

## Chapter 13 - PSG and AY-3-8910

---

The PSG is the General Instrument AY-3-8910 and Yamaha YM2149 style sound chip: three square-wave tone channels, one shared noise generator, one shared envelope generator, and two register latches that follow the original I/O port positions. It is the right chip for bright arpeggios, simple bass lines, noise drums, and register-frame music playback.

Turn the mixer on first, then give a channel a tone divider and a level:

```
10 REM PSG FIRST TONE
20 POKE32 &H000F0800,1
30 PSG MIXER &H3E
40 PSG 0,284,15
```

Line 20 enables the global audio mixer. Line 30 enables tone on channel A and disables the other tone and noise routes. Line 40 writes channel A divider 284 and level 15, which gives a clear middle-A style tone with the default PSG clock.

Try changing the divider in line 40 to 225 or 190. Smaller dividers give higher notes.

### 13.1 Shape of the chip

Item	Value
Tone channels	3, named A, B, and C
Tone waveform	Square wave
Noise generator	1, shared through the mixer
Envelope	1, shared by any channel using it
Channel level	0 to 15, or envelope controlled
Register count	16, numbered 0 to 15
Access width	Byte registers

The PSG register block is byte-wide. Use `POKE8` when you are programming the raw registers directly. A `POKE32` to the PSG register window can update more than one adjacent byte register.

### 13.2 Register block

The PSG registers occupy consecutive byte addresses from `$F0C00` to `$F0C0F`.

Address	Reg	Purpose
<code>\$F0C00</code>	0	Channel A divider low byte
<code>\$F0C01</code>	1	Channel A divider high nibble
<code>\$F0C02</code>	2	Channel B divider low byte
<code>\$F0C03</code>	3	Channel B divider high nibble
<code>\$F0C04</code>	4	Channel C divider low byte

Address	Reg	Purpose
\$F0C05	5	Channel C divider high nibble
\$F0C06	6	Noise period, low 5 bits
\$F0C07	7	Mixer and I/O direction
\$F0C08	8	Channel A level
\$F0C09	9	Channel B level
\$F0C0A	10	Channel C level
\$F0C0B	11	Envelope period low byte
\$F0C0C	12	Envelope period high byte
\$F0C0D	13	Envelope shape
\$F0C0E	14	I/O port A latch
\$F0C0F	15	I/O port B latch

The I/O port registers are readable and writable latches in Intuition Engine. They are not connected to external pins.

## 13.3 Tone dividers

Each tone channel uses a 12-bit divider. The low byte is in the even register; the high four bits are in the following odd register.

```
divider = low + 256 * (high AND 15)
frequency = clock / (16 * divider)
```

If the divider is zero, the sound engine treats it as one. The default clock is 2000000 Hz. Useful reference clocks are:

Machine style	Clock in Hz
Atari ST	2000000
ZX Spectrum	1773400
CPC	1000000
MSX	1789773

PSG ch,divider,vol is the BASIC shortcut for this register sequence. It accepts channel 0, 1, or 2, writes the 12-bit divider pair, and writes the low four bits of vol to the matching level register.

For exact register work, type the bytes yourself:

```
10 REM PSG RAW 12 BIT TONE
20 POKE32 &H00F0800,1
30 REM SPLIT THE 12 BIT DIVIDER
40 D=284
50 POKE8 &H00F0C00,D AND 255
60 POKE8 &H00F0C01,INT(D/256) AND 15
70 REM LEVEL THEN MIXER
80 POKE8 &H00F0C08,15
90 POKE8 &H00F0C07,&H3E
```

Expected result: channel A plays the same tone as the first example. Inspecting with PEEK8(&H000F0C00), PEEK8(&H000F0C01), and PEEK8(&H000F0C08) shows 28, 1, and 15.

Lines 50 and 60 show the important AY rule: the divider is not stored as a single word. The low eight bits go to register 0; the high four bits go to register 1. Line 80 sets the channel level, and line 90 finally opens the mixer route so the tone reaches the output.

Try changing D to 568. The note drops by one octave because the divider is doubled.

## 13.4 Mixer

Register 7 decides whether each channel receives tone, noise, both, or silence. Bits 0 to 5 are inverted: write 0 to enable that route.

Bit	Field	Meaning when clear
0	TONE_A	Channel A tone on
1	TONE_B	Channel B tone on
2	TONE_C	Channel C tone on
3	NOISE_A	Noise routed to A
4	NOISE_B	Noise routed to B
5	NOISE_C	Noise routed to C
6	IO_A_OUT	Port A output mode
7	IO_B_OUT	Port B output mode

Common mixer values:

Value	Effect
&H3E	Tone A only
&H3C	Tone A and B
&H38	Tone A, B, and C
&H37	Noise A only
&H00	Tone and noise on all channels

This example makes a small three-note PSG chord:

```

10 REM PSG THREE VOICES
20 POKE32 &H000F0800,1
30 REM OPEN TONE A, B, AND C
40 PSG MIXER &H38
50 REM THREE RELATED DIVIDERS
60 PSG 0,284,14
70 PSG 1,225,11
80 PSG 2,190,9

```

Expected result: all three square-wave channels sound together.

The mixer byte is doing most of the routing work. &H38 clears tone bits 0, 1, and 2, so the three tone generators are audible, while leaving the noise routes off. The three PSG lines then set the divider and level for each channel. The lower levels on B and C keep the chord from becoming harsh.

Try changing the last number on line 80 from 9 to 3; the third voice becomes a quiet colour rather than a full note.

## 13.5 Noise and envelope

Register 6 sets the shared noise period. Only bits 0 to 4 are used. A lower value gives faster, brighter noise.

Registers 11 and 12 hold a 16-bit envelope period. Register 13 holds the shape. Writing register 13 restarts the envelope immediately.

Shape	Meaning
0 to 3	Decay and hold low
4 to 7	Attack then hold low
8	Repeating saw down
9	One-shot decay
10	Repeating triangle
11	Decay and hold high
12	Repeating saw up
13	Attack and hold high
14	Repeating inverted triangle
15	Attack and hold low

Set bit 4 in a channel level register to use the envelope instead of the fixed level. The low four level bits are ignored while envelope mode is on.

The next listing uses noise rather than tone, then lets the envelope turn that noise into a pulsing burst:

```
10 REM PSG PULSING NOISE
20 POKE32 &H000F0800,1
30 REM NOISE PERIOD AND ROUTE
40 POKE8 &H000F0C06,5
50 PSG MIXER &H37
60 REM ENVELOPE CONTROLS THE LEVEL
70 POKE8 &H000F0C08,&H10
80 PSG ENVELOPE 14,500
```

Expected result: channel A plays a bright pulsing noise burst.

Line 40 sets a short noise period, which gives the burst its bright edge. Line 50 routes only noise to channel A. Line 70 sets bit 4 in the channel A level register, so the fixed volume is ignored. Line 80 writes the envelope shape and period; writing the shape also restarts the envelope.

Try changing line 40 to POKE8 &H000F0C06,20. The burst becomes rougher and slower.

## 13.6 PSG Plus

PSG Plus follows the shared Plus rule from Chapter 11. PSG PLUS ON writes 1 to PSG\_PLUS\_CTRL at \$F0C20; PSG PLUS OFF writes 0. The AY register map stays unchanged. The PSG-specific difference is enhanced gain, oversampling, light drive, and room processing.

```
10 REM COMPARE PSG PLUS
20 POKE32 &H000F0800,1
30 PSG MIXER &H3E
40 PSG 0,284,15
50 REM LISTEN TO THE PLAIN PSG FIRST
60 FOR T=1 TO 3000
70 NEXT T
80 PSG PLUS ON
90 PRINT PEEK8(&H000F0C20)
100 REM NOW LISTEN TO THE ENHANCED PATH
110 FOR T=1 TO 3000
120 NEXT T
130 PSG PLUS OFF
140 PRINT PEEK8(&H000F0C20)
```

The tone continues while the processing mode changes. Line 80 enables the enhanced path, and line 90 proves that the control byte reads back as 1. Line 130 returns to the standard path, and line 140 reads back 0.

Try changing the divider in line 40 while leaving the rest of the listing the same. PSG Plus is a processing path, not a different register set.

## 13.7 Register-frame playback

The PSG player can stream a memory block containing PSG music data. BASIC programs usually use the higher-level media loader when playing named files, but the raw control registers are available when the data is already in memory.

Address	Name	Purpose
\$F0C10	PSG_PLAY_PTR	Start address of the music data
\$F0C14	PSG_PLAY_LEN	Length in bytes
\$F0C18	PSG_PLAY_CTRL	Write 1 start, 2 stop, 5 start loop
\$F0C1C	PSG_PLAY_STATUS	Bit 0 busy, bit 1 error

```
10 REM PSG MEMORY PLAYBACK
20 REM START A BUFFER ALREADY IN MEMORY
30 PSG PLAY &H00008000,2048
40 S=PSG STATUS
50 PRINT S
60 IF S AND 2 THEN PRINT "PSG ERROR"
```

If the memory block contains supported PSG music data, PSG STATUS reports busy while it is starting or playing. If the pointer, length, or data is not valid, bit 1 is set.

Line 30 writes the start pointer, length, and start command through the BASIC helper. Lines 40 to 60 read the status once and report only the error bit. They do not stop playback; a successful song should keep playing until it ends, loops, or the next stop command is typed.

Try changing the length to a deliberately tiny value such as 4 when testing error handling with a real PSG data block.

To stop PSG playback later:

```
10 PSG STOP
20 PRINT PSG STATUS
```

## 13.8 Side effects and limits

---

Writing a tone divider, noise period, mixer byte, or level byte takes effect immediately. Writing the envelope shape retriggers the envelope. The same envelope generator is shared by all channels that have bit 4 set in their level register.

Only channels 0 to 2 are tone channels. The BASIC shortcut ignores any higher channel number. Raw PSG register writes do not clamp values beyond the bit masks described above; unused bits are stored, but the sound calculation uses only the documented bit fields.

The next chapter covers the SN76489, another square-wave and noise chip with a different command-byte interface.