MCCI Corporation
3520 Krums Corners Road
Ithaca, New York 14850 USA
Phone +1-607-277-1029
Fax +1-607-277-6844
www.mcci.com

MCCI Transaction Translator User's Guide

Engineering Report 950000548
Rev. C
Date: 2011/09/30

PROPRIETARY NOTICE AND DISCLAIMER

Unless noted otherwise, this document and the information herein disclosed are proprietary to MCCI Corporation, 3520 Krums Corners Road, Ithaca, New York 14850 (“MCCI”). Any person or entity to whom this document is furnished or having possession thereof, by acceptance, assumes custody thereof and agrees that the document is given in confidence and will not be copied or reproduced in whole or in part, nor used or revealed to any person in any manner except to meet the purposes for which it was delivered. Additional rights and obligations regarding this document and its contents may be defined by a separate written agreement with MCCI, and if so, such separate written agreement shall be controlling.

The information in this document is subject to change without notice, and should not be construed as a commitment by MCCI. Although MCCI will make every effort to inform users of substantive errors, MCCI disclaims all liability for any loss or damage resulting from the use of this manual or any software described herein, including without limitation contingent, special, or incidental liability.

MCCI, TrueCard, TrueTask, MCCI Catena, and MCCI USB DataPump are registered trademarks of MCCI Corporation.

MCCI Instant RS-232, MCCI Wombat and InstallRight Pro are trademarks of MCCI Corporation.

All other trademarks and registered trademarks are owned by the respective holders of the trademarks or registered trademarks.

NOTE: The code sections presented in this document are intended to be a facilitator in understanding the technical details. They are for illustration purposes only, the actual source code may differ from the one presented in this document.

Document Release History

Rev. A	2008/01/03	Original release
Rev. B	2010/06/24	Updated for memory requirements changes
Rev. C	2011/09/30	Added source code disclaimer.

TABLE OF CONTENTS

LIST OF TABLES

1 Introduction

The scope of this document is to describe how to initialize the Transaction Translator (TT) and what the memory requirements are for various configurations.

2 Initializing the Transaction Translator

Before initializing the Transaction Translator, you must first ask yourself the following questions:

1. How many TTs do I wish to support?

2. Do I want to allow hubs to be configured as "one TT per port"?

3. What periodic pipe interval accuracy do I wish to achieve?

4. How much memory do I have available?

Initialization of the Transaction Translator requires changes to your application and to the HCD configuration.

2.1 Initializing the Transaction Translator in your application

To initialize the Transaction Translator, you must first define the configuration in your application:

```
static CONST USBPUMP_USBDI_USBDTT_CONFIG sk_UsbPumpUsbdTT_Config =
     USBPUMP_USBDI_USBDTT_CONFIG_INIT_V1(
          /* Allow TTperPort hub configuration */ fTTPerPort,
          /* Number of TTs */ bNumberTTs
          );
```

All hubs can be configured as "one TT per hub" (TTPerHub), but some hubs can also be configured as"one TT per port" (TTPerPort). If you want to allow hubs to be configured as TTPerPort when it is available, set `fTTPerPort` to TRUE. Otherwise, set it to FALSE. Allowing hubs to operate as TTPerPort will require more TT objects and more memory.

Set `bNumberTTs` to the number of TTs you wish to support. If you want the system to automatically calculate it, select 0. If you select 0, the system will create (nHubs * nPortsPerHub) TT objects.

Once you have defined the configuration parameters, you must call the API to initialize the Transaction Translator. This API must be called immediately after the API to initialize the USBD (after `UsbPumpUsbd_Initialize()`).

```
   if (UsbPumpUsbdTT_Initialize(
        &pHcd->ObjectHeader,
        &pUsbdObject->ObjectHeader,
        &sk_UsbPumpUsbdTT_Config
        ) == FALSE)
        {
        TTUSB_OBJPRINTF((
             &pHcd->ObjectHeader,
             UDMASK_ERRORS,
             "-App_Name: UsbPumpUsbdTT_Initialize"
             " failed to initialize TT support.\n"
             ));
        }
```

2.2 Setting HCD parameters

You will need to define the schedule levels in the HCD configuration. The following sample HCD configuration is DWC3884SP host controller HCD configuration information. Usually HCD configuration will be different for each host controller driver. The fields in **bold** are the new fields you will need to set.

```
#define EVAL460_SCHEDULE_LEVEL      9
#define EVAL460_SCHEDULE_TT_LEVEL   6

CONST USBPUMP_HCD_DWC3884SP_CONFIG_INFO gk_Eval460_HcdConfig =
   USBPUMP_HCD_DWC3884SP_CONFIG_INFO_INIT_V3(
        /* ulWiring */ 0,
        /* hBus */ (UHIL_BUSHANDLE) 0,
        /* IoPort */ (IOPORT) EVAL460_USB_BASE,
        /* hUsbInt */ EVAL460_INTERRUPT_RESOURCE_HANDLE,
        /* pPrimaryIsr */ dwc3884sphcd_PrimaryIsr,
        /* RxFifoSz */ 0,
        /* NPTxFifoSz */ 0,
        /* PTxFifoSz */ 0,
        /* MemoryPoolAlign */ DMA_ALIGNMENT,
        /* nScheduleLevel */ EVAL460_SCHEDULE_LEVEL,
        /* nScheduleTTLevel */ EVAL460_SCHEDULE_TT_LEVEL
        );
```

`nScheduleLevel` and `nScheduleTTLevel` are used for periodic transfers. `nScheduleLevel` defines the number of levels in the schedule tree for the native host bus. `nScheduleTTLevel` defines the number of levels in the schedule tree for the high-speed hub's TT bus. If you don't initialize TT support in your application, `nScheduleTTLevel` will be ignored. Otherwise, `nScheduleLevel` must always be set to at least 3 more than `nScheduleTTLevel`.

As you can see in the below table, by setting the scheduling tree levels, you can control the longest interval allowed for a periodic endpoint. Intervals are always in power of 2 units. The units are in frames for a non-high-speed bus and microframes for a high-speed bus. The host will use an interval that is as long as possible given the number of levels in the scheduling tree, but the interval will be no longer than the desired period specified by the device.

Table 1. Scheduling Tree Levels

Levels in Tree	Longest interval allowed
1	Every 1 frame (non-high-speed bus) Every 1 microframe (high-speed bus)
2	Every 2 frames (non-high-speed bus) Every 2 microframes (high-speed bus)
3	Every 4 frames (non-high-speed bus) Every 4 microframes (high-speed bus)
4	Every 8 frames (non-high-speed bus) Every 8 microframes (high-speed bus)

The default settings for `nScheduleLevel` and `nScheduleTTLevel` are 9 and 6, respectively. That is an interval no longer than every 256 frames (or microframes) for the native host bus and no longer than every 32 frames for the high-speed hub's TT bus.

The maximum allowed setting for `nScheduleLevel` is 12 and the maximum setting allowed for `nScheduleTTLevel` is 9.

The default settings are recommended.

3 Transaction Translator Memory Requirements

Below is the equation to calculate the memory requirements. If you choose not to use the Transaction Translator, there are no additional memory requirements. The equation returns the number of bytes required. The inputs are: number of TTs, SCHEDULE_LEVEL and SCHEDULE_TT_LEVEL.

N=Number of TTs, X=SCHEDULE_LEVEL, Y=SCHEDULE_TT_LEVEL

RequiredMemory = 76 + ALIGN4 (COSTTREE_N(X) x 2) +

(N x (136 + ALIGN4(COSTTREE_N(Y) x 2)))

COSTTREE_N(Z) = ((1<<Z) -1)

ALIGN4(X) = ((X + 3) >> 2) << 2

The table below shows the memory requirements for some possible configurations.

Table 2. Transaction Translator Memory Requirements

N=Number of TTs
X=SCHEDULE_LEVEL
Y=SCHEDULE_TT_LEVEL
	Memory required in bytes

N=1, X=6, Y=3	356
N=7, X=9, Y=6	2948
N=21, X=9, Y=6	6644
N=147, X=6, Y=3	22,548
N=147, X=9, Y=6	39,908