

About this document

This document is being written to give myself a one-stop reference that contains everything I need to know about the Gameboy Advance. The guide will be split into chapters and released slowly as and when I have completed a section.

If there is anything in the document that you feel is incorrect, written by someone else and not referenced, or if you have any comments then please contact me: gbajunkie@hotmail.com I would be pleased to hear from you. Although this is a personal reference document I am publishing it so it can help others, feedback can only help this process.

This document is Copyright © gbajunkie.co.uk 2002
Date written: 15th January 2002

Chapter 1: Setting Up

What is in this chapter

- An overview of the hardware and some general facts
- What hardware you need to see your own programs running on the console
- What software you need to compile your own programs
- A guide to setting up the GCC compiler

The Hardware and some facts

- The Gameboy Advance is a 32-bit system containing the ARM7TDMI RISC chip.
- The main processor runs at 16.78MHz
- The graphics processor accelerates many of the 2D processes allowing the CPU to do other more important stuff
- It has 96Kb of video RAM, 32Kb of internal RAM and 256Kb of external RAM
- It has a 2.9inch TFT colour screen (which is NOT backlit)
- The resolution of the screen is a maximum 240x160 but other resolutions may be achieved (there are 6 modes)
- It can handle up to 32,768 colours
- It can handle 511 simultaneous colours in character mode and 32,768 colours in bitmap mode
- Total unit size is 144x82x25mm and weighs 140grams
- It is powered by 2 AA batteries which can last up to 16 hours
- There is no region locking on either the system or the cartridges so it will play any Advance cartridge bought anywhere in the world, as well as 99% of Gameboy and Gameboy Colour games.
- Non-Advance cartridges are played on a reduced screen size or you have the option of viewing them in wide-screen

So you want to play games you have coded on a Gameboy Advance?

You do not need to buy any hardware to view your own programs on a Gameboy Advance. All you need to do is download one of the many emulators, load your program and view it running on screen as it would on the actual console (check my website links section for a comprehensive list of emulators and where to get them). This however is a poor substitute when you have spent weeks crafting your work and want to show people (or companies)

what you have done. What could be better than turning up for a job interview with a cartridge full of your demos ready to stick into a GBA and play?

So what you need is a cartridge you can upload your programs to and a piece of kit that enables you to upload to the cartridge. This is where the Gameboy Advance Backup System comes in handy. Basically it is a small unit which connects to a PC via a serial port. You plug a cartridge into the unit and use the supplied software to upload your programs. Then just stick the cartridge into your console and sit back in amazement as your game loads up.

For details of how to get hold of one of these pieces of kit just go to the links section of my website and look under 'Where to buy the hardware from'. All the kits come with everything you need (including a blank cartridge) and are quite reasonable in price. Extra cartridges can be bought separately but you only really need one as you can upload new programs as much as you like.

One final piece of advise, buy a PSU with the kit. They do run on batteries (6 AA batteries to be precise) but they will have gone after an hour or so. That would get expensive and frustrating very quickly.

The Software you need

In order to produce games for the Gameboy Advance you are going to need a compiler. There are 2 main compilers available GCC and ARM. I am going to talk about the GCC compiler exclusively because there are too many issues with the ARM compiler (the main ones being it is shareware, expensive to buy and no games developers use it). All of my code is written for use with the GCC compiler and it is free to get hold of.

Thanks to some very nice people who love GBA programming there is a version of GCC especially compiled for use with the GBA architecture. All that is required of you is an approx. 15MB download and install. This will give you a fully functional GBA compiler free of charge. Improvements are always being made but the latest version can always be found here: <http://www.io.com/~fenix/devkitadv/>

Setting up the GCC compiler on your system

So here goes my attempt at explaining how you go about setting up the GCC compiler.

The download comes in a number of packages but you don't need all of them. For example you have to choose which platform you are going to develop on Windows or Linux? I am going to assume you are a Windows user. GCC is compatible with Win 95/98/98SE/ME/2000 and XP so it covers just about everyone. Below is a description of the packages available and what they contain:

[Agb-win-core-r5.zip](#)

This package contains the core of the compiler so it is an essential download for everyone. If you don't have this then nothing will work. The r5 relates to the release number of this file, currently at revision number 5.

[Agb-win-binutils-r4](#)

This package contains the files you need for assembly and linking which any programmer should know is also essential.

[Agb-win-gcc-r4.zip](#)

This is the actual C compiler part, it does not contain any ANSI standard libraries etc. (they come next) but obviously it is still an essential download.

[Agb-win-newlib-r4.zip](#)

This contains the standard library files that no one should really be without. Downloading this makes your life much easier.

[Agb-win-libstdcpp-r4.zip](#)

This package contains the C++ Standard Library Functions, if you are programming in pure C then you don't need it.

[Agb-source-r4.zip](#)

Download only if you plan on producing a new feature to add to this GCC compiler.

[Agb-win-patch-r4.zip](#)

Have no idea what this file is for but recommend you download it if using Windows for development.

[Interflip.zip](#)

Deals with the interwork bit of ARM ELF object files. Download it!

Once you have all the packages you need what do you do? See the next section.

Where do I un-package it all?

Collectively you will hear the development suite referred to as **DevKitAdv**. Now onto the installation.

I highly recommend that you create a directory on your route drive called **DevKitAdv** and unzip all of the downloaded files into that directory. This will stop a number of problems from happening when compiling your programs. So:

- Create a directory on your route drive e.g. **C:\DevKitAdv**
- Unzip all the files you have downloaded (in the order they are shown above) into this directory

That's It! You now have a fully functional compiler ready for those games you are going to learn to code.

Checking that the installation worked

I have included a small program (courtesy of Dovoto with a few changes) that should test if your installation has worked. All you have to do is run the included batch file `make.bat` from the Command Prompt and the program should compile automatically.

Currently the batch file reads:

```
path=C:\devkitadv\bin  
  
gcc -o pixmanip.elf pixmanip.cpp -lm  
  
objcopy -O binary pixmanip.elf pixmanip.bin
```

But you need to change the path if you have installed the devkitadv into another directory. If this doesn't work then something has gone wrong down the line. Check you have all the packages and re-install them. I hope it works first time for you. If you are wondering what the program does then don't get too excited. It just turns the GBA screen green.

Why is the batch file so important?

As you can see I have printed the batch file above that I use to compile my programs. What the lines actually mean isn't all that important right now and if you have programmed before you should already know. Basically the first line tells you where the compiler is on your hard drive. The second line compiles your code and the third line links it together and produces the GBA executable.

The important thing to remember is that this batch file can be used to compile all of your C/C++ programs (although changes need to be made if using assembly language in your code). All you have to do to re-use this batch file is rename the files, at the moment my .CPP file is called **pixmanip** but my next program might be called **sprite**, in this case all I do is change all the instances of **pixmanip** in the batch file with **sprite**. Simple!

Try running the sample program through an emulator (I use Boycott Advance) and if you are lucky try it through hardware. Any problem e-mail me from my website. Hope this chapter helped someone somewhere.

In the next chapter...

There will be an in-depth look at the GBA architecture which will hopefully give you a good grounding for setting up the GBA (in coding terms) for displaying graphics.