NEONSEVEN

Trace32 system for E-GoldRadio

Debugging and developing using Trace32 by Lauterbach

NEONSEVEN

Support

- Trace32 environment support for E-GoldRadio is based on JTAG interface and Monitor resources stored internally in Internal RAM
- Link between target and host application is established by:
  - PowerTrace LA-7707
  - cable LA-7755 for OCDS
  - cable LA-7857 for peripherals
- Trace32 support is disabled during boot for security reasons

NEONSEVEN

Features

- Allow access to every addressable resource from inside

  E- GoldRadio

- Allow access to external accesses by hw sniffing on address and data buses.

- Allow manual suspension of execution or predetermined breakpoint setting (up to 4) by TARGET sw routine located at 0x18000 (start of internal RAM)

Environment configuration : abs (1)

- TARGET related debug information are stored in *.abs file (IEEE 695) generated at the end of build chain.
- IEEE166.EXE is used
  - TASKING C166/ST10 IEEE-695 formatter v7.5r5 Build 049
- The generation takes place in tasking166.mk

  createabs:

  ${EMU_CONV} ${MAKEDIR}/${PROJECTNAME}.${AXFFILEEXT} ${MAKEDIR}/${PROJECTNAME}.abs
  - Macro definition
    - **EMU_CONV** = ieee166
- Debug level is set using proper debug switches.
  - make –r BUILDMODE=DEBUG to get a build with debugging information

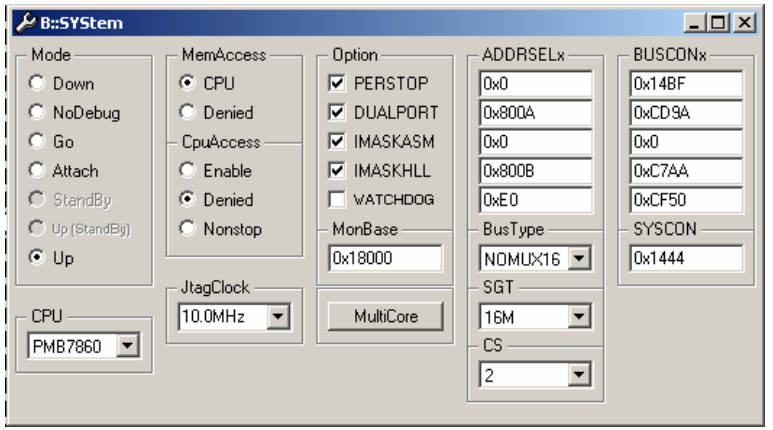
Environment configuration : abs (2)

- Debug information is generated only for those file that are rebuilt (obj not present or not up to date) when IEE166.exe is called

  →it is possible to set debug options only for a limited set of files by deleting the objs related and restarting compilation with debug flag set. This speeds up debugging process since it decreases considerably the time involved in abs generation.

Environment configuration : *.cmm (1)

- Configuration is based on a set of *.cmm scripts
- Main cmm file sets HOST system up (setting are used by Trace32 and does not affect TARGET system registers)
  - SYStem.Option MONBASE 0x18000
    - monitor routine address
  - SYStem.CPU PMB7860
    - System is EGoldLite/EGoldRadio
  - SYStem.Option or (sys.o in short format) is equivalent to manual set in B::System dialog

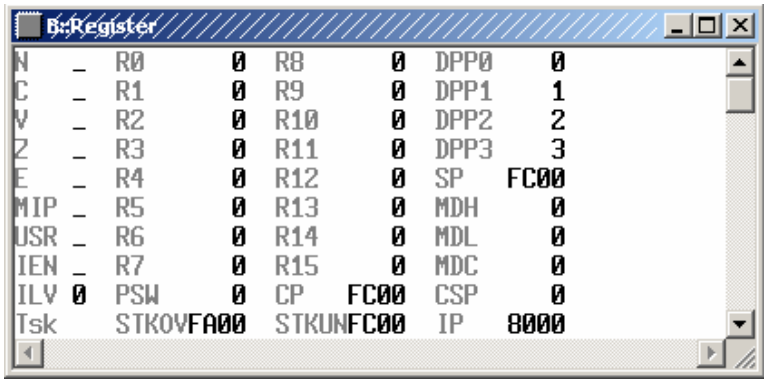


NEONSEVEN

Environment configuration : *.cmm (2)

Main cmm file sets TARGET system up

- Data.Set 0xff0c %w 0x05be ; BUSCON0
  - Every register (and peripheral) can be set by Data.Set (or d.s)
- Data.Set 0x18000 %b 0xec 0xf0 0xec 0x03 0xe6 0x03 0x03 0x00 0xcc 0x00 0xf2 0xf0 0xfc 0xf0 0x66 0xf0
  - The monitor routine is loaded to internal RAM
- R.S CSP 00
- R.S IP 8000
  - Registers are properly set
- Register.Set or (r.s in short format) is equivalent to manual set in B::Register dialog



Environment configuration : *.cmm (3)

- Main cmm loads debug informations stored in abs file
  - d.load.i ..\..\..\mmi_C166_mmi_GPRSEGL_RELEASE.abs /nocode /cfront
    - Path is relative to the path of the cmm script
  - y.spath + ..\..\..\..\..\inc\stack\target
    - Source symbol path for browse is added to current path
  - do ..\basicmmii.cmm
    - Other cmm scripts are executed by do batch command
  - d.v %SYMBOL.LONG register(sp) /TRACK
    - Stack pointer is accessed in order to show current stack fifo
    - Data.View (or short form d.v) enables runtime access to values)

Method Analyzer

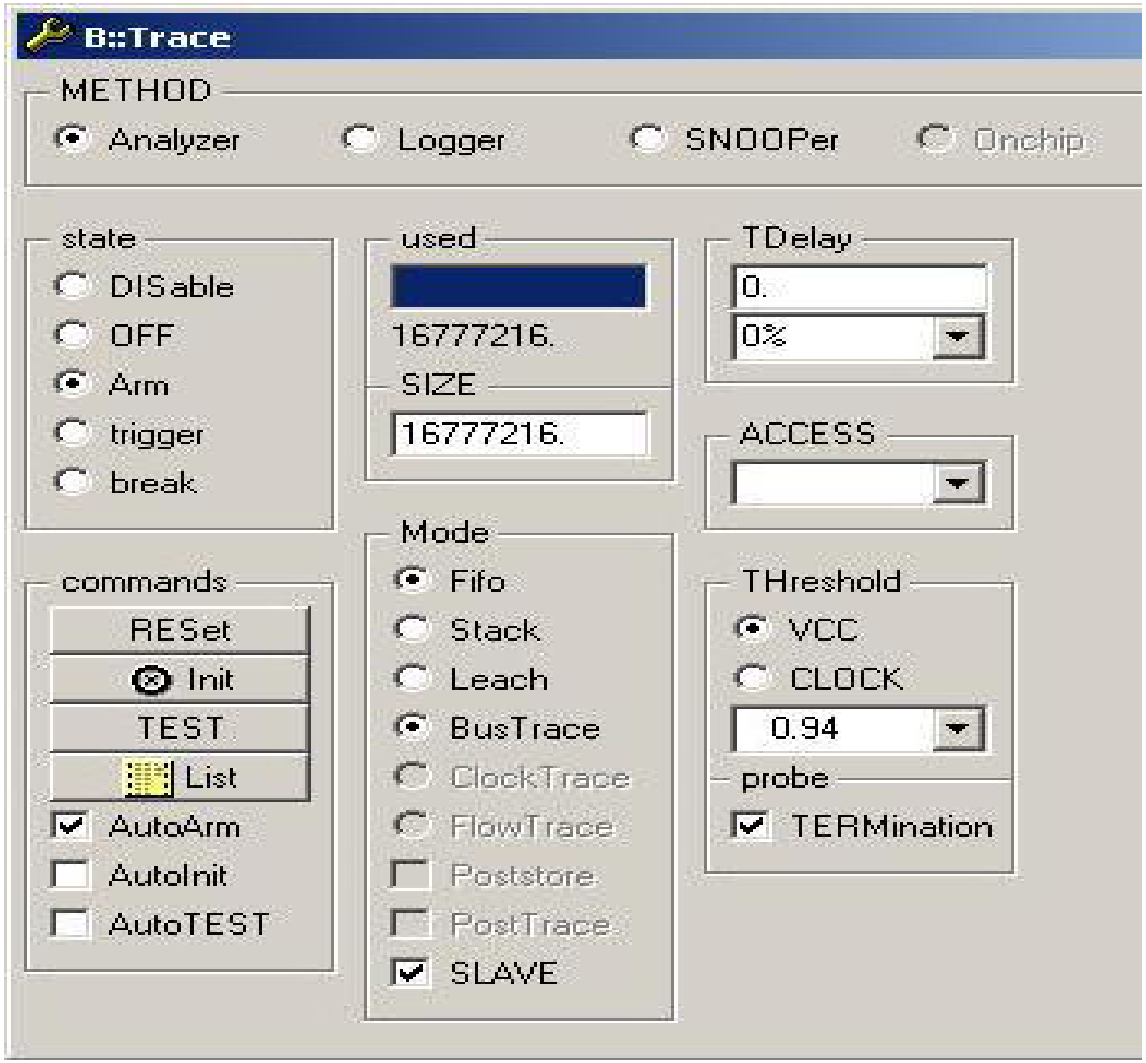
- To use the TRACE32-PowerTrace, TRACE32 development tool must be equipped with a physical trace memory with a trace depth of 16M frames or more.

- The intention of the method Analyzer is to provide a real-time trace to sample the program and data flow

  – While the user program is running the program and data flow is sampled into the trace memory

  – No influence on the real-time behavior

Method Analyzer: Configuration (1)



Method Analyzer: Configuration (2)

State Displays current trace status

- **OFF:** Indicates that trace memory and trigger unit are deactivated. The trace memory contents can be read.
- **Arm:** Indicates that trace is activated (for sampling and triggering events). A read out is impossible
- **break:** Indicates that the specified trigger event in the trigger unit has been encountered. At the same time, the trace is in OFF state for read out
- **Arm:** The trace buffer and the trigger unit is prepared for recording and triggering

Command

- **Reset:** All functions of the trace are reset to its default settings. The trace memory and the trigger unit are cleared. All user defined presets are cleared too
- **Init:** Trace memory contents are erased before each program execution and previous records are no longer visible. The trigger unit is returned to its initial setting
- **Test:** The same as executing the commands OFF, Init and Arm
- **AutoArm:** If set the “Arm” command will be executed automatically, when the user program is started. After program stops the trace will be switched into the OFF state automatically
- **AutoInit:** The Init command will be executed automatically, when the user program is started. Trace memory contents are erased before each program execution and previous records are no longer visible

Method Analyzer: Configuration (3)

Used Displays the actual number of entries in the trace buffer while the recording is running

Mode

- **Fifo:** If the trace is full, new records will overwrite older records. The trace records always the last cycles before the break
- **Stack:** If the trace is full recording will be stopped. The trace always records the first cycles after starting the trace
- **Slave:** trace memory and the trigger unit work during user program execution

TDelay Defines the trigger delay in entries or percentage of trace depth

Threshold used to optimize the threshold level for the trace lines in order to avoid indefinable FLOWERRORS

- **VCC:** The preprocessor and the TRACE32 software measure the VCC of the target. 1/2 VCC is then automatically used as the threshold level for the trace lines
- **CLOCK:** The threshold level is changed until the duty cycle of the trace clock reaches a ratio of 1:1. This setting is only recommended if the trace clock has a duty cycle of 1:1
- **Termination:** By default the trace line termination of the preprocessor is used. Indefinable FLOWERRORS may occur if the output drivers of the CPU are not strong enough. In this case it is recommended to switch the trace line termination OFF

NEONSEVEN

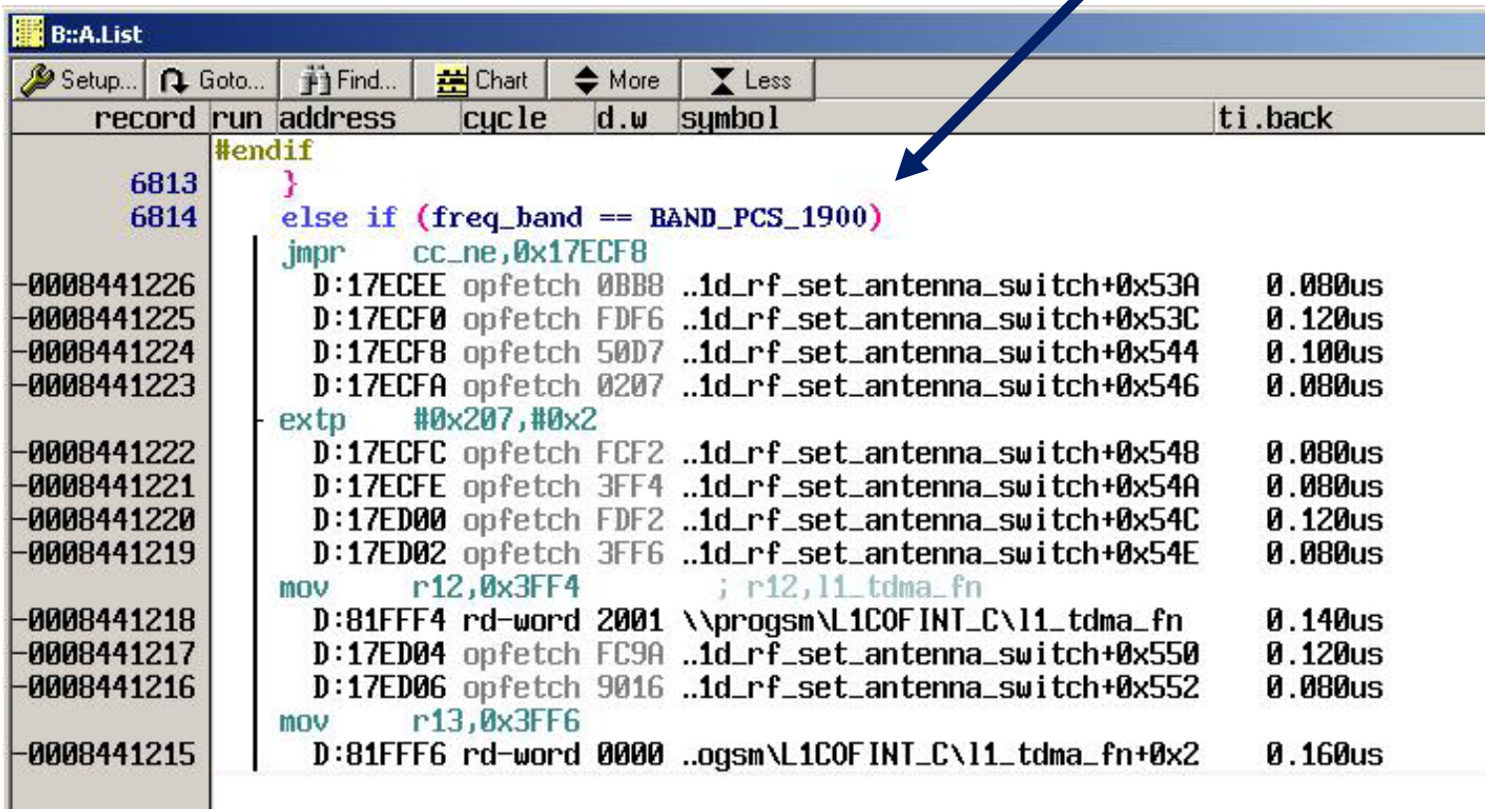
Method Analyzer: List

- List command is used to display the History buffer
- Listing the current trace memory contents is only possible when the trace status is set to OFF. Otherwise the list window is frozen and displays previous memory contents.
- If no parameters are specified, a predefined set of items will appear in the window in an order as defined by the command SETUP.ALIST

Method Analyzer: List Items (1)

- DEFault: Default selections The contents of the default selection can be configured with the command SETUP.ALIST
  - Command: Analyzer.List

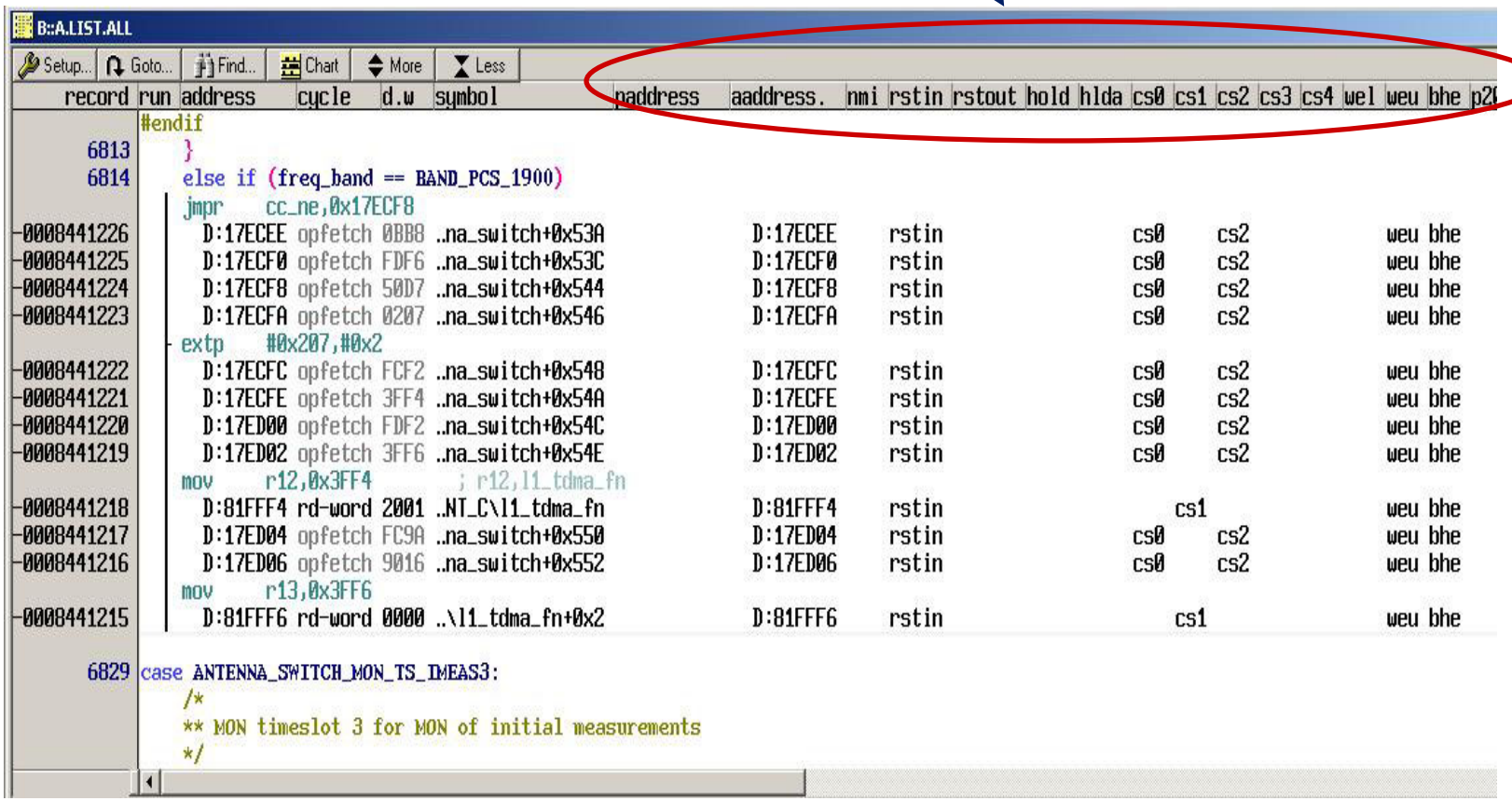
Default data



Method Analyzer: List Items (2)

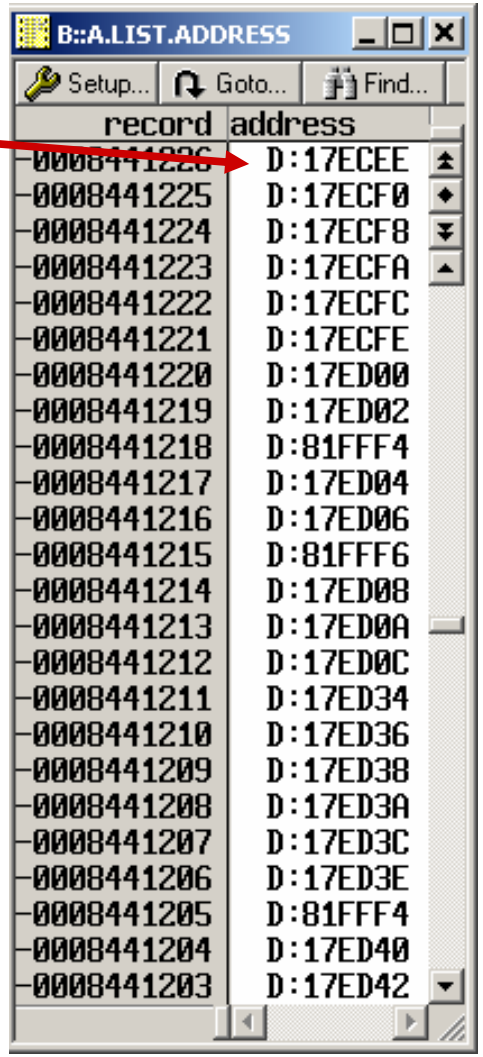
- ALL: Select all recorded data
  - Command: Analyzer.List.ALL

All record data



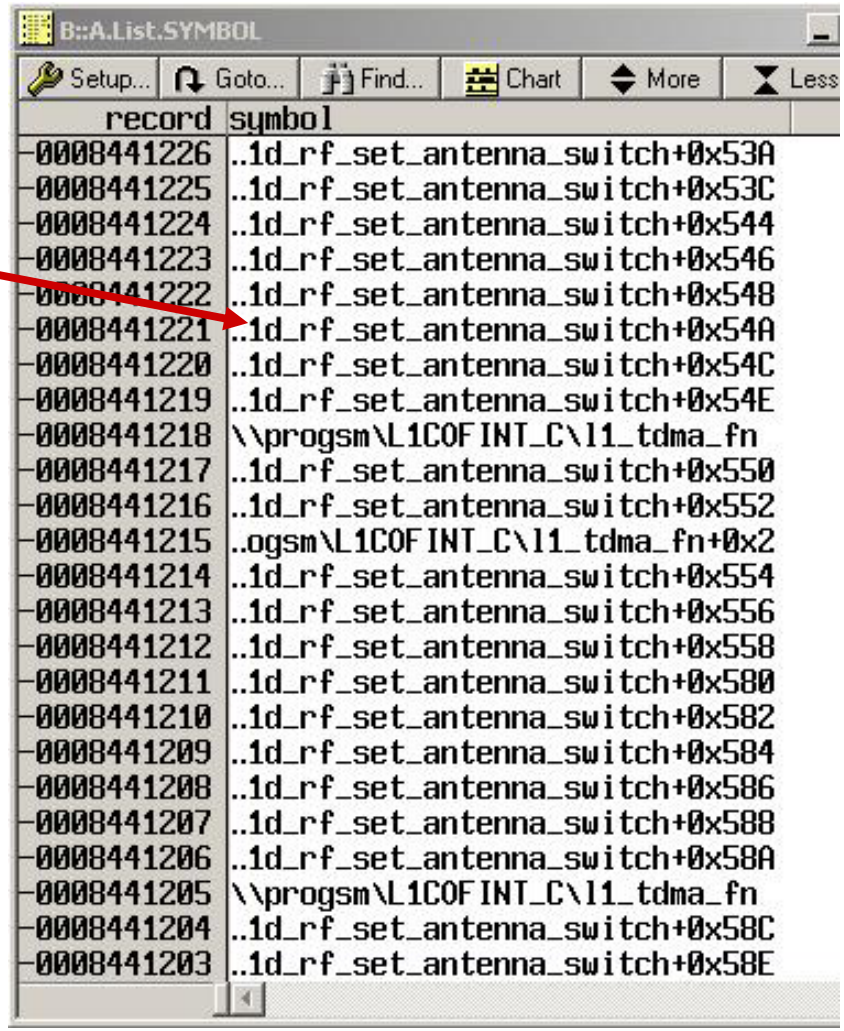
Method Analyzer: List Items (3)

- Address: Show only the CPU address
  - Command: Analyzer.List.ADDRESS



Method Analyzer: List Items (4)

- Symbol: Show only the Symbolic address with path and offset
  - Command: Analyzer.List.SYMBOL



Method Analyzer: List Items (5)

- Data.0--15: Show the CPU data bits 0 to 15 as single bits (16 bit processor)
  - Command Analyzer.List.DATA.0--15

Data 0BB8

B::A.LIST.DATA.0--15

Setup... Goto... Find... Chart More Less

record	d.0	d.1	d.2	d.3	d.4	d.5	d.6	d.7	d.8	d.9	d.10	d.11	d.12	d.13	d.14	d.15
-0008441226				d.3	d.4	d.5		d.7	d.8	d.9		d.11				
-0008441225		d.1	d.2		d.4	d.5	d.6	d.7	d.8		d.10	d.11	d.12	d.13	d.14	d.15
-0008441224	d.0	d.1	d.2		d.4		d.6	d.7					d.12		d.14	
-0008441223	d.0	d.1	d.2							d.9						
-0008441222		d.1			d.4	d.5	d.6	d.7			d.10	d.11	d.12	d.13	d.14	d.15
-0008441221			d.2		d.4	d.5	d.6	d.7	d.8	d.9	d.10	d.11	d.12	d.13		
-0008441220		d.1			d.4	d.5	d.6	d.7	d.8		d.10	d.11	d.12	d.13	d.14	d.15
-0008441219		d.1	d.2		d.4	d.5	d.6	d.7	d.8	d.9	d.10	d.11	d.12	d.13		
-0008441218	d.0													d.13		
-0008441217		d.1		d.3	d.4			d.7			d.10	d.11	d.12	d.13	d.14	d.15
-0008441216		d.1	d.2		d.4								d.12			d.15
-0008441215																
-0008441214		d.1	d.2			d.5	d.6	d.7								
-0008441213	d.0	d.1			d.4					d.9						
-0008441212			d.2	d.3			d.6	d.7								
-0008441211	d.0	d.1	d.2		d.4		d.6	d.7					d.12		d.14	
-0008441210	d.0	d.1	d.2							d.9						
-0008441209		d.1			d.4	d.5	d.6	d.7			d.10	d.11	d.12	d.13	d.14	d.15
-0008441208			d.2		d.4	d.5	d.6	d.7	d.8	d.9	d.10	d.11	d.12	d.13		
-0008441207		d.1			d.4	d.5	d.6	d.7	d.8		d.10	d.11	d.12	d.13	d.14	d.15
-0008441206		d.1	d.2		d.4	d.5	d.6	d.7	d.8	d.9	d.10	d.11	d.12	d.13		
-0008441205	d.0													d.13		
-0008441204		d.1		d.3	d.4			d.7			d.10	d.11	d.12	d.13	d.14	d.15
-0008441203	d.0	d.1	d.2	d.3	d.4											d.15

Method Analyzer: List Items (6)

- **Time**
  - Command: Analyzer.List.DEFAULT ti.back ti.zero ti.ref ti.fore

Relative time to previous record

Relative to Global reference

Relative to reference point

Relative time to next record

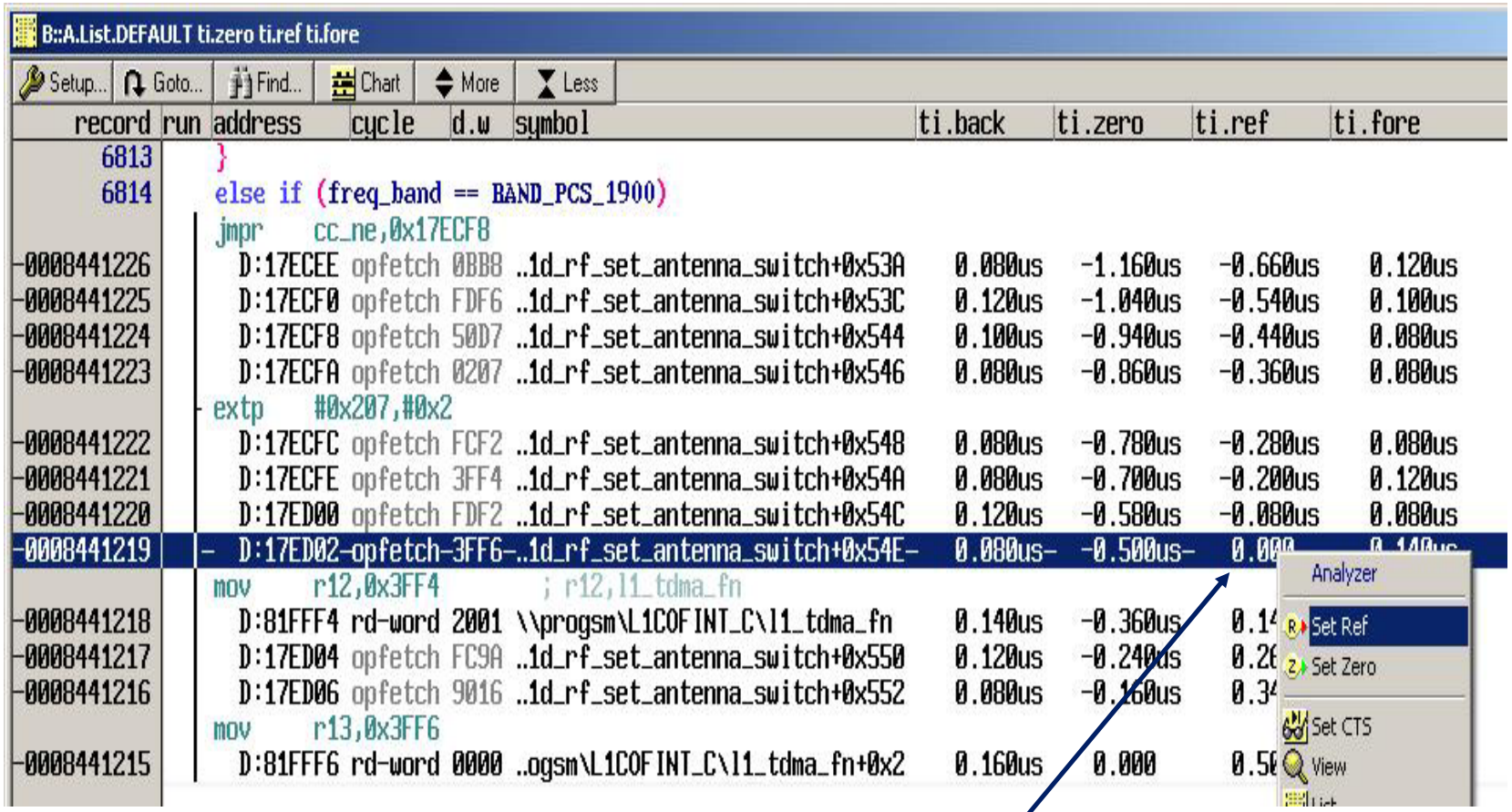
B::A.List.DEFAULT ti.zero ti.ref ti.fore

Setup... Goto... Find... Chart More Less

record	run	address	cycle	d.w	symbol	ti.back	ti.zero	ti.ref	ti.fore
6813		}							
6814		else if (freq_band == BAND_PCS_1900)							
		jmpr cc_ne,0x17ECF8							
-0008441226		D:17ECEE opfetch 0BB8			..1d_rf_set_antenna_switch+0x53A	0.080us	89.501ks	-1.580s	0.120us
-0008441225		D:17ECF0 opfetch FDF6			..1d_rf_set_antenna_switch+0x53C	0.120us	89.501ks	-1.580s	0.100us
-0008441224		D:17ECF8 opfetch 50D7			..1d_rf_set_antenna_switch+0x544	0.100us	89.501ks	-1.580s	0.080us
-0008441223		D:17ECFA opfetch 0207			..1d_rf_set_antenna_switch+0x546	0.080us	89.501ks	-1.580s	0.080us
		extp #0x207,#0x2							
-0008441222		D:17ECFC opfetch FCF2			..1d_rf_set_antenna_switch+0x548	0.080us	89.501ks	-1.580s	0.080us
-0008441221		D:17ECFE opfetch 3FF4			..1d_rf_set_antenna_switch+0x54A	0.080us	89.501ks	-1.580s	0.120us
-0008441220		D:17ED00 opfetch FDF2			..1d_rf_set_antenna_switch+0x54C	0.120us	89.501ks	-1.580s	0.080us
-0008441219		D:17ED02 opfetch 3FF6			..1d_rf_set_antenna_switch+0x54E	0.080us	89.501ks	-1.580s	0.140us
		mov r12,0x3FF4 ; r12,l1_tdma_fn							
-0008441218		D:81FFF4 rd-word 2001			\\progsm\L1COFINT_C\l1_tdma_fn	0.140us	89.501ks	-1.580s	0.120us
-0008441217		D:17ED04 opfetch FC9A			..1d_rf_set_antenna_switch+0x550	0.120us	89.501ks	-1.580s	0.080us
-0008441216		D:17ED06 opfetch 9016			..1d_rf_set_antenna_switch+0x552	0.080us	89.501ks	-1.580s	0.160us
		mov r13,0x3FF6							
-0008441215		D:81FFF6 rd-word 0000			..ogsm\L1COFINT_C\l1_tdma_fn+0x2	0.160us	89.501ks	-1.580s	0.100us

Method Analyzer: List Items (7)

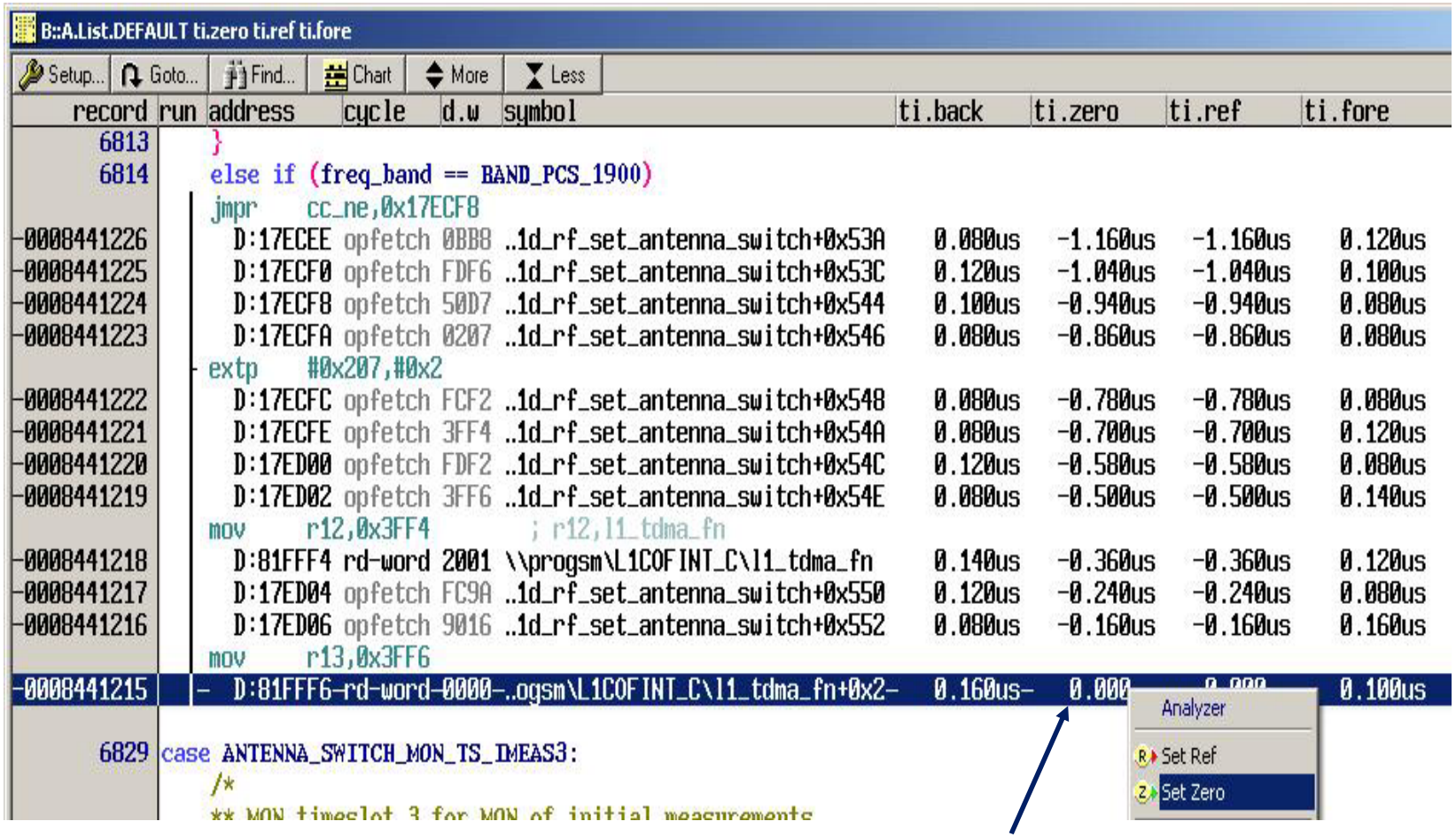
- **Time: Set Ti.REF**



Set ti.ref

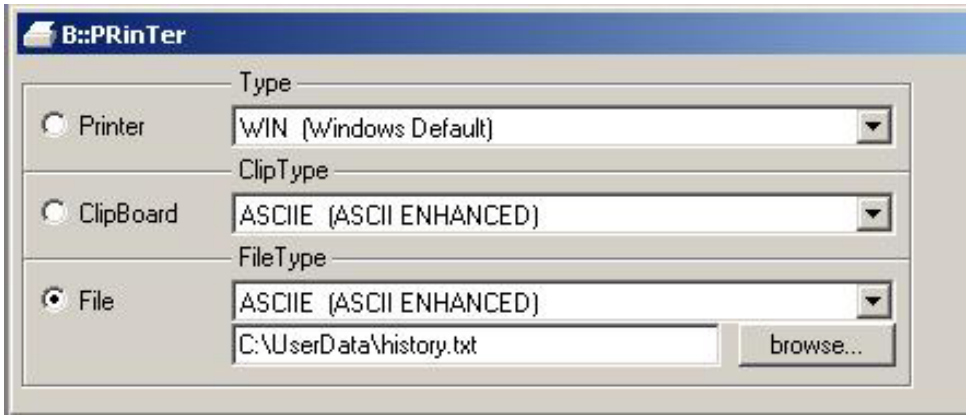
Method Analyzer: List Items (8)

- **Time: Set Ti.zero**



Method Analyzer: Storing history buffer on file (1)

1. Configure the file type and file name



2. Use the command:

   winprint.[windows name] (Start index)--(End index) [param]

   Note! Index is in HEX format. To express it in DEC format is needed to insert "." at the end of index number

Method Analyzer: Storing history buffer on file (2)

- E.g.

  Store the history buffer from -2000 to -1000

  winprint. Analyzer.List (-2000.)--(-1000.) Default

  Dec format

  Store the Default params

- Store only address, d.w and symbol from -8192 to -4096

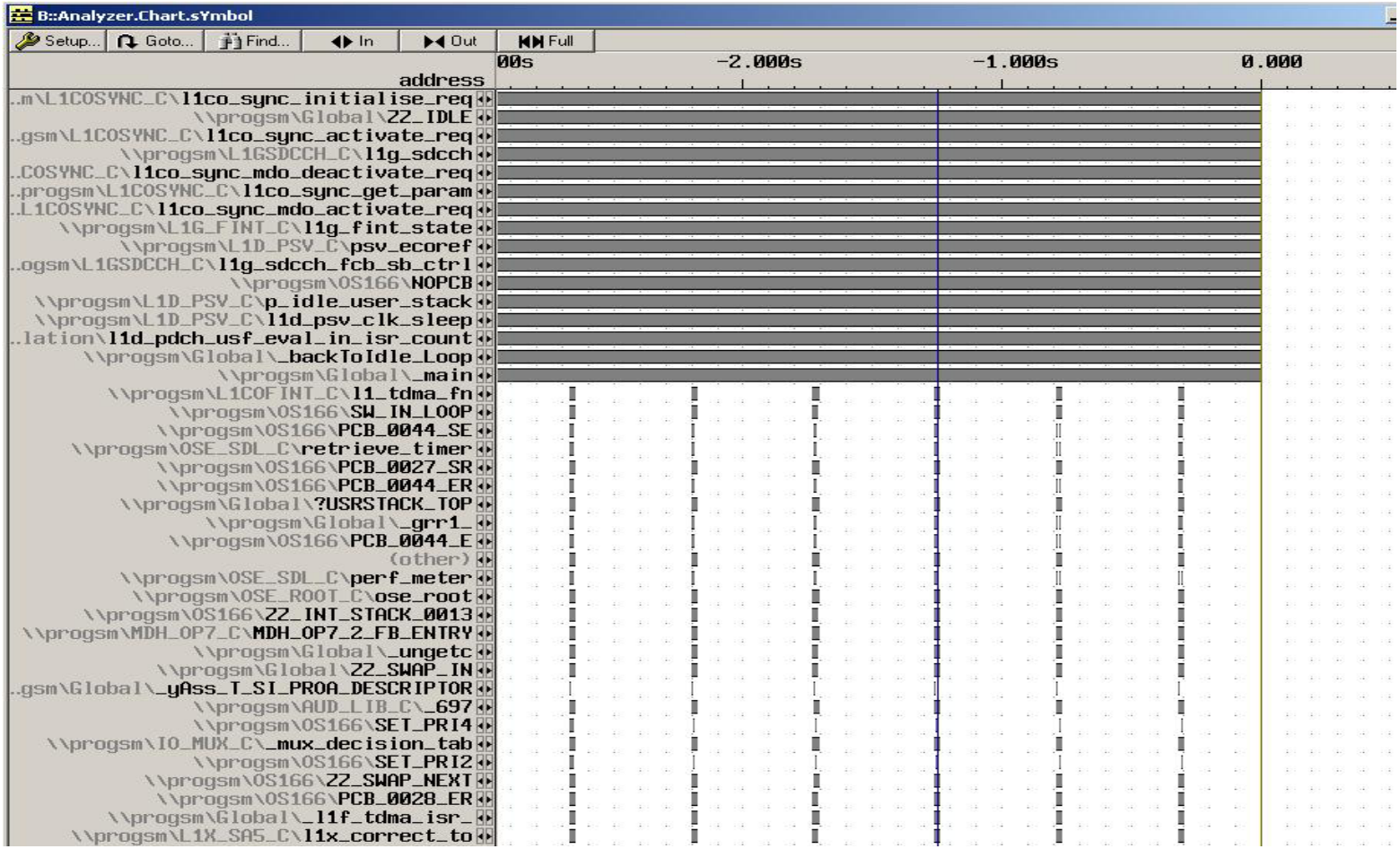
  winprint. Analyzer.List (-2000)--(-1000) address d.w symbol

  HEX format

  Store only address d.w and symbol

Method Analyzer: Chart (1)

The result of a task or function analysis can be displayed in graphical form by the chart commands



Method Analyzer: Chart (2)

- **Sort**
  - Use command: Analyzer.STATISTIC.SORT [<mode>]

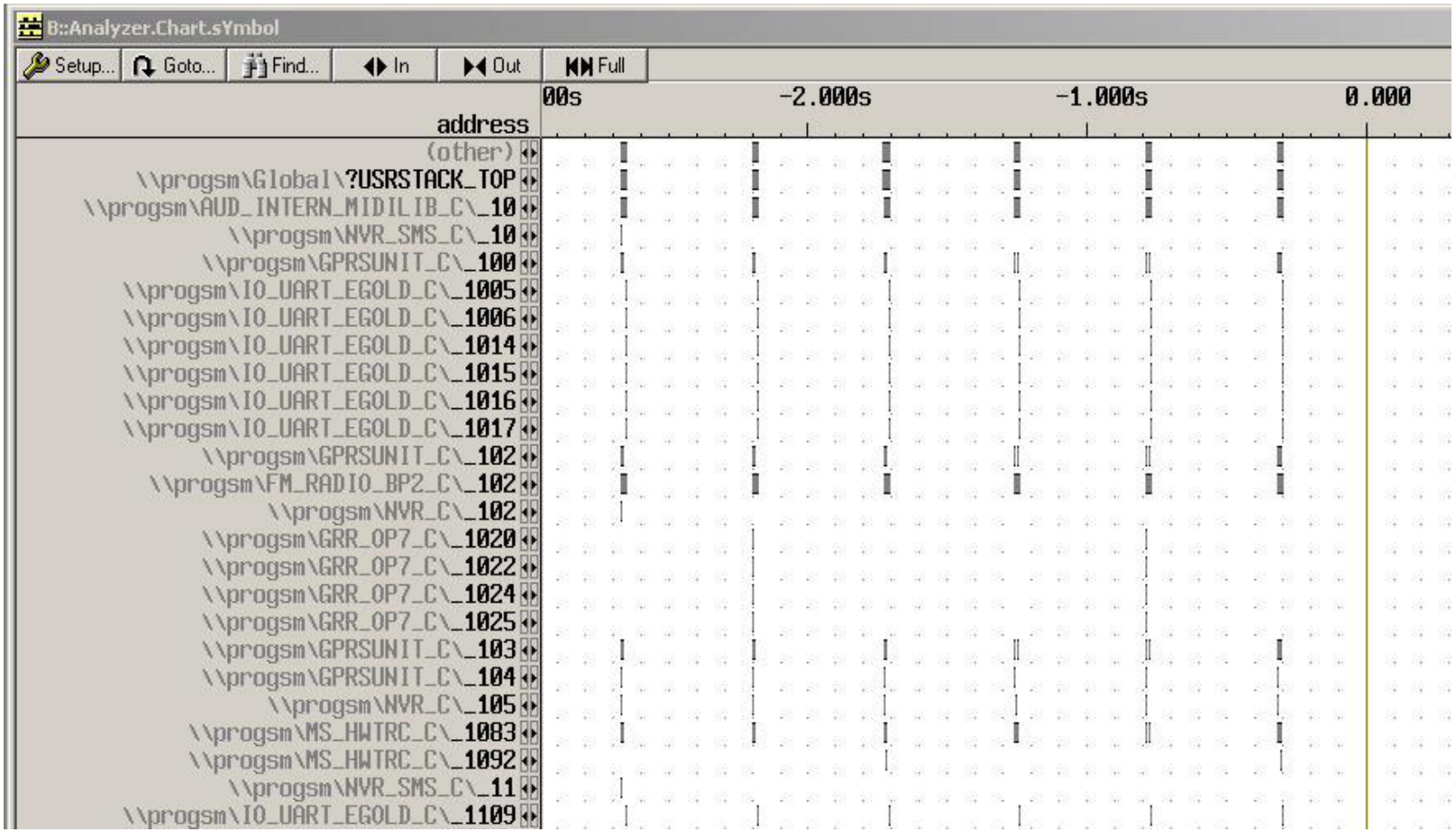
<mode>: OFF | Ratio | Address | sYmbol | Count

This command selects the display mode for the commands <trace>.Chart.

If OFF mode is selected, the entries will be displayed in the order as they are encountered in the trace memory (this is the default).

Method Analyzer: Chart (2)

- E.g: Sort by sYmbol -> Analyzer.STATISTIC.SORT SYMBOL



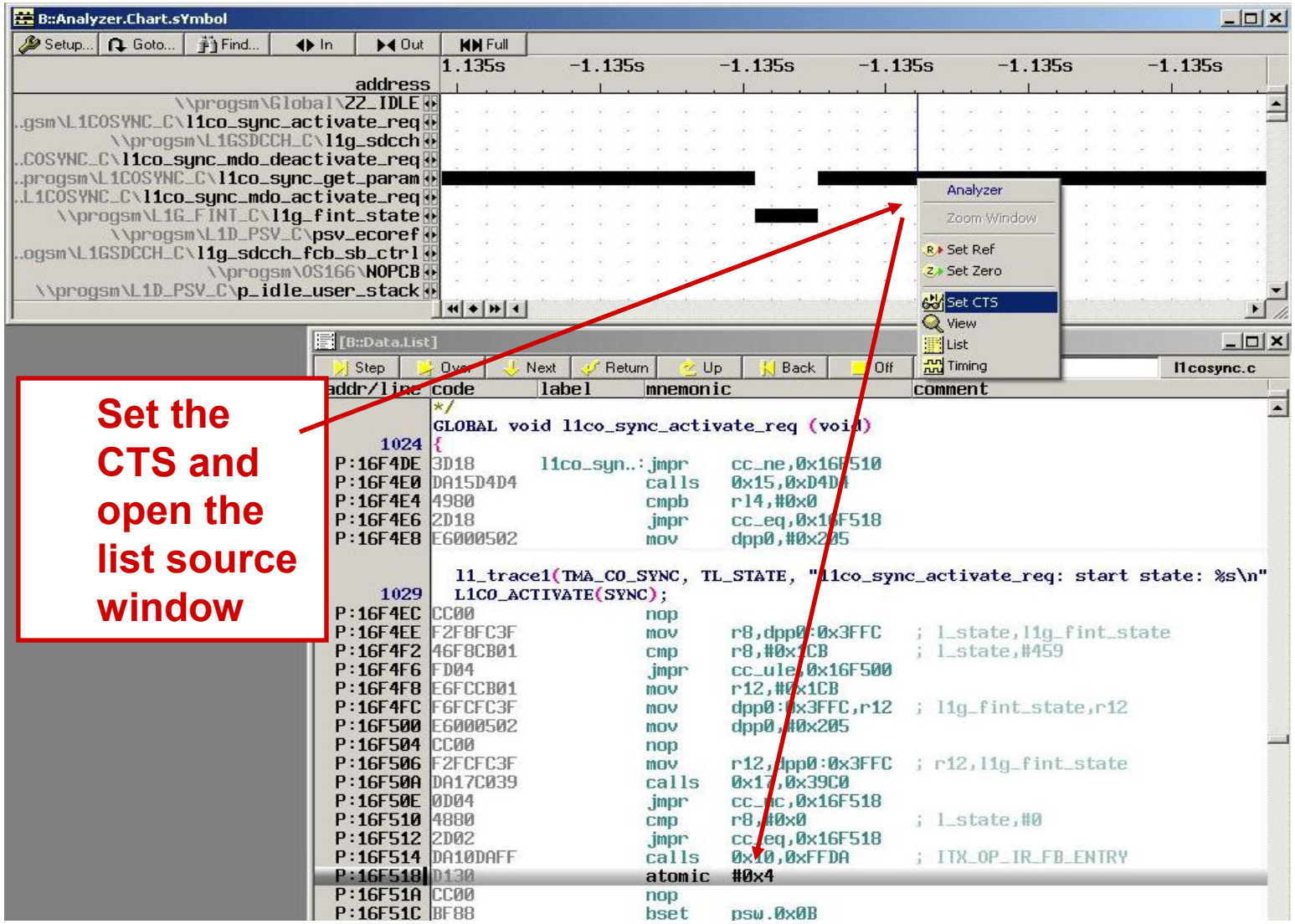
NEONSEVEN

Method Analyzer: CTS (1)

- The Context Tracking System (CTS) can recover the target context (i.e. register and memory contents) at an arbitrary time sampled in the tracebuffer. This allows to "replay" and single step program execution that was sampled before in a real-time run.

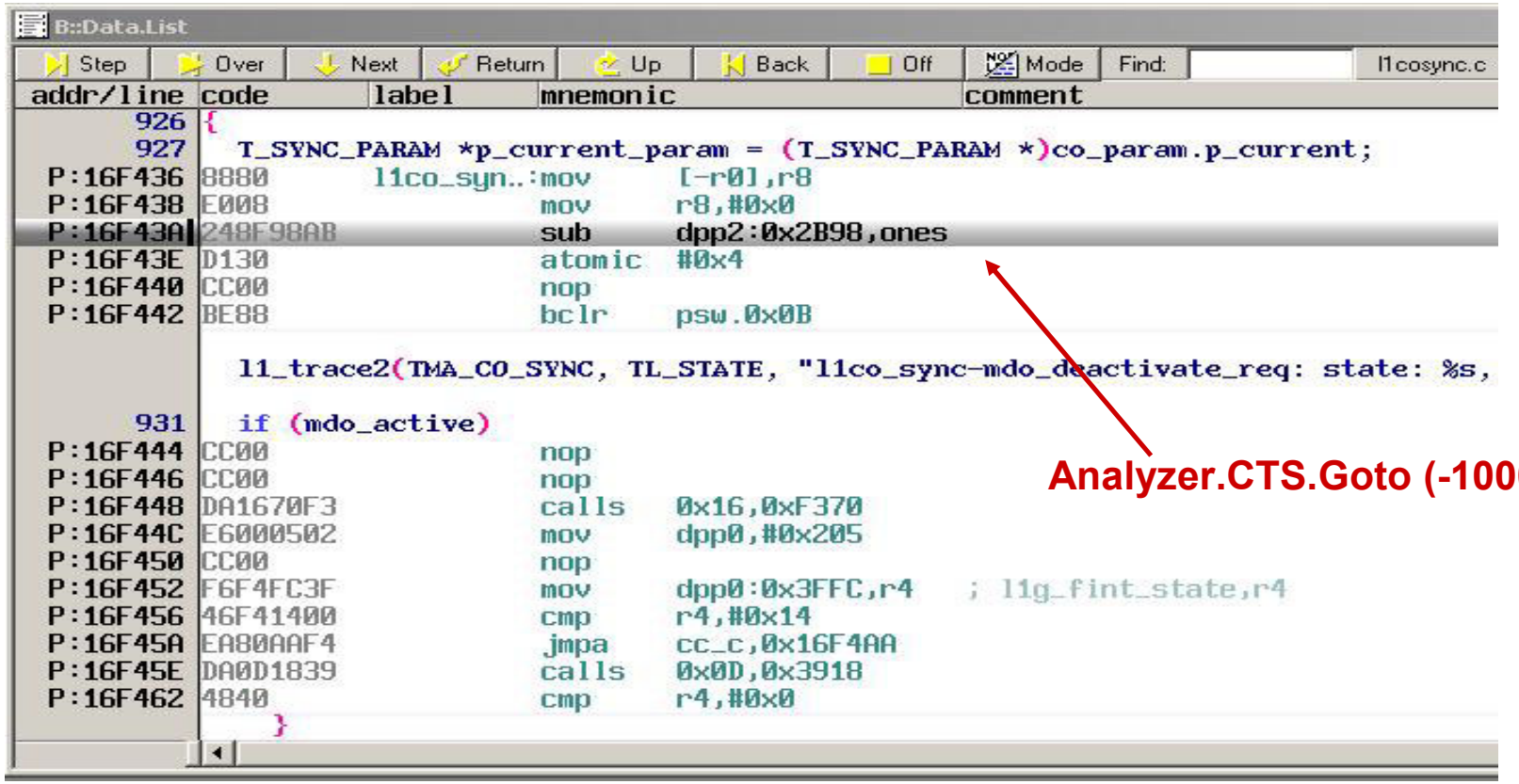
- All basic debugging commands, e.g. Single Step or Step-Over-Call can be executed in this history and variables be monitored in the same way like a "regular" debugging session.

Method Analyzer: CTS (2)



Method Analyzer: CTS (3)

- Views the context at the given record
  - Command Analyzer.CTS.Goto [<record>]



NEONSEVEN

Method Analyzer: Remote debug (1)

- To perform a remote debug all memory status, CPU register and history buffer shall be stored on file.
  - Save External RAM contents
    - Data.SAVE.Binary c:\userdata\photo_of_ram.bin 0x0800000--0x0bffffe /word
  - Save of PRAM
    - Data.SAVE.Binary c:\userdata\photo_of_pram.bin 0x018000--0x01fffe /word
  - Save SFR-ESFR-CPU Regs
    - Data.SAVE.Binary c:\userdata\photo_of_dram.bin 0x0d000--0x0fffe /word
  - Save history buffer
    - Winprint.Analyzer.List (startindex)--(endindex) [param]

NEONSEVEN

Method Analyzer: Remote debug (2)

- All stored files must be reloaded into the local Trace Analyzer to perform the remote debug
  - The steps are:

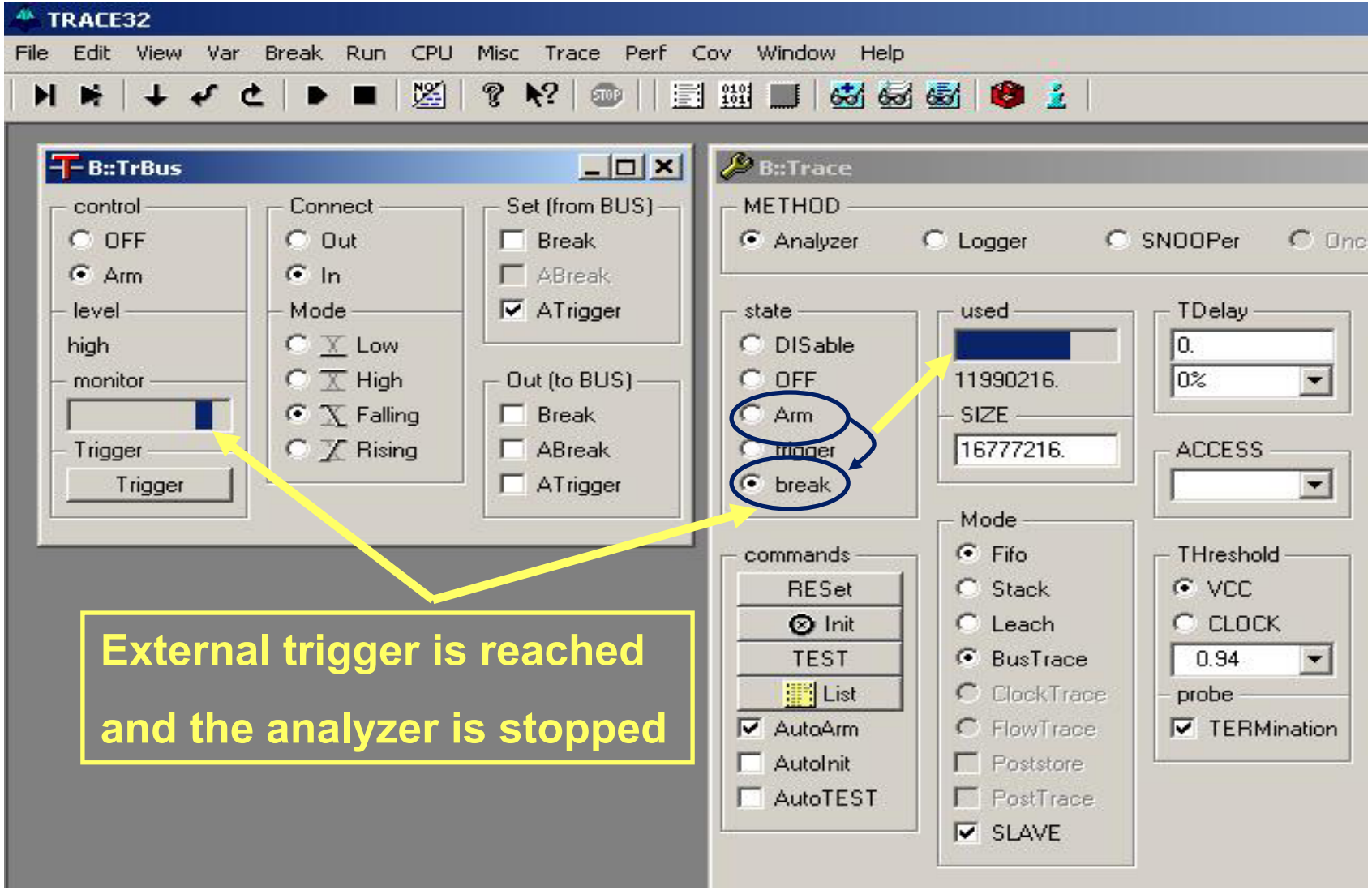
1. Load Cmm configuration file
2. Load External RAM contents
   - **Data.load.Binary c:\userdata\photo_of_ram.bin 0x0800000--0x0bffffe /word**
3. Load of PRAM
   - **Data.load.Binary c:\userdata\photo_of_pram.bin 0x018000--0x01fffe /word**
4. Load SFR-ESFR-CPU Regs
   - **Data.load.Binary c:\userdata\photo_of_dram.bin 0x0d000--0x0fffe /word**
5. Load abs file
   - **d.load.i absfile.abs /nocode**

NEONSEVEN

Method Analyzer: External trigger (1)

- An External trigger can be used to stop the Analyzer trace when the external event is reached
- Steps
  - Connect an external trigger to the TRIGGER connector of Lauterbach
  - Select Trigger BUS from Break Menu
  - Set Connect In for external input
  - Set Mode, it depends on type of external trigger
  - Set ATrigger
  - Arm the analyzer
  - When the external trigger is reached the analyzer is stopped. Push on list button to open the history buffer and to see the list of functions that are running when the trigger occur.

Method Analyzer: External trigger (2)



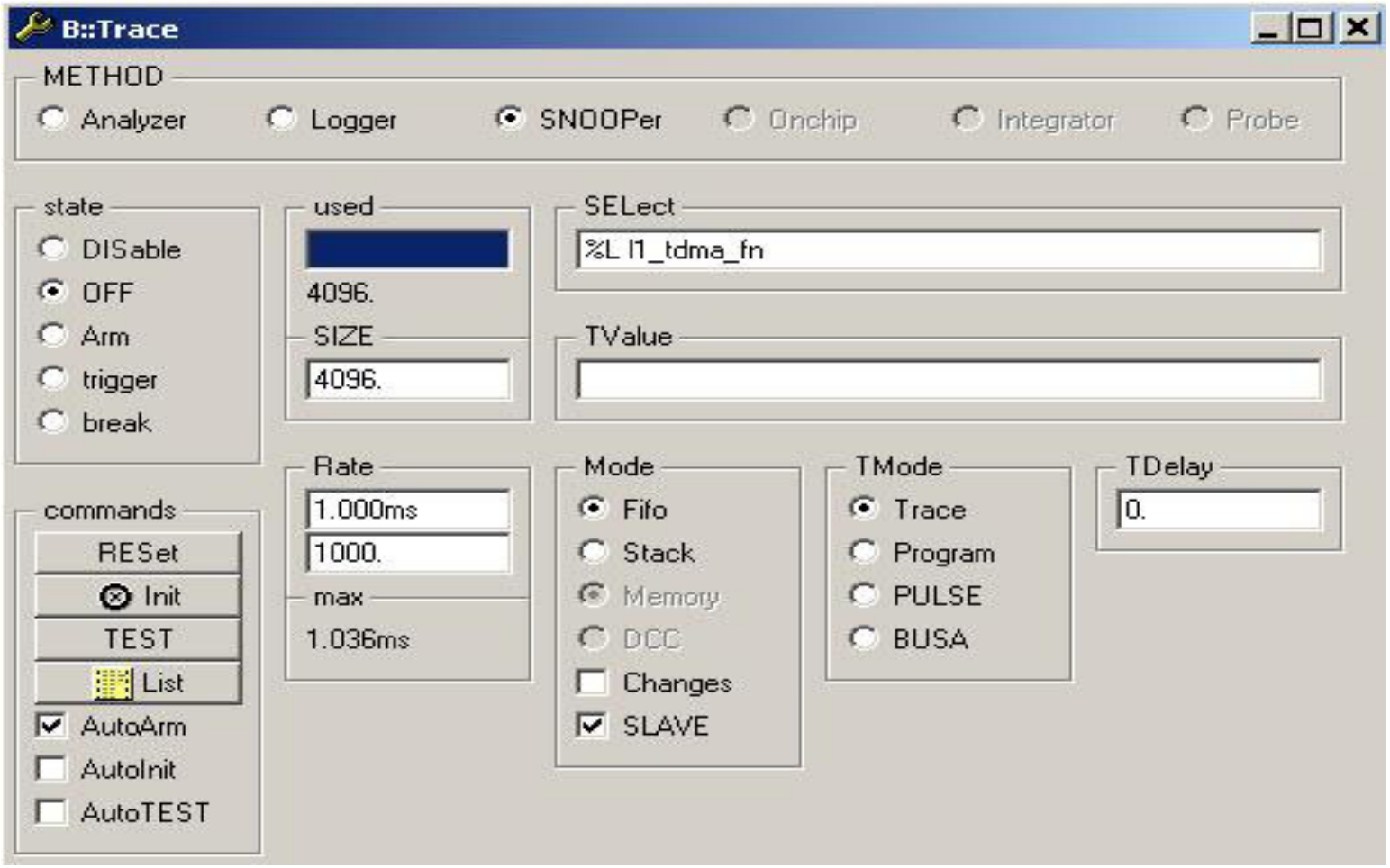
NEONSEVEN

Method Snooper

- The intention of the SNOOPer trace is to sample data information over the time
- The user define a sampling rate and the TRACE32 software reads out the requested information in this sampling rate while the CPU is executing the program and transfers the information to the SNOOPer trace
- Max. trace size: Up to 1 M frames
- Fastest sampling rate: 20 -100 us

Method Snooper: Example

Sample of the I1_tdma_fn variable: Rate 1ms, buffer size 4096.



Method Snooper: Example

Result of Snooper buffer

Value of l1d_tdma_fn

Rate 1ms

B::SNOOP.List

Setup... Goto... Find...

record	address	d.w	symbol	ti.back
-0000003996	D:81FFF4	7160	\\progsm\L1COF INT_C\l1_tdma_fn	999.400us
-0000003995	D:81FFF4	7160	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003994	D:81FFF4	7160	\\progsm\L1COF INT_C\l1_tdma_fn	1.002ms
-0000003993	D:81FFF4	7160	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003992	D:81FFF4	7161	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003991	D:81FFF4	7161	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003990	D:81FFF4	7161	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003989	D:81FFF4	7161	\\progsm\L1COF INT_C\l1_tdma_fn	999.440us
-0000003988	D:81FFF4	7161	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003987	D:81FFF4	7162	\\progsm\L1COF INT_C\l1_tdma_fn	999.420us
-0000003986	D:81FFF4	7162	\\progsm\L1COF INT_C\l1_tdma_fn	1.002ms
-0000003985	D:81FFF4	7162	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003984	D:81FFF4	7162	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003983	D:81FFF4	7162	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003982	D:81FFF4	7163	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003981	D:81FFF4	7163	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003980	D:81FFF4	7163	\\progsm\L1COF INT_C\l1_tdma_fn	1.002ms
-0000003979	D:81FFF4	7163	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003978	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	999.440us
-0000003977	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	1.001ms
-0000003976	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003975	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	999.440us
-0000003974	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003973	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003972	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003971	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	1.000ms
-0000003970	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	999.440us
-0000003969	D:81FFF4	7164	\\progsm\L1COF INT_C\l1_tdma_fn	1.002ms

NEONSEVEN

Power Trace debug using two boards

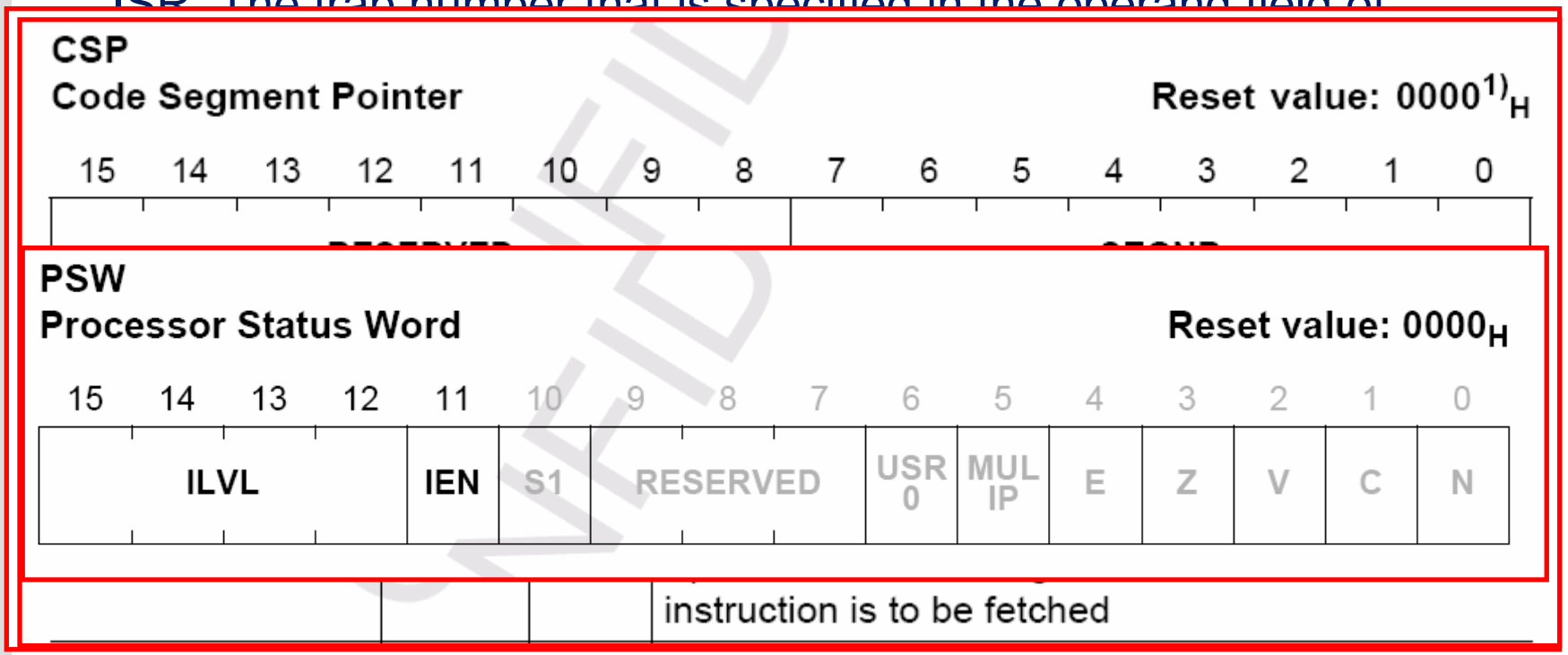
- Connection of the Power Trace to the DUT and the JTAG to another board
  - **Main reasons**
    1. Trap “Non-treatable” exceptions
    2. Trace DUT in power saving

“Treatable” exceptions

- To investigate the occurrence of a crash, an appropriate set of breakpoints shall be selected and used with the real-time debugger tool (Lauterbach).
- Once the system has halted, the tool will make some useful information available to the developer, i.e:
  - After an exception, **PSW**, **CSP** (in segmentation mode), and **IP** have been pushed into the system stack. PSW, Stack Pointer and Registers are accessible via CPU→CPU Registers
  - All chipset registers, available via menu/view/peripherals once the file <chipset>.per is made visible.
  - In case the .cmm file does not provide it by default, the system stack can be inspected by entering the following command:
  - d.v %SYMBOL.LONG register(sp) /TRACK

Software traps

- The TRAP instruction is used to cause a software call to an ISR. The trap number that is specified in the operand field of



CSP
Code Segment Pointer **Reset value: 0000 1)H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CSP							

PSW
Processor Status Word **Reset value: 0000H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ILVL				IEN	S1	RESERVED			USR 0	MUL IP	E	Z	V	C	N

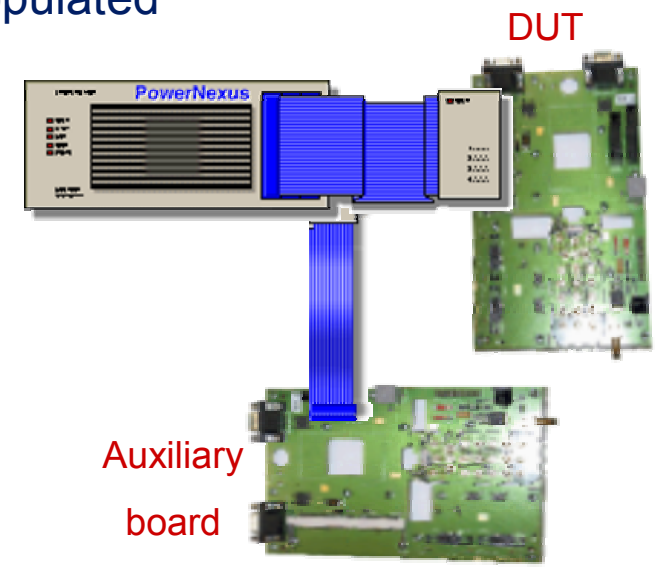
instruction is to be fetched

- **TFR** |= TRAP_NMI; /* Envoke the Non Maskable Interrupt Trap flag */
- }
- This function is called by ms_exit and ose_exception_handler.

Non-treatable exceptions: Connection lost

- When no BP is available for debugging, the DUT goes to "connection loss" state. In this case you have two options
  - Select "CPU→System Settings→No debug" and then "Trace→List→All" backtrace will be populated
  - Connect the Power Trace to the DUT and the JTAG to another board.

    When the DUT crashes, stop the auxiliary board and look to the backtrace. Please make sure you are using the .abs file of the version loaded on the DUT.

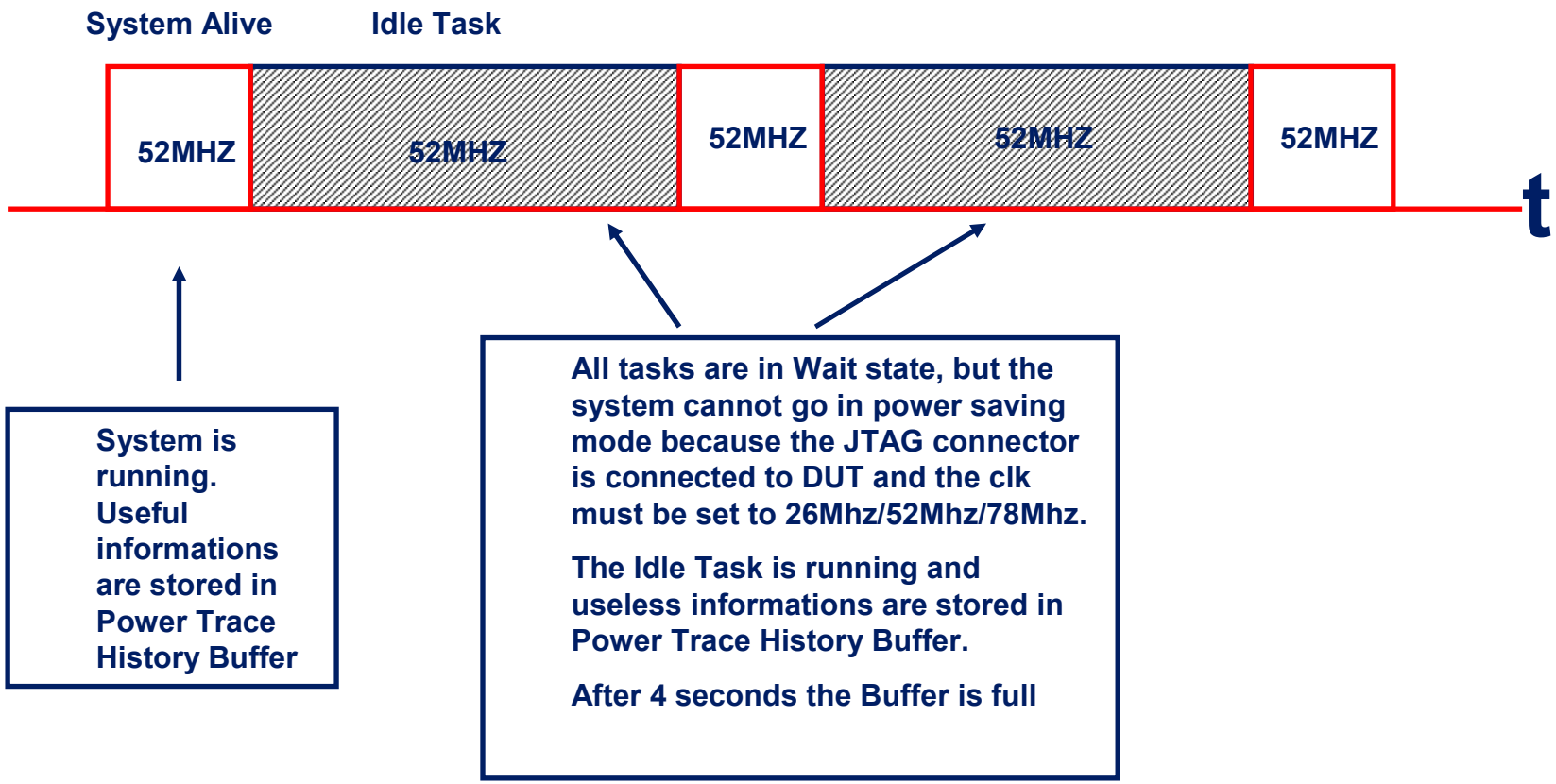


Trace DUT in power saving

- When JTAG Connector and Power Trace connected on same DUT:
  - When all tasks are in Wait state, the system cannot go in power saving mode because the JTAG connector is connected to DUT and the master clock must be set to 26Mhz/52Mhz/78Mhz otherwise the JTAC loss the connection to the DUT.
  - When all task are in wait state only the Idle Task is running and useless informations are stored in Power Trace History Buffer → There are a lot of call to Idle Task in the History
  - After 4 seconds the Buffer is complete full.
  - Is not possible to analyze the real behavior because DUT never go in power saving mode.

Trace DUT in power saving

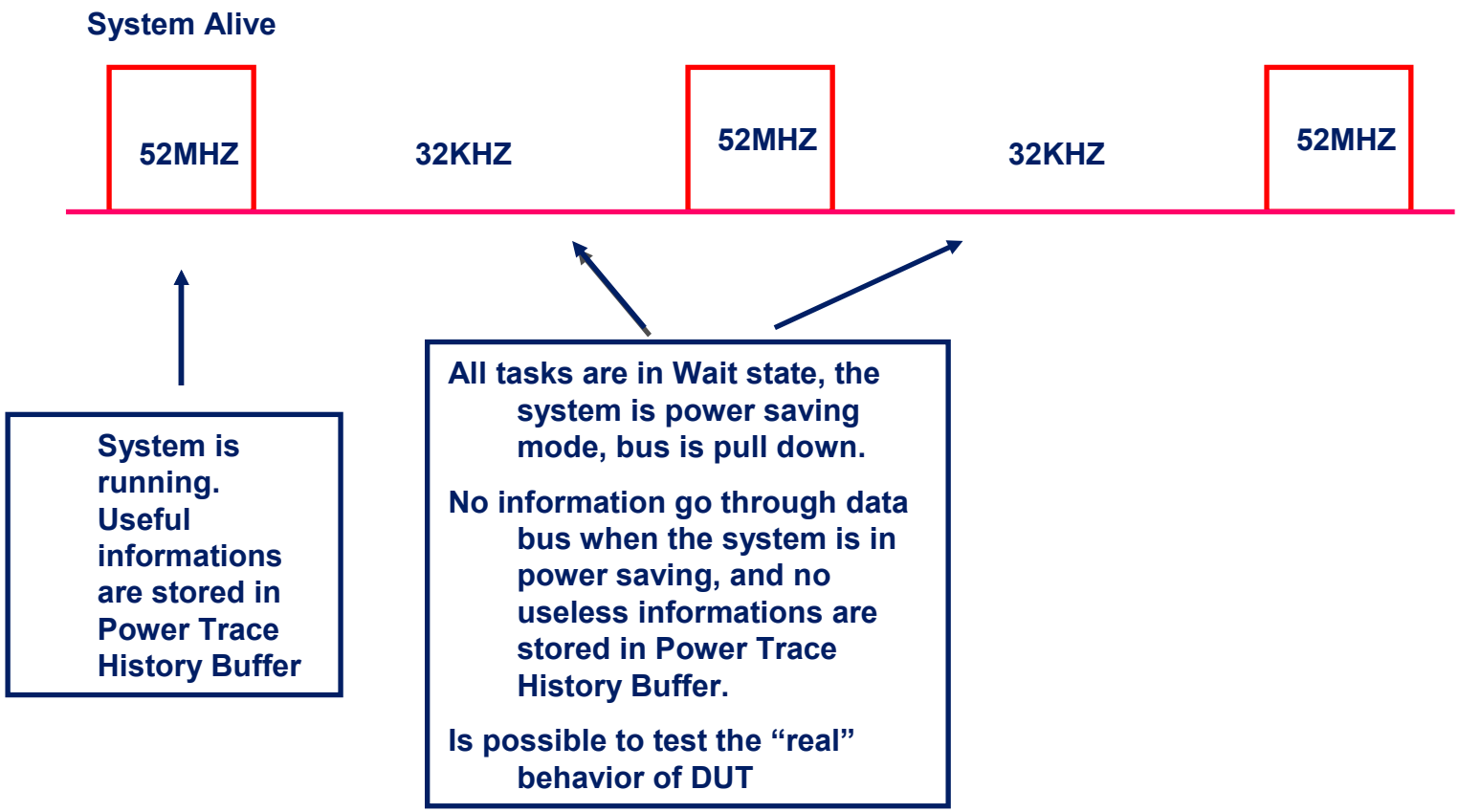
- JTAG Connector and Power Trace connected on same DUT:



Trace DUT in power saving

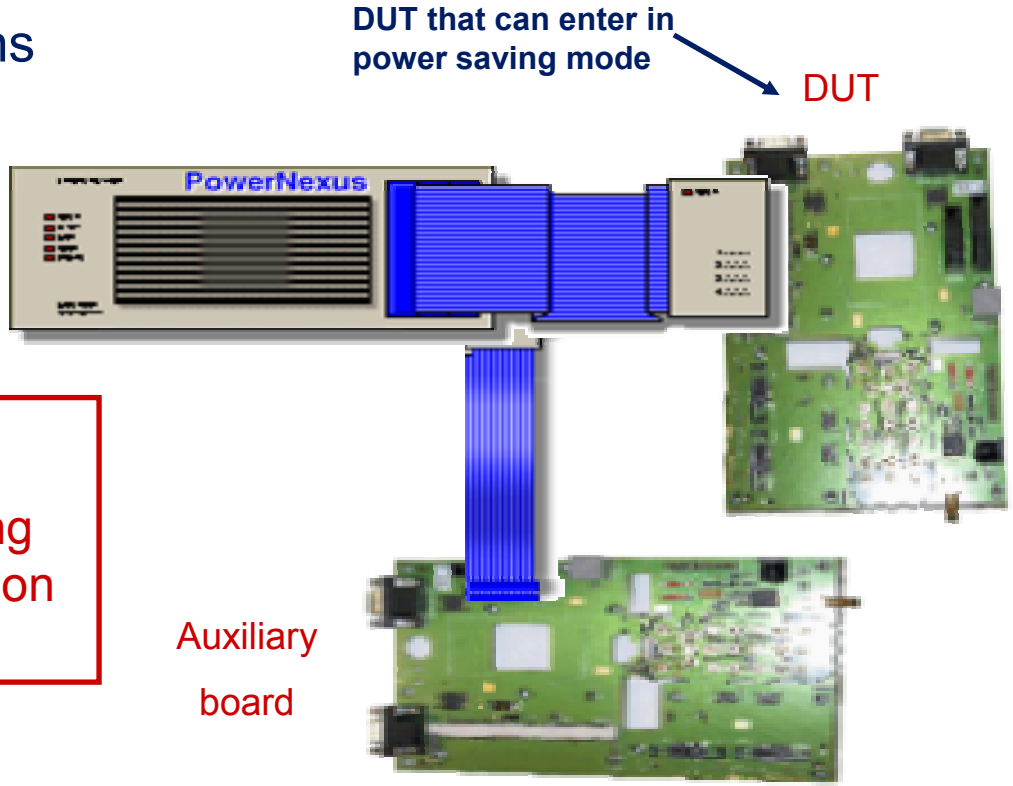
- By Connecting the Power Trace to the DUT and the JTAG to another boar
  - Is useful to trace DUT in power saving mode, because it prevents the JTAG connector from keeping the MS awaken and lengthens the temporal duration of the backtrace.
  - Only the useful information are stored in the History buffer. When the system is alive the data are stored, when the system go in power saving mode the bus is put in pull down, no task is running, no data through to the bus → no useless data stored.
  - Is possible to analyze about 1 minute of real life of DUT (the maximum time stored depend on of the status of DUT, and on the activities performed)

Trace DUT in power saving



Trace DUT in power saving

- Example of connections



IMPORTANT!!

Make sure you are using the .abs file of the version loaded on the DUT