

Web51-C – Getting Started

Web51-C is a development kit for creating embedded Ethernet UDP and SNMP applications based on the 8051 microcontroller family.

Web51-C is a migration of the Web51 project that was completely written in assembler, to the C language. The free SDCC or the commercial KEIL C51 compiler can compile the application code.

The Web51-C project should be compiled based on the current HW (starting with the first KB of internal RAM if using a newer x51 microcontroller (e.g. AT89C51RD2). More memory (RAM) is required if programming SNMP applications.

An example of target hardware is the **Charon I** module, which contains enough memory to embed all of the Web51-C applications.

This document describes the installation and testing of some of the Web51-C applications.

The basic description of Web51-C project

- Rapid application development
- Source code in ANSI C
- Uses an Atmel AT89C51RD2 microcontroller and Realtek RTL8019AS Ethernet controller
- Fully open HW and SW platform.
- Compatible with our simple, low cost and powerful HW solution (**Charon I + Development Kit** or **Charon I + Charon I&II Development Board**).
- Available in fully documented source code.
- **UDP/IP stack** implemented.
- Examples are **SDCC** (Small Device C Compiler) compatible
- Examples for the commercial **Keil C51** compiler available
- Fully developed SNMP application called “**SNMP I/O Thermometer**” using JAVA visualization
- Detailed description and getting started guide

Web51 Project

Web51

8051 GNU binutils (assembler, linker..)
WWW server on 256 B RAM,
Ethernet - RS-232 client/server converter,
Charon I module compatible

Web51-C

8051 ANSI C UDP stack
SDCC & Keil C51 core compatible,
SNMP applications available,
Charon I module compatible

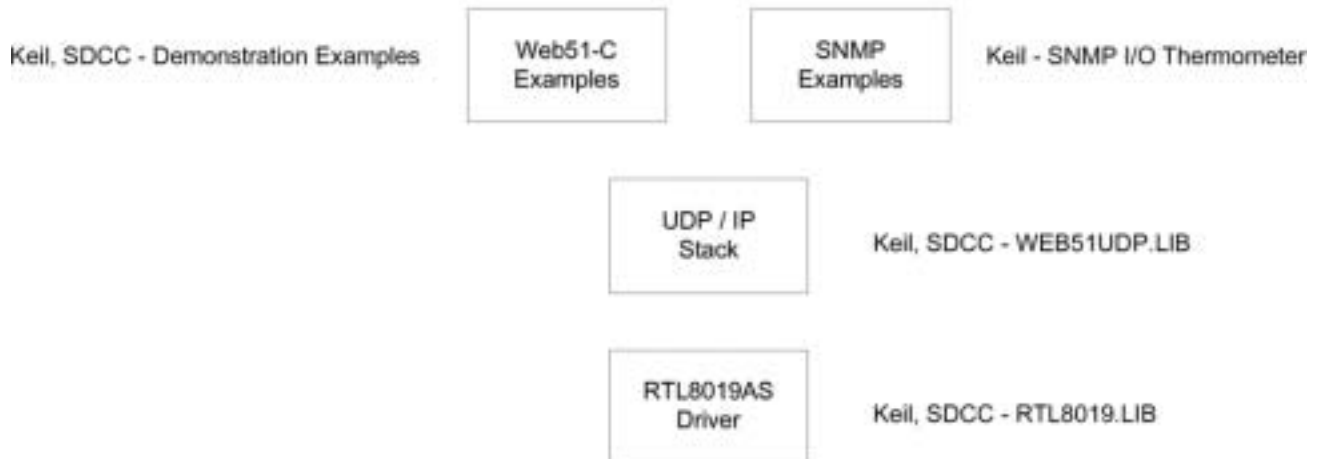


HW devices

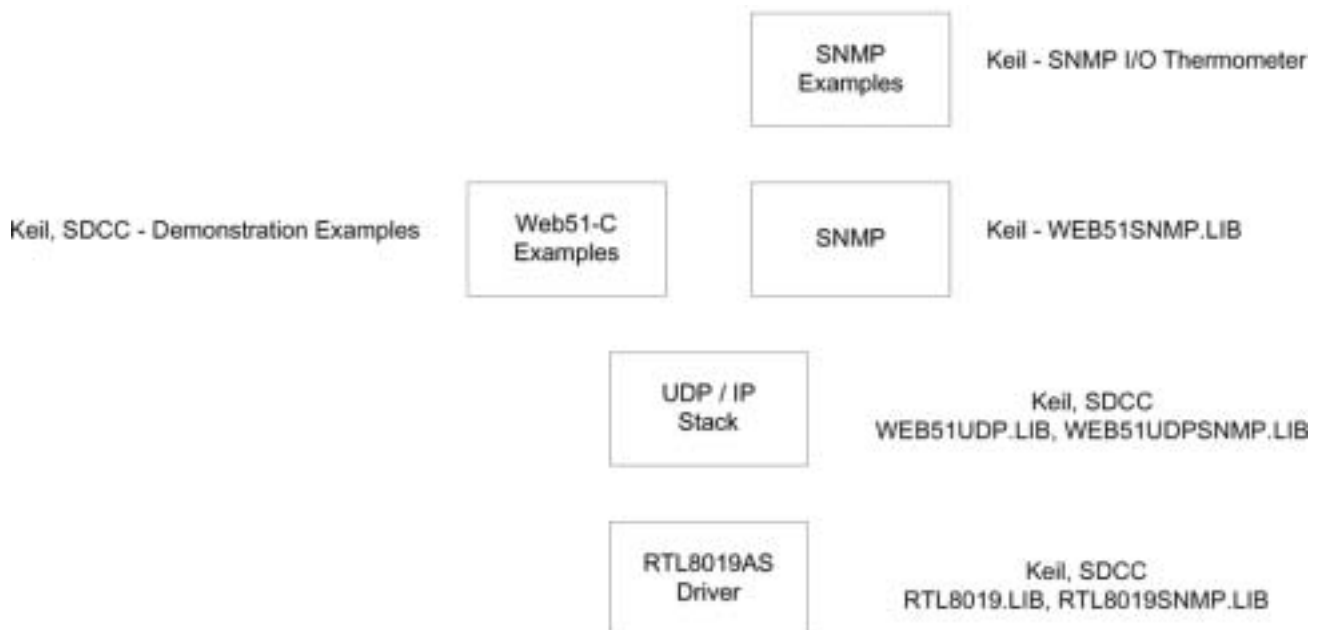


Functional Block Diagram of Web51-C

Demo Version



ANSI C - SNMP Development System



Project Hardware

The project is based on the Atmel AT89C51RD2 microcontroller and the Realtek RTL8019AS Ethernet controller. We use our **Charon I** built-in module for the target applications. You can find the detailed description of this module in the attachments section or on our web site.

The memory sub-system is made by the

- Integrated FLASH 64 kB memory (program code),
- Integrated EEPROM 2 kB memory (configuration parameters),
- External SRAM 32 kB memory (XRAM memory access),
- Directly mapped peripherals - RTL8019AS Ethernet controller.

Address Space	Used
0x0000 – 0xFBFF	FLASH 63 kB (CODE)
0xFC00 – 0xFFFF	Bootloader 1 kB

Sheet.1 The memory Sub-system of Web51-C – internal program **CODE** memory

Address space	Used
0x0000 – 0x7FFF	SRAM 32 kB (XRAM)
0x8000 – 0x801F	RTL8019as Ethernet controller
0x8020 – 0x80FF	Free for additional HW
0x8100 – 0xFFFF	Overlaps 0x8000 – 0x80FF address space

Sheet.2 The memory Sub-system of Web51-C – external data **XRAM** memory

Address space	Used	Address space	Used
0x0000 – 0x03FF	SRAM 1 kB (XRAM)	0x0000 – 0x07FF	EEPROM 2 kB

Tab.3 The memory Sub-system of Web51-C – internal data **XRAM** and **EEPROM** memory

Using the ISP method with the AT89C51RD2 serial port (see chapter “Firmware Programming”) performs the programming of the internal FLASH and EEPROM memory. This method requires no additional hardware and fully eliminates using of a classical microcontroller programmer.

The Realtek RTL8019AS Ethernet converter is connected in the minimal mode without the external 93c46 EEPROM memory. The communication with the controller is in 8-bit mode, which allows you to address the 8kB of controller’s internal SRAM memory. This internal memory is used for receiving and transmitting the Ethernet packets. After the Ethernet controller receives the packet, an interrupt signal generated.

Recommended Hardware

The examples provided were originally developed for the Charon I module and work with the following development kits:

Charon I DK (*Development Kit*) is a simple and low cost development kit which you can use for testing and final applications. It contains only basic peripherals.

Charon I&II DB (*Development Board*) is a development kit which contains direct inputs and outputs, shift registers for external inputs/outputs, LCD display connection and direct 1-Wire output. The Charon I&II DB also contains a second serial RS-232 port, JTAG, SERIAL FLASH up to 4 MB and an ISP programming interface.

Either of these two development boards is recommended for testing your application. Your production product should use the Charon I or II module combined with your product specific peripheral circuitry.

Supported Peripherals on the Charon I&II Development Board

Serial shift register I/O Ports

The DB series boards contain serial shift register support for extending parallel I/O. The input/output width (maximum 32) depends on the number of connected shift registers. This extended peripheral mode can be enabled by a **special** set command issued from either the JAVA application or snmp client. In the "**parallel**" mode, you can access the microcontroller's P1 8 bit I/O port directly.

- 0..32x shifted binary inputs
- 0..32x shifted binary outputs
- **Serial port RS-232**
All incoming serial port data are packetized and sent as traps to the SNMP client.
- **Up to 4x 1-Wire thermometers.**
The 1-Wire interface is typically used with the DS18S22 or DS18B20 temperature sensors. It's connected to GND, +5V and DATA. You can connect up to 4 1-Wire thermometers, there are auto detect routines for searching more temperature sensors

Software Methodology

Because the Web51-C project relies on using every bit of horsepower in a resource limited MICROCONTROLLER, the project needed to be architected to minimize these limitations. Project prerequisites are:

- The use of the higher programming language such as 'C'
- Dynamic memory control
- Peripheral access via special pre-programmed functions
- Basic network protocol implementation
- Fast system response time
- Well documented for support and application integration

Why use the C language?

There are many operating systems and applications written in C. Embedded, hardware specific code is easily written or ported in C. C compilers are now available for most microcontrollers.

C language advantages:

- Easy source code portability
- Multi-platform programming
- Quicker application development (standard libraries usage)
- Optimization and validation tools
- The language is well known and supported by the microcontroller vendors
- Many SW vendors available

C language disadvantages:

- Higher price for the high quality development tool
- Higher data and code memory requirements

The C language implementation in the 8051 microcontroller family

Due to its popularity, there are many compilers available for the 8051 family of microcontrollers. The basic property of the C language is that it uses a dynamic stack (parameter exchange, local variables, return addresses and many other data) for program operation. This poses the greatest issue using C compiled programs together with 8bit microcontrollers. It very depends on the C compiler producer how the stack issue is addressed.

Choosing the C language compiler for Web51

Basic Environment and operating criteria:

- The Web51-C system kernel must be compiled by at least two different compilers (a commercial and free one)
- The application development must be available for both Windows and Linux machines
- Applications must be able to be quickly developed

GNU SDCC Compiler (*Small Device C Compiler*), a freeware compiler, is much better as compared to many commercial projects. We can say, with confidence, that it is situated somewhere in the middle of the compiler ranking list. However, It lacks an IDE and many additional development tools found in commercial products. The expanded C language 8051 syntax is included and it's the same as in the Keil compiler. So, it is easy to write programs for SDCC that are directly portable to Keil.

However, because of the lack of an IDE, you need to be well skilled in using the GNU tool chain (make, ar, etc.) when programming applications with many separate modules.

Keil C51 is one of, if not the best C compiler in the 8051 world. It has an integrated IDE, so the project development cycle is easier.

Note: *The Keil compiler uses the Big Endian method for saving the data to the memory compared to SDCC, which uses the Little Endian method.*

The project folder structure

The Web51-C project is divided to the following folders:

- web51clapp** ... The user applications (for instance the demonstration examples)
- web51c\bin** ... The other Web51-C system. Add on programs (e.g. RD2-Flasher)
- web51c\dev** ... Hardware device drivers (e.g. RTL8019AS Ethernet controller driver)
- web51c\doc** ... Documentation
- web51c\include** ... The Web51-C system header files
- web51c\lib** ... Pre-Compiled libraries (udp, snmp library, etc.)
- web51c\net** ... The Web51-C system source code.

Network protocols implemented in Web51-C project

The integrated UDP/IP protocol stack covers most of the OSI reference model. The following layers are implemented:

Data link layer

- Sending and receiving packets using the Ethernet II standard.
- ARP (Address Resolution Protocol) which allows one machine to find a peer machine's MAC address given it's IP address.

Network layer

- IP (Internet Protocol) – The basic network service, which sends the packets, based on the network IP addresses in the packet's description.
- ICMP (Internet Control Message Protocol) – The control report protocol, which allows the system to send the error reports or other necessary information (i.e. ping).

Transport layer

- UDP (User Datagram Protocol) – Connectionless transport service between two stations on an inter/intranet.

Application layer

- SNMP (Simple Network Management Protocol) – This protocol supports the information transfer for network management. It is used for the monitoring, control and administration of devices on a network.
- DNS (Domain Name System) – It allows the IP address -> Name mapping.
- WOL (Wake On Lan) – Switching on the device in the local network based on packets destined for it.
- Application specific protocols, which are used in the user defined applications (for instance The Temperature Measurement example)

The SNMP implementation was developed only for the Keil C compiler.

Developing and debugging applications

Application development is done on either a Linux or Windows platform. The programmer should choose between the SDCC (compiled using make files) and Keil (μ Vision2 IDE environment) compilers. There are several example programs for better understanding the Web51-C programming in the **web51clapp** folder.

Developing Applications in Keil C51

Linking the applications is done in the μ Vision2 IDE environment, which contains an integrated compiler, linker and debugger.

The AT89c51RD2 processor's EEPROM memory configuration

The Web51-C system uses the internal EEPROM memory to save configuration parameters (see example 1). It is necessary to configure it in the following manner:

The source codes of the example are co-located with the web51.uv2 project. The project includes the information useful for the compiler and linker together. You can open the project in *Project* → *Open Project* menu, where you have to add the following path: `\web51c\app\00.eeprom.setup` and choose *web51.uv2* project.

The project's information is displayed in the left window:

- The SRC folder – the source code of the eesetup.c (EEPROM SETUP) program,
- The INC folder – the header files of the Web51-C system,
- The LIB folder – the Web51Udp.LIB used libraries Web51Udp.LIB (The whole Web51-C system including the UDP protocol stack).

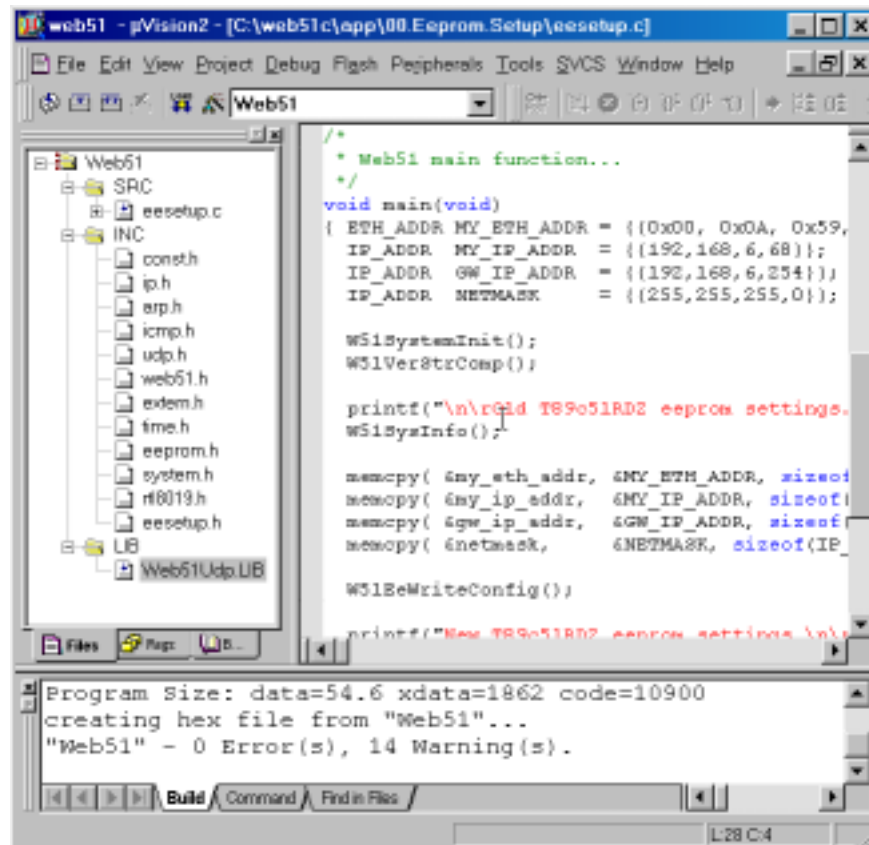


fig. 1: Keil C51 -

μVision2 IDE

```

ETH_ADDR MY_ETH_ADDR = {{0x00, 0x0A, 0x59, 0x00, 0x00, 0x00}}; // Web51 MAC address
IP_ADDR MY_IP_ADDR = {{192,168,6,68}}; // Web51 IP address
IP_ADDR GW_IP_ADDR = {{192,168,6,254}}; // mask
IP_ADDR NETMASK = {{255,255,255,0}}; // Gateway IP address

```

The IP address values must be set to correspond with the other IP addresses in the local network, where the Charon device is connected.

Build a project in the *Project* → *Build target* menu. The following files are created after the compilation is done:

File name	Description
WEB51	input data for debugger
WEB51.HEX	the program code (Intel Hex file is transferred to the MCU)
WEB51.M51	symbol table
WEB51.OPT	current settings of μVision2 (opened files, window size, ...)
WEB51.UV2	project file
EESETUP.C	source code
EESETUP.LST	the output compilation list

note: *The compilation result is in the bottom of Figure 1. The Web51Udp.LIB library contains some of the information, which is not used in the program. That's why the compiler reports 14 warnings. These warnings are not an error, but only a notice, that we have increased the code size without any effect. The user can remove the functions, which are not used during the compilation process in the \include\config.h file. This step is recommended for more advanced users.*

You can upload the firmware (the result of the compilation `web51.hex`) to the Charon I module using the `flash.bat` file (saved in the same directory as the project as well).

```

Web51c - Embedded Ethernet Project ver 1.0.0 May 20 2003, 21:20:13 - Keil 7.05

Old T89c51RD2 eeprom settings.

Web51 network interface parameters

MAC address : 00:0A:59:00:00:01
IP address  : 192.168.2.6
Gateway    : 192.168.2.1
Netmask    : 255.255.255.0

New T89c51RD2 eeprom settings.

Web51 network interface parameters

MAC address : 00:0A:59:00:00:00
IP address  : 192.168.6.68
Gateway    : 192.168.6.254
Netmask    : 255.255.255.0

```

Fig. 2: The program's output to the serial port 9600, N, 8, 1

The EEPROM memory should now be configured. If the data were stored successfully, the screen in Figure 2 is displayed following module reset.

Web51-C system applications

The following describes the basic use of the Web51-C system. Communication between the Charon I module and PC is performed using the UDP example shown. The program receives the sent data by PC at the IP address, port 2000. The UDP packet's data are displayed to the standard output port (serial port).

Open the project in the μ Vision2 environment (see the guide in the previous example). The source code of the example is stored in the `web51c\app\02.udp.client` folder.

The main function

The whole program is in the `udpclient.c` file. First, reserve the memory for the incoming UDP packets:

```

u_char EthFrmBuf[ ETH_FRM_SIZE ];    /* The memory for the received UDP datagram */
data WORD EthProt;                  /* The type of the detected protocol */

```

The next step is Web51-C system initialization

- Setting the serial port parameters (9600,N,8,1),
- Reading the configuration from the EEPROM memory,
- Initialization of the ARP CACHE
- Initialization of the RTL8019as Ethernet controller.

```

W51SystemInit(); /* Web51-C system initialization*/
W51VerStrComp(); /* Displaying the actual system version to the standard output */
W51SysInfo();    /* Displaying the configuration (MAC, IP address act.) */

```


Application functionality check

Build the program with *Project* → *Build target* menu, upload the web51.hex file to the Charon I module. Try to ping the module to check the availability (for instance ping 192.168.6.68).

```

Sending the requirement to the host 192.168.6.68 with 32 bytes of data:

Response from 192.168.6.68: bytes=32 time=5ms TTL=64
Response from 192.168.6.68: bytes=32 time=5ms TTL=64
Response from 192.168.6.68: bytes=32 time=5ms TTL=64
Response from 192.168.6.68: bytes=32 time=5ms TTL=64
Response from 192.168.6.68: bytes=32 time=5ms TTL=64

```

Fig. 3: The Web51-C system response

There is an `udp_client.exe` PC application for testing the example stored in the `web51c\app\02.udp.client\udp.client.pc` folder. The application reads the data from the standard input (a keyboard) and sends them to the port 2000 of the module.

Running the UDP client on PC: `udp_client 192.168.6.68`.

```

--[ Web51 Demo Application ver 0.1.0 (C) 2002 by www.HW.cz ]--

UDP client sends data to remote Web51 board via UDP protocol.

Local host   : 192.168.6.67 - 192.168.6.67
Remote host  : 192.168.6.68 - 192.168.6.68

Enter any characters or press ESC to terminate program.

Web51-C project based on T89c51RD2 and RTL8019as.

```

Fig. 4: `udp_client.exe` running on the PC

Enter the test string „*Web51-C project based on T89c51RD2 and RTL8019as.*“. You can see the received data by Charon I module on Fig. 5:

```

Web51c - Embedded Ethernet Project ver 1.0.0 May 20 2003, 21:20:13 - Keil 7.05
--[ Web51 UDP Client - Demo Application ver. 0.2.0 (C) 2003 by www.HW.cz ]--

UDP client receives data from PC via UDP protocol on port 2000 and
displayes it on serial port.

Web51 network interface parameters
MAC address : 00:0A:59:00:00:00
IP address  : 192.168.6.68
Gateway    : 192.168.6.254
Netmask    : 255.255.255.0

Received data

Web51-C project based on T89c51RD2 and RTL8019as.

```

Fig. 5: The received data by the Charon I module displayed by serial port terminal.

Web51-C system demonstration examples

The Web51-C system contains the following demonstration examples for better understanding of the topics:

- **EEPROM** – The EEPROM memory of the Web51-C system.
- **Wake On Lan** – Switching on the station using the UDP datagram.
- **UDP client** – Receiving the UDP datagram and displaying the data on the serial port.
- **UDP server** – Sending an UDP datagram to the set IP address.
- **UDP echo server** – Receiving and then sending back the UDP datagram.
- **ICMP ping** – The remote computer availability test using the ICMP test report.
- **SNMP I/O Thermometer** – Using the SNMP protocol for the temperature monitoring and controlling the I/O pins.

Note: *The Web51-C system uses the serial port of the T89c51RD2 processor as the standard input/output. You can use any standard serial terminal to display or send the data. You need to set the communication parameters to 9600, N, 8, 1 and disable data flow control.*

Example 1 – EEPROM

The Web51-C system uses the T89c51RD2 internal EEPROM for saving the configuration parameters as for instance IP, mask, gateway and so on.

Address	Description	Length [B]
0x000	Web51 MAC address	6
0x006	Web51 IP address	4
0x00A	Mask	4
0x00E	Gateway IP address	4
0x012	DNS server IP address	4

Table.4 The internal EEPROM memory structure of the Web51-C system

The program sets the Web51-C configuration parameters to the following values:

- Web51 MAC - The MAC address reserved for the Web51-C project. 00:0A:59:00:00:00.
- Web51 IP address – 192.168.6.68
- Mask – 255.255.255.0
- Gateway IP – 192.168.6.254

The source code of the example:

`web51c\app\00.eeprom.setup.`

Note: *The SNMP protocol implementation uses its own adaptive EEPROM memory structure, which is not the same as described in the tab. 4.*

Example 2 – Wake On Lan (WOL)

The Wake On Lan technology (Magic Packet Technology) was developed by AMD and Hewlett Packard companies. WOL allows switching on the computer on the local network using a special UDP datagram. The UDP datagram has to contain 6xFF sequence, where there is 16xMAC address sent behind this starting FF sequence. The computer must have the mainboard and the LAN board, which supports WOL. The WOL support has to be enabled in the BIOS.

The source code of the example: `web51c\app\01.wake.on.lan.`

Example 3 – UDP client

The program receives the data from PC sent to Web51 IP address and port 2000. The UDP datagram's data are displayed on the standard output (serial port).

There is an application developed for testing this example called `udp_client.exe`, which reads the data from the standard input (a keyboard) and sends everything to Web51 system. The application is written in C language and you can build it using the MS Visual C++ environment or by using a free LCC-WIN32 project. IP address of the Web51 system is passed as a command line parameter for instance `udp_client.exe 192.168.6.68`.

The source code of the example: `web51c\app\02.udp.client.`

Example 4 – UDP server

The program sends a UDP datagram each second from the Web51 system to the IP address of the PC server to the port 4000. The UDP datagram contains the information about a temperature (simulated as an incremental value incremented each time the datagram is sent).

There is also a PC server application developed for testing this example called `udp_server.exe`, which receives the incoming data from the Web51 system. The application is written in C language and you should build it in MS Visual C++ or LCC-WIN32 as well.

The source code of the example: `web51c\app\03.udp.server.`

Example 5 – UDP echo server

The program reads the data from any UDP port, displays them using the standard output and replies the original datagram back as quick as possible. It is an enhancement of the example 3 demonstration, when the system works in a query-answer mode. You can use the `udp_client.exe` application for testing the example.

The source code of the example: `web51c\app\04.udp.echo.server.`

Example 6 – ICMP ping

The program tests the remote PC availability using the echo request and echo reply ICMP messages. It's a ping command normally used by the PC.

The source code of the example: `web51c\app\05.icmp.ping.`

Example 7 – SNMP

The example provides a detailed demonstration of Web51 SNMP. The application controls the binary inputs/outputs, LCD display, serial link RS-232 and 1-Wire thermometer sensor.

You have to use the Keil C51 compiler to build it. Unfortunately the SDCC version is not ready yet.

In the last part of this text, an in-depth description on how to use the “**SNMP I/O Thermometer**” application is provided. This is also available as a standalone document.

The SNMP source codes of the SNMP examples are stored in the following folders:

- `web51c\app\10.snmp.LED,`
- `web51c\app\10.snmp.LED_table`
- `web51c\app\ 10.snmp.LED_variable_all`
- `web51c\app\10.snmp.serial_io`

Programming the firmware

Atmel Flip

The Charon Module has the **Converter Ethernet / RS-232** firmware pre-installed, so the first step is to change to the firmware containing SNMP application. We will use the original Atmel's **Flip** program for uploading to the T89C51RD2 processor.

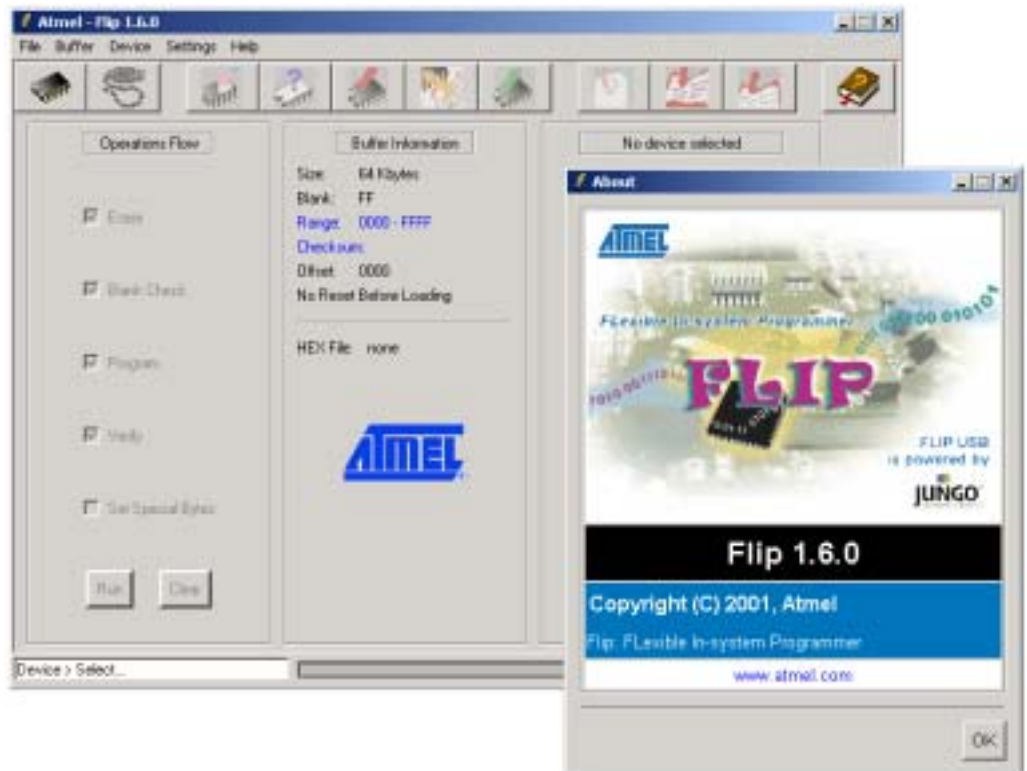
You should download the FLIP from official web pages of Atmel company or you should download an older, stable version from

www.HWgroup.cz

see page **DOWNLOAD**.

PSEN – programming

Connect the module Charon to the RS-232 serial link (using the serial cable, which is included in the development kit) and connect the **PSEN** jumper.



You can find the PSEN jumper's position on the development board (**Charon I&II DB** on the left side-the bottom position "Forced", **Charon I DK** under the LINK and POWER LEDs).

After you have connected the PSEN jumper, switch the power supply on. Select the processor type (Device → Select → T89C51RD2) in the Flip program and open the firmware file with the **HEX** extension.

Set the RS-232 programming port (Settings → Communication → RS-232). If there was an error displayed, check the PSEN jumper position, or check the serial cable type (3 wire, RX, TX, GND) (if you are using **Charon I&II DB**, check the RESET jumper position - must be in x51 pos. and reset the module)..

- Do not forget to check the ERASE and BLANK CHECK checkboxes.
- Disconnect the Atmel Flip program after programming has done.
- Finally disconnect the PSEN jumper.

- **Connect the SETUP (T0) jumper.**

Start any type of serial terminal (TeraTerm, Terminal), or use our "Ethernet Converter SETUP", which you can download from **DOWNLOAD** page on www.HWgroup.cz.

Programming the firmware – RD2-Flasher

The RD2-Flasher program is for programming the Atmel's MICROCONTROLLER using the ISP method and serial port in a Windows system.

The ISP method works together with the Loader program, which is stored in the last 1kB of the internal Flash memory of the MICROCONTROLLER. The Flasher program is a typical 32b command line application working under Windows. It's quite easy to implement it with any development kit. (e.g. Keil μ Vision2, SDCC, act.) The source codes are available in the Web51 project.

The RD2-Flasher is an alternative solution. It is fully compatible with the original microcontroller's loader, but it allows you to program internal EEPROM memory using the ISP method. It can also switch the MICROCONTROLLER to the programming mode automatically using the PSEN pin and CTS output pin from RS-232 port.

With the AT89C51RE2 microcontrollers you can use original FLIP from Atmel for programming internal EEPROM too.

ISP programming method

Setting up the PSEN jumper is the same as in the previous case. The detailed RD2-Flasher description is available at http://www.hw.cz/software/rd2_flasher/flasher22.html.

An example how to load the snmp.hex file into the Charon I module:

```
RD2F -f snmp.hex -c COM1 -b 115200 -m 0 -p 1 -t 2 -q 18 -i 32 -v 1
```

Getting started - SNMP I/O Thermometer application

RS-232 SETUP for the SNMP I/O Thermometer Application

Set the serial port parameters to 9600 Bd 8N1, no handshake, in the Terminal application. Be sure you have connected the SETUP jumper and PSEN jumper is disconnected. Reset the module.

ATTENTION

After each firmware upload, you must make **RESET** all the module's **SETUP** parameters to default. The internal **EEPROM** must be initialized by this way. Otherwise, the module does not work properly.

Don't forget to use this command:

`d ... load default setup`

After the setup is reinitialized, set the IP address, Gateway, Mask and "**target trap address**".

`t ... enter target trap address`

Target trap address is the IP address where all the UDP packets with SNMP traps will be sent. This address is the only one that should communicate with the module.

If the target is your PC...

and you are using a DHCP server, or you can't remember the IP address, run the "IPconfig" utility in command line – The utility will report the DNS, IP, MASK and GATEWAY. The other properties are 0.0.0.0 = OFF (do not send any UDP packets), 255.255.255.255 = UDP broadcast.

`v ... set port value`

This command defines the I/O port's value after power on. For example, you may want to switch off some devices after the system restart. The value is a decimal value in range 0..255

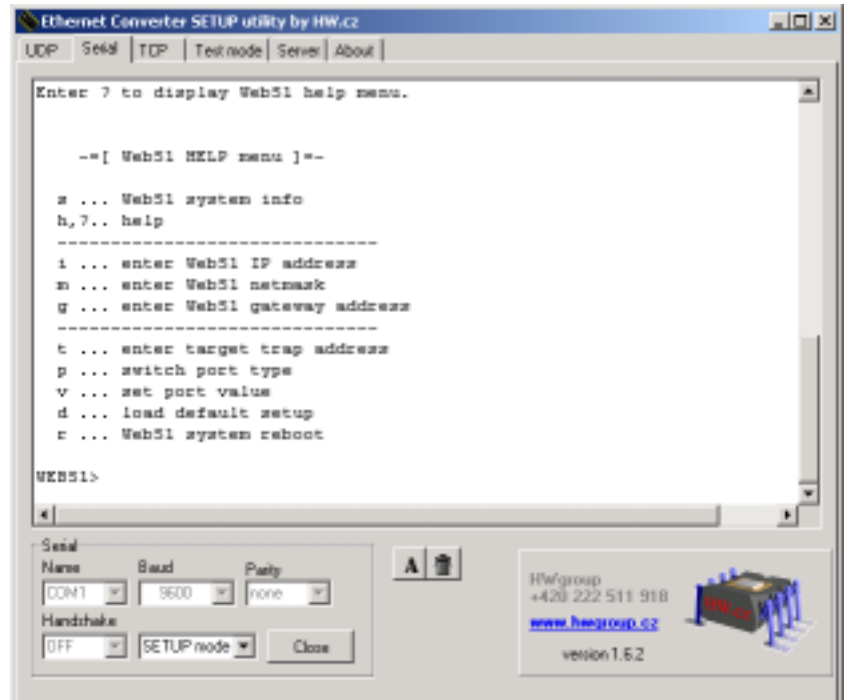
`p ... switch port type`

This command changes the port properties. **Direct** means the direct port access (you can have only one 8bit I/O port). The **shifted** property is used with the shift registers and 1-Wire thermometers. If you are using the **Charon I&II DB** do not forget to set the version.

`s ... Web51 system info`

This command displays the actual network parameters (MAC, IP, GW, MASK)

Once you have finished your settings, **disconnect the SETUP jumper** and reset the module. The message "EEPROM loading, preserving MAC, getting" will be displayed in the serial terminal window.



Conclusion

The module's firmware setup is now complete. It's time to connect the module to the Ethernet and try to communicate via SNMP. If you would like to make sure the module is communicating, try to Ping the module's IP address from your PC. Ping is available through command line or by simply typing it into the "Start"->"Run" box (i.e. ping -t 192.168.6.16). The "-t" means continuous ping until interrupted.

Installing and running the JAVA "Thermometer" Application

The next step is to install the application, which can communicate with the module via SNMP. You can use any SNMP client or browser. If you are beginner or you don't want to develop your own application, you should try our JAVA app. "**Thermometer**". This JAVA app. will show you all in graphics and you can use it to set up the module.

The basics about JAVA language

Java is becoming the most popular object programming language. Here are the basic terms.

- **JAVA SDK (Software Development Kit)**

The SDK contains a compiler to compile the .java source codes to the .class files and is all you need to develop your own JAVA applications. If you would like to use only the pre-programmed applications, it's not necessary to install the whole package.

- **JRE – JAVA runtime environment**

The basic environment that can run the JAVA code. You will need **JRE** to run all your JAVA applications. The most popular JRE is by SUN Microsystems (the JAVA developer) is **Java Web Start 1.2**. It's a kind of package of extensions, which can run the applications from one file, or directly from the web. It's a standard in the basic JRE.

- **.java**

A text file containing the JAVA source code.

- **.class**

A "byte code" = a universal executable code, which does not depend on the Operating System or the CPU. It is created by the .java file compilation.

- **.jar**

Several .class files in one file .jar. This .jar file is the entire application file.

You can download the last version of the JAVA **Thermometer** application from:

<http://www.dfsoft.cz/Charon/>

THERMOMETER
- Sample for Charon Board

Installation

1. **Install Java Web Start Application** (if not already installed - it comes bundled with SUN's Java Runtime Environment since JRE 1.2.2) Use most recent JRE version at top of page.
2. **Download management SW or Install it with Java Web Start or Run directly from this link**

Application Abstract

- it was developed to demonstrate ease of custom MIB implementation
- part of MIB-II is implemented
- asynchronous trap functionality
- application stores all parameters into a non-volatile memory
- some parameters are configurable during setup made from RSD32
- all of them can be edited/read through snmp
- CodeGen (source code skeleton generator/merger)
- maximally simplifies custom MIB implementation
- you will never believe that you can save so much time with this tool
- optimized for JSL environment (static pointers to reduce code size)
- template based (allows easy modification of generated code)
- code functional just after generation (integer values of 1 and empty string voids)

How to work with the Thermometer Application?

You have to install the JAVA SDK or JRE if you would like to work with a JAVA application. This is not a standard part of Windows, so you can download it from the web:

<http://java.sun.com/products/javawebstart/download-windows.html>

After that you have installed the JAVA support on your computer, you can just only click on the .jar file. You can run the application from the previous web page as well.

There is a file named "**charon_properties.txt**", which is used for setting up the module parameters. The file's structure is the following:

```
max=50
graph_time=1
min=-20
timeout=5000
IP=192.168.6.16
```

The Application uses a freeware Westhawk, which is a JAVA SNMP implementation for JAVA applications.

Controlling the Thermometer application

Before you run the JAVA application, set the Charon module's IP address in the text file.

Options

If you haven't set the IP address in the text file correctly, the application can't find your Charon module. You can set the IP address also in "Options → Host IP" menu, together with the SNMP timeout, after which the SW will output an SNMP communication error. You can also set the graph parameters in the "Options" menu.

Thermo

There are threshold limits set in the in the Thermo. If there is an out of range temperature detected, the module will send this information to the application without intervention. The Alarm's value is displayed on the thermometer and also in the graph. The alarms are displayed in the text window up to the graph.



Test the alarms – Set the upper temperature limit (the red sign), for instance, 31°C and keep the temperature sensor in your hand. The Temperature goes up and after the alarm value is excited, the Alarm will be displayed:

```
"04:35:07 [0] '31.13 °C' Temperature High from 192.168.6.16"
```

A note : *If your normal temperature is for example 32 – 36 °C you might try to set the lower limitation alarm for example 15°C and a piece of ice.. ☺.*

TEA

TEA is an algorithm, which is used for data encryption and authorization.

Ethernet

This menu is useful for remote access and remote module settings. You can change the module's IP address, Gateway, RS-232 parameters and so on. You can also change the module's receiving port (2000 by default), and an client IP address where the module sends the SNMP traps over UDP. This is usually the IP of the computer where JAVA app. or standard SNMP app is running.

The advanced property "**Allowed IP address range**" is an address range, which is defined by the IP address and Mask for the local communication resolution/gateway communication.

Serial

This menu is for setting up the RS-232 serial link on the module. The “**serial timeout**” is for setting up the packet making algorithm which packets the continuous data flow from the RS-232 link and sends the packets to the Ethernet network. The parameter defines the maximum space length after the last received character. After this time has expired, the module makes a packet and sends to the network. The timeout is entered in characters. There is no need to change this timeout, if you have changed the baud rate of the RS-232 serial link as the module adapts to the changed baud rate.

If the Baud rate is 9600 Bd, **serial timeout = 7** and there is a continuous stream of 15 incoming characters followed by a pause. The module will wait 7mS after the last character (7 characters @ 9600Bd) and then will send the whole packet of 15 characters to the Ethernet as an SNMP trap.

Note: All parameters, which are set remotely, are also stored to the internal EEPROM, so they are available and read by the application (using SNMP) after an applications restart.

Port

This menu sets the type of the 8bit I/O port's access.

- **Parallel**

I/O port P1 is used as a direct 8bit port. The port width is 8 pins I/O port P1 1..9, where 9th bit is an INT0 input (the second jumper next to the SETUP one). If you set the port width to 4, the upper 4 bits are used for the network indication as follows:

- P1.4 – **PING**
- P1.5 – **ARP**
- P1.6 – **SNMP**
- P1.7 – **UDP**

You cannot use the shift registers and 1-Wire bus in parallel settings.

- **Special**

The I/O ports are extended by the shift registers and other peripherals. You can find the detailed description in the Charon I&II DB datasheet.

- The 8bit I/O port is extendible to 32 bits of I/O
- You can connect up to 4 1-Wire thermometers to the P1.0 port

The described applications have no support for the LCD display connectivity or A/D converters, which are supported by the CharonI&II Development Board.

Using the application with other SNMP clients

If you would like to use any other SNMP Client application, you will need a MIB table for the Charon module. You can get this table in the original “**ANSI C - SNMP Development System**” directory or you can download it together with the .HEX file and with the **sThermometer.jar** application.

If you would like to test the firmware, you can use any SNMP Client. You have to load the .MIB file into the same directory as your SNMP client was installed to. Than you have to compile this table to the list of MIB tables in your Client. If the installation/compilation process succeeded, the SNMP client will find the Web51 device and display the tree with the module’s variables.

How to use the SNMP protocol?

You can use a development kit **ANSI C - SNMP Development System** for the x51 hardware (not only the Charon modules), which contains solved examples, an easy MIB table generator and so on. Detailed information are available on <http://Web51.HW.cz>.

Conclusion

The Web51 project and its numerous applications have resulted from several years of development. It is regarded as a very stable platform in the embedded systems world. The Web51-C version greatly enhances network management capabilities and opens the way for monitoring, controlling and configuring network device settings.

Related documentation

- **web51c\app\Readme.txt**
All the SNMP applications contain detailed readme file in English.
- **Charon I&II Development Board** – The DB scheme and device placements.
- **Charon I** – Documentation – Scheme, description of the Charon I module.
- **Project Web51** - <http://web51.hw.cz/>
- www.Hwgroup.cz – The final product and solution documentation.

Other Literature & interesting links

- [1] Charon I module - <http://www.hwgroup.cz/products/charon1/>
- [2] Project Web51 - <http://web51.hw.cz/>
- [3] Keil C51 – <http://www.keil.com/>
- [4] SDCC – <http://sdcc.sourceforge.net/>
- [5] RD2-Flasher - http://www.hw.cz/software/rd2_flasher/flasher22.html
- [6] Atmel FLIP – <http://www.atmel.com/>
- [7] Charon I - SNMP I/O Thermometer – <http://www.hwgroup.cz/products/charon1/snmp/>
- [8] LCC-WIN32 – <http://www.cs.virginia.edu/~lcc-win32/>
- All about SNMP - <http://www.snmplink.org/>
- <http://www.castlerock.com> - **SNMPc Network Manager**
- **SNMP Client**, which might be used as a 30 days trial version - <http://www.mg-soft.com>
- The server with an interesting SNMP idea - <http://www.pcmeasure.com>
- **CodeGen** - The code generator, parser and merger – a program, which makes the MIB analysis and which generates the kernel using a template. It helps to create a MIB table. - <http://www.dfsoft.cz/products.htm>



Contacts and detailed information

- Czech Republic: **HW group** www.HW-group.com
 Rumunská 26/122, Praha 2, 120 00,
 Phone +420 222 511 918, Fax. +420 222 513 833
- Germany: **egnite Software GmbH** www.egnite.de
 Westring 303, 44629 Herne, Germany
 Phone +49 23 23-92 53 75, Fax. +49 23 23-92 53 74
- United Kingdom: **TR Control Solutions** www.TRcontrolsolutions.com
 Global House, Ashley Avenue, Epsom, Surrey, KT18 5AD UK
 Phone: +44 208 823 9230, Fax: +44 208 823 9240
- USA: **Capitol Automation** www.CapitolAutomation.com
 500 Main Street, Clinton, Ma. 01510, U.S.A.
 Phone: +1 800 550 9672