

#### 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](mailto: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: 15<sup>th</sup> April 2002

## Chapter 5: Input on the GBA

### What is in this chapter

- The Buttons Register
- How to Identify Button Presses
- Multiple Button Presses
- A Handy GetInput() Function

The GBA has 6 buttons and the directional pad to allow you to control what is happening on-screen. You need to know how to get feedback from these buttons so that you can plan for them in your code.

### The Buttons Register

The GBA button register is called `REG_P1` which can be found in `gba.h`. This however is not a very good name for it so I (like many) have redefined it as `KEYS`, below is the actual declaration as seen in `keypad.h`:

```
int* KEYS = (int*)0x04000130;
```

By doing this your input function will look a lot clearer as you will see.

### How to identify Button Presses

If you take another look in `keypad.h` you will see a number of `#defines`. The `KEYS` register is used to tell which button is being pressed and each bit within the register relates to a specific key, the defines just give names to the values that relate to the different button presses. This makes it much easier (in terms of readability) to check for specific button presses.

So using `keypad.h` checking whether the A button has been pressed is as easy as:

```
if(!(*KEYS & KEY_A))
```

All this does is check to see if the bit that relates to the A button is clear (hence the ! before the 2<sup>nd</sup> bracket), if it is then it has been pressed.

### But what if I want to identify multiple button presses?

Good question. What if you wanted to allow for diagonal directions on the D-pad? This is also very simple. For example, if you only wanted something to occur if the user pushed the UP and B buttons together, it would be achieved with the following check:

```
if(!(*KEYS & KEY_UP) && !(*KEYS & KEY_B))
```

Here you just use a `&&`(and) to check if two bits in the `KEYS` register are clear together. So if you wanted diagonal directions on the D-pad you would check for:

- UP and RIGHT

- **UP** and **LEFT**
- **DOWN** and **RIGHT**
- **DOWN** and **LEFT**

In the same way as outlined in the code snippet above.

### **A Handy GetInput() Function**

I have included a program with this document called **ch5.bin**, if you run it a sprite should appear on screen that you can move around using the D-pad, it also performs simple bounds checking with the edges of the screen. The source code is included and you may want to take a look at the **GetInput()** function that can be found in **ch5.cpp**.

Although the program only uses the D-pad I have included if-statements for all key presses just so you get the general idea. It also makes it easier to copy-paste the function into your own code :0). I also recommend you keep a copy of **keypad.h** handy because you will probably use it in every GBA program you write.

This has been a very short chapter but there isn't much to write about input. This is good because it means it is a simple thing to do and at the same time very important for games.

### **What you should know from reading this chapter:**

- What the button register does
- How to check for each button on the GBA
- How to identify multiple button presses
- How the GetInput() function works