FAQ 260

Migrating a Hand coded Application into an S-Function

Keywords

C code, hand-coded application, RTI generated code, C-coded S-Function, main(), init(), RTLIB_BACKGROUND_SERVICE, endless loops, global_enable, RTLIB_INT_ENABLE, RTLIB_SRT_ISR_BEGIN(), RTLIB_SRT_ISR_END(), RTLIB_SRT_START(), install_phs_int_vector(), deinstall_phs_int_vector(), host_service(), mdlStart(), mdlInitializeConditions(), mdlOutputs

Question

Suppose that you have implemented a hand-coded application (no RTI generated code) for your dSPACE hardware. The application works fine without any problems. Now you intend to migrate the functionality of the hand-coded application into a Simulink model by means of a C-coded S-Function or user C code. Please find the following guidelines

Solution

When you are using RTI, the following **function** and **macros** are called automatically in the simulation frame. You must not call these functions and macros inside an C-coded S-function or user C-code.

- **void main(void):**
- **init()**
- **endless loop** with or without a call to **RTLIB_BACKGROUND_SERVICE()**
- **global_enable()** or **RTLIB_INT_ENABLE()**
- **RTLIB_SRT_ISR_BEGIN()** and **RTLIB_SRT_ISR_END()** or equivalent function calls which check for an overrun in the interrupt service routine (e.g. ds1005_begin_isr_timerA())

The following macros and function will be called if specific RTI blocks are part of the Simulink model.

- **RTLIB_SRT_START() or equivalent function calls which install a function as an interrupt service routine for a specific timer source (e.g. ds1005_start_isr_timerA()):** Please place the S-Function to the Simulink model and decide if the fixed step time is correct for the execution of the S-Function. Otherwise use atomic subsystems or a function-call generator to specify a sample time multiple of the fixed step size for calling the S-Function. Alternatively the RTI Timer Interrupts can be used instead when no integer multiplies of the fixed step size have to implemented for calling the S-Function.
- **install_phs_int_vector(), deinstall_phs_int_vector() or specific macros for enabling or disabling I/O interrupts:** The interrupts from the I/O boards are integrated to Simulink via different RTI hardware interrupt blocks. The RTI hardware interrupt blocks are available from the related RTI I/O block libraries. Please connect the output port of the RTI hardware interrupt blocks to a function-call subsystem und place the S-Function inside the function-call subsystem.
- **host_service(…)**: With RTI the data capturing (default behaviour) is realized via an internal call to host_service() inside the fastest timer task. If a data capturing with a different sample time is intended, place the Data Capture block of the RTI Extras block library either inside the timer task or the function-call subsystem. With user C code it is not possible to handle asynchronous executed code. User C code is called synchronously with the base sample time of the model.

The following points show where to place code from the hand-coded application in the S-Function or user C code.

Please use the templates for C-coded S-Function or for user C code referenced in the dSPACE HelpDesk chapters *Inserting Custom C Code > Implementing S-Functions* or *Inserting Custom C Code > Implementing User-Code* of the *RTI and RTI-MP Implementation Guide* as a starting point:

- If you need to initialize I/O devices that are not initialized by init(), use the specific initialization functions mdlStart() and/or mdlInitializeConditions() (S-Function) or usr_initialize() (user C code).

- The code of an interrupt service routine (ISR) triggered periodically has to be placed inside mdlOutputs() (S-Function) or the usr_output(), usr_sample_input() or usr_input() (user C-code). The S-Function block has to be placed inside the timer task identified by a certain sample time.
  The code of an interrupt service routine (ISR) triggered asynchronously by a hardware interrupt has to be placed inside the mdlOutputs() (S-Function), too. The S-Function block has to be placed inside the function-call subsystem triggered by a specific RTI hardware interrupt block. The used C code does not provide any functionality for asynchronous execution. Alternatively the Real-Time Workshop Custom Code blocks System Outputs can be used to place code inside an asynchronously triggered subsystem (type ‘custcode’ at the MATLAB Command window to open the Custom code library of the Real-Time Workshop).

- If you need to call specific code (e.g. termination values of output devices) when the simulation is stopped, use the specific termination functions mdlTerminate() (S-Function) or usr_terminate() (user C code). The mentioned functions are executed once whenever the SimState variable is changed to STOP mode.

- If you need to call specific code in the background task, use the specific user C code function usr_background().
  The S-Function template does not provide a specific function to be called inside the background task. As an alternative for S-Functions place the S-Function block inside a function-call subsystem triggered by a RTI Background block from the RTI Extras block library. The code of mdlOutputs() is executed in the background task.

- It is possible to define global variables in the S-function and user C code. The global variables can be monitored and modified in ControlDesk if they have been defined in an additional user variable description file (USR.TRC File).

Related and additional information to the mentioned issues can be found in the following chapters of the *RTI and RTI-MP Implementation Guide*

- *Inserting Custom C Code > Implementing S-Functions > Purpose of the C Methods*

- *Inserting Custom C Code > Implementing User-Code > How to Handcraft the User-Code*

- *Handling Tasks > Assigning Parts of the Model to the Background Task* and *RTI and RTI-MP Implementation Reference:*

- *RTI and RTI-MP File Reference > File Details > User‐ Code File (USR.C File)*

- *RTI and RTI-MP File Reference > File Details > User Variable Description File (USR.TRC File)*

Related FAQs

- FAQ 202 – C S-Function Techniques

- FAQ 255 – Implementing C coded S-Functions for RTI

How to Contact dSPACE Support

dSPACE GmbH
Rathenaustr. 26
D-33102 Paderborn
Germany

++49 5251 1638-941

mailto:support@dspace.de
http://www.dspace.com/support

dSPACE recommends that you use the support request form on the internet to contact dSPACE support.
It is available under

- http://www.dspace.com/goto?supportrequest

Software Updates and Patches

dSPACE strongly recommends that you download and install the most recent patches for your current dSPACE installation. Visit http://www.dspace.com/goto?patches for software updates and patches.

FAQ

FAQ documents are available under http://www.dspace.com/goto?faq.

Important Notice