APU

From NESdev Wiki

The NES **APU** is the **audio processing unit** in the NES console which generates sound for games. It is implemented in the RP2A03 (NTSC) and RP2A07 (PAL) chips. Its registers are mapped in the range $4000–$4013, $4015 and $4017.

Contents

Overview

For a simple subset of APU functionality oriented toward music engine developers rather than emulator developers, see APU basics.

The APU has five channels: two pulse wave generators, a triangle wave, noise, and a delta modulation channel for playing DPCM samples.

Each channel has a variable-rate timer clocking a waveform generator, and various modulators driven by low-frequency clocks from the frame counter. The DMC plays samples while the other channels play waveforms. Each sub-unit of a channel generally runs independently and in parallel to other units, and modification of a channel's parameter usually affects only one sub-unit and doesn't take effect until that unit's next internal cycle begins.

The read/write status register allows channels to be enabled and disabled, and their current length counter status to be queried.

The outputs from all the channels are combined using a non-linear mixing scheme.

Conditions for channel output

The pulse, triangle, and noise channels will play their corresponding waveforms (at either a constant volume or at a volume controlled by an envelope) only when (and in the model given here, *precisely* when) their length counters are all non-zero (this includes the linear counter for the triangle channel). There are two exceptions for the pulse channels, which can also be silenced either by having a frequency above a certain threshold (see below), or by a sweep towards lower frequencies (longer periods) reaching the end of the range.

The DMC channel always outputs the value of its counter, regardless of the status of the DMC enable bit; the enable bit only controls automatic playback of delta-encoded samples (which is done through counter updates).

Notes

- This reference describes the *abstract* operation of the APU. The *exact* hardware implementation is not necessarily relevant to an emulator, but the Visual 2A03 (http://www.qmtpro.com/~nes/chipimages/visual2a03/) project can be used to determine detailed information about the hardware implementation.
- The Famicom had an audio return loop on its cartridge connector allowing extra audio from individual cartridges. See Expansion audio for details on the audio produced by various mappers.
- For a basic usage guide to the APU, see APU basics, or Nerdy Nights: APU overview.
- The APU may have additional diagnostic features if CPU pin 30 is pulled high. See diagram by Quietust.

Specification

Registers

- See APU Registers for a complete APU register diagram.

Registers	Channel	Units
$4000–$4003	Pulse 1	Timer, length counter, envelope, sweep
$4004–$4007	Pulse 2	Timer, length counter, envelope, sweep
$4008–$400B	Triangle	Timer, length counter, linear counter
$400C–$400F	Noise	Timer, length counter, envelope, linear feedback shift register
$4010–$4013	DMC	Timer, memory reader, sample buffer, output unit
$4015	All	Channel enable and length counter status
$4017	All	Frame counter

Pulse ($4000–$4007)

- See APU Pulse

The pulse channels produce a variable-width pulse signal, controlled by volume, envelope, length, and sweep units.

$4000 / **$4004**	`DDLC VVVV`	Duty (D), envelope loop / length counter halt (L), constant volume (C), volume/envelope (V)
$4001 / **$4005**	`EPPP NSSS`	Sweep unit: enabled (E), period (P), negate (N), shift (S)
$4002 / **$4006**	`TTTT TTTT`	Timer low (T)
$4003 / **$4007**	`LLLL LTTT`	Length counter load (L), timer high (T)

- **$4000**:
  - **D**: The width of the pulse is controlled by the duty bits in $4000/$4004. See APU Pulse for details.
  - **L**: 1 = Infinite play, 0 = One-shot. If 1, the length counter will be frozen at its current value, and the envelope will repeat forever.
    - The length counter and envelope units are clocked by the frame counter.
    - If the length counter's current value is 0 the channel will be silenced whether or not this bit is set.
    - When using a one-shot envelope, the length counter should be loaded with a time longer than the length of the envelope to prevent it from being cut off early.
    - When looping, after reaching 0 the envelope will restart at volume 15 at its next period.
  - **C**: If *C* is set the volume will be a constant. If clear, an envelope will be used, starting at volume 15 and lowering to 0 over time.
  - **V**: Sets the direct volume if constant, otherwise controls the rate which the envelope lowers.
- The frequency of the pulse channels is a division of the CPU Clock (f_{CPU}; 1.789773MHz NTSC, 1.662607MHz PAL). The output frequency (f) of the generator can be determined by the 11-bit period value (t) written to $4002–$4003/$4006–$4007. If $t < 8$, the corresponding pulse channel is silenced.
  - $f = f_{CPU} / (16 \times (t + 1))$
  - $t = (f_{CPU} / (16 \times f)) - 1$
- Writing to $4003/$4007 reloads the length counter, restarts the envelope, and resets the phase of the pulse generator. Because it resets phase, vibrato should only write the low timer register to avoid a phase reset click. At some pitches, particularly near A440, wide vibrato should normally be avoided (e.g. this flaw is heard throughout the Mega Man 2 ending).

- Registers \$4001/\$4005 control the sweep unit. Enabling the sweep causes the pitch to constantly rise or fall. When the frequency reaches the end of the generator's range of output, the channel is silenced. See APU Sweep for details.
- The two pulse channels are combined in a nonlinear mix (see mixer below).

Triangle (\$4008–\$400B)

- See APU Triangle

The triangle channel produces a quantized triangle wave. It has no volume control, but it has a length counter as well as a higher resolution linear counter control (called "linear" since it uses the 7-bit value written to \$4008 directly instead of a lookup table like the length counter).

\$4008	`CRRR RRRR`	Length counter halt / linear counter control (C), linear counter load (R)
\$4009	`---- ----`	Unused
\$400A	`TTTT TTTT`	Timer low (T)
\$400B	`LLLL LTTT`	Length counter load (L), timer high (T), set linear counter reload flag

- The triangle wave has 32 steps that output a 4-bit value.
- **C**: This bit controls both the length counter and linear counter at the same time.
  - When set this will stop the length counter in the same way as for the pulse/noise channels.
  - When set it prevents the linear counter's internal reload flag from clearing, which effectively halts it if \$400B is written after setting *C*.
  - The linear counter silences the channel after a specified time with a resolution of 240Hz in NTSC (see frame counter below).
  - Because both the length and linear counters are be enabled at the same time, whichever has a longer setting is redundant.
  - See APU Triangle for more linear counter details.
- **R**: This reload value will be applied to the linear counter on the next frame counter tick, but only if its reload flag is set.
  - A write to \$400B is needed to raise the reload flag.
  - After a frame counter tick applies the load value *R*, the reload flag will only be cleared if *C* is also clear, otherwise it will continually reload (i.e. halt).
- The pitch of the triangle channel is one octave below the pulse channels with an equivalent timer value (i.e. use the formula above but divide the resulting frequency by two).
- Silencing the triangle channel merely halts it. It will continue to output its last value rather than 0.
- There is no way to reset the triangle channel's phase.

Noise ($400C–$400F)

- See APU Noise

The noise channel produces noise with a pseudo-random bit generator. It has a volume, envelope, and length counter like the pulse channels.

$400C	`--LC VVVV`	Envelope loop / length counter halt (L), constant volume (C), volume/envelope (V)
$400D	`---- ----`	Unused
$400E	`L--- PPPP`	Loop noise (L), noise period (P)
$400F	`LLLL L---`	Length counter load (L)

- The frequency of the noise is determined by a 4-bit value in $400E, which loads a period from a lookup table (see APU Noise).
- If bit 7 of $400E is set, the period of the random bit generation is drastically shortened, producing a buzzing tone (e.g. the metallic ding during *Solstice*'s gameplay). The actual timbre produced depends on whatever bits happen to be in the generator when it is switched to periodic, and is somewhat random.

DMC ($4010–$4013)

- See APU DMC

The delta modulation channel outputs a 7-bit PCM signal from a counter that can be driven by DPCM samples.

$4010	`IL-- RRRR`	IRQ enable (I), loop (L), frequency (R)
$4011	`-DDD DDDD`	Load counter (D)
$4012	`AAAA AAAA`	Sample address (A)
$4013	`LLLL LLLL`	Sample length (L)

- DPCM samples are stored as a stream of 1-bit deltas that control the 7-bit PCM counter that the channel outputs. A bit of 1 will increment the counter, 0 will decrement, and it will clamp rather than overflow if the 7-bit range is exceeded.
- DPCM samples may loop if the loop flag in $4010 is set, and the DMC may be used to generate an IRQ when the end of the sample is reached if its IRQ flag is set.
- The playback rate is controlled by register $4010 with a 4-bit frequency index value (see APU DMC for frequency lookup tables).
- DPCM samples must begin in the memory range $C000–$FFFF at an address set by register $4012 (address = `%11AAAAAA AA000000`).
- The length of the sample in bytes is set by register $4013 (length = `%LLLL LLLL0001`).

Other Uses

- The $4011 register can be used to play PCM samples directly by setting the counter value at a high frequency. Because this requires intensive use of the CPU, when used in games all other gameplay is usually halted to facilitate this.
- Because of the APU's nonlinear mixing, a high value in the PCM counter reduces the volume of the triangle and noise channels. This is sometimes used to apply limited volume control to the triangle channel (e.g. *Super Mario Bros.* adjusts the counter between levels to accomplish this).
- The DMC's IRQ can be used as an IRQ-based timer when the mapper used does not have one available.

Status ($4015)

The status register is used to enable and disable individual channels, control the DMC, and can read the status of length counters and APU interrupts.

$4015 write	`---D NT21`	Enable DMC (D), noise (N), triangle (T), and pulse channels (2/1)

- Writing a zero to any of the channel enable bits will silence that channel and immediately set its length counter to 0.
- If the DMC bit is clear, the DMC bytes remaining will be set to 0 and the DMC will silence when it empties.
- If the DMC bit is set, the DMC sample will be restarted *only if* its bytes remaining is 0. If there are bits remaining in the 1-byte sample buffer, these will finish playing before the next sample is fetched.
- Writing to this register clears the DMC interrupt flag.
- Power-up and reset have the effect of writing $00, silencing all channels.

$4015 read	`IF-D NT21`	DMC interrupt (I), frame interrupt (F), DMC active (D), length counter > 0 (N/T/2/1)

- N/T/2/1 will read as 1 if the corresponding length counter is greater than 0. For the triangle channel, the status of the linear counter is irrelevant.
- D will read as 1 if the DMC bytes remaining is more than 0.
- Reading this register clears the frame interrupt flag (but not the DMC interrupt flag).
- If an interrupt flag was set at the same moment of the read, it will read back as 1 but it will *not* be cleared.
- This register is internal to the CPU and so the external CPU data bus is disconnected when reading it. Therefore the returned value cannot be seen by external devices and the value does not affect open bus.
- Bit 5 is open bus. Because the external bus is disconnected when reading $4015, the open bus value comes from the last cycle that did not read $4015.

Frame Counter ($4017)

- See APU Frame Counter

$4017	`MI-- ----`	Mode (M, 0 = 4-step, 1 = 5-step), IRQ inhibit flag (I)

The frame counter is controlled by register $4017, and it drives the envelope, sweep, and length units on the pulse, triangle and noise channels. It ticks approximately 4 times per frame (240Hz NTSC), and executes either a 4 or 5-step sequence, depending how it is configured. It may optionally issue an IRQ on the last tick of the 4-step sequence.

The following diagram illustrates the two modes, selected by bit 7 of $4017:

```
mode 0:    mode 1:       function
---------  -----------  -----------------------------
 - - - f    - - - - -    IRQ (if bit 6 is clear)
 - l - l    - l - - l    Length counter and sweep
 e e e e    e e e - e    Envelope and linear counter
```

Both the 4 and 5-step modes operate at the same rate, but because the 5-step mode has an extra step, the effective update rate for individual units is slower in that mode (total update taking ~60Hz vs ~48Hz in NTSC). Writing to $4017 resets the frame counter and the quarter/half frame triggers happen simultaneously, but only on "odd" cycles (and only after the first "even" cycle after the write occurs) – thus, it happens either 2 or 3 cycles after the write (i.e. on the 2nd or 3rd cycle of the next instruction). After 2 or 3 clock cycles (depending on when the write is performed), the timer is reset. Writing to $4017 with bit 7 set ($80) will immediately clock all of its controlled units at the beginning of the 5-step sequence; with bit 7 clear, only the sequence is reset without clocking any of its units.

Note that the frame counter is not exactly synchronized with the PPU NMI; it runs independently at a consistent rate which is approximately 240Hz (NTSC). Some games (e.g. *The Legend of Zelda*, *Super Mario Bros.*) manually synchronize it by writing $C0 or $FF to $4017 once per frame.

Length Counter

- See APU Length Counter

The pulse, triangle, and noise channels each have their own length counter unit. It is clocked twice per sequence, and counts down to zero if enabled. When the length counter reaches zero, the channel is silenced. It is reloaded by writing a 5-bit value to the appropriate channel's length counter register, which will load a value from a lookup table. (See APU Length Counter for the table.)

The triangle channel has an additional **linear counter** unit which is clocked four times per sequence (like the envelope of the other channels). It functions independently of the length counter, and will also silence the triangle channel when it reaches zero.

Mixer

- See APU Mixer

The APU audio output signal comes from two separate components. The pulse channels are output on one pin, and the triangle/noise/DMC are output on another, after which they are combined. Each of these channels has its own nonlinear DAC. For details, see APU Mixer.

After combining the two output signals, the final signal may go through a lowpass and highpass filter. For instance, RF demodulation in televisions usually results in a strong lowpass. The NES' RCA output circuitry has a more mild lowpass filter.

Glossary

- All APU channels have some form of frequency control. The term **frequency** is used where larger register value(s) correspond with higher frequencies, and the term **period** is used where smaller register value(s) correspond with higher frequencies.
- In the block diagrams, a **gate** takes the input on the left and outputs it on the right, unless the control input on top tells the gate to ignore the input and always output 0.
- Some APU units use one or more of the following building blocks:
  - A **divider** outputs a clock periodically. It contains a period reload value, P, and a counter, that starts at P. When the divider is clocked, if the counter is currently 0, it is reloaded with P and generates an output clock, otherwise the counter is decremented. In other words, the divider's period is P + 1.
  - A divider can also be forced to reload its counter immediately (counter = P), but this *does not* output a clock. Similarly, changing a divider's period reload value *does not* affect the counter. Some counters offer no way to force a reload, but setting P to 0 (http://forums.nesdev.org/viewtopic.php?p=186129#p186129) at least synchronizes it to a known state once the current count expires.
  - A divider may be implemented as a down counter (5, 4, 3, ...) or as a linear feedback shift register (LFSR). The dividers in the pulse and triangle channels are linear down-counters. The dividers for noise, DMC, and the APU Frame Counter are implemented as LFSRs to save gates compared to the equivalent down counter.
  - A **sequencer** continuously loops over a sequence of values or events. When clocked, the next item in the sequence is generated. In this APU documentation, clocking a sequencer usually means either advancing to the next step in a waveform, or the event sequence of the Frame Counter device.
  - A **timer** is used in each of the five channels to control the sound frequency. It contains a divider which is clocked by the CPU clock. The triangle channel's timer is clocked on every CPU cycle, but the pulse, noise, and DMC timers are clocked only on every second CPU cycle and thus produce only even periods.

References

- Blargg's NES APU Reference (http://nesdev.org/apu_ref.txt)
- Brad Taylor's 2A03 Technical Reference (http://nesdev.org/2A03%20technical%20reference.txt)