



acontis technologies GmbH

SOFTWARE

# **RTOS32Win**

Getting Started Guide

Version 7.0

© Copyright **acontis technologies GmbH**

Neither this document nor excerpts therefrom may be reproduced, transmitted, or conveyed to third parties by any means whatever without the express permission of the publisher. At the time of publication, the functions described in this document and those implemented in the corresponding hardware and/or software were carefully verified; nonetheless, for technical reasons, it cannot be guaranteed that no discrepancies exist. This document will be regularly examined so that corrections can be made in subsequent editions. Note: Although a product may include undocumented features, such features are not considered to be part of the product, and their functionality is therefore not subject to any form of support or guarantee.

# Table of Contents

## Inhalt

1.	Introduction .....	4
1.1.	Documentation .....	4
2.	Getting Started with the System Manager .....	5
2.1.	Prerequisites .....	5
2.2.	The Software Development Kit (SDK) .....	5
2.3.	System Manager .....	6
3.	Setup the Real-time Extension Runtime.....	7
4.	Real-time Demo (run pre-compiled demo) .....	9
5.	Shared Memory Demo (build ShmDemo) .....	14
	Windows part of the ShmDemo.....	19
	Running the Demo .....	21
6.	Deployment to a runtime PC .....	22
	On the development PC .....	22
	On the target PC.....	23
7.	Device Partitioning.....	24
7.1.	Real-time extension configuration (for ex. Network Card) .....	24
7.1.1.	Identify your Network Controller .....	24
8.	Building the Real-time Extension Example Programs (Windows part) .....	26
1.	Build instructions for Windows 7 / 8 / 10 .....	26
II.	Appendix.....	27
1.	Adding User settings in a System Manager configuration.....	27
2.	Technical Background of System Manager .....	27

# 1. Introduction

The System Manager for RTOS-32 is an add-on for the acontis RTOS-32-based product line (**Rtos32Win & EC-Win(RTOS-32)**) of real-time extension for Windows 7 / 8 / 10 32-Bit / 64-Bit.

The acontis **Rtos32Win** enables RTOS-32 to share a single computer with a Microsoft Windows desktop operating system while maintaining RTOS-32 hard real-time determinism and performance. Both operating systems operate independent from each other, and, therefore, RTOS-32 operation is immune to desktop Windows exceptions and faults.

This document gives you an overview about the System Manager for RTOS-32 add-on. In this document, a step-by-step walkthrough is given how to setup the Rtos32Win real-time extension and get our real-time example program ("RealtimeDemo") running under RTOS-32. In the last chapter, we give brief instructions how the shipped SDK examples can be built with Visual Studio.

The real-time extension package comes with the following components:

- Real-time extension runtime
  - Kernel driver for virtual network adapter between Windows and RTOS-32
  - Kernel driver to assign hardware resources to the real-time OS (RTOS-32)
  - Virtual Machine Framework (Runtime shared by Windows and RTOS-32)
  - Ready to Run "RTOS-32" OS Image + SDK.
- SDK's for the real-time extension
  - Integrated EVAL version of RTOS-32
  - Libraries and header files for developing your own C/C++ applications, using our RTOS framework. E.g., you can develop applications that can communicate between Windows and RTOS-32 through shared memory and signals.
  - Extensible demo applications and examples (under `SDK/examples/`)
- Tools and Utilities
  - System Manager. This is the primary configuration tool for our real-time extension product family.

## 1.1. Documentation

For detailed information about the real-time extension (Rtos32Win), please read the [RTOS32Win.pdf](#) manual.

The Rtos32Win API is described in the [RTOS-Virtual-Machine-ReferenceManual.pdf](#) document.

## 2. Getting Started with the System Manager

This manual will guide you through the following steps:

1. Short **overview** over System Manager
2. **Setting up** a real-time configuration for RTOS-32
3. **Hardware / Software setup** for a valid configuration
4. **Setup, Build and Debug the supplied ShmDemo** with Visual Studio (See chapter 5)
5. **Deployment** of the real-time configuration

### 2.1. Prerequisites

Please make sure that the following conditions are met:

- Installed .NET Framework Version 4.0 (Client Profile) (For running the System Manager)
- Installed Microsoft Visual Studio 2005 or newer with latest service pack.
- Installed On Time RTOS-32 **Eval**<sup>1</sup> package, at least Version 5.33 (for compiling the demo application)  
**Hint: Since version 5.1.0.23 of the real-time extension the On Time RTOS-32 Eval package is integrated.**
- Installed Rtos32Win or EC-Win(RTOS-32) (requires at least .NET Framework Version 4.0 Client Profile).
- Intel PRO/1000 gigabit network controller hardware installed on the target device. (optional)
- For **Windows 7 / 8 / 10 64-Bit** edition only: At least two (logically) CPU cores.

### 2.2. The Software Development Kit (SDK)

The software development kit, located in SDK (below the installation directory), contains libraries and header files to develop software on top of the real-time extension (Rtos32Win based).

Furthermore, it contains the integrated On Time RTOS-32 Eval package.

---

<sup>1</sup> **Important:** Using the FULL version of On Time RTOS-32 requires other steps for debugging. Thus please read chapter 5.2 of RTOS32Win.pdf, especially the section about Debug Monitor. The System Manager does recognize automatically the installed RTOS-32 version.

## 2.3. System Manager

The System Manager is used for configuration of the real-time part of real-time extension as well as for managing the whole development process. It provides functionality to manage the memory configuration, the CPU assignment to the RTOS and the assignment of PCI devices. Furthermore, it provides functionality to deploy applications to the real-time part and create the Visual Studio projects for the example programs.

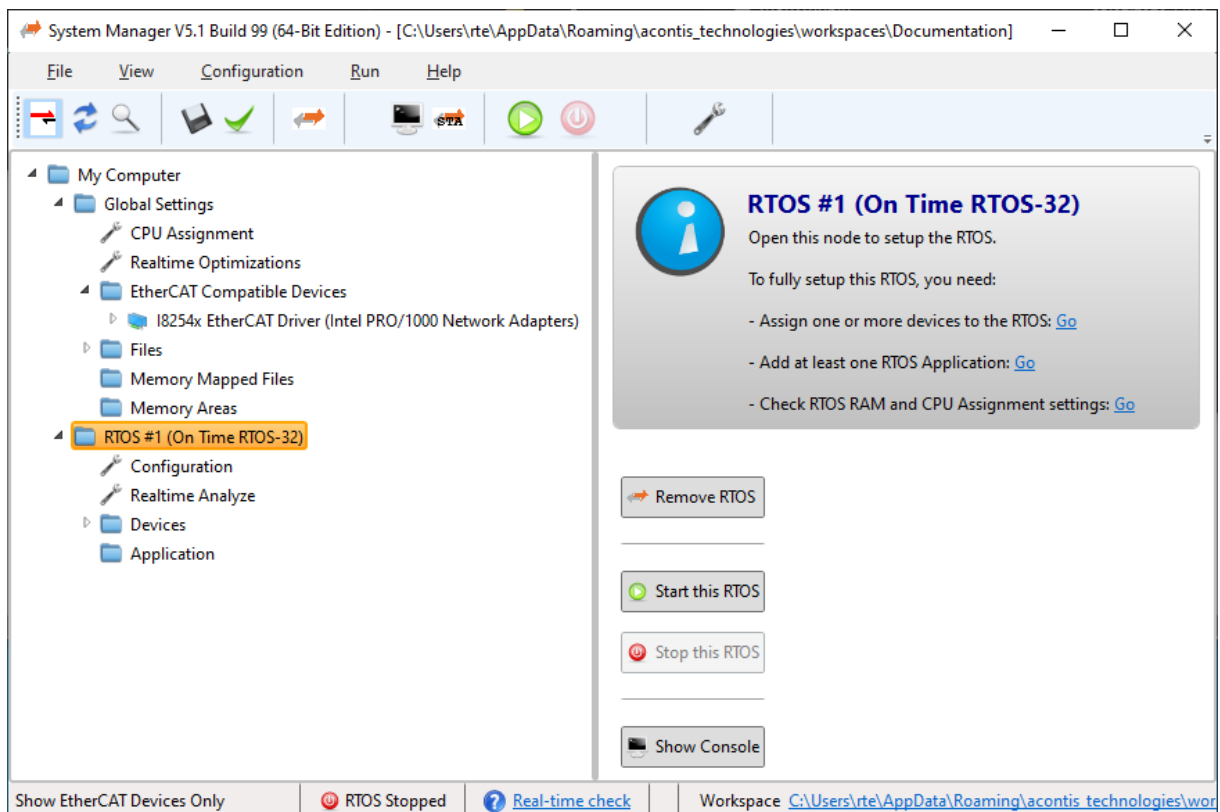


Figure 2-1 (System Manager)

### 3. Setup the Real-time Extension Runtime

This chapter describes how to setup the basic parameters of the real-time extension via the supplied System Manager tool. First step is to choose a place for your workspace. Selecting the default place is recommended.

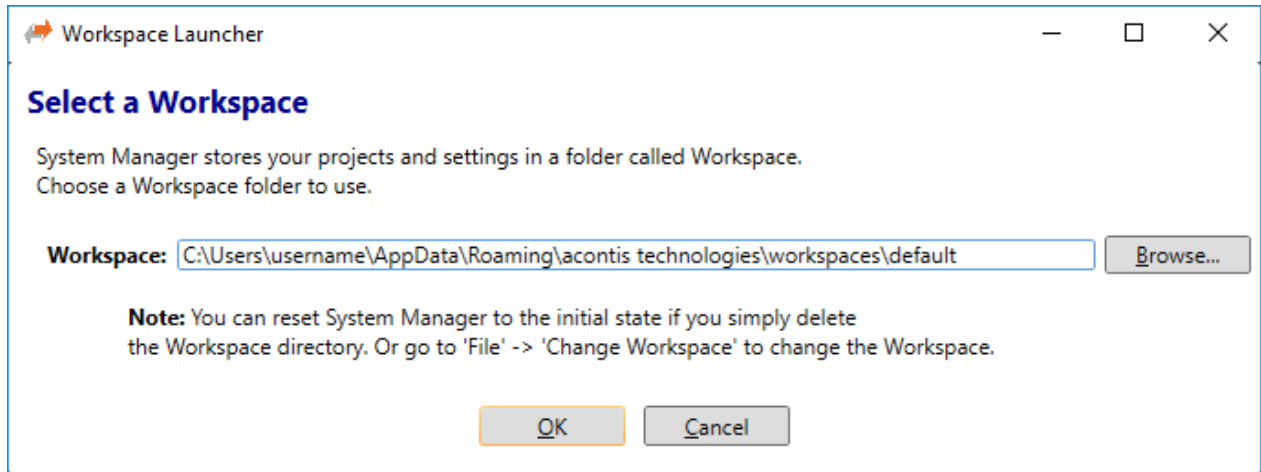


Figure 3-1 (Add Workspace)

Next step is to “Add RTOS”. In this example, it’s “On Time RTOS-32”. After adding RTOS, it is time to go to the “CPU Assignment”

If started first and Windows 7 32BIT is used all available CPUs are assigned to Windows 7 and the first CPU is additionally assigned to On Time RTOS-32. (Figure 3-3) This CPU Assignment may be valid, however it is not recommended. To achieve a recommended “CPU Assignment” assign all CPU’s to Windows, except the last one. The last CPU has to be assigned to “On Time RTOS-32”. (Figure 3-4)

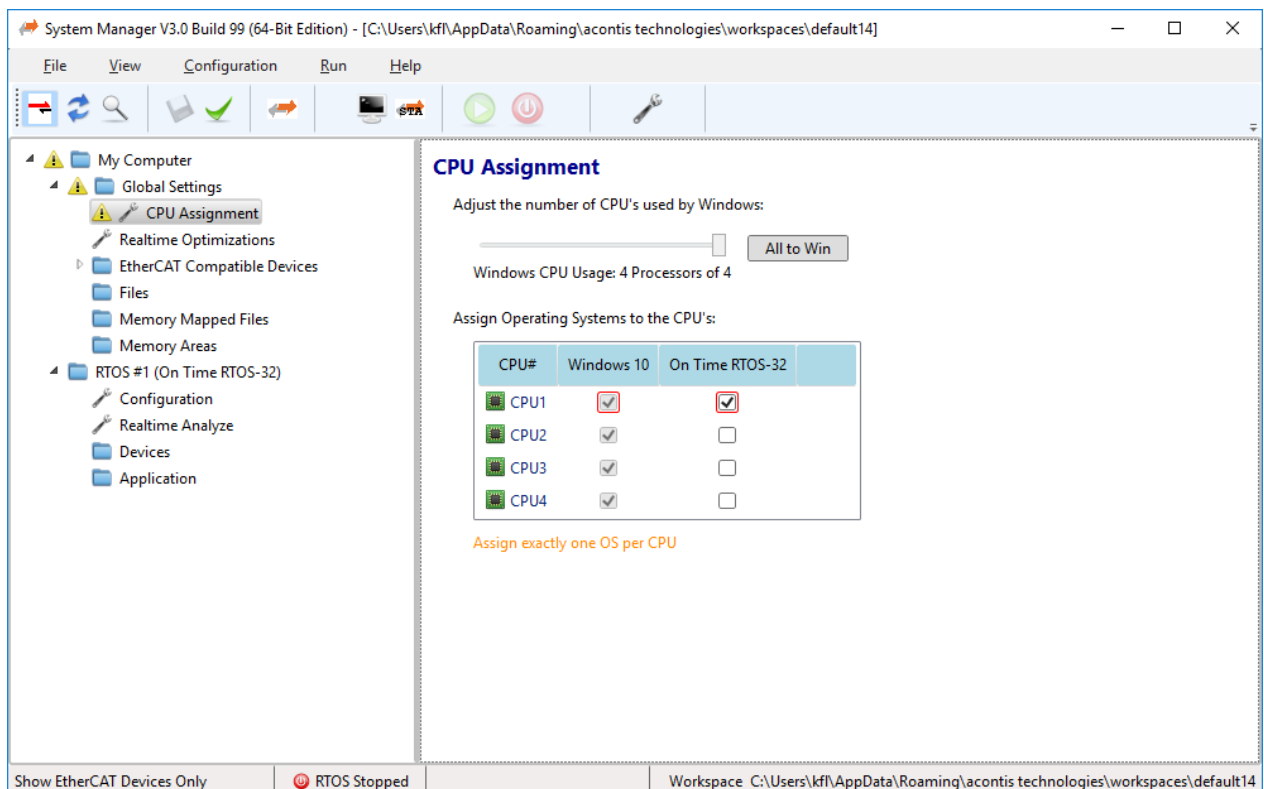


Figure 3-3 (CPU Assignment)

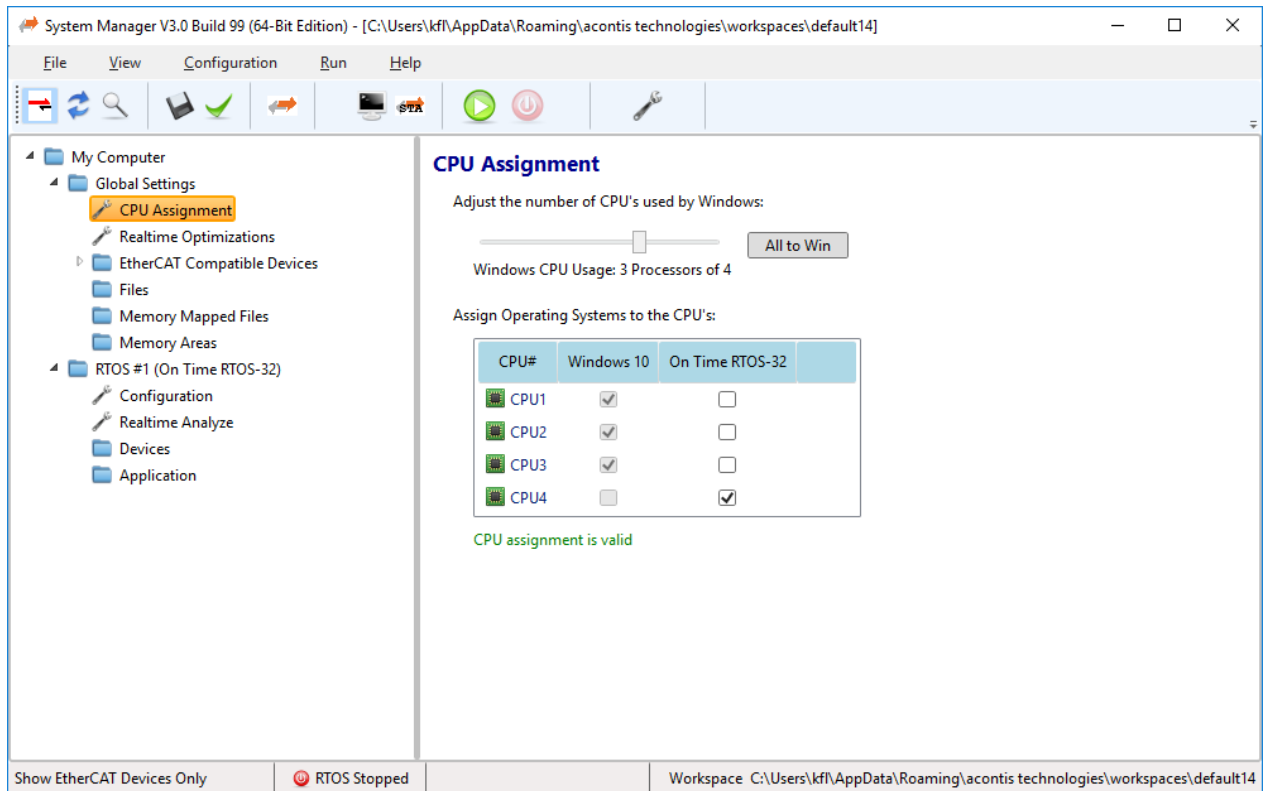


Figure 3-4 (CPU Assignment)

If Windows 7 / 8 / 10 64Bit is used, the real-time part **must** use at least one CPU exclusively. To achieve this use the “number of processor slider” above the CPU table in such a way that the real-time part has at least one CPU exclusively. After doing that, the CPU assignment should be valid.

→ *It's important that the message beyond the CPU table is green and the CPU assignment is valid!*

The System Manager will show a reboot message box, if some changes require a reboot after *save settings* has been pushed.



## 4. Real-time Demo (run pre-compiled demo)

Several demos are delivered as pre-compiled binaries, which are ready-to-use. To measure the target PC for real-time capability a RealtimeDemo is supplied. The following steps describe the needed actions in the System Manager to use the supplied, pre-compiled RealtimeDemo. As a starting point a valid System Manager configuration is needed → see previous chapter 3 Setup the Real-time Extension Runtime.

As first step select application node on the RTOS #1 sub node and push the **“Open Existing Application”** button as shown in the following screenshot.

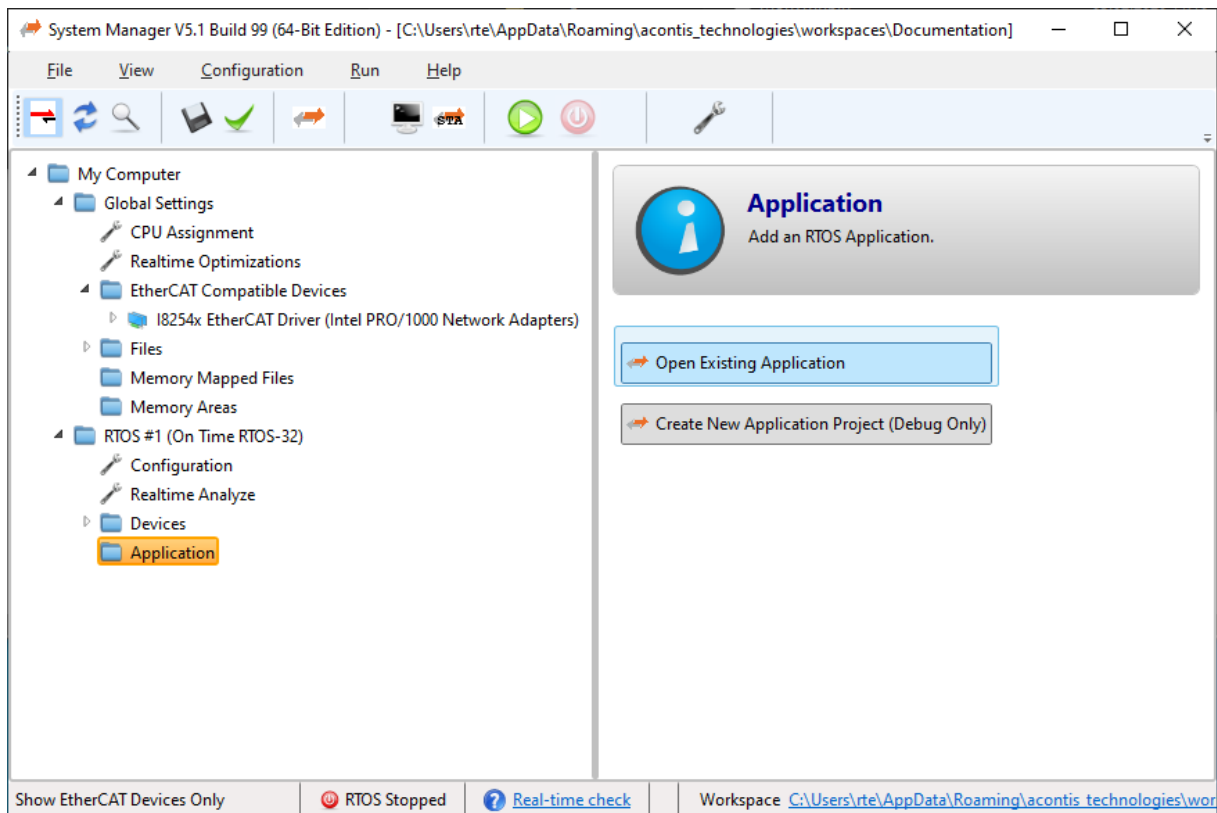


Figure 4-1

After pushing the button, the following dialog appears:

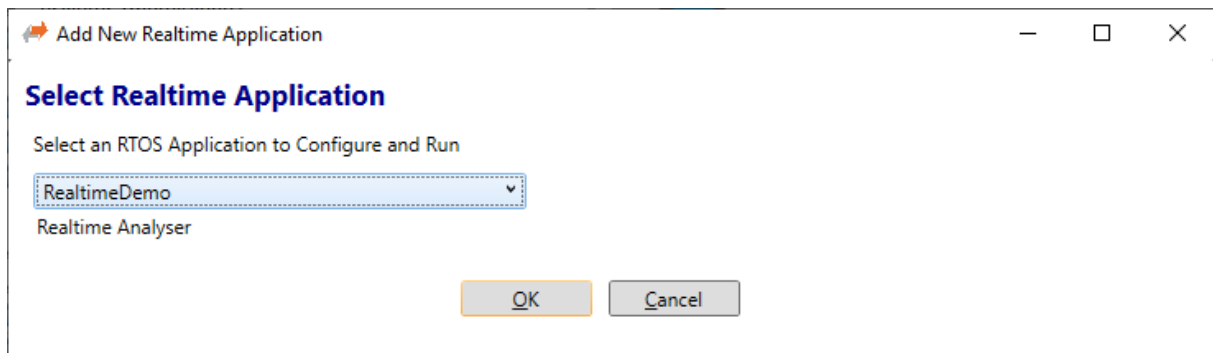


Figure 4-2

Select the RealtimeDemo and push the OK button. After successful adding of the RealtimeDemo to the application node, the following “image” should be visible:

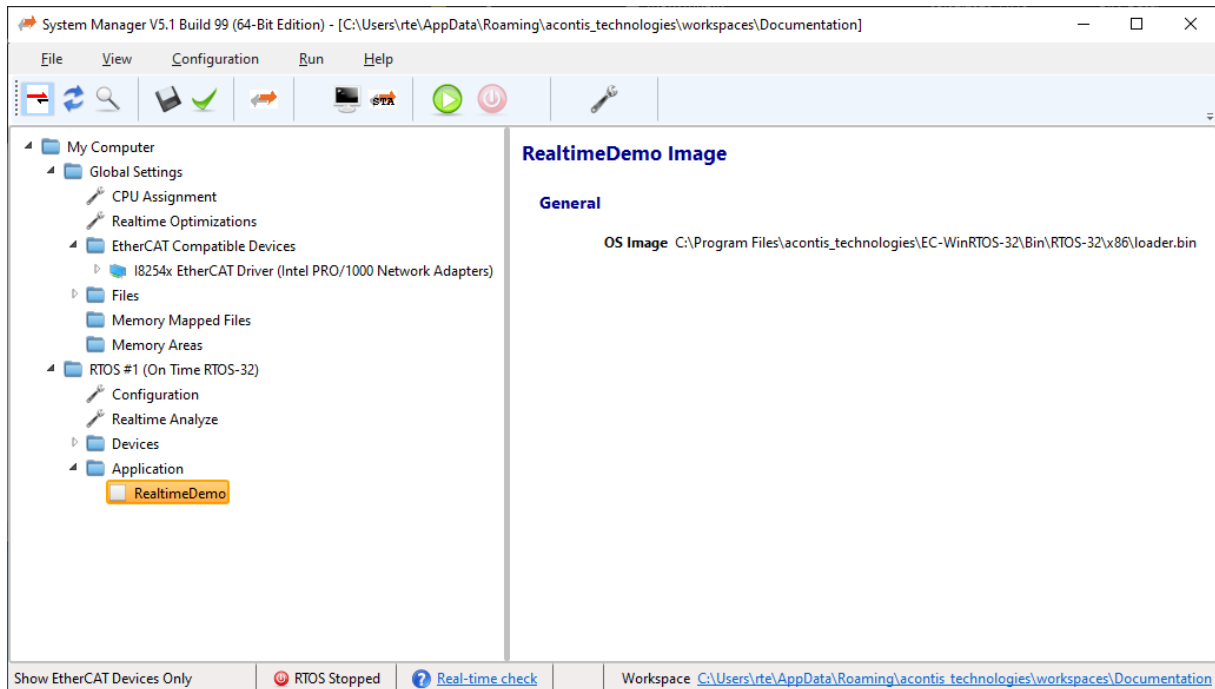


Figure 4-3

The next step requires starting the “Target Console Window”. This is not mandatory, but without the console output nothing of the RealtimeDemo, output will be shown to the user. To start the Target Console Window the monitor button must be pushed as shown in the following screenshot:

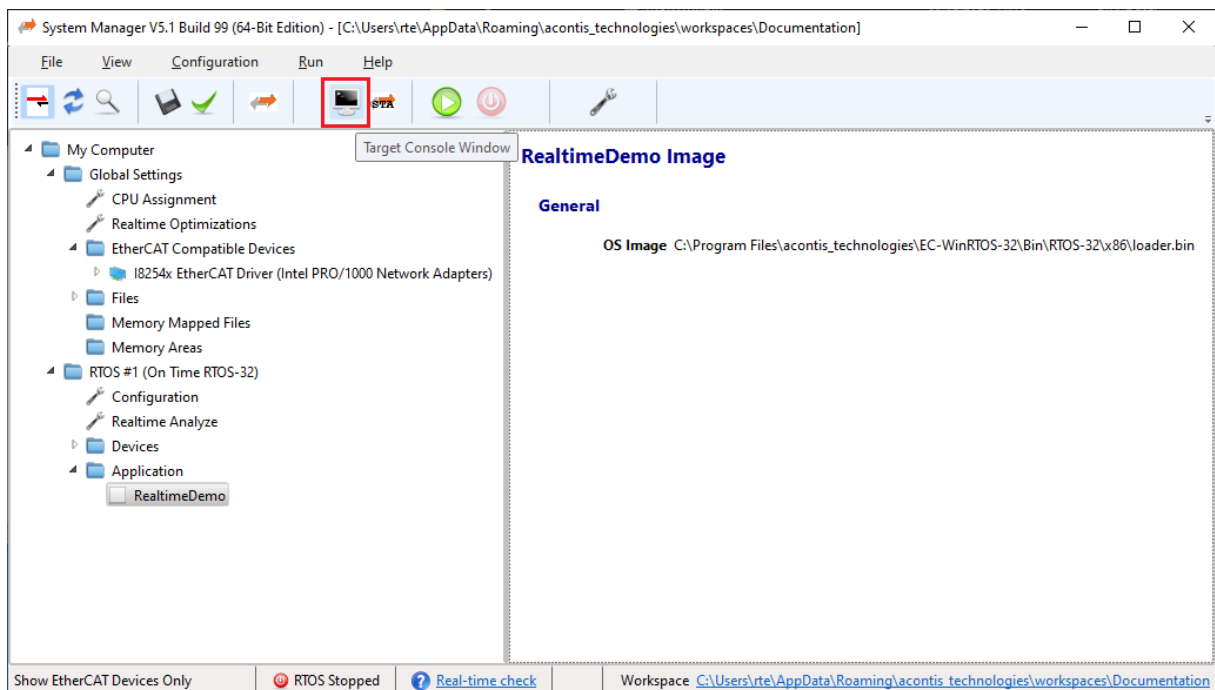


Figure 4-4

The following window appears:



Figure 4-5

The system is now ready to start the desired RealtimeDemo. To do so, please push the “green” arrow button as shown below:

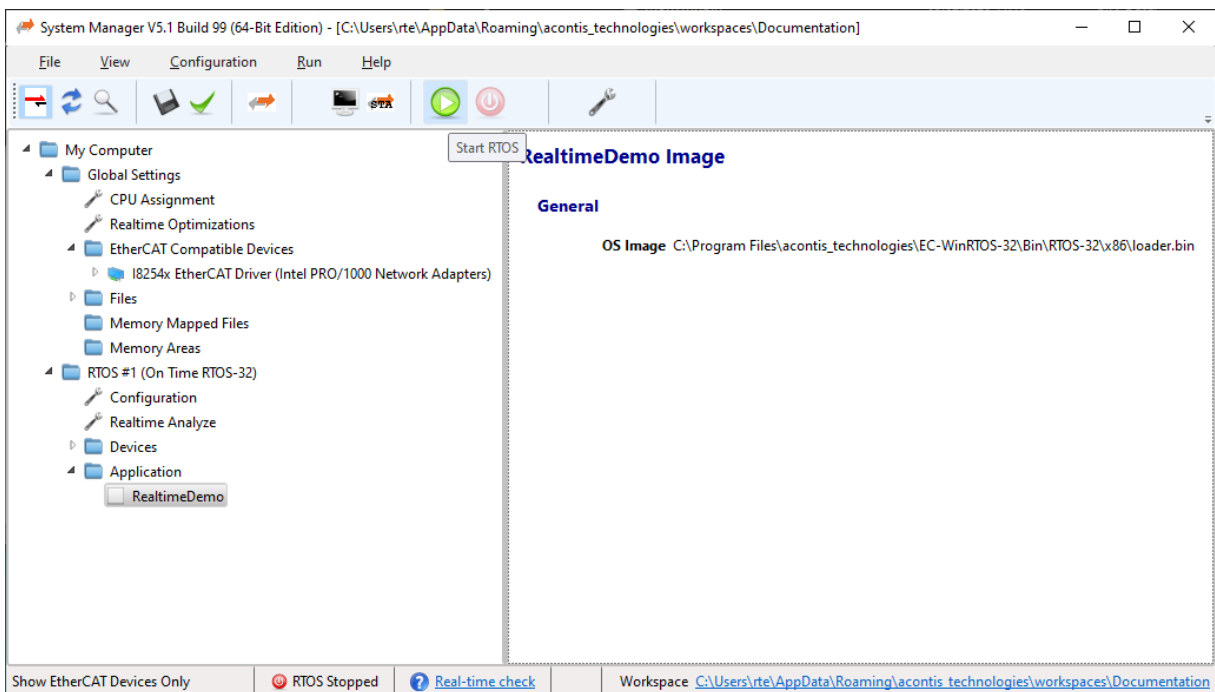
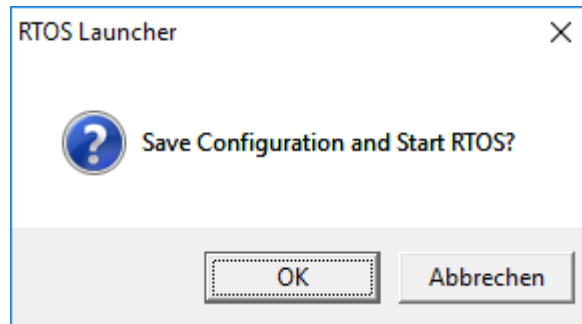


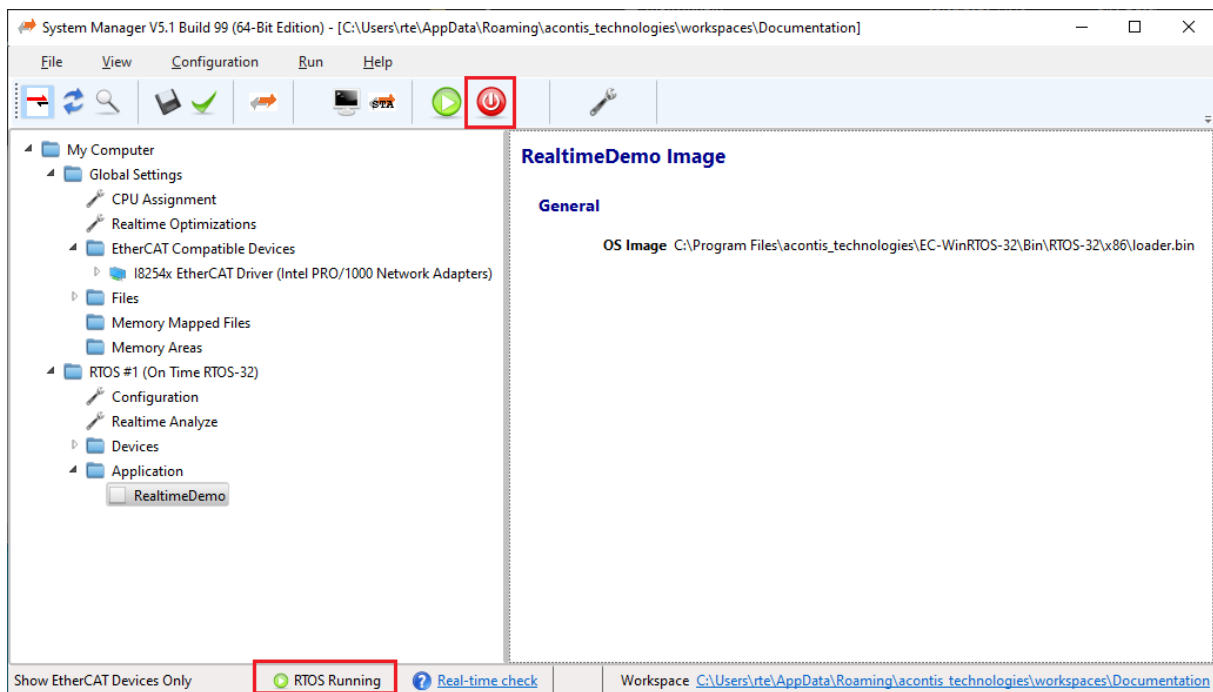
Figure 4-6

After pushing the button, the following message box appears:



**Figure 4-7**

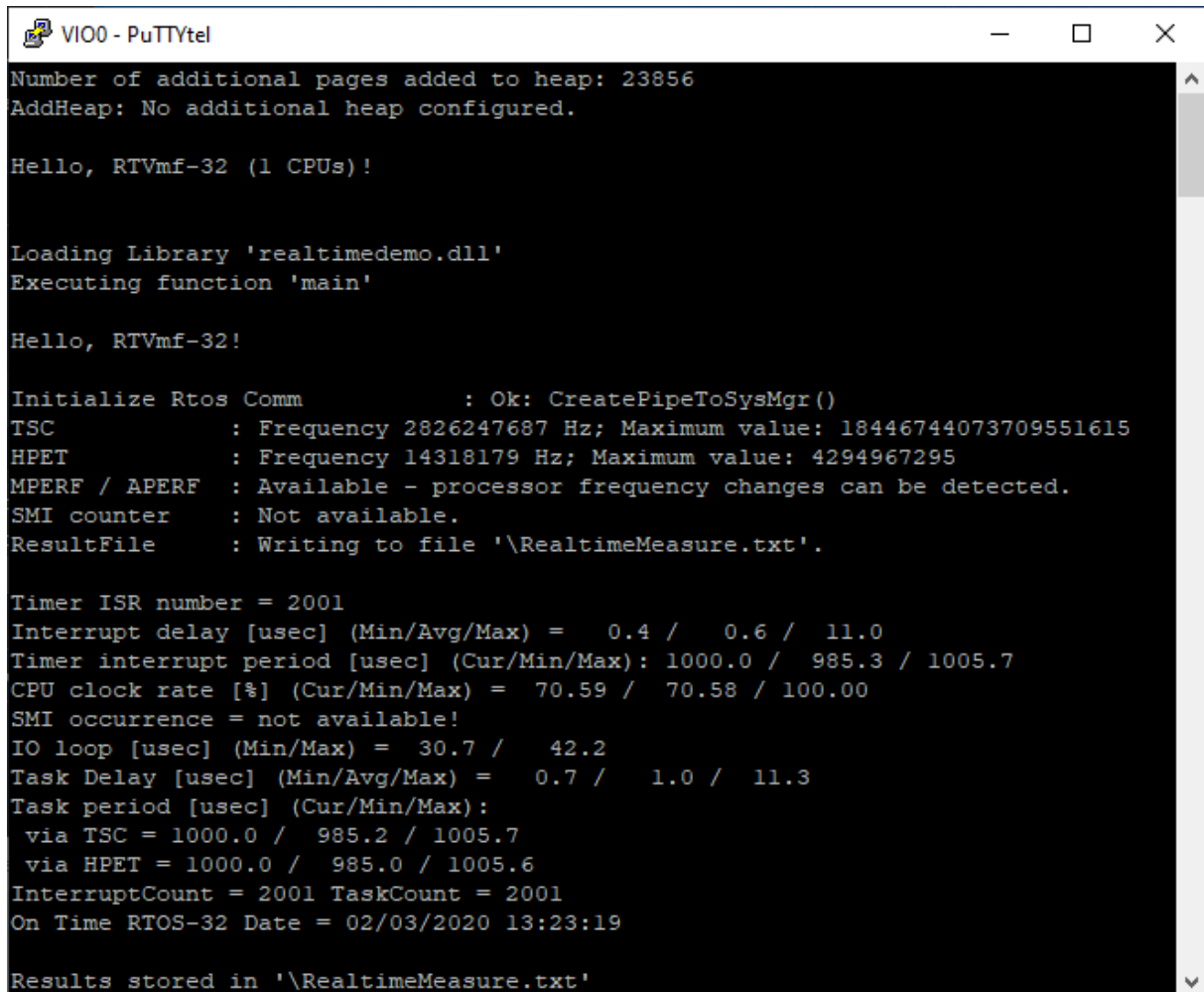
After successful starting of the RTOS-32 presents the following screenshot indicators for a successful start of RTOS-32:



**Figure 4-8**

A RTOS Running will show in the status bar and the Stop Rtos button (red) will be active.

The console output should be like the following:



```
VIO0 - PuTTYtel
Number of additional pages added to heap: 23856
AddHeap: No additional heap configured.

Hello, RTVmf-32 (1 CPUs)!

Loading Library 'realtimedemo.dll'
Executing function 'main'

Hello, RTVmf-32!

Initialize Rtos Comm      : Ok: CreatePipeToSysMgr()
TSC      : Frequency 2826247687 Hz; Maximum value: 18446744073709551615
HPET     : Frequency 14318179 Hz; Maximum value: 4294967295
MPERF / APERF : Available - processor frequency changes can be detected.
SMI counter : Not available.
ResultFile  : Writing to file '\\RealtimeMeasure.txt'.

Timer ISR number = 2001
Interrupt delay [usec] (Min/Avg/Max) =  0.4 /  0.6 / 11.0
Timer interrupt period [usec] (Cur/Min/Max): 1000.0 / 985.3 / 1005.7
CPU clock rate [%] (Cur/Min/Max) = 70.59 / 70.58 / 100.00
SMI occurrence = not available!
IO loop [usec] (Min/Max) = 30.7 / 42.2
Task Delay [usec] (Min/Avg/Max) = 0.7 / 1.0 / 11.3
Task period [usec] (Cur/Min/Max):
  via TSC = 1000.0 / 985.2 / 1005.7
  via HPET = 1000.0 / 985.0 / 1005.6
InterruptCount = 2001 TaskCount = 2001
On Time RTOS-32 Date = 02/03/2020 13:23:19

Results stored in '\\RealtimeMeasure.txt'
```

Figure 4-9

## 5. Shared Memory Demo (build ShmDemo)

Some of the supplied demos are delivered as source code only and therefore the following steps describes all needed steps to get a compiled binary of a source code only demo with support of the System Manager. The following steps are using the ShmDemo as example, but the steps also refer to the other examples in general.

As first step select application node on the RTOS #1 sub node and push the **“Create New Application Project (Debug Only)”** button as shown in the following screenshot:

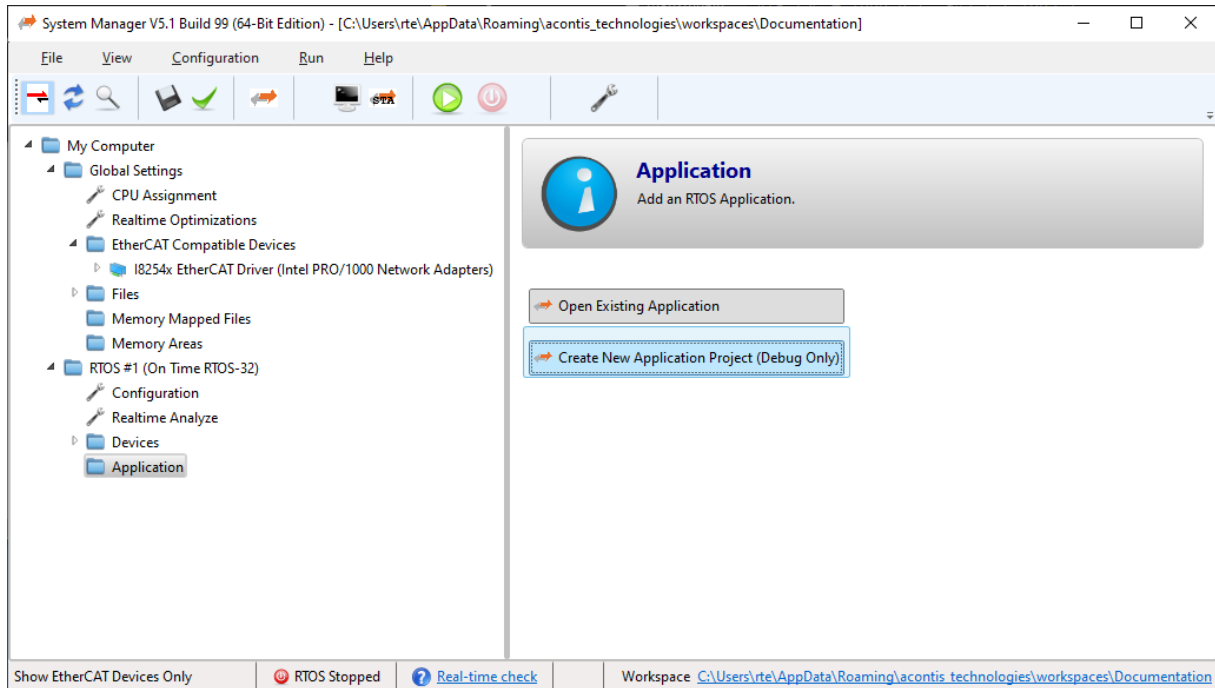


Figure 5-1

After pushing the button, the following dialog appears:

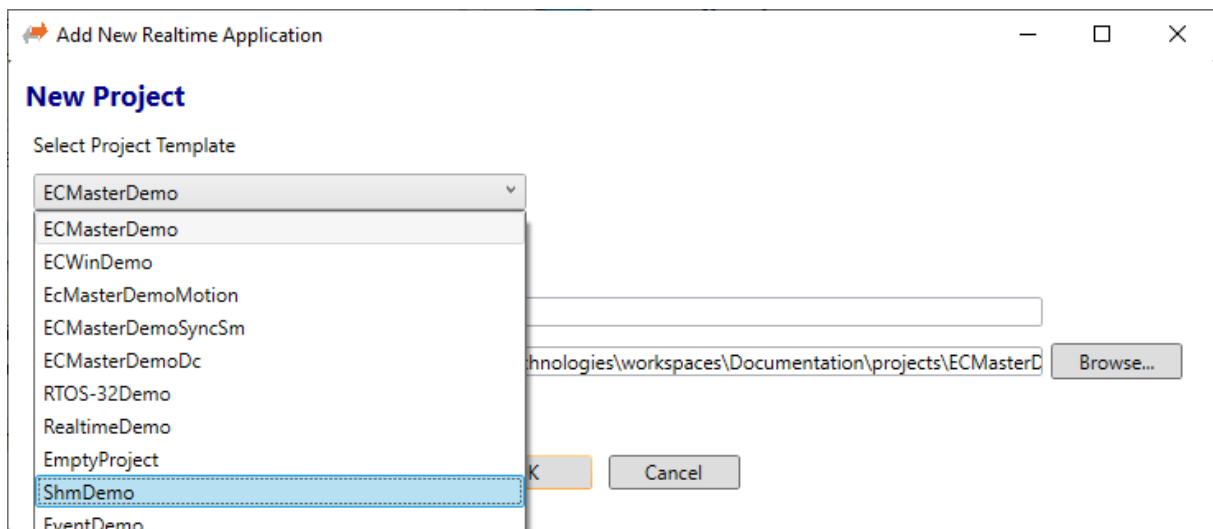


Figure 5-2

Select the ShmDemo and push the OK button. After successful adding of the ShmDemo to the application node, the following “image” should be visible:

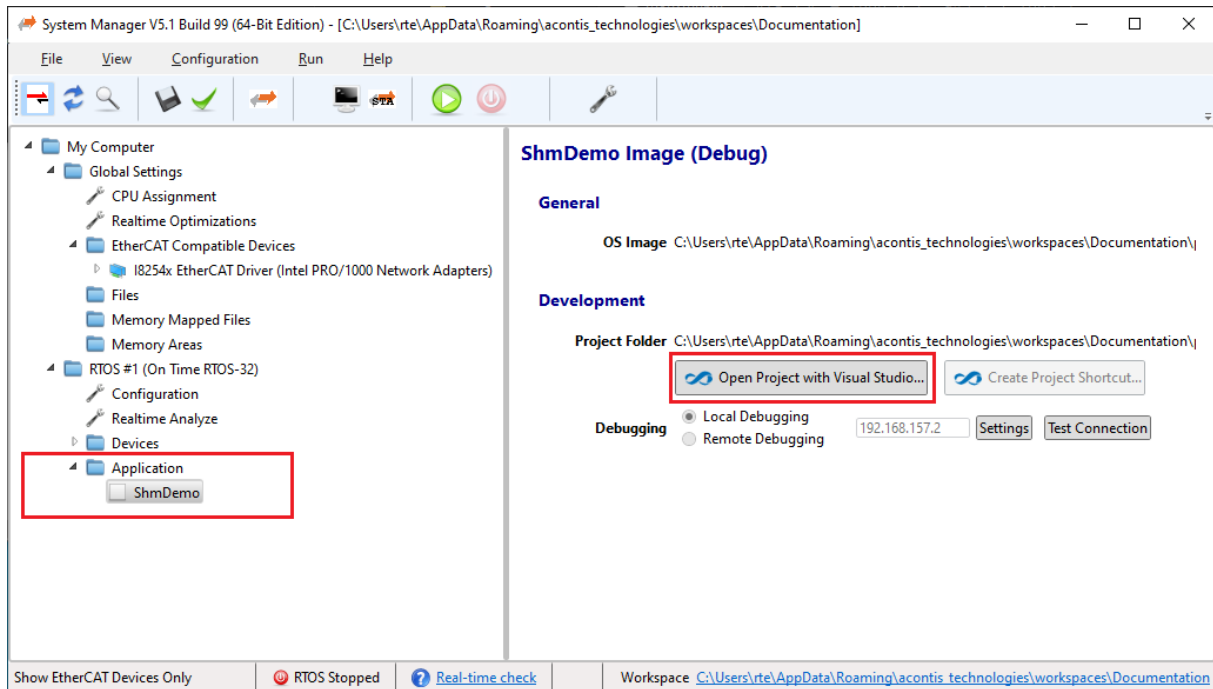


Figure 5-3

Now a new application was created on base of ShmDemo. The next step is to push the “**Open Project with Visual Studio...**” button. After pushing the Button, the following message box appears:

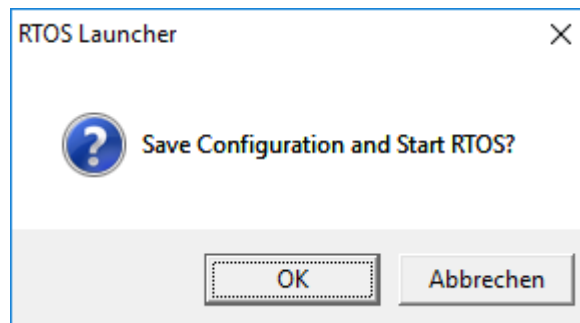


Figure 5-4

After pushing the button, the real-time part starts and if successful Visual Studio starts also and automatically a solution and a project will be created, and all source files added. After successful creation of the project, it is ready to compile and debug the example.

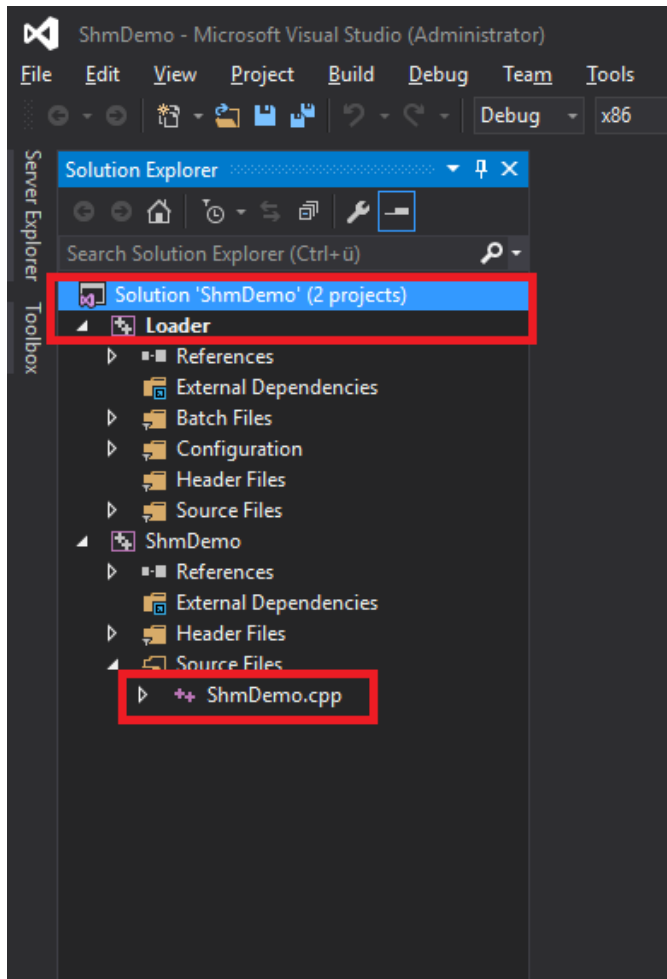


Figure 5-5

**IMPORTANT: THE SHMDEMO REQUIRES A SHARED MEMORY!** Go to node **Global Setting** to sub-node **Memory Areas**. Add a new memory area with the supplied button “Add Memory Area”. See the following screenshot:

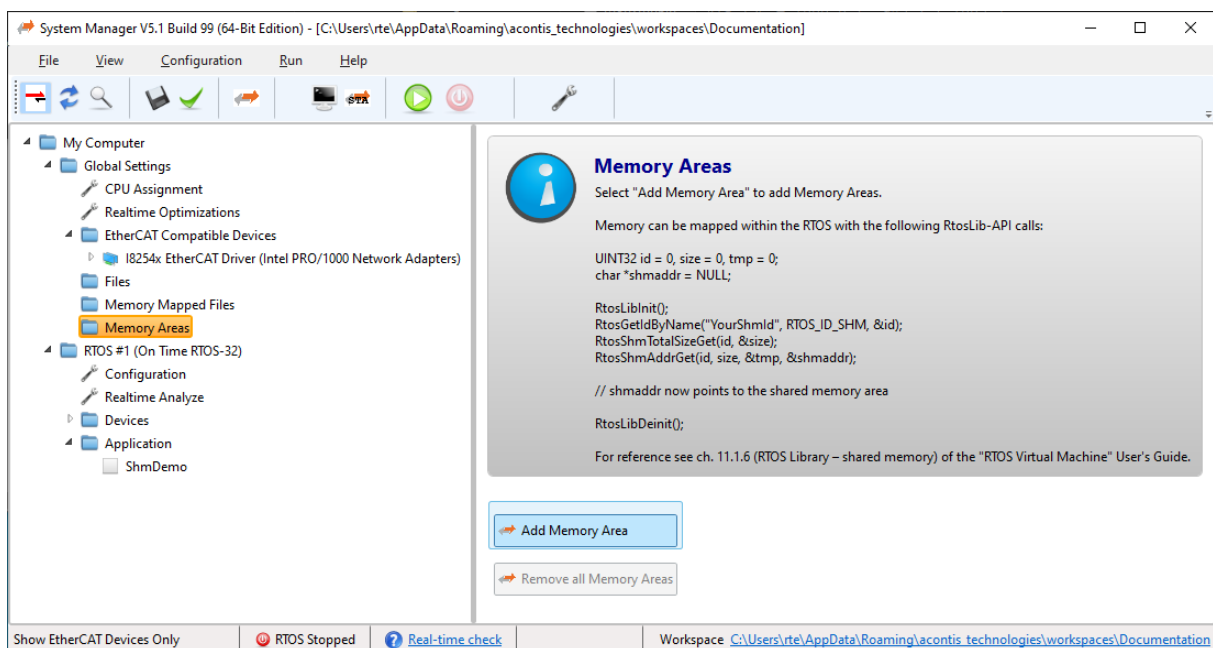


Figure 5-6



*After pressing the button, a shared memory will be created and for successful running, the demo the name of the shared memory must be "UserShm". The name is case-sensitive!*

See the following screenshot:

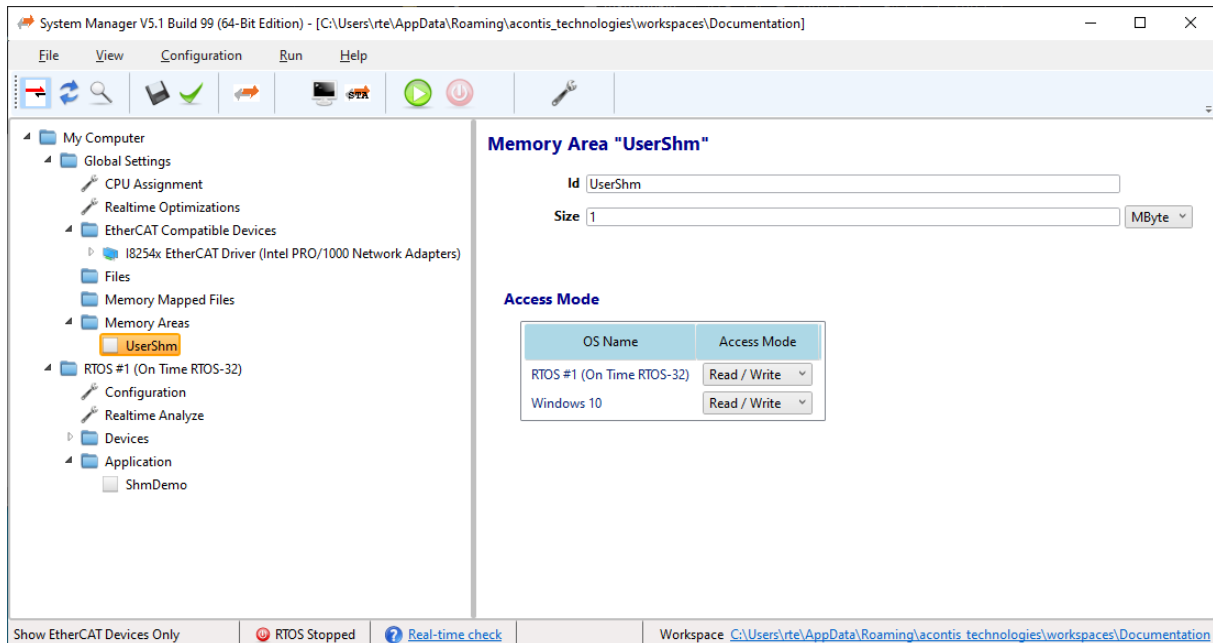


Figure 5-7

The next step requires starting the "Target Console Window". This is not mandatory, but without the console output nothing of the ShmDemo, output will be shown to the user. To start the Target Console Window the monitor button must be pushed as shown in the following screenshot:

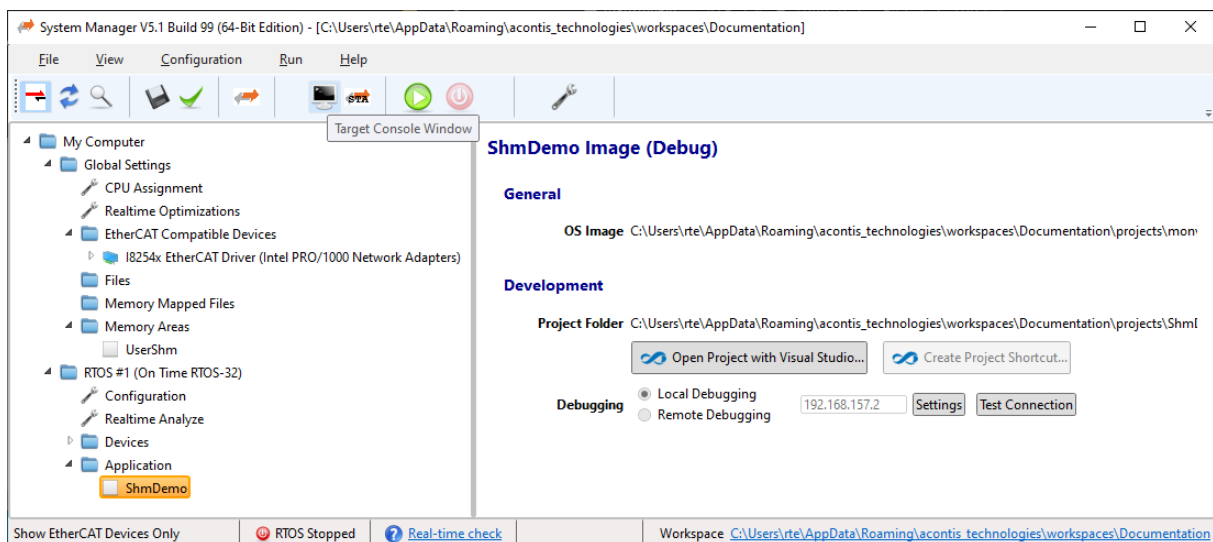


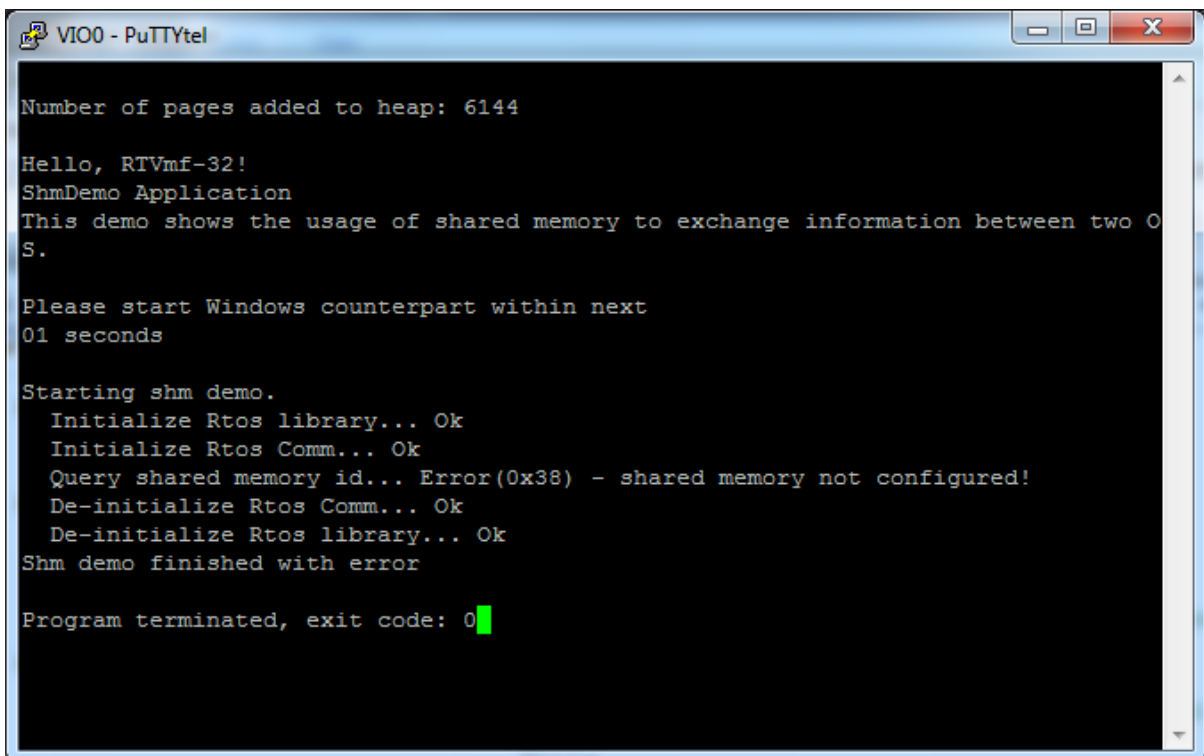
Figure 5-8

The following window appears:



**Figure 5-9**

To start the debugger press <F5>. The debugger connects to the real-time part and start the application. The following output will appear in the console window:



**Figure 5-10**

Reading the output messages leads to the conclusion, that an error occurred, because the Windows counter-part of the demo was not started.

## Windows part of the ShmDemo

To get the windows part of the ShmDemo, a new instance of the Visual must be started and a new project created. For further details, see chapter 8 Building the Real-time Extension Example Programs (Windows part).

- Create a new "Win32 Console Application", select "Console application", "Empty project"

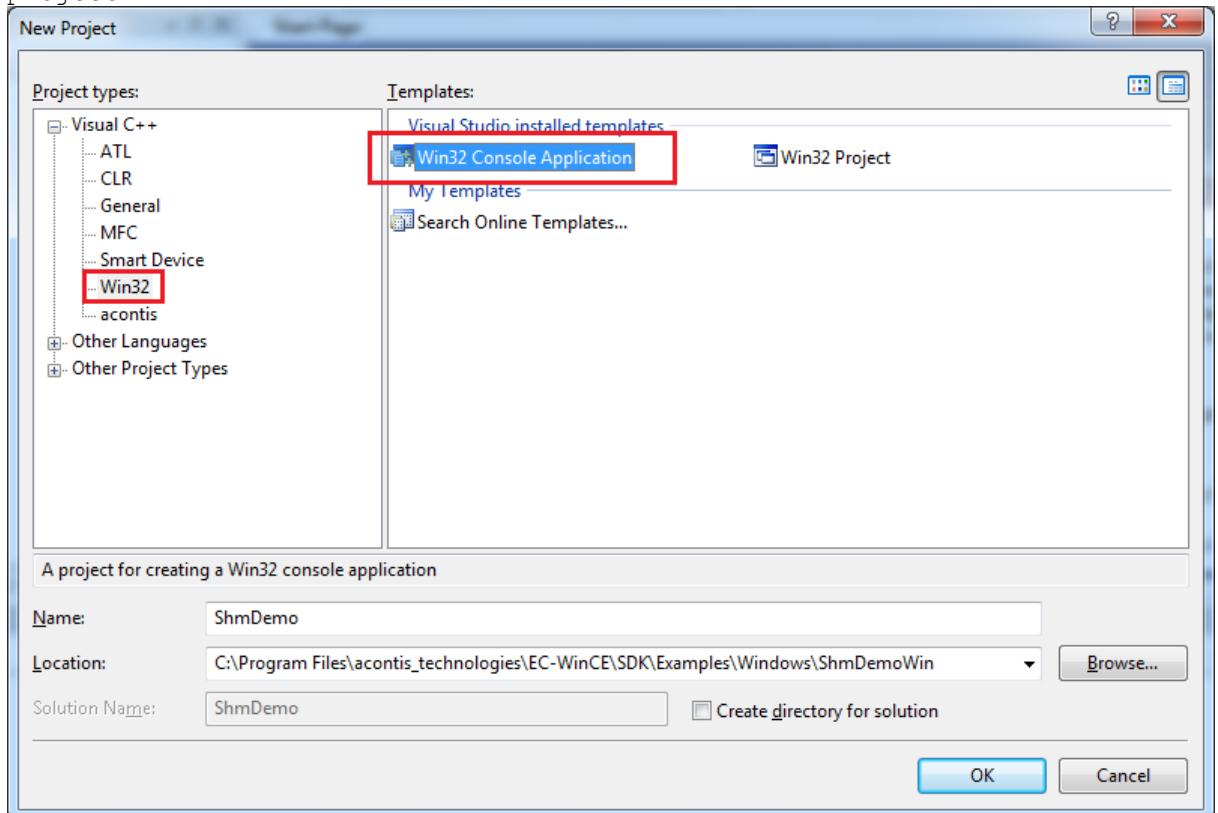


Figure 5-11

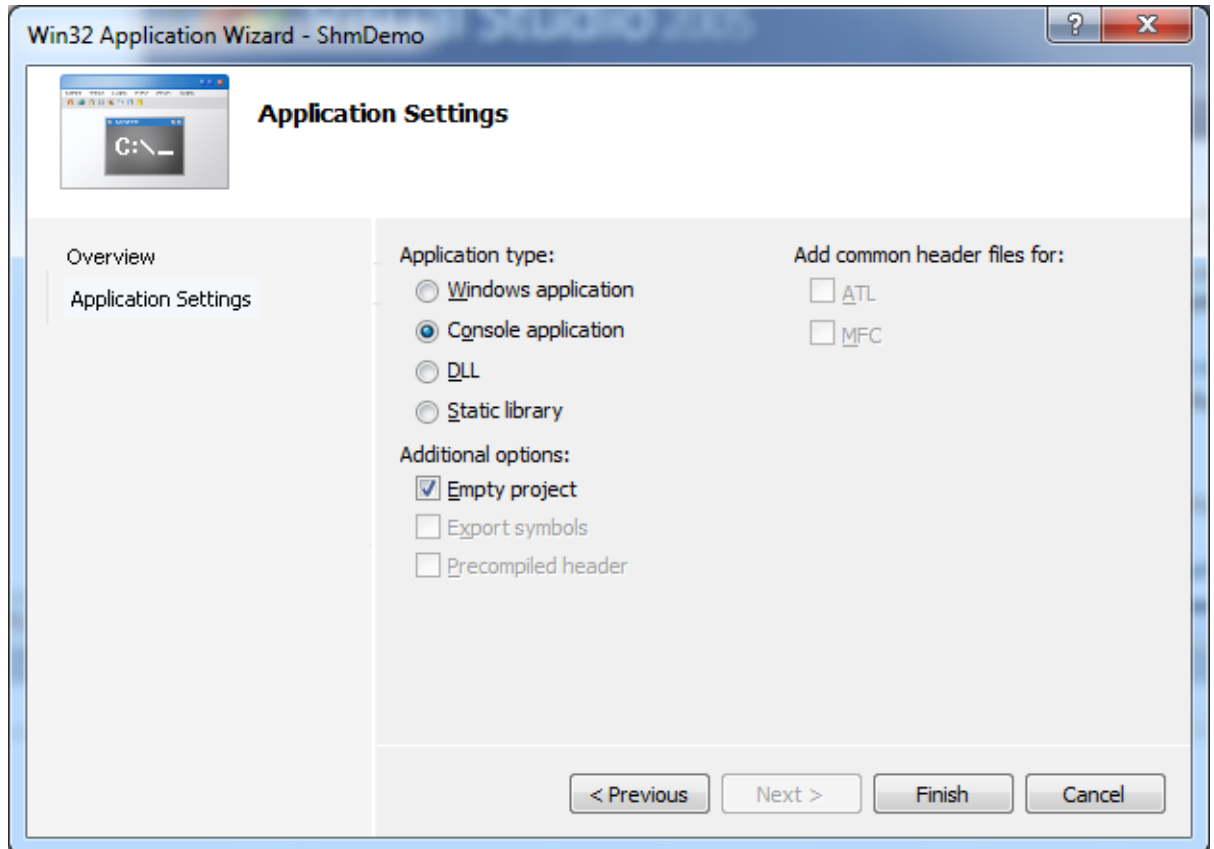


Figure 5-12

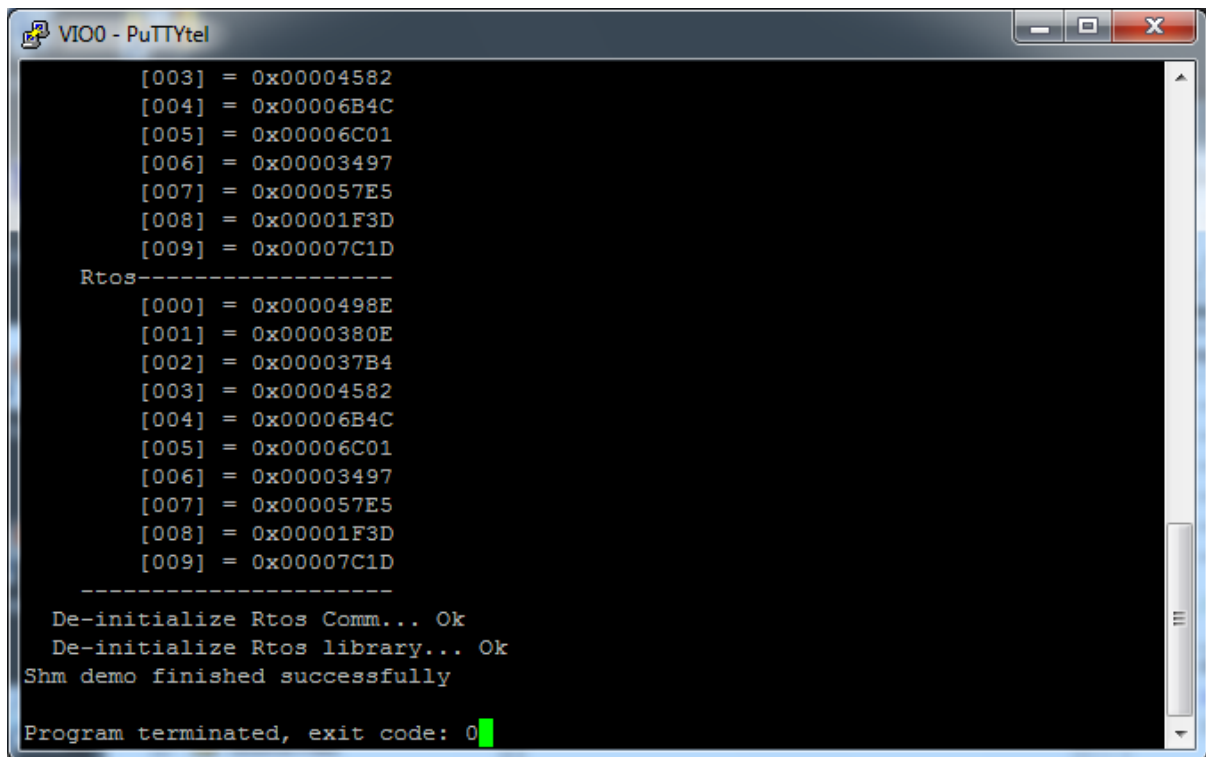
- Add all \*.cpp files located in SDK\Examples\Windows\ShmDemo\ below the real-time extension installation directory to the project.
- Go to Project "Properties" → "C/C++" → "General" → "Additional Include Directories"  
Add SDK\inc and SDK\inc\Windows (below the real-time extension installation directory) as additional include directories.
- Go to Project "Properties" → "Linker" → "General" → "Additional Library Directories"  
Add SDK\LIB\Windows\X86 (below the real-time extension installation directory)
- Go to Project "Properties" → "Linker" → "Input" → "Additional Dependencies" → "..."  
Add "RtosLib32.lib"
- Go to Project "Properties" → "C/C++" → "Language" → "Treat wchar\_t as Built-in Type" select "No"
- Go to Project "Properties" → "C/C++" → "Preprocessor"  
Add RTOS32WIN and \_CRT\_SECURE\_NO\_DEPRECATED to section "Preprocessor Definitions".
- Build the example.
- Run the example

### Running the Demo

After successful compilation of both parts (real-time part and Windows part) of the demo, the following steps are required to start the demo:

- Start the real-time part of the Demo through <F5>
- Start within 10 seconds the Windows part through <F5>

If the demo could run successful, the following output in the console window should appear:



```
[003] = 0x00004582
[004] = 0x00006B4C
[005] = 0x00006C01
[006] = 0x00003497
[007] = 0x000057E5
[008] = 0x00001F3D
[009] = 0x00007C1D
Rtos-----
[000] = 0x0000498E
[001] = 0x0000380E
[002] = 0x000037B4
[003] = 0x00004582
[004] = 0x00006B4C
[005] = 0x00006C01
[006] = 0x00003497
[007] = 0x000057E5
[008] = 0x00001F3D
[009] = 0x00007C1D
-----
De-initialize Rtos Comm... Ok
De-initialize Rtos library... Ok
Shm demo finished successfully
Program terminated, exit code: 0
```

Figure 5-13

## 6. Deployment to a runtime PC

To deploy the settings, configurations from the development PC to a target runtime PC, there are some preparations on development and target PC needed. Please look at the following subchapter.

### On the development PC

- Go to menu **Run|Create Launch Shortcuts** and press <enter>.

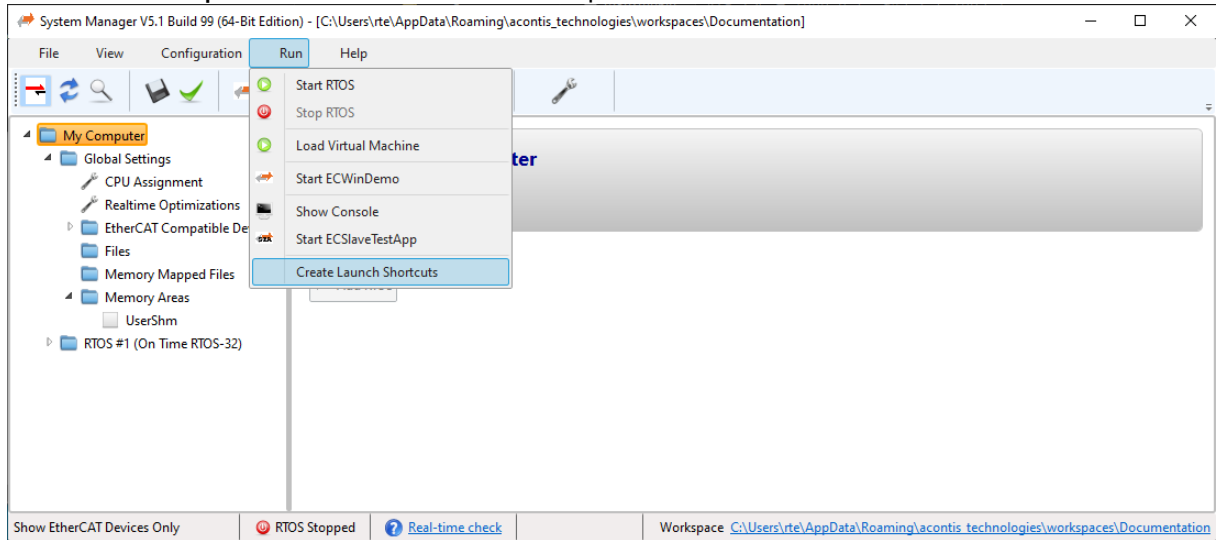


Figure 6-1

- The needed batch files and links will be created into the current active workspace directory.

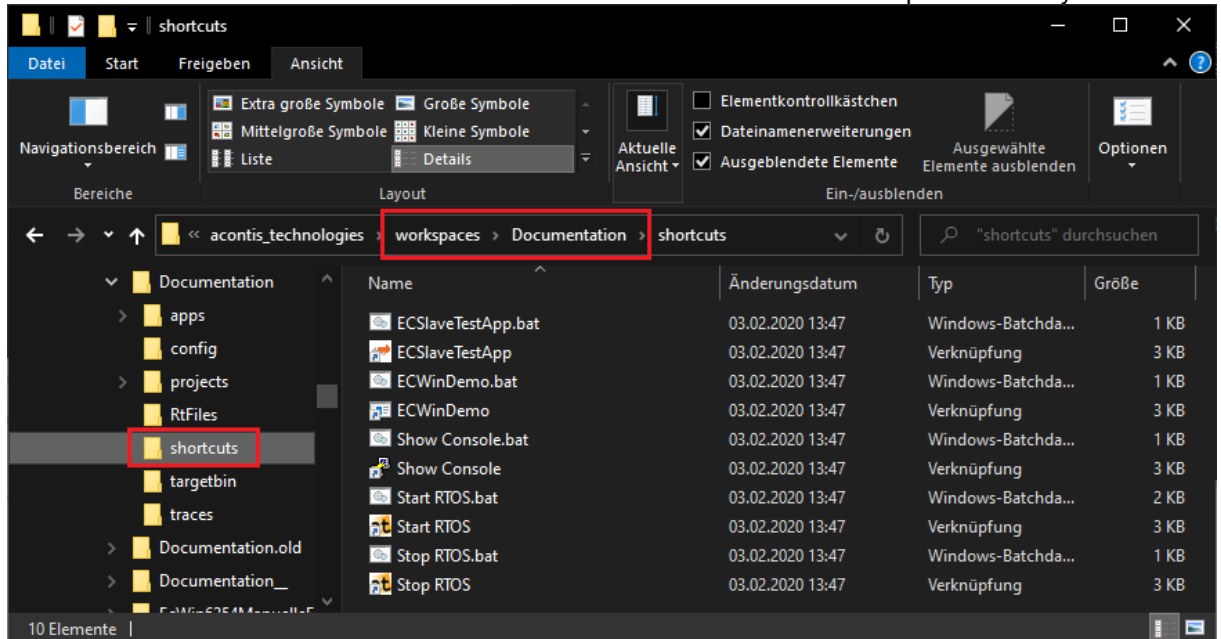


Figure 6-2

- Copy **whole** workspace directory to a location, from which the target PC has access.  
→ Maybe compress whole workspace directory to a zip file. (Optional)

### **On the target PC**

- Install Rtos32Win/EC-Win(RTOS-32) on a target runtime PC
- Copy (decompress) workspace directory to %RTE\_ROOT% directory.
- Optional: move the shortcuts from %RTE\_ROOT%\workspace\default\shortcuts\ to the desktop (if desired).
- To automatically start the real-time extension: add an entry to the auto-start with a call of %RTE\_ROOT%\workspace\default\shortcuts\Start RTOS.bat.

## 7. Device Partitioning

If exclusively the real-time OS (RTOS) should use a device, the System Manager must assign the desired device to the RTOS. Reassignment of hardware in the real-time extension is a two-step process: Isolate the device interrupts and install the RtosPnp device driver. See also the RTOS VM User Manual for more information about real-time device management. The System Manager will do most of the work.

In the following chapter, a network card will be assigned to the RTOS for example.

### 7.1. Real-time extension configuration (for ex. Network Card)

#### 7.1.1. Identify your Network Controller

The System Manager lists an overview over all available/possible network controller and other devices in the current system.

Please apply the following to identify a suitable/compatible network controller:

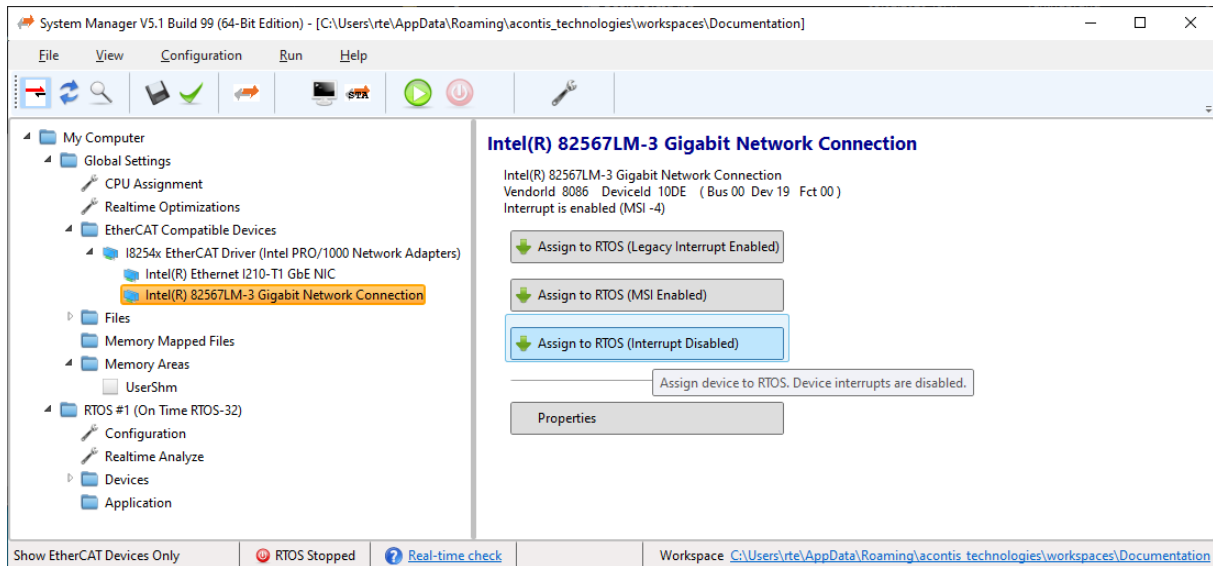
1. Open the System Manager.
2. Expand the **Global Settings**.
3. Expand the **Devices**.
4. Select the desired network adapter (if a compatible network adapter is available!)
5. Right click of the desired network adapter and select *Assign to RTOS (Interrupt disabled)*.
6. Please see Figure 7-1.

**Hint:** By using this quick start guide with EC-Win(RTOS-32) it's possible with toggling of the EtherCAT icon only devices which are supported by the EC-Master to show. The default is to show all devices of this PC (graphics adapter card, HDD, etc.)

---



Assign with Interrupts Disabled:



**Figure 7-1 (System Manager)**

*Successfully assigned RtosPnp Device Driver will be listed under the Real-time OS Devices tree node under the node “Device Configuration”.*

**WARNING: DURING THE INSTALLATION OF THE RTOS DRIVER OF THE SELECTED NETWORK CARD, WINDOWS COULD THROW SEVERAL POP-UP WINDOWS.**

**WARNING: DURING INSTALLATION OF THE RTOS DRIVER OF THE SELECTED NETWORK CARD A REBOOT COULD BE NEEDED. IN THAT CASE START THE SYSTEM MANAGER AFTER THE REBOOT AGAIN AND CHECK IF THE DESIRED NETWORK CARD IS ASSIGNED CORRECTLY TO THE RTOS.**

## 8. Building the Real-time Extension Example Programs (Windows part)

### 1. Build instructions for Windows 7 / 8 / 10

This section offers building instructions for the shipped example programs in **SDK\Examples\Windows** (below the real-time extension installation directory).

Each directory is an own example program Please look in the appropriate folders under subdirectory /SDK/Examples/\* of the installation directory.

- Open Visual Studio 2005
- Create a new "Win32 Console Application", select "Console application", "Empty project"
- Add all \*.cpp files located in SDK\Examples\Windows\<Demo-Name> below the real-time extension installation directory to the project.
- Go to Project "Properties" → "C/C++" → "General" → "Additional Include Directories"  
Add SDK\inc and SDK\inc\Windows (below the real-time extension installation directory) as additional include directories.
- Go to Project "Properties" → "Linker" → "General" → "Additional Library Directories"  
Add SDK\LIB\Windows\X86 (below the real-time extension installation directory)
- Go to Project "Properties" → "Linker" → "Input" → "Additional Dependencies" → "..."  
Add "RtosLib32.lib"
- Go to Project "Properties" → "C/C++" → "Language" → "Treat wchar\_t as Built-in Type" select "No"
- Go to Project "Properties" → "C/C++" → "Preprocessor"  
Add RTOS32WIN and \_CRT\_SECURE\_NO\_DEPRECATED to section "Preprocessor Definitions".
- Build the Example.
- Run the Example.

*Note that for EventDemo, InterlockDemo and ShmDemo a counterpart executable on the RTOS-32 side must be started (identically named; see source code below SDK\Examples\RTOS-32) for proper operation!*

## II. Appendix

### 1. Adding User settings in a System Manager configuration

In a System Manager controlled environment, all user settings shall be stored in the file `systemmgr_user.config` that is located below the workspace's config sub directory.

### 2. Technical Background of System Manager

There are some technical notes, which the user should know about the System Manager.

The most important fact in this direction is the use of a workspace directory. All modified, added, etc. files, data, etc. are located in the workspace. At first start of the System Manager the workspace directory will be created (default: `C:\Users\'username'\AppData\Roaming\aconis technologies\workspaces\` or user-defined location) and all configuration files, etc. will be copied therein. Thus, the System Manager operates on file copies and it **never** change the original files. *As a side effect of this, after deleting of the workspace directory, the start of System Manager will be like the first step before.*

All compiled applications and associated solution/project files will be found under `C:\Users\'username'\AppData\Roaming\aconis technologies\workspaces\default\projects\<ProjectName>\` if the user does not choose a user-defined location of the workspace. The used original source code file(s) of the examples reside furthermore in the examples directory in the SDK (`%RTE_ROOT%\SDK\Examples\RTOS-32\*`).

A further important notice is the fact, that the System Manager uses the newest available Visual Studio, which is available at the development PC. This means, if for ex. Visual Studio 2005 and Visual Studio 2012 are installed; Visual Studio 2012 will be preferred and automatically selected. To change the preferred Visual Studio, a manual change needed in `usersettings.xml`. This `usersettings.xml` will be found at `%RTE_ROOT%`.