Trace :-

The command Trace is a general command for trace configuration and trace display. It is available for all kind of trace methods provided by TRACE32. The following trace methods are available:

- Analyzer
- ART(Advanced Register Trace)
- FDX (Fast Data eXchange)
- Integrator
- LA
- LOGGER
- Onchip
- Probe
- PORT
- RTS
- SNOOPer

How does TRACE32 determine the default trace method?

- If the hardware module connected to your target board is a PowerTrace, then the Analyzer trace method becomes the default setting.

- If a hardware module other than a PowerTrace is connected to your target board, TRACE32 adjusts the trace method accordingly.

- if the chip has an onchip trace sink, then the **Onchip trace method** become the default setting . If the chip does not have an **trace sink** , then the **ART** trace method becomes the default setting .

-if TRACE32 runs in software -only mode as an instruction set simulator , then it is again the Analyzer trace method that becomes the default setting .

All Trace commands refer to the selected trace method.

Trace method SNOOPer :-

```
Trace.state    ; select the trace method SNOOPer for recording data

Trace.METHOD SNOOPer      ; trace data
                          ; <configuration>
                          ; <trace data is recorded using the commands GO, WAIT, Break>

Trace.List    ; Display the trace data recorded with SNOOPer as a trace listing

SNOOPer.List      ; this is the equivalent and explicit command.
```

Troubleshooting:-

1.Trace information should be analyzed while the program execution is running and the debugger has no run-time access to the target memory to read the program

code .
NOACCESS in a trace display window indicate that the debugger can not read the target memory.

You can overcome this problem by loading the program code to the TRACE32 virtual memory.

```
; load the program code additional to the TRACE32 virtual memory
; whenever you load it to the target memory
Data.LOAD.Elf diabc.x /PlusVM
```

2. Reading the target via **JTAG** is very slow therefore all trace display and analysis windows are slow.

You can overcome this problem by loading the program code to the **TRACE32 virtual memory** and by specifying **Trace.ACCESS AutoVM.**

```
; load the program code additional to the TRACE32 virtual memory
; whenever you load it to the target memory
Data.LOAD.Elf diabc.x /PlusVM
; advise TRACE32 to read the target code from the virtual memory
; if no code is loaded to the virtual memory for a program address
; TRACE32 will read the code by using the best practce procedure
Trace.ACCESS AutoVM
```

3.Trace information should be inspected , but there is no source code available.

You can overcome this problem by specifying **Trace.ACCESS Denied** to advise TRACE32 not to merge source code information . The **Trace.List** window will list the available program addresses and mark all cycles as unkown.