

BCX BASIC for the Wii Tutorial

This tutorial explains how to develop applications for the Wii using BCX BASIC to convert your basic code to C.

BCX is a small command line tool that inputs a BCX BASIC source code file and outputs a C source code file which can be compiled with many C or C++ compilers. There is no gcc version of BCX, however, so you can only use a subset of the available commands. I made a BCXWII.INC file which provides special Wii functions like USB keyboard support.

More details about BCX BASIC can be found here: <http://bcx-basic.sourceforge.net/> or here: <http://www.bcxgurus.com/> and especially the BCX help file.

This tutorial covers first how to setup a development environment for the Wii in C with the devkitpro package. Then it explains how to download BCX and install that into this development environment and how to compile and run the samples included with this tutorial.

I hope this tutorial gets you started if you are interested writing BCX BASIC code for the Wii.

1.) Setup the Wii to run your applications

To run your own applications on a Wii you have to prepare the Wii for this first.

The central site with all the information about running non-Nintendo software on the Wii is: www.wiibrew.org

a) The first thing you have to do is install the Twilight Hack.

This will allow you to start any application from a SD card in your Wii. You need the game „The Legend of Zelda“ for this. You could borrow that from a friend since you will not need it any more once you installed the Homebrew channel as described later.

Follow the instructions on this page to install the Twilight Hack: http://www.wiibrew.org/wiki/Twilight_Hack

b) Once you installed the Twilight Hack you should install the homebrew channel on your Wii to simplify the way to run your applications on the Wii.

Here is how you install the homebrew channel: http://www.wiibrew.org/wiki/Homebrew_Channel

c) To run different versions of your program on the Wii for testing quickly you should install the Wiiloader utility too.

Wiiloader will allow you to send your new program versions to the Wii via the network (WLAN). So you do not have to write it on the SD card each time. You currently can only test your code on the Wii, so better put that next to your development PC.

Here is how to install Wiiloader: <http://www.wiibrew.org/wiki/Wiiloader>

2. Now install the development environment to develop applications for the Wii in the C language.

The way to develop applications for the Wii in C is by using the Devkitppc. This is a cross-compiler package which develops applications for the Wii on Windows to be transferred to the Wii for execution.

Instructions for installing this package plus compiling and running a „Hello World“ application you can find on this page:

http://www.wiibrew.org/wiki/Getting_Started_with_devkitppc

Now you can develop applications in C for the Wii. The next step is to use BCX to generate C code that can be used to develop an application for the Wii.

3. Download and install BCX BASIC

You can download BCX from this site: <https://sourceforge.net/projects/bcx-basic/>

The bc.zip file contains bc.exe plus the source code for it. Copy bc.exe into your HELLOWORLD development directory e.g. „c:\projects\wii\helloworld“ or in my case it is „d:\devkit-dev\helloworld“

You will also need the BCX help file. This is within this (large) package which you need to install on your PC: http://www.bcxgurus.com/bcx_install.exe

The BCX user group has some additional patches for bc.exe. I have not used these yet.

I made a batch file called lcall.bat to run BCX and then do some conversions on the C file BCX has generated. This is this batch file which is included in the ZIP file you downloaded:

```
bc.exe source\%1
rem Make a small c at the end of the file - msys/make will not compile otherwise
rename source\%1.C %1.c
rem Remove some include files to avoid the warnings
ftool source\%1.c "#include <conio.h>" "" -l
ftool source\%1.c "#include <direct.h>" "" -l
ftool source\%1.c "#include <io.h>" "" -l
```

This batch file calls the ftool utility by Andreas Guenther to remove some include files in the C file generated by BCX.

The files bc.exe, ftool.exe and lcall.bat have to be placed into the HELLOWORLD directory we made during the C tests, e.g. „c:\projects\wii\helloworld“ or in my case below „d:\devkit-dev\helloworld“.

In the subdirectory SOURCE within this directory you should put your basic files. The lcall.bat file will then have bc.exe also put the generated C file into this directory.

Important note:

Keep just one C file in the source directory! The make file may compile and link all C files and this will cause linker errors.

You could now compile and run your code from the command line if you would enter:

Lcall hello -> compile hello.bas in subdirectory source to hello.c

Make -> compile hello.c with gcc to a hello.dol file

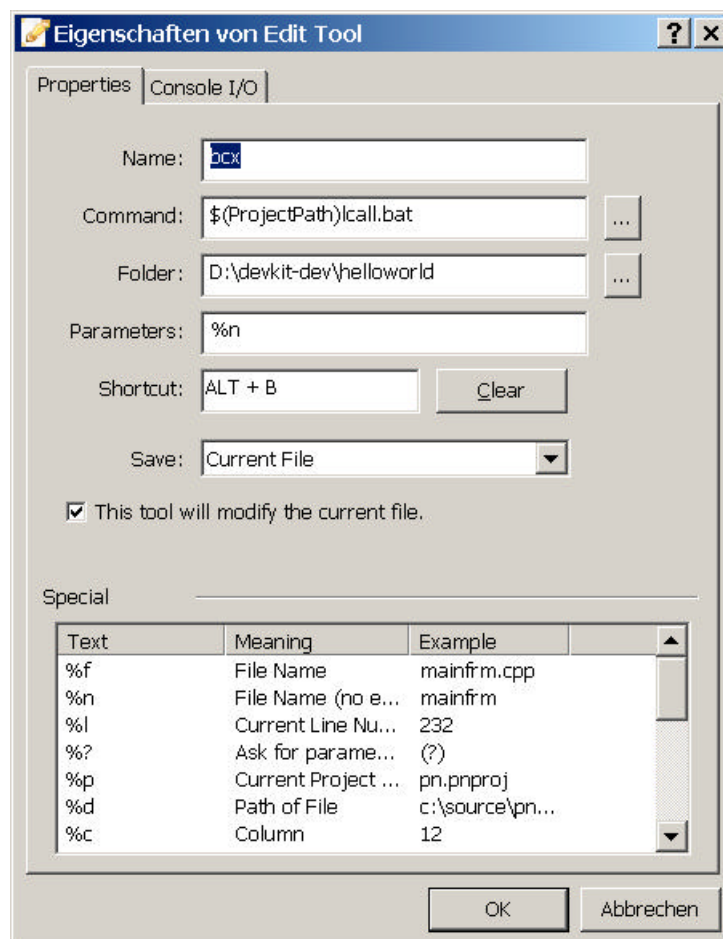
Make run -> use wiiloader to transfer hello.dol to the Wii

Usually you will use „programmers notepad“ to develop your BCX applications:

4. Add BCX to the programmers notepad to develop applications for the Wii.

You can load and develop a BCX basic program with the programmers notepad too. To compile that you should add an additional shortcut for BCX.

If you select Tools->Options you can add a new shortcut to run BCX. Here is how I set it up on my computer:



As you see the batch file lcall.bat is called to run BCX and then do some conversions on the C file BCX has generated.

After BCX generated a C file, you can select make and run from programmers notepad as before

when you compiled the helloworld sample in C. Make will compile the C code and run will call Wiiloader to transfer the program to the Wii. If you select BCX, programmers notepad will switch into a command prompt window which results in a lot of flicker on the screen!

You can now copy the samples provided with this tutorial into the source subdirectory and compile and run them this way. These samples are hello.bas, a „Hello World sample“, matches.bas and pong.bas two games using console mode and keytest.bas which needs an additional library as described in „Some general programming notes“.

To run your application from your SD card on the Wii, you should make a directory with the name of your application. Within this directory there should be your application as a boot.dol file. So you have to rename your e.g. hello.dol file in the helloworld directory (!) to boot.dol. Within this directory there should also be a PNG image file of the size 128x48 pixel to be used as the icon of your application and a meta.xml file with some information about your application. See other meta.xml files as templates. This directory then has to be copied into the „apps“ directory on your SD card.

If you want to publish your application on Wiibrew, you make a ZIP file of this directory and upload that to the Wiibrew site. You then should also write a web page describing your application and add it to the application list. See:

http://www.wiibrew.org/wiki/Homebrew_applications#Releasing

5. Some general programming notes.

Not all commands included in BCX will work with gcc. E.g. there are a number of BCX runtime functions that use optional arguments which gcc will not support. Many string functions will not work. So it would be better to generate a special version of BCX for gcc.

I made an include file named BCXWII.INC which provides some additional functions to support the Wii and some alternative string functions for those BCX functions which do not work with gcc. So instead of „if a\$=“test“ then“ you have to use the function wstrcmp: „if wstrcmp(ktest\$,“enter”) then“. Wstrcmp is short for Wii string compare.

Please take a look at this include file to see what you need to use with the Wii. The file includes the functions I needed for my projects so far. Additional functions are required, e.g. international keyboard support.

To make the samples compile without the libusbkbd library, the functions relating to that are commented out in the BCXWII.INC file. Uncomment them when using this keyboard library.

To use the keyboard support you have to download the libusbkbd library from the Wiibrew site. Copy the library libusbkbd.a into the \devkitPro\libogc\lib\wii directory and the kbd.h file into the \devkitPro\libogc\include directory.

Then edit the makefile in the HELLOWORLD directory to include the library libusbkbd.a. For this change the line:

```
LIBS := -lwiiuse -lbte -logc -lm
```

to

```
LIBS := -lwiiuse -lbte -logc -lm -lusbkbd
```

Observe that you have to omit „lib“ when using the `-l` parameter. With `#` you can comment out a line in the makefile. The result will be like this:

```
#-----  
# any extra libraries we wish to link with the project  
#-----  
#LIBS :=      -lwiiuse -lbte -logc -lm  
LIBS  :=      -lwiiuse -lbte -logc -lm -luskbd
```

You cannot just take any basic program and get that to run with BCX for the Wii. Compare that maybe trying to run an old Quickbasic program with Visual Basic.

BCX is a different BASIC dialect which differs from any other BASIC language. It is also case sensitive! For the Wii you can only use a subset of the BCX commands and have to add new ones for locate, color, keyboard input etc. These are partly provided in the BCXWII.INC file.

The error checking BCX provides is very limited. The errors are reported by gcc and you have to take a look at the C code to find out what needs to be changed in the basic code. So you should know a little bit of C to use BCX. After using BCX for a year you will be very familiar with C.

BCX allows me to develop applications faster than in C directly.

6. Graphics programming with BCX

Using GX which is part of the devkitpro toolchain for graphics programming is rather complicated. It is far easier to use the GRRLIB library which simplifies the use of GX.

GRRLIB has commands for displaying a pixel, draw a line, draw a rectangle, draw a Ngone, display a PNG image, draw characters selected from an image filled with letters of a particular font, display tiles of a PNG image and print using a font to a graphic screen. It does not support true type fonts.

Because of its simplicity we will use this library for our graphics programming now.

First you have to install this library properly into your environment. This is what you should do:

Copy the directories GFX, GRRLIB and pnglib into the source subdirectory within your Helloworld directory. Then copy the library libpng.a into the `\devkitPro\libogc\lib\wii` directory. Finally copy the files GRRLIB.C and GRRLIB_font1.C into the source subdirectory.

After that you have to edit the makefile in the Helloworld directory or even easier, replace it with the makefile within the GRRLIB package.

This has then changed the following lines to:

```
#SOURCES      :=      source  
SOURCES      :=      source source/gfx source/GRRLIB source/GRRLIB/fonts  
source/libpng source/libpng/pngu
```

And

```
LIBS := -lpng -lz -lfat -lwiiuse -lbte -logc -lm
```

So a number of directories are added to look for additional source files and the libpng.a is added to the list of libraries.

GRRLIB does not come as a compiled library, its files have to be compiled together with your program.

Compile e.g. the DAY7.c sample now from the GRRLIB site:

<http://grrlib.santo.fr/wiki/wikka.php?wakka=HomePage> to check that everything is setup properly.

If you want to display images, these have to be in the PNG format and their width and height have to be multiples of 4!!! Also the the maximum width of a PNG image in GRRLIB is 1024 pixel. In the devkitpro/devkitPPC/bin directory there is a program called raw2c.exe which will convert e.g. the MYIMAGE.PNG image into a myimage.c and a myimage.h file. These files have to be copied into your source directory and the statement #include "myimage.h" has to be added into the header of your program.

Also the following commands have to be included:

```
u8 *tex_logo=GRRLIB_LoadTexture(myimage);  
//has to point to your image and the command
```

```
GRRLIB_DrawImg(144, 176, 128, 48, tex_logo, rot, 1, 1, alpha );  
// needs the size of the PNG image as the third and fourth parameter (here: 128,48).
```

The sample DAY5.c sample from the GRRLIB site displays a PNG image of the GRRLIB logo. If you want to run this sample for a test copy the file logo.c from the gfx subdirectory into the source directory to compile. The files logo.c and logo.h are already there in the gfx subdirectory for this sample.

This tutorial contains two samples using GRRLIB: grrlibtest.bas and matchesgx.bas. Matchesgx.bas is a graphic version of the matches.bas sample we have seen above in console mode. The sample has been rewritten since for GRRLIB writing to the screen has to be done within a loop.

These samples show how to use GRRLIB with BCX BASIC. There are no special basic functions in the BCXWII.INC file since the functions from the GRRLIB library can be easily called from BCX BASIC. As you can see from the samples, the general font of GRRLIB calls for the support of better fonts.

If you need further graphics features, you can use the SDL library port for the Wii. This is a cross-platform graphics (and more) library which is used widely on many operating systems. Since it has many more features it is more complicated to use though.

6. Reading and writing files with BCX

These operations can be done on the SD card by using the LIBFAT library.

To use the SD card support you have to download the libfat library from the Wiibrew site. Copy the library libfat.a into the \devkitPro\libogc\lib\wii directory and the fat.h file into the

\devkitPro\libogc\include directory.

Then edit the makefile in the HELLOWORLD directory to include the library libfat.a. For this add the lib as the „-lfat“ parameter to the line:

```
LIBS := -lwiiuse -lbte -logc -lm -lfat
```

in the makefile. Within the program add #include <fat.h> and the additional statement
! fatInitDefault();

The sample libfattest.bas opens the file „test.txt“ in the current directory (will be the root when used with Wiiload) and writes a thousand lines into it. Then it closes this file, opens it again and prints these thousand lines to the screen.

That's all for now, I hope it was useful for you .

21th November 2008 Georg Potthast
Email: mailbox – at – georgpotthast.de