

# AP16137

## XE164

UConnect-CAN XE164 "Cookery Book" for a hello world application using the KEIL tool chain (you can do the hello world example in this document with the evaluation version of the KEIL tool chain)

Microcontrollers



Never stop thinking

**Edition 2008-07-16**

**Published by  
Infineon Technologies AG  
81726 München, Germany**

**© Infineon Technologies AG 2008.  
All Rights Reserved.**

#### **LEGAL DISCLAIMER**

THE INFORMATION GIVEN IN THIS APPLICATION NOTE IS GIVEN AS A HINT FOR THE IMPLEMENTATION OF THE INFINEON TECHNOLOGIES COMPONENT ONLY AND SHALL NOT BE REGARDED AS ANY DESCRIPTION OR WARRANTY OF A CERTAIN FUNCTIONALITY, CONDITION OR QUALITY OF THE INFINEON TECHNOLOGIES COMPONENT. THE RECIPIENT OF THIS APPLICATION NOTE MUST VERIFY ANY FUNCTION DESCRIBED HEREIN IN THE REAL APPLICATION. INFINEON TECHNOLOGIES HEREBY DISCLAIMS ANY AND ALL WARRANTIES AND LIABILITIES OF ANY KIND (INCLUDING WITHOUT LIMITATION WARRANTIES OF NON-INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS OF ANY THIRD PARTY) WITH RESPECT TO ANY AND ALL INFORMATION GIVEN IN THIS APPLICATION NOTE.

#### **Information**

For further information on technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies Office ([www.infineon.com](http://www.infineon.com)).

#### **Warnings**

Due to technical requirements components may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies Office.

Infineon Technologies Components may only be used in life-support devices or systems with the express written approval of Infineon Technologies, if a failure of such components can reasonably be expected to cause the failure of that life-support device or system, or to affect the safety or effectiveness of that device or system. Life support devices or systems are intended to be implanted in the human body, or to support and/or maintain and sustain and/or protect human life. If they fail, it is reasonable to assume that the health of the user or other persons may be endangered.

---

**AP08048**

**Revision History:** 2008-05 V2.0

Previous Version: none

Page	Subjects (major changes since last revision)

**We Listen to Your Comments**

Any information within this document that you feel is wrong, unclear or missing at all?  
Your feedback will help us to continuously improve the quality of this document.  
Please send your proposal (including a reference to this document) to:

[mcdocu.comments@infineon.com](mailto:mcdocu.comments@infineon.com)



**Note:** Table of Contents [see page 9](#).

### Introduction:

This “Appnote” is a Hands-On-Training / Cookery Book / step-by-step book.  
It will help inexperienced users to get an UConnect-CAN XE164 up and running.

With this step-by-step book you should be able to get your first useful program in less than 2 hours.

The purpose of this document is to gain know-how of the microcontroller and the tool-chain.  
Additionally, the "hello-world-example" can easily be expanded to suit your needs.  
You can connect either a part of - or your entire application to the UConnect-CAN XE164.  
You are also able to benchmark any of your algorithms to find out if the selected microcontroller fulfils all the required functions within the time frame needed.

**Note:**

The style used in this document focuses on working through this material as fast and easily as possible. That means there are full screenshots instead of dialog-window-screenshots; extensive use of colours and page breaks; and listed source-code is not formatted to ease copy & paste.

Have fun and enjoy the UConnect-CAN XE164!



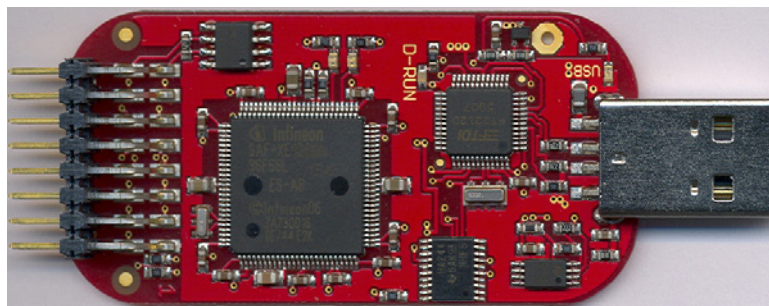
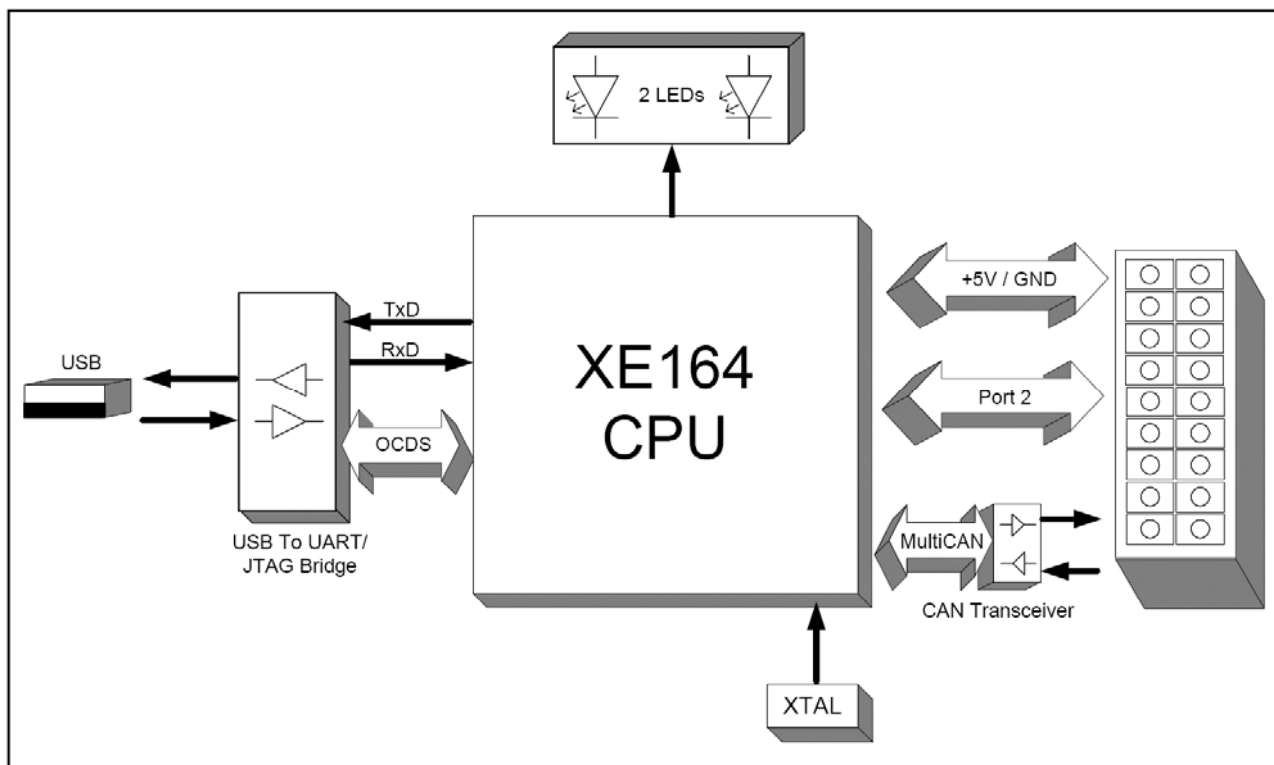


# Programming Example

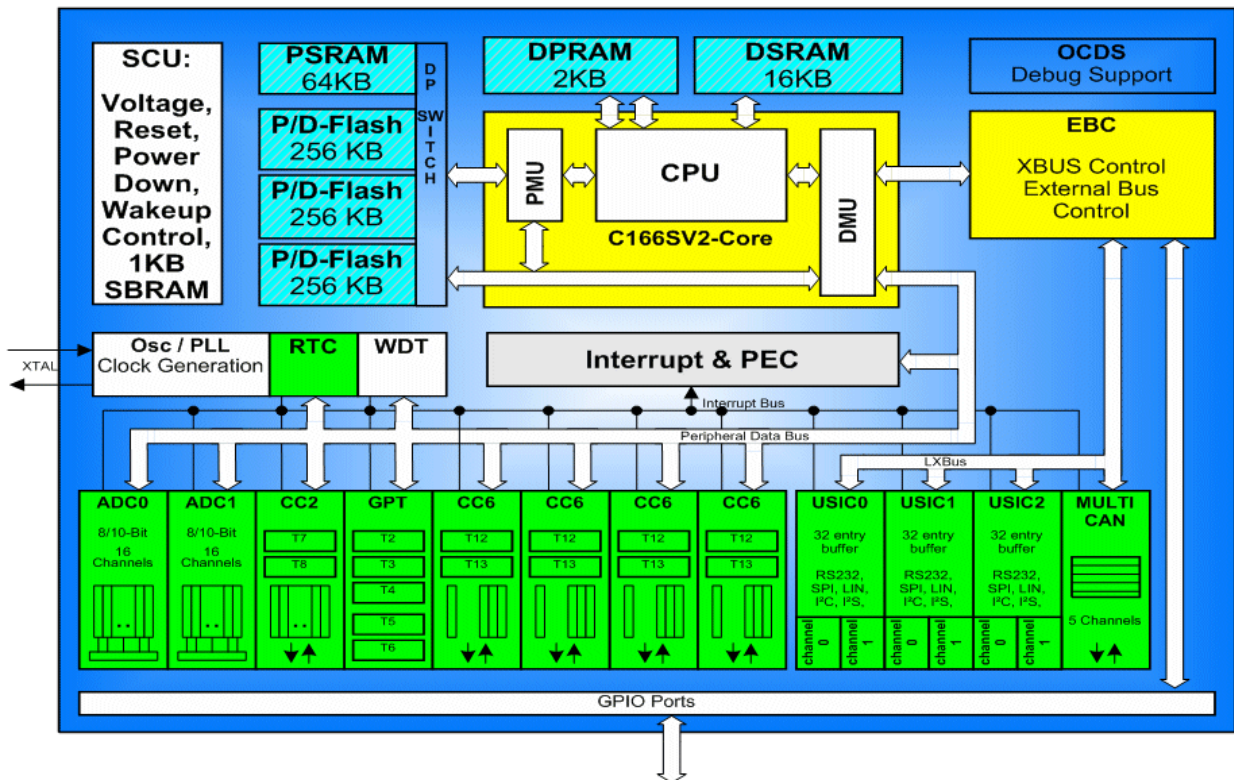
## UConnect-CAN XE164



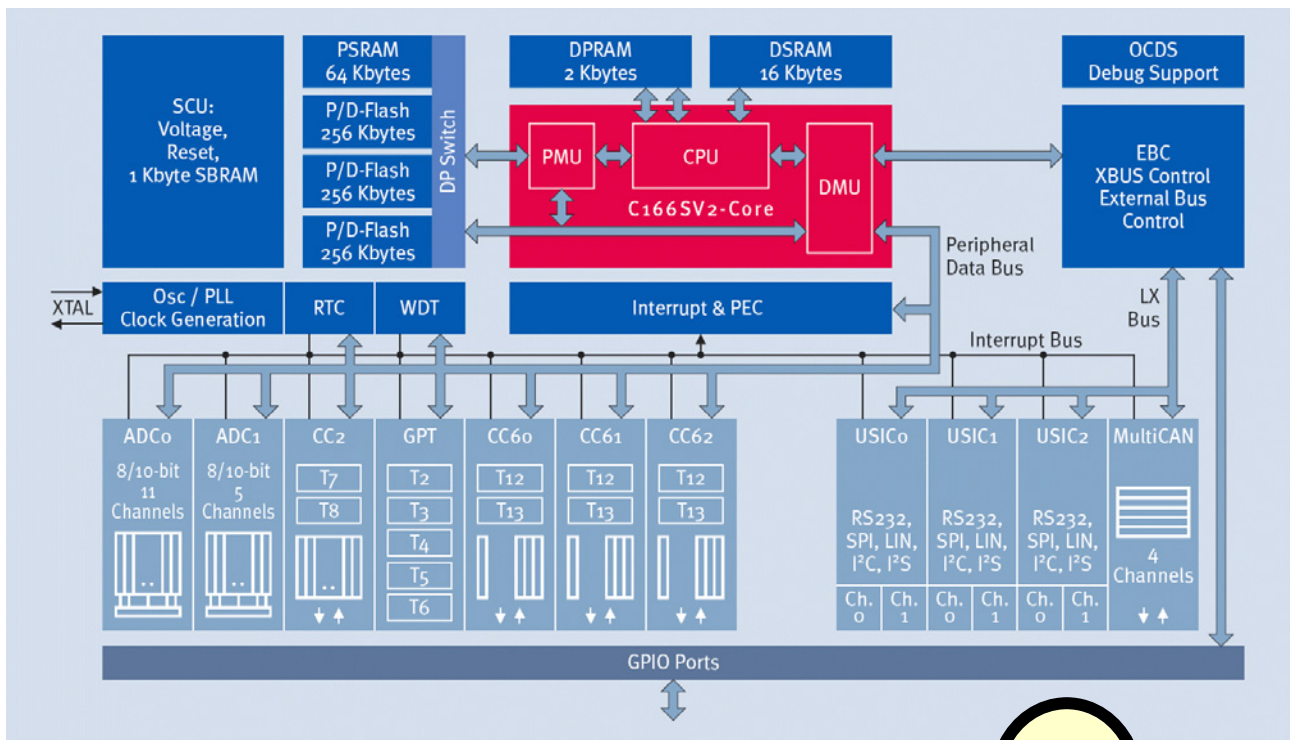
**Block Diagram** (Source: XE164 UConnect Manual)



**SAF-XE167F-96F66L Block Diagram** (Source: Product Marketing)

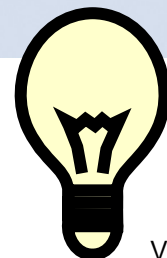


**SAF-XE164F-96F66L Block Diagram** (Source: Product Brief)



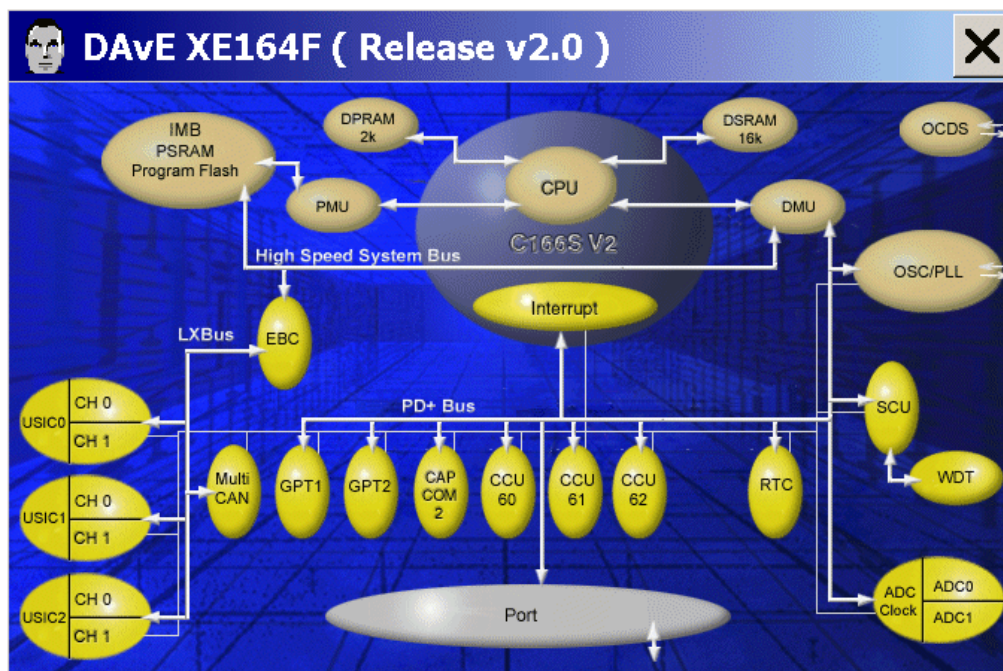
**Note:**

The XE164 microcontroller is a derivative of the XE167 microcontroller!

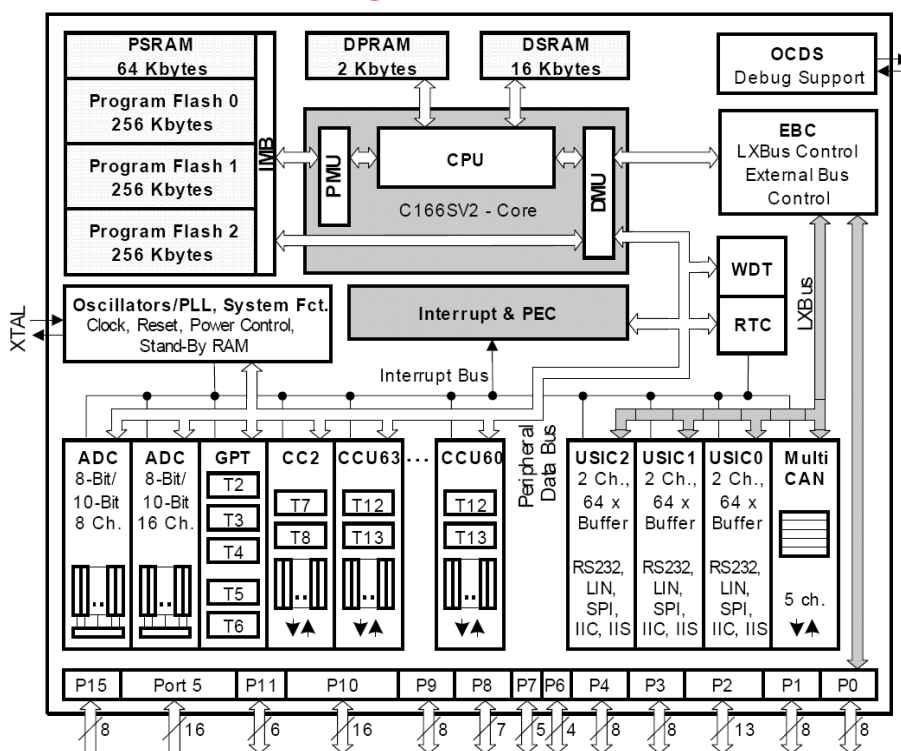




### SAF-XE164F-96F66L Block Diagram (Source: DAVe)



### **XE16x Block Diagram (Source: User's Manual)**



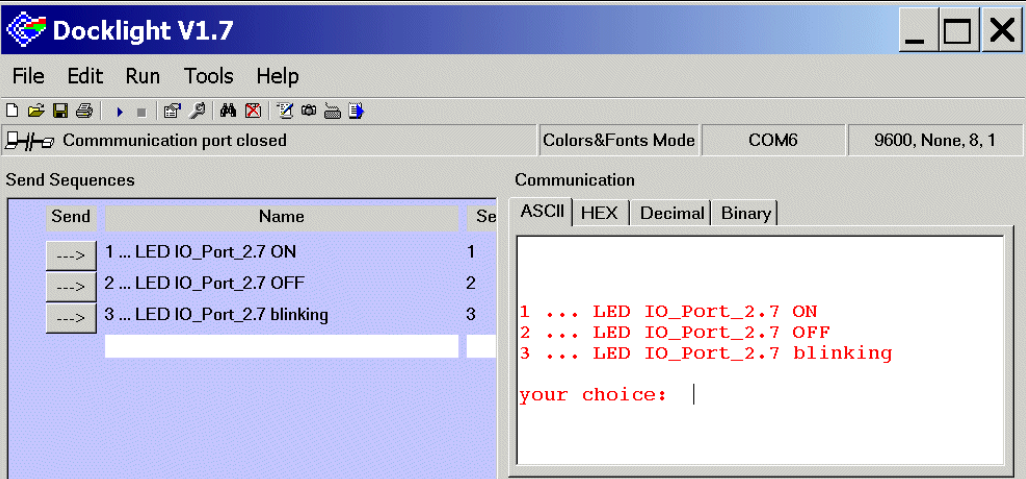
Note:

Just by comparing the different sources of block diagrams, you should be able to get a complete picture of the microcontroller and to answer some of your initial questions.



### “Cookery book“

For your first programming example for the UConnect-CAN XE164:

Your program:	
Chapter/ Step	*** Recipes ***
1.)	<a href="#"><u>DAS Installation + Connecting the UConnect-CAN XE164</u></a>
2.)	<a href="#"><u>DAvE (program generator)</u></a> <a href="#"><u>DAvE Installation (mothersystem) + DAVe Update Installation (XE16xx_Series.dip) for XE164</u></a>
3.)	<a href="#"><u>Using DAVe</u></a> <a href="#"><u>Microcontroller initialization for your programming example</u></a>
4.)	<a href="#"><u>Using the KEIL Development Tools (C-Compiler)</u></a> <a href="#"><u>Programming of your application (XE164) with the KEIL tool chain (µVision3)</u></a>
5.)	<a href="#"><u>Running your first programming example</u></a>

### Feedback

6.)	<a href="#"><u>Feedback</u></a>
-----	---------------------------------

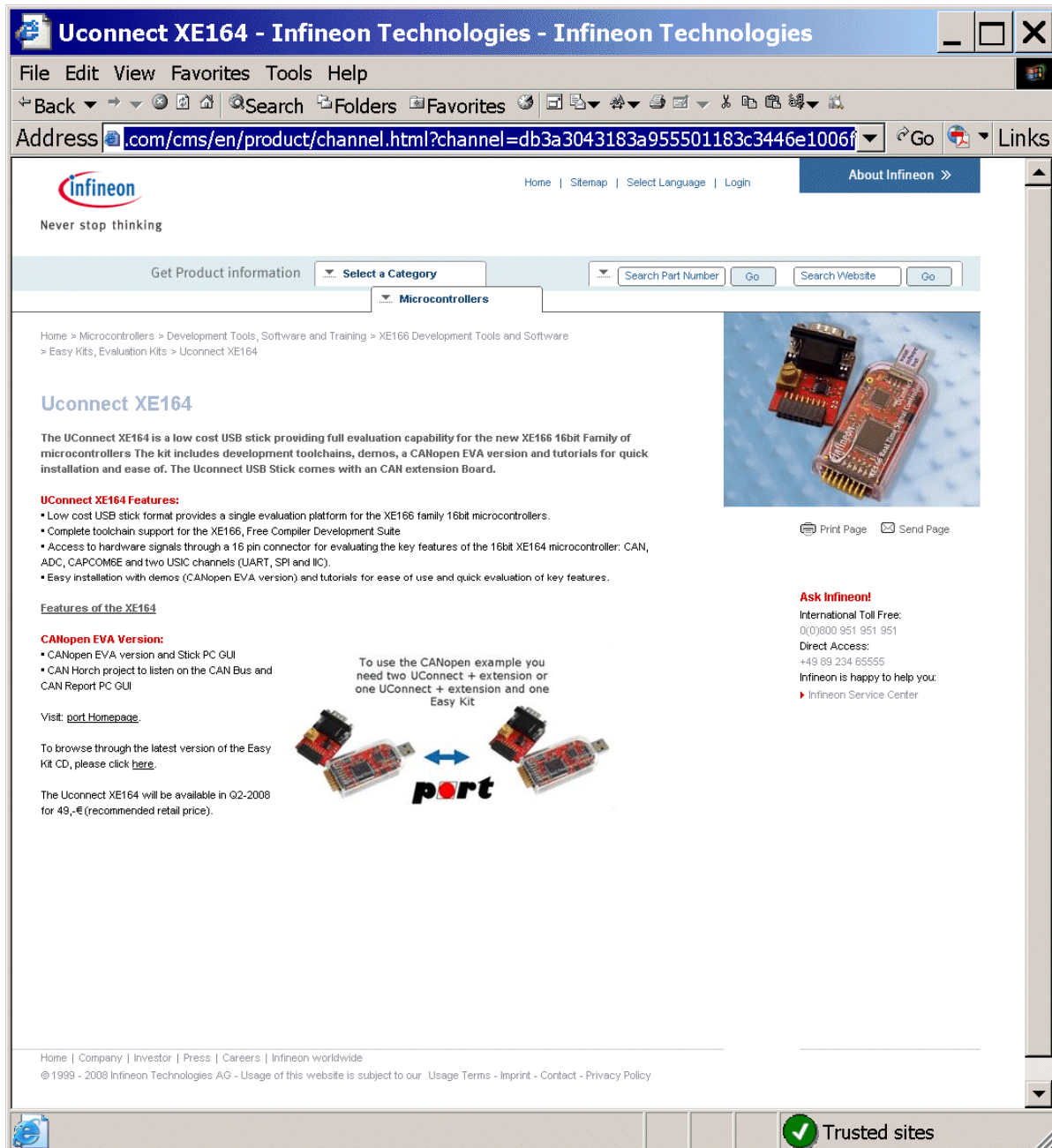


## 1.) DAS Installation + Connecting the UConnect-CAN XE164:



Screenshot of the UConnect-CAN XE164 Homepage:

<http://www.infineon.com/cms/en/product/channel.html?channel=db3a3043183a955501183c3446e1006f>



**Uconnect XE164 - Infineon Technologies - Infineon Technologies**

File Edit View Favorites Tools Help

Back Forward Stop Reload Home Search Folders Favorites Address [www.infineon.com/cms/en/product/channel.html?channel=db3a3043183a955501183c3446e1006f](http://www.infineon.com/cms/en/product/channel.html?channel=db3a3043183a955501183c3446e1006f) Go Links

infineon Never stop thinking

Home | Sitemap | Select Language | Login About Infineon >>

Get Product information Select a Category Search Part Number Go Search Website Go

Microcontrollers

Home > Microcontrollers > Development Tools, Software and Training > XE166 Development Tools and Software > Easy Kits, Evaluation Kits > Uconnect XE164

### Uconnect XE164

The UConnect XE164 is a low cost USB stick providing full evaluation capability for the new XE166 16bit Family of microcontrollers. The kit includes development toolchains, demos, a CANopen EVA version and tutorials for quick installation and ease of. The Uconnect USB Stick comes with an CAN extension Board.

**Uconnect XE164 Features:**

- Low cost USB stick format provides a single evaluation platform for the XE166 family 16bit microcontrollers.
- Complete toolchain support for the XE166, Free Compiler Development Suite
- Access to hardware signals through a 16 pin connector for evaluating the key features of the 16bit XE164 microcontroller: CAN, ADC, CAPCOM6E and two USIC channels (UART, SPI and IIC).
- Easy installation with demos (CANopen EVA version) and tutorials for ease of use and quick evaluation of key features.

**Features of the XE164**

**CANopen EVA Version:**

- CANopen EVA version and Stick PC GUI
- CAN Hirsch project to listen on the CAN Bus and CAN Report PC GUI

Visit: [port Homepage](#).

To browse through the latest version of the Easy Kit CD, please click [here](#).

The Uconnect XE164 will be available in Q2-2008 for 49,-€ (recommended retail price).

To use the CANopen example you need two UConnect + extension or one UConnect + extension and one Easy Kit

Print Page Send Page

**Ask Infineon!**  
International Toll Free: 0(0)800 951 951 951  
Direct Access: +49 89 234 65555  
Infineon is happy to help you:  
▶ Infineon Service Center

Home | Company | Investor | Press | Careers | Infineon worldwide  
© 1999 - 2008 Infineon Technologies AG - Usage of this website is subject to our Usage Terms - Imprint - Contact - Privacy Policy

Trusted sites

### Note:

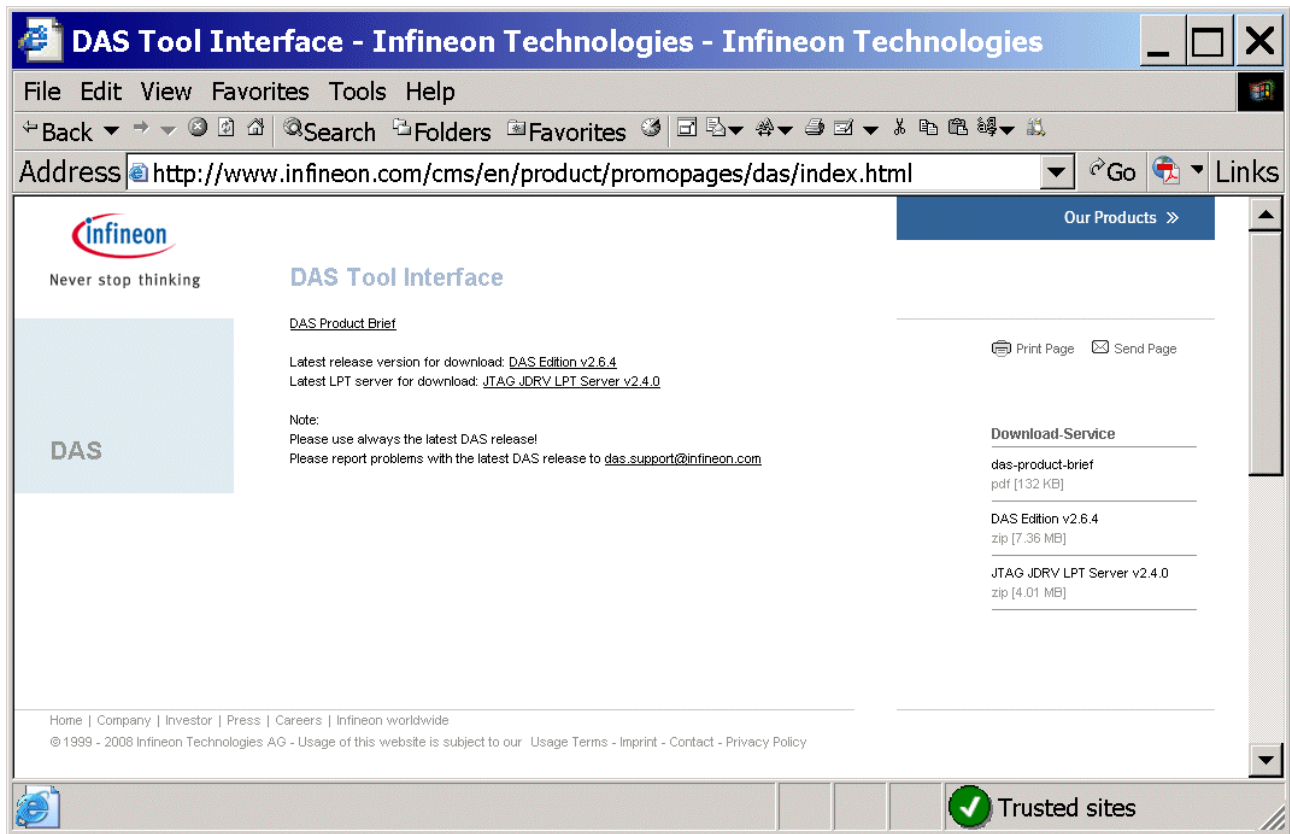
For further information, please refer to the [XE164 UConnect Manual, V.1.0](#) .  
For further information, please refer to the [XE164 UConnect Manual, V.1.1](#) .





Install the Infineon **DAS** (**D**evice **A**ccess **S**erver) Server:

Go to [www.infineon.com/DAS](http://www.infineon.com/DAS):



**Note:**

The DAS Server must be installed on your host computer!

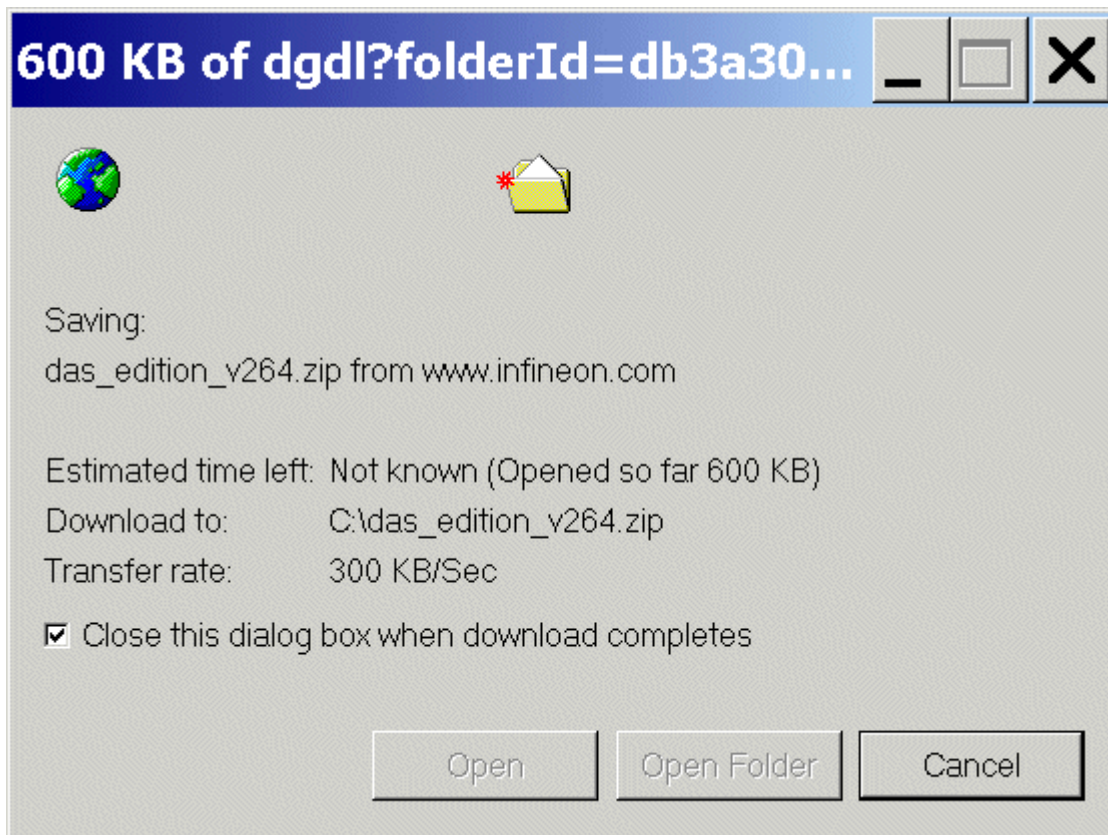
The goal of the DAS software is to provide one single interface for all types of tools.

The USB Device driver communicates with the UConnect-CAN XE164 when connected to the host computer.

The USB Device driver for the UConnect-CAN XE164 USB interface is included in the DAS software.

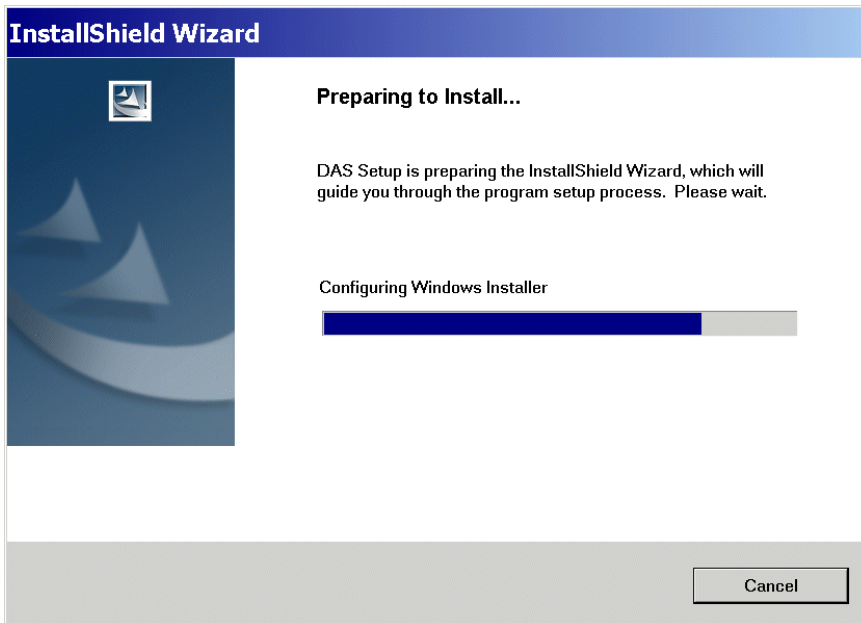
A virtual COM port driver is also included.

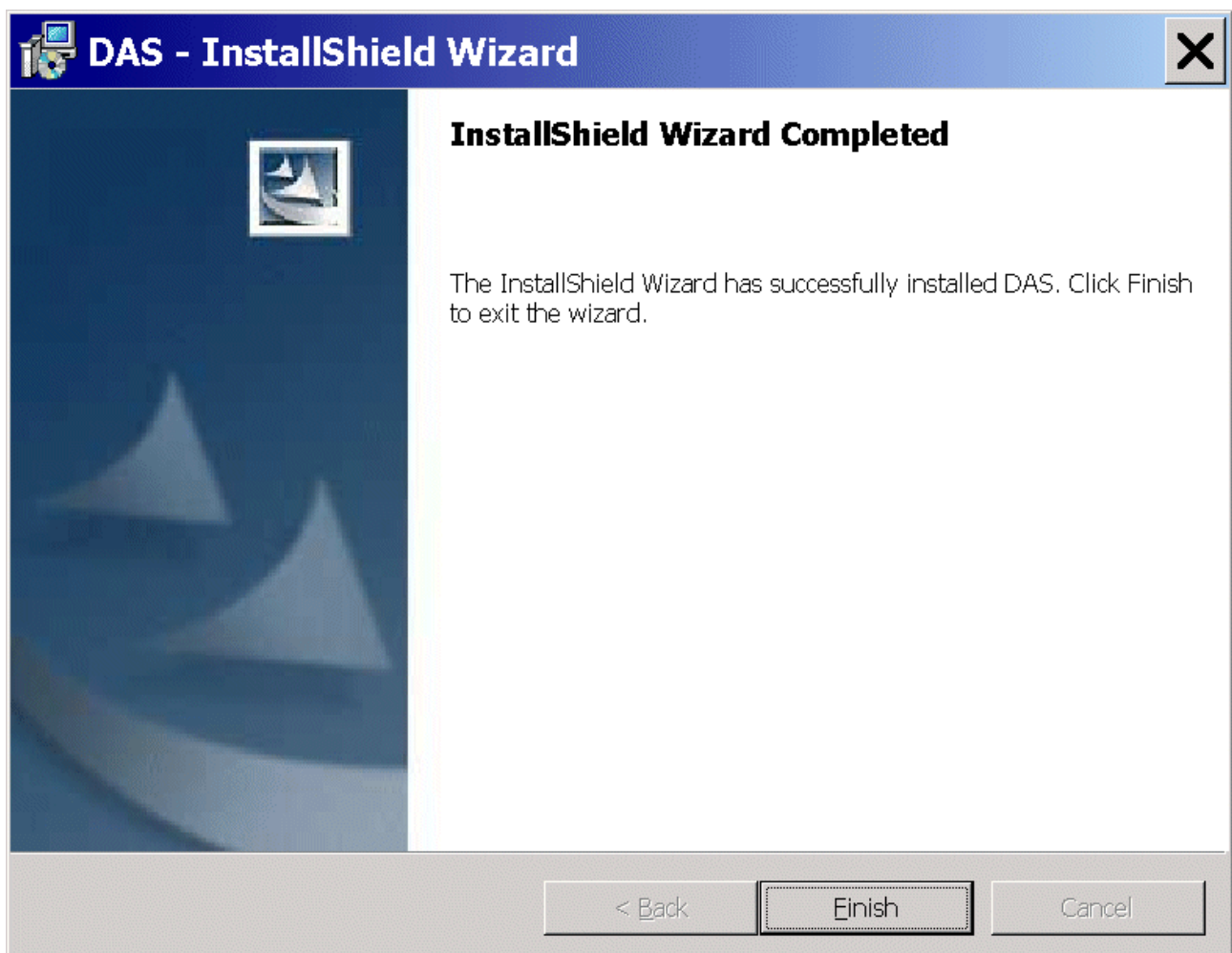
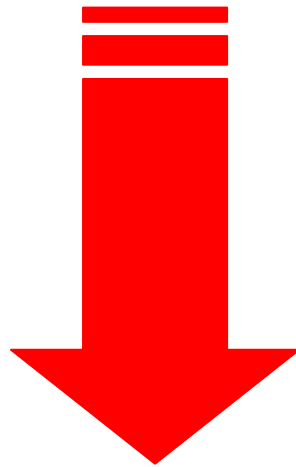
**Download** "The latest release version for download: DAS Edition v2.6.4":



**Unzip** [das\\_edition\\_v264.zip](#) and

execute “DAS\_v264\_setup.exe” to install the DAS Server.

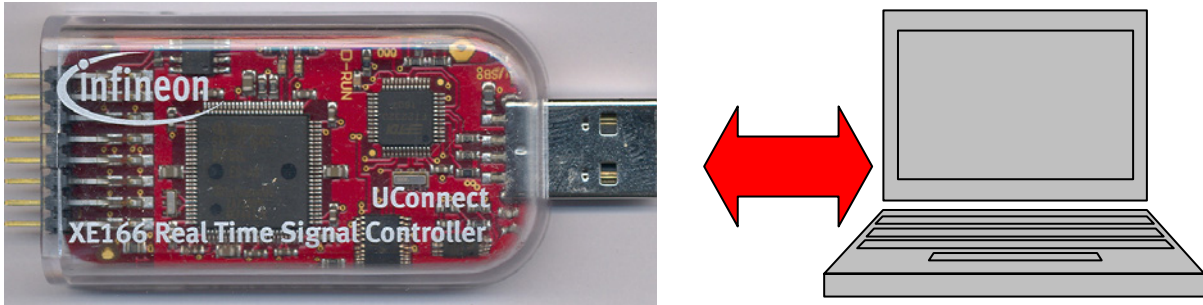




Click Finish



Connect the UConnect-CAN XE164 to the host computer:

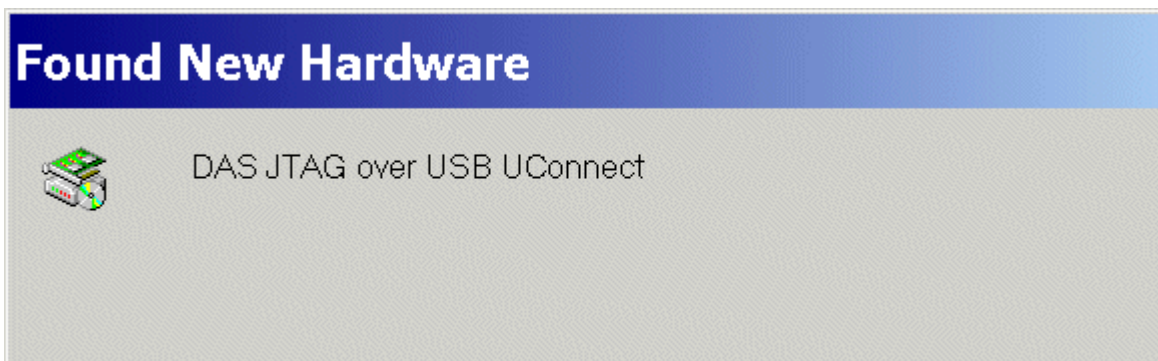


### USB Connection:

.) used for: UART communication (the USIC0\_CH0/UART/RS232/serial interface is available via USB as a virtual COM port of the second USB channel of the FTDI FT2232 Dual USB to UART/JTAG interface).

.) used for: On-Chip-Flash-Programming and Debugging (first USB channel of the FTDI FT2232 Dual USB to UART/JTAG interface).

.) the USB connection works also as the power supply.



### Note:

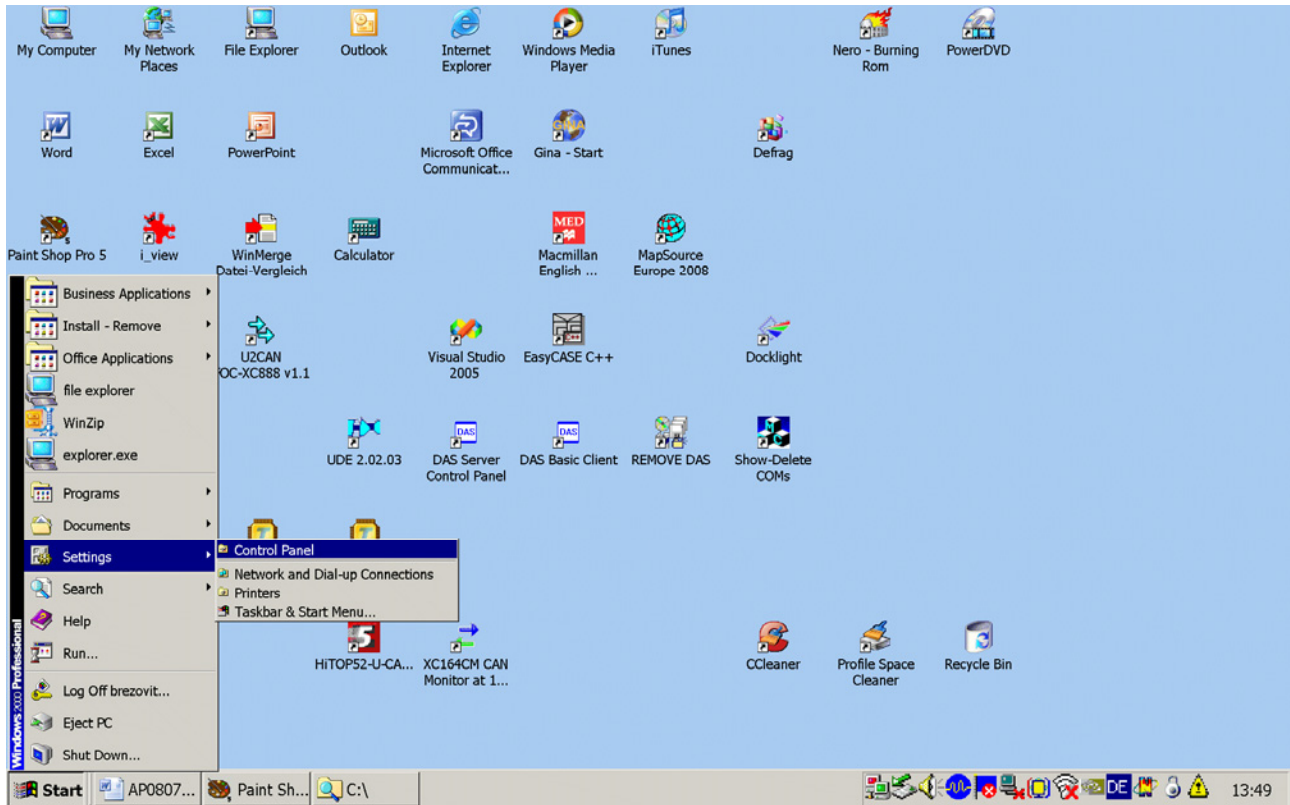
A USB driver is installed the first time while connecting the UConnect-CAN XE164 via USB to your host computer.

### Note:

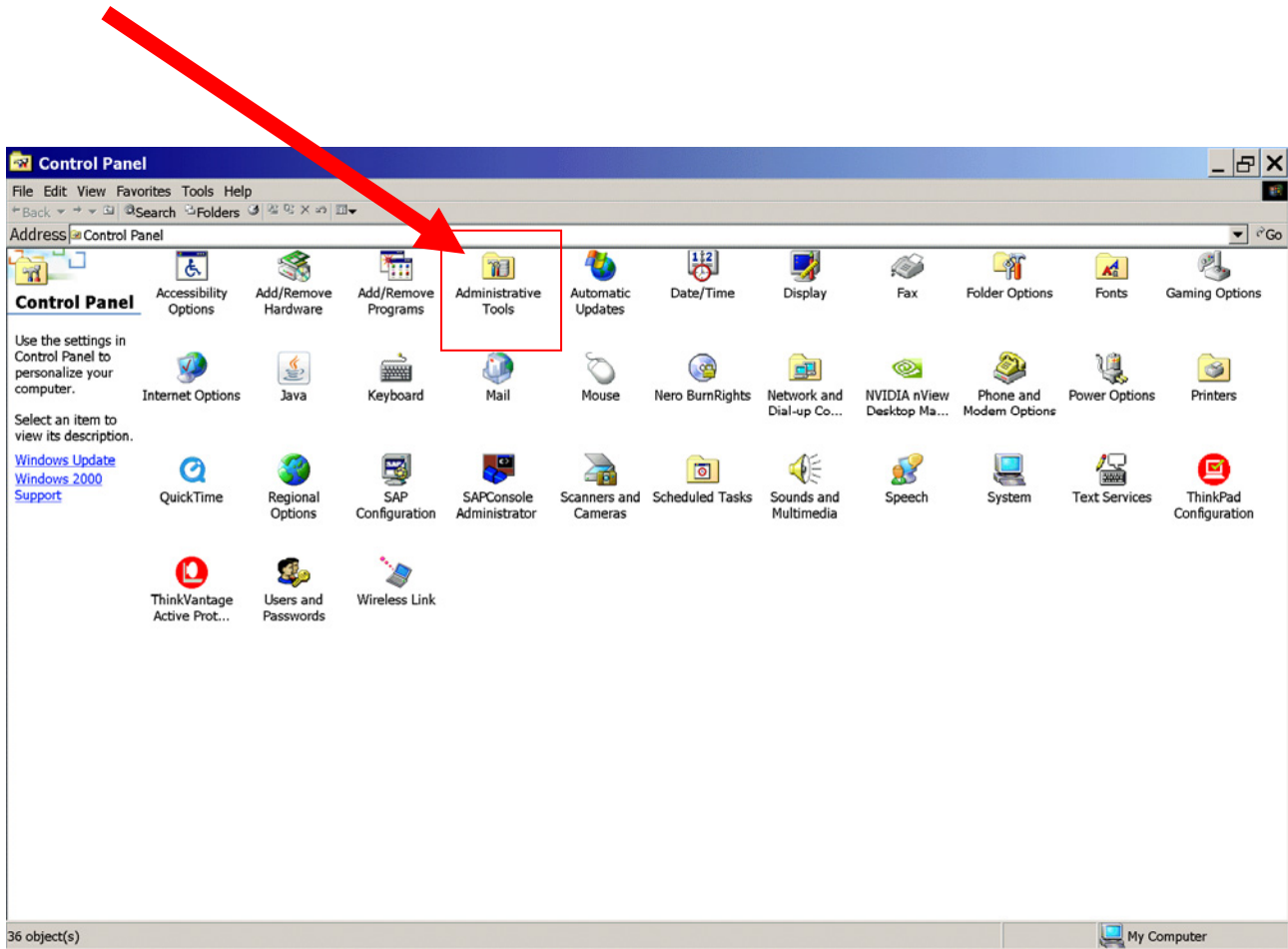
A default virtual COM Port is generated.

Using a Windows 2000 operating system, we are now going to search for the virtual COM Port which was generated after connecting our UConnect-CAN XE164:

### Start – Settings – Control Panel

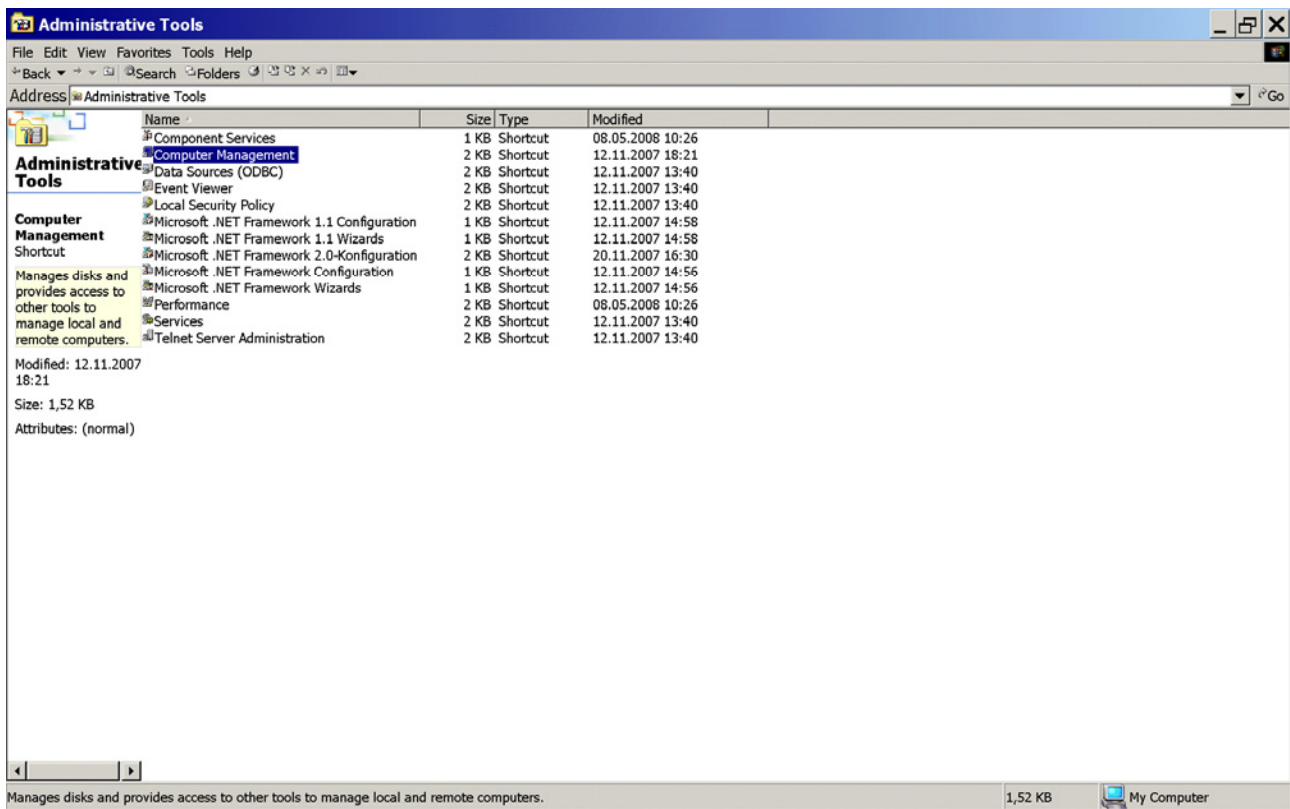


Double click: Administrative Tools

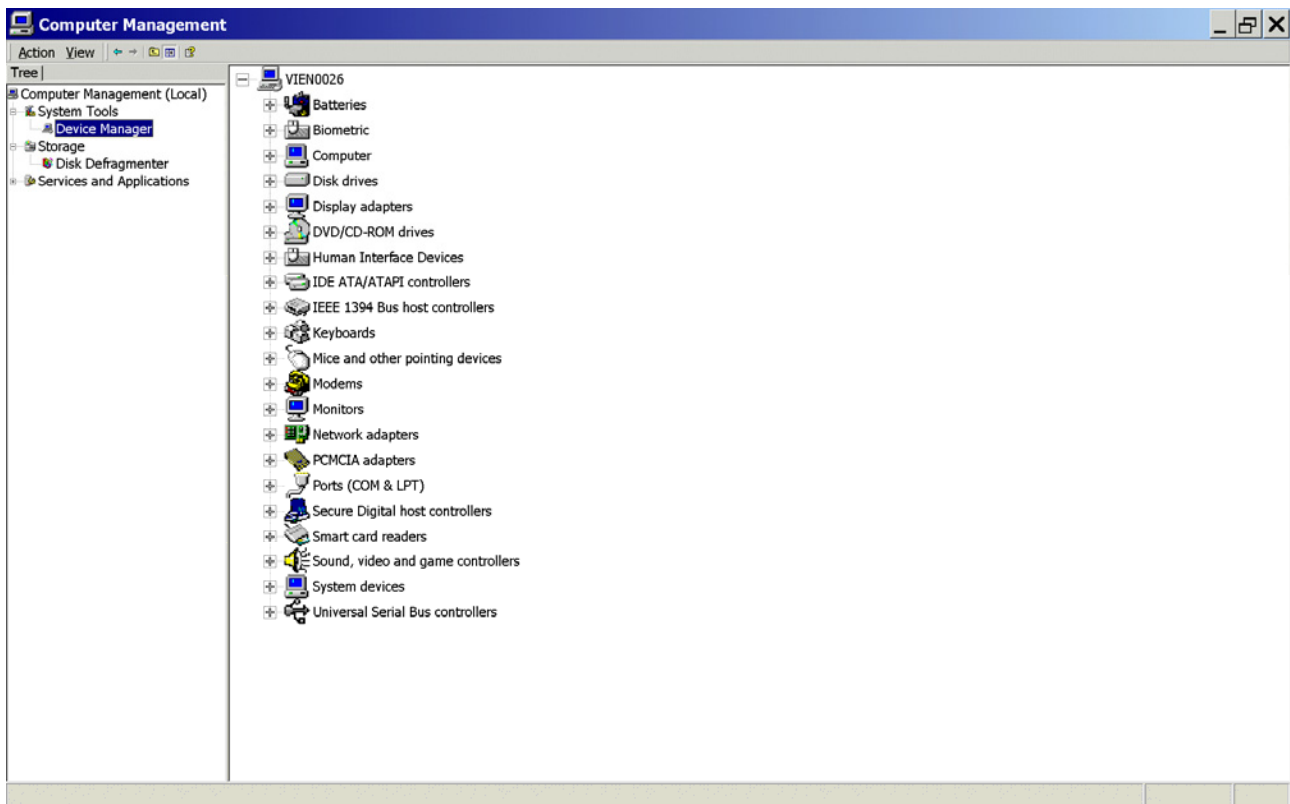




**Double click:** Computer Management

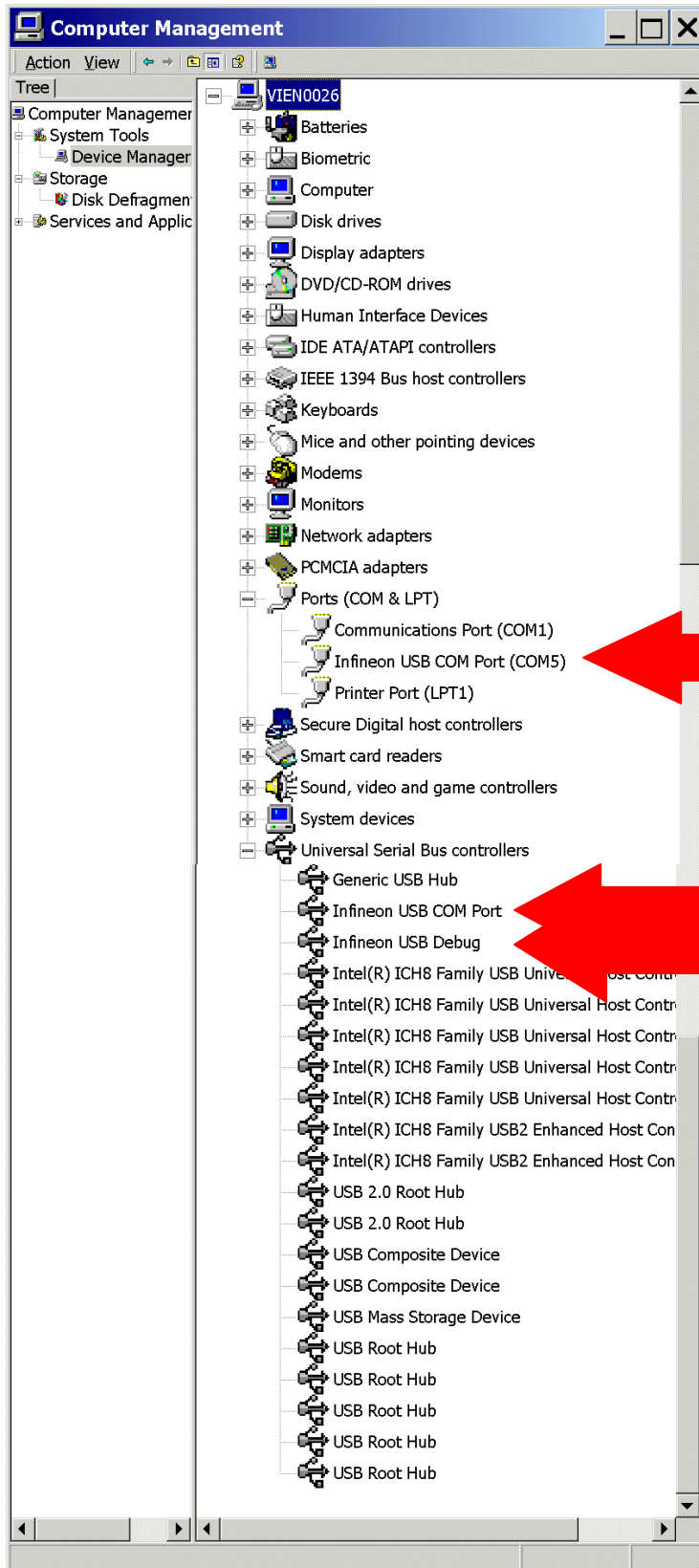


Click: Device Manager



**Expand:** Ports (COM & LPT):

**Expand:** Universal Serial Bus controllers:



**Note:**

As we can see:  
our virtual COM Port for  
UART/RS232 communication with the  
UConnect-CAN XE164 via USB is  
**COM5!**

**COM5**

## 2.) DAvE – Installation for XE16x microcontrollers:



### Install DAvE (mothersystem):

Download the DAvE-mothersystem **setup.exe** @ <http://www.infineon.com/DAvE>

Title	Date	Version	Size
<b>Tool Package</b>			
 DAvE - Mothersystem - latest version	05 Feb 2007	V2.1 r24	14.8 MB
 DAvE - Mothersystem	04 Jul 2006	V2.1 r23	15.1 MB

and execute **setup.exe** to install DAvE .

### **Note:**

Abort the installation of Acrobat Reader.



Install the XE164 microcontroller support/update (XE16xx\_Series.dip):

1.)

Download the DAVe-update-file (.DIP) for the required microcontroller


@ <http://www.infineon.com/DAvE>

## DAvE

### DAvE for the Infineon XE166 microcontroller Family

DAvE supports the 16-bit derivatives as DAVe Integration Package (DIP) files.

- All the latest DIPs are available for FREE download.


Company Name and Weblink	Product Name	XE167 Series	XE164 Series	Description
 <a href="#">DAvE home</a>	DAvE	x	x	DAvE stands for Digital Application Virtual Engineer and is Infineon Technologies' code generator for their range of 8, 16 and 32 Bit Microcontrollers. It provides initialization, configuration and driver code to ease programming for beginners as well as experts.

Documents

Contact us


### Document Types

▼ Development Tools

Title	Date	Version	Size
<b>Development Tools</b> ^			
 XE16xx-Series DIP file for DAVe (Microcontroller Configuration Tool) (XE16xx_Series_v2.0.zip)	20 May 2008	v2.0	4.2 MB

**Unzip** the zip-file “[XE16xx\\_Series\\_v2\[1\].0.zip](#)” and save “[XE16xx\\_Series.dip](#)”  
 @ e.g. [C:\DAvE\XE16x-2008-05-29\XE16xx\\_Series.dip](#).

2.)

Start DAVe - (  click DAVe )

3.)

View

Setup Wizard

Default: • Installation

Forward>

Select: • I want to install products from the DAVe's web site

Forward>

Select: C:\DAVe\XE16x-2008-05-29

Forward>

Select: Available Products

click ✓ XE16xx\_Series

Forward>

Install

End

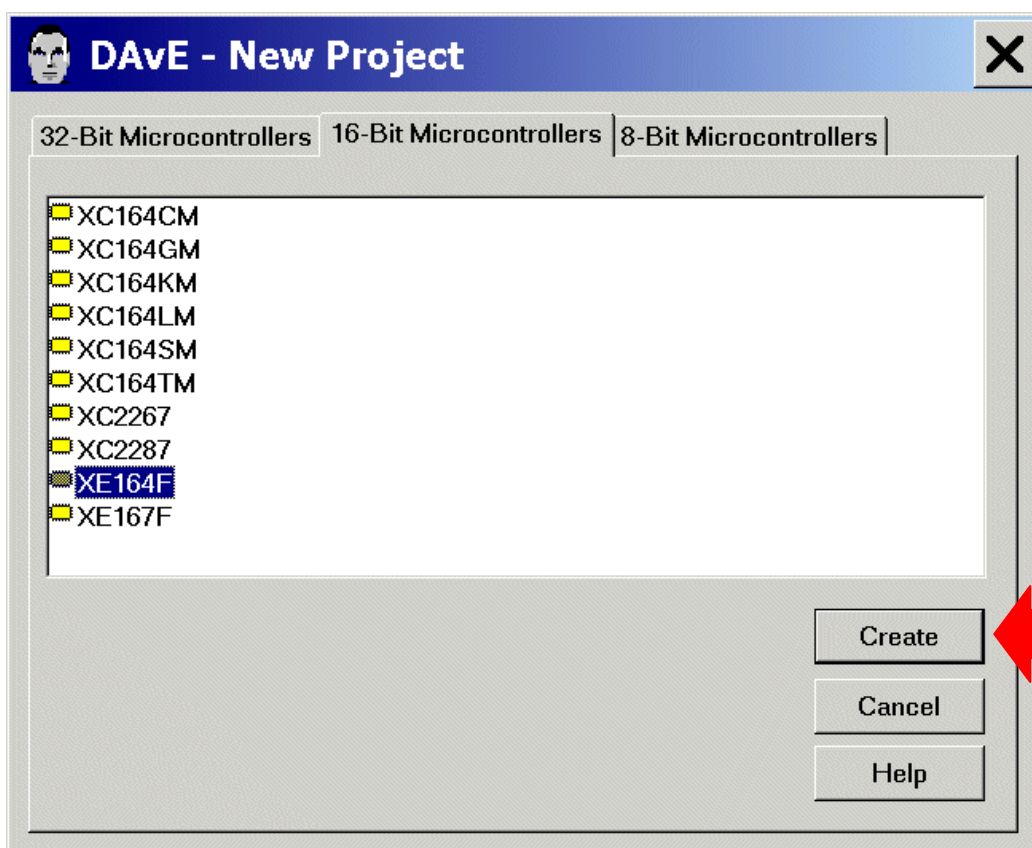
4.) DAVe is now ready to generate code for the XE16x microcontrollers.

### 3.) DAvE - Microcontroller Initialization after Power-On:



Start the program generator DAvE and select the XE164 microcontroller:

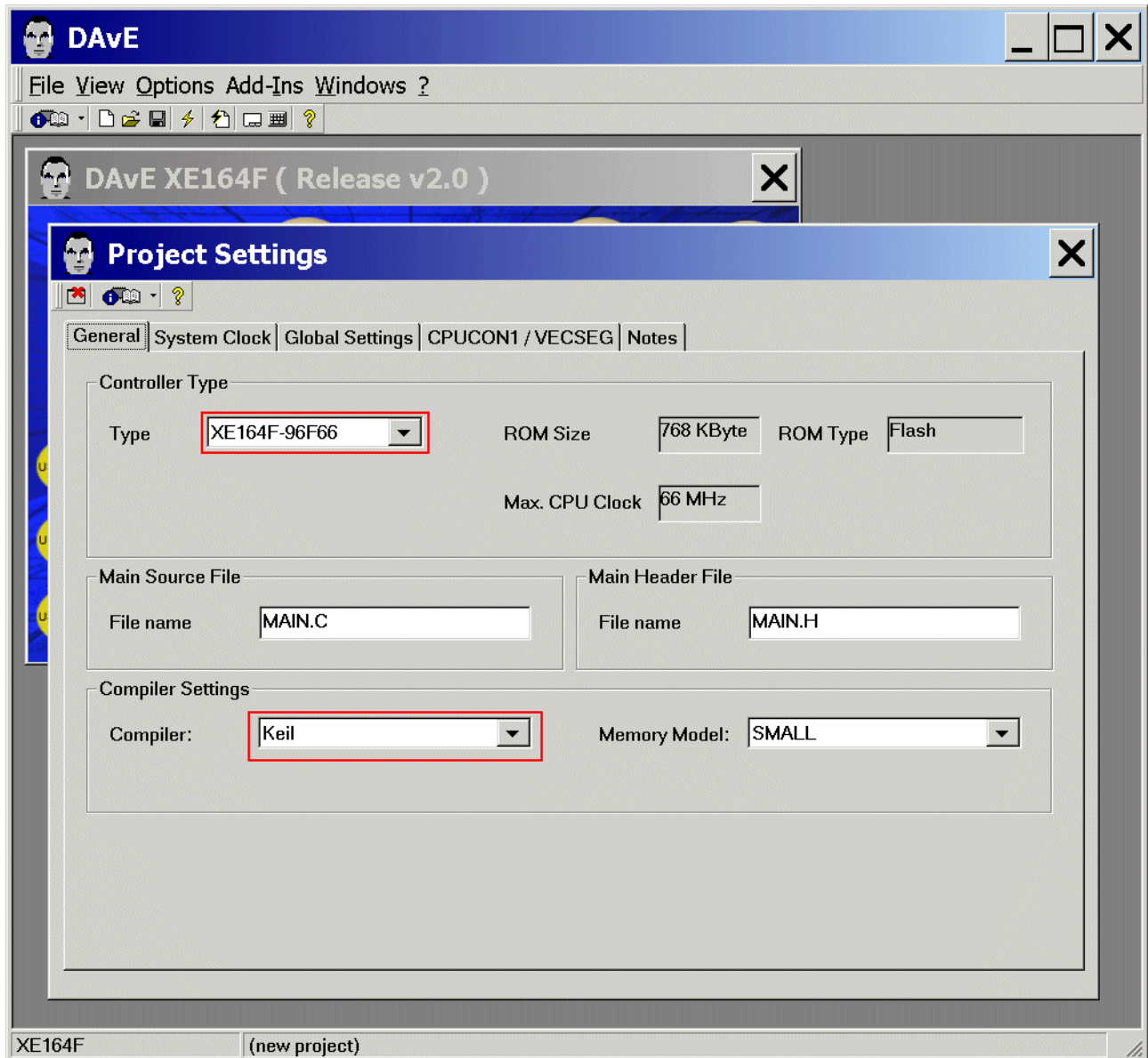
File  
New  
16-Bit Microcontrollers  
Select **XE164F**  
Create





Choose the Project Settings as you can see in the following screenshots:

General: Controller Type: Type: check/select XE164F-96F66  
General: Compiler Settings: Compiler: check/choose Keil

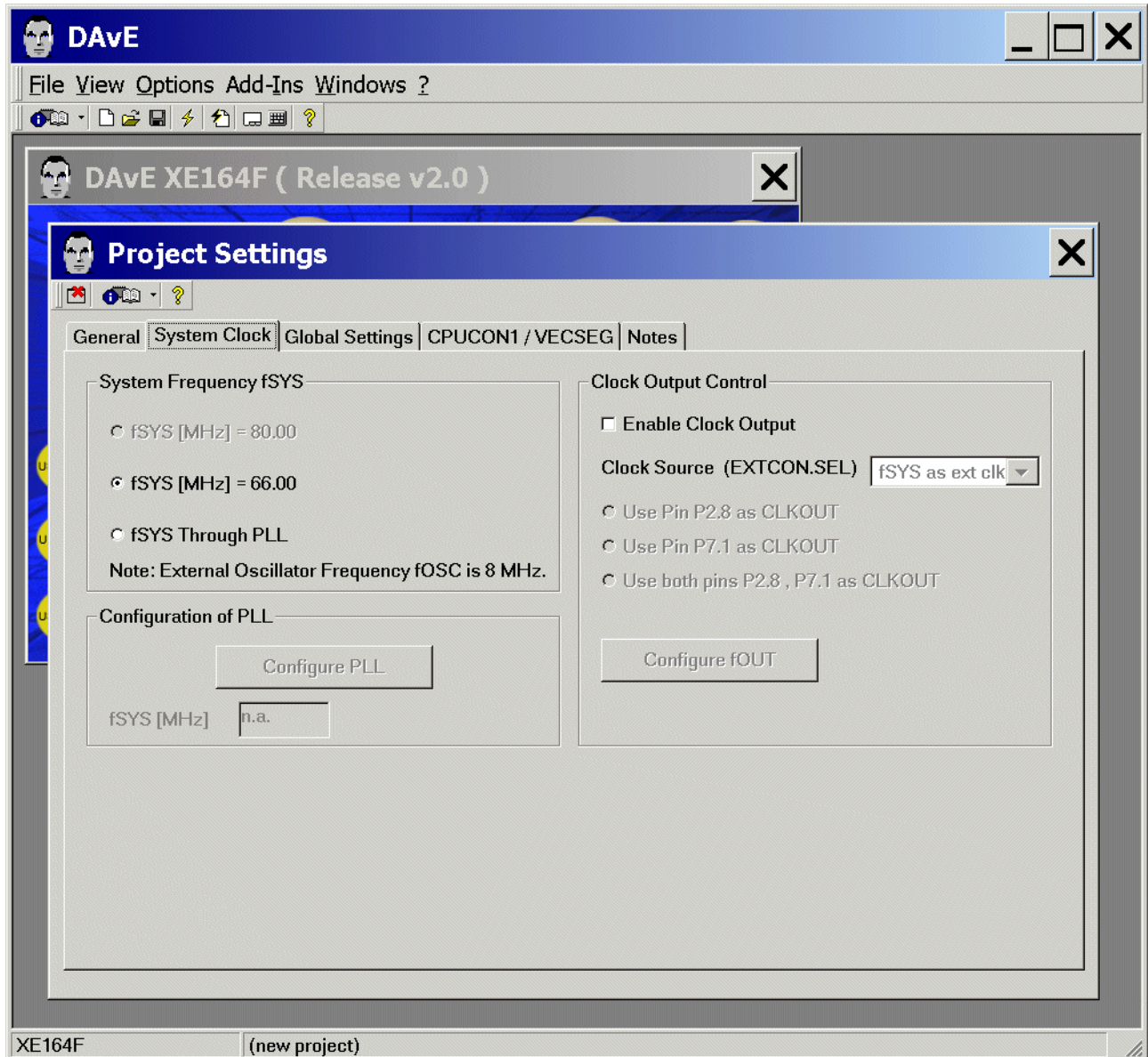


**Note:**

You can change file names (e.g. MAIN.C, MAIN.H) anytime.



System Clock: (do nothing)

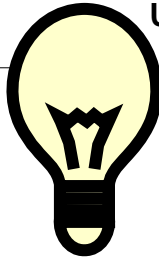


**Note (Source: DAVE):**

Configuration of the System Clock:

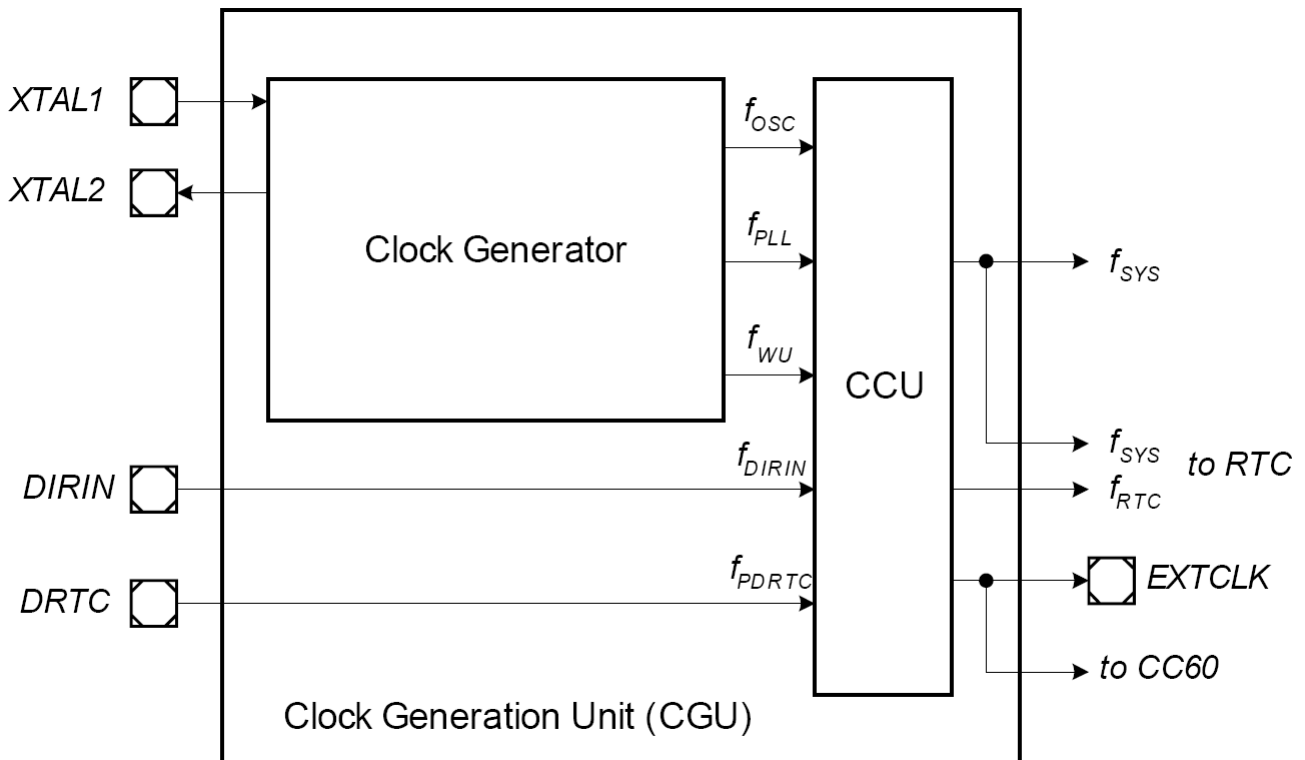
- VCO clock used, input clock is connected
- input frequency is 8,00 MHz (XTAL1)
- configured system frequency is 66,00 MHz
- system clock is 66.00 MHz





Additional information: Clock System (Source: User's Manual):

Clock Generation Unit (CGU) Block Diagram:



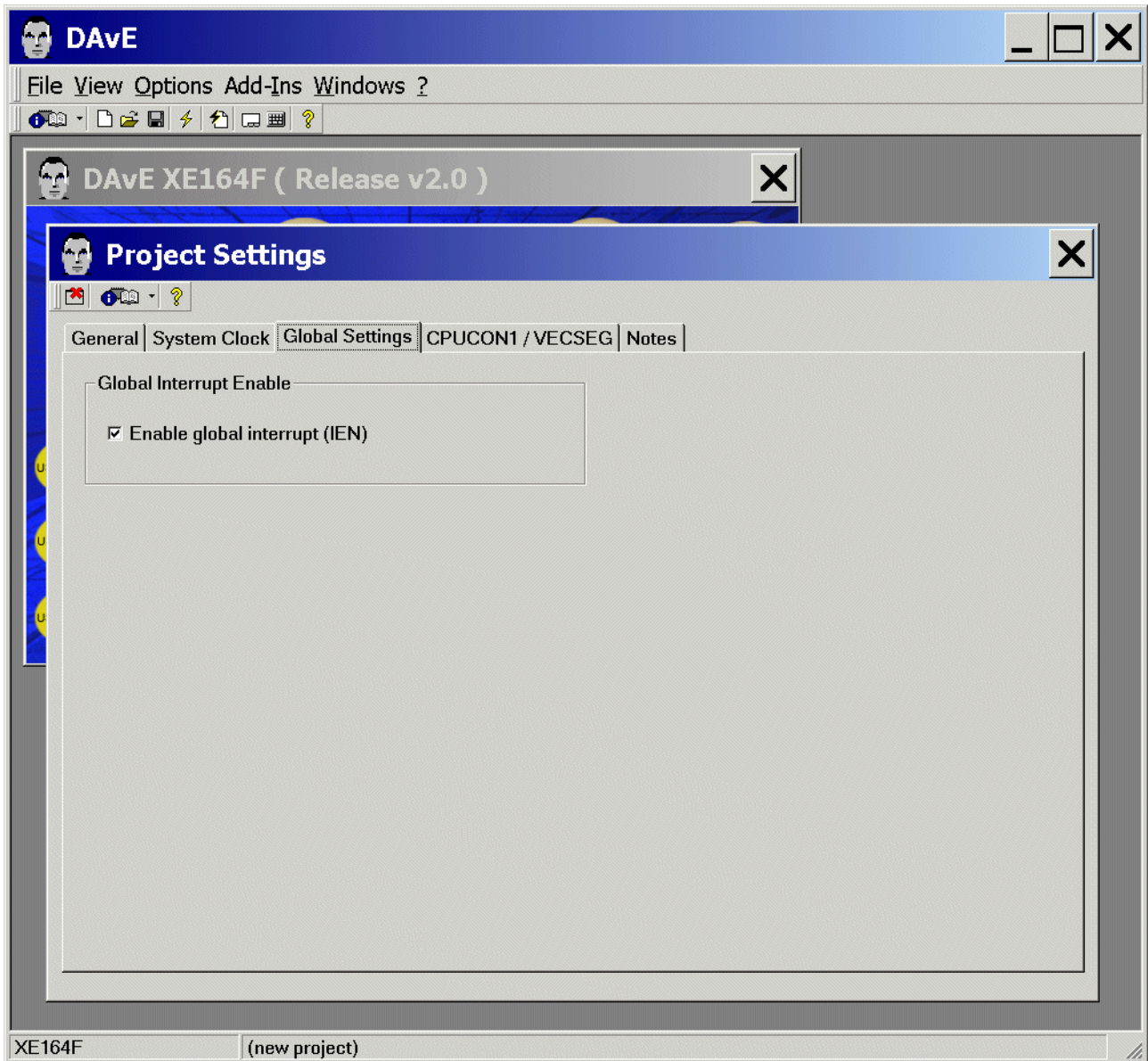
**Note:**

The CGU can convert a low-frequency external clock to a high-speed internal clock, or can create a high-speed internal clock without external input.

The system clock  $f_{SYS}$  is generated out of four selectable clocks:

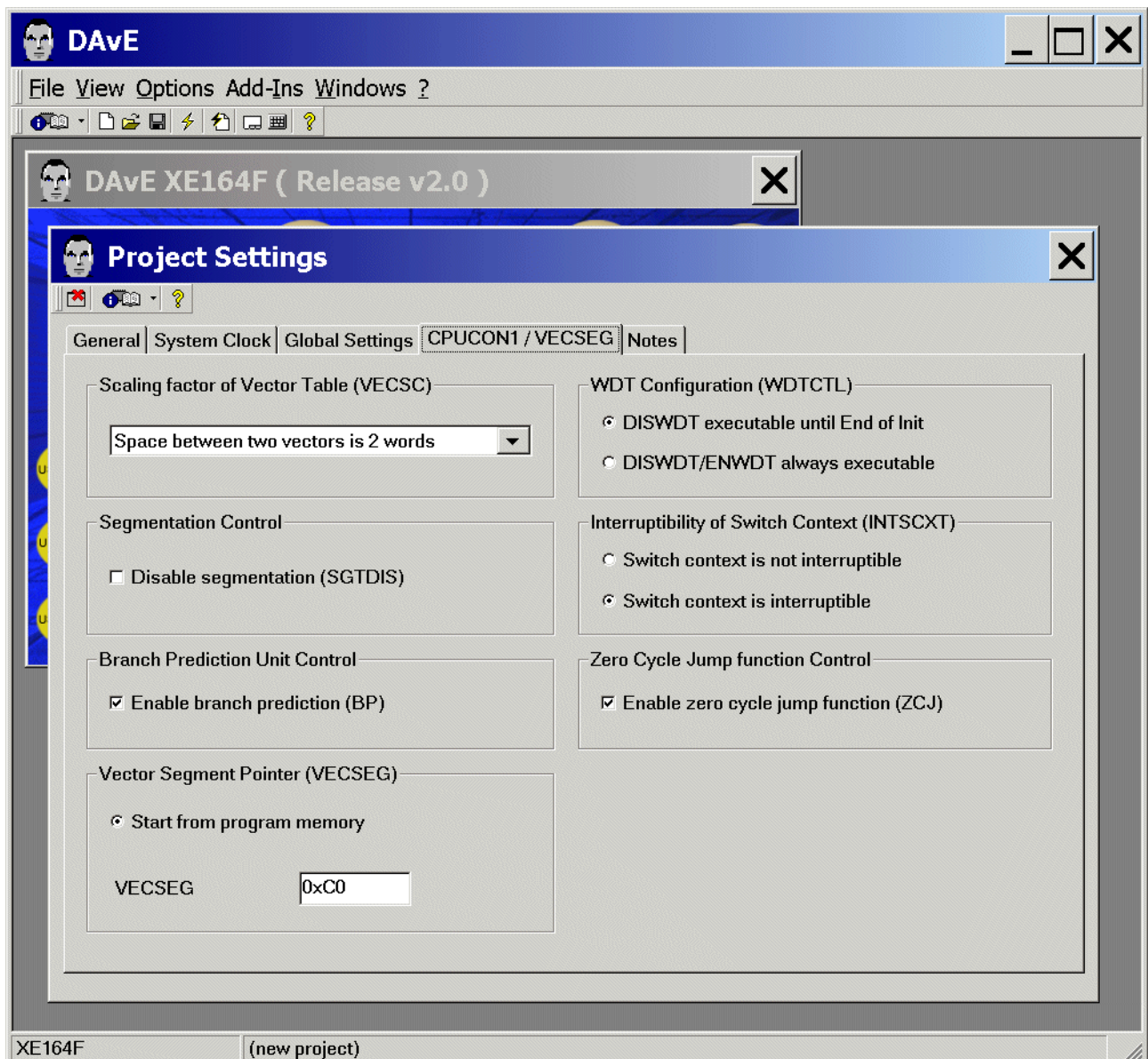
- PLL clock  $f_{PLL}$
- Wake-Up clock  $f_{WU}$
- The Direct Clock  $f_{OSC}$ , from pin XTAL1
- Input DIRIN as Direct Clock Input  $f_{DIR}$

**Global Settings:** (do nothing. Do not change configuration)

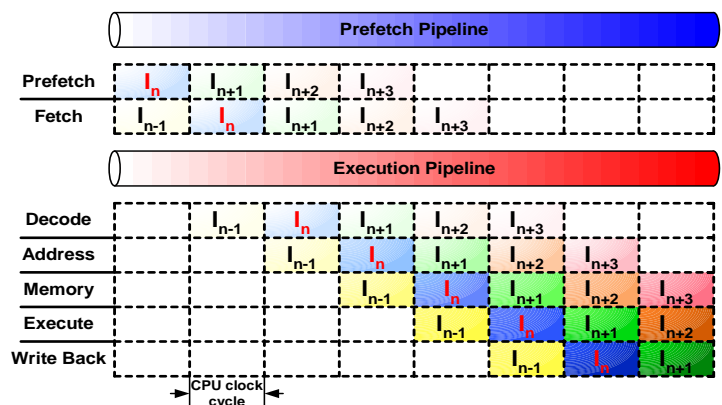




CPUCON1/VECSEG: (do nothing)



**Note:**  
We should not change the pipeline behaviour.






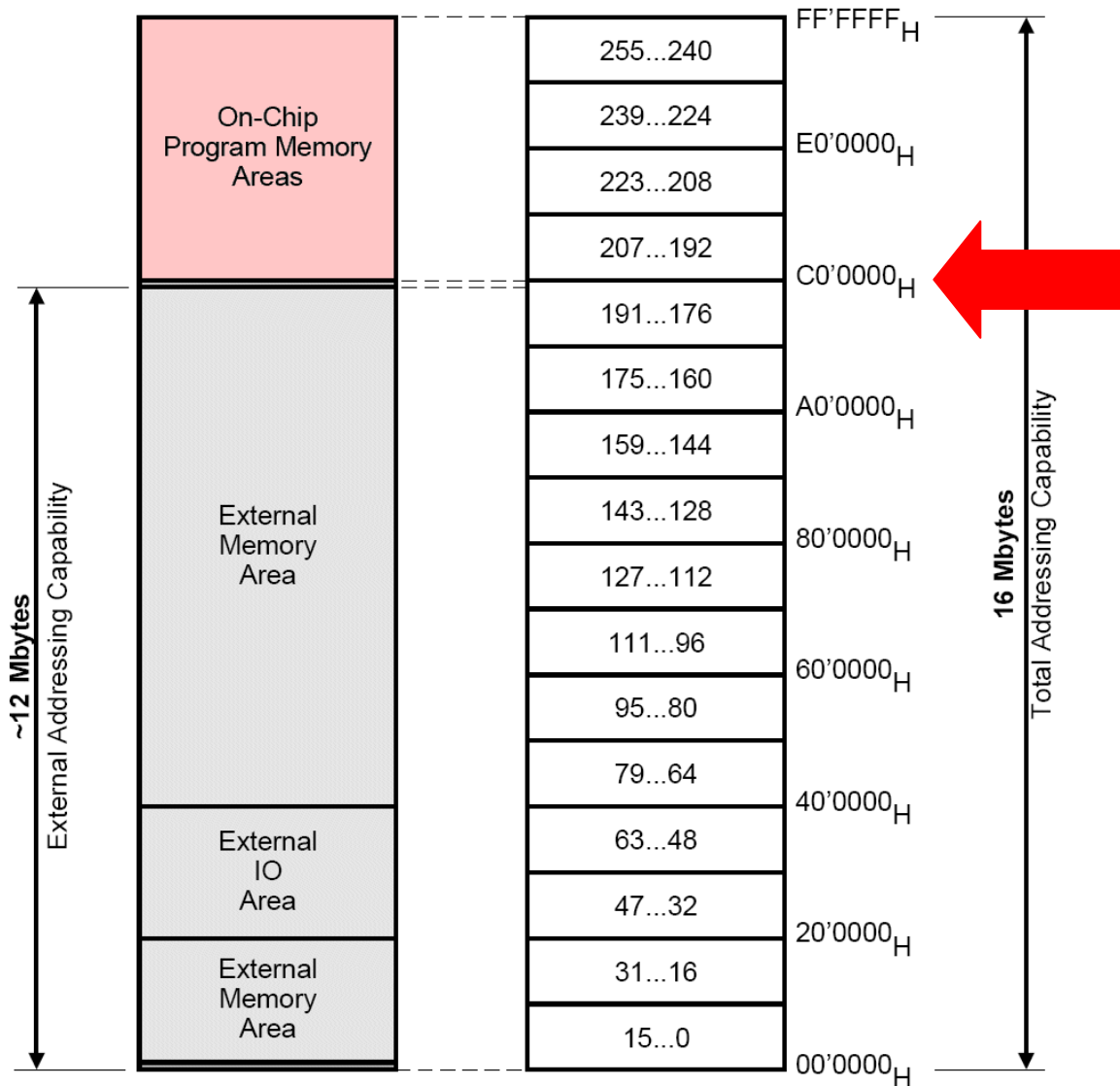


Additional information: Start from program memory (Source: User's Manual):

Vector Segment Pointer (VECSEG)


☒ Start from program memory

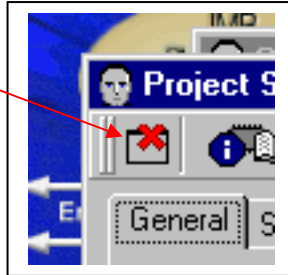
VECSEG  



Total Address Space  
16 Mbytes, Segments 255...0

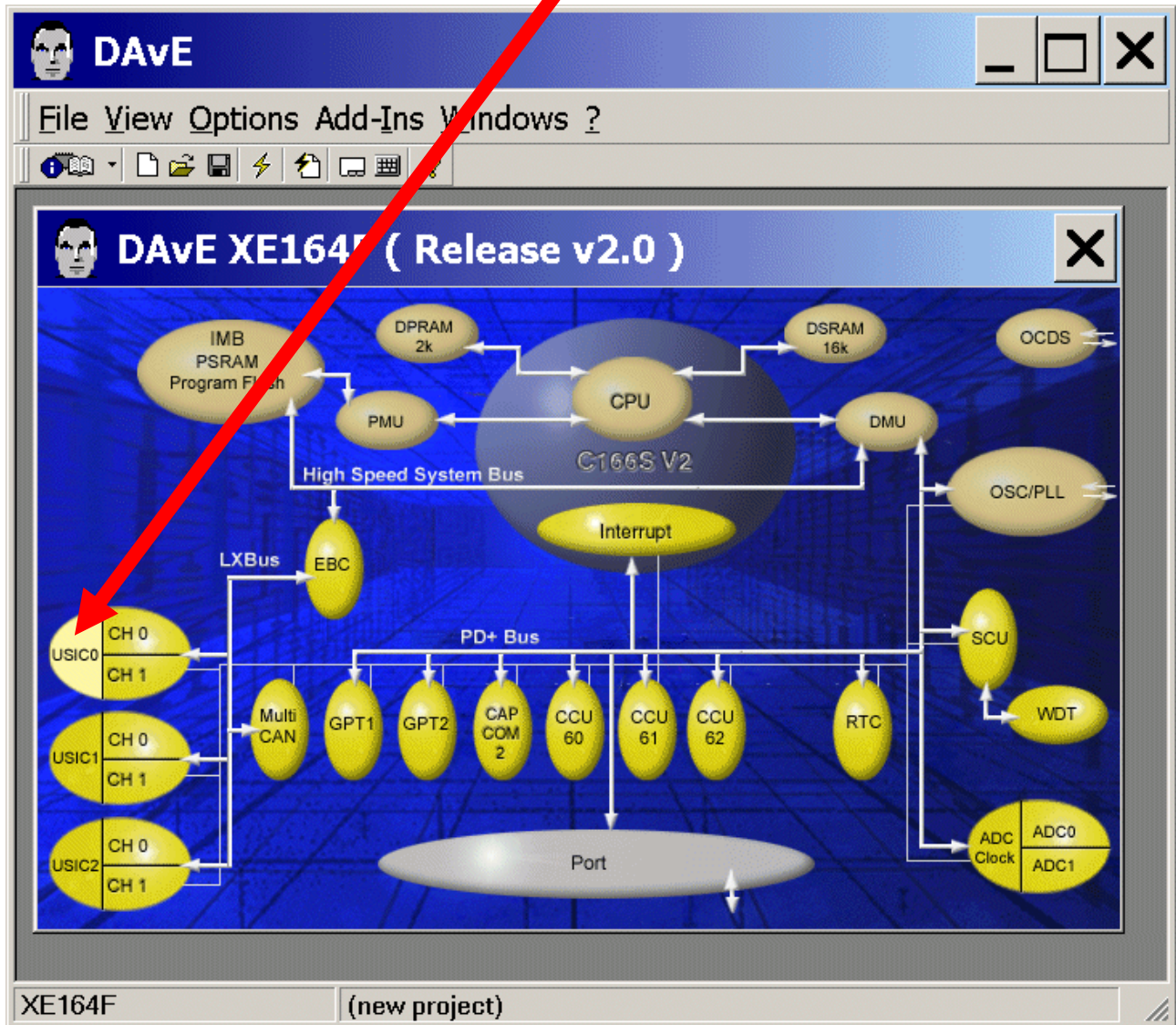
**Notes:** If you wish, you can insert your comments here.

**Exit** and **Save** this dialog now by clicking  the close button:

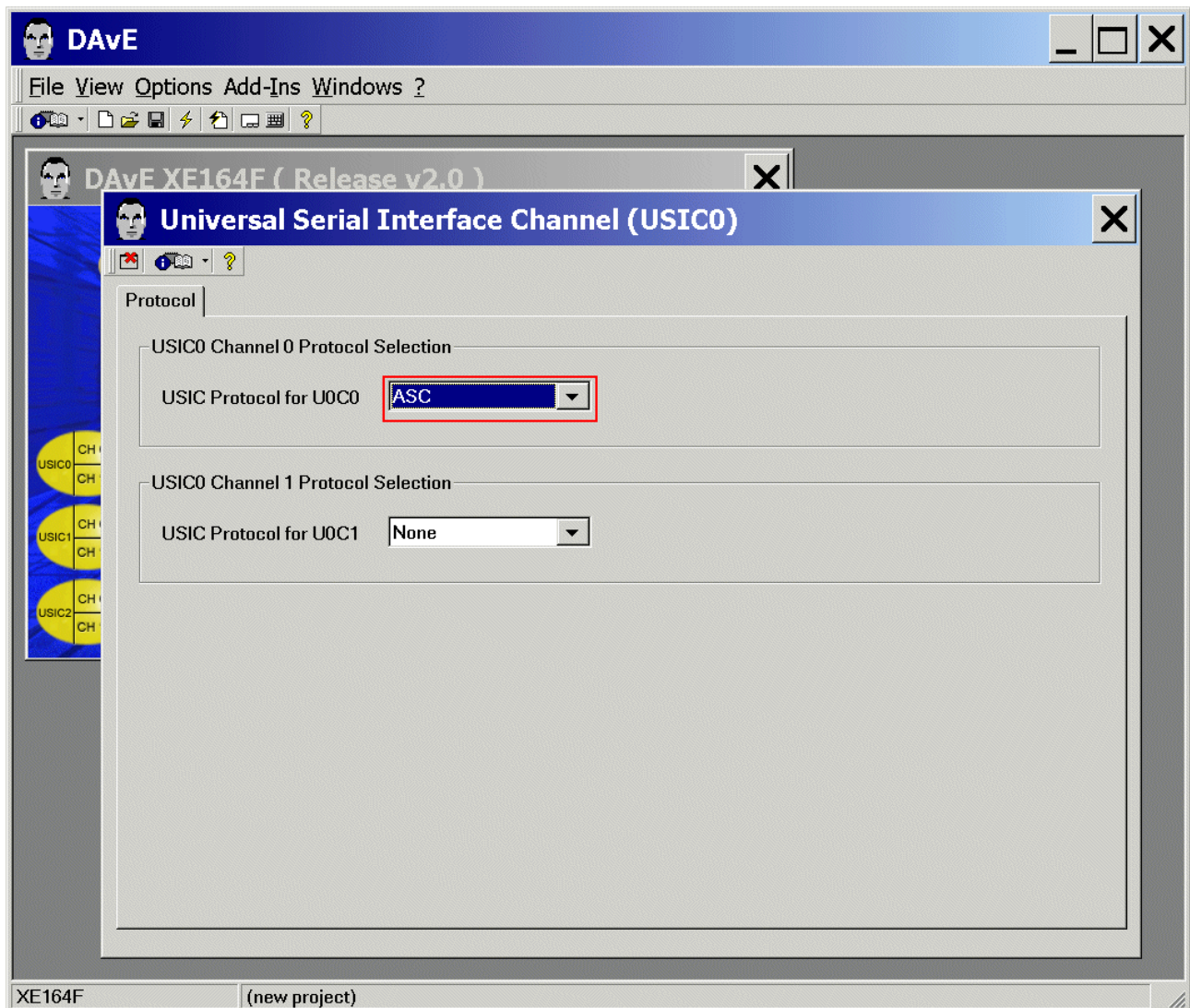



Configuration of the serial interface "ASC0" / UART / USIC0 CH0 / U0C0:

The configuration window/dialog can be opened by clicking the specific block/module (USIC0).



Protocol: USIC0 Channel 0 Protocol Selection: USIC Protocol for U0C0: select ASC

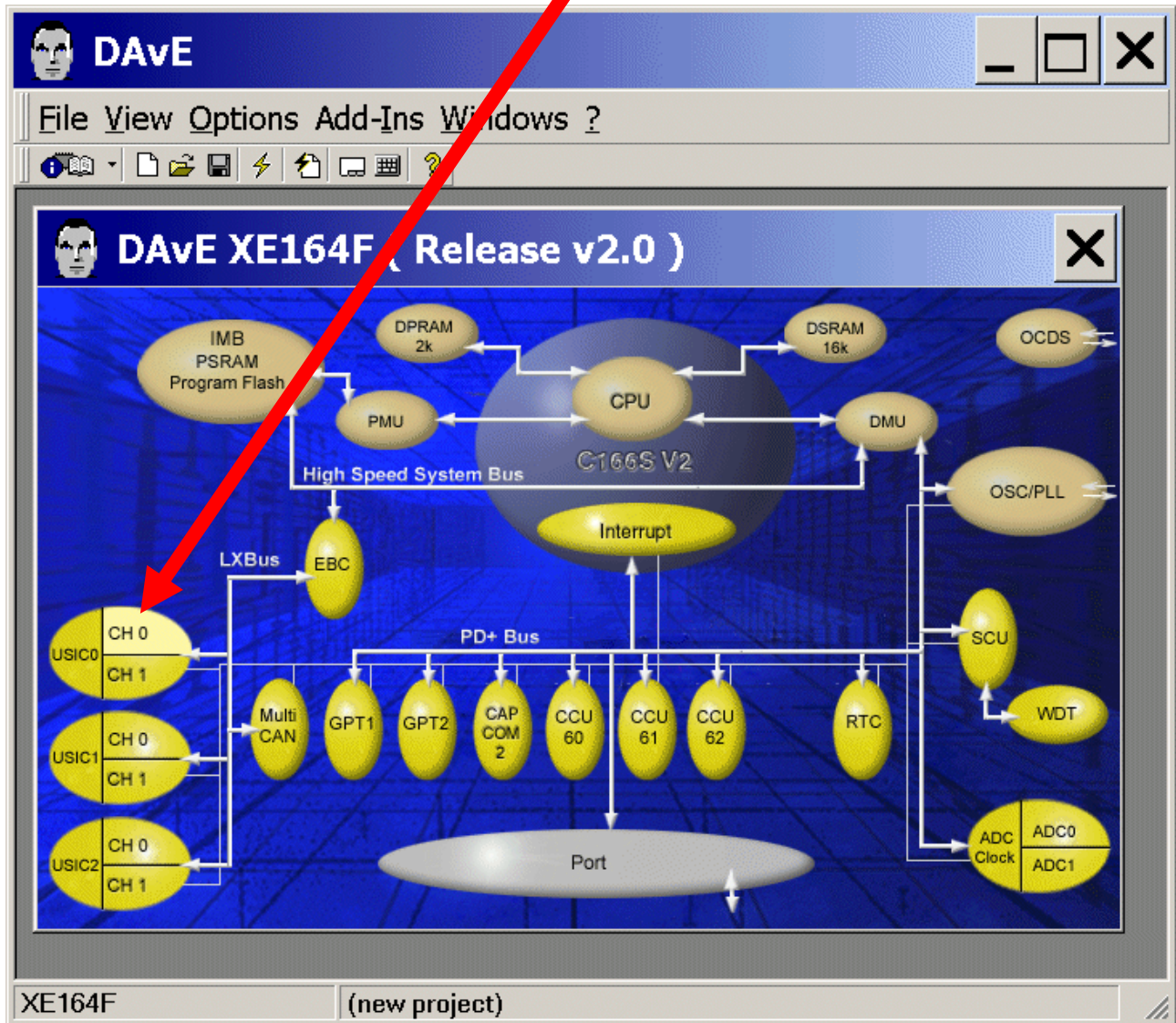


Exit and Save this dialog now by clicking  the close button.



Configuration of the serial interface USIC0\_CH0 / U0C0:

The configuration window/dialog can be opened by clicking the specific block/module (CH 0).

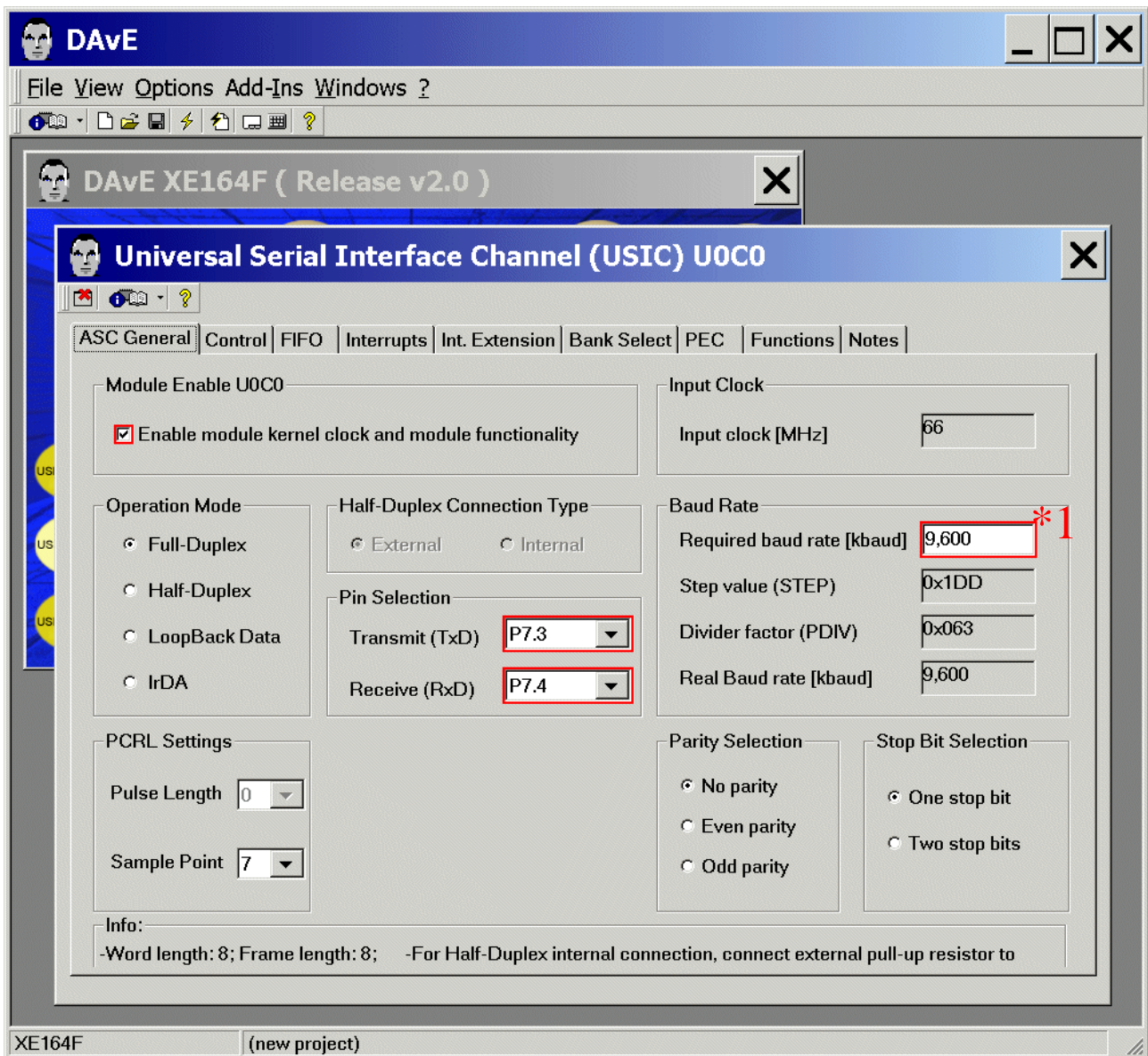


ASC General: Module Enable U0C0: click ☒ Enable module kernel clock and module functionality

ASC General: Pin Selection: Transmit (TxD): select P7.3

ASC General: Pin Selection: Receive (RxD): select P7.4

ASC General: Baud Rate: Required baud rate [kbaud]: insert 9,600 <ENTER>



Note (\*1):

Validate each alphanumeric entry by pressing <ENTER>.



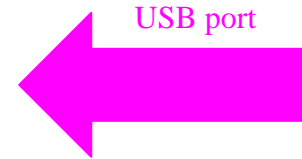
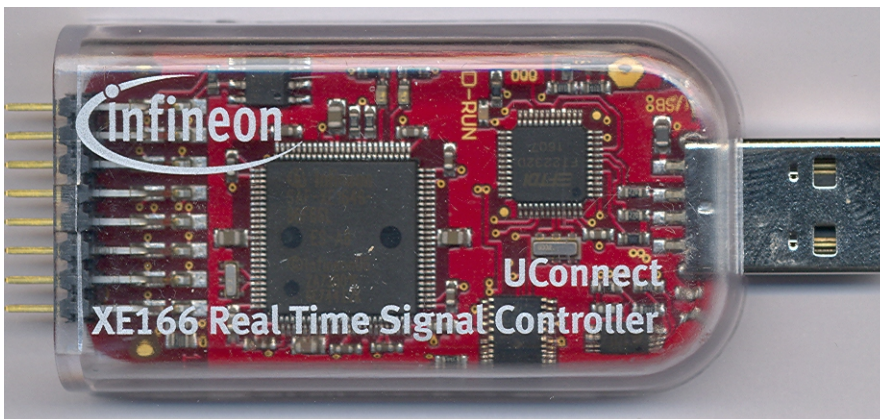




Additional information: RS232 serial interface:

**Note:**

The RS232 serial interface (USIC\_0\_Channel\_0 pins P7.3 and P7.4) is available via the [USB port](#) which converts the TTL-UART-signals to USB-signals (using a virtual COM port of the second USB channel of the FTDI FT2232 Dual USB to UART/JTAG interface).



USB port



Additional information: Standard UART Pins (Source: User's Manual):

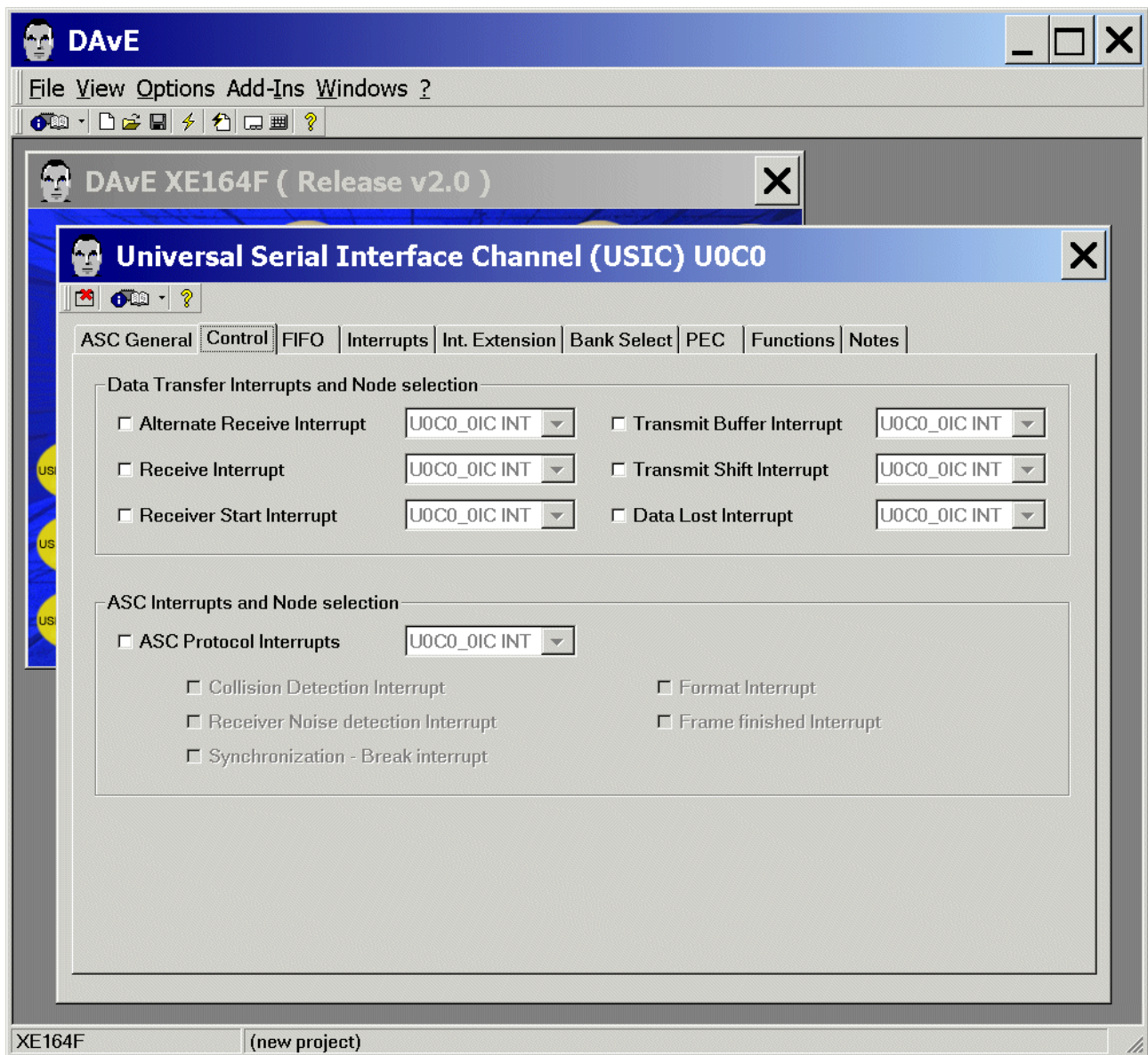
**Table 10-10 Configuration Data for Bootstrap Loader Modes**

Bootstrap Loader Mode	Configuration on P10.3-0 <sup>1)</sup>	Receive Line from Host	Transmit Line to Host	Transferred Data
Standard UART	x110 <sub>B</sub>	RxD = P7.4	TxD = P7.3	32 Bytes
Sync. Serial	1001 <sub>B</sub>	MRST = P2.4	MTSR = P2.3 SCLK = P2.5 SLS = P2.6	n Bytes; 1 ... 65,280
MultiCAN	x101 <sub>B</sub>	RxDC0 = P2.6	TxDC0 = P2.5	8 × n Bytes

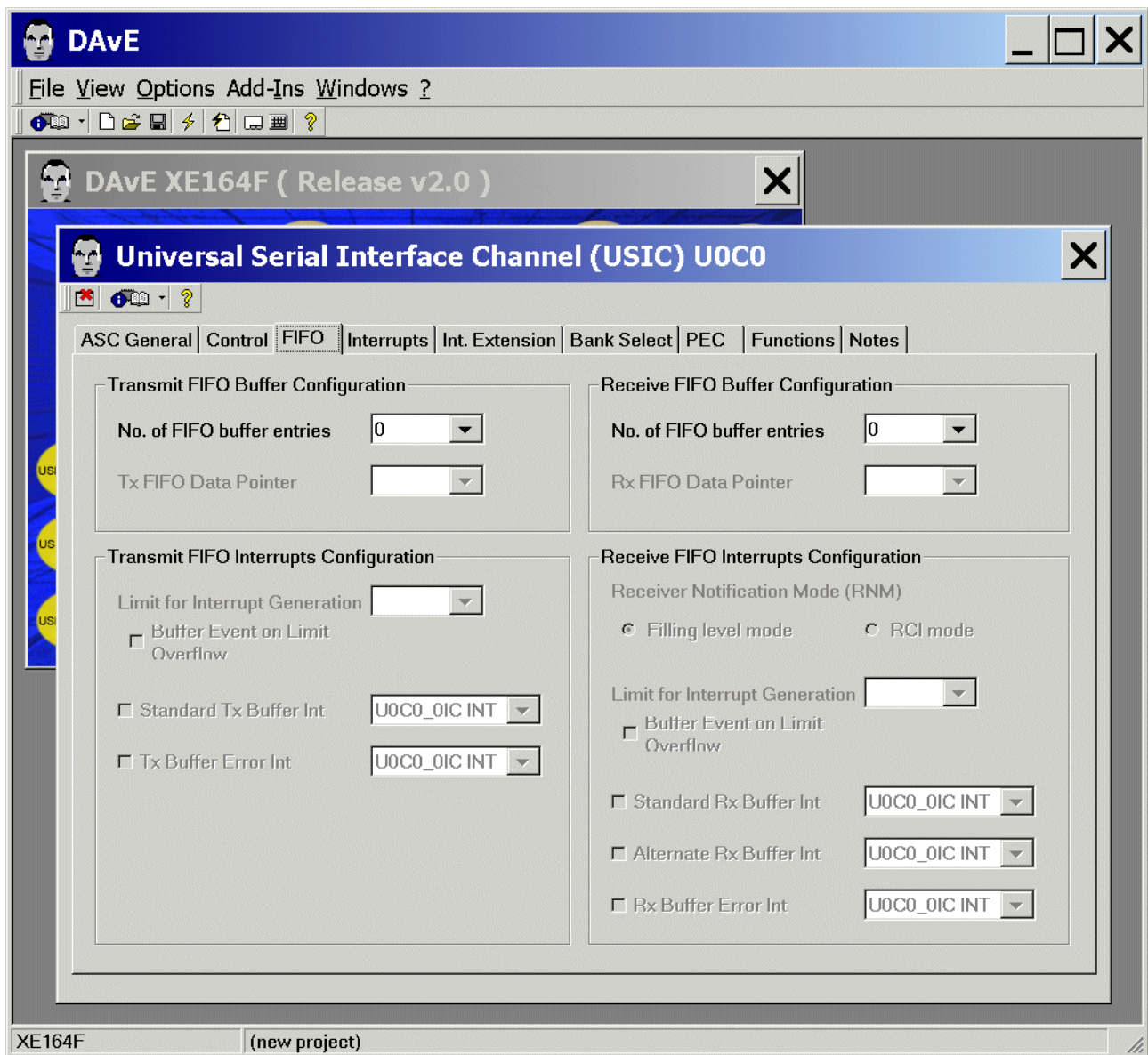
1) x means that the level on the corresponding pin is irrelevant.



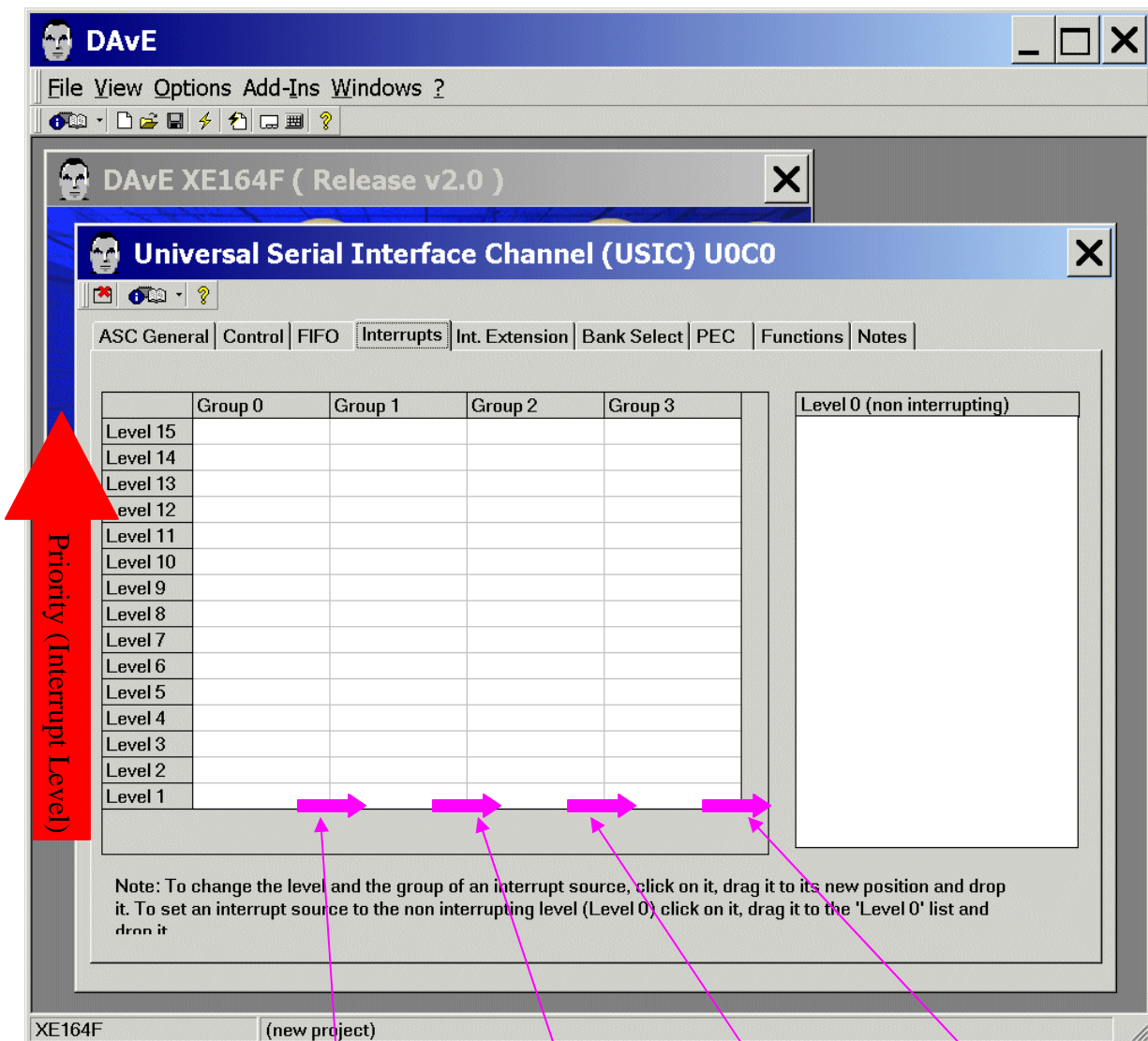
Control: (do nothing)



FIFO: (do nothing)



Interrupts: (do nothing)



Group Priority 0 → Group Priority 1 → Group Priority 2 → Group Priority 3 →

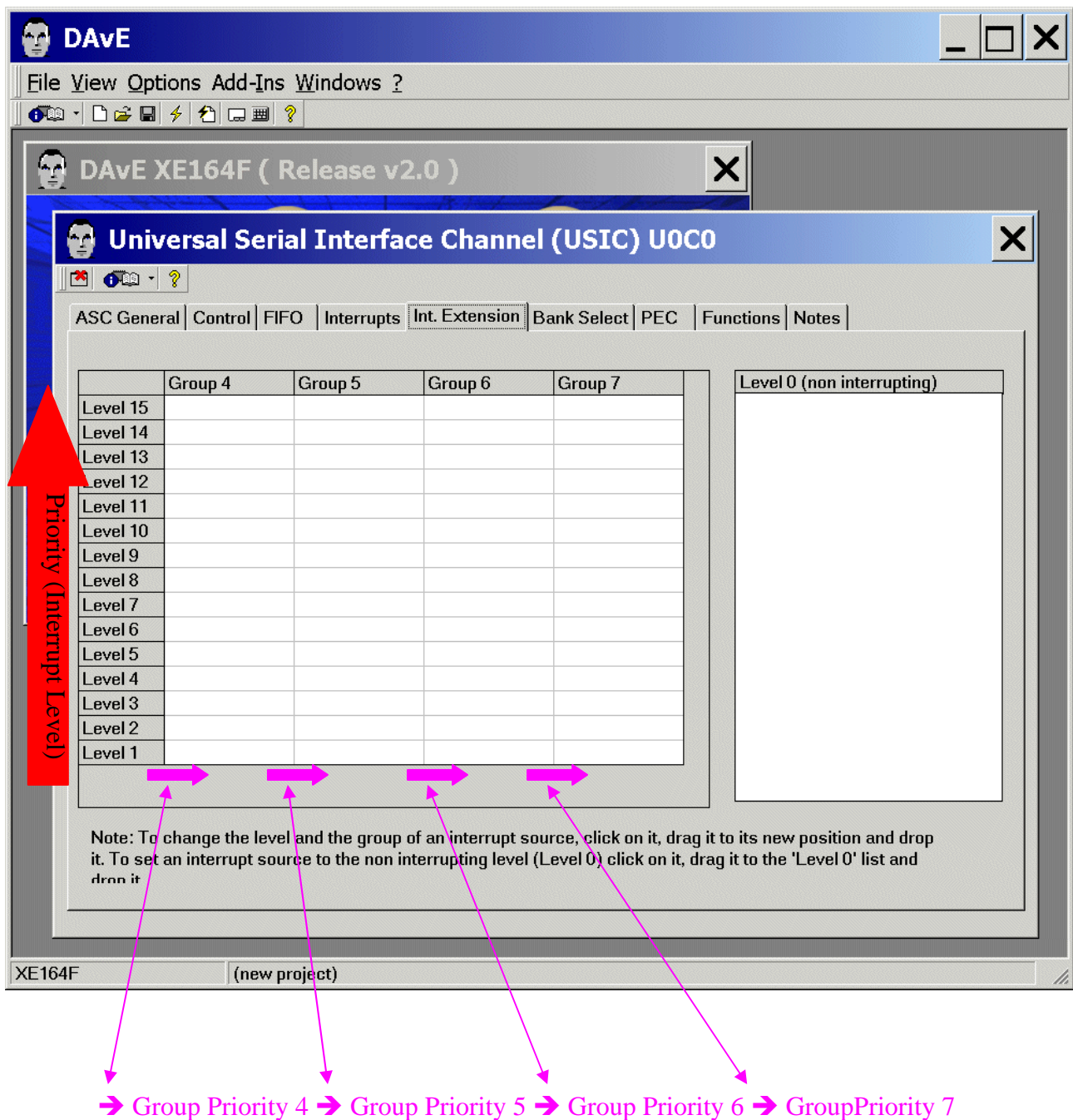


**Note:**

For the serial communication with a terminal program (e.g. Docklight, [www.docklight.de](http://www.docklight.de)) running on your host computer the myprintf function is used. The myprintf function uses Software-Polling-Mode therefore we do not need to configure any interrupts.



Int. Extension: (do nothing)



**DAve**

File View Options Add-Ins Windows ?

DAve XE164F ( Release v2.0 )

**Universal Serial Interface Channel (USIC) UOC0**

ASC General Control FIFO Interrupts **Int. Extension** Bank Select PEC Functions Notes

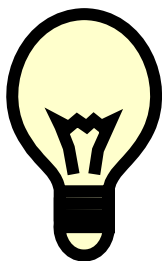
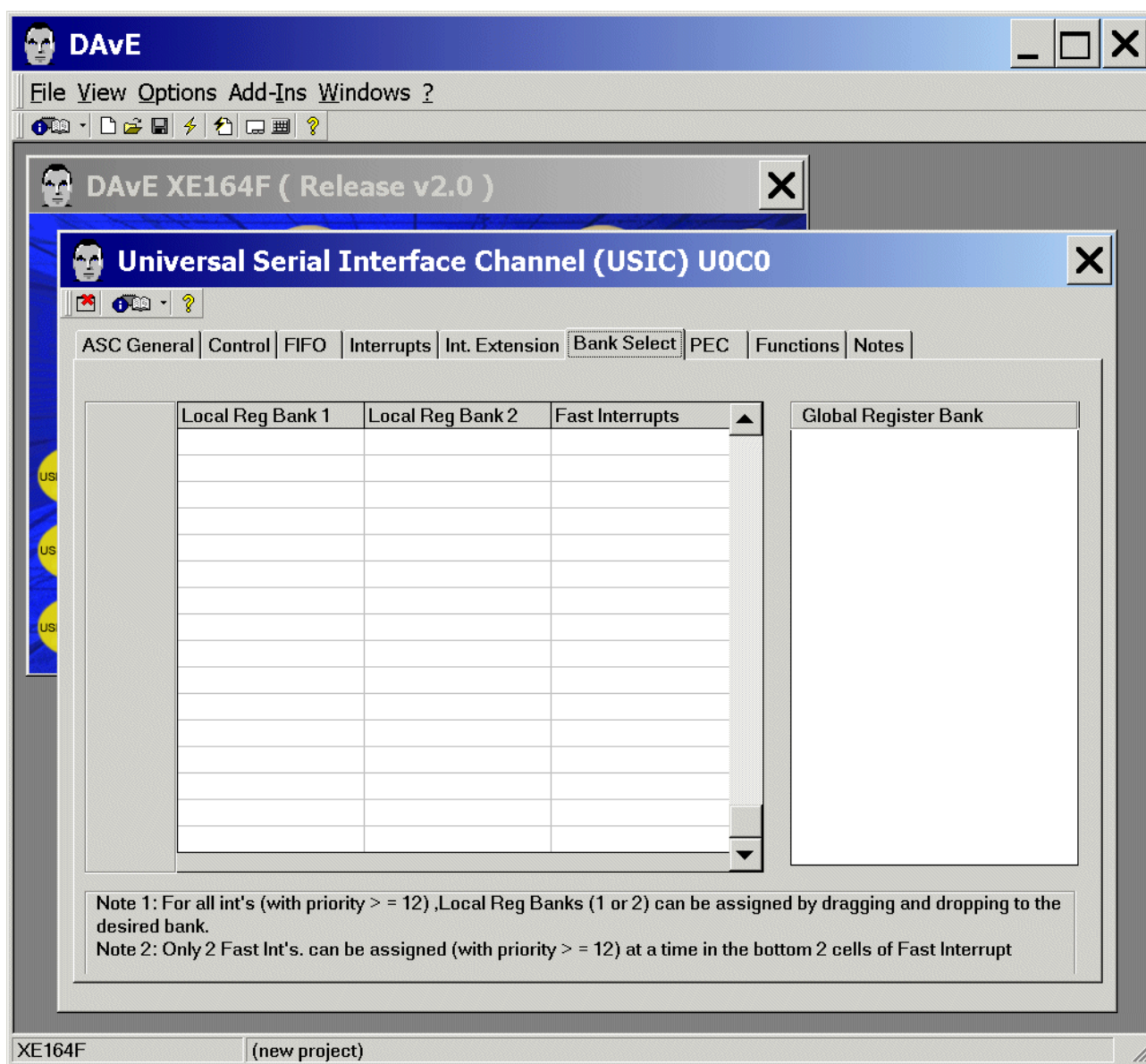
	Group 4	Group 5	Group 6	Group 7	Level 0 (non interrupting)
Level 15					
Level 14					
Level 13					
Level 12					
Level 11					
Level 10					
Level 9					
Level 8					
Level 7					
Level 6					
Level 5					
Level 4					
Level 3					
Level 2					
Level 1					

Note: To change the level and the group of an interrupt source, click on it, drag it to its new position and drop it. To set an interrupt source to the non interrupting level (Level 0), click on it, drag it to the 'Level 0' list and drop it.

→ Group Priority 4 → Group Priority 5 → Group Priority 6 → GroupPriority 7

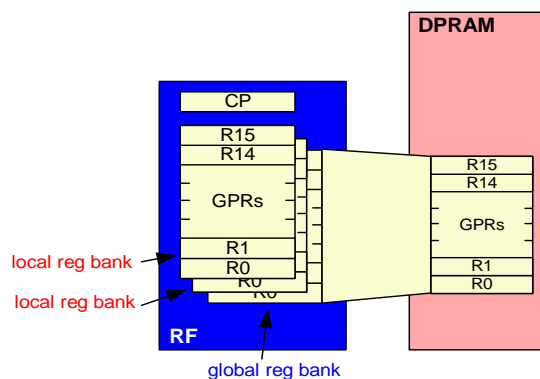


Bank Select: (do nothing)

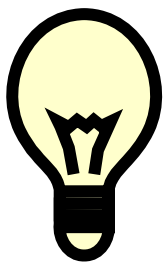
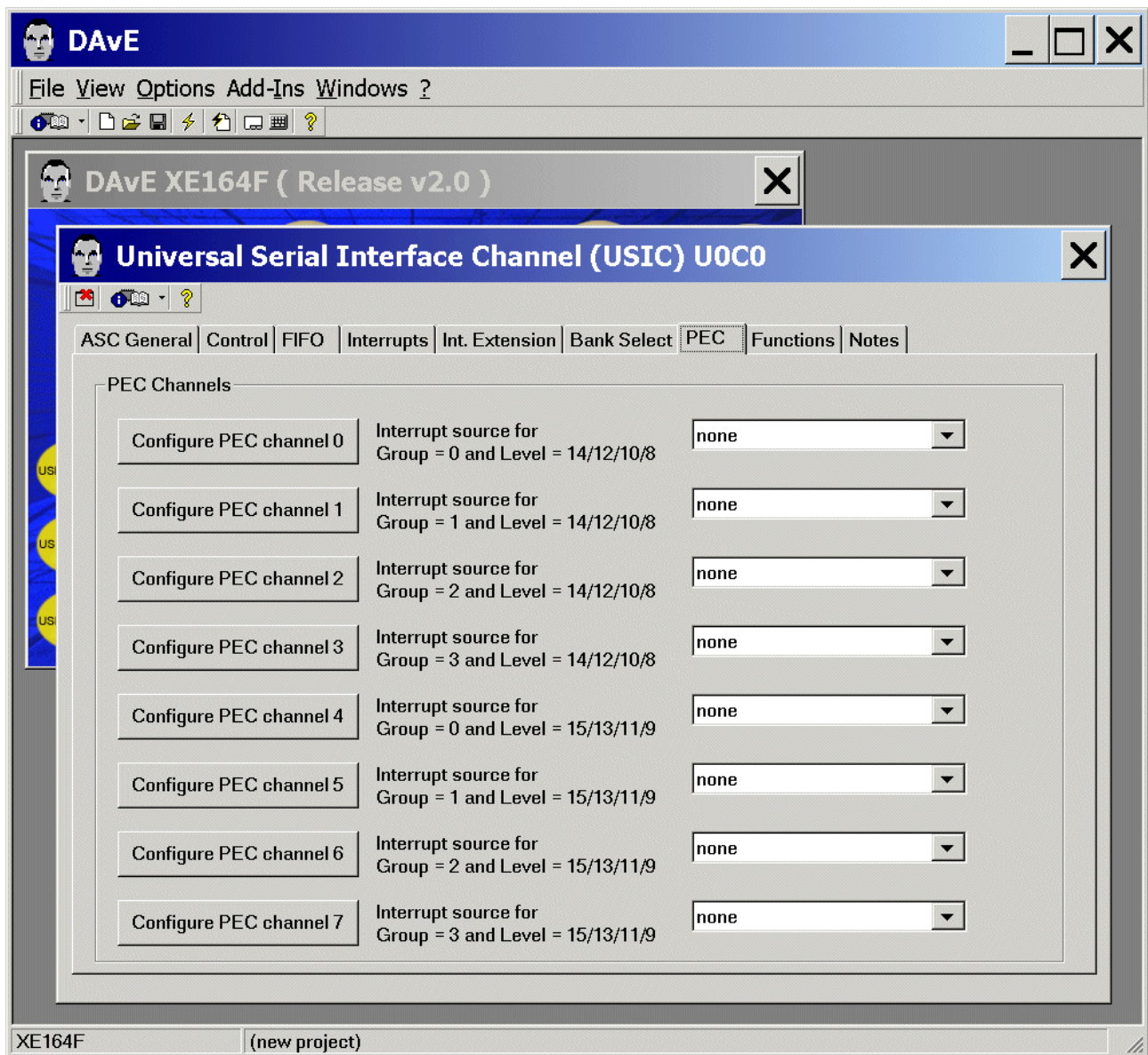


Note:

For our hello world program the 2 local register banks are not needed.

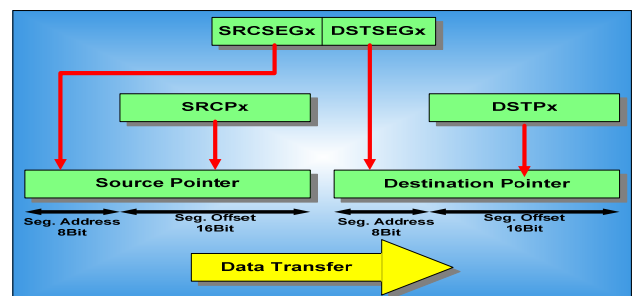


PEC: (do nothing)

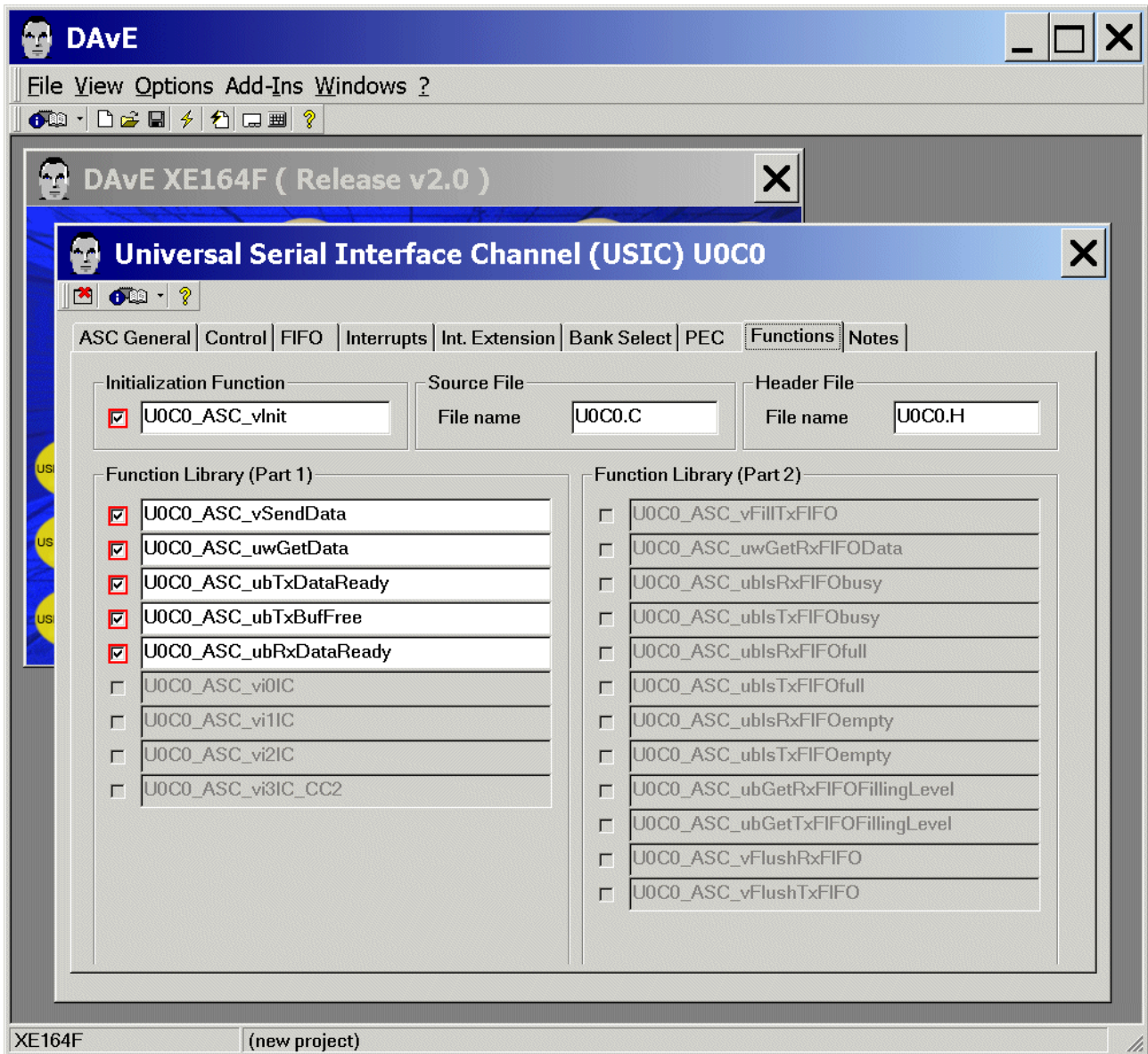


**Note:**

For our hello world program the 8 PEC Channels are not needed.



Functions: Initialization Function: **click** ☒ U0C0\_ASC\_vInit  
 Functions: Function Library (Part 1): **click** ☒ U0C0\_ASC\_vSendData  
 Functions: Function Library (Part 1): **click** ☒ U0C0\_ASC\_uwGetData  
 Functions: Function Library (Part 1): **click** ☒ U0C0\_ASC\_ubTxDataReady  
 Functions: Function Library (Part 1): **click** ☒ U0C0\_ASC\_ubTxBufFree  
 Functions: Function Library (Part 1): **click** ☒ U0C0\_ASC\_ubRxDataReady



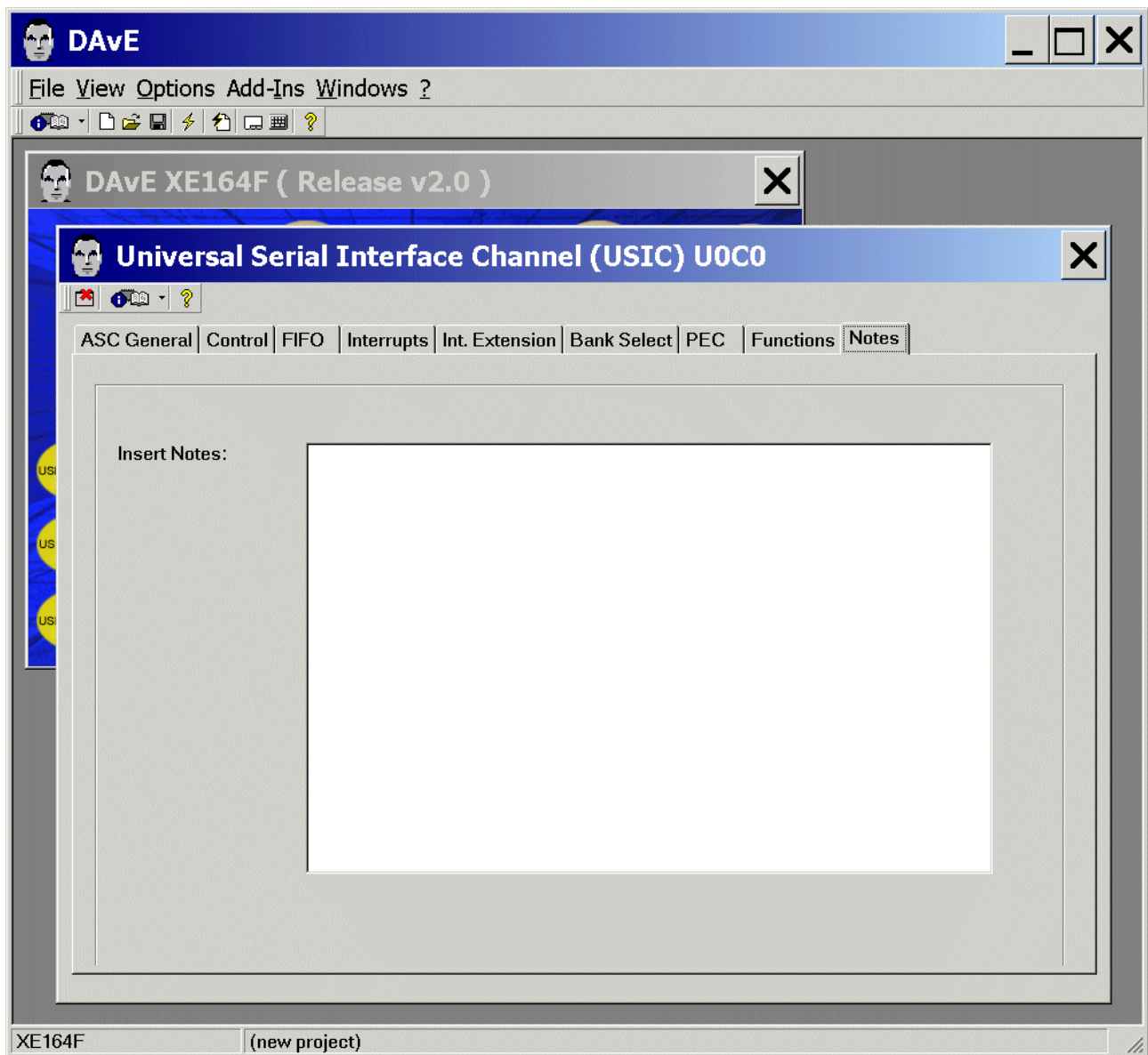
**Note:**

You can change function names (e.g. U0C0\_ASC\_vInit) and file names (e.g. U0C0.C, U0C0.H) anytime.






Notes: (do nothing)



Note:

Notes: If you wish, you can insert your comments here.

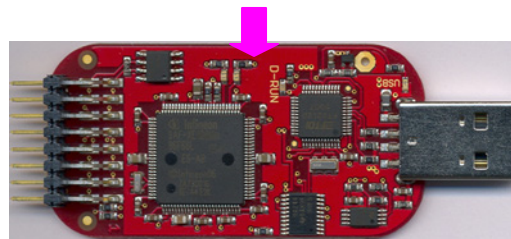
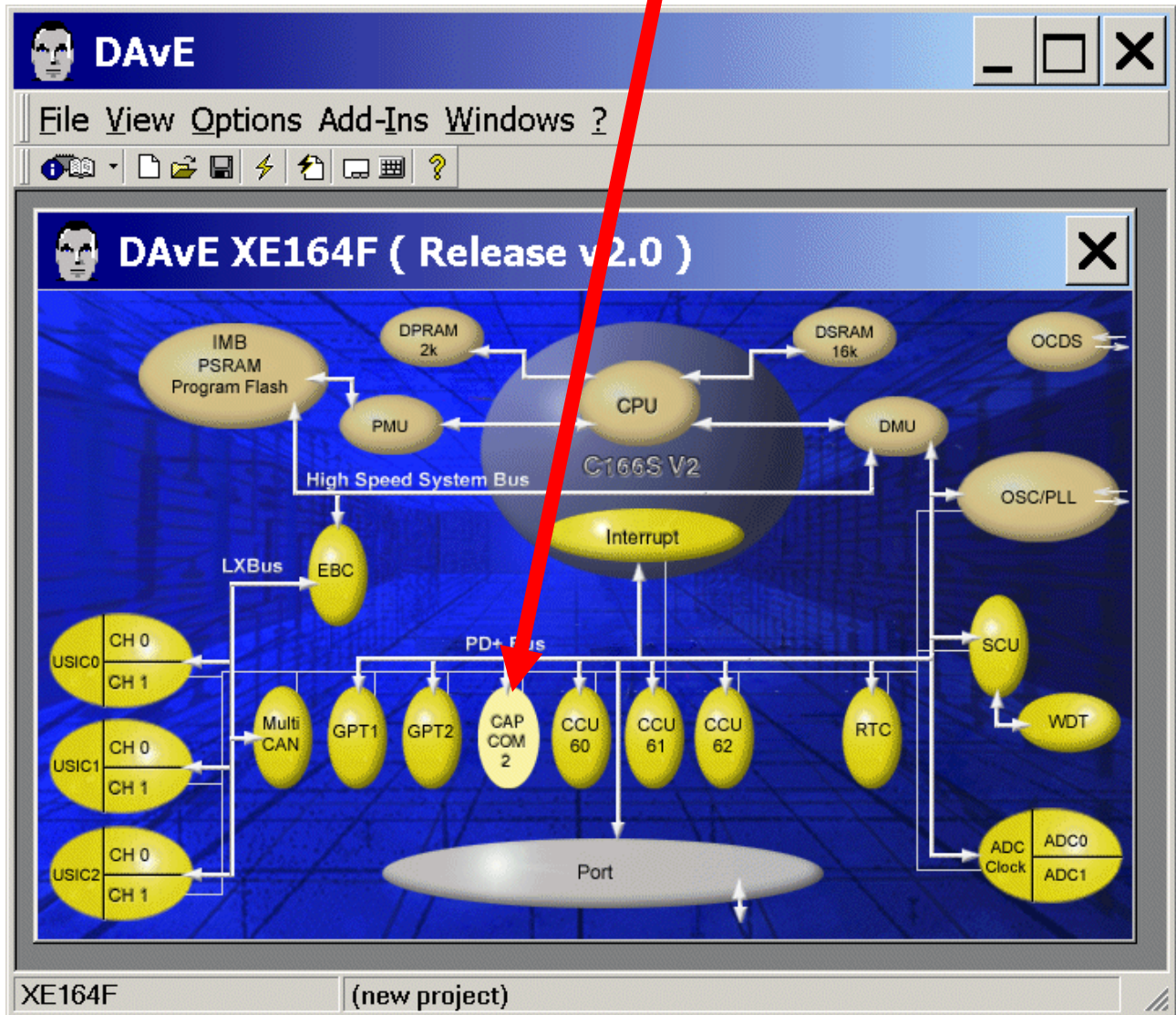


Exit and Save this dialog now by clicking  the close button.



Configure Timer T7 in the CAPCOM 2 module:

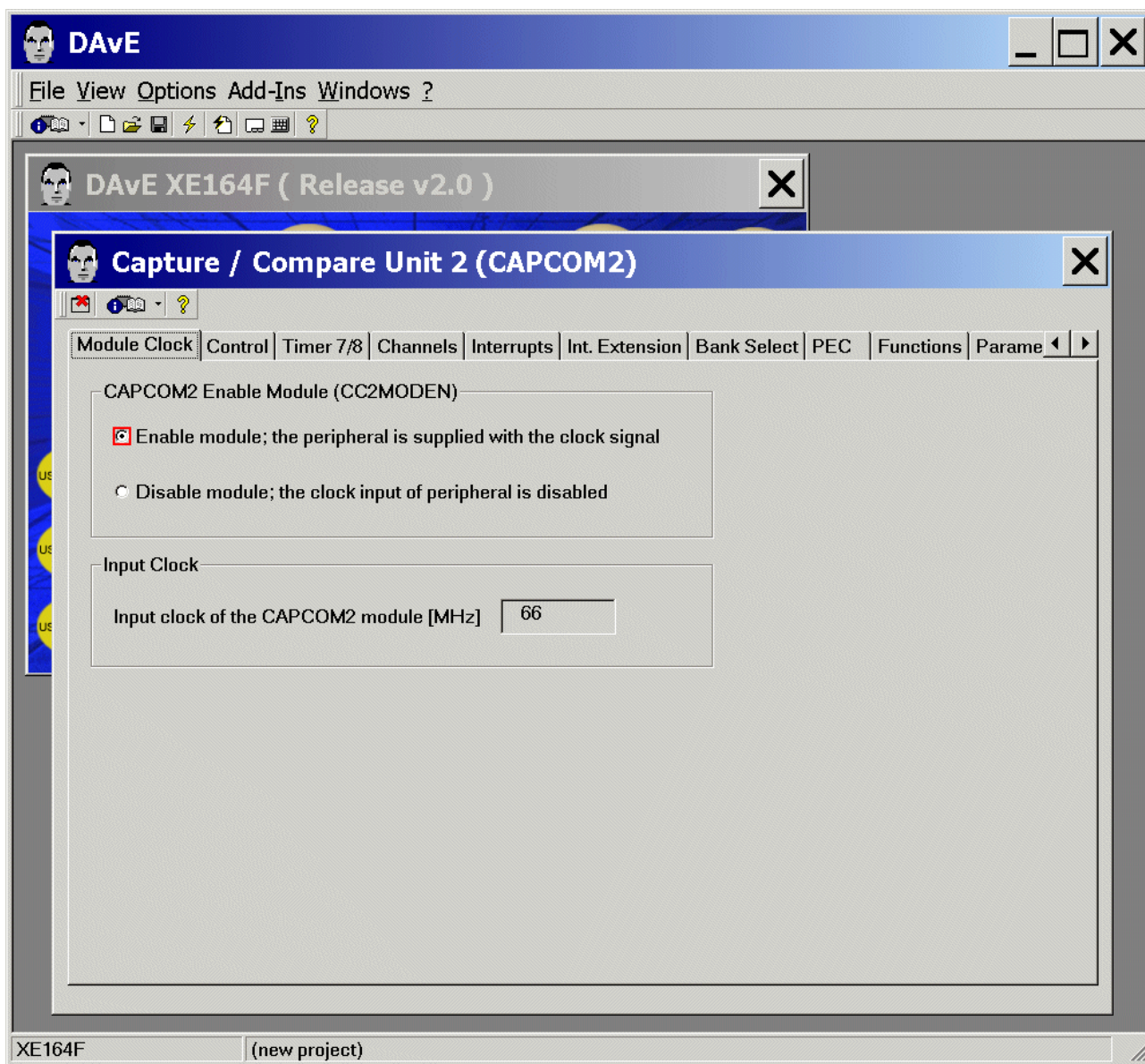
The configuration window/dialog can be opened by clicking the specific block/module (CAPCOM2).



**Note:**

The **LED on IO\_Port\_2.7** will be blinking (if selected in the main menu) with a frequency of about 1 second (done in the Timer\_7-Interrupt-Service-Routine). Therefore we have to configure Timer\_7.

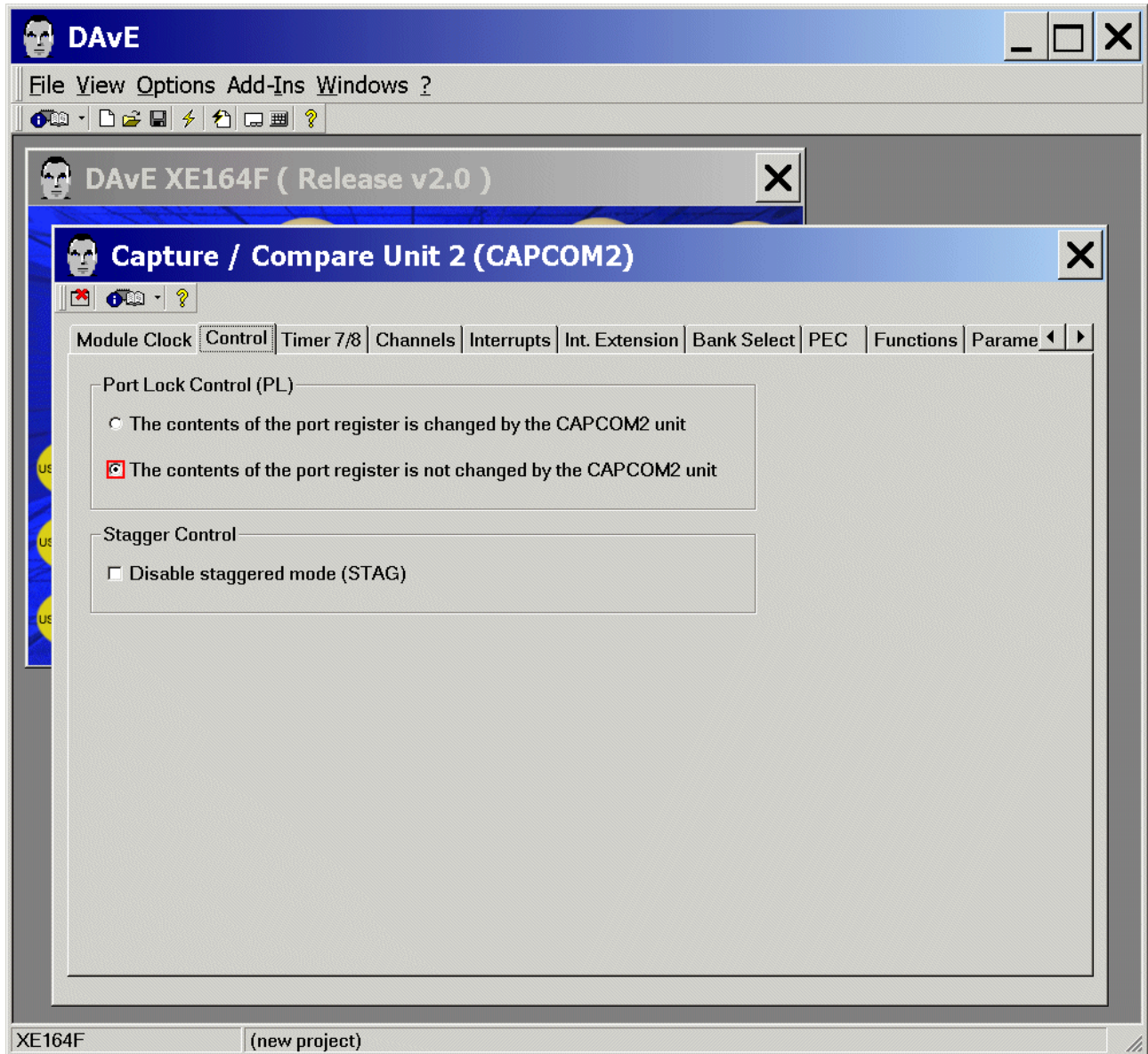
Module Clock: CAPCOM2 Enable Module: **click** ☒ Enable module



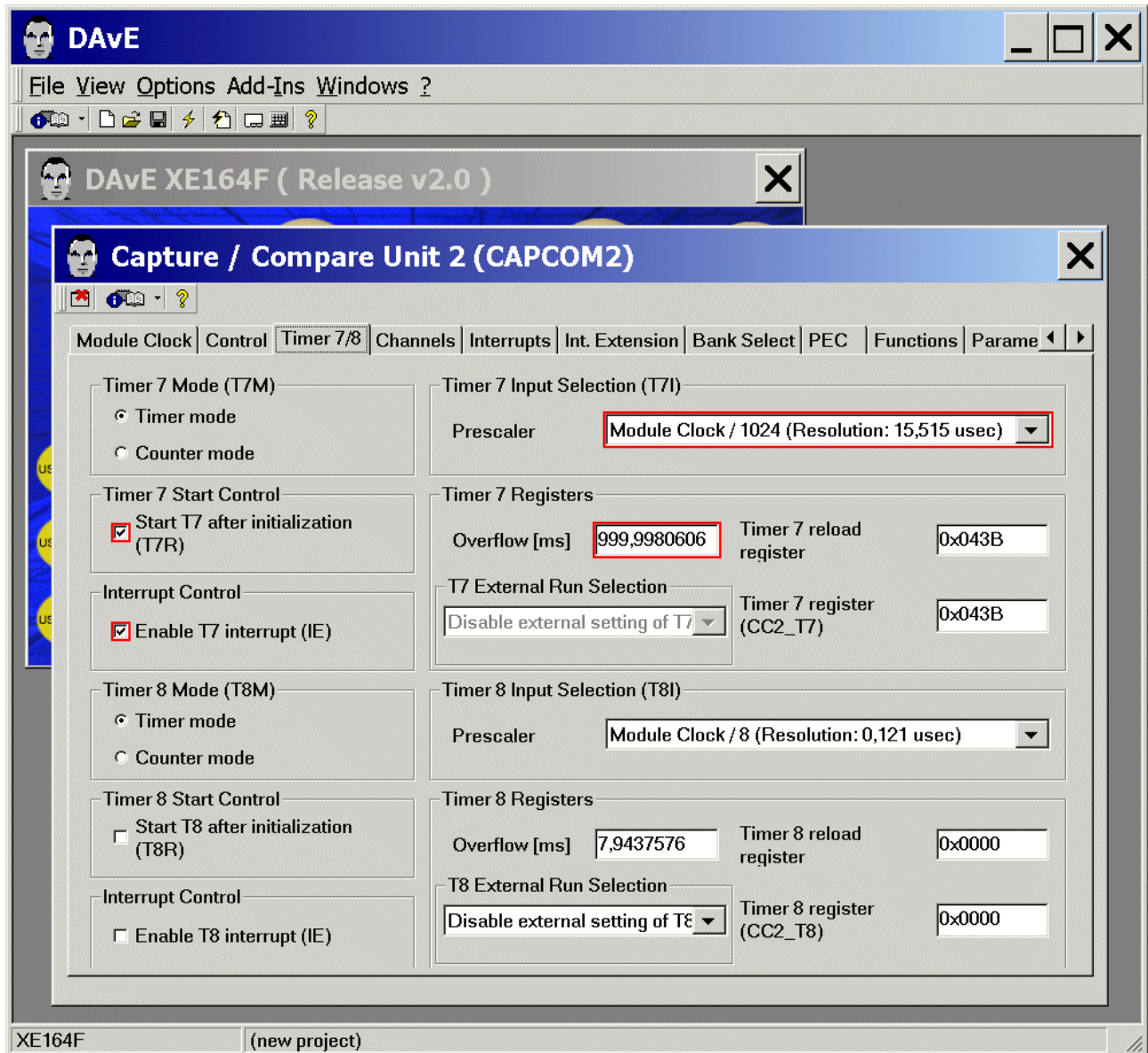


**Control: Port Lock Control:**

click ☒ The contents of the port register is not changed by the CAPCOM2 unit

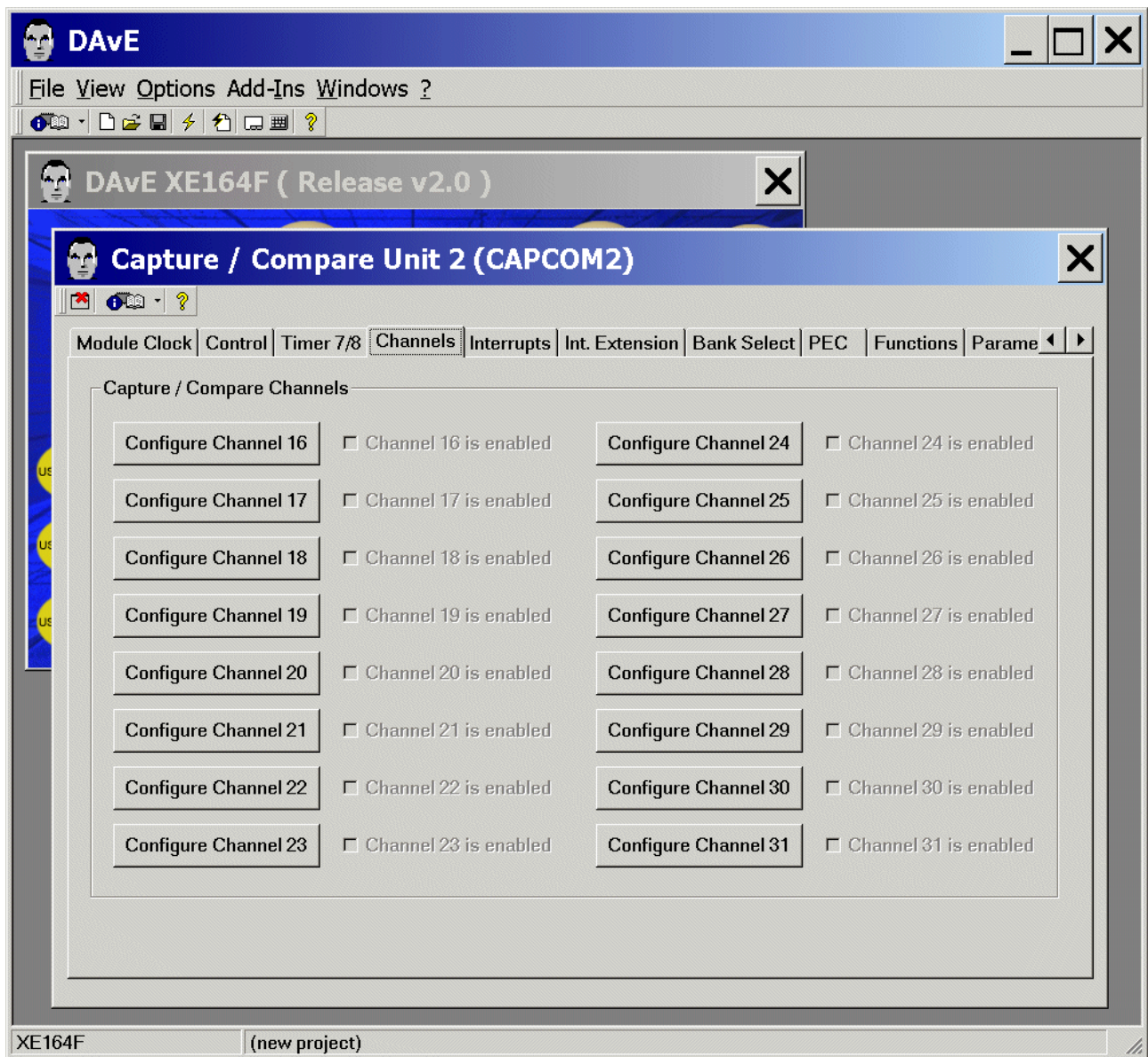


Timer 7/8: Timer 7 Start Control: **click** ✓ Start T7 after initialization (T7R)  
 Timer 7/8: Interrupt Control: **click** ✓ Enable T7 interrupt (IE)  
 Timer 7/8: Timer 7 Input Selection (T7I): **Prescaler**: **choose** Module Clock/1024  
 Timer 7/8: Timer 7 Registers: **Overflow** [s]: **insert** 1 <ENTER>

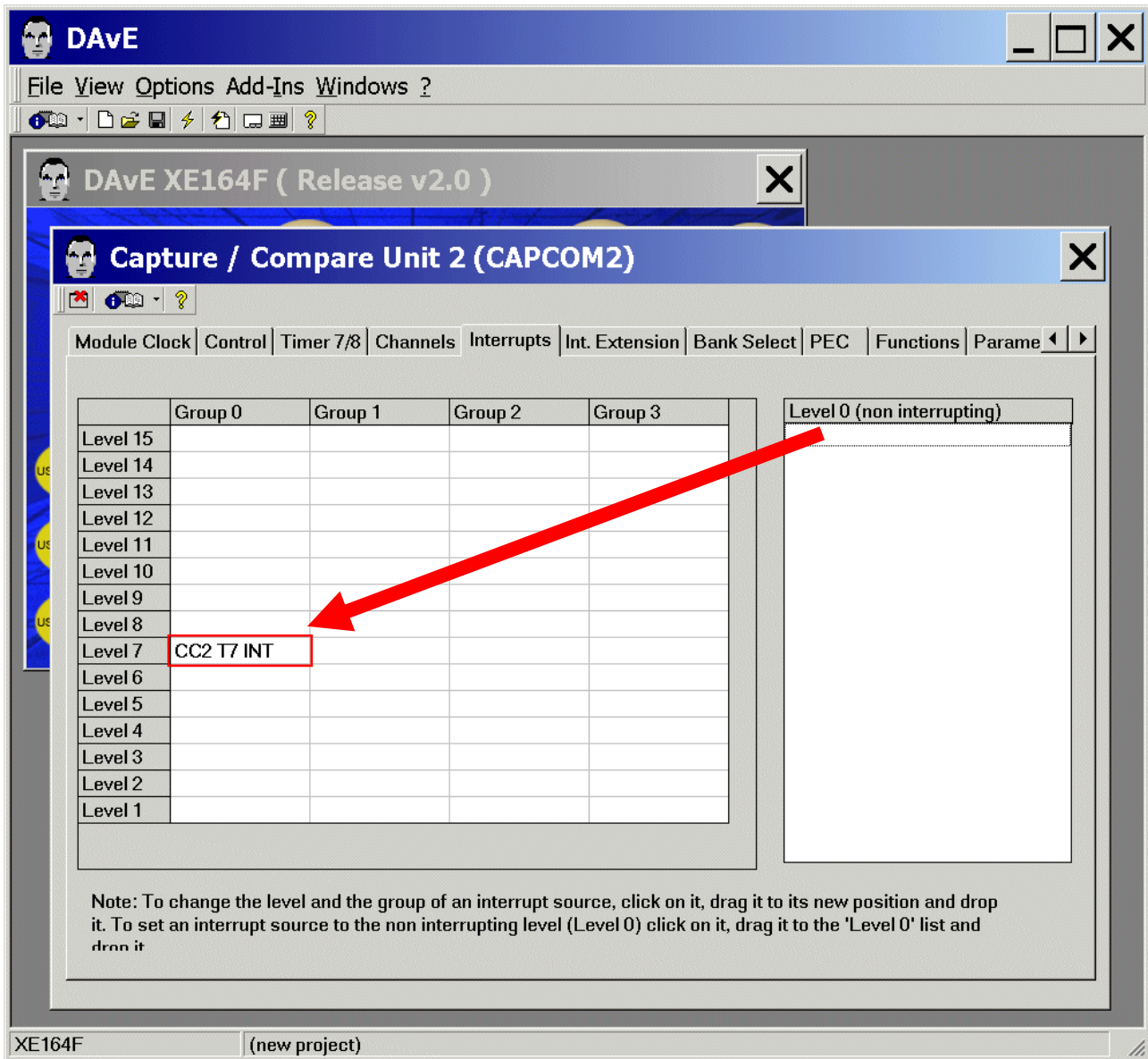




Channels: (do nothing)



Interrupts: drag and drop the CC2 T7 INT to Interrupt Level 7, Group 0

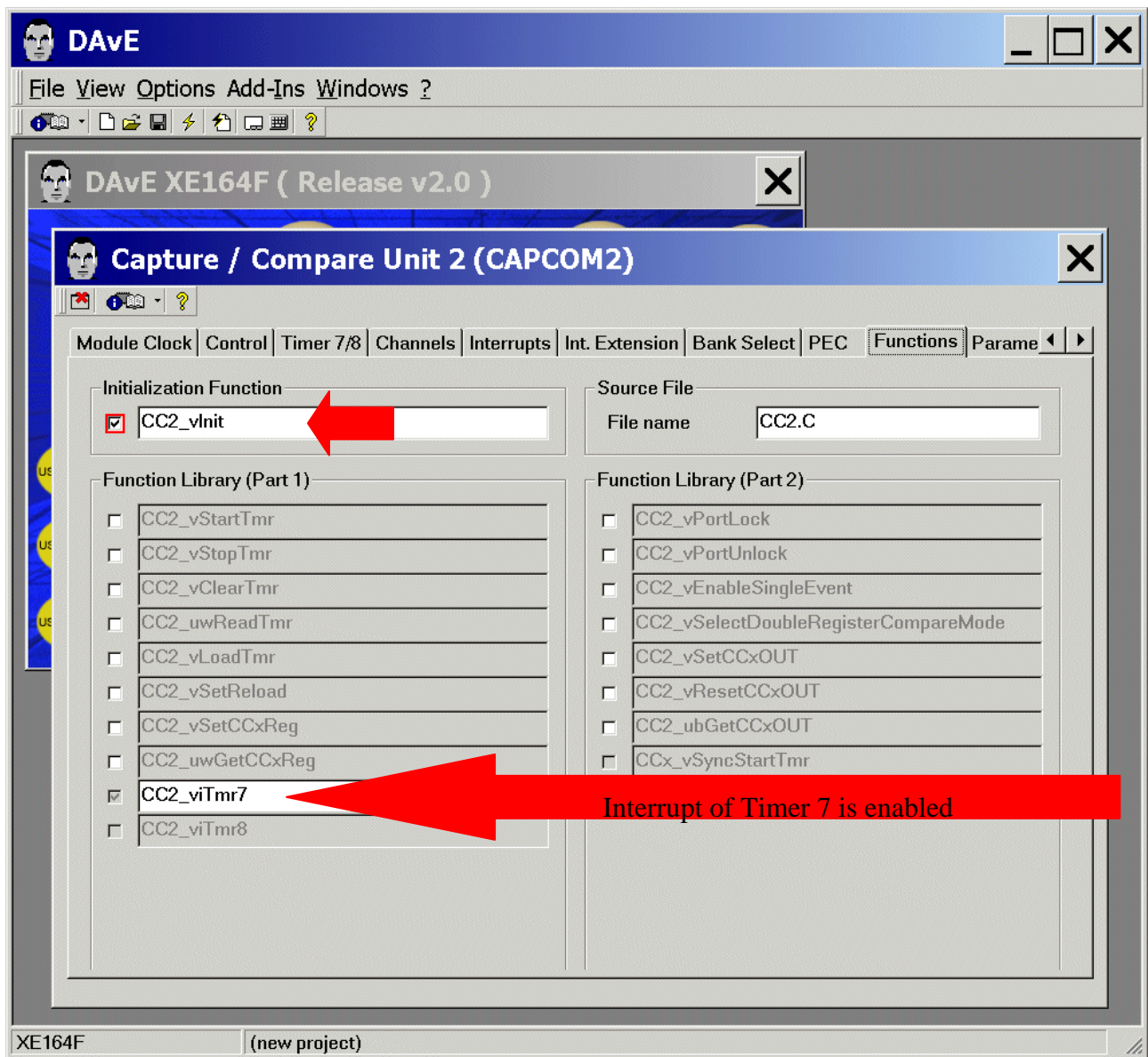


Int. Extension: (do nothing)

Bank Select: (do nothing)


PEC: (do nothing)

Functions: Initialization Function: click/check ☒ CC2\_vInit



Parameters: (do nothing)

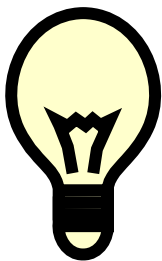
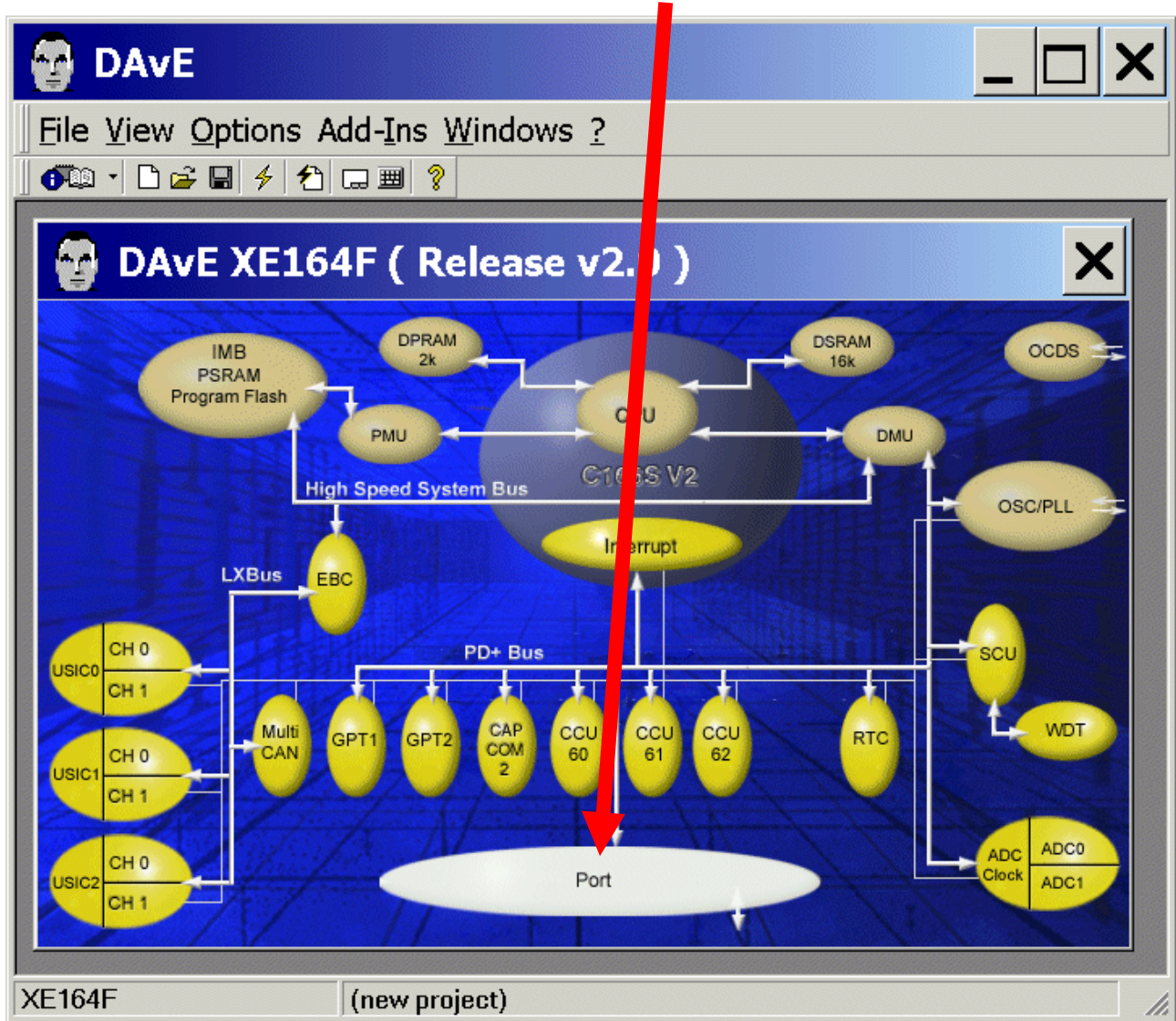
Notes: If you wish, you can insert your comments here.

Exit and Save this dialog now by clicking  the close button.

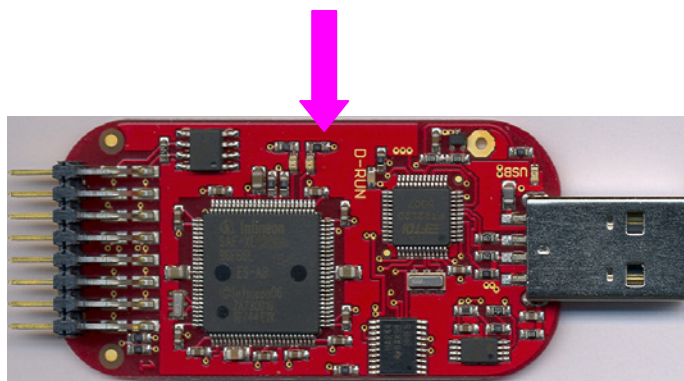


Configure Port 2 Pin 7 to Output :

The configuration window/dialog can be opened by clicking the specific block/module (Port).

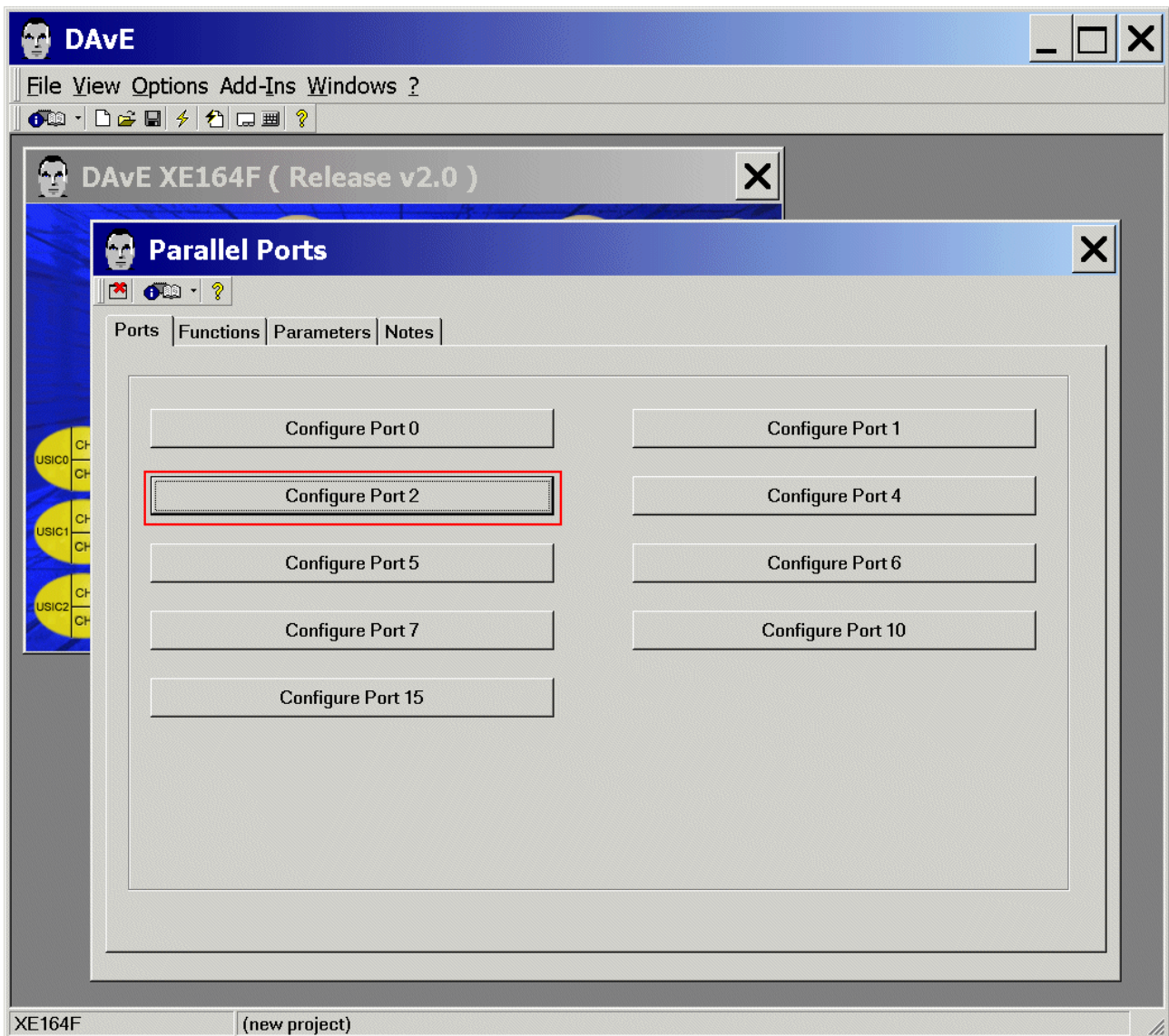


**Note:**  
The LED is connected to IO\_Port\_2.7

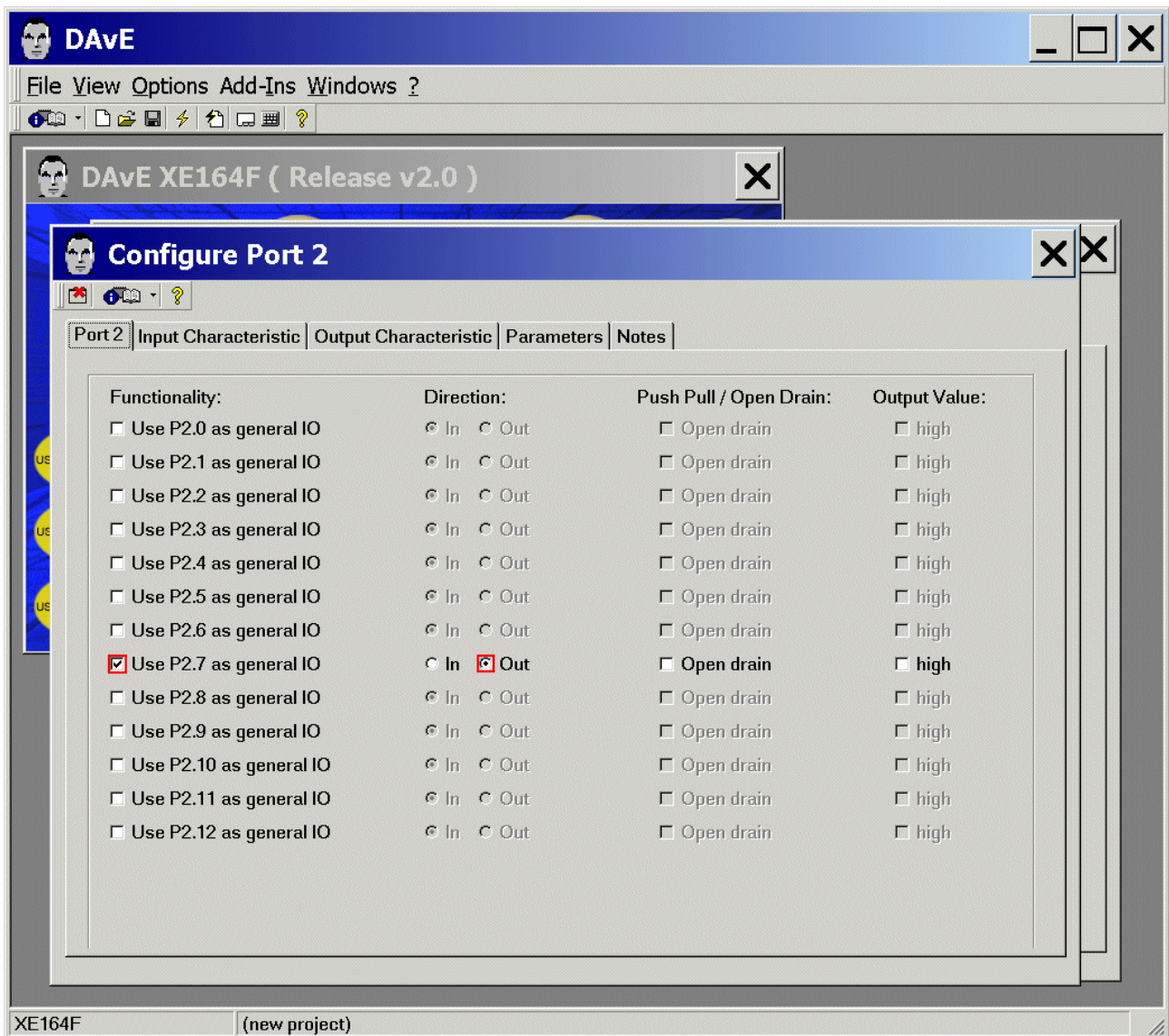




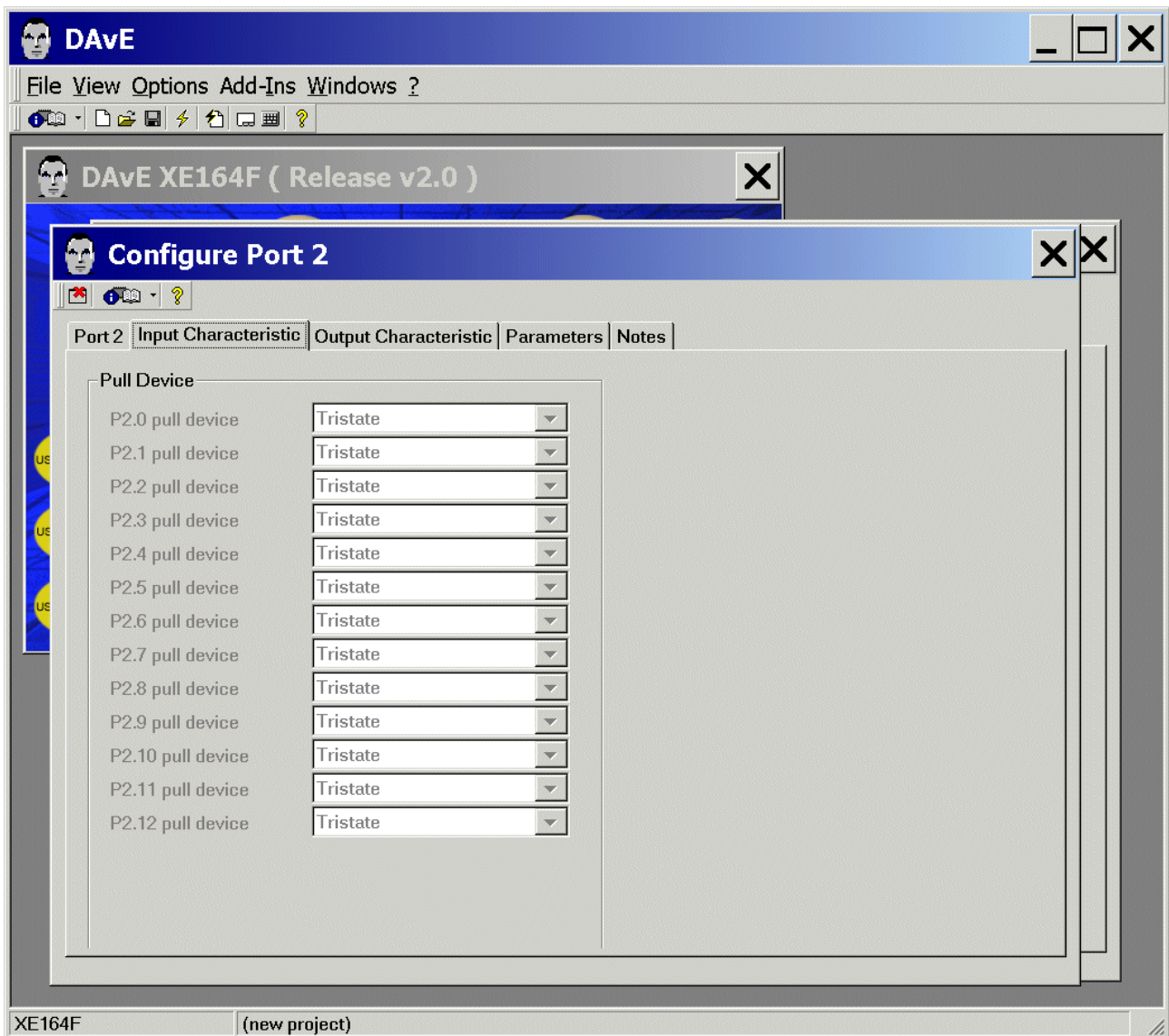
Ports: click "Configure Port 2"



Port 2: Functionality: **click** ☒ Use P2.7 as general IO - Direction: **click** ☒ Out

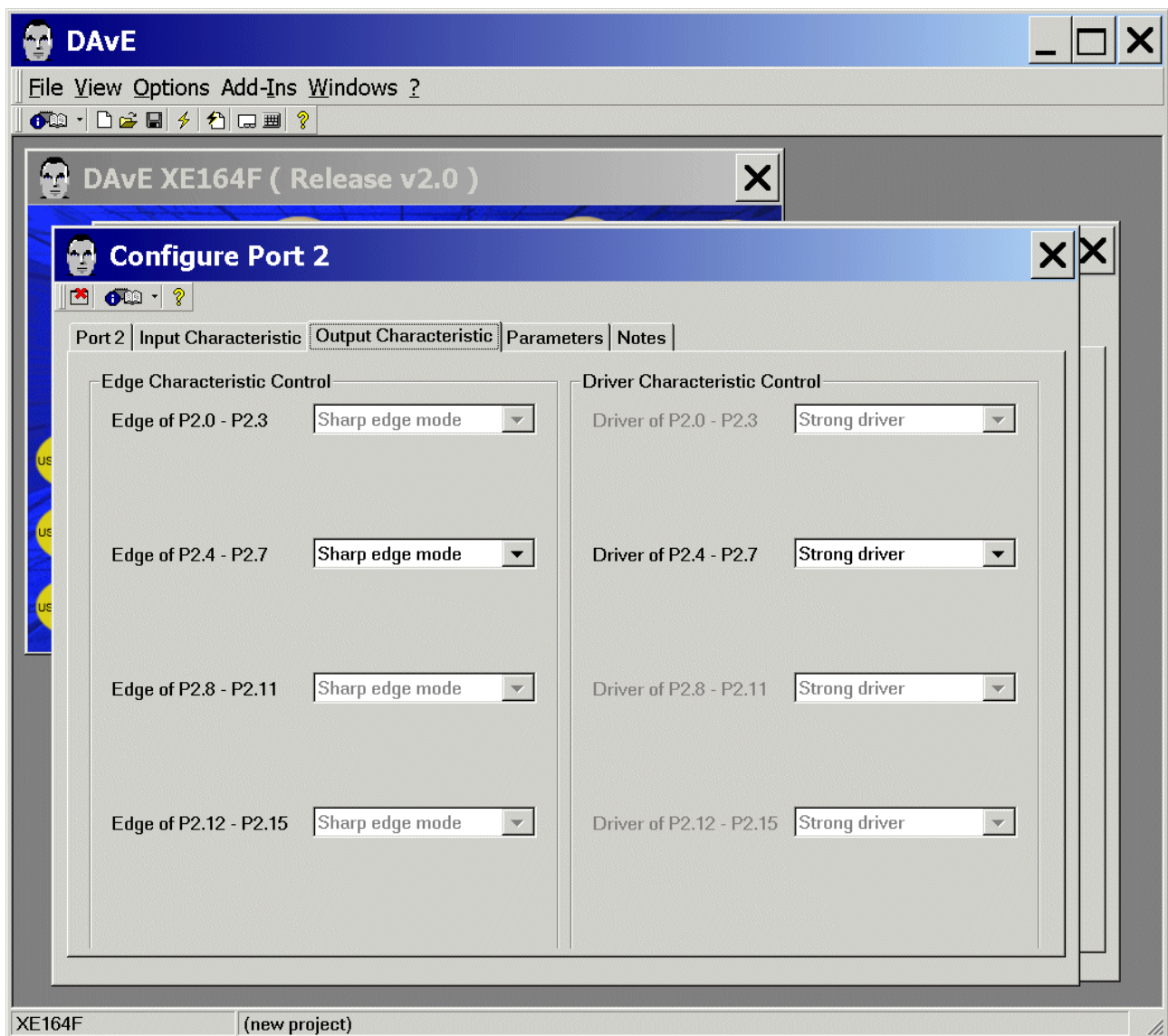


Input Characteristic: (do nothing)



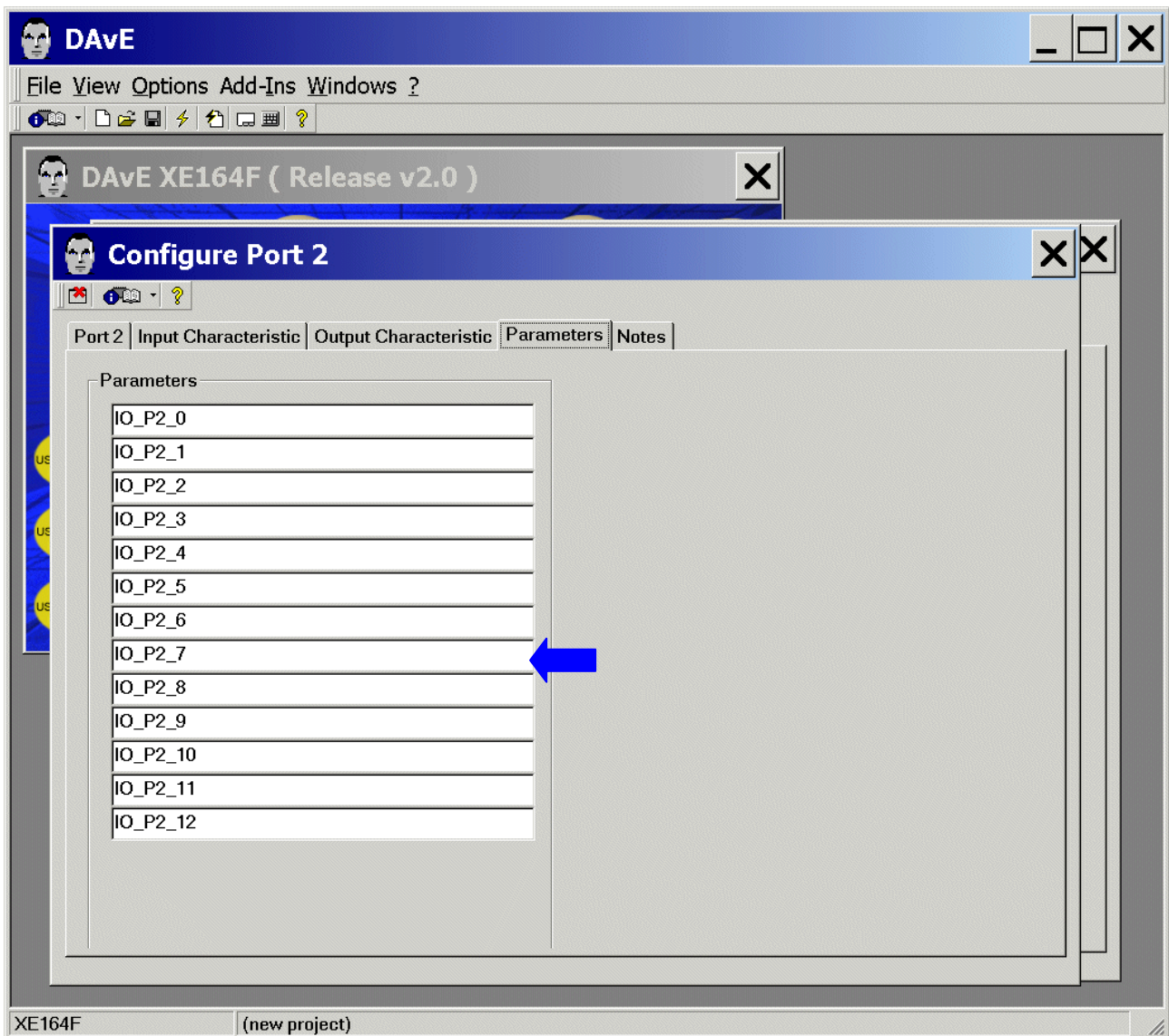


Output Characteristic: (do nothing)





Parameters: (do nothing)




**Note:**

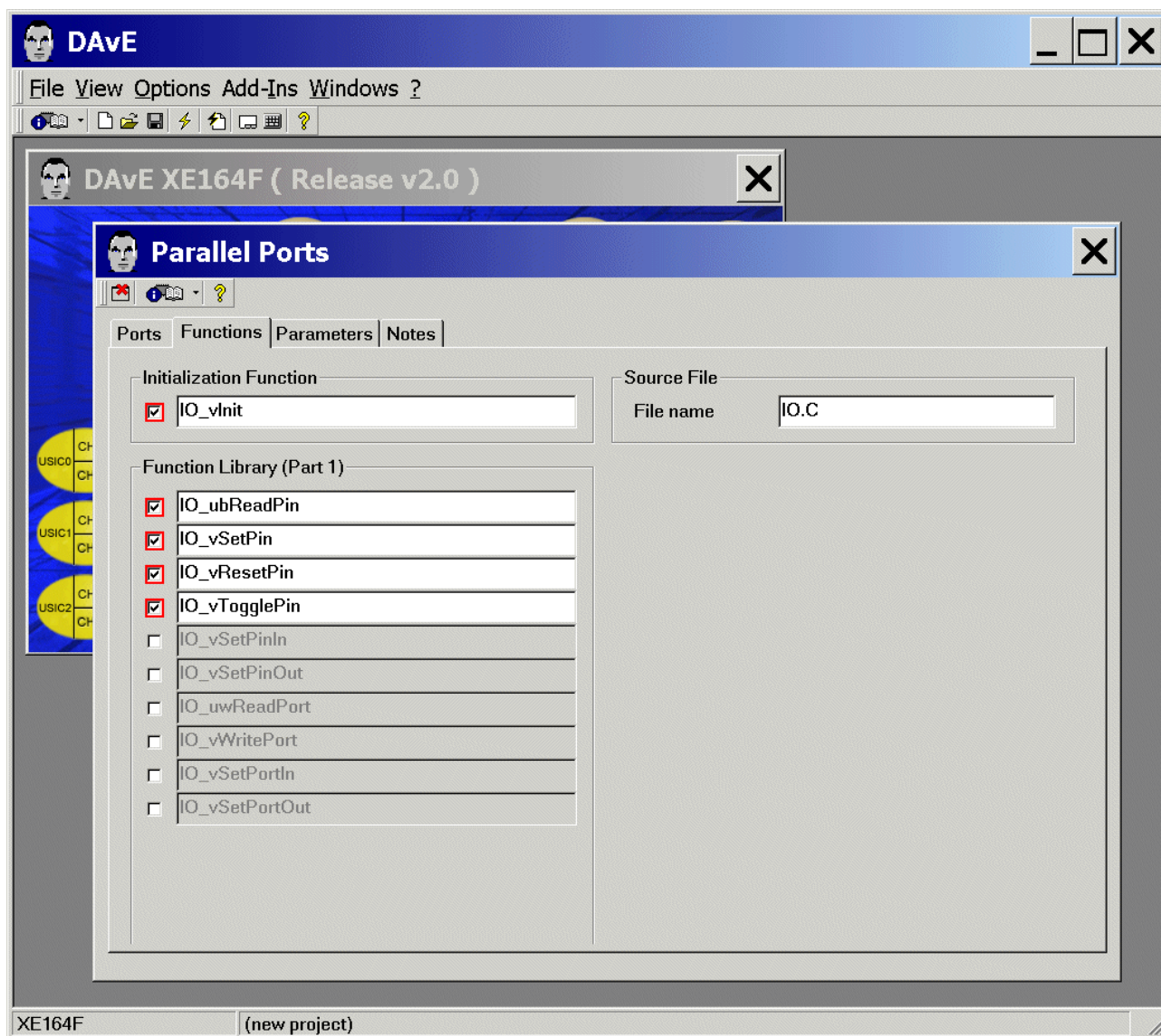
We will use the name **IO\_P2\_7** in application programming.



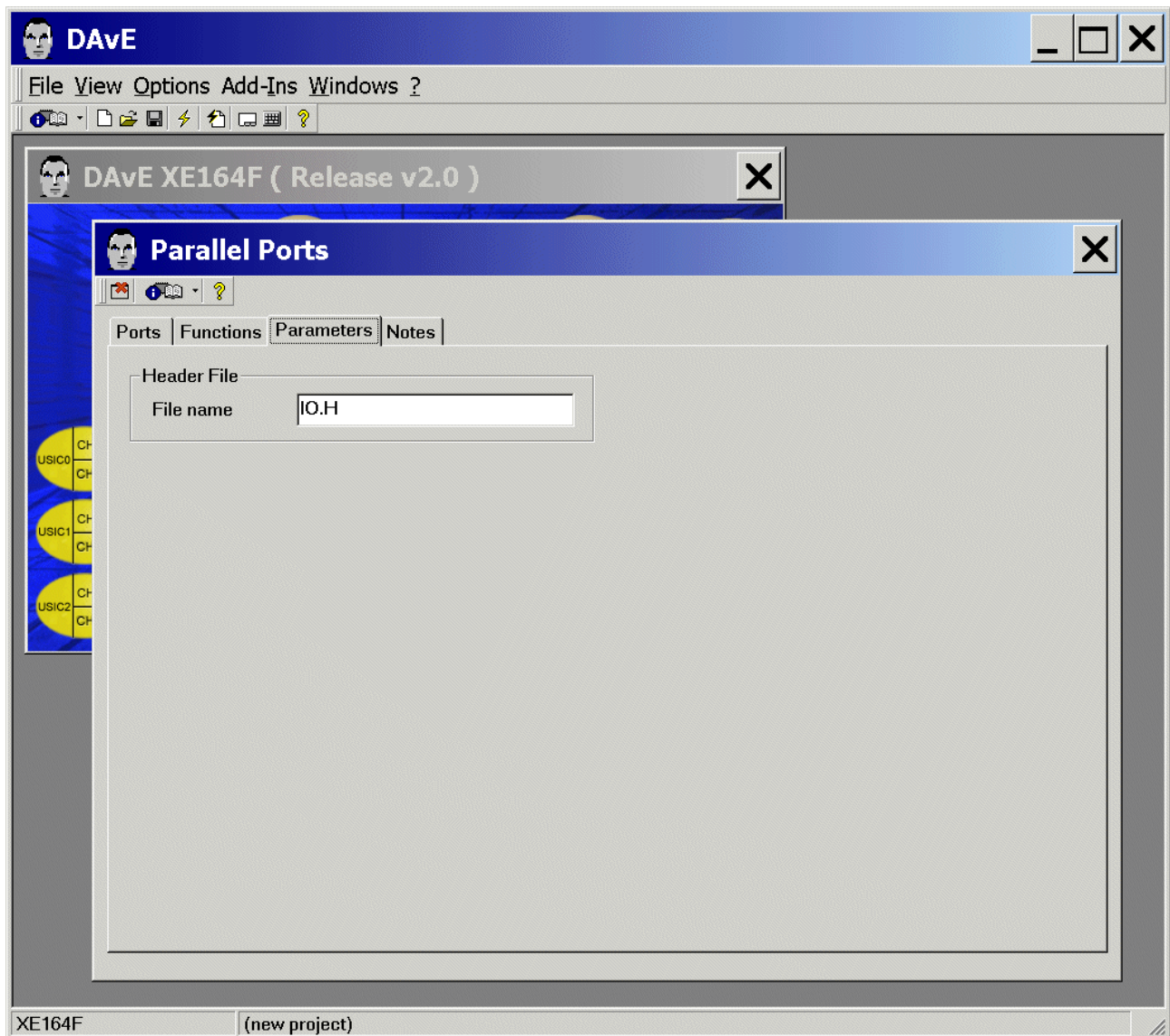
**Notes:** If you wish, you can insert your comments here.

**Exit** and **Save** this dialog now by clicking  the close button:


Functions: Initialization Functions: **click/check** ☒ IO\_vInit  
 Functions: Function Library (Part 1): **click** ☒ IO\_ubReadPin  
 Functions: Function Library (Part 1): **click** ☒ IO\_vSetPin  
 Functions: Function Library (Part 1): **click** ☒ IO\_vResetPin  
 Functions: Function Library (Part 1): **click** ☒ IO\_vTogglePin



Parameters: (do nothing)



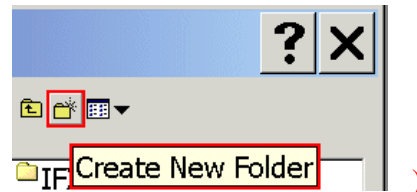
Notes: If you wish, you can insert your comments here.

Exit and Save this dialog now by clicking  the close button.

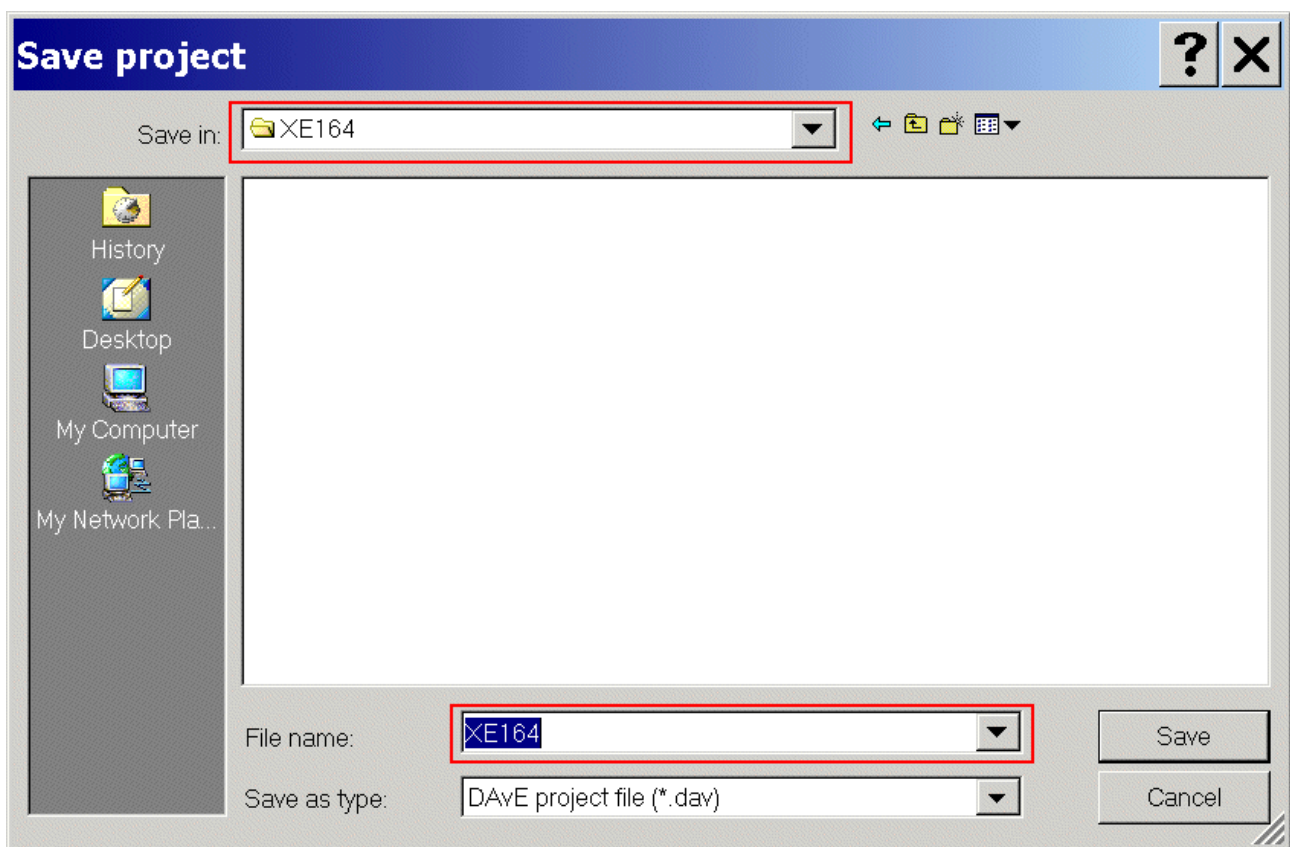


Save the project:

File  
Save




Save project: Save in C:\XE164 (create new directory  
File name: XE164

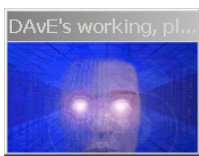


Save

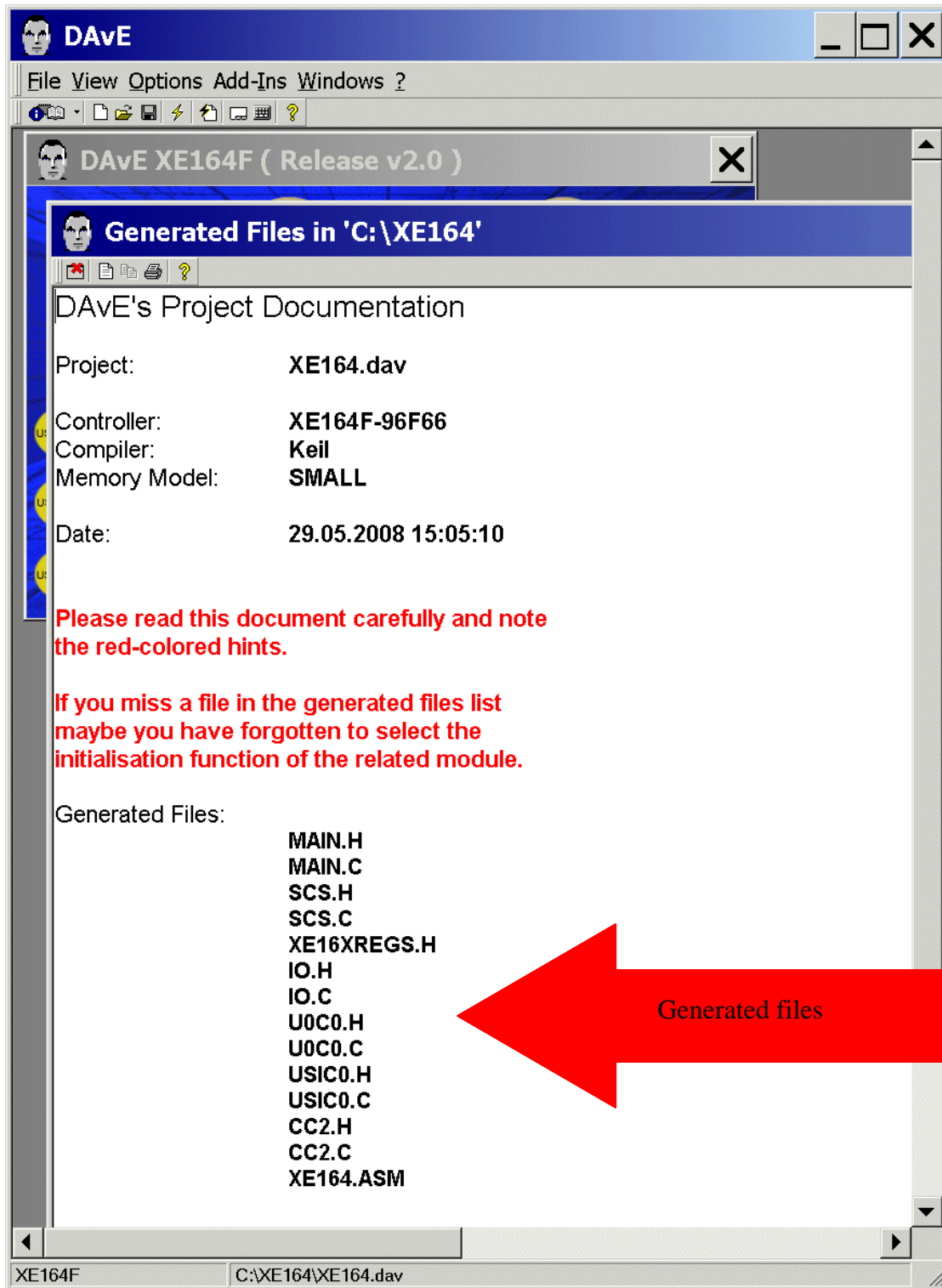


Generate Code:

<p>File Generate Code</p>	<p>or      click </p>
-------------------------------	---



DAvE will show you all the files he has generated  
(File Viewer opens automatically):



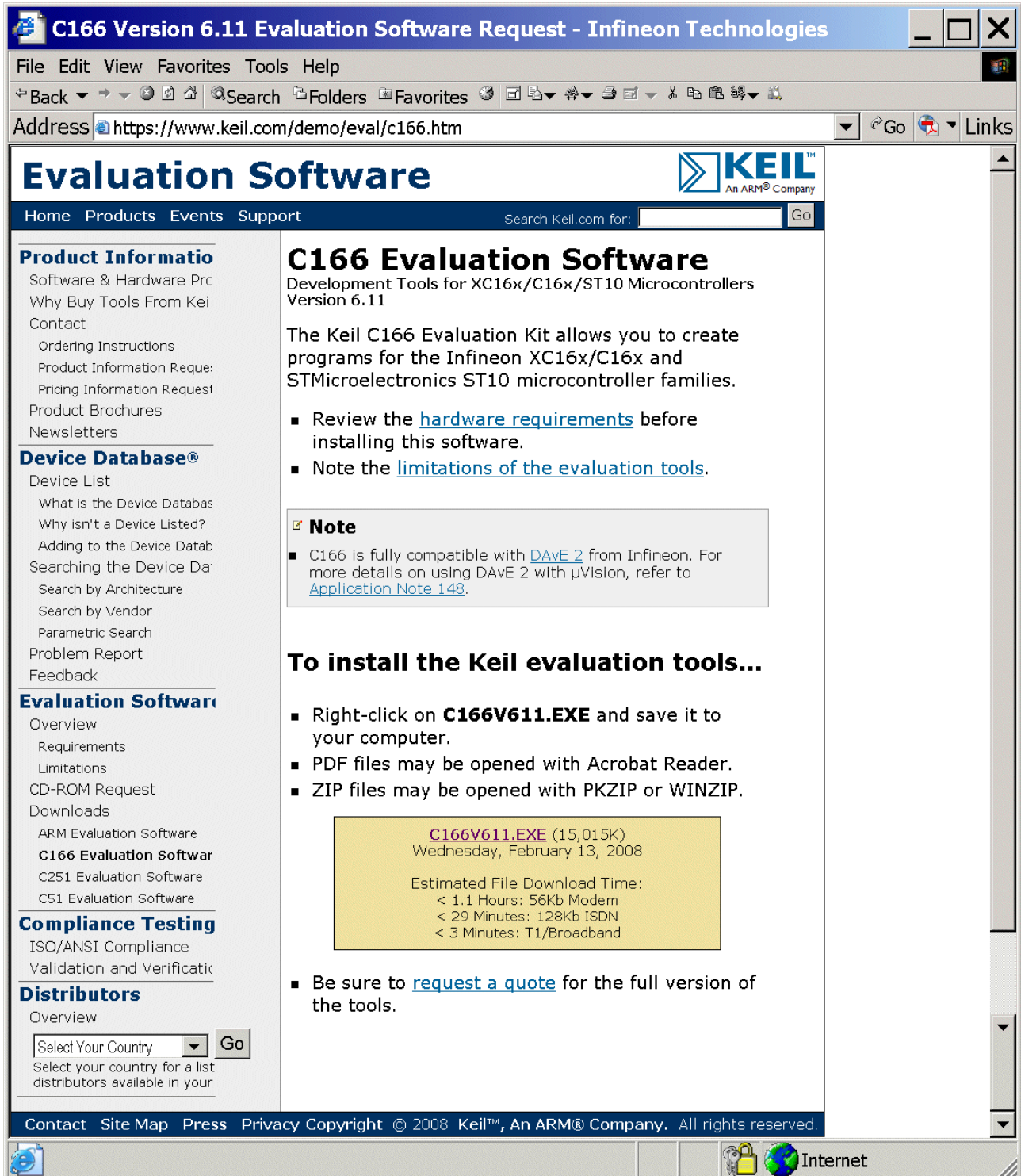
File - Exit

Save changes?

Click: **Yes**

#### 4.) Using the KEIL - $\mu$ Vision 3 Development Tools:

Install the tool chain: You can download the Keil Development Tools @ <http://www.keil.com> :



The screenshot shows a web browser window titled "C166 Version 6.11 Evaluation Software Request - Infineon Technologies". The address bar shows the URL <https://www.keil.com/demo/eval/c166.htm>. The page content includes a navigation menu with links like Home, Products, Events, and Support. The main content area is titled "Evaluation Software" and "C166 Evaluation Software". It provides information about the development tools for XC16x/C16x/ST10 microcontrollers. A list of instructions for installing the tools is provided, including right-clicking on C166V611.EXE and saving it to the computer. A note mentions compatibility with DAVE 2. A download box for C166V611.EXE (15,015K) is shown, along with estimated file download times for different connection speeds. The page also includes a section for compliance testing and distributors.

Download and Execute C166V611.EXE ( - or any higher version ) and install the Keil tool chain.



Start Keil  $\mu$  Vision3 and open the DAVe Project:

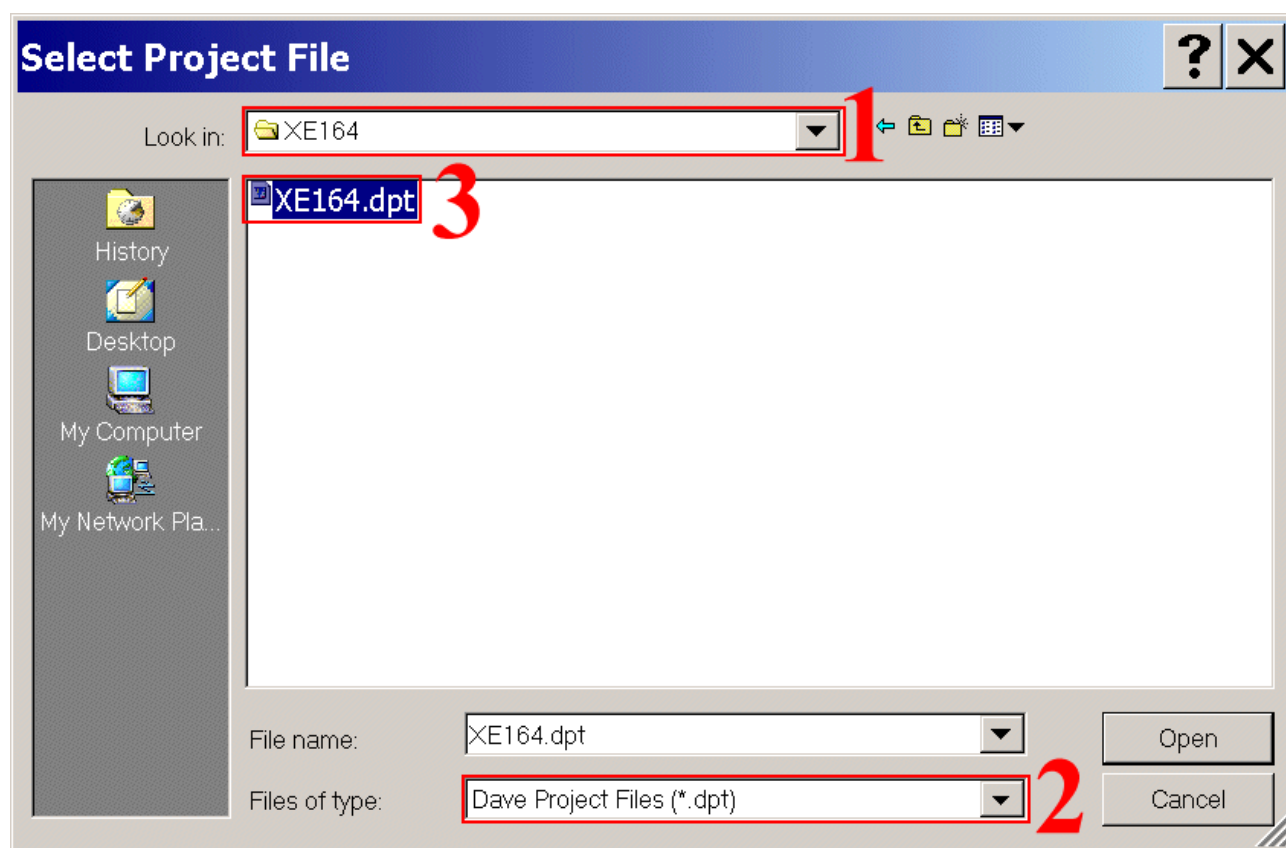
If you see an open project – close it: **Project - Close Project**

**Project - Open Project**

Select Project File: **Look in:** choose C:\XE167 (1)

Select Project File: **Files of type:** select Dave Project Files (2)

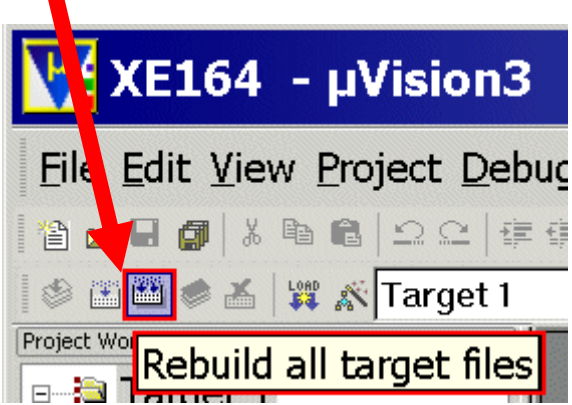
Click XE167.dpt (3)

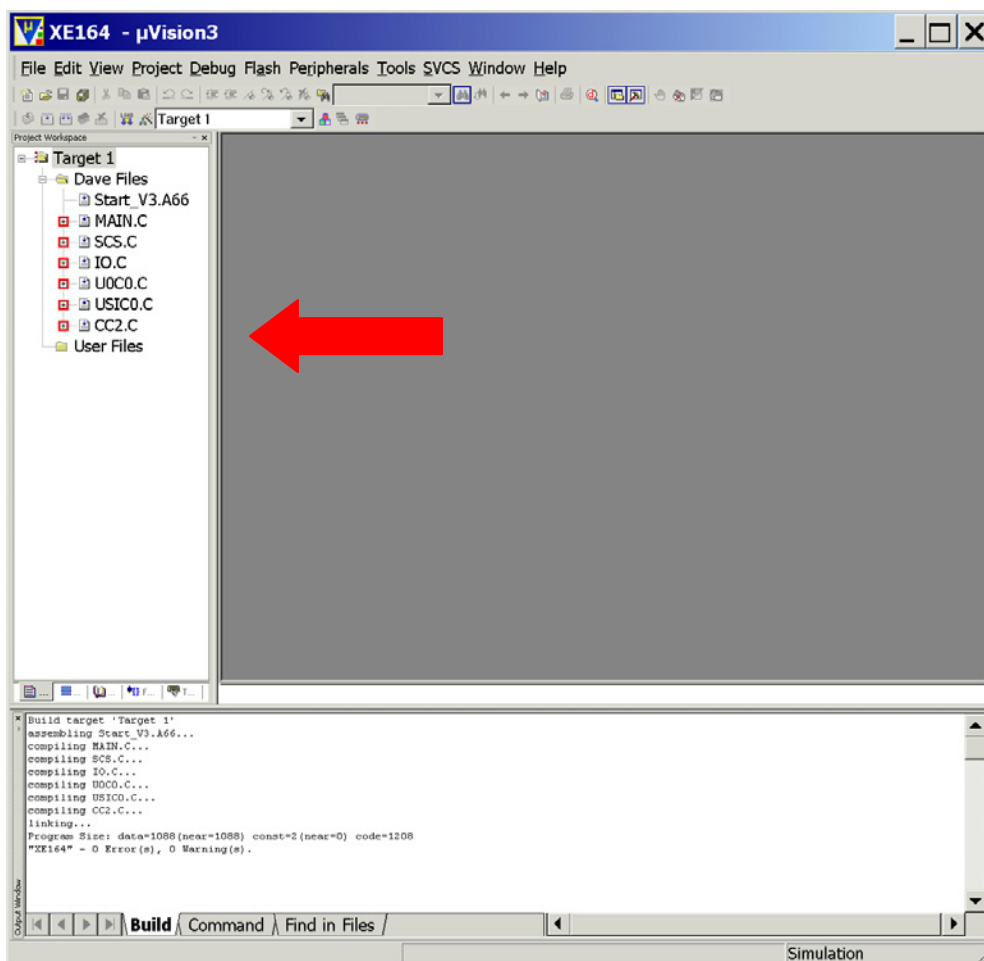


Click Open

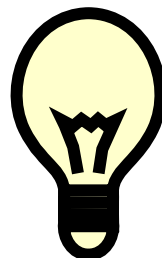


Generate „make“- file:

Project – Rebuild all target files	or	<p>click</p> 
------------------------------------	----	---

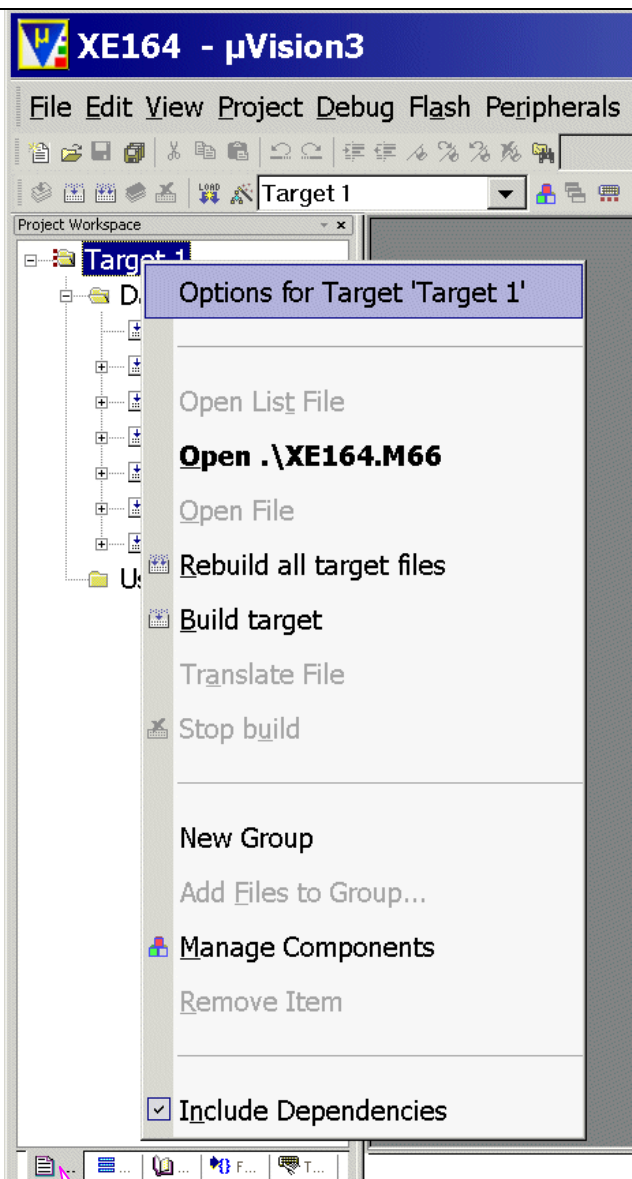
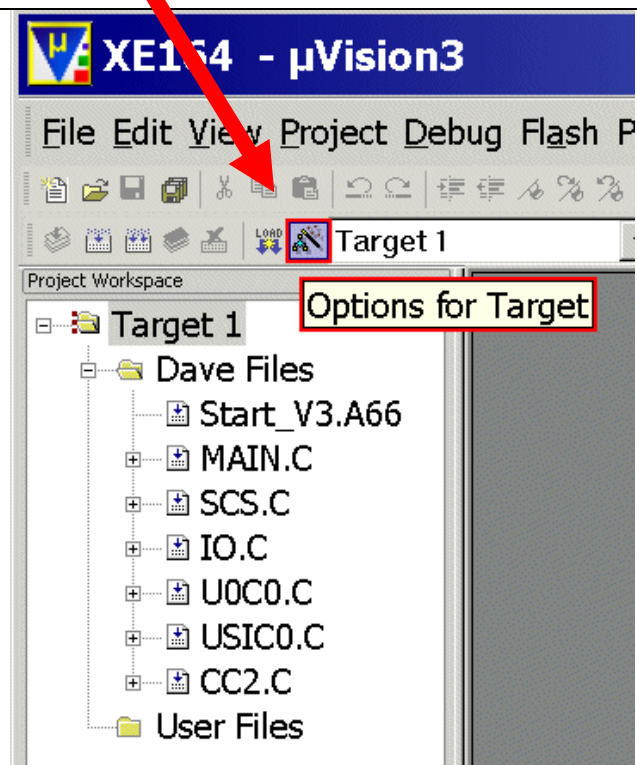


**Note:**  
This step generates a makefile and shows the include files.



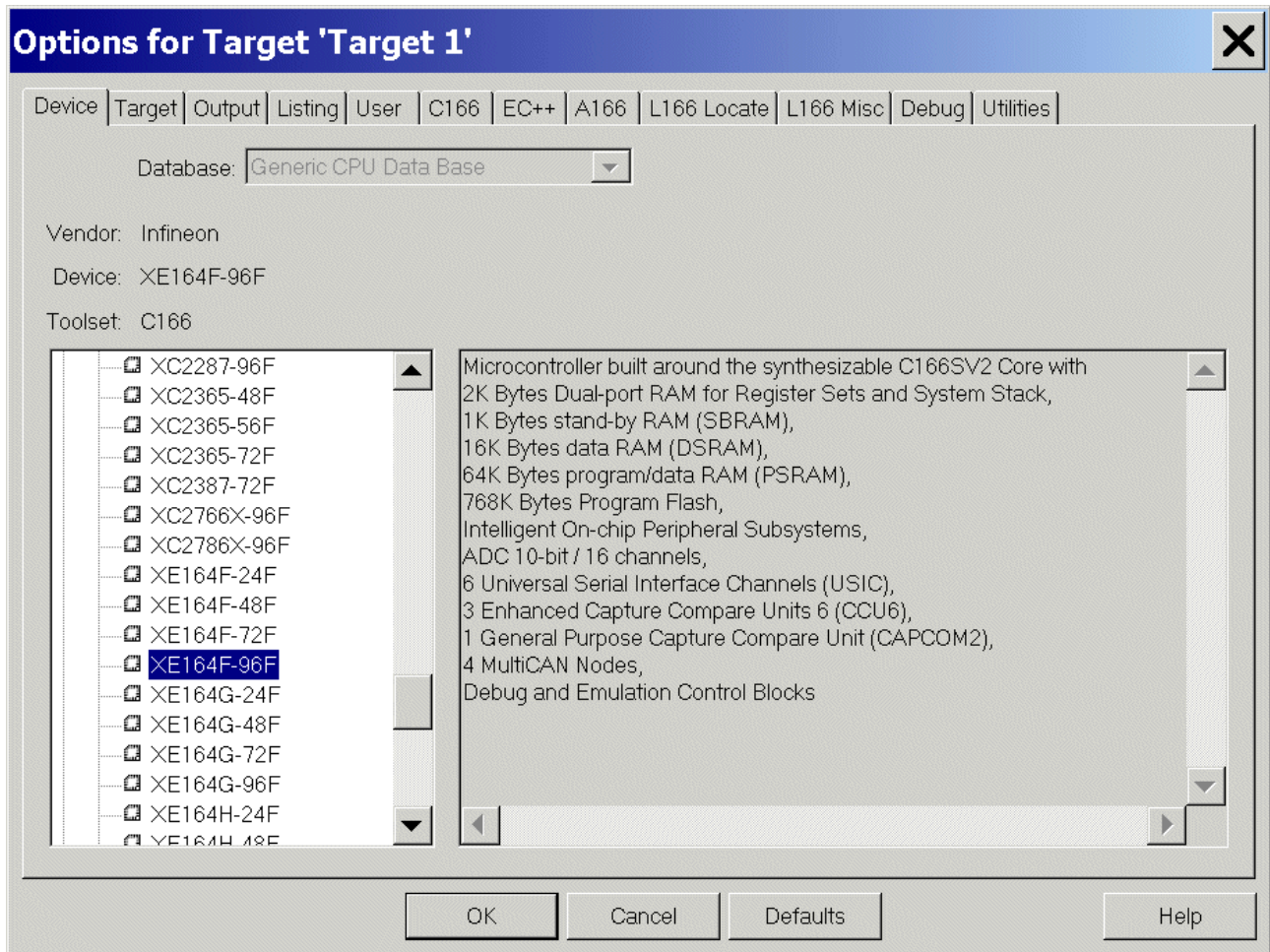
Configure:

Compiler, Assembler, Linker, Locator, Hex-Converter, Build – Control, Simulator, Debugger, Listings and Utilities (e.g. OnChip Flash Programming):

<p>mouse position: (Project Workspace, Files): Target1 click right mouse button click Options for Target 'Target1'</p>	<p>or click</p>
	

Project Workspace, Files

Device: **check** XE164F-96F



Target: Clock(MHz): check 8.0

Target: tick/check ☒ Use On-chip ROM

Target: tick/check ☒ Use On-chip ROM

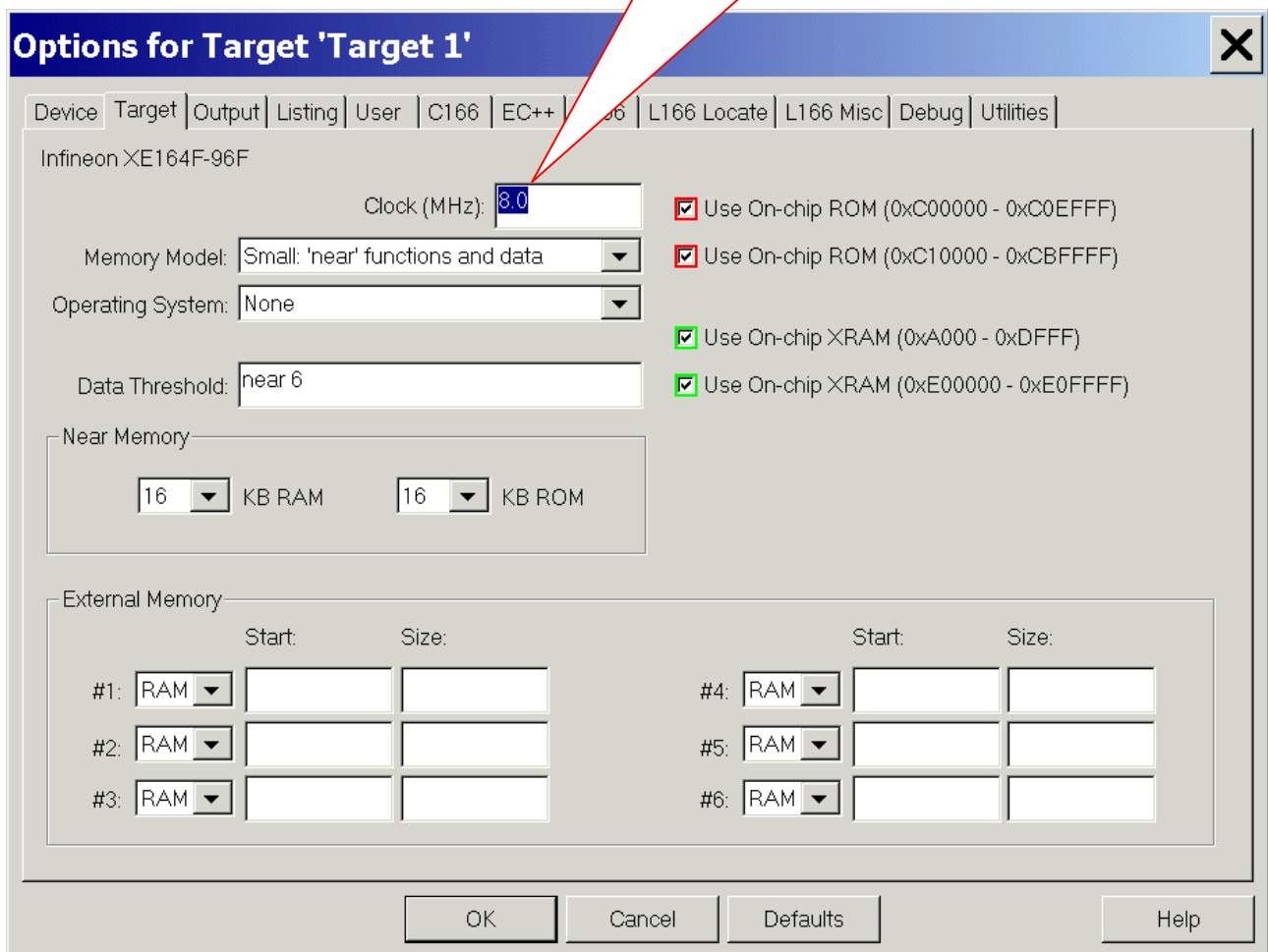
Target: tick/check ☒ Use On-chip XRAM

Target: tick/check ☒ Use On-chip XRAM

**Note (Source: DAVe):**

Configuration of the System Clock:

- VCO clock used, input clock is connected
- input frequency is 8,00 MHz
- configured system frequency is 66,00 MHz
- system clock is 66.00 MHz



The dialog box 'Options for Target 'Target 1'' contains the following settings:

- Device: Infineon XE164F-96F
- Clock (MHz): 8.0
- Memory Model: Small: 'near' functions and data
- Operating System: None
- Data Threshold: near 6
- Near Memory:
  - 16 KB RAM
  - 16 KB ROM
- External Memory:
 

	Start:	Size:
#1: RAM		
#2: RAM		
#3: RAM		
#4: RAM		
#5: RAM		
#6: RAM		
- Options:
  - ☒ Use On-chip ROM (0xC00000 - 0xC0EFFF)
  - ☒ Use On-chip ROM (0xC10000 - 0xCBFFFF)
  - ☒ Use On-chip XRAM (0xA000 - 0xDFFF)
  - ☒ Use On-chip XRAM (0xE00000 - 0xE0FFFF)
- Buttons: OK, Cancel, Defaults, Help





Additional information: Memory Map (Source: User's Manual):

**Options for Target 'Target 1'**

Device: Infineon XE164F-96F

Target:  Output:  Listing:  User:  C166:  EC++:  A166:  L166 Locate:  L166 Misc:  Debug:  Utilities:

Clock (MHz):

Memory Model:

Operating System:

Data Threshold:

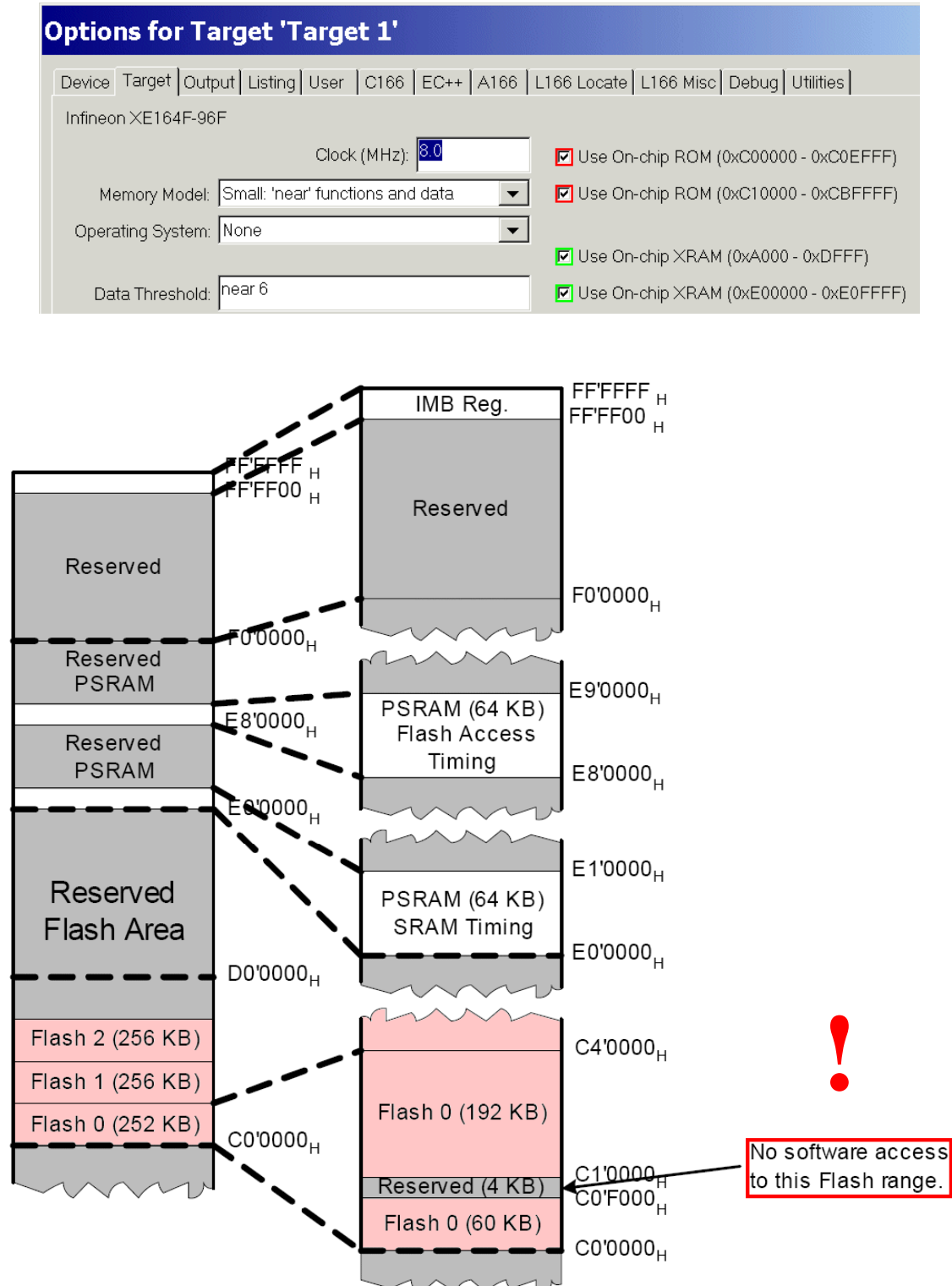
☒ Use On-chip ROM (0xC00000 - 0xC0EFFF)  
☒ Use On-chip ROM (0xC10000 - 0xCBFFFF)  
☒ Use On-chip XRAM (0xA000 - 0xDFFF)  
☒ Use On-chip XRAM (0xE00000 - 0xE0FFFF)

**Table 3-1 XE16x Memory Map <sup>1)</sup>**

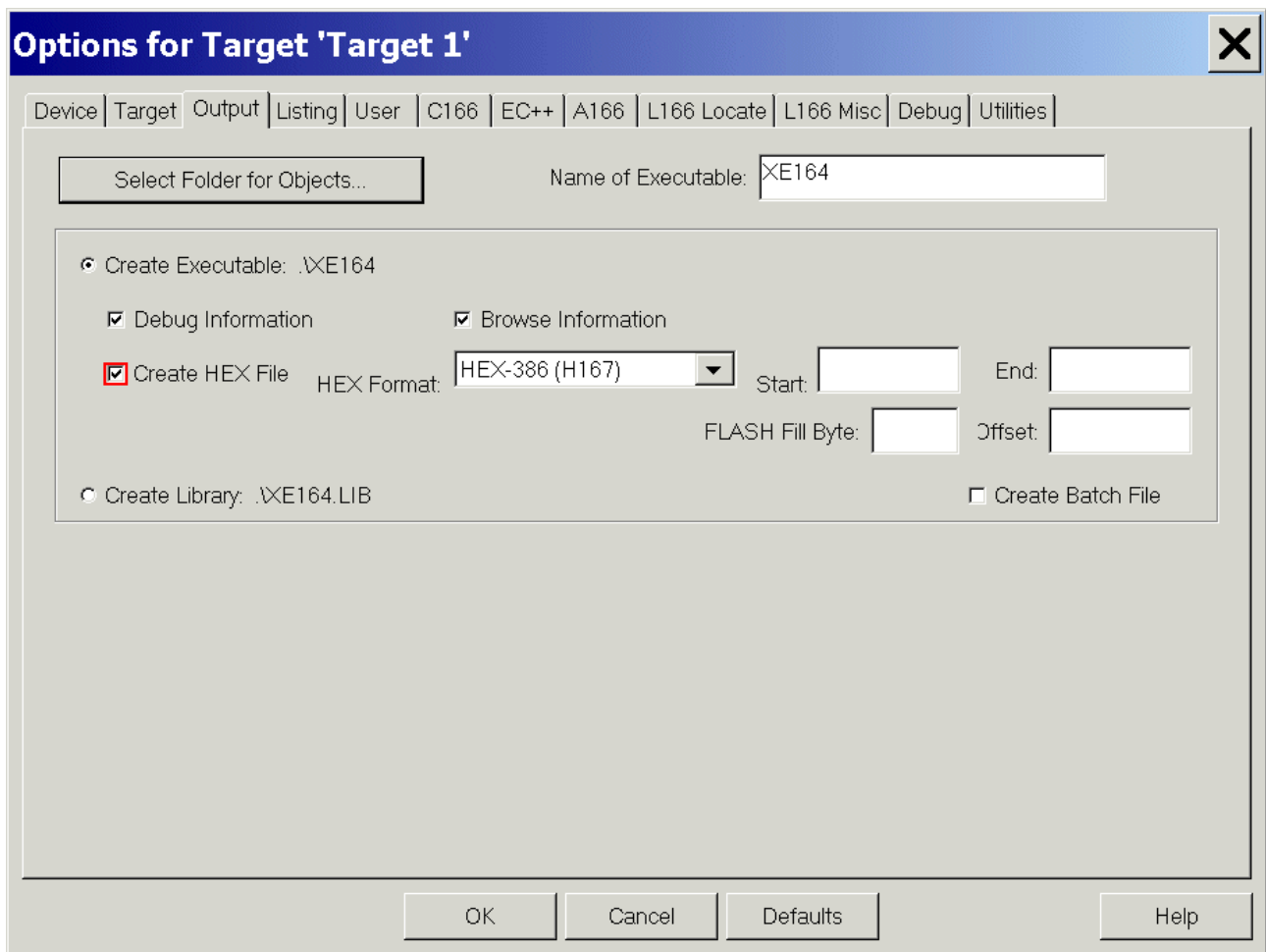
Address Area	Start Loc.	End Loc.	Area Size <sup>2)</sup>	Notes
IMB register space	FF'FF00 <sub>H</sub>	FF'FFFF <sub>H</sub>	256 Bytes	
Reserved (access trap)	F0'0000 <sub>H</sub>	FF'FEFF <sub>H</sub>	< 1 MByte	Minus IMB registers.
Reserved for EPSRAM	E9'0000 <sub>H</sub>	EF'FFFF <sub>H</sub>	448 KBytes	
EPSRAM	E8'0000 <sub>H</sub>	E8'FFFF <sub>H</sub>	64 KBytes	PSRAM with Flash timing.
Reserved for PSRAM	E1'0000 <sub>H</sub>	E7'FFFF <sub>H</sub>	448 KBytes	
PSRAM	E0'0000 <sub>H</sub>	E0'FFFF <sub>H</sub>	64 KBytes	Program SRAM.
Reserved for Flash	CC'0000 <sub>H</sub>	DF'FFFF <sub>H</sub>	<1.25 MBytes	
Flash 2	C8'0000 <sub>H</sub>	CB'FFFF <sub>H</sub>	256 KBytes	
Flash 1	C4'0000 <sub>H</sub>	C7'FFFF <sub>H</sub>	256 KBytes	
Flash 0	C0'0000 <sub>H</sub>	C3'FFFF <sub>H</sub>	252 KBytes <sup>3)</sup>	Minus res. seg.
External memory area	40'0000 <sub>H</sub>	BF'FFFF <sub>H</sub>	8 MBytes	
External IO area <sup>4)</sup>	20'5800 <sub>H</sub>	3F'FFFF <sub>H</sub>	< 2 MBytes	Minus CAN/USIC
USIC registers	20'4000 <sub>H</sub>	20'57FF <sub>H</sub>	6 KBytes	Accessed via EBC
MultiCAN registers	20'0000 <sub>H</sub>	20'3FFF <sub>H</sub>	16 KBytes	Accessed via EBC
External memory area	01'0000 <sub>H</sub>	1F'FFFF <sub>H</sub>	< 2 MBytes	Minus segment 0
SFR area	00'FE00 <sub>H</sub>	00'FFFF <sub>H</sub>	0.5 KBytes	
Dual-port RAM (DPRAM)	00'F600 <sub>H</sub>	00'FDFF <sub>H</sub>	2 KBytes	
Reserved for DPRAM	00'F200 <sub>H</sub>	00'F5FF <sub>H</sub>	1 KBytes	
ESFR area	00'F000 <sub>H</sub>	00'F1FF <sub>H</sub>	0.5 KBytes	
XSFR area	00'E000 <sub>H</sub>	00'EFFF <sub>H</sub>	4 KBytes	
Data SRAM (DSRAM)	00'A000 <sub>H</sub>	00'DFFF <sub>H</sub>	16 KBytes	
Reserved for DSRAM	00'8000 <sub>H</sub>	00'9FFF <sub>H</sub>	8 KBytes	
External memory area	00'0000 <sub>H</sub>	00'7FFF <sub>H</sub>	32 KBytes	



Additional information: Memory Map (Source: User's Manual):



Output: **click** ☒ Create HEX File



**Options for Target 'Target 1'**

Device | Target | **Output** | Listing | User | C166 | EC++ | A166 | L166 Locate | L166 Misc | Debug | Utilities

Select Folder for Objects...      Name of Executable: XE164

☒ Create Executable: .\XE164

☒ Debug Information      ☒ Browse Information

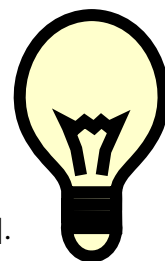
☒ Create HEX File      HEX Format: HEX-386 (H167)      Start:      End:      FLASH Fill Byte:      Offset:      ☐ Create Batch File

☐ Create Library: .\XE164.LIB

OK      Cancel      Defaults      Help

**Note:**

The HEX File could be used while working with the program MEMTOOL for OnChip-Flash-Programming via RS232-interface [Bootstrap Loader (BSL) Mode via UART/USIC0\_CH0].



**Listing:** (do nothing)

**Options for Target 'Target 1'**
✕

Device | Target | Output | Listing | User | C166 | EC++ | A166 | L166 Locate | L166 Misc | Debug | Utilities

Select Folder for Listings...

Page Width:

Page Length:

☒ C Compiler Listing: \*.lst
 

☒ Conditional
☐ Symbols
☐ #include Files
☐ Assembly Code

☐ C Preprocessor Listing: \*.i

☒ Assembler Listing: \*.lst
 

☒ Conditional
☒ Symbols
Macros: 
☐ Cross Reference

☒ Linker Listing: \*.XE164.m66
 

☒ Memory Map  
☒ Local Symbols

☒ Public Symbols  
☒ Comment Records

☒ Line Numbers  
☒ Generated Symbols  
☒ Library Symbols

☐ Cross Reference

OK
Cancel
Defaults
Help



User: (do nothing)

**Options for Target 'Target 1'**
✕

Device
Target
Output
Listing
User
C166
EC++
A166
L166 Locate
L166 Misc
Debug
Utilities

Run User Programs Before Compilation of a C/C++ File

☐ Run #1:
 

...

☐ DOS16

☐ Run #2:
 

...

☐ DOS16

Run User Programs Before Build/Rebuild

☐ Run #1:
 

...

☐ DOS16

☐ Run #2:
 

...

☐ DOS16

Run User Programs After Build/Rebuild

☐ Run #1:
 

...

☐ DOS16

☐ Run #2:
 

...

☐ DOS16

☒ Beep When Complete
 

☐ Start Debugging

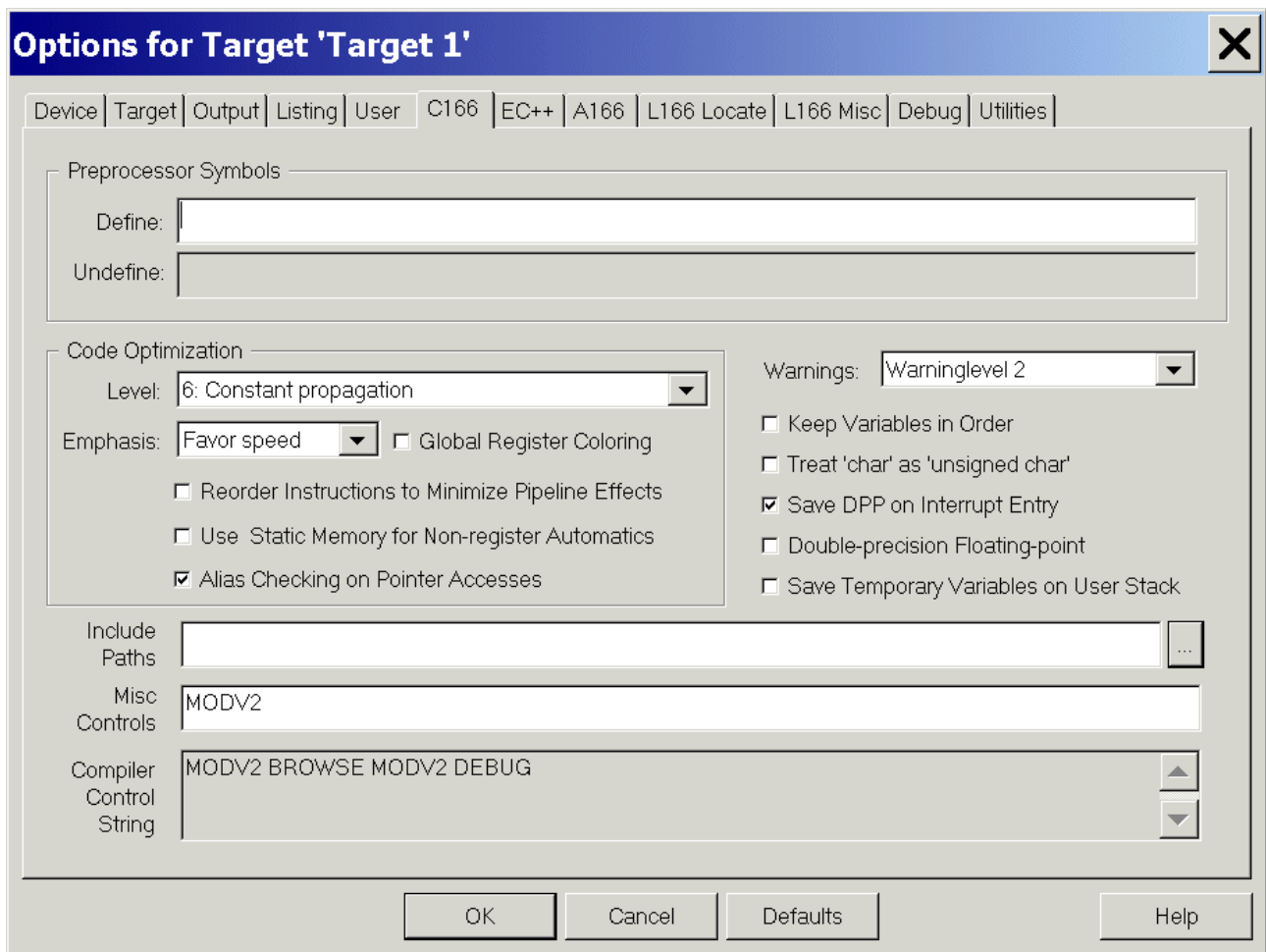
OK

Cancel

Defaults

Help

C166: (do nothing)



**Options for Target 'Target 1'**

Device | Target | Output | Listing | User | **C166** | EC++ | A166 | L166 Locate | L166 Misc | Debug | Utilities

Preprocessor Symbols

Define:

Undefine:

Code Optimization

Level: **6: Constant propagation**

Emphasis: **Favor speed** ☐ Global Register Coloring

☐ Reorder Instructions to Minimize Pipeline Effects

☐ Use Static Memory for Non-register Automatics

☒ Alias Checking on Pointer Accesses

Warnings: **Warninglevel 2**

☐ Keep Variables in Order

☐ Treat 'char' as 'unsigned char'

☒ Save DPP on Interrupt Entry

☐ Double-precision Floating-point

☐ Save Temporary Variables on User Stack

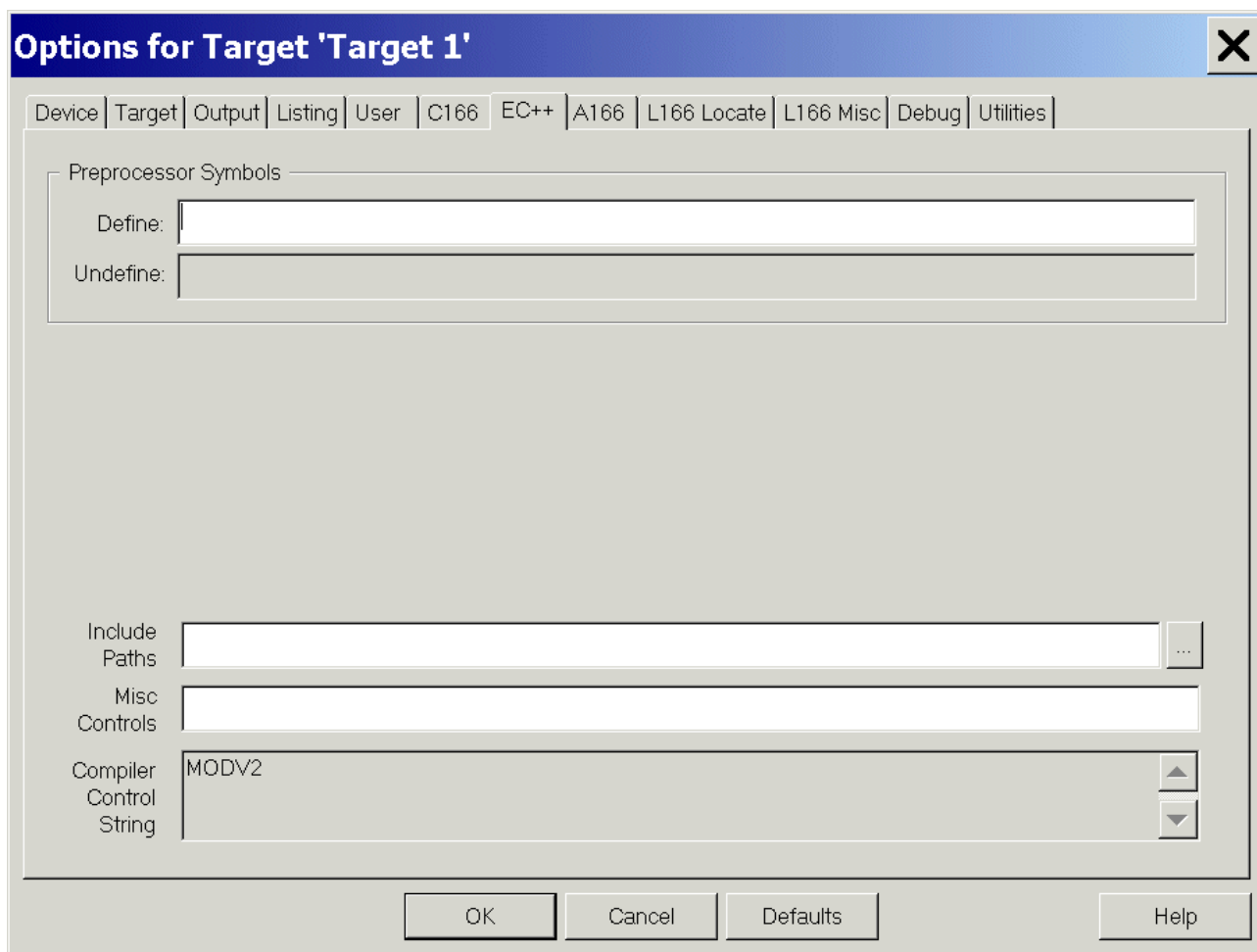
Include Paths  ...

Misc Controls **MODV2**

Compiler Control String **MODV2 BROWSE MODV2 DEBUG**

OK Cancel Defaults Help

EC++: (do nothing)



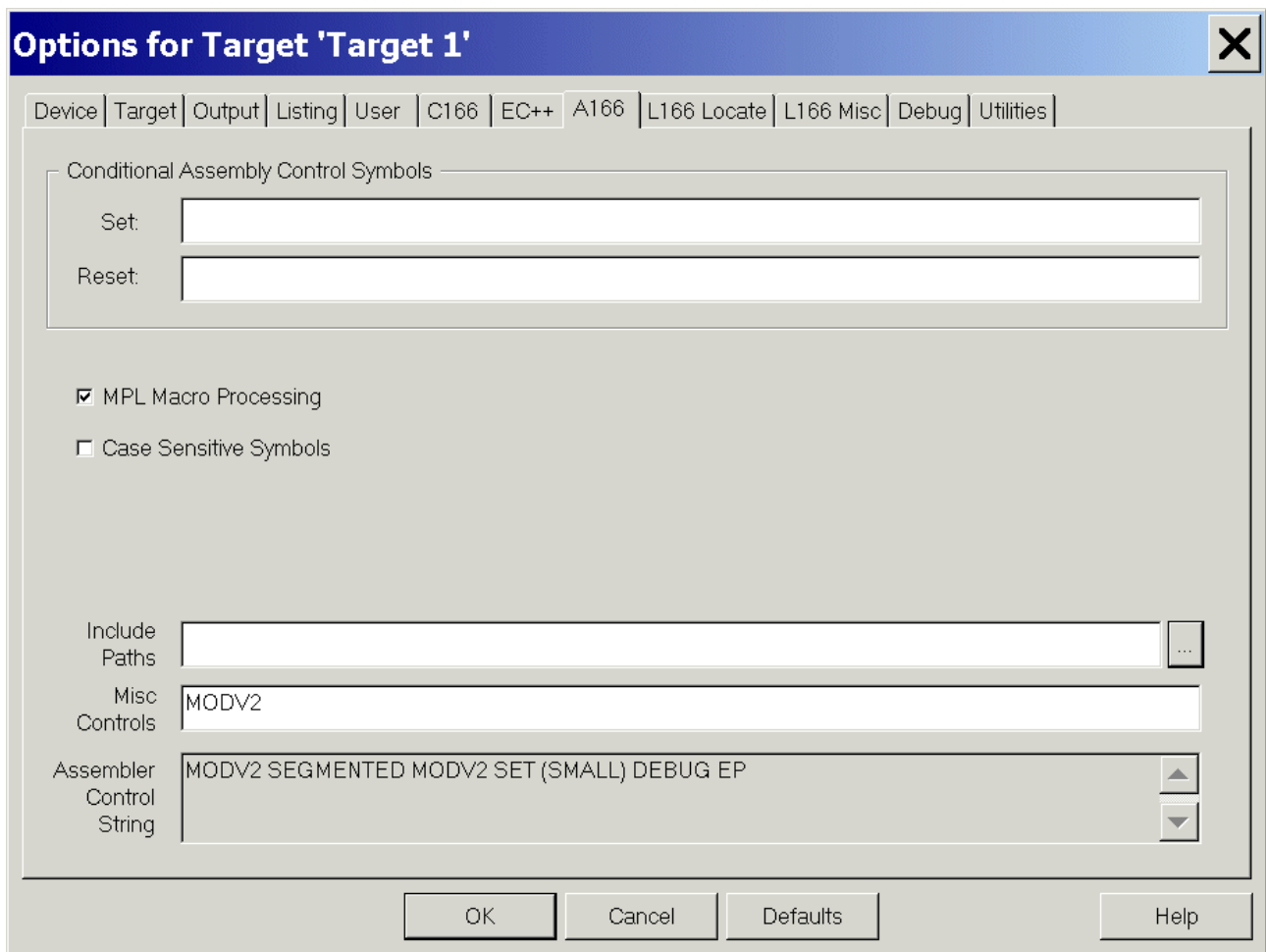
The dialog box is titled "Options for Target 'Target 1'". It features a tabbed interface with the following tabs: Device, Target, Output, Listing, User, C166, EC++, A166, L166 Locate, L166 Misc, Debug, and Utilities. The "EC++" tab is currently selected.

Under the "Preprocessor Symbols" section, there are two text input fields: "Define:" and "Undefine:". The "Define:" field is currently empty.

Below the preprocessor symbols, there are three more input fields: "Include Paths", "Misc Controls", and "Compiler Control String". The "Include Paths" field has a browse button (three dots) to its right. The "Compiler Control String" field is currently set to "MODV2" and has up and down arrow buttons to its right.

At the bottom of the dialog, there are four buttons: "OK", "Cancel", "Defaults", and "Help".

A166: (do nothing)



The dialog box is titled "Options for Target 'Target 1'". It features a tabbed interface with the following tabs: Device, Target, Output, Listing, User, C166, EC++, A166 (selected), L166 Locate, L166 Misc, Debug, and Utilities. The A166 tab is active, showing the following options:

- Conditional Assembly Control Symbols**: A section with two text input fields labeled "Set:" and "Reset:", both of which are currently empty.
- MPL Macro Processing**: A checkbox that is checked.
- Case Sensitive Symbols**: A checkbox that is unchecked.
- Include Paths**: A text input field containing an empty string, followed by a browse button (three dots).
- Misc Controls**: A text input field containing the text "MODV2".
- Assembler Control String**: A text input field containing the text "MODV2 SEGMENTED MODV2 SET (SMALL) DEBUG EP", with up and down arrow buttons on the right.

At the bottom of the dialog box are four buttons: OK, Cancel, Defaults, and Help.



**L166 Locate:** (do nothing)

**Options for Target 'Target 1'**
✕

Device

Target

Output

Listing

User

C166

EC++

A166

L166 Locate

L166 Misc

Debug

Utilities

☒ Use Memory Layout from Target Dialog

C166 Variable Initialization Tables 0xC10000 - 0xCBFFFF

DPP Usage

☐ DPPUSE

ndata

dpp2

nconst

dpp1

Target Classes

ICODE (0xC00000-0xC0EFFF), NCODE (0xC10000-0xC1FFFF),  
 FCONST (0xC00000-0xC0EFFF, 0xC10000-0xCBFFFF), HCONST (0xC00000-0xC0EFFF, 0xC10000-0xCBFFFF),  
 XCONST (0xC00000-0xC0EFFF, 0xC10000-0xCBFFFF), NCONST (0xC04000-0xC07FFF),

▲  
▼

User Classes

▲  
▼

User Sections

▲  
▼

Linker Control String

TO "XE164"  
 CLASSES (ICODE (0xC00000-0xC0EFFF), NCODE (0xC10000-0xC1FFFF),  
 FCONST (0xC00000-0xC0EFFF, 0xC10000-0xCBFFFF), HCONST (0xC00000-0xC0EFFF, 0xC10000-0xCBFFFF), XCONST (0xC00000-0xC0EFFF, 0xC10000-0xCBFFFF), NCONST (0xC04000-0xC07FFF))

▲  
▼

OK

Cancel

Defaults

Help

L166 Misc: Interrupt Vector Table Address: insert 0x0C00000

**Options for Target 'Target 1'**

Device | Target | Output | Listing | User | C166 | EC++ | A166 | L166 Locate | **L166 Misc** | Debug | Utilities

Warnings  
Level 2 ▾ Disable Warning Numbers:

☐ use linker control file:  
Create... Browse... Edit...

☐ Create Relocatable Output File (LINKONLY) Interrupt Vector Table Address:

Assign

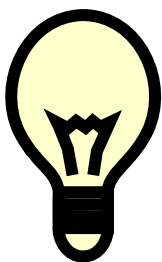
RegBank

Reserve

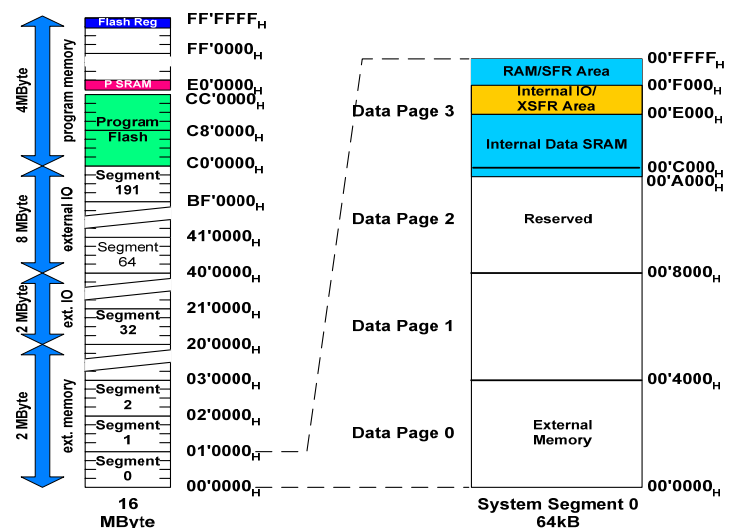
Misc Controls

Linker Control String TO "XE164"  
VECTAB (0x0C00000)  
CLASSES (ICODE (0xC00000-0xC0EFFF), NCODE (0xC10000-0xC1FFFF),

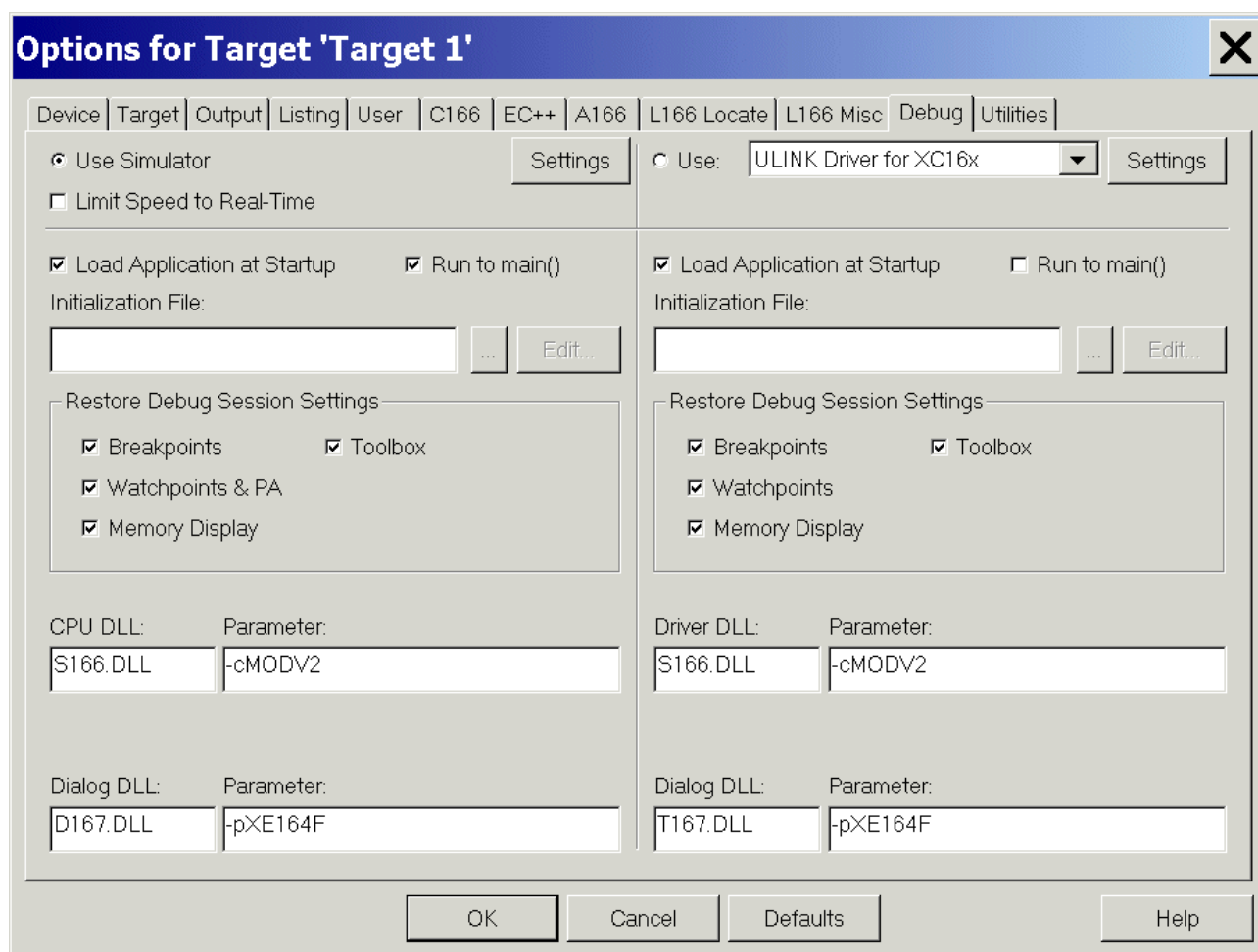
OK Cancel Defaults Help



**Note:**  
The On Chip Flash starts here.



**Debug:** (do nothing)

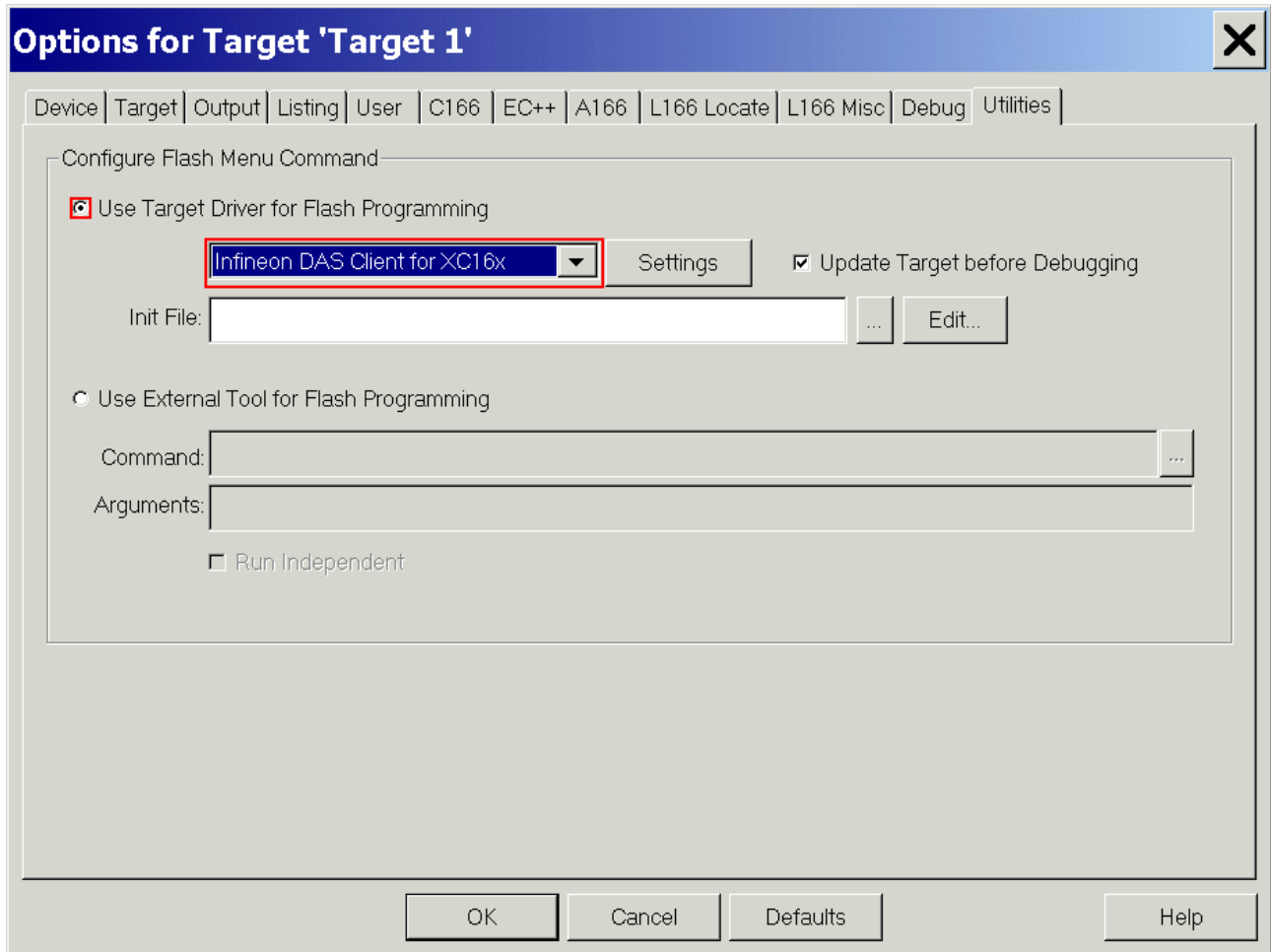


The dialog box 'Options for Target 'Target 1'' contains the following settings:

- Device:** Target | Output | Listing | User | C166 | EC++ | A166 | L166 Locate | L166 Misc | **Debug** | Utilities
- Use Simulator:** ☒ (Settings button)
- Limit Speed to Real-Time:** ☐
- Load Application at Startup:** ☒ | **Run to main():** ☒
- Initialization File:** [Empty field] (Browse button) (Edit button)
- Restore Debug Session Settings:**
  - ☒ Breakpoints | ☒ Toolbox
  - ☒ Watchpoints & PA
  - ☒ Memory Display
- CPU DLL:** S166.DLL | **Parameter:** -cMODV2
- Dialog DLL:** D167.DLL | **Parameter:** -pXE164F
- Use:** ULINK Driver for XC16x (Settings button)
- Load Application at Startup:** ☒ | **Run to main():** ☐
- Initialization File:** [Empty field] (Browse button) (Edit button)
- Restore Debug Session Settings:**
  - ☒ Breakpoints | ☒ Toolbox
  - ☒ Watchpoints
  - ☒ Memory Display
- Driver DLL:** S166.DLL | **Parameter:** -cMODV2
- Dialog DLL:** T167.DLL | **Parameter:** -pXE164F

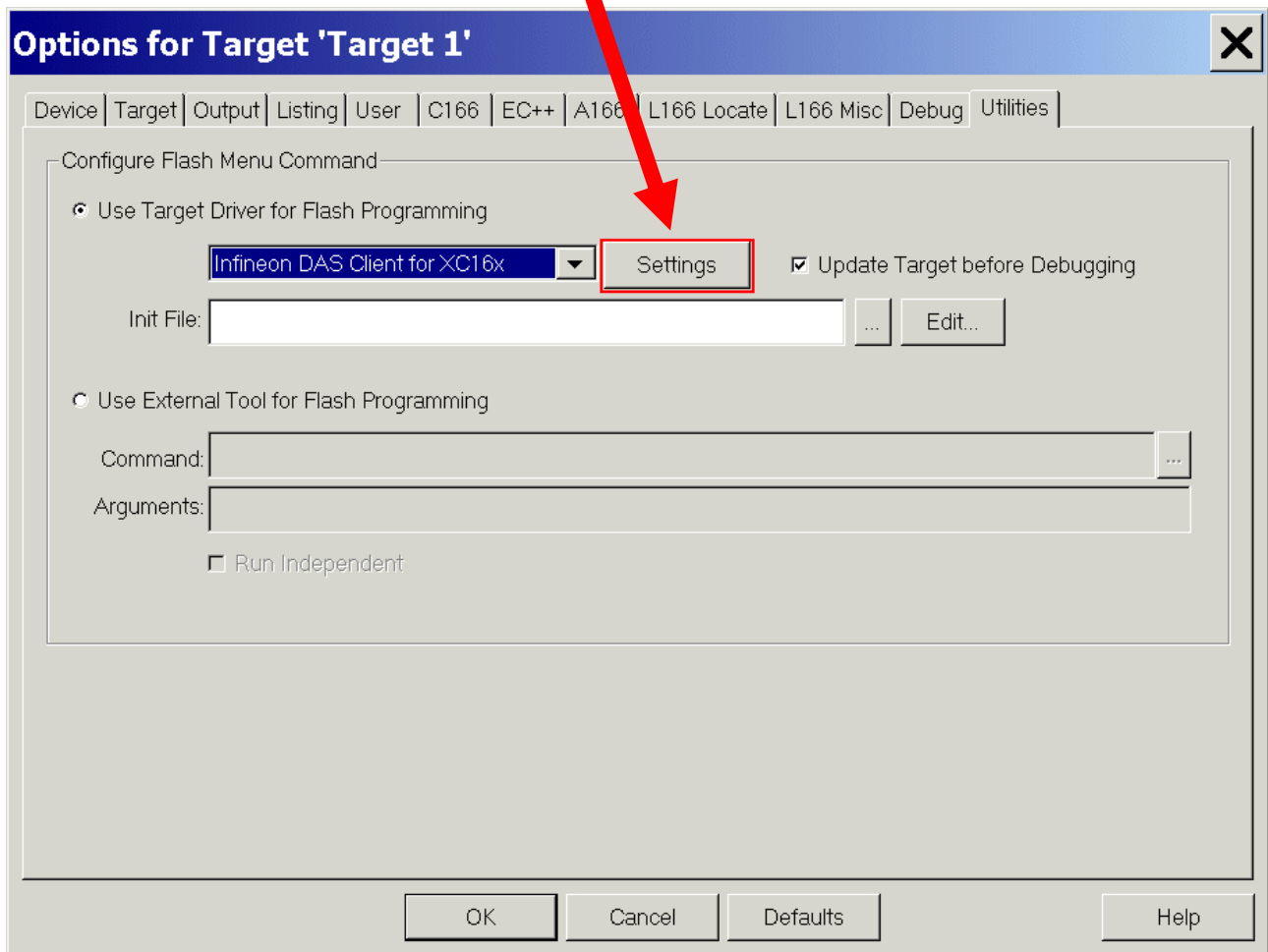
Buttons at the bottom: OK, Cancel, Defaults, Help

Utilities: Configure Flash Menu Command: **check** ☒ Use Target Driver for Flash Programming  
Utilities: Configure Flash Menu Command: **select** Infineon DAS Client for XC16x

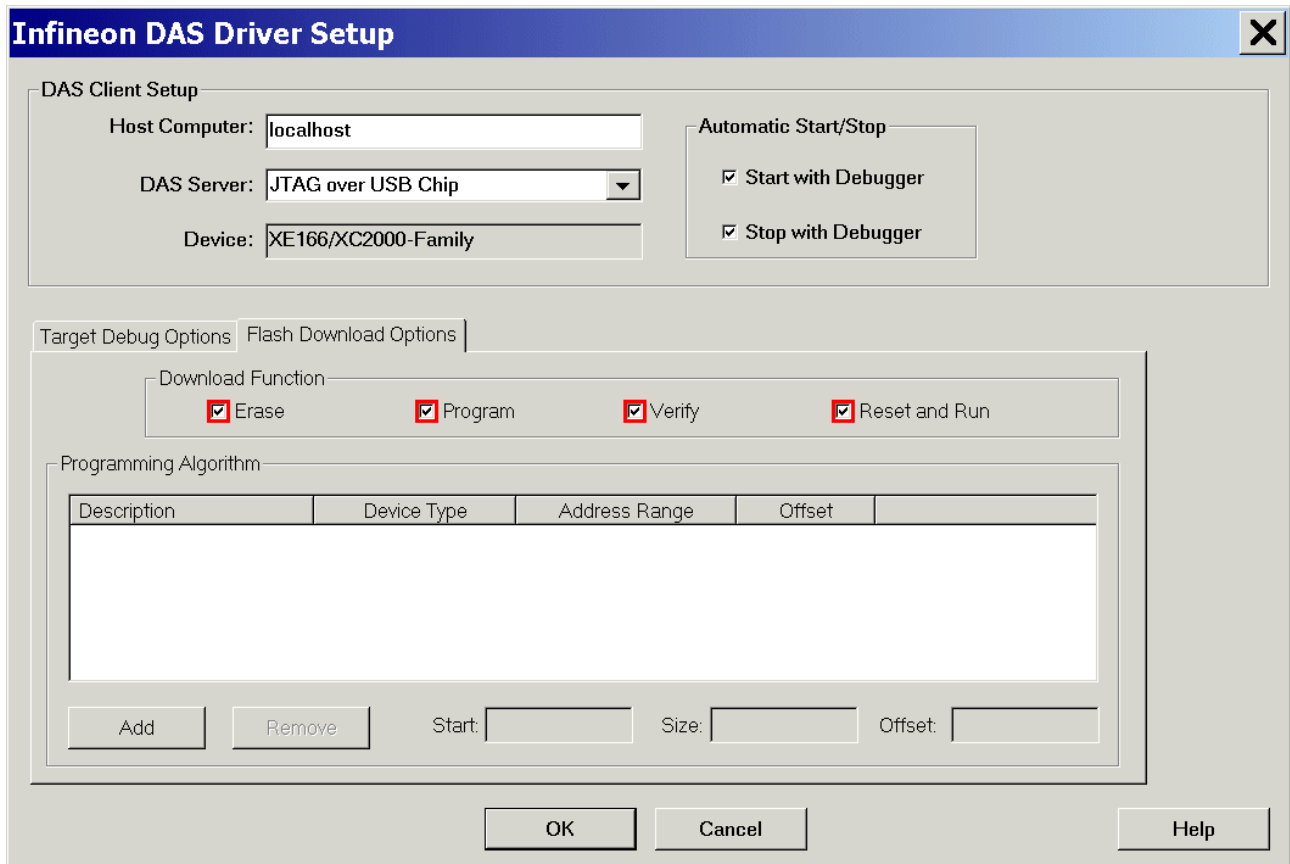




Utilities: Configure Flash Menu Command: **click** Settings



Flash Download Options: Download Function: check: ☒ Erase  
Flash Download Options: Download Function: check: ☒ Program  
Flash Download Options: Download Function: check: ☒ Verify  
Flash Download Options: Download Function: check: ☒ Reset and Run



**Infineon DAS Driver Setup**

**DAS Client Setup**

Host Computer:

DAS Server:

Device:

**Automatic Start/Stop**

☒ Start with Debugger

☒ Stop with Debugger

**Target Debug Options** | **Flash Download Options**

**Download Function**

☒ Erase    ☒ Program    ☒ Verify    ☒ Reset and Run

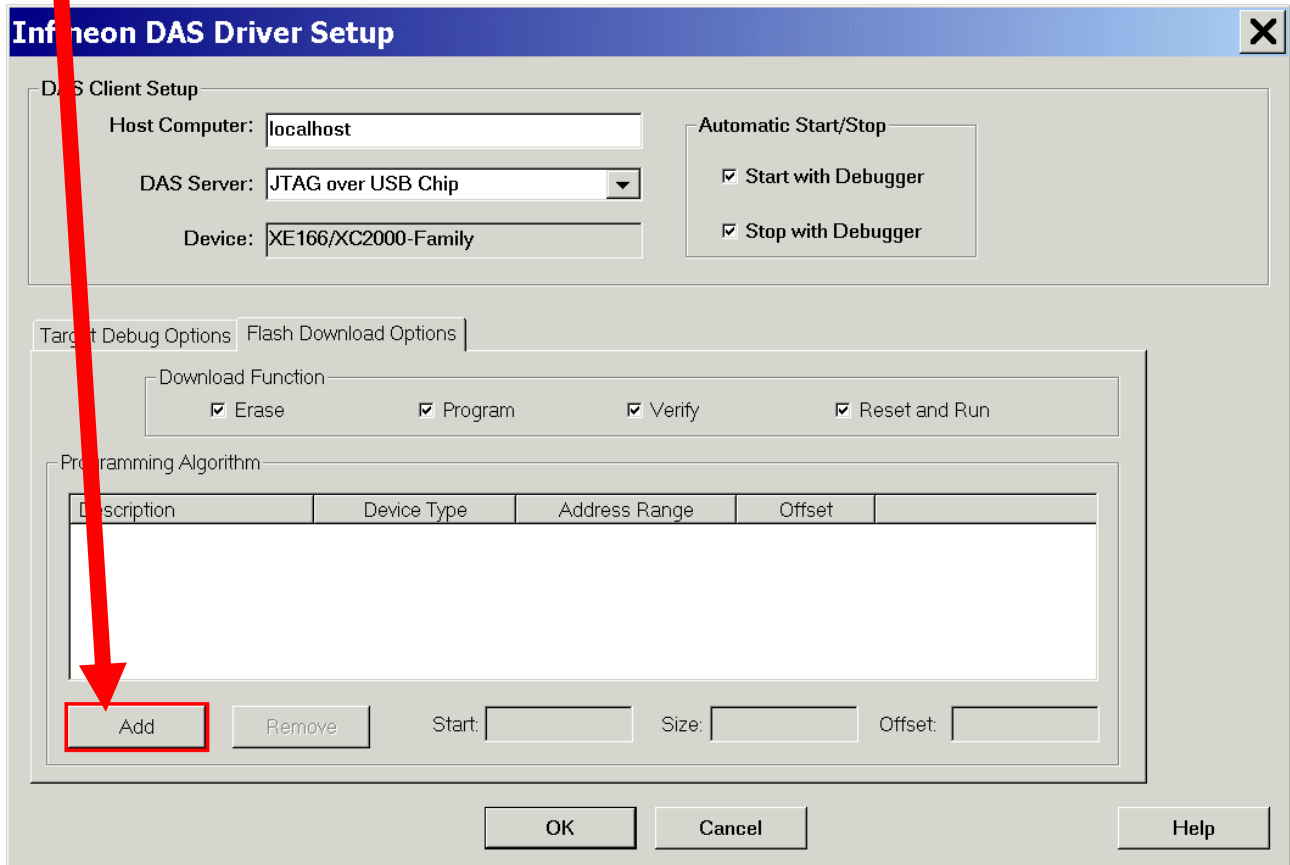
**Programming Algorithm**

Description	Device Type	Address Range	Offset

Add    Remove    Start:     Size:     Offset:

OK    Cancel    Help

Click Add



The image shows the 'Infineon DAS Driver Setup' dialog box. A red arrow points from the text 'Click Add' to the 'Add' button in the 'Programming Algorithm' section. The dialog box has a title bar with the Infineon logo and a close button. It contains several sections: 'DAS Client Setup' with fields for 'Host Computer' (localhost), 'DAS Server' (JTAG over USB Chip), and 'Device' (XE166/XC2000-Family); 'Automatic Start/Stop' with checkboxes for 'Start with Debugger' and 'Stop with Debugger'; 'Target Debug Options' and 'Flash Download Options' tabs; 'Download Function' with checkboxes for 'Erase', 'Program', 'Verify', and 'Reset and Run'; and 'Programming Algorithm' which includes a table with columns 'Description', 'Device Type', 'Address Range', and 'Offset'. Below the table are 'Add', 'Remove', 'Start', 'Size', and 'Offset' buttons. At the bottom are 'OK', 'Cancel', and 'Help' buttons.

**Infineon DAS Driver Setup**

DAS Client Setup

Host Computer: localhost

DAS Server: JTAG over USB Chip

Device: XE166/XC2000-Family

Automatic Start/Stop

☒ Start with Debugger

☒ Stop with Debugger

Target Debug Options | Flash Download Options

Download Function

☒ Erase ☒ Program ☒ Verify ☒ Reset and Run

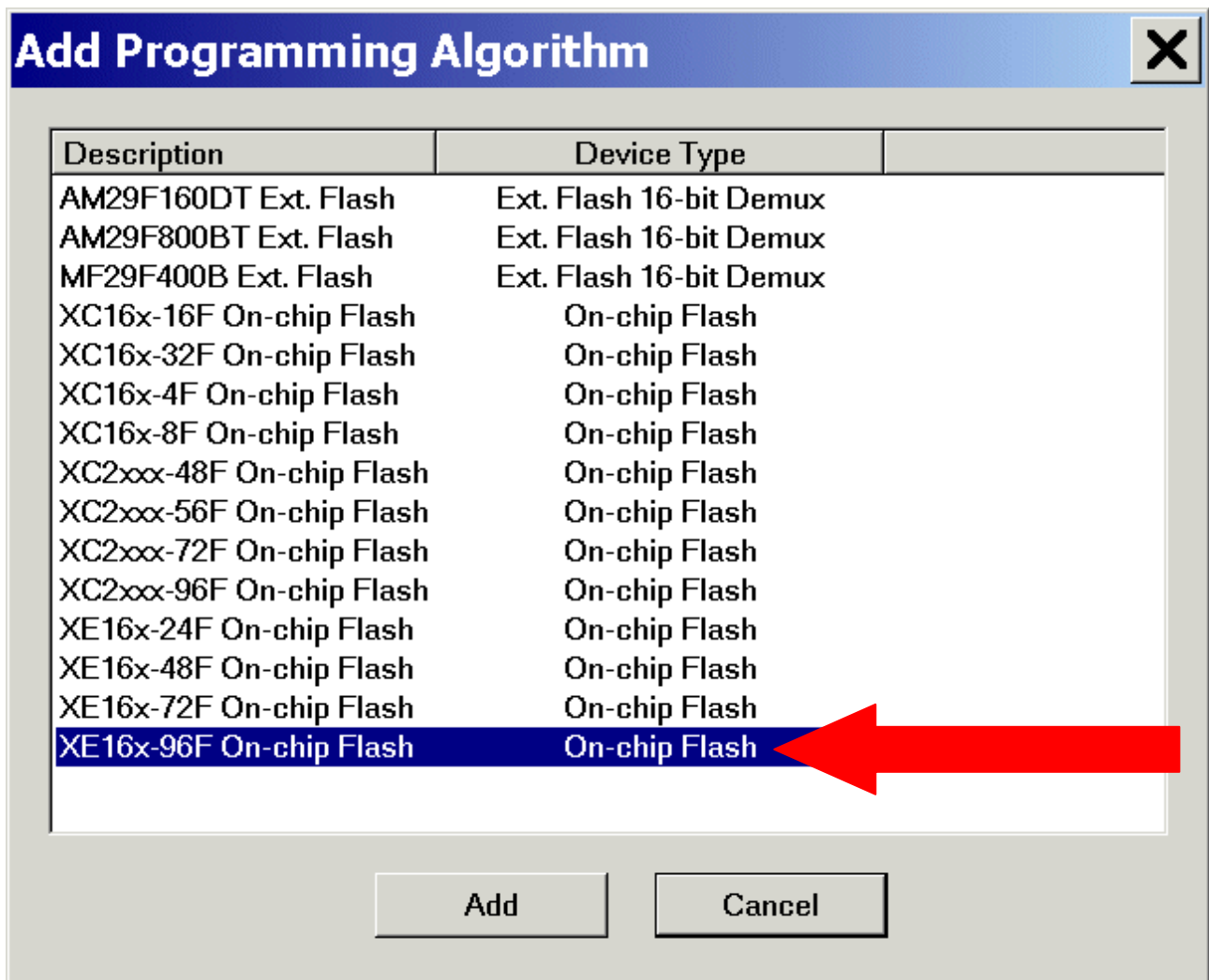
Programming Algorithm

Description	Device Type	Address Range	Offset
-------------	-------------	---------------	--------

Add Remove Start: Size: Offset:

OK Cancel Help

Select: XE16x-96F On-chip Flash



Click Add



**Infineon DAS Driver Setup**
✕

**DAS Client Setup**

Host Computer:   
DAS Server:   
Device:

**Automatic Start/Stop**  
☒ Start with Debugger  
☒ Stop with Debugger

Target Debug Options

Flash Download Options

**Download Function**

☒ Erase
☒ Program
☒ Verify
☒ Reset and Run

**Programming Algorithm**

Description	Device Type	Address Range	Offset
XE16x-96F On-chip Flash	On-chip Flash	C00000H - CBFFFFH	000000H

Start: 
Size: 
Offset:

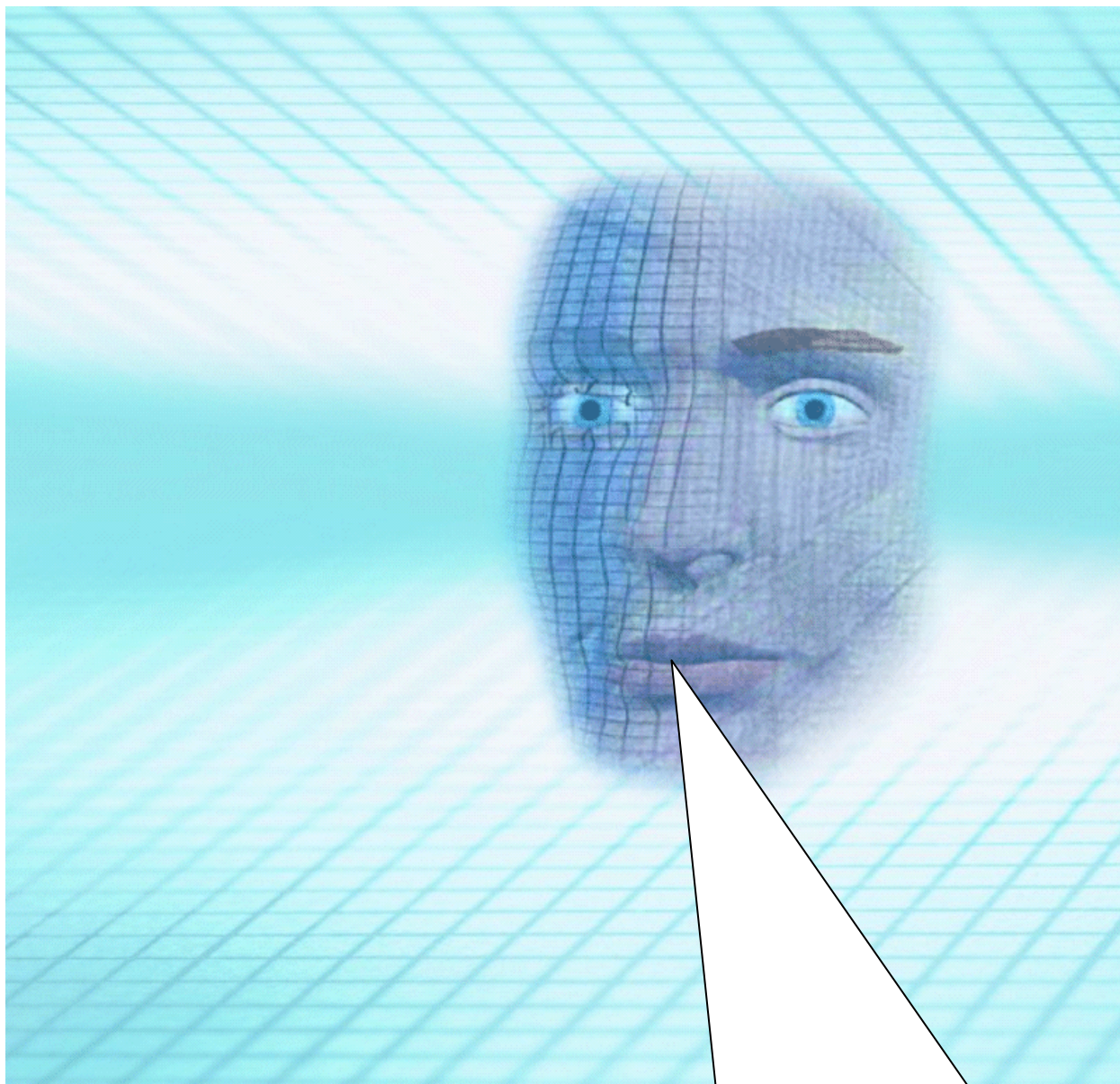
OK

Cancel

Help

OK  
OK

Insert your application specific program:



**Note:**

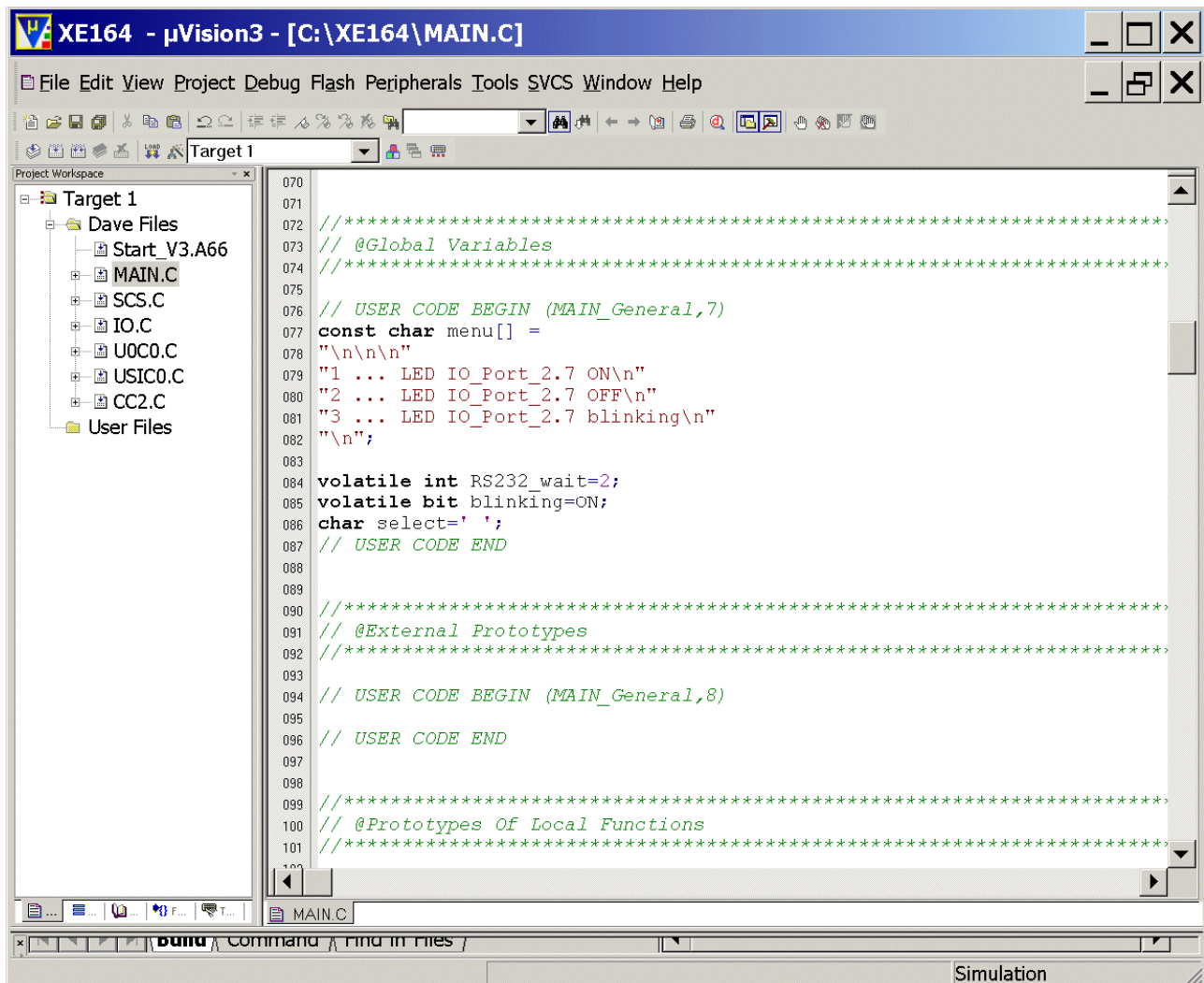
DAvE doesn't change code which is inserted between '`// USER CODE BEGIN`' and '`// USER CODE END`'. Therefore, whenever adding code to DAvE's generated code, write it between '`// USER CODE BEGIN`' and '`// USER CODE END`'.

If you wish to change DAvE's generated code or add code outside these 'USER CODE' sections you will have to insert/modify your changes each time after letting DAvE regenerate code!

Double click **MAIN.C** and insert Global Variables:

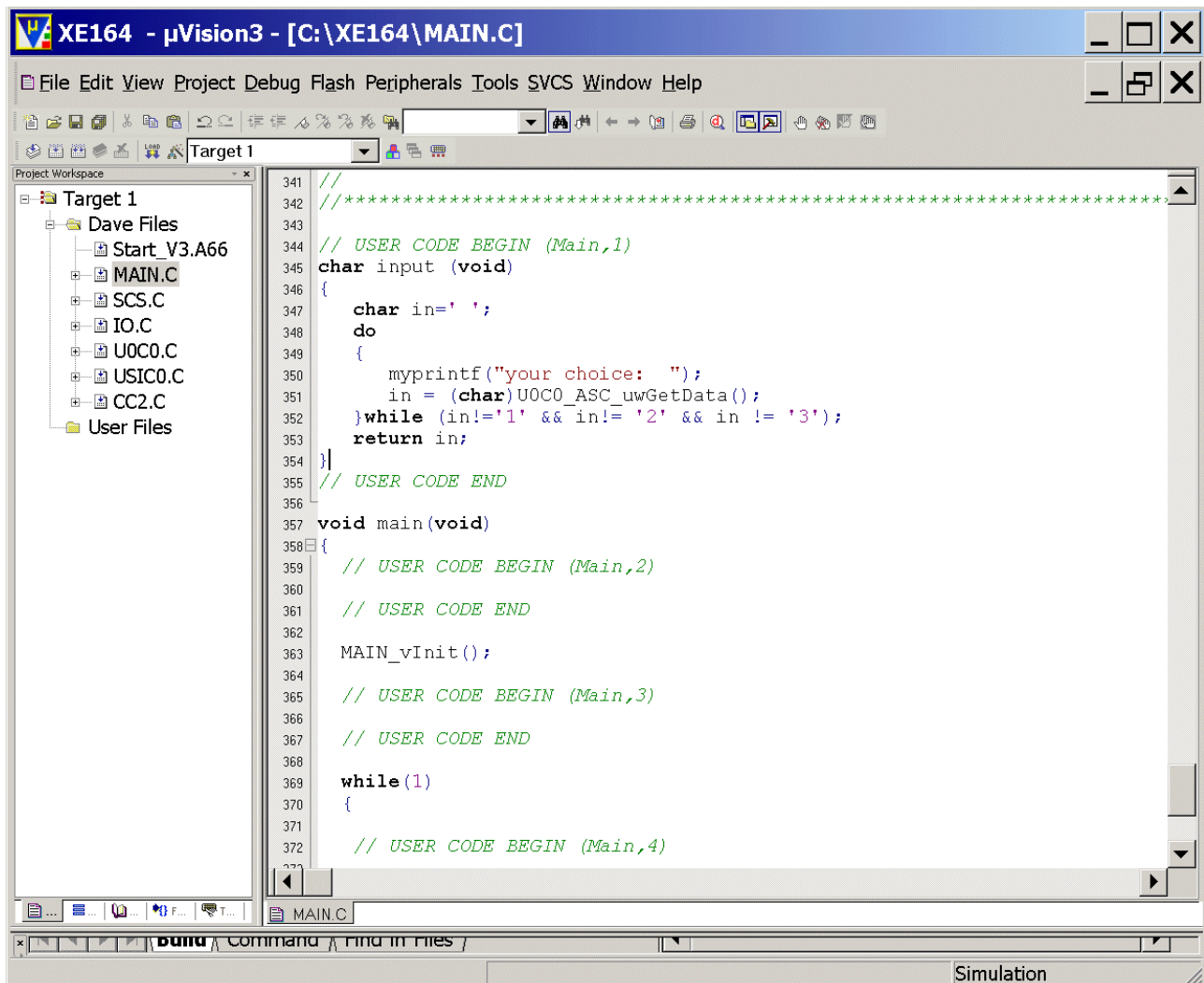
```
const char menu[] =
"\n\n\n"
"1 ... LED IO_Port_2.7 ON\n"
"2 ... LED IO_Port_2.7 OFF\n"
"3 ... LED IO_Port_2.7 blinking\n"
"\n";

volatile int RS232_wait=2;
volatile bit blinking=ON;
char select=' ';
```



Double click **MAIN.C** and insert the function **input()**:

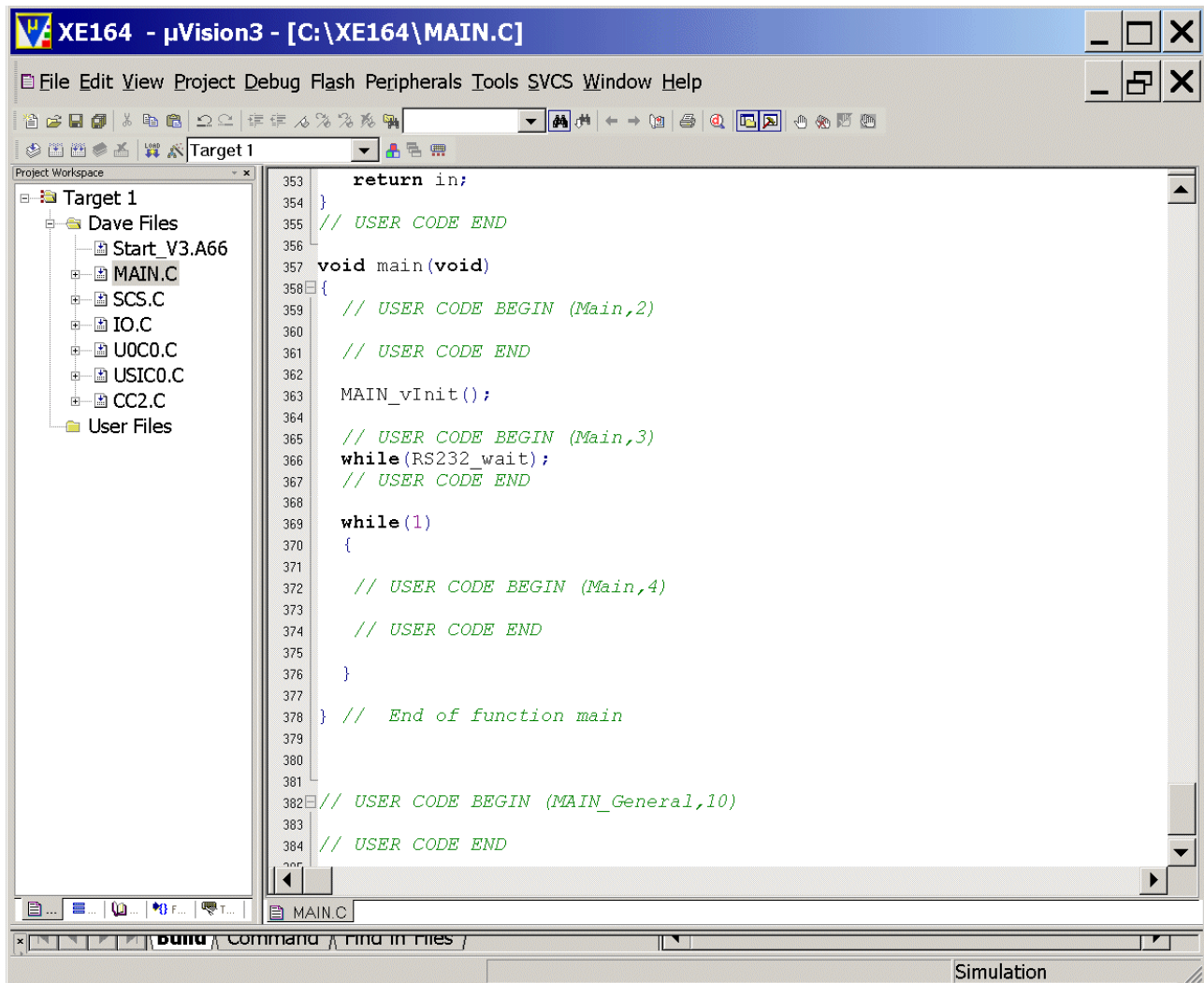
```
char input (void)
{
    char in=' ';
    do
    {
        myprintf("your choice: ");
        in = (char)U0C0_ASC_uwGetData();
    }while (in!='1' && in!= '2' && in != '3');
    return in;
}
```





Double click **MAIN.C** and insert the following code in the **main** function:

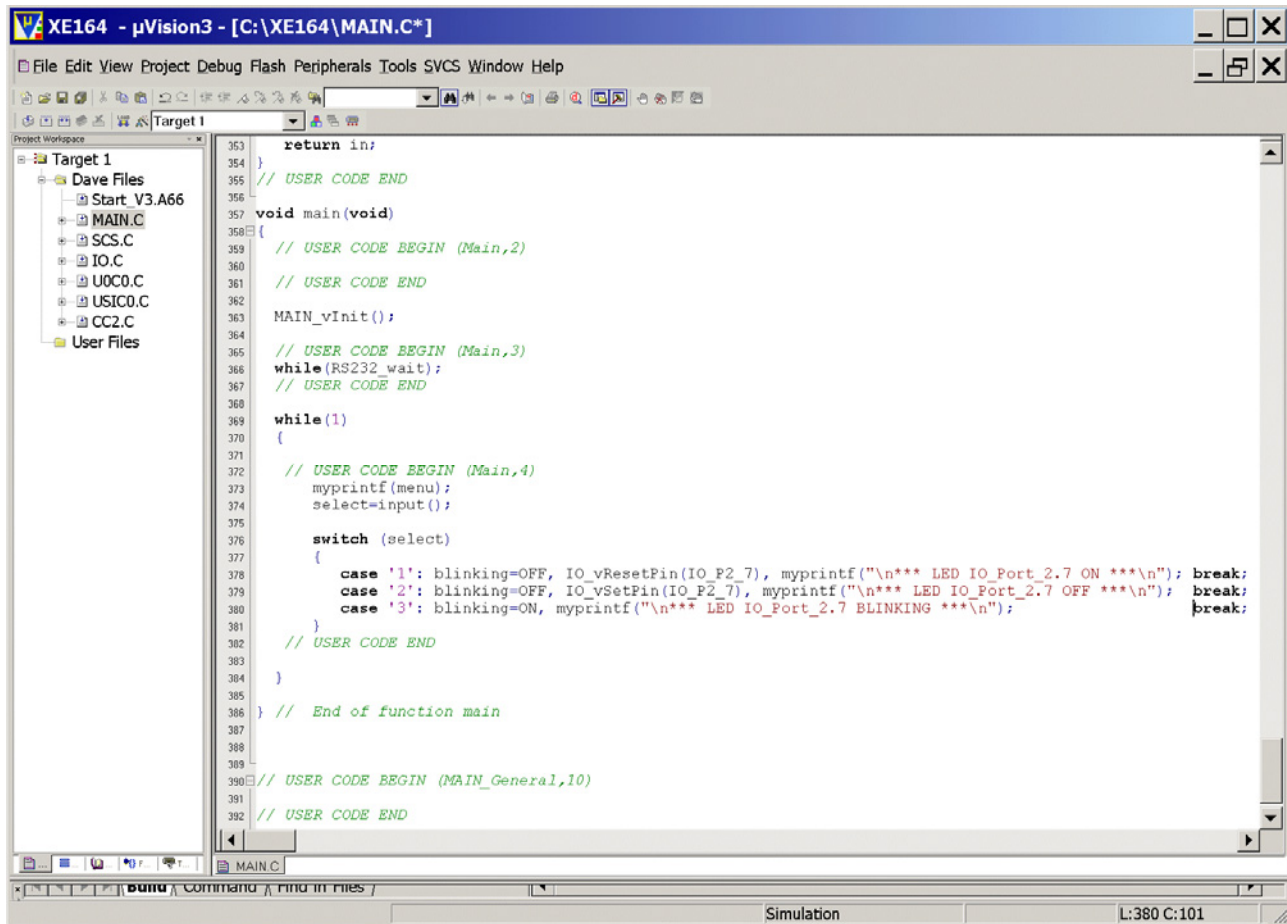
```
while(RS232_wait);
```



Double click **MAIN.C** and **insert** the following code in the **main** function into the **while(1)** loop:

```
myprintf(menu);
select=input();

switch (select)
{
    case '1': blinking=OFF, IO_vResetPin(IO_P2_7), myprintf("\n*** LED IO_Port_2.7 ON
***\n"); break;
    case '2': blinking=OFF, IO_vSetPin(IO_P2_7), myprintf("\n*** LED IO_Port_2.7 OFF
***\n"); break;
    case '3': blinking=ON, myprintf("\n*** LED IO_Port_2.7 BLINKING ***\n");
break;
}
```



```

353     return in;
354 }
355 // USER CODE END
356
357 void main(void)
358 {
359     // USER CODE BEGIN (Main,2)
360     // USER CODE END
361
362     MAIN_vInit();
363
364     // USER CODE BEGIN (Main,3)
365     while(RS232_wait);
366     // USER CODE END
367
368     while(1)
369     {
370     }
371
372     // USER CODE BEGIN (Main,4)
373     myprintf(menu);
374     select=input();
375
376     switch (select)
377     {
378         case '1': blinking=OFF, IO_vResetPin(IO_P2_7), myprintf("\n*** LED IO_Port_2.7 ON ***\n"); break;
379         case '2': blinking=OFF, IO_vSetPin(IO_P2_7), myprintf("\n*** LED IO_Port_2.7 OFF ***\n"); break;
380         case '3': blinking=ON, myprintf("\n*** LED IO_Port_2.7 BLINKING ***\n"); break;
381     }
382     // USER CODE END
383
384 }
385 // End of function main
386
387
388
389
390 // USER CODE BEGIN (MAIN_General,10)
391
392 // USER CODE END
  
```



Additional information: Port Output Modification Register (Source: User's Manual):

### Pn\_OMRL (n=6-11)

Port n Output Modification Register LowXSFR (E9C0<sub>H</sub>+4\*n) Reset Value: XXXX<sub>H</sub>

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC	PC	PC	PC	PC	PC	PC	PC	PS	PS	PS	PS	PS	PS	PS	PS
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W

Field	Bits	Type	Description
<b>PSx</b> (x = 0-7)	x	W	<b>Port Set Bit x</b> Setting this bit sets or toggles the corresponding bit in the port output register Pn_OUT (see <a href="#">Table 7-4</a> ). On a read access, this bit returns 0.
<b>PCx</b> (x = 0-7)	x + 8	W	<b>Port Clear Bit x</b> Setting this bit clears or toggles the corresponding bit in the port output register Pn_OUT. (see <a href="#">Table 7-4</a> ). On a read access, this bit returns 0.

### Function of the PCx and PSx bit fields

Table 7-4 Function of the Bits PCx and PSx

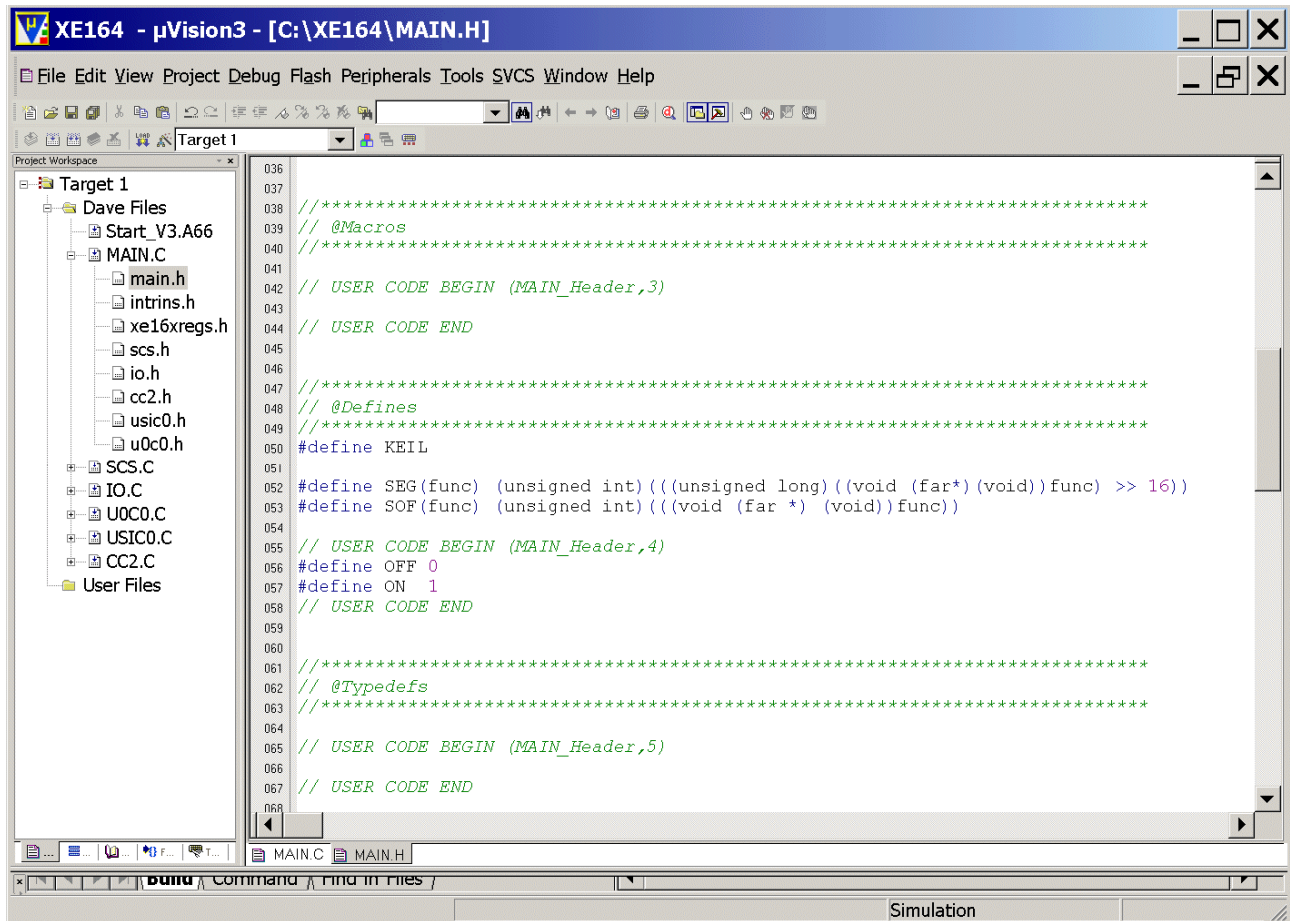
PCx	PSx	Function
0 or no write access	0 or no write access	Bit Pn_OUT.Px is not changed.
0 or no write access	1	Bit Pn_OUT.Px is set.
1	0 or no write access	Bit Pn_OUT.Px is cleared.
1	1	Bit Pn_OUT.Px is toggled.

*Note: If a bit position is not written (one out of two bytes not targeted by a byte write), the corresponding value is considered as 0. Toggling a bit requires one 16-bit write.*



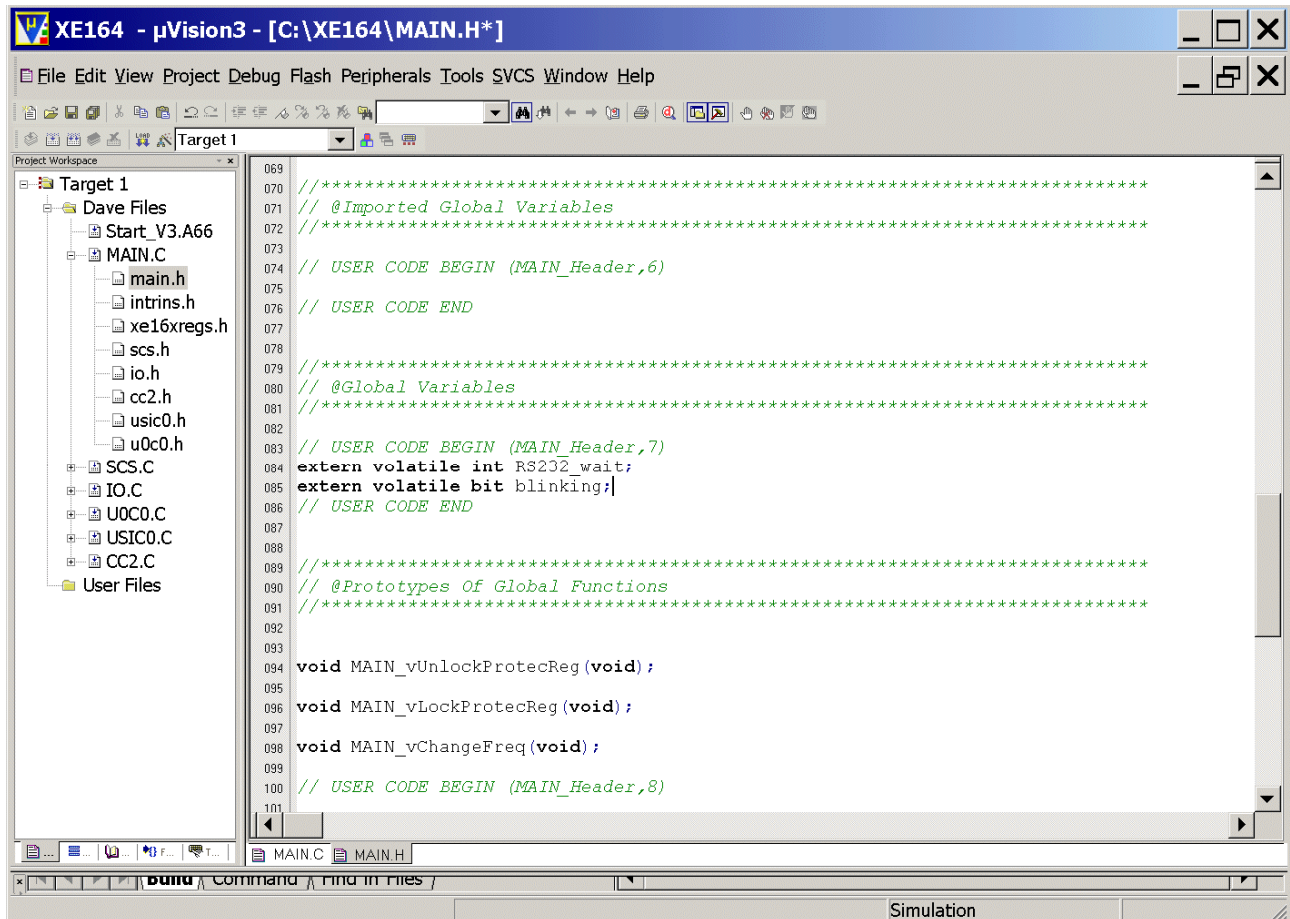
Double click **Main.h** and **insert** the following Defines:

```
#define OFF 0
#define ON 1
```



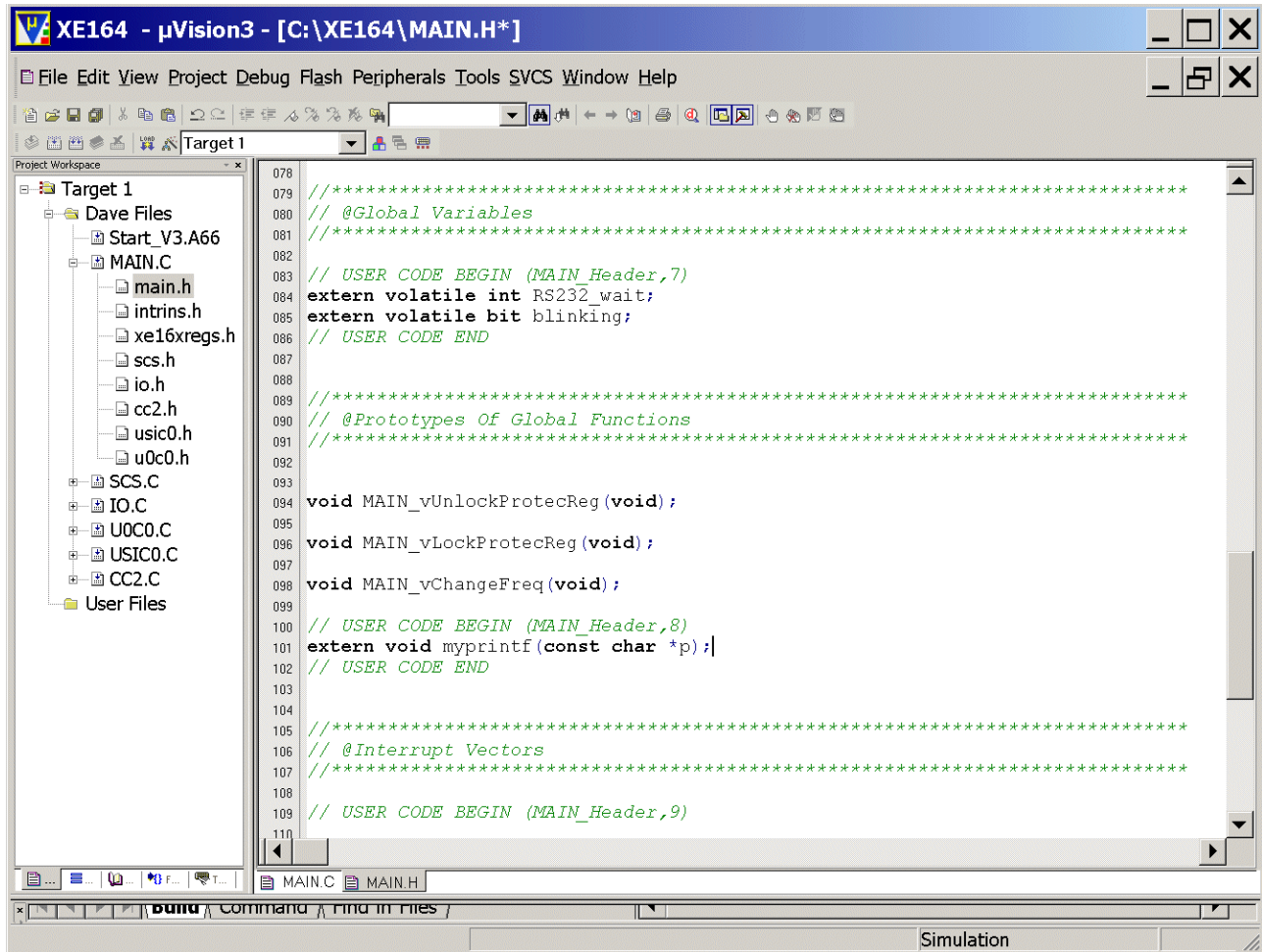
Double click **Main.h** and insert extern declarations "Global Variables":

```
extern volatile int RS232_wait;
extern volatile bit blinking;
```



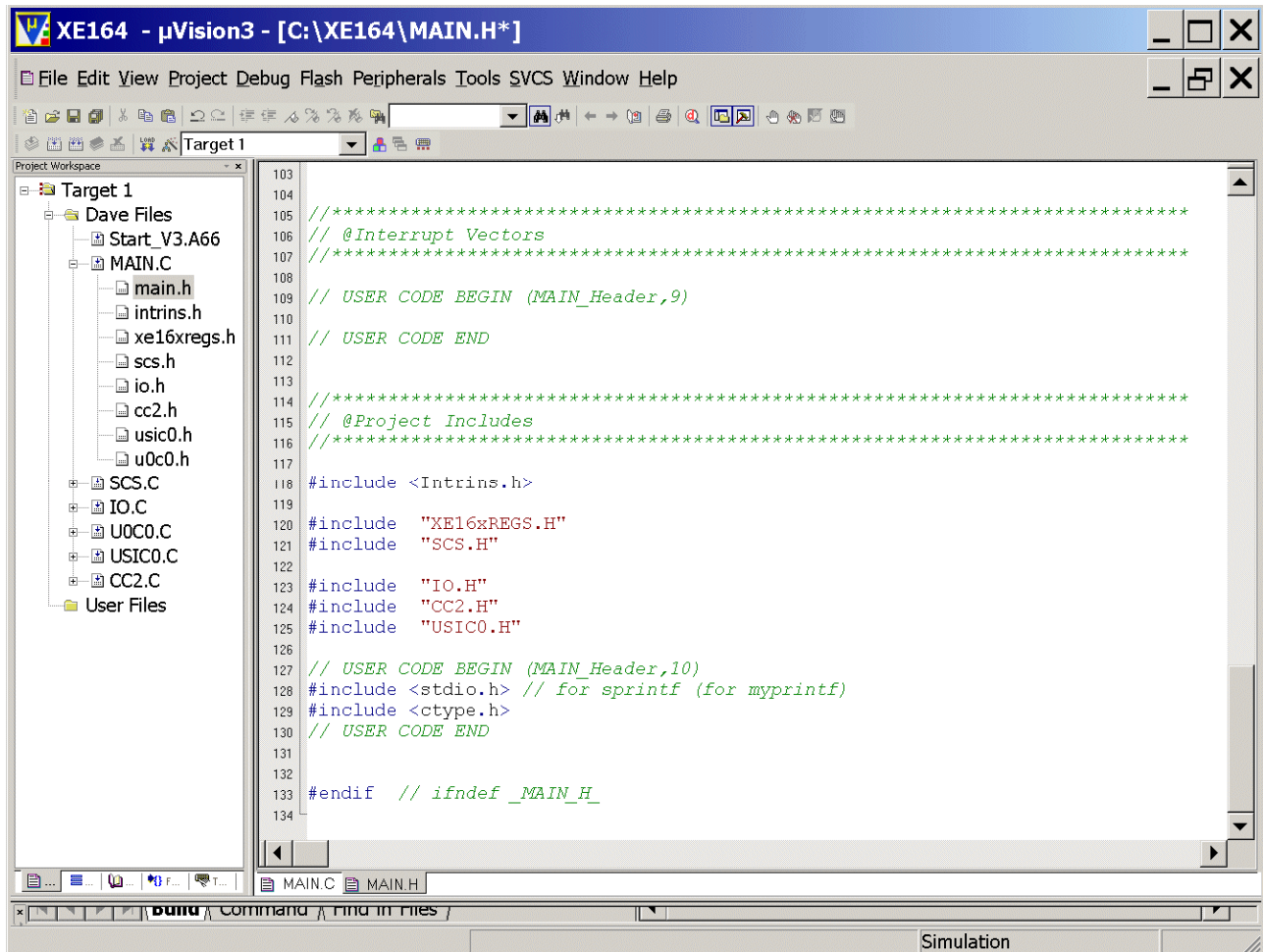
Double click **Main.h** and insert extern declarations "Global Functions":

```
extern void myprintf(const char *p);
```



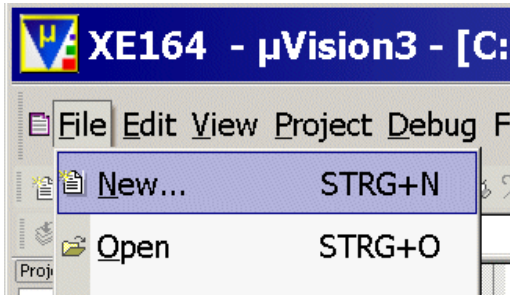
Double click **Main.h** and **insert** include files:

```
#include <stdio.h> // for sprintf (for myprintf)
#include <ctype.h>
```





## File – New



## Insert:

```
#include "main.h"

void myprintf(const char *p)
{
    while(*p)
    {
        U0C0_ASC_vSendData(*p++);
    }
}

/*

// Example 1 (use of myprintf):
// =====

void main(void)
{
    myprintf("Hello World!\r\n");
}

// Example 2 (use of myprintf):
// =====

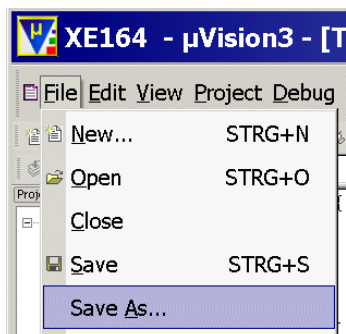
char mb[200]; // message buffer for sprintf()

void main(void)
{
    int dummy;

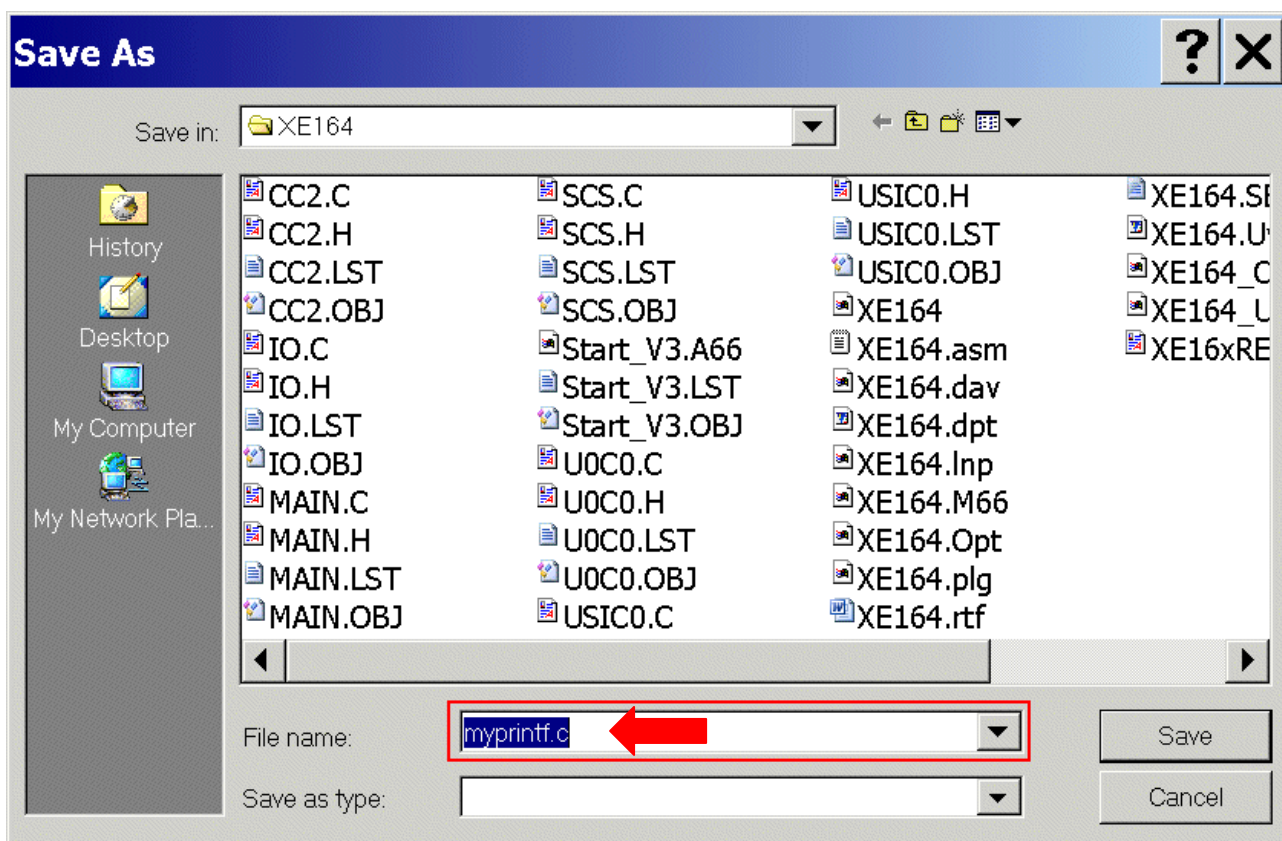
    sprintf(mb,"Variable dummy = %d",dummy); // Write formatted data to string mb
    myprintf(mb);
}

*/
```

File – Save As...

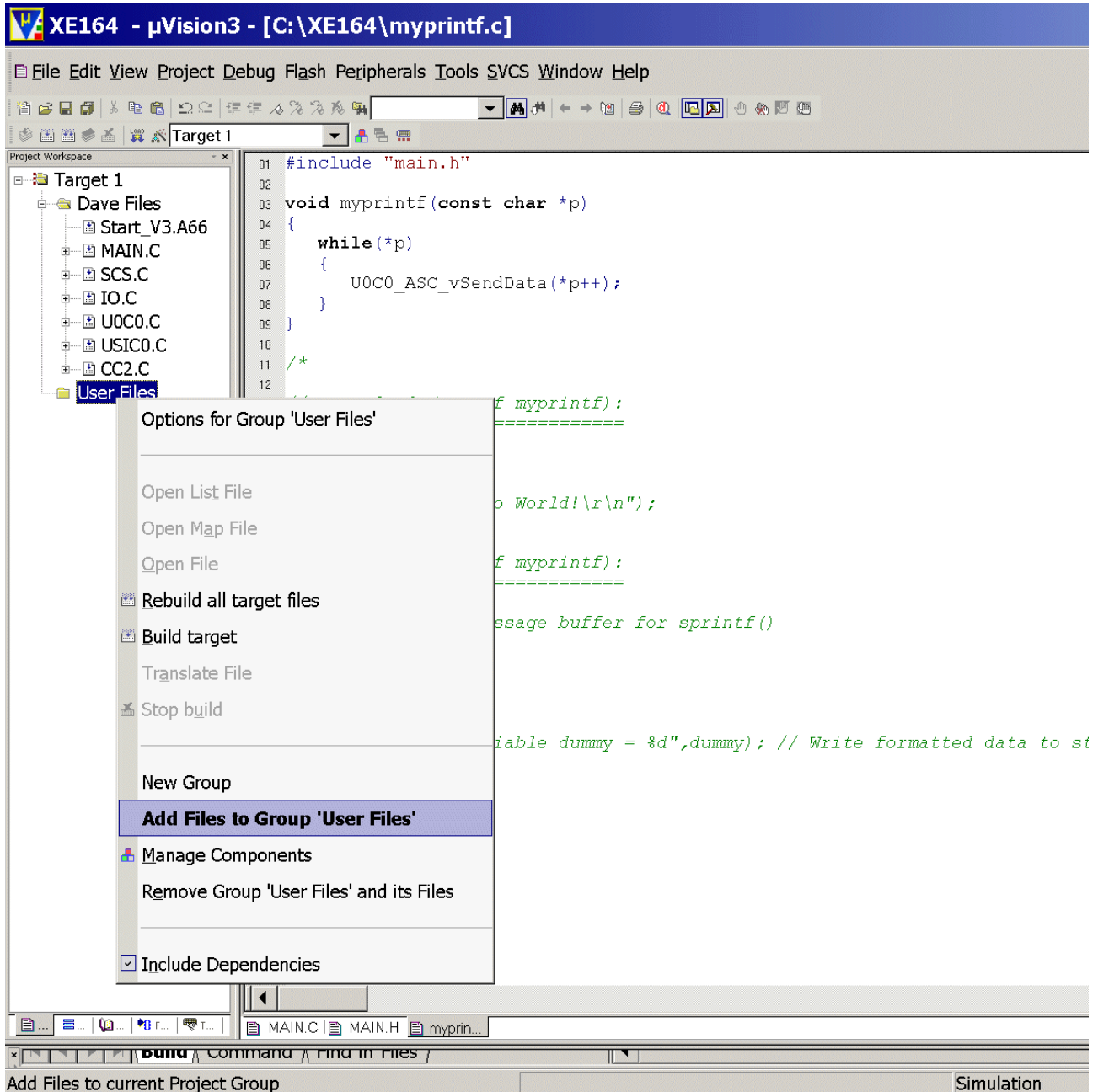


Insert: myprintf.c

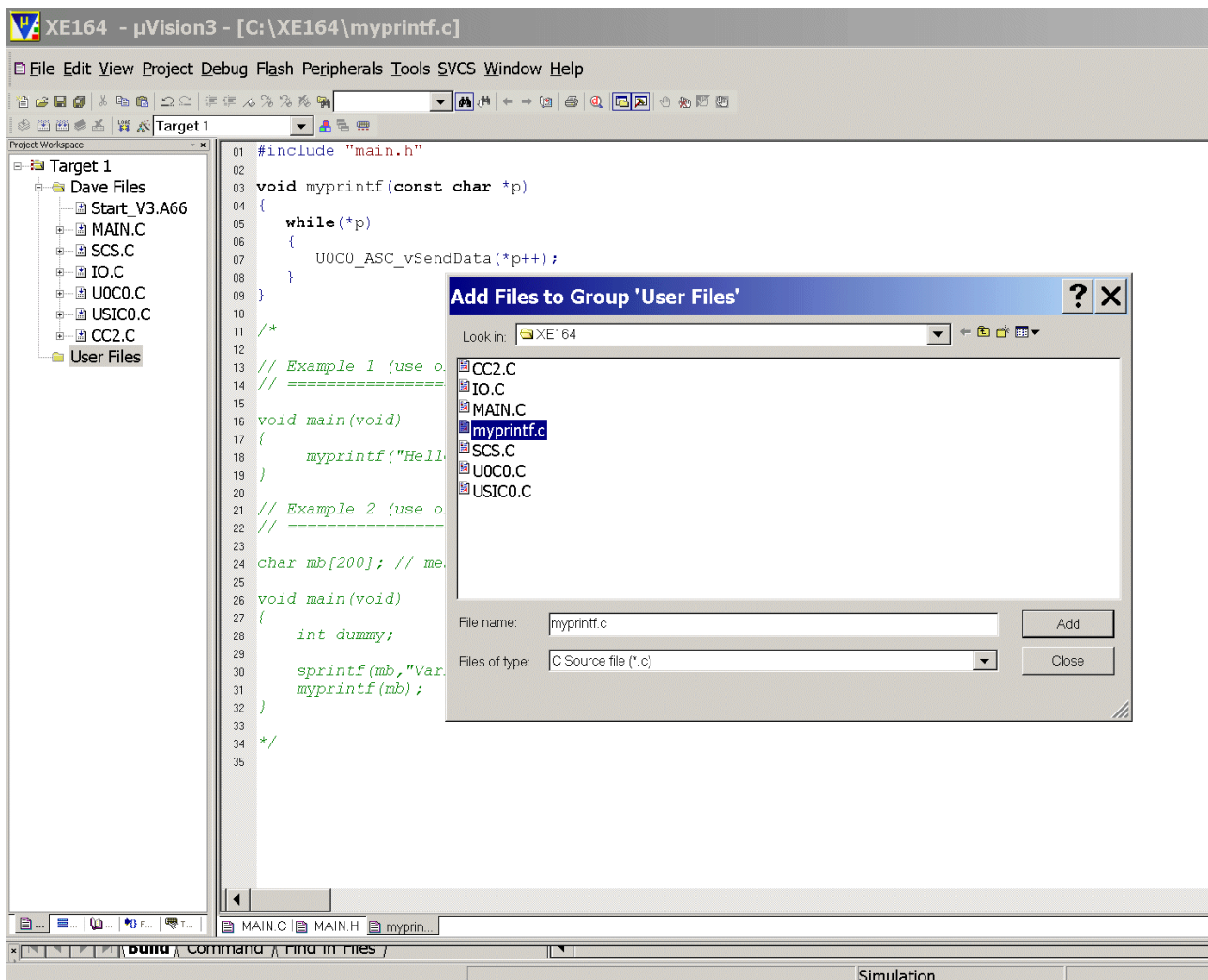


Save

Mouse position: **Project Window**, User Files: **click right mouse button**  
**click** Add Files to Group 'User Files'

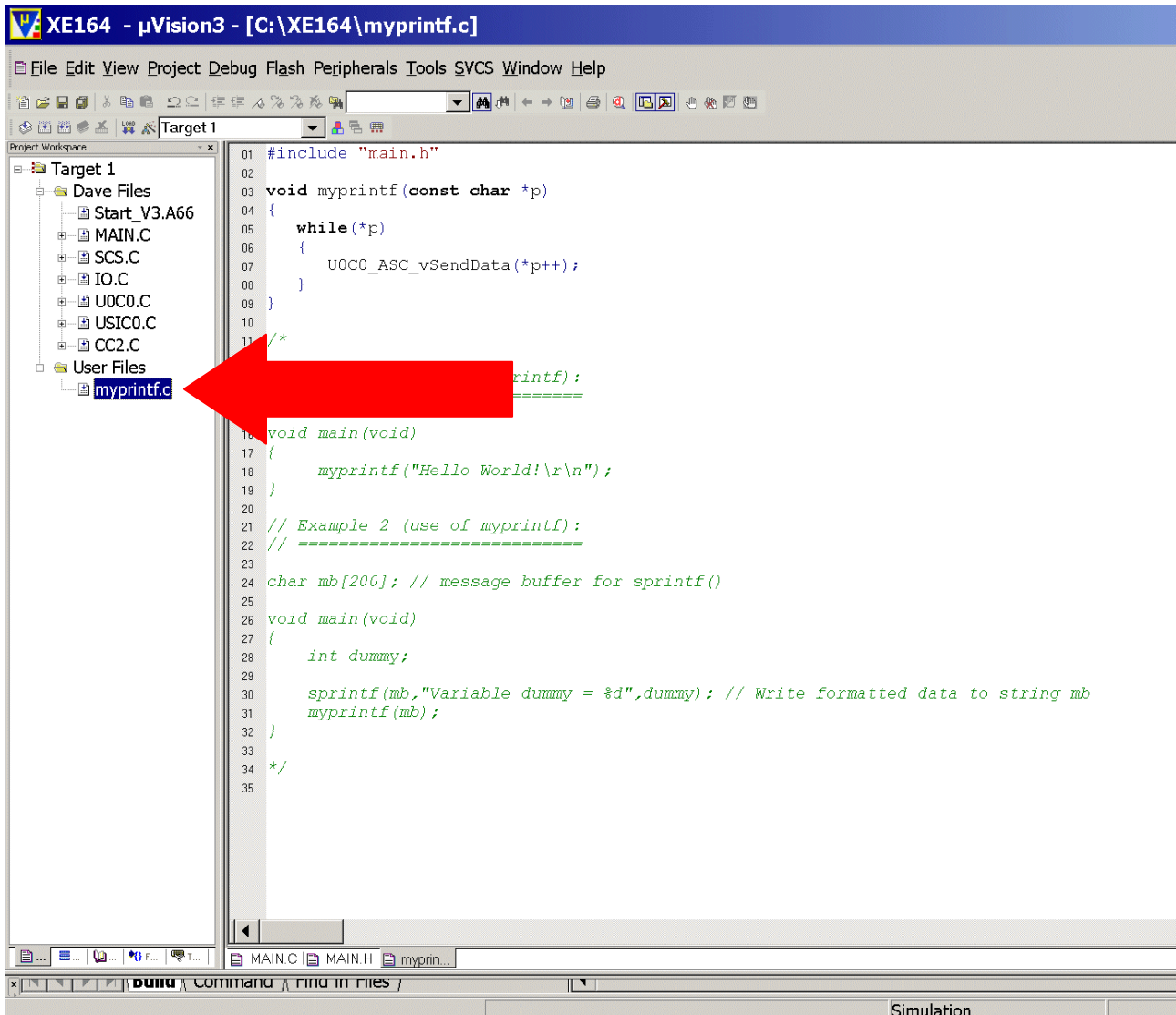


Click myprintf.c



Add  
Close

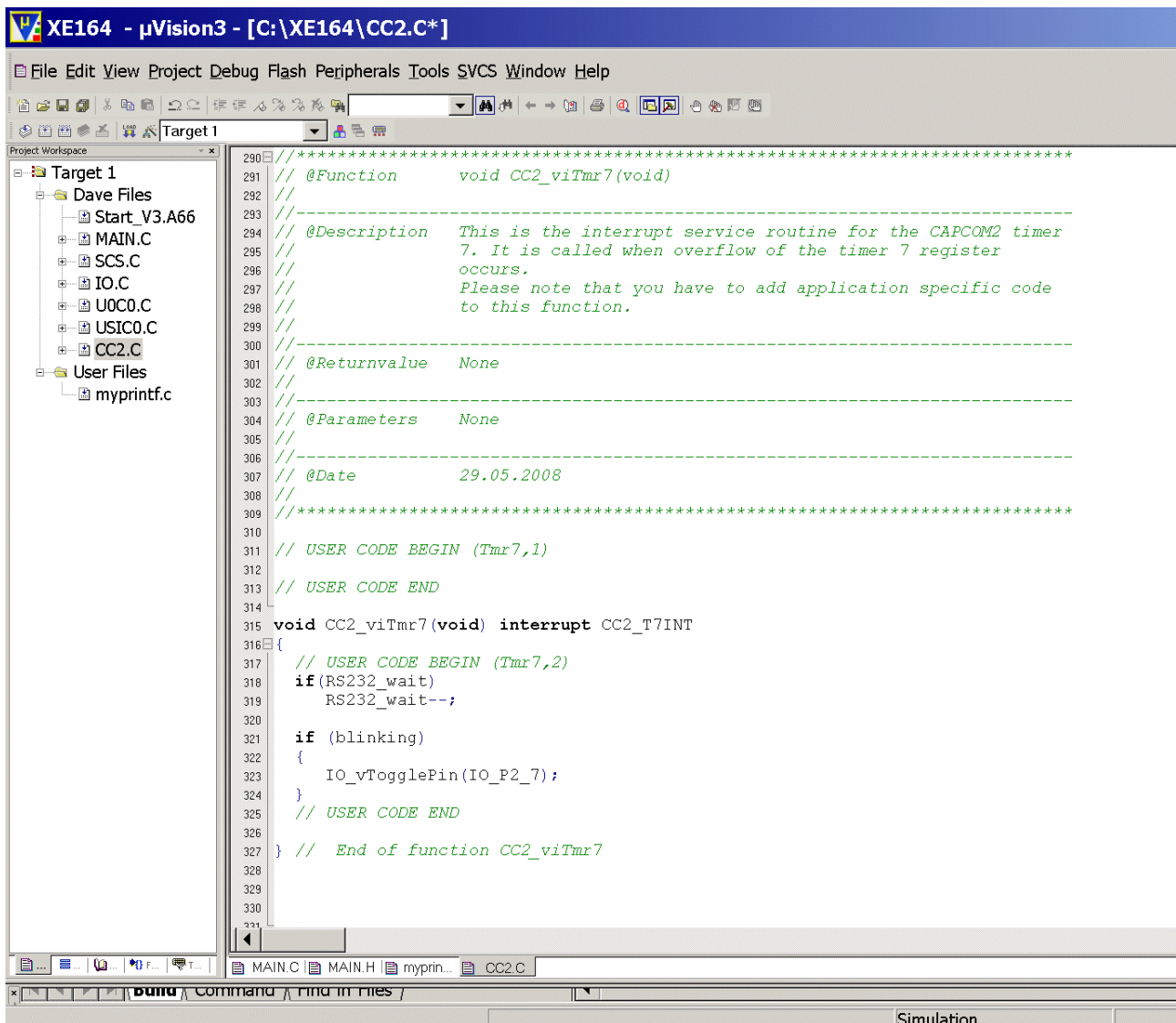




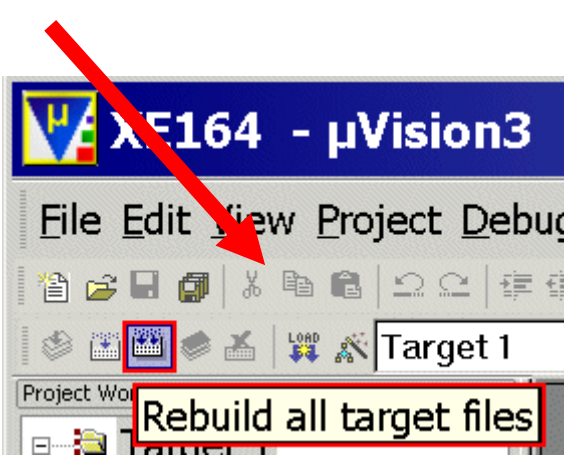
Double click **CC2.C** insert Code (CAPCOM 2 Timer 7 Interrupt Service Routine):

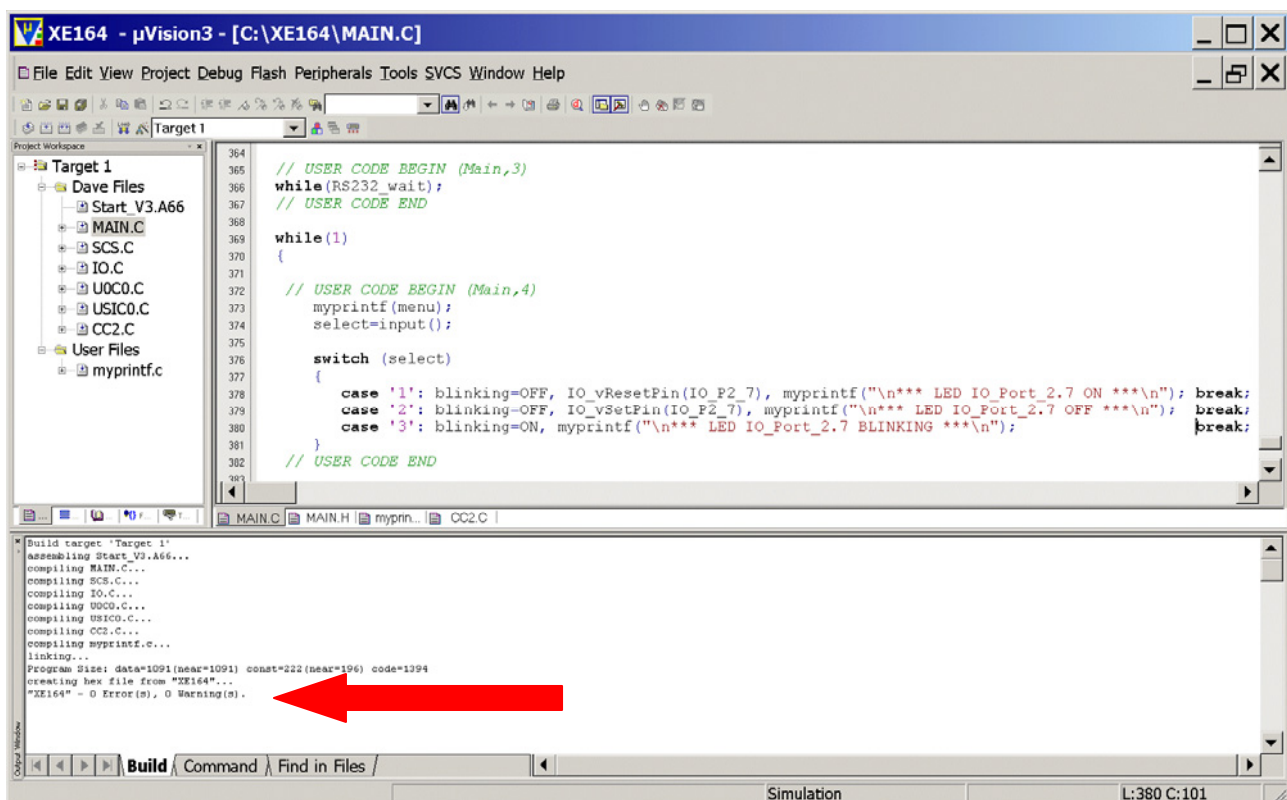
```
if(RS232_wait)
    RS232_wait--;

if (blinking)
{
    IO_vTogglePin(IO_P2_7);
}
```



Generate your application program:

<p>Project – Rebuild all target files</p>	<p>or</p>	<p>click</p> 
---	-----------	---





**Note:**

Programming is now complete.

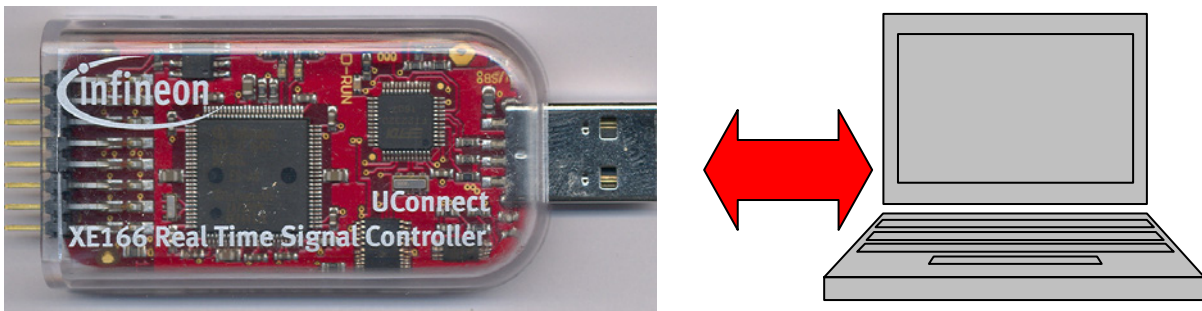
Unfortunately it is not possible to test your program with the Keil Simulator because this feature is currently not supported.

Therefore we are going to **load** (On Chip Flash Programming) and **run** your program on the UConnect-CAN XE164 in the next chapter.



### 5.) Running your first programming example:

Make sure that the UConnect-CAN XE164 is still connected to the host computer:



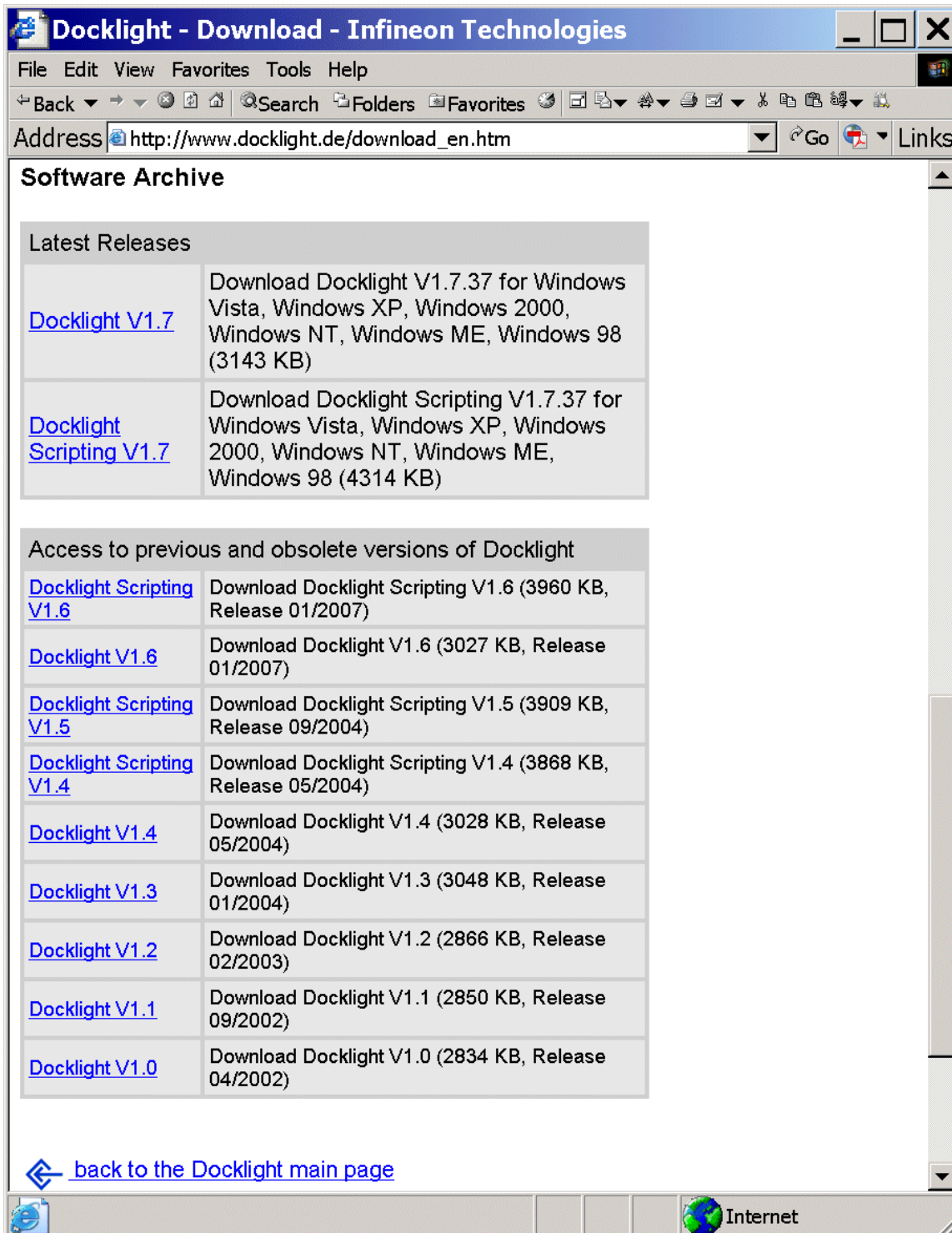
#### USB Connection:

- .) used for: UART communication (the USIC0\_CH0/UART/RS232/serial interface is available via USB as a virtual COM port of the second USB channel of the FTDI FT2232 Dual USB to UART/JTAG interface).
- .) used for: On-Chip-Flash-Programming and Debugging (first USB channel of the FTDI FT2232 Dual USB to UART/JTAG interface).
- .) the USB connection works also as the power supply.



**Note:**

Now we need a terminal program which is able to handle a virtual COM port (COM5)!  
As an example of “any terminal program” we are going to use Docklight.  
Docklight can be downloaded @ <http://www.docklight.de> :



The screenshot shows a web browser window titled "Docklight - Download - Infineon Technologies". The address bar shows the URL [http://www.docklight.de/download\\_en.htm](http://www.docklight.de/download_en.htm). The page content is titled "Software Archive" and lists the latest releases and previous versions of Docklight.

Latest Releases	
<a href="#">Docklight V1.7</a>	Download Docklight V1.7.37 for Windows Vista, Windows XP, Windows 2000, Windows NT, Windows ME, Windows 98 (3143 KB)
<a href="#">Docklight Scripting V1.7</a>	Download Docklight Scripting V1.7.37 for Windows Vista, Windows XP, Windows 2000, Windows NT, Windows ME, Windows 98 (4314 KB)

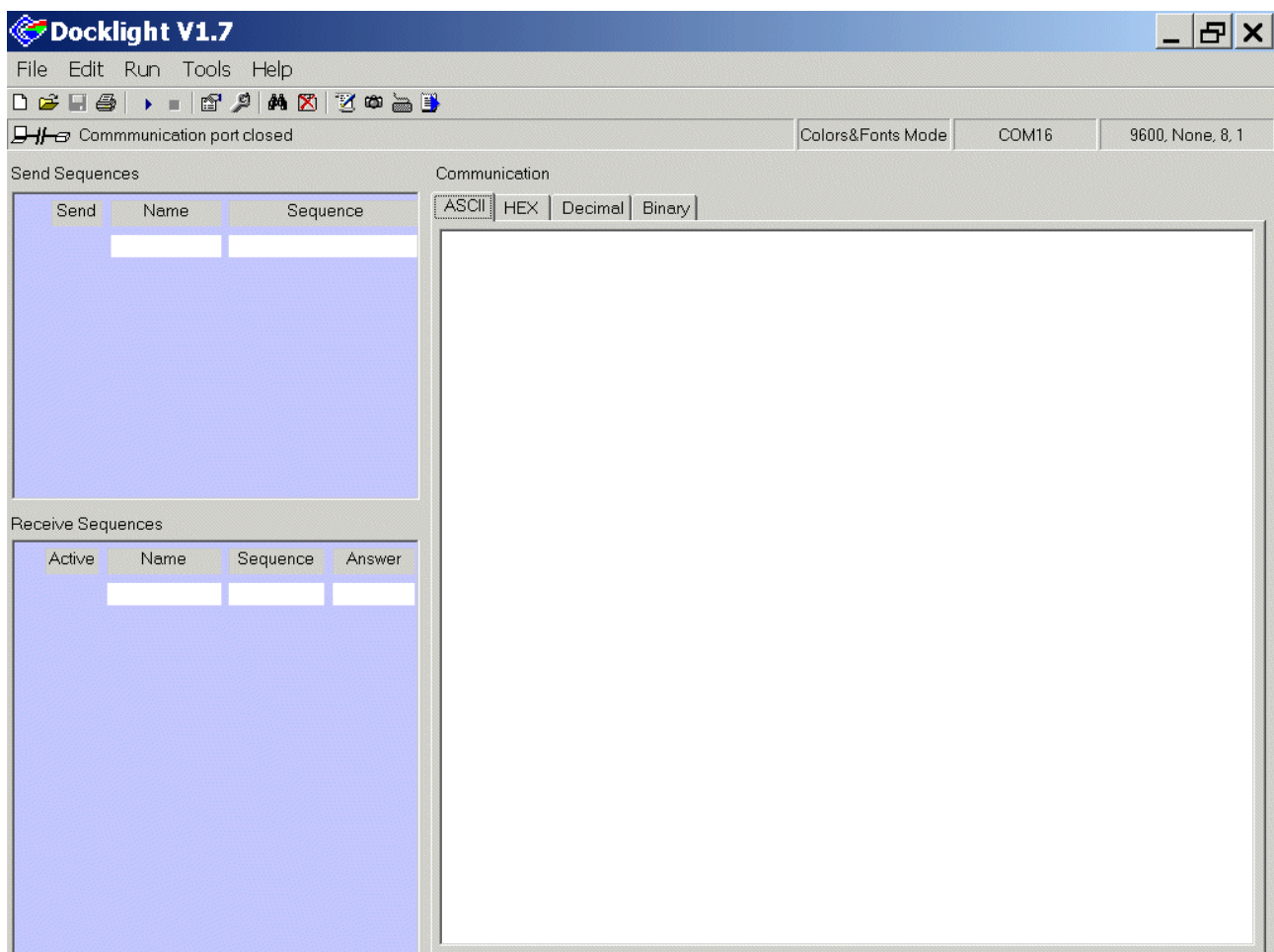
  

Access to previous and obsolete versions of Docklight	
<a href="#">Docklight Scripting V1.6</a>	Download Docklight Scripting V1.6 (3960 KB, Release 01/2007)
<a href="#">Docklight V1.6</a>	Download Docklight V1.6 (3027 KB, Release 01/2007)
<a href="#">Docklight Scripting V1.5</a>	Download Docklight Scripting V1.5 (3909 KB, Release 09/2004)
<a href="#">Docklight Scripting V1.4</a>	Download Docklight Scripting V1.4 (3868 KB, Release 05/2004)
<a href="#">Docklight V1.4</a>	Download Docklight V1.4 (3028 KB, Release 05/2004)
<a href="#">Docklight V1.3</a>	Download Docklight V1.3 (3048 KB, Release 01/2004)
<a href="#">Docklight V1.2</a>	Download Docklight V1.2 (2866 KB, Release 02/2003)
<a href="#">Docklight V1.1</a>	Download Docklight V1.1 (2850 KB, Release 09/2002)
<a href="#">Docklight V1.0</a>	Download Docklight V1.0 (2834 KB, Release 04/2002)

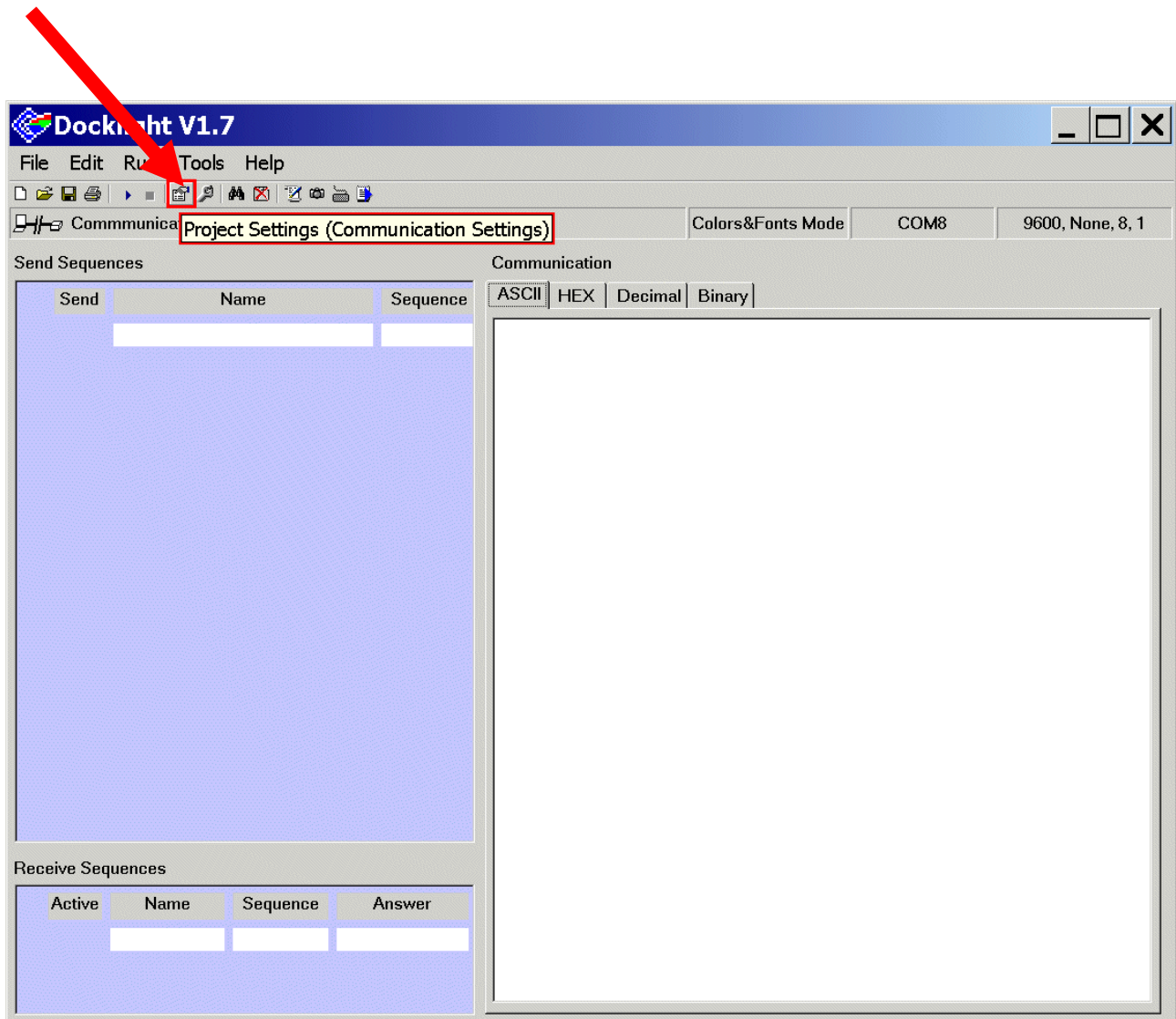
[back to the Docklight main page](#)



Now, **start** Docklight:



Click: Project Settings





Project Settings:

Communication: Communication Mode: **click** ☒ Send/Receive

Project Settings:

Communication: Communication Mode: Send/Receive on comm. channel: **select** COM5

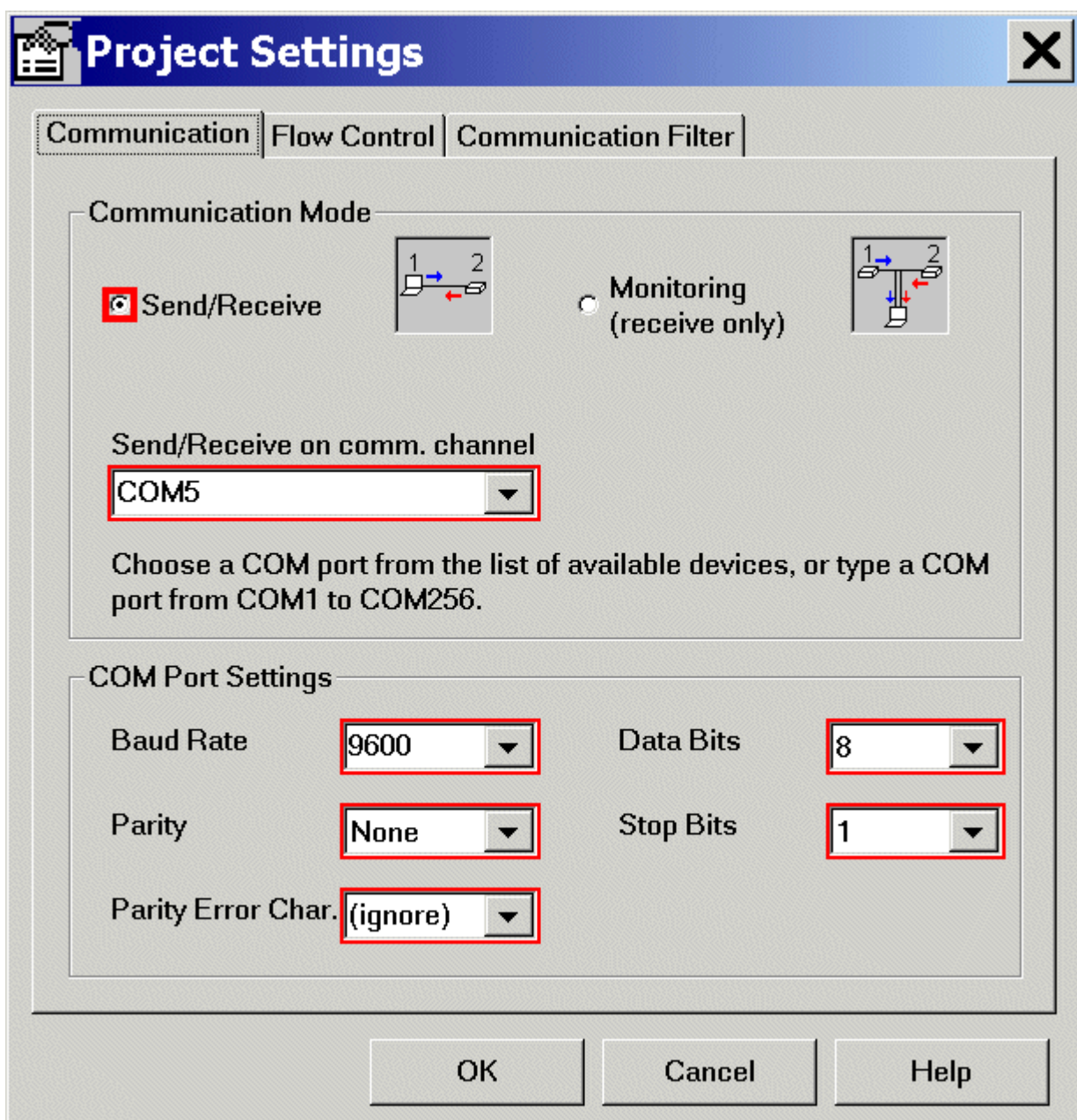
Project Settings: Communication: COM Port Settings: Baud Rate: **select** 9600

Project Settings: Communication: COM Port Settings: Parity: **select** None

Project Settings: Communication: COM Port Settings: Parity Error Char.: **select** (ignore)

Project Settings: Communication: COM Port Settings: Data Bits: **select** 8

Project Settings: Communication: COM Port Settings: Stop Bits: **select** 1



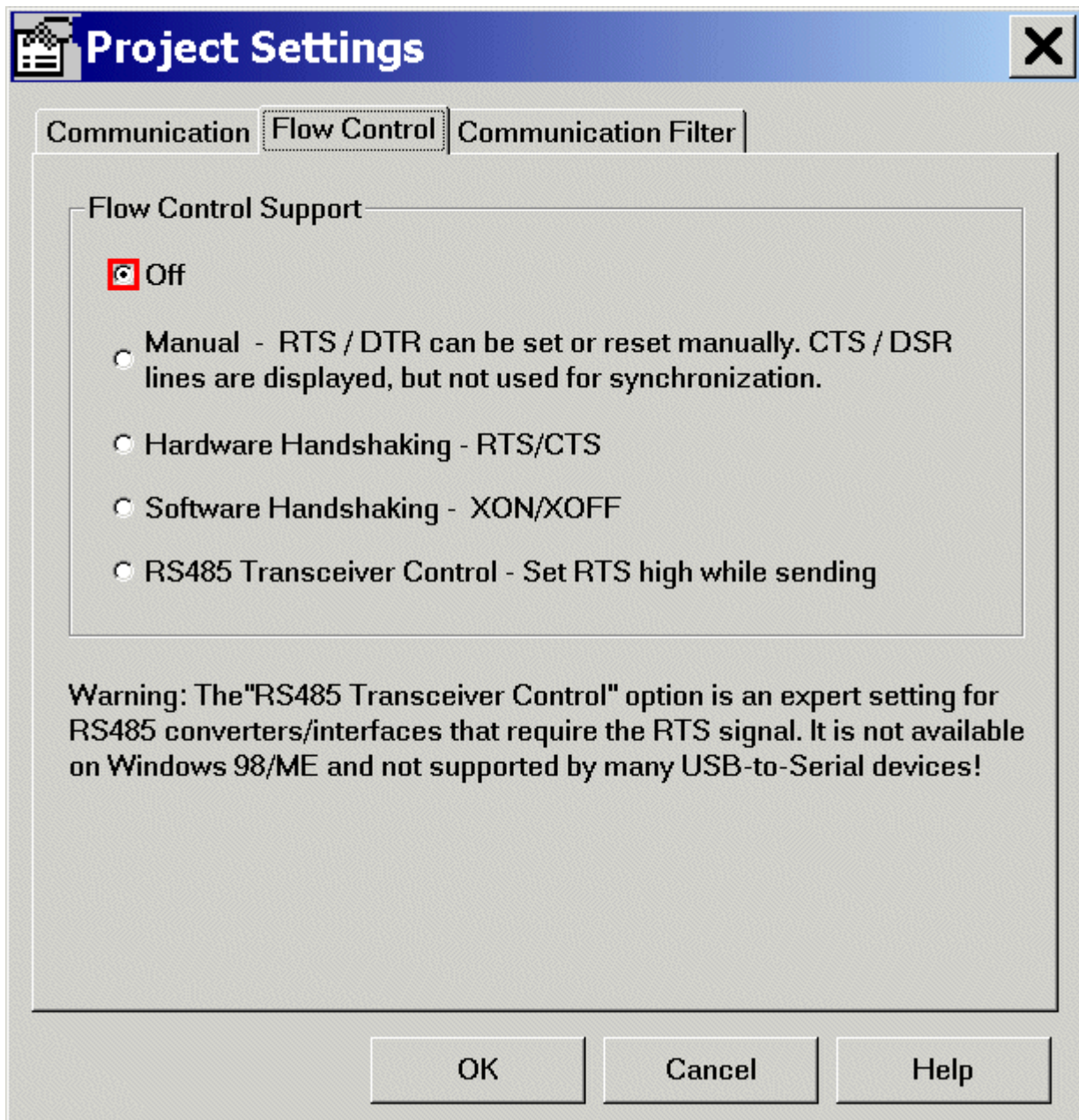
The image shows a 'Project Settings' dialog box with three tabs: 'Communication', 'Flow Control', and 'Communication Filter'. The 'Communication' tab is active. It contains two main sections: 'Communication Mode' and 'COM Port Settings'.

**Communication Mode:** This section has two options. The first is 'Send/Receive', which is selected with a checked radio button and accompanied by a diagram showing bidirectional communication between two devices. The second is 'Monitoring (receive only)', which is unselected with an unchecked radio button and accompanied by a diagram showing only reception from one device. Below these options is a dropdown menu labeled 'Send/Receive on comm. channel' with 'COM5' selected. A note below the dropdown states: 'Choose a COM port from the list of available devices, or type a COM port from COM1 to COM256.'

**COM Port Settings:** This section contains five dropdown menus, each with a red border: 'Baud Rate' is set to 9600, 'Data Bits' is set to 8, 'Parity' is set to None, 'Stop Bits' is set to 1, and 'Parity Error Char.' is set to (ignore).

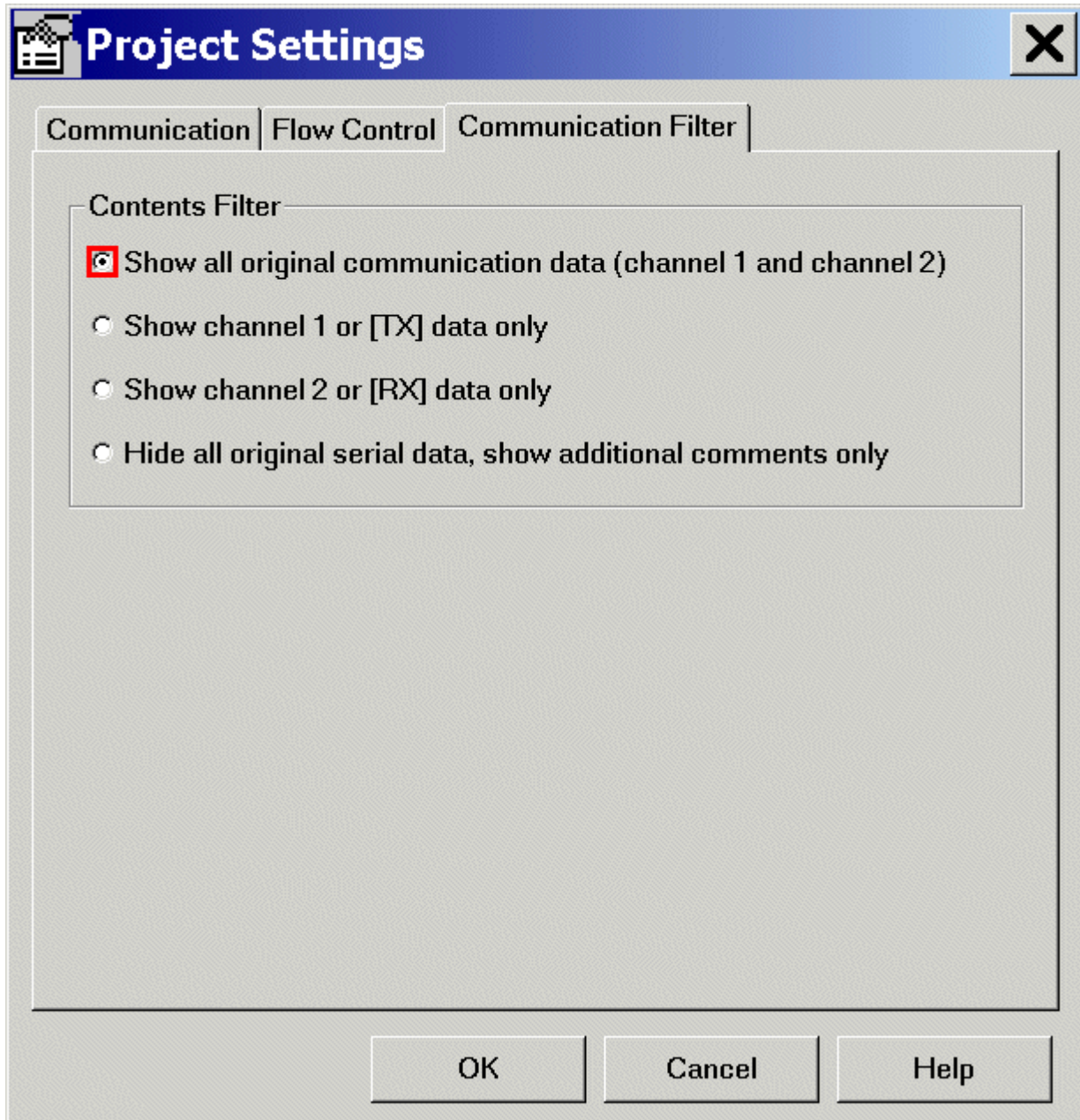
At the bottom of the dialog box are three buttons: 'OK', 'Cancel', and 'Help'.

Project Settings: Flow Control: Flow Control Support: click ☒ Off



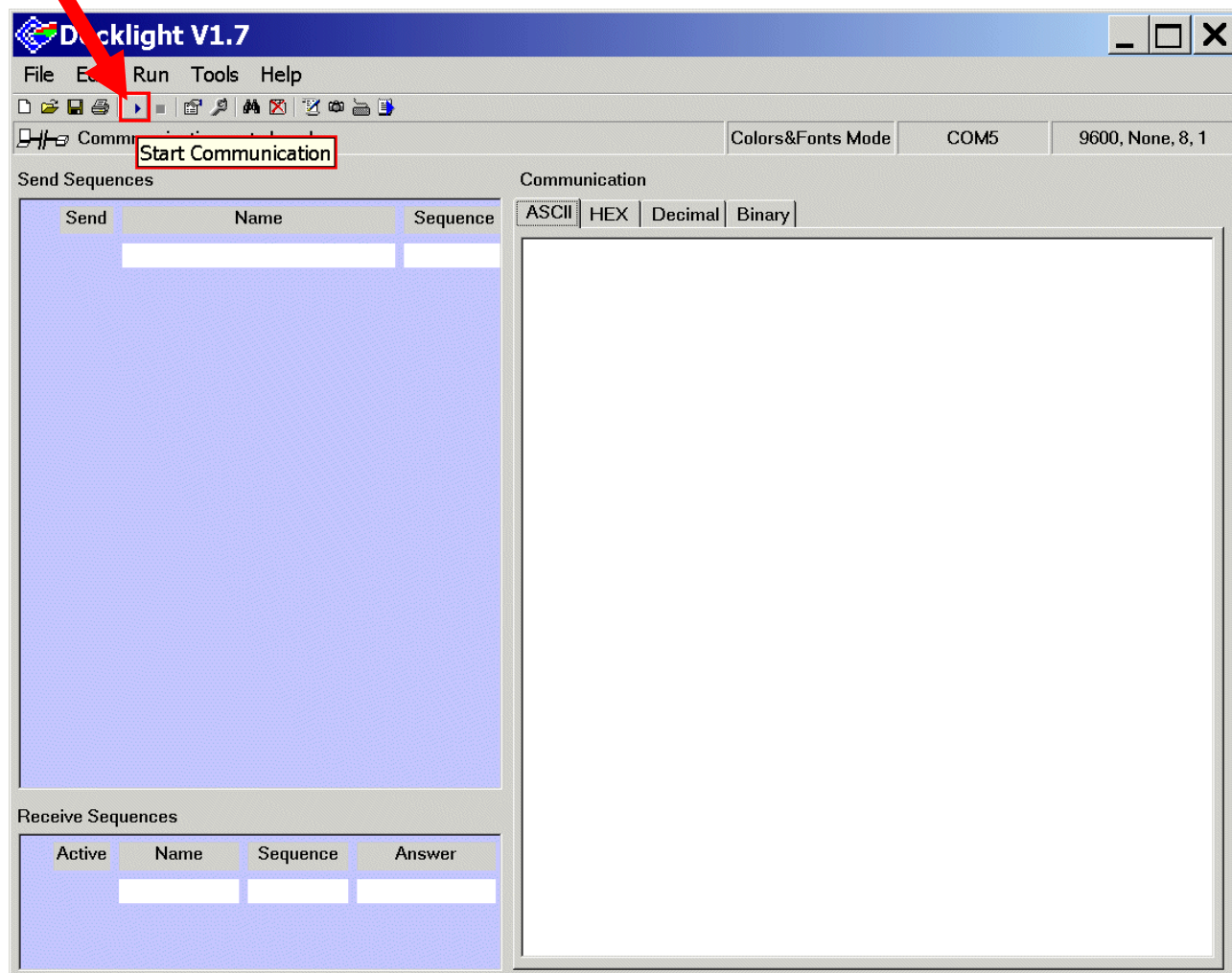
Project Settings:

Communication Filter: Contents Filter: **click** ☒ Show all original communication data

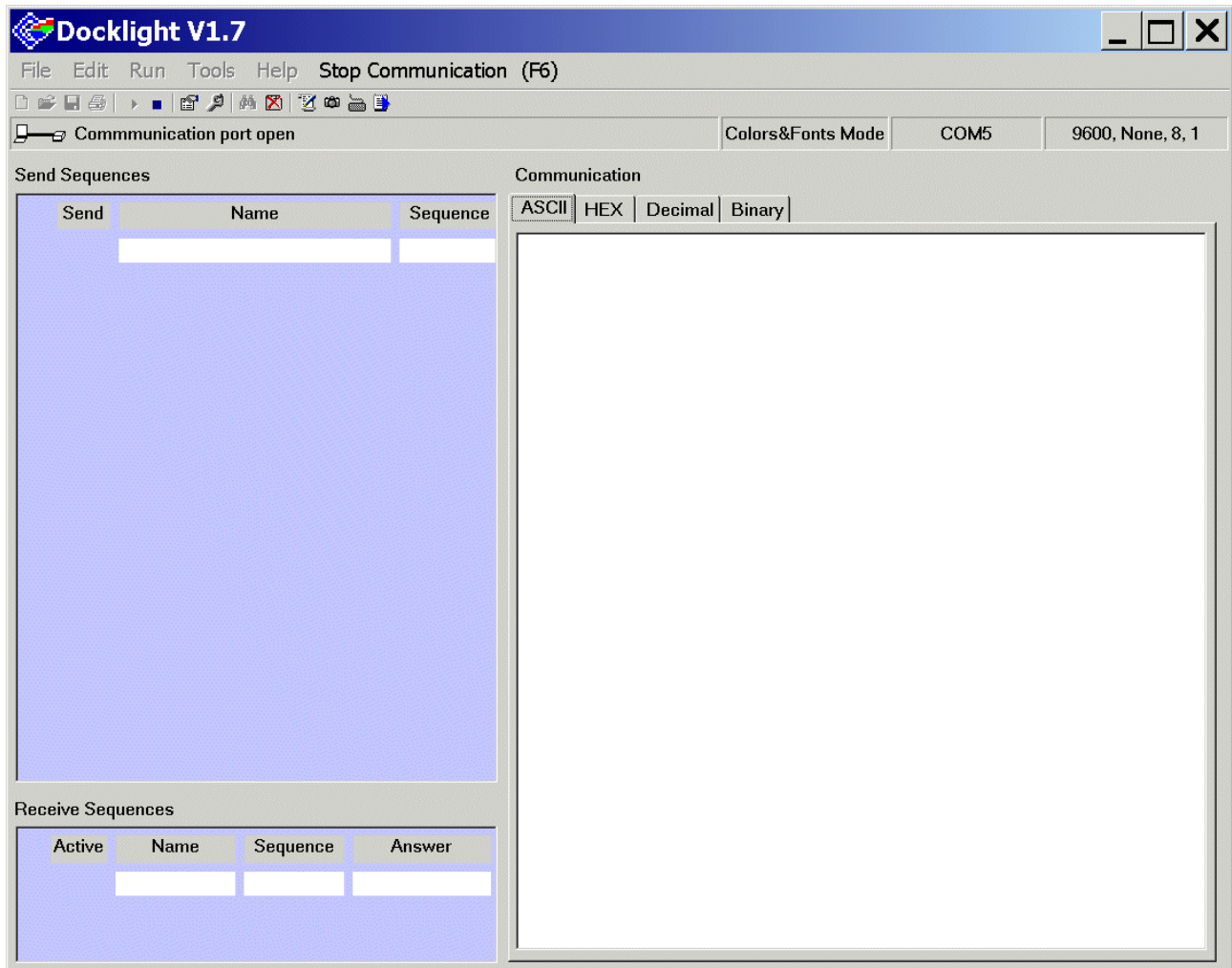


OK

Click: 









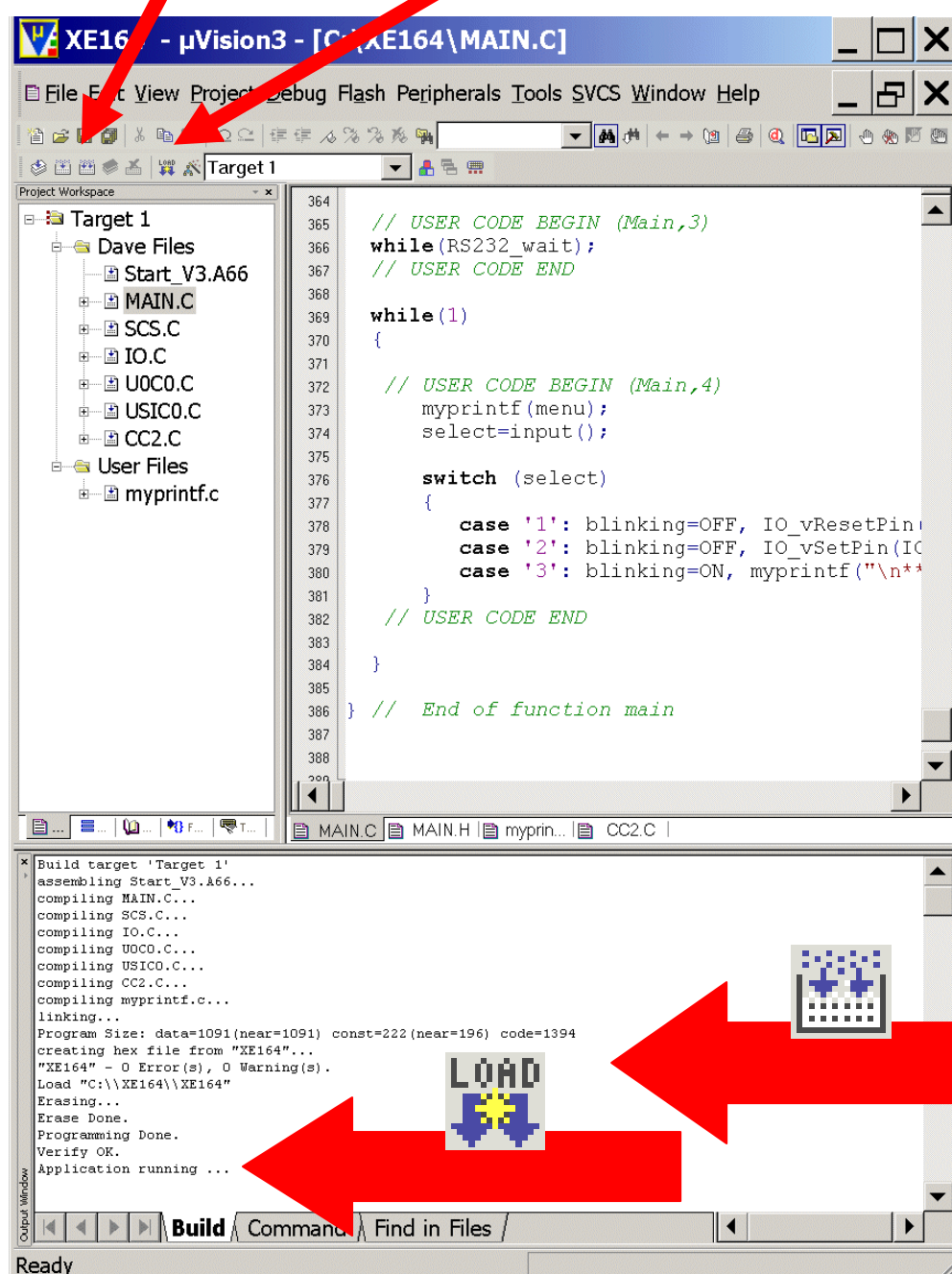
**Note:**  
Docklight is now ready for serial communication!





Go to  $\mu$ Vision:

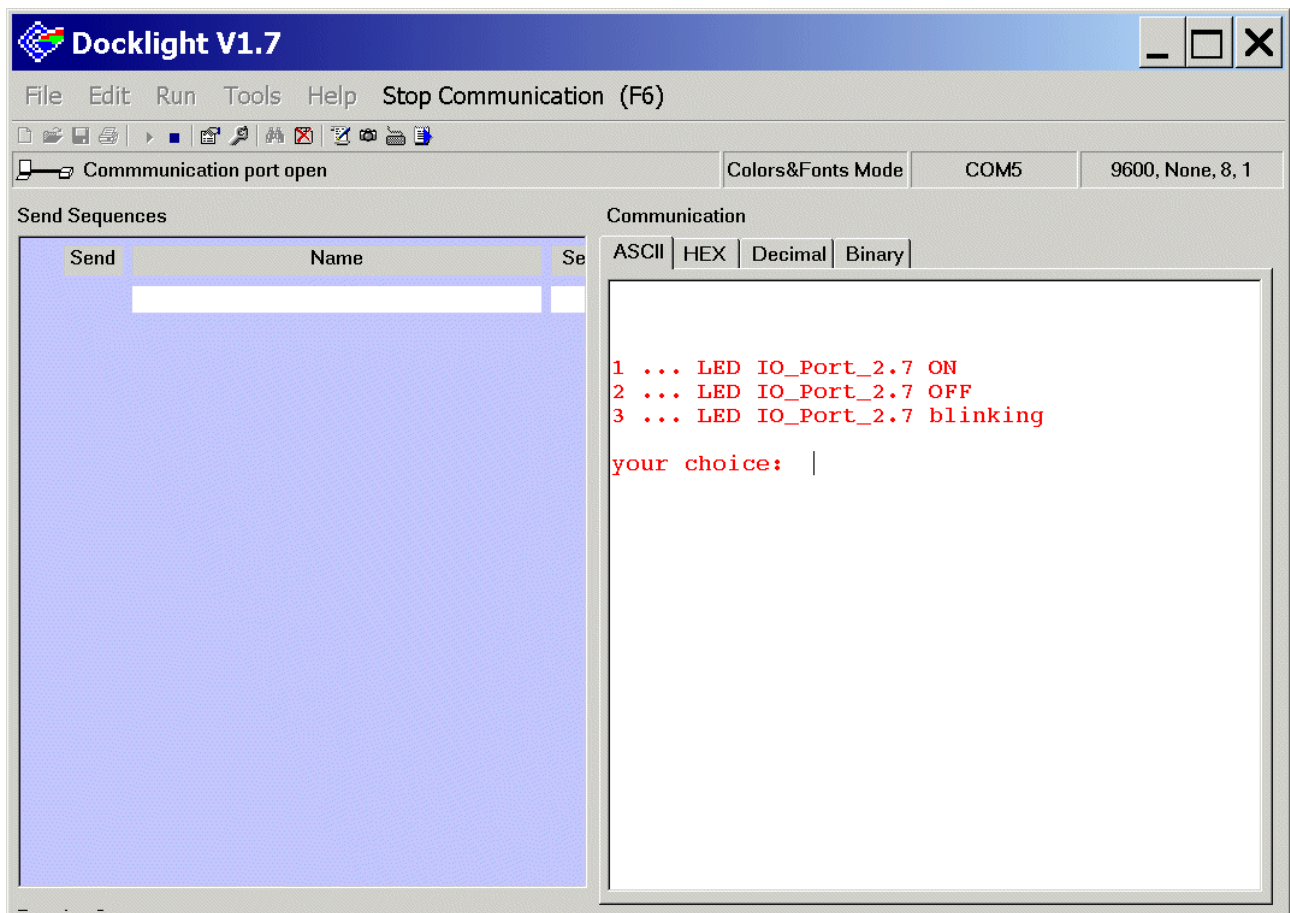
1.) click:  2.) click: 



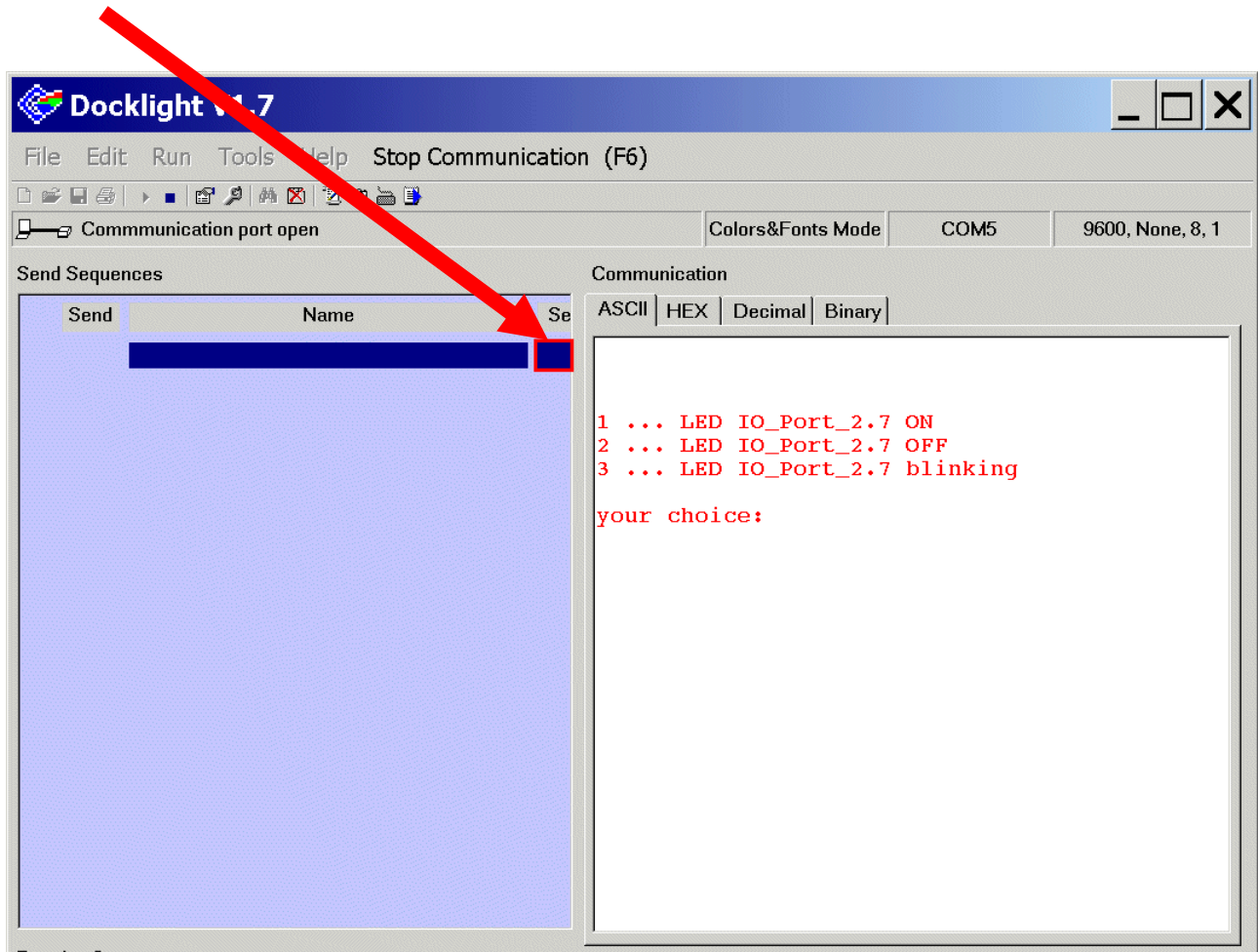
The screenshot shows the µVision3 IDE interface. The top menu bar includes File, Edit, View, Project, Debug, Flash, Peripherals, Tools, SVCS, Window, and Help. The Project Workspace on the left shows a project named 'Target 1' with a folder structure including 'Dave Files' (Start\_V3.A66, MAIN.C, SCS.C, IO.C, UOC0.C, USIC0.C, CC2.C) and 'User Files' (myprintf.c). The main editor window displays the source code for MAIN.C, showing a while loop and a switch statement. The bottom Output Window shows the build process, including compiling and linking files, and the final application running. Red arrows indicate the sequence of actions: clicking the 'Load' button in the toolbar and then clicking the 'LOAD' button in the output window.



Go to Docklight and see the result:



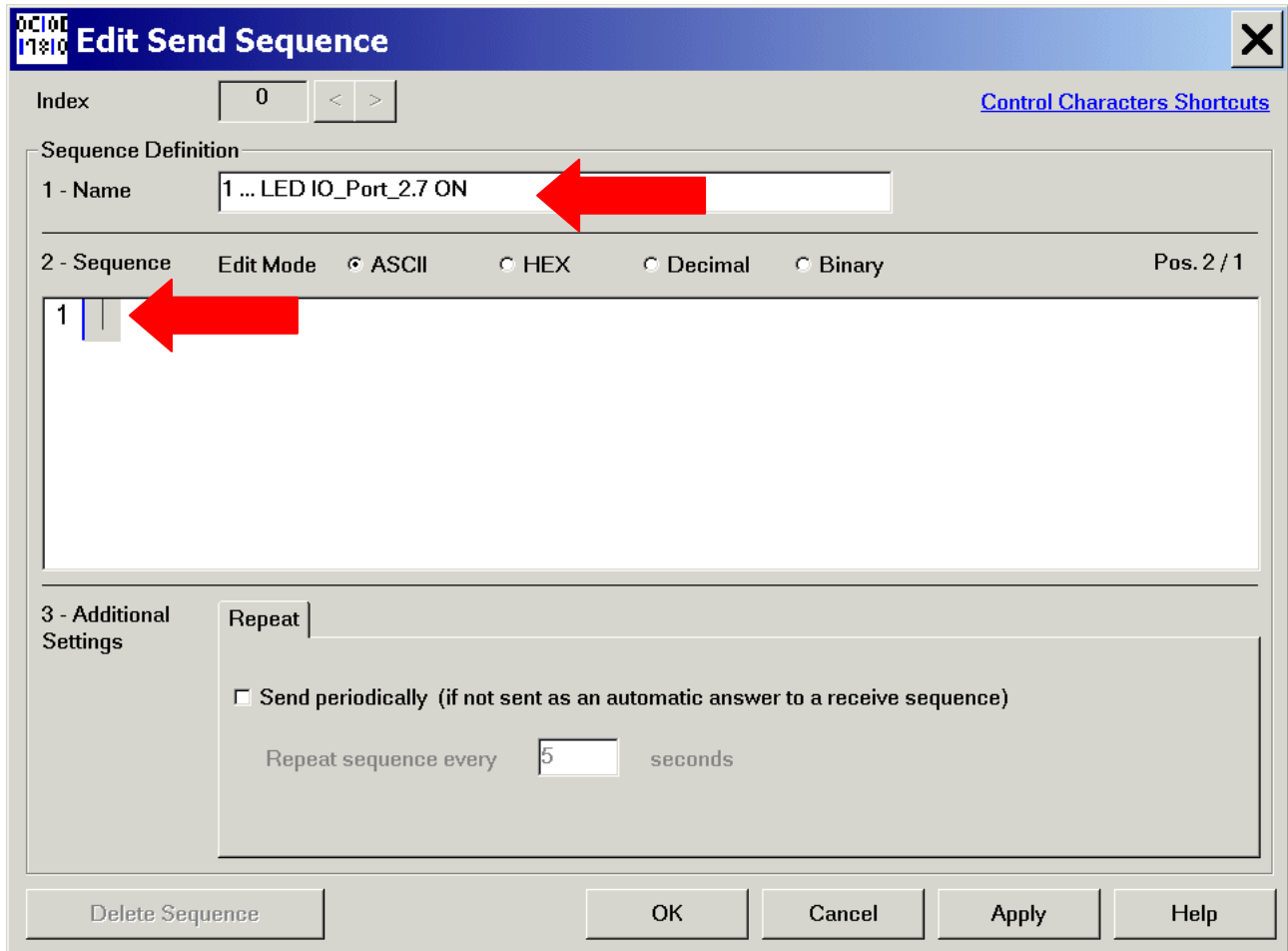
Double click inside the red box:





Edit Send Sequence: Sequence Definition: 1- Name: insert: 1 ... LED IO\_Port\_2.7 ON

Edit Send Sequence: Sequence Definition: 2- Sequence: insert: 1



**Edit Send Sequence**

Index: 0

[Control Characters Shortcuts](#)

**Sequence Definition**

1 - Name: 1 ... LED IO\_Port\_2.7 ON

2 - Sequence: Edit Mode: ☒ ASCII ☐ HEX ☐ Decimal ☐ Binary Pos. 2 / 1

1

3 - Additional Settings

Repeat

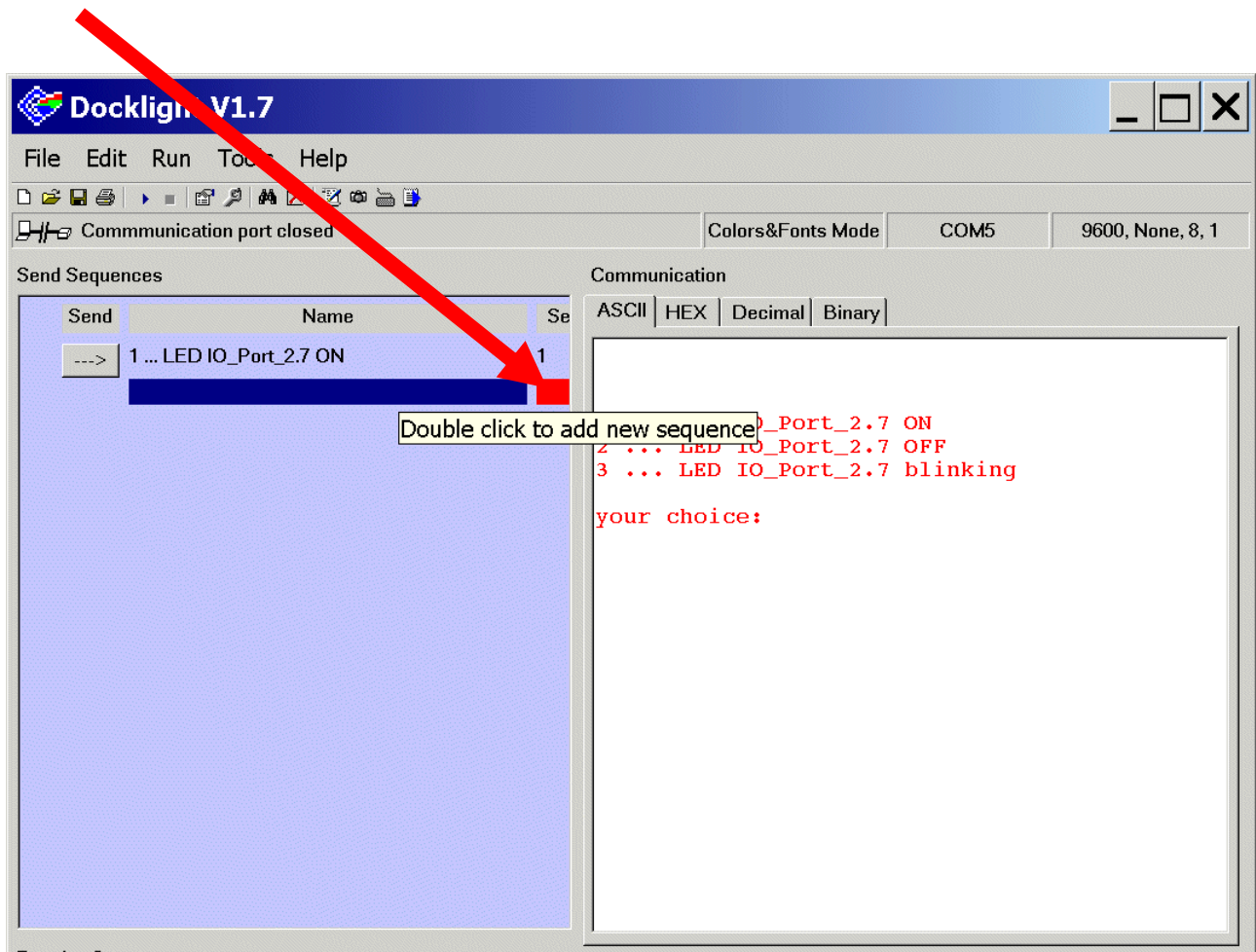
☐ Send periodically (if not sent as an automatic answer to a receive sequence)

Repeat sequence every 5 seconds

Delete Sequence OK Cancel Apply Help

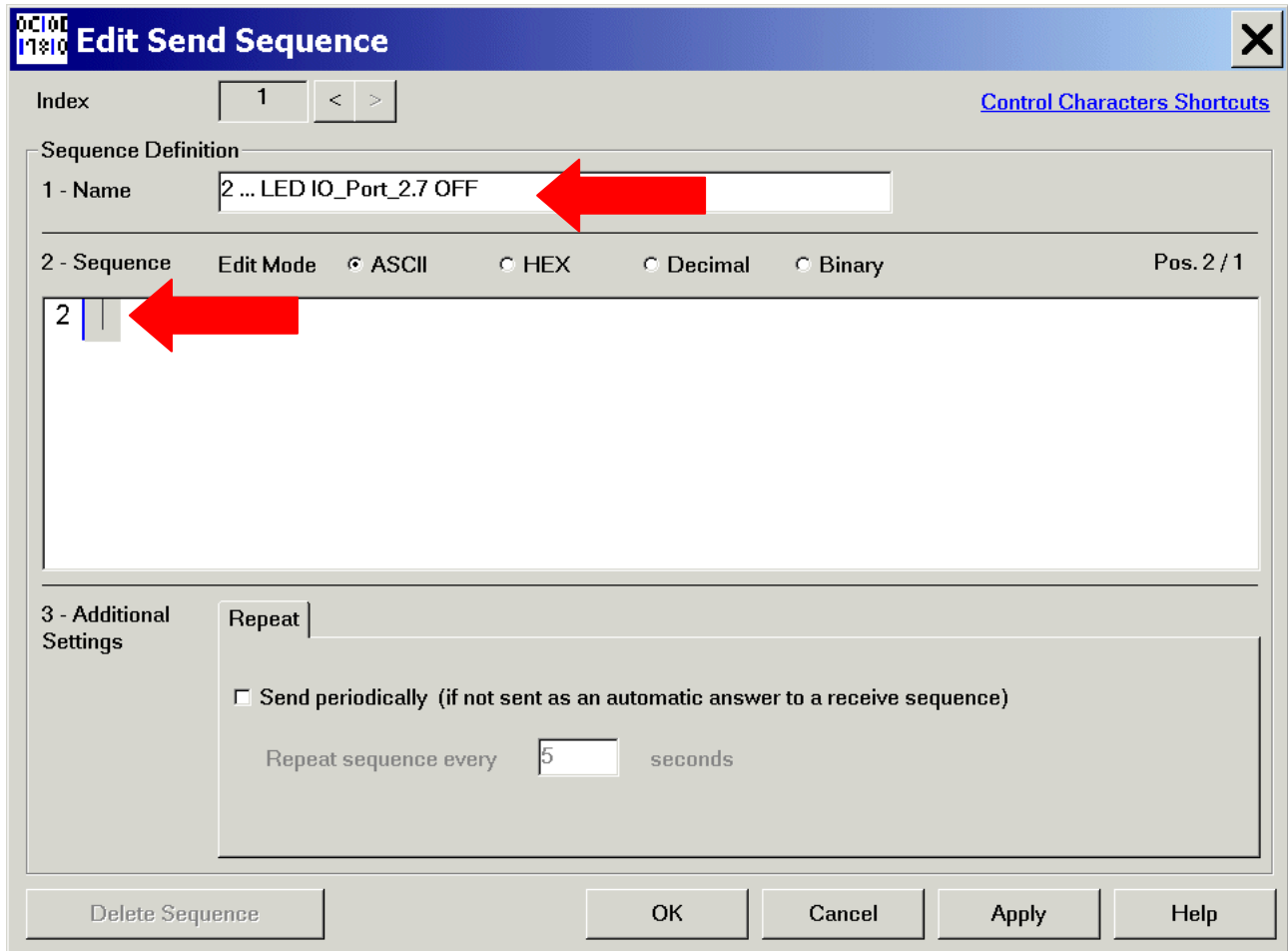
OK

Double click inside the red box:



Edit Send Sequence: Sequence Definition: 1- Name: insert: 2 ... LED IO\_Port\_2.7 OFF

Edit Send Sequence: Sequence Definition: 2- Sequence: insert: 2



**Edit Send Sequence**

Index: 1 < >

[Control Characters Shortcuts](#)

**Sequence Definition**

1 - Name: 2 ... LED IO\_Port\_2.7 OFF

2 - Sequence: Edit Mode: ☒ ASCII ☐ HEX ☐ Decimal ☐ Binary Pos. 2 / 1

2

3 - Additional Settings

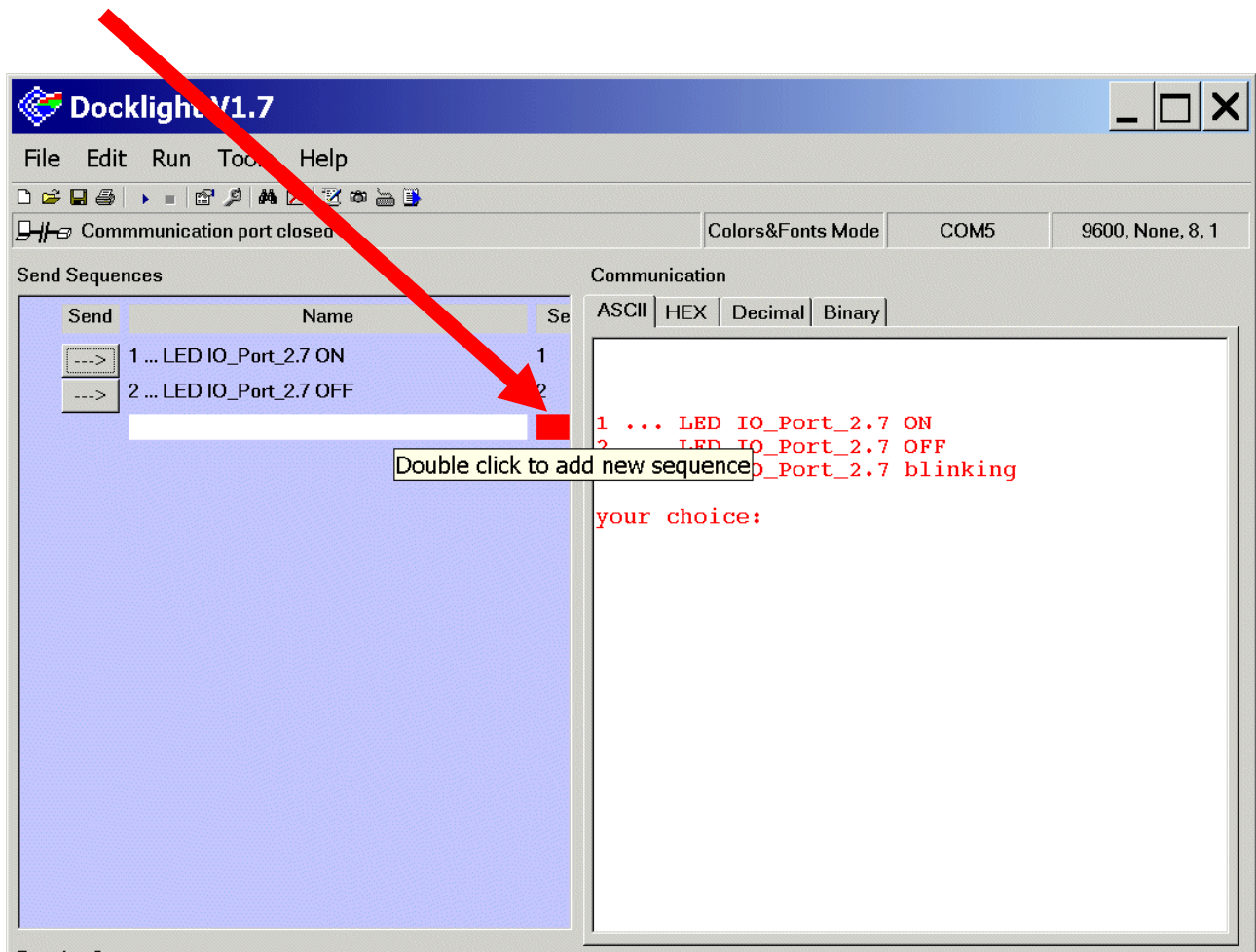
Repeat ☐ Send periodically (if not sent as an automatic answer to a receive sequence)

Repeat sequence every 5 seconds

Delete Sequence OK Cancel Apply Help

OK

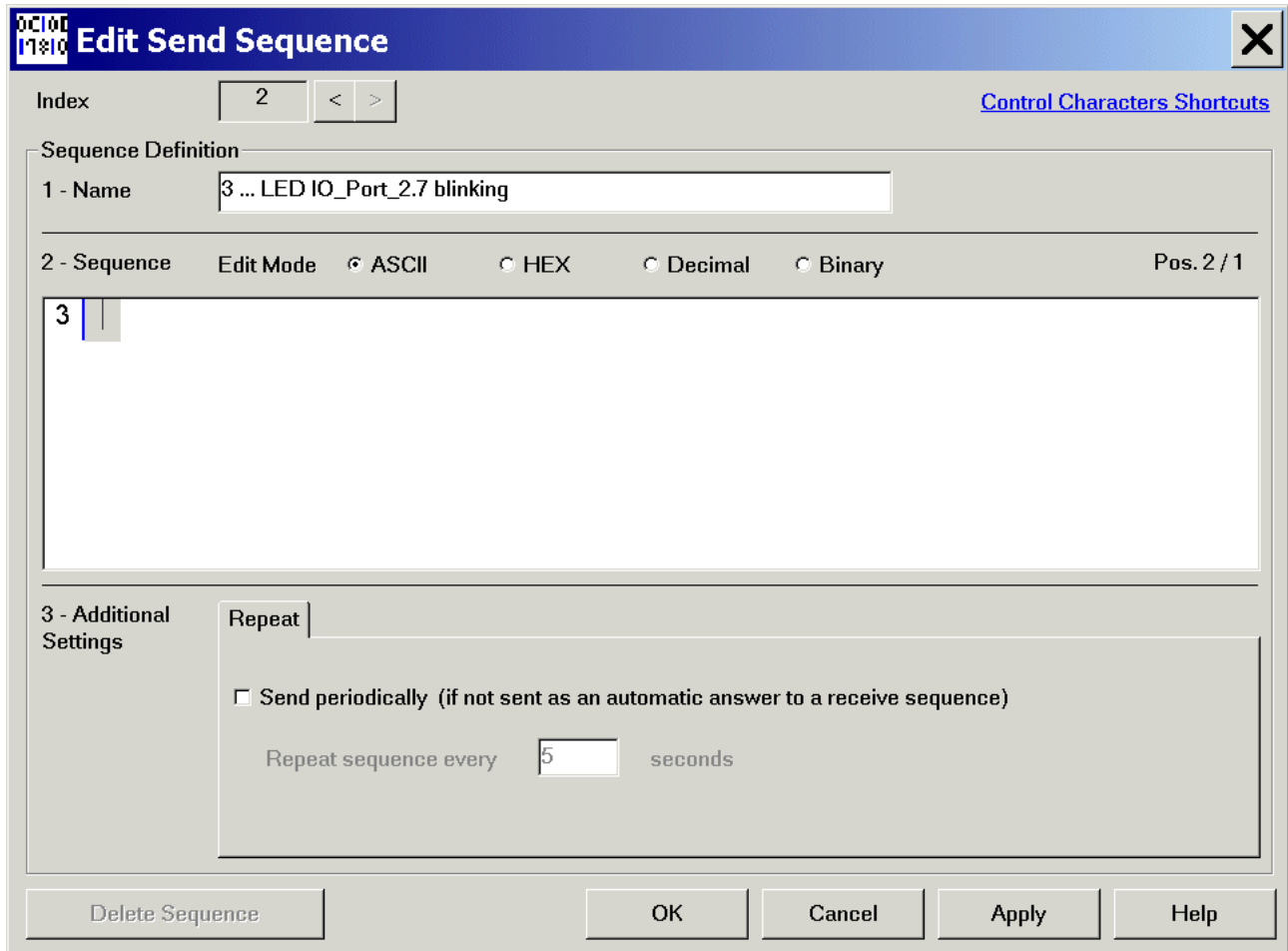
Double click inside the red box:





Edit Send Sequence: Sequence Definition: 1- Name: insert: 3 ... LED IO\_Port\_2.7 blinking

Edit Send Sequence: Sequence Definition: 2- Sequence: insert: 3



**Edit Send Sequence**

Index: 2 < > [Control Characters Shortcuts](#)

**Sequence Definition**

1 - Name: 3 ... LED IO\_Port\_2.7 blinking

2 - Sequence: Edit Mode: ☒ ASCII ☐ HEX ☐ Decimal ☐ Binary Pos. 2 / 1




3 - Additional Settings

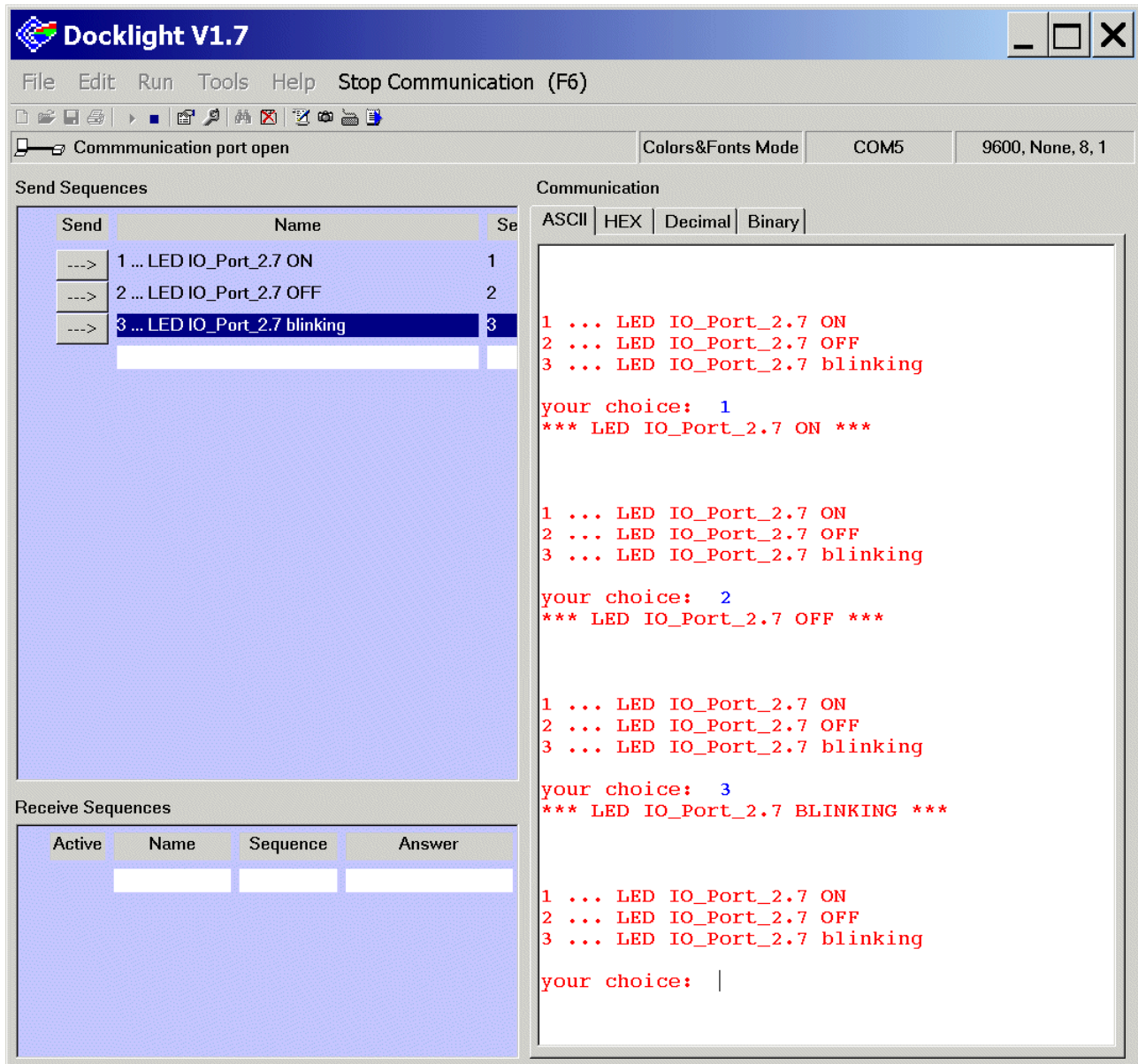
Repeat: ☐ Send periodically (if not sent as an automatic answer to a receive sequence)

Repeat sequence every: 5 seconds

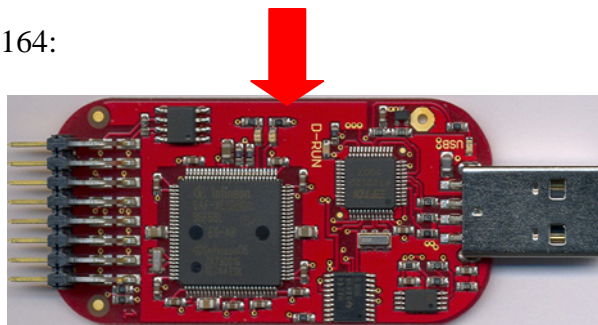
Buttons: Delete Sequence, OK, Cancel, Apply, Help

OK

- Click  1 ... LED IO\_Port\_2.7 ON or  
 Click  2 ... LED IO\_Port\_2.7 OFF or  
 Click  3 ... LED IO\_Port\_2.7 blinking



and **check** the results on your UConnect-CAN XE164:



Now we close our project and  $\mu$ Vision 3:

Project - Close Project

File  
Exit



### Conclusion:

In this step-by-step book you have learned how to use the UConnect-CAN XE164 together with the Keil tool chain.

Now you can easily expand your "hello world" program to suit your needs!

You can connect either a part of - or your entire application to the UConnect-CAN XE164.

You are also able to benchmark any of your algorithms to find out if the selected microcontroller fulfils all the required functions within the time frame needed.

Have fun and enjoy working with XE16x microcontrollers!

### Note:

There are step-by-step books for 8 bit microcontrollers (e.g. XC866 and XC888), 16 bit microcontrollers (e.g. C16x, XC16x and XE16x/XC2xxx) and 32 bit microcontrollers (e.g. TC1796 and TC1130).

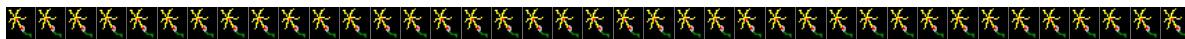
All these step-by-step books use the same microcontroller resources and the same example code.

This means: configuration steps, function names and variable names are identical.

This should give you a good opportunity to get in touch with another Infineon microcontroller family or tool chain!

There are even more programming examples using the same style available [e.g. ADC-examples, CAPCOM6-examples (e.g. BLDC-Motor, playing music), Simulator-examples, C++ examples] based on these step-by-step books.

**6.) Feedback (UConnect-CAN XE164, Keil tools):**  
**Your opinion, suggestions and/or criticisms**



**Contact Details (this section may remain blank should you wish to offer feedback anonymously):**

---

---

---

If you have any suggestions please send this sheet back to:

**email:** [mcdocu.comments@infineon.com](mailto:mcdocu.comments@infineon.com)

**FAX:** +43 (0) 4242 3020 5783



**Your suggestions:**

---

---

---

---

---

---

---

---

---



<http://www.infineon.com>