

Run “Hello World” Example on EVBPlus2 Using ImageCraft’s ICC11 and Buffalo

Application Note AN0000

Author : Lin Zhao

Data : August 4, 2003

Abstract: The goal of this application note is to help you get started using ICC11 with your EVBPlus2 evaluation board that boots from Buffalo monitor. A simple “Hello World” program was used as the example and three options to run the “Hello World” program on EVBPlus2 evaluation board were introduced. The first option, “call Buffalo’s Print Routine”, is applicable to most 68HC11 evaluation boards that boot from Buffalo monitor, although this application note was written for the EVBPlus2 board.

In this application note, three options to run “Hello World” example on the EVBPlus2 evaluation board that boots from Buffalo monitor will be introduced. The ICC11 V6 from ImageCraft was used as the C compiler.

Notice that you only need to add your C source file to your project when you want your program to run under Buffalo monitor. DO NOT include “vectors.c” in your project.

1. Call Buffalo’s Print Routine:

There was a “Hello World” example that came with ICC11 V6. You can find it at “c:\icc\examples.11\hello.c” if you have already installed ICC11 V6 on your C drive. The following steps will help you get this example running on your EVBPlus2 board.

(1) Create a Project:

Once you invoke the IDE, choose Project -> New from the menu system. Navigate to the “c:\icc\examples.11\” and type in a project name, such as “hello.prj” and click “Save”.

(2) Add file “hello.c”:

In the project window, which is on the right side of the IDE, right click on “Files” and click “Add File(s)...”. In “Add Files...” dialog frame, choose “hello.c” and click “Open”. Now, the C file is added to the project.

(3) Set Options:

Choose Project -> Option from the menu system. The “Compiler Options” dialog window appears as shown in Figure 1.

Under the “Target” tab, select “Custom” as the target processor under “Device Configuration”. Under “Memory Addresses”, set “Program Memory” as 0xC000, set “Data Memory” as 0x0000 and set “Stack Pointer” as 0x01FF. The settings are shown in Figure 2.

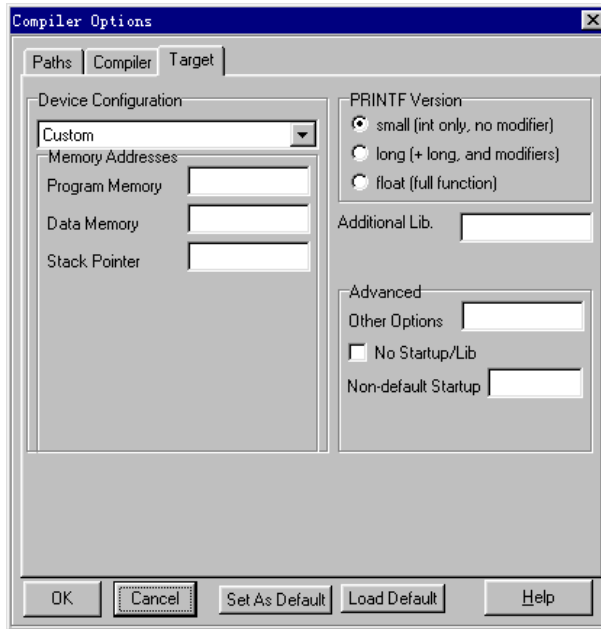


Figure 1 Compiler Options

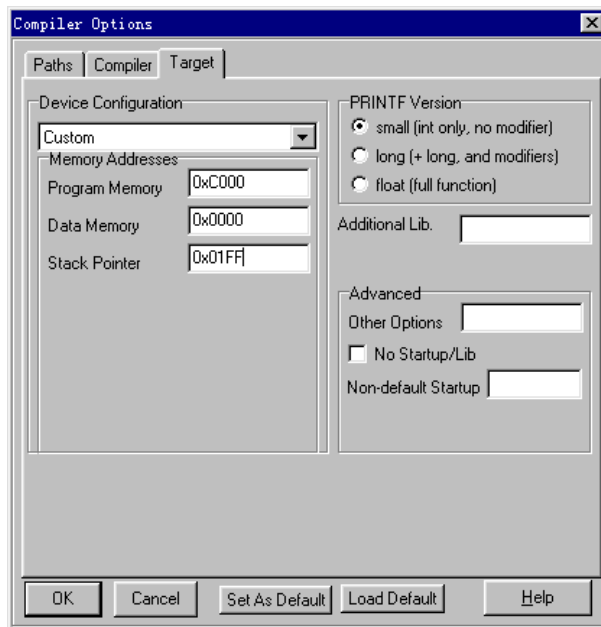


Figure 2 Set Target

Next, click on “Compiler” tab. Make sure “Accept Extensions (C++ comments, binary constants)” is checked as shown in Figure 3.

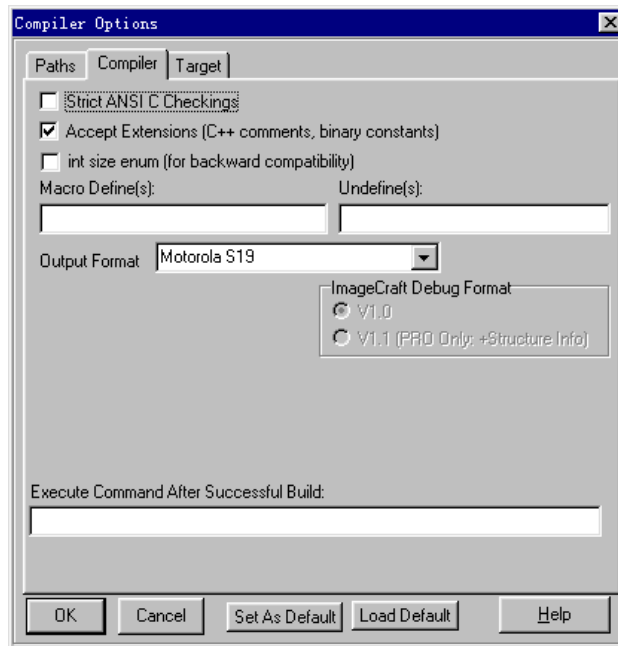


Figure 3 Set Compiler

Finally, click on “Paths” tab. Make sure the “Include Path” and the “Library Path” are correctly set as shown in Figure 4. Then, click “OK”.

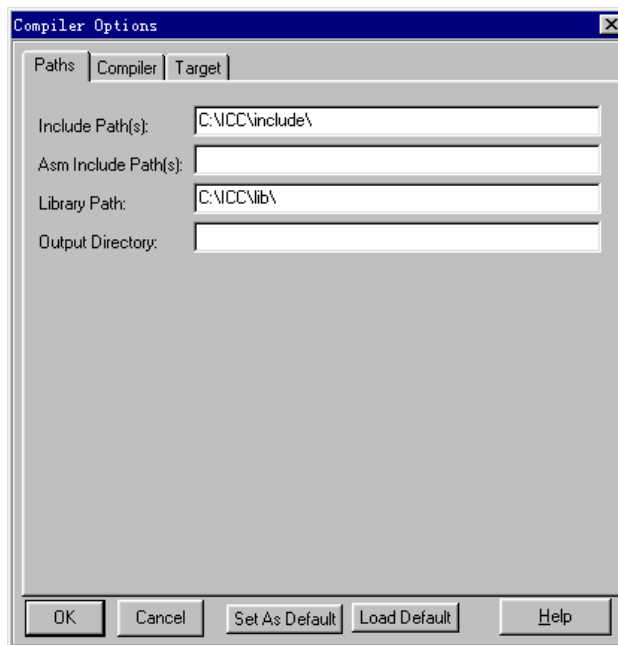


Figure 4 Set Paths

(4) Build the Project:

Before building the project, we need to change “# if 0” to “# if 1” to activate the “putchar” function in the source file “hello.c” because we are going to use Buffalo’s print routine. In putchar function, the Buffalo’s putchar subroutine is called to print out characters on the PC screen. Please refer to the comments in the source file for detail.

Now, choose Project -> Make Project or press F9 to build the project. You should have no problem to have the program compiled and get a S19 file in your folder.

(5) Load and Run the Program:

Start “AsmIDE” by double clicking on “AsmIDE” which is under “Ep2IDE” folder (you can also use your own terminal program such HyperTerminal and set the serial port as 9600, 8 data bits, no parity and 1 stop bit).

Once AsmIDE is started, click on “Terminal” tab. Then, press the reset button on the EVBPlus2 and hit “Enter” on your keyboard. Now, the AsmIDE terminal window should be same as Figure 5.

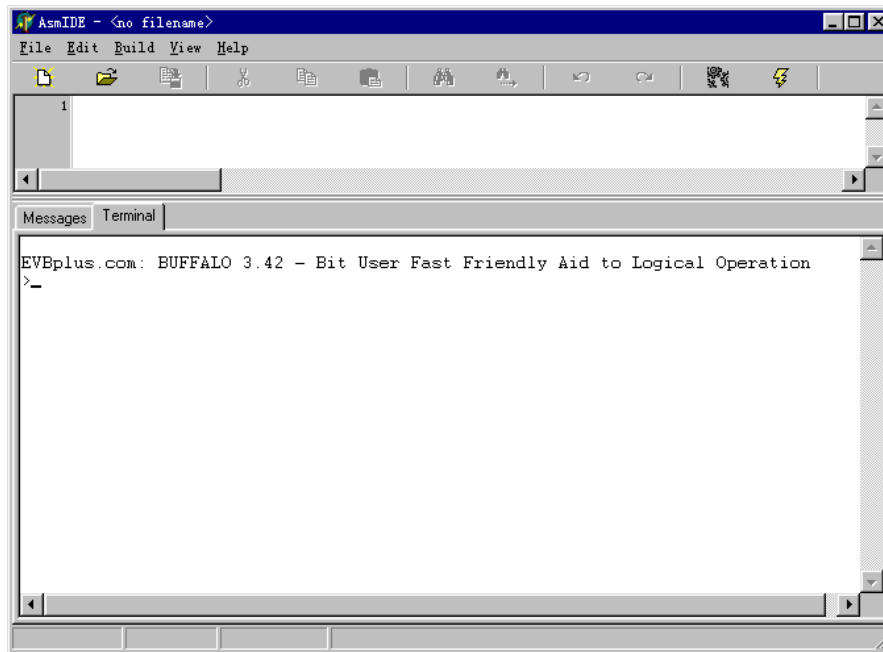


Figure 5 AsmIDE Terminal Window

Type “Load T” in the terminal window and hit “Enter” on your keyboard. Then, choose Build -> Download from the menu system. Navigate to c:\icc\examples.11, select “hello.s19” and click “Open”.

After the S19 file is loaded to EVBPlus2, type “g c000” in the terminal window to run the program. You got it! “Hello World” is displayed in the terminal window now, just like what you can see in Figure 6.

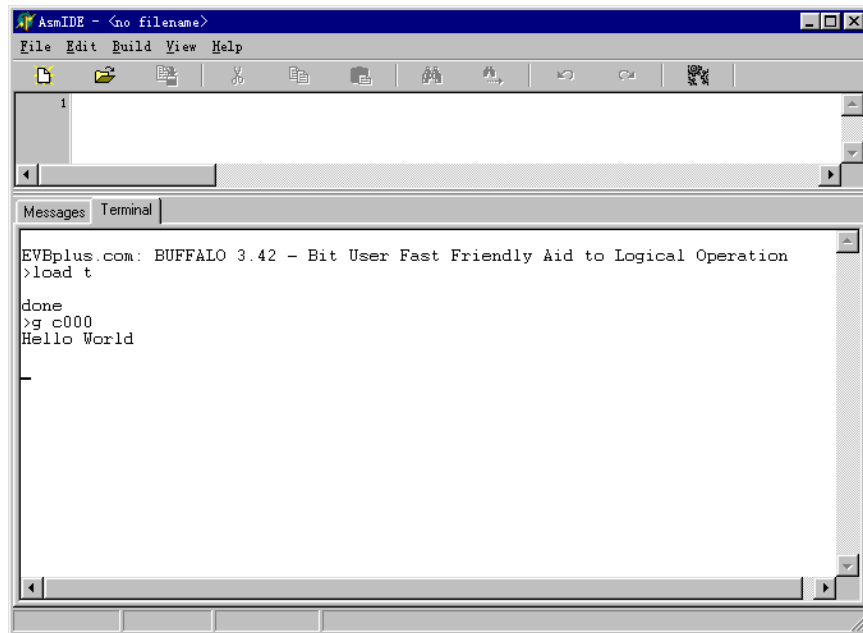


Figure 6 "Hello World"

2. Write Your Own Print Routine:

If you do not like to call Buffalo’s print subroutine, you can write your own. Since the MC6850 UART, an Asynchronous Communication Interface Adapter (ACIA), is directed to P2 on EVBPlus2 board, we have to redirect the SCI serial output from 68HC11 microcontroller to P2. This can be easily done by defining the ACIA at the beginning of the C program:

```
#define ACIA_CTRL *(unsigned char volatile *) (0x6400)  
#define ACIA_STAT *(unsigned char volatile *) (0x6400)  
#define ACIA_XMIT *(unsigned char volatile *) (0x6401)  
#define ACIA_RCV *(unsigned char volatile *) (0x6401)
```

(Note: if you use the FOX11 board, replace 0x6400 with 0x2400, and replace 0x6401 with 0x2401:

```
#define ACIA_CTRL *(unsigned char volatile *) (0x2400)  
#define ACIA_STAT *(unsigned char volatile *) (0x2400)  
#define ACIA_XMIT *(unsigned char volatile *) (0x2401)  
#define ACIA_RCV *(unsigned char volatile *) (0x2401)  
)
```

Then, add the following “getchar”, “putchar” and “my_print” functions to your C program:

```

char getchar()
{
char ch;

while (!(ACIA_STAT & 0x01)); /* wait until receive data register is full */
ch = ACIA_RCV & 0x7F; /* mask out parity bit */
return ch;
}

void putchar(char ch)
{
while (!(ACIA_STAT & 0x02)); /* wait until transmit data register is empty */
ACIA_XMIT = ch;
}

void my_print (char *msg)
{
while (*msg !=0)
putchar (*msg++);
}

```

The complete “Hello World” C program is shown in Listing 1. Refer to part 1, Call Buffalo’s Print Routine, for the steps of building your project and running the application on the EVBPlus2 board.

Listing 1: “Hello World” C Program Using User-defined Print Routine

```

/*
* Program name: hello.c
* Author       : Lin Zhao
* Data        : August 1, 2003
* Description  : This program prints out "Hello World" on the PC screen.
*              : In this program, the SCI output of 68HC11 microcontroller
*              : is redirected to P2 on the EVBPlus2 board.
*/

#include <hc11.h>

/* Define ACIA */
#define ACIA_CTRL    *(unsigned char volatile *)0x6400
#define ACIA_STAT   *(unsigned char volatile *)0x6400
#define ACIA_XMIT   *(unsigned char volatile *)0x6401
#define ACIA_RCV    *(unsigned char volatile *)0x6401

char getchar();
void putchar(char ch);
void my_print(char *msg);

main()
{
    setbaud(BAUD9600);
    my_print("Hello World\n");
}

```

```

char getchar()
{
char ch;

while (!(ACIA_STAT & 0x01)); // wait until receive data is ready
ch = ACIA_RCV & 0x7F; // mask out parity bit
return ch;
}

void putchar(char ch)
{
while (!(ACIA_STAT & 0x02)); //wait until transmit data register is empty

ACIA_XMIT = ch;
}

void my_print(char *msg)
{
while(*msg !=0)
putchar(*msg++);
}

```

3. Modify ICC11's libc11.a:

You can also modify the ICC11's "libc11.a" so you do not need to define the ACIA and add "getchar", "putchar" and "my_print" routines in your source file. The "libc11.a" locates at "c:\icc\lib\".

First, compile the "iochar.c" shown in Listing 2 using ICC11. **Note: if you use the FOX11 board, replace 0x6400 with 0x2400, and replace 0x6401 with 0x2401:**

```

#define ACIA_CTRL *(unsigned char volatile *) (0x2400)
#define ACIA_STAT *(unsigned char volatile *) (0x2400)
#define ACIA_XMIT *(unsigned char volatile *) (0x2401)
#define ACIA_RCV *(unsigned char volatile *) (0x2401)

```

Listing 2: iochar.c

```

/*
* Program Name: iochar.c
* Author       : Wayne Chu
* Date        : July 28, 2003
*/

/* Define ACIA */
#define ACIA_CTRL *(unsigned char volatile *) (0x6400)
#define ACIA_STAT *(unsigned char volatile *) (0x6400)
#define ACIA_XMIT *(unsigned char volatile *) (0x6401)
#define ACIA_RCV *(unsigned char volatile *) (0x6401)

char getchar()
{

```

```

char ch;
while (!(ACIA_STAT & 0x01)); // wait until receive data is ready
ch = ACIA_RCV & 0x7F; // mask out parity bit
return ch;
}

void putchar(char ch)
{
while (!(ACIA_STAT & 0x02)); //wait until transmit data register is empty
ACIA_XMIT = ch;
}

```

Second, replace the “iochar.s” in “libc11.a” with the contents of “iochar.o”, which is the object file of “iochar.c”. Save the “libc11.a” at “c:\icc\lib\”.

Now, you just need to write a “Hello World” program as shown in Listing 3 and have it compiled and run on the EVBPlus2 board by following the instructions in part 1, Call Buffalo’s Print Routine.

Listing 3: “Hello World” C Program

```

#include <hc11.h>

main()
{
setbaud(BAUD9600);
printf("Hello World\n");
}

```