LBA classic engine doc

[2.21]

Jul 31, 2023

LBA1

This documentation aim to explain how the Little Big Adventure engines work.

Please **Little Big Adventure** game assets (art, models, textures, audio, etc.) are not open-source and therefore aren't redistributable.

CHAPTER

ONE

GENERAL INFORMATION

This documentation is hosted by Read the docs and built with Sphinx. You can pull the project and edit locally.

The files from the engines are encoded in OEM-852 or Code page 852 : https://en.wikipedia.org/wiki/Code_page_852

CHAPTER

TWO

OTHER VERY USEFUL RESOURCES

LBA Community Wiki : http://lbafileinfo.kaziq.net/index.php/Main_Page

CHAPTER

THREE

LBA1 ENGINE

3.1 Compile

3.1.1 Prerequisites

- DOSBox - DOS emulator which we will use to compile the game inside.
- 4DOS - Command line interpreter, which supports the `copy` command with binary inputs and output.
- Watcom 10 compiler - For compiling C sources and running MAKEFILEs
- MASM (Microsoft Macro Assembler) 6.0 - For compiling ASM sources

3.1.2 Getting prerequisites and sources

DOSBox and 4DOS are freely available. For getting Watcom 10 and MASM 6.0, you need to search the internet. Note that we did not manage to build the game with Open Watcom. Also, for some reason the MASM version 6.11 compiler did run very slowly in the DOSBox, so it was basically unusable. We had to use the version 6.0.

All directories and files will placed in the `~/lba-hacking` directory on the host machine. Feel free to change this path, but then adjust the DOSBox configuration below correspondingly. This directory will be mounted to `C:` in DOSBox.

- Extract 4DOS into `4dos`.
- Extract Watcom and MASM installers into `install`. These will be needed to be installed.
- Clone https://github.com/2point21/lba1-classic-community into `lba`.

The dir structure at this point should like something like this:

```
~/lba-hacking
├── 4dos
├── install
│   ├── masm
│   └── watcom
└── lba
```

3.1.3 DOSBox configuration

Change the `autoexec` section of you DOSBox configuration like below. The configuration path of DOSBox is usually shown when you start it.

```
[autoexec]
mount C ~/lba-hacking

PATH c:\watcom\binw;c:\masm\bin;%PATH%
set INCLUDE=c:\watcom\h;c:\lba\lib386
set WATCOM=c:\watcom
set EDPATH=c:\watcom\eddat
set WIPFC=c:\watcom\wipfc

C:
C:\4DOS\4DOS.COM
```

3.1.4 Install tools

- Launch DOSBox (e.g. with `dosbox`).
- On the first run, 4DOS will prompt some configuration values.
- Install Watcom by running `C:\INSTALL\WATCOM\SETUP.EXE` and following the instructions. Leave the default installation path `C:\WATCOM`. The step which proposes to modify `AUTOEXEC.EXE` and `CONFIG.SYS` can be skipped.
- Install MASM by running `C:\INSTALL\MASM\DISK1\SETUP.EXE`. Leave the default installation paths `C:\MASM\BINB`, etc…

Check the installation by typing in:

- `wmake`: this should show the installed Watcom make version; in my case 10.5
- `wcc386`: this should show the help of the Watcom C compiler; in my case 10.5
- `ml`: this should show the version of the Microsoft Macro Assembler; in my case 6.00

Now we are ready to build the game.

3.1.5 Build

Run inside the DOSBox

```
cd C:\LBA\LIB386

cd LIB_3D
wmake

cd ..\LIB_MENU
wmake

cd ..\LIB_MIDI
wmake
```

(continues on next page)

(continued from previous page)

```
cd ..\LIB_MIX
wmake

cd ..\LIB_SAMP
wmake

cd ..\LIB_SVGA
wmake

cd ..\LIB_SYS
wmake

cd ..\..\SOURCES
wmake
link
```

The last command will link the `LBA0.exe`.

3.1.6 Run

To run the game, you will need some assets of the original game.

- copy HQR files,
- copy `M_SB16.DLL`, `S3.DLL`, and `W_SB16.DLL`,
- copy `LBA.CFG`,

into the directory containing `LBA0.exe`, in our case `C:\LBA\SOURCES`.

Run

```
dos4gw LBA0.exe
```

Enjoy!

3.2 Compile

3.2.1 Prerequisites

- Open Watcom v2 - C/C++ Compiler capable of building DOS applications
- *MASM (Microsoft Macro Assembler)* - For compiling assembler files

3.2.2 Getting prerequisites and sources

The prerequisites are freely available, MASM as part of Visual Studio Community (Tested with versions 2019 and 2022). Both can be installed at their default locations.

For Open Watcom, be sure to select full instalation and to modify environment variables later.

To get the sources, clone the lba1-classic-community repository into some folder.

```
git clone https://github.com/2point21/lba1-classic-community.git
```

3.2.3 Environment configuration

Create or edit the file `SETENV.BAT` on the `lba1-classic-community` repository folder, with the following content, making sure to double check if the Microsoft Visual Studio Community and Open Watcom folders are the same on your system.

```
@echo off
echo LBA Build Environment
call "C:\Program Files\Microsoft Visual Studio\2022\Community\VC\Auxiliary\Build\
↪vcvarsamd64_x86.bat"
call C:\WATCOM\owsetenv.bat
SET LIB386_PATH=%CD%\LIB386
SET INCLUDE=%LIB386_PATH%;%INCLUDE%
```

3.2.4 Build

In a Windows command prompt inside the `lba1-classic-community` repository folder, run

```
cd LIB386\LIB_3D
wmake

cd ..\LIB_CD
wmake

cd ..\LIB_MENU
wmake

cd ..\LIB_MIDI
wmake

cd ..\LIB_MIX
wmake

cd ..\LIB_SAMP
wmake

cd ..\LIB_SVGA
wmake

cd ..\LIB_SYS
wmake
```

(continues on next page)

(continued from previous page)

```
cd ..\..\SOURCES
wmake
link
```

The expected output is the `LBA0.exe` executable inside the SOURCES folder.

3.2.5 Run

To run the game, you will need the original assets of the game and the `LBA0.exe` generated executable.

- copy game assets,
- copy `LBA0.exe`,

into the same directory. The compiled file was verified to run with DOSBox Staging.

3.2.6 Troubleshooting

Q: When I execute LBA0.exe, an error appears: "SVGA card BIOS does not support VESA extensions. Please refer to your SVGA card documentation for installing VESA driver". What can I do?

A: To solve this, change the SvgaDriver configuration in LBA.CFG to:

```
SvgaDriver: TSENG.DLL
```

Where TSENG.DLL is set instead of S3.DLL. If the issue persists, other drivers may be used (check LBA.CFG to see which are available in the game assets). As of date, this was tested using Tseng.

Additionally, change the type of machine DOSBox tries to emulate. In the DOSBox configuration file, set the machine value to:

```
[dosbox]
machine=svga_et4000
```

This will change the emulation of DOSBox to Tseng Labs ET4000. If you choose to use another SVGA driver, change the machine value accordingly (check the DOSBox configuration file to see the available options).

3.3 Audio

CHAPTER

FOUR

LBA2 ENGINE

4.1 Compile

4.2 Audio

4.3 Scripts

4.3.1 Notation

Opcode fields:

char[] Embedded C-style (NUL-terminated) string.

cond One or more opcodes specifying a condition.

i16 16-bit signed value (little-endian) used for opcode arguments.

int_or_string Any of *char[]*, *i8*, *u8*, *i16*. This is used in conditions, where the type is determined by the type of the value that is being compared.

pc16 16-bit signed offset (little-endian) used as a jump destination, absolute.

pcrel16 16-bit signed offset (little-endian) used as a jump destination, relative to the current opcode.

u8 8-bit unsigned value used for opcode arguments.

u16 16-bit unsigned value (little-endian) used for opcode arguments.

u32 32-bit unsigned value (little-endian) used for opcode arguments.

4.3.2 Life scripts

Life scripts are broken down into "behaviours" ("comportement" in the source). Each time an actor's life script is executed, it executes the same behaviour as when it last exited (or the first behaviour if it is the first time running). This way, each behaviour acts as a mini AI loop for the actor, with each tailored to a particular situation (e.g. idling, with Twinsen nearby, in combat, interacting with an object, etc).

Life script operations

In the following table, you can see that there are a number of opcodes that have the same behaviour but different names. This is useful when compiling or decompiling the scripts as there is a 1:1 correspondence between the written script and the compiled bytecode.

Opcode (hex)	Name/syntax	Description
0x00	END	Marks the end of this script.
0x01	NOP	Does nothing.
0x02	SNIF cond pcrel16	Jumps always, then replaced with SWIF opcode if condition was
0x03	OFFSET pcrel16	Jumps always.
0x04	NEVERIF pcrel16	Jumps always. Used as a replacement for a ONEIF opcode.
0x0A	PALETTE u8:palette	Switches the game's palette.
0x0B	RETURN	Ends the current behaviour.
0x0C	IF cond pcrel16	Jumps if the condition is false.
0x0D	SWIF cond pcrel16	Jumps if the condition is false and then replaced with SNIF.
0x0E	ONEIF cond pcrel16	Jumps if the condition is false otherwise replaced with NEVERIF
0x0F	ELSE pcrel16	Jumps always.
0x10	ENDIF	Does nothing.
0x11	BODY u8:model	Changes the model of the actor.
0x12	BODY_OBJ u8:actor u8:model	Changes the modem of another actor.
0x13	ANIM u16:animation	Changes the animation of the actor.
0x14	ANIM_OBJ u8:actor u16:anim	Changes the animation of another actor.
0x15	SET_CAMERA u8:zone u8:flag	Enables or disables a camera zone.
0x16	CAMERA_CENTRE u8:angle_adjust	Recentres camera.
0x17	SET_TRACK i16:track	Changes this actor's move script track.
0x18	SET_TRACK_OBJ u8:actor i16:track	Changes another actor's move script track.
0x19	MESSAGE i16:index	Says a line of dialogue.
0x1A	CAN_FALL u8:fall_type	Sets whether actor can fall.
0x1B	SET_DIRMODE u8:mode	Sets this actor's movement mode.
0x1C	SET_DIRMODE_OBJ u8:actor u8:mode	Sets another actor's movement mode.
0x1D	CAMERA_FOLLOW u8:actor	Make camera follow an actor.
0x1E	SET_HERO_STANCE u8:mode	Set Twinsen's stance.
0x1F	SET_VAR_SCENE u8:var u8:value	Sets the value of a scene variable.
0x20	BEHAVIOUR u8:id	Begins a life script behaviour block.
0x21	SET_BEHAVIOUR pc16:offset	Jumps to a new behaviour block.
0x22	SET_BEHAVIOR_OBJ u8:actor pc16:off	Changes the active behaviour of another actor.
0x23	END_BEHAVIOUR	Marks the end of a life script behaviour block.
0x24	SET_VAR_GAME u8:var i16:value	Sets the value of a game variable.
0x25	KILL_OBJ u8:actor	Kills the given actor.
0x26	SUICIDE	Kills this actor.
0x27	USE_KEY	Subtracts one key from the inventory.
0x28	SUB_MONEY i16:quantity	Takes money from Twinsen.
0x29	END_LIFE	Ends life script execution for this actor.
0x2A	SAVE_CURRENT_TRACK	Saves the move script track to a hidden variable.
0x2B	RESTORE_LAST_TRACK	Restores the move script track from the hidden variable.
0x2C	MESSAGE_OBJ u8:actor i16:message	Another actor says a line of dialogue.
0x2D	INC_CHAPTER	Increment the chapter number game variable.
0x2E	FOUND_OBJECT u8:object	Display the "found object" overlay.
0x2F	SET_DOOR_LEFT i16:distance	Slides this door to the left.
0x30	SET_DOOR_RIGHT i16:distance	Slides this door to the right.

continues on n

Table 1 – continued from previous page

Opcode (hex)	Name/syntax	Description
0x31	SET_DOOR_UP i16:distance	Slides this door upwards.
0x32	SET_DOOR_DOWN i16:distance	Slides this door downwards.
0x33	GIVE_BONUS u8:remove	Gives this actor's bonus items.
0x34	CHANGE_SCENE u8:scene	Move to a different scene.
0x35	OBJ_COL u8:enabled	Enables or disables object/actor collisions for this actor.
0x36	BRICK_COL u8:collision_type	Enables or disables terrain collisions for this actor.
0x37	OR_IF cond pcrel16	Jumps if condition is true.
0x38	INVISIBLE u8:invisible	Makes the actor invisible or visible again.
0x39	SHADOW_OBJ u8:actor u8:enabled	Enables or disables the shadow for another actor.
0x3A	POS_POINT u8:point	Moves this actor to a point.
0x3B	SET_MAGIC_LEVEL u8:level	Sets Twinsen's magic level.
0x3C	SUB_MANA u8:quantity	Drains some of Twinsen's mana.
0x3D	SET_HEALTH_OBJ u8:actor u8:value	Sets the health of an actor.
0x3E	SUB_HEALTH_OBJ u8:actor u8:points	Subtracts health from another actor.
0x3F	HIT u8:victim u8:damage	Deals damage to another actor, caused by this actor.
0x40	PLAY_VIDEO char[]:name	Plays the named cutscene video.
0x41	LIGHTNING u8:duration	Display a lightning flash.
0x42	INC_CLOVER_BOX	Gives Twinsen another clover box.
0x43	SET_USED_INVENTORY u8:item	Use inventory item.
0x44	ADD_CHOICE i16:message	Adds choice to the next ask.
0x45	ASK_CHOICE i16:message	Says a line of dialogue and offers choices.
0x46	INIT_BUGGY u8:flag	Sets up Twinsen's car.
0x47	MEMO_SLATE u8:picture	Adds a picture to the memo slate.
0x48	SET_HOLO_POS u8:marker	Adds a marker to the holomap.
0x49	CLR_HOLO_POS u8:marker	Removes a marker from the holomap.
0x4A	ADD_FUEL u8:ignored	Does nothing (LBA1 leftover).
0x4B	SUB_FUEL u8:ignored	Does nothing (LBA1 leftover).
0x4C	SET_FRAGMENT u8:zone u8:enable	Enables or disables a terrain chunk.
0x4D	SET_TELEPORT_ZONE u8:zone u8:flag	Enables or disables a teleport zone.
0x4E	MESSAGE_ZOE i16:message	Says a line using Zoe's voice.
0x4F	FULL_POINT	Restores Twinsen's health, mana and healing horn.
0x50	BETA i16:angle	Rotates actor.
0x51	FADE_TO_PAL u8:palette	Fades to the given palette.
0x52	ACTION	Triggers Twinsen's action (like pressing the Z key).
0x53	SET_FRAME u8:frame	Changes the frame number of this actor's animation.
0x54	SET_SPRITE u8:sprite	Changes the sprite used for this actor.
0x55	SET_FRAME_3DS u8:frame	Changes the frame number of this actor's animated sprite.
0x56	IMPACT_OBJ u8:actor i16:anim i16:yoffset	Plays an impact animation above an actor.
0x57	IMPACT_POINT u8:point i16:anim	Plays an impact animation at a point.
0x58	ADD_MESSAGE i16:message	Same as MESSAGE.
0x59	BALLOON u8:enable	Enables or disables use of speech balloons.
0x5A	NO_HIT u8:enable	Enables or disables ignoring hits/damage to this actor.
0x5B	ASK_CHOICE u8:actor i16:message	Another actor says a line of dialogue and offers choices.
0x5C	CINEMA_MODE u8:enable	Enables or disables cutscene mode.
0x5D	SAVE_HERO	Saves Twinsen's stance to a hidden variable.
0x5E	RESTORE_HERO	Restores Twinsen's stance from a hidden variable.
0x5F	ANIM_SET u16:anim	Sets this actor's animation.
0x60	RAIN u8:duration	Makes it rain.
0x61	GAME_OVER	Kills Twinsen and ends the game.

continues on n

Table 1 – continued from previous page

Opcode (hex)	Name/syntax	Description
0x62	THE_END	Ends the game and shows the credits.
0x63	SET_CONVEYOR_ZONE u8:zone u8:flag	Enables or disables a conveyor zone.
0x64	PLAY_MUSIC u8:track	Plays a music track.
0x65	SAVE_TRACK_TO_GAME_VAR u8:var	Saves this actor's move script track to a game variable.
0x66	SET_TRACK_FROM_GAME_VAR u8:var	Sets this actor's move script track from a game variable.
0x67	ANIM_TEXTURE u8:enable	Enable or disable texture animation.
0x68	ADD_MESSAGE_OBJ u8:actor i16:msg	Same as MESSAGE_OBJ.
0x69	BRUTAL_EXIT	Ends the game without displaying the credits.
0x6A	COMMENT	Does nothing.
0x6B	SET_LADDER_ZONE u8:zone u8:enable	Enables or disables a ladder zone.
0x6C	SET_ARMOUR u8:armour	Sets this actor's armour value.
0x6D	SET_ARMOR_OBJ u8:actor u8:obj	Sets the armour value of another actor.
0x6E	ADD_HEALTH_OBJ u8:actor u8:life	Adds health to another actor.
0x6F	STATE_INVENTORY u8:item u8:state	Changes the state/variant of an inventory object.
0x70	AND_IF cond pcrel16	Jumps if condition is false.
0x71	SWITCH	Begins a switch statement.
0x72	OR_CASE pcrel16 cond	Jumps if condition fails.
0x73	CASE pcrel16 cond	Jumps if condition succeeds.
0x74	DEFAULT	Does nothing.
0x75	BREAK pcrel16	Jumps to offset.
0x76	END_SWITCH	Does nothing.
0x77	SET_SPIKE_ZONE u8:zone u8:damage	Enables or disables a spike/trap zone.
0x78	SAVE_BEHAVIOUR	Saves this actor's behaviour index to a hidden variable.
0x79	RESTORE_BEHAVIOUR	Restores this actor's behaviour from the hidden variable.
0x7A	SAMPLE i16:sample	Plays a sound sample coming from this actor.
0x7B	SAMPLE_RND i16:sample	Like SAMPLE but randomly alters the sample's frequency.
0x7C	SAMPLE_ALWAYS i16:sample	Like SAMPLE but plays the sample continuously.
0x7D	SAMPLE_STOP i16:sample	Stops the given sample if it is playing from this actor.
0x7E	REPEAT_SAMPLE i16:sample u8:count	Like SAMPLE but plays the given number of repeats.
0x7F	BACKGROUND u8:flag	Sets or clears the "background" (don't redraw) flag for this actor.
0x80	ADD_VAR_GAME u8:var i16:value	Adds a value to a game variable.
0x81	SUB_VAR_GAME u8:var i16:value	Subtracts a value from a game variable.
0x82	ADD_VAR_SCENE u8:var u8:value	Adds a value to a scene variable.
0x83	SUB_VAR_SCENE u8:var u8:value	Subtracts a value from a scene variable.
0x84	NOP	Does nothing.
0x85	SET_RAIL_ZONE u8:zone u8:enable	Enables or disables a rail zone.
0x86	INVERSE_BETA	Rotates the actor to face the opposite direction.
0x87	NO_BODY	Hides the model for this actor.
0x88	ADD_MONEY i16:quantity	Gives money to Twinsen.
0x89	SAVE_CURRENT_TRACK_OBJ u8:actor	Saves the move script track of another actor to a hidden variable.
0x8A	RESTORE_LAST_TRACK_OBJ u8:actor	Restores the move script track of another actor from the hidden va
0x8B	SAVE_BEHAVIOUR_OBJ u8:actor	Saves the life script behaviour of another actor to a hidden variabl
0x8C	RESTORE_BEHAVIOUR_OBJ u8:actor	Restores the life script behaviour of another actor from the hidder
0x8D	SPY	Does nothing.
0x8E	DEBUG	Does nothing.
0x8F	DEBUG_OBJ	Does nothing.
0x90	POPCORN	Does nothing.
0x91	FLOW_POINT u8:point u8:flow	Displays a particle animation at a point.
0x92	FLOW_OBJ u8:actor u8:flow	Displays a particle animation on an actor.

continues on n

Table 1 – continued from previous page

Opcode (hex)	Name/syntax	Description
0x93	SET_ANIM_DIAL u16:anim	Sets the animation to use when talking.
0x94	PCX u8:image	Displays a still image.
0x95	END_MESSAGE	Does nothing.
0x96	END_MESSAGE_OBJ u8:ignored	Does nothing.
0x97	PARM_SAMPLE i16:freq u8:vol i16:fbase	Configures audio sample parameters.
0x98	NEW_SAMPLE i16:sample i16:f u8:v i16:fb	Plays an audio sample on this actor with custom parameters.
0x99	POS_OBJ_AROUND u8:move_actor u8:dest	Positions an actor on or near another actor.
0x9A	PCX_MESS_OBJ u8:img u8:fx u8:act i16:msg	Show a message on a still image background.

Fall types (undocumented values are invalid):

0. actor cannot fall
1. actor can fall
2. actor can fall; stops any fall in progress

Movement modes (undocumented values are invalid):

0. no movement
1. controlled by player
2. follow actor (opcode has extra param: uint8: actor to follow)
3. invalid
4. invalid
5. invalid
6. same XZ position as other actor
7. MecaPenguin movement
8. rail cart movement
9. circle a point (opcode has extra param: uint8: point index)
10. circle a point while facing it (opcode has extra param: uint8: point index)
11. same XZ position and angle as other actor
12. car movement
13. car movement under player control

Hero stances (undocumented values are invalid):

0. normal
1. athletic
2. aggressive
3. discreet
4. protopack
5. walking with Zoe
6. healing horn
7. spacesuit normal (interior)

8. jetpack
9. spacesuit athletic (interior)
10. spacesuit normal (exterior)
11. spacesuit athletic (exterior)
12. car
13. skeleton

Collision types (undocumented values are invalid):

0. can move through terrain bricks
1. blocked by terrain bricks
2. blocked by terrain bricks but can crawl through narrow passages

Buggy init types (undocumented values are invalid):

0. no init
1. init if needed
2. force init

Effects for PCX_MESS_OBJ (undocumented values are invalid):

0. no effect
1. venetian blinds effect

Life script conditions

Opcode (hex)	Name/syntax	Description
0x00	COL -> i8	Actor this actor collided with (or -1 if none).
0x01	COL_OBJ u8:actor -> i8	Actor another actor collided with (or -1 if none).
0x02	DISTANCE u8:actor -> i16	2D distance to another actor.
0x03	ZONE -> i8	Index of sceneric zone this actor is within (or -1 if none).
0x04	ZONE_OBJ u8:actor -> i8	Index of sceneric zone another actor is within (or -1 if none).
0x05	BODY -> i8	Model used for this actor.
0x06	BODY_OBJ u8:actor -> i8	Model used by another actor.
0x07	ANIM -> i16	Animation used by this actor.
0x08	ANIM_OBJ u8:actor -> i16	Animation used by another actor.
0x09	TRACK -> u8	Life script track active on this actor.
0x0A	TRACK_OBJ u8:actor -> u8	Life script track active on another actor.
0x0B	VAR_SCENE u8:var -> u8	Value of a scene variable.
0x0C	CONE_VIEW u8:actor -> i16	Distance to another actor, if they are within a 90-degree view cone.
0x0D	HIT_BY -> i8	Actor that last hit this actor.
0x0E	ACTION -> i8	Action key was pressed.
0x0F	VAR_GAME u8:var -> i16	Value of a game variable.
0x10	LIFE_POINT -> i16	Health of this actor.
0x11	LIFE_POINT_OBJ u8:actor -> i16	Health of another actor.
0x12	KEYS -> i8	Number of keys.
0x13	MONEY -> i16	Money.
0x14	HERO_STANCE -> i8	Twinsen's stance.
0x15	CHAPTER -> i8	Game chapter.

continues on next page

Table 2 – continued from previous page

Opcode (hex)	Name/syntax	Description
0x16	DISTANCE_3D u8:actor -> i16	3D distance to another actor.
0x17	MAGIC_LEVEL -> i8	Magic level.
0x18	MANA -> i8	Twinsen's mana points.
0x19	ITEM_USED u8:item -> i8	Item being used.
0x1A	CHOICE -> i16	Choice made in last dialogue.
0x1B	FUEL -> i16	Returns junk value; do not used (lba1 leftover).
0x1C	CARRY_BY -> i8	Actor carrying this actor.
0x1D	CDROM -> i8	Whether this is the CDROM build or floppy build.
0x1E	LADDER u8:zone -> i8	Whether a ladder zone is enabled.
0x1F	RND u8:max -> u8	Random number.
0x20	RAIL u8:zone -> i8	Whether a rail zone is enabled.
0x21	BETA -> i16	Current angle of this actor.
0x22	BETA_OBJ u8:actor -> i16	Current angle of another actor.
0x23	CARRY_OBJ_BY u8:actor -> i8	Actor carrying another actor.
0x24	ANGLE u8:actor -> i16	Angle from this actor to another actor.
0x25	DISTANCE_MESSAGE u8:actor -> i16	Distance from another actor, if within an angle suitable for conversation.
0x26	HIT_OBJ_BY u8:actor -> i8	Actor that last hit another actor.
0x27	REAL_ANGLE u8:actor -> i16	Angle from this actor to another, clamped.
0x28	DEMO -> i8	Whether this is the demo build.
0x29	COL_BRICK -> i8	Whether this actor collides with scenery.
0x2A	COL_BRICK_OBJ u8:actor -> i8	Whether another actor collides with scenery.
0x2B	PROCESSOR -> i8	Whether running on an old processor.
0x2C	OBJECT_DISPLAYED u8:actor -> i8	Whether this actor was drawn to the screen.
0x2D	ANGLE_OBJ u8:actor -> i16	Angle from another actor to this actor.

Opcode (hex)	Name/syntax	Description
0x00	EQUAL int_or_string	Whether the value is equal to the constant.
0x01	GREATER int_or_string	Whether the value is greater than the constant. Not valid for strings.
0x02	LESS int_or_string	Whether the value is less than the constant. Not valid for strings.
0x03	GREATER_OR_EQUAL int_or_string	Whether the value is not less than the constant. Not valid for strings.
0x04	LESS_OR_EQUAL int_or_string	Whether the value is not greater than the constant. Not valid for strings.
0x05	NOT_EQUAL int_or_string	Whether the value is not equal to the constant.

4.3.3 Move scripts

Opcode (hex)	Name/syntax	Description
0x00	END	Ends this move script.
0x01	NOP	Does nothing.
0x02	BODY u8:model	Sets this actor's model.
0x03	ANIM u16:anim	Sets this actor's current animation.
0x04	GOTO_POINT u8:point	Actor rotates to face the given point and waits until its anim
0x05	WAIT_ANIM	Waits for the current animation to end.

Table 3 – continued from previous page

Opcode (hex)	Name/syntax	Description
0x06	LOOP u8:init u8:remaining pcrel16	Decrements remaining, jumps if non-zero, sets remaining to
0x07	ANGLE i16:angle	Actor rotates to the given angle and waits until the rotation c
0x08	POS_POINT u8:point	Instantly teleports the actor to a point.
0x09	MOVE_TRACK u8:id	Begins a track block within this move script.
0x0A	GOTO pcrel16	Jumps to another part of the move script.
0x0B	STOP	Stops executing this move script.
0x0C	GOTO_POINT_BACKWARDS u8:point	Actor rotates to face away from the given point and waits un
0x0D	WAIT_NUM_ANIM u8:count u8:zero	Waits for the actor's animation to have played a number of ti
0x0E	SAMPLE i16:sample	Plays a sound sample.
0x0F	GOTO_POINT_3D u8:point	Actor moves to the given point, if it's a 3D sprite.
0x10	SPEED i16:speed	Sets the rotation speed of the actor.
0x11	BACKGROUND u8:enabled	Enables or disables the "background" flag for this actor.
0x12	WAIT_NUM_SECOND u8:count u32:zero	Wait for the number of seconds.
0x13	NO_BODY	Sets this actor to have no model.
0x14	BETA i16:angle	Rotates this actor instantly.
0x15	OPEN_LEFT i16:distance	Door slides to the left.
0x16	OPEN_RIGHT i16:distance	Door slides to the right.
0x17	OPEN_UP i16:distance	Door slides upwards.
0x18	OPEN_DOWN i16:distance	Door slides downwards.
0x19	CLOSE	Restore door's original position.
0x1A	WAIT_DOOR	Wait until door finishes moving.
0x1B	SAMPLE_RND i16:sample	Plays a sound sample with a random frequency adjustment.
0x1C	SAMPLE_ALWAYS i16:sample	Plays a sound sample forever.
0x1D	SAMPLE_STOP i16:sample	Stops a particular sound sample.
0x1E	PLAY_VIDEO char[]:name	Plays a cutscene video.
0x1F	REPEAT_SAMPLE i16:count	Sets the number of repeats for SIMPLE_SAMPLE.
0x20	SIMPLE_SAMPLE i16:sample	Plays a sample according to REPEAT_SAMPLE and resets
0x21	FACE_HERO i16:negative_one	Actor rotates to face Twinsen and waits until the rotation co
0x22	ANGLE_RND i16:angle i16:negative_one	Actor rotates to a random angle and waits until the rotation c
0x23	COMMENT	Does nothing.
0x24	WAIT_NUM_DECISECONDS u8:count u32:zero	Waits for a number of deciseconds (tenths of a second).
0x25	DO	Does nothing.
0x26	SPRITE i16:sprite	Sets this actor's sprite.
0x27	WAIT_NUM_SECOND_RND u8:max u32:zero	Waits for a random number of seconds, up to a maximum.
0x28	AFF_TIMER	Does nothing.
0x29	SET_FRAME u8:frame	Sets the actor's animation frame.
0x2A	SET_FRAME_3DS u8:frame	Sets the actor's 3D sprite animation frame.
0x2B	SET_START_3DS u8:frame	Sets the start frame of the actor's 3D sprite animation.
0x2C	SET_END_3DS u8:frame	Sets the end frame of the actor's 3D sprite animation.
0x2D	START_ANIM_3DS u8:fps	Starts the actor's 3D sprite animation.
0x2E	STOP_ANIM_3DS	Stops the actor's 3D sprite animation.
0x2F	WAIT_ANIM_3DS	Waits until the actor's 3D sprite animation ends or is stoppe
0x30	WAIT_FRAME_3DS u8:frame	Waits until the actor's 3D sprite animation reaches the given
0x31	WAIT_NUM_DECISECONDS_RND u8:max u32:0	Waits for a random number of deciseconds, up to a maximu
0x32	INTERVAL int16:interval	Sets the interval between sample repeats.
0x33	FREQUENCY i16:frequency	Sets the frequency for sample playback.
0x34	VOLUME u8:volume	Sets the volume for sample playback.

4.4 Zones

Zones are used to demarcate 3D regions of space within scenes. There are various types of zones, each of which have different behaviours when Twinsen enters or interacts with them:

- *teleport zones* transport Twinsen to a different scene when entered
- *camera zones* change the camera position and angle when entered
- *sceneric zones* define areas of the scene used for "is actor in zone?" queries in scripts
- *fragment zones* define an area of the scene's terrain that can be dynamically shown or hidden
- *bonus zones* dispense items when interacted with
- *text zones* are used to implement signs and say dialogue when interacted with
- *ladder zones* provide vertical movement
- *conveyor zones* move actors which stand within them
- *spike zones* deal damaged when entered (floor spikes, traps, etc)
- *rail zones* are used to control minecart movement

All zones are cuboid in shape.

4.4.1 Zone format

Zones are stored as part of the scene containing the zone:

Listing 1: zone data layout

```
{
    i32         x0
    i32         y0
    i32         z0
    i32         x1
    i32         y1
    i32         z1
    i32         info0
    i32         info1
    i32         info2
    i32         info3
    i32         info4
    i32         info5
    i32         info6
    i32         info7
    i16         type
    i16         value
}
```

- *x0*, *y0*, *z0*: first corner defining the zone cuboid
- *x1*, *y1*, *z1*: opposite corner defining the zone cuboid
- *info0..7*: zone parameters; interpretation depends on the zone type
- *type*: the type of zone

- *value*: zone parameter; interpretation depends on the zone type

Zone type	Name
0	Teleport
1	Camera
2	Sceneric
3	Fragment
4	Bonus
5	Text
6	Ladder
7	Conveyor
8	Spike
9	Rail

The documentation below for each of the zone types describes how the flags are interpreted, as loaded from the scene. The LBA engine modifies some of these values at run-time in order to avoid allocating additional memory; these run-time uses and modifications are not documented here.

Teleport zones

Parameter	Description
param	Destination scene
info0	Destination x
info1	Destination y
info2	Destination z
info3	Destination angle
info4	Zone scripting ID
info5	Door flags: bit 0 - for exterior scenes, don't activate zone until Twinsen collides with the door
info6	Collision flags: bit 0 - don't adjust Twinsen to fix collisions
info7	Enable flags: bit 0 - zone is enabled

Teleport zones can be enabled or disabled from script by using the SET_TELEPORT_ZONE opcode.

Camera zones

Parameter	Description
param	Zone scripting ID
info0	Camera x
info1	Camera y
info2	Camera z
info3	Camera alpha angle
info4	Camera beta angle
info5	Camera gamma angle
info6	View distance
info7	Enable flags: bit 0 - zone is enabled

Camera zones can be enabled or disabled from script by using the SET_CAMERA opcode.

Sceneric zones

Parameter	Description
param	Zone scripting ID
info0	-unused-
info1	-unused-
info2	-unused-
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-

Sceneric zones can be used from scripts by checking whether an actor is within them by using the ZONE and ZONE_OBJ conditions.

Fragment zones

Parameter	Description
param	Zone scripting ID
info0	Fragment number
info1	-unused-
info2	Enable flags: bit 0 - zone is enabled
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-

Fragment zones can be enabled or disabled from script by using the SET_FRAGMENT opcode.

Bonus zones

Parameter	Description
param	-unused-
info0	Bonus type
info1	Bonus quantity
info2	-unused-
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-

Bonus zones are not scriptable.

Text zones

Parameter	Description
param	Message ID
info0	Text colour
info1	Associated camera zone (zero for none)
info2	Direction
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-

Text zones are not scriptable.

The direction values seem appropriate for use as a bitmask but the LBA engine checks for equality, not a bit test, so each text zone can only face a single direction. Double-sided signs would require two sign zones, one on each side.

Zone direction	Description
1	Sign faces North
2	Sign faces South
4	Sign faces East
8	Sign faces West

Ladder zones

Parameter	Description
param	Zone scripting ID
info0	Enabled
info1	-unused-
info2	-unused-
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-

Ladder zones can be enabled or disabled from script using the SET_LADDER_ZONE opcode. Their enabled state can be queried using the LADDER condition.

Conveyor zones

Parameter	Description
param	Zone scripting ID
info0	-unused-
info1	Enabled
info2	Direction
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-

Conveyor zones can be enabled or disabled from script using the SET_CONVEYOR_ZONE opcode.

The direction values seem appropriate for use as a bitmask but the LBA engine checks for equality, not a bit test.

Zone direction	Description
1	Conveyor travels North
2	Conveyor travels South
4	Conveyor travels East
8	Conveyor travels West

Spike zones

Parameter	Description
param	Zone scripting ID
info0	-unused-
info1	Damage
info2	Rearm time
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-

The damage of spike zones can be controlled from script using the SET_SPIKE_ZONE opcode. Setting the damage to zero will disable the spike zone; setting it to a non-zero value will enable it.

Rail zones

Parameter	Description
param	Zone scripting ID
info0	Enabled
info1	Switch set
info2	-unused-
info3	-unused-
info4	-unused-
info5	-unused-
info6	-unused-
info7	-unused-