

CC32xx Idle Profile Application

Overview

Idle profile enables the user to measure current values, power consumption and other such parameters for CC3200, when the device is essentially idle(both NWP and APPS subsystems in low power deep sleep condition). The other main objective behind this application is to introduce the user to the easily configurable power management framework.

[Return to CC31xx & CC32xx Home Page](#)
[Return to CC31xx Sample Applications](#)

Application details

Power Management Framework

CC3200 device have multiple power modes, which can be exercised by the user as per the requirement of their application. Power management framework makes it easier to specify the power policy for APPS MCU. All user has to configure is the lowest power mode the device should go into. The framework tries to hide the intricacies of the underlying settings and helps the user to experience the low power mode capabilities of the device in a simple manner. If one wishes to have the custom settings for different power modes or use the framework for a different device, it can be accomplished with changing a couple of files. For more information regarding Power Management Framework, please refer to CC32xx Power Management Framework

Current Measurement

This application specify the Low Power Deep Sleep(LPDS) as the lowest power mode. At times, both NWP and APPS subsystem will be in LPDS. In which case, the current values can be as low as in the order of hundreds of microseconds. The procedure for measuring current can be found at CC3200 Low Power Modes Current Measurement page.

Program Flow

Most the parameters user will need to modify are specified as MACROS

- SSID_NAME - Name of the Access Point used for this application.
- GPIO_SRC_WKUP - Gpio number used as wake up source.
- APP_UDP_PORT - Port number on which the device will wait for udp packets.
- LPDS_DUR_SEC - Time (in seconds) after which the device will come out of LPDS.
- LPDS_DUR_NSEC - Time (in addition to HIB_DUR_SEC) in nanoseconds after which the device will come out of LPDS.

```
#define SSID_NAME          "cc3200demo"
#define GPIO_13            13
#define GPIO_SRC_WKUP      GPIO_13
#define APP_UDP_PORT       5001
#define LPDS_DUR_SEC       60
#define LPDS_DUR_NSEC      0
```

Note: Security parameters for the AP can be configure inside **WlanConnect** function in "main.c".

Note: Whatever Gpio is used as wake up source has to added in **gpio_list** array in "user_app_config.h" and pinmux

for the same Gpio has to be added in "pinmux.c".

To start with, the board must be initialized and the power management framework have to be loaded.

```
//  
// Board Initialization  
//  
BoardInit();  
  
//  
// Configure the pinmux settings for the peripherals exercised  
//  
PinMuxConfig();  
  
//  
// Initialize the platform  
//  
platform_init();  
  
//  
// Configuring UART  
//  
g_tUartHndl = uart_open(PRCM_UARTA0);
```

Which will be followed by creation of two tasks.

1. One for creating and handling UDP Server
2. Other for setting timer and GPIO as wake source from low power modes

```
//  
// Task creation for providing host_irq as a wake up source(LPDS)  
//  
osi_TaskCreate(UDPServerTask,  
               (const signed char *)"UDP Server waiting to recv packets",  
               OSI_STACK_SIZE, NULL, tskIDLE_PRIORITY+1, NULL );  
  
//  
// setting up timer and gpio as source for wake up from LPDS  
//  
osi_TaskCreate(TimerGPIONTask, (const signed char*)  
               "Configuring Timer and GPIO as wake src",  
               OSI_STACK_SIZE, NULL, 1, NULL );
```

Task 1: UDPServerTask

1. Blocked on a message from the Other Task(waiting for simplelink start and connection to AP).
2. Creates a UDP server and wait for UDP packet in a while loop. As soon as the NWP subsystem receives any UDP packet, it will bring the APPS subsystem out of LPDS, if not already.

```
//
// waiting for the other task to start simplelink and connection to the AP
//
osi_MsgQRead(&g_tConnectionFlag, &ucSyncMsg, OSI_WAIT_FOREVER);
.
.
//
// creating a UDP socket
//
iSockDesc = sl_Socket(SL_AF_INET, SL_SOCKET_DGRAM, 0);
.
.
//
// waiting on a UDP packet
//
iRetVal = sl_RecvFrom(iSockDesc, g_cBuffer, BUFF_SIZE, 0,
                     ( SlSockAddr_t *)&sClientAddr
                     (SlSocklen_t*)&iAddrSize );

if(iRetVal > 0)
{
    //
    // signal the other task about receiving the UDP packet
    //
    osi_MsgQWrite(&g_tWkupSignalQueue, &ucQueueMsg, OSI_WAIT_FOREVER);
}
```

Task 2: TimerGPIOTask

1. Starts simplelink, switch to station mode(if not already), set NWP power policy and connects to the AP.
2. Upon successful connection unblocks the other task.
3. Set timer and GPIO as wake sources from low power modes.
4. Set Power policy for the APPS MCU.

```
//
// starting the simplelink
//
iMode = sl_Start(NULL, NULL, NULL);

//
// Switch to STA mode if device is not
//
SwitchToStaMode(iMode);

//
```

```
// Set the power management policy of NWP
//
sl_WlanPolicySet(SL_POLICY_PM, SL_NORMAL_POLICY, NULL, 0);
.
.
//
// connecting to the Access Point
//
if(-1 == WlanConnect())
{
    sl_Stop(SL_STOP_TIMEOUT);
    UART_PRINT("Connection to AP failed\n\r");
}
else
{
    UART_PRINT("Connected to AP\n\r");
    //
    //signal the other task about the sl start and connection to the AP
    //
    osi_MsgQWrite(&g_tConnectionFlag, &ucSyncMsg, OSI_WAIT_FOREVER);
}
.
.
//
// setting Timer as one of the wakeup source
//
tTimerHndl = SetTimerAsWkUp();

//
// setting some GPIO as one of the wakeup source
//
tGPIOHndl = SetGPIOAsWkUp();
.
.
//
// setting Apps power policy
//
lp3p0_setup_power_policy(POWER_POLICY_STANDBY);
```

Porting to TI-RTOS

For porting this application on TI-RTOS, please refer to **Steps to build a new application on CC3200 with TI-RTOS:** and **Notes** section on CC32xx_TI-RTOS page. **cc_idle_task_pm()** is the function needed to be hooked to the idle task.

Source Files briefly explained

- **main.c** - The main file implementing the idle profile.
- **lp3p0_board.c** - Board specific initialization for Power Management framework .
- **lp3p0_plat_ops.c** - Board specific APIs like IO parking and framework loading.

Supporting Files

- **pinmux.c** - Generated by the PinMUX utility.

Common Files

- **startup_ccs.c** - CCS related functions
- **startup_ewarm.c** - IAR related functions
- **wdt_if.c** - Interface file for Watchdog Timer

Prerequisites

This application facilitates the user to get the current numbers for LPDS mode. The prerequisites for running this application are the following:

- Ensure that the device is in Station mode
- Delete all previously stored AP profiles.

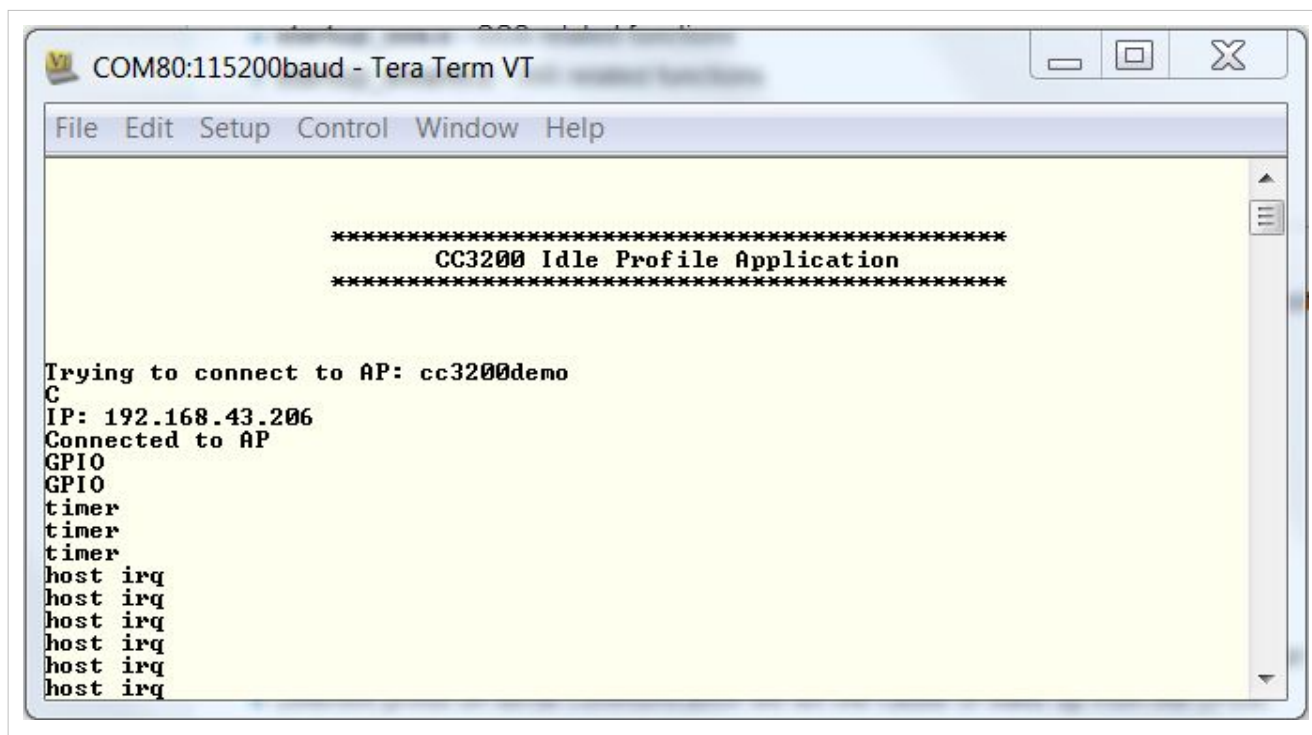
Note: One can simply run UDP Socket application before executing this application, that will take care of the above mentioned points.

Usage

- Modify MACROS according to the requirement and recompile.
- 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
 - Run the application preferably from FLASH rather than from the debugger as debugger would disconnect in LPDS.
 - Different prints on serial communication will tell the cause of wake up from the LPDS.
 - Red LED (GPIO 09) will turn off whenever the device enters LPDS.
-



The screenshot shows a Tera Term window titled "COM80:115200baud - Tera Term VT". The window has a menu bar with "File", "Edit", "Setup", "Control", "Window", and "Help". The main text area displays the following output:

```
*****  
CC3200 Idle Profile Application  
*****  
  
Trying to connect to AP: cc3200demo  
C  
IP: 192.168.43.206  
Connected to AP  
GPIO  
GPIO  
timer  
timer  
timer  
host irq  
host irq  
host irq  
host irq  
host irq  
host irq
```

Limitations/Known Issues

Refer to CC32xx_Power_Management_Framework for limitations and known issues.

Article Sources and Contributors

CC32xx Idle Profile Application *Source:* <http://processors.wiki.ti.com/index.php?oldid=179787> *Contributors:* A0221015, Codycooke, Jitgupta, Malokyle

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:Cc32xx return sample apps.png *Source:* http://processors.wiki.ti.com/index.php?title=File:Cc32xx_return_sample_apps.png *License:* unknown *Contributors:* A0221015

Image:Idle_profile_screenshot.JPG *Source:* http://processors.wiki.ti.com/index.php?title=File:Idle_profile_screenshot.JPG *License:* unknown *Contributors:* Jitgupta