

phyCORE-P87C591

QuickStart Instructions

**Using PHYTEC FlashTools98 for Windows and the Raisonance
Integrated Development Environment (RIDE)
for 8051 and XA Demo Version**

Note: The PHYTEC Spectrum CD includes the electronic version of
the English phyCORE-P8xC591 Hardware Manual

Hinweis: Die PHYTEC Spektrum CD beinhaltet die elektronische
Version des deutschen phyCORE-P8xC591 Hardware Manuals

Edition: July 2002

A product of a PHYTEC Technology Holding company

In this manual are descriptions for copyrighted products that are not explicitly indicated as such. The absence of the trademark (™) and copyright (©) symbols does not imply that a product is not protected. Additionally, registered patents and trademarks are similarly not expressly indicated in this manual.

The information in this document has been carefully checked and is believed to be entirely reliable. However, PHYTEC Meßtechnik GmbH assumes no responsibility for any inaccuracies. PHYTEC Meßtechnik GmbH neither gives any guarantee nor accepts any liability whatsoever for consequential damages resulting from the use of this manual or its associated product. PHYTEC Meßtechnik GmbH reserves the right to alter the information contained herein without prior notification and accepts no responsibility for any damages which might result.

Additionally, PHYTEC Meßtechnik GmbH offers no guarantee nor accepts any liability for damages arising from the improper usage or improper installation of the hardware or software. PHYTEC Meßtechnik GmbH further reserves the right to alter the layout and/or design of the hardware without prior notification and accepts no liability for doing so.

© Copyright 2002 PHYTEC Meßtechnik GmbH, D-55129 Mainz.

Rights - including those of translation, reprint, broadcast, photomechanical or similar reproduction and storage or processing in computer systems, in whole or in part - are reserved. No reproduction may occur without the express written consent from PHYTEC Meßtechnik GmbH.

	EUROPE	NORTH AMERICA
Address:	PHYTEC Technologie Holding AG Robert-Koch-Str. 39 D-55129 Mainz GERMANY	PHYTEC America LLC 203 Parfitt Way SW, Suite G100 Bainbridge Island, WA 98110 USA
Ordering Information:	+49 (800) 0749832 order@phytec.de	1 (800) 278-9913 info@phytec.com
Technical Support:	+49 (6131) 9221-31 support@phytec.de	1 (800) 278-9913 support@phytec.com
Fax:	+49 (6131) 9221-33	1 (206) 780-9135
Web Site:	http://www.phytec.de	http://www.phytec.com

2nd Edition: July 2002

1	Introduction to the Rapid Development Kit	1
1.1	Rapid Development Kit Documentation	1
1.2	Overview of this QuickStart Instruction.....	2
1.3	System Requirements	3
1.4	The PHYTEC phyCORE-P87C591	4
1.5	The Raisonance Integrated Development Environment (RIDE) for 51+XA.....	7
2	Getting Started.....	11
2.1	Installing Rapid Development Kit Software.....	11
2.2	Interfacing the phyCORE-P87C591 to a Host-PC	18
2.3	Starting PHYTEC FlashTools98 for Windows	20
2.4	Downloading Example Code with FlashTools	21
2.4.1	"Blinky"	25
2.4.2	"Hello"	27
3	Getting More Involved	33
3.1	Starting the Raisonance Tool Chain	33
3.2	Creating a New Project and Adding an Existing Source File.....	34
3.3	Modifying the Source Code.....	38
3.4	Saving the Modifications.....	38
3.5	Setting Tool Chain Options	39
3.6	Building the Project	42
3.7	Downloading the Output File	43
3.8	"Hello2"	44
3.8.1	Creating a New Project.....	44
3.8.2	Modifying the Example Source	45
3.8.3	Setting Tool Chain Options	45
3.8.4	Building the New Project	45
3.8.5	Downloading the Output File	46
3.8.6	Starting the Terminal Emulation Program.....	47

4	Debugging	49
4.1	Preparing the Target Hardware to Communicate with ROM Monitor.....	50
4.2	Creating a Debug Project and Preparing the Debugger	51
4.2.1	Creating a New Project	51
4.2.2	Setting Options for Target.....	52
4.3	Preparing the Debugger.....	59
4.4	Starting the Debugger.....	61
4.5	Raisonance Debug Features	64
4.6	Using the Raisonance Debug Features	65
4.6.1	Watch Window.....	65
4.6.2	Run to	66
4.6.3	Step Into and Step Over	68
4.6.4	Breakpoints.....	69
4.7	Running, Stopping and Resetting	72
4.8	Changing Target Settings for the "Final Version"	73
5	Advanced User Information.....	75
5.1	FlashTools98	75
5.2	Linking and Locating	77

Index of Figures

Figure 1:	Mounting the phyCORE-P87C591 onto the phyCORE Development Board LD 5V	18
Figure 2:	Important Connectors, Buttons and Suitable Jumper Settings on the phyCORE Development Board LD 5V	19
Figure 3:	Power Connector	19
Figure 4:	Memory Model for Use with the Raisonance Monitor (64 kByte RAM).....	53

1 Introduction to the Rapid Development Kit

This QuickStart provides:

- general information on the PHYTEC phyCORE-P87C591 Single Board Computer (SBC)
- an overview of Raisonance's Integrated Development Environment (RIDE) for 51+XA, and
- instructions on how to run example programs on the phyCORE-P87C591, mounted on the PHYTEC phyCORE Development Board LD 5V, in conjunction with RIDE for 51+XA

Please refer to the [phyCORE-P8xC591 Hardware Manual](#) for specific information on such board-level features as [jumper configuration](#), [memory mapping](#) and [pin layout](#). Selecting the links on the electronic version of this document links to the applicable section of the phyCORE-P8xC591 Hardware Manual.

1.1 Rapid Development Kit Documentation

This “Rapid Development Kit” (RDK) includes the following electronic documentation on the enclosed “PHYTEC Spectrum CD-ROM”:

- the PHYTEC [phyCORE-P8xC591 Hardware Manual](#) and [phyCORE Development Board LD 5V Hardware Manual](#)
- controller [User's Manuals and Data Sheets](#)
- this QuickStart Instruction with general “Rapid Development Kit” description, software installation hints and three example programs enabling quick out-of-the box start-up of the phyCORE-P87C591 in conjunction with the Raisonance Integrated Development Environment (RIDE) for 51+XA

1.2 Overview of this QuickStart Instruction

This QuickStart Instruction gives a general “Rapid Development Kit” description, as well as software installation hints and three example programs enabling quick out-of-the box start-up of the phyCORE-P87C591 in conjunction with the Raisonance Integrated Development Environment (RIDE) for 51+XA. It is structured as follows:

- 1) The *"Getting Started"* section uses two example programs: *"Hello"* and *"Blinky"* to demonstrate the download of user code to the Flash device using PHYTEC FlashTools98 for Windows.
- 2) The *"Getting More Involved"* section provides step-by-step instructions on how to modify both examples, create and build new projects and generate and download output files to the phyCORE-P87C591 using the Raisonance tool chain and FlashTools98.
- 3) The *"Debugging"* section provides a third example program - “Debug” - to demonstrate monitoring of the board and simple debug functions using the Raisonance RIDE debug environment.

In addition to dedicated data for this Rapid Development Kit, this CD-ROM contains supplemental information on embedded microcontroller design and development.

1.3 System Requirements

Use of this “Rapid Development Kit” requires:

- the phyCORE-P87C591 SBC module
- the phyCORE Development Board LD 5V with the included DB-9 serial cable and AC adapter supplying 5 VDC /min. 500 mA
- the PHYTEC Spectrum CD
- an IBM-compatible host-PC (486 or higher running at least Windows95/98)

For more information and example updates, please refer to the following sources:

PHYTEC

<http://www.phytec.com> - or - <http://www.phytec.de>
support@phytec.com - or - support@phytec.de

RAISONANCE

<http://www.raisonance.com> - or - <http://www.amrai.com> (US)
support@raisonance.com - or - support@amrai.com (US)
support@raisonance.fr (Europe)

1.4 The PHYTEC phyCORE-P87C591

The phyCORE-P87C591 represents an affordable, yet highly functional Single Board Computer (SBC) solution in subminiature dimensions (40 x 55 mm). The standard board is populated with a Philips P87C591 controller, featuring a 6-channel on-chip A/D-converter with 10-bit resolution and an integrated CAN controller.

All applicable data/address lines and applicable signals extend from the underlying logic devices to standard-width (2.54 mm /0.10 in.) pin headers lining the circuit board edges. This enables the phyCORE-P87C591 to be plugged like a “big chip” into target hardware.

The standard memory configuration of the phyCORE-P87C591 features 128 kByte external SRAM and 128 kByte external Flash for code storage (64 kByte for FlashTools firmware and 64 kByte for storage of user code). The Flash device allows direct on-board programming. Three Chip Select signals are available for external I/O connectivity.

The module communicates by means of an RS-232 transceiver and operates within a standard industrial range of 0 to +70 degrees C. It requires only a 250 mA power source.

PHYTEC FlashTools98 enables easy on-board download of user programs.

phyCORE-P87C591 Technical Highlights

- SBC in subminiature dimensions (40 x 55 mm) achieved through advanced SMD technology
- populated with an 44-pin packaged (PLCC) Philips 8051-compatible P87C591 controller featuring 2.0B on-chip CAN with extended Philips PeliCAN
- instruction cycle time of 375 ns at 16 MHz clock speed (no internal clock prescaler)
- 128 kByte external SRAM
- 128 (to 512) kByte external Flash supporting on-board downloading of user code from a host-PC in conjunction with PHYTEC FlashTools98 firmware
- RS-232 serial interface
- 2.0B CAN bus interface supporting 11-bit and 29-bit message identifiers
- 6-channel on-chip A/D converter with 10-bit resolution
- three Chip Select signals for connection to external peripherals
- requires only a +5 V/250 mA power source
- operates in a temperature range of 0... 70°C (optional -40... 85°C temperature range available)

The phyCORE Development Board LD 5V, in EURO-card dimensions (160 x 100 mm), is fully equipped with all mechanical and electrical components necessary for the speedy and secure insertion, and subsequent programming, of PHYTEC phyCORE series Single Board Computers with standard width (2.54 mm/ 0.10 in.) pin header connectors. Simple jumper configuration readies the Development Board's connection to any phyCORE module (standard header pins), which plug pins-down into the contact strips mounted on the phyCORE Development Board LD 5V.

phyCORE Development Board LD 5V Technical Highlights

- Reset signal controlled by push button or RS-232 control line CTS0
- Boot signal controlled by push button or RS-232 control line DSR0
- low voltage socket for supply with regulated input voltage 5 VDC
- additional supply voltage 3.3 VDC
- two DB-9 sockets (P1A, P1B) configurable as RS-232 interfaces
- two additional DB-9 plugs (P2A, P2B) configurable as CAN interfaces, connector P2B optionally configurable as RS-485 interface
- simple jumper configuration allowing use of the phyCORE Development Board LD 5V with various PHYTEC phyCORE SBC's
- one control LED D3 for quick testing of user software
- 2 x 160-pin Molex connector (X2) enabling easy connectivity to expansion boards (e.g. PHYTEC GPIO Expansion Board)

1.5 The Raisonance Integrated Development Environment (RIDE) for 51+XA

The Raisonance tool chain fully supports the entire Philips XA and 8051 derivative microcontroller families. It includes an ANSI-C compiler, macroassembler, linker/locator, simulator/debugger and ROM monitor/debugger within the RIDE development environment.

From the unique RIDE user interface projects can be developed for an application based either on an 8051 derivative, a XA derivative, or both. Such flexibility makes migrating from 8-bit to 16-bit architectures easier.

The Raisonance tool chain produces OMF object files that are supported by most in-circuit emulators. The OMF-to-HEX utility converts a Raisonance .aof format file into an Intel hexfile that is suitable for programming into the external Flash on the PHYTEC phyCORE-P87C591 target board.

The Raisonance tool chain consists of the following tools; all integrated in RIDE:

- **C Compiler**
- **Assembler**
- **RTOS**
- **Linker/Locator**
- **Simulator/Debugger**
- **ROM Monitor**

All these tools and many utility programs are available in two versions: as Win32 DLL called internally from RIDE, and DOS-based executables (*.exe) that can be called externally. These two versions are strictly equivalent regarding the generated code.

The evaluation version of the RIDE development environment is limited in manipulable code size as follows:

- 4 kByte for the 8051.
- 8 kByte for the XA.

Other than these restrictions, the evaluation tool chain functions exactly as the full version does, enabling full evaluation of the features and functionality of Raisonance development tools. This demo version can be upgraded by entering a serial number.

Raisonance Integrated Development Environment (RIDE)

RIDE is a Windows-based Graphical User Interface for all Raisonance tools. All compiler, assembler, linker/locator and debugger options are configured with simple mouse clicks. RIDE runs under Windows 95/98/2000 and NT.

All RIDE commands and functions are accessible via intuitive pull-down menus with prompted selections. An extensive help utility and complete set of online manuals are included. External executables can be run from within RIDE, including emulator software.

RC51 C Compiler

The RC51 compiler and MA51 assembler are designed specifically for 8051 controllers.

The Raisonance RC51 compiler provides the fastest and smallest code using industry benchmarks.

The Getting Started Guide from Raisonance provides more information on these tools. It also includes an introduction on 'Migrating from the 8051 to the XA'.

Debug Environment

RIDE includes a simulator/debugger and a ROM Monitor that supports debugging either via software on a host-PC or in target hardware. All the debugging functions are enabled in the demo version with the same restrictions in manipulable code size as follows:

- 4 kByte for the 8051,
- 8 kByte for the XA (Page Zero mode only).

2 Getting Started

What you will learn with this Getting Started example:

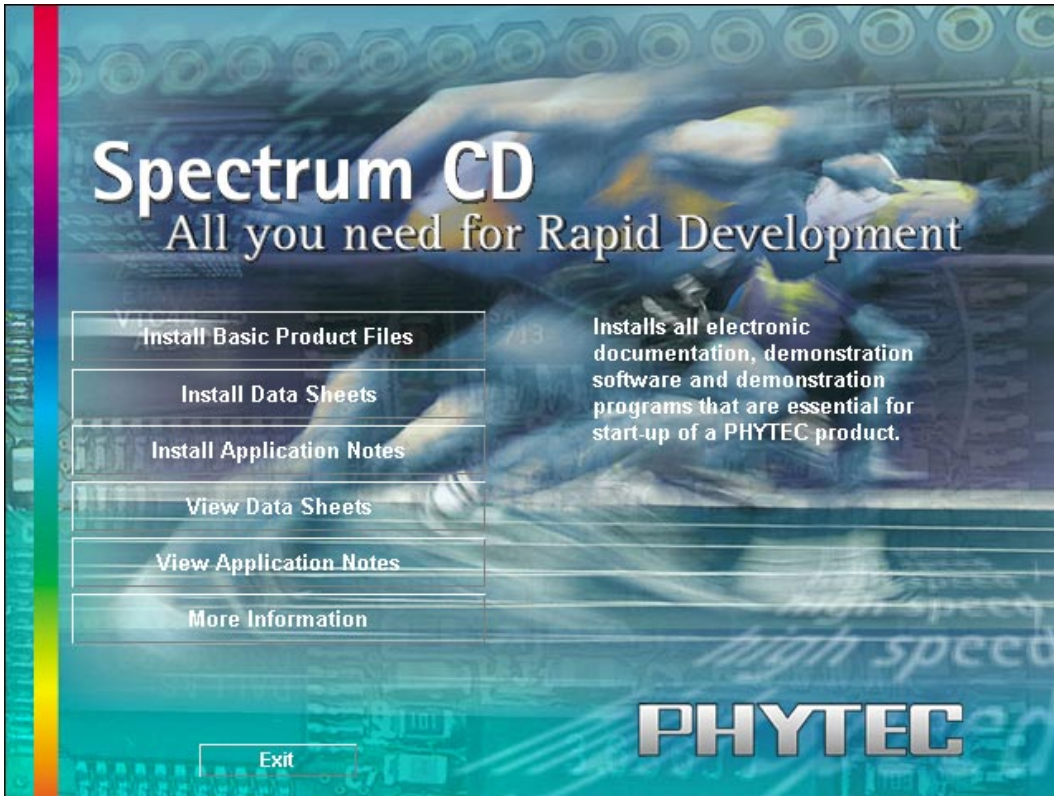
- installing Rapid Development Kit software
- starting PHYTEC FlashTools98 for Windows download utility
- interfacing the phyCORE-P87C591, mounted on the phyCORE Development Board LD 5V, to a host-PC
- downloading example user code in Intel hexfile format from a host-PC to the external Flash memory using FlashTools98

2.1 Installing Rapid Development Kit Software

- Insert the PHYTEC Spectrum CD into the CD-ROM drive of your host-PC.

The PHYTEC Spectrum CD should automatically launch a setup program that installs the software required for the Rapid Development Kit as specified by the user. Otherwise the setup program *start.exe* can be manually executed from the root directory of the PHYTEC Spectrum CD.

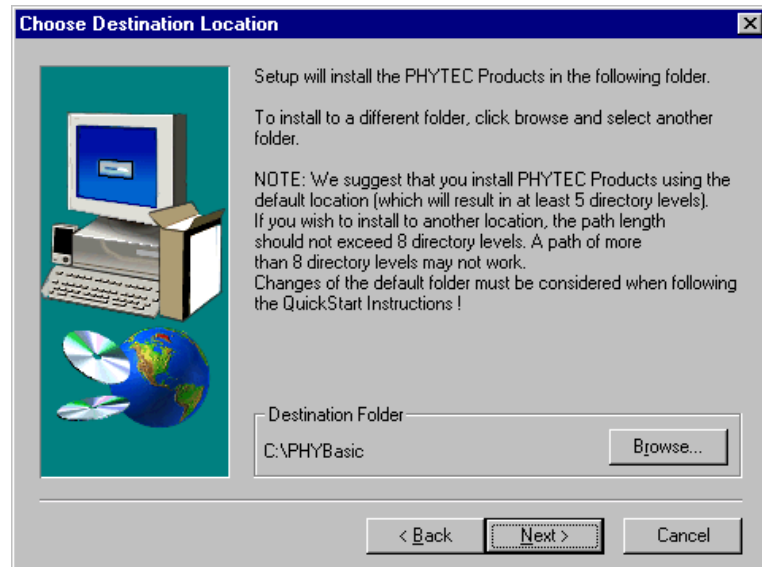
The following window appears:



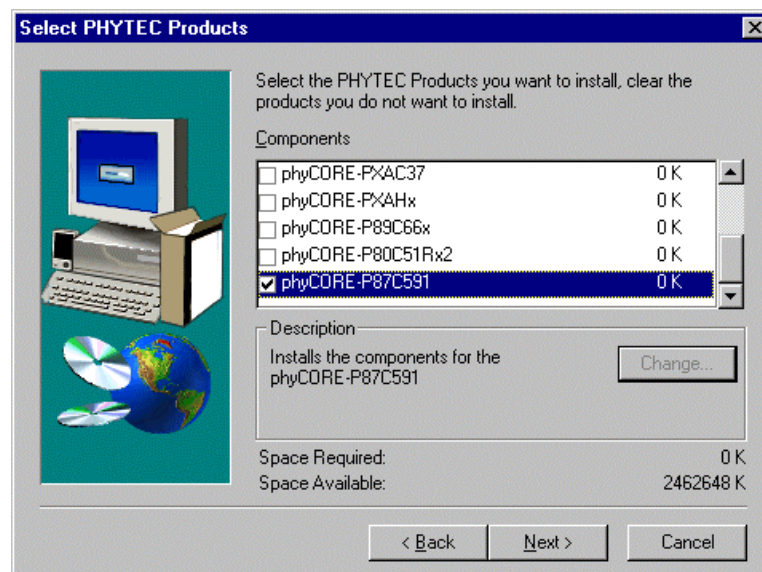
- Choose the *Install Basic Product Files* button.
- After accepting the Welcome window and license agreement, select the destination location for installation of Rapid Development Kit software and documentation.

The default destination location is **C:\PHYBasic**. All path and file statements within this QuickStart Instruction are based on the assumption that you accept the default install paths and drives. If you decide to individually choose different paths and/or drives you must consider this for all further file and path statements.

We recommend that you accept the default destination location.



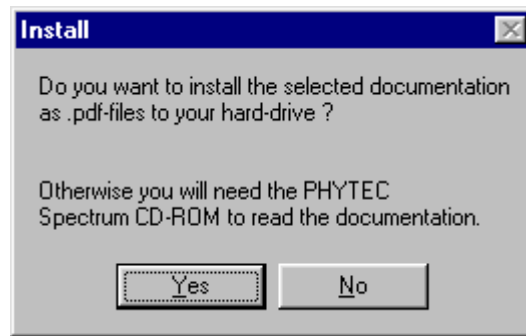
- In the next window, select your Rapid Development Kit of choice from the list of available products. By using the *Change* button, advanced users can select in detail which options should be installed for a specific product.



All Kit-specific content will be installed to a Kit-specific subfolder of the Rapid Development Kit root folder that you have specified at the beginning of the installation process.

All software and tools for this phyCORE-P87C591 RDK will be installed to the **|PHYBasic** folder on your hard drive.

- In the next dialog you must choose whether to copy the selected documentation as ***.pdf** files to your hard drive or to install a link to the file on the Spectrum CD.



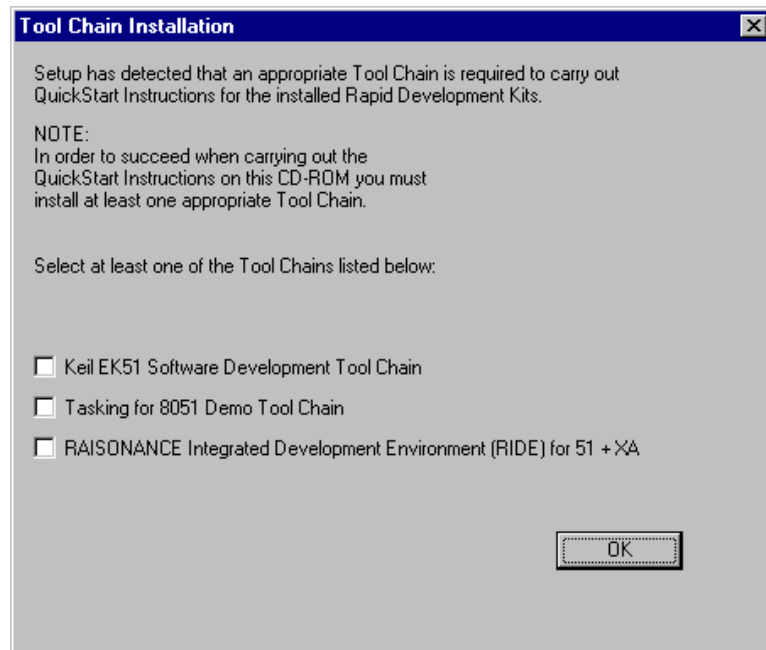
If you decide **not** to copy the documentation to your hard drive, you will need the PHYTEC Spectrum CD-ROM each time you want to access these documents. The installed links will refer to your CD-ROM drive in this case.

If you decide to copy the electronic documentation to your hard drive, the documentation for this phyCORE-P87C591 RDK will also be installed to the kit-specific subfolder. The manuals of the phyCORE Development Board LD 5V are copied to their own specific subfolder (e.g. **|PHYBasic|DevBLD5V**) because each Development Board is suitable for multiple SBC's and is not dedicated to a specific RDK.

Setup will now add program icons to the program folder, named **PHYTEC**.

- Click on *Finish* to complete the installation of PHYTEC products.

- In the next window, choose to install the Raisonance Software Development tool chain for 51 + XA¹.



The applicable Raisonance tool chain must be installed to ensure successful completion of this QuickStart Instruction. Failure to install the proper software could lead to possible version conflicts, resulting in functional problems.

We recommend that you install the Raisonance tool chain from the Spectrum CD-ROM even if other versions of RIDE is already installed on your system. These QuickStart Instructions and the demo software included on the CD-ROM have been specifically tailored for use with one another.

- After accepting the Welcome window and license agreement, select the destination location for installation of the Raisonance tool chain. The default location is **C:\Ride**.

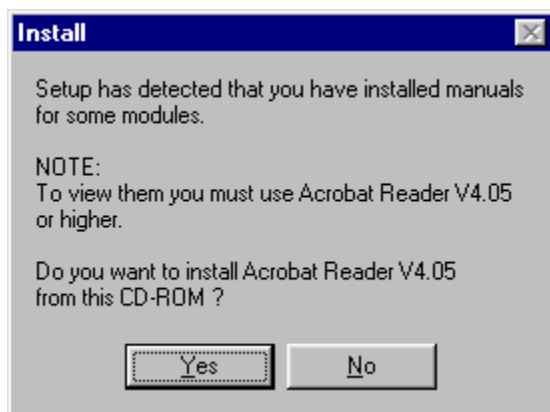
¹: If installing a different software development tool chain, *please refer to the applicable version of the QuickStart manual*.

The applicable Raisonance Software Development tool chain for 51+XA will be installed to your hard drive. Additional software, such as Adobe Acrobat Reader, will also be offered for installation.

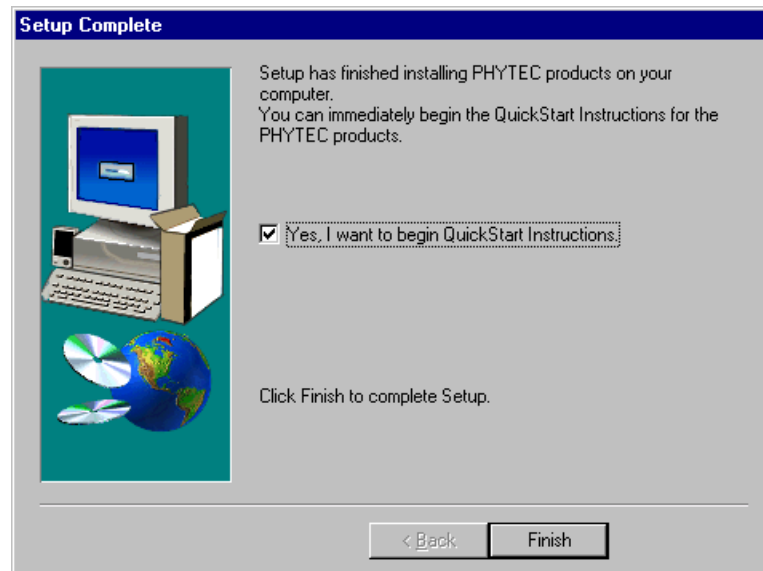
In the following windows you can decide to install FlashTools98 software and the Acrobat Reader.



The applicable FlashTools software must be installed to ensure successful completion of this QuickStart Instruction. Failure to install the proper software could lead to possible version conflicts, resulting in functional problems.



- Decide if you want to begin the QuickStart Instruction immediately by selecting the appropriate checkbox and click on *Finish* to complete the installation.



2.2 Interfacing the phyCORE-P87C591 to a Host-PC

Connecting the phyCORE-P87C591, mounted on the phyCORE Development Board LD 5V, to your computer is simple:

- As shown in the figure below, if the phyCORE module is not already preinstalled, mount it pins-down onto the Development Board's receptacle footprint (X6).
- Ensure that pin 1 of module (denoted by the hash stencil mark on the PCB) matches pin 1 of the receptacle on the phyCORE Development Board LD 5V.
- Ensure that there is a solid connection between the module pins and the phyCORE Development Board LD 5V receptacle.

Caution:

Take precautions not to bend the pins when the phyCORE module is removed from and inserted onto the phyCORE Development Board LD 5V.

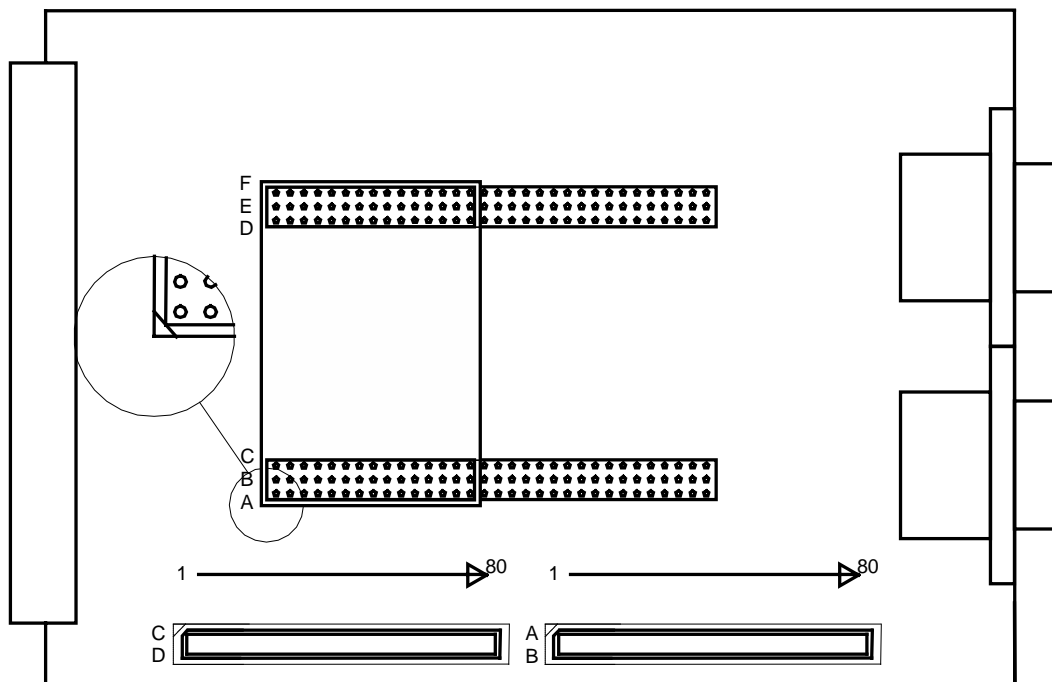


Figure 1: Mounting the phyCORE-P87C591 onto the phyCORE Development Board LD 5V

- Configure the jumpers on the phyCORE Development Board LD 5V as indicated below. This correctly routes the RS-232 signals to the DB-9 connector (P1A = bottom) and connects the Development Board's peripheral devices to the phyCORE module.

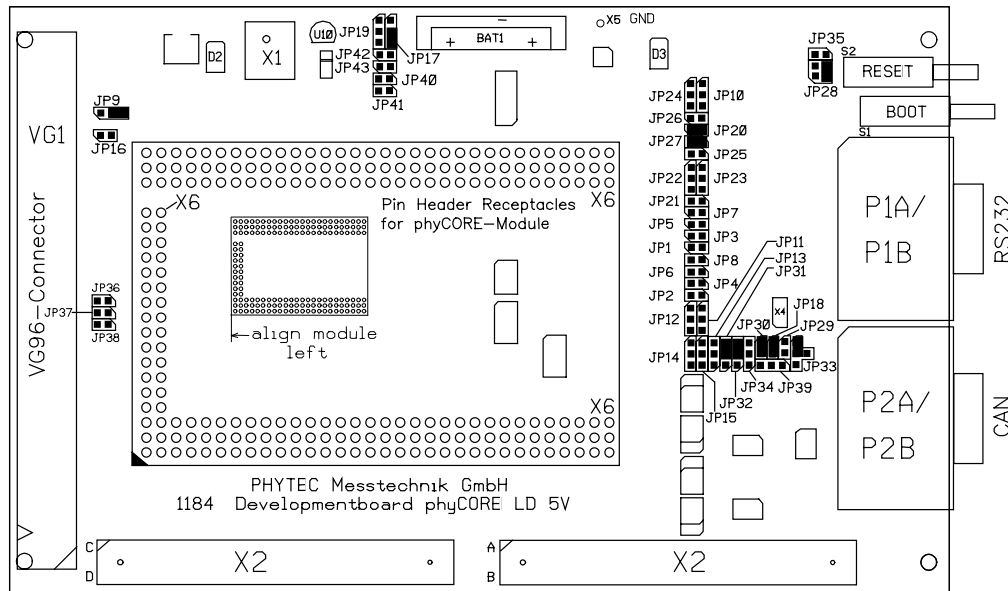


Figure 2: Important Connectors, Buttons and Suitable Jumper Settings on the phyCORE Development Board LD 5V

- Connect the RS-232 interface of your computer to the DB-9 RS-232 interface on the phyCORE Development Board LD 5V (P1A = bottom) using the included serial cable.
- Using the included power adapter, connect the power socket on the board (X1) to a power supply (*refer to Figure 3 for the correct polarity*).

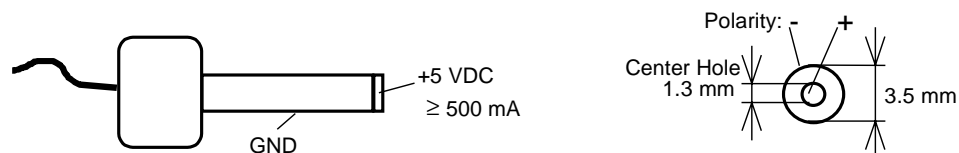


Figure 3: Power Connector

- Simultaneously press the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board LD 5V, first releasing the Reset and then, two or three seconds later, release the Boot button.

This sequence of pressing and releasing the Reset (S2) and Boot (S1) button renders the phyCORE-P87C591 into the Flash programming mode (FPM). Use of FlashTools98 always requires the phyCORE-P87C591 to be in FPM. See *section 2.4, “Downloading Example Code with FlashTools”* for more details.

The phyCORE module should now be properly connected via the phyCORE Development Board LD 5V to a host-PC and power supply. After executing a Reset and rendering the board in Flash programming mode, you are now ready to program the phyCORE-P87C591. This phyCORE module/phyCORE Development Board LD 5V combination is also referred to as “target hardware”.

2.3 Starting PHYTEC FlashTools98 for Windows

FlashTools98 should have been installed during the initial setup procedure as described in *section 2.1*. If not, you can manually install it using the *setup.exe* file located in the `\Software\Flasht98\` folder of your PHYTEC Spectrum CD.

FlashTools98 for Windows is a utility program that allows download of user code in Intel **.hex* file format from a host-PC to a PHYTEC SBC via an RS-232 connection.

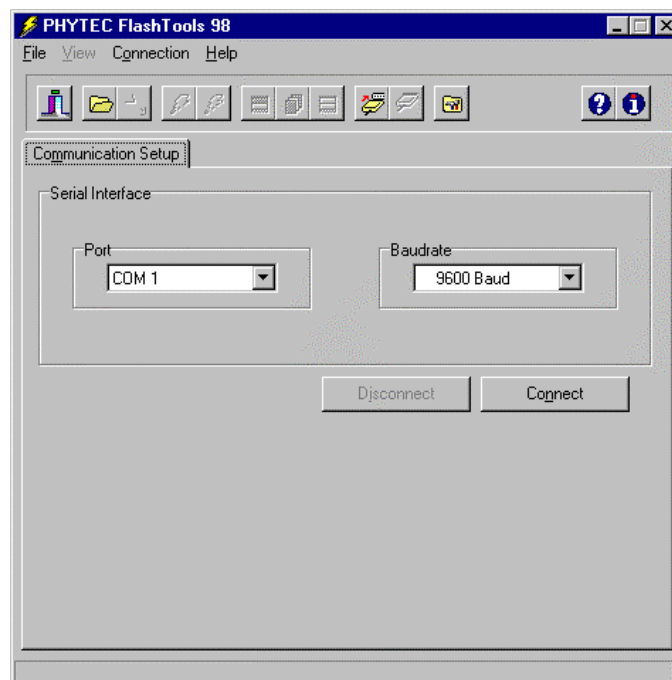
FlashTools98 consists of firmware resident in the external Flash and corresponding software installed on the host-PC. Proper connection of a PHYTEC SBC to a host-PC enables the software portion of FlashTools98 to recognize and communicate to the firmware portion.

- You can start FlashTools98 by selecting it from the *Programs* menu using the Windows *Start* button.

It is recommended that you drag the FlashTools98 icon onto the desktop of your PC. This enables easy start of FlashTools98 by double-clicking on the icon.

2.4 Downloading Example Code with FlashTools

- Start FlashTools98 for Windows by double-clicking on the FlashTools98 icon or by selecting *FlashTools98* from within the *Programs/Phytec* program group.
- The Communication Setup tab of the FlashTools98 tabsheet window will now appear. Here you can specify connection properties to the phyCORE-P87C591.



- Choose the correct serial port for your host-PC and a 9,600 baud rate.

Note:

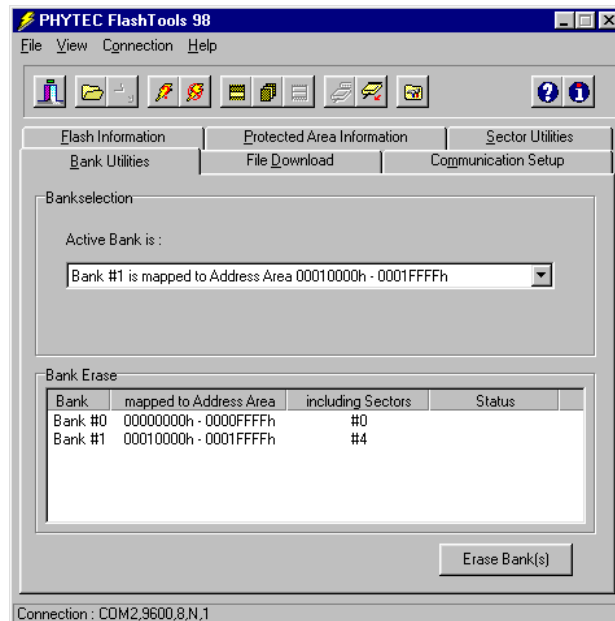
Always ensure that the phyCORE-P87C591 is in Flash programming mode before pressing the *Connect* button.

- Click the *Connect* button to establish connection to the target hardware.

The microcontroller firmware tries to automatically adjust to the baud rate selected within the baud rate tab. However, it may occur that the selected baud rate can not be attained. This results in a connection error. In this case, try other baud rates to establish a connection. Before attempting each connection, be sure to reset the target hardware and render it into Flash programming mode (FPM) as described in *section 2.2*.

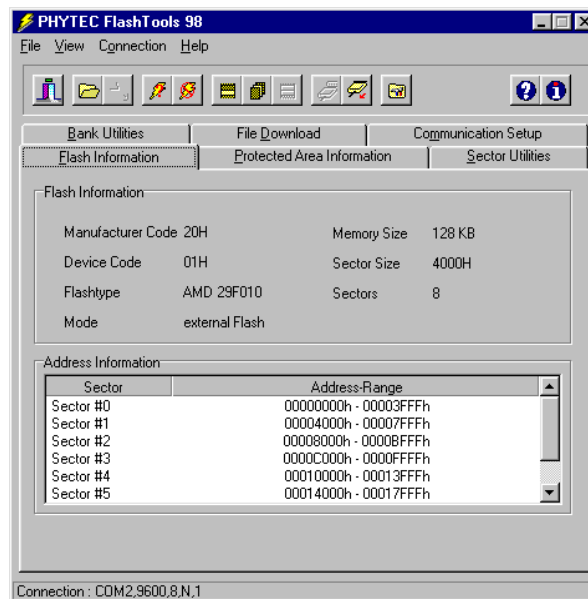
Returning to the FlashTool98 tabsheet window, you will see tabs for the following:

*Bank Utilities*² enable erasure and status check of whole banks of memory specified by the user:

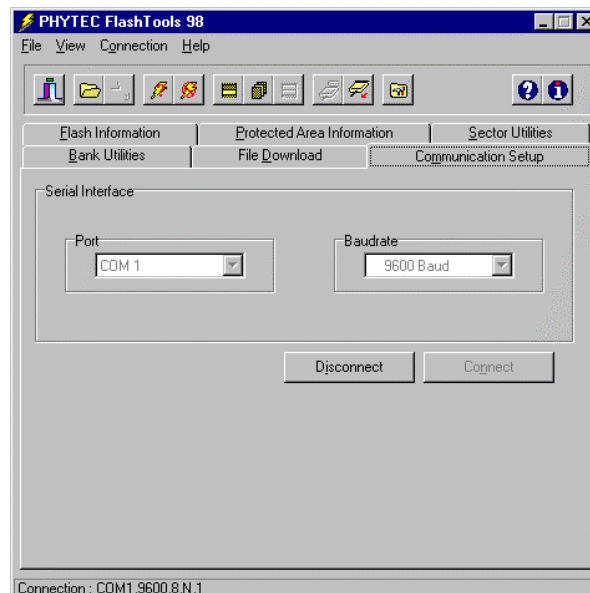


²: The number of banks shown on the *Bank Utilities* tabsheet varies depending on the size and type of the Flash mounted on the phyCORE-P87C591.

*Flash Information*³ shows Flash type, sector and address ranges in Flash memory:

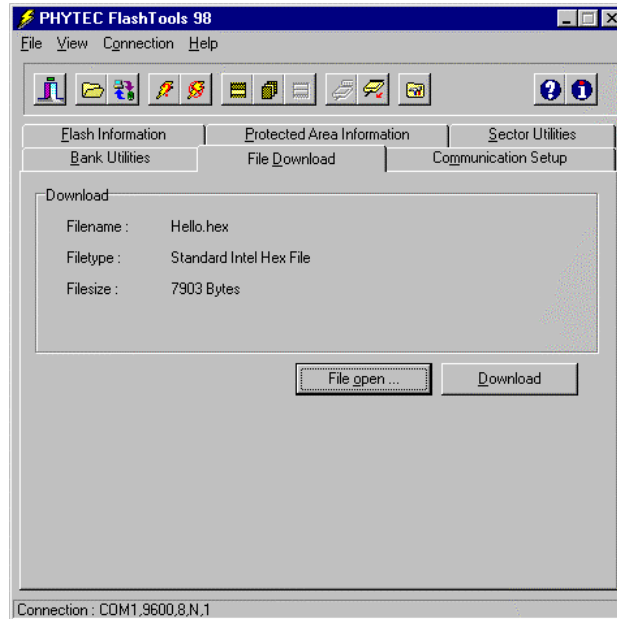


Communication Setup allows selection of the serial port and speed before the communication is initialized, or to disconnect the ongoing communication:

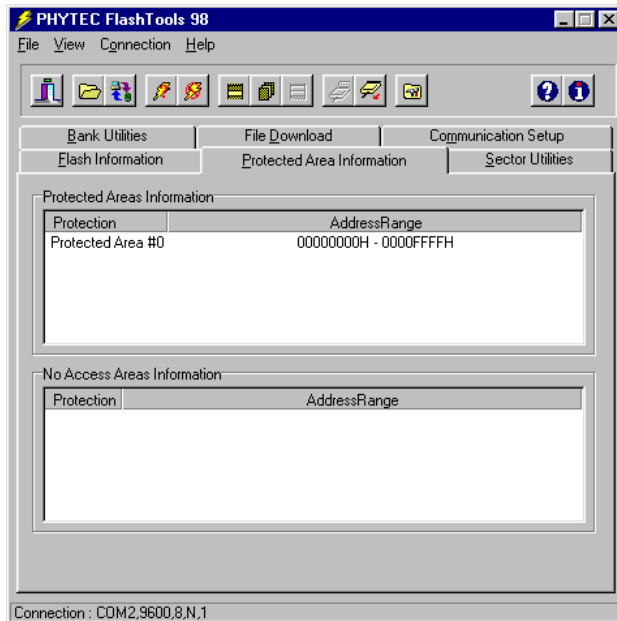


³: The appearance of the *Flash Information* tabsheet varies depending on the size and type of the Flash mounted on the phyCORE-P87C591.

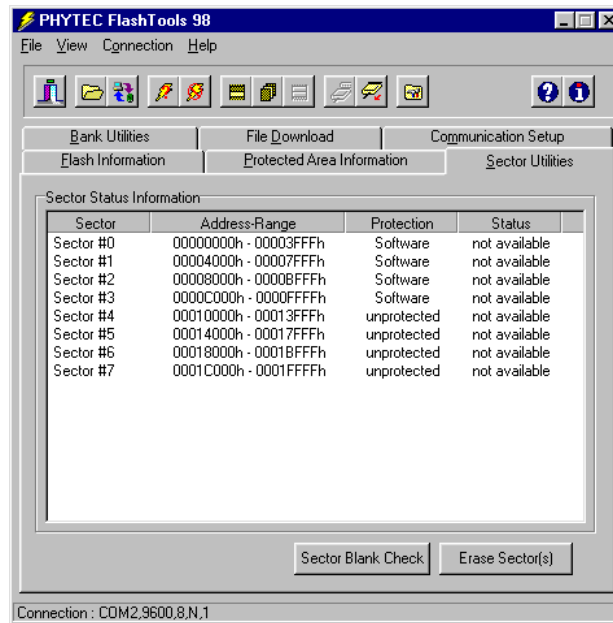
File Download downloads specified hexfiles to the target hardware:



Protected Areas Information shows protected areas of Flash memory:



*Sector Utilities*⁴ enable erasure and status check of individual sectors of Flash memory specified by the user:



2.4.1 "Blinky"

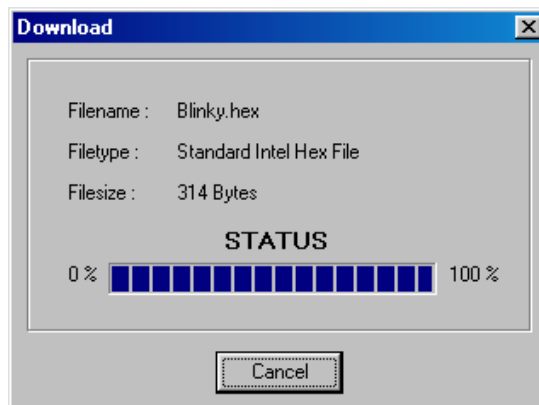
The “Blinky” example downloads a program to the Flash that, when executed, manipulates the LED D3 on the phyCORE Development Board LD 5V that is located above the jumper field (refer to Figure 2).

- Returning to the FlashTools98 tabsheet, choose the *Bank Utilities* tab, highlight *Bank #1* within the *Bank Erase* section, and click on the *Erase Bank(s)* button to erase this memory bank.
- Wait until the status check in the lower left corner of the FlashTools98 tabsheet finishes, returning the connection properties description to the lower left corner of the window.
- Next choose the *File Download* tab and click on the *File Open* button.

⁴: The appearance of the *Sector Utilities* tabsheet varies depending on the size and type of the Flash mounted on the phyCORE-P87C591.

The hexfile has already been installed to your hard drive during the installation procedure.

- Browse to the correct drive and path for the phyCORE-P87C591 Demo folder (default location **C:\PHYBasic\pC-P87C591\Demos\Raisonance\Blinky\Blinky.hex**) and click *Open*.
- Click on the *Download* button. You can watch the status of the download of the **Blinky.hex** into the external Flash memory in the Download window.



If the selected Flash bank into which you wish to download code is not empty (i.e. erased), a warning dialog box will appear, indicating “Location not empty! Please erase location and try again”. In this event, select the *Bank Utilities* tab from the FlashTools98 tabsheet, highlight *Bank #1* and erase the bank. Then repeat the download procedure.

- At the end of the download, a sector-by-sector status check of the Flash memory can be viewed in the lower left corner of the FlashTools98 tabsheet window. Wait until the status check finishes before returning to work with the board. Once the status check is complete, the downloaded code can be executed.
- Returning to the *Communication* tab, click on the *Disconnect* button and exit FlashTools98.

- Press the Reset button (S2) on the phyCORE Development Board LD 5V to reset the target hardware and to start execution of the downloaded software.
- Successful execution of the program will flash the LED D3 with equal on and off durations.

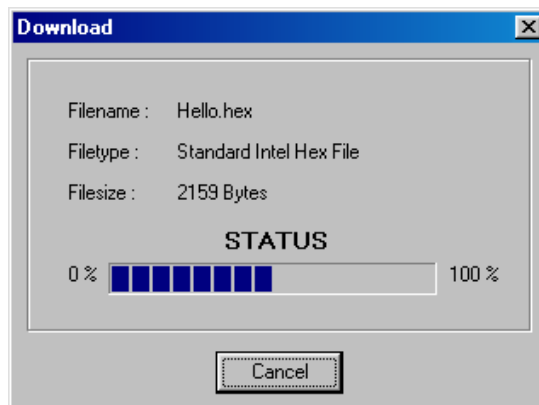
2.4.2 "Hello"

The “Hello” example downloads a program to the Flash that, when executed, performs an automatic baud rate detection and sends a character string from the target hardware back to the host-PC. The character string can be viewed with a terminal emulation program. This example program provides a review of the FlashTools98 download procedure. For detailed commentary on each step, described below in concise form, *refer back to sections 2.2 through 2.4.1.*

- Ensure that the target hardware is properly connected to the host-PC and a power supply.
- Reset the target hardware and force it into Flash programming mode by simultaneously pressing the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board LD 5V and then releasing first the Reset and, two or three seconds later, the Boot button.
- Start FlashTools98.
- At the *Communication Setup* tab of the FlashTools98 tabsheet, specify the proper serial port and transmission speed (9,600 baud) for communication between host-PC and target hardware and click the *Connect* button to establish connection to the target hardware.
- Returning to the FlashTools98 tabsheet, choose the *Bank Utilities* tab, highlight *Bank #1* within the *Bank Erase* section, and click on the *Erase Bank(s)* button to erase this memory bank.
- Wait until the status check in the lower left corner of the FlashTools98 tabsheet finishes, returning the connection properties description to the lower left corner of the window.
- Next choose the *File Download* tab and click on the *File Open* button.

The demo hexfile has already been installed to your hard drive during the installation procedure.

- Browse to the correct drive and path for the phyCORE-P87C591 Demo folder (default location **C:\PHYBasic\pC-P87C591\Demos\Raisonance\Hello\Hello.hex**) and click *Open*.
- Click on the *Download* button. You can watch the status of the download of the **Hello.hex** into the external Flash memory in the Download window.

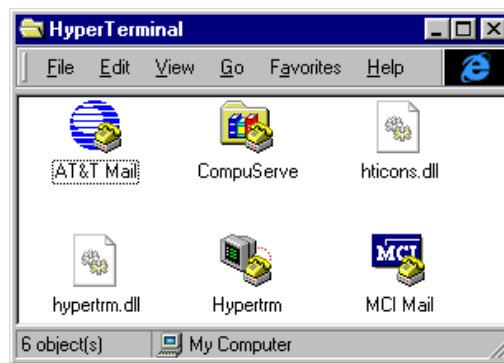


If the selected Flash bank into which you wish to download code is not empty (i.e. erased), a warning dialog box will appear, indicating “Location not empty! Please erase location and try again”. In this event, select the *Bank Utilities* tab from the FlashTools98 tabsheet, highlight *Bank #1* and erase the bank. Then repeat the download procedure.

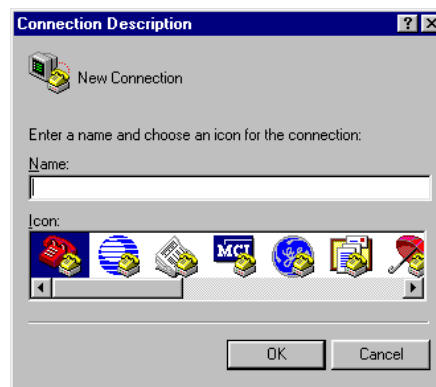
- At the end of the download, a sector-by-sector status check of the Flash memory can be viewed in the lower left corner of the FlashTools98 tabsheet window. Wait until the status check finishes before returning to work with the board. Once the status check is complete, the downloaded code can be executed.
- Returning to the *Communication* tab, click on the *Disconnect* button and exit FlashTools98.

Monitoring the execution of the Hello demo requires use of a terminal program, such as the HyperTerminal program included within Windows.

- Start the HyperTerminal program within the *Programs/Accessories* bar.
- The HyperTerminal main window will now appear⁵:
- Double-click on the HyperTerminal icon “*Hypertrm*” to create a new HyperTerminal session.



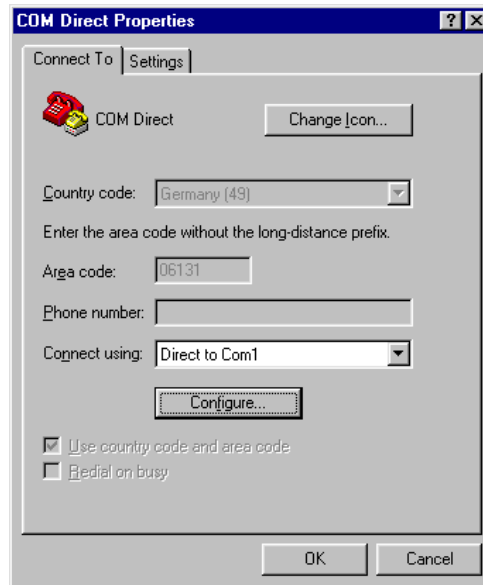
- The Connection Description window will now appear. Enter “COM Direct” in the *Name* text field.



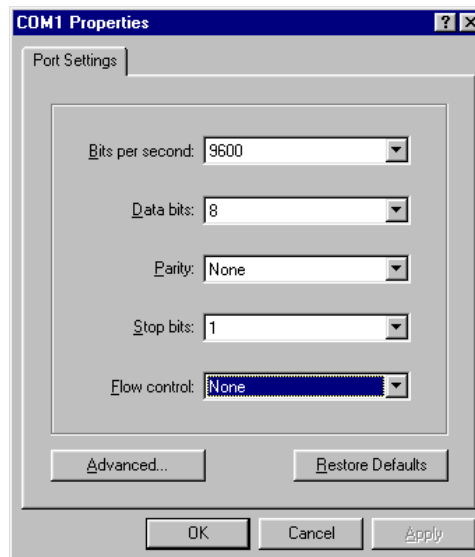
- Next click on *OK*. This creates a new HyperTerminal session named “COM Direct” and advances you to the next HyperTerminal window.

⁵: The HyperTerminal window has a different appearance for different versions of Windows.

- The *COM Direct Properties* window will now appear. Specify *Direct to COM1/COM2* under the *Connect Using* pull-down menu (be sure to indicate the correct COM setting for your system).

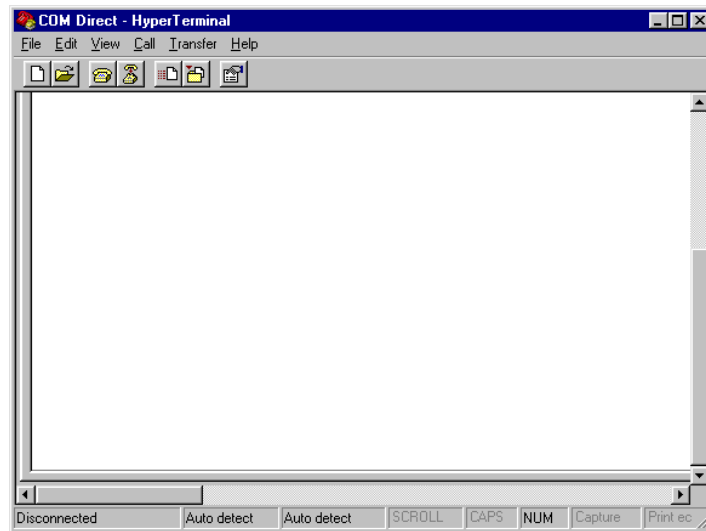


- Click the *Configure* button in the *COM Direct Properties* window to advance to the next window (*COM1/COM2 Properties*).




- Then set the following COM parameters: Bits per second = 9600; Data bits = 8; Parity = None; Stop Bits = 1; Flow Control = None.

- Selecting *OK* advances you to the *COM Direct–HyperTerminal* monitoring window. Notice the connection status report in the lower left corner of the window.



- Resetting the phyCORE Development Board LD 5V (at S2) will execute the *Hello.hex* file loaded into the Flash.
- Now push the <Space> bar on your keyboard once to start the automatic baud rate detection on phyCORE-P87C591 module.
- Successful execution will send the character string "Hello World" from the target hardware to the HyperTerminal window.

Pressing any other key than the <Space> bar leads to an improper baud rate since the automatic baud rate detection is based on the timing measurement during the transmission of a well known character – the <Space> character. As a result you may get incoherent characters in the HyperTerminal window.

- Click the disconnect icon  in HyperTerminal toolbar and exit HyperTerminal.
- If no output appears in the HyperTerminal window check the power supply, the COM parameters and the RS-232 connection.

You have now successfully downloaded and executed two pre-existing example programs in Intel **.hex* file format.

3 Getting More Involved

What you will learn with this example:

- how to start the Raisonance tool chain
- how to configure the Raisonance tools within the Integrated Development Environment (RIDE) for 51+XA demo version
- how to modify the source code from our examples, create a new project and build and download an output *.hex file to the target hardware

3.1 Starting the Raisonance Tool Chain

The Raisonance Integrated Development Environment (RIDE) for 51+XA demo software should have been installed during the install procedure, as described in *section 2.1*.

You can also manually install the tool chain by executing *install.exe* from within the `\Software\Raisonance` folder of your PHYTEC Spectrum CD.

Note:

It is necessary to use the Raisonance tool chain provided on the accompanying Spectrum CD in order to complete this QuickStart Instructions successfully. Use of a different version could lead to possible version conflicts, resulting in functional problems.

- Start the tool chain by selecting *Ride IDE* from within the *Programs/Raisonance Kit* program group.

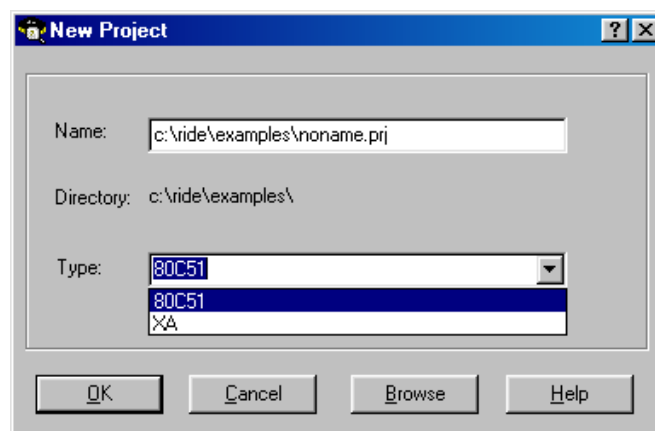
After you start RIDE, the window shown below appears. From this window you can create projects, edit files, configure tools, compile, assemble, link and debug.



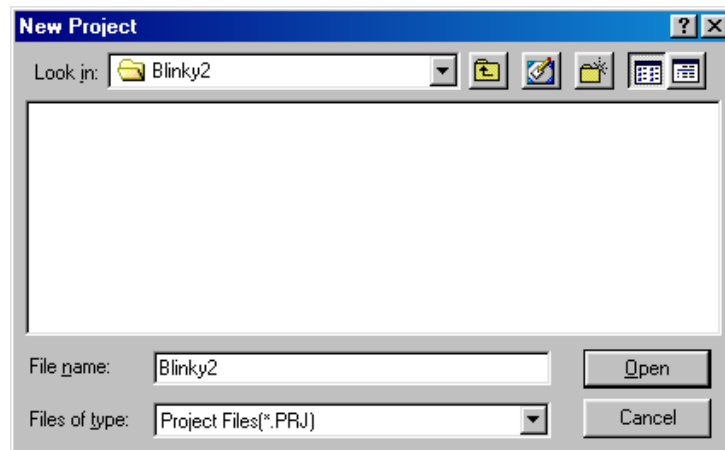
3.2 Creating a New Project and Adding an Existing Source File

RIDE automatically loads the most recently opened project. If you find an existing project when starting RIDE, close it by selecting the **Project** menu and *Close* the project.

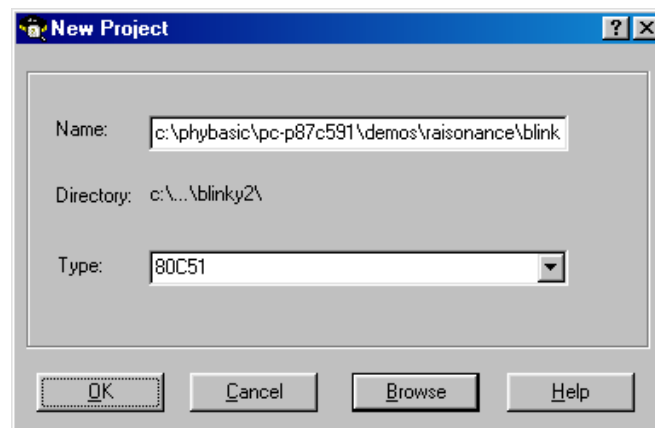
- To create a new project file open the **Project** menu and choose *New* within the RIDE menu bar. The window as shown below appears.



- Make sure that the correct architecture *Type* is selected. For this example select *80C51* (XA may be currently selected).
- Click on the *Browse* button and change to the project directory created by the installation procedure (default location *C:\PHYBasic\pC-P87C591\Demos\Raisonance\Blinky2*)
- In the text field '*File name*', enter the file name of the project you are creating. For this example, enter the name *Blinky2*.

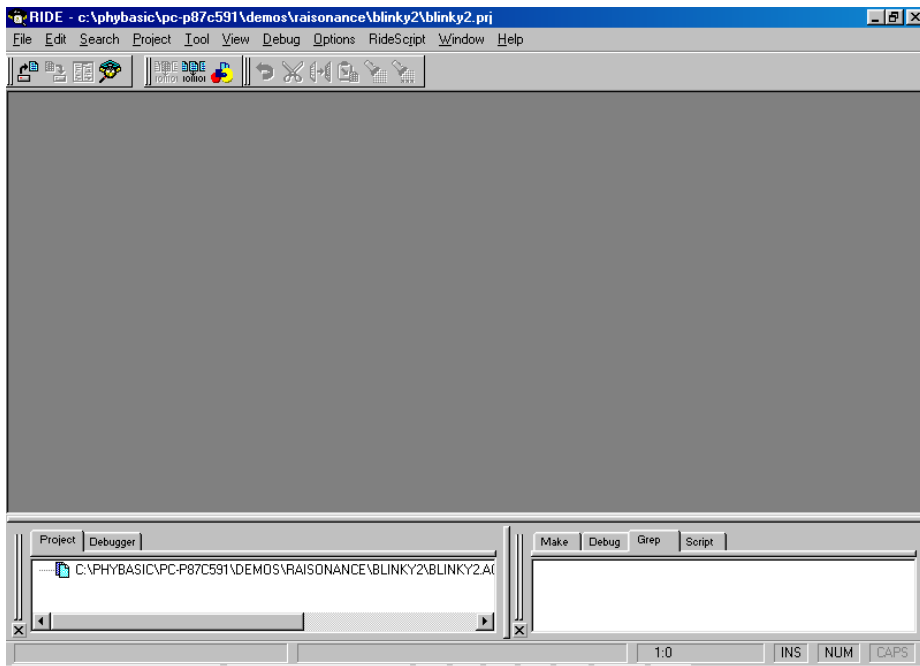


- Click on *Open*.

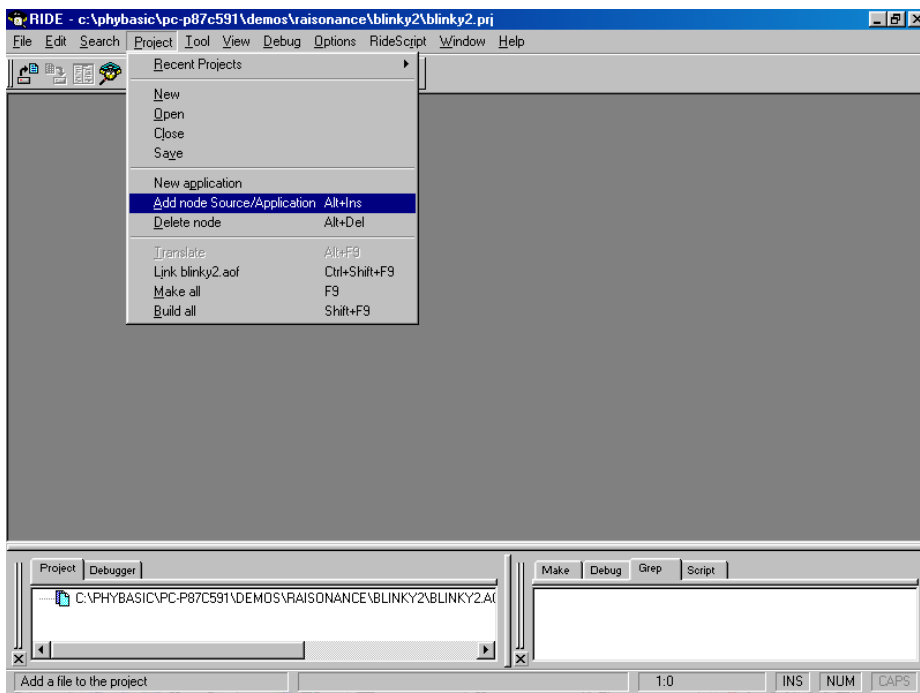


- Click on *OK* in the *New Project* window.

- The following window will appear:

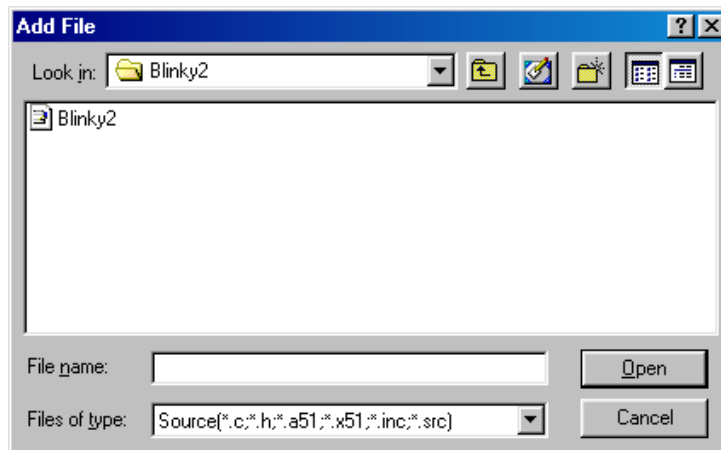


- From here, you will be able to add various files to your project.

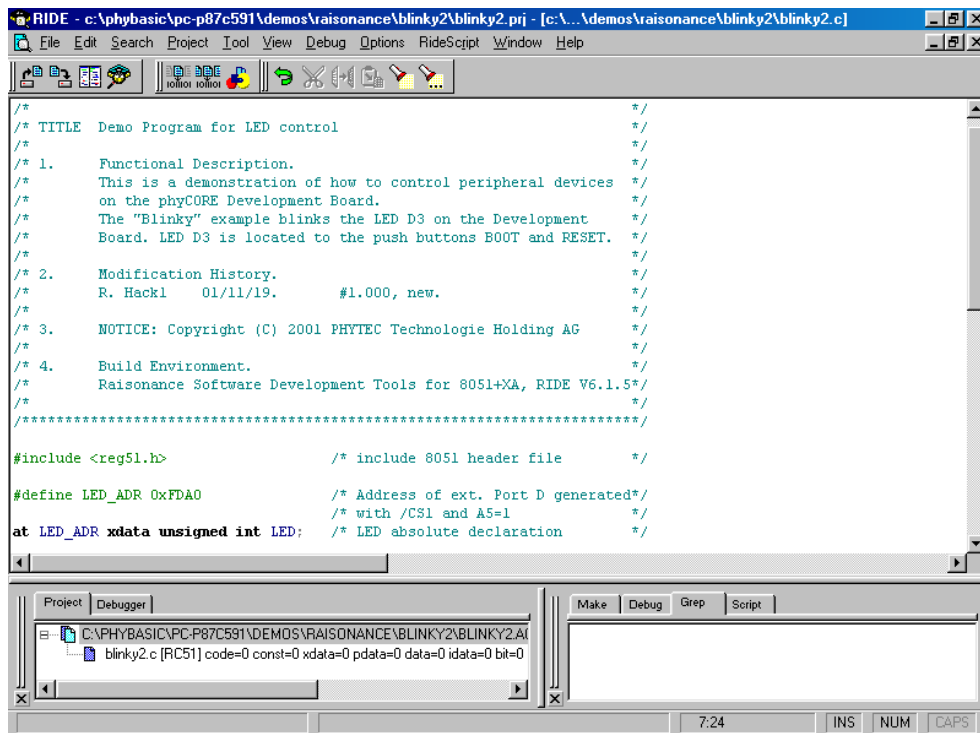


- Open the *Project* menu and choose *Add node Source/Application*.
-

- The following window will appear:



- Select the file *Blinky2.c* in the *Add File* window and click on *Open*.



At this point you have created a project called *blinky2.prj* and added an existing C source file called *blinky2.c*.

The next step is to modify the C source before building your project. This includes compiling, linking, locating and creating the hexfile.


3.3 Modifying the Source Code

- The source file *blinky2.c* is now open within the RIDE editor. If you closed the file, double click on the reference inside the project tree.
- Locate the following code section. Modify the section shown below (the values shown in bold and italic font) from the original counts to the indicated values:

```
while (1)                                /* loop forever                */
{
    LED = LED & 0xFE;                     /* output over PD port to LED D3 */
                                           /* Bit 1 of port PD = LED D3 = off*/
    for (i=0; i< 30000; i++) /* delay for 30000 counts        */
    {
        wait ();                          /* call wait function            */
    }
    LED = LED | 0x01;                     /* output over PD port to LED D3 */
                                           /* Bit 1 of port PD = LED D3 = on */
    for (i=0; i< 40000; i++) /* delay for 40000 counts        */
    {
        wait ();                          /* call wait function            */
    }
}                                           /* end of while(1)              */
                                           /* EOF                           */
```

This will change the LED on/off ratio.

3.4 Saving the Modifications

- Save the modified file by choosing *File/Save* or by clicking the *Save* icon .

3.5 Setting Tool Chain Options

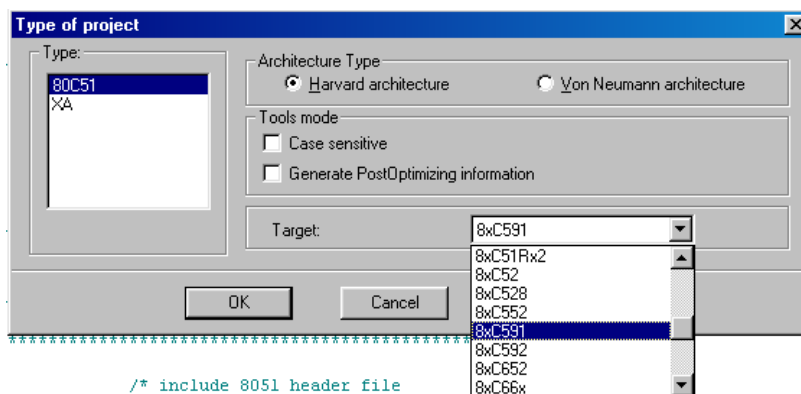
Raisonance tools include a Make utility that controls compiling and linking of various source files. Before using the macro preprocessor, assembler, C compiler or linker/locator, you must configure the corresponding options. Enter the changes as indicated below and leave all other options set to their default values. RIDE allows you to set various options with mouse clicks and these are all saved in your project file.

Note:

In most cases, options can be set at the project level. However, specific local options sometimes have to be set up differently when a file or a group of files require special options. In this case, a popup menu allows you to specify options at the level of the node.

To configure the Target:

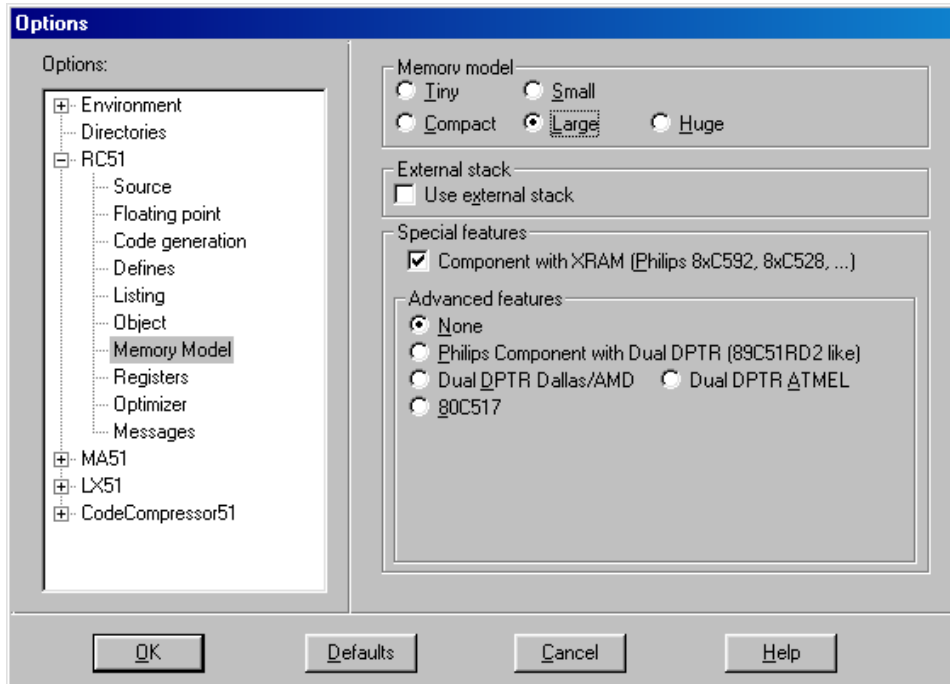
- Open the *Options/Target* menu and select the *8xC591* as shown below:



- Click on *OK* to save the configuration.

To configure the RC51 Compiler:

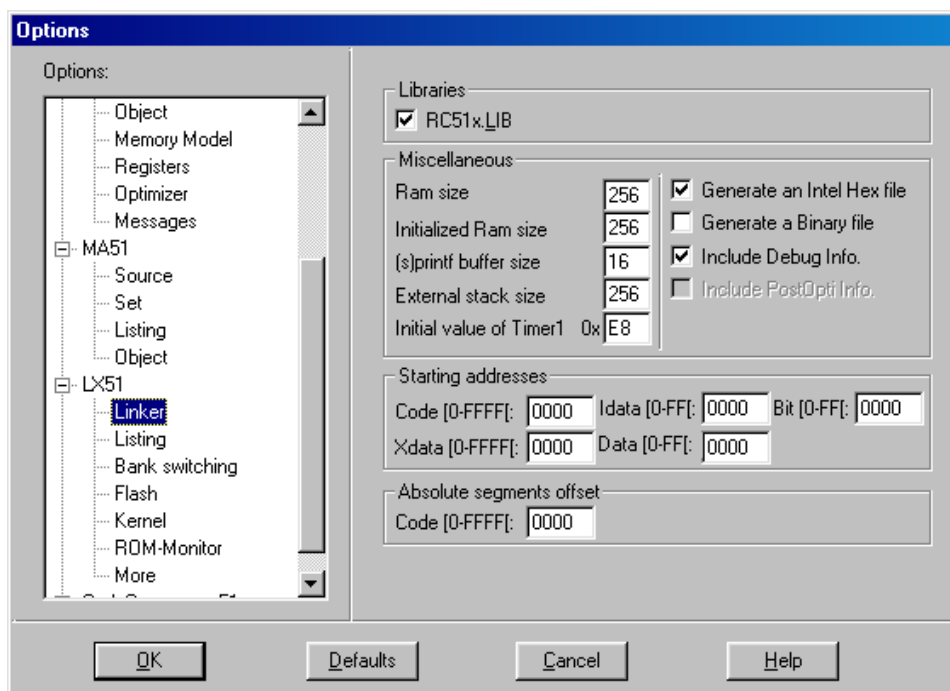
- Open the *Options/Project/RC51* menu and choose *Memory Model*.
- Select the *Large* memory model and activate the checkbox *Component with XRAM* under *Special features* and keep the other RC51 options at their default settings.



- Click on *OK* to save these settings.

To configure the LX51 Linker/Locator:


- Open the *Options/Project* menu and choose *LX51\Linker*.
- Check that the *Generate an Intel Hex file* checkbox is active. This option should be enabled by default.
- All others options are correct to run our first example. Click on *OK* to save the configurations.



The linker/locator options are now suitable for the *Blinky2* project, enabling you to build an absolute object file (*.aof) and a hexfile.

3.6 Building the Project

You are now ready to run the compiler and linker using the Make utility.

- Click on the 'Make All' Command icon  from the RIDE toolbar or open the *Project* menu and select *Build All* or *Make All*.

If the program specified (*Blinky2.c*) contains any errors, they will be shown in the *Message Window* at the bottom of the screen.

If there are no errors, the code is compiled and linked and the executable code is ready to be downloaded to the module. The created hexfile will have the name of the project with *.hex* as the filename extension (in this case *Blinky2.hex*).

Note:

A machine-readable, executable hexfile has been created. Other files (e.g. list files **.lst* and map files **.m51*) are generated to help the debugging or troubleshooting and error searching process.

- If a list of errors appears, double-click on the error to open the file and locate the error. Use the editor to correct the error(s) in the source code and (re-)build the project.

3.7 Downloading the Output File

- Reset the target hardware and force it into Flash programming mode by simultaneously pressing the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board LD 5V and then releasing first the Reset and, two or three seconds later, the Boot button.
- Start FlashTools98.
- At the *Communication Setup* tab of the FlashTools98 tabsheet, specify the proper serial port and transmission speed (9,600 Baud) for communication between host-PC and target hardware and click the *Connect* button to establish connection to the target hardware.
- Returning to the FlashTools98 tabsheet, choose the *Bank Utilities* tab, highlight *Bank #1* within the *Bank Erase* section, and click on the *Erase Bank(s)* button to erase this memory bank.
- Wait until the status check in the lower left corner of the FlashTools98 tabsheet finishes, returning the connection properties description to the lower left corner of the window.
- Next choose the *File Download* tab and click on the *File Open* button.
- Browse to the correct drive and path for the phyCORE-P87C591 Demo folder (default location **C:\PHYBasic\pC-P87C591\Demos\Raisonance\Blinky2\Blinky2.hex** and click *Open*.
- Click on the *Download* button and view the download procedure in the status window.
- Returning to the *Communication* tab, click on the *Disconnect* button and exit FlashTools98.
- Press the Reset button (S2) on the Development Board.

If the modified hexfile properly executes, the LED should now flash in a different mode with different on and off durations.

You have now modified source code, recompiled the code, created a modified downloadable hexfile, and successfully executed this modified code.

3.8 “Hello2”

A return to the “Hello” program allows a review of how to modify source code, create and build a new project, and download the resulting output file from the host-PC to the target hardware. For detailed commentary on each step, described below in concise form, *refer back to the “Blinky2” example starting at section 3.2.*

3.8.1 Creating a New Project

- Start the Raisonance RIDE environment and close all projects that might be open.
- Open the **Project** menu and create a new project called ***Hello2.prj*** within the existing project folder
C:\PHYBasic\pC-P87C591\Demos\Raisonance\Hello2
(default location) on your hard-drive. Select the 80C51 architecture for this project.
- Add ***Hello2.c*** and ***Serinit.c*** from within the project folder to the project ***Hello2.prj***.
- *Save* the project.

At this point you have created a project called ***Hello2.prj*** consisting of the C source files ***Hello2.c*** and ***Serinit.c***.

3.8.2 Modifying the Example Source

- Double-click the file *Hello2.c* from within the project window.
- Use the editor to modify the *printf* command:

```
printf ("\x1AHello World\n")
```

to

```
printf ("\x1APHYTEC... Stick It In!\n")
```

- Save the modified file under the same name *Hello2.c*.

3.8.3 Setting Tool Chain Options

The same tool chain options can be used as for the *Blinky2* project described in *section 3.5*.

3.8.4 Building the New Project


- Build the project.
- If any source file in the project contains errors, they will be shown in an error dialog box on the screen. Use the editor to correct the error(s) in the source code, save the file and (re-)build the project.

If there are no errors, the code is assembled and linked and the executable code is ready to be downloaded to the board.

3.8.5 Downloading the Output File

- Reset the target hardware and force it into Flash programming mode by simultaneously pressing the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board LD 5V and then releasing first the Reset and, two or three seconds later, the Boot button.
- Start FlashTools98.
- At the *Communication Setup* tab of the FlashTools98 tabsheet, specify the proper serial port and transmission speed (9,600 baud) for communication between host-PC and target hardware and click the *Connect* button to establish connection to the target hardware.
- Returning to the FlashTools98 tabsheet, choose the *Bank Utilities* tab, highlight *Bank #1* within the *Bank Erase* section, and click on the *Erase Bank(s)* button to erase this memory bank.
- Wait until the status check in the lower left corner of the FlashTools98 tabsheet finishes, returning the connection properties description to the lower left corner of the window.
- Next choose the *File Download* tab and click on the *File Open* button.
- Browse to the correct drive and path for the phyCORE-P87C591 demo folder (default location ***C:\PHYBasic\pC-P87C591\Demos\Raisonance\Hello2\Hello2.hex*** directory (default location).
- Click on the *Download* button and view the download procedure in the status window.
- Returning to the *Communication* tabsheet, click on the *Disconnect* button and exit FlashTools98.

3.8.6 Starting the Terminal Emulation Program

- Start HyperTerminal and connect to the target hardware using the following COM parameters: Bits per second = *9600*; Data bits = *8*; Parity = *None*; Stop Bits = *1*; Flow Control = *None*.
- Resetting the phyCORE Development Board LD 5V (at S2) will execute the *Hello2.hex* file loaded into the Flash.
- Now push the <Space> bar on your keyboard once to start the automatic baud rate detection on phyCORE-P87C591 module.
- Successful execution will send the modified character string "**PHYTEC... Stick It In!**" to the HyperTerminal window.
- Click the Disconnect icon .
- Close the Hyper Terminal program.

You have now modified source code, recompiled the code, created a downloadable hexfile, and successfully executed this modified code.

4 Debugging

This Debugging section provides a basic introduction to the debug functions included in the Raisonance RIDE tool chain. Using an existing example, the more important features are described. For a more detailed description of the debugging features, please refer to the appropriate manuals provided by Raisonance.

The Raisonance RIDE integrated debugger offers two operating modes that can be selected in the *Options\Project\LX51\ROM-Monitor* and *Options\Debug* dialog:

- The **Simulator** allows PC-based microcontroller simulation of most features of the 8051 microcontroller family without actually having target hardware. You can test and debug your embedded application before the hardware is ready. RIDE simulates a wide variety of peripherals, including the serial port, external I/O, and timers.
- The **Real Mode**, using either the Raisonance ROM monitor or an In-Circuit Emulator, allows target-based debugging. When using the ROM monitor, the debugger communicates with the target hardware via a monitor kernel that is running on the target system.

The following examples utilize the Real Mode/ROM monitor interface.

Note:

Use of the monitor program requires protection (reservation) of some controller resources, such as the serial interface, the serial interrupt and timer 1. These resources are necessary to allow communication between the monitor program on the target hardware and the RIDE Debugger (*refer to the Raisonance manuals for further information*). Do not use these resources when developing an application program to be debugged using the ROM monitor interface.

Before using the ROM monitor interface, a special ***.hex** file (the monitor loader firmware) must be downloaded to the target hardware.

4.1 Preparing the Target Hardware to Communicate with ROM Monitor

- Ensure that the target hardware is properly connected to the host-PC and a power supply.
- Reset the target hardware and force it into Flash programming mode by simultaneously pressing the Reset (S1) and Boot (S2) buttons on the Development Board and then releasing first the Reset and, two or three seconds later, the Boot button.
- Start FlashTools98 for Windows.
- At the *Serial Interface* tab of the FlashTools98 tabsheet, specify the proper serial port and transmission speed for communication between host-PC and target hardware and click the *Connect* button to establish connection to the target hardware.
- Returning to the FlashTools98 tabsheet, choose the *Bank Utilities* tab, highlight *Banks #1* and click on the *Erase Bank(s)* button.
- Next choose the *File Download* tab and click on the *File Open* button.
- Download the file *load51.hex* from the *Tools* folder *C:\PHYBasic\pC-P87C591\Tools\Raisonance\Loader* (default location).

The PHYTEC Spectrum CD-ROM also contains the *loadxa.hex* monitor file. This version is made for XA-compatible phyCORE modules. *Please refer to readme files within the Loader directory for details.*

- Click on the *Download* button and view the download procedure in the status window.

If download is successful, the loader kernel has been programmed into the external Flash memory. The target hardware is now prepared to communicate with the Raisonance RIDE debugging tools installed on the host-PC.

- Disconnect from the target hardware after the download has finished either by clicking the *Disconnect* button on the *Communication Setup* tabsheet or choosing the *Connect/Disconnect* icon from the FlashTools98 toolbar.
- Exit FlashTools98.

4.2 Creating a Debug Project and Preparing the Debugger

4.2.1 Creating a New Project

- Start the RIDE environment and close all projects that might be open.
- Open the *Project* menu and create a new project called *Debug.prj* within the existing project folder
C:\PHYBasic\pC-P87C591\Demos\Raisonance\Debug
(default location) on your hard-drive. Select the 80C51 architecture for this project.
- Add *Debug.c* and *Serinit.c* from within the project folder to the project *Debug.prj*.
- *Save* the project.

At this point you have created a project called *Debug.prj*, consisting of two C source files called *Debug.c* and *Serinit.c*.

4.2.2 Setting Options for Target

When setting the memory configuration, the memory layout necessary for the monitor must be taken into consideration.

The standard 8051 controller uses a Harvard memory architecture. In this architecture, access to CODE and XDATA memory space goes to physically different memory devices. Normally, for access to CODE space, a non-volatile memory is utilized, i.e. ROM or Flash. For access to XDATA space, a RAM is used. Using this memory model with an 8051 derivative allows access to up to 64 kByte of memory for CODE and 64 kByte for XDATA.

When debugging with the Raisonance monitor, it is important that the user program (CODE) can be changed during runtime (e.g. to enable setting of breakpoints). This requires the user program to be stored in RAM and **not** in Flash. In order to ensure that the user program is running in RAM, the monitor loader automatically configures a von Neumann memory architecture in the address range 0000H-EFFFH after reset. Here, in contrast to the Harvard architecture, access to CODE and XDATA space is directed towards the same physical memory device, normally RAM. With this von Neumann memory architecture, it is now possible to change the application program during runtime.

The following figure (see Figure 4) depicts the memory layout that is configured by the Raisonance monitor for 64 kByte RAM.

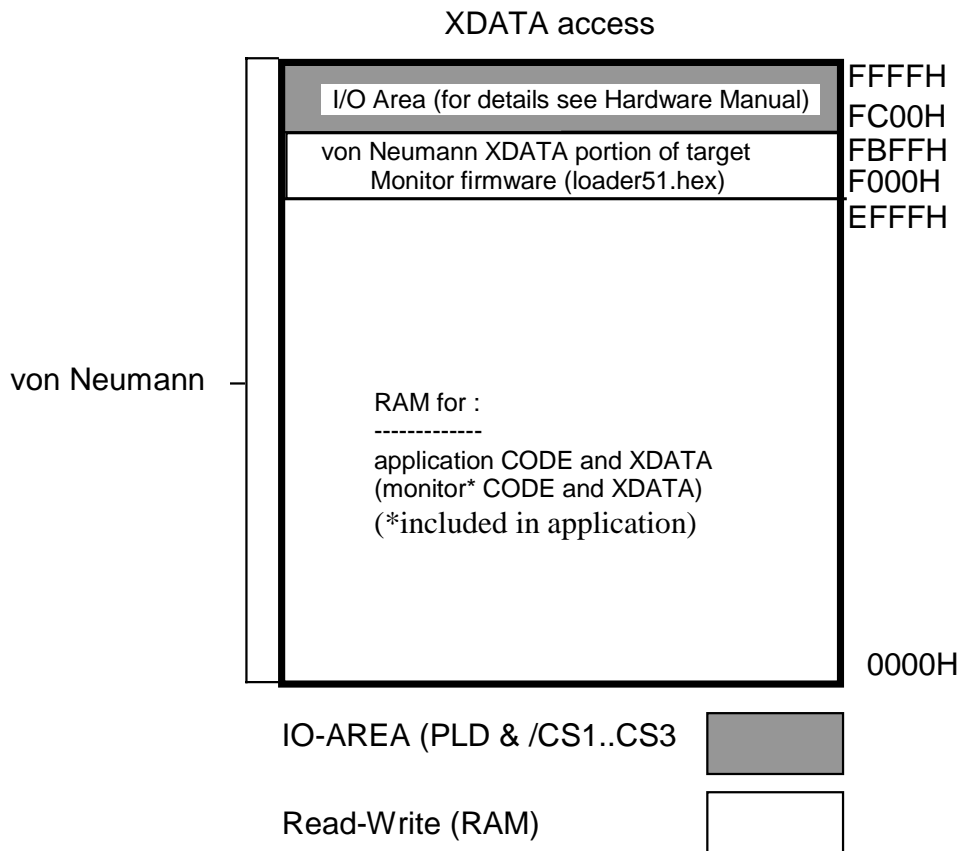
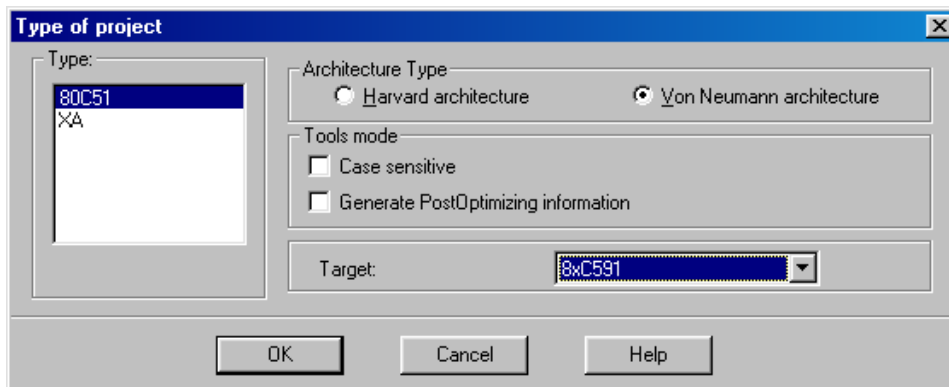


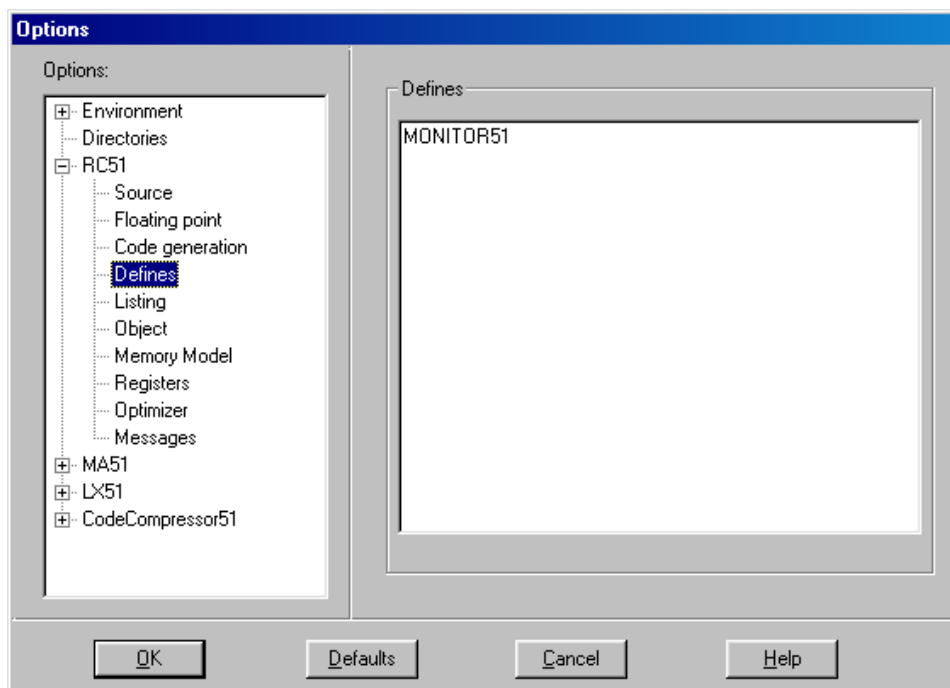
Figure 4: Memory Model for Use with the Raisonance Monitor (64 kByte RAM)

Note:
 When using the von Neumann memory architecture, ensure that the CODE and XDATA areas within the application program do **not** overlap. This is important because otherwise portions of the program (CODE) will be overwritten by e.g. variables (XDATA), resulting in an error when executing user code.

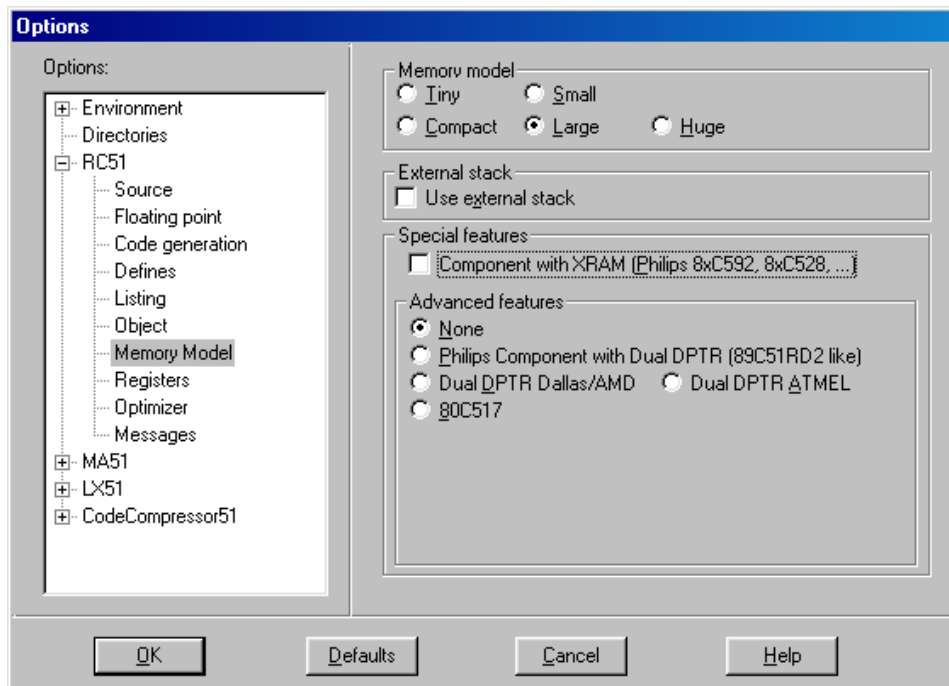
- Open the **Options/Target** menu, select the *8xC591* and a *Von Neumann architecture* as shown below:



- Click on *OK* to save the configuration.
- Open the **Options/Project/RC51** menu and choose *Defines*. Add **MONITOR51** in the *Defines* input field.

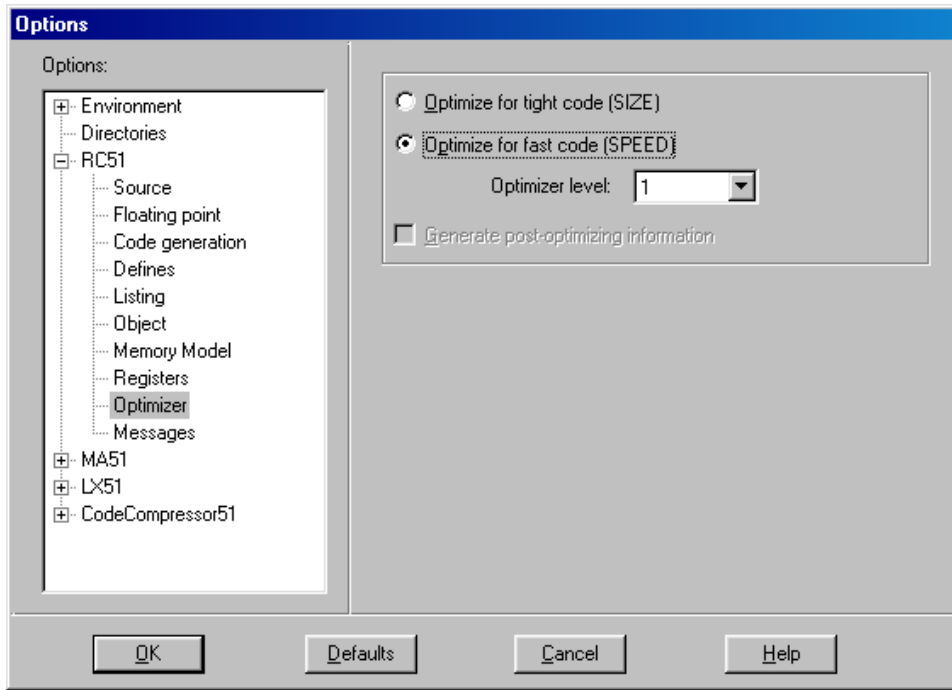


- In the *Options/Project/RC51* menu, now choose *Memory Model*.
- Select the *Large* memory model. Disable the checkbox *Component with XRAM* under *Special features*. This is necessary because the XRAM portion can not be accessed as von Neumann memory architecture.

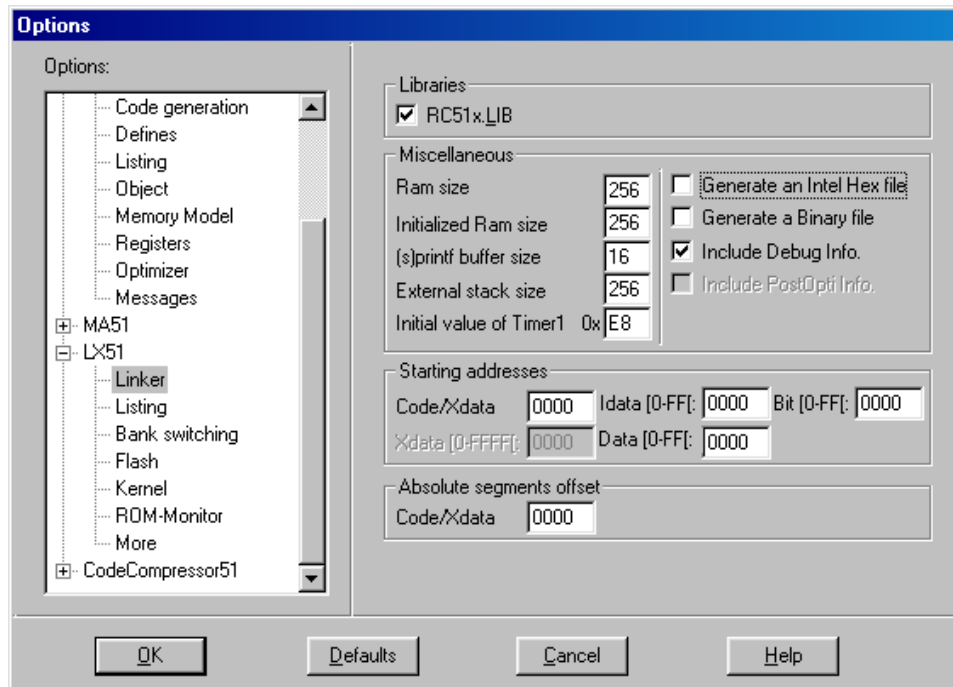


- Click on *OK* to save these settings.

- In the **Options/Project/RC51** menu, now choose **Optimizer**.
- Select the **Optimizer level 1** as shown below. This is necessary because the created **Debug.aof** file can be better debugged when the C file is compiled with this optimization level.

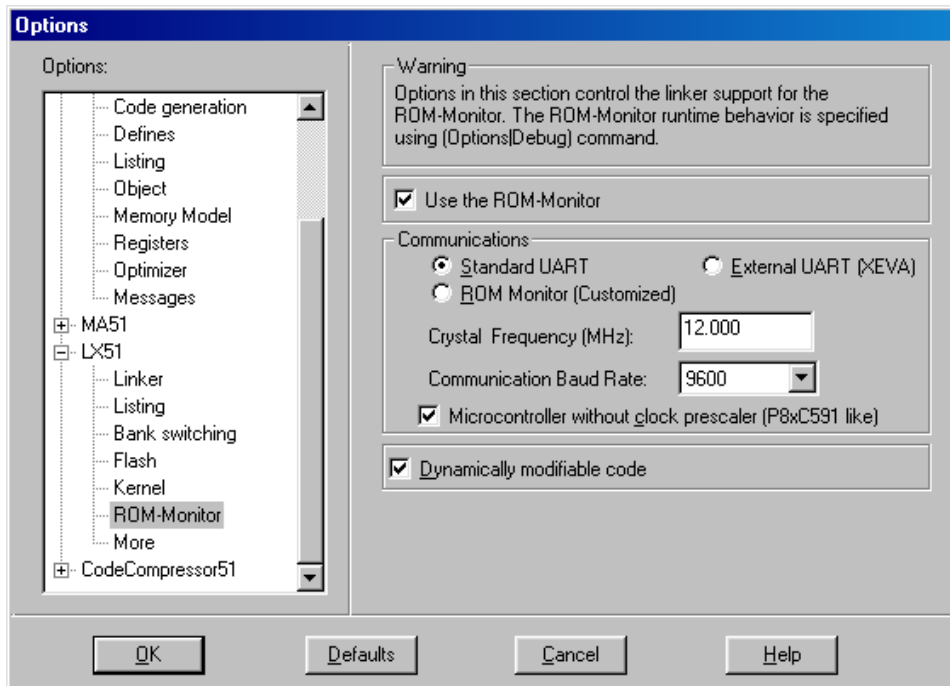



- Open the *Options/Project/LX51* menu and choose *Linker*.
- Check that the *Generate an Intel Hex file* checkbox is disabled. This option should be enabled by default.



The memory ranges for off-chip CODE and off-chip XDATA memory are configured to fit within the von Neuman memory space as configured by the the Raisonance *load51.hex* monitor file (refer to Figure 4).

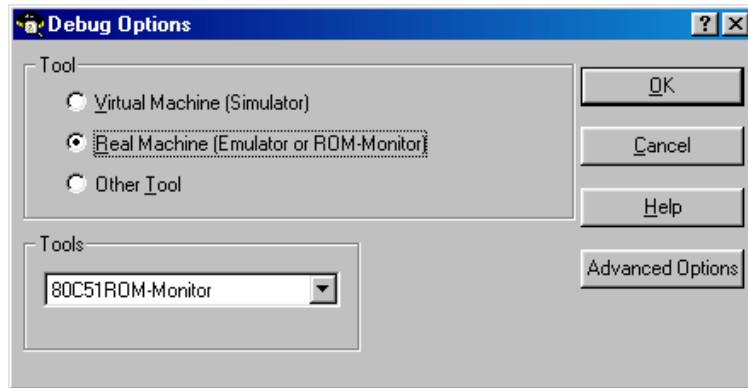
- In the **Options/Project/LX51** menu now choose **ROM-Monitor**. Activate the checkbox **Use the ROM-Monitor**, select the **Standard UART** radio button, a **Crystal Frequency** of **12.000 MHz** and a **Communication Baud Rate** of **9600**. Make sure the checkbox **Microcontroller without clock prescaler (P8xC591 like)** is enabled.



- Click on **OK** to save these settings.
- The linker/locator options are now suitable for the **Debug** project, enabling you to build an absolute object file (*.aof).
- Click on the **'Make All' Command** icon  from the **RIDE** toolbar or open the **Project** menu and select **Build All** or **Make All**.

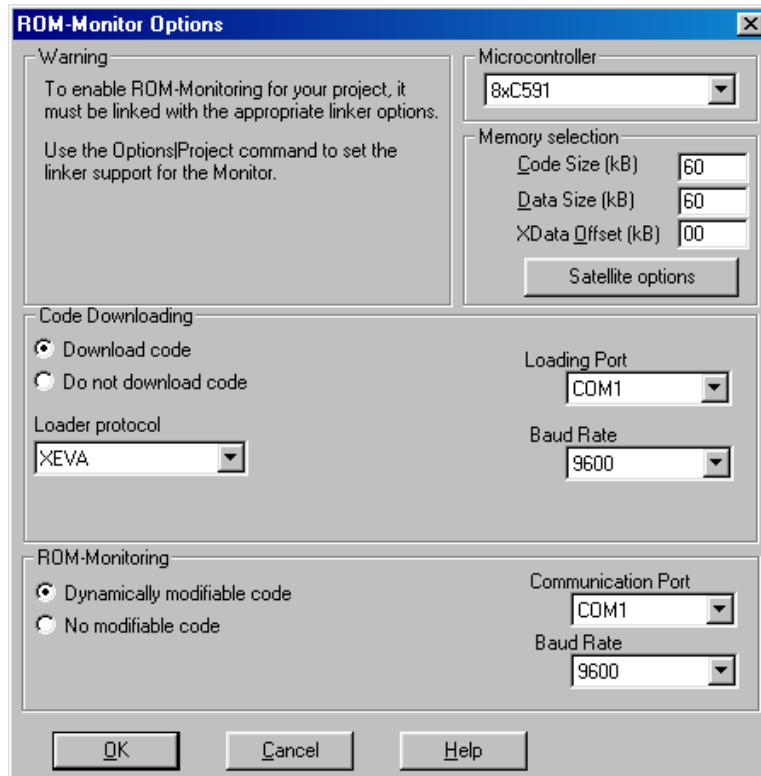
4.3 Preparing the Debugger

- Open the *Options/Debug* menu.
- Click on the *Real Machine* button and select *80C51ROM-Monitor* as shown below:

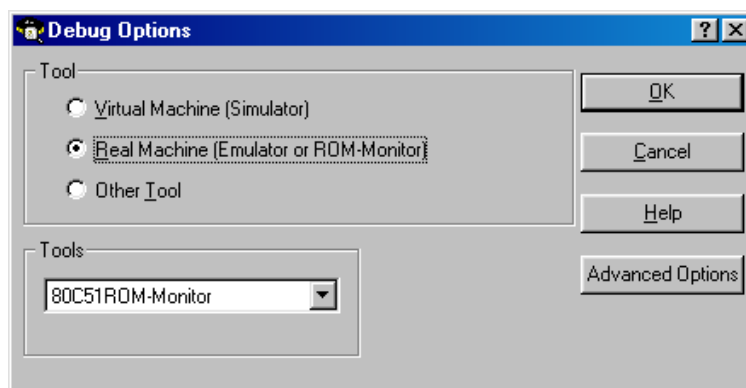


- Click on the *Advanced Options* button to specify additional debugging options. Make sure that the *8xC591* is selected in the microcontroller pull-down menu. Select the correct COM port and baud rate in both the *Loading Port* and the *Communication Port* menus as shown in the screen capture on the following page. Make sure *XEVA* is selected as *Loader protocol*.

- The maximum code and data size that can be configured in the memory selection section is 60 kB. This is because the monitor loader code itself is located at address F000H. In addition, the I/O area of the phyCORE module occupies the memory range between FC00H and FFFFH. These memory areas can not be used by the application code.



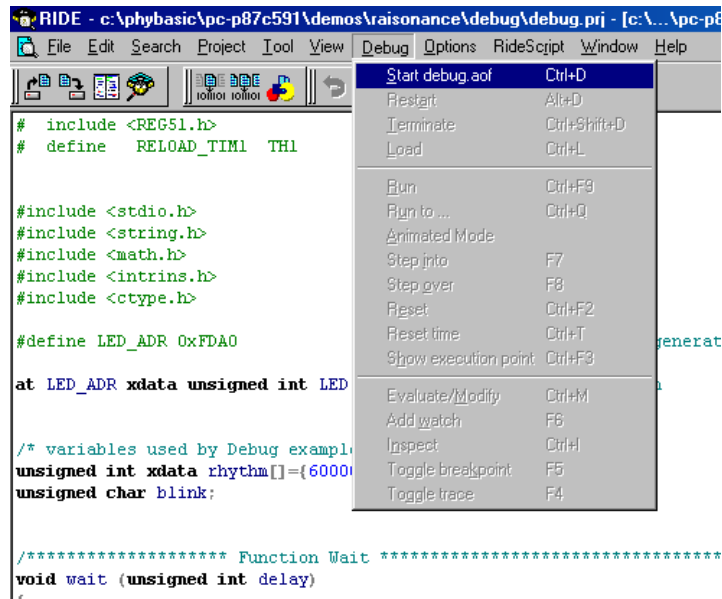
- Click on the *OK* button to exit the *ROM-Monitor Options* window.
- The *Debug Options* window will appear.



- Click on the *OK* button again.

4.4 Starting the Debugger

- Before starting the Debugger on the host-PC, press the Reset button (S1) on the phyCORE Development Board LD 5V to start the previously downloaded monitor kernel.
- To start the RIDE debug environment, select *Start debug.aof* in the *Debug* menu.

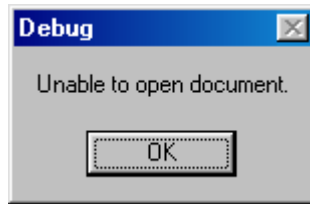


- The RIDE debug environment now initializes the ROM-Monitor and loads the debug program. You will see a blue status bar from left to right within each window indicating the progress of these steps.

If a problem occurs during data transfer, the following window will appear:



- Click on *OK*. A new window will appear:

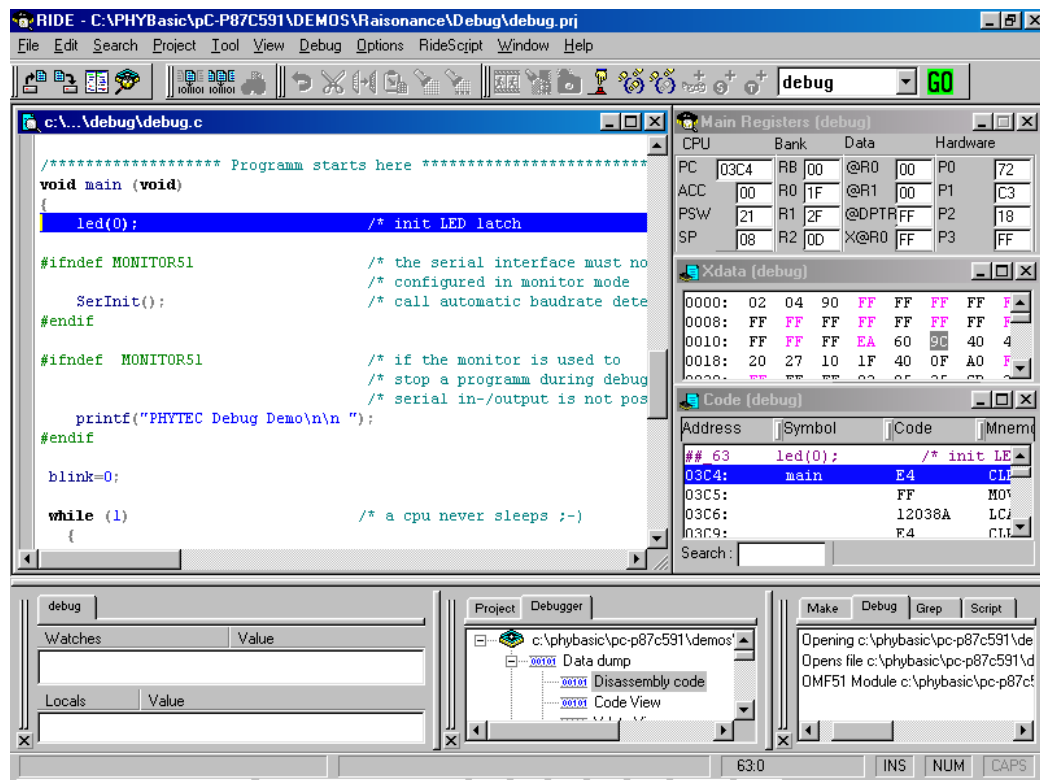


- Click on *OK* again.
- Verify the COM port and the baud rate (9,600 baud) settings in the ***Options/Project/LX51/ROM-Monitor*** tab and the ***Options/Debug*** menu. Make sure the settings are the same in each configuration menu.
- Push the Reset button S2 on the phyCORE Development Board LD 5V and start the RIDE debug environment by selecting *Start debug.aof* in the ***Debug*** menu.
- If the data transfer was successful, the following window will appear:



- Click on *OK* to continue.
- The RIDE debug environment is now connecting to the ROM-Monitor.






If the data transfer was successful, a screen similar to the one shown below will appear. The **Project** window changed to the **Debugger** page. The debug toolbar is also displayed. In the lower part of the debug screen you will see the **Command** and **Watch** window. The **Xdata** window is shown in the lower right section of the screen.



You may need to open, resize and /or move some windows to make your screen look similar to the screen capture. You can open inactive windows by choosing the desired window from the **View** pull-down menu.

- The debugger will automatically run to the 'main' function and stop.

4.5 Raisonance Debug Features

- The **Debugger** toolbar gives access to the following debug commands: *Reset*, *Go*, *Stop*, *Step Into*, *Step Over*, *Step Out* and *Run to Cursor line*.
- RIDE uses *Step (into function calls)* to single step one line at a time. *Step (into function calls)* is also used to enter a function in the same fashion. Depending on the current window (either Disassembly or Source), the meaning of “Stepping” will be slightly different, and automatically adapted to the context. In a Source window, stepping will be performed at the source level (e.g. line to line). In a Disassembly window, stepping will be performed at the instruction level. To *Step into*, click on the  button, or press <F7>, or open the *Debug / Step Into* menu.
- *Step (over function calls)* means to skip over a function that you are not interested in. To *Step over*, click on the  button, or press <F8>, or open the *Debug / Step Over* menu.
- To reach the cursor location, open the *Debug / Run to* menu.
- You can reset the application by clicking on the *Reset application*  button in the debug toolbar. The program will arrive at `main()` when the Reset is performed from a *Source* window, or at the reset vector if it is performed from the *Code disassembly* window.
- The *Go*  icon will change into a *Stop*  icon during the program execution.

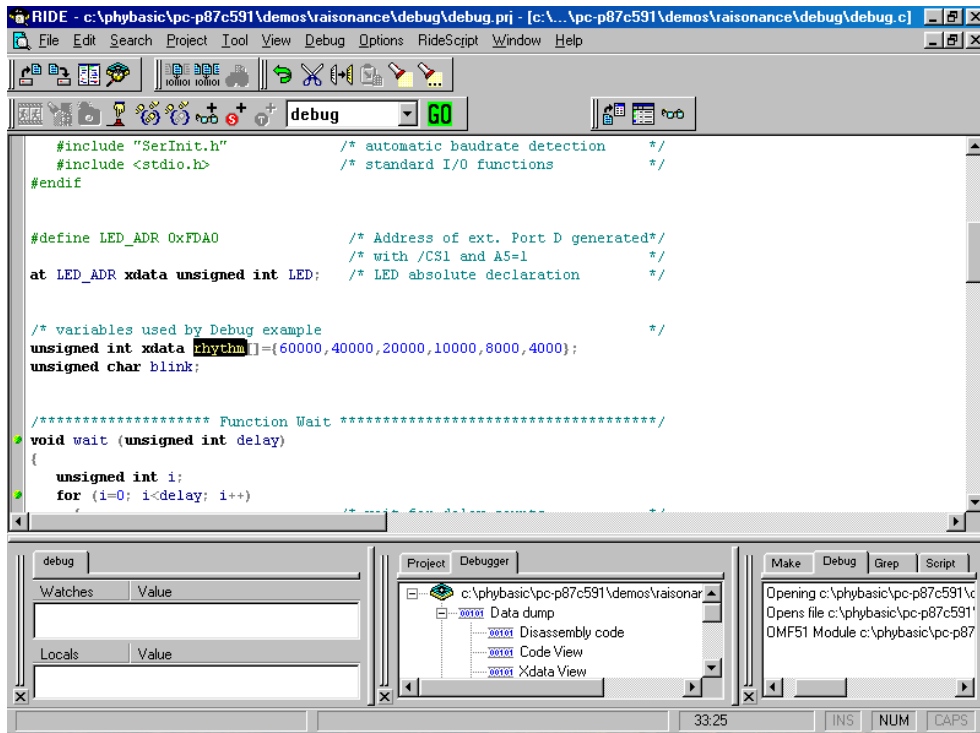
Clicking the *Go* icon runs the program without active debug functions. To stop program execution at a desired point, a breakpoint can be placed before the *Go* icon is clicked.

The *Stop* icon interrupts and stops the running program at an undetermined location.

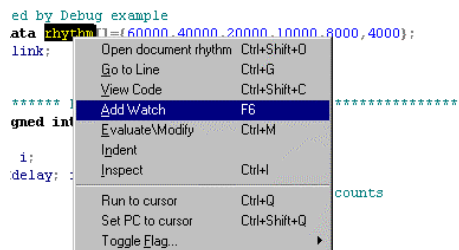
4.6 Using the Raisonance Debug Features

4.6.1 Watch Window

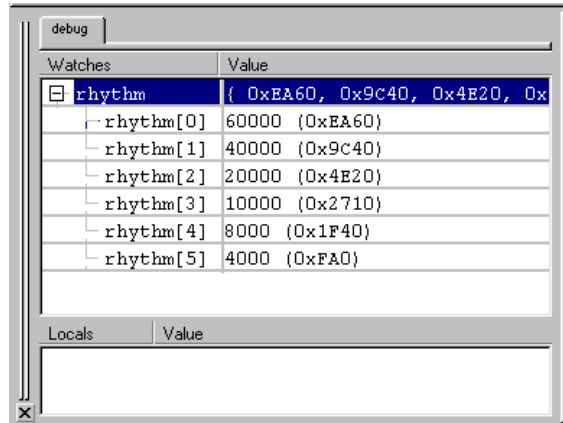
- Go to the code line where the constant *rhythm* is defined. Select the constant by double-clicking on the constant name.



- Right-click on the constant *rhythm* and select *Add Watch* in the pop-up window. You may also use the <F6> function key to enable the Add Watch window.

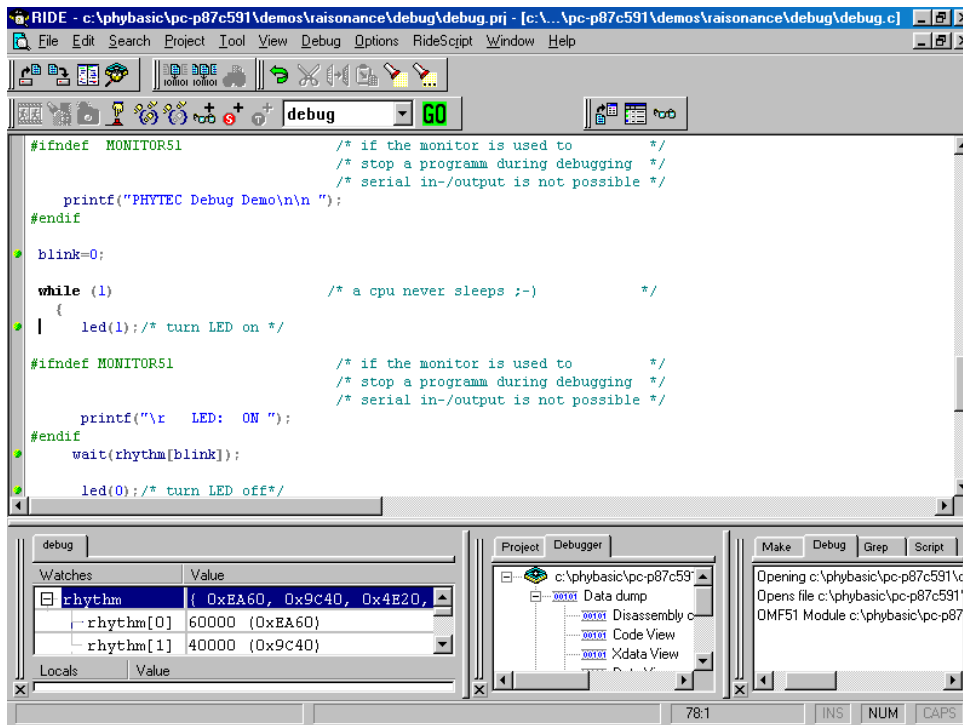


- The **Watch** window now shows the constant "**rhythm[]**". The small **+** sign in front of **rhythm** indicates that this is an array with a group of array elements. Click the **+** sign to expand the view and to see all array elements of "**rhythm[]**".

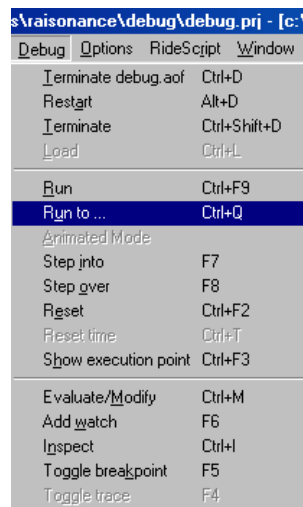


4.6.2 Run to ...

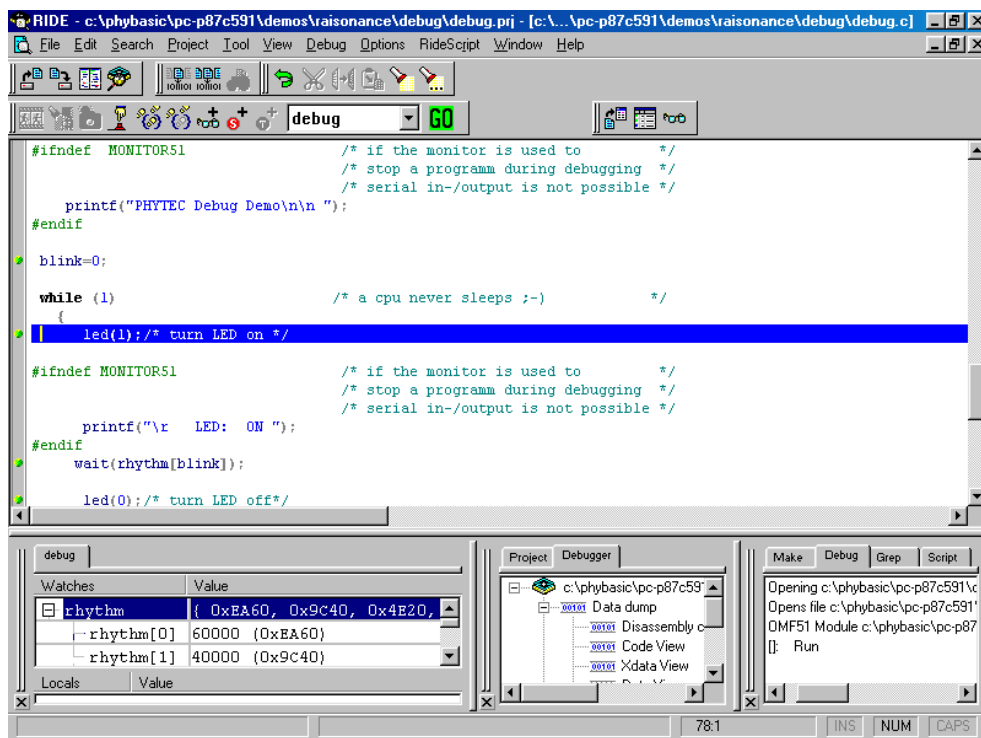
- The **Run to...** command executes the program until it reaches the code line where the cursor is currently located. Go with the cursor to the code line *led(1)*;



- To run the program and stop at the selected code line, select **Run to...** in the **Debug** menu.

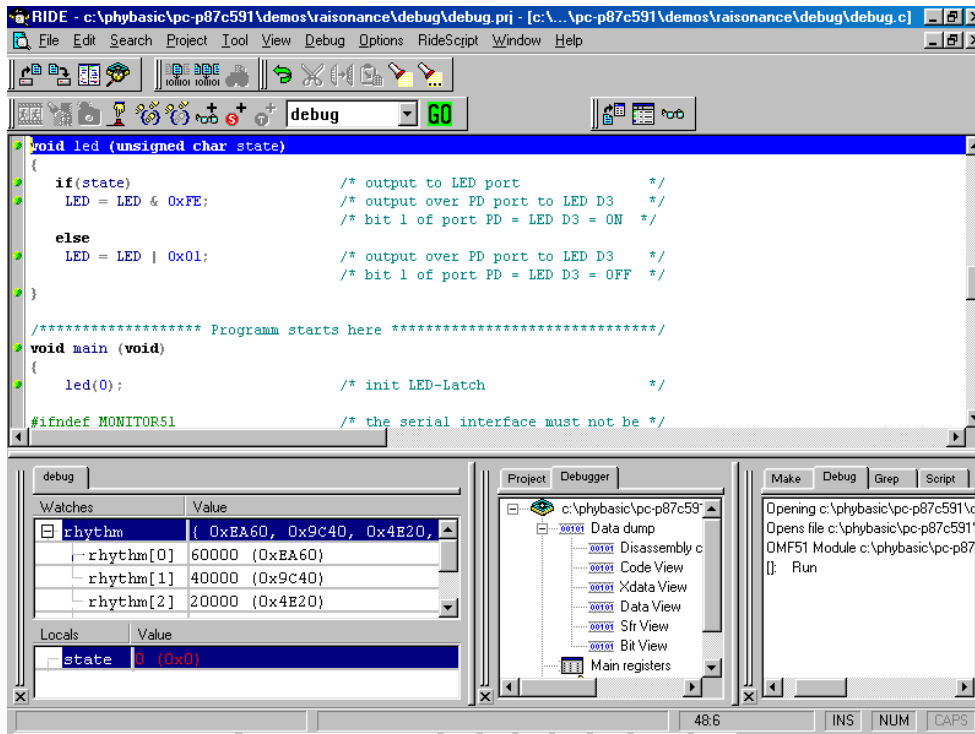


- The Debugger executes the program until it reaches the code line where the cursor is currently located. This code line is now highlighted in blue color as shown below:



4.6.3 Step Into and Step Over

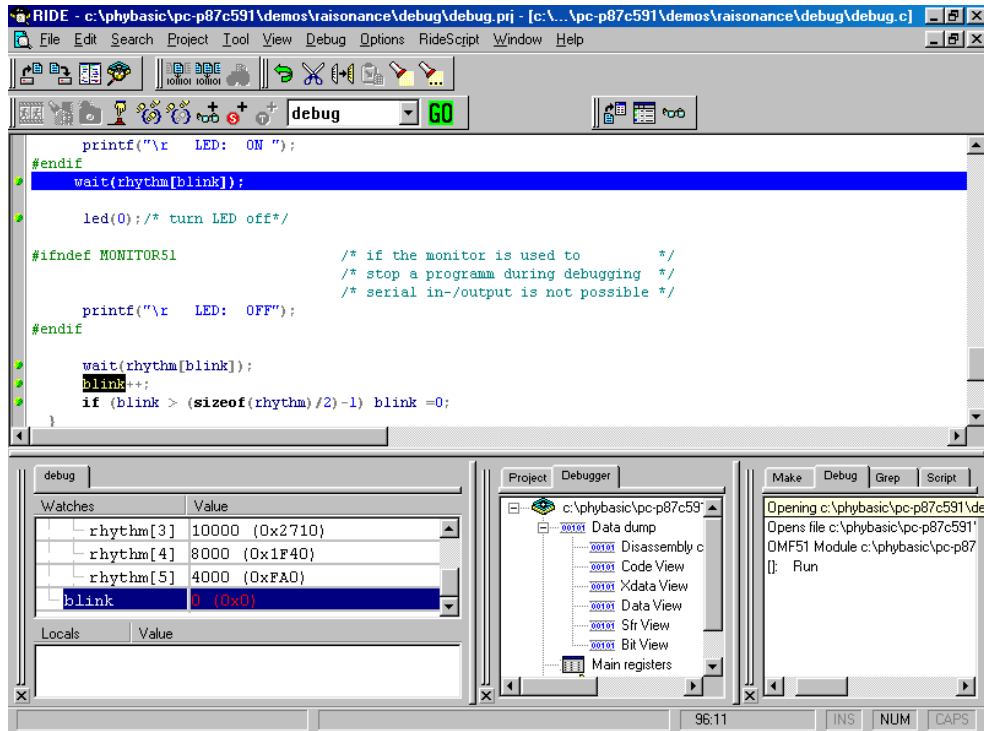
- Click on the *Step Into* icon to enter the 'led()' function.
- Notice that the local variable *state* is now shown in the *Watch* window. The initial value is '0'.



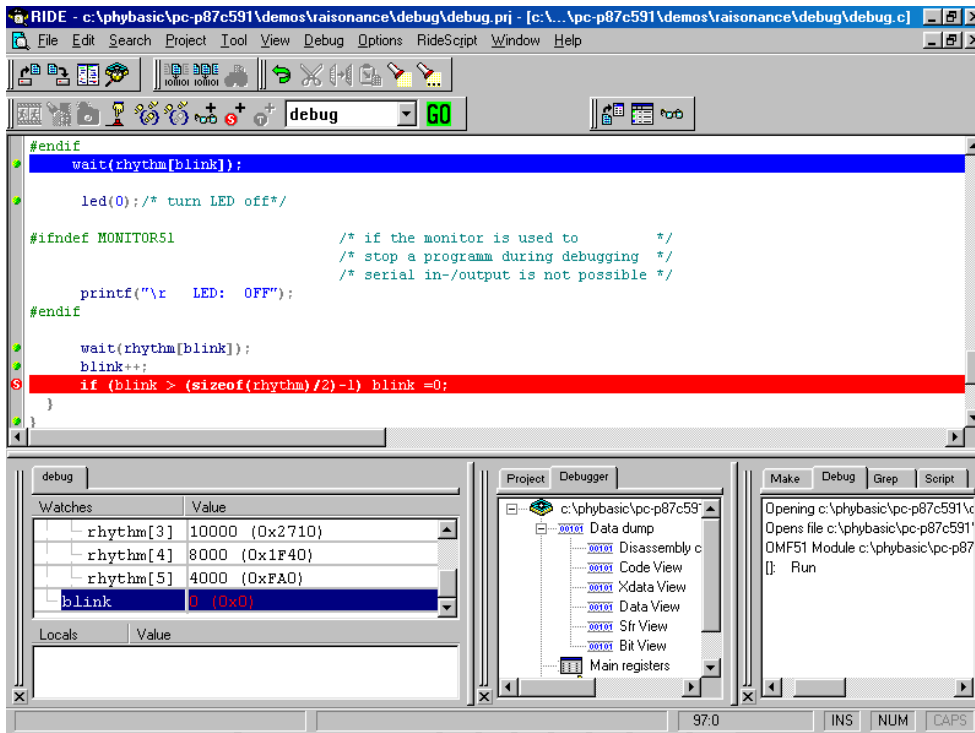
- Now click on the *Step Over* icon four times to single-step through the 'led()' function. Notice that a new value ('1') for the local variable *state* is shown in the *Watch* window after the first step over command.
- Notice that the LED (D3) on the Development Board illuminates after the third time you clicked on the *Step Over* icon.


4.6.4 Breakpoints

- Select the variable *blink* and add *blink* to the *Watch* window.

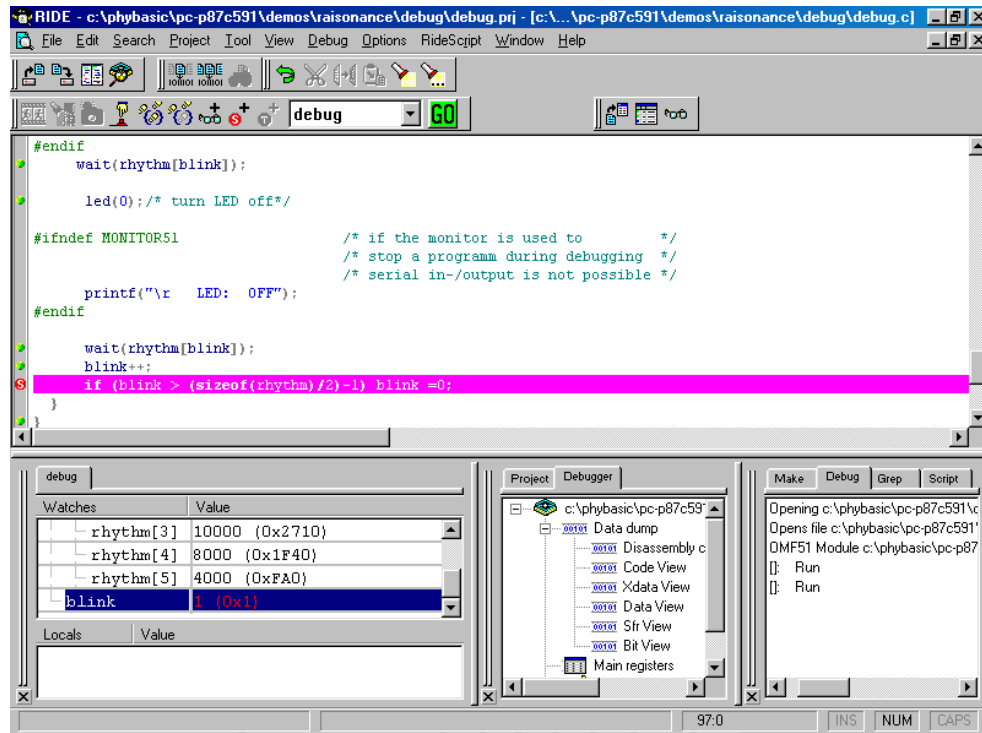


- Click on the green icon in front of the code line *if(blink >...)* to set a breakpoint here.
- The red marker on the left-hand side of the selected line indicates the breakpoint.




- Click on the Go  icon and the program will run and stop at the breakpoint.
- Notice that the LED (D3) on the Development Board now goes off. This is because the *led(0)* function call has been executed.
- Also notice that the variable *blink* in the Watch window has changed its value to '1(0x1)'.
(Note: The screenshot shows the value as 0, but the text description says it changes to 1. This is likely a typo in the original document.)

- The code line `if(blink >...)` is now highlighted in pink color.
- Repeat clicking on the Go **GO** icon and watch the variable `blink` and the LED D3 on the Development Board.



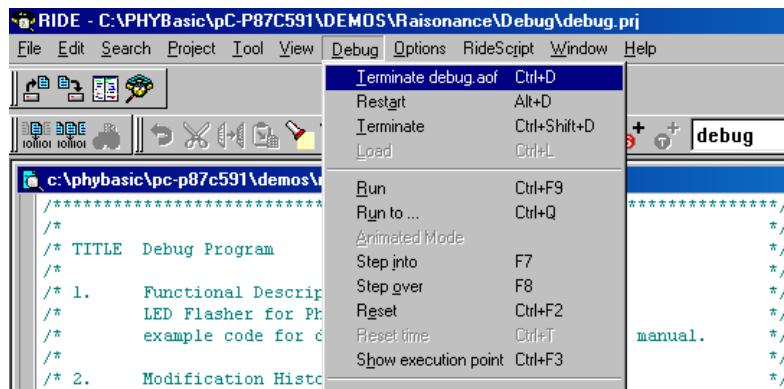
- Click on red marker on the left-hand side of the selected code line to remove the breakpoint.

4.7 Running, Stopping and Resetting

- To run your program without stopping at any time, delete all breakpoints by clicking on the red icon in front of the code line.
- Click on the Go  icon.

The LED now blinks with alternating on and off durations.

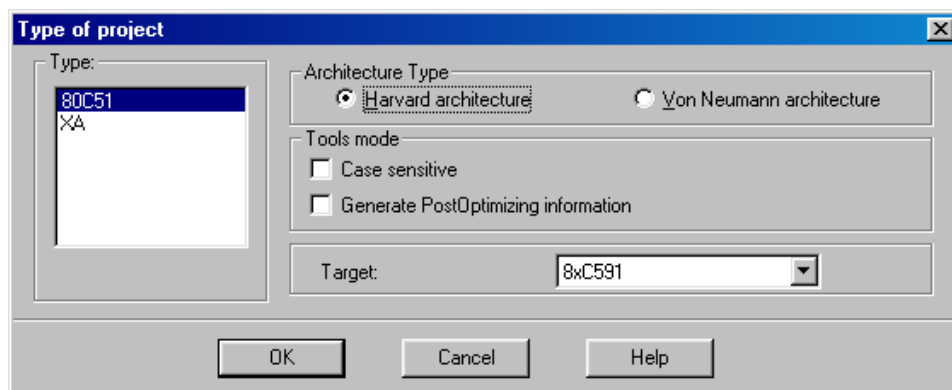
- To exit the current debug session go to the *Debug* menu and click on *Terminate debug.aof*.




4.8 Changing Target Settings for the "Final Version"

After successfully debugging the program, next change the target settings in order to create an Intel hexfile. This can then be downloaded to the Flash memory of the phyCORE-P87C591.

- Open the *Options/Target* menu and select the *Harvard architecture* as shown below:



- Click on *OK* to save this setting.
- Open the *Options/Project/RC51* menu and choose *Defines*. Delete the **MONITOR51** define in the *Defines* input field. This will include various *printf* statements in the application program that can be viewed with a terminal emulation program. Use of the *printf* statements is now possible because the serial interface is no longer required for other communication tasks.
- Click on *OK* to save this setting.
- Open the *Options/Project/LX51* menu and choose *Linker*.
- Enable the *Generate an Intel Hex file* checkbox.
- In the *Options/Project/LX51* menu disable the checkbox *ROM-Monitor*.
- The linker/locator options are now suitable for the *Debug* project, enabling you to build an absolute object file (*.aof) and a hexfile.
- Click on the 'Make All' Command icon  from the RIDE toolbar or open the *Project* menu and select *Build All* or *Make All*.

- Download the created ***Debug.hex*** file (located in ***C:\PHYBasic\pC-P87C591\Demos\Raisonance\Debug***) to the Flash memory. *For general download procedure information refer to sections 2.2 through 2.4.*
- Press the Reset button S2 on the Development Board to start the program.
- The application is now waiting for receipt of a known character over the serial interface. Start the HyperTerminal program and push the <Space> bar as described in *section 2.4.2*. This starts the automatic baud rate detection. Now you can watch your final debug example execute.

5 Advanced User Information

This section provides advanced information for successful operation of the phyCORE-P87C591 in conjunction with the Raisonance tools.

5.1 FlashTools98

Flash is a highly functional means of storing nonvolatile-data. One of its advantages among many others is the possibility of on-board programming. Programming tools for the Flash device are always included with the phyCORE-P87C591 in the form of a pre-programmed Flash with a resident microcontroller firmware and a counterpart software serving as the user interface on a host-PC. Once the firmware communicates with the PC-based software, FlashTools98 allows the download of user code from the host-PC into the Flash. Additionally, the re-programmable Flash device on the phyCORE-P87C591 allows you to easily update your own code and the target application in which the phyCORE-P87C591 has been implemented.

Currently the phyCORE-P87C591 can be populated by two different sized Flash devices: a 29F010 with 128 kByte or a 29F040 with 512 kByte. To support the entire memory area of these devices the address decoder of the phyCORE-P87C591 is equipped with an integrated banking mechanism that allows code-bank switching in code-banks of 64 kByte each.

Please note that the FlashTools98 kernel always occupies the first 64 kByte bank (bank 0, FA[18..15] = 0000b) of the Flash memory. This bank is pre-programmed upon delivery of the phyCORE-P87C591. The remaining banks are available to house your application. This makes one user application bank available if the phyCORE-P87C591 is mounted with a 29F010 and seven user application banks if the phyCORE-P87C591 is mounted with a 29F040 Flash memory device. Multiple user application banks can easily be managed by using the Code Banking mechanism of the Raisonance tool chain.

The following description is valid only for the FlashTools98 included with the phyCORE-P87C591 and is not intended as a guideline for using any other program.

FlashTools98 incorporates a safety mechanism that ensures that the system bank (bank 0), in which the firmware is resident, can not be overwritten during programming of the available user banks of the Flash device.

Resetting the phyCORE-P87C591 also activates the system bank (bank 0) of the Flash device, which automatically starts the FlashTools98 firmware. Then the firmware either enters the Flash programming mode or starts your user application.

To distinguish between download and execution modes, the firmware latches the /BOOT signal after reset (/BOOT=0 => start Flashtools, /BOOT=1 => start user program). This signal can be set to a low level by pressing the Boot (S1) button located on the phyCORE Development Board LD 5V. To enter the Flash programming mode you must simultaneously press the Reset (S2) and the Boot (S1) button, release the Reset (S2) button first and then, two to three seconds later, release the Boot (S1) button.

Execution of your user application will always start in the second 64 kByte bank (bank 1, FA[18..15] = 0010b). This is to be noted when preparing a software copy of the contents of the address decoder's internal write-only registers.

The extended features of the address decoder on the phyCORE-P87C591 allows flexibility when configuring the memory model according to your needs and addressing additional Flash banks.

Do not use Flash bank 0 in your application program in order to preserve the FlashTools98 microcontroller firmware and the associated Flash re-programming capability.

5.2 Linking and Locating

The Linker must combine several relocatable object modules contained in object files and/or libraries to generate a single absolute object.

In addition, the linker must locate several segments of code and data to fixed address locations within the address space in regards to the memory types of the phyCORE-P87C591. XDATA segments always must be located to Random Access Memory (e.g. RAM), CODE segments should be located to non-volatile memory (e.g. Flash). The 8051 family supports a Harvard memory architecture that distinguishes between non-volatile and randomly accessible memory and has two physically different signals for separate fetching of data and code.

The Raisonance tool chain distinguishes the following segment types:

- CODE: code
- XDATA: external data (max. 64 kByte)
- DATA: direct addressable on-chip data (max. 128 Byte)
- IDATA: indirect addressable on-chip data (max. 256 Byte)
- BIT: bit-addressable on-chip data (max. 128-bits)

The segment types DATA, IDATA and BIT always reside in the on-chip RAM of the controller.

The segment types XDATA and CODE will usually reside in external memory devices.

To ensure proper execution of your application, it is required that all XDATA segments are located to the external RAM of the phyCORE-P87C591 and that all CODE segments are located to the external Flash memory of the phyCORE-P87C591. Exceptions may occur if you use a 8051 derivative with on-chip portions of XDATA (e.g. internal XRAM) or CODE (e.g. internal ROM).

Since the phyCORE-P87C591 is equipped with a software configurable address decoder instead of simple programmable logic device, you can configure the memory model to your needs at runtime.

To ensure proper execution of your application, you must take the runtime memory model into consideration when linking and locating. This means that you must instruct the linker where to assume external RAM for locating data segments and Flash for locating code segments.

The standard configuration of the phyCORE-P87C591 is equipped with 128 kByte of external RAM and 128 kByte of external Flash. During runtime the RAM will be addressable at 0x0000 to 0xFFFF. The user bank (bank 1, FA[18..15] = 0010b) will be addressable at 0x0000 to 0xFFFF. This default runtime memory model requires no additional linker settings because both RAM and Flash start at 0x0000. This is also the default start address of the linker's segment types.

Since you can not define any end address, you should always ensure that the size of the segments fits within the available size of the mounted memory devices. For instance all XDATA segments should end below 0x7FFF if a 32 kByte RAM device is mounted on the phyCORE-P87C591. We recommend generation of a **.m51* map file for your project and inspection of the memory map information within this file.

Whenever you modify the memory model (e.g. use von Neumann rather than Harvard memory), which leads to different start addresses of CODE or XDATA memory, you must configure this in the linker settings.

Document: phyCORE-P87C591 QuickStart Instructions
Document number: L-586e_2, July 2002

How would you improve this manual?

Did you find any mistakes in this manual? _____ page

Submitted by:

Customer number: _____

Name: _____

Company: _____

Address: _____

Return to:

PHYTEC Technologie Holding AG
Postfach 100403
D-55135 Mainz, Germany
Fax : +49 (6131) 9221-33

Published by

PHYTEC

© PHYTEC Meßtechnik GmbH 2002

Ordering No. L-586e_2
Printed in Germany