Microsemi VPROC Apps

V1.0.1

Generated by Doxygen 1.8.6

Wed Aug 2 2017 15:37:38

Contents

Chapter 1

Main Page

1.1 Introduction

This is an document summarizes test suite for verifying Microsemi Voice Device SDK.

1.2 Build Instructions

Sample Makefile available with apps that build sample applicatons for linux platform as userspace apps. User can modify Makefile as per their platform. In exisiting Makefile,all sample applications are built as an independent execuatable binaries.hbi_test sample app is built by default. To enable compilation of other sample, app following TEST Macros are defined

HBI_LOAD_GRAMMAR - Builds grammar loading

HBI_LOAD_FIRMWARE - Build firmware and configuration record loading app

Example Make Command:

make apps HBI_TEST=1- This build only hbi_test sample app

make apps HBI_LOAD_GRAMMAR=1 - Builds hbi_load_grammar

make apps TEST_LDFWRCFG=1 - Builds hbi_load_firmware

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

Chapter 3

Data Structure Index

3.1 Data Structures

An initialized instance of that structure below must be defined within the example apps.

.bus_num:	specifies the SPI or I2C bus number of the host controller to which the ZL380xx device is interfaced to
.dev_add:	specifies the SPI chip select of I2C address of the ZL380xx device
.pfirmware:	must point to the firmware data array if the firmware *.h is to be compiled with the app statically, or NULL
.pconfig:	must point to the config record data array if the config *.h is to be compiled with the app statically, or NULL

Example definition for two ZL380xx devices interfaced to the host via SPI bus 0 and chip selects 0 and 1.
The device Id for these definition pertains to the index of the array. Example definition below aloows two device Ids 0 and 1

```
static hbi_dev_info_t hbi_devices_info[] = {


   {
      .bus_num = 0,   /*SPI or I2C bus number*/
      .dev_addr = 0,  /*SPI chip select or I2C address*/
      .pFirmware = NULL, /*a pointer to the firmware data array  from *.h if in c code format*/
      .pConfig = NULL,  /*a pointer to the config data array  from *.h if in c code format*/
   },
   {
      .bus_num = 0,
      .dev_addr = 1,
      .pFirmware = NULL,
      .pConfig = NULL,

   }
};
```

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Module Documentation

5.1 hbi_test

hbi_test is a simple application to test HBI driver functions and demonstrate its usage.

5.1.1 Detailed Description

Test is based on various TEST macros and command line options User can enable/disable them at compile-time to validate specific set of functions.

Test Macros to test various hbi driver feature

TEST_RST - Resets the device. A successful reset verify HBI_read and HBI_write function calls

TEST_LOAD_FWRCFG_FROM_FLASH - Test firmware loading from flash. Load image number 1 from flash.

TEST_ERASE_IMAGE - Test Erasing a particular image from flash. Erase image number 1 from flash

TEST_ERASE_FLASH - Test Erasing whole flash

To Execute test case, simply run:

. hbi_test

To read specific register

. hbi_test -d <device Id> -r <reg address="" in="" hex>=""> <number of="" bytes="" to="" read="" in=""

decimal>="">, example To read 10 bytes from address 0x200

. hbi_test -d 0 -r 0x200 10

To write single word at specific register

hbi_test -d <device Id> -w <reg address="" in="" hex>=""> <16-bit length word>

Example, to write 0xDEAD at register 0x200 to device Id 0

hbi_test -d 0 -w 0x200 0xDEAD

To write multiple words starting register 0x200

hbi_test -d 0 -w 0x200 0xDEAD 0xBEEF

To Display help menu

. hbi_test -h

5.2 hbi_load_grammar

hbi_load_grammar is a sample application to load an ASR grammar into the ZL380xx device and optionally save it to flash.

5.2.1 Detailed Description

The application allows to load a *.bin grammar file into the device at runtime. The binary (*.bin) grammar image

must be generated using the Python tool /apps/Python/tw_grammar_converter.py

The generated *.bin grammar file is a combination of both an ASR command file and an ASR trigger file

Press 'ctrl-c' to exit test.

For more description, please run 'hbi_load_grammar -h'.

Test support two input options at run time: -h = help display

```
Usage: ./hbi_load_grammar -[d deviceId] [-q] [-l binPath] [-f] [-s flashSlot] [-h]
  - "-q":
     Query the number of grammars stored in flash
  - "-l binPath":
     Load a grammar from a bin file to RAM
     Note: generate the bin file using "tw_grammar_converter.py"
  - "-f":
     Save the grammar from RAM to flash
  - "-s flashSlot":
     Swap grammars from flash
  - "-h":
     Prints that help

Ex: hbi_load_grammar -d 0 -l my_grammar.bin
```

5.3 hbi_load_firmware

This sample app demonstrates how to use the SDK functions to load a compatible ZL380xx firmware and configuration record into a ZL380xx Microsemi Vproc device.

It supports both Dynamic and Static image loading.

Firmware and Configuration Record Image can be compiled as static C-based (*.h) array along with the app or can be loaded dynamically at run time.

For static compilation, user needs to un-comment or define the following two macros within the /apps/C/makefile at compile time.

LOAD_FWR_STATIC - Defined at Makefile. if defined, expects user to provide a C header file containing firmware boot image for static compilation

FWR_C_FILE - C header file containing firmware boot image for static compilation. if not provided, then hbi_test would throw compile-time error

LOAD_CFGREC_STATIC - Defined at Makefile.if defined, expects user to provide C files containing configuration record

CFGREC_C_FILE C files to be included during compilation
. hbi_load_firmware

For dynamic bootimage loading to device (undef LOAD_FWR_STATIC), run

. hbi_load_firmware -d <device Id> -i <firmware filename>=""> -c <cfgrec filename>="">

where firmware filename = binary firmware image with .bin extension.

configuration record filename = configuration record file with .cr2 extension

For loading and saving to flash, use -s option

. hbi_load_firmware -d <device Id> -i <filename> -c <cfgrec filename>=""> -s , or

.
. For static loading use the command below
. hbi_load_firmware -d <device Id>
. or, if the firmware must be save to flash

. hbi_load_firmware -d <device Id> -s

NOTE - firmware binary image or c-output is generated by running Microsemi Provided Image convertor

tool.

Chapter 6

Data Structure Documentation

6.1 cfgrec_t Struct Reference

Data Fields

- void ∗**pImage**
- unsigned short **ImageLen**
- unsigned short **configBlockSize**

6.1.1 Detailed Description

Definition at line 4 of file app_util.h.

The documentation for this struct was generated from the following file:

- apps/app_util.h

Chapter 7

File Documentation

7.1 apps/hbi_test.c File Reference

```
#include <unistd.h>
#include <stdint.h>
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <string.h>
#include "typedefs.h"
#include "chip.h"
#include "hbi.h"
```

- #define **MAX_RW_SIZE** 64
- char ∗ **strtoh** (char ∗str, int len, unsigned char ∗val)
- int **hbi_test** (int argc, char ∗∗argv)
- void **main** (int argc, char ∗∗argv)

Index