interval [us]	Sampling interval in time domain. Shortest interval is 50µs for DC-Profiling and 2µs for Event-Insertion. Recommend ≥ 2.2 µs.*	√	√
CurrentRange_Max [mA]	Max value for measurement range	√	√
CurrentRange_Min [mA]	Min value for measurement range		√
ToStopCycle	Till which cycle the measurement event is inserted (Max is last cycle of vector)		√
Frequency [MHz]	Frequency which the pattern is running at.*		√
Smoothing Mode	Smoothing method using N-Points-Moving-Average or Discrete-Time-LPF		√
Num Pts for Moving Average	Specify number of points for moving average. Must be odd number.		√
Smoothing Factor	Specify LPF smoothing factor, α with 0 < α < 1.		√

Test Vector Requirement

The TM could be used for all CSDPS current profiling, but test vectors changes could be needed depending on vectors “type”:

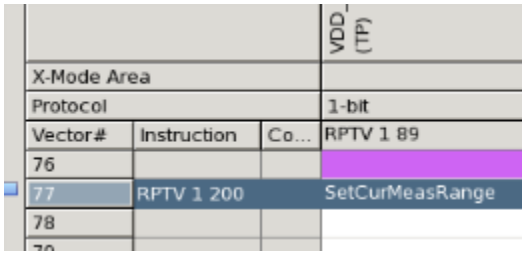
- **Flat vector without repeat cycle:**
  - Can directly use both DC-Profiling and Event-Insertion.



Signal Size: 720 bytes			Vcc (TP)	Vdd (TP)	Vee (TP)	inout0 (DVC)	inout1 (DVC)	inout2 (DVC)	inout3 (DVC)	inout4 (DVC)	inout5 (DVC)	inout6 (DVC)	inout7 (DVC)	inout8 (DVC)	inout9 (DVC)	output0 (DVC)	output1 (DVC)
X-Mode Area																	
Protocol			1-bit	1-bit	1-bit												
Vector#	Instruction	Comm...															
25						1	0	1	1	0	0	0	0	0	0	H	L
26						1	0	1	1	0	0	0	0	0	0	H	L
27						1	0	1	1	0	0	0	0	0	0	H	L
28						1	0	1	1	0	0	0	0	0	0	H	L
29						1	0	1	1	0	0	0	0	0	0	H	L
30						1	0	1	1	0	0	0	0	0	0	H	L
31						1	0	1	1	0	0	0	0	0	0	H	L
32						1	0	1	1	0	0	0	0	0	0	H	L
33						1	0	1	1	0	0	0	0	0	0	H	L
34						1	0	1	1	0	0	0	0	0	0	H	L
35						1	0	1	1	0	0	0	0	0	0	H	L
36						1	0	1	1	0	0	0	0	0	0	H	L
37						1	0	1	1	0	0	0	0	0	0	H	L
38						1	0	1	1	0	0	0	0	0	0	H	L
39						1	0	1	1	0	0	0	0	0	0	H	L
40						1	0	1	1	0	0	0	0	0	0	H	L
41			eCurMeasure	eCurMeasure	eCurMeasure	1	0	1	1	0	0	0	0	0	0	H	L

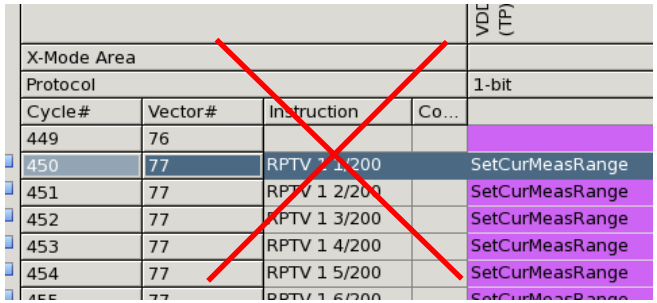
Test Vector Requirement

- **Flat vector with repeat cycle:**
  - For DC-Profiling, can directly use with flat vector.
  - For Event-Insertion, needs to convert flat vectors into multiport MBURST.
    - ❑The TM code is automated to insert event at specified anchor vector #.
    - ❑Potentially **violate the limitation** if the anchor is on the vector cycle with RPTV:

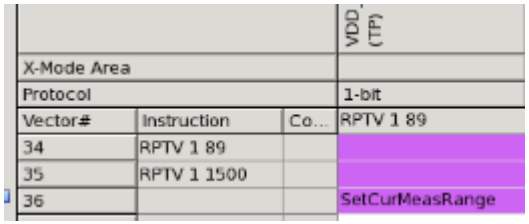


Expand

Code attempt to insert event at every cycle for that vector #, and hence violate the spec (requires >31 vectors interval).



❑Missing IDD profile "data" on those RPTV cycle:



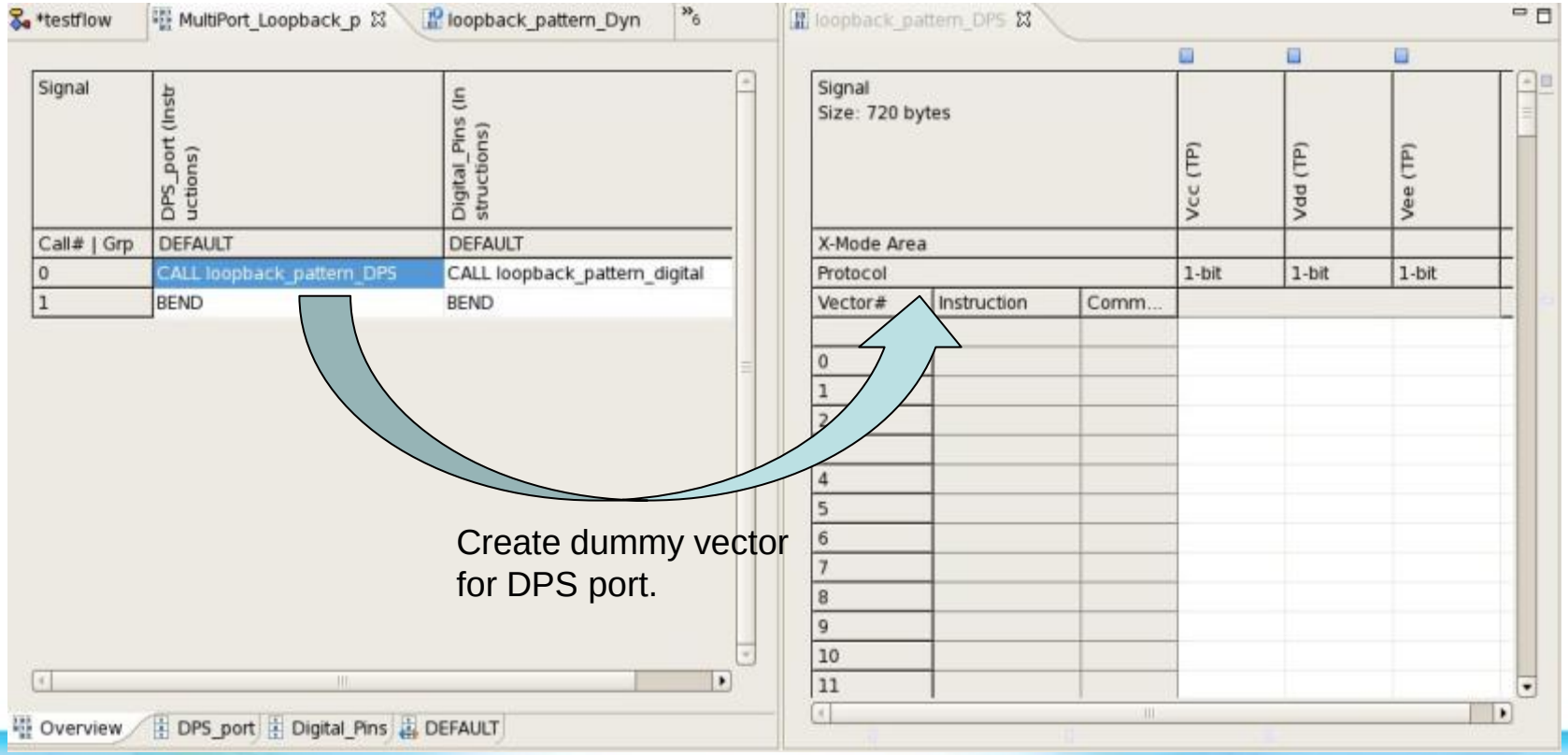
The IDD profile for the RPTV cycle is not measured. Final data sample is not "linear" in time domain (e.g. data sample 1 taken at 4us, sample 2 at 14us (due to 6us of RPTV cycle), sample 3 at 18us, ……. etc

❑Create a port for all non-DPS pin and a port for all DPS pin.

❑Convert the original flat vector to MBURST to include the DPS port which is pointing to a dummy vector having the same vector length as original flat vector.

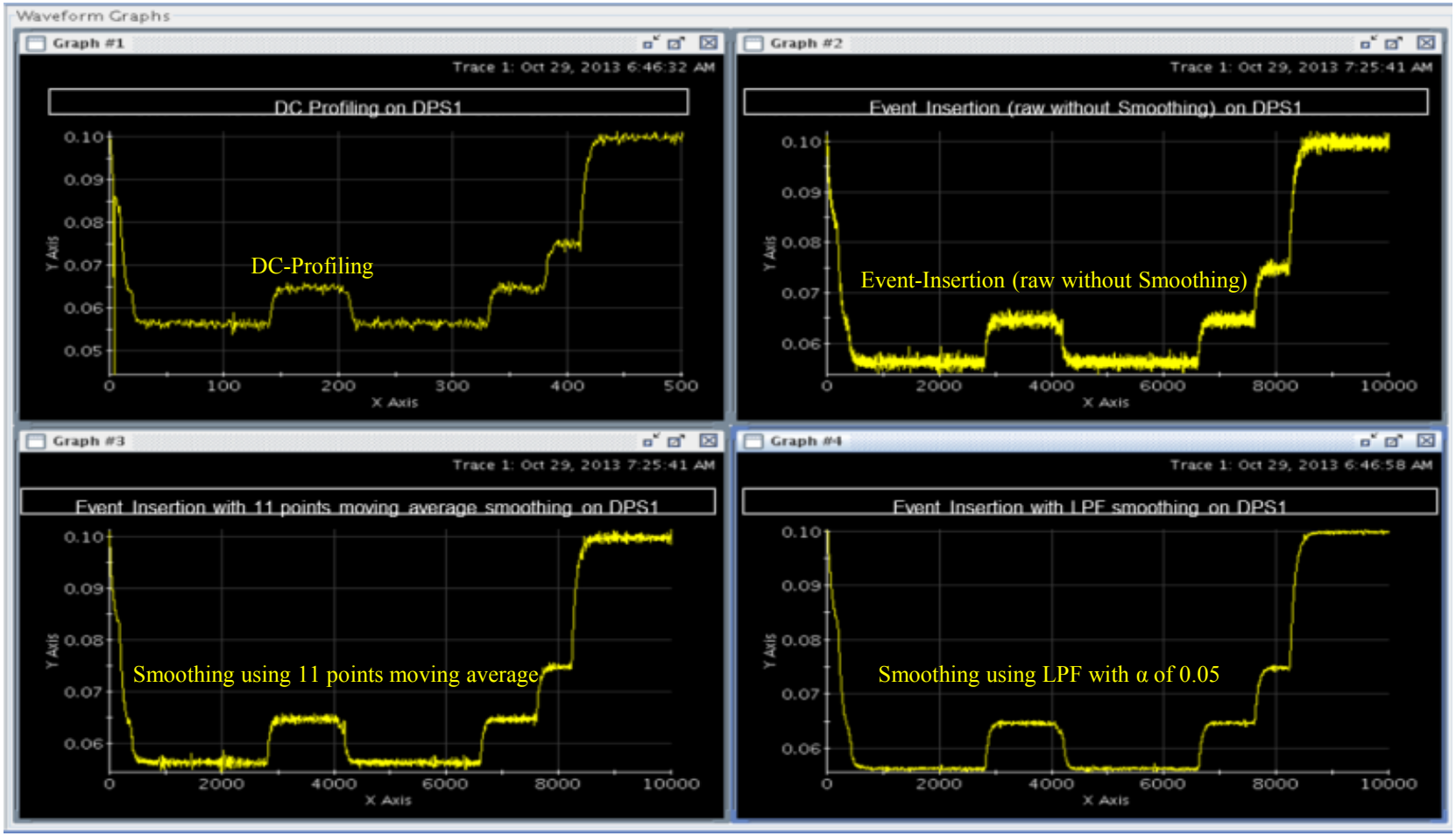
Test Vector Requirement

- **Multiport MBURST vector with or without repeat cycle:**
  - For both DC-Profiling and Event-Insertion method, need to create new MBURST vectors to include DPS port.
  - The DPS port needs to be pointing to a dummy vector with the same vector length as original vectors.



Sample Results (1) – “Large” Current Profiling

- Current measurement done on DPS1 on a vector A:

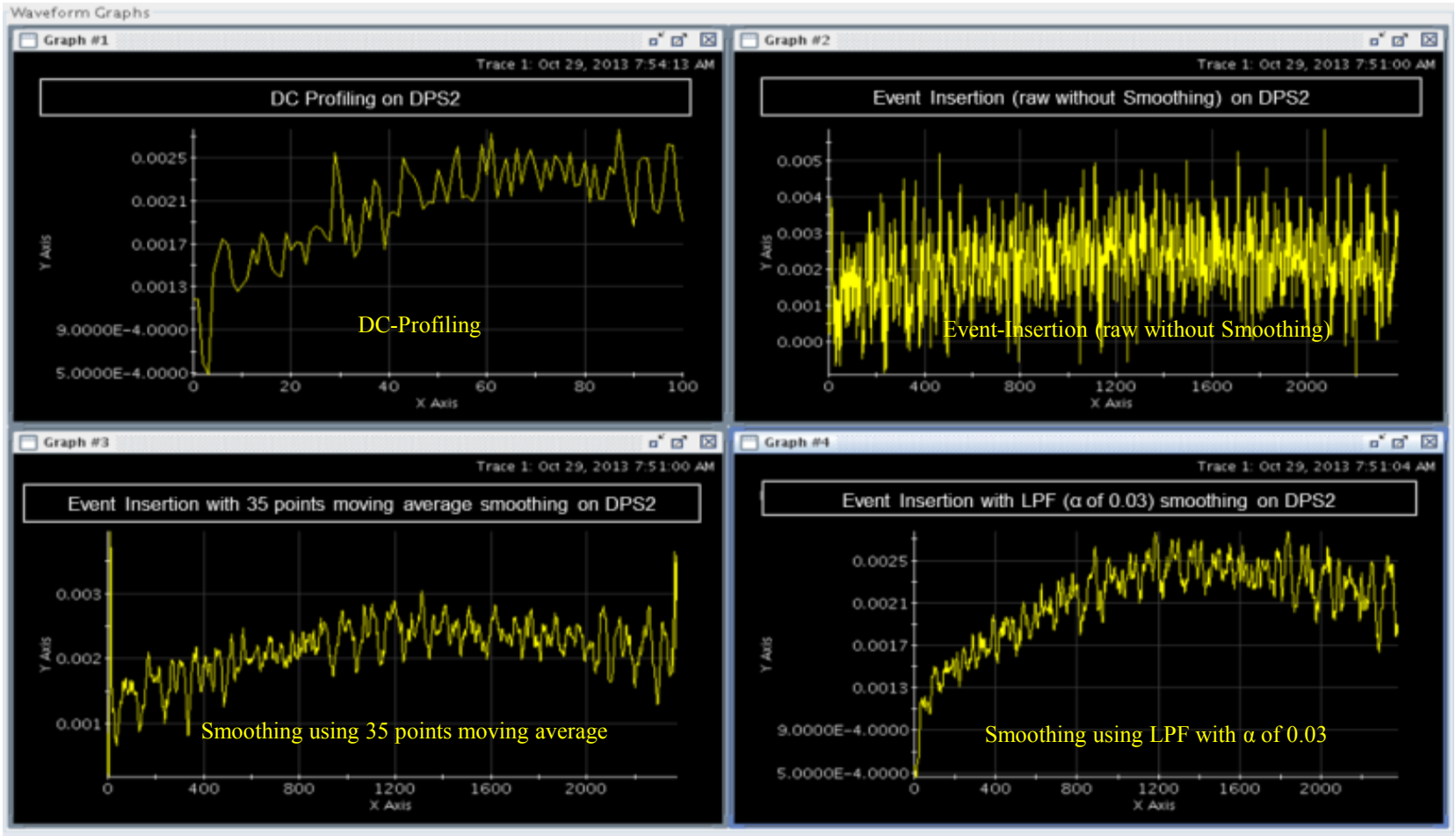


X-Axis: Measurement samples

Y- Axis: Current Measure Value, units is A

Sample Results (2) – “Small” Current Profiling

- Measurement done on DPS2 on a vector B:

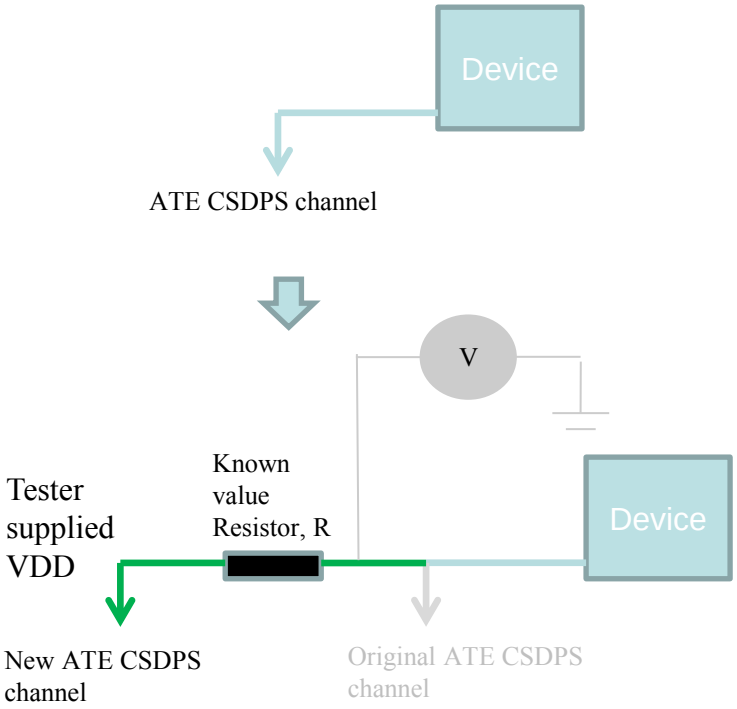


X-Axis: Measurement samples

Y- Axis: Current Measure Value, units is A

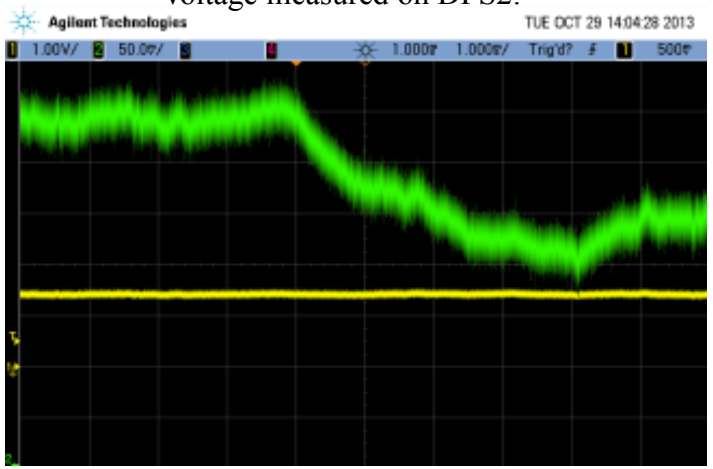
External Scope – "Current" Capture

- Load board used in this experiment/evaluation is modified to enable external scope measurement:

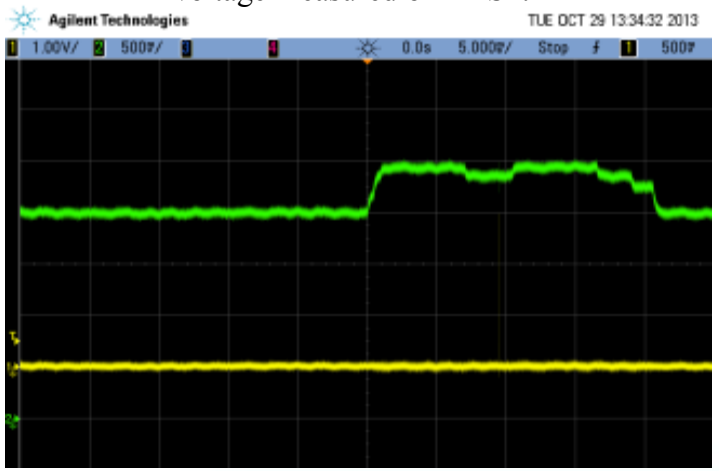


$$IDD = (V-VDD)/R$$

Voltage measured on DPS2:



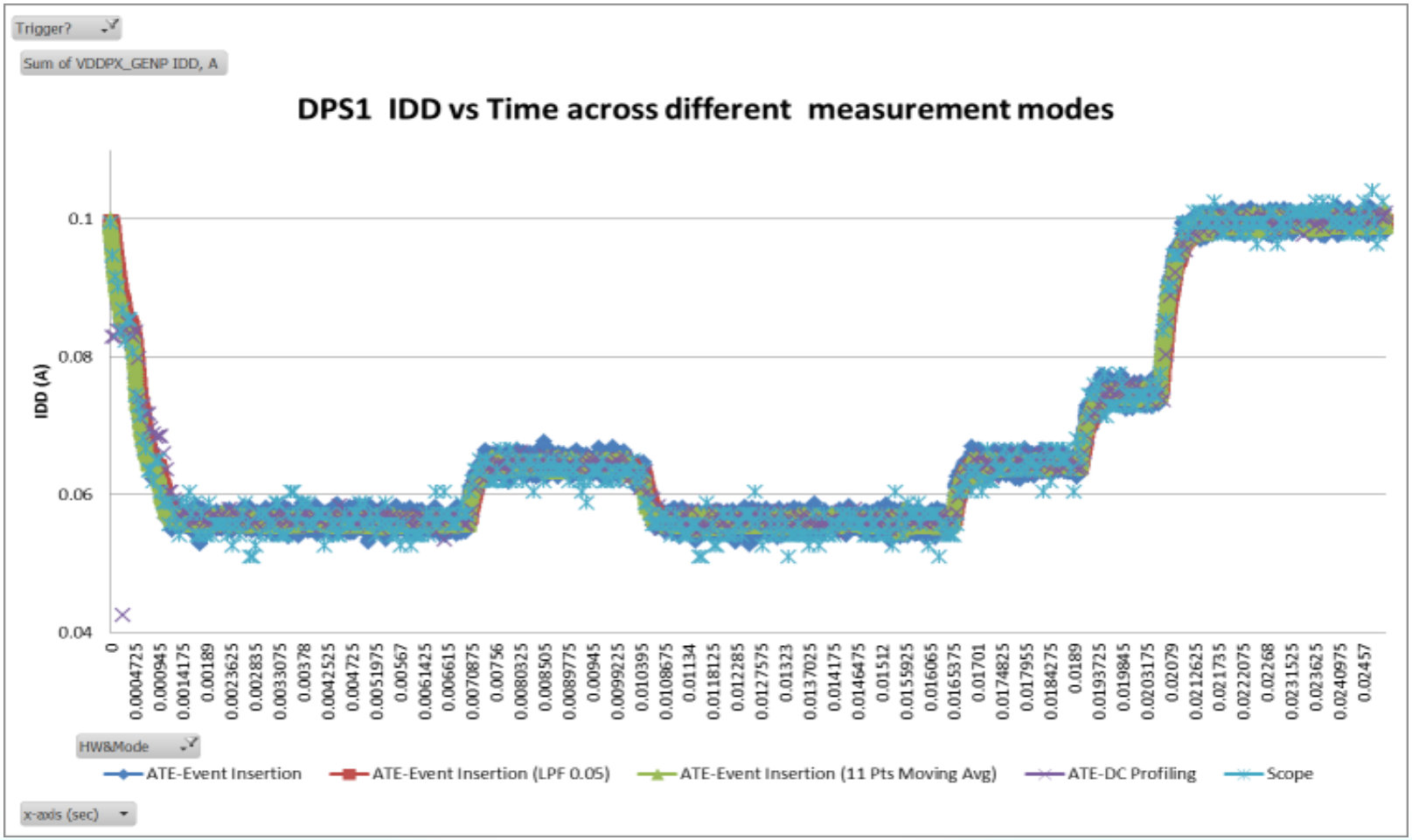
Voltage measured on DPS1:



Note: External trigger required to capture the scope result.

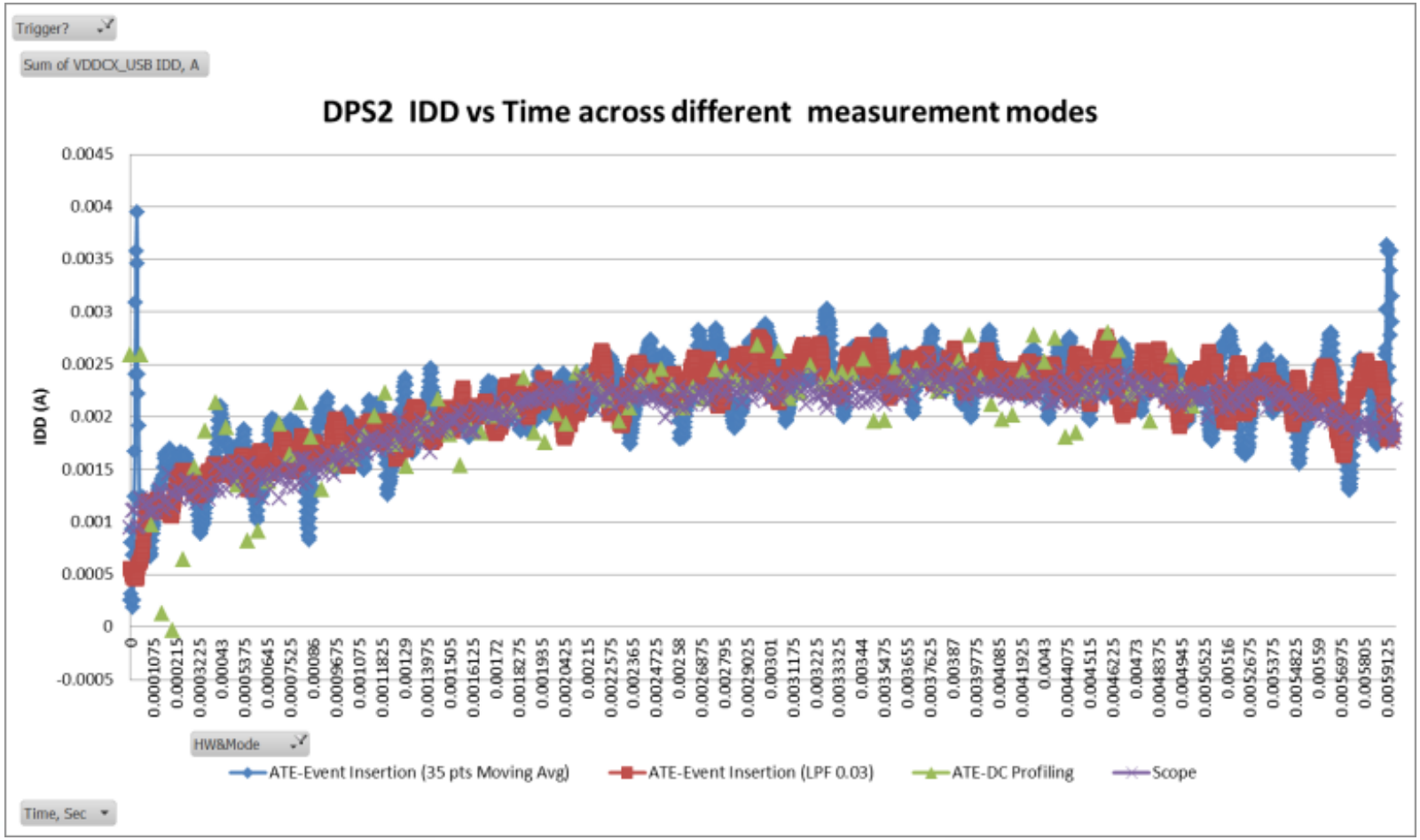
Test Method vs External Scope (1)

- Good matching between test method vs Scope for “large” current measurements on DPS1:



Test Method vs External Scope (2)

- Test method vs Scope for “small” current profiling on DPS2 shows good matching across DC profiling, smoothed Event-Insertion result and Scope:



DC Profiling vs Event Insertion

	DC Profiling	Event Insertion
Resolution	Poor with minimum sampling interval of 50µs	Better resolution with minimum sampling interval of 2µs
# Measurement per Sample	Worst case of 4 measurement per sample at 50µs sampling interval	1 measurement per sample
Ease of Use	Easy with very minimal test case changes	Slightly more difficult on certain vector type
Trigger Point	Throughout the vector	Flexible based on user-defined start and stop point
Accuracy	Good matching to scope	Poorer matching to scope on “small” current if without smoothing

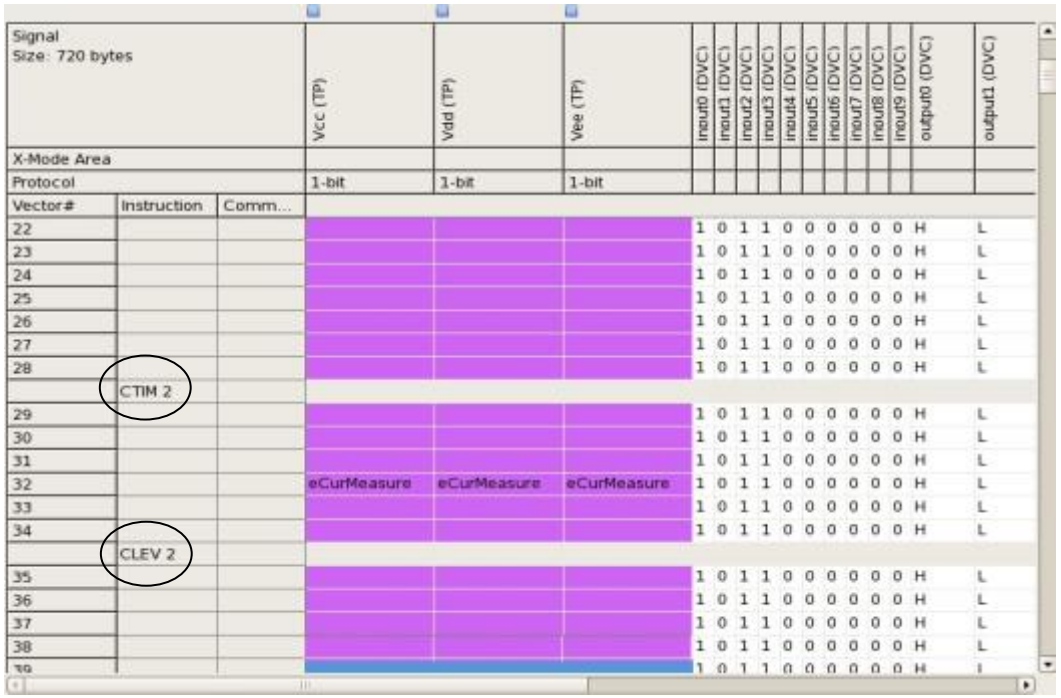
New CSDPS Hardware Support

- Other than DPS32, DC Profiling and Event Insertion both support new CSDPS hardware DPS128/DPS64.

	DPS128/DPS64		DPS32	
	DC Profiling	Event Insertion	DC Profiling	Event Insertion
Resolution	20µs	5µs	50µs	2µs
#Measurement per Sample	>4	1	>4	1
Trigger Point	throughout the vector	user-defined trigger point	throughout the vector	user-defined trigger point
limitation (>31 vectors interval)	NA	Yes	NA	Yes
Ease of Use	easy	depend on vector type	easy	depend on vector type
CTIM/CLEV support	Yes	Yes	No	No

Limitations and Enhancement Requests

- Limitation:
  - DC Profiling has poor resolution due to large sampling interval(50µs), not flexible if needs to delay trigger or stop trigger.
  - When there are CTIM and CLEV instructions used in vectors.



Both DC Profiling and Event Insertion don't support vectors with CTIM and CLEV instruction when DPS32 is used.

However, both TMs can support vectors with CTIM and CLEV instruction when DPS128/DPS64 are used.

- Enhancement request:
  - Need additional parameter to delay trigger for DC Profiling APIs. (CR-85492).

Summary

- Two test methodology are introduced: DC Profiling and Event Insertion both have good usability, and provide an efficient method to examine instantaneous current with CSDPS.
- The results from both methods are cross-checked with external scope measurement and good matching is obtained.
- Two smoothing methods: N-points moving average and Discrete time LPF are studied for more accurate current profiling.
- Real device data are shown and usability of these two method are discussed.
- Test methodology on the new CSDPS hardware: DPS64 &DPS128 are explored.

THANK YOU!