[ARC] Why does PowerView open a TERM.GATE window automatically?

2022-10-05 - Comments (0) - ARC

Since TRACE32 build 18086, PowerView opens a **TERM.GATE** window automatically when your target application would hang otherwise.

This might happen by mistake when you've created a symbol named __HOSTLINK__ by your own.

By default, the MetaWare® C/C++ Compiler adds the MetaWare hostlink library. This hostlink library enables your target application to communicate with the host PC running the debugger for e.g. doing a "printf".

When a target application is compiled with the MetaWare hostlink library, it will always go in a polling-loop before going to the main() routine to fetch argc/argv from the host PC. Without a debugger interacting with the target memory, the target application would hang in this loop forever.

To support the MetaWare hostlink library, you have to execute the following two command in PowerView (after loading your ELF file):

```
TERM.METHOD METAWARE TERM.GATE
```

When starting your target application with the TRACE32 Go command, PowerView checks if your ELF file contains the magic symbol called __HOSTLINK__ , which usually indicates that you are using the MetaWare hostlink library. If this magic symbol is there but a suitable TERM.GATE-window is not yet open, PowerView opens the window automatically.

Thus, PowerView might have opened TERM.GATE when you are not using the hostlink library but has created a symbol named __HOSTLINK__ by your own.

What can you do to avoid an automatically popping up TERM.GATE-window?

- Open the window on purpose after loading your ELF file by using the commands TERM.METHOD METAWARE and **TERM.GATE**.
- Remove the MetaWare hostlink library from your target application by linking it with

the compiler switch "-Hhostlib=" with nothing after the equals sign (=).

- Rename symbol __HOSTLINK__ in your target application.
- Rename symbol __HOSTLINK__ in the debugger after loading the ELF file with e.g.
  sYmbol.Modify.NAME __HOSTLINK__ __HOSTLINK-OFF__

Please also refer to the **"MetaWare® C/C++ Programmer’s Guide for ARC®"**