

# CC32xx UDP Socket Application

## Overview

A CC3200 device can communicate over network using standard communication protocols such as TCP and UDP. This can be accomplished without even using a Real Time Operating System (RTOS).

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

## Application details

This particular application illustrates how this device can be used as a client or server for UDP communication. Developers/users can refer the function or re-use them while writing new application. The device will connect to an AP (access point), with SSID for AP stored as a macro in the application. Initially, the application implements a UDP client and sends 1000 UDP packets to a socket address, port number and ip address specified as macros. Zero will be the expected return code.

A different return code would mean that socket error has occurred.

Default setting is defined as in following MACROs, which can be changed either in source code or at runtime.

```
#define SSID_NAME      "cc3200demo"
#define IP_ADDR        0xc0a80167
#define PORT_NUM       5001
#define UDP_PACKET_COUNT 1000
```

## Source Files briefly explained

- **main.c** – main file calls simplelink APIs to connect to the network, create socket and use it to communicate over UDP by acting as a UDP client or server.
- **pinmux.c** - pinmux file to mux the device to configure UART peripheral
- **startup\_ccs.c** – CCS specific vector table implementation for interrupts.
- **startup\_ewarm.c** – IAR workbench specific vector table implementation for interrupts.

## Code flow

### Connection

```
void main()
{
    ...
    lRetVal = sl_Start(0, 0, 0);
    ...
    // Connecting to WLAN AP - Set with static parameters defined at the top
    // After this call we will be connected and have IP address
    lRetVal = WlanConnect();
    ...
    /* following calls depend on user's input at runtime */

    // Before proceeding, please make sure to have a server waiting on PORT_NUM
```

```
BsdUdpClient (PORT_NUM);

// After calling this function, you can start sending data to CC3200 IP
// address on PORT_NUM
BsdUdpServer (PORT_NUM);

...
}
```

---

## UDP Client

```
int BsdUdpClient(unsigned short usPort)
{
    ...
    //Open a socket with standard parameters
    iSockID = sl_Socket(SL_AF_INET,SL_SOCKET_DGRAM, 0);
    if( iSockID < 0 )
    {
        // error
        ASSERT_ON_ERROR(UCP_CLIENT_FAILED);
    }

    //Send packet using the sl_Send API call
    ...
    iStatus = sl_SendTo(iSockID, g_cBsdBuf, sTestBufLen, 0,(SlSockAddr_t *)&sAddr, iAddrSize);
    if( iStatus <= 0 )
    {
        // error
        sl_Close(iSockID);
        ASSERT_ON_ERROR(UCP_CLIENT_FAILED);
    }
    ...
    //Close the socket
    sl_Close(iSockID);

    return SUCCESS;
}
```

Sending the UDP Packets is a simple three step process

1. Open the socket
2. Send the packets
3. Close the socket

---

## UDP Server

```
int BsdUdpServer(unsigned short usPort)
{
    ...
    iSockID = sl_Socket(SL_AF_INET,SL_SOCKET_STREAM, 0);
    if( iSockID < 0 )
    {
```

```
// error
ASSERT_ON_ERROR(UCP_SERVER_FAILED);
}

...

iStatus = sl_Bind(iSockID, (SlSockAddr_t *)&sLocalAddr, iAddrSize);
if( iStatus < 0 )
{
    // error
    ASSERT_ON_ERROR(UCP_SERVER_FAILED);
}

iStatus = sl_RecvFrom(iSockID, g_cBsdBuf, sTestBufLen, 0, (SlSockAddr_t *)&sAddr, (SlSocklen_t*)&iAddrSize );
if( iStatus < 0 )
{
    // error
    sl_Close(iSockID);
    ASSERT_ON_ERROR(UCP_SERVER_FAILED);
}

...

sl_Close(iSockID);

return SUCCESS;
}
```

Steps for receiving UDP Packets as a UDP server are as follows

1. Open the socket
2. Create a UDP server
3. receive packets
4. Close the socket

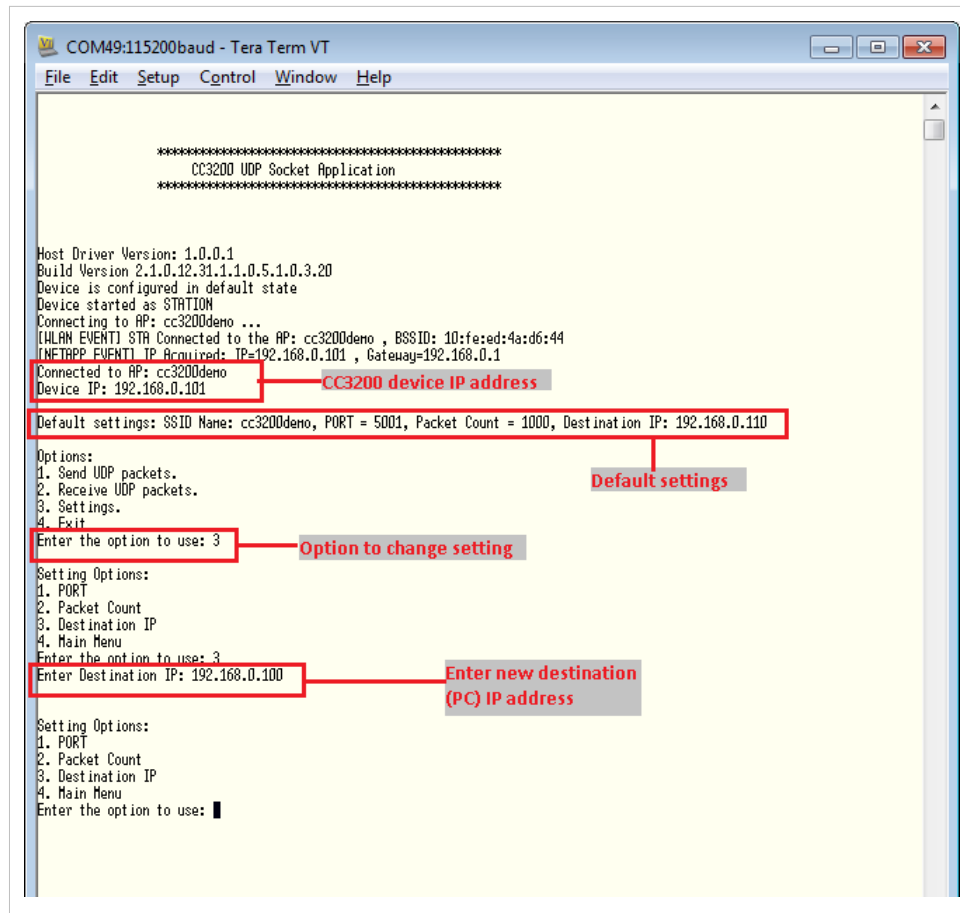
## Usage

- Setup a serial communication application (HyperTerminal/TeraTerm). For detail info visit Terminal setup

**Note:** Disable PC anti-virus while running iperf.

On the host PC. The settings are:

- **Port:** Enumerated COM port (CC3200LP Dual port)
- **Baud rate:** 115200
- **Data:** 8 bit
- **Parity:** None
- **Stop:** 1 bit
- **Flow control:** None
- Run this application (udp\_socket) from IAR/CCS or Flash the bin file to device.
- Connect a PC to same AP over which device has connected.
- Get the ip address of the PC and fill this value for IP\_ADDR macro or change the setting as specified in snapshot-



- Change the other setting (port, SSID name, packet count) as per requirement.
- Choose the options
  - 1: Send UDP packets
  - 2: Receive UDP packets

after selecting above options run **iperf** command on PC command prompt as given in TeraTerm/HyperTerminal screen.

- Observe the execution flow to understand the working.

## Limitations/Known Issues

None.

# Article Sources and Contributors

**CC32xx UDP Socket Application** *Source:* <http://processors.wiki.ti.com/index.php?oldid=184853> *Contributors:* A0221015, Beatrice, 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](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](http://processors.wiki.ti.com/index.php?title=File:Cc32xx_return_sample_apps.png) *License:* unknown *Contributors:* A0221015

**Image:CC32xx Udp Socket Terminal runScreen 1.0.0.png** *Source:* [http://processors.wiki.ti.com/index.php?title=File:CC32xx\\_Udp\\_Socket\\_Terminal\\_runScreen\\_1.0.0.png](http://processors.wiki.ti.com/index.php?title=File:CC32xx_Udp_Socket_Terminal_runScreen_1.0.0.png) *License:* unknown *Contributors:* Jitgupta