

Chapter 23 - Music from BASIC and from each CPU

Part III has covered the audio engines one at a time. This chapter is the working map: how to start music from BASIC, how to inspect the media loader, and where each CPU reaches the common sound chips.

The rule is simple. BASIC is best for quick setup, explicit-width POKE, and high-level playback. Machine code writes the same MMIO registers and ports documented in the chip chapters.

23.1 First mixed BASIC sound

Type this program. It mixes four different engines without loading any file.

```
10 REM BASIC MIXER SKETCH
20 POKE32 &H000F0800,1
30 REM SOUNDCHIP VOICE
40 SOUND 0,440,160,2
50 ENVELOPE 0,10,20,128,30
60 GATE 0,ON
70 REM SEVERAL BUS SOUND CHIPS
80 PSG 0,500,15
90 SID VOICE 1,1000,2048,65,136,240
100 SID VOLUME 12
110 POKEY 1,100,168
120 TED TONE 1,440
130 POKE8 &H000F0F03,&H18
140 REM COMMON MIXER REVERB
150 SOUND REVERB 120,160
160 FOR T=1 TO 3000
170 NEXT T
180 GATE 0,OFF
```

You should hear a bright layered tone: SoundChip sine, PSG square, SID voice, POKEY tone, TED tone, and a little mixer reverb. This is the BASIC path for short cues, test tones, and quick sound design.

Lines 40 to 60 set up a flexible SoundChip voice. Lines 80 to 130 add four other chips on the same machine bus. The TED line uses POKE8 because TED TONE sets frequency but does not enable the voice. Line 150 changes the shared mixer reverb, so every active engine is heard through the same output path.

23.2 SOUND PLAY

SOUND PLAY starts a music file from machine-visible storage. Type the filename and, when the format has subsongs, an optional subsong number:

```

10 REM PLAY BY FILENAME EXTENSION
20 SOUND PLAY "TITLE.MOD"
30 SOUND PLAY "INTRO.SID",0
40 SOUND PLAY "TUNE.AHX",1
50 SOUND PLAY "SONG.MID"
60 SOUND PLAY "SONG.MUS"
70 SOUND PLAY STOP

```

SOUND STOP is an alias of SOUND PLAY STOP. It stops media playback: SID, PSG, TED, AHX, POKEY, MOD, WAV, and MIDI/MUS players. It does not silence raw chip registers that you have written directly; turn those down with their own volume or gate controls.

The media loader chooses the engine from the filename extension.

Extension	Engine	Chapter
.sid	SID family	15
.ym, .ay, .sndh, .vtx, .vt, .pt3, .pt2, .pt1, .stc, .sqt, .asc, .ftc, .vgm, .vgz, .snd	PSG / AY	13
.ted, .prg	TED audio	16
.ahx	AHX	18
.sap	POKEY	17
.mod	MOD	19
.wav	WAV	20
.mid, .midi, .mus	MIDI/MUS and RawlandMini	21

If a start fails while BASIC is waiting for the loader, BASIC raises a PLAY error. If a player later reports an error, the media status register changes to error.

23.3 Media loader registers

BASIC normally hides these registers, but they are useful when you want to inspect loader state or start playback with raw POKE32.

Address	Name	Access	Purpose
\$F2300	MEDIA_NAME_PTR	write/read	Address of a zero-terminated filename string.
\$F2304	MEDIA_SUBSONG	write/read	Optional subsong number.
\$F2308	MEDIA_CTRL	write	1 play, 2 stop.
\$F230C	MEDIA_STATUS	read	Loader and player state.
\$F2310	MEDIA_TYPE	read	Engine selected by the filename.
\$F2314	MEDIA_ERROR	read	Last loader error.

Status values:

Value	Meaning
0	Idle.
1	Loading.
2	Playing.
3	Error.

Type values:

Value	Engine
0	None.
1	SID.
2	PSG.
3	TED.
4	AHX.
5	POKEY.
6	MOD.
7	WAV.
8	MIDI/MUS.

Error values:

Value	Meaning
0	No error.
1	File not found.
2	Bad format or read failure.
3	Unsupported extension.
4	Invalid filename.
5	File too large for the staging buffer.

This raw loader example writes the filename string itself:

```

10 REM RAW MEDIA LOADER START
20 A=&H00090000
30 REM ZERO TERMINATED FILENAME
40 POKE8 A+0,84
50 POKE8 A+1,73
60 POKE8 A+2,84
70 POKE8 A+3,76
80 POKE8 A+4,69
90 POKE8 A+5,46
100 POKE8 A+6,77
110 POKE8 A+7,79
120 POKE8 A+8,68
130 POKE8 A+9,0
140 REM POINTER, SUBSONG, PLAY COMMAND
150 POKE32 &H000F2300,A
160 POKE32 &H000F2304,0
170 POKE32 &H000F2308,1
180 PRINT PEEK32(&H000F230C),PEEK32(&H000F2310),PEEK32(&H000F2314)

```

Use filenames that belong to the machine's own storage area. Names with . . are rejected. AHX and TED use a 64 KB staging buffer. MOD, WAV, and MIDI/MUS can be larger because they load directly into their players.

Lines 40 to 130 build the zero-terminated filename in guest memory. Lines 150 to 170 point the media loader at that string, choose subsong 0, and start. Line 180 prints status, selected engine type, and the last error code.

23.4 BASIC verbs by engine

Engine	BASIC path	Chapter
SoundChip and SFX	SOUND, ENVELOPE, GATE, SFX POKE32	12
PSG / AY	PSG, PSG PLAY, PSG STOP, PSG STATUS, POKE8	13
SN76489	POKE8 byte-stream writes	14
SID family	SID VOICE, SID FILTER, SID VOLUME, SID PLAY, SID STOP, SID STATUS	15
TED audio	TED TONE, TED VOL, TED NOISE, TED PLAY, TED STOP, TED STATUS	16
POKEY	POKEY, POKEY CTRL, POKEY PLUS, SAP PLAY, SAP STOP, POKEY STATUS	17
AHX	AHX PLAY, AHX STOP, AHX PLUS, AHX STATUS	18
MOD	SOUND MOD PLAY, SOUND MOD STOP, SOUND MOD FILTER, MOD STATUS	19
WAV	SOUND PLAY or raw WAV register POKE32	20
MIDI/MUS	SOUND PLAY or raw MIDI player register POKE32	21
Live MIDI	MIDI NOTE, MIDI PROG, MIDI CTRL, MIDI SEND, MIDI RESET, or raw live MIDI POKE8	21
Paula DMA	Raw Paula register POKE32	22

Mixer-wide effects are SOUND FILTER, SOUND REVERB, and SOUND OVERDRIVE. See Chapter 11.

23.5 Full-address CPU map

IE64, IE32, M68K, and x86 can write the full sound MMIO addresses.

Engine	Address
AUDIO_CTRL	\$F0800
SoundChip channels	\$F0A80, \$F0AC0, \$F0B00, \$F0B40, then \$F0C40, \$F0C80, \$F0CC0, \$F0D40, \$F0D80, \$F0DC0
SFX channels	\$F2600-\$F29FF extended window; \$F0E80-\$F0EFF legacy aliases for channels 0 to 3
PSG	\$F0C00-\$F0C0F
PSG player	\$F0C10-\$F0C1F
SN76489	\$F0C30-\$F0C32
POKEY	\$F0D00-\$F0D0A
POKEY player	\$F0D10-\$F0D1F
SID, SID2, SID3	\$F0E00, \$F0E30, \$F0E50
SID player	\$F0E20-\$F0E2D
TED audio	\$F0F00-\$F0F05
TED player	\$F0F10-\$F0F1F
AHX	\$F0B80-\$F0B91
MIDI/MUS	\$F0BA0-\$F0BBF
Live MIDI	\$F0BF4-\$F0BF6
MOD	\$F0BC0-\$F0BD7
WAV	\$F0BD8-\$F0BF3
Paula DMA	\$F2260-\$F22AF
Media loader	\$F2300-\$F231F

The machine-code chapters in Part IV show byte-entered examples for the individual CPUs. Those examples use the monitor, not an assembler.

23.6 6502 audio map

The 6502 uses compact 16-bit ranges for the heritage chips.

Engine	6502 address
POKEY	\$D200-\$D20A
PSG	\$D400-\$D40F
SID family	\$D500-\$D55F
TED audio	\$D600-\$D605
VGA helper registers	\$D700-\$D70D
MIDI/MUS player	\$FBA0-\$FBBF
Live MIDI port	\$FBF4-\$FBF6

The SID family window is contiguous: SID starts at \$D500, SID2 at \$D520, and SID3 at \$D540.

23.7 Z80 port map

The Z80 uses select/data ports for PSG, SID, POKEY, and TED.

Engine	Select	Data
PSG / AY	\$F0	\$F1
SID family	\$E0	\$E1
POKEY	\$D0	\$D1
TED audio	\$F2	\$F3

To write a selected register, output the register number to the select port, then output the value to the data port.

SN76489 has a byte-stream port instead:

Port	Use
\$E4	Write SN76489 command/data byte; read back last written byte.
\$E5	Read ready status, bit 0.

The Z80 can also reach the MIDI/MUS player at \$FBA0 - \$FBBF and the live MIDI port at \$FBF4 - \$FBF6 through the memory MMIO mirror.

23.8 x86 port map

x86 can use the full-address MMIO map, and it also has chip-style ports:

Engine	Port
PSG select/data	\$F0 / \$F1
SID select/data	\$E0 / \$E1
POKEY direct window	\$60 - \$69
TED select/data	\$F2 / \$F3

23.9 Choosing an engine

- For general original music and effects, start with SoundChip.
- For ZX Spectrum 128, Amstrad CPC, MSX, and Atari ST chip-tunes, use PSG or SOUND_PLAY on a PSG-supported file.
- For Sega Master System style tones, use SN76489.
- For C64 music, use SID or SOUND_PLAY on a SID file.
- For C16 and Plus/4 music, use TED audio or SOUND_PLAY on a TED file.
- For Atari 8-bit music, use POKEY or SOUND_PLAY on a SAP file.
- For AHX music, use AHX.
- For ProTracker music, use MOD.
- For Standard MIDI Files or MUS files, use MIDI/MUS.
- For immediate GM-style note events, use live MIDI.
- For sampled speech and recorded effects, use WAV.
- For raw sample streaming and double-buffering, use Paula DMA.

Engines can be mixed. Keep track of shared DAC channels and player ownership when MOD, WAV, Paula, or manual DAC output are active at the same time. MIDI/MUS uses its own player mixer input, but it still passes through the shared global effects from Chapter 11.