Tactics Ogre: The Knight of Lodis Character Hacking Guide

by Soren Kanzaki Updated to v0.7 on Jun 4, 2002

```
Tactics Ogre: Knight of Lodis Character Hacking Guide v.0.7
Released on June 4, 2002
by Soren Kanzaki (soren_kanzaki@yahoo.com)

------------------
Table of Contents:
------------------

Section 1: Overview
Section 2: Version History
Section 3: Character Memory Block Basics
Section 4: Byte by Byte
Section 5: An Example / Advanced Information
  Part A: Canopus the Commander
  Part B: Game Logic
Section 6: Character Block Address Table
Section 7: Character In-Battle Block Address Table
Section 8: Class Completion Medal Code Table
Section 9: Emblem Code Table
Section 10: Alphabetic Code Table
Section 11: Picture Code Table
Section 12: Class Code Table
Section 13: Element, Alignment, and Sex Code Tables
Section 14: Month Code Table
Section 15: Item Code Table
Section 16: Skill Code Table
Section 17: Supplement: Memory, Binary, and Hexidecimal
Section 18: Credits
Section 19: Copyright / Authorization
Section 20: Miscellaneous

--------------------
Section 1: Overview
--------------------

    By know, everyone knows you can't get all the special characters in a
single game.  It's impossible.  Can't be done.

    Or can it?  Using modern science (or, at least, the cheating system of
your choice), is it possible to re-create Rictor and Orson if you've gotten
Cybil and Shiven instead?  What about making a Wizard who can use the
Shuriken Barrage?

    If you miss the ability to customize your classes, and carry over skills
from one class to another ... well, perhaps you can have your cake and eat it
too.

---------------------------
Section 2: Version History
---------------------------

    0.6 (5/23/02): First draft.  Character Block Address Table only lists the
```

first 10 characters (haven't tested to see where the Character Blocks end).

0.7 (6/04/02): Added information on Biorhythm bytes, thanks to Terence Fergusson's Biorhythm Mechanics FAQ. Added some of the Emblem Tracking bytes. Investigations so far find a perfect match between Battle and Long-Term data blocks. Character In-Battle Block Address Table added. Added correct name for 'Compressed Binary Switches' (should be bit fields, thanks to Rob Coley).

Section 3: Character Memory Block Basics

A disclaimer before we begin (as borrowed from my other hacking guides):

If you use any of these cheats, I'm not responsible for any 'weird' things happening to your game or your save data. You use these cheats at your own risk (to your game, your system, your enjoyment of Tactics Ogre: Knight of Lodis).

I made this document as a sort of educational glimpse into how the game was put together. You can make the game easier. You can make it harder. You can make it more fun. You can make it a bore. I think you get the point.

Secondly, this document is much more technical in nature than other things I have written. I cannot guarantee it's 100% correct. I cannot guarantee you'll understand it. Hopefully, both of those conditions will hold true.

Now, with the formalities over ...

Tactics Ogre: Knight of Lodis uses a two-tiered system to store information about the characters (friendly and enemy) in the game. Enemy data actually only probably has one tier (the second layer), but that's another topic altogether.

The first layer of character information is the 'long-term' memory block. This guide focuses on that block, which is 104 bytes long per character. Various information about the character is stored in this block.

During battle, certain vital statistics about the participating combatants are kept in an area of memory. The basic information for these combatants is copied before battle from the 'long-term' memory block. However, you can alter this block (and alterations to current HP/MP/SP are basically only applicable if you make alterations to this area). At the end of combat, information from this block is copied back over to the 'long-term' block (updating the statistics and values gained during combat). So far, it looks like this battle data block uses the same format as the 'long-term' block, so addresses have been computed for making changes during combat. Note that this will eventually make changes in the permanent data block (when the game copies the data over at the end of battle), and can be a MUCH faster way of making batch changes to your entire team's statistics and emblems. (Simply set the codes for the first friendly character deployed, and use the Training Mode to change that character.)

Let's look an example.

Section 4: Byte by Byte

Here's the breakdown for Alphonse's (the main character) data. He's the first character stored in data; additional soldiers are stored after him. Yes, there are a lot of unknown sections. Also, the Byte numbering is not random (you'll see why, eventually). The address listed in (parenthesis) indicates the memory address for Alphonse's data.

```
Bytes 0-3    (2000030): Unknown
Byte 4       (2000034): Class Completion Medals, Block I (see below)
Byte 5       (2000035): Class Completion Medals, Block II (see below)
Byte 6       (2000036): Class Completion Medals, Block III (see below)
Byte 7       (2000037): Unknown
Byte 8       (2000038): Emblems, Block I (see below)
Byte 9       (2000039): Emblems, Block II (see below)
Byte 10      (200003a): Emblems, Block III (see below)
Byte 11      (200003b): Emblems, Block IV (see below)
Bytes 12-16  (200003c - 2000040): Unknown
Byte 17      (2000041): Special Holy Kills
Byte 18      (2000042): Lancer Success Toggle*
Byte 19      (2000043): Philosopher's Stone Success Toggle*
Byte 20      (2000044): Self Preservation Success Toggle*
Byte 21      (2000045): Berserk Success Toggle*
Byte 22      (2000046): Successful Persuasions (for Arbitration)
Byte 23      (2000047): Unsuccessful Female Persuasions (if male)
                          OR Successful Male Persuasions (if female)
Byte 24      (2000048): War God Success Toggle*
Byte 25      (2000049): Frontal or Side Attacks Made
Byte 26      (200004a): Successfully Dodged or Blocked Attacks in a Row
Byte 27      (200004b): Miracle Success Toggle*
Byte 28      (200004c): Successful Ranged Attacks in a Row
Byte 29      (200004d): Fist Fight Success Toggle*
Byte 30      (200004e): Successful Heals Cast in Combat
Byte 31      (200004f): Don Quixote Success Toggle*
Byte 32      (2000050): Treasures Opened
Byte 33      (2000051): Archangel's Feather Toggle*
Bytes 34-43  (2000052 - 200005b): Unknown
Byte 44      (200005c): Total Kills
Byte 45      (200005d): Dragon Kills
Byte 46      (200005e): Beast Kills
Byte 47      (200005f): Exorcisms Performed
Bytes 48-63  (2000060): Character's Name
Bytes 64-65  (2000070): Character's Current HP
Bytes 66-67  (2000072): Character's Maximum HP
Bytes 68-69  (2000074): Character's Current MP/SP
Bytes 70-71  (2000076): Character's Maximum MP/SP
Bytes 72-73  (2000078): Character's Strength
Bytes 74-75  (200007a): Character's Intelligence
Bytes 76-77  (200007c): Character's Agility
Byte 78      (200007e): Character's Picture
Byte 79      (200007f): Character's Class
Byte 80      (2000080): Alternate Character's Class (see below)
Byte 81      (2000081): Character's Number
Byte 82      (2000082): Character's Level
Byte 83      (2000083): Character's Experience
Byte 84      (2000084): Character's Element
Byte 85      (2000085): Character's Alignment
Byte 86      (2000086): Character's Birth Month
Byte 87      (2000087): Character's Birth Day
Byte 88      (2000088): Biorhythm Set 1 [Definitely Amplitude]
```

```
Byte 89      (2000089): Biorhythm Set 2 [Probably Period]
Bytes 90-91  (200008a): Biorhythm Set 3 [Probably Offset]
Byte 92      (200008c): Biorhythm Set 4 [Definitely Altitude]
Byte 93      (200008d): Character's Allegiance (see below)
Byte 94      (200008e): Character's Sex
Byte 95      (200008f): Unknown
Byte 96      (2000090): Equipped Item Slot 1
Byte 97      (2000091): Equipped Item Slot 2
Byte 98      (2000092): Equipped Item Slot 3
Byte 99      (2000093): Equipped Item Slot 4
Byte 100     (2000094): Skill Slot 1
Byte 101     (2000095): Skill Slot 2
Byte 102     (2000096): Skill Slot 3
Byte 103     (2000097): Skill Slot 4
```

Okay! That's all 104 bytes (0-103). Now, what does all this mean?

Class Completion Medals - Whenever a character successfully transfers into a new class, a medal is awarded. If you have all 14 medals, you receive a specific emblem (Relix or Ripple's Emblem).

Emblems - All emblems are stored as bit fields. Block I refers to row 1 emblems, Block II to row 2 emblems, etc. etc.

Special Holy Kills - This is the number of kills that count towards the 'Gibe of Fallen Angel' emblem.

Toggles with * - Theses are used to determine success or failure for the conditions set for an emblem being earned. If the value is non-zero, the emblem will be awarded during the next emblem checking phase.

Character's Name - If the preceding byte doesn't have a letter, than the current byte will not be displayed (i.e., names must be continuous, you can't skip from one byte to another randomly). Also, beyond a certain point the character's name is overlap and obscure the class display.

Character Statistics (Current/Max HP, MP, Strength, Agility, Intelligence) - the 'current' values will only work during the Organization screens. They won't carry over to battle phases.

Character's Picture - if the character has a special picture, then this byte remembers which picture to use. If this value is 0, the character will not appear in the Organization screen.

Character's Class/Alternate Class - it seems that for special characters (Alphonse), you can use the alternate class byte. For regular characters, the regular class byte is the only one that seems to work. I have yet to test this on other special characters.

Character's Number - the number as given in the Organization screen.

Character's Level / Experience - again, these are the Organization screen values - if you fix the Character's Experience value, they could still level up in combat (since during combat a different address holds the temporary character data). The Experience value ranges from 0-100 (that is, it is non-cumulative).

Character's Element/Alignment - self-explanatory.

Character's Birth Month - ranges from Deus to Raio.

Character's Birth Day - the day of the month in which the character was born.

Biorhythm Sets - Thanks in large part to the mechanics provided by Terence Fergusson's Biorhythm Mechanics document, I have definitely isolated the Amplitude and Altitude values (sets 1 and 4, respectively). Changing Sets 2 and 3 seem to alter the current Fortune without changing into another group of fortunes. I can also verify that the values given for Amplitude and Altitude relating to which set of Fortunes you cycle through are correct.

Character's Allegiance - I haven't tested it in combat; all I know is that a value of 255 will turn the status screen background colors to the normal 'enemy' color scheme. (As such, there's no section for this one.)

Character's Sex - as a side note, beyond 2, it starts reading from the City Names. (Yes, a character will then have a sex of Lutra, or Solea, or ...) I'm not sure if this affects the perceived sex of the character. With the class and/or picture code, you can make 'cross-dressing' characters (male sex, female sprites or female sex, male sprites).

Equipped Item / Skill Slots - self explanatory.

To find out the address for any other character, add the byte number as given above to the base address given in Section 6, the Character Block Address Table. (This is why the bytes start at 0 ...). To find out what value to insert into the address, look it up in the appropriate section, or use common sense (e.g. Maximum HP).

Section 5: An Example / Advanced Information

Part A: Canopus the Commander

Everyone remembers Canopus, right? The original Hawkman from the original Ogre Battle: March of the Black Queen. Anyway ... we'll borrow him for this little demonstration. Let's turn Alphonse into Canopus!

Well, what's involved? First, we need to transform Alphonse into a Normal Hawkman. His picture, therefore, is no longer Alphonse (2) but should reflect his class (which we'll be changing to Hawkman). Therefore, we need to change 200007e to 1.

Now we need to turn him into a hawkman. Hawkmen are 23, so we put that in the class addresses, 200007f. (May as well put 23 in 2000080 to be sure.)

What else could we do? Well, let's give Canopus a few special skills. In his original incarnation, he could become an Eagleman (with Thunder) or a Ravenman (with Inferno). Let's just say he's been practicing a bit, and can do both.

To give him a 'Thunder', let's use Thunder Arrow (72), and to simulate 'Inferno', let's use Firestorm (9). Putting these in Slot 3 and 4, we get: address 2000096, value 9 and address 2000097, value 72.

You could do other additions or changes, but now 'Alphonse' is 'Canopus'. (Well, if you named him Canopus at the start, anyway ...)

(Of course, the game will lock when trying to start battle dialogues, since it's looking for an Alphonse, and hawkmen can't use the spell Firestorm. You'd be better off adding a few skills to another character, but this section is intended only as an example of how to alter characters.)

Part B: Game Logic

Some tidbits of game logic. First of all, changing to certain classes during the Organization screen has a high chance of crashing the game - primarily, enemy classes. My guess is that the game is not sure what to display.

Secondly, remember that this only edits the 'long-term' character data. If things happen during battle to change this data, the game will attempt to over-write this data with the temporary data that is has copied in the temporary data block. Granted, if you fix the values, any attempt to over-write them will fail. But if you, for example, fix the experience of a character (to prevent them from leveling up), they can still gain levels - you haven't fixed their level, and you haven't fixed the temporary experience value, either. If that hits 100 (and the character gains a level), you'll find that your character has indeed gained a level.

Known things to avoid:

If you change a generic (Character's Picture = 1) character into certain specialty classes, the game will crash. These are classes like High Priest and General that must be paired with a different picture than 1. This is probably because there is no 'generic' class picture for these characters. Use the Alternate Data bytes, otherwise bad things will definitely happen, and set the picture to anything but 1.

One small note. EVEN if you fix the HP of a character, if they receive enough damage in one attack to kill them, the character WILL die.

Section 6: Character Block Address Table

Remember, this is the beginning of each character block. To find a particular address inside this block, add the byte number to the listed address. As this are memory addresses, they are in hexadecimal, and all hexadecimal math rules apply!

2000030 - Alphonse / Character # 1
2000098 - Character # 2
2000100 - Character # 3
2000168 - Character # 4
20001d0 - Character # 5
2000138 - Character # 6
20001a0 - Character # 7
2000208 - Character # 8
2000270 - Character # 9
20002d8 - Character # 10

Section 7: Character In-Battle Block Address Table

2001420 - Friendly Character Deployed #1
2001488 - Friendly Character Deployed #2

2001770 - Enemy Character Deployed #1
20017c8 - Enemy Character Deployed #2

(Yes, I know there's only 2 and 2 ... I'll be adding more later. I've been busy ^^)

Section 8: Class Completion Medal Code Table

Okay, this section introduces a different type of data system called a bit field. (Thanks to Rob Coley for sending in the real name!) What this means is, the computer looks at these bytes not as whole bytes, but as 8 bits in a row. Each bit has a function, and can be on or off.

What does this mean in practical terms? This means you must compute the value you want on the fly! (Since you can't enter cheats in binary for the most part.) As Section 11 goes into the math behind all this, I won't go into too much depth here. Sufficed to say:

To calculate the value to place in a bit field byte, look up the 'to-add' values given. Let's say you want to give a character the medals for Soldier, Ninja, Swordmaster, and Dragoon. We add: +1, +2, +64, +128 = 195. Therefore, you would put 195 into the appropriate memory address. See how that works? People who understand binary should see why these values are the way they are immediately.

Class Completion Medals, Block I:
+1 = Soldier
+2 = Ninja
+4 = Archer
+8 = Wizard
+16 = Cleric
+32 = Knight
+64 = Swordmaster (Male Only)
+128 = Dragoon (Male Only)

Class Completion Medals, Block II:
+1 = Valkyrie (Female Only)
+2 = Siren (Female Only)
+4 = Warlock (Male Only)
+8 = Witch (Female Only)
+16 = Priest
+32 = Dragon Tamer (Female Only)
+64 = Beast Tamer (Male Only)
+128 = Lich

Class Completion Medals, Block III:
+1 = Angel Knight
+2 = Ghost
+4 = Unknown (possibly no function/special class only)
+8 = Unknown (possibly no function/special class only)
+16 = Unknown (possibly no function/special class only)
+32 = Unknown (possibly no function/special class only)
+64 = Unknown (possibly no function/special class only)
+128 = Unknown (possibly no function/special class only)

Section 9: Emblem Code Table

    Everything that was true about the Class Completion Medal Codes in Section 7 applies here.

Emblems, Block I:
+1 = Blood Reign
+2 = Dragon's Scale
+4 = Animal Hunter
+8 = Exorcist
+16 = The Pen and the Sword
+32 = Gibe of Fallen Angel
+64 = Lancer
+128 = Philosopher's Stone

Emblems, Block II:
+1 = Self Preservation
+2 = Berserk
+4 = Arbitration
+8 = Broken Heart / Vixen's Whisper
+16 = War God
+32 = Knight's Certificate
+64 = Book of Initiation
+128 = Miracle

Emblems, Block III:
+1 = Sniper
+2 = Fist Fight
+4 = Heavenly Spirit
+8 = Don Quixote
+16 = Embodiment of Desires
+32 = Archangel's Feather
+64 = The Cycle of Life
+128 = Mark of the Elite

Emblems, Block IV:
+1 = Centurion
+2 = Charisma
+4 = Bullpen Ace
+8 = Bogus Hero
+16 = Lucky Soldier
+32 = Mark of Valor
+64 = Veteran Soldier
+128 = Relix's Emblem / Ripple's Emblem

Section 10: Alphabetic Code Table
---

0 - 25: A - Z (capital letters)
26 - 51: a - z (lowercase letters)
52 - 61: 0 - 9
62: ! (exclamation mark)
63: ? (question mark)
64: . (period)
65: , (comma)
66: : (colon)

67: & (ampersand)
68: _ (underscore)
69: % (percent sign)
70: ' (apostrophe)
71: ((starting parenthesis)
72:) (closing parenthesis)
73: < (less than, but narrow)
74: > (greater than, but narrow)
75: + (plus)
76: - (minus)
77: * (asterisk, multiplication)
78: / (division, slash)
79: = (equals)
80: Deneb's Cat's Paw (in the Witch Deneb's class)
81: Opening Double Quotation Marks
82: Ellipsis
83: Some graphical mark
84: Like 83, only the mirror-image
85: Small Space
86: Full Space
87: Full Space
88 - 254: Nothing (not even a space; the game skips this character)
255 - End of Word (after this, no characters will be placed into the name)

Section 11: Picture Code Table

I'm not sure of some of these - I haven't had the time to get past the second battle in the game with everything else ... Corrections / additions / input would be appreciated.

0 - None (character will not be displayed in Organization at all)
1 - Default (use the correct picture for the current class)
2 - Alphonse
3 - Eleanor
4 - Cybil
5 - Rictor
6 - Ivanna
7 - Shiven
8 - Orson
9 - special mermaid
10 - some special male character
11 - some special female character
12 - special angel knight
13 - Glycinia
14 - Lubinia
15 - some sort of demon (possibly Saya)
16 - special male
17 - Deneb
18 - older male (see what I mean about additions ...)
19 - younger knight
20 - female swordswoman (Ivanna's former instructor)
21 - female (looks like a priest or something)
22 - demonic looking beast
23 - demonic looking beast
24 - fallen angel
25 - Standard Male Soldier
26 - Standard Female Soldier
27 - Standard Male Ninja

28 - Standard Female Ninja
29 - Standard Male Archer
30 - Standard Female Archer
31 - Standard Male Wizard
32 - Standard Female Wizard
33 - Standard Male Cleric
34 - Standard Female Cleric
35 - Stan (the Bandit)
36 - Mullin (the Wizard)
37 - Enemy Female Archer
38 - Enemy Male Knight (I think ...)
39 - Enemy Male Ninja (or possibly Dark Stalker)
40 - Enemy Beast Tamer
41 - Enemy Cleric
42 - Enemy Dragoon (or possibly Predator)
43 - Cybil (in enemy colors, I think)
44 - Lich (possibly in enemy colors, it's hard to tell)
45 - Angel Knight
46 - some sort of demon (possibly Saya)
47 - Mermaid
48 - Duke Knight
49 - Standard Male Soldier
50 - Standard Female Soldier
51 - Stan (but the palette is messed up)
52 - Female Knight (but the palette is messed up)
53 - Duke Knight (but the palette is messed up)
54 - Unknown Female (blue version)
55 - Enemy Female Ninja
56 - Unknown Female (green version of 54)
57 - Elric (in enemy colors)
58 - Swordmaster
59 - Female Priest (but in yellow ...)

Beyond that, you get garbage graphics. Again, I can recognize less than half - if someone want to help me out here, I'd appreciate it.

Section 12: Class Code Table

---

0 - None
1 - Soldier, Male
2 - Soldier, Female
3 - Ninja, Male
4 - Ninja, Female
5 - Archer, Male
6 - Archer, Female
7 - Wizard, Male
8 - Wizard, Female
9 - Cleric, Male
10 - Cleric, Female
11 - Knight, Male
12 - Knight, Female
13 - Swordmaster
14 - Dragoon
15 - Valkyrie
16 - Siren
17 - Warlock
18 - Witch
19 - Priest, Male

20 - Priest, Female
21 - Dragon Tamer
22 - Beast Tamer
23 - Hawkman
24 - Mermaid
25 - Fairy
26 - Lich
27 - Angel Knight, Male
28 - Angel Knight, Female
29 - Ghost
30 - Gorgon
31 - Gremlin
32 - Daemon
33 - Duke Knight
34 - Dark Angel
35 - Dark Angel, Female
36 - Thunder Dragon
37 - Red Dragon
38 - Earth Dragon
39 - Blue Dragon
40 - Mushus
41 - Naga
42 - Vrtra
43 - Dragon Zombie
44 - Griffin
45 - Cockatrice
46 - Octopus
47 - Cerberus
48 - Giant
49 - Bandit
50 - Undead Soldier, Male
51 - Undead Soldier, Female
52 - Undead Wizard, Male
53 - Undead Wizard, Female
54 - Undead Knight, Male
55 - Undead Knight, Female
56 - Doll (dummy)
57 - Dark Stalker
58 - Hell Gigantes
59 - Predator

Avoid 60 - 65. I think it's reading from non-class related data, so you'll get odd performance. 60 - 63 just flat out cause errors; 64 - 65 may defer to the alternate Class byte.

66 - Sorceress (no pic, probably for Cybil)
67 - High Priest (no pic, probably for Rictor)
68 - Mermaid (no pic, probably for the special Mermaid)
69 - Summoner (has Elric's picture, oddly enough)
70 - Shaman (has Euphaire's picture, oddly enough)
71 - Angel Knight (no pic, probably for the special Angel Knight)
72 - Fairy (no pic, probably for Glycinia)
73 - Fairy (no pic, probably for Lubinia)
74 - Lesser Daemon (no pic, probably for Saya)
75 - Ench (dummy)
76 - Witch (no pic, for Deneb)
77 - General (no pic)
78 - Esquire (no pic)
79 - Swordmaster (no pic, female, special version)
80 - Venefic (no pic)

81 - Venefica (no pic)
82 - ??? (Unknown) [literally what it says. It's for Rimmon.]
83 - Fallen Angel
84 - Sacred Demon

    While 85 supposedly gives you a male soldier (for the picture), the class is actually a Fire Dragon. In other words ... don't use it.

Section 13: Element, Alignment, and Sex Code Tables

Element Codes:

0 - None
1 - Wind
2 - Fire
3 - Earth
4 - Water
5 - Virtue
6 - Bane

Alignment Codes:

0 - None
1 - Lawful
2 - Netural
3 - Chaotic

Sex Codes:
0 - None
1 - Male
2 - Female

(Beyond this, you start reading city names; Lutra, Solea, Scabellum ...)

Section 14: Month Code Table
---

0 - None
1 - Deus
2 - Tierra
3 - Agua
4 - Sombra
5 - Branca
6 - Flama
7 - Vento
8 - Ouro
9 - Trueno
10 - Trevas
11 - Oceano
12 - Preta
13 - Gemeo
14 - Fogo
15 - Raio

Section 15: Item Code Table

The codes are identical to the codes used in the Item Hacking Guide. However, they are reproduced here for ease of use.

0 - Nothing
1 - Short Sword
2 - Long Sword
3 - Claymore
4 - Sum Mannus
5 - Firedrake Sword
6 - Ice Blade
7 - Fafnir
8 - Sword of Tiamat
9 - Balmung
10 - Notos
11 - Laevateinn
12 - Gram
13 - Oracion
14 - Fragarach
15 - Anbicion (fully powered version)
16 - Anbicion ('Anbicion' effect version)
17 - 'None' (Wind Snapdragon Sword)
18 - 'None' (Fire Snapdragon Sword)
19 - 'None' (Earth Snapdragon Sword)
20 - 'None' (Water Snapdragon Sword)
21 - 'None' (Virtue Snapdragon Sword)
22 - 'None' (Bane Snapdragon Sword)
23 - Matsukaze
24 - Kagari-bi
25 - Yomogi-u
26 - Yu-giri
27 - Rapier
28 - Estoc
29 - Dragon Gem Sword
30 - Inca Rose
31 - Peridot Sword
32 - Needle of Light
33 - Answerer
34 - Francisca
35 - Prox
36 - Earth Dragon Axe
37 - Frozen Axe
38 - Bloody Cleaver
39 - Boreas
40 - Halt Hammer
41 - Euros
42 - Flame Flail
43 - Sanscion
44 - Hammer of Tears
45 - Mystic Hammer
46 - Battle Fan
47 - Caldia
48 - Hyacinth Fan
49 - Gypsy Queen
50 - Scipplay's Staff
51 - Wind Wand
52 - Fire Wand
53 - Earth Wand
54 - Ice Wand
55 - Ripple's Staff

56 - Kerykeion
57 - Dowsing Rod
58 - Sugar Cane
59 - Pike
60 - Trident
61 - Zephyrus
62 - Volcaetus
63 - Earth Javelin
64 - Osric's Spear
65 - Longicolnis
66 - Brionac
67 - Leather Whip
68 - Beast Whip
69 - Holy Comet
70 - Rapture Rose
71 - Short Bow
72 - Great Bow
73 - Thunder Bow
74 - Flame Bow
75 - Sandstorm Bow
76 - Tundra Bow
77 - Crescente
78 - Sherwood Bow
79 - Bow Gun
80 - Composite Bow
81 - Tathlum
82 - Tower Shield
83 - Dragon Shield
84 - Thunder Shield
85 - Flame Shield
86 - Earth Shield
87 - Ice Shield
88 - Saint's Shield
89 - Dark Shield
90 - Chocolate Shield
91 - Hard Leather
92 - Chain Mail
93 - Thunder Chain
94 - Flame Leather
95 - Earth Leather
96 - Ice Chain
97 - Saint's Garb
98 - Cursed Garment
99 - Plate Mail
100 - Heavy Armor
101 - Peregrine Mail
102 - Phoenix Mail
103 - Nathalork Mail
104 - Leviathan Mail
105 - Rune Plate
106 - Black Armor
107 - Southern Cross
108 - Dragon Armor
109 - Grincer Coat
110 - Candy Armor
111 - Brigandine
112 - Robe
113 - Spell Robe
114 - Robe of the Wise
115 - Wind Garb

116 - Fire Garb
117 - Earth Garb
118 - Water Garb
119 - Cloak of Oath
120 - Robe of Abyss
121 - Fur Coat
122 - Pure-White Dress
123 - Cloak of Authority
124 - Leather Hat
125 - Bandanna
126 - Plumed Headband
127 - Iron Helm
128 - Goblin Helm
129 - Holy Crown
130 - Freude Helm
131 - Dragon Helm
132 - Sherwood Hat
133 - Candy Helm
134 - Pointy Hat
135 - Circlet of Wisdom
136 - Winged Shoes
137 - Warp Shoes
138 - Greasy Boots
139 - Snow Boots
140 - Forest Boots
141 - Ring of Flight
142 - Warp Ring
143 - Ring of Flotation
144 - Armlet of Wisdom
145 - Armlet of Agility
146 - Wind Ring
147 - Firedrake Ring
148 - Earth Ring
149 - Water Ring
150 - Sacred Ring
151 - Dark Ring
152 - Dragon Eyes
153 - Ring of the Dead
154 - Necklace of Resist
155 - Pearl Necklace
156 - Amulet
157 - Dragon Gem
158 - Cassowary Feather
159 - Glass Pumpkin
160 - Firecrest
161 - Sacrificial Doll
162 - Sacred Stone of Bliss
163 - Transferring Stone
164 - Seraph's Plume
165 - Healing Leaf
166 - Healing Seed
167 - Healing Salve
168 - Healing Essence
169 - Magic Leaf
170 - Magic Seed
171 - Magic Salve
172 - Magic Essence
173 - Wisdom Fruit
174 - Angel Fruit
175 - Revive Stone

176 - Antidote
177 - Spirit Fruit
178 - Orb
179 - Savage Bugle
180 - Coral Harp
181 - Snapdragon
182 - Reincarnation
183 - Sword Emblem
184 - Crown of Intellect
185 - Stone of Swiftness
186 - Tome of Discipline
187 - Urn of Chaos
188 - Mirror of the Gods
189 - Cup of Life
190 - Sorcerer's Cup
191 - Altar of Resurrection

I'm not 100% sure how the game, therefore, handles the use of Snapdragon created swords.

Section 16: Skill Code Table

---

0 - None
1 - Thunder Bird
2 - Thunder Blade
3 - Air Blade
4 - Teleport
5 - Summon Tempest
6 - Harnella's Influence
7 - Haste
8 - Salamander
9 - Firestorm
10 - Fireball
11 - Clear Sky
12 - Ray of Paralysis
13 - Zoshonel's Influence
14 - Molten Blade
15 - Gnome
16 - Acid Vapor
17 - Crag Crush
18 - Petrifying Cloud
19 - Hurdle Wall
20 - Berthe's Influence
21 - Constrain
22 - Fenrir
23 - Ice Javelin
24 - Ice Field
25 - Slumber Mist
26 - Grueza's Influence
27 - Poison Squall
28 - Purify
29 - Star Tiara (dummy)
30 - Exorcism
31 - Lightning Bow
32 - Divine Radiance
33 - Tranquilize
34 - Faith
35 - Cleanse

36 - Heal
37 - Heal Plus
38 - Full Heal
39 - Resurrection
40 - Ignis Fatuus
41 - Fiend's Grip
42 - Nightmare
43 - Pain (dummy)
44 - Brain Sap
45 - Enfeeble
46 - Cursed Existence
47 - Fluid Magic
48 - Time Flux
49 - Necromancy
50 - Thunder Breath
51 - Fire Breath
52 - Poison Breath
53 - Cold Breath
54 - Mesmerize
55 - Petrifying Breath
56 - Rotten Breath
57 - Strangling Tentacles
58 - Windstorm
59 - Titan Crush
60 - Hell's Gate
61 - Anbicion (steal souls)
62 - Anbicion (use HP, unlock power)
63 - Summon Darkness
64 - Summon Golem
65 - Energy Transfer
66 - Barren Soul
67 - Swallow's Daze
68 - Fascination
69 - Temptation
70 - Poignant Memory
71 - Banish
72 - Thunder Arrow
73 - Cheer
74 - Lullaby
75 - Fairy's Embrace
76 - Fairy's Kiss
77 - Magic Missile
78 - Magic Barrage
79 - Fairy's Embrace (dark Fairy)
80 - Fairy's Kiss (dark Fairy)
81 - Evil Eye
82 - Ice Requiem
83 - Day of Reckoning
84 - Apocalypse
85 - Cataclysm
86 - Descent (the version used by the Sacred Demon)
87 - Atropos
88 - Star Tiara (the correct version Eleanor uses in the game)
89 - Clotho
90 - Descent
91 - Lachesis
92 - Shuriken Barrage
93 - Pelting Fury

Section 17: Supplement: Memory, Binary, and Hexidecimal

Okay, a few words before we begin. Why am I including this, and why here? This is a little guide to memory, binary, hexidecimal, and how and why hacking codes work. People ask me a lot of questions about the basics, and I figure that since this guide has almost all of the different type of codes in one place, it is an excellent place to put it.

This information is probably covered in a much more professional manner in your local mathematics textbook and computer science course. Of course, since a lot of gameplayers are in high-school, they may not have a decent computer science department. Not to fret.

Let's start by talking about numbers. Most of us are familiar with this subject, but let's review a little, shall we? Numbers are made up of numerals (the representations of numbers, which in the English alphabet are written as 0, 1, 2, 3, 4, 5, 6, 7, 8 and 9. We could have just as well used different smiley faces, pictures of trees, or letters.) While there is no problem when writing a number from 0 to 9, what happens when we want to write a 10? Well, as you can see, we use the concept of digits. The '1' in the number '10' is in the tens place. In the number '42,375', the number 2 is in the thousands place, and so on.

You of course know this, otherwise ... well, you know all this. But what we really mean when we write '42,375' is:

"I want the number whose value equals 4 X 10,000 + 2 X 1,000 + 3 x 100 + 7 x 10 + 5 x 1." 10 is, of course, 10 to the first power; 100 is 10 to the second power; 1000 is 10 to the third power, and so on and so forth.

This system of representing numbers is known as decimal. (From the Greek for 10.) But what if we don't want to use decimal?

To a computer, decimal is far to hard and inefficient. Computers understand two basic things: On (electrons flowing) and Off (electrons not flowing). If On = 1 and Off = 0, we can still represent any number we like. This is known as binary. So, we can write '27' in binary as 11011. (That's 1 x 16 + 1 x 8 + 0 x 4 + 1 x 2 + 1 x 1.) 16 is 2 to fourth power, 8 is 2 to the third power, etc. etc.

Hopefully, you understand binary now. Hexidecimal is similar, except instead of using 10 or 2 as a base, we use 16. In order to represent 10, 11, 12, 13, 14, and 15 in a single symbol, we use A, B, C, D, E, and F respectively. (16 in hexidecimal is, of course, 10.) So 100 in decimal is 6 x 16 + 4 or 64 in hexidecimal.

Why even bother with hexidecimal? Ahh ... well, historically, we group 8 binary switches into a single unit known as a byte. (10011011, for example.) Therefore, a byte can store values from 255 (1111111) to 0 (00000000). It so happens that 255 is equivalent to FF in hexidecimal. See where we're going with this?

A two-digit hexidecimal value, then, can represent all the values that a byte can. This is why we use hexidecimal extensively in computers.

Okay, so now we know about hexidecimal and why we use it in computers. Memory addresses are written in hexidecimal. Why? Well, a computer needs to be able to manipulate that address and store it. As a number. How does a computer store numbers? (If you said in binary, you are right.) What's the

way we write binary numbers? (If you said in hexidecimal, you are again right.)

Now, we come to the interesting part. How is data stored in memory? (Not physically, but conceptually.) Well, we store all sorts of things in a game's memory. If we want to store a number, we usually store it in a single byte (if it runs from 0-255), or 2 bytes (0-65535) or 4 bytes (basically anything bigger, up to 4,294,967,295.) This is different if you want to store potentially negative values (and I won't get into it, we rarely run into negative numbers in game memory.) Things like HP, EN, how many lives you have left, the number of shots in that rifle, etc. are usually stored in this manner.

We could also store it, instead of in true hexidecimal, as though it were decimal (since hexidecimal includes 0-9. I believe this is referred to as Binary Coded Decimal.) This tends to waste space (you ignore the power of A-F), but sometimes this is done in games. Not so much nowadays, though. So, if I stored 142 in 2-bytes, in the 'value' I'm really storing is 1 in the first of those two bytes, and the decimal value of 66 in the second. (I omit a discussion of byte-ordering until another time, simply because there are two different ways to store a multi-byte number. Don't worry about it for the present.)

We can also store a value that corresponds to something else (like a pilot's background data.) If information is kept like this, the game knows (it's built-in) what, say, a 12 stands for. We have to plug in values to figure out what each value stands for and make long lists. Sometimes, in cases like these, a 0 is empty (doesn't correspond to something, like in Tactics Ogre: Knight of Lodis). Sometimes (like in Super Robot Taisen A) it does stand for something (Pilot 0 is Lalah Sune.)

Finally, we can use memory in a BINARY fashion. Remember: FF is REALLY 11111111. Or 8 little switches in a row than can be ON or OFF. So if we want to look at things that can be on or off (like the Emblem Toggle), we can cram 8 of those things into one byte, instead of using 8 bytes. The 'Enable Bytes' are variations on this. Basically, a 0 stands for Off, and anything else is On. (The reason for this lies within the realm of assembly language, and will not be covered here. Sufficed to say, the game has a way of quickly checking if something is on or off, and only 0 stands for off.) This type of data, known as a bit field, is very commonly used for certain on-off purposes.

Well, this ends the brief supplement of Binary, Hexidecimal, and how memory is used in most games. Hopefully, this answers a few questions.

Section 18: Credits

There are several people without whose publicly available resources this document could have never been complied:

GameFAQs (www.gamefaqs.com), for being the comprehensive game information site.

Special thanks goes to Terence Fergusson for writing a very detailed Biorhythm Mechanics FAQ, which helped in decoding which bytes would do what. Otherwise, I'd probably never have figured out the exact addresses.

Also, special thanks goes to Waverlyt and Waspinator, who provided plenty

of information on pictures as well as doing extensive play-testing for these codes. Both have given this project much more impetus than I originally planned ^^

Section 19: Copyright / Authorization

Section 20: Miscellaneous

Several sections of this guide are copied directly from my previous hacking guides for Super Robot Taisen A. (Especially the supplement.) If you really want to know more about how computer memory and hacking into it works, consult your local computer science professional, or try the local library.

Next?:
If there are certain game logic questions, I will be happy to answer them (and probably add an explanation in the guide.) I am NOT, however, going to answer questions on how to specifically apply these codes to get specific results, or on how to cheat in general - there are probably plenty of guides written on that subject.

Both of the special play-testers and myself are looking into the exact mechanics of the Class Byte, Alternate Class Byte, and Picture Bytes. Eventually, I plan on updating this information and adding explanations to the Game Logic section.