THEZZAZZGLITCH'S GLITCH RESEARCH ARCHIVES

Making "Super Glitch" moves useful (Red/Blue/Green)
First found and published on: April 02, 2013, 06:35:13 AM

1.1. FOREWORD

Well, I've been doing a lot of Super Glitch research for a year now, and I finally decided to show off some things that I found.

At the moment, Super Glitches are considered only as a fun little distraction without real use. And my goal is to turn things around and make them useful and predictable. Because if you think long enough, these moves have quite the power. They can write large numbers of values to the RAM by just displaying their names. You may also think "if there was a way to manipulate values or range of Super Glitch's memory corruption, that would be amazing".

OK, done with this boring foreword. **Yes, there is a way to manipulate (and exploit) the way Super Glitch modifies the RAM.** It worked on 4 different savefiles on emulator and on the actual cart, so I'm pretty sure it will work for everyone. But before we go into how to do it and before I show you some examples of useful "Super Glitch glitches", let's talk about some theory.

1.2. TECHNICAL DETAILS ON SUPER GLITCH

We all know Super Glitch heavily messes up the game's RAM. But from where it gets those values from? And where does it write? And why it even writes those values?

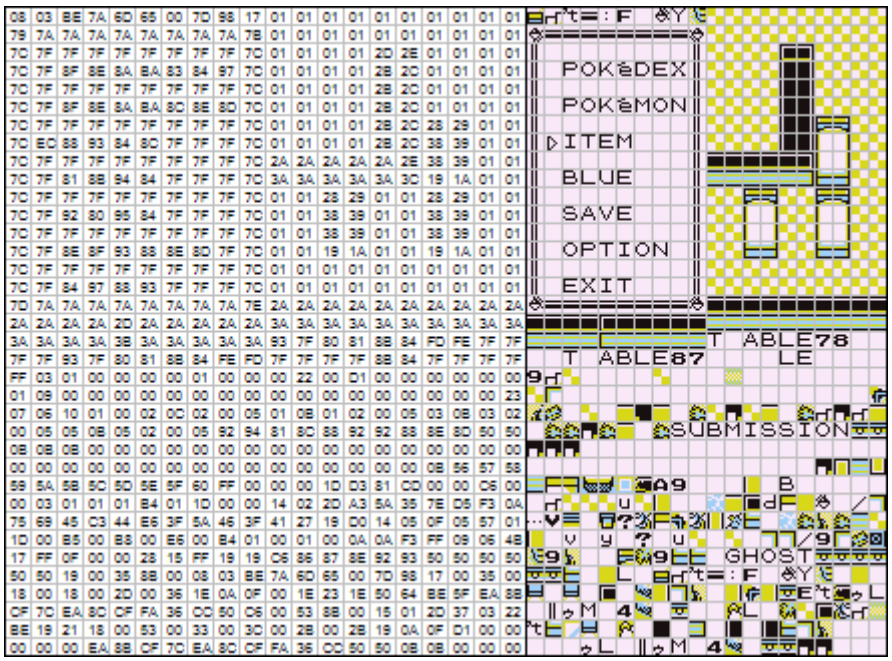
This is a subject worth a 10-page research document, but I will at least try to make this short and quick, presenting only a few facts needed. If you want a more detailed, technical explanation, go to the section "EVEN MORE DETAILS" at the end of this document.

First of all, it's not the move that is real dangerous, it's the name of the move that makes the game go weird. Super Glitch's name pointer is read wrongly by the game, and it makes the game believe the name of this move resides in the RAM. Which is an absolute nonsense, RAM has no move names at all. And when the game tries to read a move name, it buffers it byte by byte until it encounters a terminator byte (0x50). The problem is, RAM most likely won't ever contain a terminating character. So the game copies way more data than it should, resulting in a buffer overflow. And this is why Super Glitches corrupt the RAM.

The data is copied from address $CD6D, the buffer for move loading subroutine. And it is copied to $D0E1, the moveset buffer if you activate Super Glitch by viewing stats/learning a move/opening the fight menu, or to $CF4B if you activate Super Glitch by selecting/using it in battle. The buffer overflow causes a huge number of bytes from $CD6D to be copied to either $D0E1 or $CF4B.

How the buffer looks like normally; Moveset: WATER GUN, DOUBLESLAP, BODY SLAM, THUNDERPUNCH	How the buffer looks like with Super Glitch; Moveset: WATER GUN, DOUBLESLAP, BODY SLAM, [Super Glitch]
Array delimiter / First move / Second move / Third move / Fourth move / Other addresses	
The buffer at RAM:CF4B with SKULL BASH loaded. Everything looks nice and clean.	The buffer at RAM:CF4B with Super Glitch move loaded. That's a very long move name...

And if we look what does the $CD6D address contains, we will find out that there's a full screen copy here!



What does it mean? By changing what's on the screen at the moment, we can manipulate the values Super Glitch writes! And by positioning 0x50 overworld tiles on the exact places on the screen, we can also manipulate the length of Super Glitch corruption! However, the screen information here isn't just ordinary screen copy at the moment, it is refreshed only at certain times:

- While opening the Pokemon menu (both overworld and in-battle)
- While opening the Item menu (both overworld and in-battle)
- While opening the Pokedex menu
- While opening the trainer card screen
- While opening the Save menu
- While opening the Options menu
- While accessing the PC
- While watching the title screen
- Every time the game has to display a full screen message and needs to cloak the overworld map

This is even more convenient for us! If you for example open up the item menu in overworld, close it immediately, and go into a battle, if you don't open any menus while battling, the overworld screen copy will still be there as there was no need to overwrite it. By opening and closing the Pokemon menu in certain spots and not opening the start menu anymore, we can cause a single Super Glitch effect to happen almost 100% of the time.

1.3. CORRUPTION STARTING OFFSET

Also, because $D0E1 is not a single move buffer but a move list, a number of characters the move names before the Super Glitch have can also affect the length and values of corruption, shuffling the corrupted address by around 1-10 bytes.

Examples:

```
Moveset: Agility, Agility, TM28, [Super Glitch]
Agility [7 chars] + Agility [7 chars] + TM28 [4 chars] = 18 chars
0xD0E1 + 18 + 3 = 0xD0F6
The Super Glitch will start its corruption at address $D0F6
```

```
Moveset: Barrage, Clamp, [Super Glitch], Hi Jump Kick
Barrage [7 chars] + Clamp [5 chars] = 12 chars
0xD0E1 + 12 + 2 = 0xD0EF
The Super Glitch will start its corruption at address $D0EF
```

Note that while calculating the resulting starting address you have to add number of moves before the Super Glitch, as they are separated within the code by an invisible whitespace line feed character 0x4E.

1.4. “SUPER GLITCH GLITCHES”

Now we have some information on how to modify the values Super Glitch writes - so let's jump into some useful applications of it!

Harmless Super Glitch trick

Use: You can use this trick to learn/forget Super Glitches without any problem, view stats of Pokemon with Super Glitch moves, or swap/use Super Glitches in battle without crazy effects.
Statistics: Worked on all 5 tested saves on the first try.
Prerequisites:
- A Pokemon with Super Glitch, obviously.
- Access to Celadon City.

Execution:

1. Go to the exact spot shown on the screenshot below (1st floor of Celadon Mansion). Open up your Pokemon menu while still standing on that spot.

2. Congratulations. You just now immune to Super Glitches' glitchiness. Now you can learn, forget, view stats with Super Glitches involved without risking your save file.
3. You can also carry this effect to your local patch of grass in order to swap or use Super Glitches in battle without risk. Just open your Pokemon menu again, close it, go into a patch of grass and fight.
Note: Do not open your start menu at all while going to the grass or while fighting. This will reset Super Glitch to its usual glitchness.

Access Pokemon beyond the sixth slot

Use: With a corrupted Pokemon list and a corrupted item list (achievable using corrupted Pokemon list), you can make a lot of serious memory modifications - it's like having a memory viewer in your GB.
Statistics: Save #1,4,5 - Worked on the 1st try, Save #2,3 - Worked on the second try
Prerequisites:
- Access to Celadon City.
- A Pokemon meeting very specific moveset requirements:
  a) It needs to have a Super Glitch as a 4th move,
  b) Its three moves besides the Super Glitch have to contain 28 characters in total
  (for example: BODY SLAM [9 chars], DOUBLESLAP [10 chars], WATER GUN [9 chars])
- At least 5 Pokemon in your party, a party of 6 is recommended.

These moveset requirements aren't a real problem if you're using LM4, as its default starting moveset has 3 moves with 28 characters in total.
LM4 will have to learn no new moves till level 24, and at level 24 Hypnosis should be replaced with Super Glitch. Then you will be able to use previously described Harmless Super Glitch trick to switch the first move with the last.
Note: 4 first Pokemon in your party will change their species, but it isn't a problem as you're probably not going to save after this glitch anyways.

Execution:

1. Go to the exact spot shown on the screenshot below (second to last house on Celadon's south-east). Open up and close immediately your Pokemon menu while still standing on that spot.

2. Go into a patch of grass and encounter a wild Pokemon. Again, do not open your start menu while going there.
3. Open and close your fight menu a few times, the run from the battle.
4. Check your Pokemon list. Try to scroll past your 6th Pokemon. If you can't, repeat step 1. If you can - congratulations, you did it. If your game crashes, you obviously did something wrong.

Erase player's name

Use: This generates a perfectly blank properly terminated name, allowing you to save the game after you do something really game-breaking (Super Glitch, ZZAZZ, 2x2x2x2 after messing with 3906 for a while etc.), or you can just amaze your friends with an unobtainable name.
Statistics: Save #1,2,4 - Worked on the 1st try, Save #3,5 - Worked on the second try
Prerequisites:
- Access to Cerulean City.
- A Pokemon meeting very specific moveset requirements:
  a) It needs to have a Super Glitch as a 4th move,
  b) Its three moves besides the Super Glitch have to contain 28 characters in total
  (for example: BODY SLAM [9 chars], DOUBLESLAP [10 chars], WATER GUN [9 chars])
- Access to the field move Fly

Execution:

1. Go to the exact spot shown on the screenshot below (south-west corner of Cerulean City). Open up and close immediately your Pokemon menu while still standing on that spot.

2. Go into a patch of grass and encounter a wild Pokemon. Again, do not open your start menu while going there.
3. Open up and close your fight menu a few times, then run from the battle.
4. Your name should be now blank. However, it is still unsafe to save your progress.
5. Open up the start menu and select 'SAVE'. Don't freak out. When a glitched yes/no box appears, press B to cancel out.
6. You should end up in a glitch city. Fly away anywhere and you're now free to save. Saving is even recommended, as glitched trainers will now also appear instead of normal ones, and reloading the game will fix this problem. It shouldn't erase your game, I saved on 4 files (wasn't brave enough to do this on a cart) and the game was perfectly fine.

100% TMTRAINER

Use: Has no real use; It is here just to show how you can make Super Glitches predictable.
Statistics: Worked on all 5 tested saves on the first try.
Prerequisites:
- A Pokemon with Super Glitch, obviously.

Execution:

1. Encounter a wild Pokemon.
2. Open your Pokemon menu (in-battle) and close it shortly afterwards.
3. Open the fight menu and select (don't have to use) the Super Glitch move.
4. TMTRAINER 100% guaranteed. If it doesn't work, back out, open your Pokemon menu again, go back and try again.

Catch a level 82 Hitmonchan with (almost) infinite HP

Use: Making your way through E4, trolling people on link battles, having a partner for your 'Mew Smash'
Statistics: Save #1: 3 tries, Save #2: 1 tries, Save #3: 5 tries, Save #4: 7 tries, Save #5: 4 tries
Prerequisites:
- Access to Celadon City.
- A Pokemon meeting very specific moveset requirements:
  a) It needs to have a Super Glitch as a 4th move,
  b) Its three moves besides the Super Glitch have to contain 28 characters in total
  (for example: BODY SLAM [9 chars], DOUBLESLAP [10 chars], WATER GUN [9 chars])
- Access to Route 14

Execution:

1. Go to the exact spot shown on the screenshot below (Route 14's northmost field of grass). Open up and close immediately your Pokemon menu while still standing on that spot.

2. Optional step: Save your game. This will help you out as you may need a few tries to get this to work. After reloading the save you have to open your Pokemon menu again to rewrite your screen data, just saying (testers have been shouting LOLZ IT DOESNT WORK!!!1 while they forgot about opening the Pokemon menu after reloading the save).
3. Go find a wild Pokemon
4. Open the fight menu and continuously try to select the Super Glitch move. It will eventually change its type to either the name of last trainer battled, or a blank space.
5. Press B to exit out. You should end up fighting a Pokemon named with a bunch of player name characters. If it crashes, you failed - try again.
6. Open your item menu and use anything you're not supposed to (a bicycle or something). It should say your name followed by usual "not the time to use that". If your name is glitched, you failed - try again.
7. Throw all the Pokeballs. After you succeed, the game will state you caught a Hitmonchan. Congratulations!

1.5. EVEN MORE DETAILS

The game has something called the move name table. And it's a list of every move in the game, separated by terminating characters (0x50). When the game needs to buffer a name of a move, it calls a special subroutine which loads the table, loops through that table until it reaches an indexth move, and writes 20 characters from the found list location to a temporary location $CD6D.

When you open up the fight menu, in order to load the names of moves your Pokemon has, the game uses that subroutine mentioned before. After it completes, the game reads characters from the address $CD6D until a null terminator (0x50) is encountered. Then the name is appended to the move list so it can be displayed on the screen later.

As we can see, there is completely no error checking in any of these parts, the game heavily relies that supplied data will be valid - so what will happen if we try to request a name for an index number greater than 0xA5, like 0xB6? The searching function will start searching beyond the table, as the table has only 0xA5 entries and it has no more data programmed. It will keep searching and searching until it finds a null character (0x50) somewhere. The loop will eventually end up going for so long, that the game will start looking for the names in the RAM. As the RAM's contents constantly change, it's impossible to predict the location the function will choose. Let's say the subroutine luckily and completely by accident finds the character at $CC9F.

This makes the function believe that this move's name resides at address CC9F. It's obvious that this address does not contain any move names at all – it may contain internal timers, screen data or various buffers, depending on how far did the search go. Then, the subroutine copies these 20 bytes of garbage to the buffer and returns to the main function.

Now, the very important fact: At the time Generation I Pokémon series were developed, every byte and every execution cycle was important. Because every normal, nonglitched move has length of at most 14 bytes (13 characters and a null terminator), and the subroutine always copies 20 bytes, programmers have decided not to make sure the buffer is always terminated, as when dealing with a nonglitched move, the terminating 0x50 character will always get copied along with the name. However, that's not the case when we are dealing with glitch moves. What happens is the game will try to copy this buffer to the moveset array, and will keep copying until it finds a null character. However, the buffer now contains just garbage and there's absolutely no guarantee that the game manages to find the terminator.

If the game finds the terminator by accident, nothing will happen, you are safe. But if it won't find a null terminator soon enough, the game will attempt to read and copy characters beyond the buffer. Gameboy has no memory protection, so this operation will be happily accepted and executed, causing bytes after the move array to be overwritten.

So what we have here is a standard buffer overflow.