CC32xx Dynamic Library Loader

Overview

Return to CC31xx & CC32xx Home Page

This application illustrates how to hook different libraries at run time having common interface APIs, but (maybe) differing in their implementation. On systems with limited RAM, this feature can be of utmost importance if the application intends to load a library into memory at run-time based on some conditions.

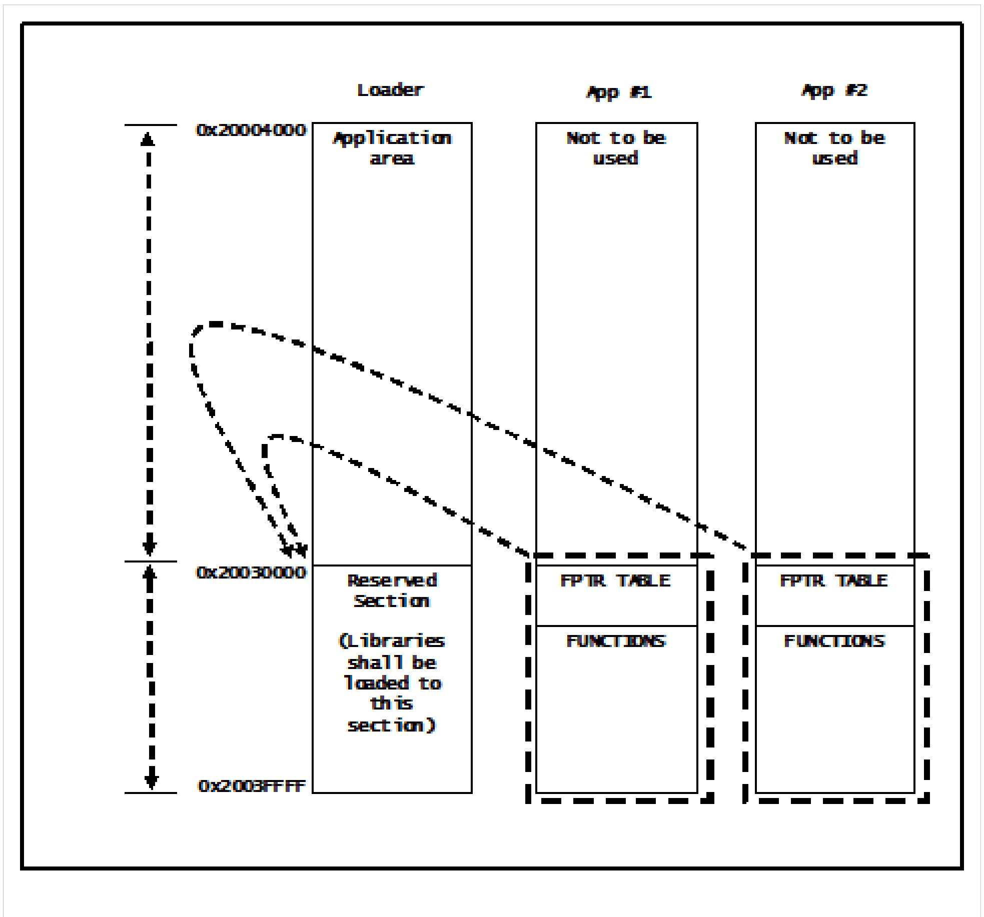
Application Details

This a very basic application where two different library having same interface APIs are being loaded one by one, which is followed by their invocation in the same order. Both library differs in their implementations, thus giving different output when called with the same arguments.

Design Approach

The following steps will walk you through the design approach:

1. The application shall determine a predefined function-pointer table to interface with different libraries. All libraries are expected to have the same interface.
2. It shall identify a region in the memory for paging – And the predefined function-pointer table shall be put at the beginning of this region.
3. Before invoking any library function(s), the loader/main-application shall load the contents of the library from SFLASH into the defined region on SRAM.
4. The application shall then use the function-pointer table to access the library function(s).



Note: The memory addresses shown in above diagram are for reference only – Developers shall define the sections based on their application’s needs. The workspace for this application consists of three projects(for IAR) or three configurations (for CCS):

app1

app1 is the first library with four routines

app2

app2 is the second library (also) with four routines

loader

loader loads the above libraries from non-volatile memory to SRAM in run time, then invokes their routines (common)

Usage

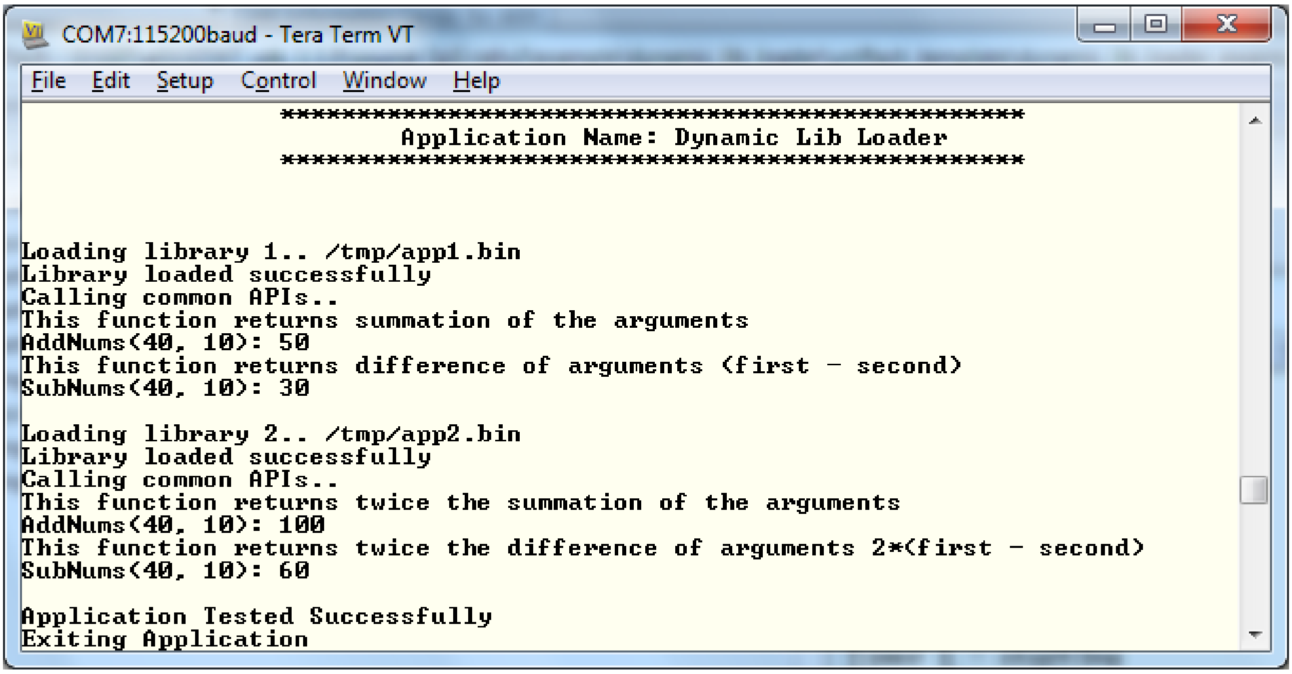
- Modify the implementation(not mandatory) of the libraries and compile them.
- Setup a serial communication application (HyperTerminal/TeraTerm). For detail info visit CC31xx & CC32xx Terminal Setting

On the host PC. The settings are:

- **Port**: Enumerated COM port
- **Baud rate**: 115200
- **Data**: 8 bit

- **Parity**: None
- **Stop**: 1 bit
- **Flow control**: None

- Flash the libraries and the loader application (ucf files provided for IAR and CCS workspace) using **uniflash**.
- Run the loader application either from Debugger or the Flash.
- Different output for the same APIs will showcase the dynamic loading of libraries.


COM7:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
                                 *************************************************
                                         Application Name: Dynamic Lib Loader
                                 *************************************************



Loading library 1.. /tmp/app1.bin
Library loaded successfully
Calling common APIs..
This function returns summation of the arguments
AddNums(40, 10): 50
This function returns difference of arguments (first - second)
SubNums(40, 10): 30

Loading library 2.. /tmp/app2.bin
Library loaded successfully
Calling common APIs..
This function returns twice the summation of the arguments
AddNums(40, 10): 100
This function returns twice the difference of arguments 2*(first - second)
SubNums(40, 10): 60

Application Tested Successfully
Exiting Application
```

Limitations/Known Issues

None

Article Sources and Contributors

CC32xx Dynamic Library Loader *Source*: http://processors.wiki.ti.com/index.php?oldid=195034 *Contributors*: Jitgupta

Image Sources, Licenses and Contributors

File:Cc31xx cc32xx return home.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Cc31xx_cc32xx_return_home.png *License*: unknown *Contributors*: A0221015
File:Dynamic lib loader design approach.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Dynamic_lib_loader_design_approach.png *License*: unknown *Contributors*: Jitgupta
File:Dynamic Lib Loader.png *Source*: http://processors.wiki.ti.com/index.php?title=File:Dynamic_Lib_Loader.png *License*: unknown *Contributors*: Jitgupta