

# NUC472 uIP Client

Example Code Introduction for 32-bit NuMicro® Family

## Information

Application	This code uses Ethernet function to report a message to a host periodically.
BSP Version	NUC472 Series BSP CMSIS V3.03.000
Hardware	NuTiny-SDK-NUC472

*The information described in this document is the exclusive intellectual property of Nuvoton Technology Corporation and shall not be reproduced without permission from Nuvoton.*

*Nuvoton is providing this document only for reference purposes of NuMicro microcontroller based system design. Nuvoton assumes no responsibility for errors or omissions.*

*All data and specifications are subject to change without notice.*

*For additional information or questions, please contact: Nuvoton Technology Corporation.*

[www.nuvoton.com](http://www.nuvoton.com)

# 1 Function Description

## 1.1 Introduction

This example code uses NUC472's Ethernet function to report a message to a host on the network periodically. The network data processing uses uIP stack. uIP is a simple network stack. It can directly call the API to perform various network actions, including connecting to a host, sending and receiving packets.

## 1.2 Principle

This example code connects to the host IP: 192.168.1.1 : 80 by tcp\_client\_appcall(). Then it will send the string "Hello World!!" every six seconds. Finally, it disconnects the TCP/IP communication. It can be applied to the system that needs to report the host periodically.

## 1.3 Demo Result

The screenshot shows a serial communication software interface with the following sections:

- Serial Option:** COM Port (COM1), Baud Rate (19200), Data Bits (8), Parity (None), Stop Bits (1), Flow Control (None). Includes STEAL and Connect buttons.
- TCP Option:** Server IP (192.168.1.1), Port (80), radio buttons for Server and Client.
- UDP Option:** Target IP, Port, and Connect button.
- User Mode:** Font Set, File On.
- Char Mode:** Show HEX, Send HEX, Send CR, Send LF.
- Error Check Mode:** Error Code (NONE), Data(Char) Start/End (\$, \*), Number Start/End (0, 0).
- Communication Data:** A log showing TCP communication cycles and received data:
 

```
TCP Communication End
TCP Communication End
0000 R [16:50:07'339] : Hello World!!
TCP Communication Start
TCP Communication Start
TCP Communication End
TCP Communication End
0000 R [16:50:01'346] : Hello World!!
TCP Communication Start
TCP Communication Start
TCP Communication End
TCP Communication End
0000 R [16:49:55'334] : Hello World!!
TCP Communication Start
TCP Communication Start
TCP Communication End
TCP Communication End
0000 R [16:49:49'339] : Hello World!!
TCP Communication Start
TCP Communication Start
```
- Edit Data(Send Data):** A text input field at the bottom.

## 2 Code Description

Main function sets IP address :

```
int main(void)
{
    uip_ipaddr_t ipaddr = {0, 0};

    SYS_Init();
    UART_Open(UART0, 115200);

    printf("NUC472 uIP sample code start\n");

    /* Select RMI interface by default*/
    EMAC_Open(ethaddr.addr);
    NVIC_EnableIRQ(EMAC_TX_IRQn);
    NVIC_EnableIRQ(EMAC_RX_IRQn);
    EMAC_ENABLE_RX();
    EMAC_ENABLE_TX();
    /*Setting Timer*/
    TIMER_Open(TIMER0, TIMER_PERIODIC_MODE, 1);
    TIMER_EnableInt(TIMER0);
    NVIC_EnableIRQ(TMR0_IRQn);
    TIMER_Start(TIMER0);

    /* TCP/IP address setting*/
    uip_ipaddr(ipaddr, 192,168,1,2);
    uip_sethostaddr(ipaddr);
    uip_ipaddr(ipaddr, 192,168,1,10);
    uip_setdraddr(ipaddr);
    uip_ipaddr(ipaddr, 255,255,255,0);
    uip_setnetmask(ipaddr);

    /* MAC address setting*/
    uip_setethaddr(ethaddr);
    /*TCP connection*/
    tcp_client_connect();
    /*uIP network processing*/
    while(1) {
        uip_polling();
        if((uip_closed()||uip_aborted())&&((curTime-sendTime)>=6)) {
```

```
        tcp_client_connect();
        sendTime=curTime;
    }
}
}
```

Use the port 80 to handle the message communication :

```
void httpd_appcall(void)
{
    switch(uiplib_conn->rport) {
        case HTONS(80):
            tcp_client_appcall(); /*application callback*/
            break;
        default:
            break;
    }
}

void tcp_client_senddata(void)
{
    struct tcp_appstate *s = (struct tcp_appstate *)&uiplib_conn->appstate;
    if(s->textlen>0) uip_send(s->textptr, s->textlen);
}

void tcp_client_appcall(void)
{
    uint8_t string_pass[] = "Hello World!!";
    struct tcp_appstate *s = (struct tcp_appstate *)&uiplib_conn->appstate;

    if((curTime-sendTime)>=6) {
        s->textptr=string_pass;
        s->textlen=strlen((const char*)string_pass);
        tcp_client_senddata(); /*Send the message*/
        sendflag=1;
        sendTime=curTime;
    }
    if((sendflag==1)&&(uiplib_acked()!=0)) {
        uip_close(); /*Disconnect network*/
        sendflag=0;
    }
}
```

```
}  
}
```

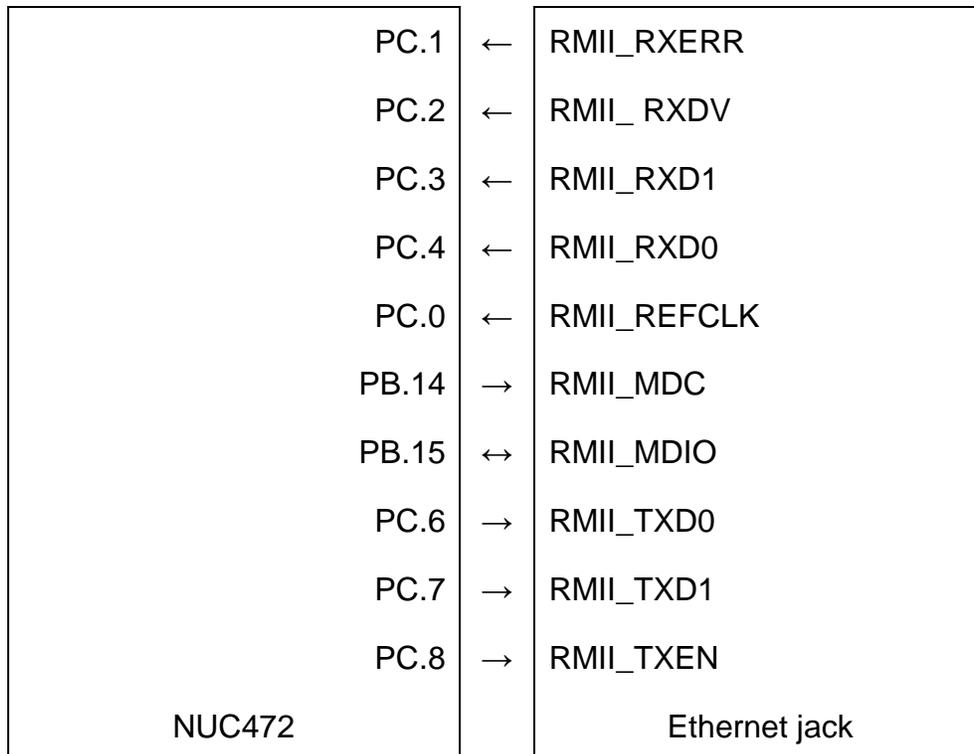
### 3 Software and Hardware Environment

- **Software Environment**

- BSP version
  - ◆ NUC472 Series BSP CMSIS V3.03.000
- IDE version
  - ◆ Keil uVersion 5.26

- **Hardware Environment**

- Circuit components
  - ◆ NuTiny-SDK-NUC472
- Diagram



## 4 Directory Information

📁 EC\_NUC472\_uIP\_Client\_V1.00

📁 Library	Sample code header and source files
📁 CMSIS	Cortex <sup>®</sup> Microcontroller Software Interface Standard (CMSIS) by Arm <sup>®</sup> Corp.
📁 Device	CMSIS compliant device header file
📁 StdDriver	All peripheral driver header and source files
📁 SampleCode	
📁 ExampleCode	Source file of example code
📁 ThirdParty	
📁 uip-0.9	uIP is a very small implementation of the TCP/IP stack that is written by Adam Dunkels <adam@sics.se>. More information can be obtained from the uIP homepage at <a href="https://github.com/adamdunkels/uip">https://github.com/adamdunkels/uip</a>

## 5 How to Execute Example Code

1. Browsing into sample code folder by Directory Information (section 4) and double click NUC472\_ulP\_Client.uvproj.
2. Enter Keil compile mode
  - a. Build
  - b. Download
  - c. Start/Stop debug session
3. Enter debug mode
  - a. Run

## 6 Revision History

Date	Revision	Description
Jul. 11, 2019	1.00	1. Initially issued.

---

### **Important Notice**

Nuvoton Products are neither intended nor warranted for usage in systems or equipment, any malfunction or failure of which may cause loss of human life, bodily injury or severe property damage. Such applications are deemed, "Insecure Usage".

Insecure usage includes, but is not limited to: equipment for surgical implementation, atomic energy control instruments, airplane or spaceship instruments, the control or operation of dynamic, brake or safety systems designed for vehicular use, traffic signal instruments, all types of safety devices, and other applications intended to support or sustain life.

All Insecure Usage shall be made at customer's risk, and in the event that third parties lay claims to Nuvoton as a result of customer's Insecure Usage, customer shall indemnify the damages and liabilities thus incurred by Nuvoton.

---

*Please note that all data and specifications are subject to change without notice.  
All the trademarks of products and companies mentioned in this datasheet belong to their respective owners.*