

- コード名の表記
- Constant Write(連続書き込み)
- In/Decrement(加減算コード)
- Test Code(条件判定コード)
  - Multiple Skip
  - Adress Compare+Multiple Skip(ver0.2.1A以降)
- Multi Address Writes(シリアルコード)
- Boolean Commands(論理演算コード)
- Pointer Commands(ポインタコード)
  - Reverse Pointer(逆ポインタコード ver0.1.8 REV. C以降)
  - Extra Pointer(追加オプション, Ver0.2.1以降, nnnn>1の場合)
- Joker Code(パッドコード)
  - Inverse Joker Code
- Copy bytes(コピー)
- Code stopper(停止コード)
- Time command(遅延コード)
- FREECHEAT Special Format
  - MULTI(HIGH LV) POINTER(080323 later version)
- CheatMasterFusion Special Format
  - CODE Encryption(R19D ~)
  - Restore Value(R20 ~)

## コード名の表記

\_S            ゲームID  
 \_G            ゲームタイトル  
 \_C0 コード名自動実行しない  
 \_C1 コード名自動実行する

[http://cwcheat.myconsole.it/wiki/doku.php?id=english:code:psp\\_types](http://cwcheat.myconsole.it/wiki/doku.php?id=english:code:psp_types)

## Constant Write(連続書き込み)

8bit 0x0aaaaaaaa 0x000000bb アドレスaaaaaaaaに値bbを書き込み続ける  
 16bit 0x1aaaaaaaa 0x0000bbbb アドレスaaaaaaaaに値bbbbを書き込み続ける  
 32bit 0x2aaaaaaaa 0xbbbbbbbb アドレスaaaaaaaaに値bbbbbbbbを書き込み続ける

## In/Decrement(加減算コード)

8bit 0x301000nn 0x0aaaaaaaa アドレスaaaaaaaaの値にnnを加算し続ける  
      0x302000nn 0x0aaaaaaaa アドレスaaaaaaaaの値からnnを減算し続ける  
 16bit 0x3030nnnn 0x0aaaaaaaa アドレスaaaaaaaaの値にnnnnを加算し続ける  
      0x3040nnnn 0x0aaaaaaaa アドレスaaaaaaaaの値からnnnnを減算し続ける  
      0x30500000 0x0aaaaaaaa アドレスaaaaaaaaの値にnnnnnnnnを加算し続ける  
 32bit 0xnnnnnnnn 0x00000000  
      0x30600000 0x0aaaaaaaa アドレスaaaaaaaaの値からnnnnnnnnを減算し続ける  
      0xnnnnnnnn 0x00000000

## Test Code(条件判定コード)

	0xDaaaaaaa 0x20000dd	アドレスaaaaaaaの値がddと等しければ下のコード実行
8bit(ver0.1.4以降)	0xDaaaaaaa 0x201000dd	アドレスaaaaaaaの値がddと等しくなければ下のコード実行
	0xDaaaaaaa 0x202000dd	アドレスaaaaaaaの値がddより小さければ下のコード実行
	0xDaaaaaaa 0x203000dd	アドレスaaaaaaaの値がddより大きければ下のコード実行
16bit	0xDaaaaaaa 0x0000dddd 0x0010dddd	アドレスaaaaaaaの値がddddと等しければ下のコード実行
	0xDaaaaaaa 0x0020dddd	アドレスaaaaaaaの値がddddより小さければ下のコード実行
	0xDaaaaaaa 0x0030dddd	アドレスaaaaaaaの値がddddより大きければ下のコード実行

### Multiple Skip

	0xE0nndddd 0x0aaaaaaa	アドレスaaaaaaaの値がddddと一致するとき下のnn行分のコード実行
16bit	0xE0nndddd 0x1aaaaaaa	アドレスaaaaaaaの値がddddと一致しないとき下のnn行分のコード実行
	0xE0nndddd 0x2aaaaaaa	アドレスaaaaaaaの値がddddより少ないとき下のnn行分のコード実行
	0xE0nndddd 0x3aaaaaaa	アドレスaaaaaaaの値がddddより大きいとき下のnn行分のコード実行
8bit(ver0.1.9 REV.I以降)	0xE1nn00dd 0x0aaaaaaa	アドレスaaaaaaaの値がddと一致するとき下のnn行分のコード実行
	0xE1nn00dd 0x1aaaaaaa	アドレスaaaaaaaの値がddと一致しないとき下のnn行分のコード実行
	0xE1nn00dd 0x2aaaaaaa	アドレスaaaaaaaの値がddより少ないとき下のnn行分のコード実行
	0xE1nn00dd 0x3aaaaaaa	アドレスaaaaaaaの値がddより大きいとき下のnn行分のコード実行

### Address Compare+Multiple Skip(ver0.2.1A以降)

Address Equal	0xDaaaaaaa 0x4bbbbbbb 0x000000nn 0x0000000Y	アドレスaaaaaaaにある値とアドレスbbbbbbbの値が一致するとき 下のnn行分のコード実行 Y =0(8bit), 1(16bit), 2(32bit)
Address Not Equal	0xDaaaaaaa 0x5bbbbbbb 0x000000nn 0x0000000Y	アドレスaaaaaaaにある値とアドレスbbbbbbbの値が一致しないとき 下のnn行分のコード実行 Y =0(8bit), 1(16bit), 2(32bit)
Address Less Than	0xDaaaaaaa 0x6bbbbbbb 0x000000nn 0x0000000Y	アドレスaaaaaaaにある値よりアドレスbbbbbbbの値が多いとき 下のnn行分のコード実行 Y =0(8bit), 1(16bit), 2(32bit)
Address Greater Than	0xDaaaaaaa 0x7bbbbbbb 0x000000nn 0x0000000Y	アドレスaaaaaaaにある値よりアドレスbbbbbbbの値が少ないとき 下のnn行分のコード実行 Y =0(8bit), 1(16bit), 2(32bit)

---

### Multi Address Writes(シリアルコード)

8bit(ver0.1.6以降)	0x8aaaaaaaa 0xbbbbyyyy 0x000000dd 0x000000ee	アドレスaaaaaaaaからbbbb個のアドレスになるようにyyyyアドレスずつずらしながら値ddを書き込み続ける このときddもee分ずつ増加して書き込む
16bit(ver0.1.6以降)	0x8aaaaaaaa 0xbbbbyyyy 0x1000dddd 0x0000eeee	アドレスaaaaaaaaからbbbb個のアドレスになるようにyyyy*2アドレスずつずらしながら値ddddを書き込み続ける このときddddもeeee分ずつ増加して書き込む
32bit	0x4aaaaaaaa 0xbbbbyyyy 0xdddddddd 0xeeeeeeee	アドレスaaaaaaaaからbbbb個のアドレスになるようにyyyy*4アドレスずつずらしながら値ddddddddを書き込み続ける このときddddddddもeeeeeeee分ずつ増加して書き込む

### Boolean Commands(論理演算コード)

8bit	OR 0x7aaaaaaaa 0x000000vv	アドレスaaaaaaaaの値とvvの論理和を書き込む
	AND 0x7aaaaaaaa 0x000200vv	アドレスaaaaaaaaの値とvvの論理積を書き込む
	XOR 0x7aaaaaaaa 0x000400vv	アドレスaaaaaaaaの値とvvの排他的論理和を書き込む
16bit	OR 0x7aaaaaaaa 0x0001vvvv	アドレスaaaaaaaaの値とvvvvの論理和を書き込む
	AND 0x7aaaaaaaa 0x0003vvvv	アドレスaaaaaaaaの値とvvvvの論理積を書き込む
	XOR 0x7aaaaaaaa 0x0005vvvv	アドレスaaaaaaaaの値とvvvvの排他的論理和を書き込む

### Pointer Commands(ポインタコード)

8bit	0x6aaaaaaaa 0x000000vv	アドレスaaaaaaaaの値を基準アドレスとし 値vvを基準アドレス-0x8800000+iiiiiiiに書き込む
16bit	0x6aaaaaaaa 0x0000vvvv 0x00010001 0xiiiiiii	アドレスaaaaaaaaの値を基準アドレスとし 値vvvvを基準アドレス-0x8800000+iiiiiiiに書き込む
32bit	0x6aaaaaaaa 0xvvvvvvvv 0x00020001 0xiiiiiii	アドレスaaaaaaaaの値を基準アドレスとし 値vvvvvvvvを基準アドレス-0x8800000+iiiiiiiに書き込む

### Reverse Pointer(逆ポインタコード ver0.1.8 REV. C以降)

8bit	0x6aaaaaaaa 0x000000vv	アドレスaaaaaaaaの値を基準アドレスとし 値vvを基準アドレス-0x8800000-iiiiiiiに書き込む
16bit	0x6aaaaaaaa 0x0000vvvv 0x00040001 0xiiiiiii	アドレスaaaaaaaaの値を基準アドレスとし 値vvvvを基準アドレス-0x8800000-iiiiiiiに書き込む
32bit	0x6aaaaaaaa 0xvvvvvvvv 0x00050001 0xiiiiiii	アドレスaaaaaaaaの値を基準アドレスとし 値vvvvvvvvを基準アドレス-0x8800000-iiiiiiiに書き込む

### Extra Pointer(追加オプション, Ver0.2.1以降, nnnn>1の場合)

type null	0x6aaaaaaaa 0xvvvvvvvv 0xqqq2nnnn 0xiiiiiii 0x00000000 0x00000000	nnnn>1のときqqqを使いたい場合入れる必要となる? アドレスaaaaaaaa+qqq*4(nnnn-1)の値を基準アドレス Nとする
-----------	---	---

	0x6aaaaaaaa		
	0x000000vv		nnnn>1のときオフセットiiiiiiiをssssssずつずらしなが
	0xqqq0nnnn	0xiiiiiii	ら書き込む
	0x9sssssss		vvもww分ずつ増加して書き込まれる。
	0x000000ww		
	0x6aaaaaaaa		
	0x0000vvvv		nnnn>1のときオフセットiiiiiiiをssssss*2ずつずらしな
multi adress write	0xqqq1nnnn	0xiiiiiii	がら書き込む
	0x9sssssss		vvvvもwwww分ずつ増加して書き込む
	0x0000wwww		
	0x6aaaaaaaa		
	0xvvvvvvvv		nnnn>1のときオフセットiiiiiiiをssssss*4ずつずらしな
	0xqqq2nnnn	0xiiiiiii	がら書き込む
	0x9sssssss		vvvvvvvもwwwwwww分ずつ増加して書き込む
	0xwwwwwww		
	0x6aaaaaaaa		
	0xvvvvvvvv		
copy byte	0xqqq00002	0xiiiiiii	基準アドレス-0x8800000+iiiiiiiの値を基準アドレス
	0x1sssssss		2-0x8800000+sssssssにvvvvvvv分コピー
	0x00000000		
			aaaaaaa=第1ベースアドレスがあるポインタアドレ
			ス、vvvvvvvv=最終到達アドレスに書き込む値
			t=最終到達アドレスに書き込むbit数とzzzzの加減方向
	0x6aaaaaaaa		0~2が加算の0=8bit, 1=16bit, 2=32bit, 3~5が減算の
	0xvvvvvvvv		8bit,16bit,32bit
multi	0xqqqt00nn		nn=追跡回数、zzzz=最後に足す/引くオフセット値(最
pointer(0.2.2REVA以	0x0000zzzz		終書込アドレス=第nnベースアドレス
降)	0xS000iiii		-0x8800000+/-zzzzになる)
	0x00000000		S=オフセットの加減 S=2でプラス, S=3でマイナス、
	0xS000jjjj ...		iiii=第1ベースから足す/引くオフセット値
	0xS000yyyy ...		これより下の行は3回以上の追跡をする場合追跡回数
			に応じて追加
			jjjj=第2ベースから足す/引くオフセット値
			yyyy=第(nn-1)ベースから足す/引くオフセット値

## Joker Code(パッドコード)

0xD00000dd      ボタンを押している間dd+1行分下のコード実行 nnnnはパッドの合計  
0x1000nnnn      値

## Inverse Joker Code

0xD00000dd      ボタンを押している間dd+1行分下のコード無効 nnnnはパッドの合計  
0x3000nnnn      値

ボタン	パッド値(=nnnn)
セレクト	0x0001
スタート	0x0008
上	0x0010
右	0x0020
下	0x0040
左	0x0080
L	0x0100
R	0x0200
	0x1000
	0x2000
x	0x4000

0x8000  
HOME 0x10000  
HOLD 0x20000  
NOTE 0x800000  
SCREEN 0x400000  
VOLUME UP 0x100000  
VOLUME DOWN 0x200000  
WLAN UP 0x40000  
REMOTE HOLD 0x80000

---

### Copy bytes(コピー)

0x5aaaaaaa 0xnccccccc アドレスaaaaaaaからbbbbbbbbにnnnnnnnnバイト分コピー  
0xbbbbbbbb 0x00000000

### Code stopper(停止コード)

0xCaaaaaaa 0xvvvvvvvv アドレスaaaaaaaの値がvvvvvvvvでないときコードを停止

### Time command(遅延コード)

0xB0000000 0xnccccccc nnnnnnnn分だけ遅らせる,0x10で1秒

---

## FREECHEAT Special Format

### MULTI(HIGH LV) POINTER(080323 later version)

	aaaaaaa is pointer address,the first baseaddress position
	t = write bit, see CWC
	vvvvvvv=write value
	u=add/substract offset.6 is adding,7 is substracting
	iiii=add/substract offset to 1st baseaddress
	jjjj=add/substract offset to 2nd baseaddress
	kkkk=add/substract offset to 3rd baseaddress
	....
	zzzz=add/substract offset to last baseaddress
	....
	write a final address=last baseaddress-0x8800000+/-zzzz

# Multi address write and Copy bytes can be used with multi pointer.

---

## CheatMasterFusion Special Format

### CODE Encryption(R19D ~)

Encrypted 0xF0XXXnn next nn lines asm subroutine code is executed.zz=00 raw codes 0xYYYYYYzz asm code,other is encrypted.

### Restore Value(R20 ~)

Restore 0xCaaaaaaa this code is same "type 0x2" 32bit write,difference is restoring codes 0xbbbbbbbb original value when you unlock code.