A Boy and His Blob: Trouble on Blobolonia Save State Hacking Guide

by MASTERNO

Updated to v1.00 on Apr 4, 2016

Table of Contents

Introduction & Copyrights

Greetings, and welcome to my hex editing guide for David Crane's A Boy and His Blob for the NES! With this guide, I will show you how to turn the Sewers of Hell into a peaceful walk through Wonderland! And once you're done doing that, you can turn Willy Wonka's Chocolate Factory into a CVS. I mean in *theory* you could actually hack graphics like that into the game, but that's not what this guide is here for. I mean, that's a bit too literal.

In case you didn't notice, my name is MASTERNO, and this is my first HTML-based guide. I wrote a number of hex editing guides for other NES games a few years ago and have recently picked the hobby back up, so hopefully you like the new format. I think it's a lot easier to work with and a lot easier on the eyes. If for some reason you want a text version in my older formats, feel free to shoot me a private message or an email and I can provide a copy. My current email(s) will always be listed on my Profile and/or Contributor Page.

A Boy and His Blob is a game designed and developed by David Crane and a bunch of his Activision buddies that joined him at Imagineering. The game was a pretty neat staple of the NES in its mid-life, being the winner of numerous awards for its ingenuity, graphics, and ability to keep players' attention. Age hasn't treated the game very well, as it proved to have a fairly simplistic design without much replay value. Still, for fans of platformers, puzzles, and adventures, this game is a fun way to spend an afternoon or two. Especially so if you're following this guide!

One interesting note to add about this game, for people who have been morbidly curious for nearly 30 years, is that the Ketchup Jellybean was originally supposed to be a Grape Jellybean with the pun being the "Grape 'Wall'" of Blobert. The wall was intended to repel enemies, but got replaced by the Ketchup Jellybean when a Nintendo manager noticed that you could become separated from Blobert, making the game impossible to complete. Crane made the change before the game was published, leaving part of the code intact for the Grape Jellybean. Unfortunately, the address and lines of code that made the wall function were replaced with the Ketchup effect, so the wall is now just a useless easter egg.

Known Hex Addresses

The following sections list the hex addresses and appropriate values for them that I have found to be meaningful to gameplay. A Boy and His Blob is actually fairly well programmed for a game of its time. Since the game is very simple in design, there isn't a wide variety to what you can reasonably play with when hex editing, though there are some really clever tricks I managed to figure out while digging around.

Addresses & Values

Gameplay

- **004B: Falling Speed**
  This affects how fast or slow you fall. The value is normally affected only by turning Blobert into either the Umbrella or the Bubble. The base value is 04 and can be increased or decreased as you wish. The Umbrella modifies this to 03 and the Bubble sets it to 02. As an added bonus, if you have the value set to 03 or lower while landing, you won't die from drop distance.
- **0071: Holding State** Determines whether or not you are holding a wieldable transformation of Blobert. In a nutshell, you can use it instead of a Ketchup Jellybean or to reposition Blobert in more point-sensitive areas. You can also get some humorous graphical glitches if you throw his Coconut form with this turned on. It's a simple switch address, so 01 turns it on and 00 turns it off.
- **0073: Throw Lock** While I haven't really found a use for this address, it's worth noting that keeping it on will prevent you from throwing anything. Best to leave it alone, but if you want something to play around with and figure out, here you go!
- **0075: Bubble State** This godsend of an address determines whether you're inside Bubble Blobert or not. It's a simple switch value, so 01 turns it on and 00 turns it off. If you keep it locked at 01, you can protect yourself from those notorious stalactites and stalagmites in the underwater section of the caverns. Keep in mind, though, that while this is turned on, you cannot throw jellybeans and you cannot whistle yourself out of the bubble (because the value to determine that is locked...duh!). When using this address alone, you will come across a number of graphical glitches, a few of which are described below.
- **007A: Jumping Height** When using Trampoline Blobert, this value determines how high you go after you bounce on him. It's a logical scale: higher values make you go higher. The values do change rapidly while you hold the up and/or down buttons, so having access to precise measurements can be a real boon. Normal minimum value is 03 while the maximum depends on which world you're on: Earth's maximum is value F4 (~7 screens) and Blobolonia's maximum value is 4F (~1¾ screen).
- **0082: Blobert's Sprite State** Interestingly enough, Crane programmed Blobert's various states and animations into a pair of hex addresses (the other is a couple of addresses down on this list). This specific address determines what sprite Blobert takes on. Normally, it's pretty glitchy unless the codes match up properly. The base value is 00, which if locked in will prevent Blobert from completing any transformations while the value is locked. He will go through the initial animations, but won't complete the change until you unlock the address, at which point he will assume whatever form the most recent jellybean you fed him makes. Some flavors/forms do share values: 0C works for Cinnamon "Blow Torch", Apple "Jack", and Tangerine "Trampoline", 10 works for Coconut "Coconut", Cola "Bubble", Lime "Key", Root Beer "Rocket", Honey "Hummingbird", and Punch "Hole", and 1C works for Vanilla "Umbrella" and Orange "Vitablaster". The remaining flavors/forms have unique values: 18 for Ketchup/Grape "Wall", 54 for Licorice "Ladder", and 64 for Strawberry "Bridge".
- **0083: Blobert's Form State** This address determines if Blobert is transformed or not. The normal value is 01, which says that he is not transformed. Locking that value in will prevent him from transforming at all, even after unlocking the address. Consequently, if he is already transformed when the value is set to 01,

he will remain in his current form until he is returned to normal, after which he will be unable to change until the address is unlocked. Conversely, if you set the value to 00 while he is in his normal form, the opposite will happen; he will be able to transform into whatever you feed him, but be unable to change back. This isn't a terribly useful address unless you want to avoid inadvertently changing him back and forth.

- **0086: Blobert's Form** The sole reason that bug where you can have Blobert turn into a wall is this value right here and the script that references it. Specifically-speaking, this address defines the type of bean you're feeding to Blob to determine what he turns into. The reason that bug exists is due purely to timing; the script doesn't reference this address until after a certain point in Blobert's transformation animation, during which you can change the value by throwing a different bean. There isn't a lot of time to perform it, so you can't really feed him a Licorice Jellybean and expect to make him turn into an Umbrella that easy. At least not without setting this address to the appropriate value. Be aware that keeping the value locked will prevent Blobert from changing form, though some issues can result from changing values after he's already been transformed (at least without using some of the other above addresses).

Form	Value	Form	Value
Licorice "Ladder"	00	Vanilla "Umbrella"	07
Strawberry "Bridge"	01	Tangerine "Trampoline"	08
Coconut "Coconut"	02	Root Beer "Rocket"	09
Cola "Bubble"	03	Honey "Hummingbird"	0A
Cinnamon "Blowtorch"	04	Ketchup/Grape "Wall"	0B
Apple "Jack"	05	Punch "Hole"	0C
Lime "Key"	06	Orange "VitaBlaster"	0D

- **00A2: Extra Lives** Not that it needs any explanation, but if you set this to anything higher than 00, you will never game over if you die. Values 01, 02, and 03 will cause that number of extra life sprites to appear in the top text box, so you may want to keep it set to 01 or 04 or higher. Mostly, just set it to 01. Life is easier (and more fun) that way.
- **00AF: Peppermint Count** Simply put, this address determines how many mints you've collected. Not really useful unless you're obsessive-compulsive about maxing everything out because setting this value to anything will prevent you from earning extra lives by collecting them. Not that you would need extra lives anyway considering I also provided that address here.
- **00B2: Pharmacy Access** This address flags whether you've been to the pharmacy or not. When the value is set to 00, you can enter it as many times as you like given that you have physically collected at least one treasure. Normally, it is set to 00 at the beginning of the game, but if you try entering the pharmacy, nothing will happen because whatever other value it references indicates that you have not collected any treasure and cannot define which Assortment of vitamins you've earned. When you return after collecting some treasure, though, entering the pharmacy changes the value to whatever Assortment you earned, preventing you from barring the pharmacy due to the script requiring the address to be set to 00. As of the initial release of this guide, I've been unable to determine the exact address(es) that interact with this one to decide what Assortments you get or whether or not you've physically collected treasure (as opposed to just changing some numbers around like you've been doing). Or at least, I haven't figured out how the script is written to reference, check, and verify the information it's requesting (I'm not a programmer myself, just a hobbyist editor). One thing I do know is that it is oddly possible to trick the game into giving you an Assortment 7 by setting the Treasure Count address to 00, but you are still required to collect at least one physical treasure before the game will let you into the pharmacy. This was probably the most frustrating address I tested in the whole game. Of course, it is a lot less useful when you can just give yourself infinite vitamins outright, but if you're looking for just a few trivial quality-of-life addresses, this would certainly be one of them.
- **02BF~02D9: Jellybeans** This address range determines how many you have of a given jellybean. I used the tilde to denote that it's not the entire range used, rather every *second* address represents a different bean. Setting the value to anything greater than 00 will give you more jellybeans than a Jelly Belly factory could ever produce, though, oddly enough, the values normally count in decimals, ignoring the additional hex values. That means if you want to have 99 beans, you set the value to 99. These are the most useful addresses in the game, especially when you want to explore, experiment, and generally have fun messing around with the game.

Bean Type	Address	Bean Type	Address
Licorice	02BF	Vanilla	02CD
Strawberry	02C1	Tangerine	02CF
Coconut	02C3	Root Beer	02D1
Cola	02C5	Honey	02D3
Cinnamon	02C7	Ketchup/Grape	02D5
Apple	02C9	Punch	02D7
Lime	02CB	Orange	02D9

- **02DA-02DF: Vitamins** As with the Jellybeans address, these values determine the number of vitamins you have. Each type of vitamin has two adjacent address to determine this value: A=DA/DB, B=DC/DD, C=DE/DF. The first address is the marker for the hundreds value (X00) and the second address is for the remainder (0XX). As with Jellybeans, these addresses use decimal values to determine the quantity and ignore the additional hex values. This address range can be very useful if you want to play the game relatively straight, but otherwise there are enough strategies and glitches to make vitamins completely useless in the end.
- **0356: Treasure Count** Pretty much accounting for 2/3 of the game, collecting treasure can get a bit tedious after a while. Setting this to 00 will make it so you don't have to (at least if you intend on using other glitches or addresses to allow you finish the game without them). This address is particularly interesting in that it seems to be the only one directly tied to all the treasures. As noted in the Pharmacy Access address above, adjusting this address has some weird issues unless you set it to 00. Without the programming knowledge I'd need to access and read the scripting, I can't really determine what's going wrong when you mess with this address. I can safely say, though, that setting it to 00 is probably the best thing you can do with this address. Feel free to play around with it on your own and see what you can come up with.

Text & Other Addresses

- **00A9-00AE: Score** This range is where the actual score value is held. As with jellybeans and vitamins, it uses decimals to define its values while ignoring additional hex values. Each address contains information for only one digit, so they use a text database instead of a calculated output for display. The important thing to keep in mind here is that the actual text box addresses overwrite the display of the score, but the score will remain as you set it or will keep being calculated since it's stored in these addresses and not in the ones used for text box editing.
- **0286-02BD: Text Boxes** One of the most entertaining finds in this game was that you could edit the text boxes at the top and bottom of the screen without having to do a lot of programming wizardry. All you need are these addresses, a basic understanding of hexadecimal, and the layout of the text database (the former two of which you now know and should have, the latter you are about to find out). Each box has a range of 27 characters total: the upper box has the range 0286-02A1 while the lower box has the range 02A2-02BD. The extra lives and peppermint indicator images are sprites overlaying the text, so they will partially block the spaces they're sitting on. Note that overwriting these values will supersede anything else the game would otherwise point to; whatever you write in the lower box range will appear even if you change the type of jellybean you're using or your score will be hidden on the upper box when you write something to that range (as mentioned above).

Text	Value	Text	Value	Text	Value	Text	Value	Text	Value
(Space)	00	7	08	F	10	N	18	V	20
0	01	8	09	G	11	O	19	W	21
1	02	9	0A	H	12	P	1A	X	22
2	03	A	0B	I	13	Q	1B	Y	23
3	04	B	0C	J	14	R	1C	Z	24
4	05	C	0D	K	15	S	1D	=	25
5	06	D	0E	L	16	T	1E		
6	07	E	0F	M	17	U	1F		

Glitches, Issues, and Other Information

This section is dedicated to various glitches and issues I've encountered along the way, along with any miscellaneous information I can pass on that's relevant to this guide. Unlike previous guides I've written, I have not been terribly thorough with the testing for this section. A Boy and His Blob is really such a simple game that there isn't a lot of fun to be had digging around for such things. It's not a large and complex game like Metroid or Zelda, nor is it stocked with a lot of features or functionality like most action games, so there isn't a lot of real depth to the design and code of the game to make it fun to break. Nevertheless, there are a few notes of interest to write home about.

- **004B: Falling Speed** When setting this address to 06, you will "jump" to the platform directly above you if you fall, regardless of how many screens above you that platform is. It results in some glitch animations while it's happening, but otherwise does not appear to adversely affect the game.
- **0071: Holding State** Most of the glitches come from attempting to drop or throw whatever form you were holding. This can result in some pretty humorous animations, particularly with Coconut Blobert. There are a number of graphical glitches that can occur too. Since it's just a switch, I haven't tried setting the values to anything other than 00 and 01, so setting it to undefined values might cause other issues or instability and crashes.
- **0075: Bubble State** As mentioned before, if used without adjusting other related addresses, it creates graphical glitches, generally in the form of Blobert's sprite layered directly over the Boy's sprite. As with other switch-value addresses, I haven't tested this beyond 00 and 01.
- **0082, 0083, and 0086** These three related hexes create all sorts of insane graphical anomalies depending upon how you set their values. Some of the funniest come from turning Blobert into Ladder form before hard-setting his form to something else. I haven't particularly noticed any stability or crashing issues related to these three addresses yet, but I'm sure there's bound to be with as silly as they get.
- **00A2 & 00AF: Extra Lives & Peppermint Count** Purely text-based glitches. Setting the values too high without covering up the number displays will give graphical text issues. Otherwise they function without any real issues, though the possibility may exist that the game could become unstable or crash if you set the values too high. I doubt it though.
- **00B2: Pharmacy Access** Ah, so much time spent frustratingly trying to nail this address down to the Mr. T, but so many problems. Where to begin? Well, for starters, messing with the Treasure Count illegitimately has negative implications when it comes to this address. Oftentimes I found that if you changed the Treasure Count sometime before or after you start collecting treasures, it will often just give you Assortment 1 unless the Treasure Count is set to 00. I am well aware that there are 7 actual assortments available, so it's a bit odd how this address interacts with others. As for the values for this address itself, it seems like it's coded so that the Pharmacy is only open when it's set to 00, after which it will add the Assortment number to the value, preventing you from using the Pharmacy again. A clever way to keep the code relatively clean and open up an address or two that would otherwise be required. I just can't seem to figure out what else the script(s) point to in order to find a way to make this abuseable for quality-of-life improvements. About the best I could suggest is to set this to 00, grab the first treasure in the subway, set 0356 to 00, then abuse other addresses to quickly return to the surface to get your vitamins stocked. If anyone figures anything else out, feel free to let me know and I'll give you full credit for the discovery.
- **02BF~02D9, 02DA-02DF: Jellybeans & Vitamins** Similar to the Extra Lives and Peppermint Count issues, it only really affects text graphics. While not mentioned as a "feature" above, you can make it easier to scroll through various jellybeans and vitamins by setting ones you don't need to 00, thus making them not show up when you cycle through the inventory - a standard feature of the game.
- **0356: Treasure Count** When sticking with the natural values it is programmed with, there don't appear to be too many glitches. Since it's not an issue with normal gameplay, it probably wasn't even considered that hard-setting this address could have some adverse effects or unintentional consequences with regards to the physical treasures themselves. There doesn't appear to be a script written to verify either the treasure count or the presence or absence of specific treasures when such values would change. Ergo, changing this address doesn't cause treasures to disappear at all when - for stability reasons - a script could have been written to eliminate treasures in a logical collection order should this address be modified independently. It's understandable why it wasn't, though, and a simple LUA script may be able to resolve that if the specific addresses for the treasures are actually discovered (I have not looked for them, but I may decide to do so for a future update).
- **00A9-00AE & 0286-02BD: Score & Text Boxes** As is standard for NES games, text-based addresses get unique glitches that often don't affect the stability of the game. A Boy and His Blob is no different. The standard values defined for the text database range from 00 to 25 (as laid out in the table above). Everything higher than 25 is system graphics and sprites. Keep in mind that actual sprites are layered over text, such as is the case with the Peppermint and Extra Lives indicators. I don't think any values that can be set into these addresses will be considered sprites, I guess it was worth mentioning again. I've played with the values a bit for these, but there isn't a lot of weird stuff to find, at least not until the much higher values that I didn't test out.

Version Summary

Version 1.00 - Initial release version. Contains most of the important relevant information.