Run Mode Debugging Manual Symbian

Run Mode Debugging Manual Symbian

Version 18-Nov-2020

History

28-Aug-18	The title of the manual was changed from “RTOS Debugger for *<x>* - Run Mode” to “Run Mode Debugging Manual *<x>*”.

Basic Concepts

For Integrated Run & Stop Mode Debugging, Stop Mode Debugging via the JTAG interface is extended by a DCC communication interface between TRACE32 and the debug agent.

TRK Monitor

The TRK application is the debug agent on the target.

The Debug Communications Channel (DCC)

The JTAG interface of the ARM architecture includes a so-called Debug Communications Channel (DCC). Information exchange via DCC is possible between

- the TRACE32 software running on the host
- and the TRK application running as symbian process on the target

For details about DCC refer to your **ARM Technical Reference Manual**.

Switching to Run Mode Debugging

After TRACE32 was started and configured for Stop Mode Debugging switching to Run Mode debugging is performed as follows:

```
SYStem.MemAccess TrkMON

Go.MONitor
```

SYStem.MemAccess TrkMON	Select TRK as monitor
Go.MONitor	Switch to Run Mode Debugging

After the communication is configured, debugging can be performed completely via the TRACE32 PowerView GUI.

To debug a process proceed as follows:

1. **Check if the process is already running.**

TASK.List.tasks	List all running processes

```
TASK.List.tasks
```

2. **Load the process for debugging or select a process from the list.**

TASK.RUN *<process>*	Load a process
TASK.SELect *<id>*	Select a process from the list

If the process is not running, the command **TASK.RUN** can be used to load the process for debugging. **TASK.SELect** selects a running process from the list.

3. **Load the symbol and debug information for the process.**

Data.LOAD.*<file_format> <file>* **/NoCODE /NoClear /NOREG**

NoCODE - load only symbol information.

NoClear - obtain the symbol information loaded for other processes.

NOREG - avoid any unintended change to the CPU registers.

Quick Start Example

Integrated Run & Stop Mode Debugging requires that **Stop Mode Debugging** is working properly before Run Mode Debugging can be activated. The following commands represent a basic TRACE32 setup for Stop Mode Debugging. It is assumed that the target setup and symbian booting is performed by the code in the boot FLASH.

```
SYStem.CPU ARM920                       ; Select the target CPU

TrOnchip.Set DABORT OFF                 ; Debug mode is not entered at data
                                        ; abort exception.

TrOnchip.Set PABORT OFF                 ; Debug mode is not entered at
                                        ; prefetch abort exception.

TrOnchip.Set UNDEF OFF                  ; Debug mode is not entered at an
                                        ; UNDEF instruction.

SYStem.Mode Attach                      ; Establish the communication
                                        ; between the debugger and the CPU

TASK.CONFIG symbian2                    ; Activate OS Awareness and
                                        ; menu

Data.Load.SYMBIAN symbian.symbol        ; Load the kernel symbols into the
                                        ; debugger

TRANSlation.ON                          ; Switch debugger MMU to ON
```

The target setup and the preparations for Linux debugging might be more complex for your system. It is strongly recommanded to refer to **"ARM Debugger"** (debugger_arm.pdf) and for details.

To configure **Run Mode Debugging**:

1. Select TrkMON as monitor:

```
SYStem.MemAccess TrkMON
```

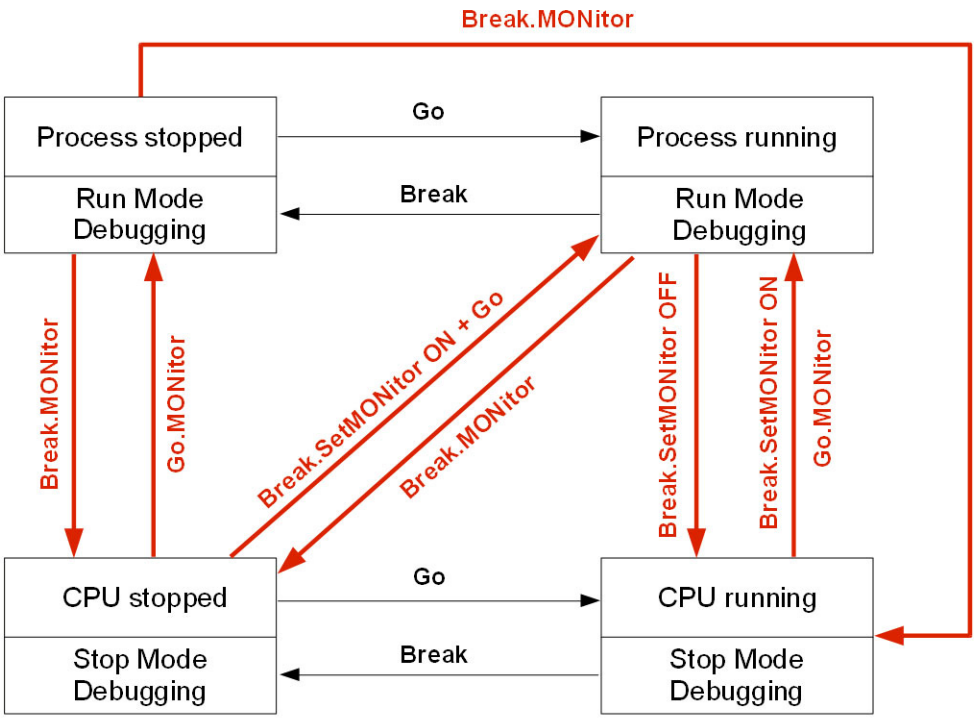
2. Start the TRK application on the target.
3. Then continue to enter the following command in TRACE32:

```
Go.MONitor                  ; Switch to Run Mode Debugging
```

or

```
Break.SetMONitor ON         ; Switch to Run Mode Debugging
```

Switching between Run & Stop Mode



The graphic above shows a simple schema of the switching between Run Mode Debugging and Stop Mode Debugging.

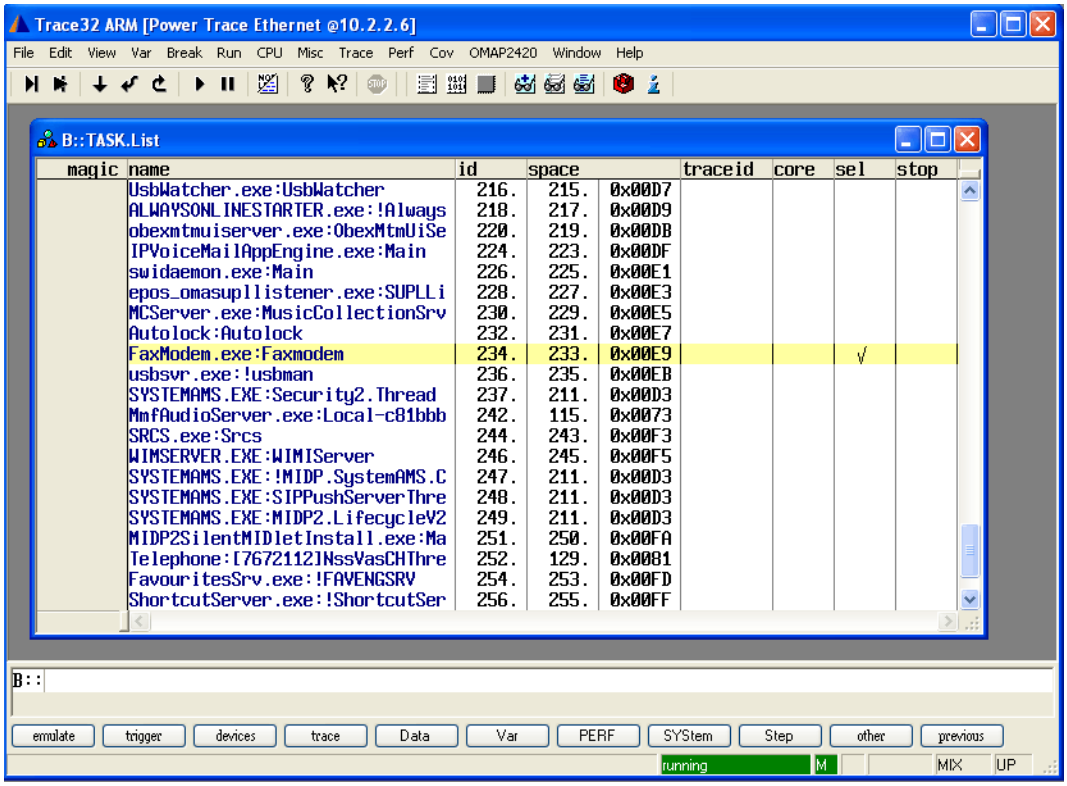
The following commands are used to switch between Run & Stop Mode Debugging:

Break.MONitor	Switch to Stop Mode Debugging and stop the program execution on CPU.
Break.SetMONitor OFF	Switch to Stop Mode Debugging. If the selected process was running or no process was selected, the CPU stays running in Stop Mode. If the selected process was stopped, the CPU is stopped in Stop Mode.
Break.SetMONitor ON	Switch to Run Mode Debugging. If the CPU is stopped, the run mode is active with the next **Go**.
Go.MONitor	If the CPU is stopped, the program execution is started. Switch to Run Mode Debugging.

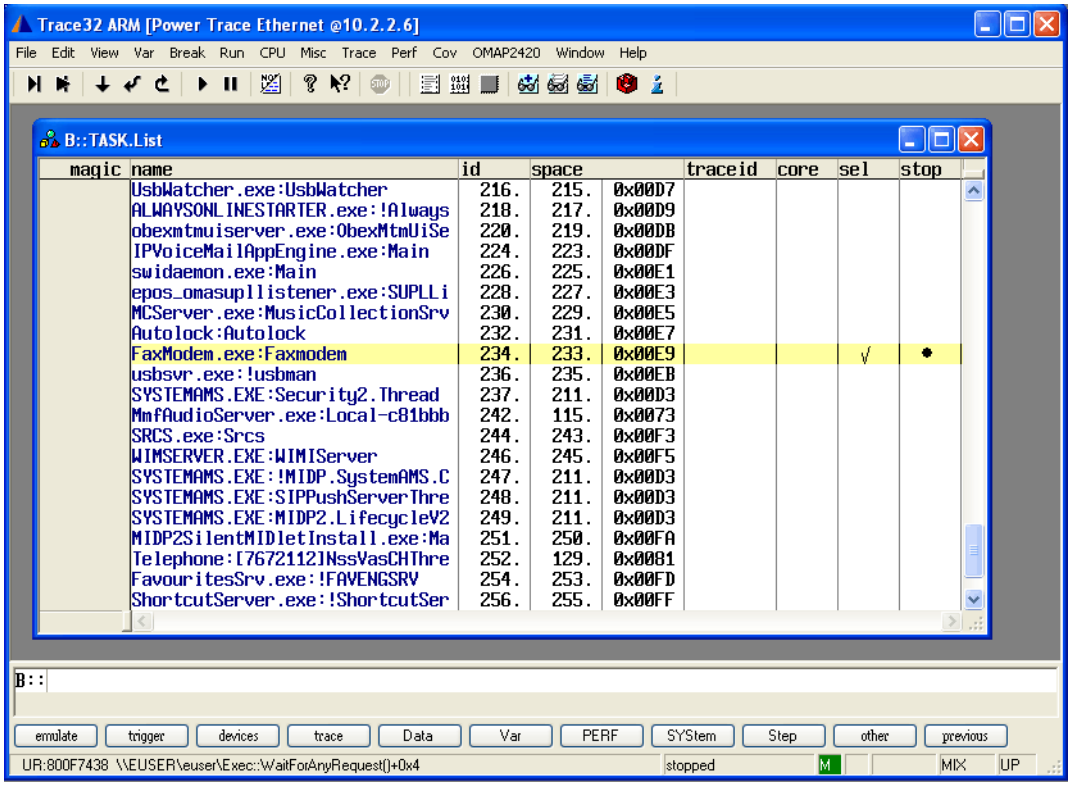
If Run Mode Debugging is active, a **green M** is displayed in the state line of TRACE32.

The following states are possible in Run Mode Debugging:

1. Run Mode Debugging active (**green M**), selected process running



2. Run Mode Debugging active (**green M**), selected process stopped.



Commands for Run Mode Debugging

TASK.COPYDOWN<*src*> <*dest*>	Copy a file from the host into the target
TASK.COPYUP<*src*> <*dest*>	Copy a file from the target into the host
TASK.INSTALL <*drive*>	
TASK.KILL <*id*>	Kill the process
TASK.List.tasks	List all running processes
TASK.RUN <*process*>	Load a process for debugging
TASK.SELect <*id*>	Select a process for debugging

Breakpoint Conventions

For Integrated Run & Stop Mode Debugging please keep the following breakpoint convention:

- Use on-chip breakpoints only for Stop Mode Debugging

  If an on-chip breakpoint is hit in Run Mode Debugging, the CPU is stopped in Stop Mode debugging.

- Use software breakpoints only for Run Mode Debugging

Frequently-Asked Questions

TASK.CONFIG symbian Errors Ref: 0257	**Why does the debugger report errors when executing 'TASK.CONFIG symbian'?** This command already checks the target system for the availability of necessary debug information. Symbian OS has to be up and the debugger in stopped mode to get the information. Also, the Symbian internal debug hooks have to be available, and the symbols of ekern.exe have to be loaded. check chapter "Hooks and Internals" in the Symbian OS Awareness Manual.