

Final Release Documentation ver 1.0

Contents

	<i>Page</i>
API Details	2
New CSLv3.05 Based description.....	2
Tool Details	3
Test Description	3
Test Instructions	4
<i>C55xx EVM end:</i>	4
<i>PC end:</i>	4
Test Results	5
<i>Sample snapshot of a typical run on C55xx EVM and Windows 7 PC (below):</i> ..	5

API Details

New CSLv3.05 Based description

csl_gdc.c – cs/src file for Generic Device Class (GDC) API's

Chief contents:

```
CSL_StatusGDC_Ep_Setup(pGdcClassHandlepHandle,  
BoolusbSpeedCfg)  
  
pHandle    address of handle to the chief GDC Class structure  
usbSpeedCfg  TRUE - High Speed mode  FALSE - Full Speed Mode
```

Function to setup the Generic Device Class Control and Bulk, IN and OUT Endpoints.

USB_requestEndpt() and USB_configEndpt() used to initialize and configure the endpoints.

csl_gdc.h – Header file for Generic Device Class (GDC) structures and extern'ed variables

Chief contents:

GDC Object structure - Holds the bulk and interrupt endpoint object pointers

```
typedef struct CSL_GdcObject {  
    CSL_UsbEpHandle bulkInEpHandle;  
    /** \brief Bulk Out Endpoint Object */  
    CSL_UsbEpHandle bulkOutEpHandle;  
    /** \brief Bulk In Endpoint Object */  
    CSL_UsbEpHandle intrInEpHandle  
    /** \brief Bulk Out Endpoint Object */  
    CSL_UsbEpHandle intrOutEpHandle;  
} CSL_GdcObject;
```

GDC Control structure - Holds the Control endpoint object pointers

```
typedef struct CSL_GdcCtrlObject {  
    /** \brief Control In Endpoint Object */  
    CSL_UsbEpHandle ctrlInEpHandle;  
    /** \brief Object of Type Device Number */  
    CSL_UsbDevNum devNum;  
    /** \brief Control Out Endpoint Object */  
    CSL_UsbEpHandle ctrlOutEpHandle;  
} CSL_GdcCtrlObject;
```

GDC Class structure - Holds all components for the chief GDC Class Structure

```
typedef struct CSL_GdcClassStruct {  
    /** \brief Handle to the selected USB Device instance */  
    CSL_UsbDevHandle usbDevHandle;  
    /** \brief Handle to Control Object */  
    CSL_GdcCtrlObject ctrlHandle;  
    /** \brief Handle to GDC Transfer Object */  
    CSL_GdcObject gdcHandle;  
} CSL_GdcClassStruct;
```

GDC class handle - Global handle for the chief GDC Class Structure

```
typedef CSL_GdcClassStruct *pGdcClassHandle;
```

Tool Details

Code Composer Studio version: 6.1.2 or above

Chip Support Library (CSL) Version: 3.05 or above

DSP/BIOS Version: 5.42.1.09 or above

Code Generation Tool (CGT) version: Bundled with CCS 6.1.2 or above

Test Description

This test is to verify the operation of the CSL USB module's WinUSB Generic Device Class (GDC) driver. This test runs in interrupt mode. USB interrupts are configured and ISR is registered using CSL INTC module. After initializing and configuring the USB module test waits on a while loop. When there is any request from the USB host application (WinUSB Test App) USB ISR is triggered and the requested operation is performed inside the ISR.

There are 3 endpoints of concern:

1 Control Endpoint 0

1 Bulk IN Endpoint 3

1 Bulk OUT Endpoint 2

A 320x200 bytes image is stored in "image.h" file. When a command is sent from the PC host WinUSB Test App to the C55 to send an image chunk (command: 0xAA followed by chunk ID - 1 byte for 0xAA, 2 bytes for chunk ID), the same is sent across and when all chunks have been received, the image is displayed on the PC host using the WinUSB Test App.

USB enumeration as a Generic Device Class WinUSB device requires a specific inf file – C5517_WINUSB.inf.


Commands are received on BULK OUT endpoint 2 and Image is sent out via BULK IN endpoint 3.

Test Instructions

C55xx EVM end:

1. Connect a USB A/B cable from your host PC's USB port to port 'EMU USB'(J201) on the C55xx EVM. Connect another USB port on the PC using a Type B cable to the on-board Type B port.
2. Open the CSL_USB_WinusbExample CCSproject using CCSv6.x.
3. Import all the threecsl projects - atafs_bios_drv_lib, C55XXCSL_LP and CSL_USB_WinUSBExample_Out.
4. Rebuild, Launch and run the CSL_USB_WinUSBExample_Out project.
5. Check if an Unknown device pops up on the PC Device Manager.
6. R-click, Goto Properties > Update driver and install C5517_WINUSB.inf.
7. Check if a new category 'Universal Serial Bus Devices' pops up.
8. Check if C55xx EVM is recognized here as C5517 (on Win 7) or C5517_WINUSB (Win7).

PC end:

1. One may use Microsoft Visual Studio 2013 to open/run the WinUsbTest c# solution Winusb\Winusb.sln.
2. Click Debug > Start Debugging or the  icon or Press F5.
3. If required, R-click on Solution 'Winusb' in the Solution Explorer window and click Rebuild Solution and then proceed to launch the application as described in 2.
4. One may also directly run the application exe. Double-click the application exe WinUsbTest.exe (Application).
5. In the USB Device GUID field, type in 9f543223-cede-4fa3-b376-a25ce9a30e74 and click 'Find' tab.
6. Once the device is detected, 'Attached' is displayed below the 'Find' tab. Until then, it displays 'Detached'.
7. The CSL_USB_WinUSBExample_OutCCS project should be already running on the C55xx EVM or can be run now to 'Attach' the device.
8. Once the C55xx EVM is 'Attached', only then, one must proceed further.
9. Click 'Image Test' tab.
10. A vertical bar on a white background is displayed under 'Received Image'. The bar sweeps from the left edge to the right edge and then starts sweeping all over from left edge to the right again. It goes on continuously.
11. Clicking the 'Image Test' tab again stops the moving bar.
12. Click the 'Image Test' tab once more and the vertical bar resumes the sweep from where it last stopped.
13. The vertical bar also changes colour randomly.

Test Results

The setup was run on the C55xx EVM and Win 7 PC over long durations. The final version of the code based on CSLv3.05 has been run without any complications. Please find below a snapshot of a typical run on C55xx EVM and Win 7 PC.

Sample snapshot of a typical run on C5517 EVM and Windows 7 PC (below):

