Doom 64 Tech Bible

Samuel 'Kaiser' Villarreal

Revision 1.0 – 05/22/2011

1 – Introduction

1.1 What is Doom 64

Developed by Midway - San Diego, Doom 64 is a unofficial sequel to Doom II developed for the Nintendo 64. Based on the Jaguar and Playstation codebase, Doom 64 is considered the most technically advanced Doom game of its time. In addition to the completely revamped rendering engine, Doom 64 went through numerous changes and improvements from that of Doom II. No graphics or sounds from Doom II have been reused in Doom 64; instead, it features completely new sprites, textures, levels and sounds. There are new sprites for enemies and player weapons as well as a completely different art style.

1.2 About this Document

The goals of this document are to cover all of the technical aspects of Doom 64 and to compare the differences in gameplay and level format.. Other technical information like Doom 64's rendering system will also be covered. This document will also assume that the end user already has a firm knowledge of Doom II's lump and wad format and at least basic technical knowledge of Doom's inner workings. Only the things specific to Doom 64 will be covered.

1.3 Authors and Contributors

<Text goes here>

2 – WAD Format

2.1 Lump Markers

Doom 64 still uses S_START and S_END for sprites but in addition to that, it uses T_START and T_END for textures. The last lump in the IWAD is a unique marker labeled ENDOFWAD which serves no purpose other than to act as additional padding in the IWAD.

2.2 Lump Compression

Like the Jaguar and Playstation versions, Doom 64 uses a LZSS compression algorithm for its lumps to preserve ROM space. In Doom 64, however, a second compression algorithm is used for the levels, demos, and textures, which is similar to the Huffman algorithm, using a lookup table for compressed bytes. The algorithm to decompress these lumps has been identified and is used in the Wadgen utility for Doom64 Ex (see deflate64.c)

Lumps are identified as compressed if the high bit of the first character of the lump name is set. Example:

```
ÓARGA2A8 – Compressed - (charname[0] & 0x80) == true
SARGA2A8 – Not compressed
```

W_CacheLumpNum and W_ReadLump have added an additional argument which defines what compression type to use: 0 = no compression, 1 = LZSS, 2 = Huffman

2.3 Map Lumps (Pwads)

Doom 64 has no tolerance for lumps with duplicate names, which is a problem for level-specific lumps such as THINGS, LINEDEFS, SSECTORS, etc. Levels are stored in separate PWADS, which are then stored within the IWAD and are treated as individual lumps. When the P_SetupLevel routine is called, the level lump is parsed and a secondary WAD handler reads the contents and grabs the level lumps.

Example of how level-specific lumps are read:

```
void* W_GetMapLump(int lump)
{
    if(lump >= numMapLumps)
        I_Error("W_GetMapLump: lump %d out of range", lump);

    return (mapLumpData + mapLump[lump].filepos);
}
```

2.4 Byte Alignment

On the Nintendo 64, lump data must be aligned to a 4-byte boundary; otherwise, Doom 64 will fail to read the lumps. The size of that lump in the lumpinfo_t structure will still contain the 'true' size.

3 – Level Format

3.1 Level Lumps

There are 14 lumps which make up a level (or map). Below is the list of each lump and a description for each:

Name	Description
MAPXX	Header for level. All level-specific lumps must follow after it.
THINGS	Represent players, monsters, items and other objects
LINEDEFS	Wall shared by two vertices
SIDEDEFS	Contains the wall texture data for each linedef
VERTEXES	X, Y coordinates. Used by linedefs. Extra vertices are generated by node builder.
SEGS	Segments of linedefs. Generated by node builder
SSECTORS	Range of segs, grouped to form a convex polygon. Generated by node builder
NODES	Constitutes a binary space partition of the level. Generated by node builder
SECTORS	An area referenced by sidedefs on the linedefs to make up the ceiling and floor.
REJECT	A resource table attached to levels which is used to speed up line-of-sight calculations. Generated by node builder
BLOCKMAP	A data structure used for collision detection. Generated by node builder
LEAFS	Contains groups of vertices and segs sorted in counter-clockwise order to form a convex polygon. Generated by node builder
LIGHTS	RGB table containing colored lighting data
MACROS	Data for scripted linedefs

3.2 Things

3.2.1 Structure

Size (bytes)	Description
2	X position
2	Y position
2	Z position
2	Facing angle
2	DoomEd thing type
2	Flags

2	Thing id (tid)

3.2.2 Flags

Bit	Description
0x1	Thing is on skill levels 1 & 2
0x2	Thing is on skill level 3
0x4	Thing is on skill levels 4 & 5
0x8	Deaf monsters/do not react to sound
0x10	Thing is not in single player
0x20	Don't spawn until triggered in level
0x40	Trigger tagged linedef matching TID when picked up
0x80	Trigger tagged linedef matching TID when killed
0x100	Count as secret for intermission when picked up
0x200	Ignore other attackers (No infighting)

3.3 Linedefs

3.3.1 Structure

Size (bytes)	Description
2	Start vertex
2	End vertex
4	Flags
2	Special flags
2	Sector tag
2	Left sidedef
2	Right sidedef

3.3.2 Flags

Bit	Description
0x1	Blocks players and monsters
0x2	Blocks monsters
0x4	Two sided
0x8	Unpeg upper texture
0x10	Unpeg lower texture
0x20	Secret – shows as one-sided wall in automap and cannot be interacted by enemies

0x40	Block sound
0x80	Never show on automap
0x100	Always show on automap
0x200	Show middle texture on a two-sided line
0x400	Line is not clipped against the occlusion buffer
0x800	Unpeg middle texture. Also blocks projectiles
0x1000	Linedef is triggered by a killed enemy. Line tag and thing TID must match
0x2000	Switch mask 1
0x4000	Switch mask 2
0x8000	Switch mask 3
0x10000	Check for player floor height. Checks for ceiling height if flag is not set
0x20000	Scroll texture to the right
0x40000	Scroll texture to the left
0x80000	Scroll texture up
0x100000	Scroll texture down
0x200000	Do color blending only on top texture
0x400000	Do color blending only on bottom texture
0x800000	Use upper/lower light color blendings. Uses thing color if flag is not set
0x1000000	Linedef can only be triggered from the front
0x2000000	Unknown/Unused
0x4000000	Reverse color blending
0x8000000	Unknown/Unused
0x10000000	Unknown/Unused
0x20000000	Unknown/Unused
0x40000000	Set texture wrap mode to horizontal mirror
0x80000000	Set texture wrap mode to vertical mirror

3.3.3 Special Flags

Unlike Doom1/2, the linedef special property contains both line types and behavior flags which specify how this linedef can be activated. Because of this setup, Doom 64 is limited to only 255 generic line specials and 255 macro specials.

To easily retrieve the line type ID, the following macro can be used with X = linedef->special:

```
#define SPECIALMASK(x) (x & 0x1FF)
```

Bit	Description

0x1 – 0xFF	Normal line type special
0x100 – 0x1FF	Macro line type special
0x200	Requires red key
0x400	Requires blue key
0x800	Requires yellow key
0x1000	Crossing line triggers
0x2000	Shooting line triggers
0x4000	Using line triggers
0x8000	Line can be re-triggered

3.3.4 Switch Masks

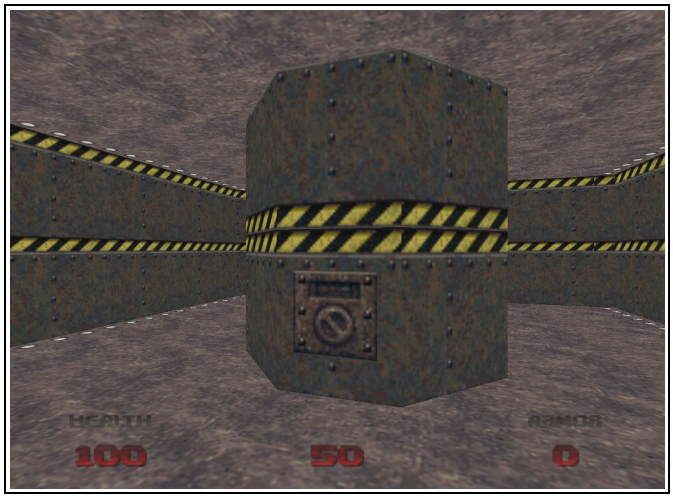


Figure 3.3.4a: Basic representation of a switch in Doom64

Doom 64 also uses bit flags on how switches are represented. Line flags 0x2000, 0x4000, 0x8000 and 0x10000 are used for switches. In Doom 64, switches are shown as 32x32 boxes that are drawn on top of either the top, middle or lower part of the linedef. The textures that the switches use are based on the switch flag, and where the texture is set on the sidedef. All names for textures used as switch boxes must begin with the SWX prefix.

The following table explains the different combinations and results for switches. 'Where drawn' means where the 32x32 box is drawn on the linedef and 'texture set' is where on the sidedef the switch expects to look up the switch texture. 'Bit' is what flags were set. The switch is also affected if the linedef is two-sided or not.

Bits Set	**Two-Sided**	**Where Drawn**	**Texture Set**
0x2000 + 0x10000	Yes	Bottom	Top
0x2000 + 0x8000 + 0x10000	No	Middle	Top
0x4000 + 0x8000	Yes	Top	Bottom
0x4000 + 0x8000 + 0x10000	No	Middle	Bottom

0x2000 + 0x4000 + 0x10000	Yes	Bottom	Middle
0x2000 + 0x4000 + 0x8000	Yes	Top	Middle

3.4 Sidedefs

Sidedefs underwent only minimal change: 8-character arrays are no longer used for referencing the top, bottom and middle textures. Instead, 2-byte indices are used to reference the texture ID.

3.5 Vertexes

Vertex X/Y data has been changed to 4-byte integers instead of 2-byte words and are also read in 16.16 fixed format. When LEAFS is compiled by the node builder, it creates additional vertices which are added to the VERTEXES lump. These extra vertices are exactly the same as vertices stored in the GL_VERTS lump used by OpenGL-based source ports.

3.6 Sectors

3.6.1. Structure

Size (bytes)	Description
2	Floor height
2	Ceiling height
2	Floor pic index
2	Ceiling pic index
10	Color table look up index
2	Type
2	Tag number
2	Flags

3.6.2. Flags

New to Doom 64, sectors now possess flags which define random behavior and actions for that sector. This may include sectors that damage the player, play reverb effects, or scroll/pan texture offsets.

Bit	Description
0x1	Reverb sounds played in sector
0x2	Reverb sounds played in sector and multiply factor by 2
0x4	Water effect (floors only)
0x8	All flagged sectors with same special type will sync all light flickers/strobe across all sectors
0x10	Scrolling floors/ceilings will scroll at twice the speed

0x20	Count as secret when entering this sector
0x40	Damage player x5
0x80	Damage player x10
0x100	Damage player x20
0x200	Hide sector in automap (textured mode)
0x400	Enable ceiling scrolling
0x800	Enable floor scrolling
0x1000	Scroll texture west
0x2000	Scroll texture east
0x4000	Scroll texture north
0x8000	Scroll texture south

3.7 Leafs

First featured in the Sony Playstation version of Doom, Leafs is a new lump generated by the node builder which stores information on how segs are sorted in order to convert subsectors into convex polygons. Each leaf is generated for each subsector and the number of leafs must match the number of subsectors otherwise Doom 64 will produce an error.

The leafs lump is very similar to how the GL-specific nodes are built using the GLBSP node builder. The only difference is that the data is stored in GL_VERT, GL_SEG, and GL_SSECT lumps. The leafs lump is basically those three lumps combined into one.

3.7.1. Structure

Each structure begins with a 2-byte variable which indicates how many vertices this leaf contains.

Size (bytes)	Description
2	Vertex reference ID
2	Seg reference ID (0xFFFF if miniseg)

Example of how leafs can be set up as a data structure for programming:

```
typedef struct
{
    int vertexID;
    int segID;

} leafchild_t;

typedef struct
{
    int leafcount;
    leafchild_t* children;
```

```
} leafparent_t;
```

3.8 Lights

The lights lump is a new level lump for Doom 64 which contains an RGB table for the colored lighting that's referenced by sectors. Each sector contains an array of 5 different RGB lookup values; each one pertaining to the ceiling, floor, thing and walls. By default the initial lookup ID starts at 256 because Doom 64 generates a default grayscale table that ranges from 0, 0, 0 to 255, 255, 255, followed by the actual RGB table.

3.8.1. Structure

Size (bytes)	Description
1	Red value
1	Green value
1	Blue value
1	Padding (not used)
2	Tag

3.8.2. Light Tags

In addition to the RGB values, there is another value which is somewhat similar to sector tags. Line special 234 is used to copy one RGB value to another RGB value from within the table. For example it will copy the RGB values of the light data of the specified tag to another light data with the same tag. The only place to see this feature is in Map30 (The Lair) and is not used anywhere else in the entire game.

3.9 Macros

The macros lump is a new level lump for Doom 64 that contains data for scripted events. More documentation covering the macro system is explained later in this document.

3.9.1. Header Structure

Size (bytes)	Description
2	Total events
2	Total actions
…	Begin macro events

3.9.2. Event Structure

Size (bytes)	Description

2	Number of actions in this event
...	Begin macro definition

3.9.3. Definition Structure

Size (bytes)	Description
2	Script line ID. Usually in multiples of 10
2	Sector tag
2	Line type/action

4 – Sprite Format

4.1 Color Format

Sprites in Doom 64 are stored in both RGB8 and RGB4 format (which is mostly used by sprites that aren't the player weapons or monsters). First RGB entry in the palette will always indicate the color in which to mask out the pixels.

4.2 Palette Format

The Nintendo 64 color table is 5 bits per pixel and each RGB entry is stored as 2 bytes instead of 3. The following pseudocode below demonstrates how to convert the Nintendo 64 RGB format into the traditional 8 bits per pixel:

```
b = (val & 0x003E) << 2;
g = (val & 0x07C0) >> 3;
r = (val & 0xF800) >> 8;
```

4.3 Palette Lookup

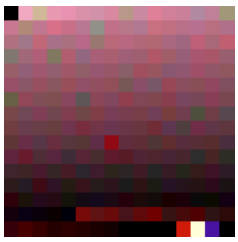
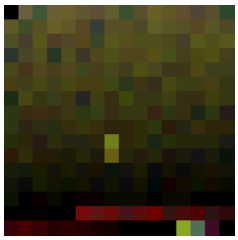
Every sprite lump also contains its own palette table (256 color table for RGB8 sprites and 16 color tables for RGB4 sprites). Monster sprites have the option of using external palette tables that are not embedded within the sprite lump. These external palettes are stored as individual lumps and are used to allow color swaps for certain monsters like the Imp and Nightmare Imp.

In order to lookup these palette lumps, all palette lumps are named in the following naming convention:

```
“PAL<sprite name><table index>”
```

And an example of a valid name for a external palette would be: “PALSARG0”. This palette would be looked up by a thing using the SARG sprite, which is the sprite of the Bulldog Demon and Specture, which shares the same sprite. In order to make them distinctive, they both used different palette tables.

The following table below demonstrates the effects of using multiple palettes for one sprite:

Palette Lump	SARGE2E8 Sprite (Result)
 **PALSARG0**	
 **PALSARG1**	

4.4 Lump Format

Size (bytes)	Description
2	Number of tiles the sprite is divided into
2	-1 for RGB4 format or 0 for RGB8 format
2	Unknown
2	Draw X offset
2	Draw Y offset
2	Draw width
2	Draw height
2	Tile height
...	Raw data
…	Palette data (if not using external palettes)

4.4.1. Tiles

Because the Nintendo 64 cannot render large-scale textures, sprites larger than 64x64 are broken up into smaller chunks. Some of the entries in the header define how many tile pieces into which the sprite should be broken up and the height for each tile.

4.5 Displaying the sprite

Again, due to some differences in how the Nintendo 64 hardware renders the sprites, the rows of the sprite's pixels are swapped (usually in groups of 4 pixels) and will need some simple conversion to swap the pixels in order to be properly viewed on the PC. Normally this only has to be done for the larger sprites.

Figure 4.4.2a: A typical Doom 64 sprite when extracted directly from the game. Larger sprites like these need to be processed in order to display on PC.

5 – Texture Format

5.1 Color Format

Unlike sprites, all textures in Doom 64 are in RGB4 only. First RGB entry in the palette that is at full 0, 0, 0 will always indicate the color in which to mask out the pixels.

5.2 Palette Format

The Nintendo 64 color table is 5 bits per pixel and each RGB entry is stored as 2 bytes instead of 3. The following pseudocode below demonstrates how to convert the Nintendo 64 RGB format into the traditional 8 bits per pixel:

```
b = (val & 0x003E) << 2;
g = (val & 0x07C0) >> 3;
r = (val & 0xF800) >> 8;
```

5.3 Lump Format

Size (bytes)	Description
2	Unknown
2	Number of palette tables used (1 for non-animating textures)
2	Width shift (1 << w)
2	Height shift (1 << h)

...	Raw data
...	Palette data

5.4 Texture Lookup

The lump names of the textures are not actually referenced in Doom 64. Instead they are looked up as indexes based on the order the lumps between the T_START and T_END markers.

5.5 Displaying the Texture

Like sprites, the rows of the textures' pixels are swapped (usually in groups of 4 pixels) and will need some simple conversion to swap the pixels in order to be properly viewed on the PC.

6 – Hud Sprite Format

Hud sprites are basically simple graphic lumps used for 2D display like the HUD, fonts, menu icons, etc. The format is somewhat similar to sprites.

6.1 Lump Format

Size (bytes)	Description
2	0 for RGB4 format or -1 for RGB8 format
2	Padding/Unused
2	Width
2	Height
...	Raw data
...	Palette data

7 – Demo Format

7.1 Lump Format

Size (bytes)	Description
56	Array to force-set button bind configuration
...	Button inputs (4 bytes per tic)

8 – Animation Definitions

The animdef system has gone through several changes compared to the original Doom. To recap, animdefs are definitions that define the animation of a texture. In Doom, it simply cycles from one pic to another from within the texture list. Doom 64 does the same but with additional features/options.

8.1 Animdef Structure

Size (bytes)	Description
4	Cycle restart delay
12	Name (max is 8 but pads to 12)
4	Number of frames
4	Tic speed between each frame
4	Reverse cycle at last pic (boolean)
4	Cycle palettes (boolean)

8.2 Types of animations

Below illustrate various animation types in Doom 64:

Normal


1


2


3

Reverse


1

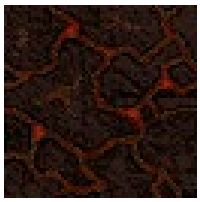

2

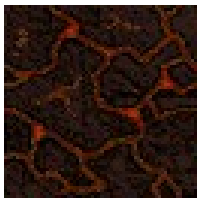

3

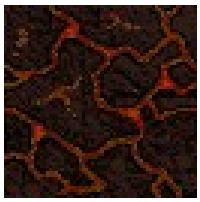

2

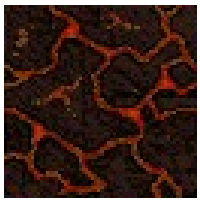

1

Palette









8.3 Palette Animations

Doom 64 can also use the texture's palette for animation effects, saving memory and the number of texture frames, allowing just one texture to do the animation. Instead of cycling through texture pics, Doom 64 will cycle through rows of 16 colors in the texture's palette to create the effect. Because it goes through 16 colors for each frame, the max number of frames for palette-animating textures is 16.

9 – Thing Definitions

9.1 Stateinfo Structure

The stateinfo structure in Doom 64 isn't really different than that of Doom 1 or 2. Mostly the unused fields were removed.

Size (bytes)	Description
4	Sprite number
4	Frame
4	Frame tics
4	Action codepointer
4	Next frame

9.2 Mobjinfo Structure

Size (bytes)	Description
4	Doom ed number
4	Spawn state
4	Spawn health
4	See state
4	See sound
4	Reaction time
4	Attack sound
4	Pain state
4	Pain chance
4	Pain sound
4	Melee state
4	Missile state
4	Death state
4	X-Death state

4	Speed
4	Radius
4	Height
4	Mass
4	Damage
4	Active sound
4	Flags
4	Palette LUT
4	Alpha

9.3 Codepointers

Offset	Name	Notes
0x80011340	A_Look	Bug in see sound logic. Possessed/Shotgun guy play only two of the three see sounds defined. Player 'bots' look for nearest shootable thing.
0x8001146C	A_Chase	
0x800116A8	A_FaceTarget	
0x80011740	A_Scream	
0x800117E4	A_XScream	
0x80011804	A_Pain	
0x8001185C	A_Fall	
0x80011870	A_Explode	
0x80011894	A_OnDeathTrigger	
0x80011954	A_PosAttack	Damage formula is changed
0x800119FC	A_SPosAttack	
0x80011B1C	A_PlayAttack	
0x80011BD4	A_CposFire	Not used. Calls P_CheckSight instead of MF_SEETARGET checks. Unoptimized for the N64.
0x80011C50	A_BspiFace	
0x80011C74	A_BspiAttack	Fires two plasma bolts
0x80011CBC	A_SpidRefire	
0x80011D78	A_TroopMelee	
0x80011DEC	A_TroopAttack	
0x80011E28	A_SargAttack	Damage formula is changed
0x80011E90	A_HeadAttack	Damage formula is changed
0x80011F08	A_CyberAttack	Location of where the projectile is spawned is different.
0x80011F44	A_CyberSpecial	

0x80011FC4	A_BruisAttack	Damage formula is changed
0x8001204C	A_SpawnSmoke	
0x80012088	A_Tracer	
0x800122F4	A_FatRaise	
0x80012320	A_FatAttack1	
0x800123B0	A_FatAttack2	
0x80012440	A_FatAttack3	
0x80012528	A_SkullAttack	
0x8001267C	A_PainShootSkull	Does path traverse checks for lost souls getting stuck behind walls
0x80012804	A_PainAttack	Spread angle is different. Fires two lost souls
0x8001285C	A_PainDie	Calls A_OnDeathTrigger
0x800128C4	A_RectChase	
0x8001296C	A_RectGroundFire	
0x80012B1C	A_RectMissile	
0x80012EA4	A_MoveGroundFire	
0x80012F34	A_RectTracer	
0x80012F6C	A_RectSpecial	
0x80012FEC	A_TargetCamera	
0x80013070	A_BarrelExplode	
0x800130E0	A_Hoof	
0x80013110	A_Metal	
0x80013140	A_BabyMetal	
0x8001333C	A_FadeAlpha	
0x80013364	A_PainSpecial	Alpha of target is set to 0x3F
0x80013378	A_SkullSetAlpha	Alpha of target is reduced by 75%
0x8001338C	A_MissileSetAlpha	Alpha of target is reduced by half
0x800133A0	A_FadeOut	
0x80013428	A_FadeIn	
0x8001B83C	A_WeaponReady	
0x8001B91C	A_ReFire	
0x8001B9A0	A_CheckReload	
0x8001B9C0	A_Lower	
0x8001BA84	A_Raise	
0x8001BAD8	A_GunFlash	Sets the alpha of the weapon's flash frame to 100 (excluding the BFG)
0x8001BB2C	A_Punch	
0x8001BC1C	A_Saw	
0x8001BDA8	A_ChainSawReady	
0x8001BDE4	A_FireMissile	Sets view pitch recoil. Thrusts user back

0x8001BE78	A_FireBFG	
0x8001BED8	A_AnimatePlasma	
0x8001BF2C	A_FirePlasma	
0x8001C0B4	A_FirePistol	
0x8001C138	A_FireShotgun	Sets view pitch recoil
0x8001C210	A_FireShotgun2	Sets view pitch recoil. Thrusts user back
0x8001C3F8	A_FireCGun	Sets view pitch recoil. Weapon sprite X and Y coordinates are randomized
0x8001C548	A_BFGFlash	Sets the alpha of the weapon's flash frame to 170
0x8001C560	A_BFGSpray	Calls A_FadeAlpha
0x8001C698	A_BFGSound	
0x8001C6C0	A_LoadShotgun2	
0x8001C6E8	A_CloseShotgun2	
0x8001CAC0	A_FireLaser	Performs nested loops and BSP traversing. Very expensive.

9.4 Sprite Array

Sprite Name
SPOT
PLAY
SARG
FATT
POSS
TROO
HEAD
BOSS
SKUL
BSPI
CYBR
PAIN
RECT
MISL
PLSS
BFS1
LASS
BAL1
BAL3
BAL2
BAL7
BAL8
APLS
MANF
TRCR

DART
FIRE
RBAL
PUF2
PUF3
PUFF
BLUD
A027
TFOG
BFE2
ARM1
ARM2
BON1
BON2
BKEY
RKEY
YKEY
YSKU
RSKU
BSKU
ART1
ART2
ART3
STIM
MEDI
SOUL
PINV
PSTR
PINS
SUIT
PMAP
PVIS
MEGA
CLIP
AMMO
RCKT
BROK
CELL
CELP
SHEL
SBOX
BPAK
BFUG
CSAW
MGUN
LAUN

PLSM
SHOT
SGN2
LSRG
CAND
BAR1
LMP1
LMP2
A031
A030
A032
A033
A034
BFLM
RFLM
YFLM
A006
A021
A003
A020
A014
A016
A008
A007
A015
A001
A012
A010
A018
A017
A026
A022
A028
A029
A035
A036
TRE3
TRE2
TRE1
A013
A019
A004
A005
A023
SAWG
PUNG

PISG
SHT1
SHT2
CHGG
ROCK
PLAS
BFGG
LASR

9.5 Mobj flags

Bit	Description
0x1	Pick up item
0x2	Solid
0x4	Can be hit
0x8	Don't use the sector links (invisible but touchable)
0x10	Don't use the blocklinks (inert but displayable)
0x20	Not to be activated by sound, deaf monster
0x40	Will try to attack right back
0x80	Will take at least one step before attacking
0x100	Hang from ceiling instead of stand on floor
0x200	Apply gravity (every tic)
0x400	This allows jumps from high places
0x800	For players, will pick up items
0x1000	Clip through walls (unused)
0x2000	Keep info about sliding walls (unused)
0x4000	For active floaters (unused)
0x8000	Object is teleporting; used for special ray-trace checks (unused in PC)
0x10000	Projectile
0x20000	Dropped item
0x40000	Trigger line special on touch/pickup
0x80000	Don't bleed when shot
0x100000	Don't stop moving halfway off a step
0x200000	Don't auto float to target's height
0x400000	On kill, count this enemy object towards intermission kill total
0x800000	On picking up, count this item object towards intermission item total
0x1000000	Special handling: skull in flight

0x2000000	Don't spawn this object in death match mode (e.g. key cards)
0x4000000	Target is visible to source
0x8000000	Count as secret when picked up (for intermissions)
0x10000000	Render as a flat laser-like projectile
0x20000000	Trigger line special on death
0x40000000	Invisibility powerup (players only)
0x80000000	Do not switch targets

9.6 Mobjinfo definitions

Name	Doom ED ##	Health	Reaction Time	Pain Chance	Speed	Radius	Height	Mass	Damage	Palette ID	Alpha
Player	-1	100	0	255	0	19	64	100	0	0	255
Red Bot	3008	100	0	255	16	32	87	100	0	1	255
Aqua Bot	3009	100	0	255	16	32	87	100	0	2	255
Green Bot	3010	100	0	255	0	32	87	100	0	0	255
Demon	3002	150	8	180	12	44	100	400	0	0	255
Spectre	58	150	8	180	12	50	100	400	0	1	255
Mancubus	67	600	8	80	8	60	108	1000	0	0	255
Possessed	3004	20	8	200	8	32	87	100	0	0	255
Shotgun Guy	9	30	8	170	8	32	87	100	0	1	255
Red Imp	3001	60	8	200	8	42	94	100	0	0	255
Blue Imp	3007	60	8	128	16	42	94	100	0	1	180
Cacodemon	3005	400	8	128	8	55	90	400	0	0	255
Baron	3003	1000	8	50	8	24	100	1000	0	1	255
Hell Knight	69	500	8	50	8	24	100	1000	0	0	255
Lost Soul	3006	60	8	256	8	28	64	50	3	0	192
Arachnotron	68	500	8	128	12	64	80	600	0	0	255
Cyberdemon	16	4000	8	20	16	70	170	1000	0	0	255
Cyberdemon (Title map)	3014	4000	8	0	0	40	110	1000	0	0	255
Pain Elemental	71	400	8	128	8	60	112	400	0	0	255
Resurrector	3013	5000	8	50	30	80	150	1000	0	0	255
Camera	0	1000	8	0	0	20	16	100	0	0	255
Teleport Dest	14	1000	8	0	0	20	16	100	0	0	255
Projectile Dest	2050	1000	8	0	0	20	16	100	0	0	255
Fake Item	89	1000	8	0	0	32	16	100	0	0	255
Laser Node	90	1000	8	0	0	20	16	100	0	0	255

Rocket Projectile	-1	1000	8	0	30	11	8	100	20	0	255
Plasma Projectile	-1	1000	8	0	40	13	8	100	5	0	255
BFG Projectile	-1	1000	8	0	40	13	8	100	100	0	255
Laser Projectile	-1	1000	8	0	0	8	8	100	0	0	255
Red Imp Projectile	-1	1000	8	0	10	6	8	100	3	0	255
Blue Imp Projectile	-1	1000	8	0	20	6	8	100	3	0	100
Cacodemon Projectile	-1	1000	8	0	20	6	8	100	5	0	255
Barron Projectile	-1	1000	8	0	15	6	8	100	8	0	255
Knight Projectile	-1	1000	8	0	15	6	8	100	8	0	255
Arachnotron Projectile	-1	1000	8	0	25	13	8	100	3	0	255
Mancubus Projectile	-1	1000	8	0	20	6	8	100	8	0	255
Tracer	-1	1000	8	0	10	11	8	100	10	0	255
Dart	-1	1000	8	0	16	13	8	100	4	0	255
Resurrector Fire	-1	1000	8	0	20	16	64	100	5	0	180
Resurrector Projectile	-1	1000	8	0	18	11	8	100	10	0	255
Gray Smoke	-1	1000	8	0	0	20	16	100	0	0	120
Red Smoke	-1	1000	8	0	0	20	16	100	0	0	120
Small Smoke	-1	1000	8	0	0	20	16	100	0	0	255
Blood	-1	1000	8	0	0	20	16	100	0	0	255
Crushed Gib	24	1000	8	0	0	20	16	100	0	0	255
Teleport Fog	-1	1000	8	0	0	20	16	100	0	0	140
BFG Spread	-1	1000	8	0	0	20	16	100	0	0	255
Item Green Armor	2018	1000	8	0	0	20	16	100	0	0	255
Item Blue Armor	2019	1000	8	0	0	20	16	100	0	0	255
Item Potion	2014	1000	8	0	0	20	16	100	0	0	255
Item Helmet	2015	1000	8	0	0	20	16	100	0	0	255
Item Blue Card	5	1000	8	0	0	20	16	100	0	0	255
Item Red Card	13	1000	8	0	0	20	16	100	0	0	255

Item Yellow Card	6	1000	8	0	0	20	16	100	0	0	255
Item Yellow Skull	39	1000	8	0	0	20	16	100	0	0	255
Item Red Skull	38	1000	8	0	0	20	16	100	0	0	255
Item Blue Skull	40	1000	8	0	0	20	16	100	0	0	255
Item Red Artifact	1042	1000	8	0	0	20	16	100	0	0	255
Item Aqua Artifact	1043	1000	8	0	0	20	16	100	0	0	255
Item Violet Artifact	1044	1000	8	0	0	20	16	100	0	0	255
Item Simpack	2011	1000	8	0	0	20	16	100	0	0	255
Item Medkit	2012	1000	8	0	0	20	16	100	0	0	255
Item Soulsphere	2013	1000	8	0	0	20	16	100	0	0	255
Item Invul Sphere	2022	1000	8	0	0	20	16	100	0	0	255
Item Beserk	2023	1000	8	0	0	20	16	100	0	0	255
Item Invis Sphere	2024	1000	8	0	0	20	16	100	0	0	255
Item Rad Sphere	2025	1000	8	0	0	20	16	100	0	0	255
Item Automap	2026	1000	8	0	0	20	16	100	0	0	255
Item Amp Goggles	2045	1000	8	0	0	20	16	100	0	0	255
Item Megasphere	83	1000	8	0	0	20	16	100	0	0	255
Item Clip	2007	1000	8	0	0	20	16	100	0	0	255
Item Ammo Box	2048	1000	8	0	0	20	16	100	0	0	255
Item Rocket	2010	1000	8	0	0	20	16	100	0	0	255
Item Rocket Box	2046	1000	8	0	0	20	16	100	0	0	255
Item Cell	2047	1000	8	0	0	20	16	100	0	0	255
Item Cell Pack	17	1000	8	0	0	20	16	100	0	0	255
Item Shell	2008	1000	8	0	0	20	16	100	0	0	255
Item Shell Box	2049	1000	8	0	0	20	16	100	0	0	255
Item Backpack	8	1000	8	0	0	20	16	100	0	0	255

Item BFG	2006	1000	8	0	0	20	16	100	0	0	255
Item Chainsaw	2005	1000	8	0	0	20	16	100	0	0	255
Item Chaingun	2002	1000	8	0	0	20	16	100	0	0	255
Item Rocket Launcher	2003	1000	8	0	0	20	16	100	0	0	255
Item Plasma Gun	2004	1000	8	0	0	20	16	100	0	0	255
Item Shotgun	2001	1000	8	0	0	20	16	100	0	0	255
Item Super Shotgun	82	1000	8	0	0	20	16	100	0	0	255
Item Laser Weapon	84	1000	8	0	0	20	16	100	0	0	255
Big Fire	2051	1000	8	0	0	16	64	100	0	0	140
Candle	34	1000	8	0	0	20	16	100	0	0	255
Barrel	1001	20	8	0	0	16	50	100	0	0	255
Explosion 1	-1	1000	8	0	0	20	16	100	0	0	80
Explosion 2	-1	1000	8	0	0	20	16	100	0	0	80
Tech Lamp 1	1015	1000	8	0	0	20	54	100	0	0	255
Tech Lamp 2	1016	1000	8	0	0	20	12	100	0	0	255
Blue Torch	1003	1000	8	0	0	20	16	100	0	0	255
Yellow Torch	1039	1000	8	0	0	20	16	100	0	0	255
Red Torch	1025	1000	8	0	0	20	16	100	0	0	255
Pole Base Long	1050	1000	8	0	0	12	16	100	0	0	255
Pole Base Short	1051	1000	8	0	0	8	16	100	0	0	255
Blue Fire	1033	1000	8	0	0	20	16	100	0	0	192
Red Fire	1034	1000	8	0	0	20	16	100	0	0	192
Yellow Fire	1035	1000	8	0	0	20	16	100	0	0	192
Meat Stick	1005	1000	8	0	0	20	16	100	0	0	255
Hanging Meat	1006	1000	8	0	0	20	95	100	0	0	255
Hanging Torso	1007	1000	8	0	0	20	83	100	0	0	255
Ribs	1008	1000	8	0	0	20	16	100	0	0	255
Twitching Gibs	1009	1000	8	0	0	20	16	100	0	0	255
Pool of Blood	1010	1000	8	0	0	20	16	100	0	0	255
Bloody Bones	1011	1000	8	0	0	20	16	100	0	0	255

Meaty Ribs	1012	1000	8	0	0	20	16	100	0	0	255
Meat Rib Cage	1013	1000	8	0	0	20	16	100	0	0	255
Hook and Chains	1014	1000	8	0	0	20	95	100	0	0	255
Hanging Cage	1017	1000	8	0	0	20	91	100	0	0	255
Hanging Pinser	1018	1000	8	0	0	20	101	100	0	0	255
Hanging Arm	1019	1000	8	0	0	20	58	100	0	0	255
Hanging Mace	1020	1000	8	0	0	20	80	100	0	0	255
Head on Stick 1	1022	1000	8	0	0	8	16	100	0	0	255
Head on Stick 2	1023	1000	8	0	0	8	16	100	0	0	255
Meat on Double Stick	1024	1000	8	0	0	8	16	100	0	0	255
Statue 1	1028	1000	8	0	0	20	16	100	0	0	255
Statue 2	1029	1000	8	0	0	20	16	100	0	0	255
Tech Pole Long	1031	1000	8	0	0	8	80	100	0	0	255
Tech Pole Short	1032	1000	8	0	0	8	62	100	0	0	255
Small Tree Stump	1036	1000	8	0	0	16	16	100	0	0	255
Large Tree Stump	1037	1000	8	0	0	16	16	100	0	0	255
Tree	1038	1000	8	0	0	16	16	100	0	0	255
Bloody Pole	1045	1000	8	0	0	8	16	100	0	0	255
Bloody Mace	1046	1000	8	0	0	20	56	100	0	0	255
Hanging White Meat	1047	1000	8	0	0	20	64	100	0	0	255
Hanging Head	1048	1000	8	0	0	20	60	100	0	0	255
Hanging Ribs	1049	1000	8	0	0	20	98	100	0	0	255

10 – Specials

10.1 Linedef specials

Type ID	Description	Tag

1	Vertical Door	Manual
2	Open Door	Line Tag
3	Close Door	Line Tag
4	Raise Door	Line Tag
5	Raise Floor	Line Tag
6	Ceiling Crush & Raise	Line Tag
8	Build Stairs	Line Tag
10	Platform Down Wait Up	Line Tag
16	Door Close Wait 30 Seconds Open	Line Tag
17	Spawn Light Strobe	Line Tag
19	Lower Floor	Line Tag
22	Plat Raise and Change	Line Tag
25	Ceiling Crush and Raise	Line Tag
30	Raise Floor To Nearest	Line Tag
31	Vertical Door Open Once	Manual
36	Lower Floor Fast	Line Tag
37	Lower Floor and Change	Line Tag
38	Lower Floor to Lowest	Line Tag
39	Teleport to Dest	Thing TID
43	Ceiling Lower to Floor	Line Tag
44	Ceiling Crush and Raise	Line Tag
52	Exit Level	Manual
53	Perpetual Platform Raise	Line Tag
54	Platform Stop	Line Tag
56	Raise Floor Crush	Line Tag
57	Ceiling Crush Stop	Line Tag
58	Raise Floor 24 Units	Line Tag
59	Raise Floor 24 Units and Change	Line Tag
66	Platform Down Up Raise 24 Units	Line Tag
67	Platform Down Up Raise 32 Units	Line Tag
90	Artifact Switch 1	Matching Line Tag + 1
91	Artifact Switch 2	Matching Line Tag + 1
92	Artifact Switch 3	Matching Line Tag + 1
93	Modify Thing Flags	Thing TID
94	Alert Thing	Thing TID

100	Build Stairs Fast 16 Units	Line Tag
108	Door Raise Fast Once	Line Tag
109	Door Open/Close Fast	Line Tag
110	Door Close Fast	Line Tag
117	Vertical Door Fast	Manual
118	Vertical Door Open Fast	Manual
119	Raise Floor to Nearest	Line Tag
121	Platform Down Wait Up Stay Fast	Line Tag
122	Platform Up Wait Down Stay	Line Tag
123	Platform Up Wait Down Stay Fast	Line Tag
124	Secret Exit	Tag = Level/Map ##
125	Monster Teleport to Dest	Thing TID
141	Silent Crusher	Line Tag
200	Clear Camera View	Manual
201	Set Camera	Thing TID
202	Invoke Dart	Thing TID
203	Delay Timer	Tag = Delay Amount
204	Set Macro Integer	Tag = Set Integer
205	Modify Sector Color 1	Line Tag, Integer = Light ID
206	Modify Sector Color 2	Line Tag, Integer = Light ID
207	Modify Sector Color 3	Line Tag, Integer = Light ID
208	Modify Sector Color 4	Line Tag, Integer = Light ID
209	Modify Sector Color 5	Line Tag, Integer = Light ID
210	Custom Ceiling	Line Tag, Integer = Move Amount
212	Custom Floor	Line Tag, Integer = Move Amount
214	Move Elevator Sector	Line Tag, Integer = Move Amount
218	Change Line Flags	Tag = Dest Line Tag, Integer = Source Line Tag
219	Modify Line Texture	Tag = Dest Line Tag, Integer = Source Line Tag
220	Modify Sector Flags	Tag = Dest Sector Tag, Integer = Source Sector Tag
221	Modify Sector Specials	Tag = Dest Sector Tag, Integer = Source Sector Tag
222	Modify Sector Lights	Tag = Dest Sector Tag, Integer = Source Sector Tag
223	Modify Sector Flats	Tag = Dest Sector Tag, Integer = Source Sector Tag
224	Spawn Thing	Thing TID
225	Quake	Tag = Duration
226	Custom Ceiling Fast	Line Tag, Integer = Move Amount

227	Custom Ceiling Instant	Line Tag, Integer = Move Amount
228	Custom Floor Fast	Line Tag, Integer = Move Amount
229	Custom Floor Instant	Line Tag, Integer = Move Amount
230	Modify Line Special	Tag = Dest Line Tag, Integer = Source Line Tag
231	Invoke Revenant Missile	Thing TID
232	Fast Ceiling Crush & Raise	Line Tag
233	Freeze Player	Tag = Duration
234	Change Light by Light Tag	Tag = Dest Light ID, Integer = Source Light ID
235	Modify Light Data	Tag = Dest Sector Tag, Integer = Source Sector Tag
236	Custom Down/Up Platform	Line Tag, Integer = Move Amount
237	Custom Down/Up Platform Fast	Line Tag, Integer = Move Amount
238	Custom Up/Down Platform	Line Tag, Integer = Move Amount
239	Custom Up/Down Platform Fast	Line Tag, Integer = Move Amount
240	Trigger Random Lines	Tag = Triggered Lines With Matching Tag
241	Split Open Sector	Line Tag, Integer = Move Amount
242	Fade Out Thing	Thing TID
243	Move and Aim Camera	Tag = Thing TID, Integer = Next Camera Spot
244	Set Floor Height	Line Tag, Integer = Height
245	Set Ceiling Height	Line Tag, Integer = Height
246	Restart Macro at Script Line	Tag = Repeat count, Integer = Script Line ##
247	Move Floor by Height	Line Tag, Integer = Z Height
248	Suspend Macro Script	Tag = Macro Line Type (256 - 511)
249	Telefrag to Dest	Thing TID
250	Toggle Macros On	Tag = Macro Line Type (256 - 511)
251	Toggle Macros Off	Tag = Macro Line Type (256 - 511)
252	Move Ceiling by Height	Line Tag, Integer = Z Height
253	Unlock Cheat Menu	Manual
254	Morph Logo on F_SKYG Sky	Manual

10.2 Sector specials

Type	Effect
1	Random flickering light
2	Blink light 0.5 seconds
3	Blink light 1.0 seconds

8	Normal pulsing light
9	Slow pulsing light
11	Random pulsing light
12	Same as type 3
13	Same as type 2
17	Fire light flickering
202	Constant blink light slow
204	Blink light rapid
205	Start of sequenced light sectors
206	Blink light very rapid
208	Constant blink light fast
666	Instant kill thing when being crushed

11 – Game Logic Changes

11.1 Doom Main Loop (d_main)

11.1.1 Mini Loops

Early console ports of Doom (Sony Playstation, Jaguar, N64, GBA) have redesigned the central loop system to be more simplistic and to reduce the executable size overall. The PC version of Doom used gamestates to define states that determine what the user is viewing: whether if its the intermission screen, the title screen, credits screen or actual game itself. Instead of using gamestates, these ports of Doom used nested mini loops which uses four arguments that are function calls. For example a typical setup would be like this:

```
next = D_MiniLoop(P_Start, P_Stop, P_Drawer, P_Ticker);

// run intermission if completed game
if(next == ga_completed)
    D_MiniLoop(WI_Start, WI_Stop, WI_Drawer, WI_Ticker);
```

The *_Start and *_Stop functions are used to initialize the next game state and cleanup the previous/current states. *_Ticker is the function that handles the tics of that specific game state. *_Drawer functions basically does the drawing. While the setup is similar to the PC version of Doom, the only major difference is that mini loops are used instead of gamestate variables.

11.2 Machine State (i_system)

Like the PC version, error handling is done through the machine state but all N64 backend stuff is handled as well. The low level rendering mechanics of the N64 is also done through the machine state.

11.3 Playloop State (p_*)

11.3.1. Collision

11.3.1.1 Wall Running

In the original Doom, wall running can occur when a player runs along a straight wall (either normally or by straferunning) while in contact with it. The player will be accelerated to a speed that is greater than normally attainable.

In Doom 64, there has been several modifications to the code for sliding along walls which prevents the player from ever achieving the wall running effect. The following code below is the change that was made in P_SlideMove that prevents wall running:

```
ld = bestslideline;

if(ld->slopetype == ST_HORIZONTAL)
    tmymove = 0;
else
    tmymove = FixedMul(mo->momy, bestslidefrac);

if(ld->slopetype == ST_VERTICAL)
    tmxmove = 0;
else
    tmxmove = FixedMul(mo->momx, bestslidefrac);

an1 = finecosine[ld->angle];
an2 = finesine[ld->angle];

if(P_PointOnLineSide(mo->x, mo->y, bestslideline))
{
    //
    // [d64] same as deltaangle += ANG180 ?
    //
    an1 = -an1;
    an2 = -an2;
}

newx = FixedMul(tmxmove, an1);
newy = FixedMul(tmymove, an2);

mo->momx = FixedMul(newx + newy, an1);
mo->momy = FixedMul(newx + newy, an2);

if(!P_TryMove(mo, mo->x + mo->momx, mo->y + mo->momy))
    goto retry;
```

11.3.1.2 Position Checks

P_CheckPosition was modified where MAXRADIUS is no longer used in the block iterator loops. Having this removed will result in thing collision checks only if two things are within the same block (in the blockmap). Because of this, its possible for two things to completely clip into each other if they are in both different blocks.

11.3.1.3 Radial Damage

PIT_RadiusAttack was updated to prevent Pain Elementals from receiving splash damage from Lost Souls if they die upon being projected out.

11.3.2 Line Triggering

PIT_CheckLine was updated with some checks to prevent the player or things from activating a line special that's in a closed sector (ceiling height = floor height). Usable line specials were also updated to prevent the player from activating a line special that's high above the player or below the floor.

11.3.3 Overflow Checks

Some of the collision related code for keeping track of special lines that was triggered and intercept checks has been patched to prevent overflow errors.

A spechits overflow occurs when any player or monster crosses and activates more than 8 linedef specials simultaneously. "Spechits" stands for "special hits" and refers to the number of linedef specials that have been "hit" by a player or monster after activating. Doom 64 fixes this by simply checking for how many lines that was hit and aborts adding any more linedef data to the spechits array.

For intercepts, if a bullet tracer crosses too many things and lines (more than 128), an internal overflow will happen, causing undefined behavior. This sometimes results in the bug known as "all-ghosts" in the original Doom. Doom 64 fixes this by aborting if 128 intercepts has been reached.

11.5 Zone Heap (z_zone)

Unlike the original version, the actual zone heap is passed as an argument in zone-related routines. As of now the reason behind this is unknown but one guess is that it was meant to work with more than one zone heap and thus needed to pass them as arguments to specify what heap to work with.

The zone heap pointer is located at offset 0x800B2240

In addition to supporting multiple heaps, two new routines were added: Z_Alloc and Z_Touch. Z_Alloc seems to be only used for allocating data for demo lumps while Z_Touch is called only in W_CacheLumpNum if the zone tag is PU_STATIC.

11.6 Misc

- VIEWHEIGHT is set from 41 to 56
- MAXMOVE is set from 30 to 16
- MELEERANGE is set from 64 to 80
- Friction value is now set at 0xd200 (player movement)
- Momentum is clamped differently in the P_XYMovement routine
- All references to the player is removed in P_XYMovement and P_ZMovement. Player movement is handled through separate routines.
- Mapthings are checked for valid spawn locations in P_SpawnMapThing. This is mainly to fix

level design issues of overlapping things placed in level
- Player projectiles are spawned at different Z-height axis based on type of projectile
- Player reaction time is set to 9 after teleporting
- Finesine and finecosine tables are generated upon game initialization
- Sound clipping distance is 1700
- Max sound distance formula is now (NORM_VOLUME * (S_CLIPPING_DIST >> FRACBITS))
- Leveltime variable is no longer used but instead uses gametic instead.
- Definitions for sprite rotations, flipping, etc is hard coded in the game instead of being generated on initialization

12 – Rendering System

The rendering features of Doom 64 is consisted of the following:

- True 3D rendering geometry/polygons
- Alpha blending
- Multi-texture blending
- Full 16-bit colored textures/sprites
- Vertex lighting/shading
- Limited to 5120 draw commands
- Limited to 3072 vertices for all geometry drawn
- Limited to 256 things/mobjs drawn on screen at once
- Limited to 256 subsectors rendered on screen at once

12.1 Rendering Pipeline

The main rendering of the game world begins in the routine P_Drawer (0x80021AB8). Below is a timeline of the rendering pipeline:

- P_Drawer (0x80021AB8)
  - Clear frame
    - Set scissor for screen clipping
    - Enable fog, shading and front face culling
  - Render automap
  - R_SetupFrame (0x80023448)
    - Setup frame by calculating player's view, angles, pitch, etc
    - R_RenderPlayerView (0x80023F30)
      - Traverse BSP nodes
        - Add any segs, subsectors, etc to occlusion buffer
        - Add any visible geometry to draw list
    - Render sky
    - Render fog
    - R_SetupRenderInfo (0x80026590)
      - Draw geometry
        - Render segs

        - Render subsectors
            - Render ceiling list if pic ID is not -1 and viewz is less than the ceiling height
            - Render floor list if viewz is above floor height
            - Render another floor plane if sector is flagged 0x4 (Water effect)
                - Draw transparent floor plane on top
        - Render things list
    - Render player sprites (weapons)
- Draw status hud
- Draw menu
- Finalize/draw frame

12.2 Texture Combiner Effects

12.2.1. Glowing Textures

Glowing textures replaces the original Doom's glowing sector effects like flickering or pulsating lights. Instead of the special effects changing the light level per sector, it applies an additive blending effect to the textures used in that sector.

Figure 12.2.1a: Standing in a sector with the flickering light effect. This effect additively blends a white constant color to the textures and sprites within that sector. The strength of the blending is based on the sector's light level.

12.2.2. Flashes

Just like the original Doom, whenever the player receives damage, or picks up an item, the screen will flash a specific color. The original Doom used the game palette to apply this effect though in Doom 64, it uses the texture combiner. Much like glowing textures, it additively blends a constant color to all textures and sprites. Flashes can also be blended together with glowing textures. Also only one flash can occur at a time and because of that, certain flashes has priorities over others. Below lists from highest to lowest priority:

Invulnerability < BFG Flash < Damage/Berserk < Radiation Suit < Item Pickup

Figure 12.2.2a: Examples of additive blending from flashes

12.3 Skies

The skies in Doom 64 is much more advanced than that of the original Doom. The skies have also been programmed to be customizable and allow for various visual effects.

In P_LoadSectors (0x8001D43C), it looks up certain flat names that begins with F_SKY*. In Doom 64, there are 11 sky flats named F_SKYA through F_SKYK and they determine what type of sky to render/display. Table below shows what sky flat determines the type of sky to render:

Flat Name	**Sky**
F_SKYA	Violet-colored clouds with thunder
F_SKYB	Red-colored clouds
F_SKYC	Orange-colored clouds with mountain backdrop
F_SKYD	Dynamic fire sky
F_SKYE	Orange-colored clouds with heavy fog
F_SKYF	Space backdrop

F_SKYG	Space backdrop with Doom 64 logo
F_SKYH	Blank/void sky
F_SKYI	Dynamic green fire sky
F_SKYJ	Violet-colored clouds with thunder and mountain backdrop
F_SKYK	Space and mountain backdrop

Each sky also has a draw and tick routine which varies based on what sky type is selected. Variables that determine color is mostly used for the clouds since they vary in style and color.

P_SetupSky (0x80025060) – Main routine for initializing the sky
fognear (0x800A8120) – Variable for fog near value
fogcolor (0x800A8124) – Variable for fog color
skyfunction (0x800A8130) – Pointer to sky draw/tick routine
cloudlump (0x800A8148) – Lump for cloud texture
skylump (0x800A8164) – Lump for 2D sky pic
backdroplump (0x800A8168) – Lump for 2D backdrop pic
cloudcolor (0x800A816C) – Variable for cloud color
flatcolor (0x800A8170) – Variable for blank/void sky color
skyflags (0x800A8174) – Variable for flags (mostly for clouds and debugging)

12.3.1. Clouds

Unlike normal skies, clouds gives off a 3D appearance and may even resemble a skybox. Clouds are nothing more than a simple plane in its own 3D matrix. The plane is rendered at a slanted angle while the top two vertices are horizontally spread out farther than the lowered ones. Below are diagrams to illustrate how the cloud plane is rendered:

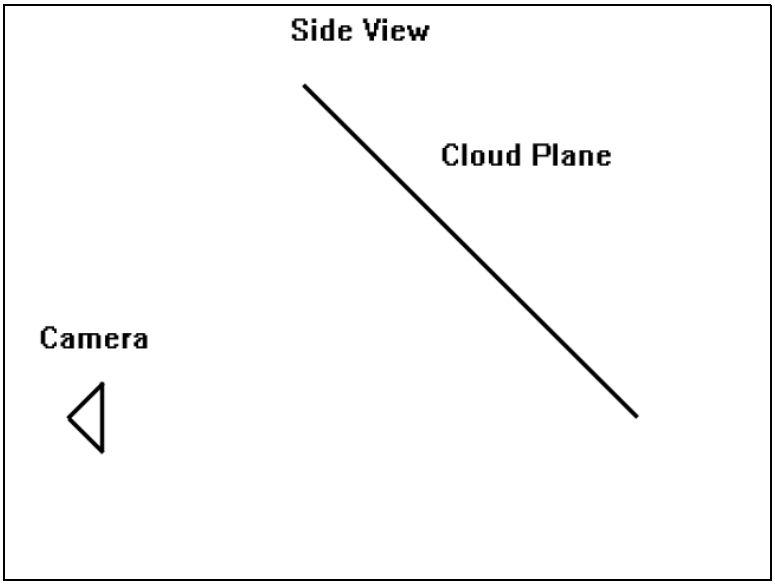


Figure 12.3.1a: Side view diagram of how cloud planes are rendered

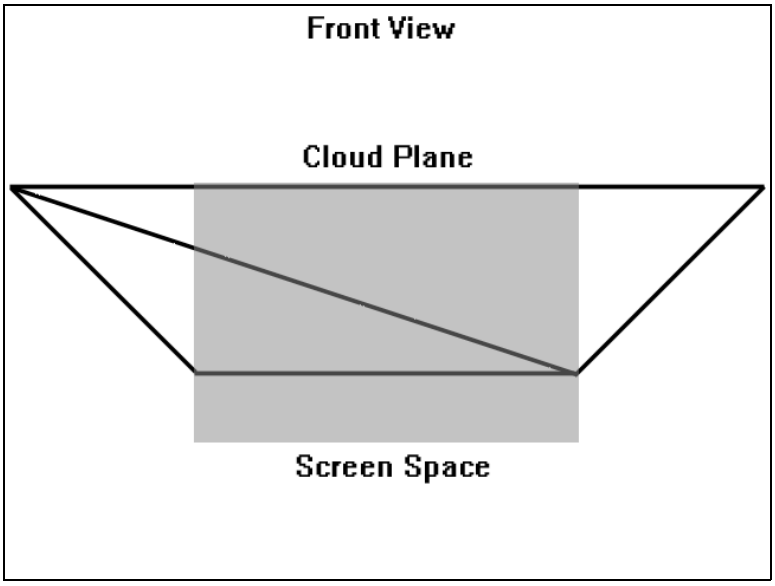


Figure 12.3.1b: Front view diagram of how cloud planes are rendered

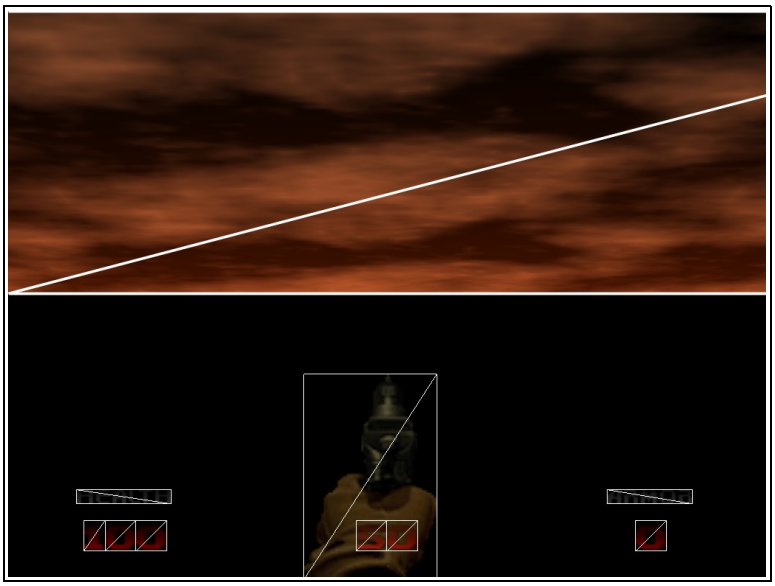


Figure 12.3.1c: In-game view of the cloud plane

The process to display the texture involves more than just the texture itself. It goes through four passes to create the final texture:

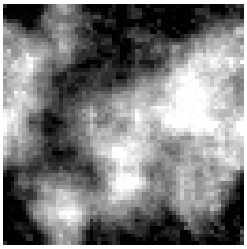


Figure 12.3.1d: Pass 1 – Initial Texture

Figure 12.3.1e: Pass 2 – Modulate by given cloud color value
This pass does the overall colorization of the cloud

Figure 12.3.1f: Pass 3 – Modulate by const RGB value '144, 144, 144'
This pass is to darken the texture

Figure 12.3.1g: Pass 4 – Add by fragment color
This creates the horizon look

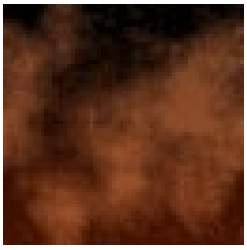


Figure 12.3.1h: Final Result

12.4 Fake 3D Geometry

Any floor or ceiling using the sky texture will not be rendered in the game. Because of this, the designers at Midway took advantage of this effect and used it to create fake 3D structures. These can be used to create things like bridges or low support beams, etc.

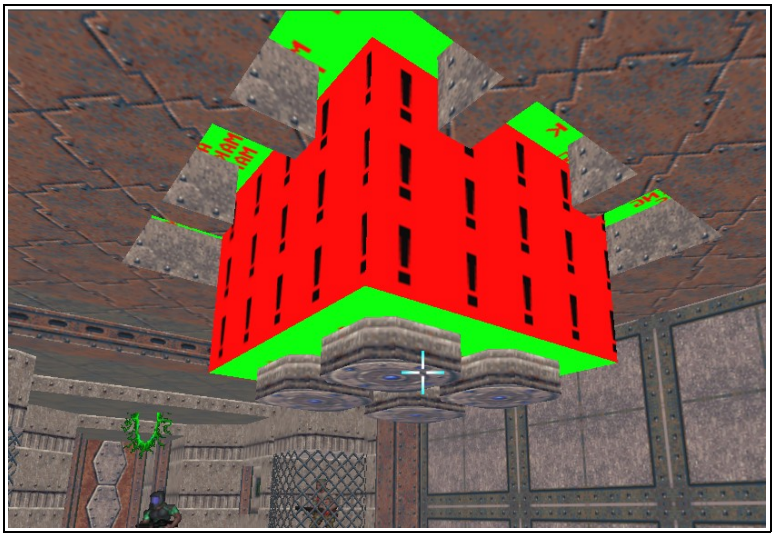


Figure 12.4a: This represents how this would actually look like in the editor. Note the surrounding sector and the four circular pods.

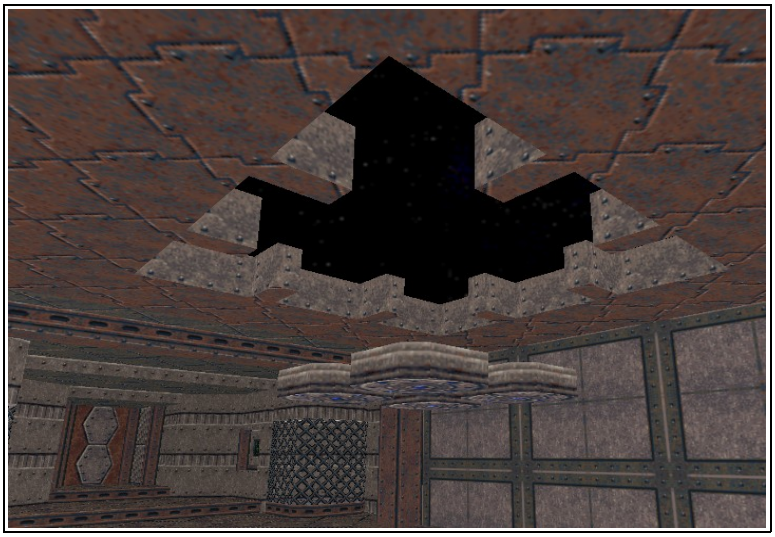


Figure 12.4b: In-game view of the effect. The lowered sky ceiling is not rendered, only leaving the four circular pods visible and giving off a 3D appearance.

12.5 Lighting

Lighting is the biggest stand-out feature in Doom 64. The lighting system is what gives off horrific atmosphere and ambiance and can be used in all sorts of ways. The lighting data is edited through the level editing tool and is implemented by the level designer. The RGB data is then stored in a table in the level's LIGHTS lump. The table has 255 pre-set RGB values, all gray scaled. These values range from 0, 0, 0 to 255, 255, 255. Following that is the actual RGB data set by the level designer.

In the rendering engine, the colored lighting is computed in this following timeline:

- Compile table of light RGB values
- Set the overall light factor (game brightness setting + player's infrared powerup)

    - Compute the H, S, V values
    - Clamp the luminosity
    - Compute the R, G, B values
- Update light table from light factor

Each sector can store up to 5 different RGB values which are used to determine the color for the ceiling, floor, thing and walls. Walls can have two RGB values which affects the top and lower portions of the wall. Below illustrates an example of a sector with configured color values in the editor:

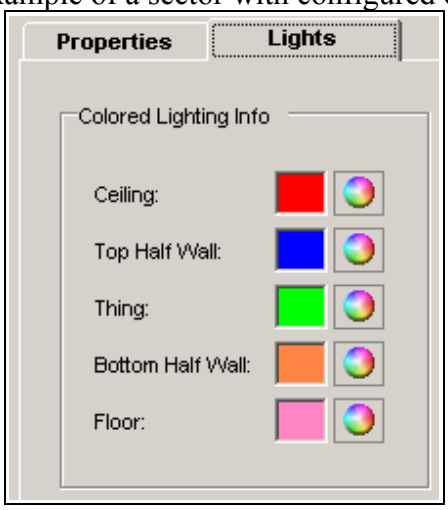


Figure 12.5a: Sector with configured lighting values. For example purposes, very distinct colors are chosen for each property.

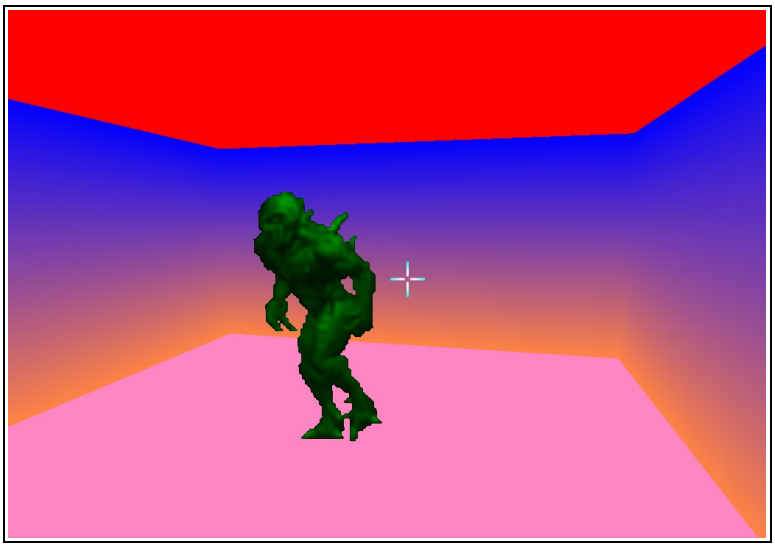


Figure 12.5b: Preview of how the room would be affected by the given color values for that sector. Texturing is disabled for demonstration purposes.

12.6 Advanced Lighting

Setting up the colored lighting is simple with one sided walls but can get complicated when working with two sided walls and sector heights.

12.6.1 Clamping

There is the option to clamp the top and bottom wall colors by using one of these linedef flags:

0x200000 – Clamp wall colors to top sidedef
0x400000 – Camp wall colors to lower sidedef

Below illustrates the differences between clamped and non-clamped:

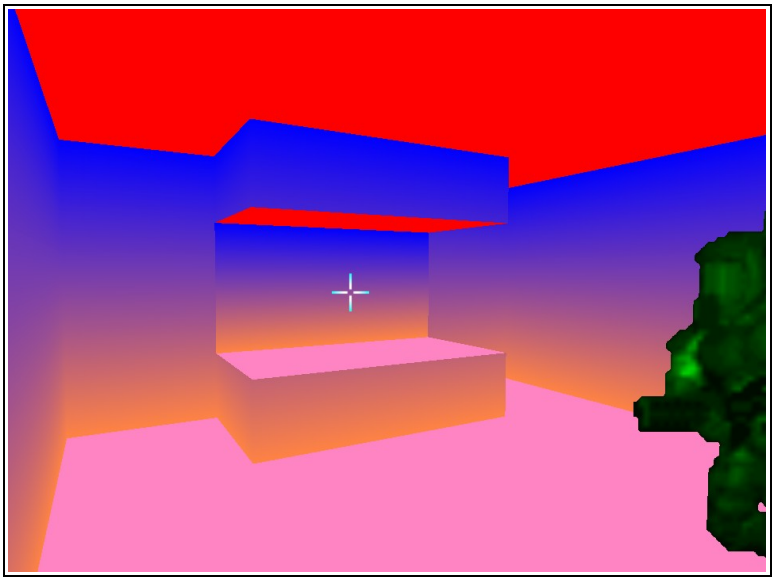


Figure 12.6.1a: Lowered ceiling and raised floor without color clamping.

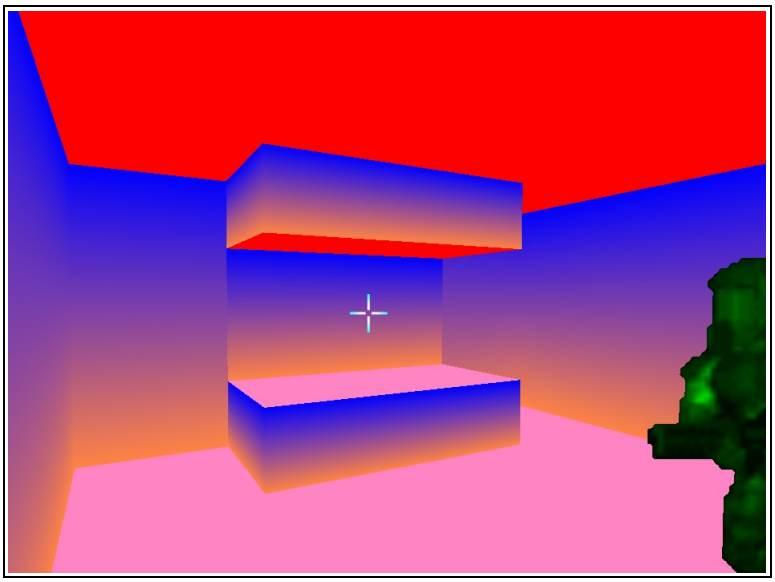


Figure 12.6.1b: Lowered ceiling and raised floor with color clamping for both top and lower sidedefs.

12.6.2 Flipping

There is another linedef flag (flag 0x4000000) that flips the top/bottom wall colors of the top sidedef.

Below illustrates the effects of this flag:

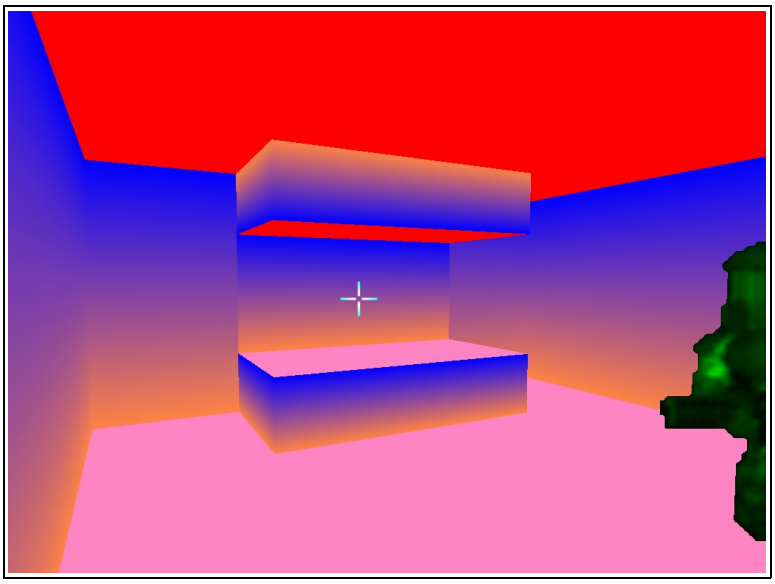


Figure 12.6.2a: Top/bottom wall colors are flipped for the upper sidedef.

12.6.3 Using Thing Color for Walls

If linedef flag 0x800000 is not set, then the walls will use the thing/mobj color values.

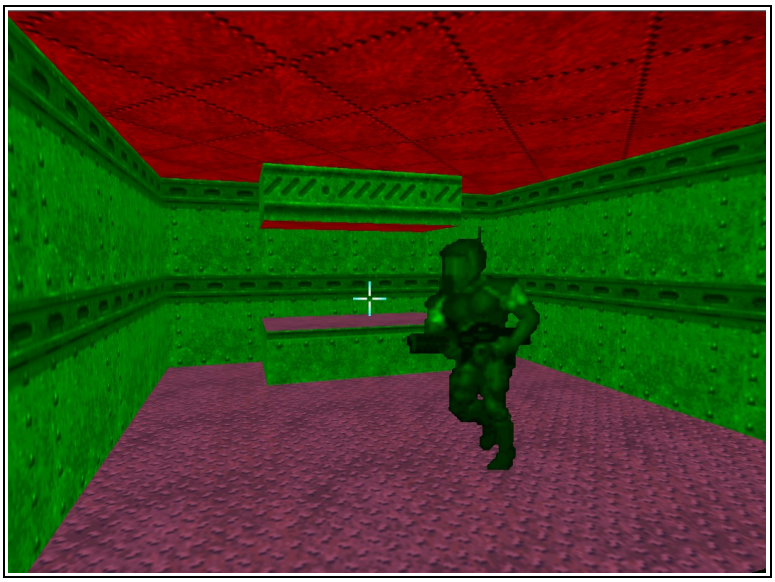


Figure 12.6.3a: Walls will use the thing color value if flag 0x800000 is not set.

12.7 Texture Mapping

Linedefs also uses some flags to manipulate how textures are mapped in Doom 64.

12.7.1 Mirroring

Because most textures are limited to size, there is the option to mirror the X/Y mapping to make them appear larger. The flags used to achieve this are:

0x40000000 – Horizontal texture mirror
0x80000000 – Vertical texture mirror

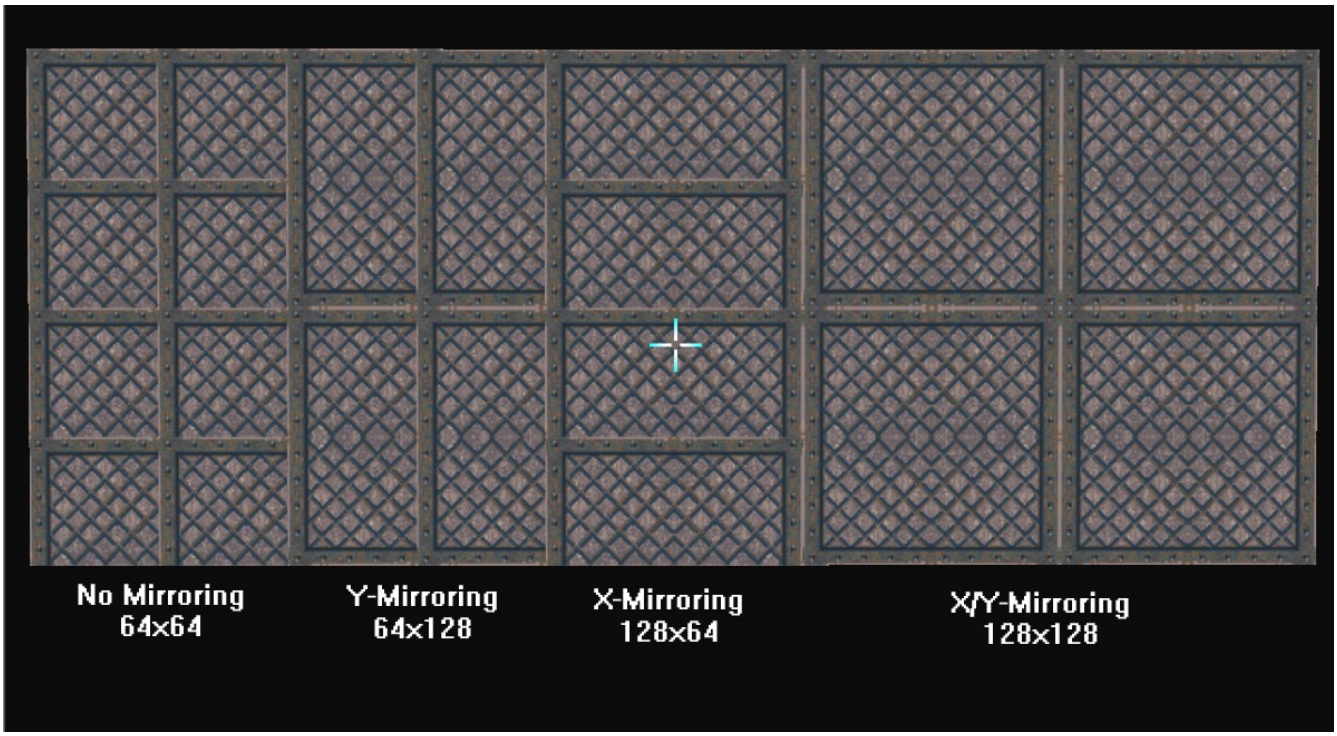


Figure 12.7.1a: Example of texture mirroring

12.7.2 Upper Unpegging

Behavior with upper and lower unpegging in Doom 64 is still the same than that of the original Doom however, Doom 64 has slightly changed how upper unpegging works for single sided walls. Flagging a solid-single sided wall to be upper unpegged will cause the row offset of the texture to snap to the nearest texture unit which is useful for aligning.

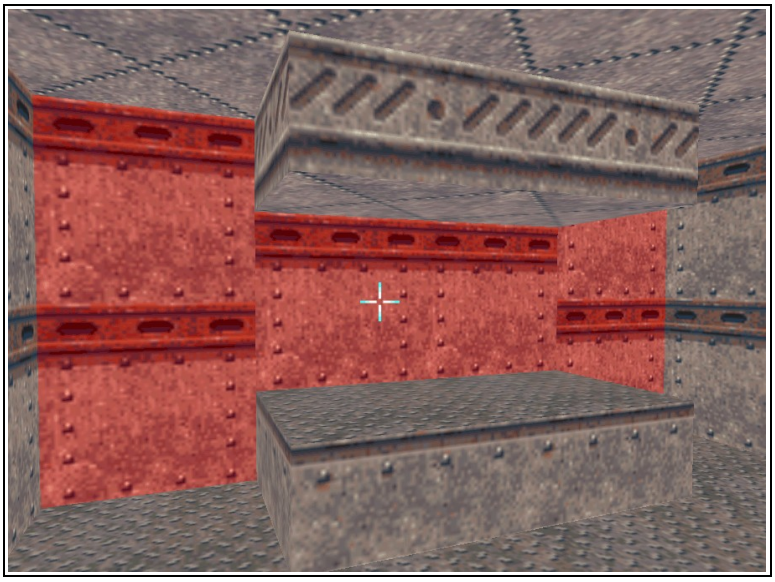


Figure 12.7.2a: Before flagging these highlighted walls to use the upper unpegged flag

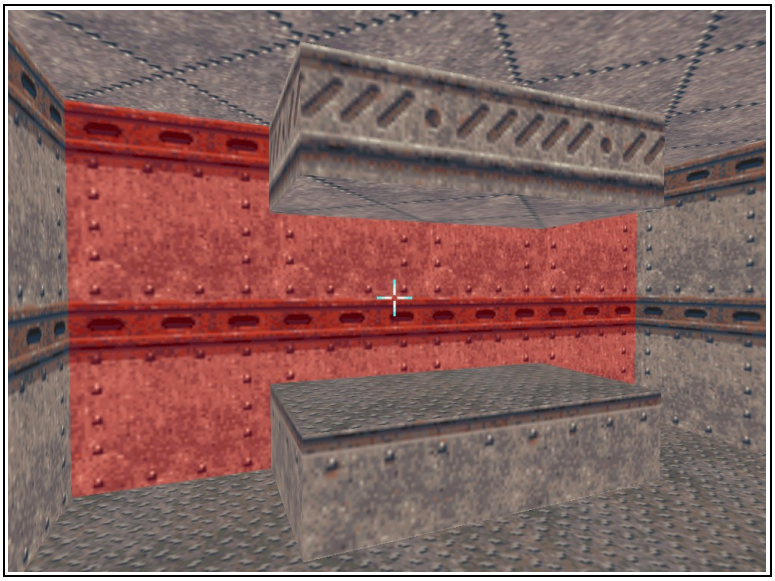


Figure 12.7.2b: Highlighted walls are flagged to be upper unpegged. Note the wall in the center is now aligned.

12.7.3 Rendering Textures for Middle Sidedefs

By default, Doom 64 will never render textures for middle sidedefs unless flagged '0x200'. The reason why this design choice was made is unknown but more likely related to technical limitations on the N64.

12.7.4 Middle Texture Unpegging

Doom 64 added a new flag (0x800) to unpeg middle sidedef textures. This flag is used only once in the entire game and can be seen in the secret exit in Map 12 (Altar of Pain). This flag also blocks projectiles but not tracer-based projectiles.

13 – Automap

As with the original Doom games, Doom 64 features the automap system with a few changes in behavior as well as new features.

13.1 Behavior Changes

- Zooming and panning speed has increased
- Blockmap width and height is used to determine the boundary of the automap to prevent the user from panning too far out of the map.
- Player arrow color pulsates
- Different colors for representing thing arrows, special linedefs and sector height variation
- Level name is displayed at x/y coordinates 20, 20 instead at the bottom of the screen
- Game keeps track of automap interaction with special automap flags that's stored in the player_t structure.

13.2 New Features

- Two modes: Textured mode and standard wireframe mode
  - Textured mode is simply rendering the floor subsectors from top down view. No sector effects are rendered with the exception of animating flats.
- Laser artifacts that the player picks up are displayed in the automap
- Automap shakes when the quake special is activated/triggered
- Instead of toggling 'follow' mode, holding down the use key toggles panning mode where the user can move around using the N64 stick.
- Entire map is rotated when user turns

13.3 Color Identification

Name	**Color RGB**
Solid Wall	164, 0, 0
Special Wall (linedef triggers, etc)	204, 204, 0
Secret Wall	164, 0, 0
Two-Sided Line	128, 80, 32
Two-Sided Line (sector height variation)	138, 92, 48
Cheating/Automap Powerup	128, 128, 128
Shootable Thing	164, 0, 0
Non-Shootable Thing	51, 115, 179

14 – Laser Projectile

The new weapon in Doom 64, the Laser Artifact, fires a straight beam (basically a series of quads) across the area. Despite of being a simple effect, there's a lot going on behind the scenes to achieve this.

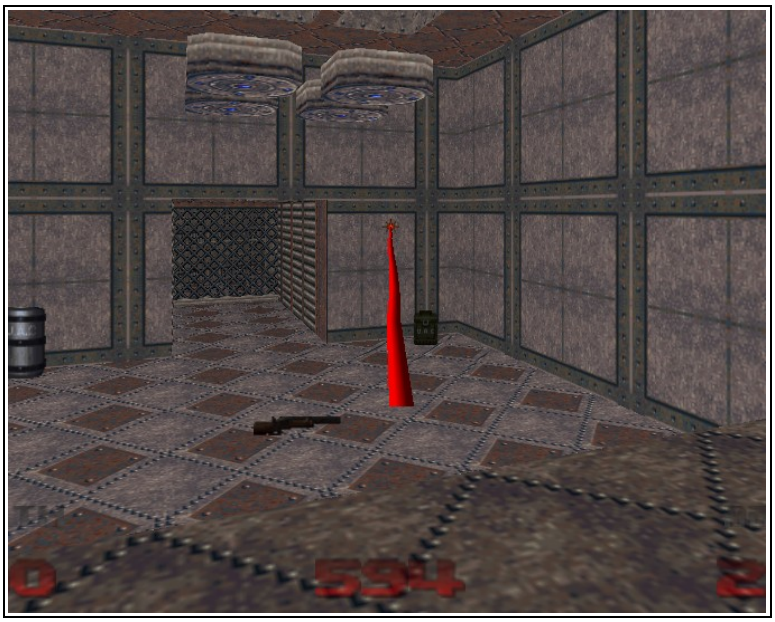


Figure 14a: Demonstration of a laser beam when fired by the Laser Artifact.

Almost everything is done through the A_FireLaser (0x8001CAC0) codepointer. There is even a special mobj, a mobj flag and a thinker that's used just to create this effect. The following timeline is as follows:

- A_FireLaser (0x8001CAC0)
  - Determine behavior by checking the artifacts that the player collected
    - 1 Artifact
      - Firing frame tic is changed to 5
      - Fires single laser beam
    - 2 Artifacts
      - Firing frame tic is changed to 4
      - Fires two laser beams
    - 3 Artifacts
      - Firing frame tic is changed to 4
      - Fires three laser beams
  - Decrement ammo per laser beam
  - P_AimLineAttack
    - Distance input is 4096
    - Z height offset is 40
    - Get the fraction of the intercepted line and assign to variable aimfrac (0x800A5720)
      - Default to 0x800 if aimfrac is 0
    - If target was found, damage it
  - Allocate laser
    - Set X/Y/Z coordinates for head and tail portion
      - Get player's firing location and then get location of the hit wall (using aimfrac)
  - Setup movement slope and distance
  - Traverse BSP (from player's firing location to aimfrac location)

- Spawn a 'laser node' in every subsector that was reached
- When done, connect all laser nodes to form a long beam. This means that the actual laser beam is consisted up of smaller segments which are chained together by the laser nodes.

Below is a visual example of how laser nodes are placed:

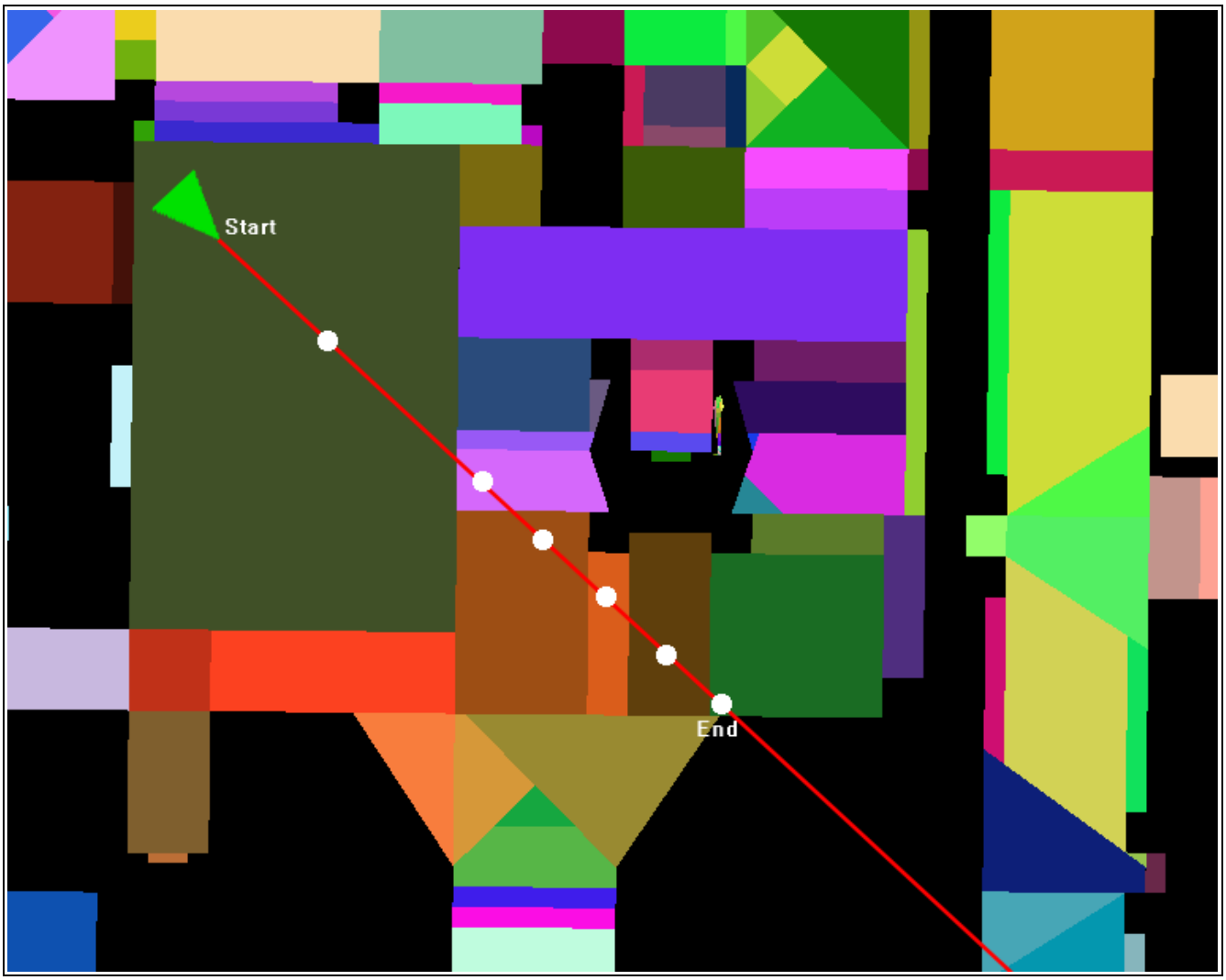


Figure 14b: Red line represents the beam and the white dots represents the laser nodes. The green arrow is the player and the colored regions are the subsectors from Map01 (Staging Area). When the end (aimfrac) is reached, all nodes are chain-linked together to form the laser beam.

Known Bugs:

- In more complicated subsectors, the beam will begin looking 'wobbly' due to the large number of segmented quads being rendered due to precision loss thanks to Doom's fracunit metrics.
- Because that the A_FireLaser codepointer calls P_AimTraverse instead of P_ShootTraverse, the end of the laser will always reach a one sided wall, meaning the laser beam will be going straight through raised sectors, stairs, sidedefs, etc. Doom64 Ex has fixed this issue though.

15 – Macro System

Another major feature in Doom 64 is a scripted system called Macros. This system works by executing a series of linedef specials in a batch and then execute the next batch when finished and so forth.. This rather simple system is very effective and can be used for all sorts of events.

15.1 Data Storing

Macros are stored in the level's WAD file though the methods used to import the macro data is unknown but may suspect that the macros where edited from through an external utility. For information on the macro lump format, see chapter 3.9.

15.2 Scripting Format

As stated before, it is unknown how macros are created since the actual tools to Doom 64 were never released by Midway. However in the Doom 64 TC interview, the designers did mention the macros were created using a scripting language similar to BASIC, but due to the nature of how macros are stored in Doom 64, this format is more likely to be closely similar to the Atari BASIC language (see http://en.wikipedia.org/wiki/Atari_BASIC). More likely macros were edited outside the level editor and through either a text file or command line. One possible guess to the scripting format would look something like this:

```
10    BATCH
12    DIM 256
14    EXEC 252, 35
16    DIM 192
18    EXEC 252, 2

20    BATCH
25    DELAY 45
30    BATCH
34    EXEC 231, 1
38    DELAY 20

40    BATCH
45    GOTO 30 : COUNTER = 14

50    BATCH
52    DIM 384
54    EXEC 252, 35
56    DIM 320
58    EXEC 252, 2
```

Because writing scripted languages can be daunting to most users, one suggestion for programmers is to implement macro editing directly into the level editor.

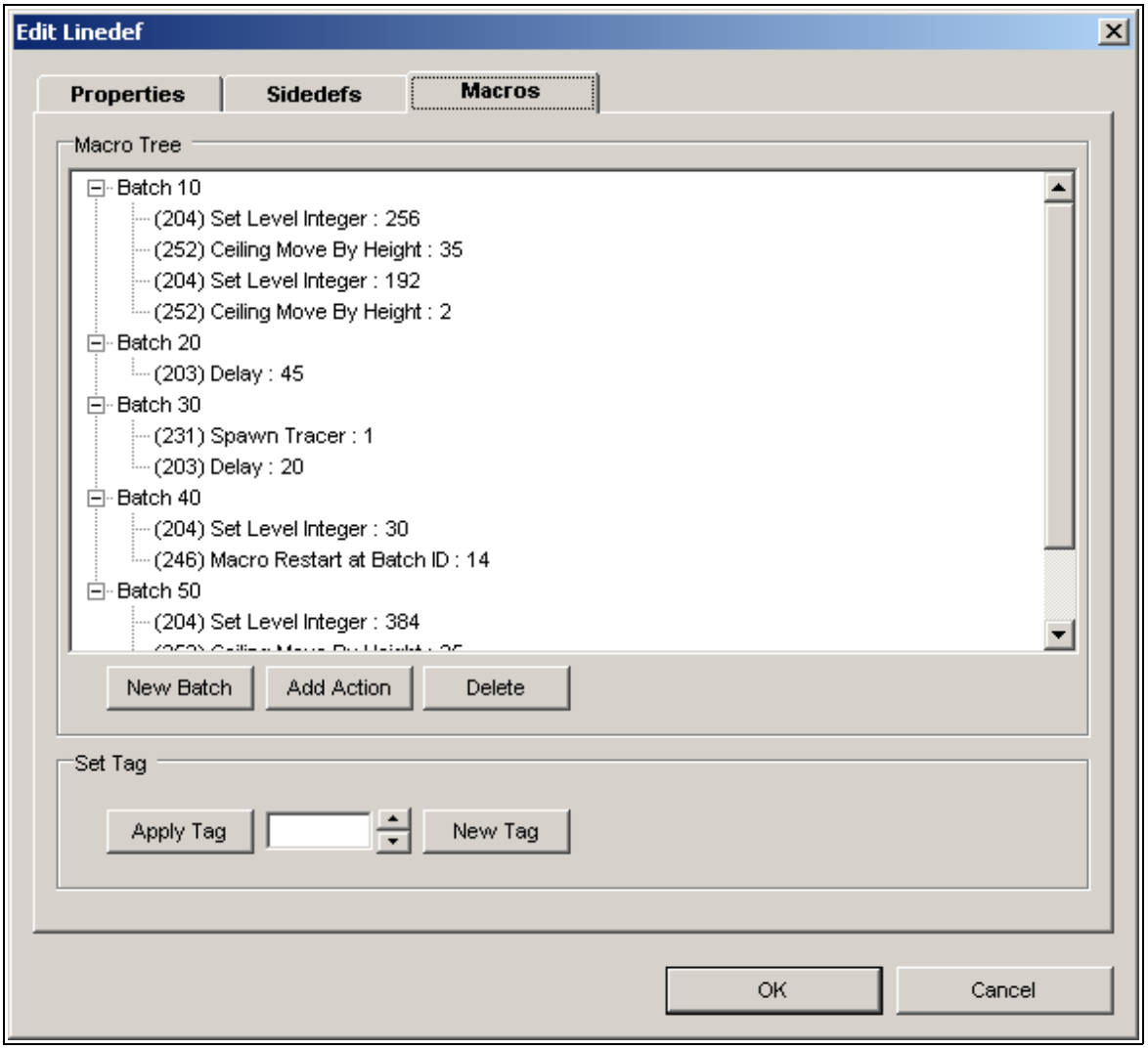


Figure 15.2a: The user-friendly macro editor which is integrated in Doom Builder 64.

15.3 Macro Processing

Macros heavily rely on thinkers (moving floors, ceilings, doors, etc) to determine when to run batches and to determine when all special actions are finished executing. The timeline below demonstrates how macros operate from within the game:

- Player triggers a macro special
- Check if macro is already running
  - If macro already running, abort current operation
  - If no macro is active, check any activated macros that's been queued
- Check for a tracked thinker initialized by the macro
  - If active, abort operation
- Read macro data
  - Execute all line specials down the batch list (ie. Run all actions with same batch ID)
  - If last line special in the list happens to trigger a thinker, then track it
    - If currently tracked thinker is NOT activated by the macro then set tracked thinker

pointer to null
    - If currently active thinker IS triggered by the macro then assign it to the tracked thinker pointer
- Wait until tracked thinker has finished
- Jump to next batch

There are several issues with this; first issue is that it will only track the thinker activated by the last action in the batch list. What can happen here is that the next batch could start while the other actions in the previous batch are still running because the thinker activated by the action at the bottom of the list finished early. If the last action happen to trigger no thinker then the macro will immediately jump to the next batch. The second issue is that macro scripts can only be triggered one at a time. Triggering other macros while a macro is active will add them to a queue and then will be automatically activated once the current macro has finished executing all actions.

15.4 Global Integer

The global integer variable (0x800A60F4) can be set by line type 204 through the tag. This variable is used as a second input for most line type specials such as the move height for custom moving ceiling/floors. Refer to chapter 10.1 for what line type actions the global integer can be used for.

16 – Sound System

Doom64 uses Midway's own sound library used to play sounds and music for the game. This library is also used in many other games developed by Midway. The library consists of Midway's own software synthesizer while leveraging the N64 midi player library. The library consists of two chunks: SN64 and SSEQ.

16.1 SN64

The SN64 chunk holds all instrument data and loop settings for sound data. It also contains the wav table that contains the offsets to the actual ADPCM data and predicator table.

16.2 SN64 Header

Size (bytes)	Description
4	Header marker (SN64)
4	Game ID (must be 2)
4	Padding
4	Version ID
4	Padding
4	Padding
4	File size (47520)
4	Padding
4	Unknown (31)

2	Number of presets
2	Total size of preset block
2	Number of instruments (505)
2	Size of instrument block (20)
2	Number of audio data (124)
2	Size of wav table block (24)
4	Padding
4	Size (excluding header) (33848)

16.3 Preset Block

Size (bytes)	Description
2	Length
2	Offset

16.4 Instrument Block

Size (bytes)	Description
1	Unity Pitch
1	Attenuation
1	Pan
1	Used for music (boolean)
1	Root Key
1	Unknown
1	Min Note
1	Max Note
1	Pitch Wheel Sensitivity
1	Unknown
2	SFX ID
2	Attack Time
1	Unknown
1	Unknown
2	Decay Time
1	Unknown
1	Unknown

16.5 Wav Table Block

Size (bytes)	Description
4	Offset Start
4	Size
4	Padding
4	Default Pitch
4	Loop ID
4	Padding

16.6 Loop Table Block

Size (bytes)	Description
4	Loop Start
4	Loop End
10	Padding / Garbage

16.7 Predicator Block

Size (bytes)	Description
4	Order
4	Count
128	Predicators

16.8 SSEQ

The SSEQ chunk contains all MIDI-related data. Following the SSEQ chunk is the encoded audio data.

16.9 SSEQ Header

Size (bytes)	Description
4	Header marker (SSEQ)
4	Game ID
4	Padding
4	Number of entries
8	Padding
4	Size of entry block
4	Padding

16.10 Entry Block

Size (bytes)	Description
4	Number of midi tracks
4	Length of entry block
4	Offset to track block
4	Padding

16.11 Track Block

Size (bytes)	Description
2	Flag
2	Initial Instrument ID
2	Padding
1	Volume
1	Pan
2	Padding
2	Beats Per Minute
2	Restart Position
2	Loop (boolean)
2	Padding
2	Length of midi track

16.12 Midi Events

Unlike standard MIDI, Midway has defined their own MIDI event types:

Event Name	Type ID	Arg 1	Arg 2	Notes
Unknown	0x02			Used in other games but not Doom64
Program Change	0x07	Instrument ID		
Pitch Bend	0x09	Bend Value LSB	Bend Value MSB	Values range from -8192 to 8192
Unknown	0x0B			Seems to be used alongside with pitch bend
Global Volume	0x0C	Volume (0 – 127)		
Global Panning	0x0D	Pan (0 – 127)		
Channel Aftertouch	0x0E	Value		

Unknown	0x0F			Used in other games but not Doom64
Play Note	0x11	Note	Velocity	Somewhat bugged. May cause other playing notes to stop or fail
Stop Note	0x12	Note		
Jump Position	0x20			Jumps to position marked with 0x23 or to position specified by track
Set Jump Position	0x23			Sets the position to jump to when event 0x20 is invoked. Ignored if position is set by the track's header.

17 – Memory Mapping

The following is a incomplete memory mapping dump of the Doom 64 ROM. Both routines and variables are listed. Some names may or may not be correct.

```
0002:00000000    start
0002:00000050    M_ClearBox
0002:00000070    M_AddToBox
0002:000000D8    AM_Stop
0002:000000F4    AM_Ticker
0002:000005AC    AM_Drawer
0002:00000EA0    AM_DrawLeafs
0002:000010C8    AM_DrawLine
0002:00001434    AM_DrawThings
0002:00001620    D_memset
0002:000016CC    D_memcpy
0002:00001CBC    R_GetLightHSV
0002:0000208C    R_GetLightRGB
0002:000023C0    D_DoomMain
0002:00002528    P_Random
0002:00002554    M_Random
0002:00002580    M_ClearRandom
0002:00002598    D_MiniLoop
0002:000027F8    D_abs
0002:000028D0    IN_Start
0002:00002A14    IN_Stop
0002:00002A44    IN_Ticker
0002:00002B14    IN_Drawer
0002:00002D3C    F_Start
0002:00002E20    F_Stop
0002:00002E58    F_Ticker
0002:000035DC    F_Drawer
0002:0000391C    R_DrawSpriteHud
0002:000040D0    FixedMul
0002:000040E4    FixedDiv
0002:00004130    G_DoLoadLevel
```

```
0002:00004198      G_PlayerFinishLevel
0002:00004230      G_PlayerReborn
0002:000042E4      G_CompleteLevel
0002:000042F4      G_InitNew
0002:00004394      G_RunGame
0002:000045D0      D_RunDemoMap
0002:000046F0      WI_Start
0002:000049B0      WI_Stop
0002:00004A24      WI_Ticker
0002:00004D64      WI_Drawer
0002:00005220      I_Start
0002:0000527C      I_StartGameThread
0002:00005310      D_Init
0002:00005330      I_SystemTicker
0002:00005850      N64_Init
0002:00005B30      I_Error
0002:00005CE8      I_CheckGFX
0002:00005F7C      I_ClearFrame
0002:00006170      I_DrawFrame
0002:000062C0      I_SetOSMessage
0002:00006324      LongSwap64
0002:00006350      LongSwap32
0002:00006370      ShortSwap
0002:00006390      I_GetInput
0002:00006564      WIPE_MeltScreen
0002:00006934      WIPE_FadeOutScreen
0002:00006C9C      alSynFreeFX
0002:00006CB0      I_CheckPeripherals
0002:00006E24      I_DeletePakFile
0002:00006F08      I_SavePakFile
0002:00006FB8      I_ReadPakFile
0002:000070D4      I_CreatePakFile
0002:00007230      D_Title
0002:000076B4      M_FadeStart
0002:000076EC      M_ChangeMenu
0002:00007848      M_Drawer
0002:00007A0C      M_MenuTicker
0002:00008718      M_DoCheat_Warp
0002:000087E8      M_DoCheat_Invul
0002:00008828      M_DoCheat_AllMap
0002:00008854      M_DoCheat_Weapons
0002:000088E4      M_DoCheat_HealthBoost
0002:00008A6C      M_FadeStop
0002:00008A7C      M_Start
0002:00008DC0      M_DrawFeatures
0002:00008EEC      M_FeatureDraw_Warp
0002:00008F1C      M_FeatureDraw_Invul
0002:00008F94      M_FeatureDraw_NoClip
0002:00009014      M_FeatureDraw_AllMap
0002:0000903C      M_FeatureDraw_Weapons
0002:00009064      M_FeatureDraw_Keys
0002:0000908C      M_FeatureDraw_Health
0002:000090DC      M_FeatureDraw_MusicTest
0002:00009484      M_Display
0002:00009668      R_SlamBackground
0002:00009B58      M_DrawOverlay
0002:00009CF8      M_ScreenTicker
0002:00009FE4      M_PakMenu
```

0002:0000A2E8 M_PakStart
0002:0000A3B4 M_PakStop
0002:0000A404 M_PakTicker
0002:0000A744 M_DrawPakMenu
0002:0000AAEC M_PassStart
0002:0000AB8C M_PassStop
0002:0000ABE4 M_PasswordMenu
0002:0000AE70 M_ControllerPak
0002:0000B204 M_CenterDisplay
0002:0000B294 M_ControlStick
0002:0000B588 M_ControlPad
0002:0000B810 M_EncodePassword
0002:0000BD94 M_DecodePassword
0002:0000C374 M_InitPassword
0002:0000C6F0 M_Password
0002:0000C9E0 P_RunMobjs
0002:0000CA74 P_MobjThinker
0002:0000CB98 P_MoveMobjXY
0002:0000CE28 P_MoveMobjZ
0002:0000CFF4 P_TryMoveMobj
0002:0000D15C P_UnsetThingPosition2
0002:0000D250 P_SetThingPosition2
0002:0000D350 P_CheckMobjPosition
0002:0000D520 P_CheckLineBBox
0002:0000D644 PIT_CheckLineAlt
0002:0000D770 P_CheckBlockedLines
0002:0000D870 PIT_CheckThingAlt
0002:0000D9D4 P_CheckBlockedThings
0002:0000DA60 T_LookAtCamera
0002:0000DB20 P_SetCamera
0002:0000DC2C P_SpawnTrapMissile
0002:0000DD60 P_SpawnDelayThinker
0002:0000DDCC P_DelayThinker
0002:0000DE20 G_ExitLevel
0002:0000DE5C G_ExitSecretLevel
0002:0000DE9C P_Telefrag
0002:0000DFA0 EV_Teleport
0002:0000E1C0 EV_SilentTeleport
0002:0000E2BC P_ModifyLineFlags
0002:0000E380 P_ModifyLine
0002:0000E42C P_ModifyLineTexture
0002:0000E528 P_ModifySector
0002:0000E6CC T_FadeThinker
0002:0000E78C P_SpawnMobjFade
0002:0000E908 P_RemoveThingFade
0002:0000E9E8 T_Quake
0002:0000EA7C P_Quake
0002:0000EAE0 P_RandomLineTrigger
0002:0000EC14 T_CameraThinker
0002:0000EEF8 P_CameraMover
0002:0000F010 R_RefreshBrightness
0002:0000F058 R_SetLightFactor
0002:0000F210 P_UpdateLightFactor
0002:0000F274 P_ModifyMobjFlags
0002:0000F2C4 P_AlertTaggedMobj
0002:0000F36C P_BossDeath
0002:0000F480 T_CeilingThinker
0002:0000F64C EV_DoCeiling

```
0002:0000F778    silentCrushAndRaise
0002:0000F77C    fastCrushAndRaise
0002:0000F7A4    crushAndRaise
0002:0000F7B4    lowerToFloor
0002:0000F7D8    raiseToHighest
0002:0000F7F0    customCeiling
0002:0000F834    customCeilingToHeight
0002:0000F8C0    P_AddActiveCeiling
0002:0000F918    P_RemoveActiveCeiling
0002:0000F988    P_ActivateInStasisCeiling
0002:0000FB74    EV_CeilingCrushStop
0002:0000FD80    P_ThingHeightClip
0002:0000FE34    PIT_ChangeSector
0002:0000FFBC    P_ChangeSector
0002:000100A0    T_VerticalDoor
0002:00010350    EV_DoDoor
0002:00010598    EV_VerticalDoor
0002:00010790    P_CheckMeleeRange
0002:00010810    P_CheckMissileRange
0002:00010908    P_Move
0002:00010A88    P_TryWalk
0002:00010AD0    P_NewChaseDir
0002:00010D5C    P_LookForPlayers
0002:00010F40    A_Look
0002:0001106C    A_Chase
0002:000112A8    A_FaceTarget
0002:00011340    A_Scream
0002:000113E4    A_XScream
0002:00011404    A_Pain
0002:0001145C    A_Fall
0002:00011470    A_Explode
0002:00011494    A_OnDeathTrigger
0002:00011554    A_PosAttack
0002:000115FC    A_SPosAttack
0002:0001171C    A_PlayAttack
0002:000117D4    A_CposFire
0002:00011850    A_BspiFace
0002:00011874    A_BspiAttack
0002:000118BC    A_SpidRefire
0002:00011978    A_TroopMelee
0002:000119EC    A_TroopAttack
0002:00011A28    A_SargAttack
0002:00011A90    A_HeadAttack
0002:00011B08    A_CyberAttack
0002:00011B44    A_CyberDeathEvent
0002:00011BC4    A_BruisAttack
0002:00011C4C    A_SpawnSmoke
0002:00011C88    A_Tracer
0002:00011EF4    A_FatRaise
0002:00011F20    A_FatAttack1
0002:00011FB0    A_FatAttack2
0002:00012040    A_FatAttack3
0002:00012128    A_SkullAttack
0002:00012254    PIT_PainCheckLine
0002:0001227C    A_PainShootSkull
0002:00012404    A_PainAttack
0002:0001245C    A_PainDie
0002:000124C4    A_RectChase
```

```
0002:0001256C    A_RectGroundFire
0002:0001271C    A_RectMissile
0002:00012AA4    A_MoveGroundFire
0002:00012B34    A_RectTracer
0002:00012B6C    A_RectDeathEvent
0002:00012BEC    A_TargetCamera
0002:00012C70    A_BarrelExplode
0002:00012CE0    A_Hoof
0002:00012D10    A_Metal
0002:00012D40    A_BabyMetal
0002:00012D70    L_ExplodeMissile
0002:00012EAC    L_SkullBash
0002:00012F3C    A_FadeAlpha
0002:00012F64    A_PainDeathEvent
0002:00012F78    A_SkullSetAlpha
0002:00012F8C    A_MissileSetAlpha
0002:00012FA0    A_FadeOut
0002:00013028    A_FadeIn
0002:000130BC    P_SpawnMissile
0002:000132C0    T_MovePlane
0002:00013520    T_MoveFloor
0002:000135FC    EV_DoFloor
0002:000139B0    EV_BuildStairs
0002:00013C98    T_CustomPlane
0002:00013E34    EV_CustomSector
0002:00013FE0    P_GiveAmmo
0002:000141C0    P_GiveWeapon
0002:00014280    P_GiveBody
0002:000142C8    P_GiveArmor
0002:00014304    P_GiveCard
0002:0001432C    P_GivePower
0002:00014410    P_TouchSpecialThing
0002:00014C80    P_KillMobj
0002:00014EDC    P_DamageMobj
0002:00015340    T_FireFlicker
0002:000153B4    P_SpawnFireFlicker
0002:00015420    T_Glow
0002:00015568    P_SpawnGlowingLight
0002:00015614    T_LightFlash
0002:000156B4    P_SpawnLightFlash
0002:00015728    T_StrobeFlash
0002:000157B4    P_SpawnStrobeFlash
0002:00015844    P_SpawnStrobeFlash2
0002:000158C4    EV_StartLightStrobing
0002:0001596C    P_ModifySectorColor
0002:00015A5C    T_Sequence
0002:00015C38    P_SpawnLightSequence
0002:00015D18    P_UpdateLightThinker
0002:00015E44    T_LightMorph
0002:00015F20    P_ChangeLightByTag
0002:00015FB8    P_DoSectorLightChange
0002:00016124    T_Combine
0002:00016178    P_CombineSectorSpecials
0002:000162E0    P_CheckThing
0002:00016324    P_TryMove
0002:00016458    P_CheckSwitchHeight
0002:00016628    PTR_UseTraverse
0002:00016810    P_UseLines
```

0002:00016A3C PIT_RadiusAttack
0002:00016B9C P_RadiusAttack
0002:00016CAC PTR_SlideTraverse
0002:00016DB0 P_SlideMove
0002:00017108 PTR_AimTraverse
0002:000173A8 PTR_ShootTraverse2
0002:00017830 P_AimLineAttack
0002:00017974 P_LineAttack
0002:00017B00 P_AproxDistance
0002:00017B40 P_LineOpening
0002:00017BE8 P_BlockLinesIterator
0002:00017D1C P_BlockThingsIterator
0002:00017DDC P_PathTraverse
0002:00018174 PIT_AddLineIntercepts
0002:0001820C PIT_AddThingIntercepts
0002:0001832C P_InterceptVector
0002:000184F0 P_TraverseIntercepts
0002:00018620 P_SpawnMobj
0002:00018824 P_SpawnMapThing
0002:00018B94 P_SpawnPlayer
0002:00018D30 P_RemoveMobj
0002:00018D84 P_SetMobjState
0002:00018E18 P_SpawnPuff
0002:00018EB8 P_SpawnBlood
0002:00019010 P_SpawnMissile2
0002:00019268 P_SpawnPlayerMissile
0002:000194B8 P_ExplodeMissile
0002:00019580 P_TryMove2
0002:00019824 P_PointOnLineSide
0002:00019910 P_UnsetThingPosition
0002:00019A20 P_SetThingPosition
0002:00019B50 P_CheckPosition
0002:00019E80 P_BoxOnLineSide
0002:00019FDC PIT_CheckLine3
0002:0001A160 PIT_CheckThing2
0002:0001A310 PIT_CheckLine
0002:0001A410 PIT_CheckThing
0002:0001A490 T_PlatRaise
0002:0001A694 EV_DoPlat
0002:0001AA6C P_ActivateInStasis
0002:0001AB2C EV_StopPlat
0002:0001ABF8 P_AddActivePlat
0002:0001AC50 P_RemoveActivePlat
0002:0001ACD0 P_SetupPsprites
0002:0001AD4C P_MovePsprites
0002:0001AE54 P_RecursiveSound
0002:0001AFA4 P_NoiseAlert
0002:0001AFFC P_SetPsprite
0002:0001B0BC P_BringUpWeapon
0002:0001B180 P_DropWeapon
0002:0001B1BC P_CheckAmmo
0002:0001B3CC P_FireWeapon
0002:0001B43C A_WeaponReady
0002:0001B51C A_ReFire
0002:0001B5A0 A_CheckReload
0002:0001B5C0 A_Lower
0002:0001B684 A_Raise
0002:0001B6D8 A_GunFlash

```
0002:0001B72C      A_Punch
0002:0001B81C      A_Saw
0002:0001B9A8      A_ChainSawReady
0002:0001B9E4      A_FireMissile
0002:0001BA78      A_FireBFG
0002:0001BAD8      A_PlasmaAnimate
0002:0001BB2C      A_FirePlasma
0002:0001BB88      P_BulletSlope
0002:0001BC24      P_GunShot
0002:0001BCB4      A_FirePistol
0002:0001BD38      A_FireShotgun
0002:0001BE10      A_FireShotgun2
0002:0001BFF8      A_FireCGun
0002:0001C148      A_BFGFlash
0002:0001C160      A_BFGSpray
0002:0001C298      A_BFGsound
0002:0001C2C0      A_LoadShotgun2
0002:0001C2E8      A_CloseShotgun2
0002:0001C310      P_LaserCrossBSP
0002:0001C5B8      T_LaserThinker
0002:0001C6C0      A_FireLaser
0002:0001CB20      P_LoadVertexes
0002:0001CC20      P_LoadSegs
0002:0001CF4C      P_LoadSubSectors
0002:0001D03C      P_LoadSectors
0002:0001D24C      P_LoadNodes
0002:0001D464      P_LoadThings
0002:0001D5B8      P_LoadLineDefs
0002:0001D8C8      P_LoadSidedefs
0002:0001DA38      P_LoadBlockMap
0002:0001DB98      P_LoadReject
0002:0001DBF8      P_LoadLeafs
0002:0001DE9C      P_LoadLights
0002:0001E078      P_LoadMacros
0002:0001E214      P_GroupLines
0002:0001E574      P_SetupLevel
0002:0001E700      P_ScanSights
0002:0001E7CC      P_CheckSight
0002:0001ED5C      PS_CrossSubsector
0002:0001EF40      P_LoadTextures
0002:0001F090      P_SpawnSpecials
0002:0001F56C      getNextSector
0002:0001F5AC      P_FindLowestFloorSurrounding
0002:0001F648      P_FindHighestFloorSurrounding
0002:0001F6E4      P_FindNextHighestFloor
0002:0001F868      P_FindLowestCeilingSurrounding
0002:0001F908      P_FindHighestCeilingSurrounding
0002:0001F9A4      P_FindSectorFromLineTag
0002:0001FA08      P_FindLightFromLightTag
0002:0001FA64      P_ActivateLineByTag
0002:0001FAC0      P_UpdateSpecials
0002:0001FE6C      getSide
0002:0001FEBC      getSector
0002:0001FF14      twoSided
0002:0001FF54      P_AddSectorSpecial
0002:000200BC      P_DoSpecialLine
0002:00020C88      P_SpawnMacro
0002:00020D48      P_RunMacroBatch
```

0002:00020E14 P_MacroToggle
0002:00020E6C P_MacroLoop
0002:00020F84 P_RestartMacro
0002:00021060 P_SwitchTexture
0002:00021208 P_StartButton
0002:00021370 P_AddThinker
0002:0002139C P_RemoveThinker
0002:000213C8 P_RunThinkers
0002:0002147C M_Ticker
0002:00021600 P_Ticker
0002:000216B8 P_Drawer
0002:00021850 P_Start
0002:00021958 P_Stop
0002:00021A20 P_PlayerXYMovment
0002:00021B38 P_PlayerZMovement
0002:00021C60 P_PlayerTic
0002:00021D54 P_BuildMove
0002:000221BC P_Thrust
0002:00022270 P_CalcHeight
0002:0002242C P_MovePlayer
0002:00022514 P_DeathThink
0002:0002271C P_PlayerInSpecialSector
0002:00022960 P_PlayerThink
0002:00022D80 R_InitData
0002:00022E7C R_InitTextures
0002:00022F78 R_InitSprites
0002:00022FE0 R_Init
0002:00023048 R_SetupFrame
0002:0002376C R_PointOnSide
0002:00023844 R_PointInSubsector
0002:00023910 R_SlopeDiv
0002:00023960 R_PointToAngle
0002:00023B30 R_RenderPlayerView
0002:00023C20 R_RenderBSPNode
0002:00023D70 R_CheckBBox
0002:0002411C R_Subsector
0002:00024204 R_AddClipLine
0002:00024698 R_ProjectSprite
0002:00024A64 R_RenderBSPNodeNoClip
0002:00024C60 P_SetupSky
0002:00024D20 F_SKYJ
0002:00024DB8 F_SKYB
0002:00024E1C F_SKYC
0002:00024EDC F_SKYD
0002:00024FA0 F_SKYF
0002:00024FB0 F_SKYG
0002:00024FC8 F_SKYH
0002:00025000 F_SKYK
0002:00025040 R_DrawSky
0002:000251B8 P_CloudTicker
0002:000252B4 R_DrawFlatSky
0002:00025338 R_DrawEvilSky
0002:00025478 P_DrawCloud
0002:000257DC R_DrawSkyPic
0002:00025B68 P_FireTicker
0002:00026018 P_ThunderTicker
0002:00026190 R_SetupRenderInfo
0002:00026238 R_RenderWorld

0002:00026644	R_RenderSeg
0002:00026D38	R_RenderLine
0002:00027254	R_RenderSwitch
0002:00027768	R_RenderLeafs
0002:00027E48	R_RenderThings
0002:000288CC	R_RenderLaser
0002:00028B20	R_RenderPSprites
0002:00029190	S_Init
0002:000293A8	S_SetSoundVolume
0002:000293F4	S_SetMusicVolume
0002:0002943C	S_StartMusic
0002:00029478	S_StopMusic
0002:000294A4	S_PauseSound
0002:000294C8	S_ResumeSound
0002:000294E8	S_StopSound
0002:0002951C	S_UpdateSounds
0002:00029570	S_StartSound
0002:00029648	S_AdjustSoundParams
0002:000297A0	ST_Init
0002:00029800	G_ResetPlayerVars
0002:00029888	ST_Ticker
0002:000299C0	ST_Draw
0002:00029F6C	ST_Message
0002:0002A39C	ST_DrawNumber
0002:0002A530	ST_DrawString
0002:0002A6F4	ST_GetUniqueSymbol
0002:0002A830	ST_UpdateFlash
0002:0002A9EC	ST_DrawSymbol
0002:0002AED0	G_PlayDemo
0002:0002AF58	D_TitleMap
0002:0002AFE8	D_WarningTicker
0002:0002B030	D_DrawWarning
0002:0002B1F8	D_LegalTicker
0002:0002B244	D_DrawLegal
0002:0002B3A0	D_NoPakTicker
0002:0002B3CC	nullsub_11
0002:0002B3D4	D_DrawNoPak
0002:0002B588	D_SplashScreen
0002:0002B634	D_Credits
0002:0002B688	D_CreditTicker
0002:0002B7E4	D_CreditDrawer
0002:0002BA28	D_FadeScreen
0002:0002BAC0	W_Init
0002:0002BCF4	W_GetLumpNum
0002:0002BDB8	W_GetNumForName
0002:0002BE04	W_LumpLength
0002:0002BE60	W_ReadLump
0002:0002C030	W_CacheLumpNum
0002:0002C17C	W_GetLumpForName
0002:0002C1B0	W_CacheMapLump
0002:0002C348	W_FreeMapLump
0002:0002C37C	W_MapLumpLength
0002:0002C490	W_GetMapLump
0002:0002C4F0	Z_Init
0002:0002C534	Z_InitZone
0002:0002C570	Z_Update
0002:0002C57C	Z_Malloc2
0002:0002C7E0	Z_Alloc

```
0002:0002CA28      Z_Free
0002:0002CA8C      Z_FreeTags
0002:0002CB9C      Z_Touch
0002:0002CBEC      Z_CheckZone
0002:0002CCF0      Z_ChangeTag
0002:0002CD88      Z_FreeMemory
0002:0002CDC8      Z_DumpHeap
0002:0002CDD0      W_GetDecodeByte
0002:0002CE14      W_WriteOutput
0002:0002CEF4      W_DecodeScan
0002:0002CFB8      W_RescanByte
0002:0002D068      W_InitDecodeTable
0002:0002D224      W_CheckTable
0002:0002D32C      W_DecodeByte
0002:0002D504      W_StartDecodeByte
0002:0002DB98      nullsub_12
0002:0002DBA0      W_Decompress
0002:0002DDF4      W_DecodeJaguar
0002:0002DF34      read_rom
0002:0002E01C      S_InitOS
0002:0002E210      S_InitAL
0002:0002E7D8      I_SetAudioFrame
0002:0002ECF0      osAckRamromRead
0002:0002ED00      S_InitPlayer
0002:0002ED54      S_ALPlayerHandler
0002:0002ED80      S_SaveChannelData
0002:0002EF48      S_InitChannels
0002:0002F038      S_ClearSn64Bank
0002:0002F074      S_GetSoundInfoPointer
0002:0002F0CC      S_GetSSEQTrack
0002:0002F28C      nullsub_13
0002:0002F29C      nullsub_14
0002:0002F370      S_LoadSN641
0002:0002F63C      S_LoadSN642
0002:0002FC2C      S_CalculateSeqClock
0002:0002FEF8      S_AddTrackToPlaylist
0002:0002FF6C      S_PlayTrack
0002:000302B8      nullsub_15
0002:000303AC      SEQ_SetStopState
0002:000304BC      SEQ_StopTrack
0002:000304FC      nullsub_16
0002:00030504      SEQ_KillAllNotes
0002:00030778      SEQ_SetResetState
0002:00030868      I_UpdateTracks
0002:000308A8      nullsub_17
0002:000308B0      SEQ_ResetNotes
0002:000308D4      SEQ_UpdateNotes
0002:00030BE8      nullsub_18
0002:00030D18      nullsub_19
0002:00030D90      SEQ_SetSndVolState
0002:00030DCC      SEQ_UpdateSoundVolume
0002:00030FDC      SEQ_SetMusVolState
0002:00031018      SEQ_UpdateMusicVolume
0002:0003122C      nullsub_20
0002:000313B0      nullsub_21
0002:00031674      nullsub_22
0002:00031814      SEQ_SetPauseState
0002:000318CC      SEQ_PauseNotes
```

0002:00031AF4 SEQ_SetResumeState
0002:00031B9C SEQ_ResumeNotes
0002:00031D60 S_DoPlayMusic
0002:00031DA8 S_StartSoundAtVolume
0002:000320A4 nullsub_23
0002:00032358 SEQ_SetStopSoundState
0002:00032468 SEQ_StopSoundSource
0002:000324A8 nullsub_24
0002:000324B0 SEQ_StopSounds
0002:000324E4 SEQ_StopNotes
0002:00034E90 S_SaveHeap
0002:00034E9C S_ReadHeap
0002:00034F1C nullsub_1
0002:00034F24 nullsub_2
0002:00034F5C I_LockMutex
0002:00034F9C I_UnlockMutex
0002:00034FDC S_GetDefaultTimeDiv
0002:00034FE4 S_CalculateTimeDivision
0002:00035058 S_VoiceHandler
0002:00035120 W_Read
0002:000351B4 nullsub_56
0002:000358C0 SEQ_WaitOnGameState
0002:000358F0 SEQ_SetGameState
0002:00035948 SEQ_FinishGameState
0002:00035A38 SEQ_ExecGameState
0002:00035B1C nullsub_57
0002:00035B40 nullsub_58
0002:00035B78 nullsub_59
0002:00035B80 SEQ_GetTimeDiv
0002:00035EA0 SEQ_PreSetSample
0002:00035ED0 SEQ_PreSetPitch
0002:00035F04 SEQ_PreSetGlobalVolume
0002:00035F20 SEQ_PreSetPanning
0002:00036CD8 SEQ_PreSetTempo
0002:00036DC4 SEQ_PreJump
0002:00036E8C SEQ_PreEndTrack
0002:00036FAC SEQ_ReadMidi
0002:00037600 alCents2Ratio
0002:00037664 SEQ_StartVoice
0002:000379A8 SEQ_SetupSound
0002:00037C6C SEQ_Null_0x01
0002:00037C74 SEQ_UpdateVoices
0002:00037DAC SEQ_Null_0x03
0002:00037DB4 SEQ_Null_0x04
0002:00037DBC SEQ_EndTrack
0002:00037E54 SEQ_StopVoices
0002:00037FF8 SEQ_SetSample
0002:0003801C SEQ_Null_0x08
0002:00038024 SEQ_SetPitchBend
0002:000382C8 SEQ_Null_0x0A
0002:000382D0 SEQ_Null_0x0B
0002:000382D8 SEQ_SetGlobalVolume
0002:000384FC SEQ_SetPanning
0002:000386A0 SEQ_ChannelAfterTouch
0002:000387D8 SEQ_Unknown_0x0F
0002:0003892C SEQ_Null_0x10
0002:00038934 SEQ_BeginVoice
0002:000389E4 SEQ_KillVoice

0002:00038AF8	SEQ_KillSound
0002:00038BBC	SEQ_ReleaseSound
0002:00038DF8	SEQ_PlaySound
0002:00039034	SEQ_PlayNote
0002:000391B4	SEQ_StopNote
0002:0003930C	S_GetSfxCount
0002:00039820	S_LoadSSEQ_Instruments
0002:000398E4	S_LoadSSEQ_Sounds
0002:00039FE0	I_SignalOn
0002:0003A000	I_SignalOff
0002:0003A044	sprintf
0002:0003A0A0	osInitialize
0002:0003A340	osCreateThread
0002:0003A490	osStartThread
0002:0003A5E0	osCreatePiManager
0002:0003A770	osSetThreadPri
0002:0003A850	osYieldThread
0002:0003A8A0	osRecvMesg
0002:0003A9E0	osSpTaskYield
0002:0003AA00	osWritebackDCacheAll
0002:0003AB4C	osSpTaskLoad
0002:0003ACAC	osSpTaskStartGo
0002:0003ACF0	osContStartReadData
0002:0003ADB4	osContGetReadData
0002:0003AF50	osSendMesg
0002:0003B120	osSpTaskYielded
0002:0003B1A0	osViSwapBuffer
0002:0003B1F0	osViBlack
0002:0003B260	osCreateMesgQueue
0002:0003B2D0	osCreateViManager
0002:0003B61C	n_alSynFreeFX
0002:0003B630	osViSetMode
0002:0003B6A0	osViSetSpecialFeatures
0002:0003B860	osViSetXScale
0002:0003B990	osViSetYScale
0002:0003B9F0	osSetEventMesg
0002:0003BA60	osViSetEvent
0002:0003BAD0	osContInit
0002:0003BCC8	__osContGetInitData
0002:0003BD98	__osPackRequestData
0002:0003BE90	osJamMesg
0002:0003BFE0	osViGetCurrentMode
0002:0003C030	osPfsIsPlug
0002:0003C1D0	__osPfsRequestData
0002:0003C2CC	__osPfsGetInitData
0002:0003C3A0	osPfsInit
0002:0003C454	__osPfsGetStatus
0002:0003C560	osPfsNumFiles
0002:0003C6B0	osPfsChecker
0002:0003CD74	corrupted_init
0002:0003CF28	corrupted
0002:0003D110	osPfsFileState
0002:0003D400	osPfsDeleteFile
0002:0003D6E0	__osPfsReleasePages
0002:0003D918	__osBlockSum
0002:0003DB0C	osPfsReadWriteFile
0002:0003DF10	osPfsFindFile
0002:0003E0D0	osPfsAllocateFile

```
0002:0003E554    __osPfsDeclearPage
0002:0003E880    D_Sqrt
0002:0003E890    sinf
0002:0003EA50    guFrustumF
0002:0003EBB0    guFrustum
0002:0003EC20    alHeapInit
0002:0003EC60    alHeapDBAlloc
0002:0003ECC0    W_CopyData
0002:0003ED70    osPiStartDma
0002:0003EE80    osAiSetFrequency
0002:0003EFE0    alUnlink
0002:0003F010    alLink
0002:0003F034    alClose
0002:0003F06C    alInit
0002:0003F0A0    osVirtualToPhysical
0002:0003F120    osAiSetNextBuffer
0002:0003F1D0    osAiGetLength
0002:0003F1E0    nullsub_60
0002:0003F1E8    _timeToSamples
0002:0003F240    _freePVoice
0002:0003F278    _collectPVoices
0002:0003F2D8    __freeParam
0002:0003F2F0    __allocParam
0002:0003F320    nullsub_61
0002:0003F328    alAudioFrame
0002:0003F5C0    alSynNew
0002:0003F8C0    alSynAddPlayer
0002:0003F910    __ull_rshift
0002:0003F93C    __ull_rem
0002:0003F978    __ull_div
0002:0003F9B4    __ll_lshift
0002:0003F9E0    __ll_rem
0002:0003FA1C    __ll_div
0002:0003FA78    __ll_mul
0002:0003FAA8    __ull_divremi
0002:0003FB08    __ll_mod
0002:0003FBA4    __ll_rshift
0002:0003FBD0    _allocatePVoice
0002:0003FCB8    alSynAllocVoice
0002:0003FE00    alSynStartVoiceParams
0002:0003FEF0    alSynSetPitch
0002:0003FF80    alSynSetVol
0002:00040020    alSynSetPan
0002:000400B0    alSynStopVoice
0002:00040130    alSynFreeVoice
0002:000401E0    alSynSetPriority
0002:00040860    _Printf
0002:00040EB0    memcpy
0002:00040EDC    strlen
0002:00040F04    strchr
0002:00040F50    __osSetSR
0002:00040F60    __osGetSR
0002:00040F70    __osSetFpcCsr
0002:00040F80    __osSiRawReadIo
0002:00040FD0    __osSiRawWriteIo
0002:00041020    __osExceptionPreamble
0002:00041030    __osException
0002:00041554    send_mesg
```

```
0002:00041608      handle_CpU
0002:0004163C      __osEnqueueAndYield
0002:0004173C      __osEnqueueThread
0002:00041784      __osPopThread
0002:00041794      __osDispatchThread
0002:00041910      __osCleanupThread
0002:000419A0      osInvalICache
0002:00041A20      osMapTLBRdb
0002:00041A80      osPiRawReadIo
0002:00041AE0      _blkclr
0002:00041B80      __osSetHWIntrRoutine
0002:00041BD0      __osLeoInterrupt
0002:00042490      __osDisableInt
0002:000424B0      __osRestoreInt
0002:000424D0      __osDequeueThread
0002:00042510      __osPiCreateAccessQueue
0002:00042560      __osPiGetAccess
0002:000425A4      __osPiRelAccess
0002:000425D0      osGetThreadPri
0002:000425F0      osPiRawStartDma
0002:000426D0      osEPiRawStartDma
0002:000427B0      __osDevMgrMain
0002:00042B58      nullsub_62
0002:00042B60      __osSpSetStatus
0002:00042B70      _bcopy
0002:00042E80      __osSpSetPc
0002:00042EC0      __osSpRawStartDma
0002:00042F50      __osSpDeviceBusy
0002:00042F80      __osPiCreateAccessQueue_0
0002:00042FD0      __osPiGetAccess_0
0002:00043014      __osPiRelAccess_0
0002:00043040      __osSiRawStartDma
0002:000430F0      __osViInit
0002:00043240      __osSpGetStatus
0002:00043250      __osTimerServicesInit
0002:000432DC      __osTimerInterrupt
0002:00043520      __osSetTimerIntr
0002:00043594      __osInsertTimer
0002:00043660      __osViSwapContext
0002:000439C0      osGetCount
0002:000439D0      osGetTime
0002:00043A60      osSetTimer
0002:00043B40      __osSumcalc
0002:00043B9C      __osIdCheckSum
0002:00043C04      __osRepairPackId
0002:0004401C      __osCheckPackId
0002:000441B4      __osGetId
0002:00044410      __osCheckId
0002:0004450C      __osPfsRWInode
0002:0004482C      __osPfsSelectBank
0002:000448A0      __osContRamRead
0002:00044C50      __osContRamWrite
0002:00045000      osPfsFreeBlocks
0002:00045150      guMtxF2L
0002:00045250      guMtxIdentF
0002:000452D8      guMtxIdent
0002:00045308      guMtxL2F
0002:000453C0      osPiGetCmdQueue
```

```
0002:000453F0      alSynDelete
0002:00045400      __osProbeTLB
0002:000454C0      __osAiDeviceBusy
0002:000454F0      alSaveNew
0002:00045534      alMainBusNew
0002:00045588      alAuxBusNew
0002:000455DC      alResampleNew
0002:00045664      alLoadNew
0002:0004570C      alEnvmixerNew
0002:000457B0      init_lpfilter
0002:00045850      alFxNew
0002:00045C90      alSynAllocFX
0002:00045D30      alMainBusParam
0002:00045D60      alMainBusPull
0002:00045E80      alLoadParam
0002:00046054      alRaw16Pull
0002:00046400      _decodeChunk
0002:0004652C      alAdpcmPull
0002:000469D0      alResampleParam
0002:00046ABC      alResamplePull
0002:00046CD0      _ldexpf
0002:00046CF8      _frexpf
0002:00046DE4      alEnvmixerParam
0002:000473DC      alEnvmixerPull
0002:00047930      alAuxBusParam
0002:00047960      alAuxBusPull
0002:00047A40      alSaveParam
0002:00047A74      alSavePull
0002:00047B00      osSetIntMask
0002:00047BA0      _Litob
0002:000483A8      nullsub_63
0002:000483B0      _Ldtob
0002:00048900      __osSiDeviceBusy
0002:00048930      osDestroyThread
0002:00048A30      osLeoDiskInit
0002:00048B20      __osResetGlobalIntMask
0002:00048B80      osEPiRawWriteIo
0002:00048BD0      __osSetCompare
0002:00048BE0      __osContAddressCrc
0002:00048C90      __osContDataCrc
0002:00048D60      alFilterNew
0002:00048D80      _doModFunc
0002:00048E28      _filterBuffer
0002:00048EE0      _saveBuffer
0002:00049068      _loadBuffer
0002:000491F4      _loadOutputBuffer
0002:00049418      alFxParamHdl
0002:00049674      alFxParam
0002:0004968C      alFxPull
0002:000499D0      alCopy
0002:00049A50      lldiv
0002:00049B50      ldiv
0002:0004A2F0      nullsub_64
0002:0004AC10      nullsub_65
0002:0004B864      nullsub_66
0002:0004BF10      nullsub_67
0002:0004BF60      nullsub_68
0002:0004BFB8      osAckRamromRead_0
```

0002:0004C108	nullsub_69
0002:0004C134	nullsub_70
0002:0004CFF4	StateInfo
0002:0004D008	StateDefs
0002:0004D01C	states
0002:00050E38	MobjInfo
0002:00059D80	demolump
0002:00059D84	demplump_ptr
0002:00059D88	rndindex
0002:00059D8C	prndindex
0002:00059D90	m_randtable
0002:0005A070	gamestate
0002:0005A3A0	mRunHecticDemo
0002:0005A3A8	menualpha
0002:0005A3BC	st_drawstatus
0002:0005A3C0	s_soundvolume
0002:0005A3C4	s_musicvolume
0002:0005A3C8	brightness
0002:0005A3D0	bEnableFeatures
0002:0005A414	buttonmapping
0002:0005A440	buttonmappingdemo
0002:0005A474	buttonmappingmenu
0002:0005A664	passwordSymbols
0002:0005A7A0	MenuStrings
0002:0005A8A8	aRvnh3ct1cd3m0???
0002:0005A8BC	in_password
0002:0005AA70	plasma_animatepic
0002:0005AA80	AnimPicsDef
0002:0005AA84	aSmonaa
0002:0005AAA4	aSmonba
0002:0005AAC4	aSmonca
0002:0005AAE4	aCfacea
0002:0005AB04	aSmonda
0002:0005AB24	aSmonea
0002:0005AB44	aSporta
0002:0005AB64	aSmonf
0002:0005AB84	aStrakr
0002:0005ABA4	aStrakb
0002:0005ABC4	aStraky
0002:0005ABE4	aC307b
0002:0005AC04	aCtel
0002:0005AC24	aCasfl98
0002:0005AC44	aHtela
0002:0005AC60	forwardmove
0002:0005AC68	sidemove
0002:0005AC70	angleturn
0002:0005ADD0	backskycolor
0002:0005AE10	firecolor
0002:0005AE50	prev_mus
0002:0005AE60	symboldata
0002:0005B490	finecosine
0002:0005B498	tantoangle
0002:0005D4C0	audioframe
0002:0005D4EC	SN64Count
0002:0005D567	numChannelInfo
0002:0005D598	mutexcnt
0002:0005D5A0	seq_gamestate
0002:0005D5DC	seq_gamestatecounter

0002:0005D5E0 seq_statequeuefull
0002:0005D68C sequence_event_functable
0002:0005D6D8 midi_info
0002:0005D6F0 seq_midievents
0002:0005E4F0 ALSynthDriver
0002:0005E534 a00000000000000
0002:0005EA30 aSpot
0002:0005EA38 aPlay
0002:0005EA40 aSarg
0002:0005EA48 aFatt
0002:0005EA50 aPoss
0002:0005EA58 aTroo
0002:0005EA60 aHead
0002:0005EA68 aBoss
0002:0005EA70 aSkul
0002:0005EA78 aBspi
0002:0005EA80 aCybr
0002:0005EA88 aPain
0002:0005EA90 aRect
0002:0005EA98 aMisl
0002:0005EAA0 aPlss
0002:0005EAA8 aBfs1
0002:0005EAB0 aLass
0002:0005EAB8 aBal1
0002:0005EAC0 aBal3
0002:0005EAC8 aBal2
0002:0005EAD0 aBal7
0002:0005EAD8 aBal8
0002:0005EAE0 aApls
0002:0005EAE8 aManf
0002:0005EAF0 aTrcr
0002:0005EAF8 aDart
0002:0005EB00 aFire
0002:0005EB08 aRbal
0002:0005EB10 aPuf2
0002:0005EB18 aPuf3
0002:0005EB20 aPuff
0002:0005EB28 aBlud
0002:0005EB30 aA027
0002:0005EB38 aTfog
0002:0005EB40 aBfe2
0002:0005EB48 aArm1
0002:0005EB50 aArm2
0002:0005EB58 aBon1
0002:0005EB60 aBon2
0002:0005EB68 aBkey
0002:0005EB70 aRkey
0002:0005EB78 aYkey
0002:0005EB80 aYsku
0002:0005EB88 aRsku
0002:0005EB90 aBsku
0002:0005EB98 aArt1
0002:0005EBA0 aArt2
0002:0005EBA8 aArt3
0002:0005EBB0 aStim
0002:0005EBB8 aMedi
0002:0005EBC0 aSoul
0002:0005EBC8 aPinv

0002:0005EBD0 aPstr
0002:0005EBD8 aPins
0002:0005EBE0 aSuit
0002:0005EBE8 aPmap
0002:0005EBF0 aPvis
0002:0005EBF8 aMega
0002:0005EC00 aClip
0002:0005EC08 aAmmo
0002:0005EC10 aRckt
0002:0005EC18 aBrok
0002:0005EC20 aCell
0002:0005EC28 aCelp
0002:0005EC30 aShel
0002:0005EC38 aSbox
0002:0005EC40 aBpak
0002:0005EC48 aBfug
0002:0005EC50 aCsaw
0002:0005EC58 aMgun
0002:0005EC60 aLaun
0002:0005EC68 aPlsm
0002:0005EC70 aShot
0002:0005EC78 aSgn2
0002:0005EC80 aLsrg
0002:0005EC88 aCand
0002:0005EC90 aBar1
0002:0005EC98 aLmp1
0002:0005ECA0 aLmp2
0002:0005ECA8 aA031
0002:0005ECB0 aA030
0002:0005ECB8 aA032
0002:0005ECC0 aA033
0002:0005ECC8 aA034
0002:0005ECD0 aBflm
0002:0005ECD8 aRflm
0002:0005ECE0 aYflm
0002:0005ECE8 aA006
0002:0005ECF0 aA021
0002:0005ECF8 aA003
0002:0005ED00 aA020
0002:0005ED08 aA014
0002:0005ED10 aA016
0002:0005ED18 aA008
0002:0005ED20 aA007
0002:0005ED28 aA015
0002:0005ED30 aA001
0002:0005ED38 aA012
0002:0005ED40 aA010
0002:0005ED48 aA018
0002:0005ED50 aA017
0002:0005ED58 aA026
0002:0005ED60 aA022
0002:0005ED68 aA028
0002:0005ED70 aA029
0002:0005ED78 aA035
0002:0005ED80 aA036
0002:0005ED88 aTre3
0002:0005ED90 aTre2
0002:0005ED98 aTre1

```
0002:0005EDA0       aA013
0002:0005EDA8       aA019
0002:0005EDB0       aA004
0002:0005EDB8       aA005
0002:0005EDC0       aA023
0002:0005EDC8       aSawg
0002:0005EDD0       aPung
0002:0005EDD8       aPisg
0002:0005EDE0       aSht1
0002:0005EDE8       aSht2
0002:0005EDF0       aChgg
0002:0005EDF8       aRock
0002:0005EE00       aPlas
0002:0005EE08       aBfgg
0002:0005EE10       aLasr
0002:0005EE20       aLevelDS
0002:0005EE70       aDemo1lmp
0002:0005EE7C       aDemo2lmp
0002:0005EE88       aDemo3lmp
0002:0005EE94       aDemo4lmp
0002:0005EEA0       aHahahaha
0002:0005EEAC       aYouShouldnTHaveDoneThat_
0002:0005EECC       aTryAnEasierLevel___
0002:0005EEE4       aWowLookAtThoseDemonFeet_
0002:0005EF04       aIPlayDoomAndICanTGetUp_
0002:0005EF24       aOuchThatHadToHurt_
0002:0005EF3C       aLookAtMeIMFlat
0002:0005EF54       aThanksForPlaying
0002:0005EF68       aYouLazy@
0002:0005EF78       aHaveYouHadEnough?
0002:0005EF90       aTheDemonsGaveY
0002:0005EFB0       aTheLeadDemonVa
0002:0005EFD0       aYouCackleAsThe
0002:0005EFE4       aFamiliarityOfT
0002:0005EFF8       aSituationOccur
0002:0005F014       aTheGatewayToTh
0002:0005F030       aDomainWasTooAc
0002:0005F04C       aYouRealizeTheD
0002:0005F068       aYouWithTheirIn
0002:0005F084       aItDoesNotMatte
0002:0005F09C       aTheDemonsSpawn
0002:0005F0B8       aAndYouHaveTheG
0002:0005F0D4       aU_a_c_PoisonTh
0002:0005F0F0       aYourBloodthirs
0002:0005F10C       aShattersTheTel
0002:0005F128       aOnceAgainYouFi
0002:0005F148       aAmidst___
0002:0005F158       aTheVastSilence
0002:0005F174       aYouOfTheMilita
0002:0005F194       aYouKnewTheInst
0002:0005F1B0       aHadAClassified
0002:0005F1CC       aTheU_a_c_HadSo
0002:0005F1E8       aReasonToHideTh
0002:0005F208       aYouWonderWhatIt
0002:0005F21C       aCouldBe___
0002:0005F22C       aYouSmile_
0002:0005F23C       aWhatStrangePlaceHave
0002:0005F254       aYouStumbledUpon?
```

```
0002:0005F26C       aTheDemonsDidNotExpect
0002:0005F288       aYouToSurviveThisFar_
0002:0005F2A4       aYouFeelTheirDe
0002:0005F2BC       aPresenceWaitin
0002:0005F2DC       aLetThemTasteTh
0002:0005F2FC       aYouWretchAsASt
0002:0005F314       aAcridOdorAssau
0002:0005F334       aDeathAndDemonCarcass
0002:0005F354       aNoNightmareCouldHave
0002:0005F36C       aPreparedYouForThis_
0002:0005F388       aYouRealizeThatThis
0002:0005F3A0       aPlaceWasNotMeantFor
0002:0005F3B8       aLivingHumans_
0002:0005F3CC       aCongratulation
0002:0005F3F4       aHectic_0
0002:0005F400       aOnlyTheBestWillReap
0002:0005F418       aItsRewards_
0002:0005F42C       aFinally___
0002:0005F438       aTheMotherOfAllDemons
0002:0005F454       aIsDead
0002:0005F464       aTheBloodPoursFrom
0002:0005F47C       aYourEyesAsYouS
0002:0005F494       aInDefiance_
0002:0005F4A8       aAsTheOnlyMarineTo
0002:0005F4C0       aEndureTheSlaughter
0002:0005F4D8       aYouDecideToRemain
0002:0005F4F0       aInHellAndEnsur
0002:0005F508       aDemonEverRises
0002:0005F524       aTheEnd_
0002:0005F534       aZombieman
0002:0005F540       aShotgunGuy
0002:0005F54C       aImp
0002:0005F550       aNightmareImp
0002:0005F560       aDemon
0002:0005F568       aSpectre
0002:0005F570       aLostSoul
0002:0005F57C       aCacodemon
0002:0005F588       aHellKnight
0002:0005F594       aBaronOfHell
0002:0005F5A4       aArachnotron
0002:0005F5B0       aPainElemental
0002:0005F5C0       aMancubus
0002:0005F5CC       aTheCyberdemon
0002:0005F5DC       aOurHero
0002:0005F5E8       aEvil
0002:0005F5F0       aFinal
0002:0005F5F8       aFinal_0
0002:0005F600       aEvil_0
0002:0005F6E0       aStagingArea
0002:0005F6F0       aTheTerraformer
0002:0005F700       aMainEngineerin
0002:0005F714       aHoldingArea
0002:0005F724       aTechCenter
0002:0005F730       aAlphaQuadrant
0002:0005F740       aResearchLab
0002:0005F750       aFinalOutpost
0002:0005F760       aEvenSimpler
0002:0005F770       aTheBleeding
```

```
0002:0005F780       aTerrorCore
0002:0005F78C       aAltarOfPain
0002:0005F79C       aDarkCitadel
0002:0005F7AC       aEyeOfTheStorm
0002:0005F7C0       aDarkEntries
0002:0005F7D0       aBloodKeep
0002:0005F7DC       aWatchYourStep
0002:0005F7EC       aSpawnedFear
0002:0005F7FC       aTheSpiral
0002:0005F808       aBreakdown
0002:0005F814       aPitfalls
0002:0005F820       aBurntOfferings
0002:0005F830       aUnholyTemple
0002:0005F840       aNoEscape
0002:0005F84C       aCatAndMouse
0002:0005F85C       aHardcore
0002:0005F868       aPlayground
0002:0005F874       aTheAbsolution
0002:0005F884       aOutpostOmega
0002:0005F894       aTheLair
0002:0005F8A0       aInTheVoid
0002:0005F8AC       aHectic
0002:0005F8B4       aTitle_0
0002:0005F8BC       a2_2d2_2d
0002:0005F8D0       aEvil_2
0002:0005F8D8       aFinished
0002:0005F8E4       aKills
0002:0005F8F0       aItems
0002:0005F8FC       aSecrets
0002:0005F908       aTime
0002:0005F910       aEntering
0002:0005F91C       aPassword
0002:0005F940       aI_checkgfxGfxO
0002:0005F960       aI_checkgfxGfxC
0002:0005F980       aI_checkvtxVtxO
0002:0005F9A0       aI_checkgfxVtxC
0002:0005F9C0       aI_drawframeGfx
0002:0005F9E4       aI_drawframeVtx
0002:0005FA10       aDefaultD
0002:0005FA1C       aRight
0002:0005FA24       aLeft
0002:0005FA2C       aForward
0002:0005FA34       aBackward
0002:0005FA40       aAttack
0002:0005FA48       aUse
0002:0005FA4C       aMap
0002:0005FA50       aSpeed
0002:0005FA58       aStrafeOn
0002:0005FA64       aStrafeLeft
0002:0005FA70       aStrafeRight
0002:0005FA80       aWeaponBackward
0002:0005FA90       aWeaponForward
0002:0005FAA0       aControlPad
0002:0005FAAC       aVolume
0002:0005FAB4       aDisplay
0002:0005FABC       aPassword_1
0002:0005FAC8       aMainMenu
0002:0005FAD4       aRestartLevel
```

```
0002:0005FAE4       aRReturn
0002:0005FAF0       aMusicVolume
0002:0005FB00       aSoundVolume
0002:0005FB10       aBrightness
0002:0005FB1C       aResume
0002:0005FB24       aOptions
0002:0005FB2C       aDefault
0002:0005FB34       aDefault_0
0002:0005FB3C       aNewGame
0002:0005FB48       aBeGentle
0002:0005FB54       aBringItOn
0002:0005FB64       aIOwnDoom
0002:0005FB70       aWatchMeDie
0002:0005FB80       aNightmare
0002:0005FB8C       aYes
0002:0005FB90       aNo
0002:0005FB94       aFeatures_0
0002:0005FBA0       aWarpToLevel
0002:0005FBB0       aInvulnerable
0002:0005FBC0       aHealthBoost
0002:0005FBD0       aSecurityKeys
0002:0005FBE0       aWeapons
0002:0005FBE8       aExit
0002:0005FBF0       aDebug
0002:0005FBF8       aTextureTest
0002:0005FC08       aWallBlocking
0002:0005FC18       aCenterDisplay
0002:0005FC28       aMessages
0002:0005FC34       aStatusBar
0002:0005FC40       aLockMonsters
0002:0005FC50       aScreenshot
0002:0005FC5C       aMapEverything
0002:0005FC6C       aMacroPeek
0002:0005FC78       aMusicTest
0002:0005FC84       aWarpToFun
0002:0005FC90       aControlStick
0002:0005FCA0       aDefault_1
0002:0005FCA8       aSensitivity
0002:0005FCB4       aManagePak
0002:0005FCC0       aDoNotUsePak
0002:0005FCD0       aTryAgain
0002:0005FCDC       aCreateGameNote_0
0002:0005FCF0       aTitle
0002:0005FCF8       aPause
0002:0005FD00       aPressNToResume
0002:0005FD14       aChooseYourSkil
0002:0005FD2C       aOptions_0
0002:0005FD34       aQuitGame?
0002:0005FD40       aDeleteGameNote
0002:0005FD54       aControllerPakB
0002:0005FD68       aControllerPakF
0002:0005FD7C       aCreateGameNote
0002:0005FD90       aFeatures
0002:0005FD9C       aS
0002:0005FDA0       aOn
0002:0005FDA4       aOff
0002:0005FDA8       aOn_0
0002:0005FDAC       aOff_0
```

0002:0005FDB0 aOn_1
0002:0005FDB4 aOff_1
0002:0005FDB8 aOff_2
0002:0005FDBC aOn_2
0002:0005FDC0 aOn_3
0002:0005FDC4 aOff_3
0002:0005FDC8 aOn_4
0002:0005FDCC aOff_4
0002:0005FDD0 aOn_5
0002:0005FDD4 aOff_5
0002:0005FDD8 a100
0002:0005FDE4 a100_0
0002:0005FDF0 a100_1
0002:0005FDFC aOn_6
0002:0005FE00 aOff_6
0002:0005FE04 aD
0002:0005FE08 aNotImplemented
0002:0005FE18 aVolume_0
0002:0005FE20 aControlStick_0
0002:0005FE30 aDisplay_0
0002:0005FE38 aOn_7
0002:0005FE3C aOff_7
0002:0005FE40 aOn_8
0002:0005FE44 aOff_8
0002:0005FE48 aControllerPak
0002:0005FE58 aControllerPakRemoved
0002:0005FE70 aPressNToExit_0
0002:0005FE80 aEmpty
0002:0005FE88 aPMore___
0002:0005FE94 aOMore___
0002:0005FEA0 aPagesUsedDFreeD
0002:0005FEBC aPressNToExit_1
0002:0005FECC aPressDeToDelete
0002:0005FEE0 aLevel2_2d
0002:0005FEEC aEvil_3
0002:0005FEF4 aControllerPak_0
0002:0005FF04 aControllerPakRemoved_0
0002:0005FF1C aGameCannotBeSaved_
0002:0005FF34 aPressNToExit_2
0002:0005FF44 aEmpty_0
0002:0005FF4C aPMore____0
0002:0005FF58 aOMore____0
0002:0005FF64 aPressNToExit_3
0002:0005FF74 aPressDeToSave
0002:0005FF88 aControllerPak_1
0002:0005FF9C aNoSave
0002:0005FFA4 aPMore____1
0002:0005FFB0 aOMore____1
0002:0005FFBC aPressNToExit_4
0002:0005FFCC aPressDeToLoad
0002:0005FFE0 aCenterDisplay_0
0002:0005FFF0 aUseControlPadToAdjust
0002:0006000C aPressNToExit_5
0002:0006001C aControlPad_0
0002:00060028 aPMore____2
0002:00060034 aOMore____2
0002:00060040 aPressNToExit_6
0002:00060160 aPassword_0

```
0002:0006016C      aInvalidPasswor
0002:00060180      aPressNToExit
0002:00060190      aPressDToChange
0002:000601B0      aP_addactivecei
0002:000601DC      aP_removeactive
0002:000602D0      aP_newchasedirC
0002:000603D0      aP_giveammoBadT
0002:000603E8      aYouPickUpAHeal
0002:00060404      aYouPickUpAnArm
0002:00060420      aSupercharge
0002:00060430      aMegaSphere
0002:00060440      aPickedUpAClip_
0002:00060454      aPickedUpABoxOf
0002:00060470      aPickedUpARocke
0002:00060484      aPickedUpABox_0
0002:000604A0      aPickedUpAnEner
0002:000604BC      aPickedUpAnEn_0
0002:000604DC      aPickedUp8Shotg
0002:000604F8      aPickedUp4Shotg
0002:00060514      aPickedUpABox_1
0002:00060530      aYouGotTheBackp
0002:00060548      aYouGotTheBfg90
0002:00060568      aAChainsawFindS
0002:00060588      aYouGotTheChain
0002:000605A0      aWhatThe@IsThis
0002:000605B8      aYouGotTheRocke
0002:000605D8      aYouGotThePlasm
0002:000605F0      aYouGotTheShotg
0002:00060608      aYouGotTheSuper
0002:00060624      aYouPickUpTheAr
0002:0006063C      aYouGotTheMegaa
0002:00060654      aYouPickUpABlue
0002:00060670      aYouPickUpAYell
0002:00060690      aYouPickUpARedK
0002:000606AC      aYouPickUpABl_0
0002:000606CC      aYouPickUpAYe_0
0002:000606EC      aYouPickUpARedS
0002:0006070C      aYouPickUpAStim
0002:00060724      aYouPickUpAMedi
0002:00060750      aYouPickUpAMe_0
0002:00060768      aInvulnerabilit
0002:0006077C      aBerserk
0002:00060788      aPartialInvisib
0002:000607A0      aRadiationShiel
0002:000607B4      aComputerAreaMa
0002:000607C8      aLightAmplifica
0002:000607E4      aYouHaveAFeelin
0002:00060818      aWhateverItIsIt
0002:0006084C      aItMustDoSometh
0002:000609E0      aP_spawnmapthin
0002:00060A20      aP_addactivepla
0002:00060A40      aP_removeacti_0
0002:00060AC0      aF_skya
0002:00060AC8      aP_loadthingsDo
0002:00060AEC      aP_loadlinedefs
0002:00060B1C      aP_loadleafsLea
0002:00060B50      aP_loadleafsVer
0002:00060B74      aP_loadleafsSeg
```

```
0002:00060B94       aP_grouplinesMi
0002:00060BB0       aP_setuplevelNo
0002:00060BE0       aPs_crosssubsec
0002:00060C10       aYouNeedABlueKe
0002:00060C28       aYouNeedAYellow
0002:00060C40       aYouNeedARedKey
0002:00060C54       aYouLackTheAbil
0002:00061000       aYouFoundASecre
0002:00061020       aT_start
0002:00061028       aT_end
0002:00061030       aSwx
0002:00061034       aS_start_0
0002:0006103C       aS_end_0
0002:0006104C       M_PI
0002:00061050       aR_subsectorSsI
0002:00061080       aSpace
0002:00061088       aCloud
0002:00061090       aMountc
0002:00061098       aCloud_0
0002:000610A0       aCloud_1
0002:000610A8       aMountb
0002:000610B0       aFire_0
0002:000610B8       aMounta
0002:000610C0       aEvil_1
0002:00061100       aS_initAudioHea
0002:00061120       aSfont
0002:00061128       aStatus
0002:00061130       aSymbols
0002:00061140       aWarning
0002:0006114C       aNintendo64Cont
0002:00061164       aIsNotConnected
0002:00061178       aPleaseTurnOffY
0002:00061190       aNintendo64Syst
0002:000611A4       aPlugInYourNint
0002:000611C0       aControllerAndT
0002:000611DC       aUslegal
0002:000611E4       aHoldNToManageP
0002:000611FC       aNoControllerPa
0002:00061210       aYourGameCannot
0002:00061224       aBeSaved_
0002:00061230       aPleaseTurnOffYour
0002:00061248       aNintendo64Sy_0
0002:0006125C       aBeforeInsertin
0002:00061270       aControllerPak_
0002:00061280       aIdcred1
0002:00061288       aIdcred2
0002:00061290       aWmscred1
0002:0006129C       aWmscred2
0002:000612B0       aIwad
0002:000612B8       aW_initInvalidM
0002:000612D8       aW_getnumfornam
0002:000612F8       aW_lumplengthLu
0002:0006131C       aW_readlumpLump
0002:00061340       aW_cachelumpnum
0002:00061368       aW_maplumplengt
0002:0006138C       aW_getmaplumpLu
0002:000613B0       aZ_malloc2Faile
0002:000613D4       aZ_mallocAnOwne
```

0002:00061408 aZ_allocFailedA
0002:0006142C aZ_allocFaile_0
0002:00061450 aZ_allocAnOwner
0002:00061484 aZ_freeFreedAPo
0002:000614AC aZ_touchTouched
0002:000614D8 aZ_checkzoneBlo
0002:000614FC aZ_checkzoneZon
0002:0006152C aZ_checkzoneB_0
0002:00061564 aZ_checkzoneNex
0002:0006159C aZ_changetagFre
0002:000615C8 aZ_changetagAnO
0002:00061600 aOverflowedOutp
0002:00061620 aAudio
0002:00061628 aOsaialign8
0002:00061634 aMaxrspcmds
0002:00061640 aDmaptrnull
0002:0006164C aDmanotdone
0002:00061660 aSn64
0002:00061668 sSN64
0002:00061D00 aRspGfxUcodeF3d
0002:00062500 aRspGfxUcodeL3d
0002:00062D30 latetic
0002:00062D34 leveltime
0002:00062D3C gametic
0002:00062D40 lasttic
0002:00062D44 movefactor
0002:00062D48 btn_setstrafe
0002:00062D4C btn_setspeed
0002:00062DF0 finaleoffset
0002:00062DF8 finaletext
0002:00062DFC finalefadetext
0002:00062E30 gameaction
0002:00062E34 gameskill
0002:00062E38 gamemap
0002:00062E3C nextmap
0002:00062E40 players
0002:00062E44 resetplayer
0002:00062E58 viewheight
0002:00062F08 playercheats
0002:00062F20 st_message
0002:00062F84 am_flags
0002:00062F90 in_starttime
0002:00062F94 in_endtime
0002:00062F98 totalkills
0002:00062F9C totalsecret
0002:00062FA0 totalitems
0002:00062FA8 showhud
0002:00062FB0 in_advance
0002:00062FB4 in_stage
0002:00062FB8 in_killpercent
0002:00062FBC in_itempercent
0002:00062FC0 in_secretpercent
0002:00062FC4 in_kills
0002:00062FC8 in_items
0002:00062FCC in_secret
0002:00062FD0 in_time
0002:000A4600 glistp
0002:000A4604 gfx_head

0002:000A4608 vtx_tail
0002:000A460C vtx_head
0002:000A4B40 gfx_messagequeue
0002:000A4BA0 PiMessageQ
0002:000A4BB8 OSMesgQueue
0002:000A4BD0 PiMessages
0002:000A4E44 gfx_task
0002:000A5180 MenuOptionsCount
0002:000A5184 MenuFunction
0002:000A5198 prev_gametic
0002:000A51A8 pakconnected
0002:000A51D0 mapthing
0002:000A51D4 mapthingflags
0002:000A51D8 mapthingx
0002:000A51DC mapthingy
0002:000A51E0 mapthingradius
0002:000A51E8 mapbbox
0002:000A51F8 mapsubsector
0002:000A51FC mapthingblocker
0002:000A5200 tmhitline
0002:000A5204 mapfloorz
0002:000A5208 mapceilingz
0002:000A520C mapdropoffz
0002:000A5210 activeceilings
0002:000A5290 crushchange
0002:000A5294 nofit
0002:000A52A0 deathmocktics
0002:000A52B0 tmthing
0002:000A52B4 tmx
0002:000A52B8 tmy
0002:000A52BC bombsource
0002:000A52C0 bombspot
0002:000A52C4 bombdamage
0002:000A52C8 blockthing
0002:000A52F8 linetarget
0002:000A52FC shotlineheight
0002:000A5300 shootthing
0002:000A5304 attackrange
0002:000A5308 topslope
0002:000A530C bottomslope
0002:000A5310 aimslope
0002:000A531C shootz
0002:000A5320 aimfrac
0002:000A5324 la_damage
0002:000A5328 bestslidefrac
0002:000A532C bestslideline
0002:000A5330 slidemo
0002:000A5340 opentop
0002:000A5344 openbottom
0002:000A5348 openrange
0002:000A5350 intercepts
0002:000A5950 intercept_p
0002:000A5970 cameraTargetMobj
0002:000A5974 spawnthing_list
0002:000A5978 spawnthing_count
0002:000A5980 moveok
0002:000A5984 floatok
0002:000A5988 tmfloorz

0002:000A598C tmceilingz
0002:000A5990 tmdropoffz
0002:000A5994 tmsubsec
0002:000A5998 validcheckthing
0002:000A59A0 oldtmx
0002:000A59A4 oldtmy
0002:000A59A8 tmbbox
0002:000A59B8 tmflags
0002:000A59BC spechit
0002:000A59C0 spechit2
0002:000A59E0 numspechit
0002:000A59E4 activeplats
0002:000A5A74 soundtarget
0002:000A5A78 bulletslope
0002:000A5A80 vertexes
0002:000A5A84 segs
0002:000A5A88 sectors
0002:000A5A8C subsectors
0002:000A5A90 nodes
0002:000A5A94 lines
0002:000A5A98 sidedefs
0002:000A5A9C lights
0002:000A5AA0 macroData
0002:000A5AA4 lightdata
0002:000A5AA8 blockmaplump
0002:000A5AAC blockmap
0002:000A5AB0 bmapwidth
0002:000A5AB4 bmapheight
0002:000A5AB8 bmaporgx
0002:000A5ABC bmaporgy
0002:000A5AC0 blocklinks
0002:000A5AC4 rejectmatrix
0002:000A5AD0 playerstarts
0002:000A5AE4 numsegs
0002:000A5AE8 numsectors
0002:000A5AEC numsubsectors
0002:000A5AF0 numnodes
0002:000A5AF4 numlines
0002:000A5AF8 numsides
0002:000A5AFC numlights
0002:000A5B00 nummacros
0002:000A5B04 numleafs
0002:000A5B08 displaykeytype_blue
0002:000A5B0C displaykeytype_red
0002:000A5B10 displaykeytype_yellow
0002:000A5B14 skypic
0002:000A5B60 linespeciallist
0002:000A5C90 AnimPics
0002:000A5C94 macroActive
0002:000A5C98 macroSpecial
0002:000A5C9C macroTag
0002:000A5CA0 macroLineDummy
0002:000A5CEC macroLine
0002:000A5CF0 macroThinker
0002:000A5CF4 globalInt
0002:000A5CFC macroRestartCounter
0002:000A5D00 macroqueuedata
0002:000A5D20 macroqueue

0002:000A5D24 macroqueue_unknowna6124
0002:000A5D28 buttonlist
0002:000A5E70 paused
0002:000A5E78 thinkercap
0002:000A5E88 mobjhead
0002:000A5F20 s_start
0002:000A5F24 s_end
0002:000A5F2C t_start
0002:000A5F30 t_end
0002:000A5F38 SwitchTextureData
0002:000A5F40 frontsector
0002:000A5F48 occlusion
0002:000A6088 rSubsector_tail
0002:000A6488 rSubsector_head
0002:000A648C renderview
0002:000A6490 viewx
0002:000A6494 viewy
0002:000A6498 viewz
0002:000A649C viewangle
0002:000A64A0 viewcos
0002:000A64A4 viewsin
0002:000A64A8 bOcclude
0002:000A64AC rThingCount
0002:000A64B0 thing_drawcount
0002:000A64B8 rFrustumMatrix
0002:000A6500 validcount
0002:000A6508 vissprite_list
0002:000A7D08 vissprite
0002:000A7D0C infraredfactor
0002:000A7D10 r_flashcolor
0002:000A7D14 quakeviewy
0002:000A7D18 quakeviewx
0002:000A7D1C camerapitch
0002:000A7D20 fognear
0002:000A7D24 fogcolor
0002:000A7D2C scrollfrac
0002:000A7D30 skyfunc
0002:000A7D48 cloudlump
0002:000A7D4C skyTitleOn
0002:000A7D5C thundertic
0002:000A7D64 skylump
0002:000A7D68 backdroplump
0002:000A7D6C cloudcolor
0002:000A7D70 flatskycolor
0002:000A7D74 skyflags
0002:000A7DF8 arrSFont
0002:000A7DFC arrStatus
0002:000A7E04 arrSymbols
0002:000A7E10 finesine
0002:000B1E10 creditscreenShowStage
0002:000B1E14 creditscreenAlpha
0002:000B1E18 creditscreenTextAlpha
0002:000B1E1C creditscreenStage
0002:000B1E20 lumpinfo
0002:000B1E24 iNumLumps
0002:000B1E28 lmpLumpCache
0002:000B1E30 iNumMapLumps
0002:000B1E34 lmpMapLumpDir

0002:000B1E38	arrMapLumpData
0002:000B1E40	mainzone
0002:000B3CE0	ALVoice
0002:000B3D40	ALPlayerClient
0002:000B3D54	ALPlayerClientData
0002:000B3D60	SN64FILE
0002:000B3DCC	sn64SoundInfoPointer
0002:000B3E18	iStatusVariable
0002:000B6230	cur_timediv1
0002:000B6234	cur_timediv2
0002:000B62E0	sequence_struct
0002:000B62EC	sequence_event
0002:000B630C	seq_patches
0002:000B6310	seq_subpatches
0002:000B6580	voice_volume
0002:000B6584	voice_panning
0002:000B65C0	seq_forcednote
0002:000B65D0	seq_pitch
0002:000B65EC	seq_globalvol
0002:000B65FE	seq_panning
0002:000B663C	cur_patchoffset
0002:000B6640	seq_note
0002:000B6641	note_velocity
0002:000B6642	cur_patchlen
0002:000B6644	cur_patch
0002:000B6648	cur_subpatch
0002:000B664C	cur_sfx
0002:000B6684	SSEQ_Address
0002:000B6688	NUMSFX
0002:000B9EF0	zoneheap
0002:0030E158	bufferArray01
0002:00325400	audioheap
0002:0062E658	aS_start
0002:006322B8	aS_end
0002:006322C8	aT_start_0
0002:00634248	aT_end_0
0002:006345B8	aEndofwadsn64
0002:0063FFA0	aSseq

18 – Conclusion

This concludes the Doom 64 Tech Bible. There are many more things that need to be covered as well as the majority of the sound system that yet needs researching but should provide enough information to understand the changes and improvements in Doom 64. I've spent several years on and off learning the mechanics and reverse engineering the game. I hope that this document is proven useful to anyone who wishes to add Doom 64 support to their source port or just simply wants to learn about the tech. Please provide credit where possible in your source port if you do add support and insure the user, who is playing the source port, knows this as well.

Any further questions, concerns, or contribution requests please contact me at svkaiser@gmail.com.