NUC240 ENC28J60 Driver for uIP/HTTP

Example Code Introduction for 32-bit NuMicro®Family

Information

Application	This sample code uses NUC240 to drive ENC28J60 to achieve uIP/httpd function
BSP Version	NUC240BSP_v3.00.001
Hardware	NuTiny-SDK-NUC240V

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

1 Function Description

1.1 Introduction

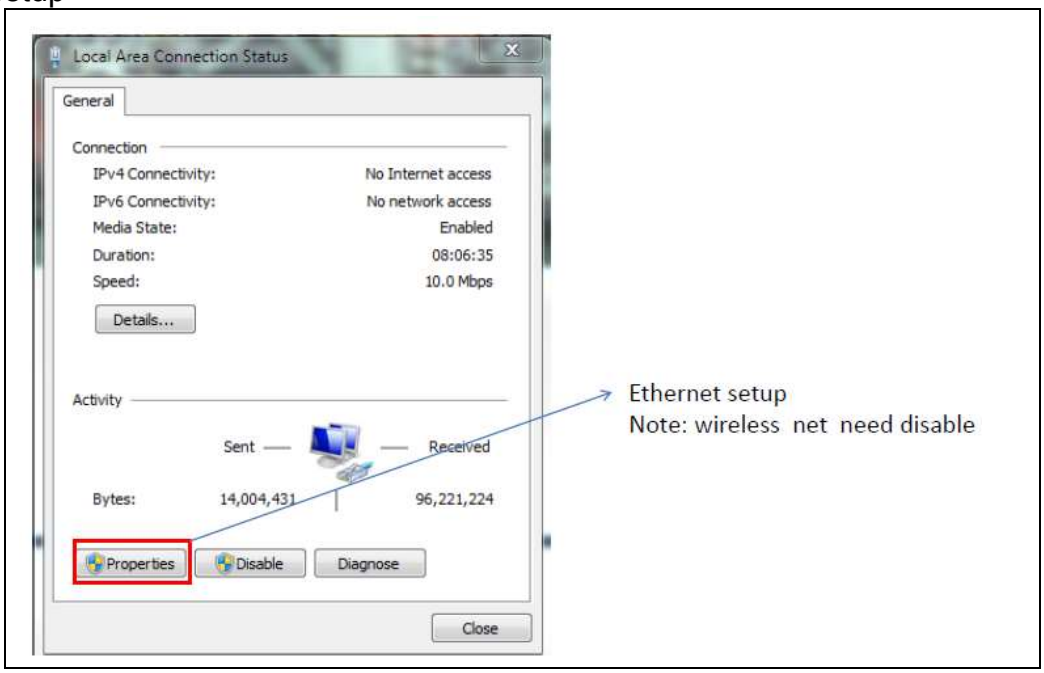
This sample code uses NUC240 to drive ENC28J60 though SPI interface to achieve uIP/httpd function

1.2 Principle

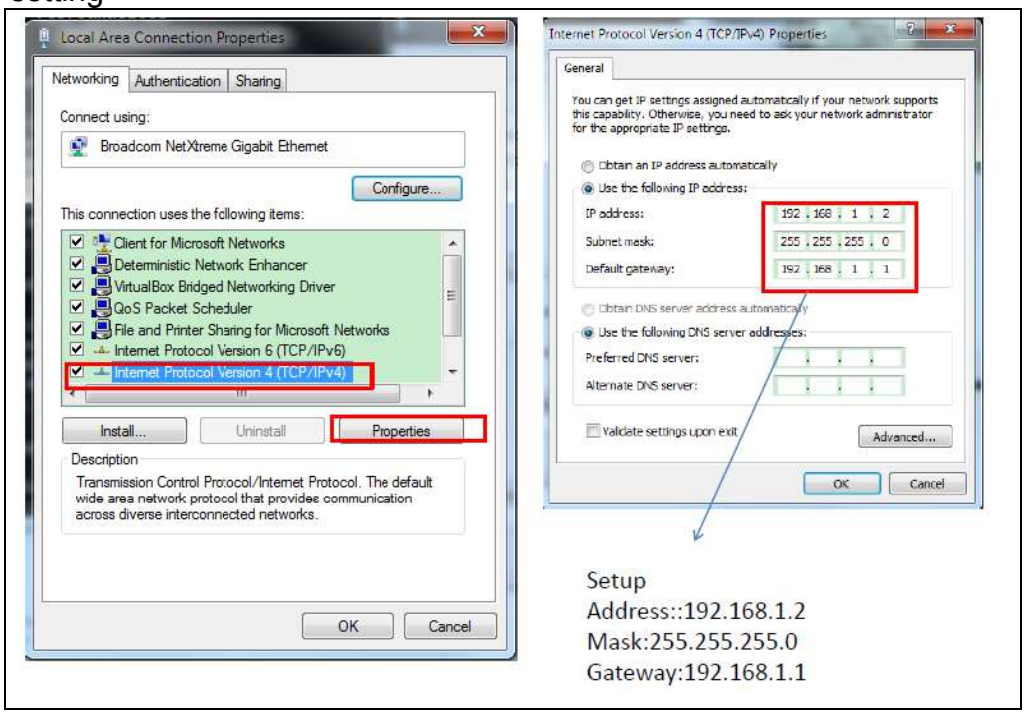
The uIP implementation the TCP/IP network protocol stack, the ENC28J60 adopts IEEE802.3 as an Ethernet controller and the interface is SPI, so we can combine the uIP, ENC28J60 with NUC240 for web services application.

1.3 Demo result

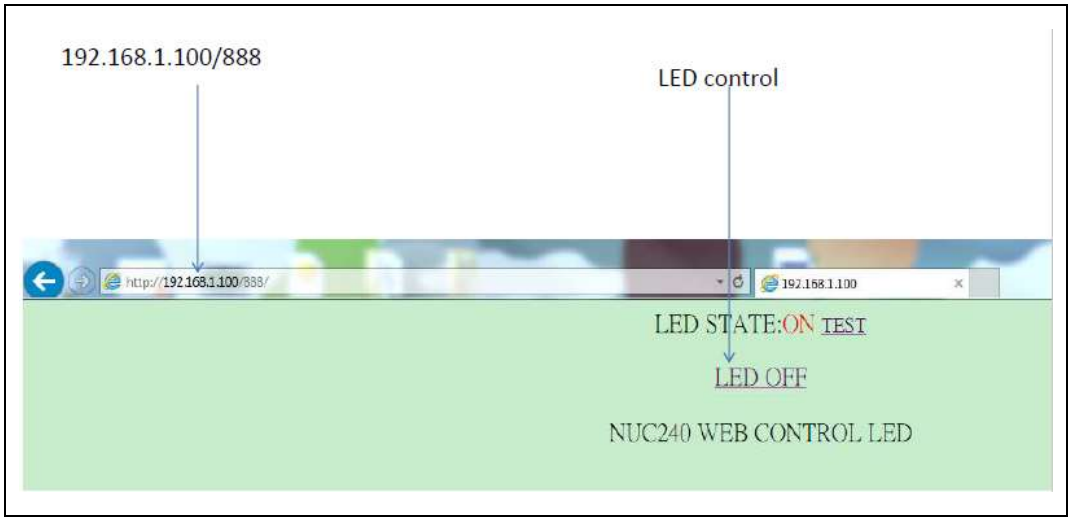
Ethernet setup



IP address setting



IE to connect web services



2 CodeDescription

Initialize ENC28J60：

```
int32_t ENC28J60_Init(unsigned char * macaddr)
{
 unsigned int retry=0;
 ENC28J60_Reset();
 ENC28J60_Write_Op(ENC28J60_SOFT_RESET,0,ENC28J60_SOFT_RESET);
. . . . . .
}
```

uIP initialize and IP address setup：

```
int simple_server(void)
{
   unsigned int plen;
   unsigned int dat_p;
   unsigned char i = 0;
   unsigned char cmd_pos = 0;
   unsigned char cmd;
   unsigned char payloadlen = 0;
   char str[30];

   char cmdval;

   /*initialize enc28j60*/
   ENC28J60_Init(mymac);

   str[0]=(char)ENC28J60_Get_EREVID();

   init_ip_arp_udp_tcp(mymac, myip, mywwwport);
    . . . . . .

}
```

ENC28J60 transmit data：

```
void ENC28J60_Packet_Send(unsigned int len, unsigned char *packet)
{
    // Set the write pointer to start of transmit buffer area
    ENC28J60_Write(EWRPTL, TXSTART_INIT & 0xFF);
    ENC28J60_Write(EWRPTH, TXSTART_INIT >> 8);

    // Set the TXND pointer to correspond to the packet size given
    ENC28J60_Write(ETXNDL, (TXSTART_INIT + len) & 0xFF);
    ENC28J60_Write(ETXNDH, (TXSTART_INIT + len) >> 8);
    // write per-packet control byte (0x00 means use macon3 settings)
    ENC28J60_Write_Op(ENC28J60_WRITE_BUF_MEM, 0, 0x00);
    //printf("len:%d\r\n",len);
    // send the contents of the transmit buffer onto the network
    ENC28J60_Write_Buf(len, packet);

    ENC28J60_Write_Op(ENC28J60_BIT_FIELD_SET, ECON1, ECON1_TXRTS);

    // Reset the transmit logic problem. See Rev. B4 Silicon Errata point 12.
    if ((ENC28J60_Read(EIR)&EIR_TXERIF))
        ENC28J60_Write_Op(ENC28J60_BIT_FIELD_CLR, ECON1, ECON1_TXRTS);
}
```

ENC28J60 recive data：

```
unsigned int ENC28J60_Packet_Receive(unsigned int maxlen, unsigned char *packet)
{
    unsigned int rxstat;
    unsigned int len;

    // check if a packet has been received and buffered
    if (ENC28J60_Read(EPKTCNT) == 0)
        return 0;

    // Set the read pointer to the start of the received packet
    ENC28J60_Write(ERDPTL, (NextPacketPtr));
    ENC28J60_Write(ERDPTH, (NextPacketPtr) >> 8);

    // read the next packet pointer
    NextPacketPtr = ENC28J60_Read_Op(ENC28J60_READ_BUF_MEM, 0);
    NextPacketPtr |= ENC28J60_Read_Op(ENC28J60_READ_BUF_MEM, 0) << 8;
```

```
        // read the packet length (see datasheet page 43)
        len = ENC28J60_Read_Op(ENC28J60_READ_BUF_MEM, 0);
        len |= ENC28J60_Read_Op(ENC28J60_READ_BUF_MEM, 0) << 8;
        len -= 4; //remove the CRC count
        // read the receive status (see datasheet page 43)
        rxstat = ENC28J60_Read_Op(ENC28J60_READ_BUF_MEM, 0);
        rxstat |= ENC28J60_Read_Op(ENC28J60_READ_BUF_MEM, 0) << 8;

        // limit retrieve length
        if (len > maxlen - 1)
            len = maxlen - 1;

        // check CRC and symbol errors (see datasheet page 44, table 7-3):
        // The ERXFCON.CRCEN is set by default. Normally we should not
        // need to check this.
        if ((rxstat & 0x80) == 0)
            len = 0; // invalid
        else
            ENC28J60_Read_Buf(len, packet); // copy the packet from the receive buffer

        // Move the RX read pointer to the start of the next received packet
        // This frees the memory we just read out
        ENC28J60_Write(ERXRDPTL, (NextPacketPtr));
        ENC28J60_Write(ERXRDPTH, (NextPacketPtr) >> 8);
        // decrement the packet counter indicate we are done with this packet
        ENC28J60_Write_Op(ENC28J60_BIT_FIELD_SET, ECON2, ECON2_PKTDEC);
        return (len);
    }
```

Data process:

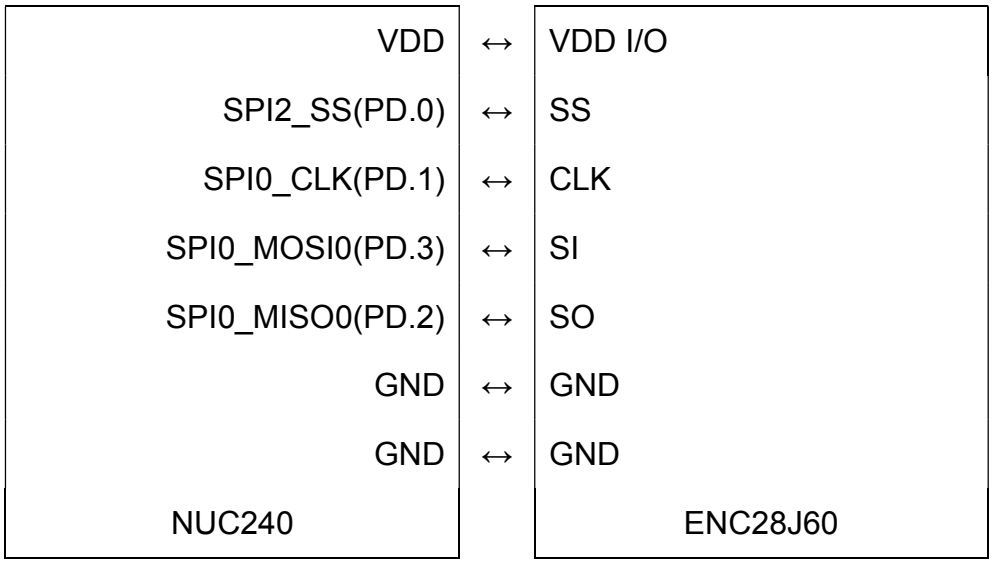
```
    //init the ethernet/ip layer:
    while (1)
       {
          // get the next new packet:
          plen = ENC28J60_Packet_Receive(BUFFER_SIZE, buf);

          /*plen will ne unequal to zero if there is a valid packet (without crc error) */
          if (plen == 0)
          {
             continue;
          }
          // arp is broadcast if unknown but a host may also
```

```
        // verify the mac address by sending it to
        // a unicast address.
        if (eth_type_is_arp_and_my_ip(buf, plen))
        {
           make_arp_answer_from_request(buf);
           continue;
        }
```

3 Software and Hardware Environment

- **Software Environment**
  - BSP Version
    - NUC240BSP_v3.00.001
  - IDE Version
    - Keil uVersion 5.2
- **Hardware Environment**
  - Circuit Components
    - NuTiny-SDK-NUC240
    - ENC28J60
  - Diagram

NUC240driver ENC28J60 though SPI interface.



4 Directory Information

EC_NUC240_SPI_ENC28J60_uIP_HTTP _V1.00	
Library	Sample code header and source files
CMSIS	Cortex® Microcontroller Software Interface Standard (CMSIS) V4.5.0 definitions by ARM® Corp.
Device	CMSIS compliant device header file
StdDriver	All peripheral driver header and source files
SampleCode	
ExampleCode	Source file of example code

5 How to Execute Example Code

1. According to Directory Information enter the KEIL folder in path “ExampleCode”, double click “NUC240_SPI_ENC28J60_uIP_HTTP.uvproj”
2. Enter the compilation mode interface
   a. Compilation
   b. Download code to Flash
   c. Enter/leave debugging mode
3. Enter the debugging mode interface
   a. Execution code

6 Revision History

Date	Revision	Description
Oct. 1, 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.