

PSG32

Table of Contents

Overview	3
Features	3
Core Hierarchy	4
Clocks	5
Computing Frequency Resolution	5
Maximum Frequency Generated	5
Example Tone Frequency Calc.	5
Registers	6
Frequency Register	8
Pulse Width Register	8
Control Register	8
ADSR Register	9
Wave Table Base Address	11
Global Registers	11
Interrupt Enable	11
Interrupt Occurred	12
Playback Ended Status	12
I/O Ports	12
Operation:	14
Frequency Synthesis	14
Wave Table	14
Filter	14
Filter Sample Frequency	14
Taps	14
Filter Coefficients	14
Filter Output	15
Software Sample	16
WISHBONE Compatibility Datasheet	17

Overview

PSG32 is an audio interface circuit (sound interface device) for use within a programmable system to interface the system to an audio output. It supports eight ADSR audio output channels with a wavetable option and a single input channel. The wave table option allows arbitrary waveforms to be played.

Features

- eight ADSR / wave table channels (“voices”)
- programmable frequency and pulse width control
- 0.02328 Hz frequency resolution (with 100.0MHz reference clock)
- attack, decay, sustain and release
- test, ringmod, sync and gate controls
- six voice types: triangle, sawtooth, pulse, noise and wave
- frequency modulation
- digital exponential decay and release modelling (2^{**n})
- 31 tap digital FIR filter

Core Hierarchy

- ▼ ● upsg1 : PSG32 (PSG32.sv) (39)
 - > ● urr1 : RoundRobinArbiter (RoundRobinArbiter.sv) (2)
 - u8 : PSGFilter3 (PSGFilter3.v)
 - gWtaOffs.genblk1[0].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[1].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[2].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[3].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[4].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[5].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[6].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[7].uedacc : edge_det (edge_det.v)
 - gWtaOffs.genblk1[8].uedacc : edge_det (edge_det.v)
 - ▼ ● gToneGenerators.genblk1[0].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - u2 : PSGHarmonicSynthesizer (PSGHarmonicSynthesizer.sv)
 - u4 : PSGNoteOutMux (PSGNoteOutMux.sv)
 - > ● gToneGenerators.genblk1[1].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - > ● gToneGenerators.genblk1[2].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - > ● gToneGenerators.genblk1[3].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - > ● gToneGenerators.genblk1[4].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - > ● gToneGenerators.genblk1[5].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - > ● gToneGenerators.genblk1[6].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - > ● gToneGenerators.genblk1[7].u1 : PSGToneGenerator (PSGToneGenerator.sv) (2)
 - gEnvelopeGenerators.genblk1[0].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gEnvelopeGenerators.genblk1[1].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gEnvelopeGenerators.genblk1[2].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gEnvelopeGenerators.genblk1[3].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gEnvelopeGenerators.genblk1[4].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gEnvelopeGenerators.genblk1[5].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gEnvelopeGenerators.genblk1[6].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gEnvelopeGenerators.genblk1[7].u2 : PSGEnvelopeGenerator (PSGEnvelopeGenerator.sv)
 - gShaper.genblk1[0].u5 : PSGShaper (PSGShaper.sv)
 - gShaper.genblk1[1].u5 : PSGShaper (PSGShaper.sv)
 - gShaper.genblk1[2].u5 : PSGShaper (PSGShaper.sv)
 - gShaper.genblk1[3].u5 : PSGShaper (PSGShaper.sv)
 - gShaper.genblk1[4].u5 : PSGShaper (PSGShaper.sv)
 - gShaper.genblk1[5].u5 : PSGShaper (PSGShaper.sv)
 - gShaper.genblk1[6].u5 : PSGShaper (PSGShaper.sv)
 - gShaper.genblk1[7].u5 : PSGShaper (PSGShaper.sv)
 - gAudout.genblk1[0].u10 : PSGVolumeControl (PSGVolumeControl.sv)
 - gAudout.genblk1[1].u10 : PSGVolumeControl (PSGVolumeControl.sv)
 - gAudout.genblk1[2].u10 : PSGVolumeControl (PSGVolumeControl.sv)
 - gAudout.genblk1[3].u10 : PSGVolumeControl (PSGVolumeControl.sv)

Clocks

The PSG32 core uses two clocks. The first of which is the system bus clock. The second clock is a 100MHz timing reference clock. The 100MHz reference clock is used during tone and the envelope generation. In order to have consistent values for frequency settings and envelope settings between systems a 100MHz reference clock should be used.

Computing Frequency Resolution

The frequency resolution depends on the reference clock used. 32-bit harmonic synthesizers are used as frequency generators. The minimum frequency resolution is then the clock frequency divided by 2^{32} . For a 100MHz clock this would be $100\text{MHz}/(2^{32}) = 0.02328 \text{ Hz}$.

Maximum Frequency Generated

The maximum frequency that can be generated is $2^{20} * \text{the minimum frequency resolution}$. For a 100MHz clock this would be 24.4kHz.

Example Tone Frequency Calc.

For a tone of 1kHz with a 100MHz clock, the value needed in the frequency control register is $1\text{kHz}/0.02328 = 42950$.








Registers

Registers are selected with the s_cs_i input port. A 512 byte register range is required. All register read / write operations are 32 bit. A block of eight registers is used for each channel.






Registers	Channel
00 to 1C	Output channel 0
20 to 3C	Output channel 1
40 to 5C	Output channel 2
60 to 7C	Output channel 3
80 to 9C	Output channel 4
A0 to BC	Output channel 5
C0 to DC	Output channel 6
E0 to FC	Output channel 7
100 to 11C	Input Channel
120 to 13C	Global control regs
140 to 17C	reserved
180 to 1FC	Filter coefficients

There are two views of the channel registers depending on whether the ADSR generator or a wave sample table is used. This is controlled by the voice type – the control register is in common between the views.

ADSR view

reg	Bits	R/W	Brief
00	 Frequency ₂₂	R/W	tone frequency
04	 Pulse Width ₂₂	R/W	pulse width
08	R-----o000 trsgFef- vvvvvv-a	R/W	Control
0C	 Attack ₂₉	R/W	Attack
10	 Decay ₃₀	R/W	Decay
14	 Sustain ₈	R/W	sustain
18	 Release ₂₄	R/W	release
1C	 0 ₅	R/W	Not used

Wave Table View

reg	Bits	R/W	Brief
00	 Frequency ₂₂	R/W	playback rate
04	 ~ ₂₂	R/W	Not used
08	R-----o000 trsgFef- 000001--	R/W	control
0C	 ~ ₂₉	R/W	Not used
10	 ~ ₃₀	R/W	Not used
14	 Volume ₈	R/W	volume
18	Length ₃₂	R/W	Sample length
1C	Address _{31...5} 0 ₅	R/W	Sample base address

Global Registers

120	-----m	R/W	master volume	
124	nnnnnnnn nnnnnnnn nnnnnnnn nnnnnnnn	R	Osc7 oscillator 7 output	
128	-----n	R	Env7 envelope 7 output	
12C	-sss-sss-sss-sss -sss-sss-sss-sss	R	envelope state	
130	----oooo -----RRRRRRRR RRRRRRRR	R/W	filter control, sample rate divider	
134	-----i ooooooooo	R/W	Interrupt enable	
138	-----i ooooooooo	R/W	Interrupt occurred	
13C	-----i ooooooooo	R/W	Playback ended	

Filter Co-efficients

180-1F8	-----s---kkkk kkkkkkkk	W	filter coefficients	

Frequency Register

This register sets the tone frequency for the voice or the audio playback frequency for samples. To set the frequency specify a value that is a multiple of the base frequency step. For example, for an 800 Hz tone with a 100MHz clock, $800/0.02328 = 34360$ would need to be specified.

For 22kHz audio playback, 947040 would be specified ($22.050e6/0.02328$).

Pulse Width Register

This register controls the pulse-width when the pulse output waveform is selected. Pulse frequency is controlled by the frequency register.

Control Register

‘R’ bit is a software reset bit for the envelope generator, automatically clears

‘o’ bits enable the output for the voice. Each channel may be routed to any output.

‘F’ bit enables frequency modulation by the output of the previous voice.

‘e’ bit when set routes the tone generator through the envelope generator. When clear the raw tone generator is used without an envelope. This is primarily for debugging.

‘f’ bit tells the sound generator to route the voice’s output to the filter

‘vvvvvv’ sets the output voice type

101000 = reverse sawtooth

010000 = triangle wave

001000 = sawtooth wave

000100 = pulse (or possibly square)

000010 = noise

000001 = wave

‘g’ bit ‘gates’ the envelop generator which when set causes it to begin generating the envelope for the voice. When the gate is turned off, the envelope generator enters the release phase.

ADSR Register

‘a’ - Attack

The attack code controls the attack rate of the sound envelope. The attack slope is triggered when the gate signal is activated. The envelope travels from a zero level to it's peak during the attack phase.

The value required in the rate register can be calculated as:

reg value = $1 / ((1 / \text{clock frequency}) / \text{desired time})$

Example: reg value = $1 / ((1 / 100\text{e}6) / 2\text{e-}3)$
= 200000

Some typical attack values:

Rate Divider	Attack Time
200000	2 ms
800000	8 ms
1600000	16 ms
2400000	24 ms
	38 ms
	56 ms
	68 ms
	80 ms
	100 ms
	239 ms
	500 ms
80000000	800 ms
	1 s
	3.2 s
	5.3 s
800000000	8 s

‘d’ = Decay

The decay code controls the decay rate of the sound envelope just after the peak has been reached from the attack phase. The envelop decays from it’s peak value down to the value set by the sustain code.

The value required in the rate register can be calculated as:

reg value = $1/((1/\text{clock frequency})/\text{desired time})/256$

Example: reg value = $1/((1/100\text{e}6) / 2\text{e-}3)/256$
= 390.625

Rate Divider	Decay/Release Time
2344	6 ms
9376	24 ms
	48 ms
	72 ms
	114 ms
	168 ms
	204 ms
	240 ms
	300 ms
	750 ms
	1.5 s
	2.4 s
	3.0 s
	9.0 s
	15.0 s
9375000	24.0 s

‘s’ = Sustain

Sustain sets the signal level at which the signal is ‘sustained’ relative to it’s peak value. There are 255 sustain levels from 0x0 to 0xFF with 0x0 being the lowest and 0xFF the maximum.

‘r’ = Release

The release code controls the rate at which the signal is ‘released’ after the gate is turned off. When the gate signal is made inactive, the release phase of the ADSR envelope begins. This is an exponential of 2 release.

Rate Divider	Decay/Release Time
	6 ms
	24 ms
	48 ms
	72 ms
	114 ms
	168 ms
	204 ms
	240 ms
	300 ms
	750 ms
	1.5 s
	2.4 s
	3.0 s
	9.0 s
	15.0 s
	24.0 s

Wave Table Base Address

This register sets the beginning address for the wave table scan. When this register is set, an internal address counter is reset to zero. The address counter will increment every time the accumulator crosses zero when generating the playback frequency. If the ‘a’ bit of the control register is set, then playback will automatically repeat when the sample length is reached.

Global Registers

120h – VOL – master volume.

12Ch - ES – reflects the envelope state for each of the four envelope generators.

SSS	Envelope State
0	IDLE
1	ATTACK
2	DECAY
3	SUSTAIN
4	RELEASE
5-7	reserved

Interrupt Enable

‘o’ enables an interrupt for the channel when the audio playback is finished.

‘i’ enables an interrupt for when the audio input buffer is full.

Interrupt Occurred

This register shows which interrupts occurred.

Playback Ended Status

This register indicates the channels for which playback has ended. The bit is set automatically but must be cleared programmatically.

I/O Ports

I/O is via a standard WISHBONE slave port with the addition of a circuit select line. An additional WISHBONE master port is used to access the wave table memory. All register accesses are 32 bit word wide accesses. The master port is 256-bits wide.

Reading the PSG has a three cycle latency before the core responds with an ack. Writing the PSG is single cycle.

Name	Wid	I/O	Description	
rst i	1	I	This is the active high reset signal	
clk i	1	I	system bus clock	
clk100 i	1	I	100MHz timing reference clock	
s_cs i	1	I	circuit select	
s_cyc i	1	I	cycle active	
s_stb i	1	I	data strobe	
s_ack o	1	O	data transfer acknowledge	
s_we i	1	I	write cycle	
s_adr i	9	I	decode / register address, the two LSB's are not used in the core but must still be supplied.	
s_dat i	32	I	data input	
s_dat o	32	O	data output	
m_cyc o	1	O	Cycle active	
m_stb o	1	O	Data strobe	
m_ack i	1	I	Data transfer acknowledge	
m_we o	1	O	Write cycle	
m_sel o	32	O	Byte lane selects	
m_adr o	32	O	master address for wave table memory	
m_dat i	256	I	master data input from wave table memory	
m_dat o	256	O	Master data output to wave table	
aud0 o	16	O	16 bit audio output	
aud1 o	16	O		
aud2 o	16	O		
aud3 o	16	O		
aud i	16	I		

Operation:

Frequency Synthesis

The PSG uses a harmonic frequency synthesizer with a 32 bit accumulator. This gives the generator a base frequency step of 0.02328Hz. ($100\text{e}6 / 2^{32}$). The upper bits of the accumulator are used as a source for audio waves.

Wave Table

The wave table may be located in main memory.

The PSG uses the tone generator accumulator to set the playback frequency at which to read.

Filter

The filter is a time domain multiplexed (TDM) filter in order to conserve resources. A digital FIR (finite impulse response) filter is used.

Filter Sample Frequency

The filter's sampling frequency is one of the characteristics controlling filter output. The sampling frequency factors into the calculations for the filter coefficients.

The sample frequency of the filter may be set using a sixteen-bit control register which contains a clock divider value. This register is provided to make it easier to use the same filter coefficients in systems with different clock rates. The filter sample rate should be set to a rate substantially higher than the highest frequency to be filtered. For example 100kHz. To get a 100kHz sample rate from a 100MHz reference clock the clock needs to be divided by 1000. So the clock rate divider (CRD) register should be set to 1000.

Taps

The filter contains a number of taps, which are points at which filter coefficients are applied to the input signal. The filter has a fixed number of 31 taps. Filter coefficients must be supplied for each tap.

Filter Coefficients

The filter coefficients control the resulting type of filter (low pass, band pass, high pass, band stop) and the frequency response of the filter. The filter coefficients are 12 fractional bits plus a sign bit. Filter coefficients range in value from -.9999 to +.9999

Filter Output

The output channel filtered audio appears on can be set with the 'oooo' bits of the filter control register.

Software Sample

```
-----  
; Beep: a 800Hz tone for 1 sec.  
; Using a 37.5MHz bus clock.  
-----  
beep:  
    tgt  
    mark1  
    ldi        r5,$FFD50000  
    lea        r1,$B0[r5]  
    ldi        r1,$FF  
    stt        r1,$B0[r5]          ; set volume to 100%  
    ldi        r1,#91626          ; 800Hz  
    stt        r1,$00[r5]          ; frequency 0  
    ldi        r1,#200            ; attack  
    stt        r1,$0C[r5]  
    ldi        r1,#100            ; decay  
    stt        r1,$10[r5]  
    ldi        r1,$80            ; 128 sustain level  
    stt        r1,$14[r5]          ;  
    ldi        r1,#500            ; release  
    stt        r1,$18[r5]  
    ldi        r1,$1504          ; gate, output, triangle wave  
    stt        r1,$008[r5]  
    csrrw     r2,#2,r0            ; get tick  
    add       r1,r2,#37500000  
.  
.ipsg1:  
    csrrw     r2,#2,r0  
    bltu      r2,r1,.ipsg1  
    ldi        r1,$0000  
    stt        r1,$008[r5]          ; turn off gate  
    ret
```


WISHBONE Compatibility Datasheet

The PSG core may be directly interfaced to a WISHBONE compatible bus.

WISHBONE Datasheet WISHBONE SoC Architecture Specification, Revision B.3		
Description:	Specifications:	
General Description:	PSG32 – programmable ADSR sound generator	
Supported Cycles:	SLAVE, READ / WRITE SLAVE, BLOCK READ / WRITE SLAVE, RMW	
Data port, size:	32 bit 32 bit 32 bit Little Endian any (undefined)	
Data port, granularity:		
Data port, maximum operand size:		
Data transfer ordering:		
Data transfer sequencing		
Clock frequency constraints:	16 MHz minimum to 300 MHz maximum	
Supported signal list and cross reference to equivalent WISHBONE signals	Signal Name: ack_o adr_i(7:0) clk_i dat_i(31:0) dat_o(31:0) cyc_i stb_i we_i	WISHBONE Equiv. ACK_O ADR_I() CLK_I DAT_I() DAT_O() CYC_I STB_I WE_I
Special Requirements:		