RTA-OSEK for PC

Getting Started Guide

Contact Details

ETAS Group

www.etasgroup.com

Germany

ETAS GmbH
Borsigstraße 14
70469 Stuttgart

Tel.:+49 (711) 8 96 61-102
Fax:+49 (711) 8 96 61-106

www.etas.de

USA

ETAS Inc.
3021 Miller Road
Ann Arbor, MI 48103

Tel.: +1 (888) ETAS INC
Fax: +1 (734) 997-94 49

www.etasinc.com

Japan

ETAS K.K.
Queen's Tower C-17F,
2-3-5, Minatomirai, Nishi-ku,
Yokohama, Kanagawa
220-6217 Japan

Tel.: +81 (45) 222-0900
Fax: +81 (45) 222-0956

www.etas.co.jp

France

ETAS S.A.S.
1, place des États-Unis
SILIC 307
94588 Rungis Cedex

Tel.: +33 (1) 56 70 00 50
Fax: +33 (1) 56 70 00 51

www.etas.fr

Korea

ETAS Korea Co., Ltd.
4F, 705 Bldg. 70-5
Yangjae-dong, Seocho-gu
Seoul 137-899, Korea

Tel.: +82 (2) 57 47-016
Fax: +82 (2) 57 47-120

www.etas.co.kr

Great Britain

ETAS UK Ltd.
Studio 3, Waterside Court
Third Avenue, Centrum 100
Burton-upon-Trent
Staffordshire DE14 2WQ

Tel.: +44 (0) 1283 - 54 65 12
Fax: +44 (0) 1283 - 54 87 67

www.etas-uk.net

Copyright Notice

Version: RTA-OSEK for PC v5.0.0 (issue 1)

Disclaimer

The information in this document is subject to change without notice and does not represent a commitment on any part of LiveDevices. While the information contained herein is assumed to be accurate, LiveDevices assumes no responsibility for any errors or omissions.

In no event shall LiveDevices, its employees, its contractors or the authors of this document be liable for special, direct, indirect, or consequential damage, losses, costs, charges, claims, demands, claim for lost profits, fees or expenses of any nature or kind.

Trademarks

RTA-OSEK and LiveDevices are trademarks of LiveDevices Ltd.

Windows and MS-DOS are trademarks of Microsoft Corp.

OSEK/VDX is a trademark of Siemens AG.

The XL Driver Library is owned by Vector Informatik GmbH.

All other product names are trademarks or registered trademarks of their respective owners.

Certain aspects of the technology described in this guide are the subject of the following patent applications:

UK - 0209479.5 and USA - 10/146,654,

UK - 0209800.2 and USA - 10/146,239,

UK - 0219936.2 and USA - 10/242,482.

Table of Contents

1 About this Guide

RTA-OSEK for PC consists of tools and libraries that enable the creation of Windows-hosted applications that emulate the behavior of applications running on microcontroller hardware – including the ability to run automotive-style OSEK and AUTOSAR applications. This guide describes the tools and libraries used to create and monitor such applications.

RTA-OSEK for PC includes a Windows port of the popular RTA-OSEK kernel. You should consult the documents *RTA-OSEK User Guide* and *RTA-OSEK Reference Guide* for general information on how to use RTA-OSEK.

This guide describes how to get started with *RTA-OSEK for PC*. It covers installation of tools, how to build and run the example applications and where to go next.

1.1 Who Should Read this Guide?

It is assumed that you are a developer who wants to know how to create and monitor OSEK or AUTOSAR applications on Windows PCs.

1.2 Conventions

Important: Notes that appear like this contain important information that you need to be aware of. Make sure that you read them carefully and that you follow any instructions that you are given.

In this guide you'll see that program code, header file names, C/C++ type names, C/C++ functions and API call names all appear in the `courier` typeface.

2 Overview

RTA-OSEK for PC is a complete environment for developing OSEK and AUTOSAR applications. Mostly you'll be using it to prototype a new application before migrating it on to the production hardware, but you will also find that it is a good tool for training your engineers in developing applications for embedded targets.

But you needn't stop there. Because *RTA-OSEK for PC* is a complete and fast implementation of OSEK, you can use CAN to add inter-application communication. You can write applications that sit on your CAN network as test or simulation units. You can remotely monitor the state and progress of your applications using the *RTA-OSEK for PC* monitor, *RTA-TRACE* or your PC-based debugger. And of course the development turnaround time is tiny – just recompile and run; no downloading of hex files to an emulator, no programming Flash.

2.1 Terms

This section introduces some terminology that you will need to understand the rest of this guide. The complete set of terminology is explained in the "RTA-OSEK for PC User Guide".

RTA-OSEK for PC is typically used in automotive environments where the term *ECU* (Electronic Control Unit) is commonly used to refer to the target hardware on which the application runs. The ECU can be considered as a black box with inputs and outputs that performs a specific set of functions.

Other than the PC that you run it on, an application built under *RTA-OSEK for PC* does not need any real hardware. Instead, you create a *Virtual ECU (VECU)* in software that simulates the real-life devices such as switches or sensors that will be present in your ECU. These devices are built around a core *Virtual Machine (VM)* that provides services such as the interrupt controller, application control and diagnostic links.

Within this document we will use the terms VM and VECU extensively. Remember that *VM* represents the 'core' of the simulated hardware, and that *VECU* is the whole 'black-box'.

We also introduce the term *VRTA* as the short form of (Virtual) RTA-OSEK for PC. This short form is used to prefix executables such as `vrtaMonitor.exe`, `vrtaVM.dll` and many of the supplied source files.

The term `<rta>` is used for the root of the RTA-OSEK installation. Normally this would be `c:\rta`.

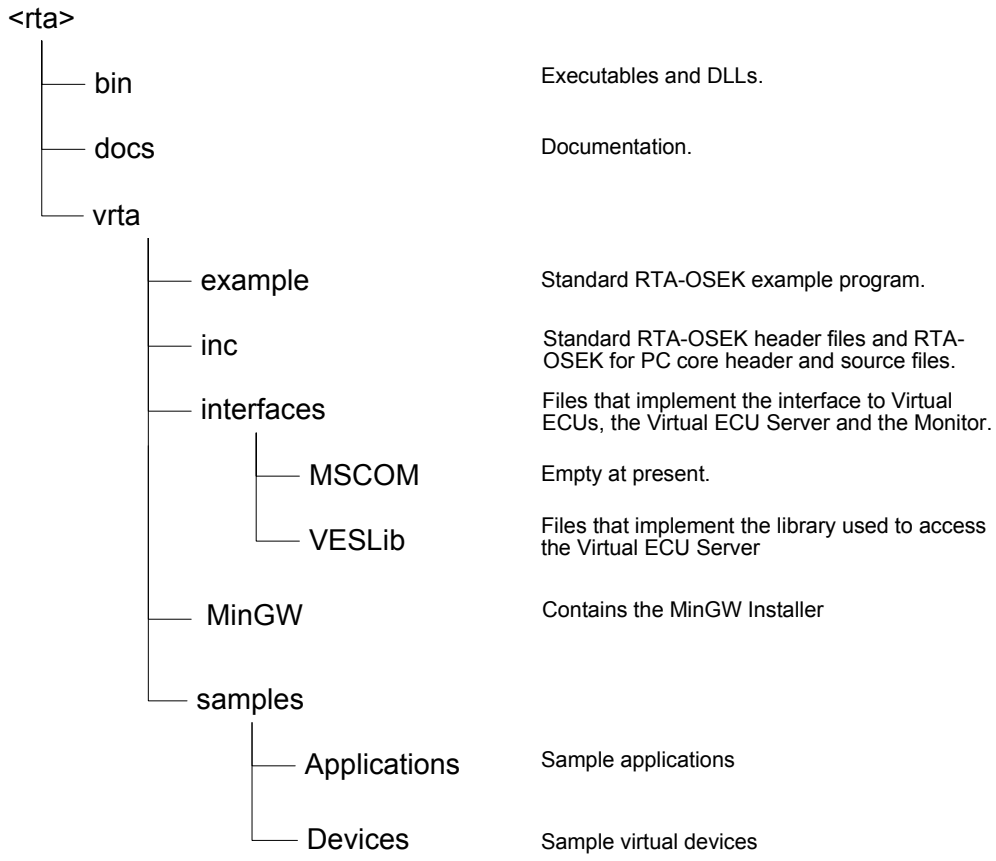
3 Installation

RTA-OSEK for PC is an RTA-OSEK target and should be installed **after** you have installed the RTA-OSEK tools CD and optionally the RTA-TRACE CD. Once you have installed these you should install *RTA-OSEK for PC* by inserting the *RTA-OSEK for PC* CD into your PC's CD drive. Normally the installer will run automatically. If you have auto-run disabled, run the `setup.exe` program from the root of the *RTA-OSEK for PC* CD.

Important: you will need to have Administrator privileges on the PC to install *RTA-OSEK for PC* (since `vrtaServer.exe` is installed as a Windows service). If you are unsure about this please contact your system administrator.

3.1 What is Installed

After installation you will have the following directory tree on your PC.



3.1.1 The Executable Files

The directory `<rta>\bin` contains the standard RTA-OSEK executable files as well as *RTA-OSEK for PC* specific executables and DLLs. The *RTA-OSEK for PC* specific files are:

`vrtaServer.exe`	The Virtual ECU server.
`vrtaMonitor.exe`	The Virtual ECU monitor.
`vrtaVM.dll`	The base Virtual Machine (no OS).
`vrtaVMxxx.dll`	The Virtual Machines that contain the different RTA-OSEK builds. "xxx" is the RTA-OSEK build type.
`VesLib.dll`	The Virtual ECU server interface library compiled as a DLL.
`vrtaMSCOM.dll`	The COM bridge.
`rtcVRTAlink.dll`	The RTA-TRACE communications DLL for *RTA-OSEK for PC*.

A standard RTA-OSEK port includes a set of library files containing the RTA-OSEK kernel and the RTA-TRACE target implementation. In the *RTA-OSEK for PC* port these libraries are bound into the `vrtaVMxxx.dll` DLLs that contain the Virtual Machine. For example, the RTA-OSEK standard build kernel without RTA-TRACE support library is bound into `vrtaVMs.dll`. The Virtual ECU start-up code (provided) automatically loads the correct DLL based on the RTA-OSEK build type.

3.1.2 Source and Header Files

The directory `<rta>\vrta\inc` contains the standard RTA-OSEK header files plus some additional header and source files for use with *RTA-OSEK for PC*. The *RTA-OSEK for PC* specific files start with a "`vrta`" prefix.

The directory tree starting at `<rta>\vrta\interfaces` contains header and source files that are used to provide interfaces to Virtual ECUs and the Virtual ECU server.

3.1.3 Sample Code

The directory tree starting at `<rta>\vrta\samples\Applications` contains sample applications as follows. These applications are described in section 4.

`BuildTemplates`	The `BuildTool.exe` utility and templates used to build some of the sample applications.
`NonOSEK`	An example that makes use of the standard sample virtual devices.
`RTA-OSEK Example 1`	An example that uses RTA-OSEK.
`RTA-OSEK Example 2`	An example that uses RTA-OSEK.

The directory tree starting at `<rta>\vrta\samples\Devices` contains sample virtual devices as below.

`Logger`	The source for a logging virtual device.
`Message`	The source for a messaging virtual device.
`Standard`	The source for standard virtual devices (timers etc).
`Test`	The source for a test virtual device.

3.2 Licensing

Before you can build and run anything with *RTA-OSEK for PC* you will need to obtain a license. Please contact your ETAS sales office to obtain the relevant license.

The RTA-OSEK license provided should be installed as described in section 4.1 "Licensing" of the "RTA-OSEK Getting Started Guide" (which can be found in `<rta>\docs\RTA-OSEK Getting Started Guide.pdf`).

3.3 Compiler Configuration

RTA-OSEK for PC has been designed to work with most PC C/C++ compilers and the `toolinit.bat` file (see section 3.4) is pre-configured to work with several such compilers. These are:

- MinGW / gcc
- Microsoft Visual C++ 5.0
- Microsoft Visual Studio 2003
- Borland C++ 5.5.1 / Borland C++ Builder 5
- Borland C++ 5.8.1 / Borland Developer Studio 2006

The example applications described in section 4 default to using the MinGW compiler – however you can change this if you like. If you do not already have access to one of the above compilers, or wish to build the example applications without modification, then you will need to install MinGW.

3.3.1 Installing MinGW

The installer for MinGW is contained in `<rta>\VRTA\MinGW\MinGW-4.1.0.exe`. To install MinGW run this installer and follow the instructions displayed by the installer. The installer downloads the MinGW executables and headers from an Internet mirror site so you will need to have Internet access.

Important: you must accept the license terms displayed by the MinGW installer.

The "4.1.0" in `MinGW-4.1.0.exe` refers to the version number of the installer rather than the version number of the gcc compiler. The installer offers you a choice of MinGW versions to install. Choose the "Current" version. At this time of writing this would install gcc version 3.4.2. This is the version of gcc with which *RTA-OSEK for PC* was tested. Unfortunately LiveDevices does not have any control over the availability of version 3.4.2 of the gcc compiler.

Important: LiveDevices has provided the MinGW installer for your convenience. We have downloaded this installer from the MinGW web site (www.mingw.org) and have not modified it in anyway. LiveDevices and the ETAS Group do not claim any rights to MinGW/gcc nor does LiveDevices or the ETAS Group provide any support for MinGW/gcc.

3.4 Configuring toolinit.bat

`<rta>\vrta\toolinit.bat` is a DOS batch file that gets run during the build stage for a VECU. It simply sets certain environment variables that tell the RTA-OSEK GUI and example application build scripts where to find the compiler elements, and sets up some default values. If you look in `toolinit.bat` you will see that it is already set up to recognize a range of compilers. The value of the environment variable `VRTA` is used to determine which compiler to select:

e.g.

```
@echo off
< ...snip... >
if not @%1==@ set VRTA=%1
if @%VRTA%@==@MinGW@ goto MINGW
if @%VRTA%@==@BorlandC@ goto BCPP
if @%VRTA%@==@BDS2006@ goto BDS_2006
if @%VRTA%@==@VisualC5@ goto VCPP5
if @%VRTA%@==@VS2003@ goto VS2003
echo !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
echo Compiler not specified in environment variable
VRTA
echo Valid settings are:
echo     MinGW
echo     BorlandC
echo     BDS2006
echo     VisualC5
echo     VS2003
echo !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
goto exit
```

Important: Please maintain the current structure of `toolinit.bat` if you change it to avoid incompatibilities with compiler support in the RTA-OSEK tools.

3.4.1 Configuring the MinGW Section

Assuming that you wish to use MinGW you should edit the MINGW section of `toolinit.bat` so it sets the correct path for your MinGW installation. Assuming that you have installed MinGW in `c:\mingw` the MINGW section should look something like this:

```
:MINGW
rem tools installation directory
set CBASE=c:\mingw

rem location of C compiler
set CC=%CBASE%\bin\gcc.exe

rem location of C++ compiler
set AS=%CBASE%\bin\gcc.exe

rem location of linker
set LNK=%CBASE%\bin\g++.exe

rem location of Archiver / librarian
set AR=%CBASE%\bin\ar.exe

rem Set location of C include files
set CBASE_INC=%CBASE%\include

rem Default settings
SET _LIBS=-lwinmm -lws2_32
SET _OBJ=o

goto check
```

The environment variables set in `toolinit.bat` have the following uses:

`CBASE`	The root of the compiler installation.
`CC`	The path of the compiler executable.
`AS`	The path of the assembler executable. For *RTA-OSEK for PC* this is set to the compiler since the RTA-OSEK configuration is created in a C++ file (`osgen.cpp`) rather then an assembly file.
`LNK`	The path of the linker.
`AR`	The path of the archiver.
`_LIBS`	Default libraries needed. Usually this is the Window's multi-media library and the Winsock library.
`_OBJ`	The object file suffix.

Now open a console window and execute `<rta>\vrta\toolinit MinGW`[1], followed by `%cc% --version`. Your results should be like this:

```
C:\>c:\rta\vrta\toolinit MinGW
C:\>%cc% --version
gcc.exe (GCC) 3.4.2 (mingw-special)
Copyright (C) 2004 Free Software Foundation, Inc.
This is free software; see the source for copying
conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS
FOR A PARTICULAR PURPOSE.
C:\>
```

3.4.2 Configuring for a Different Compiler

If you wish to use one of the other compilers for which `toolinit.bat` is pre-configured instead of MinGW, modify that compiler's section in `toolinit.bat` so that `CBASE` is set to the correct directory for your installation of the compiler.

After doing this we recommend that you check that the settings within `toolinit.bat` are correct using a similar approach to that described for MinGW above.

3.5 vrtaServer

When *RTA-OSEK for PC* is installed the Virtual ECU Server application, `vrtaServer.exe`, is installed as a Windows service.

3.6 vrtaVMxxx.dll Location

When a Virtual ECU is started it tries to load the appropriate VM DLL (`vrtaVMs.dll` etc.). The VECU first tries to load the VM DLL using the normal DLL search rules. That is, it searches the following locations in the specified order:

1. The directory containing the VECU.
2. The current directory.
3. The 32-bit Windows system directory.
4. The Windows directory.
5. The directories listed in the `PATH` environment variable.

If the VECU fails to find the VM DLL then by default it tries to load it from the directory `c:\rta\bin`.

Therefore, if you have installed *RTA-OSEK for PC* in `c:\rta` (the default location) a VECU will always be able to find the appropriate VM DLL. If you

[1] `toolinit.bat` will use the first command-line parameter passed to it if the environment variable VRTA is not set.

have installed *RTA-OSEK for PC* in a different location then there are three ways of ensuring that VECUs can find VM DLLs:

- Add the directory `<rta>\bin` to the `PATH` environment variable.
- Edit the file `<rta>\vrta\inc\vrtaCore.cpp` and set the `DLL_SEARCH` #define to be the directory containing the VM DLLs (i.e. `<rta>\bin`).
- Define the `DLL_SEARCH` macro to be the directory containing the VM DLLs when `vrtaCore.cpp` or `osgen.cpp` is compiled. For example, define the `DLL_SEARCH` macro using a compiler command line option.

3.7 RTA-TRACE

RTA-TRACE, available as a separate product provides a very detailed graphical display showing in real-time the execution of all Tasks, ISRs and processes in your RTA-OSEK application.

RTA-OSEK for PC comes complete with a special high-bandwidth virtual device that can be used to connect to RTA-TRACE. If you have installed RTA-TRACE in the same location as RTA-OSEK then this link will be detected automatically. If not you must copy the file `rtcVRTAlink.dll` from RTA-OSEK's 'bin' to RTA-TRACE's 'bin'.

4 Running the Example Applications

In addition to the standard example application supplied with an RTA-OSEK port, *RTA-OSEK for PC* is supplied with three example applications that demonstrate *RTA-OSEK for PC* specific features.

For the sake of brevity the term `<APP>` is used as shorthand for `<rta>\VRTA\samples\Applications`.

4.1 NonOSEK

The directory `<APP>\NonOSEK` contains an example VECU that does not use RTA-OSEK. The example uses a collection of standard virtual devices such as a clock, counters, sensors and actuators.

4.1.1 What it does

The file `devices.cpp` creates a collection of standard devices. Comments in `devices.cpp` describe the configuration of the devices. The file `main.c` contains the executable entry-point `main()`, the application thread entry-point `OS_MAIN()`, the interrupt vector table and some ISRs. See the comments in the code.

The application is in two parts. The first part uses a `vrtaUpCounter` device called `UpCounter` and a `vrtaCompare` device called `UpCompare` attached to a `vrtaClock` device called `Clock` to generate an interrupt on vector 1 every 2 seconds. The ISR attached to interrupt vector 1 increments the value of a `vrtaActuatorLight` device called `Light`, a `vrtaActuatorDimmableLight` device called `DimmableLight` or a `vrtaActuatorMultiColorLight` device called `MultiColorLight` depending on the value of the `vrtaSensorMultiwaySwitch` device called `MultiwaySwitch`.

The second part uses a `vrtaDownCounter` device called `DownCounter` and a `vrtaCompare` device called `DownCompare` attached to the `vrtaClock` device called `Clock` to generate an interrupt on vector 2 every second. The ISR attached to interrupt vector 2 increments the 0th element of a `vrtaIO` device called `I/O` if a `vrtaToggleSwitch` device called `ToggleSwitch` has the value 1.

4.1.2 Building it

To build the example with MinGW use the command:

```
mk_mingw.bat
```

To build the example with Borland C++ Builder 5 use the command:

```
mk_borland5.bat
```

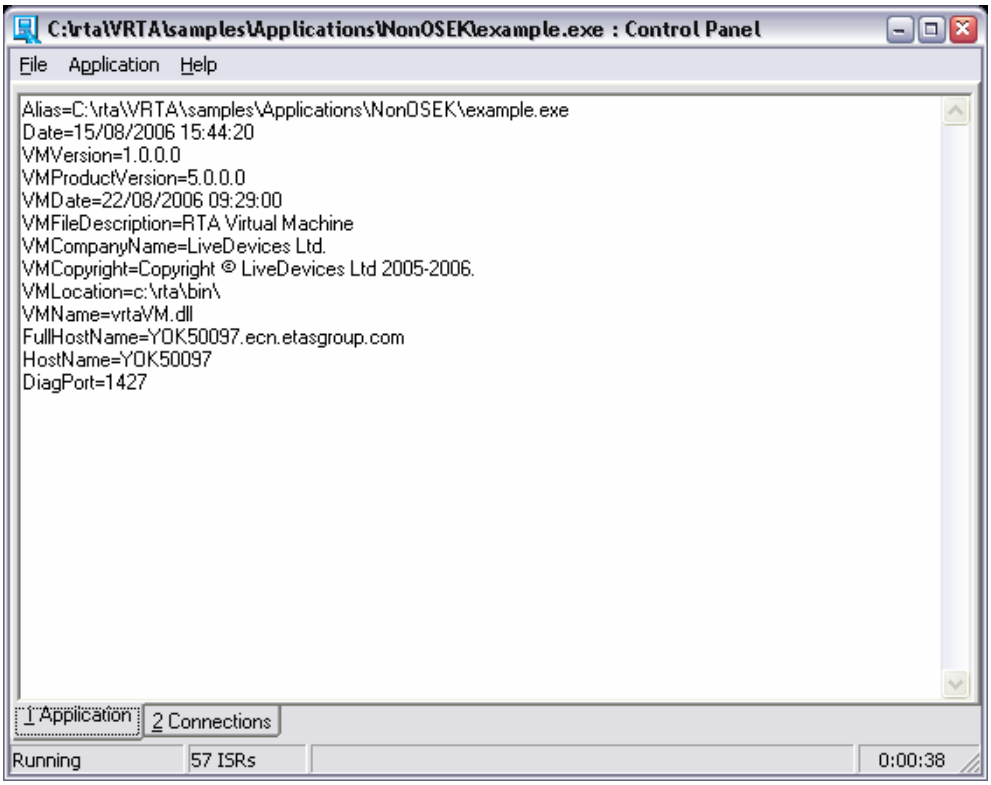
To build the example with Microsoft Visual Studio 2003 use the command:

```
mk_visual2003.bat
```

This will create a VECU executable called `example.exe`.

4.1.3 Running it

Since a VECU is just a Windows executable you can simply run `example.exe` (either from the command line or from Windows Explorer). If you do this you will see a GUI like this:

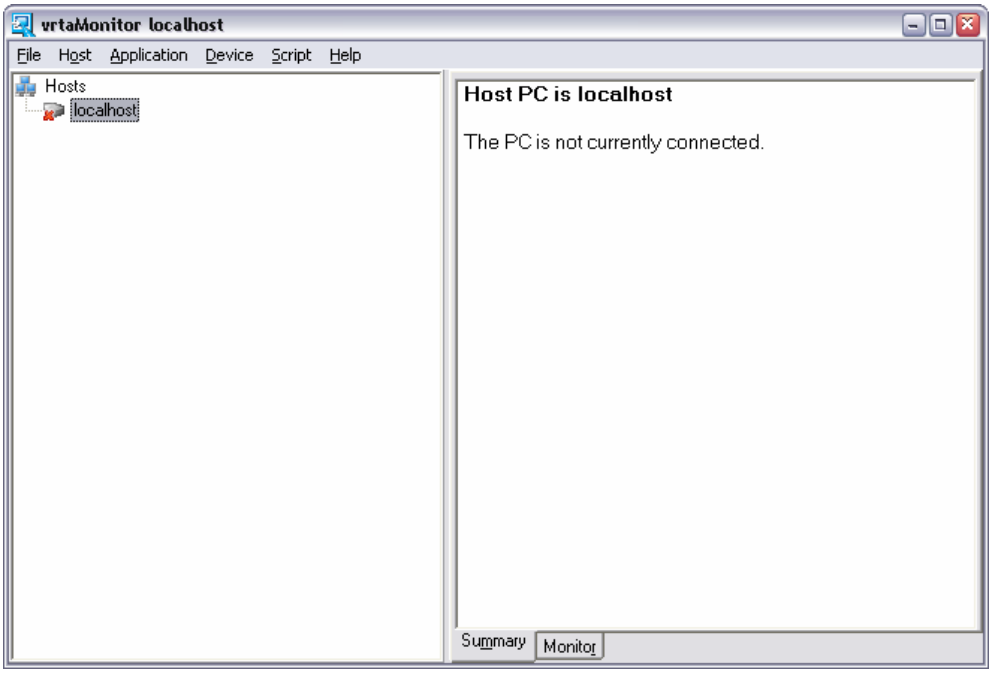


Unfortunately this is not very exiting! All you will see is the ISR count increasing in the status bar at the bottom. Select the main menu item **Application / Terminate** to close the VECU.

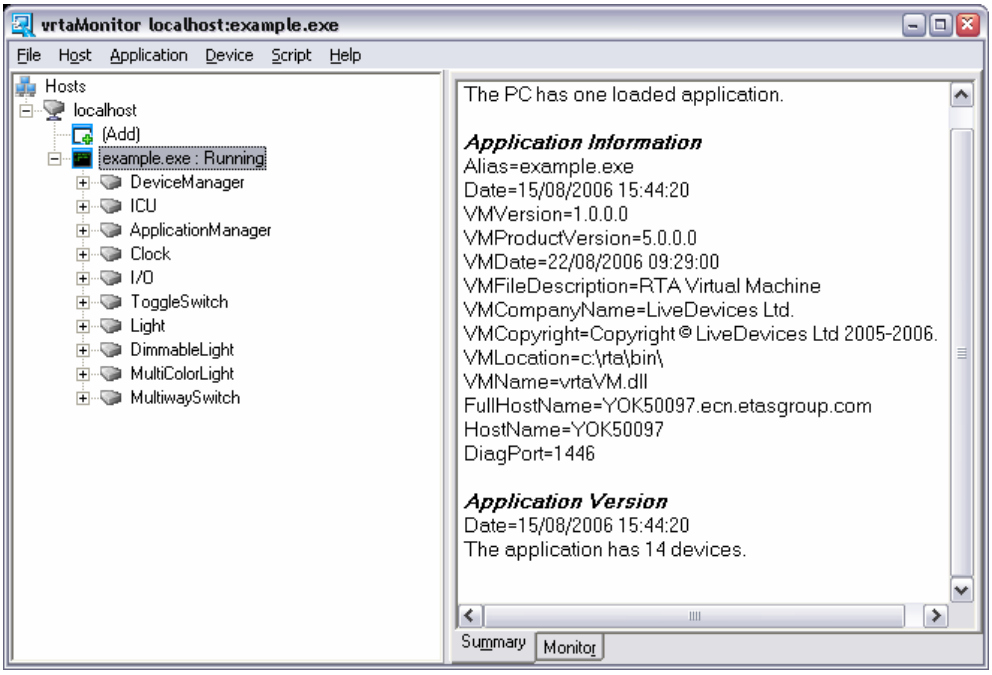
It is more interesting to run `example.exe` using the Virtual ECU monitor, `vrtaMonitor`. Run the command:

```
<rta>\bin\vrtaMonitor
```

You will now see the `vrtaMonitor` GUI as below:

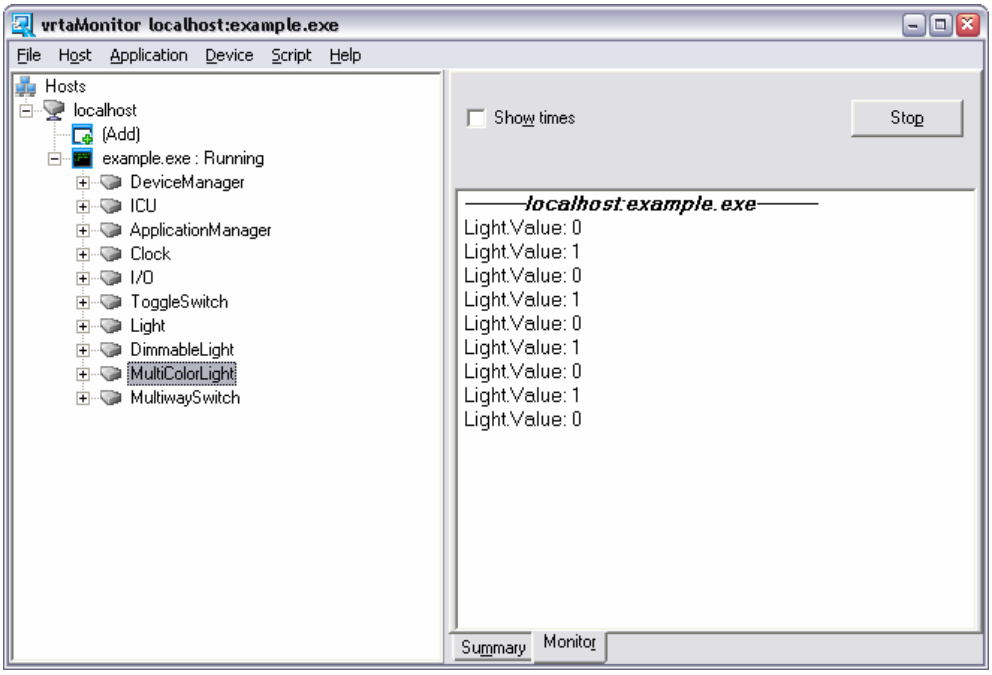


Double click on the "**localhost**" item and then on the "**(Add)**" item that appears below it. A file selection dialog will now appear. Select the `example.exe` file. `vrtaMonitor`'s GUI will now look like:

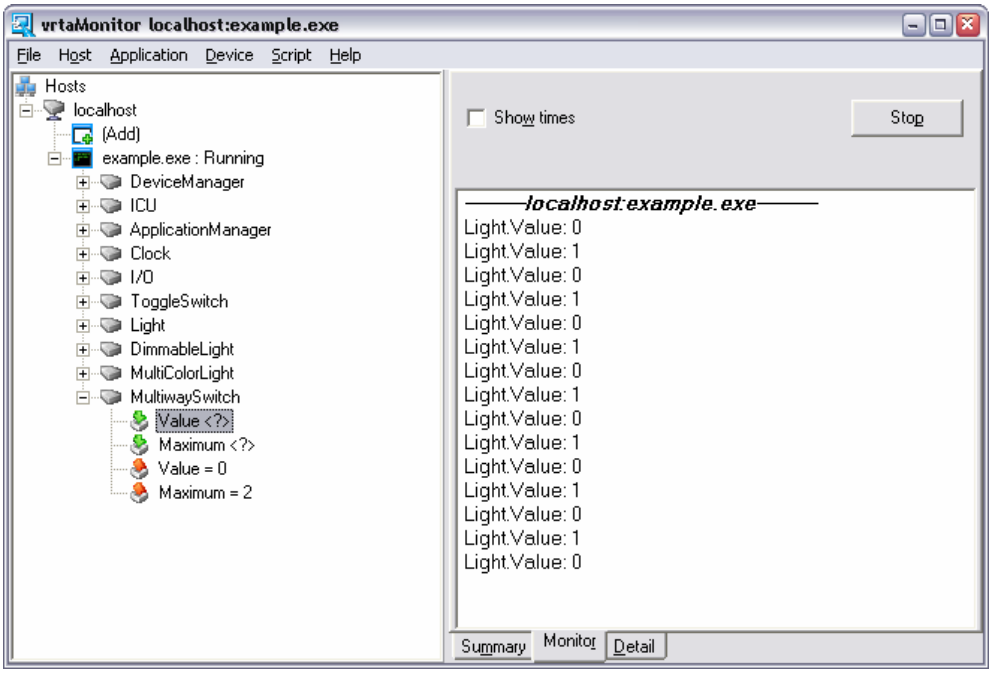


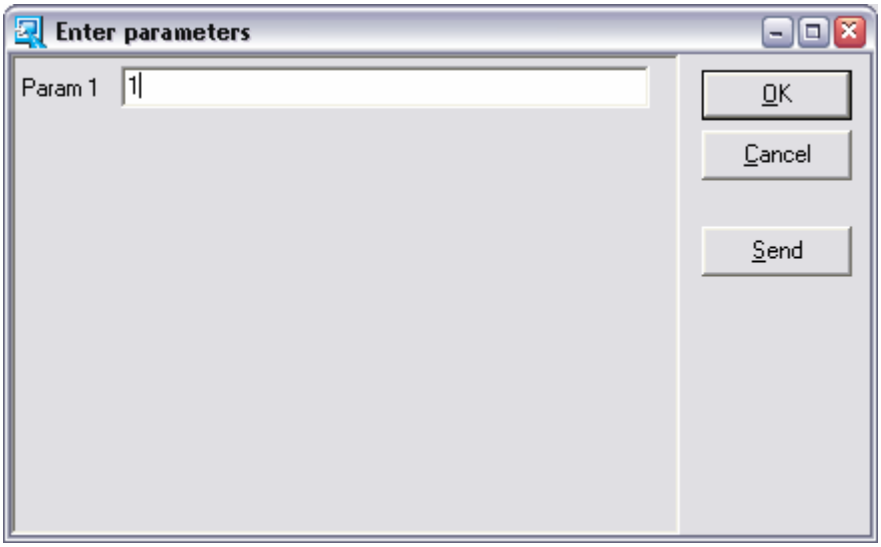
The right-hand pane contains information about the VECU. The left-hand pane contains a list of the virtual devices that are present inside the VECU.

Switch to the '**Monitor**' tab and then drag the "**I/O**", "**Light**", "**DimmableLight**" and "**MultiColorLight**" devices onto the right-hand pane. The GUI should now look something like:

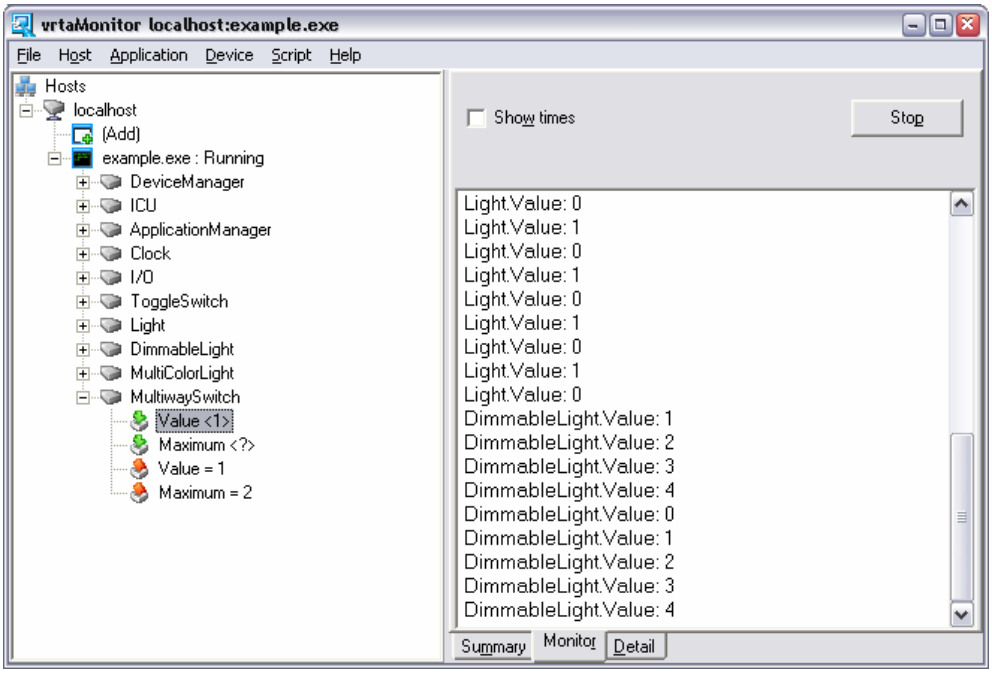


The right-hand pane now shows *events* that are being raised by the devices in the VECU. At present the right-hand pane shows that the value of the "Light" device is toggling between 0 and 1 every two seconds. Expand out the "MultiwaySwitch" device by clicking on the "+". The green down-arrows represent *actions* that can be sent to the device. The red up-arrows represent events that the device can raise. Send a "Value" action to "MultiwaySwitch" to change its value by double-clicking on its "Value" action. In the dialog box that appears set "Param1" to 1 and then press the 'OK' button.





Having done this the GUI should now look something like:



As you can see "Light" has stopped toggling and "DimmableLight" is now incrementing every two seconds. If you now send another "Value" action to "MultiwaySwitch" specifying "Param1" as 2 you will see that "DimmableLight" stops incrementing and "MultiColorLight" starts incrementing. By default when you double-click on an action a second time it sends the same parameters as used previously. To change the parameters, right-click on the action and choose the "Params..." option from the pop-up menu.

You can also experiment with the "I/O" device. If you send a "GetValues" action to the I/O device by expanding the tree view for the I/O device and

double-clicking on the "GetValues" action you should see a list of I/O cell values appear in the right-hand pane (the "GetValues" action causes a "Values" event to be raised). If you send this action several times you should see that the I/O cell values do not change. Now send a "Position" action to the "ToggleSwitch" device with the parameter value 1. If you now send "GetValues" actions to the "I/O" device you should see that the I/O cell values change every second.

Once you have finished, select the main menu item Application / Kill to close the VECU and then File / Exit to close `vrtaMonitor`.

4.2 RTA-OSEK Example 1

The directory `<APP>\RTA-OSEK Example 1` contains the standard RTA-OSEK example ported to *RTA-OSEK for PC*. Although extra code has been added to support virtual devices, the core files of the example application have not been changed.

4.2.1 What it does

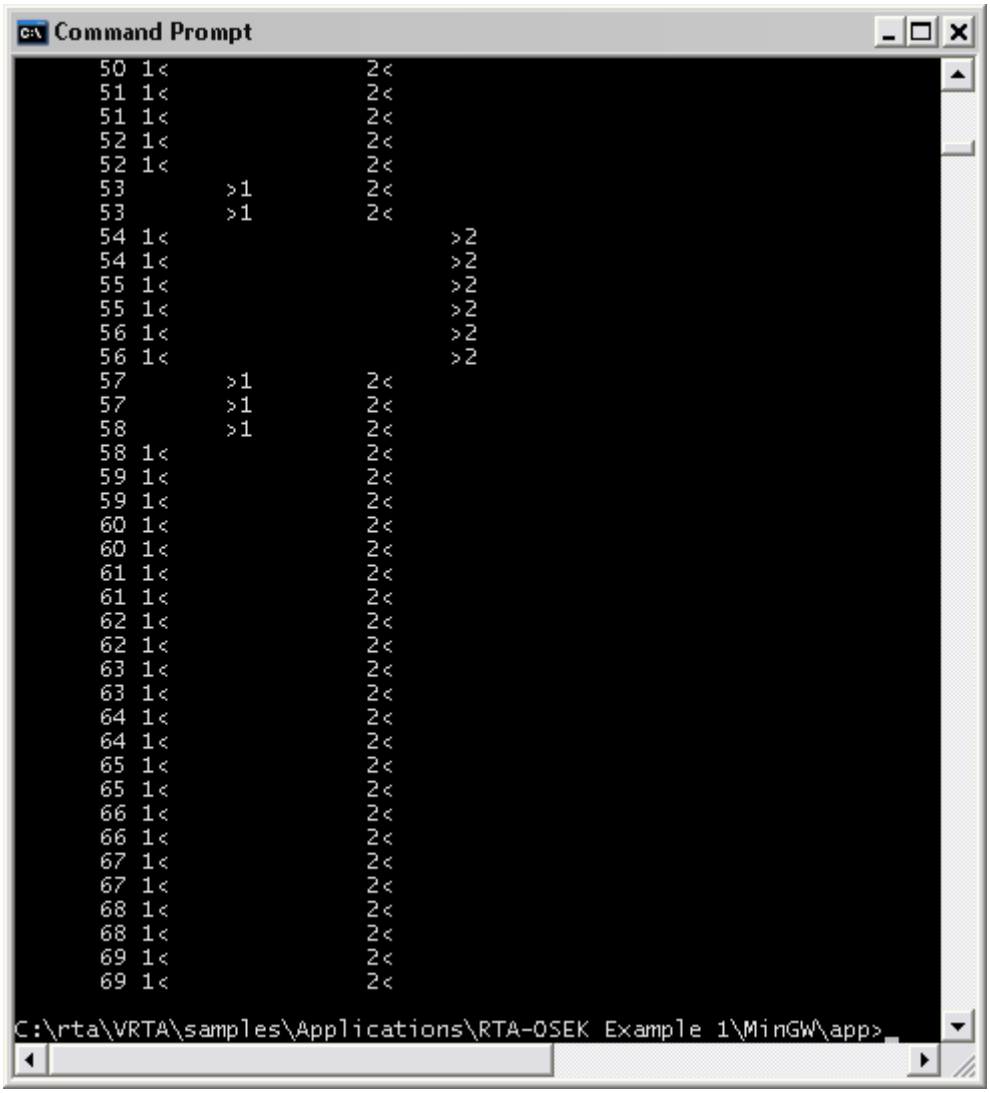
The standard RTA-OSEK example uses two tasks to toggle a pair of I/O pins to generate a trace that can be viewed on an oscilloscope. This example is available in `<rta>\VRTA\example`. The example in `<APP>\RTA-OSEK Example 1` is basically the same except it outputs the "oscilloscope trace" using `printf()`. It also uses a 1 second clock tick rather than a 1 millisecond clock tick to make the output readable.

4.2.2 Building it

The example is built with RTA-OSEK in the normal way – see the "RTA-OSEK User Guide" for details (`<rta>\docs\RTA-OSEK User Guide.pdf`). Start RTA-OSEK and then from the main menu select File / Open.... In the file selection dialog that appears select `<APP>\RTA-OSEK Example 1\example.oil`. Switch to the '2 Builder' tab and select 'Custom Build' from the list on the left-hand side. Now press the 'Build Now (F9)' button. This will build the example using the MinGW compiler – see section 4.2.5 if you wish to use a different compiler.

4.2.3 Running it

The VECU executable will be created in `<APP>\RTA-OSEK Example 1\MinGW\app\example.exe`. Run this program from a command window. You will see output that looks something like:



The numbers in the left-hand column represent elapsed seconds. The `1<` numbers represent the oscilloscope trace for I/O pin 1 and the `2<` numbers represent the oscilloscope trace for I/O pin 2.

When you have finished close the VECU by selecting main menu option Application / Terminate from its GUI.

4.2.4 Using RTA-TRACE

If you have RTA-TRACE available you can use it to see what is going on inside the VECU. In RTA-OSEK open the file `<APP>\RTA-OSEK Example 1\example_plus.oil`. This OSEK configuration uses a 1 millisecond clock tick and is configured for RTA-TRACE. Switch to the '2 Builder' tab and select 'Custom Build' from the list on the left-hand side. Now press the 'Build Now (F9)' button. This will build the example.

Start RTA-TRACE and from the main menu select File / New Connection .... A dialog will appear asking for the location of the target; accept the "localhost"

default. Another dialog will now appear asking you to select the OS interface. Select "RTAOSEK-VRTA". Next in the file selection dialog that appears select the file `<APP>\RTA-OSEK Example 1\MinGW\app\example_plus.rta`. After a short pause you will see data appearing in the RTA-TRACE window. Once some data has appeared, bring the VECU's GUI to the foreground and close it by selecting main menu option Application / Terminate.

4.2.5 Using a Different Compiler

If you wish to build the example using one of the other pre-configured compilers instead of MinGW you need to do the following:

- Make sure that you have configured `toolinit.bat` for your compiler installation – see section 3.4.
- Run RTA-OSEK and load the `example.oil` (or `example_plus.oil`) file as described above.
- In the '1 Planner' tab select the Target / Target Type options from the list on the left-hand side. Press the 'Change Variant' button. In the dialog that appears choose the compiler variant you wish to use. The compiler variants correspond to the `VRTA` value used by `toolinit.bat`. E.g. `MinGW` for MinGW and `BorlandC` for Borland C++ 5.5.1 / Borland C++ Builder 5.
- Build the example as described above. The executable will be built in a directory tree that has the name of the variant.

4.3 RTA-OSEK Example 2

The directory `<APP>\RTA-OSEK Example 2` contains an example that simulates a car with a simple set of controls.

4.3.1 What it does

The simulated car has a throttle with range 0 to 100, a brake (similarly 0 to 100), gears 0 to 5 and a steering wheel. Changing their values effects the engine revs, the speed and direction.

Plus if the Audio device is on, you get to annoy your co-workers!

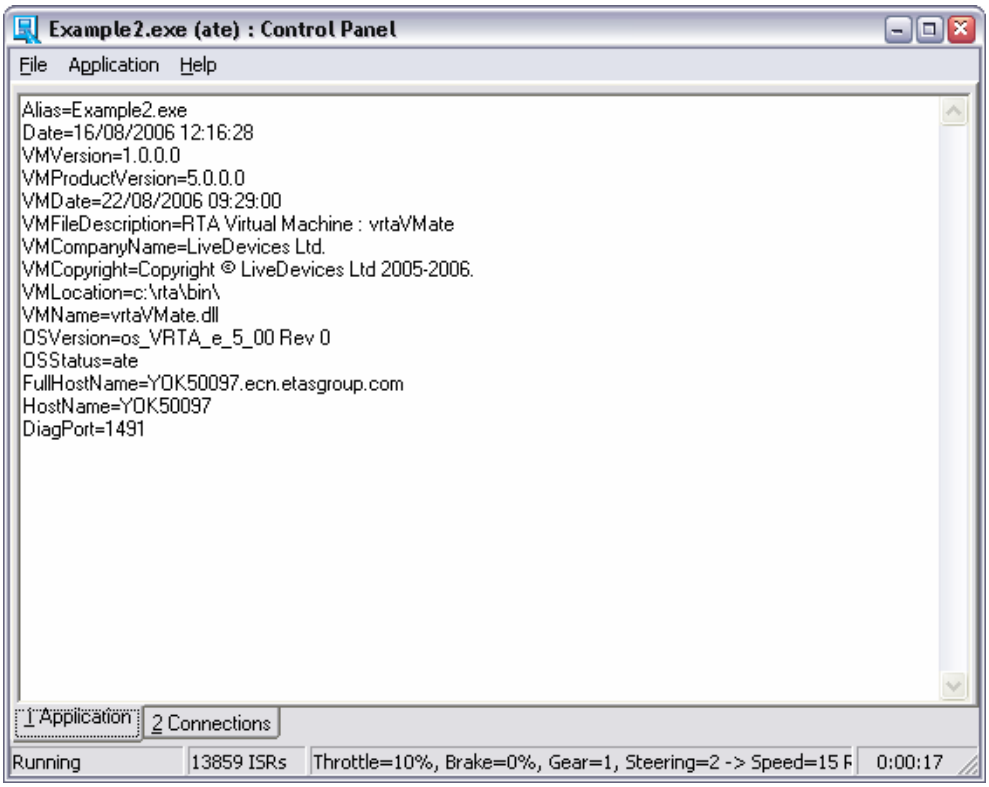
4.3.2 Building it

The example is built with RTA-OSEK in the normal way – see the "RTA-OSEK User Guide" for details (`<rta>\docs\RTA-OSEK User Guide.pdf`). Start RTA-OSEK and then from the main menu select File / Open.... In the file selection dialog that appears select `<APP>\RTA-OSEK Example 2\Example2.oil`. Switch to the '2 Builder' tab and select 'Custom Build' from the list on the left-hand side. Now press

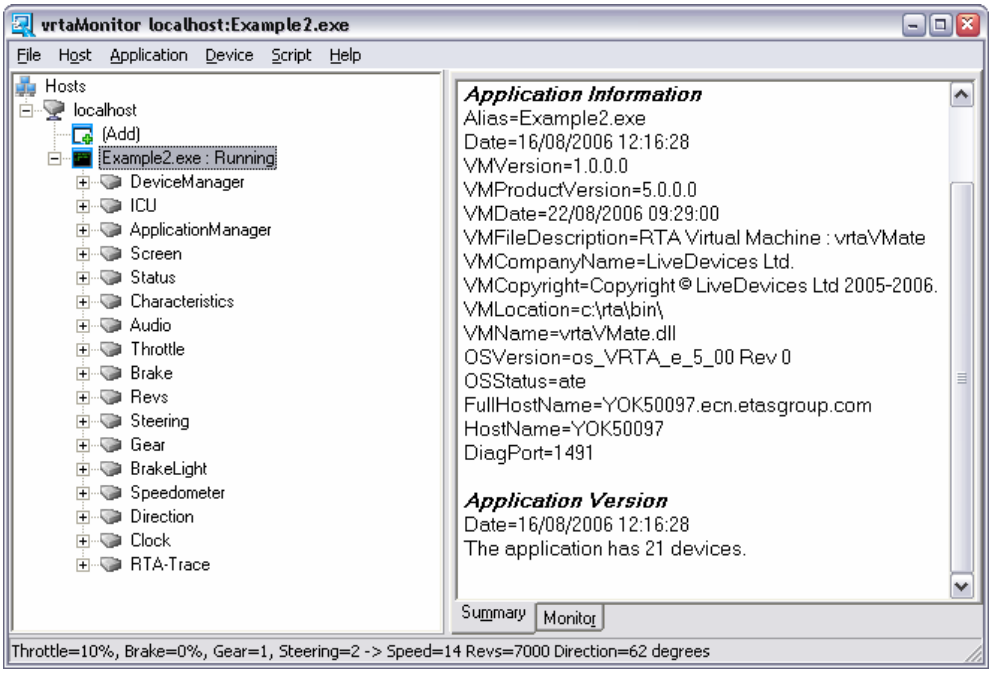
the 'Build Now (F9)' button. This will build the example using the MinGW compiler – see section 4.2.5 if you wish to use a different compiler.

4.3.3 Running it

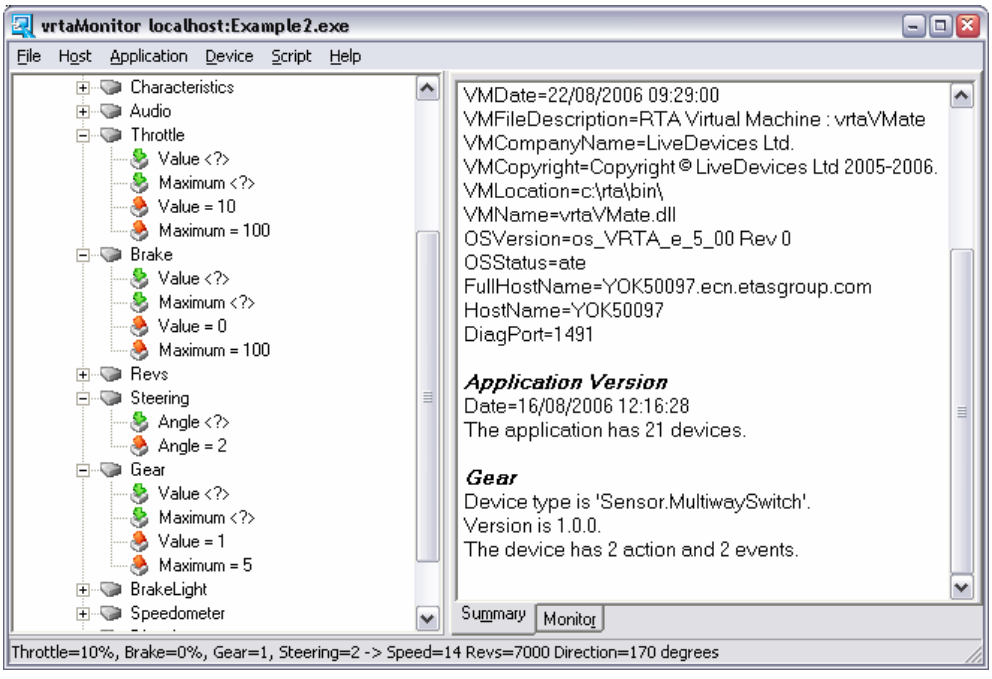
The VECU executable will be created in `<APP>\RTA-OSEK Example 2\MinGW\app\Example2.exe`. Run this program from a command window. When the VECU's GUI appears select the main menu option Application / Monitor. This will launch `vrtaMonitor` and automatically connect it to the VECU. The VECU's GUI should look something like:



And `vrtaMonitor` will look something like:



You will notice that both the VECU's GUI and `vrtaMonitor` display status information at the bottom of their windows. This status information includes the Throttle, Break, Gear and Steering wheel settings as well as the car's Speed, Revs and Direction. Try sending "**Value**" or "**Angle**" actions to the "**Throttle**", "**Brake**", "**Steering**" and "**Gear**" devices and notice the effect on the Speed, Revs and Direction values. For example, expand out "**Thottle**" by clicking on the "**+**" and then send a "**Value**" action to it by double-clicking on the "**Value**" action (remember that actions have green down-arrows).



4.3.4 Using RTA-TRACE

You can use RTA-TRACE with this example in the same way you did with the previous example – see section 4.2.4. The example is configured for RTA-TRACE. When RTA-TRACE asks for a file select `<APP>\RTA-OSEK Example 2\MinGW\app\Example2.rta`. When the VECU GUI appears attach `vrtaMonitor` to the VECU by selecting Application / Monitor from the VECU's main menu.

5 Where Next?

5.1 Further Documentation

In addition to this Getting Started Guide the following *RTA-OSEK for PC* specific documentation is provided:

RTA-OSEK for PC User Guide
(`<rta>\docs\RTA-OSEK for PC User Guide.pdf`)

This document contains information on how to use the *RTA-OSEK for PC* specific tools and how to build Virtual ECUs.

RTA-OSEK Binding Manual PC
(`<rta>\docs\RTA-OSEK Binding Manual PC.pdf`)

This document describes how the *RTA-OSEK for PC* component binds to the Virtual ECU environment.

You may also find the following generic RTA-OSEK documentation useful:

RTA-OSEK Getting Started Guide
(`<rta>\docs\RTA-OSEK Getting Started Guide.pdf`)

RTA-OSEK User Guide
(`<rta>\docs\RTA-OSEK User Guide.pdf`)

RTA-OSEK Reference Guide
(in `<rta>\docs\RTA-OSEK Reference Guide.pdf`)

5.2 The Next Step

Having worked through this Getting Started Guide we recommend that you move on to the "Overview" and "Tutorial" chapters of the "RTA-OSEK for PC User Guide".

Support

For product support, please contact your local ETAS representative.

Office locations and contact details can be found on the ETAS Group website www.etasgroup.com.