M451 Driver DM9051

Example Code Introduction for 32-bit NuMicro®Family

Information

Application	This sample code uses M451 to drive DM9051 to achieve uIP/httpd function
BSP Version	M451BSP_v3.01.003
Hardware	NuTiny-SDK-M451V

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

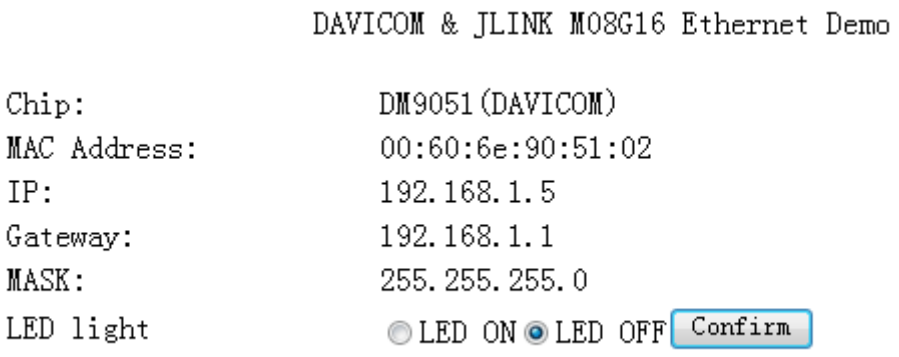
1.1 Introduction

This sample code uses M451 to drive DM9051 though SPI interface to achieve uIP/httpd function.

1.2 Principle

The DM9051 is a fully integrated and cost-effective low pin count single chip Fast Ethernet controller with a Serial Peripheral Interface (SPI), a 10/100M PHY and MAC, and 16K-byte SRAM. It is designed with low power and high performance process interface that supports 3.3V with 5V IO tolerance.
The PHY of the DM9051 can interface to the UTP3, 4, 5 in 10Base-T and UTP5 in 100Base-TX with HP Auto-MDIX. It is fully compliant with the IEEE 802.3u Spec. Its Auto-Negotiation function will automatically configure the DM9051 to take the maximum advantage of its 10M or 100M abilities.
The DM9051 supports IEEE 802.3az in PHY and MAC to save power consumption when Ethernet is idle. The IEEE 802.3x Full-Duplex flow control and Half-Duplex back-pressure functions are also supported to avoid Ethernet packet loss with link partner.
The slave SPI interface is designed to support SPI clock mode 0 and 3 that are compatible with the master SPI interface of CPU. The clock speed can be up to 50Mhz to co-operate with highest throughput master SPI. The SPI burst command format is code-effective to minimize the command overhead in access to DM9051 internal registers and packet data in memory.

1.3 Demo Result



2 Code Description

Initialize DM9051：

```
int32_t DM9051_INIT()
{
    int i, oft, lnk;
    uint32_t value=0;
    DM9051_Configuration();
  /* identfy DM9051   */
  value |= DM9051_Read_Reg(DM9000_VIDH) << 8;
  value |= DM9051_Read_Reg(DM9000_PIDL) << 16;
  value |= DM9051_Read_Reg(DM9000_PIDH) << 24;
  value |= DM9051_Read_Reg(DM9000_VIDL);
  . . . . . .
}
```

uIP initialize and IP address setup：

```
int main(void)
{
    . . . . . .
    uip_init();
    uip_arp_init();    // Clear arp table.
#ifdef __DHCPC_H__
  // setup the dhcp renew timer the make the first request
    timer_set(&dhcp_timer, CLOCK_SECOND * 600);
     dhcpc_init(&uip_ethaddr, 6);
#endif
    uip_ipaddr(ipaddr, 192,168,1,5);         //Host IP address
    uip_sethostaddr(ipaddr);
    uip_ipaddr(ipaddr, 192,168,1,1);         //Default Gateway
    uip_setdraddr(ipaddr);
    uip_ipaddr(ipaddr, 255,255,255,0);       //Network Mask
    uip_setnetmask(ipaddr);
    httpd_init();
    iperf_init();
    . . . . . .

}
```

DM9051 transmit data：

```
uint32_t DM9051_tx(void)
{
    uint16_t calc_MWR;
    uint16_t SendLength;
    //printf("DM9051_tx()....\n");
    SendLength = uip_len;

    /* disable DM9051a interrupt           */
     while (DM9051_Read_Reg(DM9051_TCR) & DM9051_TCR_SET)
    {
        _DM9051_Delay(5);
    }
  . . . . . .
}
```

DM9051 receive data：

```
uint16_t DM9051_rx(void)
{
    . . . . . .

    rxbyte = DM9051_Read_Reg(DM9051_MRCMDX);
    rxbyte = DM9051_Read_Reg(DM9051_MRCMDX);

    //printf("DM9051_rx process...1st byte = 0x%x\n",rxbyte);
    if ((rxbyte != 1) && (rxbyte != 0))
    {
        printf("rxbyte = %02X.\n", rxbyte);
        DM9051_TRACE2("DM9051 rx: rx error, stop device \n");

        /* Reset RX FIFO pointer */
        DM9051_Write_Reg(DM9051_RCR, RCR_DEFAULT);  //RX disable
        DM9051_Write_Reg(DM9051_MPCR, 0x01);        //Reset RX FIFO pointer
        _DM9051_Delay(2);
        DM9051_Write_Reg(DM9051_RCR, (RCR_DEFAULT | RCR_RXEN));     //RX Enable
        /* restore receive interrupt                */
        //DM9051_device.imr_all |= IMR_PRM;
        //DM9051_device.imr_all = IMR_DEFAULT;
        //DM9051_Write_Reg(DM9051_IMR, DM9051_device.imr_all);
```

```
            return 0;
        }  . . . . . .
    }
```

Data process:

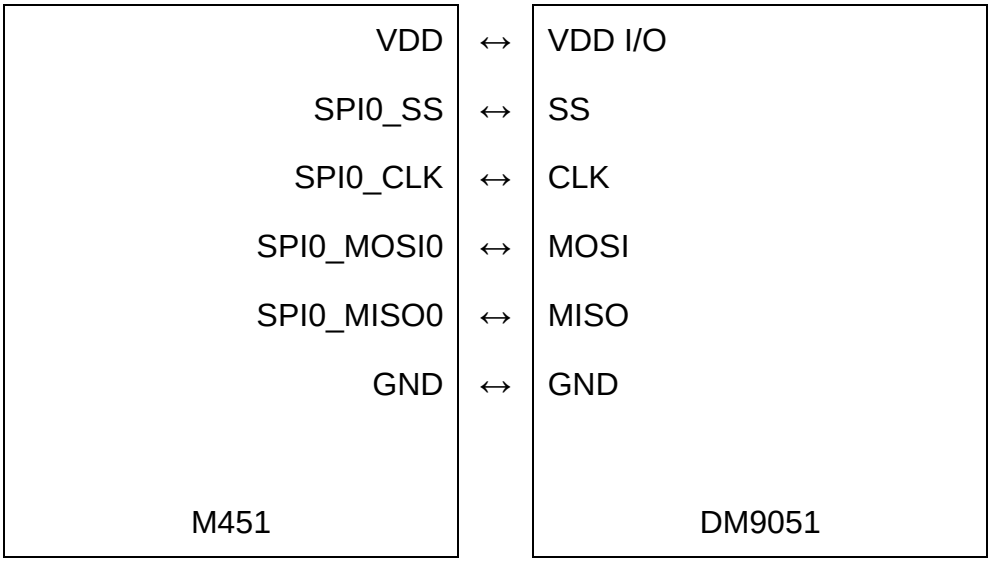
```
    While(1){
     uip_len = tapdev_read();
    if (uip_len > 0)
    {
       //printf("Rx length = %d.\r\n", uip_len);
        if (BUF->type == htons(UIP_ETHTYPE_IP))
        {
               //printf("UIP_ETHTYPE_IP..\r\n");
               uip_input();        // uip_process(UIP_DATA)
         . . . . . .
    }
    else if (timer_expired(&periodic_timer))
    {
         timer_reset(&periodic_timer);

      . . . . . .
    }
```

3 Software and Hardware Environment

- **Software Environment**
  - BSP Version
    - M451BSP_v3.01.003
  - IDE Version
    - Keil uVersion4.74
- **Hardware Environment**
  - Circuit Components
    - NuTiny-SDK-M451V
    - DM9051
  - Diagram

M451driver DM9051 though SPI interface.



4 Directory Information

- EC_M451_SPI_DM9051_uIP _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 and double click “M451_SPI_DM9051_uIP.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
Apr. 24, 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.