Microcontroller Networking

for Embedded

Measurement & Control Applications

M C N F

MicroController Networking Framework

- Getting Started -

Dr.-Ing. Andreas Eppinger

2012-12-17 Rev 1.0.0

MCNF Installation

Pre-Requisites

Before installing MCNF, please make sure you have installed the following free software on your PC:

(1) **Microsoft .NET Framework 4** (mandatory)

Microsoft Visual Studio C# 2010 (Express version will work)

http://www.microsoft.com/visualstudio/eng/products/visual-studio-2010-express

The Microsoft .NET Framework 4 and Microsoft Visual Studio C# 2010 is mandatory.

(2) **Arduino 1.0 plus the TimerOne library**

http://arduino.cc/en/Main/Software

http://www.arduino.cc/playground/code/timer1

Arduino 1.0 is the IDE supporting the Arduino embedded platforms.

Recommended for AVR based and .NET MF based embedded applications,
but not needed for the demos in this deocument:

(3) **Atmel Studio 6** (optional)

http://www.atmel.com/microsite/atmel_studio6/default.aspx

Atmel Studio 6 supports AVR based embedded solution development.

(4) **Microsoft .NET Micro Framework SDK 4.2** (optional)

http://netmf.codeplex.com/releases/view/82448

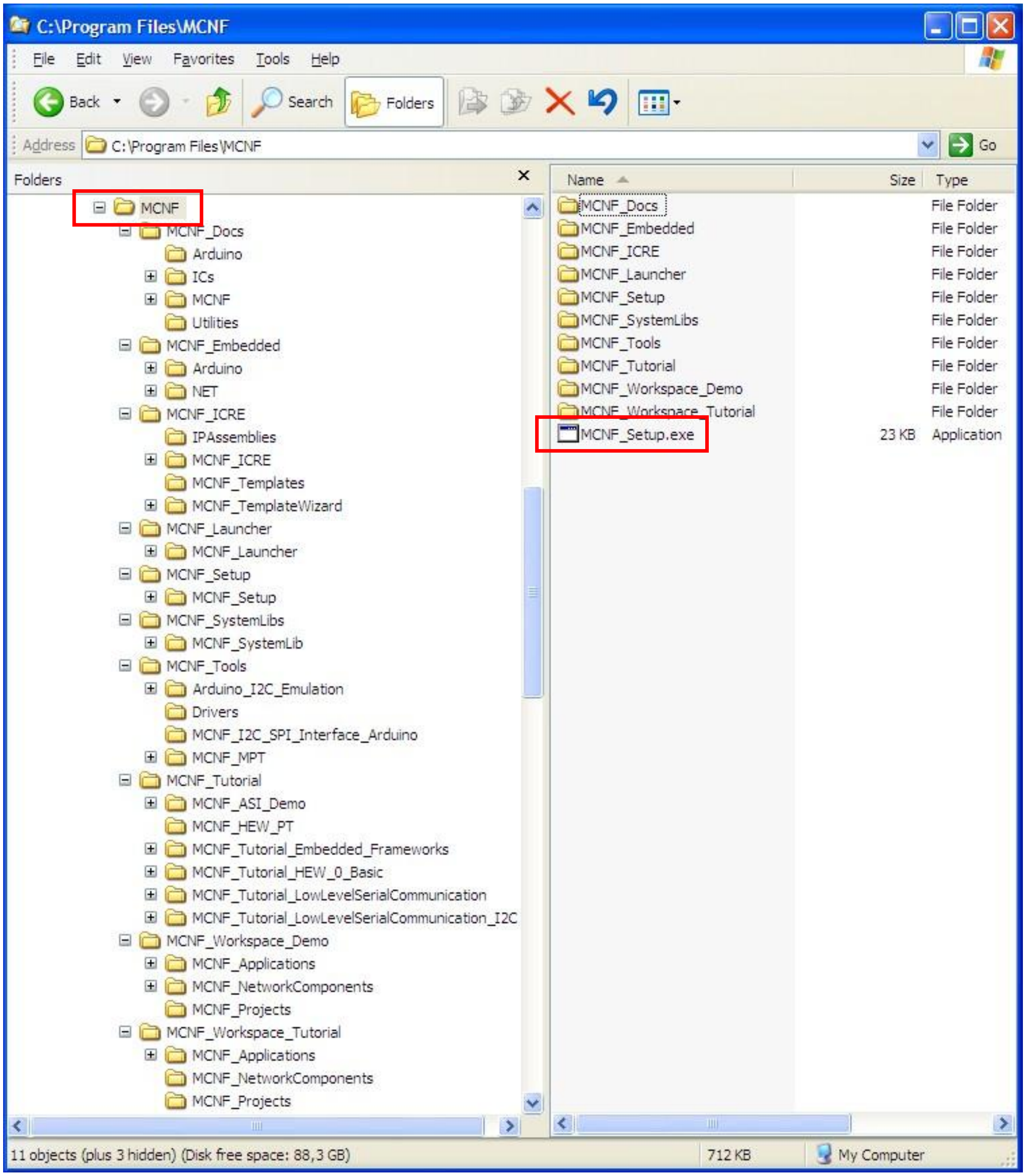
Microsoft .NET Micro Framework SDK 4.2 is needed for .NET MF based embedded solutions.

Installation of MCNF Framework on the PC

(1) Copy file MCNF.zip to a directory of your choice, e.g. “C:\Program Files”.

(2) Extract MCNF.zip with option “Extract Here”.

This will create a directory “…\MCNF”, e.g. “C:\Program Files\MCNF” with the following structure:



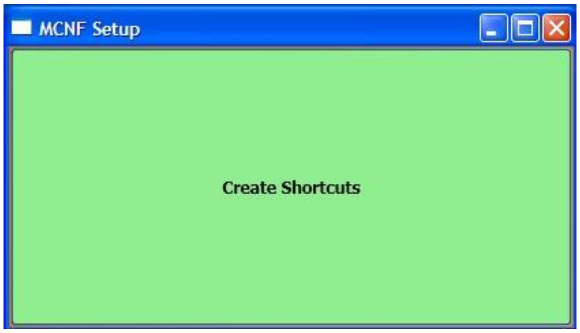
- MCNF
  - MCNF_Docs
    - Arduino
    - ICs
    - MCNF
    - Utilities
  - MCNF_Embedded
    - Arduino
    - NET
  - MCNF_ICRE
    - IPAssemblies
    - MCNF_ICRE
    - MCNF_Templates
    - MCNF_TemplateWizard
  - MCNF_Launcher
    - MCNF_Launcher
  - MCNF_Setup
    - MCNF_Setup
  - MCNF_SystemLibs
    - MCNF_SystemLib
  - MCNF_Tools
    - Arduino_I2C_Emulation
    - Drivers
    - MCNF_I2C_SPI_Interface_Arduino
    - MCNF_MPT
  - MCNF_Tutorial
    - MCNF_ASI_Demo
    - MCNF_HEW_PT
    - MCNF_Tutorial_Embedded_Frameworks
    - MCNF_Tutorial_HEW_0_Basic
    - MCNF_Tutorial_LowLevelSerialCommunication
    - MCNF_Tutorial_LowLevelSerialCommunication_I2C
  - MCNF_Workspace_Demo
    - MCNF_Applications
    - MCNF_NetworkComponents
    - MCNF_Projects
  - MCNF_Workspace_Tutorial
    - MCNF_Applications
    - MCNF_NetworkComponents
    - MCNF_Projects

Name	Size	Type
MCNF_Docs		File Folder
MCNF_Embedded		File Folder
MCNF_ICRE		File Folder
MCNF_Launcher		File Folder
MCNF_Setup		File Folder
MCNF_SystemLibs		File Folder
MCNF_Tools		File Folder
MCNF_Tutorial		File Folder
MCNF_Workspace_Demo		File Folder
MCNF_Workspace_Tutorial		File Folder
MCNF_Setup.exe	23 KB	Application

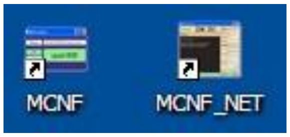
11 objects (plus 3 hidden) (Disk free space: 88,3 GB) 712 KB My Computer

The newly created directory will be refered to as **“MCNF_Directory”.**

(3) Double-click onto MCNF_Setup.exe in order to open the following application



(4) Then create MCNF icons on your desktop by clicking “Create Shortcuts”:



- MCNF is the shortcut to the **MCNF Launcher** which starts the MCNF Integrated Configuration and Runtime Environment.
- MCNF_NET is a MCNF embedded application running on the .NET platform. It works like the other “Hello Embedded World (HEW)” embedded demo applications – just on the .NET platform in your PC instead of e.g. an AVR ATmega328P chip.

(5) Finish installation by closing the window.

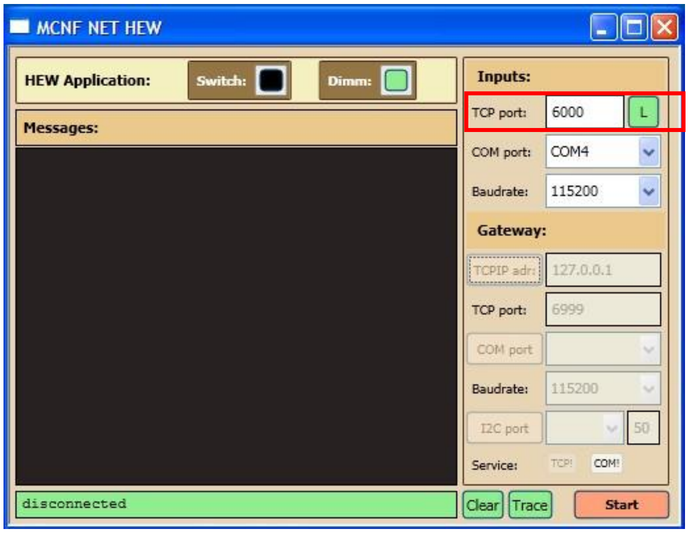
“Hello Embedded World” Demos

The equivalent to “Hello World” examples in the embedded world is to switch a LED on/off. We’re going to use that example in a basic version “HEW_1”, switching an LED on/off and in a version “HEW_2” where we add PWM based dimming of a LED.

Executing the .NET HEW Demo

First we're going to execute a demo that works entirely on the .NET platform. The "embedded" node and device are implemented in C# on the .NET platform. We will create a TCP/IP connection locally (port 6000 on "localhost") to connect the HEW application to its embedded counterpart.

(1) Open the HEW embedded part of the demo by clicking onto the "MCNF_NET" icon on your desktop. A window opens showing two LEDs "Switch" and "Dimm" on the top and a number of communication options on the right.



TCP port: 6000
COM port: COM4
Baudrate: 115200
TCPIP adr: 127.0.0.1
TCP port: 6999
Baudrate: 115200
I2C port 50
disconnected

6000 is the TCP/IP port

"L" green activates local host

We will use a TCP/IP connection locally on port 6000 on the PC.

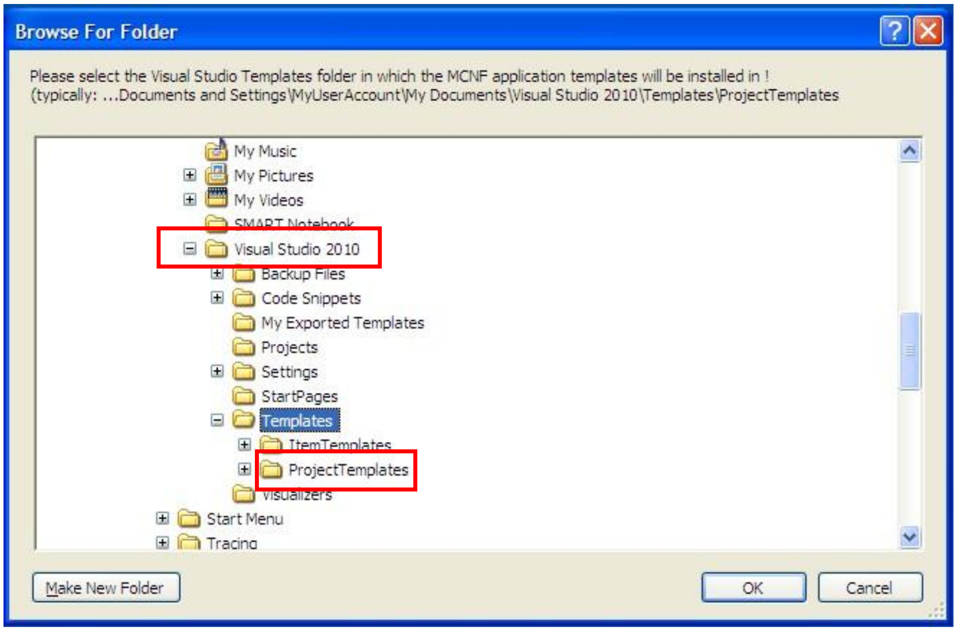
Make sure, the "L" button is activated = green color.

(2) Open the MCNF PC environment by clicking the "MCNF" icon on your desktop.

If you never launched MCNF before and your VisualStudio 2010 file path is not known by the system, you will be prompted to select the directory that holds the VisualStudio 2010 Project Templates.

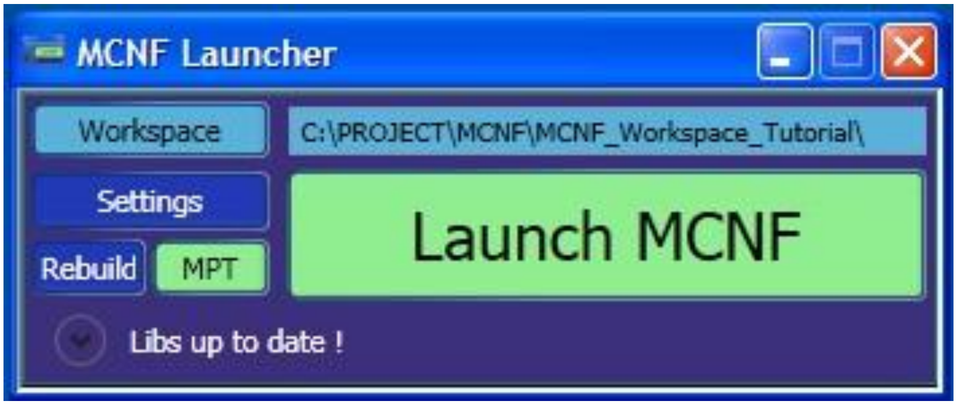
Usually, this directory will be under in your user account under "… Documents and Settings\MyUserAccount\My Documents\Visual Studio 2010\Templates\Project Templates".

| ← "MyUserAccount" is just a placeholder for your own personal user account



Please select the Visual Studio Templates folder in which the MCNF application templates will be installed in !
(typically: ...Documents and Settings\MyUserAccount\My Documents\Visual Studio 2010\Templates\ProjectTemplates

(3) The “MCNF Launcher” window will open and the content is grayed out while the system rebuilds our workspace (may take a little time).



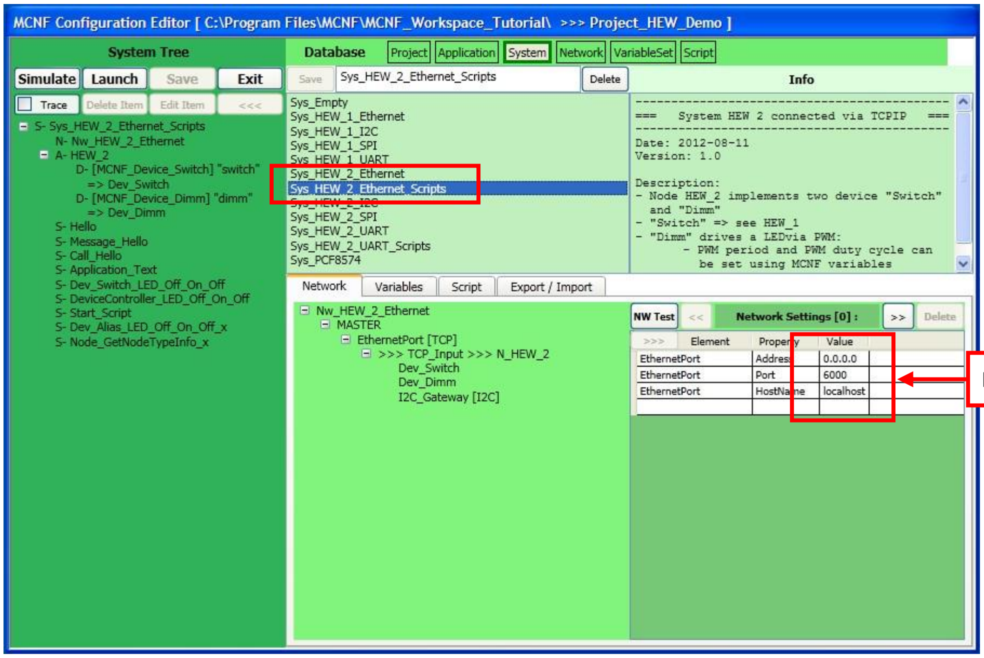
Workspace: C:\PROJECT\MCNF\MCNF_Workspace_Tutorial\

Libs up to date !

(4) Once the rebuild is finished, you should see the default workspace setting as

“MCNF_Directory”\MCNF\MCNF_Worksapce_Tutorial\

Press the “Launch MCNF” button to launch the MCNF Configuration Editor:


MCNF Configuration Editor [C:\Program Files\MCNF\MCNF_Workspace_Tutorial\ >>> Project_HEW_Demo]

Sys_HEW_2_Ethernet_Scripts

```
S- Sys_HEW_2_Ethernet_Scripts
  N- Nw_HEW_2_Ethernet
  A- HEW_2
      D- [MCNF_Device_Switch] "switch"
         => Dev_Switch
      D- [MCNF_Device_Dimm] "dimm"
         => Dev_Dimm
  S- Hello
  S- Message_Hello
  S- Call_Hello
  S- Application_Text
  S- Dev_Switch_LED_Off_On_Off
  S- DeviceController_LED_Off_On_Off
  S- Start_Script
  S- Dev_Alias_LED_Off_On_Off_x
  S- Node_GetNodeTypeInfo_x
```

```
Sys_Empty
Sys_HEW_1_Ethernet
Sys_HEW_1_I2C
Sys_HEW_1_SPI
Sys_HEW_1_UART
Sys_HEW_2_Ethernet
Sys_HEW_2_Ethernet_Scripts
Sys_HEW_2_I2C
Sys_HEW_2_SPI
Sys_HEW_2_UART
Sys_HEW_2_UART_Scripts
Sys_PCF8574
```

```
----------------------------------------------
===   System HEW 2 connected via TCPIP    ===
----------------------------------------------

Date: 2012-08-11
Version: 1.0

Description:
- Node HEW_2 implements two device "Switch"
  and "Dimm"
- "Switch" => see HEW_1
- "Dimm" drives a LEDvia PWM:
     - PWM period and PWM duty cycle can
       be set using MCNF variables
```

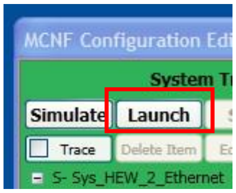
```
Nw_HEW_2_Ethernet
  MASTER
    EthernetPort [TCP]
      >>> TCP_Input >>> N_HEW_2
            Dev_Switch
            Dev_Dimm
            I2C_Gateway [I2C]
```

>>>	Element	Property	Value
	EthernetPort	Address	0.0.0.0
	EthernetPort	Port	6000
	EthernetPort	HostName	localhost

Port 6000 on local host

(5) Select the MCNF system “Sys_HEW_2_Ethernet_Scripts” as shown above.

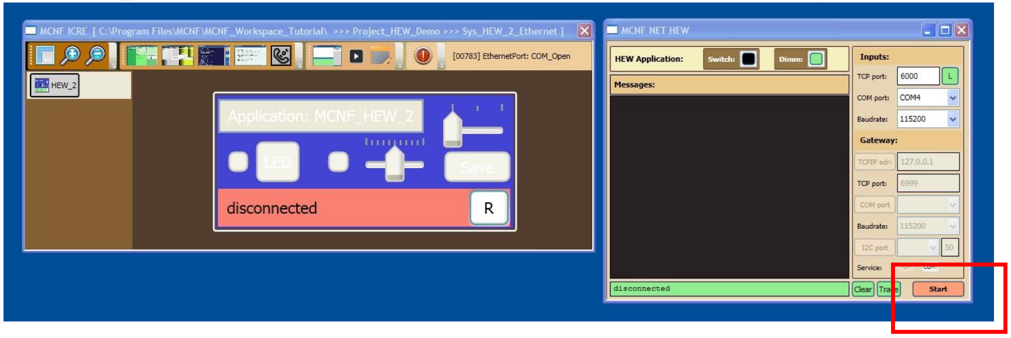
(6) Click the “Launch” button in the upper left corner of the MCNF Configuration Editor:



(7) The MCNF Execution Window is activated. You should now have two windows on your screen:

MCNF Execution Window with HEW_2 User Control

MCNF HEW_2 .NET embedded system


MCNF ICRE [C:\Program Files\MCNF\MCNF_Workspace_Tutorial\ >>> Project_HEW_Demo >>> Sys_HEW_2_Ethernet]

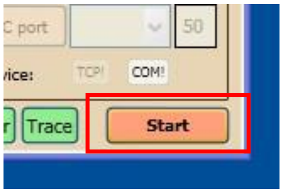
[00783] EthernetPort: COM_Open

- TCP port: 6000
- COM port: COM4
- Baudrate: 115200
- TCPIP adr: 127.0.0.1
- TCP port: 6999
- COM port:
- Baudrate: 115200
- I2C port: 50
- Service:

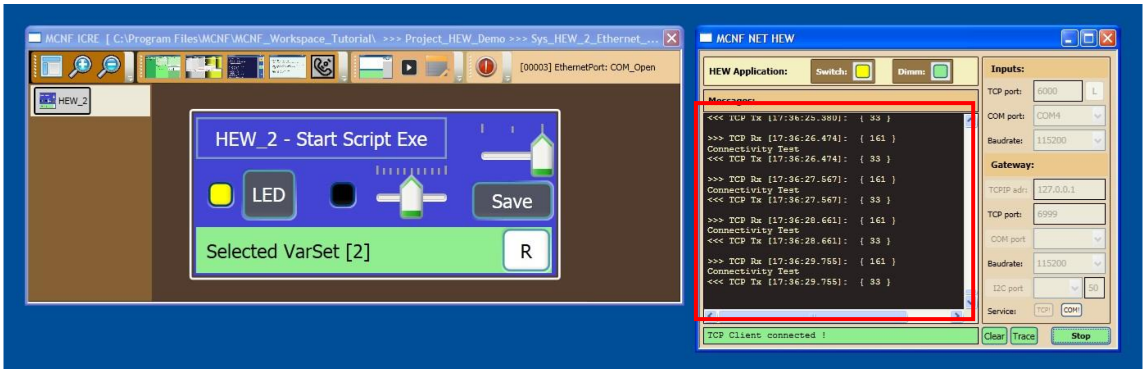
disconnected

(8) The HEW_2 user control still shows “disconnected” because we have not started the .NET embedded system yet. Press the “Start” button in the MCNF NET HEW window to start the

embedded system execution:



(9) Your system should change its appearance to show the active status and you should see the trace output of connectivity messages in the MCNF NET HEW window.


MCNF ICRE [C:\Program Files\MCNF\MCNF_Workspace_Tutorial\ >>> Project_HEW_Demo >>> Sys_HEW_2_Ethernet_...
[00003] EthernetPort: COM_Open
HEW_2
HEW_2 - Start Script Exe
LED
Save
Selected VarSet [2]
R
MCNF NET HEW
HEW Application:
Switch:
Dimm:
Messages:

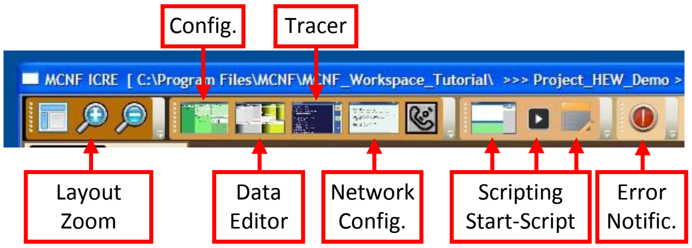
```
<<< TCP Tx [17:36:25.380]:  { 33 }

>>> TCP Rx [17:36:26.474]:  { 161 }
Connectivity Test
<<< TCP Tx [17:36:26.474]:  { 33 }

>>> TCP Rx [17:36:27.567]:  { 161 }
Connectivity Test
<<< TCP Tx [17:36:27.567]:  { 33 }

>>> TCP Rx [17:36:28.661]:  { 161 }
Connectivity Test
<<< TCP Tx [17:36:28.661]:  { 33 }

>>> TCP Rx [17:36:29.755]:  { 161 }
Connectivity Test
<<< TCP Tx [17:36:29.755]:  { 33 }

TCP Client connected !
```

Inputs:
TCP port: 6000
COM port: COM4
Baudrate: 115200

Gateway:
TCPIP adr: 127.0.0.1
TCP port: 6999
COM port
Baudrate: 115200
I2C port 50
Service: TCP! COM!

(10)Try out the HEW_2 user control and MCNF tools:

- Press “LED” and see the LED on/off messages and the visual change of LED “Switch”
- Press the second (green) LED and toggle between blink and dimming mode; use the slider to set dimming (PWM duty cycle); watch the HEW embedded application window.
- Use the “Save” button and the slider above the Save button to save and recall variable settings for the green LED.
- Try the tool bar icons to access the various MCNF tools:



- Re-launch the system from the Configuration Editor with “Simulate” instead of “Launch”.

(11) Close the MCNF Execution Window and the MCNF NET HEW window.

Executing the Arduino HEW Demo

This demo will execute the HEW_2 embedded application on an Arduino platform.

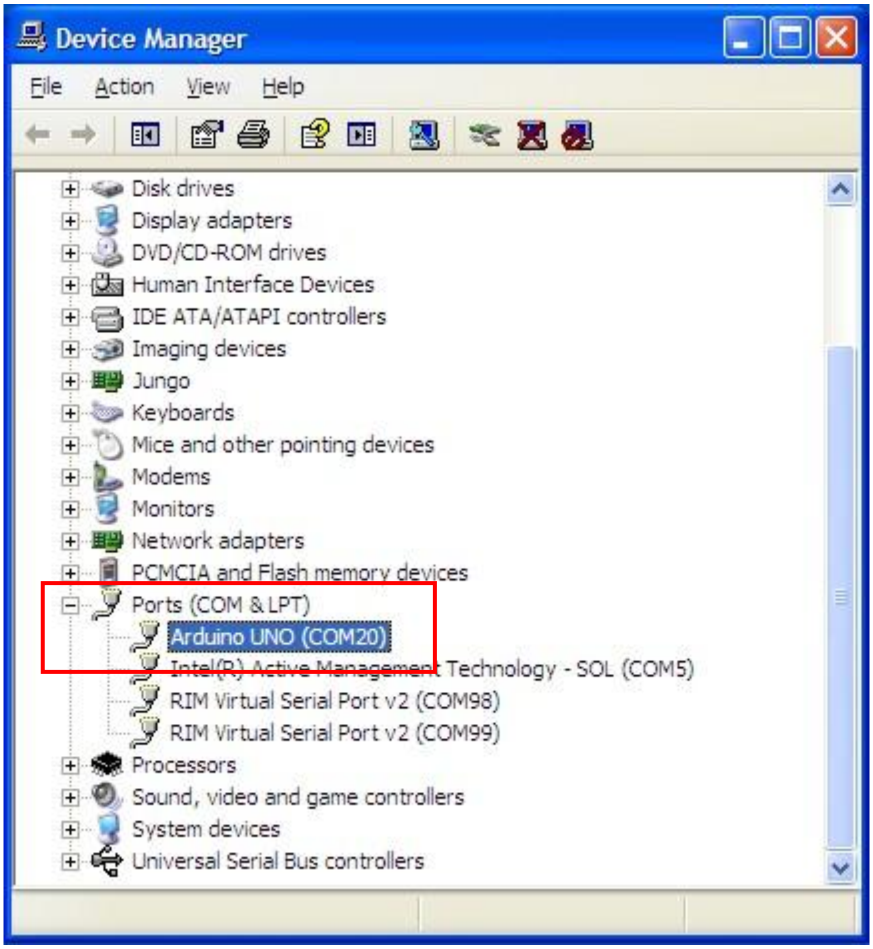
We're using an Arduino Uno in this example.

(1) Prepare the Arduino hardware by connecting two LEDs:

- yellow LED from Digital Pin 5 via a 1K resistor to ground.
- green LED from Digital Pin 9 (PWM) via a 1K resistor to ground.

(2) Connect the USB cable and find your COM port in the Windows Device Manager; in the example we assume COM20.

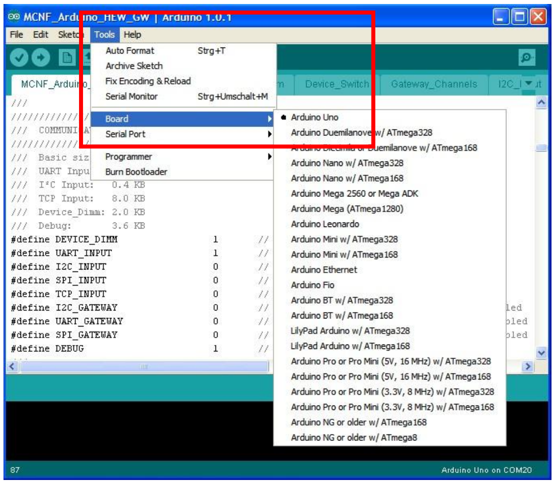
Remark: You can change the COM port asignment in the Windows Device Manager under Port Settings – Advanced.


Device Manager
File
Action
View
Help
Disk drives
Display adapters
DVD/CD-ROM drives
Human Interface Devices
IDE ATA/ATAPI controllers
Imaging devices
Jungo
Keyboards
Mice and other pointing devices
Modems
Monitors
Network adapters
PCMCIA and Flash memory devices
Ports (COM & LPT)
Arduino UNO (COM20)
Intel(R) Active Management Technology - SOL (COM5)
RIM Virtual Serial Port v2 (COM98)
RIM Virtual Serial Port v2 (COM99)
Processors
Sound, video and game controllers
System devices
Universal Serial Bus controllers

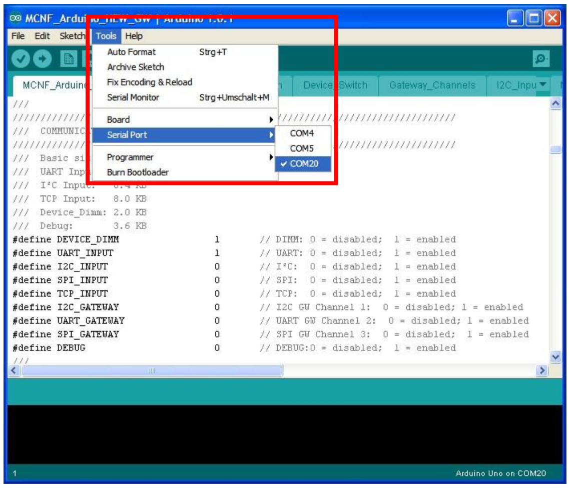
(3) Configure, compile and download the HEW application onto the Arduino hardware:

- goto *"MCNF_Directory"\MCNF_Embedded\Arduino\MCNF_Arduino_HEW_GW* and launch the Arduino IDE by double-clicking onto *MCNF_Arduino_HEW_GW.ino*.

- select Arduino UNO in Tools-Board and COM20 in Tools-Serial Port

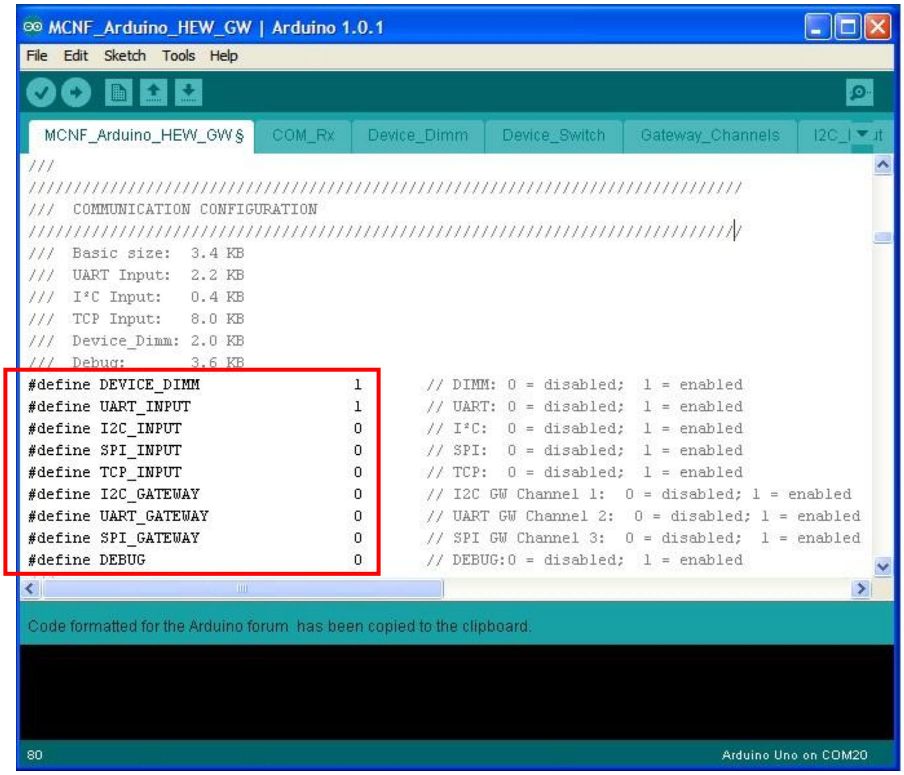

MCNF_Arduino_HEW_GW | Arduino 1.0.1
File Edit Sketch Tools Help
Auto Format Strg+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Strg+Umschalt+M
Board
Serial Port
Programmer
Burn Bootloader
● Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168
Arduino Nano w/ ATmega328
Arduino Nano w/ ATmega168
Arduino Mega 2560 or Mega ADK
Arduino Mega (ATmega1280)
Arduino Leonardo
Arduino Mini w/ ATmega328
Arduino Mini w/ ATmega168
Arduino Ethernet
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8
MCNF_Arduino_HEW_GW
Device_Switch
Gateway_Channels
I2C_Input
87
Arduino Uno on COM20

```
///
/////////////
///  COMMUNI
/////////////
///  Basic siz
///  UART Inpu
///  I²C Input:    0.4 KB
///  TCP Input:    8.0 KB
///  Device_Dimm: 2.0 KB
///  Debug:        3.6 KB
#define DEVICE_DIMM              1        //
#define UART_INPUT               1        //
#define I2C_INPUT                0        //
#define SPI_INPUT                0        //
#define TCP_INPUT                0        //
#define I2C_GATEWAY              0        //
#define UART_GATEWAY             0        //
#define SPI_GATEWAY              0        //
#define DEBUG                    1        //
```



```
///
/////////////                     //////////////////////////////////////////
///  COMMUNIC
/////////////                     //////////////////////////////////////////
///  Basic si
///  UART Inp
///  I²C Input:
///  TCP Input:    8.0 KB
///  Device_Dimm: 2.0 KB
///  Debug:        3.6 KB
#define DEVICE_DIMM              1        // DIMM: 0 = disabled;  1 = enabled
#define UART_INPUT               1        // UART: 0 = disabled;  1 = enabled
#define I2C_INPUT                0        // I²C:  0 = disabled;  1 = enabled
#define SPI_INPUT                0        // SPI:  0 = disabled;  1 = enabled
#define TCP_INPUT                0        // TCP:  0 = disabled;  1 = enabled
#define I2C_GATEWAY              0        // I2C GW Channel 1:  0 = disabled; 1 = enabled
#define UART_GATEWAY             0        // UART GW Channel 2:  0 = disabled; 1 = enabled
#define SPI_GATEWAY              0        // SPI GW Channel 3:  0 = disabled;  1 = enabled
#define DEBUG                    0        // DEBUG:0 = disabled;  1 = enabled
///
```

- In tab MCNF_Arduino_HEW_GW in section COMMUNICATION CONFIGURATION, set the #defines DEVICE_DIMM and UART_INPUT to 1, all others to 0.

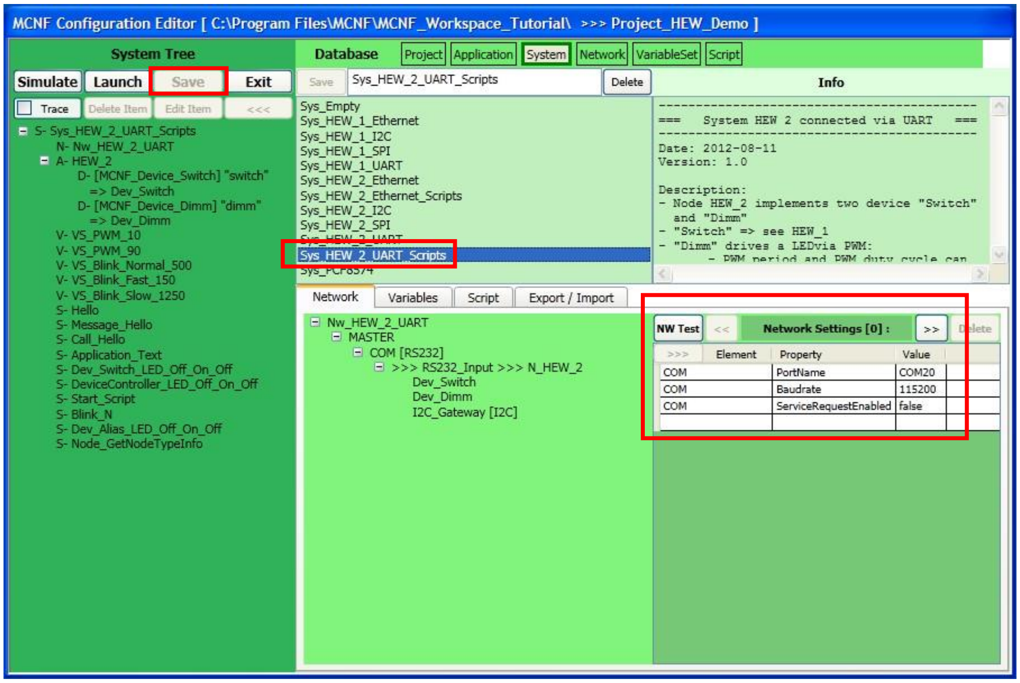


```
///
//////////////////////////////////////////////////////////////////////////////
///  COMMUNICATION CONFIGURATION
//////////////////////////////////////////////////////////////////////////////
///  Basic size:   3.4 KB
///  UART Input:   2.2 KB
///  I²C Input:    0.4 KB
///  TCP Input:    8.0 KB
///  Device_Dimm: 2.0 KB
///  Debug:        3.6 KB
#define DEVICE_DIMM              1        // DIMM: 0 = disabled;  1 = enabled
#define UART_INPUT               1        // UART: 0 = disabled;  1 = enabled
#define I2C_INPUT                0        // I²C:  0 = disabled;  1 = enabled
#define SPI_INPUT                0        // SPI:  0 = disabled;  1 = enabled
#define TCP_INPUT                0        // TCP:  0 = disabled;  1 = enabled
#define I2C_GATEWAY              0        // I2C GW Channel 1:  0 = disabled; 1 = enabled
#define UART_GATEWAY             0        // UART GW Channel 2:  0 = disabled; 1 = enabled
#define SPI_GATEWAY              0        // SPI GW Channel 3:  0 = disabled;  1 = enabled
#define DEBUG                    0        // DEBUG:0 = disabled;  1 = enabled
```

Code formatted for the Arduino forum has been copied to the clipboard.

- save the changes, compile and upload the code

(4) Launch the MCNF environment and select system “Sys_HEW_2_UART_Scripts”. In the lower right section you can define the Network Settings for your configuration. Make sure that the Arduino’s COM port is defined in the network settings; edit and save the changes if necessary.



MCNF Configuration Editor [C:\Program Files\MCNF\MCNF_Workspace_Tutorial\ >>> Project_HEW_Demo]

```
S- Sys_HEW_2_UART_Scripts
   N- Nw_HEW_2_UART
   A- HEW_2
      D- [MCNF_Device_Switch] "switch"
         => Dev_Switch
      D- [MCNF_Device_Dimm] "dimm"
         => Dev_Dimm
   V- VS_PWM_10
   V- VS_PWM_90
   V- VS_Blink_Normal_500
   V- VS_Blink_Fast_150
   V- VS_Blink_Slow_1250
   S- Hello
   S- Message_Hello
   S- Call_Hello
   S- Application_Text
   S- Dev_Switch_LED_Off_On_Off
   S- DeviceController_LED_Off_On_Off
   S- Start_Script
   S- Blink_N
   S- Dev_Alias_LED_Off_On_Off
   S- Node_GetNodeTypeInfo
```

Sys_HEW_2_UART_Scripts

```
Sys_Empty
Sys_HEW_1_Ethernet
Sys_HEW_1_I2C
Sys_HEW_1_SPI
Sys_HEW_1_UART
Sys_HEW_2_Ethernet
Sys_HEW_2_Ethernet_Scripts
Sys_HEW_2_I2C
Sys_HEW_2_SPI
Sys_HEW_2_UART
Sys_HEW_2_UART_Scripts
Sys_PCF8574
```

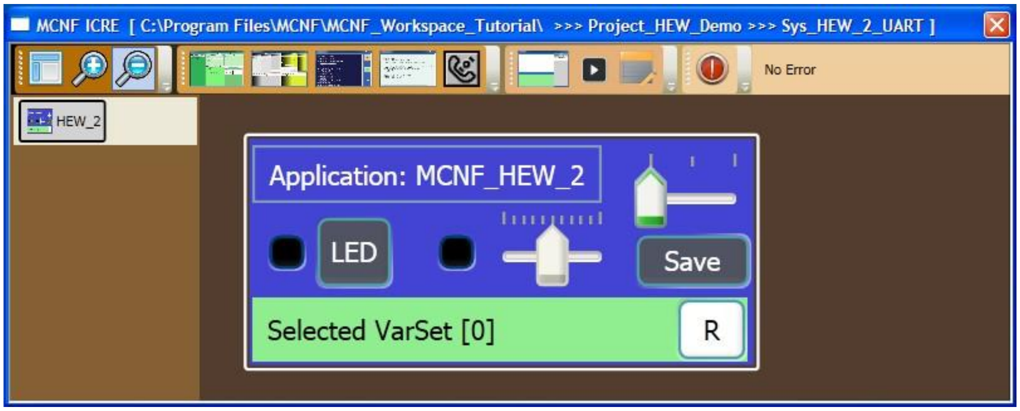
```
-----------------------------------------
===   System HEW 2 connected via UART   ===
-----------------------------------------
Date: 2012-08-11
Version: 1.0

Description:
- Node HEW_2 implements two device "Switch"
  and "Dimm"
- "Switch" => see HEW_1
- "Dimm" drives a LEDvia PWM:
     - PWM period and PWM duty cycle can
```

```
Nw_HEW_2_UART
   MASTER
      COM [RS232]
         >>> RS232_Input >>> N_HEW_2
            Dev_Switch
            Dev_Dimm
            I2C_Gateway [I2C]
```

Network Settings [0] :

>>> Element	Property	Value
COM	PortName	COM20
COM	Baudrate	115200
COM	ServiceRequestEnabled	false

(5) Eventually save all changes with “Save” and press “Launch” to start executing the demo.



MCNF ICRE [C:\Program Files\MCNF\MCNF_Workspace_Tutorial\ >>> Project_HEW_Demo >>> Sys_HEW_2_UART]

No Error

(6) Press the Arduino hardware reset button to see the effect of hardware disconnecting and re-connecting.

(7) Experiment with the tools, run scripts etc.

(8) Close the MCNF Execution Window.