

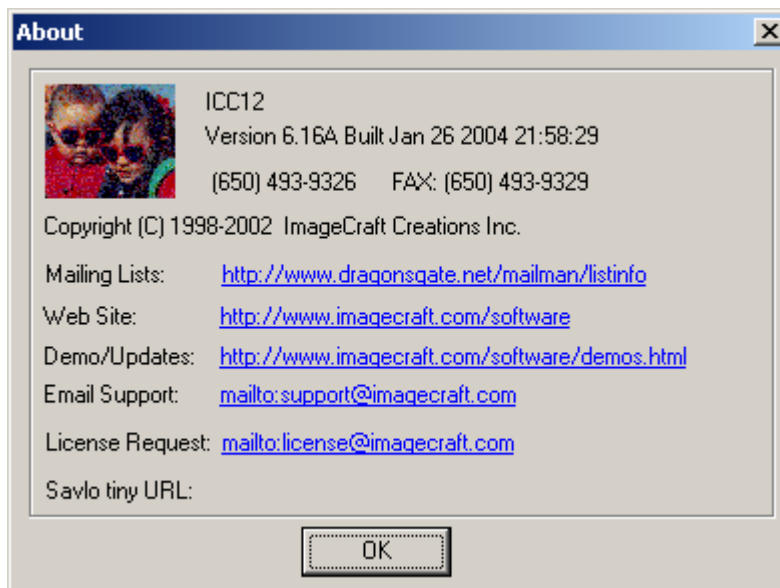
## How to use ICC12 with NC12 and uBUG12

This document will show and demonstrate the use of ImageCraft ICC12 Latest **Version 6** with Technological Arts' NC12 module. uBUG12 GUI is used to erase and program FLASH after the compilation of a test program. Other method can be used to also erase and program the FLASH but in this example it will be the uBUG12.

This document assumes that the user is familiar with C and so will not teach how to program C here.

This document further assumes that the Serial Monitor has not been erased and is presently in the 9S12C32 MCU.

### ImageCraft Links:



<http://www.imagecraft.com/software/>

<http://www.ece.utexas.edu/%7Evalvano>

<http://www.dragonsgate.net/FAQ/cache/20.html>

<http://www.imagecraft.com/software/mdevtools.html>

<http://www.dragonsgate.net/mailman/listinfo>

### Technological Arts Links:

<http://www.technologicalarts.com/myfiles/nc12.html>

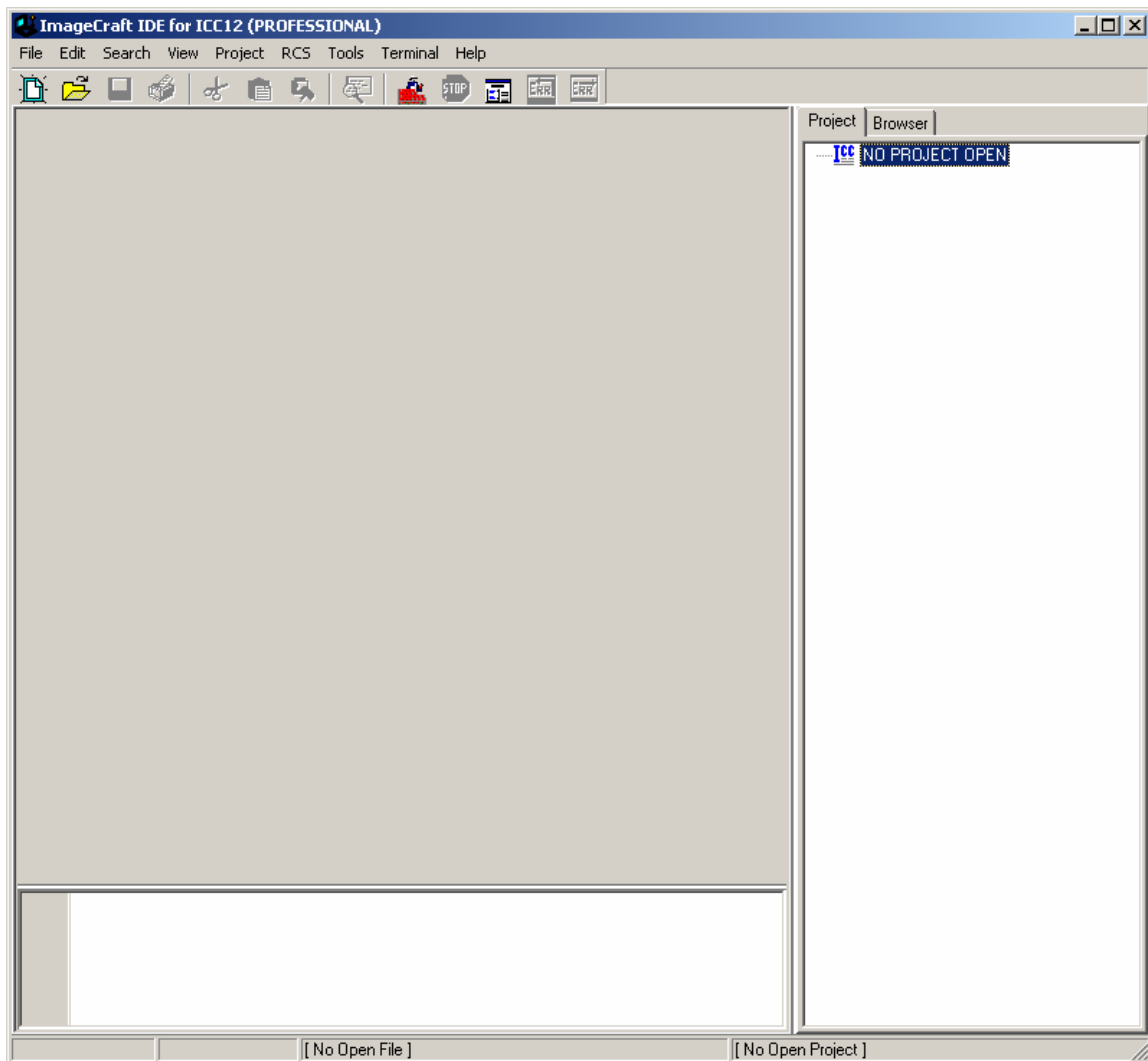
<http://support.technologicalarts.ca/files/uBug12.zip>

## Getting Started:

Double click on the ICC12 icon. If a user has not read the ICC12 manual and just open the IDE one will wonder what to do next. Well wonder no more.

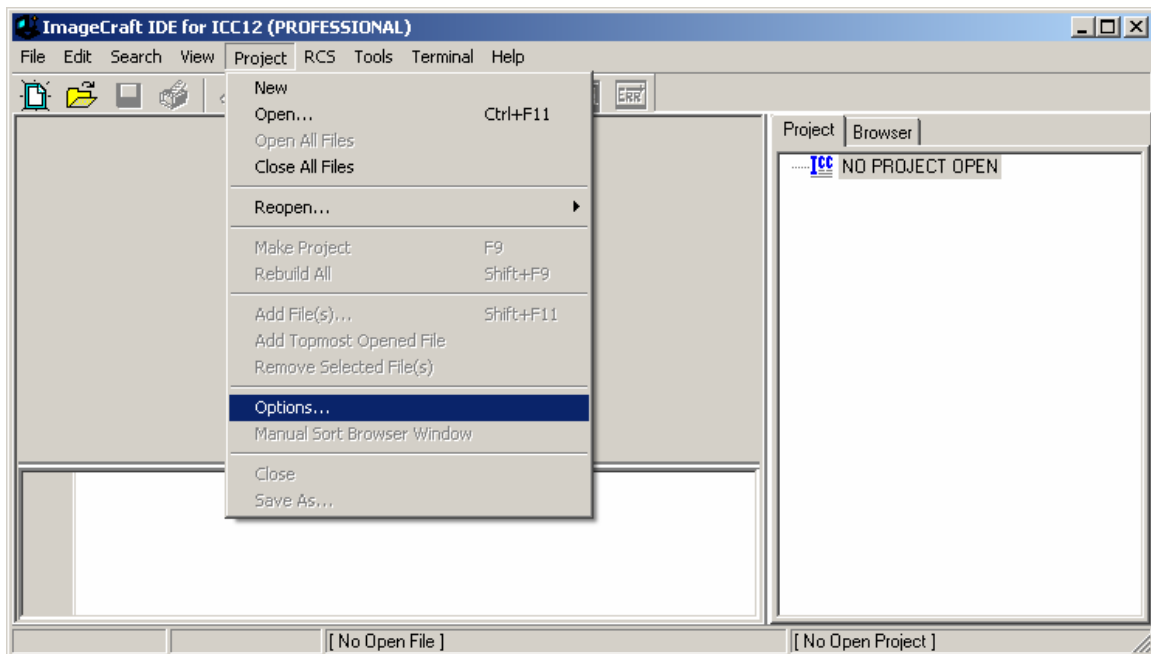
Note the 3 window panes. The top left most is greyed out and the right is the project window. The left bottom pane is where the error messages are displayed during compilation.

Before creating a new Project, the hardware target in the Compiler Options must be setup properly for the target MCU. This is to ensure that the compiler will setup the type of MCU the C program will compile for. In this example it is the NC12.

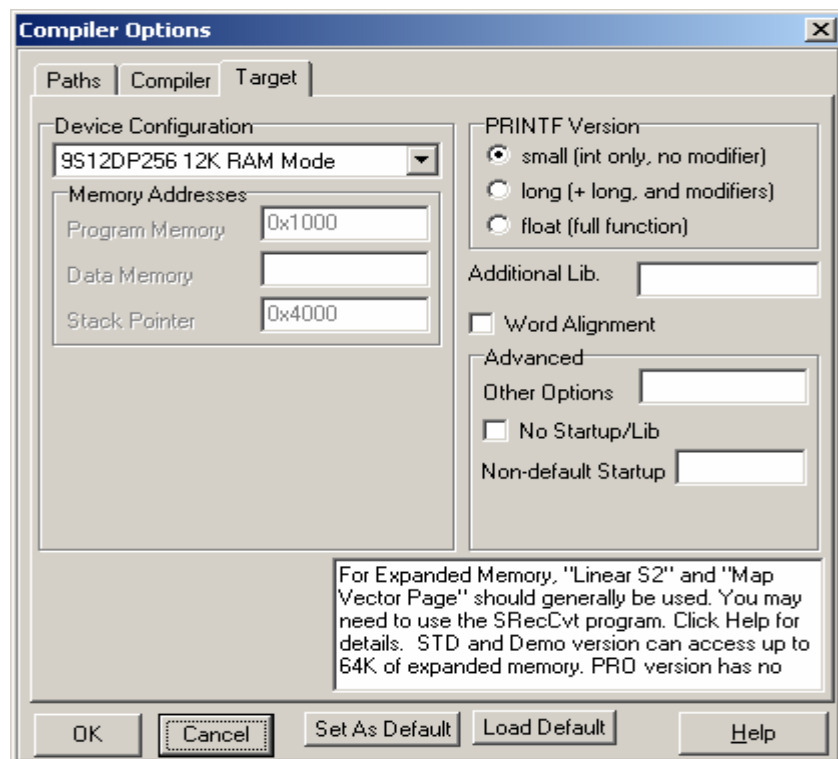


## Compiler Setup:

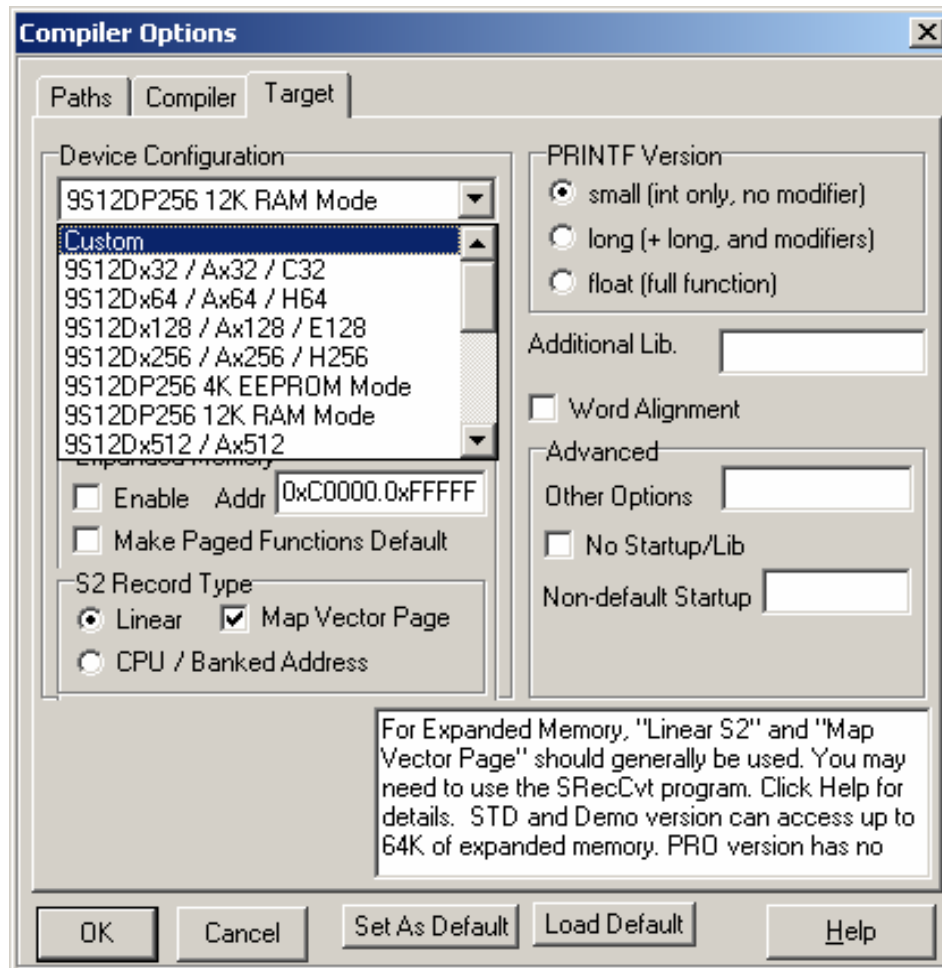
Click on Project Menu – Options – Target Tab.



Please note the Device Configuration. Click on the pull down arrow to change the device type.



Scroll up or down to select Custom as shown. Note that an 9S12C32 device Configuration does already exist. Unfortunately, the addresses are not setup properly with using uBUG12. Therefore the Custom configuration must be selected and the memory parameters are changed to reflect uBUG12 usage.



### Custom Device Configuration:

Program Memory: **0x4000.0x7FFF:0xC000.0xFFFF**

Data Memory: **0x3800**

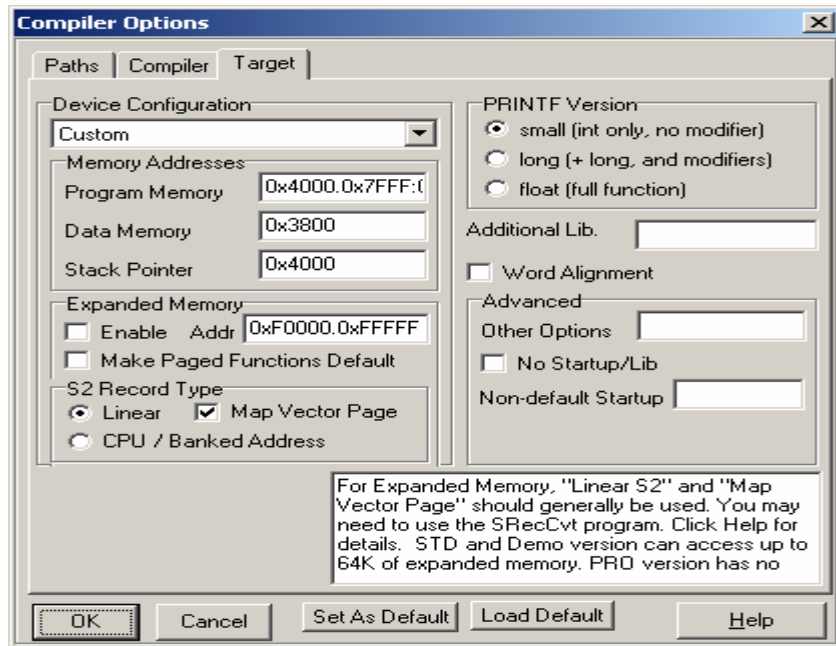
Stack Pointer: **0x4000**

### Expanded Memory:

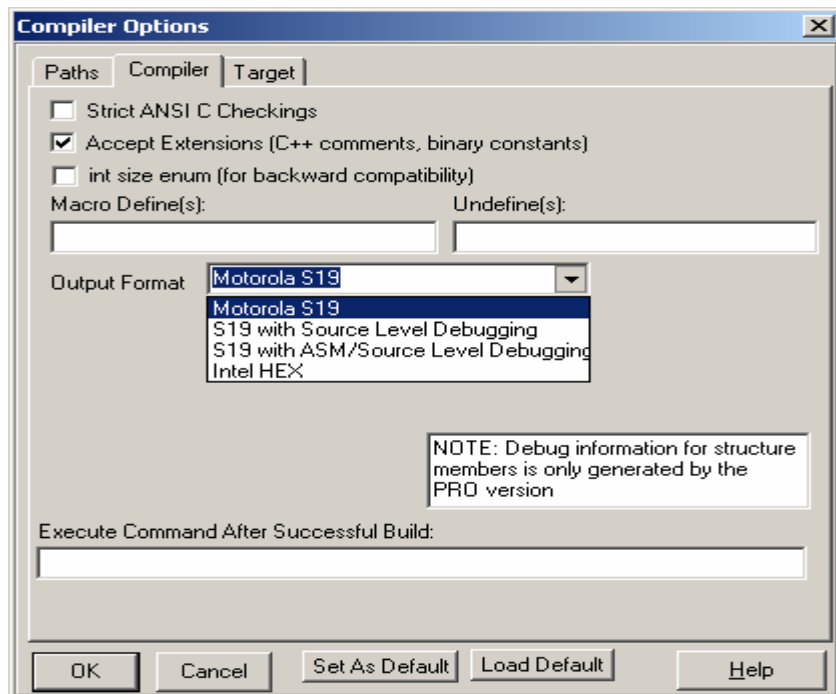
PPAGE \$3E and \$3F are the fixed memory and are allocated for **0x4000.0x7FFF:0xC000.0xFFFF**

## S2 Record Type:

Make sure to select Linear and the Map Vector Page is check marked.

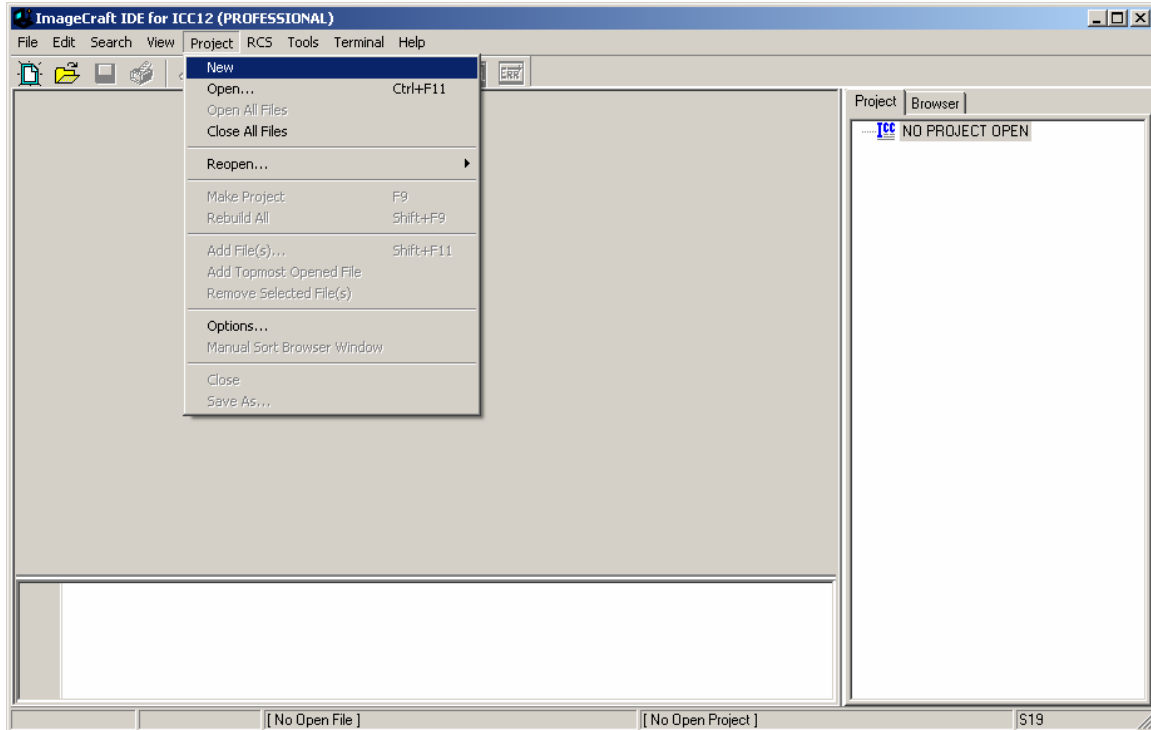


On the compiler tab there are several choices of S-record output as shown. Select which one that suits you.

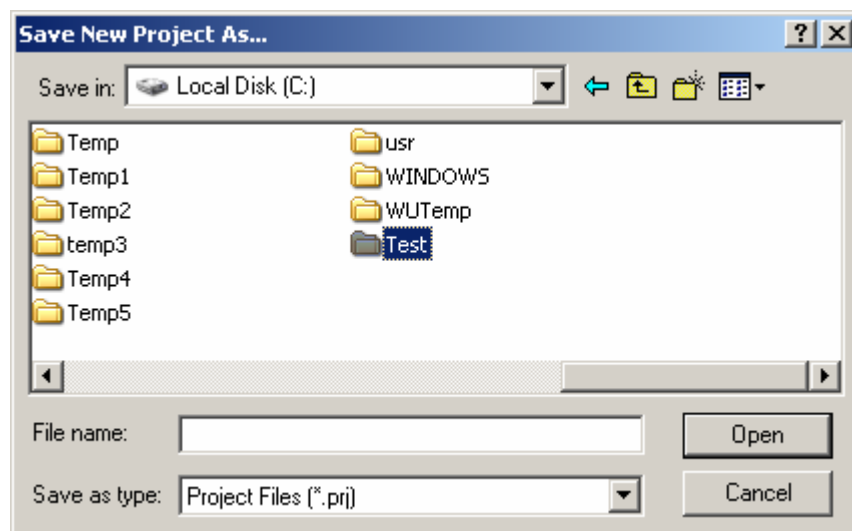


## Starting a new Project:

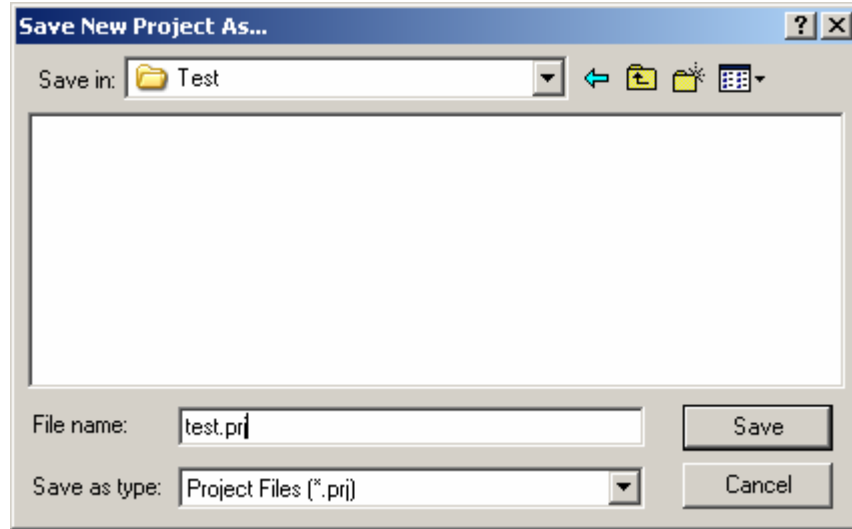
Once the compiler options are setup, a new project can be created. Click Project menu – New.



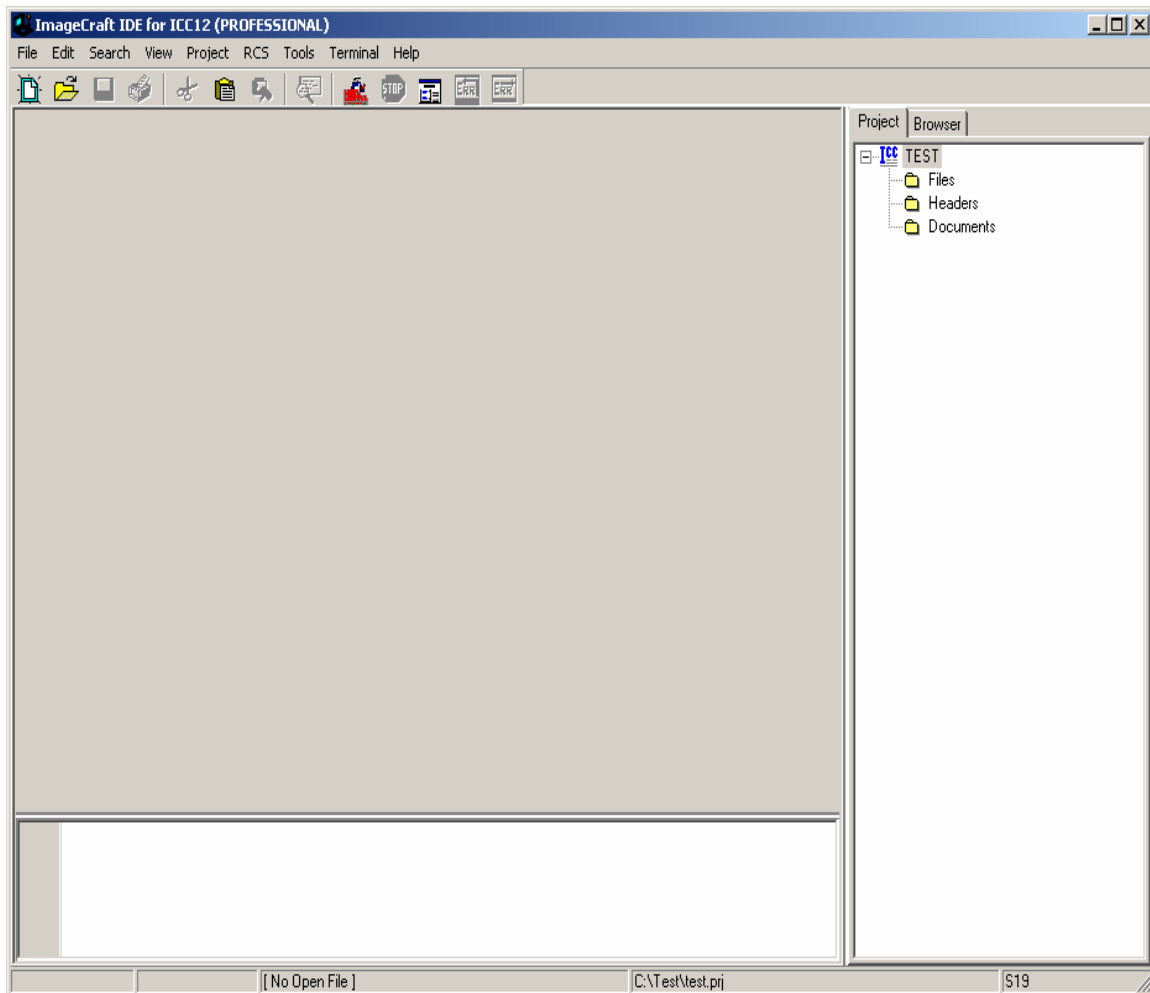
The ICC12 will prompt to save the new project. The user should decide whether to create a new directory to save the new project. In this example a new directory called **Test** is created and the file is saved as file **test.prj**.



Type the filename as **test.prj** and click on the Save button.

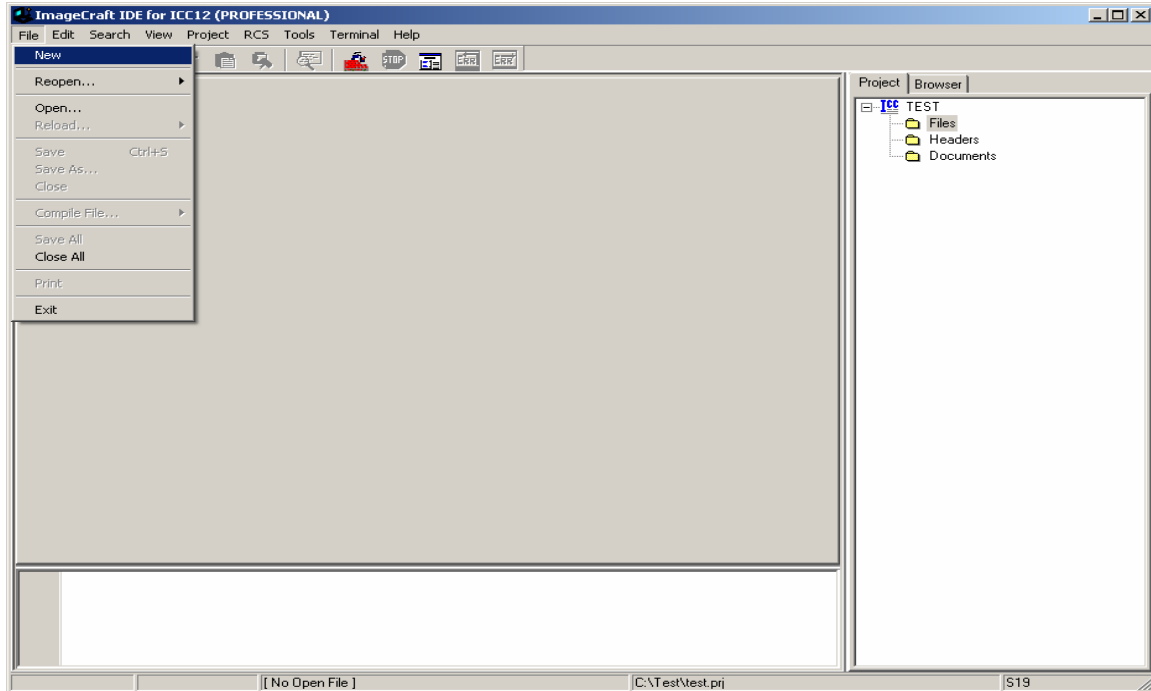


Note that the project window has changed to add Files, Headers and Documents.

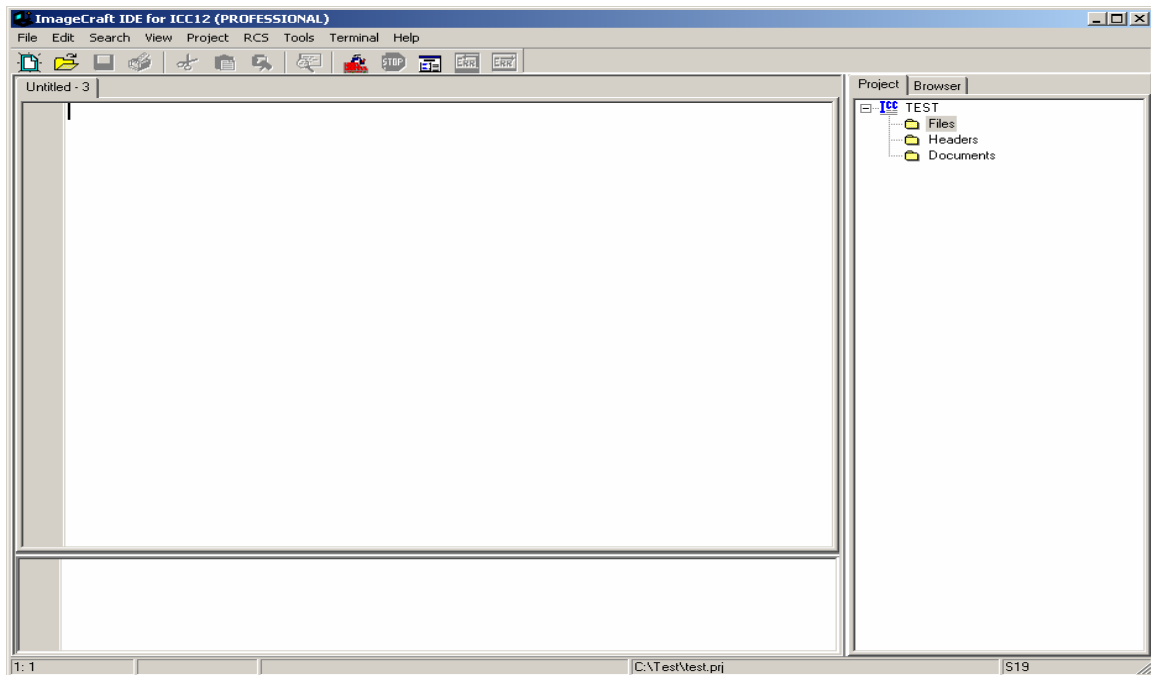


## Creating a new file to the project:

To add files to the project, click on the File menu – new as shown.

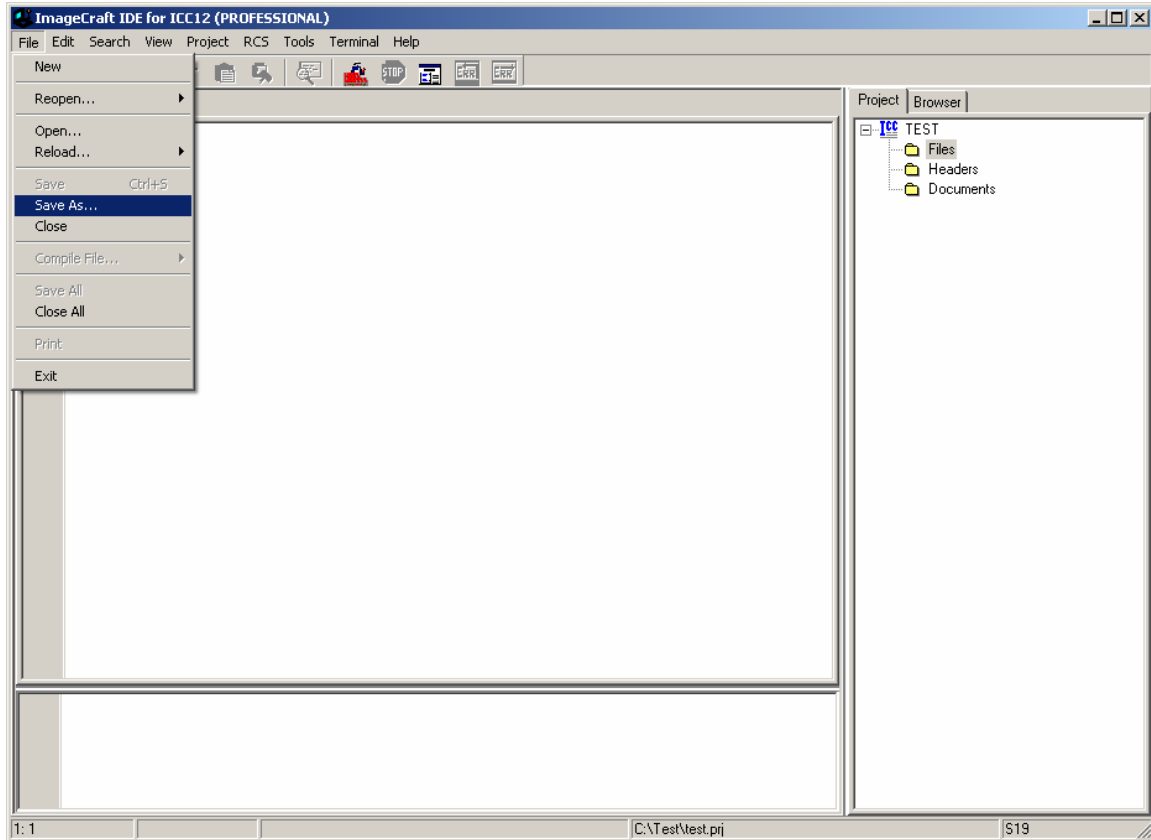


Note that ICC12 created an untitled file. Save the file as **BlinkLED.C**.

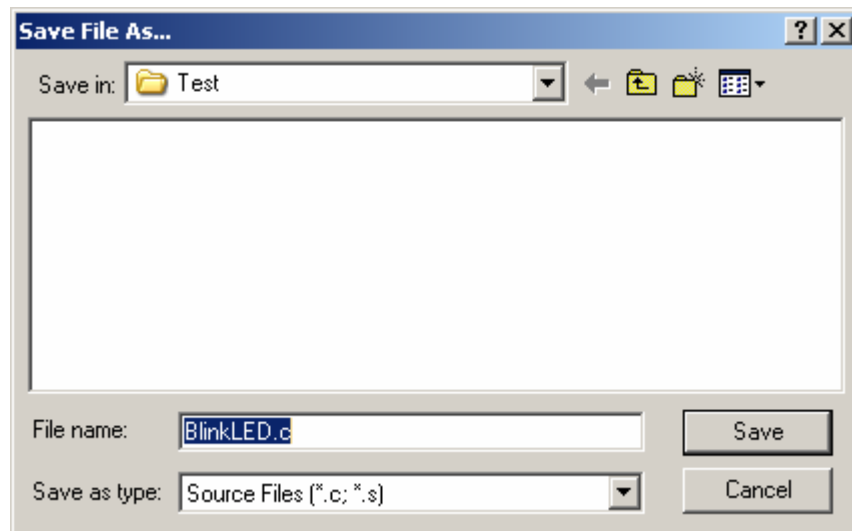




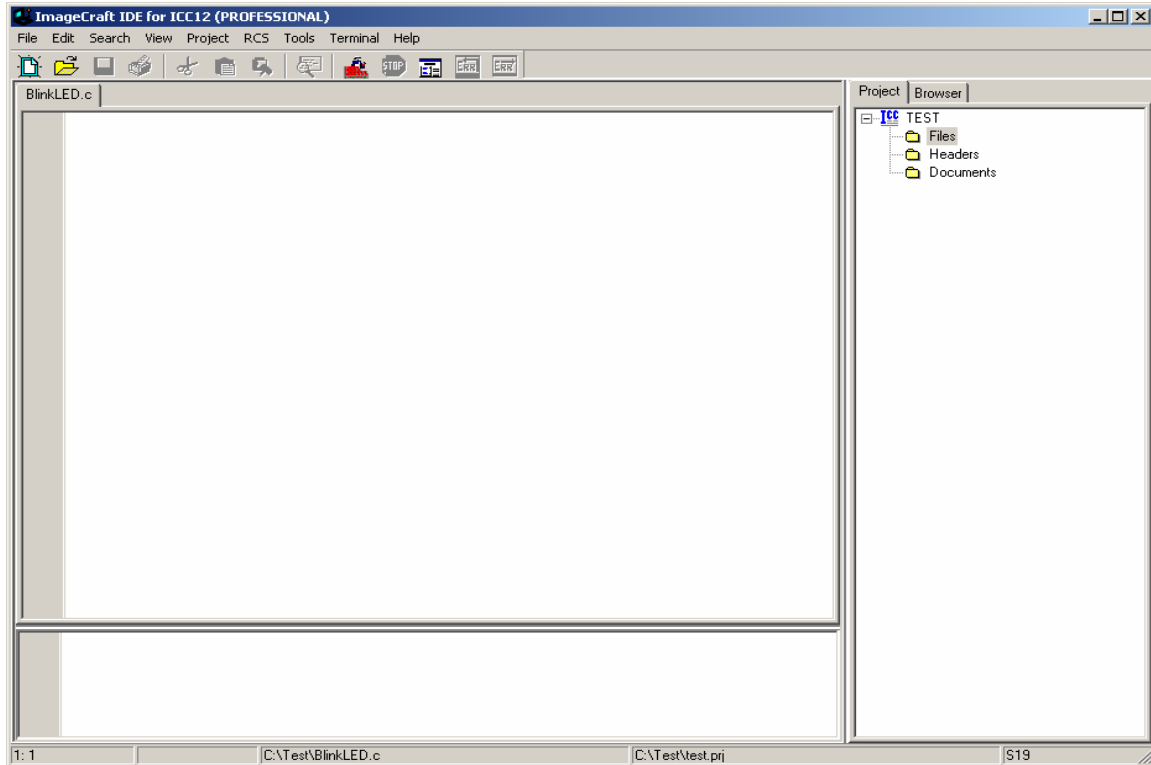
To save, click on File menu – Save As



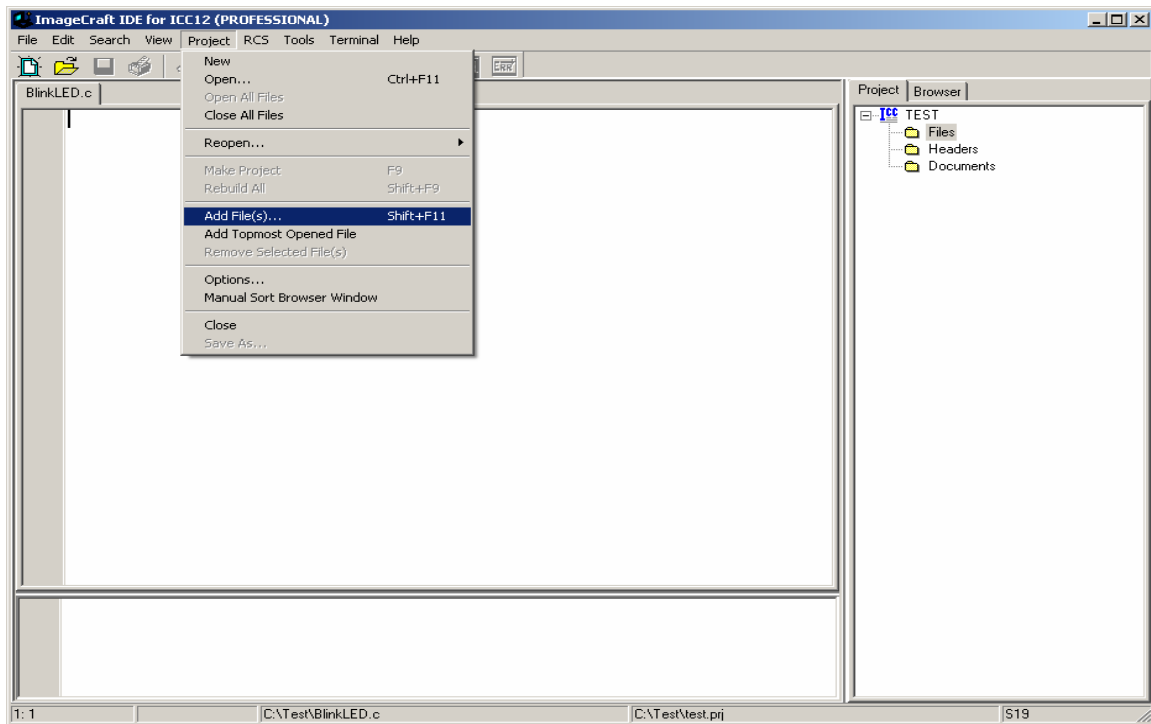
ICC12 will open an explorer window to help save the file. Type BlinkLED.c then press the save button.



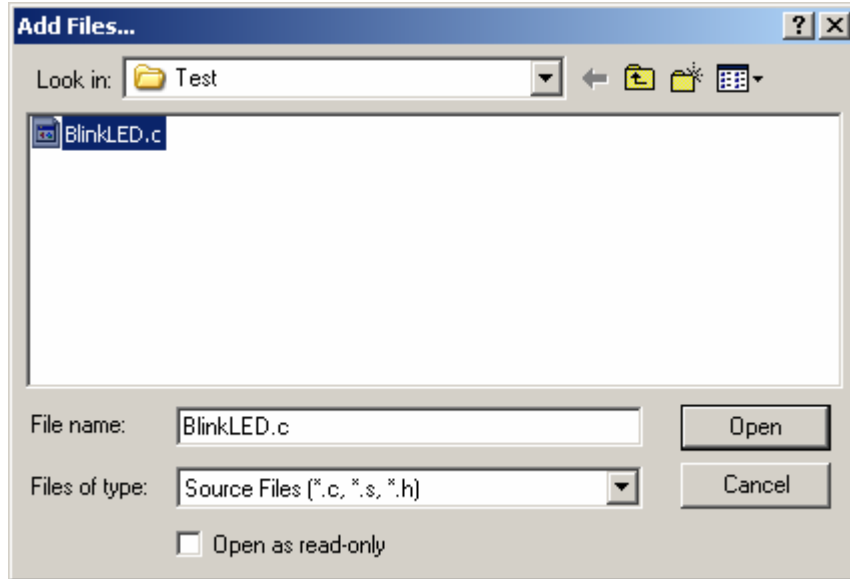
Note that ICC12 has renamed the file to BlinkLED.c.



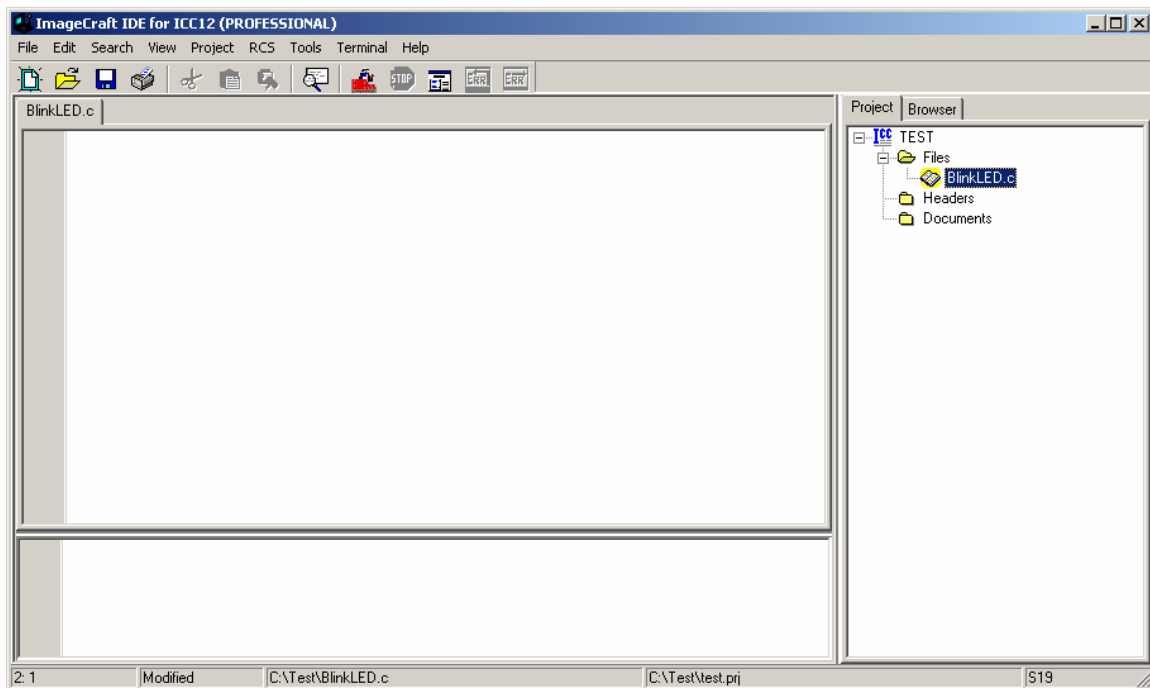
To add BlinkLED.c to the Project, click on the Project menu – Add File(s)



ICC12 will open an explorer window to help and locate the file of interest.

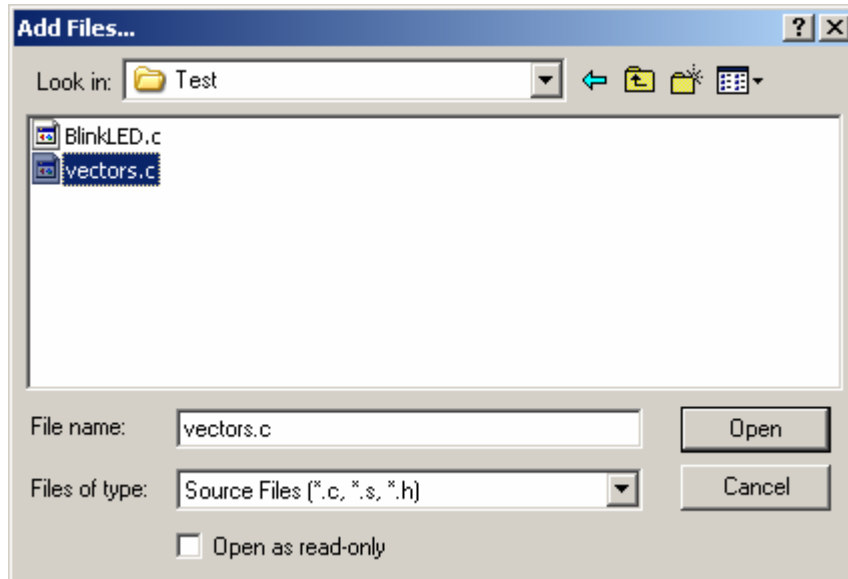


Note that the right window pane has changed to include BlinkLED.c under the Files Project.

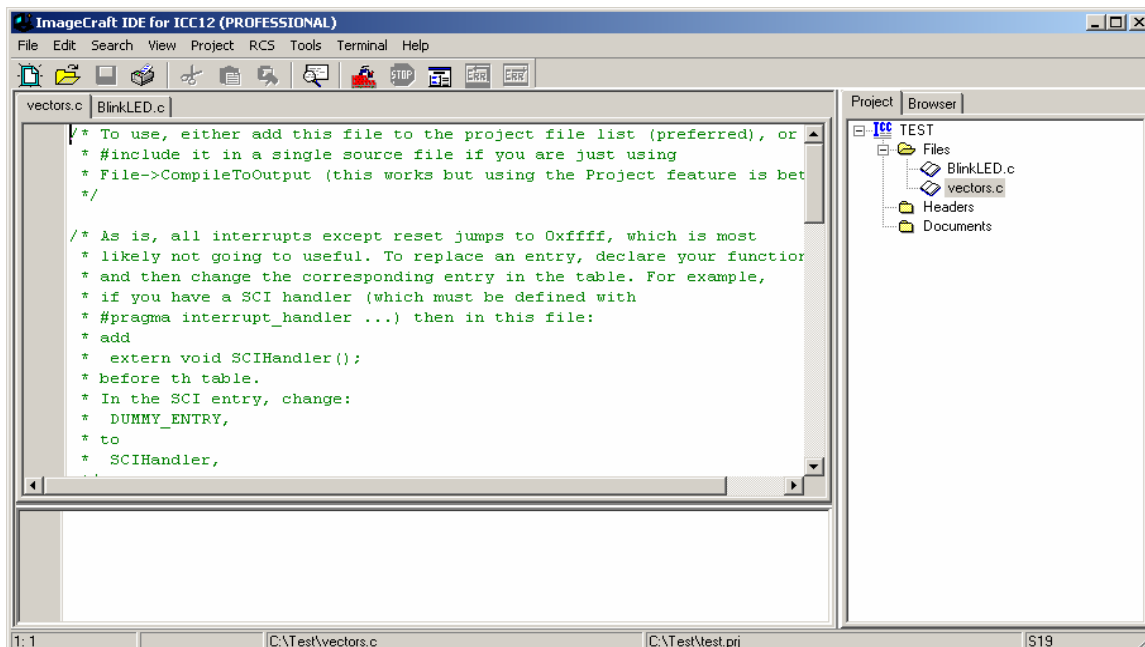


Locate **vectors.c** and copy file to Test directory. The major reason why this must be done is because of project to project dependency. It is not good to keep editing a single **vectors.c** if other projects are using this same file. It becomes a problem to keep track of the changes made to the different projects.

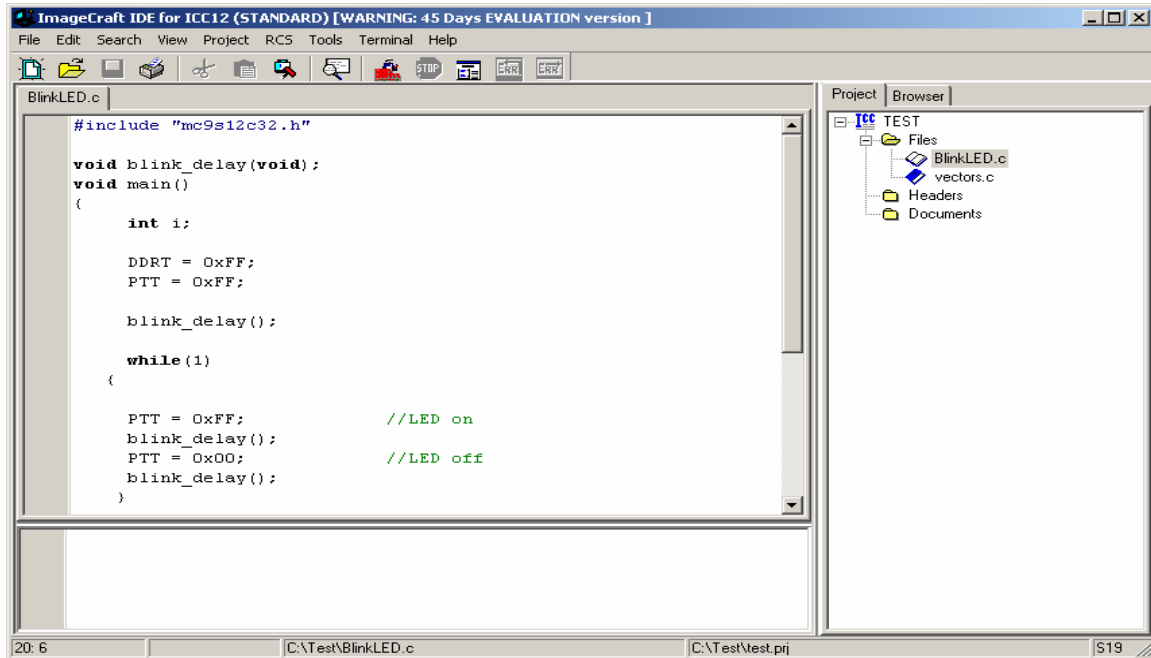
To add **vectors.c** to the Project, click on the Project menu – Add File(s)



Note that ICC12 has changed to include **vectors.c**. It is important to note that the **vectors.c** was written for the 68HC912B32 and 812A4 MCUs. One should edit the file to include other ISR addresses for the 9S12C32. This example uses the file as is.



Write the codes below into BlinkLED.c file. Once it is written we can then compile/make/build the code.



```
#include "mc9s12c32.h"
```

```
void blink_delay(void);
```

```
void main()
```

```
{
```

```
    int i;
```

```
    DDRT = 0xFF;
```

```
    PTT = 0xFF;
```

```
    blink_delay();
```

```
    while(1)
```

```
{
```

```
    PTT = 0xFF;
```

```
    //LED on
```

```
    blink_delay();
```

```
    PTT = 0x00;
```

```
    //LED off
```

```
    blink_delay();
```

```
}
```

```
}
```

```
void blink_delay(void)
```

```
{
```

```
    int i;
```

```
    for(i=0;i<64000;i++)
```

```
    {
```

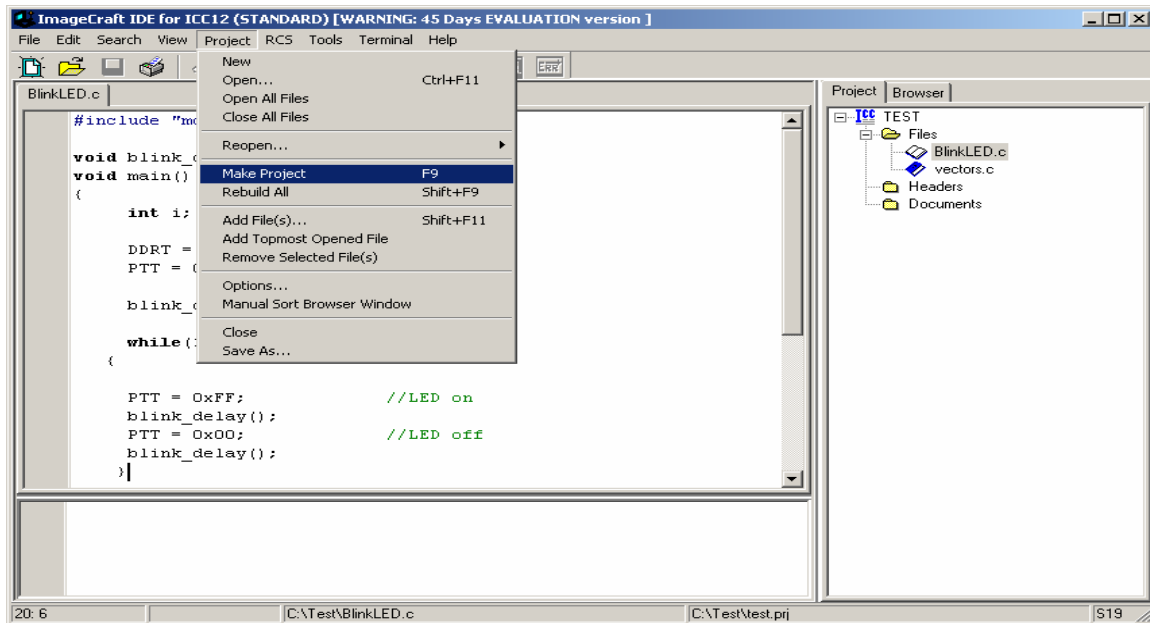
```
        ;
```

```
    }
```

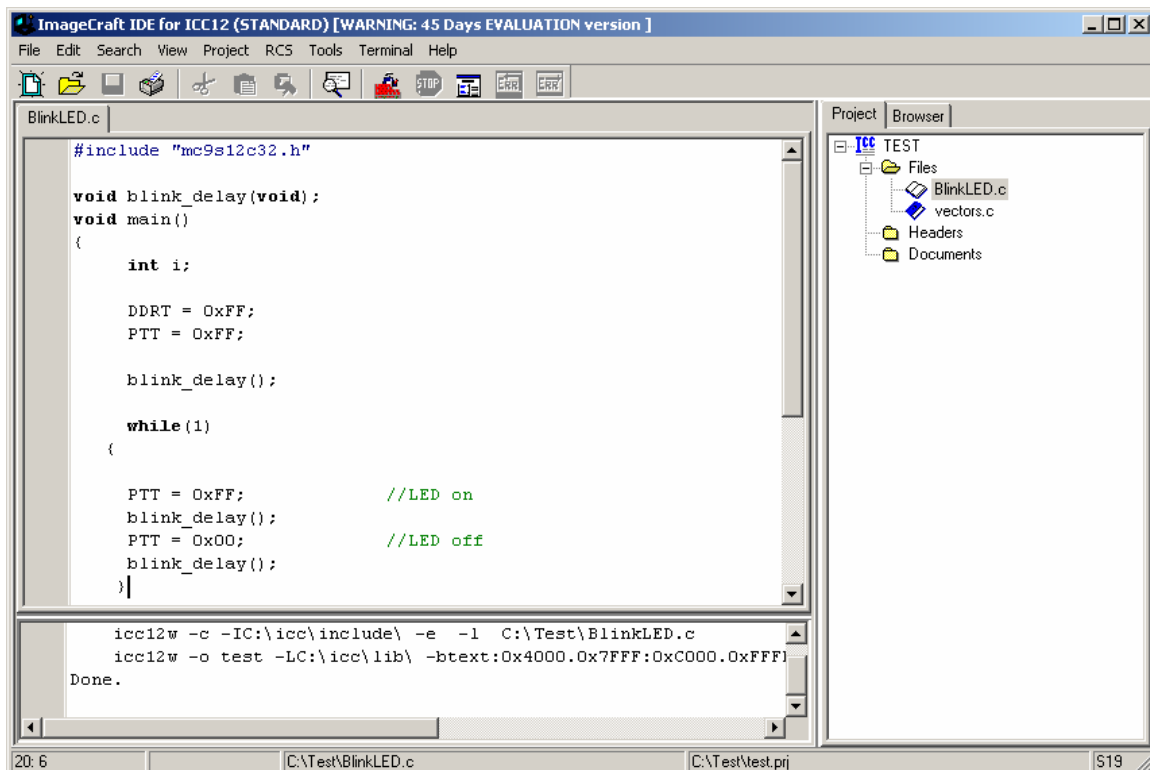
```
}
```

## Compiling/Build/Make the file:

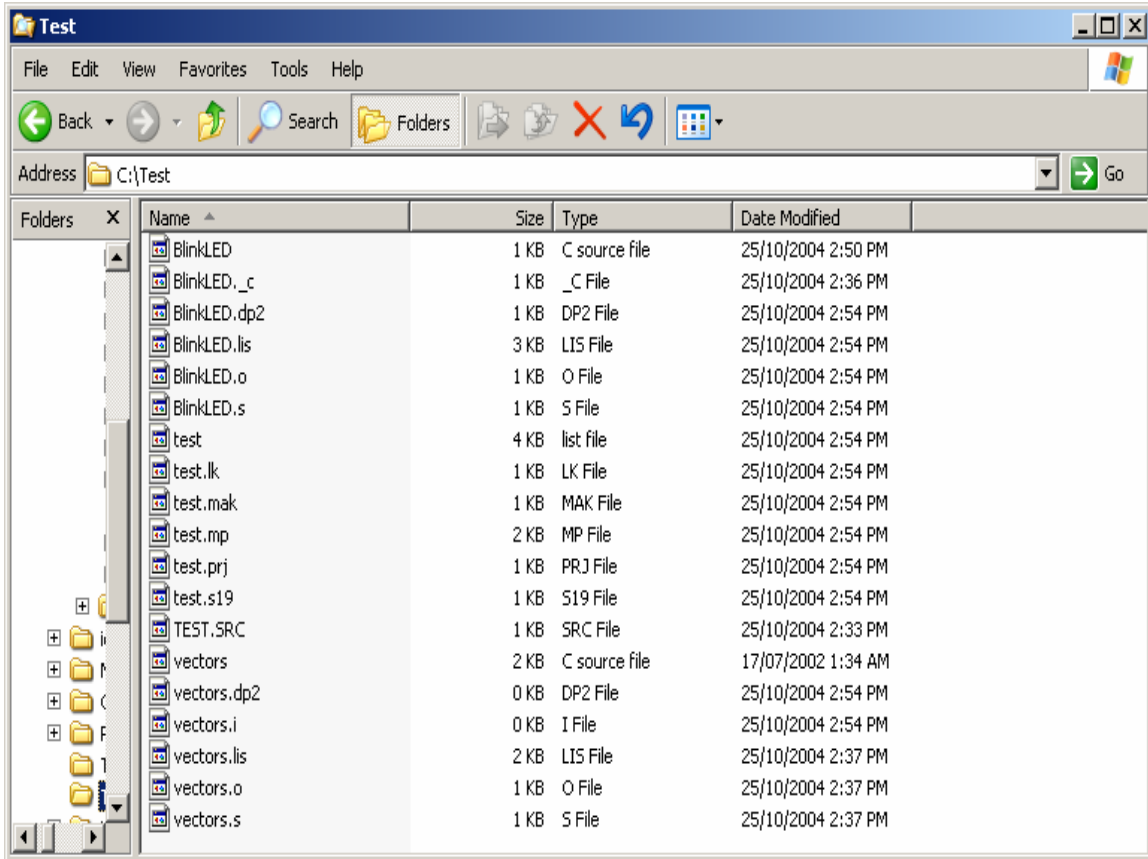
To make the file click Project menu – make project as shown.



Note the bottom window pane will show messages to display how the build progressed. Any errors, if any, are shown in this window. The build was without error so we can progress to erasing and programming the 9S12C32.



Note the other extraneous files are created after a make.



Using WordPad to check the content of test.s19

```
S10E400CF400016407087CE38008EC1
S110400B380027056A000820F6CE4075CD68
S111401838008E40752706180A307020F51601
S1074026402A20FE0A
S110402A34B7751B9EC6FF7B0242C6FF7BA8
S11040370240164052200EC6FF7B024016C8
S1114044405279024016405220F0B757303DEA
S111405234B7751B9ECC00006C1E2007EC1EBC
S1114060C300016C1EEC1E8CFA0025F2B7574B
S105406E303DDF
S111FFD0FFFFFFFFFFFFFFFFFFFFFFFF2D
S111FFDEFFFFFFFFFFFFFFFFFFFFFFFF1F
S111FFECFFFFFFFFFFFFFFFFFFFFFFFF11
S109FFFAFFFFFFFF4000C1
S10840701D0016073DD0
S9034000BC
```

If one looks closely at the S-record one can see the S1 records are programmed into the **\$4000 - \$7FFF** and **\$C000 - \$FFFF** memory blocks. The main reason has to do with uBUG12. It adjusts the memory configuration into 2 separate blocks.

Below is the vector address as will be programmed into \$C000 to \$FFFF block.

```
S111FFD0FFFFFFFFFFFFFFFFFFFFFFFFFFFF2D
S111FFDEFFFFFFFFFFFFFFFFFFFFFFFFFFFF1F
S111FFECFFFFFFFFFFFFFFFFFFFFFFFFFFFF11
S109FFFAFFFFFFFF4000C1
```

Note the address at \$FFFE = \$4000, the start of code from power up or RESET.

```
S109FFFAFFFFFFFF4000C1
```

Below is the start of code address as will be programmed into the \$4000 to \$7FFF block.

```
S10E4000CF400016407087CE38008EC1
S110400B380027056A000820F6CE4075CD68
S111401838008E40752706180A307020F51601
S1074026402A20FE0A
S110402A34B7751B9EC6FF7B0242C6FF7BA8
S11040370240164052200EC6FF7B024016C8
S1114044405279024016405220F0B757303DEA
S111405234B7751B9ECC00006C1E2007EC1EBC
S1114060C300016C1EEC1E8CFA0025F2B7574B
S105406E303DDF
```

Below is the start of code address

```
S10E4000CF400016407087CE38008EC1
```



## Programming:

This document assumes that the Serial monitor is not erased and is present on the NC12.

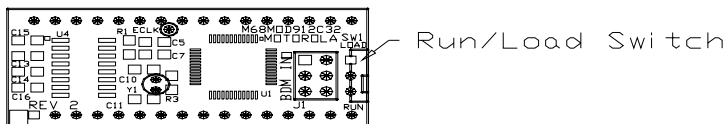
Download uBUG12 from Technological Arts which can be found at the link below <http://support.technologicalarts.ca/files/uBug12.zip>

or from the CD that came with the NC12 kit.

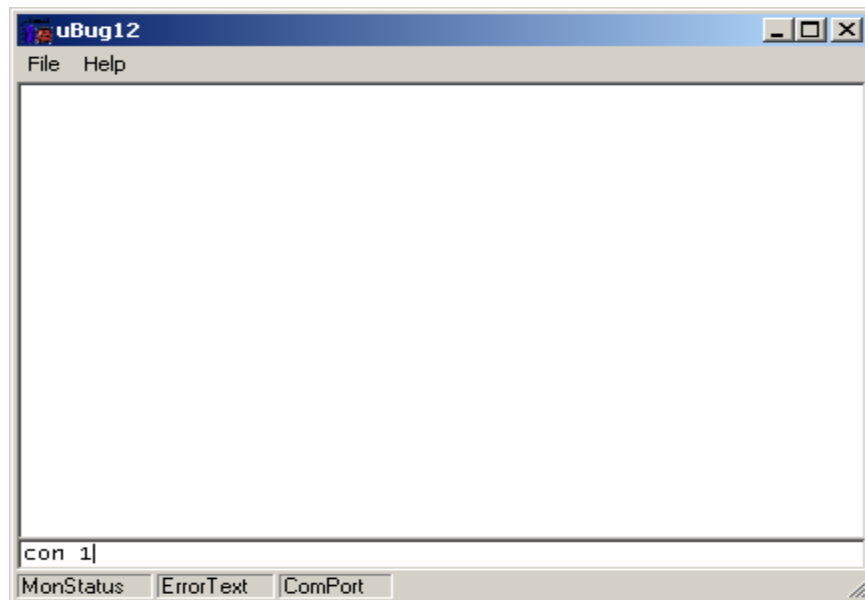
For windows 98 users the .NET framework must be installed before running uBUG12. The MS site link is

<http://www.microsoft.com/downloads/details.aspx?FamilyID=d7158dee-a83f-4e21-b05a-009d06457787&displaylang=en>

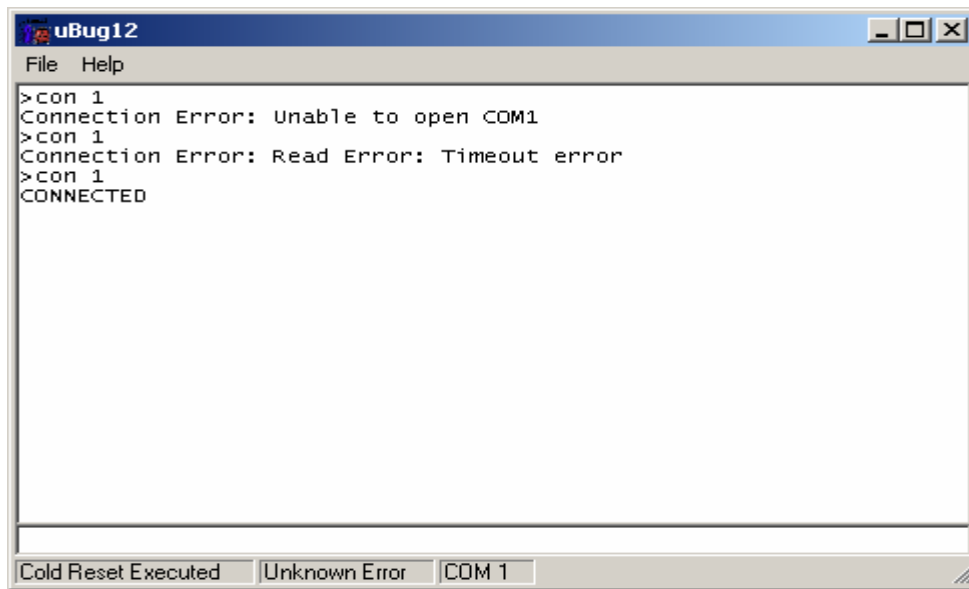
Switch the Run/Load switch to Load position and apply power to the board.



Double click on uBUG12 icon to execute program.



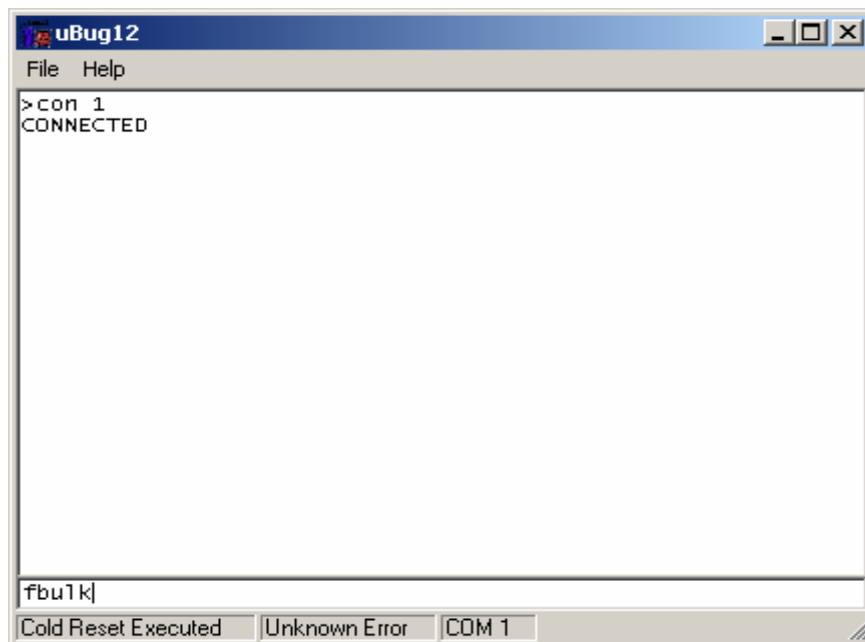
In the command bar type **con 1** for COM 1 or **con 2** for COM 2. A **CONNECTED** message will appear to indicate that a connection between PC and NC12 has been established.



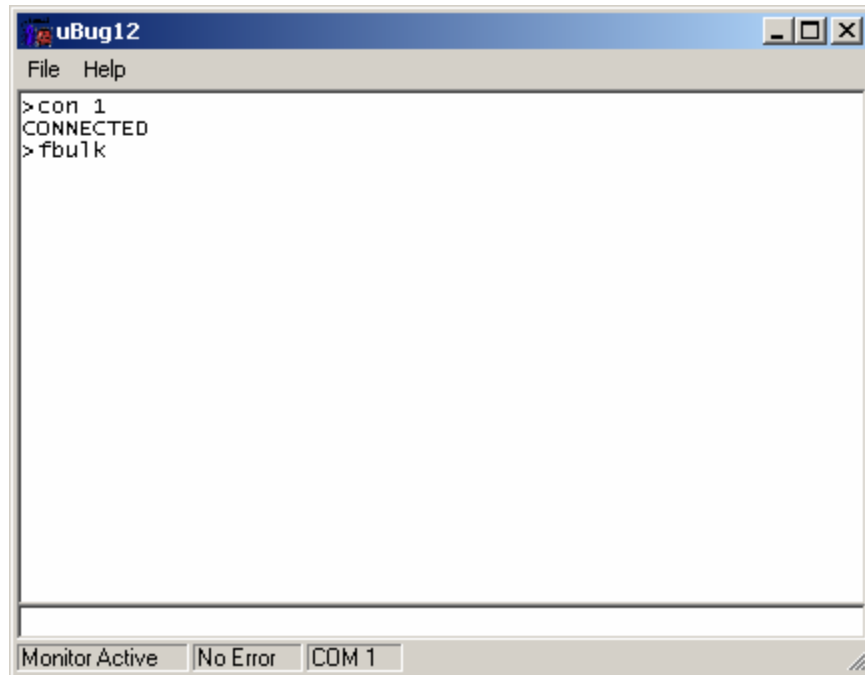
2 possible errors can occur:

**Connection Error: Unable to open COM1** <- Another application is using the COM port

**Connection Error: Read Error: Timeout error** <- The MCU not currently in LOAD mode or the cable is disconnected from either PC or Docking Module. Lastly, the serial cable is connected to the wrong COM port.



Make sure to erase the FLASH memory by typing the command **FBULK**.



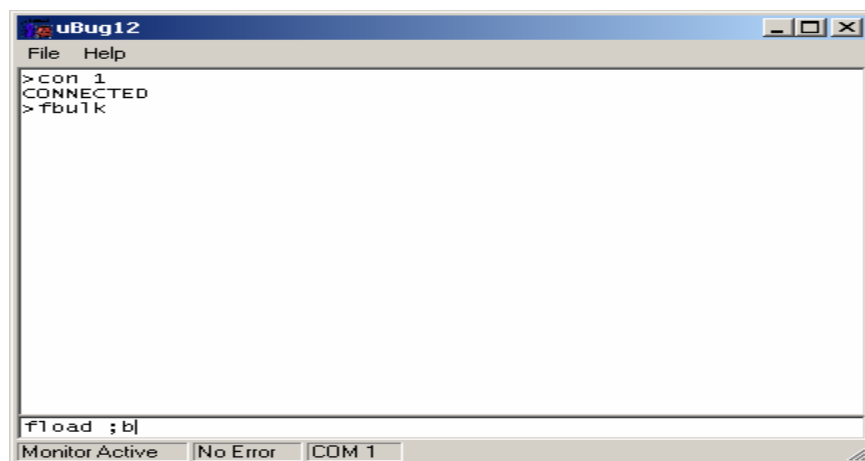
```
uBug12
File Help
> con 1
CONNECTED
> fbulk

Monitor Active No Error COM 1
```

To program, type the command **FLOAD ;B** for banked S19, S2, SX and formatted S19 (went thru SrecCVT program) records. For non-banked S2 the command is **FLOAD**.

### Uploading Banked S-record:

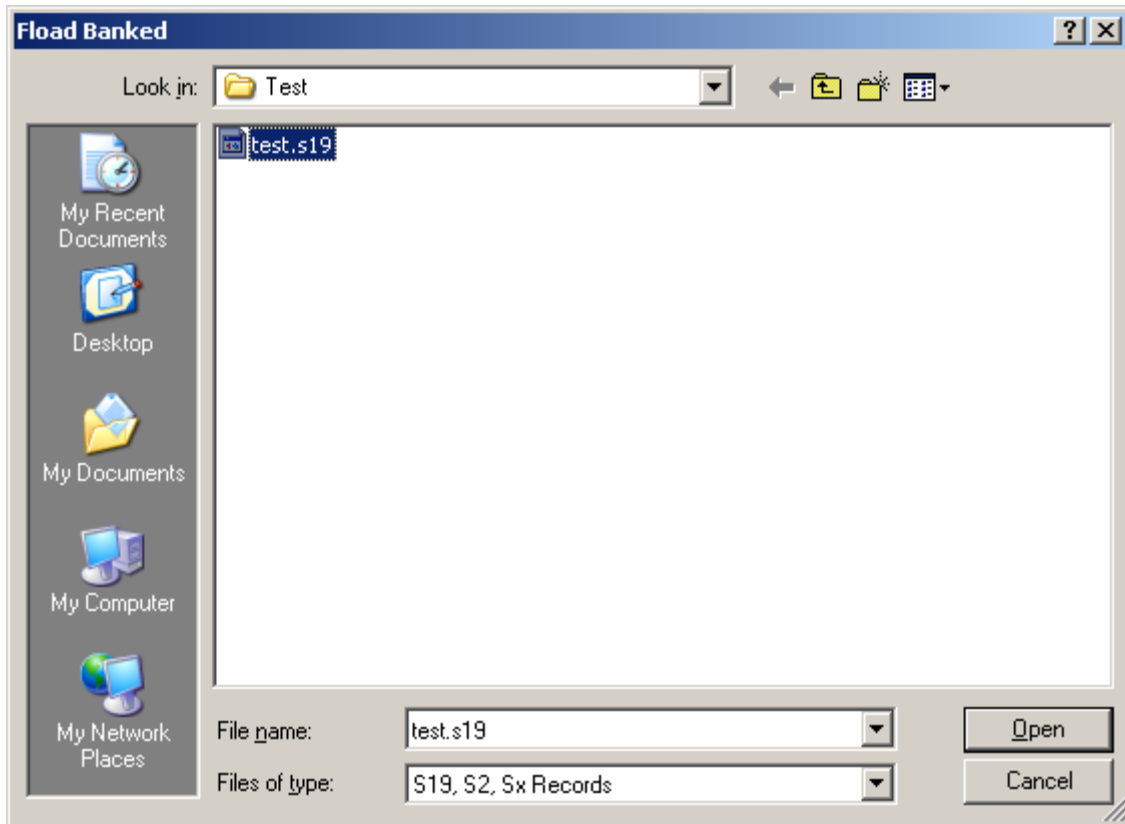
The command to upload banked S-record is **FLOAD ;B**. Note to include the **;B** option to let uBUG12 know that the S-record is banked. Users should be familiar with the differences between S19, SX, S2. See Appendix A for S-record explanation.



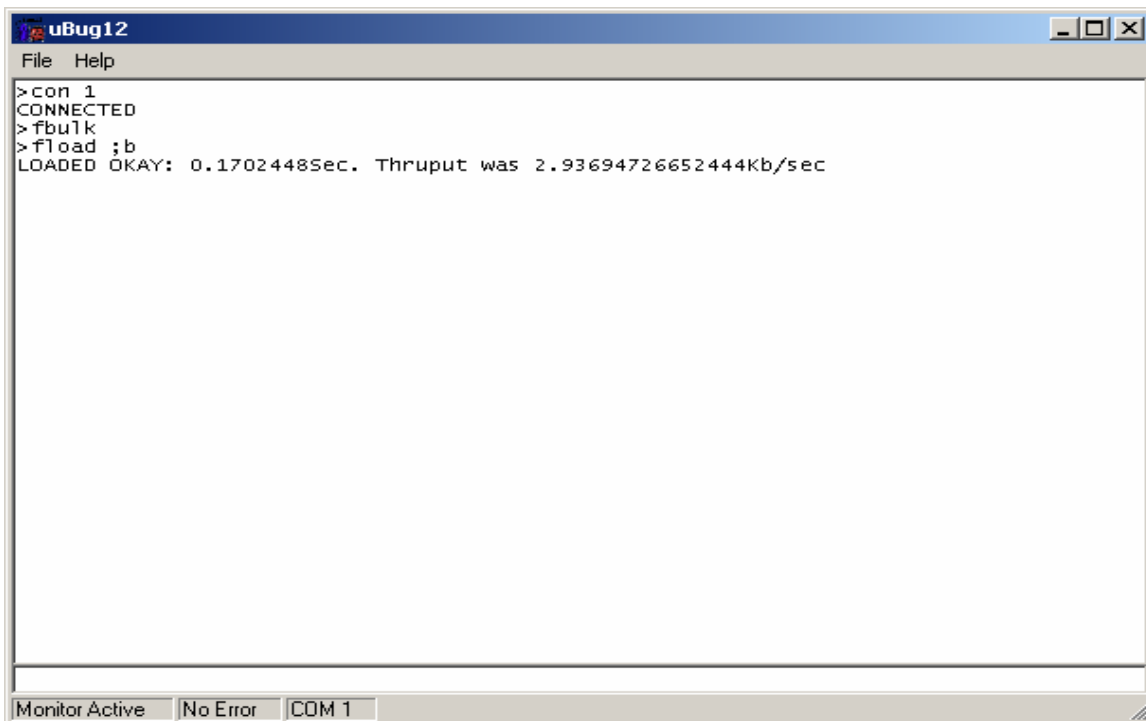
```
uBug12
File Help
> con 1
CONNECTED
> fbulk

Fload ;b

Monitor Active No Error COM 1
```



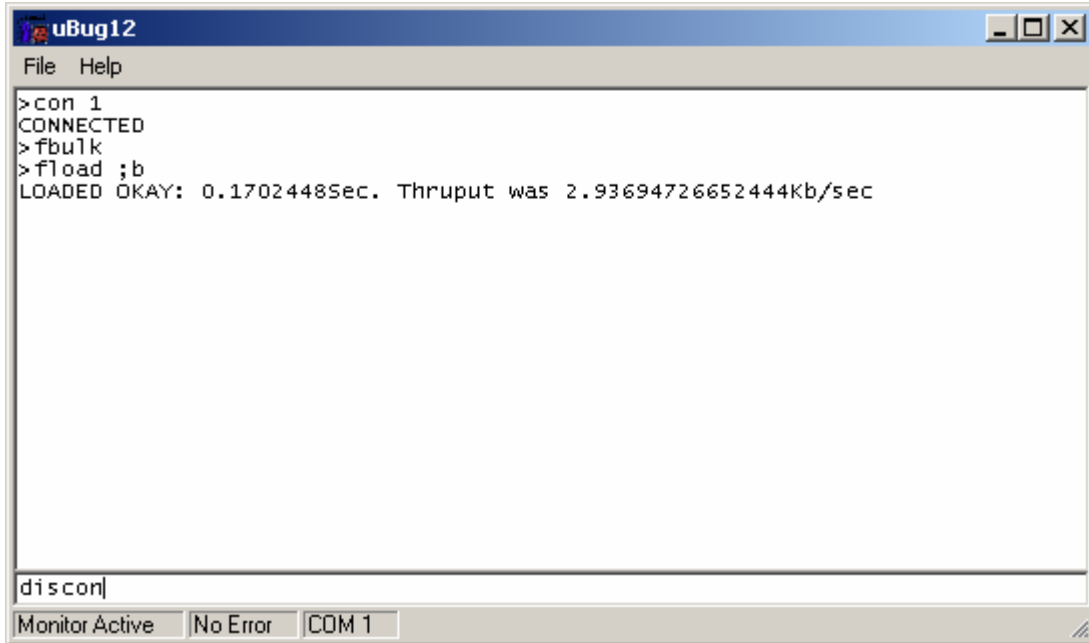
Double click on the file to initiate upload.



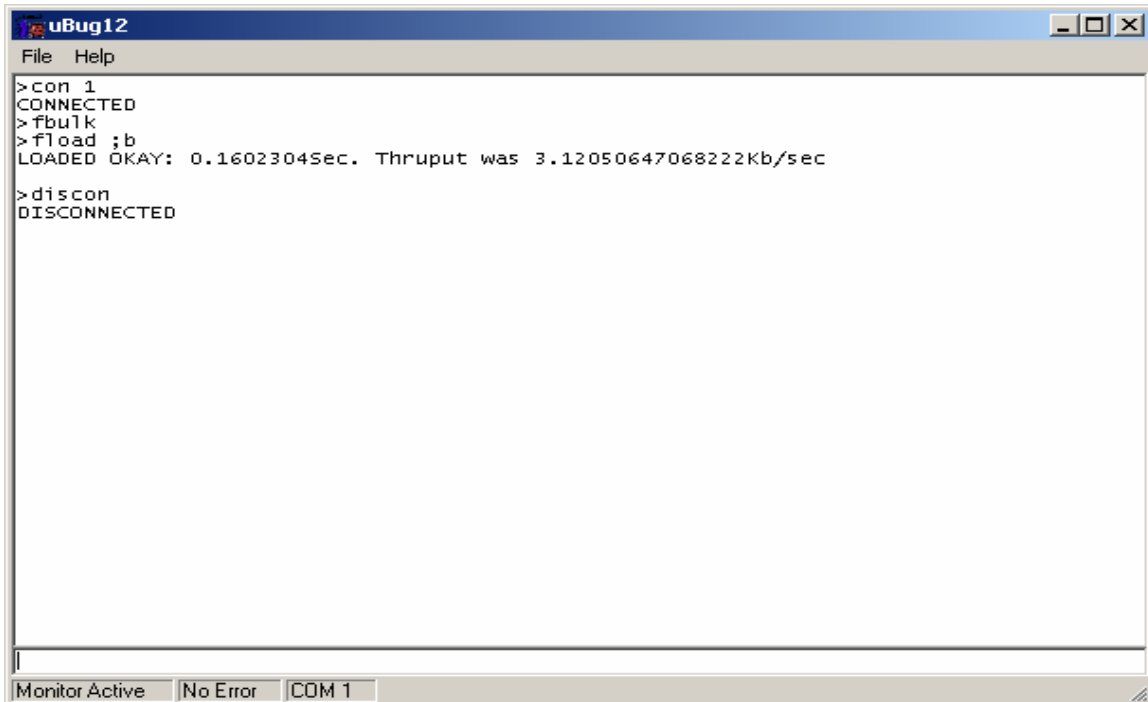
A good upload will show **LOADED OKAY** messages.

After successful programming slide the Run/Load switch to Run and press the reset button. The application is blinking the LED connected to at PP0

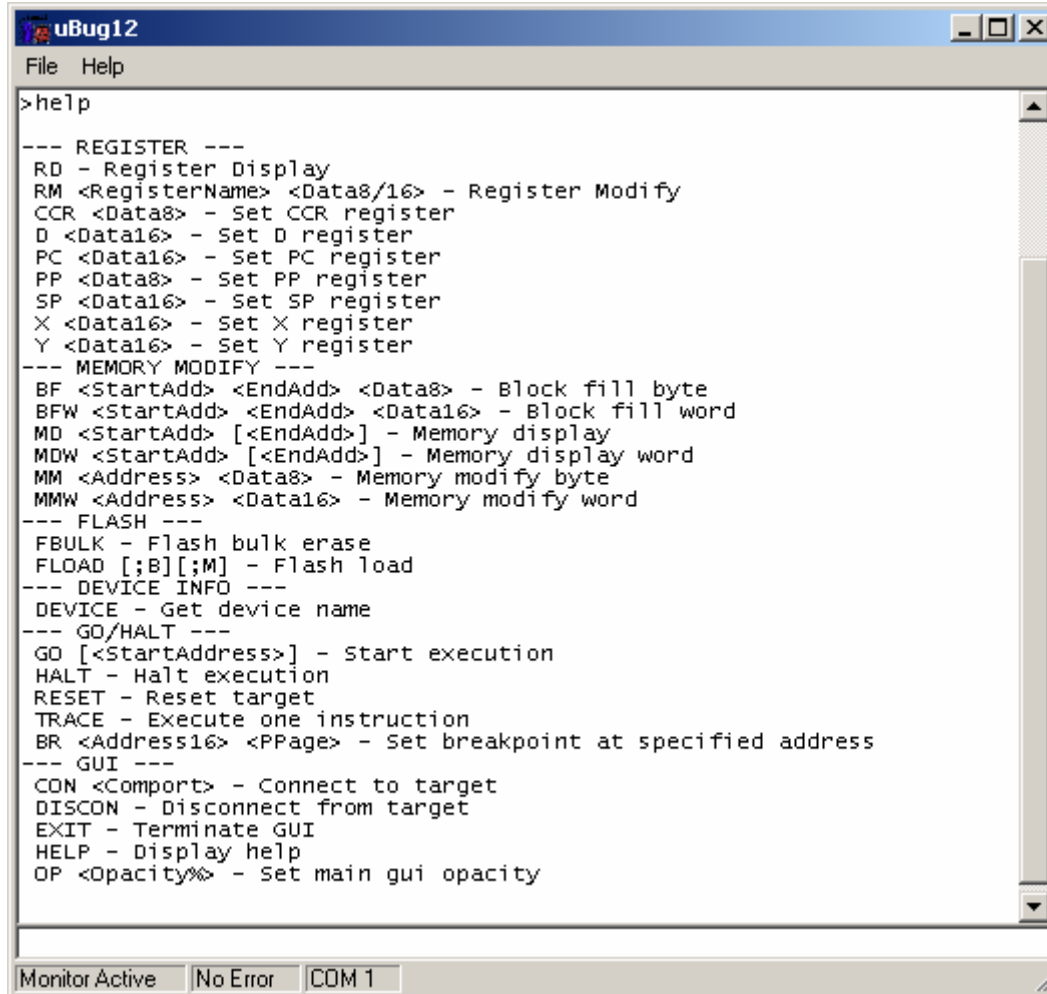
To disconnect uBUG12 GUI to serial port type the command is **discon**.



Disconnected message will appear to indicate that the serial is ready to be use by another application like **HyperTerm** or **Tera Term**.



Other uBUG12 commands are available by typing the **help** command.



The screenshot shows a window titled "uBug12" with a menu bar containing "File" and "Help". The main area displays the output of the ">help" command. The output is organized into several sections: REGISTER, MEMORY MODIFY, FLASH, DEVICE INFO, GO/HALT, and GUI. Each section lists various commands with their syntax and descriptions. At the bottom of the window, there are three status indicators: "Monitor Active", "No Error", and "COM 1".

```
>help
--- REGISTER ---
RD - Register Display
RM <RegisterName> <Data8/16> - Register Modify
CCR <Data8> - Set CCR register
D <Data16> - Set D register
PC <Data16> - Set PC register
PP <Data8> - Set PP register
SP <Data16> - Set SP register
X <Data16> - Set X register
Y <Data16> - Set Y register
--- MEMORY MODIFY ---
BF <StartAdd> <EndAdd> <Data8> - Block fill byte
BFW <StartAdd> <EndAdd> <Data16> - Block fill word
MD <StartAdd> [<EndAdd>] - Memory display
MDW <StartAdd> [<EndAdd>] - Memory display word
MM <Address> <Data8> - Memory modify byte
MMW <Address> <Data16> - Memory modify word
--- FLASH ---
FBULK - Flash bulk erase
FLOAD [;B];[M] - Flash load
--- DEVICE INFO ---
DEVICE - Get device name
--- GO/HALT ---
GO [<StartAddress>] - Start execution
HALT - Halt execution
RESET - Reset target
TRACE - Execute one instruction
BR <Address16> <PPage> - Set breakpoint at specified address
--- GUI ---
CON <Comport> - Connect to target
DISCON - Disconnect from target
EXIT - Terminate GUI
HELP - Display help
OP <Opacity%> - Set main gui opacity

Monitor Active  No Error  COM 1
```

The command help are self explanatory but one should try them out to be familiar with their usage and capability.

**Note: For NC12 families**

Note that the Serial Monitor resides at \$F800 - \$FFFF. Therefore SerialMon will automatically re-locate the vector addresses at below \$F800.

SerialMon moves the internal RAM to \$3800 - \$3FFF. Make sure your code stack begins at \$4000 or at \$3F80 if you intend to use uBUG12 as limited debugger. To make sure this is done you can add the code below to your code.

```
STACK          equ    $3F80          ;Stack at below Ubug12

movb    #$00,INITRG          ;set registers at $0000
movb    #$39,INITRM          ;move and set ram to end at $3fff
```

The last thing to note is that uBUG12 enabled the PLL during Load mode. In Run mode the PLL is NOT enabled as the user maynot want this feature enabled. The code below shows how to enable the PLL.

```
OscFreq      equ      8000          ;Enter Osc speed
initSYNR     equ      $02          ; mult by synr + 1 = 3 (24MHz)
initREFDV    equ      $00          ;
PLLSEL       equ      %10000000    ;PLL select bit
LOCK         equ      %00001000    ;lock status bit
PLLON        equ      %01000000    ;phase lock loop on bit

; Initialize clock generator and PLL
bclr  CLKSEL,PLLSEL      ;disengage PLL to system
bset  PLLCTL,PLLON      ;turn on PLL

movb  #initSYNR,SYNR     ;set PLL multiplier
movb  #initREFDV,REFDV   ;set PLL divider

nop
nop
nop
nop

brclr  CRGFLG,LOCK,*+0   ;while (!(crg.crgflg.bit.lock==1))
bset  CLKSEL,PLLSEL     ;engage PLL to system
```

### Examining the content of the MCU:

Below is the S-record (test.s19) that was programmed into the NC12.

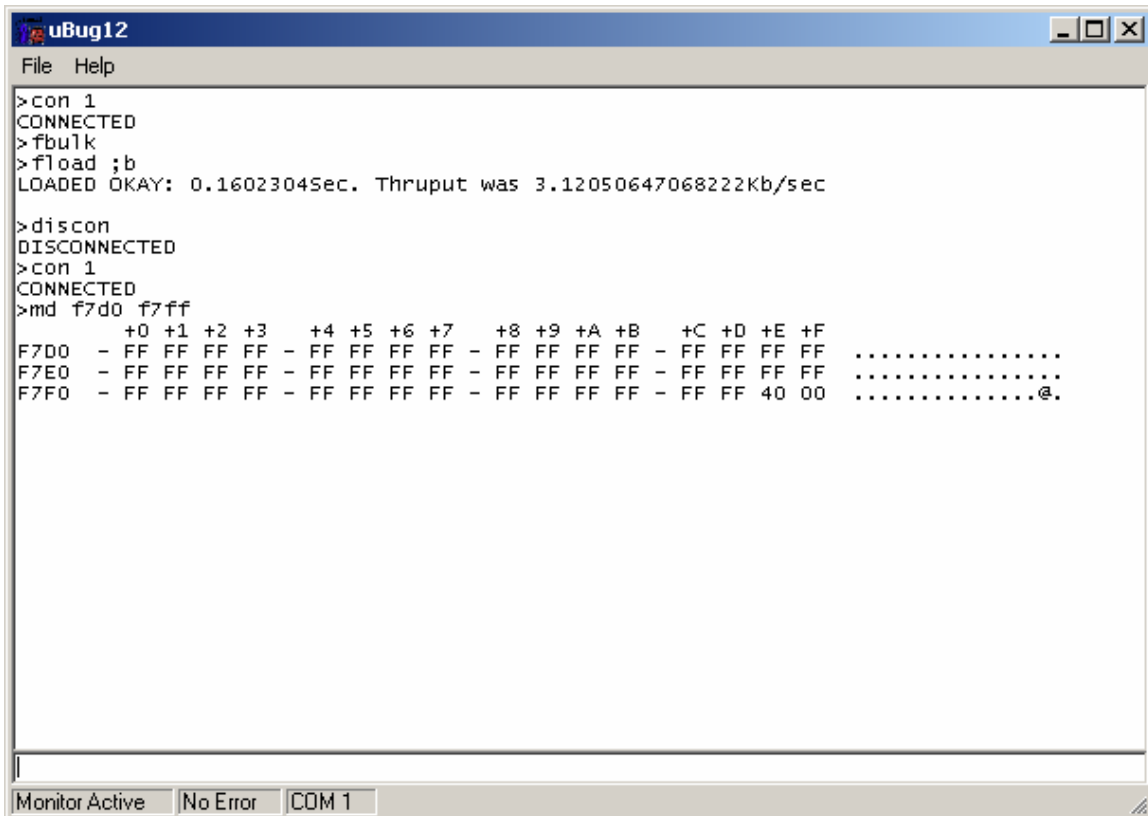
```
S10E4000CF400016407087CE38008EC1
S110400B380027056A000820F6CE4075CD68
S111401838008E40752706180A307020F51601
S1074026402A20FE0A
S110402A34B7751B9EC6FF7B0242C6FF7BA8
S11040370240164052200EC6FF7B024016C8
S1114044405279024016405220F0B757303DEA
S111405234B7751B9ECC00006C1E2007EC1EBC
S1114060C300016C1EEC1E8CFA0025F2B7574B
S105406E303DDF
S111FFD0FFFFFFFFFFFFFFFFFFFFFFFFFFFF2D
S111FFDEFFFFFFFFFFFFFFFFFFFFFFFFFFFF1F
S111FFECFFFFFFFFFFFFFFFFFFFFFFFFFFFF11
S109FFFAFFFFFFFF4000C1
S10840701D0016073DD0
S9034000BC
```

These are the area of interest where the S-record is programmed to. Let us start with the interrupt vector area. As stated previously, Serial Monitor re-locates the vector address at below \$F800.

```
S111FFD0FFFFFFFFFFFFFFFFFFFFFFFF2D
S111FFDEFFFFFFFFFFFFFFFFFFFFFFFF1F
S111FFECFFFFFFFFFFFFFFFFFFFFFFFF11
S109FFFAFFFFFFFF4000C1
```

Use uBUG12 to memory dump from \$F7D0 to \$F7FF by the command ***md f7d0 f7ff***.

Note the content of the memory address at \$F7FE:\$F7FF is \$4000, the RESET vector.

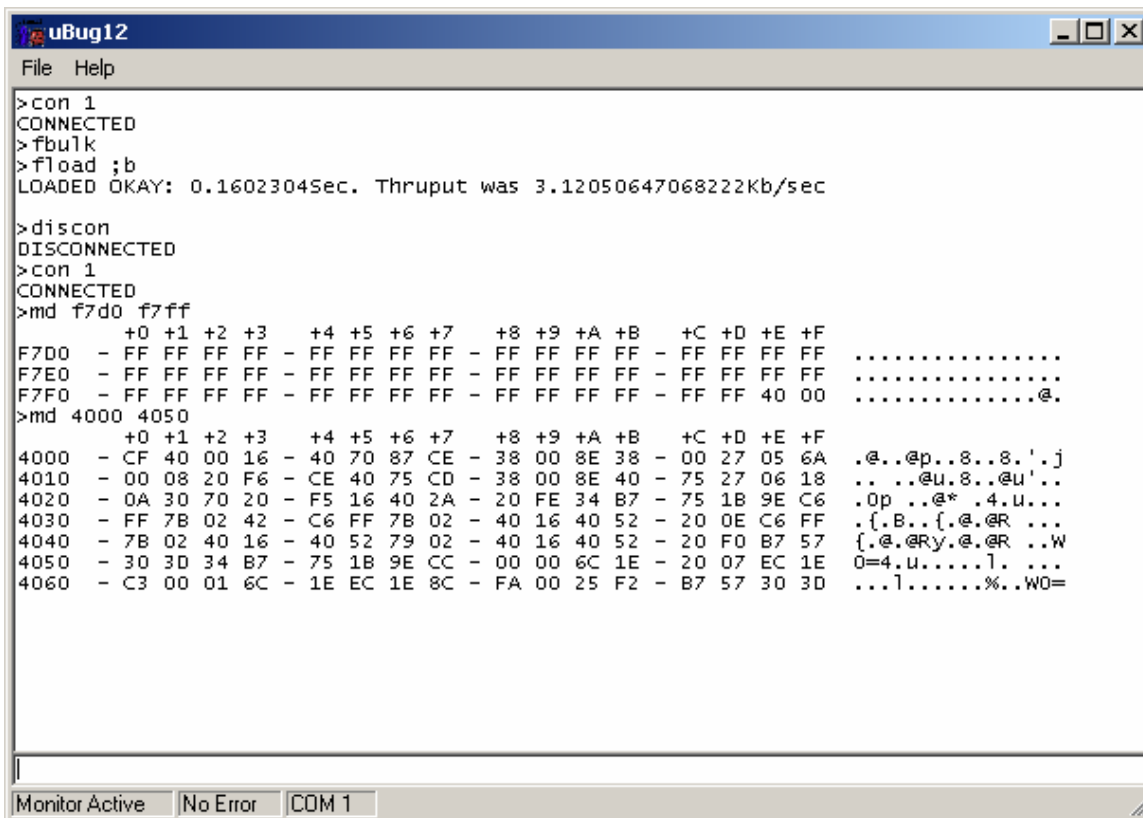




The content of address beginning at \$4000 to \$4050

```
S10E400CF400016407087CE38008EC1
S110400B380027056A000820F6CE4075CD68
S111401838008E40752706180A307020F51601
S1074026402A20FE0A
S110402A34B7751B9EC6FF7B0242C6FF7BA8
S11040370240164052200EC6FF7B024016C8
S1114044405279024016405220F0B757303DEA
S111405234B7751B9ECC00006C1E2007EC1EBC
S1114060C300016C1EEC1E8CFA0025F2B7574B
S105406E303DDF
```

Note that the memory dump is the same as the S-record.



The screenshot shows the uBug12 software interface. The main window displays a command-line interface with the following text:

```
>con 1
CONNECTED
>fbulk
>fload ;b
LOADED OKAY: 0.1602304Sec. Thruput was 3.12050647068222Kb/sec

>discon
DISCONNECTED
>con 1
CONNECTED
>md f7d0 f7ff
+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F
F7D0 - FF FF FF FF - FF FF FF FF - FF FF FF FF - FF FF FF FF .....
F7E0 - FF FF FF FF - FF FF FF FF - FF FF FF FF - FF FF FF FF .....
F7F0 - FF FF FF FF - FF FF FF FF - FF FF FF FF - FF FF 40 00 .....@.
>md 4000 4050
+0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +A +B +C +D +E +F
4000 - CF 40 00 16 - 40 70 87 CE - 38 00 8E 38 - 00 27 05 6A .@..@p..8..8.'.j
4010 - 00 08 20 F6 - CE 40 75 CD - 38 00 8E 40 - 75 27 06 18 .. ..@u.8..@u'..
4020 - 0A 30 70 20 - F5 16 40 2A - 20 FE 34 B7 - 75 1B 9E C6 .0p ..@* .4.u...
4030 - FF 7B 02 42 - C6 FF 7B 02 - 40 16 40 52 - 20 0E C6 FF .{.B...{.@.@R ...
4040 - 7B 02 40 16 - 40 52 79 02 - 40 16 40 52 - 20 F0 B7 57 {.@.@Ry.@.@R ..W
4050 - 30 3D 34 B7 - 75 1B 9E CC - 00 00 6C 1E - 20 07 EC 1E 0=4.u.....]. ...
4060 - C3 00 01 6C - 1E EC 1E 8C - FA 00 25 F2 - B7 57 30 3D ...]......%.W0=
```

At the bottom of the window, there are three status indicators: "Monitor Active", "No Error", and "COM 1".

This concludes the use of ICC12 from erasing and programming FLASH with using uBUG12.