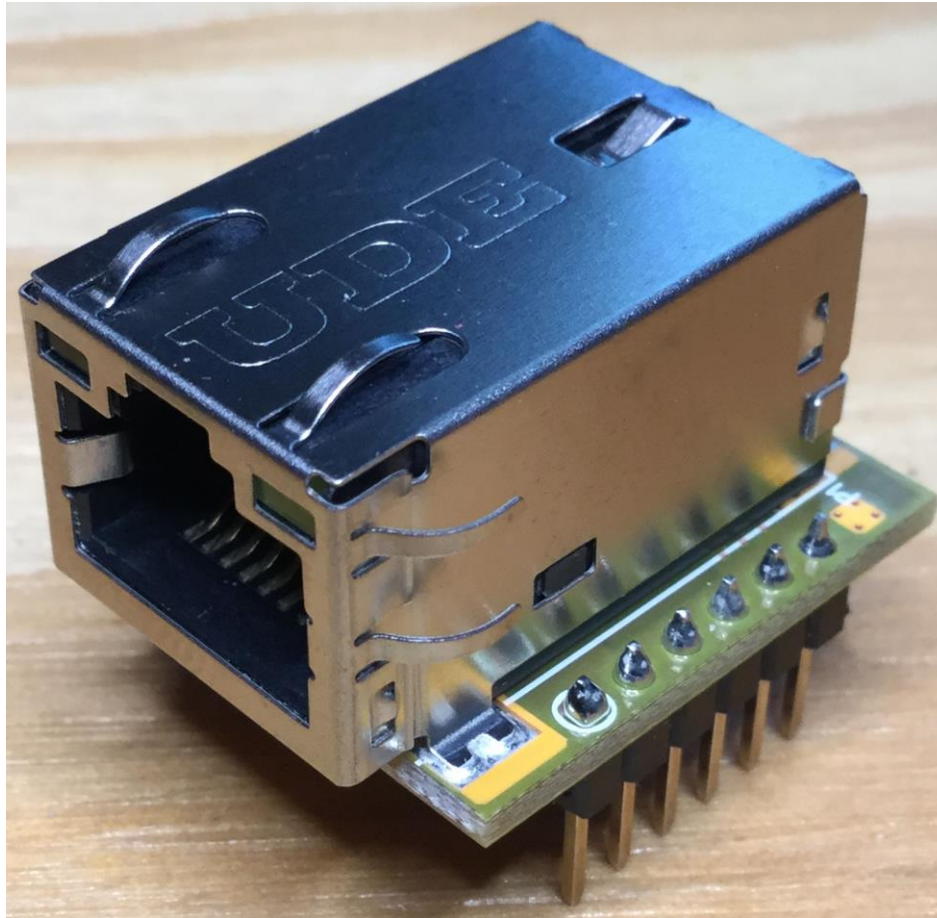


Easy TCP/IP for IoT



TCP/IP in a Can

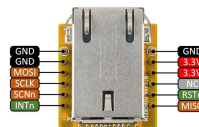
October 23, 2019

Fred Eady

Easy TCP/IP for IoT

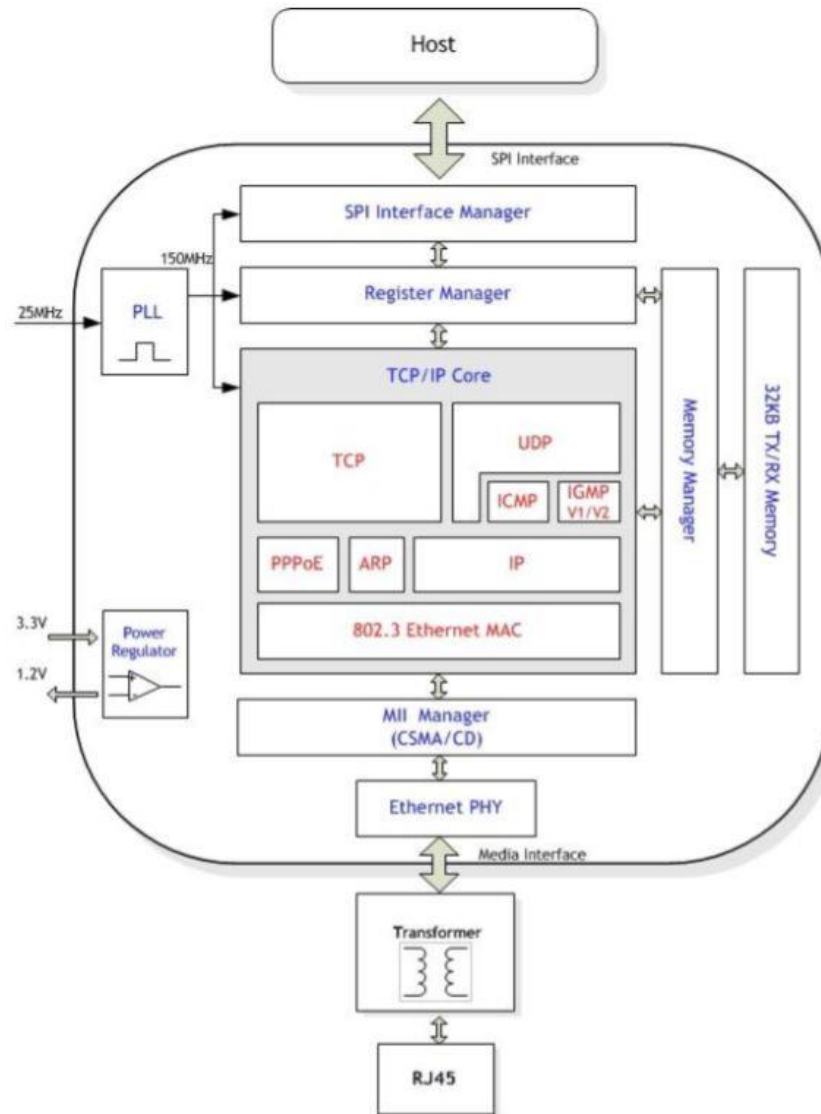
AGENDA

- Hardware – The WIZ850io
- Firmware – Integrating the WIZnet ioLibrary
- Day 3 Summary



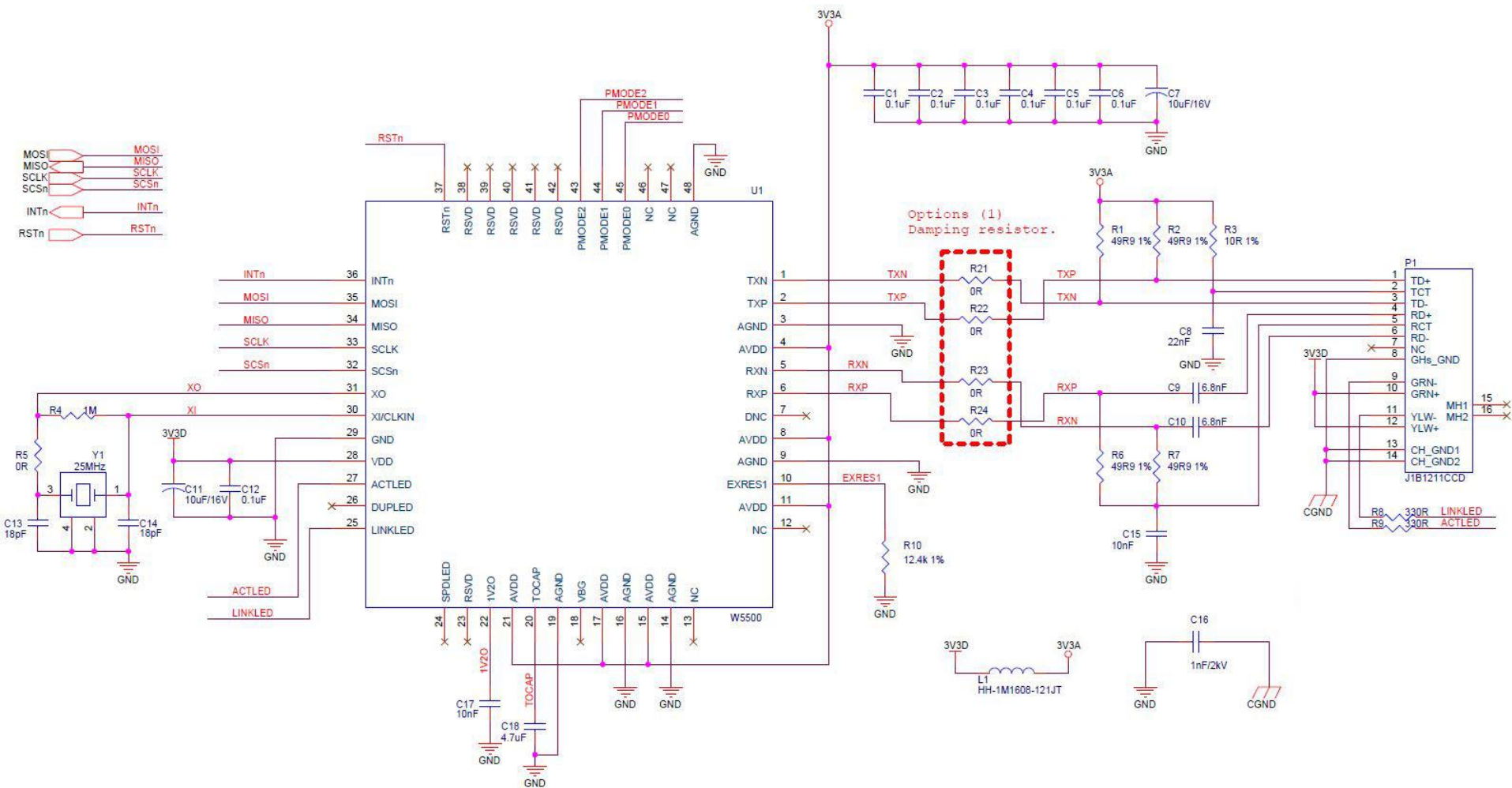
Easy TCP/IP for IoT

Hardware - The WIZ850io: W5500 Block Diagram

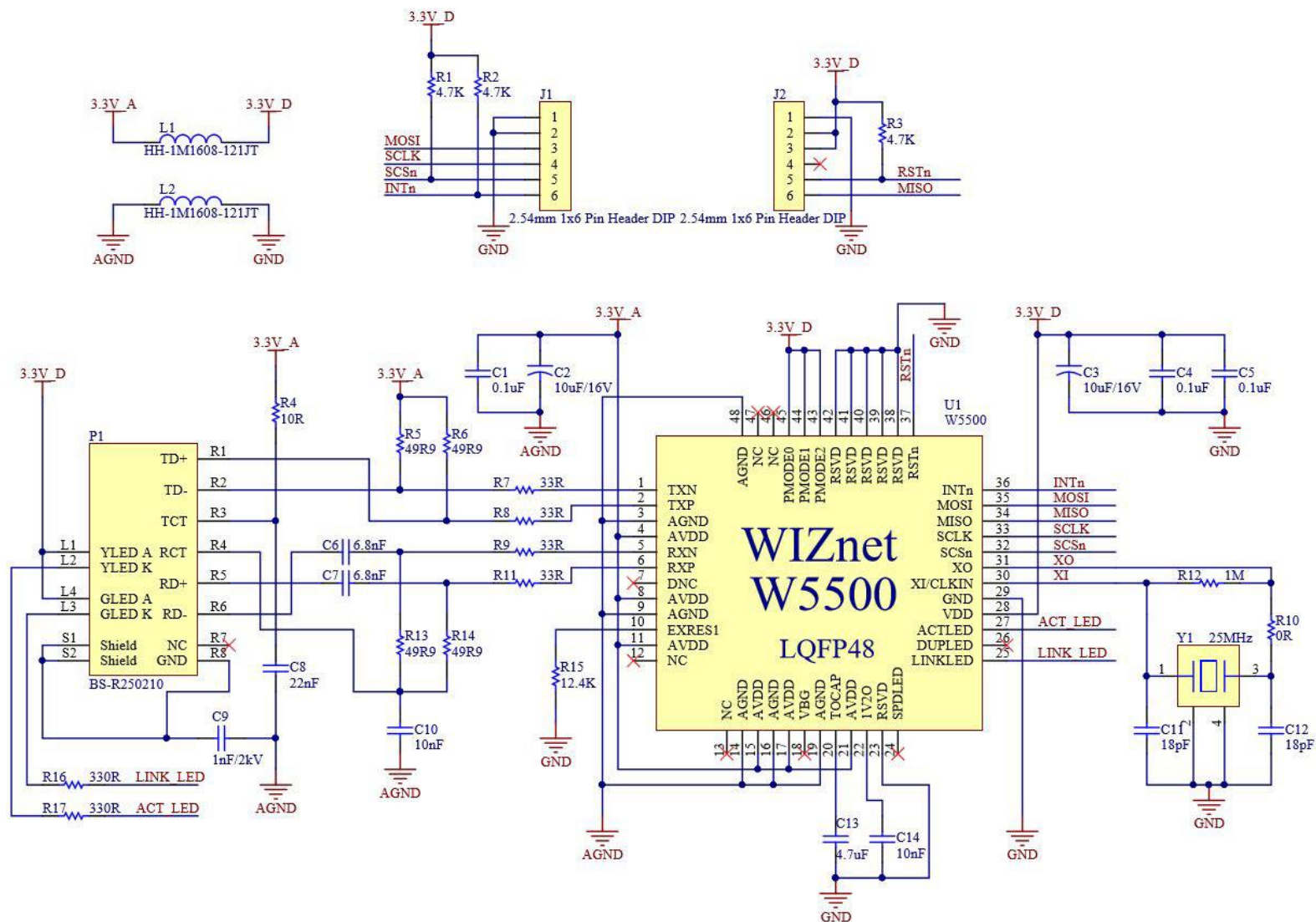


Easy TCP/IP for IoT

Hardware - The WIZ850io: W5500 Reference Design



Hardware - The WIZ850io: Module Schematic

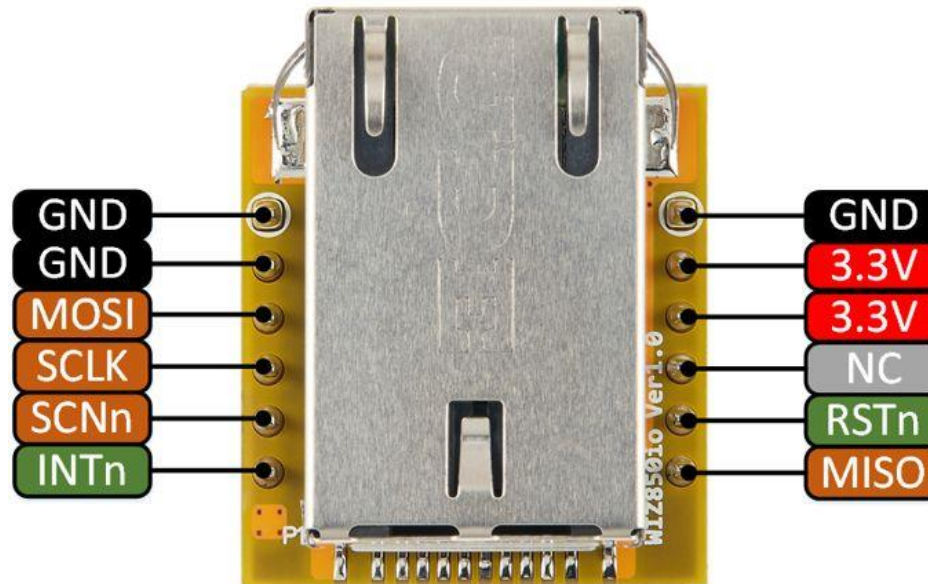
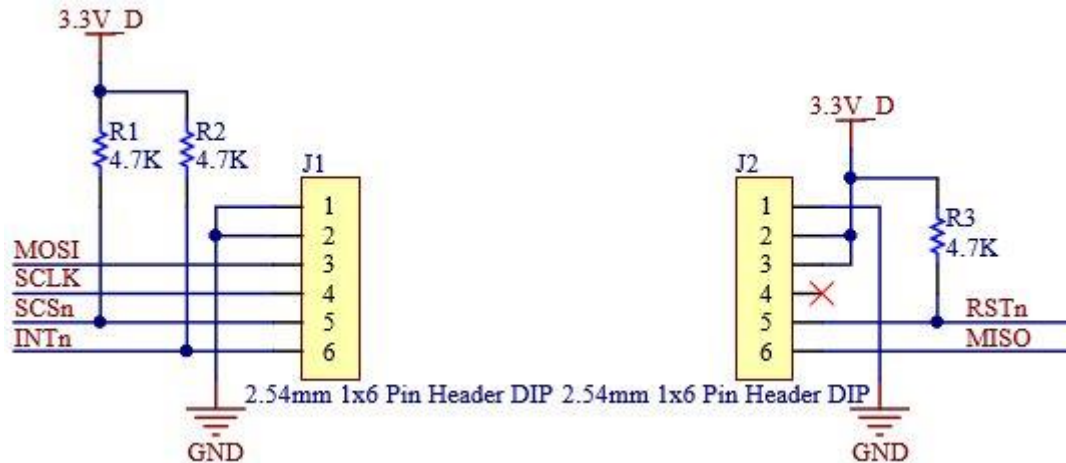


Presented by:



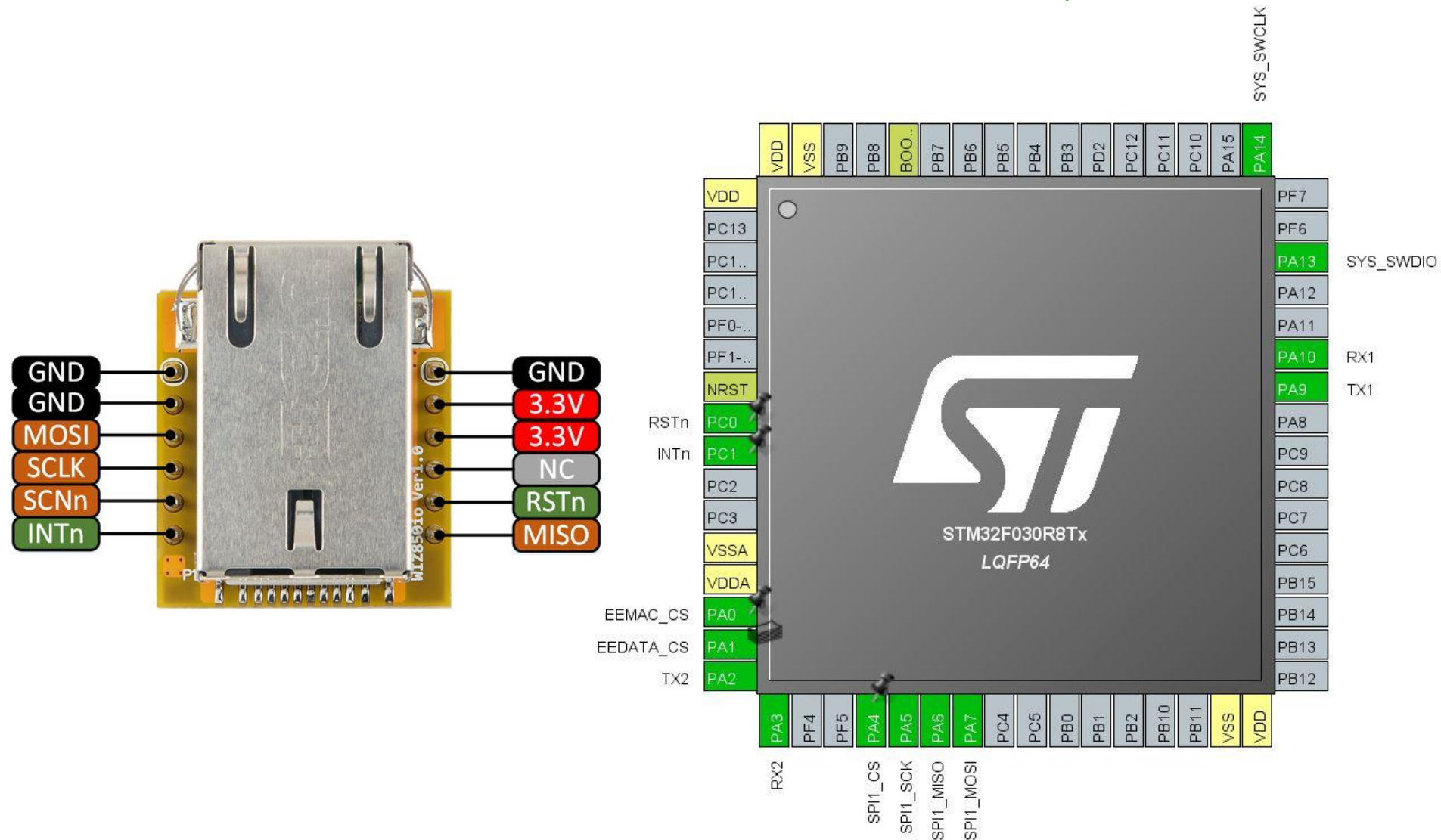
Easy TCP/IP for IoT

Hardware - The WIZ850io: Module Pinout



Easy TCP/IP for IoT

Hardware - The WIZ850io: Module ARM Interface

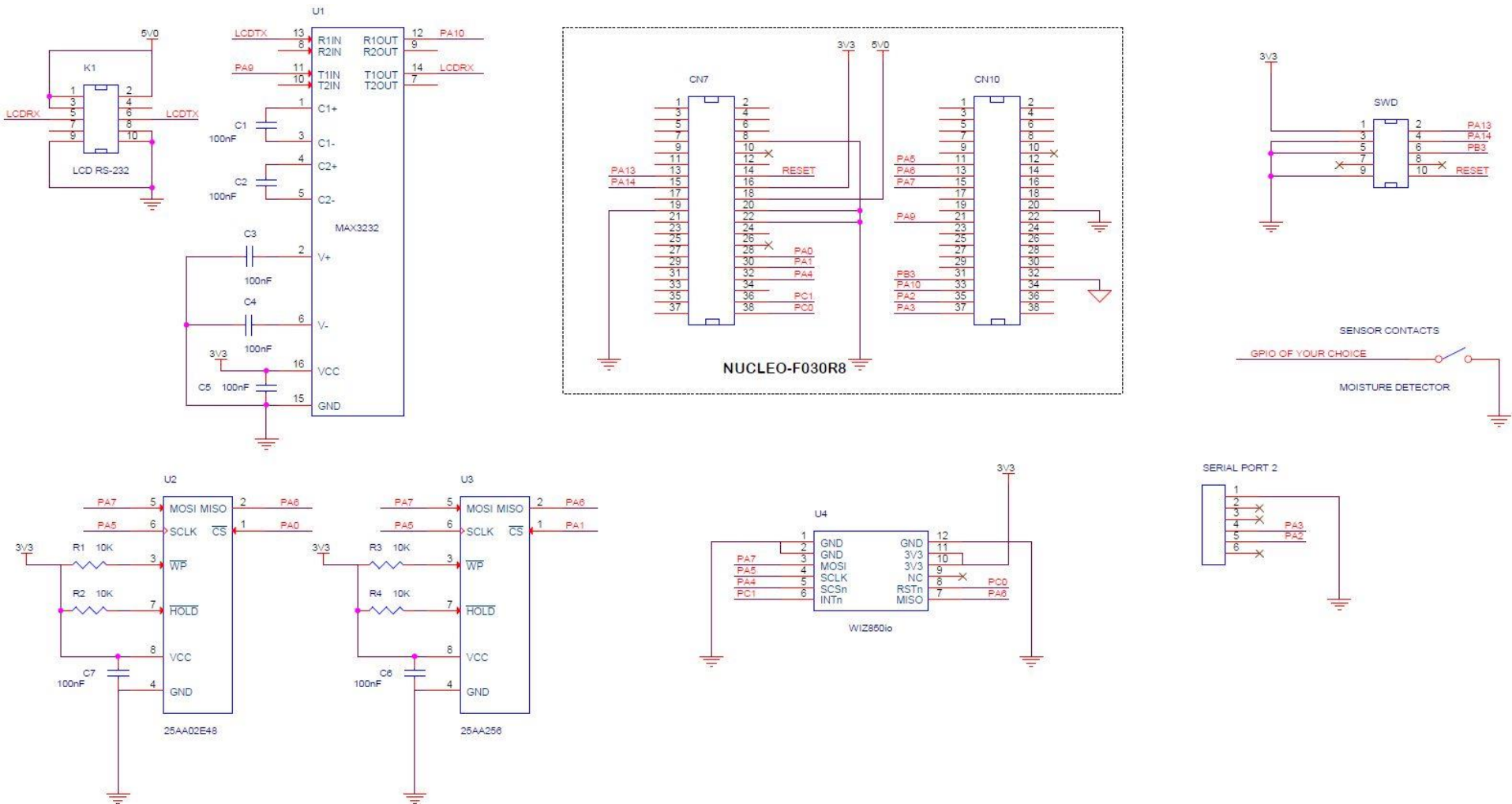


Presented by:



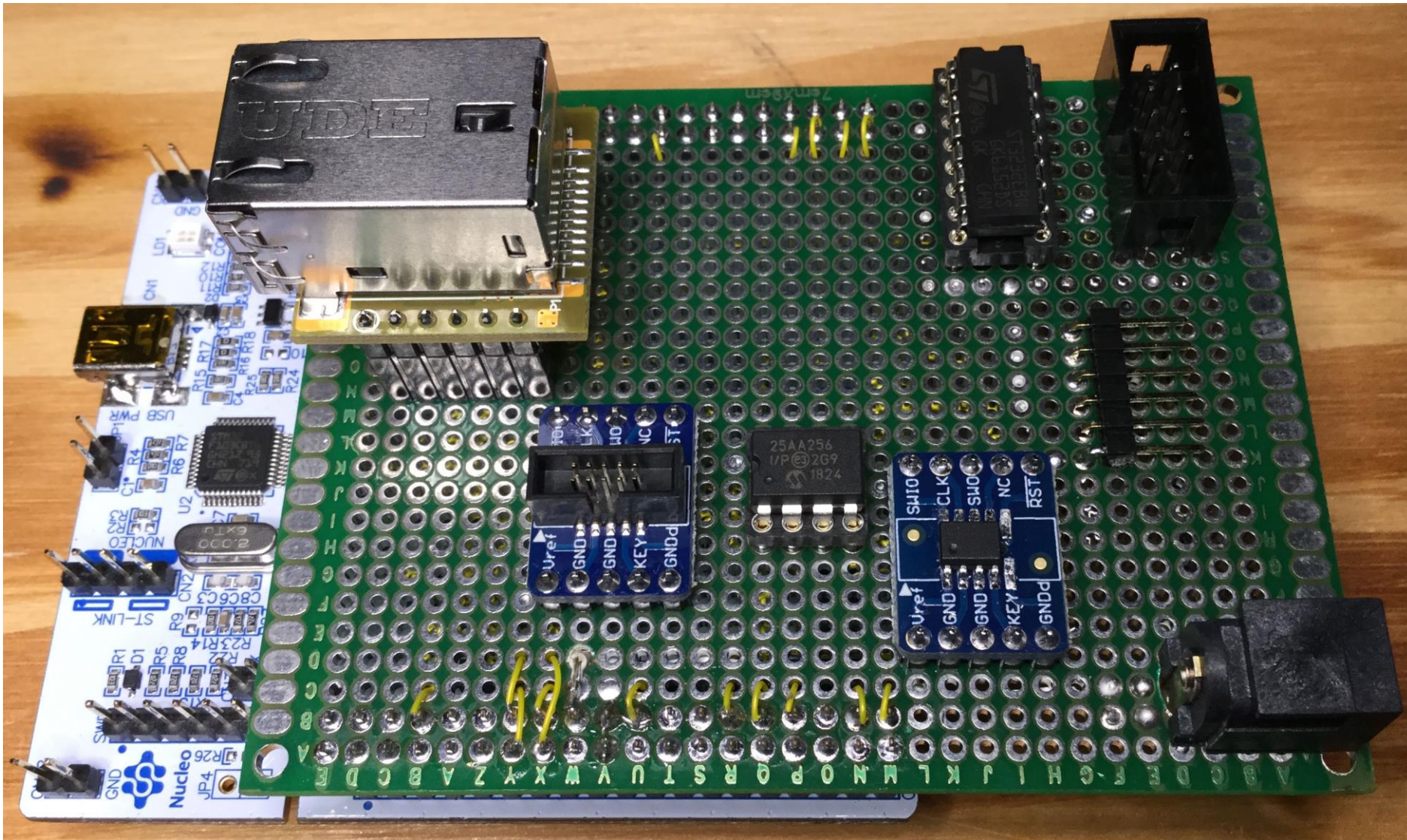
Easy TCP/IP for IoT

Hardware - The WIZ850io: ARM System



Easy TCP/IP for IoT

Hardware - The WIZ850io: ARM System

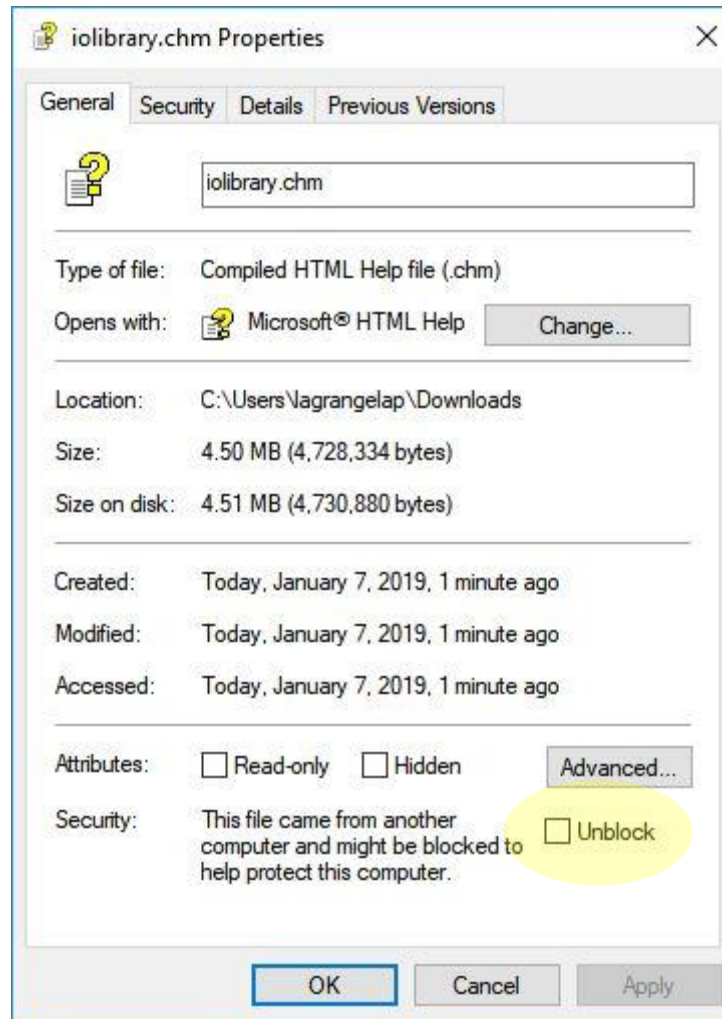


Presented by:



Easy TCP/IP for IoT

Firmware – Integrating the WIZnet ioLibrary: Unblock ioLibrary.chm



Easy TCP/IP for IoT

Firmware – Integrating the WIZnet ioLibrary: Specify W5500 and I/O Mode

wizchip_conf.h

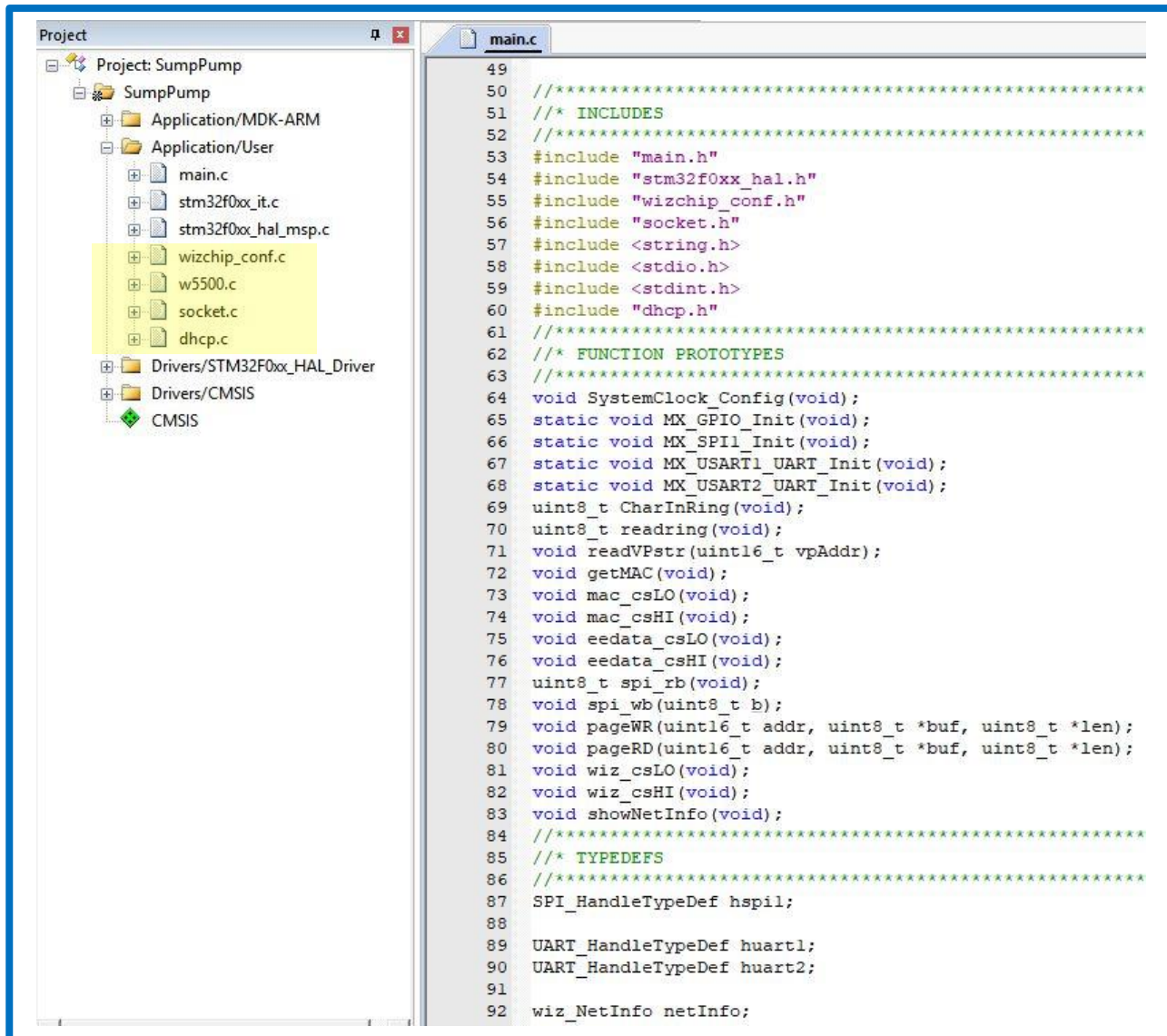
```
62  /**
63   * @brief Select WIZCHIP.
64   * @todo You should select one, \b W5100, \b W5100S, \b W5200, \b W5300, \b W5500 or etc.
65   *       ex> <code> #define _WIZCHIP_      W5500 </code>
66   */
67
68  #define W5100      5100
69  #define W5100S    5100+5
70  #define W5200      5200
71  #define W5300      5300
72  #define W5500      5500
73
74  #ifndef _WIZCHIP_
75  #define _WIZCHIP_      W5500 // W5100, W5100S, W5200, W5300, W5500
76  #endif
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151  #ifndef _WIZCHIP_IO_MODE_
152  // #define _WIZCHIP_IO_MODE_      _WIZCHIP_IO_MODE_SPI_FDM_
153  #define _WIZCHIP_IO_MODE_      _WIZCHIP_IO_MODE_SPI_VDM_
154  #endif
```



Presented by:

Easy TCP/IP for IoT

Firmware – Integrating the WIZnet ioLibrary: ioLibrary Files



Presented by:

Easy TCP/IP for IoT

Firmware - Integrating the WIZnet ioLibrary: SPI Integration

```
283 //*****
284 /* GET MAC ADDRESS
285 //*****
286 void getMAC(void)
287 {
288     mac_csLO();
289     spi_wb(ee_readCmd);
290     spi_wb(mac_eeAddr);
291     for(i=0;i<6;i++)
292     {
293         macAddr[i] = spi_rb();
294     }
295     mac_csHI();
296 }

782 reg_wizchip_cs_cbfunc(wiz_csLO, wiz_csHI);
783 reg_wizchip_spi_cbfunc(spi_rb, spi_wb);
784 //uint8_t bufSize[] = {2, 2, 2, 2};
785 wizchip_init(bufSize, bufSize);
786 getMAC();
787 for(eeIndx=0;eeIndx<6;eeIndx++)
788 {
789     netInfo.mac[eeIndx] = macAddr[eeIndx];
790 }
791 wizchip_setnetinfo(&netInfo);
```

Watch 1		
Name	Value	Type
netInfo	0x20000210 &netInfo	struct wiz_NetInfo_t
mac	0x20000210 &netInfo[]...	unsigned char[6]
[0]	0x00	unsigned char
[1]	0x04	unsigned char
[2]	0xA3 'E'	unsigned char
[3]	0x06	unsigned char
[4]	0xE7 'ç'	unsigned char
[5]	0x4B 'K'	unsigned char
ip	0x20000216 ""	unsigned char[4]
sn	0x2000021A ""	unsigned char[4]
gw	0x2000021E ""	unsigned char[4]
dns	0x20000222 ""	unsigned char[4]
dhcp	0x00	enum (uchar)

Easy TCP/IP for IoT

Firmware - Integrating the WIZnet ioLibrary: DHCP

```

793  scratch8 = 0;
794  DHCP_init(0, dhcpBuf);
795  printf("DHCP Init\r\n");
796  do{
797    switch(DHCP_run())
798    {
799      case DHCP_FAILED:
800        printf("DHCP FAILED\r\n");
801        break;
802      case DHCP_RUNNING:
803        //printf("DHCP RUNNING\r\n");
804        break;
805      case DHCP_IP_ASSIGN:
806        printf("DHCP IP ASSIGN\r\n");
807        break;
808      case DHCP_IP_CHANGED:
809        printf("DHCP IP CHANGED\r\n");
810        break;
811      case DHCP_IP_LEASED:
812        if(scratch8 == 0)
813        {
814          printf("DHCP IP LEASED\r\n");
815          wizchip_getnetinfo(&netInfo);
816          showNetInfo();
817          ++scratch8;
818        }
819        break;
820      case DHCP_STOPPED:
821        printf("DHCP STOPPED\r\n");
822        break;
823    }
824  }while(1);

```

Watch 1		
Name	Value	Type
netInfo	0x20000210 &netInfo	struct wiz_NetInfo_t
mac	0x20000210 &netInfo[...]	unsigned char[6]
[0]	0x00	unsigned char
[1]	0x04	unsigned char
[2]	0xA3 'E'	unsigned char
[3]	0x06	unsigned char
[4]	0xE7 'ç'	unsigned char
[5]	0x4B 'K'	unsigned char
ip	0x20000216 "Ä-□à"	unsigned char[4]
[0]	0xC0 'À'	unsigned char
[1]	0xA8 '¬'	unsigned char
[2]	0x01	unsigned char
[3]	0xE0 'à'	unsigned char
sn	0x2000021A "yyy"	unsigned char[4]
[0]	0xFF 'ÿ'	unsigned char
[1]	0xFF 'ÿ'	unsigned char
[2]	0xFF 'ÿ'	unsigned char
[3]	0x00	unsigned char
gw	0x2000021E "Ä-□□"	unsigned char[4]
[0]	0xC0 'À'	unsigned char
[1]	0xA8 '¬'	unsigned char
[2]	0x01	unsigned char
[3]	0x01	unsigned char
dns	0x20000222 ""	unsigned char[4]
dhcp	0x00	enum (uchar)

Presented by:

Easy TCP/IP for IoT

Firmware - Integrating the WIZnet ioLibrary: DHCP

Watch 1		
Name	Value	Type
netInfo	0x20000210 &netInfo	struct wiz_NetInfo_t
mac	0x20000210 &netInfo[0]	unsigned char[6]
[0]	0x00	unsigned char
[1]	0x04	unsigned char
[2]	0xA3 'E'	unsigned char
[3]	0x06	unsigned char
[4]	0xE7 'ç'	unsigned char
[5]	0x4B 'K'	unsigned char
ip	0x20000216 "À []"	unsigned char[4]

```
272 //*****
273 /* SHOW NETWORK CONFIGURATION
274 //*****
275 void showNetInfo(void)
276 {
277     printf("Network configuration:\r\n");
278     printf(" IP ADDRESS:  %d.%d.%d.%d\r\n", netInfo.ip[0], netInfo.ip[1], netInfo.ip[2], netInfo.ip[3]);
279     printf(" MAC ADDRESS: 0x%02X:0x%02X:0x%02X:0x%02X:0x%02X:0x%02X\r\n", netInfo.mac[0], netInfo.mac[1], netInfo.mac[2], netInfo.mac[3], netInfo.mac[4], netInfo.mac[5]);
280     printf(" NETMASK:    %d.%d.%d.%d\r\n", netInfo.sn[0], netInfo.sn[1], netInfo.sn[2], netInfo.sn[3]);
281     printf(" GATEWAY:    %d.%d.%d.%d\r\n", netInfo.gw[0], netInfo.gw[1], netInfo.gw[2], netInfo.gw[3]);
282 }
```

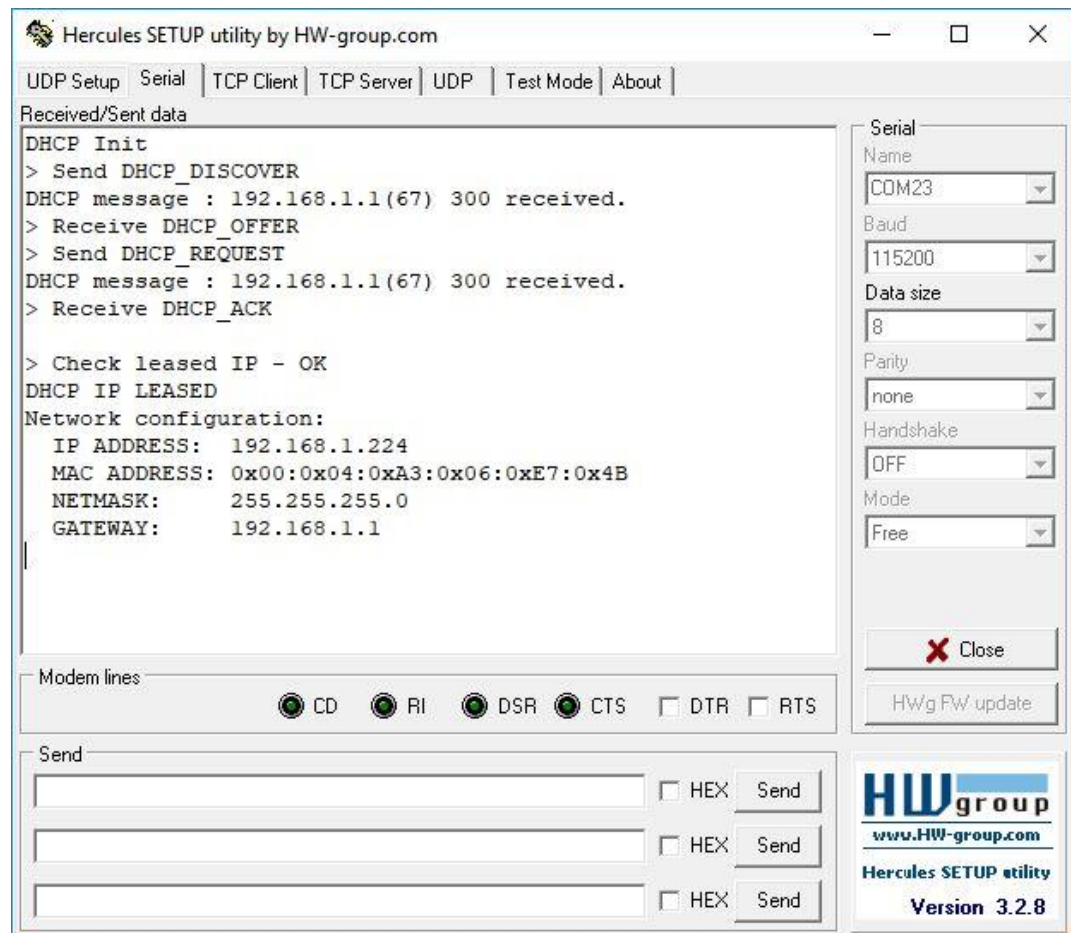
[1]	0xFF 'ÿ'	unsigned char
[2]	0xFF 'ÿ'	unsigned char
[3]	0x00	unsigned char
gw	0x2000021E "À []"	unsigned char[4]
[0]	0xC0 'À'	unsigned char
[1]	0xA8 ''	unsigned char
[2]	0x01	unsigned char
[3]	0x01	unsigned char
dns	0x20000222 ""	unsigned char[4]
dhcp	0x00	enum (uchar)

Presented by:

Easy TCP/IP for IoT

Firmware - Integrating the WIZnet ioLibrary: DHCP

```
793 scratch8 = 0;
794 DHCP_init(0,dhcpBuf);
795 printf("DHCP Init\r\n");
796 do{
797     switch(DHCP_run())
798     {
799         case DHCP_FAILED:
800             printf("DHCP FAILED\r\n");
801             break;
802         case DHCP_RUNNING:
803             //printf("DHCP RUNNING\r\n");
804             break;
805         case DHCP_IP_ASSIGN:
806             printf("DHCP IP ASSIGN\r\n");
807             break;
808         case DHCP_IP_CHANGED:
809             printf("DHCP IP CHANGED\r\n");
810             break;
811         case DHCP_IP_LEASED:
812             if(scratch8 == 0)
813             {
814                 printf("DHCP IP LEASED\r\n");
815                 wizchip_getnetinfo(&netInfo);
816                 showNetInfo();
817                 ++scratch8;
818             }
819             break;
820         case DHCP_STOPPED:
821             printf("DHCP STOPPED\r\n");
822             break;
823     }
824 }while(1);
```



Easy TCP/IP for IoT

Day 3 Summary

Hercules SETUP utility by HW-group.com

UDP Setup | Serial | TCP Client | TCP Server | UDP | Test Mode | About

Received/Sent data

```
DHCP Init
> Send DHCP_DISCOVER
DHCP message : 192.168.1.1(67) 300 received.
> Receive DHCP_OFFER
> Send DHCP_REQUEST
DHCP message : 192.168.1.1(67) 300 received.
> Receive DHCP_ACK

> Check leased IP - OK
DHCP IP LEASED
Network configuration:
IP ADDRESS: 192.168.1.224
MAC ADDRESS: 0x00:0x04:0xA3:0x06:0xE7:0x4B
NETMASK: 255.255.255.0
GATEWAY: 192.168.1.1
```

Modem lines: ☒ CD ☒ RI ☒ DSR ☒ CTS ☐ DTR ☐

Send

HEX Send

HEX Send

HEX Send

HWgroup
www.HW-group.com
Hercules SETUP utility
Version 3.2.8

Watch 1		
Name	Value	Type
netInfo	0x20000210 &netInfo	struct wiz_NetInfo_t
mac	0x20000210 &netInfo[]...	unsigned char[6]
[0]	0x00	unsigned char
[1]	0x04	unsigned char
[2]	0xA3 'E'	unsigned char
[3]	0x06	unsigned char
[4]	0xE7 'ç'	unsigned char
[5]	0x4B 'K'	unsigned char
ip	0x20000216 ""	unsigned char[4]
sn	0x2000021A ""	unsigned char[4]
gw	0x2000021E ""	unsigned char[4]
dns	0x20000222 ""	unsigned char[4]
dhcp	0x00	enum (uchar)

Presented by: