Renesas Synergy™ Platform

# USBX™ Host Class Video Module Guide

R11AN0345EU0100
Rev.1.00
Oct 29, 2018

## Introduction

This module guide will enable you to effectively use a module in your own design. Upon completion of this guide, you will be able to add this module to your own design, configure it correctly for the target application, and write code using the included application project code as a reference and efficient starting point. References to more detailed API descriptions and suggestions of other application projects that illustrate more advanced uses of the module are available on the Renesas Synergy™ Knowledge Base (as described in the References section at the end of this document) and should be valuable resources for creating more complex designs.

The USBX™ Host Class Video module is a high-level API for USBX Host Class Video applications and is implemented on `g_ux_host_class_video`. The USBX Host Class Video module configures the USBX Host Class Video Source, USBX Host Configuration, USBX Source, and USBX Port Host Controller Device. The USBX Host Class Video module uses the USB peripheral on the Synergy MCU.

## Contents

## 1. USBX Host Class Video Module Features

The video class oversees communication to the USB camera. It includes the following features:

- Supports either collect information on the video format, resolution, and frame rate of connected USB camera.
- Supports either start and stop video streaming by the specified video format, resolution, and frame rate.

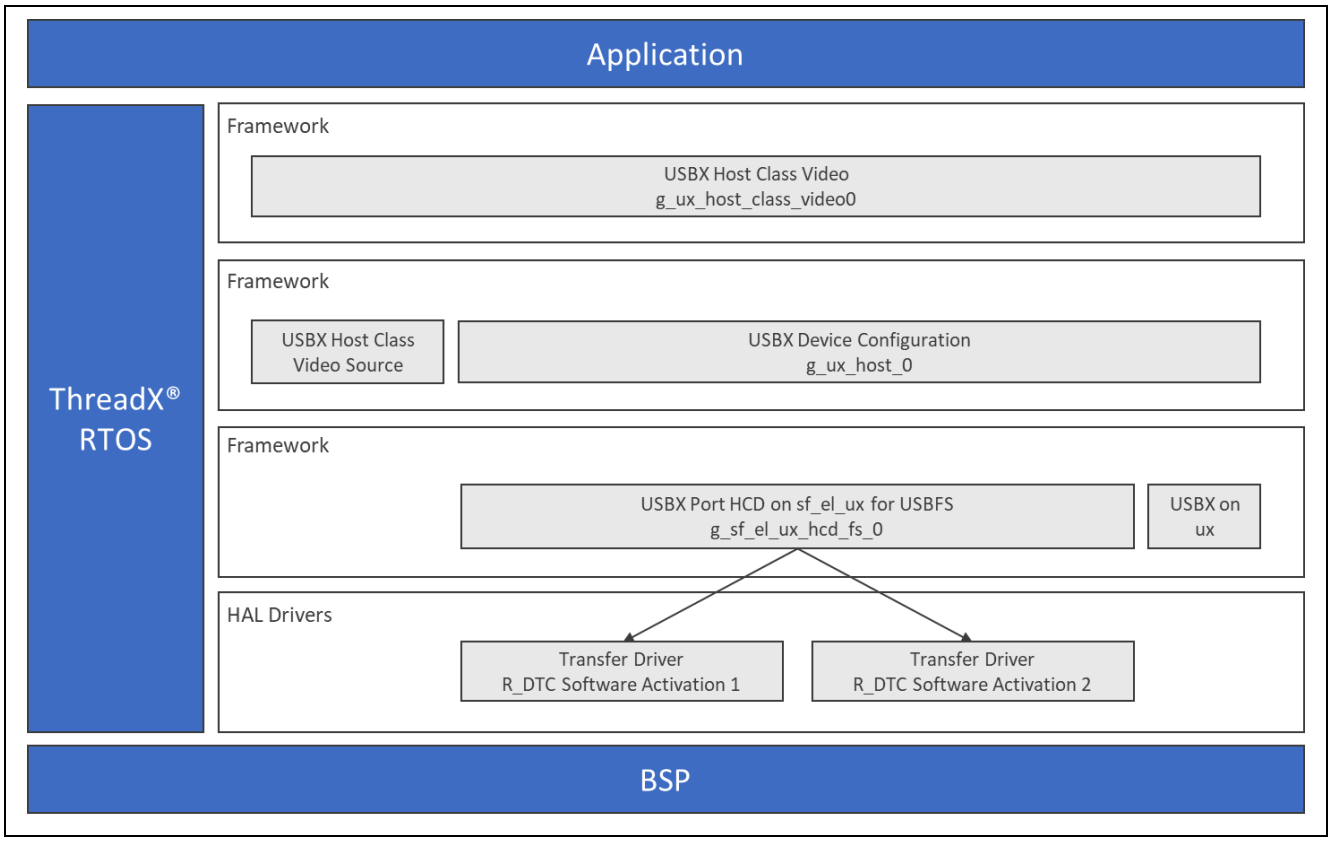


**Figure 1 USBX Host Class Video Module Organization, Options, and Stack Implementations**

## 2. USBX Host Class Video Module APIs Overview

The USBX Host Class Video Module defines APIs for start, stop, frame parameters set, max payload get, buffer add, and callback set. A complete list of the available APIs, an example API call, and a short description of each can be found in Table 1. A table of status return values is shown in Table 2.

**Table 1 USBX Host Class Video Module API Summary**

| Function Name | Example API Call and Description |
|---|---|
| ux_host_class_video_start | `status = ux_host_class_video_start(video);`<br>This function starts the video streaming. The video channel needs to be properly configured prior to calling this function. |
| ux_host_class_video_stop | `status = ux_host_class_video_stop(video);`<br>This service stops the current video channel. |
| ux_host_class_video_frame_parameters_set | `status = ux_host_class_video_frame_parameters_set(video, frame_format, width, height, frame_interval);`<br>This function sets the video parameters for the video device. |
| ux_host_class_video_max_payload_get | `length = ux_host_class_video_max_payload_get(video);`<br>This function returns the maximum payload size for a given video parameter setting. After properly configures the video streaming parameters (such as video encoding, resolution, |

| | |
|---|---|
| | frame rate), application may use this function to obtain the maximum payload size. With the maximum payload size, application can allocate memory buffers for receiving incoming video frame data. |
| ux_host_class_video_transfer_buffer_add | `Status = ux_host_class_video_transfer_buffer_add(video, buffer);`<br>This function passes a buffer to the video device, which is used to store incoming video stream data. The size of the buffer must be at least the maximum of the video payload size, which can be obtained by calling ux_host_class_video_max_payload_get. |
| ux_host_class_video_transfer_callback_set | `ux_host_class_video_transfer_callback_set(video, callback_function);`<br>This function sets the video transfer callback function. This callback function is invoked once a transfer request has been fulfilled, and the application is ready to consume the video data. |

Note: For more detailed descriptions of operation and definitions for the function data structures, typedefs, defines, API data, API structures, and function variables, review the associated Express Logic User's Manual accessible as described in the Reference section later in this document.

**Table 2 Status Return Values**

| Name | Description |
|---|---|
| UX_SUCCESS | The data transfer was completed. |
| UX_TRANSFER_TIMEOUT | Transfer timeout, reading/writing not completed. |
| UX_MEMORY_INSUFFICIENT | Not enough memory. |
| UX_HOST_CLASS_INSTANCE_UNKNOWN | The video instance is not valid. |
| UX_HOST_CLASS_VIDEO_PARAMETER_ERROR | The desired video parameters are not supported by this camera. |

Note: Lower level drivers may return Common Error Codes. Refer to the *SSP User's Manual* API References for the associated module for a definition of all relevant status return values.

## 3. USBX Host Class Video Module Operational Overview

**Initialization of USBX resources**

The USBX has its own memory manager. The memory needs to be allocated to the USBX before the host or device side of the USBX is initialized. The USBX memory manager can accommodate systems where memory can be cached.

**Definition of USB Host Controllers**

It is required to define at least one USB host controller for USBX to operate in host-mode. The application-initialization file should contain this definition. SSP defines USB host controller when USB host controller driver is added to thread stacks.

**Definition of Device Classes**

It is required to define one or more device classes(s) with the USBX. A USB class is required to drive a USB device after the USB stack has configured the USB device. A USB class is very specific to the device; one or more classes may be required to drive a USB device depending on the number of interfaces contained in the USB device descriptors.

**USB Class Binding**

When the device is configured, the topology manager will let the class manager continue the device discovery by looking at the device-interface descriptors. A device can have one or more interface descriptors.

An interface represents a function in a device. For instance, a USB speaker has three interfaces, one for audio streaming, one for audio control, and one to manage the various speaker buttons.

The class manager has two mechanisms to join the device interface(s) to one or more classes. It can either use the combination of a PID/VID (product ID and vendor ID) found in the interface descriptor or the combination of Class/Subclass/Protocol.

The PID/VID combination is valid for interfaces that cannot be driven by a generic class. The Class/Subclass/Protocol combination is used by interfaces that belong to a USB-IF certified class such as a printer, hub, storage, audio, or Human Interface Design (HID).

The class manager contains a list of registered classes from the initialization of the USBX. The class manager will call each class one-at-a-time until one class accepts to manage the interface for that device; each class can only manage one interface. In the case of the USB audio speaker, the class manager will call all the classes for each of the interfaces.

Once a class accepts an interface, a new instance of that class is created; the class manager will then search for the default alternate setting for the interface. A device may have one or more alternate settings for each interface. The alternate setting 0 will be the one used by default until a class decides to change it.

For the default alternate setting, the class manager will mount all the endpoints contained in the alternate setting. If the mounting of each endpoint is successful, the class manager will complete its job by returning to the class that will finish the initialization of the interface.

## 3.1 USBX Host Class Video Module Important Operational Notes and Limitations

### 3.1.1 USBX Host Class Video Module Operational Notes

- When the User Callback for Host Event Notification is called, check class name and save the instance.
- Set valid camera parameters and set a transfer request done callback.
- After starting the camera, add buffers.
- When the camera sends a frame data, transfer request done callback is called.
- In the user application, it is necessary to add the buffer again each time frame data is received.

### 3.1.2 USBX Host Class Video Module Limitations

- The module needs the interrupt of a USB Controller enabled.
- The module uses the interrupt of a USB Controller. Set appropriate interrupt-priority level in the Synergy Configuration tool for proper operation.
- The module uses the interrupt of a transfer module (implemented as DMAC or DTC) if one is implemented. Set the appropriate priority level in the Synergy Configuration tool. The priority level must be higher than that of the USB Controller for proper operation.
- Refer to the most recent *SSP Release Note* for any additional operational limitations for this module.

# 4. Including the USBX Host Class Video Module in an Application

This section describes how to include the USBX Host Class Video module in an application using the SSP configurator.

Note: This section assumes you are familiar with creating a project, adding threads, adding a stack to a thread, and configuring a block within the stack. If you are unfamiliar with any of these items, refer to the first few chapters of the *SSP User's Manual* to learn how to manage each of these important steps in creating SSP-based applications.

To add the USBX Host Class Video module to an application, simply add it to a thread using the stacks selection sequence given in Table 3. (The default name for the USBX Host Class Video module is g_ux_host_class_video0. This name can be changed in the associated **Properties** window.)

**Table 3 USBX Host Class Video Selection Sequence**

| Resource | ISDE Tab | Stacks Selection Sequence |
|---|---|---|
| g_ux_host_class_video0 USBX Host Class Video | Threads | New Stack> X-Ware™> USBX> Host > Classes > Video > USBX Host Class Video |

When the USBX Host Class Video module is added to the thread stack as shown in the following figure, the configurator automatically adds any needed lower-level modules. Any modules that need additional configuration information will have box text highlighted in Red. Modules with a Gray band are individual modules that stand alone. Modules with a Blue band are shared or common. They need only be added once and can be used by multiple stacks. Modules with a Pink band can require the selection of lower-level modules; these are either optional or recommended. (This is indicated in the block with the inclusion of this text.) If the addition of lower-level modules is required, the module description will include Add in the text. Clicking on any Pink banded modules will bring up the **New** icon and then display the possible choices.

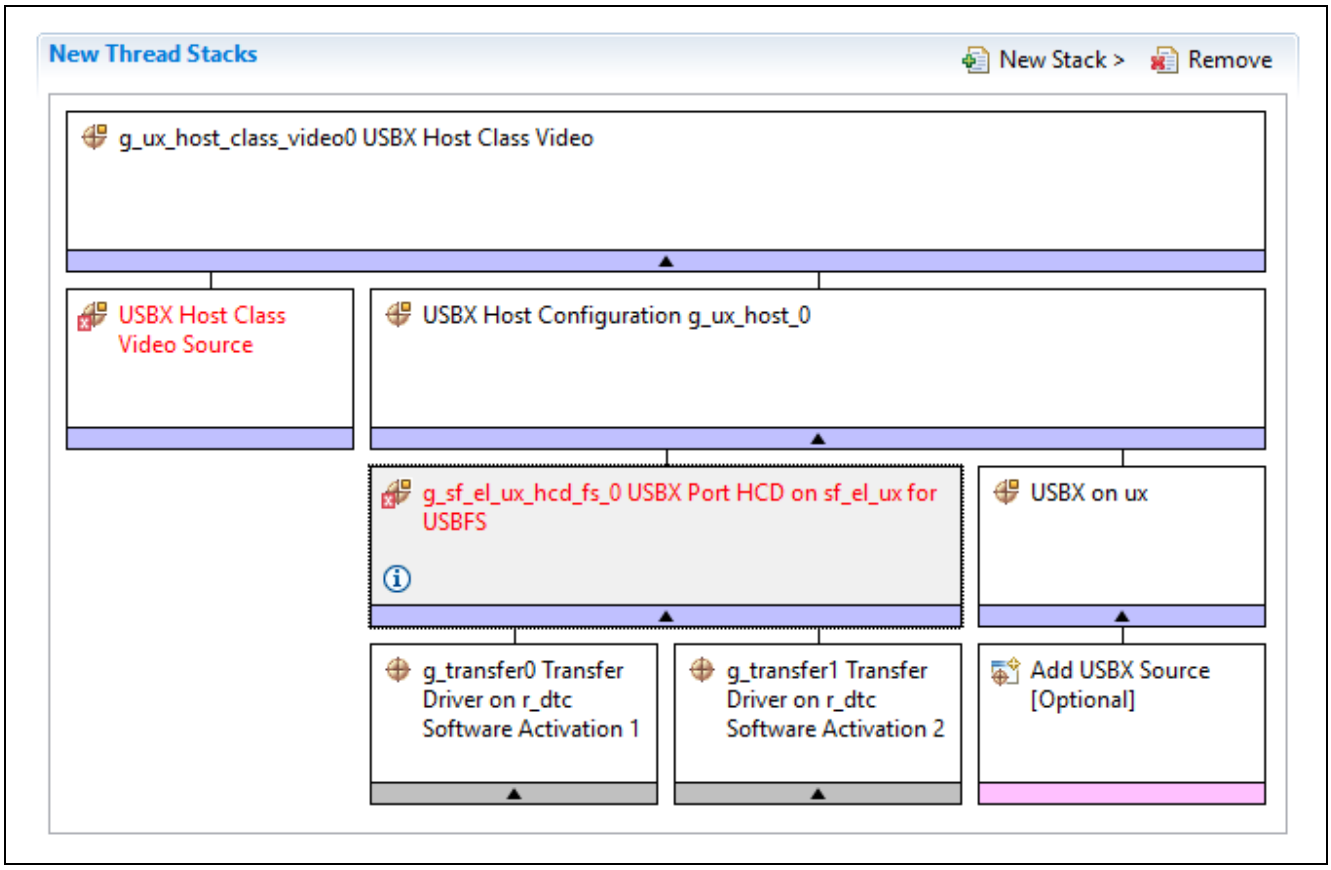


**Figure 2 USBX Host Class Video Module Stack**

## 5. Configuring the USBX Host Class Video Module

The USBX Host Class Video module has no configurable properties associated with it; configuration is performed using the lower-level modules as demonstrated in the following sections.

One of the properties most often identified as requiring a change is the interrupt priority. This configuration setting is available within the Properties window of the associated module. Simply select the indicated module and then view the Properties window. The interrupt settings are often toward the bottom of the properties list, so scroll down until they become available. Also note that the interrupt priorities listed in the Properties window in the ISDE will include an indication as to the validity of the setting based on the targeted MCU (CM4 or CM0+). This level of detail is not included in the following configuration properties tables, but is easily visible with the ISDE when configuring interrupt-priority levels.

Note: You may want to open your ISDE, create the module and explore the property settings in parallel with looking over the following configuration table values. This will help orient you and can be a useful hands-on approach to learning the ins and outs of developing with SSP.

**Table 4 Configuration Settings for the USBX Host Class Video Module**

| ISDE Property | Value | Description |
|---|---|---|
| Class Instance Name | g_ux_host_class_video0 | Class instance name. |

In some cases, settings other than the defaults for lower-level modules can be desirable. For example, it might be useful to select different channels for the data transfer driver or the size of the USBX pool memory. The configurable properties for the lower-level stack modules are given in the following sections for completeness and as a reference.

Note: Most of the property settings for lower-level modules are intuitive and usually can be determined by inspection of the associated properties window from the SSP configurator.

## 5.1 Configuration Settings for the USBX Host Class Video Module Low-Level Modules

Typically, only a small number of settings must be modified from the default for lower-level modules and these are indicated with red text in the thread stack block. Notice that some of the configuration properties must be set to a certain value for proper framework operation and will be locked to prevent user modification. Table 5 identifies all the settings within the properties section for the module.

**Table 5 Configuration Settings for the USBX Host Class Video Source**

| ISDE Property | Value | Description |
|---|---|---|
| Show linkage warning | Enable, Disable<br>Default: Enable | Notification message for users will be shown if “Enabled" option is selected. This is just to warn users of possible linkage errors by multiple symbol definitions. Selecting “Disabled” stops the notification message. |

**Table 6 Configuration Settings for the USBX Host Configuration**

| ISDE Property | Value | Description |
|---|---|---|
| Name | g_ux_host_0 | Module name. |
| Name of generated initialization function | ux_host_init0 | Name of generated initialization function selection. |
| Auto Initialization | Enable, Disable<br>Default: Enable | Auto initialization selection. |

Note: The example values and defaults are for a project using the Synergy S7 MCU Series. Other MCUs may have different default values and available configuration settings.

**Table 7 Configuration Settings for the USBX HCD on sf_el_ux for USBFS**

| ISDE Property | Value | Description |
|---|---|---|
| Full Speed Interrupt Priority | Priority 0 (highest), Priority 1:2, Priority 3 (lowest - not valid if using ThreadX), Disabled<br>Default: Disabled | Full speed interrupt priority selection. |
| VBUSEN pin Signal Logic | Active Low, Active High<br>Default: Active High | VBUSEN pin signal logic selection. |
| LDO Regulator (Only for S3 and S1 part MCUs) | Enable, Disable<br>Default: Disable | LDO regulator selection. |
| Name | g_sf_el_ux_hcd_hs_0 | Module name. |
| USB Controller Selection | USBFS | USB controller selection. |

Note: The example values and defaults are for a project using the Synergy S7 MCU Series. Other MCUs may have different default values and available configuration settings.

**Table 8 Configuration Settings for USBX on ux**

| ISDE Property | Value | Description |
|---|---|---|
| USBX Pool Memory Name | g_ux_pool_memory | USBX pool memory name selection. |
| USBX Pool Memory Size | 18432 | USBX pool memory size selection. |
| User Callback for Host Event Notification (Only valid for USB Host) | NULL | User callback for host even notification selection. |
| Name of generated initialization function | ux_common_init0 | Name of generated initialization function selection. |
| Auto Initialization | Enable, Disable<br>Default: Enable | Auto initialization selection. |

Note: The example values and defaults are for a project using the Synergy S7 MCU Series. Other MCUs may have different default values and available configuration settings.

**Table 9 Configuration Settings for the Transfer Driver on r_dtc Software Activation 1**

| ISDE Property | Value | Description |
|---|---|---|
| Parameter Checking | BSP, Enabled, Disabled<br>Default: BSP | Selects if code for parameter checking is to be included in the build. |
| Software Start | Enabled, Disabled<br>Default: Disabled | Software start selection. |
| Linker section to keep DTC vector table | .ssp_dtc_vector_table | Linker section to keep DTC vector table selection. |
| Name | g_transfer0 | Module name. |
| Mode | Normal | Mode selection. |
| Transfer Size | 1 Byte | Transfer size selection. |
| Destination Address Mode | Fixed | Destination address mode selection. |
| Source Address Mode | Incremented | Source address mode selection. |
| Repeat Area (Unused in Normal Mode) | Source | Repeat area selection. |
| Interrupt Frequency | After all transfers have completed | Interrupt frequency selection. |
| Destination Pointer | NULL | Destination pointer selection. |
| Source Pointer | NULL | Source pointer selection. |
| Number of Transfers | 0 | Number of transfers selection. |
| Number of Blocks (Valid only in Block Mode) | 0 | Number of blocks selection. |
| Activation Source (Must enable IRQ) | Software Activation 1 | Activation source selection. |
| Auto Enable | False | Auto enable selection. |
| Callback (Only valid with Software start) | NULL | Callback selection. |
| ELC Software Event Interrupt Priority | Priority 0 (highest), Priority 1:2, Priority 3 (lowest - not valid if using ThreadX), Disabled<br>Default: Disabled | ELC Software Event interrupt priority selection. |

Note: The example values and defaults are for a project using the Synergy S7 MCU Series. Other MCUs may have different default values and available configuration settings.

**Table 10 Configuration Settings for the Transfer Driver on r_dtc Software Activation 2**

| ISDE Property | Value | Description |
|---|---|---|
| Parameter Checking | BSP, Enabled, Disabled<br>Default: BSP | Selects if code for parameter checking is to be included in the build. |
| Software Start | Enabled, Disabled<br>Default: Disabled | Software start selection. |
| Linker section to keep DTC vector table | .ssp_dtc_vector_table | Linker section to keep DTC vector table selection. |
| Name | g_transfer1 | Module name. |
| Mode | Normal | Mode selection. |
| Transfer Size | 1 Byte | Transfer size selection. |
| Destination Address Mode | Fixed | Destination address mode selection. |
| Source Address Mode | Incremented | Source address mode selection. |
| Repeat Area (Unused in Normal Mode | Destination | Repeat area selection. |
| Interrupt Frequency | After all transfers have completed | Interrupt frequency selection. |

| | | |
|---|---|---|
| Destination Pointer | NULL | Destination pointer selection. |
| Source Pointer | NULL | Source pointer selection. |
| Number of Transfers | 0 | Number of transfers selection. |
| Number of Blocks (Valid only in Block Mode) | 0 | Number of blocks selection. |
| Activation Source (Must enable IRQ) | Software Activation 2 | Activation source selection. |
| Auto Enable | False | Auto enable selection. |
| Callback (Only valid with Software start) | NULL | Callback selection. |
| ELC Software Event Interrupt Priority | Priority 0 (highest), Priority 1:2, Priority 3 (lowest - not valid if using ThreadX), Disabled<br>Default: Disabled | ELC Software Event interrupt priority selection. |

Note: The example values and defaults are for a project using the Synergy S7 MCU Series. Other MCUs may have different default values and available configuration settings.

## 5.2 USBX Host Class Video Module Clock Configuration

The USB peripheral module is clocked based on the UCLK frequency. The UCLK frequency must be 48 MHz for USB operation. You can set the UCLK frequency using the clock configurator in e$^2$ studio Configuring Clocks tab or the CGC Interface at run-time.

## 5.3 USBX Host Class Video Module Pin Configuration

The USB peripheral module uses pins on the MCU to communicate to external devices. I/O pins must be selected and configured as required by the external device. Table 11 illustrates the method for selecting the pins within the SSP configuration window and Table 12 and Table 13 illustrate example selections for USB pins.

Note: Operation Mode selection determines what peripheral signals are available and thus what MCU pins are required.

**Table 11 Pin Selection Sequence for USBFS and USBHS**

| Resource | ISDE Tab | Pin selection Sequence |
|---|---|---|
| USBFS | Pins | Select Peripherals > Connectivity: USBFS> USBFS0 |
| USBHS | Pins | Select Peripherals > Connectivity: USBHS> USBHS0 |

Note: The selection sequence assumes USBFS0 or USBHS0 are the desired hardware target for the driver.

**Table 12 Pin Configuration Settings for the USBFS**

| Property | Value | Description |
|---|---|---|
| Operation Mode | Disabled, Custom, Device, Host, OTG<br>(Default: Disabled) | Select Device as the Operation Mode |
| USBDP | USBDP | USBDP Pin |
| USBDM | USBDM | USBDM Pin |
| OVRCURB | None | OVRCURB Pin |
| OVRCURA | None | OVRCURA Pin |
| VBUSEN | None | VBUSEN Pin |
| VBUS | None, P407<br>(Default: P407) | VBUS Pin |
| EXICEN | None | EXICEN Pin |
| ID | None | ID Pin |
| VCCUSB | VCCUSB | VCCUSB Pin |
| VSSUSB | VSSUSB | VSSUSB Pin |

Note: The example settings are for a project using the S7G2 Synergy MCU Group and the SK-S7G2 Synergy Kit. Other Synergy MCUs and Synergy Kits may have different available pin configuration settings.

**Table 13 Pin Configuration Settings for the USBHS**

| Property | Value | Description |
|---|---|---|
| Operation Mode | Disabled, Custom, Device, Host, OTG Default: Disabled | Select Device as the Operation Mode |
| USBHSDP | USBHSDP | USBHSDP Pin |
| USBHSDM | USBHSDM | USBHSDM Pin |
| OVRCURB | None | OVRCURB Pin |
| OVRCURA | None | OVRCURA Pin |
| VBUSEN | None | VBUSEN Pin |
| VBUS | None | VBUS Pin |
| EXICEN | None | EXICEN Pin |
| ID | None | ID Pin |
| USBHSRREF | USBHSRREF | USBHSRREF Pin |
| AVCCUSBHS | AVCCUSBHS | AVCCUSBHS Pin |
| AVSSUSBHS | AVSSUSBHS | AVSSUSBHS Pin |
| PVSSUSBHS | PVSSUSBHS | PVSSUSBHS Pin |
| VSS1USBHS | VSS1USBHS | VSS1USBHS Pin |
| VSS2USBHS | VSS2USBHS | VSS2USBHS Pin |

Note: The example settings are for a project using the Synergy S7G2 Group and the SK-S7G2 Kit. Other Synergy Kits and Synergy MCUs may have different available pin configuration settings.

## 6. Using the USBX Host Class Video Module in an Application

The configurator generates processing to register the USBX Host Class Video module. Specify the same module as **USBX Host Configuration** module registered in the class to be used at the same time.

Figure 3 shows the stack when registered with the USBX Host Class Video module at the same time.

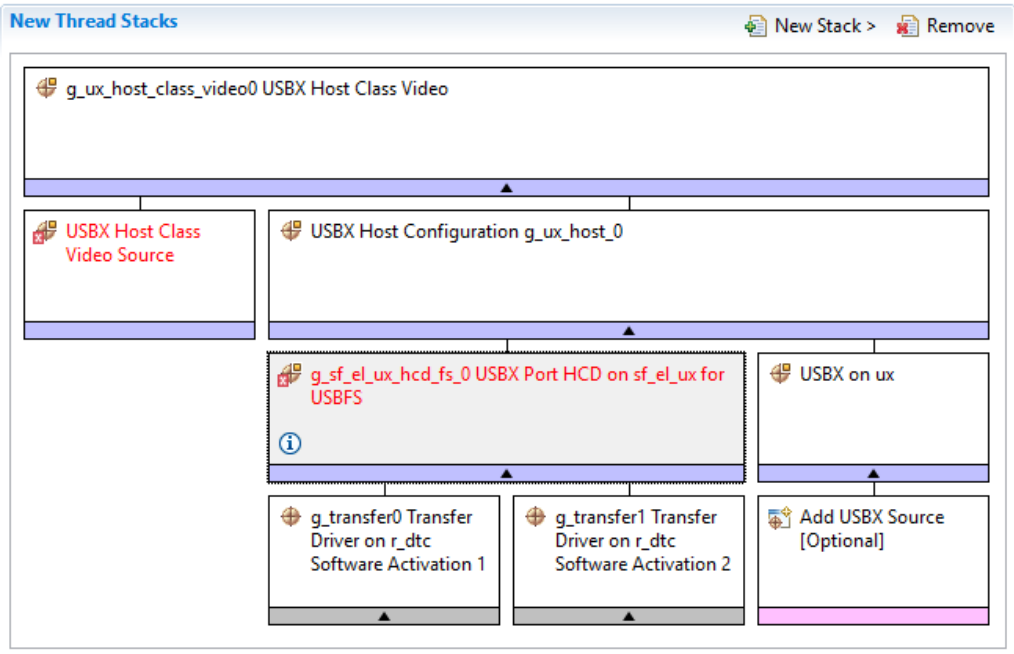


**Figure 3 USBX Host Class Video with Low-Level modules**

# 7. The USBX Host Class Video Module Application Project

The application project associated with this module guide demonstrates a full design. You may want to import and open the application project within ISDE and view the configuration settings for the USBX Host Class Video module. You can also read over the code in `camera_thread_entry.c` which is used to illustrate the USBX Host Class Video module APIs in a complete design.

The application project main thread entry waits for the connection from the callback function, and lists parameters of the camera. Then, the application sets the camera parameters and receives frame data. Receiving of frame data repeats until device is unplugged. The user-callback function for Host Event obtains a pointer of the device instance and notifies the main thread of the plugged or unplugged state. The user-callback function of video class notifies the main thread every time it receives one frame data. The following table identifies the target versions for the associated software and hardware used by the application project.

**Table 14 Software and Hardware Resources Used by the Application Project**

| Resource | Revision | Description |
|---|---|---|
| e$^2$ studio | 6.2.0 or later | Integrated Solution Development Environment |
| SSP | 1.4.0 or later | Synergy Software Platform |
| IAR EW for Synergy | 8.21.1 or later | IAR Embedded Workbench® for Renesas Synergy™ |
| SSC | 6.2 or later | Synergy Standalone Configurator |
| SK-S7G2 | v3.0 to v3.1 | Starter Kit |

A simple flow diagram of the application project is given in the following figure:

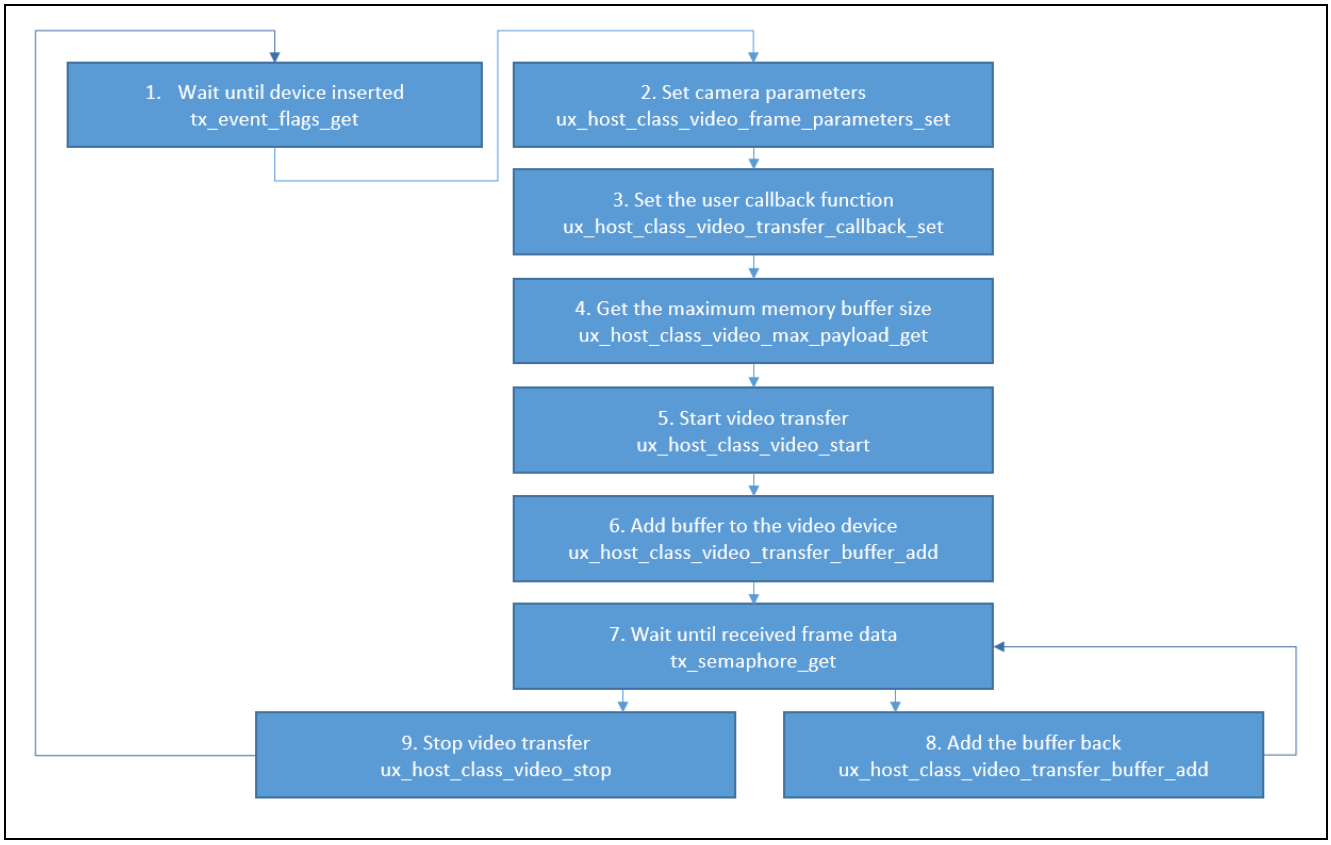


**Figure 4 USBX Host Class Video Module Application Project Flow Diagram**

The `camera_thread_entry.c` file is located in the project once it has been imported into the ISDE. You can open this file within the ISDE and follow along with the description below to help identify key uses of APIs.

The first section of the `camera_thread_entry.c` has the header files that reference the USBX structures and a code section which contains macro constants, global variables, and a function prototype. The first macro definition (EVENTFLAG_USB_DEVICE_INSERTED) defines the bit of the device status to use with the event flag. The next macro definition (MAX_NUM_BUFFERS) is the number of buffers used for frame data transfer. The next data

(video_host_class) is a pointer to the instance of video class, and the next data (video_buffer) is the buffer area. The next data (vs_type_name) is strings that converts format type data for display.

The next section is the USBX Host event function, which handles the camera device. It checks if the device has been already inserted or removed. In the first case, the instance of the device is assigned to the global variable and the application waits until the camera device status is live. In case of removal, the global variable, previously storing the instance of the camera device, is set to null. This callback always returns a code indicating that a successful operation has been performed, as it is called from the generated code. (If the error occurred here, the video class device would not be handled properly.) The next section is a callback when frame data is received, and it is notified to the main thread by semaphore. The next three sections are functions that list camera parameters.

The next section is a function that controls the entire application. It displays the parameters supported by the connected camera and sets the parameters. The application project sets the preliminary parameters, but you can change these. After initializing the data, start transfer and start receiving frame data. Receiving of frame data is done consecutively, but when it becomes unplugged, it exits the receiving loop.

The last section is the entry function of the main thread. The infinite while loop begins and if there is an attached camera, the control process is called.

A few key properties are configured in this application project to support the required operations and the physical properties of the target board and MCU device. The properties with the values set for this specific project are listed in the following tables. You can also open the application project and view these settings in the **Properties** window as a hands-on exercise.

**Table 15 USBX Port HCD for USBHS Configuration Settings for the Application Project**

| ISDE Property | Value Set |
|---|---|
| High Speed Interrupt Priority | Priority 3 |
| FIFO size for Bulk Pipes | 2048 bytes |
| Number of Isochronous Pipes | 2 |
| VBUSEN pin Signal Logic | Active High |
| Enable High Speed | Enable |
| Name | g_sf_el_ux_hcd_hs_0 |
| USB Controller Selection | USBHS |

**Table 16 USBX Configuration Settings for the Application Project**

| ISDE Property | Value Set |
|---|---|
| USBX Pool Memory Name | `g_ux_pool_memory` |
| USBX Pool Memory Size | 40960 |
| User Callback for Host Event Notification | `ux_host_usr_event_notification` |

## 8. Customizing the USBX Host Class Video Module for a Target Application

Some configuration settings will normally be changed from those shown in the application project. For example, you can also add a data transfer module for data transfer of USBX Port HCD. This data transfer module can be added simply by clicking on the box for TX or RX displayed under the USBX Port HCD box of the configurator normally.

## 9. Running the USBX Host Class Video Module Application Project

To run the USBX Host Class Video module application project and to see it executed on a target kit, you can simply import it into your ISDE, compile, and run debug.

Note: The following steps are described in sufficient detail for someone experienced with the basic flow through the Synergy development process. If these steps are not familiar, refer to the first few chapters of the *SSP User's Manual* for a description of how to accomplish these steps.

To create and run the application project simply follow these steps:

1. Import and build the example project included with this module guide according to the Synergy Project Import Guide (11an0023eu0120-synergy-ssp-import-guide.pdf.)
2. Connect to the host PC using the USB cable (use J19 DEBUG_USB connector).
3. Start to debug the application.
4. Connect to the USB camera using the USB cable (use J6 USB connector).
5. The output can be viewed in the Renesas Debug Console.

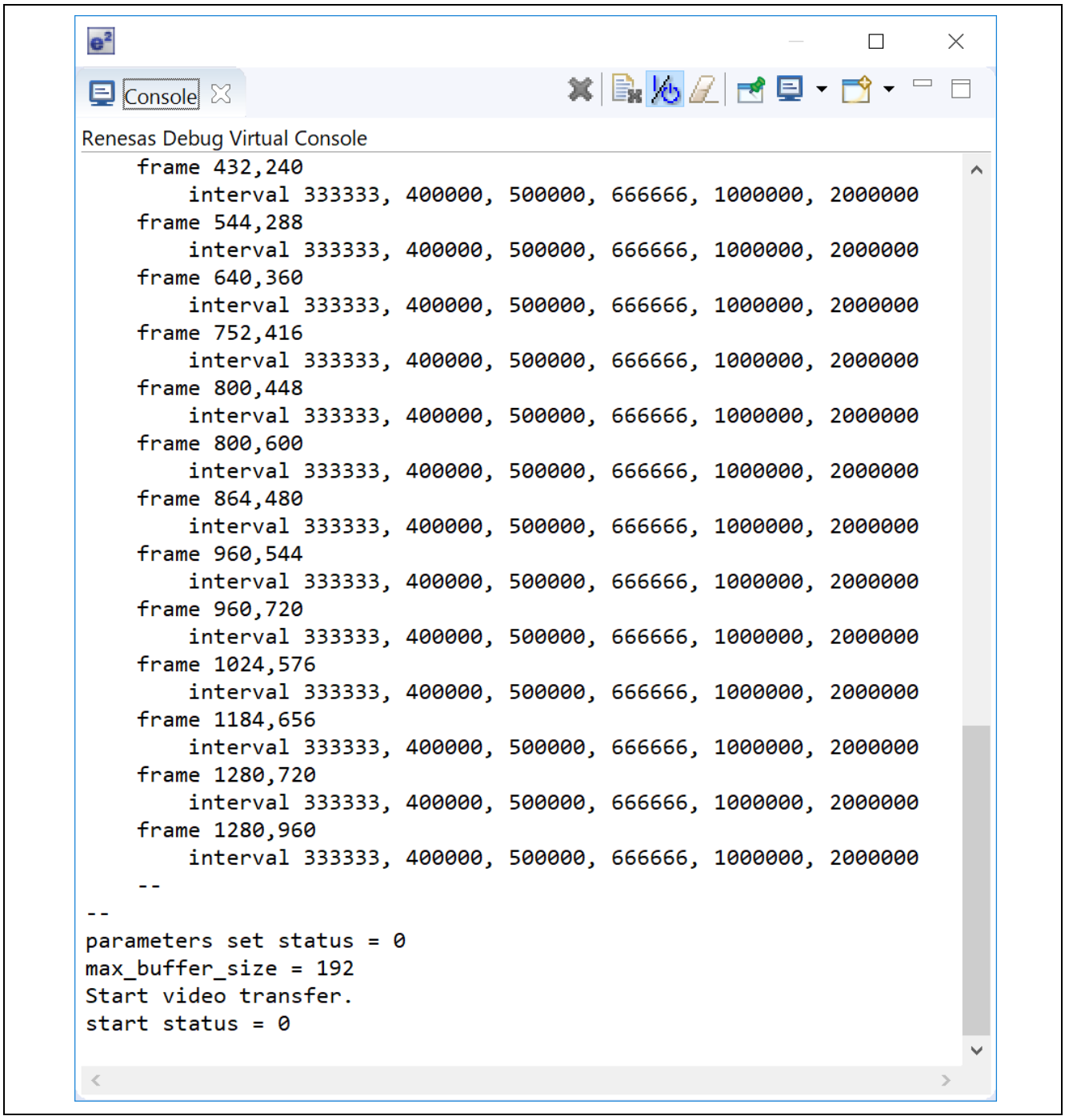


**Figure 5 Example Output from USBX Host Class Video Module Application Project**

## 10. USBX Host Class Video Module Conclusion

This module guide has provided all the background information needed to select, add, configure, and use the module in an example project. Many of these steps were time consuming and error-prone activities in previous generations of embedded systems. The SSP makes these steps much less time consuming and removes the common errors like conflicting configuration settings or incorrect selection of lower-level drivers. The use of high-level APIs (as demonstrated in the application project) illustrates additional development-time savings by allowing work to begin at a high-level and avoiding the time required to use, or, in some cases, create, lower-level drivers in older development environments.

## 11. USBX Host Class Video Module Next Steps

After you have mastered a USBX Host Class Video module project, you may want to review another example. Other application projects and application notes that demonstrate USBX use can be found as described in the References section in this document.

## 12. USBX Host Class Video Module Reference Information

*SSP User Manual:* Available in html format in the SSP distribution package and as a pdf from the Synergy Gallery.

Links to all the most up-to-date USBX Host Class Video module reference materials and resources are available on the Synergy Knowledge Base: https://en-us.knowledgebase.renesas.com/English_Content/Renesas_Synergy%E2%84%A2_Platform/Renesas_Synergy_Knowledge_Base/USBX_Host_Class_Video_Module_Guide_Resources. https://en-us.knowledgebase.renesas.com/English_Content/Renesas_Synergy%E2%84%A2_Platform/Renesas_Synergy_Knowledge_Base/USBX_Host_Class_Video_Module_Guide_Resources.

## Website and Support

Visit the following vanity URLs to learn about key elements of the Synergy Platform, download components and related documentation, and get support.

| | |
|---|---|
| Synergy Software | renesassynergy.com/software |
| Synergy Software Package | renesassynergy.com/ssp |
| Software add-ons | renesassynergy.com/addons |
| Software glossary | renesassynergy.com/softwareglossary |
| Development tools | renesassynergy.com/tools |
| Synergy Hardware | renesassynergy.com/hardware |
| Microcontrollers | renesassynergy.com/mcus |
| MCU glossary | renesassynergy.com/mcuglossary |
| Parametric search | renesassynergy.com/parametric |
| Kits | renesassynergy.com/kits |
| Synergy Solutions Gallery | renesassynergy.com/solutionsgallery |
| Partner projects | renesassynergy.com/partnerprojects |
| Application projects | renesassynergy.com/applicationprojects |
| Self-service support resources: | |
| Documentation | renesassynergy.com/docs |
| Knowledgebase | renesassynergy.com/knowledgebase |
| Forums | renesassynergy.com/forum |
| Training | renesassynergy.com/training |
| Videos | renesassynergy.com/videos |
| Chat and web ticket | renesassynergy.com/support |

## Revision History

| Rev. | Date | Description | |
|---|---|---|---|
| | | Page | Summary |
| 1.00 | Oct 29, 2018 | - | Initial document release |

# Notice

1. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation or any other use of the circuits, software, and information in the design of your product or system. Renesas Electronics disclaims any and all liability for any losses and damages incurred by you or third parties arising from the use of these circuits, software, or information.
2. Renesas Electronics hereby expressly disclaims any warranties against and liability for infringement or any other claims involving patents, copyrights, or other intellectual property rights of third parties, by or arising from the use of Renesas Electronics products or technical information described in this document, including but not limited to, the product data, drawings, charts, programs, algorithms, and application examples.
3. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
4. You shall not alter, modify, copy, or reverse engineer any Renesas Electronics product, whether in whole or in part. Renesas Electronics disclaims any and all liability for any losses or damages incurred by you or third parties arising from such alteration, modification, copying or reverse engineering.
5. Renesas Electronics products are classified according to the following two quality grades: “Standard” and “High Quality”. The intended applications for each Renesas Electronics product depends on the product’s quality grade, as indicated below.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; industrial robots; etc.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control (traffic lights); large-scale communication equipment; key financial terminal systems; safety control equipment; etc.

   Unless expressly designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not intended or authorized for use in products or systems that may pose a direct threat to human life or bodily injury (artificial life support devices or systems; surgical implantations; etc.), or may cause serious property damage (space system; undersea repeaters; nuclear power control systems; aircraft control systems; key plant systems; military equipment; etc.). Renesas Electronics disclaims any and all liability for any damages or losses incurred by you or any third parties arising from the use of any Renesas Electronics product that is inconsistent with any Renesas Electronics data sheet, user’s manual or other Renesas Electronics document.
6. When using Renesas Electronics products, refer to the latest product information (data sheets, user’s manuals, application notes, “General Notes for Handling and Using Semiconductor Devices” in the reliability handbook, etc.), and ensure that usage conditions are within the ranges specified by Renesas Electronics with respect to maximum ratings, operating power supply voltage range, heat dissipation characteristics, installation, etc. Renesas Electronics disclaims any and all liability for any malfunctions, failure or accident arising out of the use of Renesas Electronics products outside of such specified ranges.
7. Although Renesas Electronics endeavors to improve the quality and reliability of Renesas Electronics products, semiconductor products have specific characteristics, such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Unless designated as a high reliability product or a product for harsh environments in a Renesas Electronics data sheet or other Renesas Electronics document, Renesas Electronics products are not subject to radiation resistance design. You are responsible for implementing safety measures to guard against the possibility of bodily injury, injury or damage caused by fire, and/or danger to the public in the event of a failure or malfunction of Renesas Electronics products, such as safety design for hardware and software, including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult and impractical, you are responsible for evaluating the safety of the final products or systems manufactured by you.
8. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. You are responsible for carefully and sufficiently investigating applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive, and using Renesas Electronics products in compliance with all these applicable laws and regulations. Renesas Electronics disclaims any and all liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
9. Renesas Electronics products and technologies shall not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations. You shall comply with any applicable export control laws and regulations promulgated and administered by the governments of any countries asserting jurisdiction over the parties or transactions.
10. It is the responsibility of the buyer or distributor of Renesas Electronics products, or any other party who distributes, disposes of, or otherwise sells or transfers the product to a third party, to notify such third party in advance of the contents and conditions set forth in this document.
11. This document shall not be reprinted, reproduced or duplicated in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products.

(Note 1) “Renesas Electronics” as used in this document means Renesas Electronics Corporation and also includes its directly or indirectly controlled subsidiaries.

(Note 2) “Renesas Electronics product(s)” means any product developed or manufactured by or for Renesas Electronics.

(Rev.4.0-1 November 2017)

RENESAS

## SALES OFFICES

Renesas Electronics Corporation

http://www.renesas.com

Refer to "http://www.renesas.com/" for the latest and detailed information.


**Renesas Electronics Corporation**
TOYOSU FORESIA, 3-2-24 Toyosu, Koto-ku, Tokyo 135-0061, Japan

**Renesas Electronics America Inc.**
1001 Murphy Ranch Road, Milpitas, CA 95035, U.S.A.
Tel: +1-408-432-8888, Fax: +1-408-434-5351

**Renesas Electronics Canada Limited**
9251 Yonge Street, Suite 8309 Richmond Hill, Ontario Canada L4C 9T3
Tel: +1-905-237-2004

**Renesas Electronics Europe Limited**
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K
Tel: +44-1628-651-700

**Renesas Electronics Europe GmbH**
Arcadiastrasse 10, 40472 Düsseldorf, Germany
Tel: +49-211-6503-0, Fax: +49-211-6503-1327

**Renesas Electronics (China) Co., Ltd.**
Room 1709 Quantum Plaza, No.27 ZhichunLu, Haidian District, Beijing, 100191 P. R. China
Tel: +86-10-8235-1155, Fax: +86-10-8235-7679

**Renesas Electronics (Shanghai) Co., Ltd.**
Unit 301, Tower A, Central Towers, 555 Langao Road, Putuo District, Shanghai, 200333 P. R. China
Tel: +86-21-2226-0888, Fax: +86-21-2226-0999

**Renesas Electronics Hong Kong Limited**
Unit 1601-1611, 16/F., Tower 2, Grand Century Place, 193 Prince Edward Road West, Mongkok, Kowloon, Hong Kong
Tel: +852-2265-6688, Fax: +852 2886-9022

**Renesas Electronics Taiwan Co., Ltd.**
13F, No. 363, Fu Shing North Road, Taipei 10543, Taiwan
Tel: +886-2-8175-9600, Fax: +886 2-8175-9670

**Renesas Electronics Singapore Pte. Ltd.**
80 Bendemeer Road, Unit #06-02 Hyflux Innovation Centre, Singapore 339949
Tel: +65-6213-0200, Fax: +65-6213-0300

**Renesas Electronics Malaysia Sdn.Bhd.**
Unit 1207, Block B, Menara Amcorp, Amcorp Trade Centre, No. 18, Jln Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: +60-3-7955-9390, Fax: +60-3-7955-9510

**Renesas Electronics India Pvt. Ltd.**
No.777C, 100 Feet Road, HAL 2nd Stage, Indiranagar, Bangalore 560 038, India
Tel: +91-80-67208700, Fax: +91-80-67208777

**Renesas Electronics Korea Co., Ltd.**
17F, KAMCO Yangjae Tower, 262, Gangnam-daero, Gangnam-gu, Seoul, 06265 Korea
Tel: +82-2-558-3737, Fax: +82-2-558-5338




Colophon 7.2