Using Ethernet interface of Sampo

20.9.2012 ver 1.1

Contents

Overview

Nordic ID Sampo S1 is provided with the USB and Ethernet interface. The most convenient way to configure the Ethernet settings is to use Nordic ID RFID Configurator software utility. However, it is possible to read and write the Ethernet configuration of the device by using NurApi & NurApiDotNet dll libraries.

This document describes how to use the Ethernet configuration features from the NurApiDotNet.dll
The language of the code samples described in this document is C# .NET

1.1 Boot sequence.

The device can be powered using USB, Power over Ethernet (PoE) or external power supply. After plugging the power, the device beeps once and starts the boot sequence.

1. Power up / reset.
   -All four leds are ON.

2. Addressing mode: DHCP (default)
   - Acquiring IP address from the DHCP server
   - If device cannot get IP from the DHCP server or it's not available, Nordic ID Sampo enters to Auto IP state where IP will be 169.254.18.0. This may take 30 seconds after power up.

3. Addressing mode: STATIC IP
   - Nordic ID Sampo set Static IP address as configured.
   - Nordic ID Sampo sends "Gratuitous ARP" message to network allowing other network machines to update their ARP tables.

4. Device ready to use
   - Only the power led is ON.
   - For configuring and searching Nordic ID Sampo devices from network, use RFID Configurator software utility.

Note: If the Ethernet link gets lost for some reason, the Ethernet module will restart the boot sequence when the link comes back again.

1.2 Searching devices from network

Nordic ID Sampo devices listen Broadcast messages at port 4331. RFID Configurator software utility finds devices from the network by sending broadcast message using IP 255.255.255.255.

Sending a Broadcast message for searching devices from the network using NurApi.NurApiSendBroadcast function:

```
hNur.NurApiSendBroadcast(NurApi.BC_CMD_GET_DEV_INFO,NurApi.BC_FILTER_TYPE_NONE, 0, null, 0, null, 0);
```

After sending the Broadcast command NurApi.BC_CMD_GET_DEV_INFO with NurApi.BC_FILTER_TYPE_NONE, all Nordic ID Sampo devices in the network will answer with the device information.

Receiving device info data:

```
//Definition of event handler for receiving Broadcast answer from Sampo device
hNur.DevInfoEvent += new EventHandler<NurApi.DevInfoEventArgs>(hNur_DevInfoEvent);

//Event handler for receiving DevInfoData (one event per device)
void hNur_DevInfoEvent(object sender, NurApi.DevInfoEventArgs e)
{
        NurApi.DevInfoData devData = new NurApi.DevInfoData();

        //Copy and save it to list for future usage
        devData = e.data;
        devList.Add(devData);
}
```

NurApi.DevInfoEventArgs contains DevInfoData

Members of NurApi.DevInfoData structure:

Name	Description
altSerial	NUR altSerial number
boardID	Board ID (NOT USED)
boardType	Board type (NOT USED)
eth	Ethernet settings of device
hwVer	Nur HW version
IOstate	Current I/O state (NOT USED)
lightADC	Current ADC value of light sensor. Max 1023
nurAntNum	NUR Number of antennas
nurName	NUR Name
nurVer	NUR SW version (Major, Build, DevBuild)
serial	NUR Serial number
status	Current device status 0=Disconnected 1=Connected
tapADC	Current ADC value of tap sensor. Max 1023
temp	Internal temperature of Sampo Ethernet CPU (Celsius)
upTime	Up time in days since last reset

Members of NurApi.EthConfig struct

Name	Description
addrType	(Get/Set) Address type 0=DHCP(default) 1=STATIC
gw	(Get/Set) Gateway (used in static IP mode)
hostip	(Get/Set) Client host IP (if Mode=Client)
hostmode	(Get/Set) hostmode 0=Server (default) 1=Client. If server, Sampo listens port 'serverPort' for client connections (default). If Client, Sampo automatically tries to initialize connection to 'hostip' : 'hostPort' server.
hostPort	(Get/Set) Client host port (if Mode=Client)
ip	(Get) Current IP address of Sampo device
mac	(Get/(Set only if using transport as broadcast)) MAC address of device. Used for addressing when sending new settings via Broadcast.
mask	(Get/Set) Subnet mask (used in static IP mode)
reserved	(Get/Set) reserved for future usage
serverPort	(Get/Set) Server port where client connects to when hostmode=Server. default 1300.
staticip	(Get/Set) Static IP. When using static IP address, set addrType to 1=STATIC
title	(Get/Set) Title of Sampo device: Default: "Nordic ID Sampo S1"
transport	Transport method used with NurApiSetEthConfig(). if transport = 0 then ethernet settings will be set to Sampo via current connection.(USB,serial or TCP) if transport = 1 then ethernet settings will be set to Sampo using broadcast message. This method requires that correct 'mac' address has been set to structure. MAC address is used for filtering that only certain device receives new configurations. */
version	(Get) Current Revision of ethernet CPU firmware

Note: The Company Network policy may block Broadcast messages getting through the routers. Sometimes the routers need to be configured to allow broadcast messages. Nordic ID Sampo devices listen to broadcast messages at port 4331.

If broadcasting is not allowed in network, the alternative way to find out the IP address of Nordic ID Sampo is by using ARP table of the PC connected to network. The format of Nordic ID Sampo MAC address id: 00-21-ad-xx-xx-xx

1. Open the Windows command prompt (cmd.exe)
2. Type arp –s
3. Find your device IP from the list by comparing the MAC address

1.3 Using Sampo in Server mode

When Nordic ID Sampo is connected to the host, the “Communication” led will be ON.
By default, Nordic ID Sampo is in TCP/IP Server mode and listens to port 4333.

Example: Connecting to Nordic ID Sampo with IP 172.16.33.1 and port 4333 using NurApi.ConnectSocket function.

```
//Define event handler for Connected and Disconnected events
hNur.ConnectedEvent += new EventHandler<NurApi.NurEventArgs>(hNur_ConnectedEvent);
hNur.DisconnectedEvent += new EventHandler<NurApi.NurEventArgs>(hNur_DisconnectedEvent);


try
{
  // Connecting to Sampo
  hNur.ConnectSocket(“172.16.33.1”, 4333);

 }
 catch (Exception ex)
 {
    toolStripStatusLabel1.Text = "Cannot connect:" + ex.ToString();
    return;
 }

void hNur_ConnectedEvent(object sender, NurApi.NurEventArgs e)
{
    // Connected event handler
}

void hNur_DisconnectedEvent(object sender, NurApi.NurEventArgs e)
{
    // Disconnected event handler
}
```

1.4 Using Nordic ID Sampo in Client mode

When Nordic ID Sampo is configured as a Client, Nordic ID Sampo continuously (between 3 sec) tries to establish connection to specified IP and port. The host application started server for listening incoming connection at specified port.

When Nordic ID Sampo connects to the host, NurApi.ClientConnectedEvent fires and offers a NurApi handle for the just connected device. The host application needs to take care of all received client handles.
When NurApi.ClientDisconnectedEvent event fires, the host application must destroy client handle as it cannot be used anymore.

NurApi.StopServer() function will stop the server but does not disconnect already connected clients.
For disconnecting all connected clients, iterate client list and call NurApi.Disconnect() for each NurApi instance.

```
// Define “parent NUR” instance and Start server for listening incoming connections.
NurApi pNur = null;
pNur = new NurApi(this); //Create Nur Api handle

// Add event handler for receiving client connected and disconnected events
pNur.ClientConnectedEvent += new EventHandler<NurApi.ClientInfoEventArgs>(pNur_ClientConnected);
pNur.ClientDisconnectedEvent += new EventHandler<NurApi.ClientInfoEventArgs>(pNur_ClientDisonnected);

try
{
  pNur.StartServer(2000,5); //Start listening port 2000 and allowing max 5 Sampo to connect
  AddLog("Listening port:" + textPortEdit.Text);
}
catch (NurApiException ex)
{
  AddLog("StartServer Error:" + ex.Message);
}

// Client Connected. Add new Client to list.
void pNur_ClientConnected(object sender, NurApi.ClientInfoEventArgs e)
{
    //ClientInfoEventArgs passes NurApi handle of just connected device
    //Save it to the some kind of client list and use it for interfacing Sampo
    AddToClientList(e.cApi);
}

// Client Disconnected. At this point client is not useful to us anymore.
// We need to remove it from our ClientList and call NurApi.Dispose function for freeing resources.
void pNur_ClientDisonnected(object sender, NurApi.ClientInfoEventArgs e)
{
    //Stop all activities for this Sampo.
    RemoveFromClientList(e.cApi); //Remove it from list of clients
    e.cApi.Dispose(); //No use anymore. Free resources and newer use this handle anymore
}

//Stopping Server. Note! client connections will remain.
pNur.StopServer();

//See MultiClientTest sample code. C# VisualStudio 2010 project.
```

1.5 Reading Ethernet Configuration

The Ethernet configuration can be read from the device using Broadcast method or using current connection (USB, Serial, TCP/IP)

```
//Read ethconfig from Sampo device using current connection

NurApi.EthConfig eth;

try
{
       eth = hNur.GetEthConfig();
}
catch (Exception ex)
{
       //error reading EthConfig
}
```

Reading the Ethernet configuration using Broadcast, refer to section 1.2 “Searching devices from network”

1.6 Writing the Ethernet Configuration

The new Ethernet configuration can be sent to Nordic ID Sampo using NurApi.SetEthConfig function. After sending the new configuration, the Ethernet module of Nordic ID Sampo will reboot automatically and starts using the new configuration.

There are two alternative methods to send the new Ethernet configuration to Nordic ID Sampo:

1. Using current connection. (USB,Serial or TCP/IP). (NurApi.EthConfig.transport = 0)
2. Using Broadcast message (NurApi.EthConfig.transport = 1 and NurApi.EthConfig.mac of target device specified). This method is useful when Nordic ID Sampo is connected to the host as a TCP/IP client as there is no other way to create connection.

Set the configuration using current connection:

```
NurApi.EthConfig ec = new NurApi.EthConfig();

ReadEthSettings(ref ec); //Fill EthConfig struct with appropriate configuration.

ec.transport = 0;//Use current connection. (USB,Serial,TCP/IP)

try
{
	hNur.SetEthConfig(ref ec);
	//At this point our current connection is lost.
	//Sampo will reset in order to take new Ethernet settings in use.
	//User need to establish connection again
}
catch (Exception ex)
{
	//Error writing Ethernet configuration.
}
```

Set the configuration using Broadcast

```
NurApi.EthConfig ec = new NurApi.EthConfig();

ReadEthSettings(ref ec); //Fill EthConfig struct with appropriate configuration.

//Make sure ec.mac has been set for target destination

ec.transport = 1;//Use Broadcast

try
{
	hNur.SetEthConfig(ref ec);
	//At this point target device reboots.

}
catch (Exception ex)
{
	//Error writing Ethernet configuration.
}
```

1.7 Broadcast messages

Broadcast messages can be sent using NurApi.NurApiSendBroadcast method.
Filter settings defines which devices can receive Broadcast message.
Broadcast message is sent to all Sampo devices in network but only those Sampo devices receives and executes messages which met filter rules.
* If filter type has been set to NUR_BC_FILTER_TYPE_NONE then all devices recevies and executes broadcast message.

For testing how filtering works, use RFID Configurator software utility.

```
public void NurApiSendBroadcast(
        int cmd,
        int filterType,
        int filterOp,
        byte[] filterData,
        int filterSize,
        byte[] data,
        int dataLength
)
```

Nordic ID Sampo device supports following broadcast commands:

BC_CMD_BEEP	Broadcast command: Generate Beep sound
BC_CMD_GET_DEV_INFO	Broadcast command: Request for device info
BC_CMD_SET_ETHCONFIG	Broadcast command: Set Ethernet configuration

Filter types:

BC_FILTER_TYPE_ADDRTYPE	Broadcast filter type: Address type (0=server 1=client)
BC_FILTER_TYPE_IP	Broadcast filter type: IP address (x.x.x.x)
BC_FILTER_TYPE_MAC	Broadcast filter type MAC address (12 char hex string)
BC_FILTER_TYPE_MODE	Broadcast filter type: host mode (0=server 1=client)
BC_FILTER_TYPE_NAME	Broadcast filter type: Name (part of name string is enough for filtering with device name)
BC_FILTER_TYPE_NONE	Broadcast filter type (No filter)
BC_FILTER_TYPE_NURVER	Broadcast filter type: Nur version (Numeric)
BC_FILTER_TYPE_SERVERPORT	Broadcast filter type : Server port(Numeric)
BC_FILTER_TYPE_STATUS	Broadcast filter type: Status (0=disconnected 1=Connected)
BC_FILTER_TYPE_VERSION	Broadcast filter type: Version (Numeric)

Filter Operation:

BC_FILTER_OP_EQUAL	Broadcast filter operation: EQUAL
BC_FILTER_OP_HIGHER	Broadcast filter operation: HIGHER
BC_FILTER_OP_LOWER	Broadcast filter operation: LOWER

filterData [] 16-byte array of filter data. Typically string data

filterSize Length of filterData [] in bytes.

data [] Byte array of message to target device(s)

dataLength Length of data [] in bytes.

Examples:

Sending beep sound to all devices:

```
byte[] msg = new byte[2];

msg[0] = 60; //Beep length = value x 10 ms.
msg[1] = 50; //duty (10-90)

hNur.NurApiSendBroadcast(NurApi.BC_CMD_BEEP,NurApi.BC_FILTER_TYPE_NONE,0, null, 0, msg, 2);
```

Sending Beep sound to device which MAC is 00-21-ad-0a-00-07

```
string macAddr="0021ad0a0007";
byte[] msg = new byte[2];
byte[] fdata = new byte[16]; //Store filter data (mac addr)

fdata = System.Text.Encoding.ASCII.GetBytes(macAddr);

msg[0] = 60; //Beep length value x 10 ms.
msg[1] = 50; //duty (10-90)

hNur.NurApiSendBroadcast(NurApi.BC_CMD_BEEP,NurApi.BC_FILTER_TYPE_MAC, NurApi.BC_FILTER_OP_EQUAL,
fdata, macAddr.Length, msg, sizeof(msg));
```

Get DeviceInfo from devices which NUR version is lower than 2.5

```
string macAddr="0021ad0a0007";
byte[] buf = new byte[2];
byte[] fdata = new byte[1];

fdata = System.Text.Encoding.ASCII.GetBytes(macAddr);
buf[0] = 25; //Version number without dot

hNur.NurApiSendBroadcast(NurApi.BC_CMD_GET_DEV_INFO,NurApi.BC_FILTER_TYPE_NURVER,
NurApi.BC_FILTER_OP_LOWER, fdata, sizeof(fdata), null,0);
```

2 Support

For further information, contact Nordic ID support

E-mail: support@nordicid.com
Telephone: +358 2 727 7790