LAUTERBACH
DEVELOPMENT TOOLS

PowerIntegrator Setup Application Note

MANUAL

PowerIntegrator Setup Application Note

TRACE32 Online Help

TRACE32 Directory

TRACE32 Index

TRACE32 Documents .....

PowerIntegrator .....

PowerIntegrator Application Notes .....

PowerIntegrator Setup Application Note **1**

General **3**

Pin / Name Assignment **4**

Integrator Hardware Setup **5**

Threshold Level 5

Type of Recording 7

Timing Modes 7

State Mode 8

TestFocus 9

Filtering **10**

State Mode 10

Transient Detection 10

SupportPackage 11

Selective Trace 11

Analysis **12**

Protocol Analysis 12

Disassembly for Bustrace 13

PowerIntegrator Setup Application Note

Version 06-Jun-2024

General

The PowerIntegrator setup can be divided into four parts:

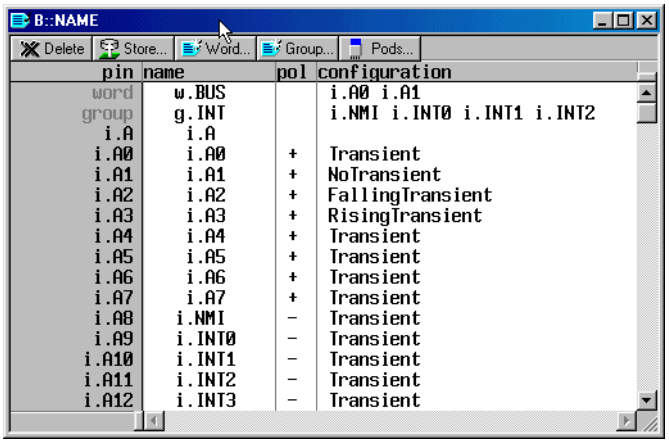
1. Definition of logical names for the input probe lines and target buses.
2. Setup of the PowerIntegrator hardware (voltage and timing).
3. Filter setup (qualify target events of interest).
4. Software analysis of the traced data.

Pin / Name Assignment

The NAME function is used to assign logical names to the input probe lines. These names are used by the display functions and the trigger unit programming.

Logical name assignment for busses (e.g. addressbus, databus) is done with the NAME.WORD function. A word is a set of bits, that will be displayed in hex, binary or decimal.

The target connector assignment typically can be found in the target schematics or the Processor Architecture Manuals.



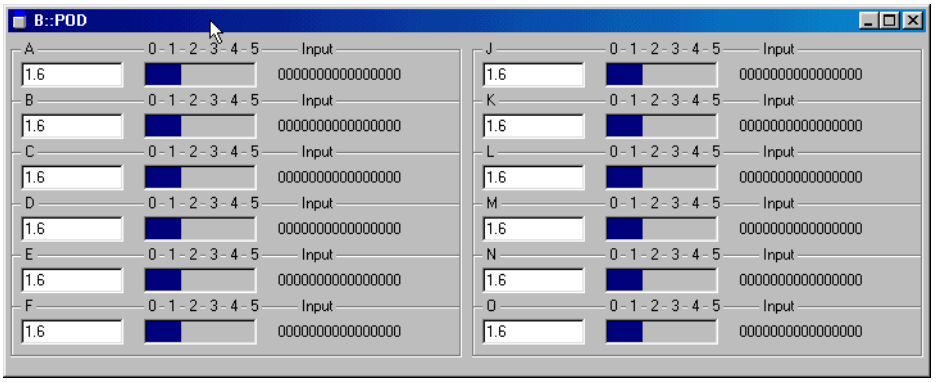
Integrator Hardware Setup

The PowerIntegrator can be used to trace a wide range of applications. This applications of course differ in voltage levels and timings, so a target-specific configuration is needed.

Threshold Level

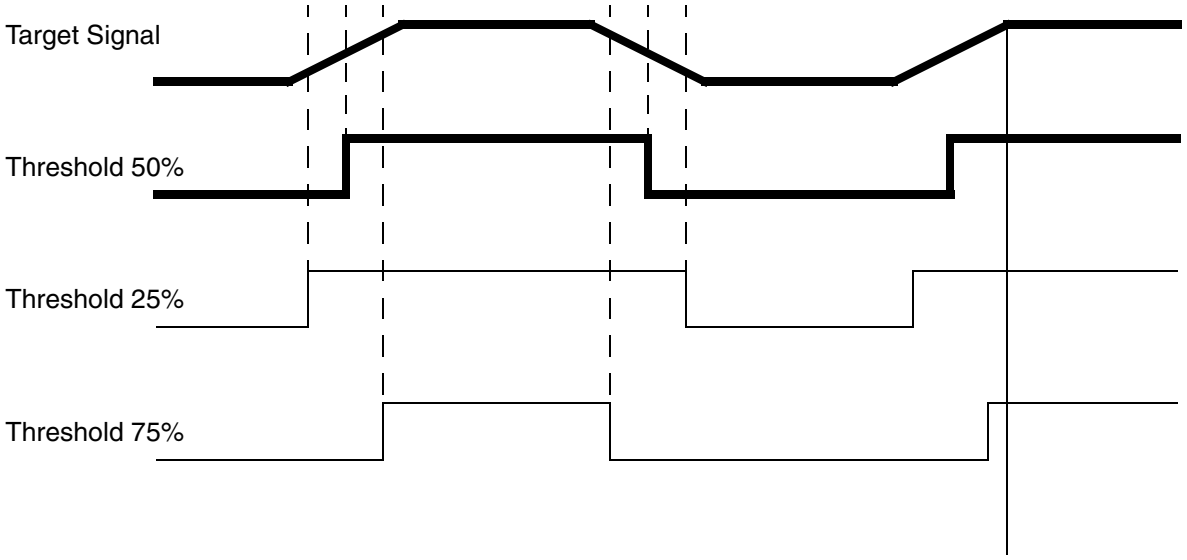
Each input probe has its own adjustable threshold level which can be set with the POD.LEVEL command.

For a first try it should be set to half of the target voltage level.



The POD window shows the selected threshold levels and the activity of the target signals.

A bad setting of the threshold level will cause wrong trace results. See here the different trace results influenced by the threshold level settings.



There are two scenarios which require a fine tuning of the threshold level.

- display of a wrong duty cycle (see diagram: different threshold level settings cause different duty cycle meassurements)
- display of spikes (caused by target signal overshoots)

Type of Recording

The selection of the appropriate recording type is essential for a good trace result. It depends on the application which method is best.

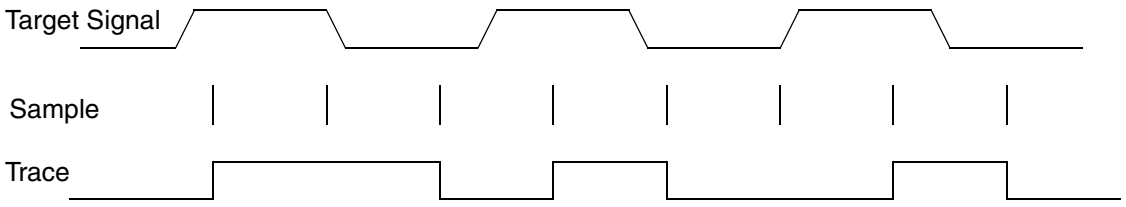
For a first try the timing mode **250MHZ** and especially the **Fixed500MHZ** mode gives a good impression of the target signal timing.

Timing Modes

250MHZ, 500MHZ and Fixed500MHZ mode work fine for signal pulses of minimum 5 ns (250MHZ mode) or 3 ns (Fixed500MHZ mode).

For 250MHZ mode the sampling interval is 4 ns. If the target signal pulse length is about the sampling interval it is no more guarantied that each signal pulse is detected. So the target signal pulse length has to be minimum 4ns plus some reserve. Pulses of 5ns can be detected without problem. Of course the timing display will show them as 4 ns or 8 ns. This depends on if the 5 ns pulse was detected once or twice.

So the resolution of the PowerIntegrator (4 ns/2 ns) causes a wrong display of the signals duty cycle. This is no fault, it is a consequence of the sampling method.



State Mode

State Recording is the appropriate method to trace synchronous bus protocols. A typical application is a clocked SDRAM bus where valid data is driven on the bus relative to the SDRAM clock.

Of course State Recording also can be used for a standard RAM interface where address, data and strobe information is valid on each rising edge of a chip select.

State Recording means:

Only the change in state of a single reference signal enables the trace recording.

The PowerIntegrator supports various methods for State recording.

- Timing Mode, with transient detection on one target signal (up to 100 MHz).
- State Mode (special Timing Mode), with transient detection on predefined probe CLK-pin (up to 200 MHz)
- StatePLL, the target clock itself generates the sampling signal (up to 200 MHz DDR)

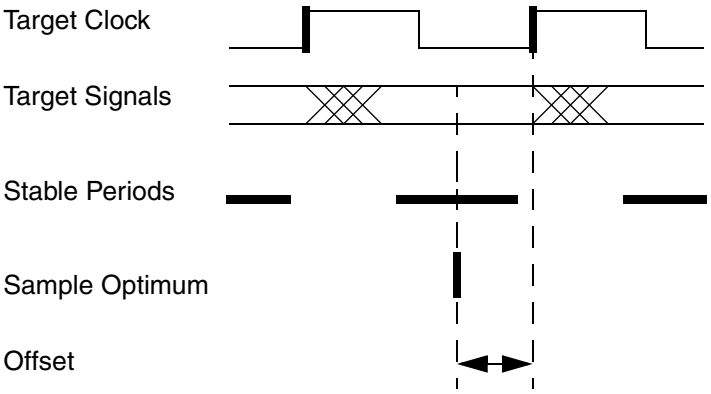
The setup for State Recording requires:

- selection of a **reference signal** (e.g. Target clock)
- selection of the **change in state type** (rising, falling clock edge)
- definition of a **data sampling offset** (data sampling before/after clock edge)

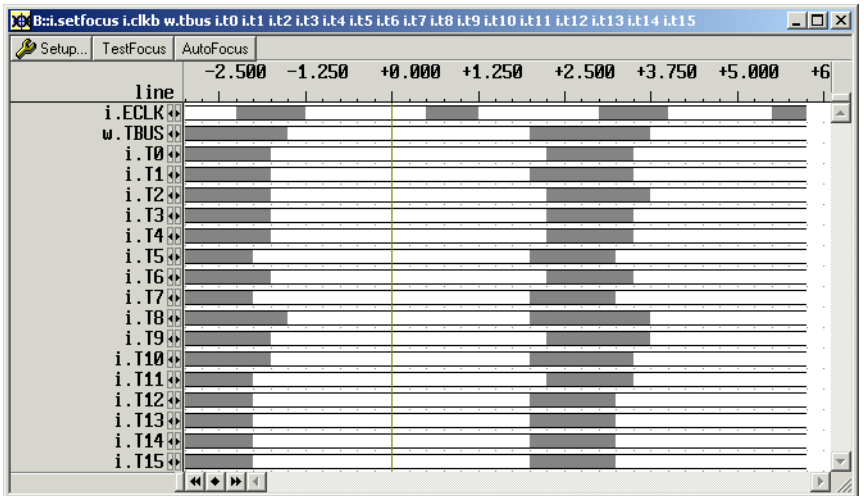
For more details see application note: STATE Modes

For State Mode the PowerIntegrator allows to adjust the sampling offset relatively to the clock edge. But which offset is best for the application under test?

In general the sampling should be done at that time where all target signals are stable (not toggling). The optimum is the middle in between the signal toggling.



To find Stable Periods the PowerIntegrator includes a comfortable analysis option called '**TestFocus**'. It does a signal analysis in the range of -3ns to +6ns around the clock edge. The signal has to be 'quiet' for a quarter of the target clock period to be marked as stable period.



The screenshot shows the measurement for a 200 MHz single data rate bus.

The current SAMPLE setup is 0.0ns (cursor line). A clock transition is detected 0.5 … 1.25 ns and 3.0 … 4.0 ns after the SAMPLE point. This indicates a rise and fall time of about 1 s.

All I.Tx lines are displayed separately. This way the different delays of all lines can be seen. The toggling summary of the complete W.TBUS can be displayed if all I.Tx lines are defined as a WORD (NAME.WORD …). The stable period for the TBUS is between -1.5 … 2 ns, so the sampling point of 0.0 ns fits perfect.

Filtering

To get a better utilization of the trace buffer and to simplify the analysis of the target protocol, it is best to filter not required target signal changes.

There are various levels of filtering:

- State Mode
- Transient detection
- Support Package (if available)
- Selective Trace (Complex Trigger)

State Mode

Uses a target specified sampling period (target clock period). Toggling of other target signals which might be present in between of two sampling points is ignored.

Transient Detection

Only the change in state of certain signals will cause a trace record. All toggling of other signals is ignored and will not cause a trace record.

Example: A typical scenario is an addressbus. Normally all address lines should toggle at the same time, but this is not true in real life. Because of capacitive load and PCB layout problems the signal traveling time varies. So it might happen it needs several nanoseconds till all address lines are stable. With a default setting of transient detection on each address line, this would cause several trace records till all address lines have found their final state. So it is better to use a transient detection on an address-strobe (address valid signal) and to disable the transient detection on the address lines.

SupportPackage

Does a preprocessing of the target signals in real-time before they are traced.

Example 1: A serial link needs several clocks to transfer a complete data package. Without Support Package all signal changes (clock and data) have to be traced. The trace for a complete data package will need a lot of trace records. A Support Package could convert the serial data flow into a parallel one. Only complete data packages would be traced.

Example 2: A SDRAM protocol needs several clocks for a complete data transfer. It generates a RAS cycle a CAS cycle and depending on burst configuration it generates several Data cycles. Without Support Package any clock transition has to be traced. Only this way the trace records with valid Data cycles can be calculated (SDRAM latency of Read and Write cycles is defined by a certain number of clocks). A Support Package could calculate the logical address in real-time. Only the calculated address with it's Data cycle would be traced.

Selective Trace

The combination of Data Selector, Address Selector and Complex Trigger unit allows to generate trace records on very special conditions. This is the most flexible and powerful tool for filtering.

Example: Trace only target memory cycles which access a certain address range.

Analysis

In many cases it is tedious to analyze the traced raw data manually. It is far more convenient to let the computer transform the traced raw data into an higher level of abstraction, for example into the display of transferred bytes, or even into a summary of sent/received packets.

Protocol Analysis

The TRACE32 software offers an extremely flexible feature to support such a protocol analysis: There is an open programming interface which allows to transfer the trace raw data to an user made analysis software (DLL) which returns the results to the TRACE32 software. The TRACE32 software can display the results in a separate window and it is easy to switch between different levels of abstraction.

More details see **"Protocol Analyzer Application Note"** (protocol_app.pdf).

For traced memory busses it is very important to get the linkage

- to the instruction set of the CPU in use to get a correct disassembly and
- to the source code of the executed program to show HLL source code and variables.

For the TRACE32 software this linkage works as soon as the trace data meets a special internal format (buscycle format).

A valid 'buscycle' is defined by

- Address
- Data
- Cycle Type (Fetch, Read, Write)
- Access Size (Byte, Word, Long, Quad)

In many cases the required 'buscycle' information is not present in the same trace record, so a special 'converter' algorithm is needed to generate a valid 'buscycle' record.

The TRACE32 software offers two ways to define a 'converter' algorithm.

The first one is based on TRACE32 commands which should be part of a script file. The command keyword is 'DisConfig'. It defines where to find Data (currently processed trace record) and where to find the belonging Address, Cycle Type and Access Size relatively to the current trace record in process.

For details see **"PowerIntegrator Trace DisConfig Application Note"** (powerintegrator_app_dc.pdf)

The second way is based on the Protocol Analysis (DLL) which has to fill a 'virtual' trace with the required 'buscycle' information.

A special DISCONFIG command will load/use the DLL.

For details see **"Protocol Analyzer Application Note"** (protocol_app.pdf).