The Gameboy cartridge “Gamepak”.

The simplest gamepak has 32kB of ROM that stores the program code for a game.

Implementation for this is pretty straightforward as you all is necessary is to allocate 32kB of space and load a cartridge’s ROM image into it and connect those memory addresses to the Gameboy’s bus.

However, as development for the Gameboy software continued, developers found 32kB to be limiting as well as the amount of ram the system had. Nintendo addressed this by introducing new types of cartridges that would extend the Gameboy’s hardware to support new features.

Notably the Memory Bank Controller(MBC), this chip would allow for the Cartridge to bypass the 32kB limit and store up to potentially 8MB of data, the MBC would control what banks of 32kB chunks of data the Gameboy would see, allowing developers to store more data and switch out banks as needed. The MBC is used to swap out the address space seen by the rest of the bus whilst staying in the Gameboy’s 65536 unique addresses. MBCs are controlled by attempting to write to the ROM space of a Gamepak.

Not only that but provisions for extending the amount ram was all so added, some cartridges could add up to 128kB of extra ram or other hardware like rumble support, camera support etc.

Some of the RAM chips have a dedicated coin cell battery attached to them, this keeps the volatile data active and prevents it from erasure, this type of RAM is usually used to keep game saves for in between game sessions (progression, inventory, etc).

The 147th, 148th and 149th byte of a gamepak let’s us know three important parts of information. Firstly, the type of hardware included with cartridge (Which MBC is used? Is there any extra ram? Is there a battery?).  
Secondly, how large is the ROM and therefore how many 32kB banks are there.  
And finally, how much extra ram is present, if any.

There are 28 different cartridge types and 9+ different sizes of ROMs as well as 5+ different sizes of RAM that a cartridge can have. For this reason, we need to create a system that can handle all of these variations as well as allowing us to extend it with more support as we go along without breaking any previous functionality.

For this reason we need to capture the memory access