LAUTERBACH
DEVELOPMENT TOOLS

Integration for Rhapsody in MicroC

MANUAL

Integration for Rhapsody in MicroC

Integration for Rhapsody in MicroC

Version 06-Jun-2024

Overview

This interface serves as a GBA (Graphical Back Animation) driver for the Rhapsody in MicroC / Statemate CASE tool from I-Logix. The data for the GBA is read from the target by the TRACE32 debugger.

Current supported Rhapsody in MicroC / Statemate versions are:
Rhapsody in MicroC Version 3.1,
Statemate Version 4.2

Hosts: Windows

Brief Overview of Documents for New Users

Architecture-independent information:

- **"Training Basic Debugging"** (training_debugger.pdf): Get familiar with the basic features of a TRACE32 debugger.
- **"T32Start"** (app_t32start.pdf): T32Start assists you in starting TRACE32 PowerView instances for different configurations of the debugger. T32Start is only available for Windows.
- **"General Commands"** (general_ref_<*x*>.pdf): Alphabetic list of debug commands.

Architecture-specific information:

- **"Processor Architecture Manuals"**: These manuals describe commands that are specific for the processor architecture supported by your debug cable. To access the manual for your processor architecture, proceed as follows:
  - Choose **Help** menu > **Processor Architecture Manual**.
- **"OS Awareness Manuals"** (rtos_<*os*>.pdf): TRACE32 PowerView can be extended for operating system-aware debugging. The appropriate OS Awareness manual informs you how to enable the OS-aware debugging.

Operation Theory

The interface consists of a helper application (“rimct32.exe)” that connects to TRACE32 and the Rhapsody GBA server. For using the Graphical Back Animation, Rhapsody generates requests that are transferred to the helper application. This application converts the requests and sends them to the TRACE32 display driver via sockets. The display driver transports the requests to the TRACE32, which, in turn, communicates with the target as requested. The answer goes exactly the same way backwards.

Rhapsody in MicroC --> GBA server --> rimct32.exe --> t32mxx.exe --> TRACE32 --> target

NOTE:	This integration uses internally the **TRACE32 Remote API**. The Remote API has **restrictions** if TRACE32 runs in demo mode. Please see there for further details.

Installation

Follow these steps to install the Rhapsody in MicroC / Statemate GBA driver for TRACE32:

1. Locate the file "~~/demo/env/rhapsody/microc/rimc31/rimct32.exe" in your TRACE32 installation directory.

2. Use the “amc_communication_dll.dll” of your Rhapsody / Statemate installation. Find this file in the directory “I-Logix\stmm\<*version*>\bin\”.

3. Edit your "config.t32" file: Add between two empty lines (!):

```
RCL=NETASSIST
PORT=20000
PACKLEN=1024
```

   Generate your application with GBA functionality, according to the Statemate manuals.

Command Line Parameters

rimct32.exe supports some command line parameters to customize its behavior. Please note that there must be no space in between the parameters.

-rhpn*<host>*	Host name of the Statemate GBA server default: -rhpnlocalhost
-rhpp*<port>*	Port of the Statemate GBA server default: -rhpp6666
-t32n*<host>*	Host name of the TRACE32 debugger software default: -t32nlocalhost
-t32p*<port>*	Port of the TRACE32 debugger software default: -t32p20000 (see config.t32)
-sync	Break if new data available on the target, instead of permanently read information default: read information permanently Use “-sync” if you want to watch every state change, or if the state changes are slower than the read cycle (see -timer).
-runmode	Don’t stop for memory reads default: stop target for memory reads (state checks) Use this option, if your TRACE32 system provides dual port access to the memory of the GBA variables (“gba*”).
-timer	Timer cycle rate to check target for new state information in milliseconds default: -timer200 Increase this value, if you have a slow target connection, or if you’re using the Simulator.

Startup Sequence

rimct32.exe serves as a background task (“demon”). You may once start it and let it run, as long as you use GBA. You don’t need to restart it, if you restart your application, TRACE32 or Rhapsody.

For the first try on the TRACE32 GBA driver, follow the steps below. If you once got experienced, you can customize your own startup sequence.
See also the example: ~~/demo/env/rhapsody/microc/rimc31/rimc.cmm

1. Switch on power on TRACE32 and your target.
2. Start rimct32.exe.
3. Start the TRACE32 debugger.
4. Start Rhapsody in MicroC / Statemate.
5. Open your project.
6. Start the GBA Server.
7. Open the work area browser and your chart.
8. Start your application in TRACE32.

Working with the TRACE32 GBA Driver

There are several prerequisites to work with the TRACE32 GBA driver.

- The application, of course, must include the GBA functionality.
- TRACE32 must have access to the file “indirect_gba.c” of your project.
  (More precisely: TRACE32 must be able to locate the source of the function “write_socket”, i.e. the search paths must be set correct.)
- The file “gba_data.gba” (generated by Rhapsody) must reside in the same directory as the file “indirect_gba.c”.
  (More precisely: it must reside in the same directory as the source file of the function “write_socket”).

The files “indirect_gba.c” and “gba_data.gba” provided in the ~~/demo directory are only example files and must not be used with your application.

Depending on the mode the driver is running, it may set some breakpoints on the target (especially on the function “write_socket”). Those breakpoints are managed by the driver, setting or clearing them while the GBA is running will result in unpredictable behavior.