

COP8™ Development Tools

QUICKSTART FOR THE DM and EM MODULES



August 1999

REVISION RECORD

REVISION	RELEASE DATE	SUMMARY OF CHANGES
A	03/98	FIRST RELEASE
B	08/98	USED DM PICTURES
C	03/99	REVISED ASSEMBLY STARTUP
D	08/99	REVISED, NEW LAYOUT

The information contained in this guide is for reference only and is subject to change without notice.

No part of this document may be reproduced in any form or by any means without the prior written consent of National Semiconductor Corporation.

© Copyright National Semiconductor Corporation, 1998-99

COP8 is a trademark of National Semiconductor Corporation
PC is a trademark of International Business Machine Corporation
iceMaster is a trademark of MetaLink Corporation
Windows and Windows NT are trademarks of Microsoft Corporation
InstallShield is a registered trademark of InstallShield Corporation
Acrobat® is a registered trademark of Adobe Systems Incorporated
WCOP8 IDE is a trademark of K&K Development ApS

Introduction

This QUICKSTART document will give you a quick introduction, as well as a step-by-step guide of how to get started with the COP8 Development Platform.

As an extension to this Quickstart document an Advanced Tutorial is available. The Advanced Tutorial is for the user who needs more detailed information regarding the COP8 Development Platform features.

Summary of the COP8 Development Platform documents:

1. ***Quickstart (this document)***

The Quickstart document contains a guide through software installation, hardware installation and a small tutorial which tests that the software tools, the hardware and development environment are operating correctly.

2. ***Advanced tutorial***

The Advanced Tutorial document introduces and explains many of the COP8 Development Platform features and gives an introduction to the MetaLink Debug Module debugging environment. The Advanced Tutorial takes the user through a working example. Finally, the Advanced Tutorial includes a section for troubleshooting software and hardware problems.

What you need

- Computer and monitor: 486 or higher PC™ with at least 16 MB RAM, a hard disk with at least 32MB of free disk space and a mouse.
- Windows 95, Windows 98, Windows NT, or Windows 3.11 running in enhanced mode (The descriptions in the document will assume a Windows 95 environment)
- (Optional) Printer
- A LED (Light Emitting Diode) and a resistor (at least 330 Ohms)
- Adobe Acrobat Reader installed. If you do not have Acrobat Reader installed, the installation files can be found on the CD-ROM.

Get started!

Begin by clearing the memory and exiting all tasks:

Identify any running programs by lowering the mouse cursor to the taskbar and close all open programs.

Installation of the COP8-NSDEV software package

1. Insert the CD-ROM labeled “*Technical Documentation Tools + Software*” into the CD-ROM drive.
2. If the PC has the CD-ROM “auto run” feature enabled, a small welcome program will be launched. Proceed to step 4.
3. If the PC does NOT have the CD-ROM “auto run” feature enabled then locate the SETUP.EXE program in the root directory of the CD-ROM and launch it. When done a small welcome program will be shown.
4. Press the “Install Tools” button to start the software installation.
5. Read the InstallShield® Welcome text and click **N**ext > to proceed.
6. Read the license agreement and click **Y**es to accept the terms.
7. Read the information text and click **N**ext > to proceed.
8. Enter your name and company in the dialog box and click **N**ext > to proceed.
9. On the InstallShield menu you can select between different installation options:
 - A) **Standard** (default): all tools will be installed into one user defined location.
 - B) **Custom**: only the tools selected by the user will be installed into one user defined location.
 - C) **Special**: Launch of each individual COP8 tools own installation program.
 - D) **Floppies**: This entry enables you to create a set of floppy disks for each COP8 tool package. This would enable installation of each COP8 tool package on PC’s without access to a CD-ROM drive.
10. Select the **Standard** installation and click **N**ext > to proceed.
11. Browse to your preferred directory for installation or just accept the default value of “**c : \cop8**”. Click **N**ext > to proceed.
12. Either type in a new name for the COP8 tools program folder, select an existing group in the list or just accept the default value of “**COP8 Development Platform**”. Click **N**ext > to proceed.
13. Your choices will be shown in an information window. If every thing is OK then click **N**ext > to proceed.
14. All files will now be installed to the directory you selected. After the files have been installed, you can select whether to launch WCOP8 IDE immediately, or whether to close the installation. Click on **F**inish to exit the InstallShield.

Now the COP8 Development Platform is installed on your PC and ready for use.

At the end of the installation you can verify that the correct tools have been installed by using Windows Explorer and comparing your installation to that shown in Fig 1.

COP8 QUICKSTART

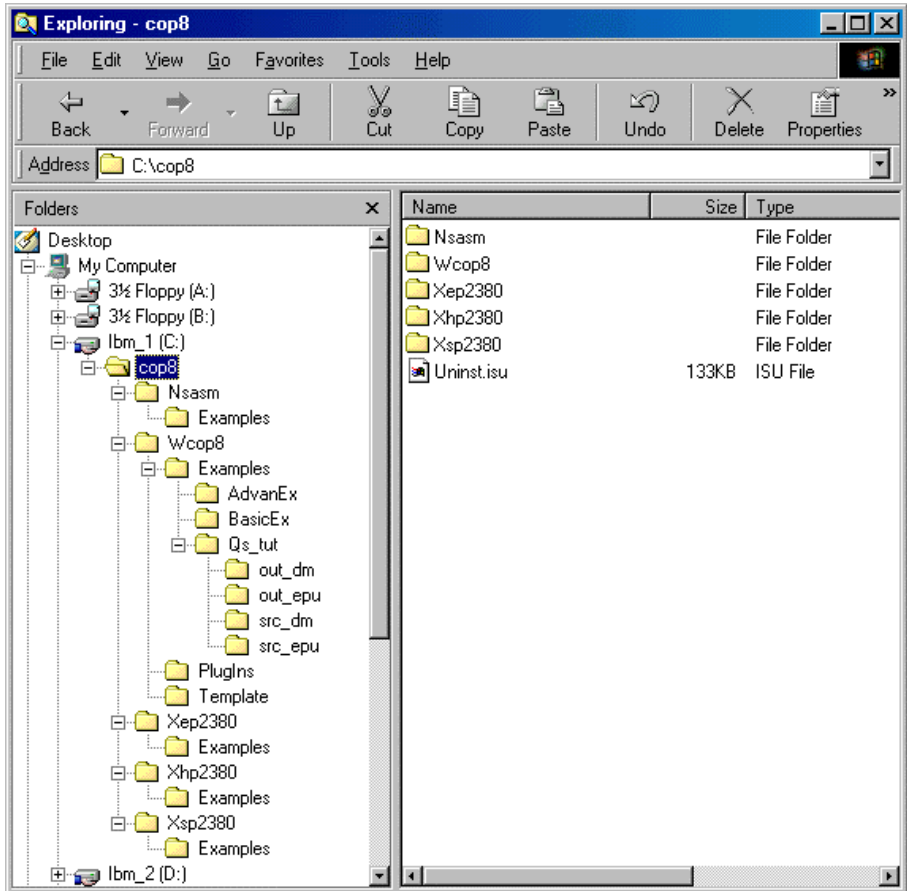


Figure 1.

Note: The exact names of the MetaLink EPU, Emulator DM/AD and Simulator directories could be **Wep2380**, **Whp2380** and **Wsp2380** instead of the names shown above. If the Standard installation has been used, the COP8-NSDEV InstallShield automatically determines your operating system and installs the 16-bit version of the software if you run under Win3.x and the 32-bit versions if you run under Windows 95/98/NT.

The MetaLink debug tool naming legend is as follows:

- | | |
|---------------------------------|---------------------------------|
| Xep2380 – EPU s/w, 32-bit | Wep2380 – EPU s/w, 16-bit |
| Xhp2380 – DM/AD s/w, 32-bit | Whp2380 – DM/AD s/w, 16-bit |
| Xsp2380 – Simulator s/w, 32-bit | Wsp2380 – Simulator s/w, 16-bit |

Installing the Debug Module Hardware

1. Begin by identifying all the components of the system. Locate the serial connection cable, base unit, power supply (the mains VAC can be 100-240V), and an assortment of pin ribbon cable with headers and adapters (types included are dependent on the Debug Module emulation model). The items included should look like the illustration shown in Fig. 2.



Figure 2. Basic content

2. Connect the power supply module (100–240V) to AC Mains and to the Debug Module. Make the connection but do not turn the Debug Module on yet.

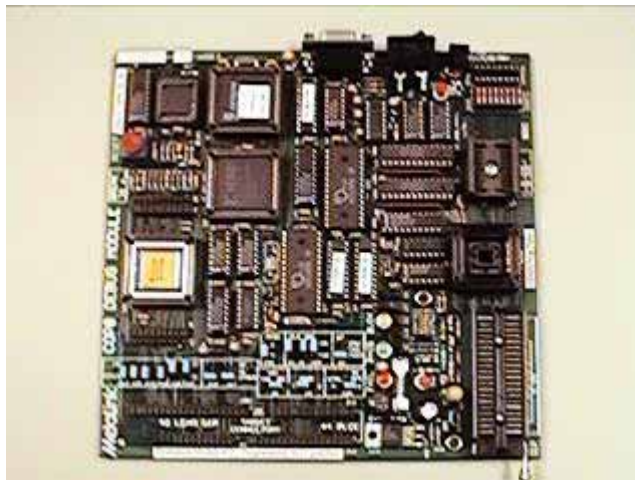


Figure 3. A typical Debug Module

3. Connect the serial cable to a 9-pin serial port connector (may be COM1-COM4) at the PC and to the Debug Module at its 9-pin connector. Connection is 1-1; a null modem adapter is not required
4. Install the appropriate ribbon cable and surface mount adapter to the Debug Module and to your target prototype if desired. For the exercises following Quickstart exercise, it is not necessary to connect the Debug Module to a target. Observe the jumper configuration on the Debug Module; it will be configured to run stand-alone at the frequency of the on board crystal oscillator. As illustrated, the serial connector, on/off switch and power connector are on top. The sockets on the right side are for programming OTP parts. The emulation target connectors are at the bottom left. The emulation bond-out part or daughter board is at center left.

The Quick Start Exercise

The sections which follow, demonstrate the typical steps and procedures for

- entering and modifying a program written for the COP8 Assembler
- running the Assembler and Linker
- downloading the program into the Debug Module for purposes of debugging and eliminating errors
- programming an (E)PROM version of the COP8

We make the assumption that you are somewhat familiar with embedded microcontrollers, software text editors and assemblers, and some form of debugging tool. By following this document closely you will be able to create one example of a working set of firmware even if you are new to developing software for microcontrollers. You do not need to have any previous experience with COP8 microcontrollers to understand and use the example program.

A Note on Developing Software

The first step in developing application software is to carefully specify the operational requirements. Flow-charts or some other technique can be used to document the program sequences in the software (such as the one shown in Fig. 4). Fig. 4 is a high level "idea chart" that we will use for our exercise program. In many cases new application software is written by modifying existing software. Some sample programs, (like `main_dm.asm`, `sub_dm.asm` and others) are supplied with WCOP8 IDE, and they are used for our example. WCOP8 IDE allows you to organize software development into **projects**. This exercise briefly delineates the steps to use a project.

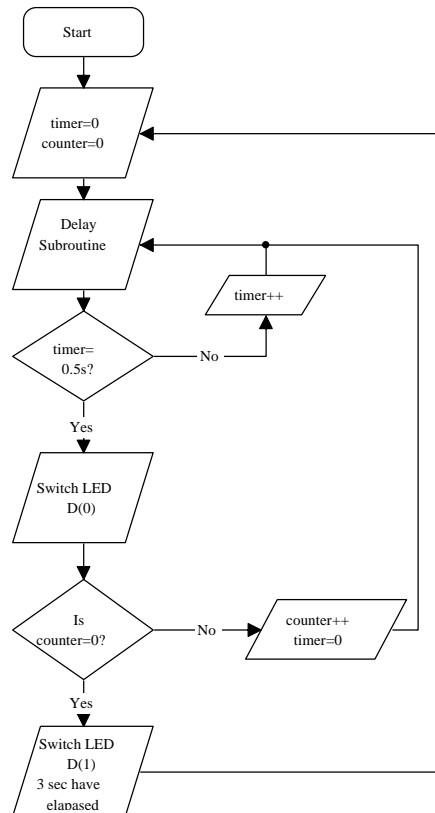
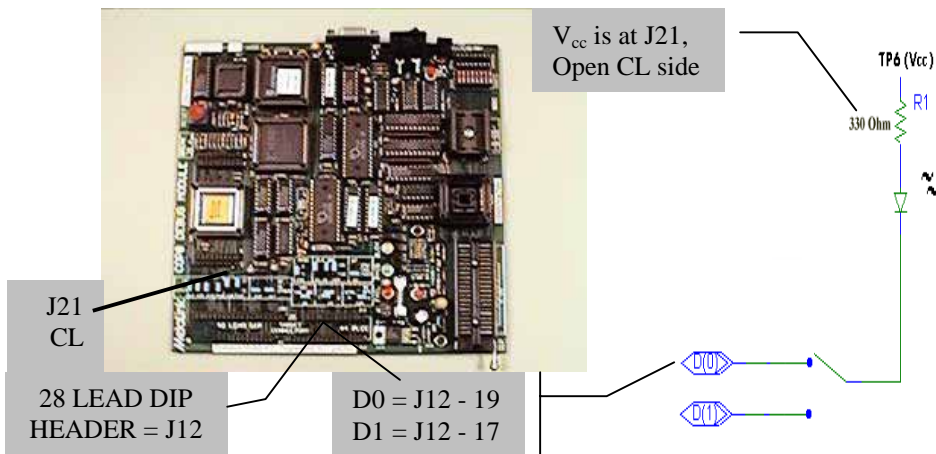


Figure 4.

Step 1. Setup of the Circuit

The modified software, when executing on your Debug Module, will blink an LED at two different rates. The LED and a series resistor are connected between Vcc and one of 2 PORT “D” I/O pins. Attach a clip lead to Vcc; e.g., the CL end of J21 is a convenient spot on the Debug Module. Connect the D0 point (See Fig. 5b) to Pin-19 of the 28 LEAD DIP connector (D1 is on Pin-17). Odd pins (1, 3, ..., 23) are on the inside set of the dual row header pins.



Figures 5a and 5b – Connecting a Test LED Circuit

When the assembled circuit (Fig 5b.) is connected, and the program is running, the LED attached at Debug Module J12-19 = D0 will blink at a high rate and if attached to J12-17 = D1, at about 1/10 that rate. If you have a logic probe or oscilloscope available, building the circuit is not necessary. The Debug Module emulates in real-time. As a consequence, the D0 LED will blink at 10 Hz and the D1 LED will blink at about 1 Hz.

Please note that pin-1 of J12 is shown on the white silk screen. With this style of connector, odd numbered pins are on one side and even numbered pins are on the opposite side. The ribbon cable assembly maps J12-1 to IC DIP header pin 28. J12-28 maps to IC DIP header pin 1, etc. The ribbon cable connects even J12 pins on every other wire and odd pins in the other wires.

Step 2. Open a project and build (assemble/link) the source files

Launch the WCOP8 IDE by selecting **Start|P**rograms|**C**OP8 Development Platform|**W**COP8 IDE from the windows **Start** menu.

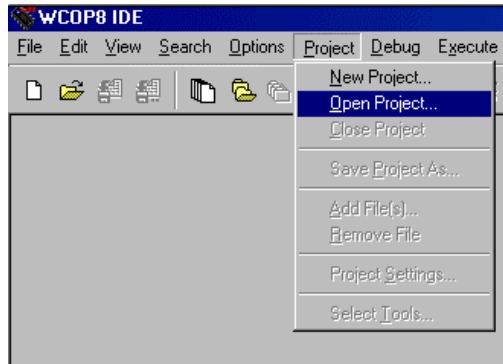


Figure 6.

Select **Project|O**pen Project, and, on the Open Project window (Fig.6), locate the directory and file `c:\cop8\wcop8\examples\qs_tut\tut_dm.prj` and click **OK** to open it. The project opens and the Project Files window appears (Fig. 7).

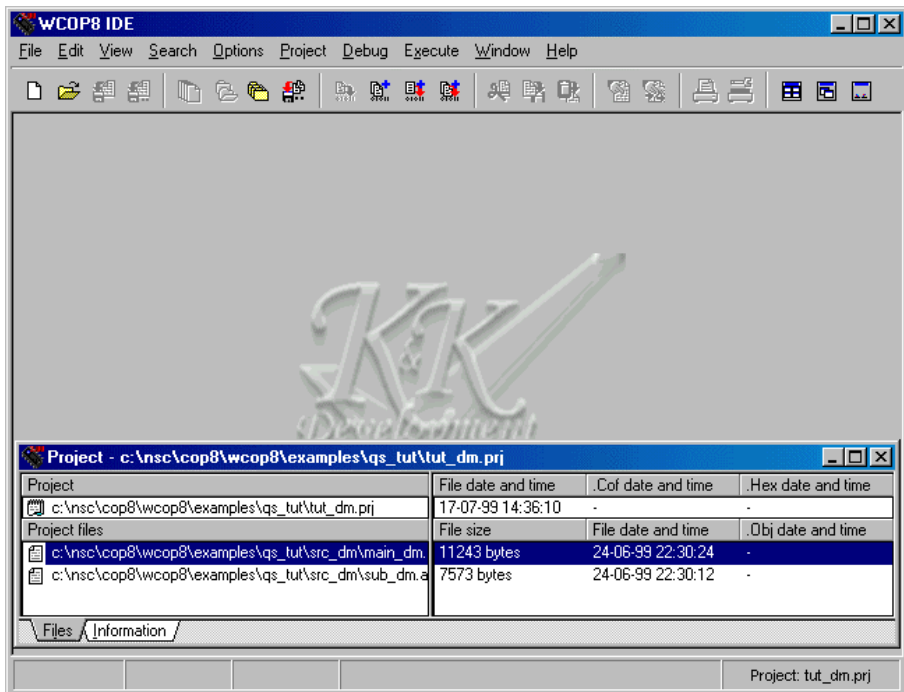


Figure 7.

Double click on the `main_dm.asm` file in the Project Files to open the file in an editor window. Select **Execute|Build** or click on the **Build** icon. A window with the title *Executing* will pop up. WCOP8 IDE will now assemble all files in the project, and will link the program if there are no assembly errors.

If there are assembly errors, an error map is displayed in the *Execute Result* window. If no error(s) occurred then a display such as the one in Fig. 8 will appear.

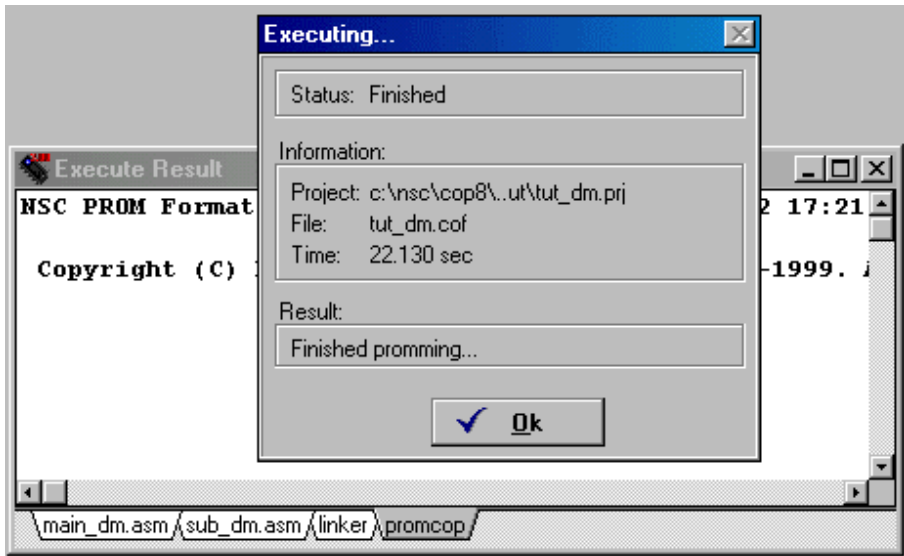


Figure 8

Click the **Ok** button to accept the result.

Note: Large embedded microprocessor projects frequently contain more than one file (module) each of which is assembled separately. The Assembler outputs are then linked together and tested as a whole. WCOP8 IDE has a **make** function that assembles only the files that have been *modified* since last assembly, and then links the files to produce the symbolic output ready for loading into the Debug Module or another MetaLink emulation tool. For this feature select **Execute|Make** or click on the **make** icon. The **build** feature (as we used in this example) assembles/compiles and links all files in the project regardless of whether they have been modified or not.

Step 3. Debugging and Testing Software on the Debug Module

Connect the LED and resistor circuit described previously (in a real project, the ribbon cable and surface mount adapters would be used to connect to a prototype target). Power up the Debug Module and click on **Debug|COP8 Emulator-DM/AD** to start the Debug Module software. A **Select Chip** window will appear. First select the Debug Module (DM) as **Emulation Base** and then select the **Emulation Device**

that matches your Debug Module and Click **OK**. (In this example we have chosen the COP8SAC in a 40-pin configuration). Another window will appear asking for the communications port (COM1-COM4) to which the Debug Module is connected. Click the appropriate COM port and click **OK**. The specific device and COM port information will be saved, so that the user will not have to re-enter information more than once for a given project directory.

The Debug Module software will start, initialize the Debug Module, and load the completed COF file from the project directory. The software and hardware will be in a reset condition with break logically active; the program will not be executing and the program counter will be at 0x0000. COP8 registers that reset will be in that state.

Step 4. Configuring the Debug Module

If there are communication problems between the Debug Module and the PC, use **Configure|Emulator** within the Debug Module software to select the serial port and baud rate. While the serial port is usually set to the highest baud rate, it is sometimes necessary to set the baud rate to a lower value to ensure reliable operation. Refer to the Debug Module manual or the COP8 EMULATOR-DM README.TXT file if you encounter any configuration problems.

In some PC configurations it may be necessary to set up the serial port for optional compatibility to the Debug Module. If communications are clean, do not reconfigure the port. If communication fails then configure the port as follows: from the Windows95 menu use **Start|Settings|Control Panel|System|Device Manager|Ports (COM & LPT)|<port in use>|Properties|Port Settings**. Set the device properties as follows:

Baud	9600	Advanced	
Data Bits	8	FIFO Buffer	Use, if possible
Stop Bits	1	Receive Buffer	0 Tick from Left
Parity	None	Transmit Buffer	0 Ticks from Left
Flow Control	Hardware		

If communication is still not successful, it may be necessary to ‘disconnect’ the port from a prior application that failed to release control; i.e. back out, shut down, power-down and then try again.

Step 5. Loading of Executable File

When activating the MetaLink COP8 Debug Module via the **Debug|Metalink Windows Tools|COP8 Emulator-DM/AD** menu (as we did in step 3), the executable file to be debugged will be transferred automatically as default to the Debug Module. So in this example it should already be there and you can proceed with step 6...!

If for some reason the automatic transfer feature is switched off, then select **File|Load** from within the COP8 MetaLink ICE software so that the executable (in this case `tut_dm.cof`) can be entered into the debugger. In such a case you will need to browse to the directory where the `tut_dm.cof` is located and select it i.e. “`c:\cop8\wcop8\examples\qs_tut\out_dm\tut_dm.cof`”

At the prompt, “Merge into current application environment?”, select **No** (merge allows multiple files to be loaded into memory without pre-initialization to all 0x00 content).

Once loaded the Debug Module is ready to execute, and following your directions, to test the example

Step 6. Running the Code

It is good practice to reset the microprocessor before starting the simulation. This is done by selecting the **RES P** tool bar button or **Run|Reset|Processor**.

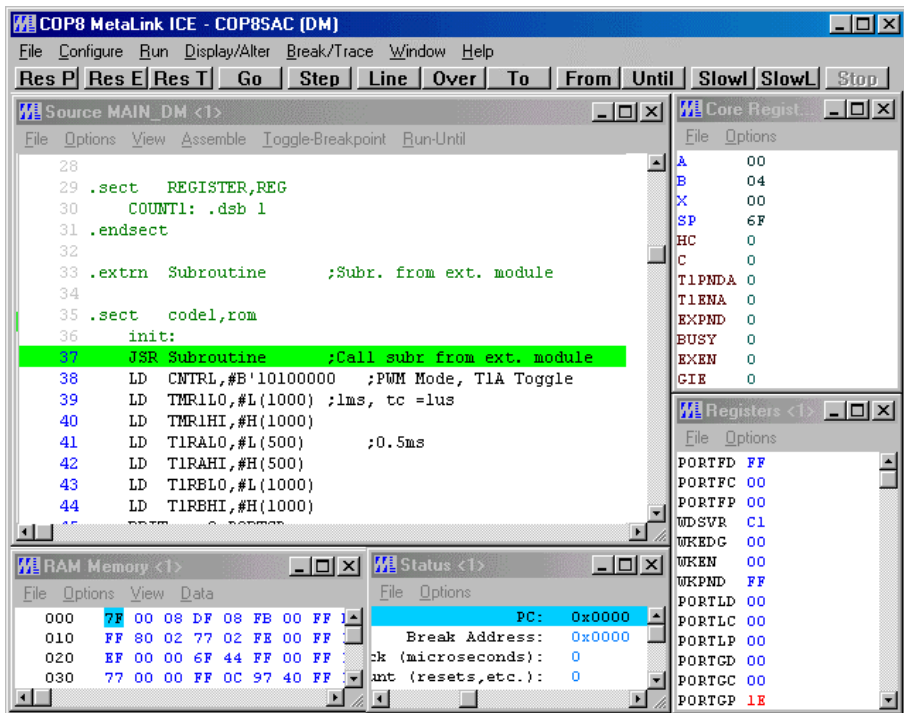


Figure 9.

One method of emulation is to step through the program one instruction at a time. While this approach can be time consuming, it is possible to determine the step-by-step status of the microcontroller. This is accomplished by selecting **Run|Step** (function key **F7**).

You can also select **Run|GO** (or just click on the **GO** button) and see the result.

Pressing the **ESC** button will halt the emulation.

Step 7. Switching between WCOP8 IDE and the Debug Module

The next step in this exercise is to show how the editing of files and debugging process is performed.

Consider the case, where you have located a bug in one of your source files after emulation in the Debug Module. In this case you need to go back to the WCOP8 IDE, edit and correct the bug, re-assemble and link the project and finally reload the executable into the Debug Module.

The above flow is handled as follows:

Switch back to WCOP8 IDE (either via Alt-Tab on the keyboard, or via the Windows 95/98/NT taskbar) where you will find your **tut_dm.prj** project still open. Double click on the **sub_dm.asm** file in the Project Files list and the file opens in an editor window. Change the lines:

```
Subroutine:
NOP
ExitLabel:
to:
Subroutine:
NOP
NOP
ExitLabel:
```

Save the changes via the **File|Save** menu or by pressing the **Save** button. Now you need to re-assemble the file that has been edited and link the project. This is done via the **Execute|Make** menu or by pressing the **Make** button. Press the **OK** button on the result window when WCOP8 IDE is finished assembling/linking.

Switch back to the Debug Module software (either via Alt-Tab on the keyboard, or via the Windows 95/98/NT taskbar).

The Debug Module software will then notify the user that the executable loaded into the Debug Module has changed. The Debug module will ask you whether you want reload the file into the Debug Module or if you would like to keep the previously loaded file.

Select **YES** for a reload, and at the prompt, “Merge into current application environment?”, select **NO** (merge allows multiple files to be loaded into memory without pre-initialization to all 0x00 content).

You should now see your executable being reloaded into the Debug Module and it is now ready for debugging once again.

Step 8. Programming an (E)PROM version

The final step in this exercise will show the procedures for programming the COP One Time Programmable (OTP) and UV erasable microprocessors supported by the Debug Module.

Select **File|PROM Programmer|Device** to display the set of COP devices that can be programmed by the Debug Module. Select the appropriate device from the list (e.g. COP8SAC740N for a 40-pin DIP version of the COP8SAC). A window similar to that of Fig. 10 should appear to allow programming of the COP8 microcontroller.

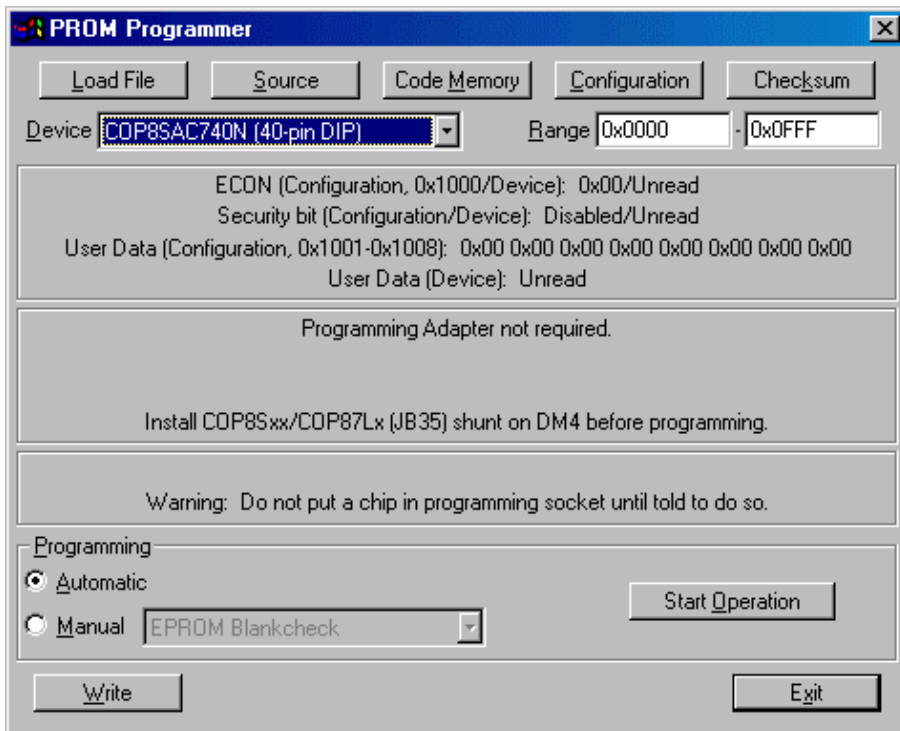


Figure 10. Programming Function Display

Clicking on the **Configuration** button will bring up another window (Fig 11.) which will allow a detailed configuration of the microcontroller.

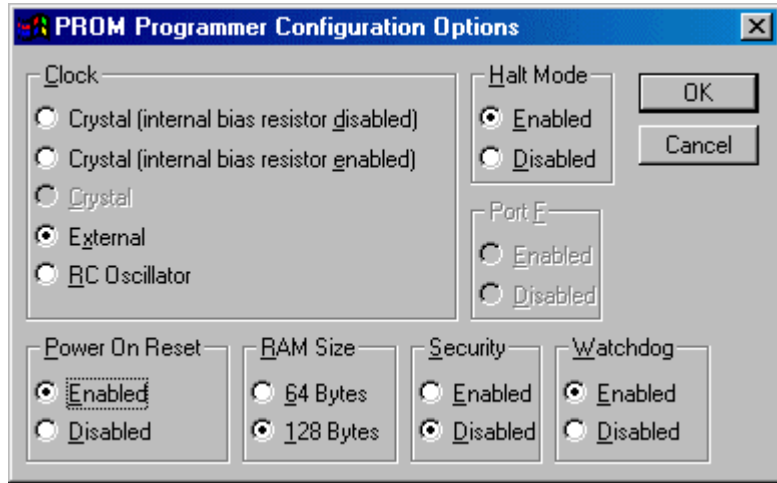


Figure 11. ECON Configuration Menu

Figure 11 displays the programmable options that may be set. For purposes of this exercise, the options should be set as shown. The functions selected in the **ECON** register are indicated by a black dot and these functions will be determined upon programming the OTPs. Not all devices configure using ECON and within ECON the configuration bits may change function from device to device. ECON is fully described in the COP8 Data Sheet for your device. Within the engineering laboratory, programming the **security** bit may not be desirable.

If the program has not been loaded, then select **L**oad to load the program into the Debug Module so that the COP8 microprocessor can be programmed. This is not necessary if the program has been loaded as part of the debugging process; the OTP will program from the same memory that was used by the debugger for emulation.

Programming the COP microcontroller is accomplished by selecting the **File|EPROM Programmer...|A**utomatic function and then clicking on the **S**tart **O**peration. This Debug Module first verifies that the COP device is blank, then programs the code with byte by byte verification, the programs Configuration (ECON) without **security**, the block verifies the ROM content and ECON content and finally, if enabled, programs **security**. These steps can be executed one by one in 'manual' mode: **File|EPROM Programmer...|M**anual and then select each function from the pull-down function menu.

Click on the button **S**tart **O**peration to begin each function. After programming Click on the **E**xit button to get back to the main client window. The software will ask you to remove the chip from the programming socket. Make sure the chip is not in the socket. Leaving the chip in the socket may cause damage to both the emulator board and/or the surrounding circuit.

After programming the microcontroller you can test the behavior of the code in a target system.

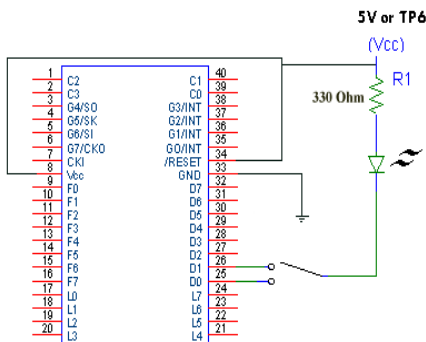
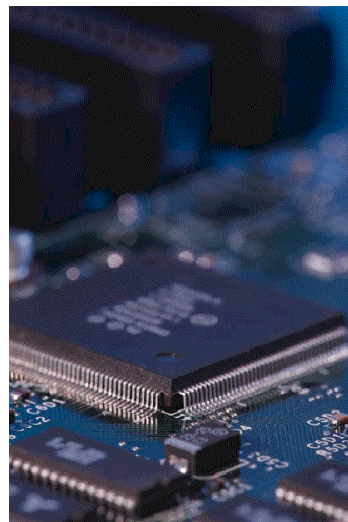


Figure 12. Test Circuit Schematic

Replace the ribbon cable header with the newly programmed chip and apply a clean 5 volts (preferably from a power supply). Detach the clip from TP6 and attach it to a supply VCC (+5V) and the microcontroller ground to that of supply GND. Make sure that the /RESET line is tied high as to enable the POR (Power On Reset) circuit. Use a square wave signal at the CKI pin as clock. The setup should look similar to that of Fig. 12.

This concludes the quick start lesson and hopefully all part of the software and hardware worked as described.

For an introduction to the many features of WCOP8 IDE and the MetaLink Debug Module please refer to the Advanced Tutorial or the respective User Manuals and Help Files of the tool in question.



Appendix A

Assembly Code For The QuickStart Lesson (main_dm.asm)

```

;*****;
;* COMPANY      : K&K Development                *;
;* PROJECT      : WCOP8 IDE Test Assembler Project *;
;* FILENAME     : Main_dm.asm                   *;
;* VERSION     : 1.0                            *;
;-----;
;*
;*
;* MAIN PROGRAM FOR DEBUG MODULE TUTORIAL        *;
;*
;* This file is part of the test project that comes *;
;* with the WCOP IDE software. The only purpose of the *;
;* project is to serve as an example when exploring the *;
;* features of WCOP8 IDE.                          *;
;*
;* K&K Development makes no warranty, representation or *;
;* guarantee regarding the suitability of this project *;
;* for any particular purpose.                     *;
;*****;

;Assembly Code For The QuickStart Lesson

; .chip COP8SAA          ; Place Chip directive for COP8SAA
;                        ; It is assumed, that a 28-pin device
;                        ; will be used in hardware.
.incl cop8saa.inc       ; dedicated Include file for COP8SAA
;                        ; supplied with the NS Assembler/linker
;                        ; WCOP8 IDE will find it there in this project

.sect REGISTER,REG
COUNT1: .dsb 1
.endsect

.extrn Subroutine          ;Subr. from ext. module

.sect codel,rom
init:
JSR Subroutine           ;Call subr from ext. module
LD CNTRL,#B'10100000    ;PWM Mode, T1A Toggle
LD TMR1LO,#L(1000)      ;lms, tc =1us
LD TMR1HI,#H(1000)
LD T1RALO,#L(500)        ;0.5ms
LD T1RAHI,#H(500)
LD T1RBLO,#L(1000)
LD T1RBHI,#H(1000)
RBIT 3,PORTGD
SBIT 3,PORTGC
LD PSW,#B'00010001      ;Enable GIE,T1ENA
SBIT 4,CNTRL            ;Start Timer T1

```

Assembly Code Continued (main_dm)

```
        WAIT:
        JSR    DELAY
        JSR    TOGGLE
        JP     WAIT
.endsect
;
;*****Interrupt handler*****
.sect   intr,rom,abs=0xff
        RBIT   5,PSW                ;Reset Timer T1A pending flag
        LD    A,PORTD
        XOR   A,#001                ;Toggle Port D0, 1.5ms
        X     A,PORTD
        RETI
.endsect
;
;*****
.sect   delay,rom
        DELAY:
        LD    COUNT1,#0FF

        LABEL1:
        DRSZ  COUNT1
        JP    LABEL1
        RET
.endsect
;
;*****
.sect   toggle,rom,abs=0x200
        TOGGLE:
        LD    A,PORTD
        XOR   A,#002
        X     A,PORTD
        JSR   DELAY
        RET
.endsect
;
;*****
.end    init
```

Assembly Code For The QuickStart Lesson (sub_dm.asm)

```
*****;  
;* COMPANY      : K&K Development                *;  
;* PROJECT      : WCOP8 IDE Test Assembler Project *;  
;* FILENAME     : Sub_dm.asm                    *;  
;* VERSION      : 1.0                          *;  
;-----*;  
;* *;  
;* SUB MODULE FOR DEBUG MODULE TUTORIAL          *;  
;* *;  
;* This file is part of the test project that comes *;  
;* with the WCOP IDE software. The only purpose of the *;  
;* project is to serve as an example when exploring the *;  
;* features of WCOP8 IDE.                          *;  
;* *;  
;* K&K Development makes no warranty, representation or *;  
;* guarantee regarding the suitability of this project *;  
;* for any particular purpose.                    *;  
*****;  
  
;Assembly Code For The QuickStart Lesson  
  
.includ      cop8saa.inc      ;supplied with the NS Assembler/linker  
.public      Subroutine      ;Subr. in this module for export  
  
*****;  
  
.sect      Sub_User,ROM  
Subroutine:  
NOP  
  
ExitLabel:  
RET  
.endsect      ;Sub_User  
  
.end      ;No label, submodule!
```