phyCORE-167

QuickStart Instructions

Using PHYTEC FlashTools 16W and the Tasking C166/ST10 Embedded Development Environment (EDE) evaluation version

Note: The PHYTEC Spectrum CD includes the electronic version of the English phyCORE-167 Hardware Manual

Edition: February 2003

	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

3rd Edition: February 2003

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 2
1.4 The PHYTEC phyCORE-167..4
1.5 The Tasking C166/ST10 Embedded Development Environment (EDE) 7
2 Getting Started...11
2.1 Installing Rapid Development Kit Software..............................11
2.2 Interfacing the phyCORE-167 to a Host-PC 18
2.3 Starting PHYTEC FlashTools 16W...20
2.4 Downloading Example Code with FlashTools 16W 21
2.4.1 "Blinky" 26
2.4.2 "Hello" 28
3 Getting More Involved 35
3.1 Starting the Tasking Tool Chain..35
3.2 "Blinky" 36
3.2.1 Creating a New Project and Adding an Existing Source File...36
3.2.2 Modifying the Source Code..40
3.2.3 Saving the Modifications..41
3.2.4 Setting Tool Chain Options 41
3.2.5 Building the Project..44
3.2.6 Downloading the Output File 45
3.3 "Hello" 47
3.3.1 Creating a New Project...47
3.3.2 Modifying the Example Source....................................48
3.3.3 Setting Tool Chain Options 48
3.3.4 Building the New Project 48
3.3.5 Downloading the Output file 49
3.3.6 Starting the Terminal Emulation Program....................50

4 Debugging 51
4.1 Creating a Debug Project and Preparing the Debugger 51
4.2 Preparing the Target Hardware to Communicate with Cross View Pro 53
4.3 Starting the Debugger 54
4.4 Using the Tasking CrossView Pro Debug Features 55
Breakpoints 55
4.4.2 Data Window 55
4.5 Single Stepping 56
4.5.1 Memory Window 56
4.5.2 Stack Window 57
4.5.3 Simulated I/O 57
4.6 Resetting CrossView Pro and the phyCORE-167 58
5 Advanced User Information 59
5.1 FlashTools 16W 59
5.2 Start.asm 60
5.3 Linking and Locating 61
5.4 Debugging Using Monitor Kernel 63
5.5 Using ROM Debug Monitor in Flash Memory 63
Appendices A: Troubleshooting 65
A.1. Initialization Error: ROM Monitor DLL could not Initialize or Access the Target Board 65
A.2 Program Download Never Stops 65
A.3 Placing Code Breakpoint at 0xXXXXXX Failed: Memory not Writeable 66

Index of Figures

Figure 1: phyCORE Development Board HD200 Overview 18
Figure 2: Default Setting for the phyCORE Development Board HD200 19
Figure 3: Power Connector 19

1 Introduction to the Rapid Development Kit

This QuickStart provides:

- general information on the PHYTEC phyCORE-167 Single Board Computer (SBC),
- an overview of Tasking's C166/ST10 Embedded Development Environment (EDE) evaluation version, and
- instructions on how to run example programs on the phyCORE-167, mounted on the PHYTEC phyCORE Development Board HD200, in conjunction with Tasking software tools

Please refer to the phyCORE-167 Hardware Manual for specific information on board-level features such as jumper configuration, memory mapping and pin layout. Selecting the links on the electronic version of this document takes you to the applicable section of the phyCORE-167 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-167 Hardware Manual and Development Board 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-167 in conjunction with the Tasking C166/ST10 Embedded Development Environment (EDE)

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-167 in conjunction with Keil µVision2. It is structured as follows:

1) The *"Getting Started"* section uses two example programs *Blinky* and *Hello* to demonstrate the download of user code to the Flash device using PHYTEC FlashTools 16W.
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-167 using the Tasking tools and FlashTools 16W.
3) The *"Debugging"* section provides a third example program - *"Debug"* - to demonstrate simple debug functions using the Tasking debug environment.

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

1.3 System Requirements

Use of this "Rapid Development Kit" requires:

- the PHYTEC phyCORE-167
- the phyCORE Development Board HD200 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/NT)

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

TASKING
Embedded software development from Altium™

http://www.tasking.com
support_de@tasking.com (Germany)
support@tasking.com (International)

1.4 The PHYTEC phyCORE-167

The phyCORE-167 represents an affordable yet highly functional Single Board Computer (SBC) solution in sub-miniature dimensions (60 x 53 mm). It is intended for use in memory-intensive applications running within a CAN bus system. The standard board is populated with an Infineon SAB C167CR controller as well as a Real-Time Clock which, like the SRAM, can be buffered by an external battery.

All applicable data/address lines and signals extend from the underlying logic devices to two high-density 100-pin Molex SMT pin header connectors (pin width is 0.635 mm) lining the circuit board edges. This enables the phyCORE-167 to be plugged like a "big chip" into target hardware.

The standard module runs at a 20 MHz internal clock speed (delivering 100 ns instruction cycle) and offers 256 kByte (up to 1 MByte) SRAM and 256 kByte (up to 2 MByte) Flash on-board for DATA and CODE storage.

The module communicates by means of an RS-232 transceiver and a CAN bus interface. An additional UART with an RS-232 transceiver provides the second asynchronous serial interface. A second CAN bus interface is optionally obtainable if the phyCORE module is populated with an Infineon C167CS controller. The phyCORE-167 operates within a standard temperature range of 0 to +70°C and requires only a 220 mA power source. Modules are also available with extended temperature range from -40 to 85 °C.

PHYTEC FlashTools 16W enables easy on-board download of user programs.

phyCORE-167 Technical Highlights

- SBC in credit card-size dimensions (60 x 53 mm) achieved through modern SMD technology
- populated with an Infineon SABC167CR controller featuring on-chip 2.0B Full-CAN and operating in 16-bit, non-multiplexed bus mode at 20 MHz CPU speed (100 ns / instruction cycle)
- 256 kByte (up to 1 MByte) external SRAM (up to 1 MByte can be buffered with an optional lithium battery)[1]
- 256 kByte (up to 2 MByte) external Flash memory supporting on-board download of user code from a host-PC in conjunction with PHYTEC FlashTools 16W[1]
- no dedicated programming voltage required through use of 5 V Flash-devices
- battery-buffered Real-Time Clock
- one serial interfaces via RS-232
- optional a UART as a second asynchronous serial interface
- 16-channel A/D converter with 10-bit resolution
- second 2.0B Full-CAN interface (C167CS only)
- requires a single power supply of 5 V/ <220 mA

[1]: Please contact PHYTEC for more information about additional module configurations.

The phyCORE Development Board HD200, 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 most PHYTEC phyCORE high-density series Single Board Computers. Simple jumper configuration readies the Development Board's connection to the phyCORE-167, which plugs into the receptacle contact strips mounted on the Development Board HD200.

phyCORE Development Board HD200 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
- simple jumper configuration allowing use of the phyCORE Development Board HD 200 with various PHYTEC phyCORE high-density SBC's
- RJ45 Ethernet transformer module
- 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 Tasking C166/ST10 Embedded Development Environment (EDE)

The Tasking software tool chain fully supports the entire Infineon C16X and ST Microelectronics ST10 microcontroller family. It includes a C/C++/EC++ compiler, macroassembler, Linker/Locator and the CrossView Pro Simulator and ROM Monitor Debugger within the EDE development environment. Specific chips supported are the Infineon 161, 163, 164, 165, 166, 167, and ST Microelectronics ST10/262. Future derivatives are easily accommodated due to the flexible Tasking C compiler design.

The Tasking software tool chain supports in-circuit emulators that adhere to the IEEE-695 debugging specification or support the Tasking a.out format. The IHEX166 Object-to-Hex converter converts an absolute object file into an Intel-hexfile that is suitable for programming into an EPROM device or downloading into external Flash on the PHYTEC phyCORE-167 target board.

The Tasking software tool chain consists of the following executables*:*

• **C++/EC++ Comp.**	cp166.exe	(not in eval version)
• **C Compiler**	c166.exe	
• **Macro Preprocessor**	m166.exe	
• **Assembler**	a166.exe	
• **Linker/Locator**	l166.exe	
• **HEX converter**	ihex166.exe	
• **Make Utility**	mk32.exe	
• **Steering Program**	cc166.exe	
• **IEEE-695 converter**	ieee166.exe	
• **SREC converter**	srec166.exe	(not in eval version)
• **Disassembler**	d166.exe	(not in eval version)
• **Object dumper**	dmp166.exe	(not in eval version)
• **Library archiver**	ar166.exe	(not in eval version)
• **CrossView Pro**	xfw166.exe	(Windows-based)
• **EDE**	ede.exe	(Windows-based)

Once installed, the default destination location for these files is the ***C:\DC166\bin*** for the evaluation version. If using the professional (i.e. full) version of the Tasking software tool chain, the default destination location for these files is the ***C:\C166\bin***. The executables are 32-bit Windows programs and run in command line mode except for CrossView Pro and EDE which are Windows-based applications. Access to these programs from Windows is done with EDE. The entire tool set can be run from EDE or directly from a command line using “batch” or “make” files. The limitations of the evaluation version are listed in the ‘Demo Package Readme’ file. Other than these restrictions, all features operate normally.
The Tasking tools are also available for the UNIX operating system.

Embedded Development Environment (EDE)

EDE is a Windows-based Graphical User Interface (GUI)for the C compiler and assembler portion of the Tasking software tools. All compiler, assembler and linker options are set with simple mouse clicks. EDE runs under Windows 95/98/2000 and NT. This Embedded Development Environment has been designed with the user in mind and includes a fully functional editor based on the popular Codewright from Premia.

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

C166 C Compiler

The C166 ANSI compiler and A166 assembler are designed specifically for the Infineon SAB 161, 163, 164, 165, 166, 167, and ST Microelectronics ST10/262 and future derivatives.

The Tasking C166 compiler provides the fastest and smallest code using industry benchmarks.

M166 Macro Preprocessor/A166 Assembler

The macro preprocessor and the assembler are started by the EDE environment. It is also possible to pass a file directly to the assembler when it does not need any macro expansion.

Debug Environment

CrossView Pro is a software simulator/ROM monitor debugger supporting debugging either via software on a host-PC or in target hardware. CrossView Pro enables the following debugging functions:

- run/halt,
- set (complex) breakpoints,
- examine/change memory,
- view the stack,
- simulated I/O
- extended logging and scripting language
- powerful C like command language
- communication via RS-232 or CAN

The CrossView Pro Simulator supports code coverage and profiling to ensure your code runs efficiently, and high level language trace. The restrictions on evaluation version of the Tasking C166 tool set are described in the 'Demo Package Readme'. Other than these restrictions the evaluation tool chain functions exactly as does the full version. The evaluation version does not have a starting address restriction and produces useful object code. This allows you to fully evaluate the features and power of Tasking products on a PHYTEC target board. The full version has no restrictions and is fully ANSI compliant.

CAN (Controller Area Network) Library

The Infineon CAN protocol driver software routines including pre-built CAN libraries are supplied with the 32-bit Windows 95/NT version of this product. The file ap292201.pdf describes the usage of these libraries. This file is located in the pdf folder on the Spectrum CD.

2 Getting Started

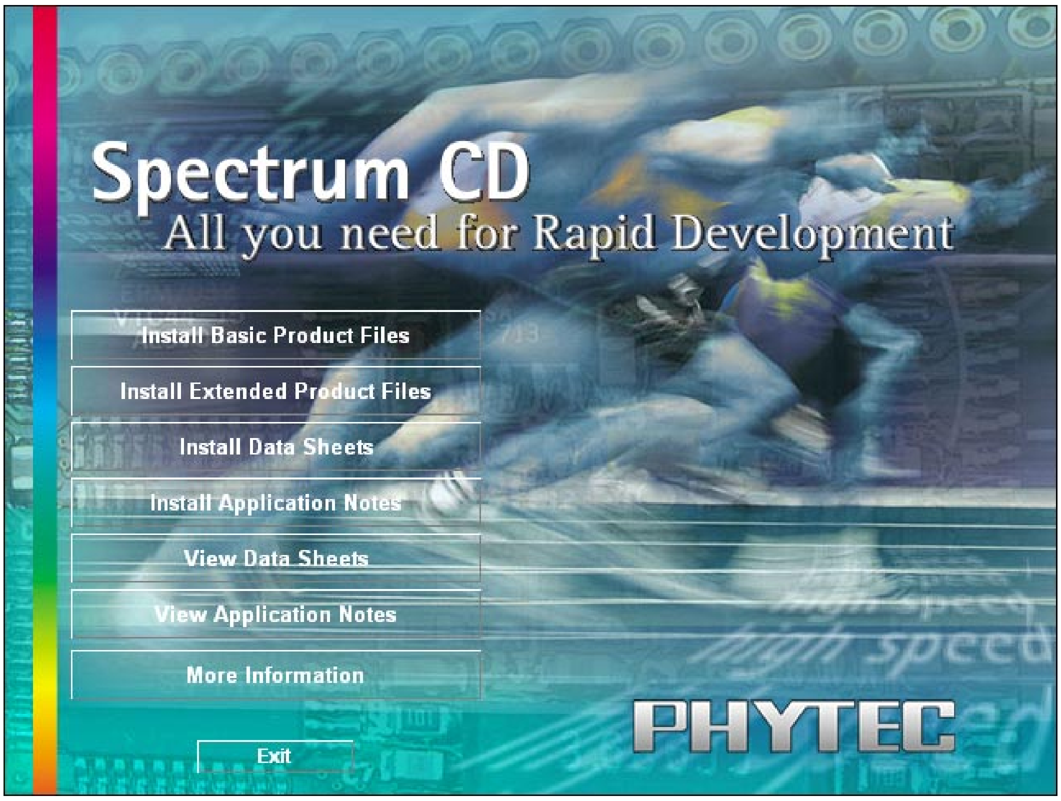
What you will learn with this Getting Started example:

- installing Rapid Development Kit software
- starting PHYTEC's FlashTools 16W download utility
- interfacing the phyCORE-167, mounted on the Development Board, to a host-PC
- downloading example user code in hexfile format from a host-PC to the external Flash memory using FlashTools 16W

2.1 Installing Rapid Development Kit Software

When you 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.

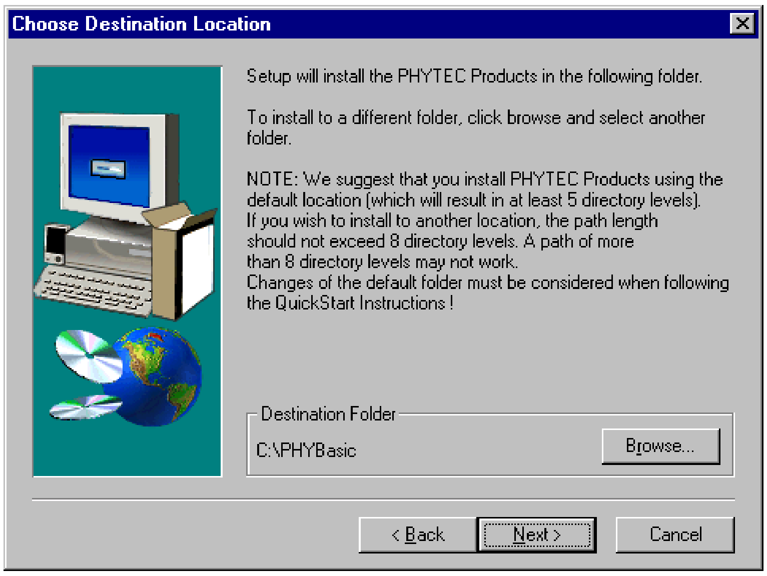
The following window appears:


Spectrum CD
All you need for Rapid Development
Install Basic Product Files
Install Extended Product Files
Install Data Sheets
Install Application Notes
View Data Sheets
View Application Notes
More Information
Exit
PHYTEC

- Choose *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.


Choose Destination Location

Setup will install the PHYTEC Products in the following folder.

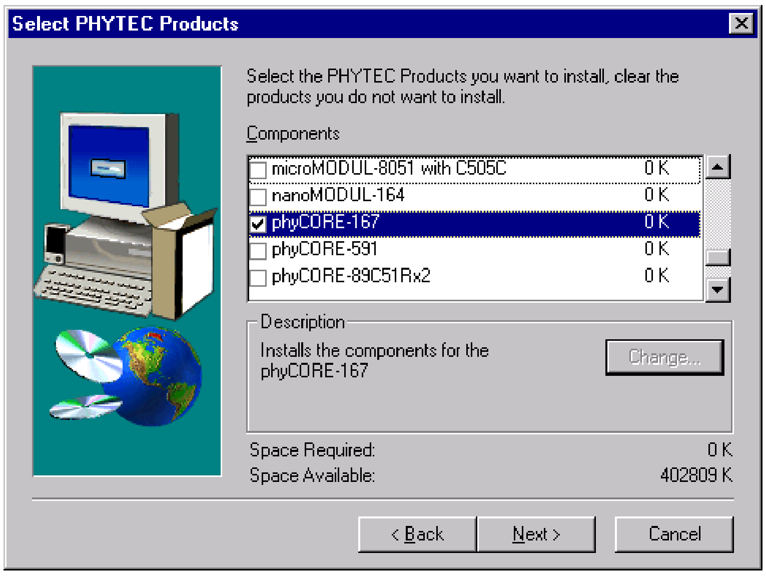
To install to a different folder, click browse and select another folder.

NOTE: We suggest that you install PHYTEC Products using the default location (which will result in at least 5 directory levels). If you wish to install to another location, the path length should not exceed 8 directory levels. A path of more than 8 directory levels may not work. Changes of the default folder must be considered when following the QuickStart Instructions !

Destination Folder

C:\PHYBasic

- 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.


Select PHYTEC Products

Select the PHYTEC Products you want to install, clear the products you do not want to install.

Components

☐ microMODUL-8051 with C505C	0 K
☐ nanoMODUL-164	0 K
☑ phyCORE-167	0 K
☐ phyCORE-591	0 K
☐ phyCORE-89C51Rx2	0 K

Description

Installs the components for the phyCORE-167

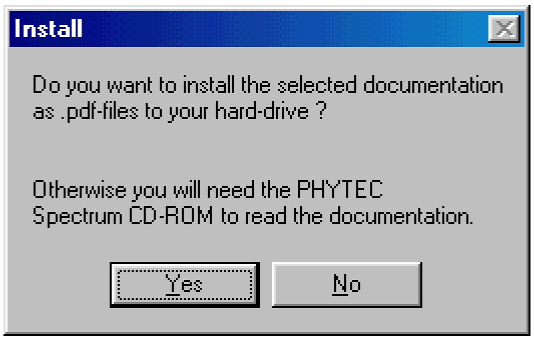
Space Required: 0 K

Space Available: 402809 K

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

All software and tools for this phyCORE-167 Kit will be installed to the ***\PHYBasic*** directory 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.


Install

Do you want to install the selected documentation as .pdf-files to your hard-drive ?

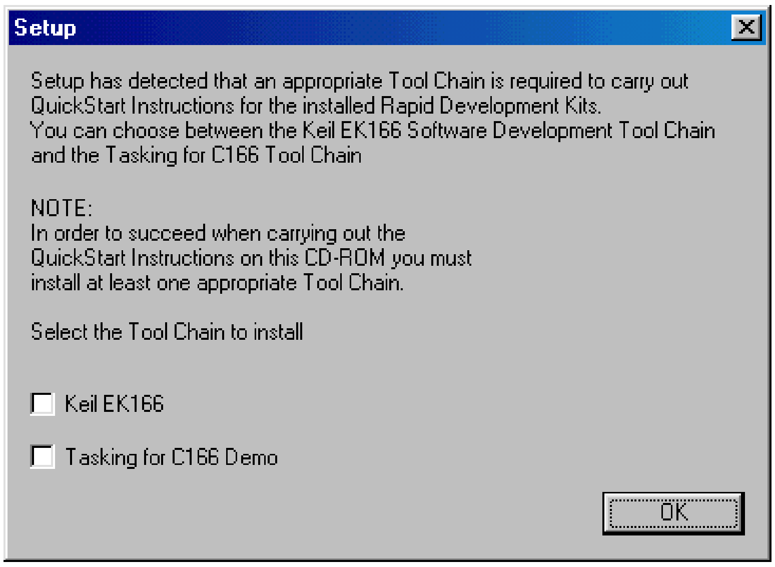
Otherwise you will need the PHYTEC Spectrum CD-ROM to read the documentation.

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-167 Kit will also be installed to the Kit-specific subdirectory.

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

- In the next window, choose Tasking for C166 Demo Software Development Tool Chain.



Setup

Setup has detected that an appropriate Tool Chain is required to carry out QuickStart Instructions for the installed Rapid Development Kits.
You can choose between the Keil EK166 Software Development Tool Chain and the Tasking for C166 Tool Chain

NOTE:
In order to succeed when carrying out the QuickStart Instructions on this CD-ROM you must install at least one appropriate Tool Chain.

Select the Tool Chain to install

☐ Keil EK166

☐ Tasking for C166 Demo

OK

After accepting the Welcome window and license agreement select the destination location for installation of the Development Tool chain.

Depending on the Rapid Development Kit software you have selected, the applicable Tasking Evaluation Development tool chain will be installed to your hard-drive. Additional software, such as Adobe Acrobat Reader, will also be offered for installation.

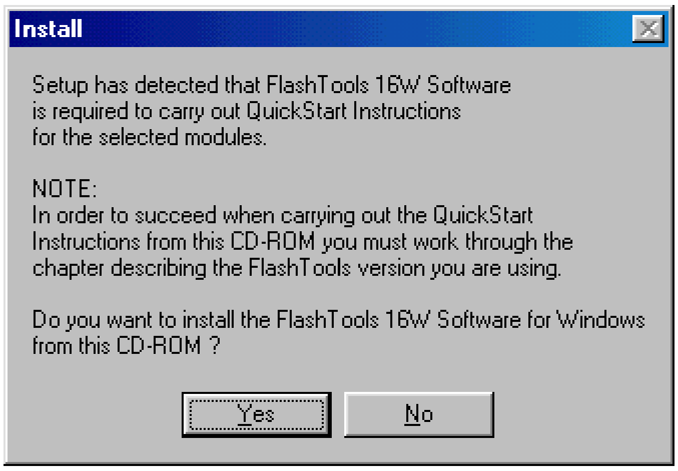
Caution:
Tasking Evaluation Development tool chain is limited for 30 days since the first installation.

The applicable Tasking 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.

To ensure that path configurations of the pre-made project match the installation of the tool chain use only the default path ***c:\dcc166*** during the installation procedure.

We recommend that you install Tasking C166/ST10 Embedded Development Environment (EDE) evaluation version from the Spectrum CD-ROM even if other versions of EDE are 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.

In the following windows you can decide to install FlashTools 16W software and the Acrobat Reader.

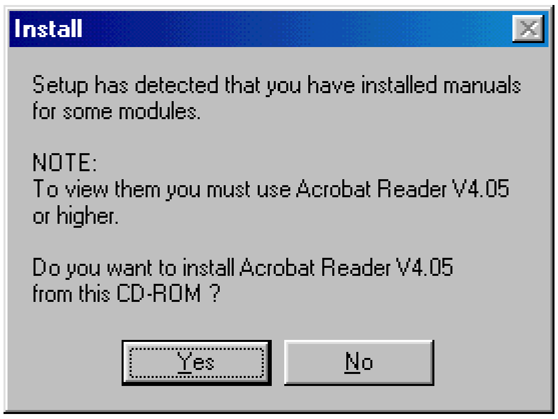


Install

Setup has detected that FlashTools 16W Software
is required to carry out QuickStart Instructions
for the selected modules.

NOTE:
In order to succeed when carrying out the QuickStart
Instructions from this CD-ROM you must work through the
chapter describing the FlashTools version you are using.

Do you want to install the FlashTools 16W Software for Windows
from this CD-ROM ?



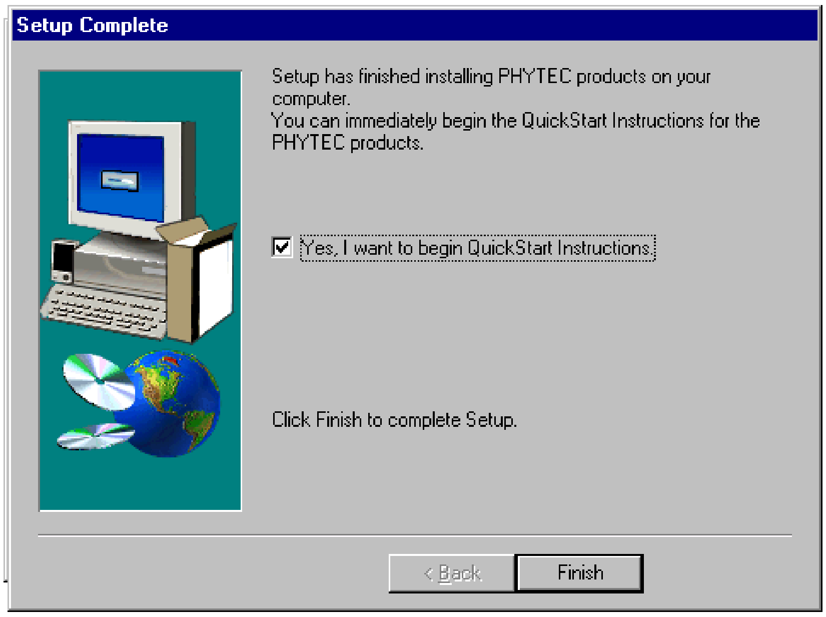
Install

Setup has detected that you have installed manuals
for some modules.

NOTE:
To view them you must use Acrobat Reader V4.05
or higher.

Do you want to install Acrobat Reader V4.05
from this CD-ROM ?

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



Setup Complete

Setup has finished installing PHYTEC products on your computer.
You can immediately begin the QuickStart Instructions for the PHYTEC products.

☑ Yes, I want to begin QuickStart Instructions.

Click Finish to complete Setup.

2.2 Interfacing the phyCORE-167 to a Host-PC

Connecting the phyCORE-167, mounted on the phyCORE Development Board HD200, to your computer is simple.

- As shown in the figure below, if the phyCORE module is not already pre-installed, mount it connector side down onto the Development Board’s receptacle footprint (X6).

Ensure that pin 1 of the phyCORE-connector (denoted by the hash stencil mark on the PCB) matches pin 1 of the receptacle on the phyCORE Development Board HD200.

Ensure that there is a solid connection between the Molex connector on the module and the Development Board receptacle.

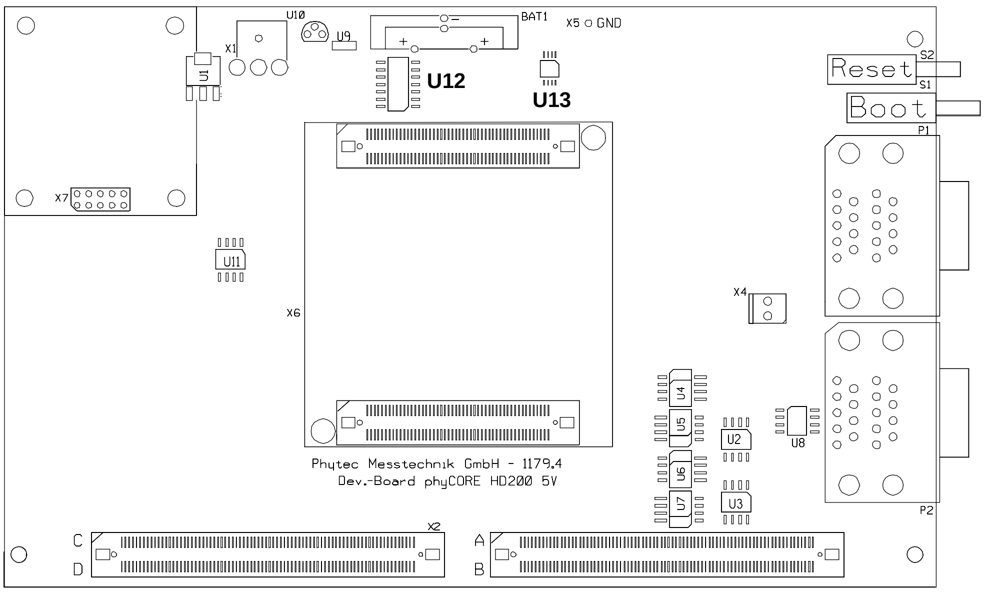


Figure 1: phyCORE Development Board HD200 Overview

- Configure the jumpers on the Development Board as indicated in *Figure 2*. This correctly routes the RS-232 signals to the DB-9 socket (P1A = bottom) and connects the Development Board's peripheral devices to the phyCORE module.

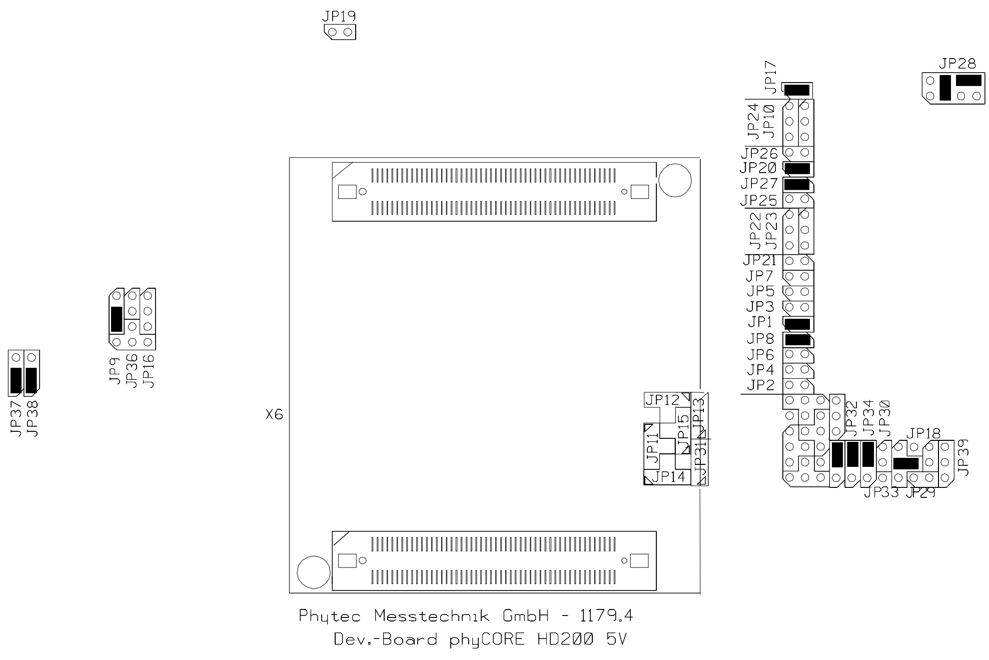


Figure 2: Default Setting for the phyCORE Development Board HD200

- Connect the RS-232 interface of your computer to the DB-9 RS-232 interface on the phyCORE Development Board HD200 (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*).

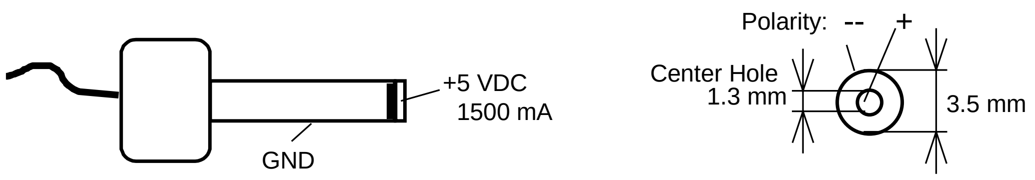


Figure 3: Power Connector

- Simultaneously press the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board HD200, 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) buttons renders the phyCORE-167 into the Bootstrap mode. Use of FlashTools 16W always requires the phyCORE-167 to be in Bootstrap mode. *See section 5.1, "FlashTools 16W" for more details.*

The phyCORE module should now be properly connected via the phyCORE Development Board HD200 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-167. This phyCORE module/phyCORE Development Board HD200 combination is also referred to as "target hardware".

2.3 Starting PHYTEC FlashTools 16W

FlashTools 16W 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 folder ***\Software\Ft16w\.***

FlashTools 16W for Windows is a utility program that allows download of user code in Intel ****.hex*** or ****.h86*** file format from a host-PC to a PHYTEC SBC via an RS-232 connection.

Proper connection of a PHYTEC SBC to a host-PC enables the software portion of FlashTools 16W to recognize and communicate to the Bootstrap loader on the phyCORE-167.

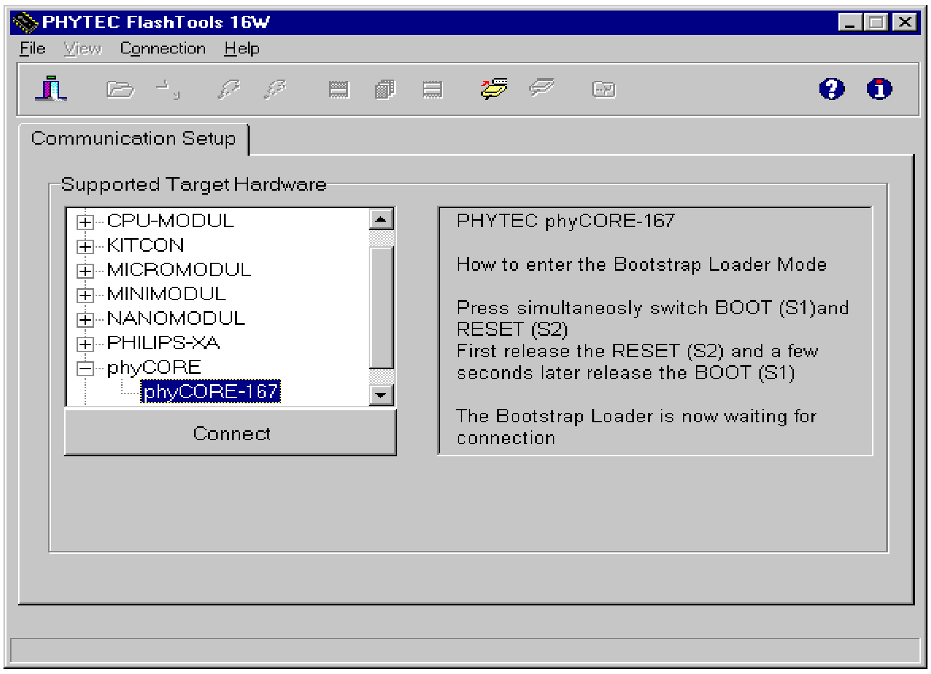
- You can start FlashTools 16W by selecting it from the *Programs* menu using the Windows *Start* button.

It is recommended that you drag the FlashTools 16W icon onto the desktop of your PC. This enables easy start of FlashTools 16W by double-clicking on the icon.

2.4 Downloading Example Code with FlashTools 16W

Start FlashTools 16W by double-clicking on the FlashTools 16W icon or by selecting *FlashTools 16W* from within the *Programs|PHYTEC* program group.

- The FlashTools 16W start window with the *Communication Setup* tab will now appear.
- Click on the + sign in front of the phyCORE string to expand the view and to see all available modules. Select the phyCORE-167 in the target hardware list.



PHYTEC phyCORE-167

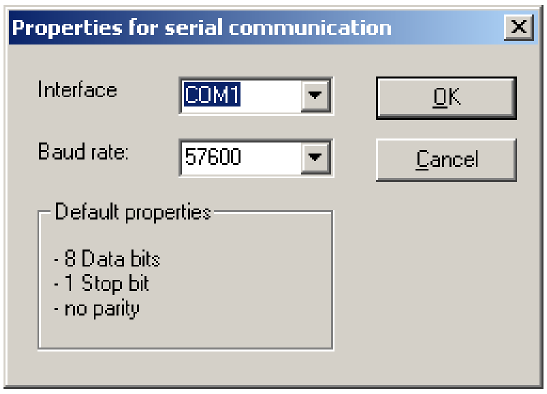
How to enter the Bootstrap Loader Mode

Press simultaneosly switch BOOT (S1)and RESET (S2)
First release the RESET (S2) and a few seconds later release the BOOT (S1)

The Bootstrap Loader is now waiting for connection

- Make sure the target hardware is in Bootstrap mode. Now click the *Connect* button.

- At the *Properties for serial communication* window of the FlashTools 16W tabsheet, choose the correct serial port for your host-PC and a 57,600 baud rate.



Properties for serial communication

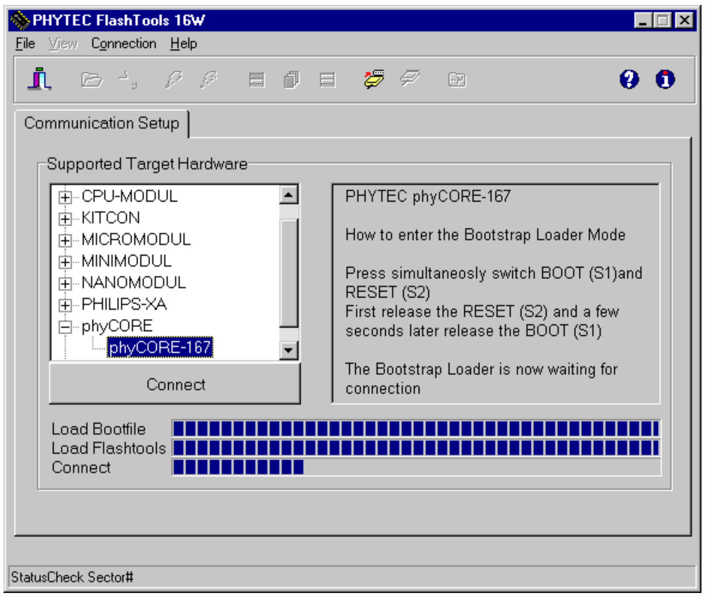
Interface: COM1

Baud rate: 57600

Default properties
- 8 Data bits
- 1 Stop bit
- no parity

- Click the *OK* 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 Bootstrap mode as described in *section 2.2.*



PHYTEC FlashTools 16W

Supported Target Hardware: CPU-MODUL, KITCON, MICROMODUL, MINIMODUL, NANOMODUL, PHILIPS-XA, phyCORE, phyCORE-167

PHYTEC phyCORE-167

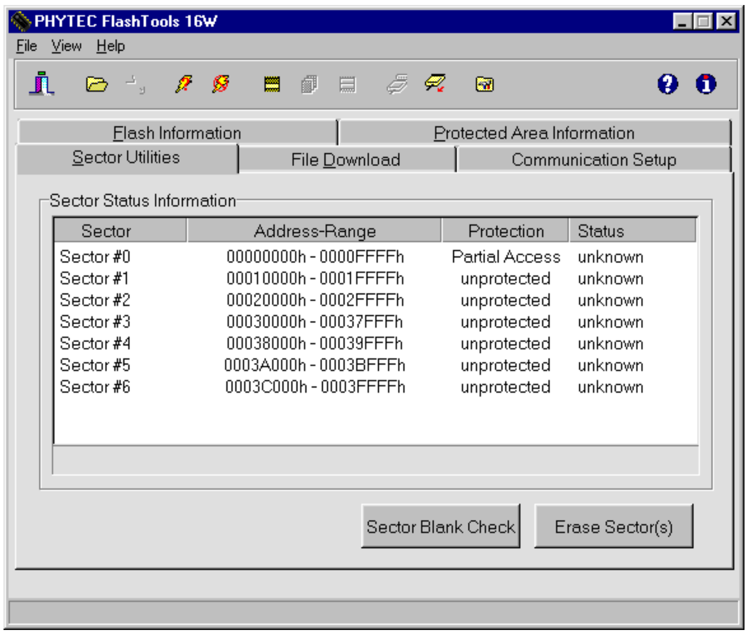
How to enter the Bootstrap Loader Mode

Press simultaneosly switch BOOT (S1)and RESET (S2)
First release the RESET (S2) and a few seconds later release the BOOT (S1)

The Bootstrap Loader is now waiting for connection

Returning to the FlashTools 16W tabsheet window, you will see tabs for the following:

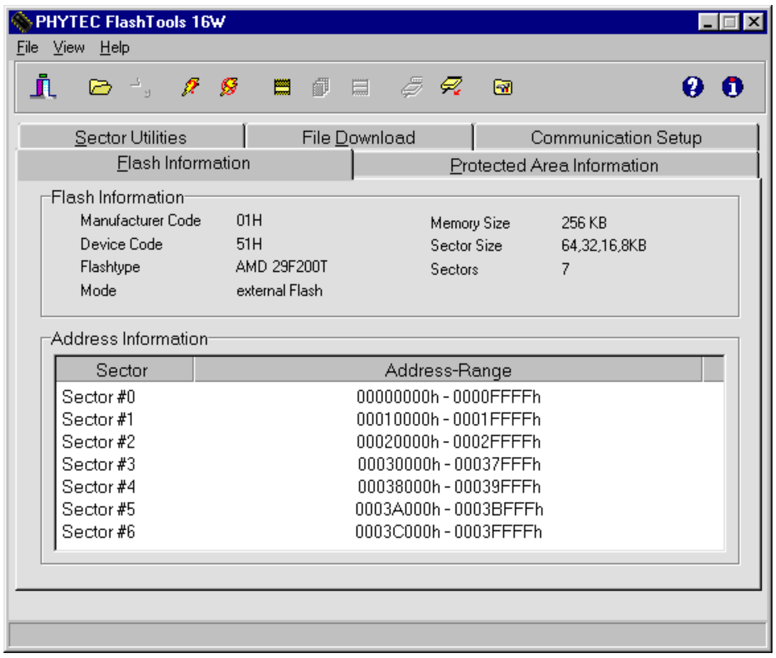
Sector Utilities[1] enable erasure and status check of individual sectors of Flash memory specified by the user:



Sector	Address-Range	Protection	Status
Sector #0	00000000h - 0000FFFFh	Partial Access	unknown
Sector #1	00010000h - 0001FFFFh	unprotected	unknown
Sector #2	00020000h - 0002FFFFh	unprotected	unknown
Sector #3	00030000h - 00037FFFh	unprotected	unknown
Sector #4	00038000h - 00039FFFh	unprotected	unknown
Sector #5	0003A000h - 0003BFFFh	unprotected	unknown
Sector #6	0003C000h - 0003FFFFh	unprotected	unknown

1 : The picture for the *Sector Utilities* tabsheet varies depending on the size and type of the Flash mounted on the phyCORE-167.

Flash Information[1] shows Flash type, sector and address ranges in Flash memory:



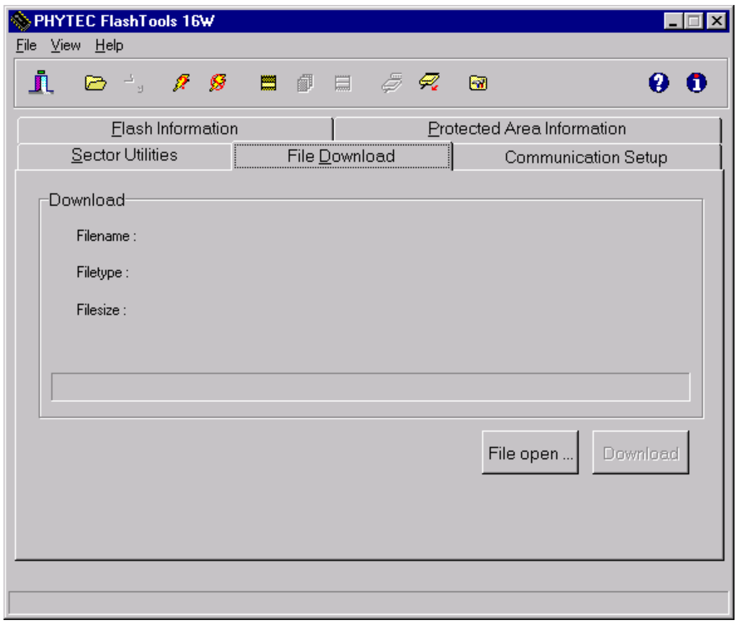
Flash Information

Manufacturer Code	01H	Memory Size	256 KB
Device Code	51H	Sector Size	64,32,16,8KB
Flashtype	AMD 29F200T	Sectors	7
Mode	external Flash		

Address Information

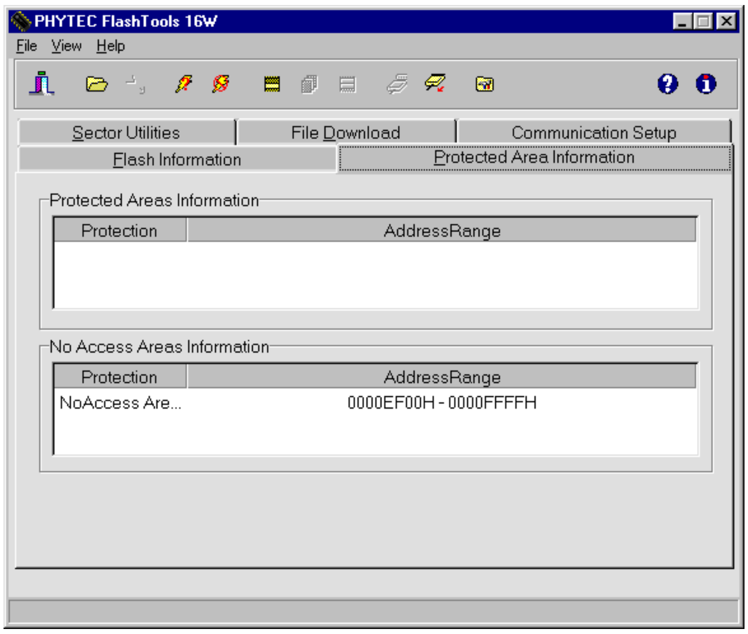
Sector	Address-Range
Sector #0	00000000h - 0000FFFFh
Sector #1	00010000h - 0001FFFFh
Sector #2	00020000h - 0002FFFFh
Sector #3	00030000h - 00037FFFh
Sector #4	00038000h - 00039FFFh
Sector #5	0003A000h - 0003BFFFh
Sector #6	0003C000h - 0003FFFFh

File Download downloads specified hexfiles to the target hardware:



1 : The appearance of the *Flash Information* tabsheet varies depending on the size and type of the Flash mounted on the phyCORE-167.

Protected Areas Information shows protected areas of Flash memory:


PHYTEC FlashTools 16W

File View Help

Sector Utilities | File Download | Communication Setup

Flash Information | Protected Area Information

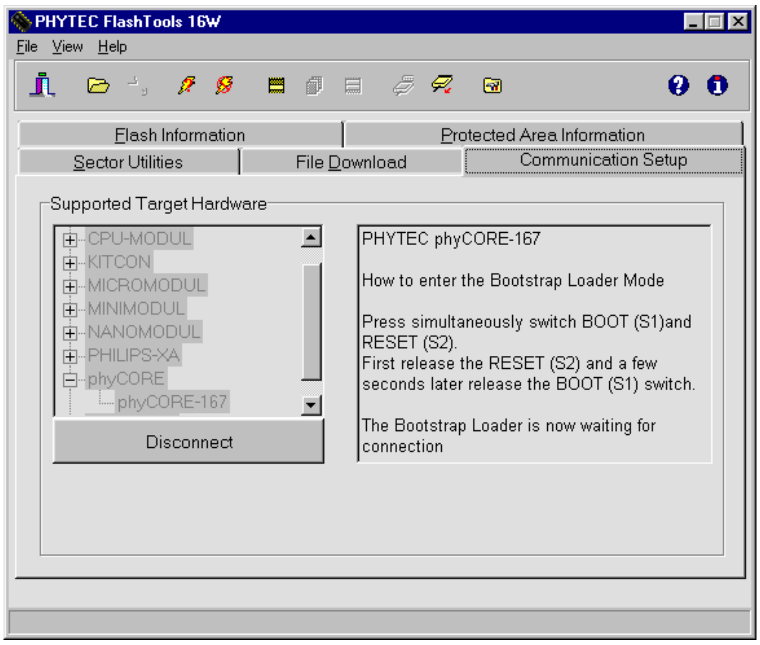
Protected Areas Information

Protection	AddressRange

No Access Areas Information

Protection	AddressRange
NoAccess Are...	0000EF00H - 0000FFFFH

Communication Setup allows connection to and disconnection from the target hardware (this is the same window that was used when you first entered FlashTools 16W):


PHYTEC FlashTools 16W

File View Help

Flash Information | Protected Area Information

Sector Utilities | File Download | Communication Setup

Supported Target Hardware

- CPU-MODUL
- KITCON
- MICROMODUL
- MINIMODUL
- NANOMODUL
- PHILIPS-XA
- phyCORE
  - phyCORE-167

Disconnect

PHYTEC phyCORE-167

How to enter the Bootstrap Loader Mode

Press simultaneously switch BOOT (S1)and RESET (S2).
First release the RESET (S2) and a few seconds later release the BOOT (S1) switch.

The Bootstrap Loader is now waiting for connection

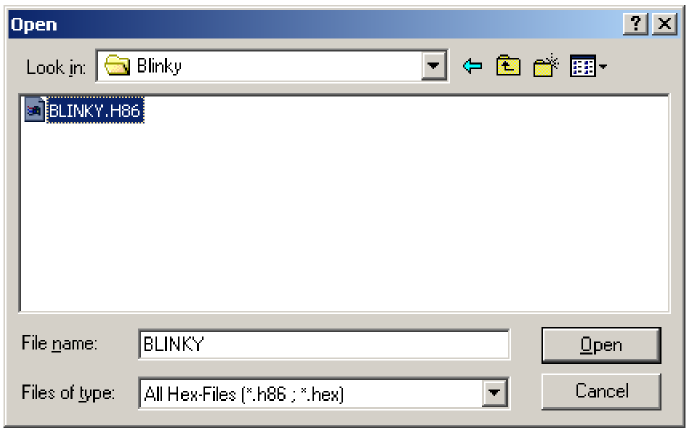
2.4.1 "Blinky"

The "Blinky" example downloads a program to the Flash that, when executed, manipulates the LED D3 on the phyCORE Development Board HD200 that is located above the jumper field (*refer to Figure 1*).

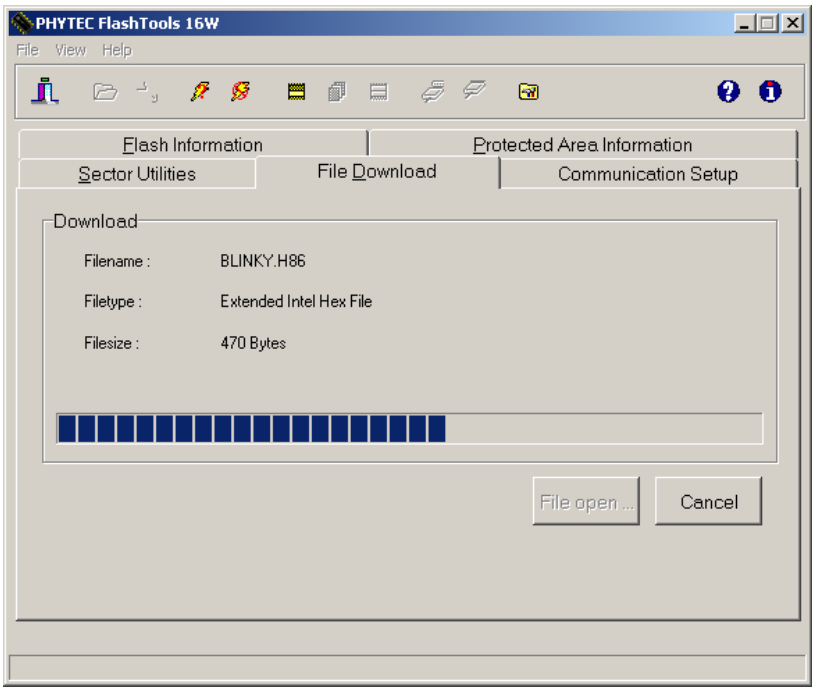
- Returning to the FlashTools 16W tabsheet, choose the *Download* tab and click on the *Open* button.

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

- Browse to the correct drive and path for the phyCORE-167 Demo folder (default location ***C:\PHYBasic\pC-167\Demos\Tasking\Blinky\Blinky.h86***) and click *Open.*



- Click on the *Download* button. You can watch the status of the download of the ***Blinky.h86*** into external Flash memory in the Download window.



Filename : BLINKY.H86
Filetype : Extended Intel Hex File
Filesize : 470 Bytes

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 *Erase Sector(s)* tab from the FlashTools 16W worksheet, highlight sector #0 and erase the sector. Then repeat the download procedure.

- Returning to the *Communication Setup* tab, click on the *Disconnect* button and exit FlashTools 16W.
- Press the Reset button (S2) on the phyCORE Development Board HD200 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 duration.

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 FlashTools 16W 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 Bootstrap mode by simultaneously pressing the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board HD200 and then releasing first the Reset and, two or three seconds later, the Boot button.
- Start FlashTools16W.
- The *Communication Setup* tab of the FlashTools 16W tabsheet window will now appear.
- Select the *phyCORE-167* as the target hardware and click on the *Connect* button.
- At the *Properties for serial communication* window of the FlashTools 16W tabsheet, choose the correct serial port for your host-PC and a 57,600 baud rate.
- Click the *OK* button to establish connection to the target hardware.
- Returning to the FlashTools 16W tabsheet, choose the *Sector Utilities* tab, highlight *Sector #0* in the *Sector Status Information* section of the tab and click on the *Erase Sector(s)* button to erase this memory location.
- Wait until the status check in the lower left corner of the FlashTools 16W 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.

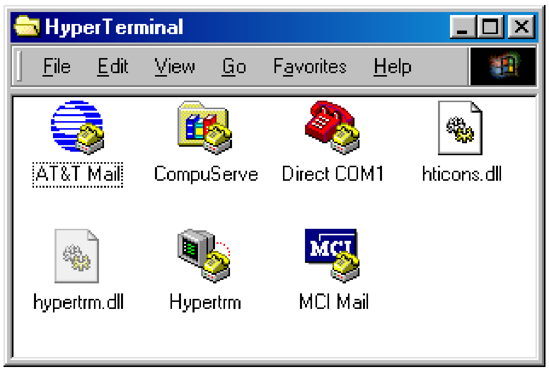
- Browse to the correct drive and path for the phyCORE-167 Demo folder (default location ***C:\PHYBasic\pC-167\Demos\Tasking\Hello\Hello.h86***) and click *Open.*
- Click on the *Download* button. You can watch the status of the download of the ***Hello.h86*** into external Flash memory in the Download window.

  If the selected Flash sector 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 *Sector Utility* tab from the FlashTools 16W tabsheet, highlight *Sector #0* and erase the sector. Then repeat the download procedure.

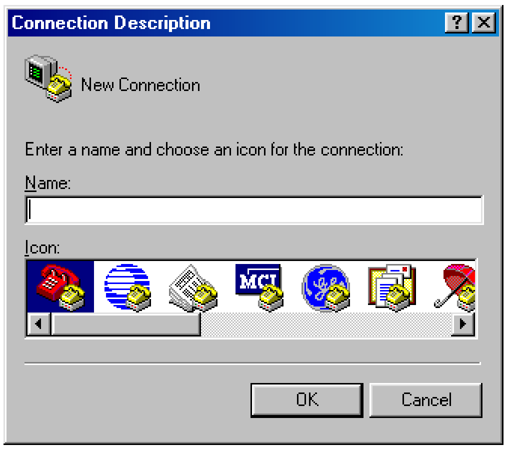
- At the end of the download, a sector-by-sector status check of the Flash memory can be viewed in the lower left corner of the FlashTools 16W tabsheet window. Wait until the status check finishes before returning to work with the board. Once the status check is complete, the downloaded code can be executed.
- Returning to the *Communication* tab, click on the *Disconnect* button and exit FlashTools 16W.

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 appear[1]:
- Double-click on the HyperTerminal icon "*Hypertrm*" to create a new HyperTerminal session.



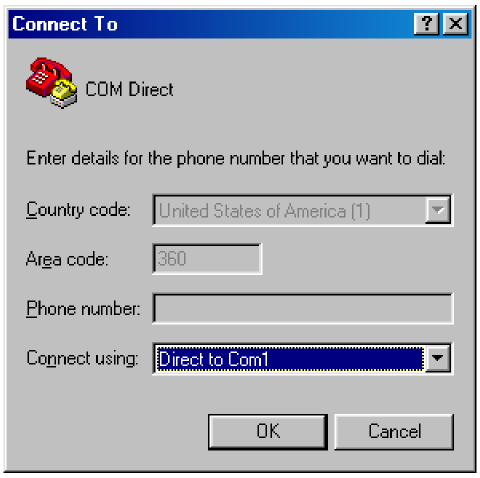
- The Connection Description window will now appear. Enter "COM Direct" in the *Name* text field.



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

[1] : 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).



COM Direct

Enter details for the phone number that you want to dial:

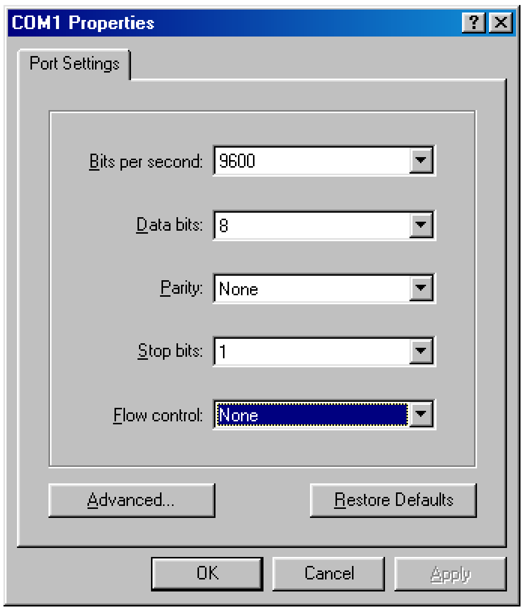
Country code: United States of America (1)

Area code: 360

Phone number:

Connect using: Direct to Com1

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



Bits per second: 9600

Data bits: 8

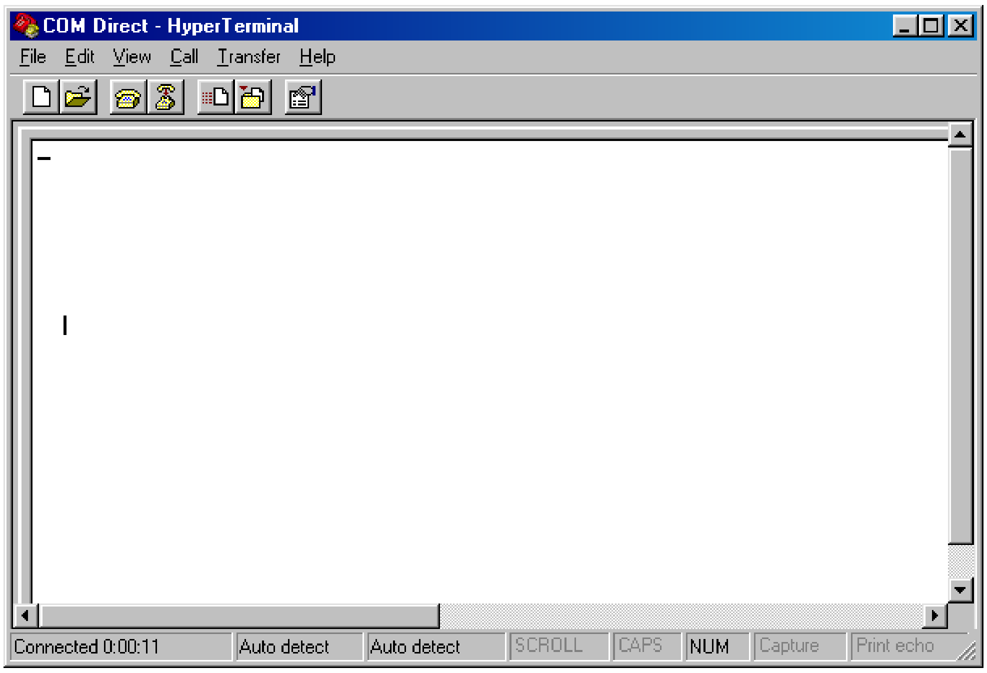
Parity: None

Stop bits: 1

Flow control: None

- 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 HD200 (at S2) will execute the ***Hello.h86*** file loaded into the Flash.
- Now push the <Space> bar on your keyboard once to start the automatic baud rate detection on phyCORE-167 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.

The demo application within the file ***Hello.h86*** initializes the serial port of your phyCORE-167 to 9600 baud. The initialization values are based on the assumption that the microcontroller runs at a 20 MHz internal clock frequency. Please note that an oscillator with a frequency of 5 MHz populates the phyCORE-167. Using the internal PLL (Phase Locked Loop) device results in an internal 20 MHz CPU frequency. If your phyCORE-167 is equipped with a different speed oscillator, the demo application might transmit using another baud rate. This may lead to incoherent characters appearing in the HyperTerminal window following execution of code.

You have now successfully downloaded and executed two pre-existing example programs in Intel ****.h86*** file format.

3 Getting More Involved

What you will learn with this example:

- how to start the Tasking software tool chain
- how to configure the Tasking tools within the C166/ST10 Embedded Development Environment (EDE)
- 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 Tasking Tool Chain

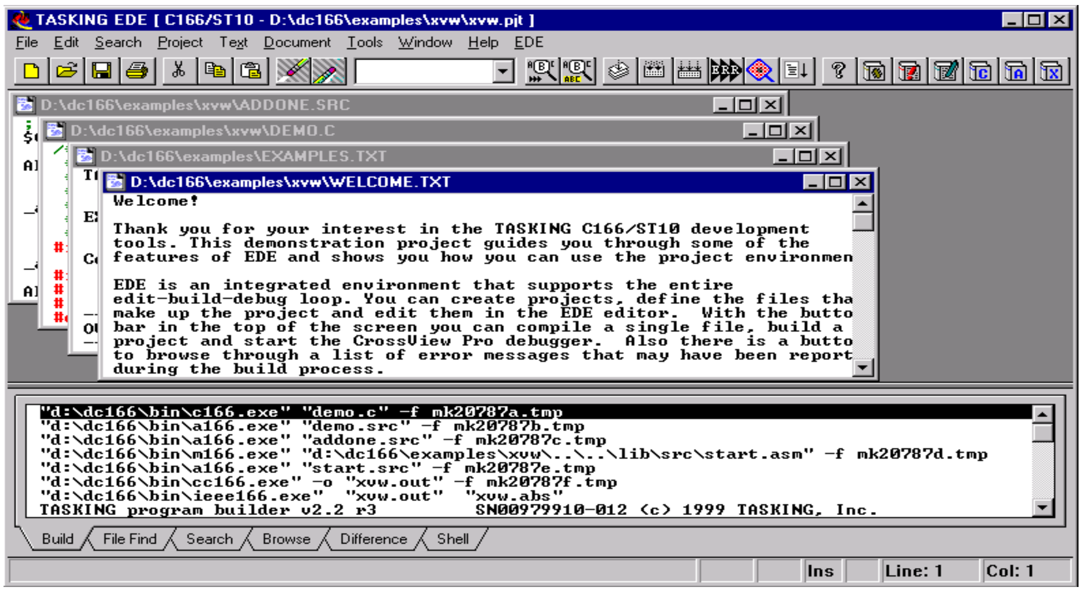
The Tasking C166/ST10 Embedded Development Environment (EDE) evaluation software should have been installed during the install procedure, as described in *section 0*

You can also manually install the tool chain by executing ***dc166.exe*** from within the ***\Software\Tasking\DC166\V6R6\Disk1*** folder of your PHYTEC Spectrum CD.

Caution:
You must use the Tasking tool chain provided on the accompanying Spectrum CD. In order to complete this QuickStart Instruction successfully. Use of a different version could lead to possible version conflicts, resulting in functional problems.

Start the tool chain by selecting *Embedded Development Environment* from within the *Programs* group.

After you start EDE, the window shown below appears. From this window you can create projects, edit files, configure tools, compile, assemble, link and start the debugger. Other 3rd party tools such as emulators can also be started from here.


TASKING EDE [C166/ST10 - D:\dc166\examples\xvw\xvw.pjt]
File Edit Search Project Text Document Tools Window Help EDE
D:\dc166\examples\xvw\ADDONE.SRC
D:\dc166\examples\xvw\DEMO.C
D:\dc166\examples\EXAMPLES.TXT
D:\dc166\examples\xvw\WELCOME.TXT
Build
File Find
Search
Browse
Difference
Shell
Ins
Line: 1
Col: 1

```
Welcome!

Thank you for your interest in the TASKING C166/ST10 development
tools. This demonstration project guides you through some of the
features of EDE and shows you how you can use the project environmen

EDE is an integrated environment that supports the entire
edit-build-debug loop. You can create projects, define the files tha
make up the project and edit them in the EDE editor.  With the butto
bar in the top of the screen you can compile a single file, build a
project and start the CrossView Pro debugger.  Also there is a butto
to browse through a list of error messages that may have been report
during the build process.
```

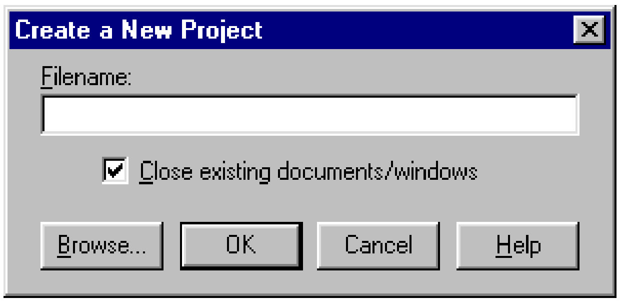
```
"d:\dc166\bin\c166.exe" "demo.c" -f mk20787a.tmp
"d:\dc166\bin\a166.exe" "demo.src" -f mk20787b.tmp
"d:\dc166\bin\a166.exe" "addone.src" -f mk20787c.tmp
"d:\dc166\bin\m166.exe" "d:\dc166\examples\xvw\..\..\lib\src\start.asm" -f mk20787d.tmp
"d:\dc166\bin\a166.exe" "start.src" -f mk20787e.tmp
"d:\dc166\bin\cc166.exe" -o "xvw.out" -f mk20787f.tmp
"d:\dc166\bin\ieee166.exe"  "xvw.out"  "xvw.abs"
TASKING program builder v2.2 r3        SN00979910-012 (c) 1999 TASKING, Inc.
```

3.2 "Blinky"

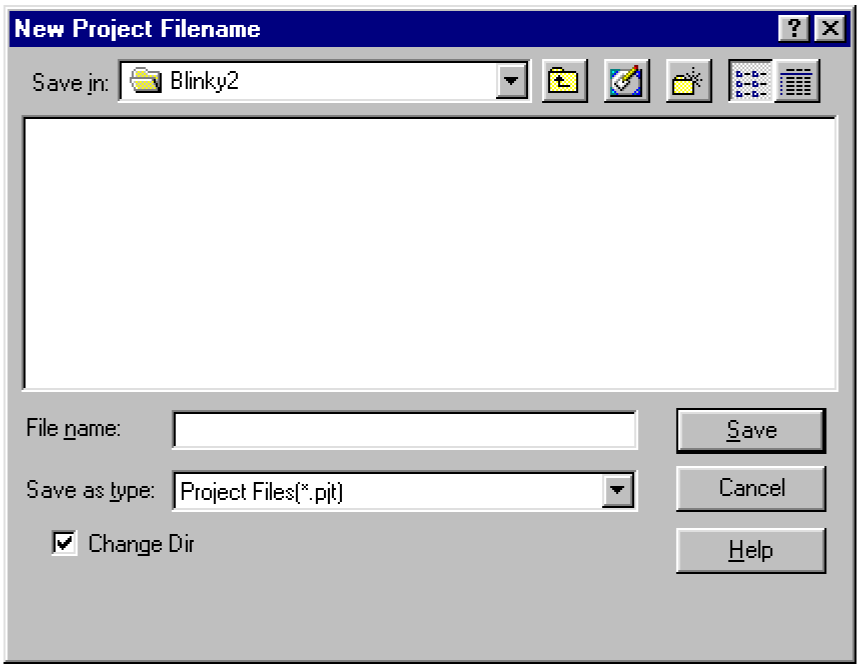
3.2.1 Creating a New Project and Adding an Existing Source File

Go to the *Project* menu and close all projects that might be open

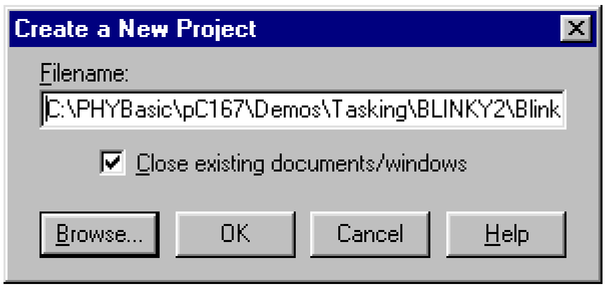
- Open the *Project* menu and choose *New*. The window as shown below appears.



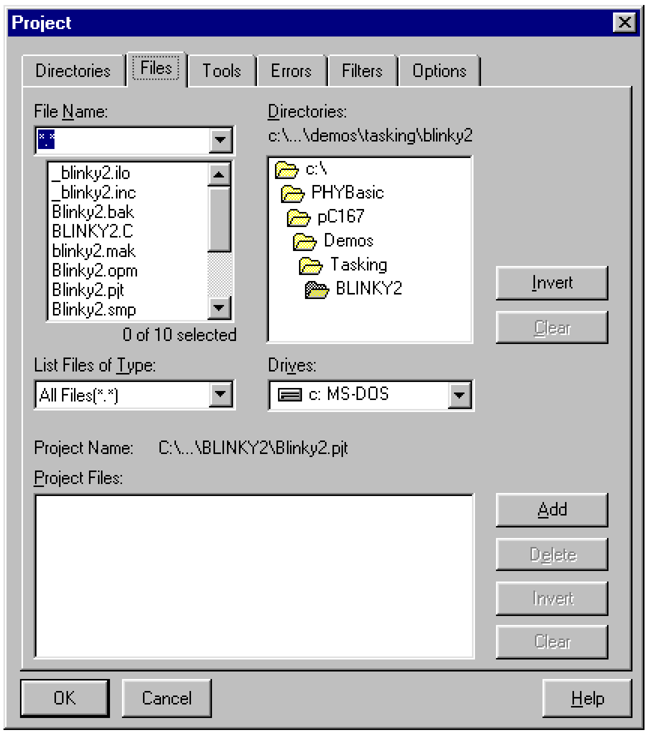
- Press the *Browse* button in order to browse to the folder ***C:\PHYBasic\pc-167\Demos\Tasking\Blinky2*** which includes the programs for this example. See the figure below.



- In the line *'Filename'*, enter the filename of the project you are creating. For this tutorial, enter the name ***Blinky2*** and press *Save*.
- Press *OK* in the *Create a new Project Window*.

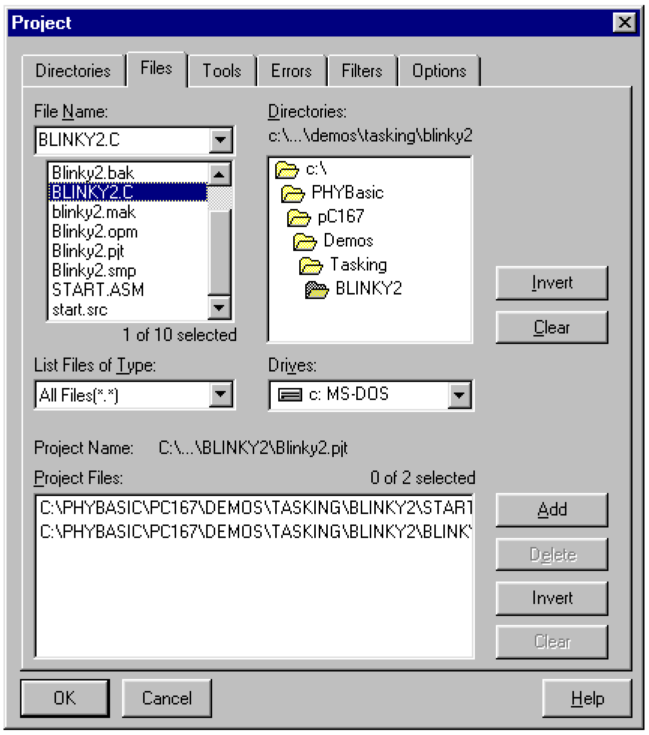

Create a New Project
Filename:
C:\PHYBasic\pC167\Demos\Tasking\BLINKY2\Blink
Close existing documents/windows
Browse...
OK
Cancel
Help

- The following window will appear:



This window is used to add various files to your project. These include ASCII files, C/C++ and assembly sourcefiles, assembler generated object files and library files. Note that the Project Files box is blanked out at this time. This project window is accessible at any time by selecting *Project|Properties|Files*, which enables easy edit of your file list.

- Double clicking on the files which should be part of the project (in this example ***start.asm*** and ***Blinky2.c***) results in the following window.



File Name: BLINKY2.C

Blinky2.bak
BLINKY2.C
blinky2.mak
Blinky2.opm
Blinky2.pjt
Blinky2.smp
START.ASM
start.src

1 of 10 selected

Directories: c:\...\demos\tasking\blinky2

c:\
PHYBasic
pC167
Demos
Tasking
BLINKY2

List Files of Type: All Files(*.*)

Drives: c: MS-DOS

Project Name: C:\...\BLINKY2\Blinky2.pjt

Project Files: 0 of 2 selected

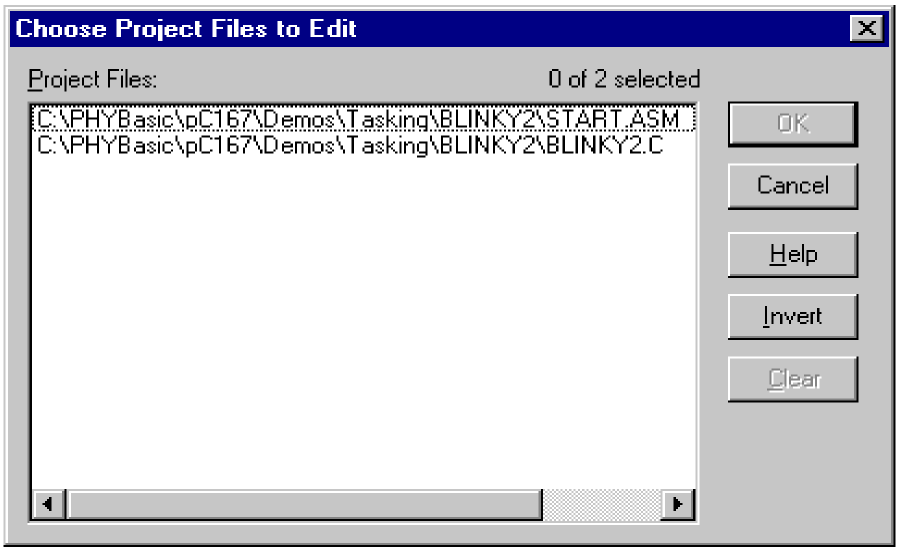
```
C:\PHYBASIC\PC167\DEMOS\TASKING\BLINKY2\STAR1
C:\PHYBASIC\PC167\DEMOS\TASKING\BLINKY2\BLINK`
```

- Choose *OK* to save the new project.

At this point you have created a project called ***blinky2.pjt*** and added an existing C source file called ***blinky2.c*** and an existing Assembler file called ***start.asm***. The next step is to modify the C source before building your project. This includes compiling, linking, locating and creating the hexfile.

3.2.2 Modifying the Source Code

- Open the *Project* menu and choose *Load Files*. The window shown below appears. This is where all project files are displayed. You can double click on the *filename* you would like to edit, or just highlight it and select *OK*. If you would like to load all project files into the editor you can select *Invert* and *OK*.
- Double click on ***blinky2.c*** to open it in the source code editor.



```
C:\PHYBasic\pC167\Demos\Tasking\BLINKY2\START.ASM
C:\PHYBasic\pC167\Demos\Tasking\BLINKY2\BLINKY2.C
```

- Locate the following code section. Modify the section shown below (the values shown in bold and italic) from the original (150,000) counts to the indicated values:

```
while (1) {                    /* loop forever              */

   P1_0 = 0;                   /* output to LED port        */
   for (i=0; i< 225000; i++)   /* delay for 225.000 counts  */
   {
    wait ();                   /* call wait function        */
    }

   P1_0 = 1;                  /* output to LED port         */
   for (i=0; i< 75000; i++)   /* delay for 75.000 counts   */
   {
     wait();                  /* call wait function         */
   }
 }
```

3.2.3 Saving the Modifications

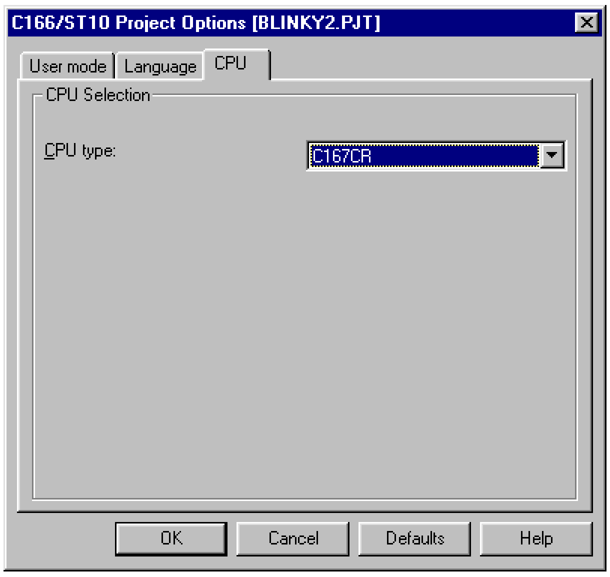
- Save the modified file by choosing *File|Save* or by clicking the floppy disk icon 🖫 .

3.2.4 Setting Tool Chain Options

Tasking includes a Make utility that can control compiling and linking source files in several programming languages. 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. EDE allows you to set various options with mouse clicks and these are all saved in your project file.

To configure the CPU:

Open the *EDE|Project Options* menu and choose *CPU*. Adjust the default settings by clicking *Defaults*. In *CPU type* select C167CR. Press *OK* to close the window after the modification.

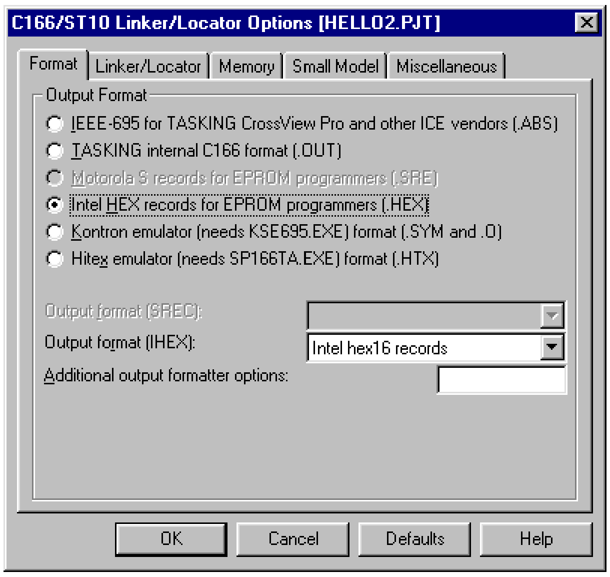

C166/ST10 Project Options [BLINKY2.PJT]

CPU Selection

CPU type: C167CR

To configure the L166 Linker/Locator:

Open the *EDE* menu and choose *Linker/Locator, Options|Format*. Adjust the default settings by clicking *Default*. Select the *Intel Hex records for EPROM programmers (HEX)* radio button and change the output format to *Intel HEX 16 records.*


C166/ST10 Linker/Locator Options [HELLO2.PJT]

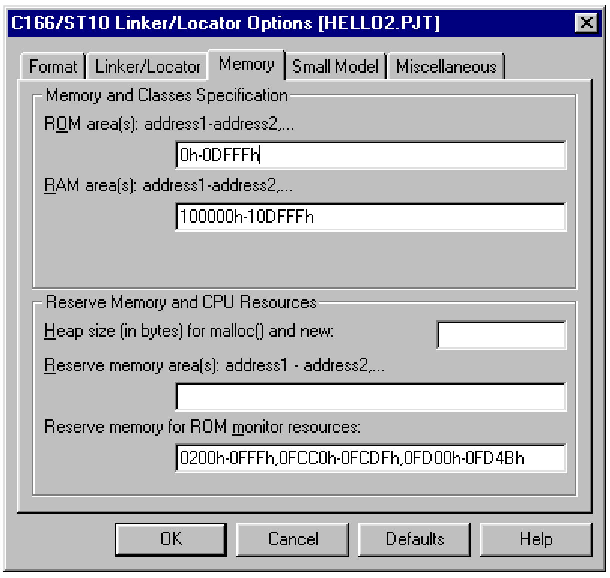
Output Format
- IEEE-695 for TASKING CrossView Pro and other ICE vendors (.ABS)
- TASKING internal C166 format (.OUT)
- Motorola S records for EPROM programmers (.SRE)
- [✓ Intel HEX records for EPROM programmers (.HEX)]
- Kontron emulator (needs KSE695.EXE) format (.SYM and .O)
- Hitex emulator (needs SP166TA.EXE) format (.HTX)

Output format (SREC):

Output format (IHEX): Intel hex16 records

Additional output formatter options:

- Select the *EDE|Linker\Locator Options...|Memory* and specify the following settings


C166/ST10 Linker/Locator Options [HELLO2.PJT]

Memory and Classes Specification

ROM area(s): address1-address2,...
0h-0DFFFFh

RAM area(s): address1-address2,...
100000h-10DFFFh

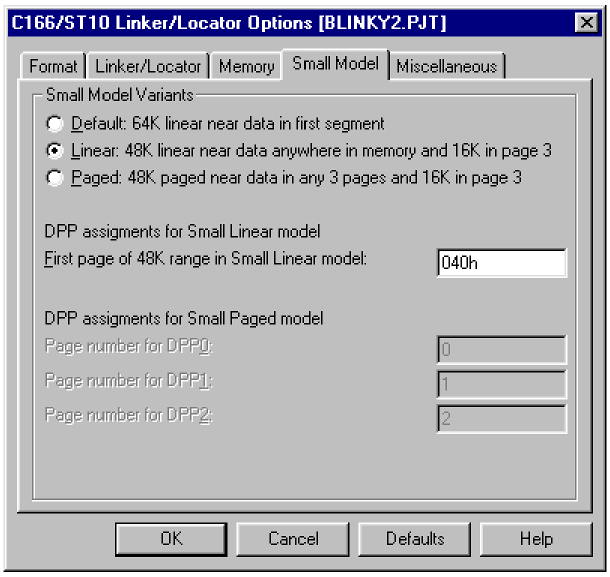
Reserve Memory and CPU Resources

Heap size (in bytes) for malloc() and new:

Reserve memory area(s): address1 - address2,...

Reserve memory for ROM monitor resources:
0200h-0FFFh,0FCC0h-0FCDFh,0FD00h-0FD4Bh

- Enter *EDE|Linker\Locator Options|Small Model* Menu select *Linear: 48K linear near data anywhere in memory and 16K in page 3* and specify *First page of 48K range in Small Linear model:* 040h. Press *OK* to close the window after the modification.


C166/ST10 Linker/Locator Options [BLINKY2.PJT]

Small Model Variants

( ) Default: 64K linear near data in first segment
(●) Linear: 48K linear near data anywhere in memory and 16K in page 3
( ) Paged: 48K paged near data in any 3 pages and 16K in page 3

DPP assigments for Small Linear model
First page of 48K range in Small Linear model: 040h

DPP assigments for Small Paged model
Page number for DPP0: 0
Page number for DPP1: 1
Page number for DPP2: 2

The Linker options are now suitable for the ***Blinky2*** project, enabling you to build an absolute object file without taking into account debugging settings. The options lead to the generation of a ***.map*** map file with included memory map.

3.2.5 Building the Project

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

- Click on the *Execute 'Rebuild' Command* *icon* from the EDE tool bar or open *Project* and select *Rebuild*.
- If the program specified (***blinky2.c***) contains any errors, they will be shown in the Build Output window on the screen.
- If there are no errors, the code is assembled and linked and the executable code is ready to be downloaded to the board. *No errors found* is shown in the bottom line. The code to be downloaded to the board will be the name of the project with ***.hex*** as filename extension (in this case ***blinky2.hex***).
- Note that a machine-readable, executable hexfile has been created. Other files (e.g. assembler input files ****.src***, list files ****.lst*** and map file ****.map***) are created to help the debugging or troubleshooting and error searching process.
- If a list of errors appears, use the editor to correct the error(s) in the source code and save the file and repeat this section.

3.2.6 Downloading the Output File

- Ensure that the target hardware is properly connected to the host-PC and a power supply.
- Reset the target hardware and force it into Bootstrap mode by simultaneously pressing the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board HD200 and then releasing first the Reset and, two or three seconds later, the Boot button.
- Start FlashTools 16W.
- The *Communication Setup* tab of the FlashTools 16W tabsheet window will now appear.
- Select the *phyCORE-167* as the target hardware and click on the *Connect* button.
- At the *Properties for serial communication* window of the FlashTools 16W tabsheet, choose the correct serial port for your host-PC and a 57,600 baud rate.
- Click the *OK* button to establish connection to the target hardware.
- Returning to the FlashTools 16W tabsheet, choose the *Sector Utilities* tab, highlight *Sector #0* in the *Sector Status Information* section of the tab and click on the *Erase Sector(s)* button to erase this memory location.
- Wait until the status check in the lower left corner of the FlashTools 16W 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-167 Demo folder (default location ***C:\PHYBasic\pC-167\Demos\Tasking\Blinky2\Blinky2.h86***) and click *Open.*
- Click on the *Download* button. You can watch the status of the download of the ***Blinky2.h86*** into external Flash memory in the Download window.

- 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 FlashTools 16W 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 FlashTools 16W.
- Press the Reset (S2) button on the Development Board.

If the modified hexfile properly executes, the LED D3 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 download hexfile, and successfully executed this modified code.

3.3 "Hello"

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 "Blinky" example starting at *section 3.1*.

3.3.1 Creating a New Project

- Start the Tasking EDE environment and close all projects that might be open.
- Open the Project menu and create a new project called ***hello2.pjt*** within the existing project folder ***C:\PHYBasic\pC-167\Demos\Tasking\Hello2*** (default location) on your hard-drive.
- *Add* **hello2.c, *serio.c*** and ***start.asm*** from within the project folder to the project ***hello2.pjt***.
- Choose *OK* to save the new project.

At this point you have created a project called ***hello2.pjt*** consisting of a C source files called ***hello2.c, serio.c*** and an assembler file called ***start.asm***.

3.3.2 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***.

3.3.3 Setting Tool Chain Options

The tool chain options can be set similar to the settings for the Blinky2 project described in *section 3.2.4.*

3.3.4 Building the New Project

- Click on the *Execute 'Rebuild' Command* *icon* from the EDE tool bar or open *Project* and select *Rebuild*.
- If any source file of the project contains any 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 and save the file and repeat this section.
- If a list of errors appears, use the editor to correct the error(s) in the source code and save the file and repeat this section.
- If there are no errors, the code is assembled and linked and the executable code is ready to be downloaded to the board.

3.3.5 Downloading the Output file

- Reset the target hardware and force it into Bootstrap mode by simultaneously pressing the Reset (S2) and Boot (S1) buttons on the phyCORE Development Board HD200 and then releasing first the Reset and, two or three seconds later, the Boot button.
- Start FlashTools 16W.
- Select the *phyCORE-167* as the target hardware and click on the *Connect* button.
- Choose the correct serial port for your host-PC and a 57,600 baud rate.
- Click the *OK* button to establish connection to the target hardware.
- Returning to the FlashTools 16W tabsheet, choose the *Sector Utilities* tab, highlight *Sector #0* in the *Sector Status Information* section of the tab and click on the *Erase Sector(s)* button to erase this memory location.
- Wait until the status check in the lower left corner of the FlashTools 16W 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-167 Demo folder (default location
  C:\PHYBasic\pC-167\Demos\Tasking\Hello2\Hello2.h86)
  and click *Open.*
- Click on the *Download* button. You can watch the status of the download of the ***Hello2.h86*** into external Flash memory in the Download window.
- 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 FlashTools 16W 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.
- Click on the *Disconnect* button and exit FlashTools 16W.

3.3.6 Starting the Terminal Emulation Program

- Start the 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 HD200 (at S2) will execute the ***Hello2.h86*** file loaded into the Flash.
- Now push the <Space> bar on your keyboard once to start the automatic baud rate detection on phyCORE-167 module.
- Successful execution will send the modified character string ***"PHYTEC... Stick It In!"*** to the HyperTerminal window.
- Click the Disconnect icon .
- Close the HyperTerminal program

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

4 Debugging

The Tasking debugger has two possible modes:

- the CrossView Pro simulator debugger environment is a software debugger and simulator
- the CrossView Pro ROM monitor debugger communicates with the target hardware via a monitor kernel that is downloaded to the RAM on the target system

This example utilizes the CrossView Pro ROM monitor debugger debug environment, which automatically downloads a special monitor kernel to the target hardware using the Bootstrap mode.

4.1 Creating a Debug Project and Preparing the Debugger

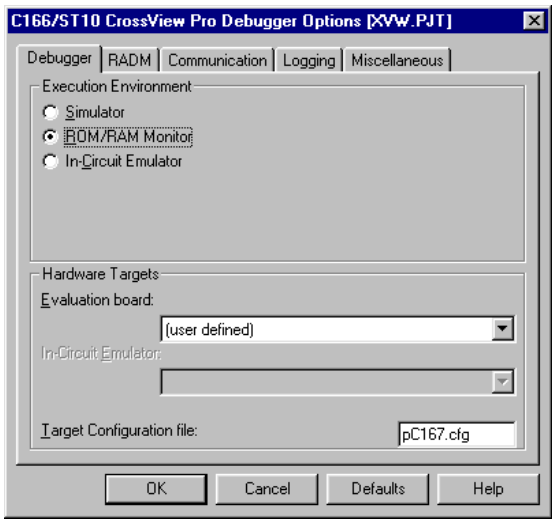
Create a new project in ***C:\PHYBasic\pC-167\Demos\Tasking\xvw*** folder named xvw. *Add* the files ***Start.asm***, ***Addone.src*** and ***Demo.c*** to the project and *load* ***Demo.c*** in the editor (*Project|Load Files)*.

To be able to debug an application on target hardware the complete application (code and data) needs to be loaded into the target hardware's RAM. Therefore the application needs to be rebuild with different settings compared to the stand-alone running version. Compared to the Blinky2 example settings *(see section 3.2.4*) the changes are listed below.

- EDE|*Projekt Options\User mode* select *Expert* mode
- *EDE|Linker\Locator Options|Format* select *IEEE-695 for Tasking CrossView Pro and other ICE vendors (ABS)*
- *EDE|Linker\Locator Options|Memory:* No entries necessary in ROM/RAM areas
- *EDE|Linker\Locator Options|Small Model* select *Default: 64K linear near data in first segment*
- Rebuild the project

Since files required for the download are specific to the target hardware the appropriate selection has to be made in *EDE|CrossView Pro Options|Debugger*.

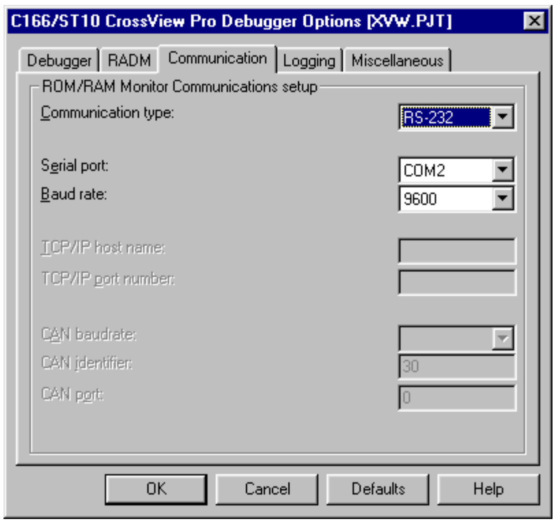
- Choose *ROM/RAM Monitor* for the Execution Environment, select (*user defined*) as Evaluation board and enter pC167.cfg as Target Configuration file.



Execution Environment
Simulator
[✓ ROM/RAM Monitor]
In-Circuit Emulator

Hardware Targets
Evaluation board: (user defined)
In-Circuit Emulator:
Target Configuration file: pC167.cfg

- The baud rate plus COM port can be set in *EDE|CrossView Pro Options|Communication* menu. Press *OK* to close the window after the modification. Press *OK* to close the window after the modification.



ROM/RAM Monitor Communications setup
Communication type: RS-232
Serial port: COM2
Baud rate: 9600
TCP/IP host name:
TCP/IP port number:
CAN baudrate:
CAN identifier: 30
CAN port: 0

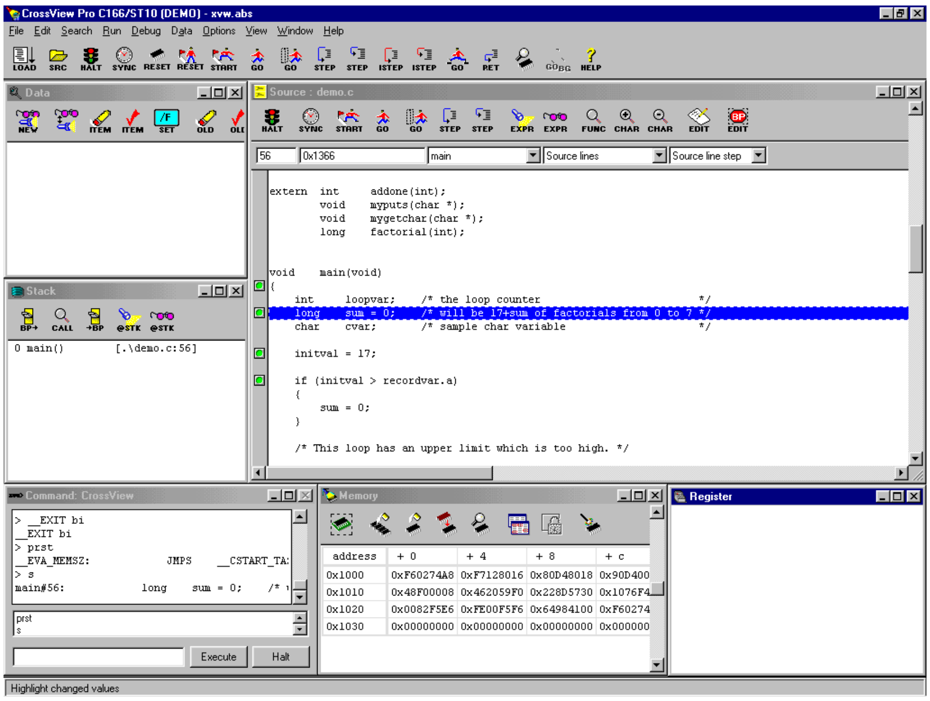
4.2 Preparing the Target Hardware to Communicate with CrossView Pro

- Ensure that the target hardware is properly connected to the host-PC and a power supply
- Reset the target hardware and put it in Bootstrap mode by simultaneously pressing the Reset (S2) and Boot (S1) switches on the Development Board and then releasing first the Reset (S2) and, two or three seconds later, release the Boot (S1) switch

Since the required microcontroller portion to communicate with CrossView Pro will be automatically downloaded by CrossView Pro using the Bootstrap mode there is no further preparation of the target system.

4.3 Starting the Debugger

- To start the Tasking debug environment, click the debugger icon on the EDE toolbar.
- The CrossView Pro debugger will download the monitor program automatically via bootstrap mode. Then the current opened project is downloaded.
- A CrossView Pro window similar to that shown below appears.


CrossView Pro C166/ST10 (DEMO) - xvw.abs
File Edit Search Run Debug Data Options View Window Help
LOAD SRC HALT SYNC RESET RESET START GO GO STEP STEP ISTEP ISTEP GO RET GOBG HELP
Data
NEW ITEM ITEM SET OLD OLI
Source : demo.c
HALT SYNC START GO GO STEP STEP EXPR EXPR FUNC CHAR CHAR EDIT EDIT
56 0x1366 main Source lines Source line step
Stack
BP+ CALL +BP @STK @STK
Command: CrossView
Execute Halt
Memory
Register
Highlight changed values

```
extern  int     addone(int);
        void    myputs(char *);
        void    mygetchar(char *);
        long    factorial(int);


void    main(void)
{
    int     loopvar;    /* the loop counter                           */
    long    sum = 0;    /* will be 17+sum of factorials from 0 to 7 */
    char    cvar;       /* sample char variable                       */

    initval = 17;

    if (initval > recordvar.a)
    {
        sum = 0;
    }

    /* This loop has an upper limit which is too high. */
```

```
0 main()          [.\demo.c:56]
```

```
> __EXIT bi
__EXIT bi
> prst
__EVA_MEMSZ:            JMPS     __CSTART_TA
> s
main#56:           long     sum = 0;     /*
```

```
prst
s
```

address	+ 0	+ 4	+ 8	+ c
0x1000	0xF60274A8	0xF7128016	0x80D48018	0x90D400
0x1010	0x48F00008	0x462059F0	0x228D5730	0x1076F4
0x1020	0x0082F5E6	0xFE00F5F6	0x64984100	0xF60274
0x1030	0x00000000	0x00000000	0x00000000	0x000000

The window marked *“Source: demo.c”* is the Debug window. The *Command: CrossView* window can be used to enter commands. You may need to open, resize and /or move some windows to make your screen look something like the screen capture. You may have to open any invisible window by opening the *View* menu and selecting the window to view.

- For troubleshooting please *refer to Appendix A:Troubleshooting*.

4.4 Using the Tasking CrossView Pro Debug Features

4.4.1 Breakpoints

- Click on a memory location such as line number 69 *for (loopvar = 0; loopvar <= 8; ++loopvar)*. A frame appears marking this position.
- You can click on *Go* to reach this point or you can set a breakpoint by pressing the *green but*ton to the left. Set a breakpoint here. The green button will become red to indicate the first breakpoint.
- Click on *Go* and the program will run and stop at the break-point.
- Click on the breakpoint again to remove it.

4.4.2 Data Window

- If not already visible activate the data window by selecting *View|Data*
- To watch the value of variable *initval* either double click on the *variable* in the source window or press the *New* button in the data window and type in the name of the variable. Then select *Add Watch* and *Close* to display the variable in the data window.
- To change the value of a variable just double click on it in the *data window* or use the command line e.g. *initval = 10* to set the value to 10
- Structures can be viewed/modified in a similar way. Just double click on *recordvar* and select *Add Watch* to display the structure in the data window. A '+' sign to the left shows a possible 'expansion' of the value.

4.5 Single Stepping

- CrossView Pro uses *Step into (function calls)* to single step one instruction at a time. *Step into (function calls)* is also used to enter a function in the same fashion.
- *Step over (function calls)* means to skip over a function that you are not interested in.
- *Return from function* is used to exit a function you are currently in. *Return from function* is very useful if you find yourself in a function you are not interested in and need to return quickly to your intended function.
- If CrossView Pro gets stuck, execute a reset using the *Reset application (but do not run)* button in the top toolbar . You can also use the hardware reset button on the board.
- As you step through the program - the appropriate values in the *Register* window will change color as their values are updated.

4.5.1 Memory Window

- If the Memory Window is not already visible, activate it by selecting *View|Memory* from the CrossView Pro tool bar.
- The Memory Window can be configured by pressing the *Setup Memory Display button* in the memory window
- Enter *Start address: 0 x 1030* in *Memory Window Setup* menu and *Base Type* Long. The addresses displayed in the Memory Window should change starting at this address. This is the memory area for the long array *table*. The recursive calculated factorial values of the variable *loopvar* are stored in this array. Single step through the for loop: *for (loopvar = 0; loopvar <= 8; ++loopvar)*.

4.5.2 Stack Window

- If not already visible view the Stack Window by selecting *View|Stack.*
- The Stack Window shows the current contents of the stack after the program has been stopped.
- The Stack Window displays the following information for each stack level: The name of the function that was called, all parameters specified to the function and the line number in the source code from which the function was called.
- Stepping into the factorial calculation *table[loopvar] = factorial(loopvar);* gives a good impression of how the Stack Window works.

4.5.3 Simulated I/O

- Simulated I/O is activated by default. Stream 0 is used for input and stream 1 is used for output. This feature lets you observe and simulate the input and output of your program before the hardware peripherals are in place.
- To activate the simulated I/O streams display in CrossView Pro select *Debug|Simulated I/O Setup* and double click on *0* and *1* in the *stream Nr* column.
- Now set a *breakpoint* at line *myputs("fail\n");* and press *Go*.
- Press *Step (over function calls)*. A new window should pop up which shows the contents of SIO stream: 1 (fail).
- The program now hangs in an endless loop *while (loopvar)* since loopvar is never zero.
- Set the variable loopvar to zero by modifying it's content to reach the end of the program (which includes a simulated input demonstration too).

4.6 Resetting CrossView Pro and the phyCORE-167

- The CrossView Pro RAM/ROM monitor debugger: The Monitor runs in the target hardware. When the *Reset the application (but do not run)* button is pressed the IP is set to zero. This command does NOT perform a reset of the target system, that is, no registers are modified except for the program counter. When the *Reset target CPU/system (but do not run)* button is pressed the target is initialized according to the power-up sequence for the processor. Almost all registers, including the system stack pointer and program counter are initialized.

- The PHYTEC Development Board does not have a hardware NMI button. A NMI is the most reliable way to stop a running program. This is even more important when you are using the Bootstrap version of the Monitor because it is difficult to re-synchronize CrossView Pro and the monitor program. For example, it is not possible for CrossView Pro to re-synchronize automatically by pressing the hardware-Reset button (S2) if the monitor has been downloaded with the Bootstrap Loader. It must be restarted manually by setting the board in bootstrap mode and downloading the monitor program again.

5 Advanced User Information

This section provides advanced information for successful operation of the phyCORE-167 with the Tasking software tool chain.

5.1 FlashTools 16W

Flash is a highly functional means of storing non-volatile data. One of its advantages is the possibility of on-board programming. Programming tools for the Flash device are delivered with the phyCORE-167 in the form a set of executable and binary files that run under Windows9x/NT. These tools make use of the Bootstrap mode to transfer executable code to the phyCORE-167 that, in turn, download user code into the Flash. Additionally, the re-programmable Flash device on the phyCORE-167 enables easy update of user code and the target application in which the phyCORE-167 has been implemented.

Currently the phyCORE-167 can be populated with four different sized Flash devices: a 29F200 with 256 kByte, a29F400 with 512 kByte, a 29F800 with 1 MB or a 29F160 with MB.

FlashTools 16W always uses the Bootstrap mode to transfer the required microcontroller firmware to the phyCORE-167. Hence, there is no restriction in using the Flash memory for storing user code.

Before Flash programming FlashTools 16W will adjust the internal Chip-Select Unit and the bus configuration of the microcontroller during the Bootstrap download. This results in the following programming Memory-Model:

- Flash Bank using /CS0 addressable at 000000H-1FFFFFH (up to 2 MB Flash)
- RAM Bank 1 using /CS1 addressable at 200000H-2FFFFFH (up to 1 MB RAM)

Using the Bootstrap mode to transfer the required microcontroller firmware to the phyCORE-167 ensures that FlashTools 16W maintains its independent Flash programming capability.

5.2 Start.asm

The code within the assembly file ***start.asm*** and it's include files is responsible for the initial controller configuration and the startup initialization of your C project. This includes adjusting the properties of the external bus signals and Chip-Select signals, setting of the system stack, initialization of variables and clearing of memory areas.

This code will execute prior to the execution of user code. The startup code occupies the Reset Vector of the application, which is the location where the microcontroller starts execution after reset (0 x 0000). After performing the initialization steps your individual *main()* function is called by the startup code.

The configuration of the startup code is done via macro preprocessor defines which are generated automatically by EDE according to the settings in *EDE|CPU Options*. Most important in combination with the phyCORE-167 is the correct Chip-Select (CS) setup via the Buscon and Addrsel registers.

Because the startup code is modified by macro preprocessor defines it is not necessary to copy the start.asm file (located in \lib\src folder) into the project folder. But it needs to be added to the project files.

5.3 Linking and Locating

The Linker has to combine several re-locatable 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 type, constants and data to fixed address locations within the address range of the microcontroller: This ensures the natural or explicitly declared properties of these segments.

Data segments must always be located in Random Access Memory (e.g. RAM), code and constant segments should be located in any kind of non-volatile memory (e.g. Flash). The C166 family has a Von-Neumann architecture which uses the same read signal to fetch data and also code or constants. To distinguish between non-volatile and modifiable memory, physically different memory devices must be addressable within different address ranges. A list of the compiler generated section names can be found in Taskings *C Compilers User's Guide* in *section 3.2.2, "Section Allocation"*.

It is required that all RAM data sections are located to any internal RAM of the C167 or to any external RAM of the phyCORE-167. All ROM code and –data sections must also be located to any internal non-volatile memory (e.g. Flash, OTPROM) of the C167 or any external Flash memory of the phyCORE-167.

The *small Memory-Model* which is the only one supported in the eval version of the Tasking C166/ST10 tool chain requires a special treatment of near RAM- and ROM-data sections. More detailed information on this subject can be found in Taskings *C Compilers User's Guide* in *section 3.2.1.2, "Small Memory-Model"*.

To ensure proper execution of your application you must take into account the runtime Memory-Model when linking and locating. This means that you must instruct the Linker/Locator where to assume external RAM for locating data classes and Flash for locating code and constant classes.

The recommended operating mode of the phyCORE-167 allows the use of the Chip-Select Unit of the C167 to define the physical memory layout. By modifying the system registers (Buscon and Addrsel) in *EDE|CPU Options* as part of your project you can adapt the memory layout to your needs.

The external use of the Chip-Select signals is predefined by the hardware in the following way:

- Flash Bank uses /CS0 (up to 2 MB Flash)
- RAM Bank uses /CS1 (up to 1 MB RAM)
- Optional UART uses /CS2

The default configuration of the phyCORE-167 has 256 kByte Flash (/CS0) and 256 kByte RAM (/CS1).

You will see multiple mirrors of a memory device that has a physical smaller address range than the associated address range of the Chip-Select signal.

For instance if you adjust Chip-Select Signal /CS1 to be active within an address range of 1 MB and the actually memory size populating the phyCORE-167 is just 256 kByte, you will get three mirrors of your RAM.

We recommend that you generate a ***.map** map file for your project and inspect the memory map information within this file. Accurately compare this information to the physical Memory-Model resulting from your settings selected within the *EDE|CPU Options* menu.

5.4 Debugging Using Monitor Kernel

Whenever you decide to use the CrossView Pro ROM monitor debugger to debug your application, some special precautions must be taken into consideration to ensure proper code execution of your application.

Your application and the ROM monitor kernel contained in the files ***b167nrb.sre*** and ***m167r.sre*** must share some memory locations within the target system. If you do not consider the physical Memory-Model already claimed by the kernel and the memory requirements of the kernel, you may get conflicts in memory use. This typically leads to variables containing not their assigned value, functions returning bad results and modified code.

To prevent this kind of conflicts you can instruct EDE to reserve the memory areas needed by the monitor kernel. This is done in *EDE|CPU Configuration|Startup* menu by checking the *ROM or RAM monitor on evaluation board* box, when you choose the Expert mode in *Project Options|User mode*.

To obtain information about the memory requirements of the Monitor, you can take a look at *Reserve memory for ROM &monitor resources:* in *EDE|Linker\Locator Options|Reserve* menu. In addition the monitor uses the NMI and the serial interrupt. This interrupts may not be used by your application.

5.5 Using ROM Debug Monitor in Flash Memory

It is also possible to load the monitor kernel into Flash memory. In this case the bootstrap mode is not necessary anymore since the monitor program is running immediately after reset. For more information about the monitor kernel in Flash memory you can take a look into the Addendum: *C166/ST10 Family Target Boards section 5.1.2, "ROM Debug Monitor Using Dual Vector Table of the CrossView Pro Debugger User's Guide"*. The sourcefiles of the monitor program are included in the full version of the product. The monitor kernel is royalty free and may be included in the customer application.

Appendices A: Troubleshooting

A.1. Initialization Error: ROM Monitor DLL could not Initialize or Access the Target Board

- board may not be in bootstrap mode. Try to activate the bootstrap mode again and press *Retry* button.
- check for correct power supply and serial connection
- the selection in *ROM/RAM Monitor Board* may be wrong in *EDE|CrossView Pro Options|Debugger* menu
- the baud rate in *EDE|Cross View Pro Options|Communication* may be set too high. With 'default' oscillator frequencies normally the maximum value is 38400 baud.
- the wrong COM port may be selected in *EDE|Cross View Pro Options|Communication* menu
- the serial FIFO buffer in Windows 95 can cause transmission problems. CrossView Pro may have problems completing the communication initialization process. This can be intermittent. The FIFO can be disabled under *Controlpanel|System|Device Manager|Port Settings|Advanced.* Make sure *Use FIFO buffers* in this menu is not activated.

A.2 Program Download Never Stops

- the downloaded program may overwrite the monitor kernel. Verify whether *ROM or RAM monitor on evaluation board* is checked in *EDE|Project Options|Startup* menu or take a look into the map file of your project.
- the address window settings may be wrong. The RAM Bank should be mapped to address 0h. Check Buscon1 and Addrsel1 settings in *EDE|CPU Options*.

A.3 Placing Code Breakpoint at 0xXXXXXX Failed: Memory not Writeable

When the CrossView Pro ROM monitor debugger is used the complete application which will be debugged must be allocated in RAM area. The error message is generated because parts of the application are allocated in read only memory. You can verify this by checking the projects mapfile (map).

Document:	**phyCORE-167 QuickStart Instructions**
Document number:	**L-548e_3, February 2003**

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 2003

Ordering No. L-548e_3
Printed in Germany