Notas sobre el algoritmo de síntesis directa empleado en la implementación del sintetizador para Arduino Due.

Naturaleza del algoritmo de síntesis.

El algoritmo de síntesis empleado en la versión para Arduino Due varia sustancialmente de la versión de Arduino Uno, ofreciendo varias mejoras gracias al rendimiento más elevado del mismo. Una de las características más notables es que ahora soporta la atenuación continua en la fase de sostenimiento de las notas, creando notas de duración prolongada que decaen suavemente, haciendo innecesario el método de síntesis mediante tablas de logaritmos de seno. Adicionalmente, esta atenuación se presenta como una envolvente exponencial amortiguada, creando un sonido que decae de forma natural.

Esto se consigue empleando una única tabla de envolvente, en vez de las 3 en el Arduino Uno: Una para ataque, otra para decaída y otra de liberación (en sostenimiento la amplitud se mantiene constante y no requiere tabla). Lo que se hace en el algoritmo de Arduino Due es que se recorre esta tabla única a ritmos y direcciones variables, para así generar las variaciones en la envolvente en cada fase.

La variación de ritmo y dirección de tabla no es eficiente en el Arduino Uno, debido a que se requieren numerosas restas y comparaciones con signo de 24 bits (que por motivos de diseño del lenguaje C++ sólo se permiten hacer en 32 bits) y los ciclos de CPU son escasos. Esto no es el caso en el Arduino Due, que opera nativamente en 32 bits y a mayor velocidad.

Consideraciones sobre el rango dinámico de salida.

Dadas las prestaciones más generosas de la plataforma, la salida PWM de 9 bits (ampliada gracias a la reprogramación del temporizador del ATMega328) se sustituye por el DAC de 12 bits presente en el Due, mejorando significativamente la calidad de audio de salida.

Aunado a la mejora de la resolución de salida, el método de síntesis directa permite usar cualquier forma de onda más allá de la función seno. Esto permite crear sonidos con composiciones armónicas más ricas y por tanto más variedad en los instrumentos sintetizables.

Sin embargo, con la mejora de resolución ocurre un fenómeno que resulta inconveniente al querer aplicar funciones de atenuación exponencial, y es que por su naturaleza, estas funciones jamás llegan a cero (llegar a cero implica evaluar una potencia infinita, lo que a su vez involucra tiempos infinitos), esto resulta en sonidos cuya amplitud decae naturalmente en su inicio, pero que tardan mucho

tiempo en desvanecerse completamente, sobretodo cuando el rango dinámico, que ahora posee más bits, puede representar las amplitudes pequeñas con mejor definición.

Esto quiere decir que los sonidos tardan mucho en terminarse completamente, sin embargo pararlos de forma abrupta puede causar artefactos que resultan audibles, aun con amplitudes pequeñas.

El inconveniente resulta evidente cuando se reproducen piezas musicales, las cuales cuentan con muchos sonidos consecutivos, y si las notas individuales se procesan durante mucho tiempo los espacios para las voces se agotan con facilidad.

Sin embargo este comportamiento de oscilación prolongada a baja amplitud es natural y hasta cierto punto deseable, pero dadas las limitaciones de los recursos de hardware y el hecho de que los sonidos de baja amplitud resultan poco perceptibles (sobretodo si se mezclan con sonidos de mayor amplitud), resulta necesario idear una manera de liberar estas voces con más agilidad, sobretodo en su etapa final.

La solución a este problema es simple: se debe evitar recortar la voz, pero se puede acelerar su atenuación cuando su amplitud disminuye lo suficiente. Para ello se altera la función exponencial en sí, y se usa una función que llamamos “pseudo-exponencial”, la cual presenta el comportamiento regular para las amplitudes elevadas, pero se convierte en una función lineal (que decae rápidamente) para las amplitudes menores.

Vale señalar que en la implementación del Arduino Uno (algoritmo por tablas de logaritmo de seno) se emplea la función exponencial sin alteraciones, pero es porque el rango dinámico se ve limitado al emplear una menor resolución, aparte que se usa un artificio que consiste en silenciar el sonido cuando la amplitud disminuye por menos de medio bit (las muestras son redondeadas hacia cero). En este caso el recorte abrupto no se nota pues se ve opacado por el elevado ruido de cuantización del DAC.

Deducción de la formula de envolvente amortiguada pseudo-exponencial.

La obtención de la envolvente con las características ya mencionadas, requiere la unión de 2 funciones matemáticas, las cuales son implementadas como funciones seccionadas, de la siguiente forma:

$$env(x)=\begin{cases} Ab^{x}, & 0\le x\le x_i \\ \frac{y_f-y_i}{x_f-x_i}(x-x_i)+y_i, & x_i\le x\le x_f \end{cases}$$

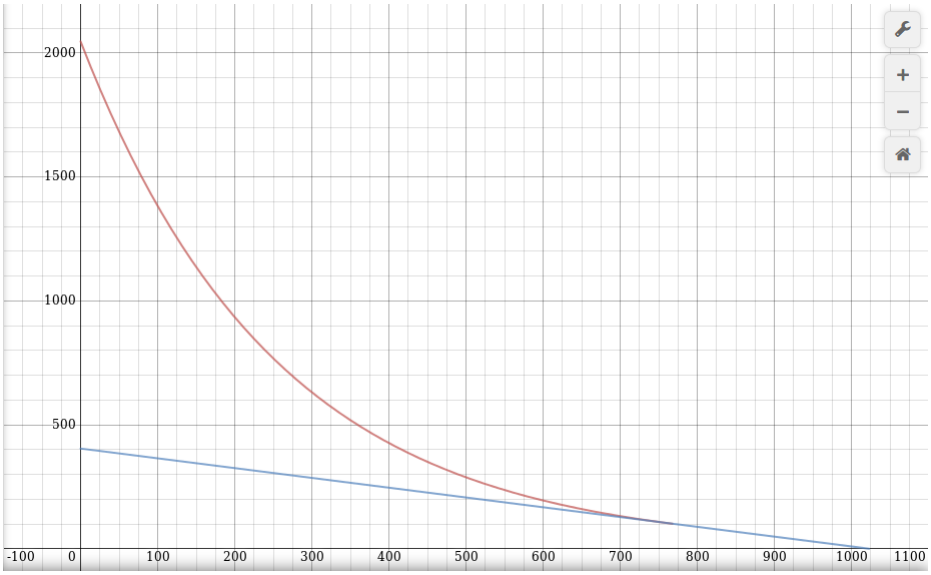
Donde:

- A representa la amplitud máxima de la señal en unidades de fracción de punto fijo. Este valor se elige de tal forma que sea exactamente $2^n - 1$ donde n es la resolución en bits de la fracción de punto fijo (16 bits en nuestro caso).
- b es la base de la función exponencial. Esta base se debe determinar mediante cálculo y será abordada más adelante.
- (x_i, y_i) es el punto donde la función exponencial se intersecta con la lineal.
- (x_f, y_f) es el punto final de la curva.

Resulta evidente que para evitar artefactos en la reproducción de los sonidos la envolvente debe tener una transición suave cuando pasa de un segmento a otro, por lo cual se imponen 2 requerimientos:

- El punto donde se intersectan ambas ecuaciones debe ser exactamente el mismo (evitar saltos).
- El punto donde se intersectan debe tener pendiente uniforme, o dicho de otra forma, ser diferenciable (para evitar deformaciones armónicas).

Si estos requerimientos se cumplen, la envolvente pseudo-exponencial deberá tener una apariencia como la siguiente:



Cabe señalar que el punto de intersección en si puede ser elegido de forma arbitraria, siempre y cuando se elija para una amplitud (y_i) pequeña. De esta forma se elige un xi que se encuentre a 3 cuartas partes del recorrido total, así durante los primeros 3 cuartos la atenuación es exponencial y en el último cuarto la atenuación es lineal:

$$x_i = \frac{3}{4}(2^n)$$

En este caso n representa la cantidad de bits de direcciones de la tabla, que para nuestro caso es de 10 bits o 1024 muestras. Luego se tiene un valor de $x_i = 768$.

Con estas consideraciones se procede a simultanear las 2 ecuaciones para hacer que coincidan en el mismo punto, cumpliendo con el primer requerimiento:

$$\begin{aligned} env(x)\Big|_{x=x_i-} &= env(x)\Big|_{x=x_i+} \\ Ab^{x_i} &= \frac{y_f - y_i}{x_f - x_i}(x_i - x_i) + y_i \\ Ab^{x_i} &= y_i \end{aligned}$$

Esta expresión será usada más adelante, pero de momento se obtiene la derivada de la ecuación exponencial, quedando de la siguiente manera:

$$\begin{aligned} env(x) &= Ab^x, \qquad 0 \le x \le x_i \\ env(x) &= A(b)^x \\ env(x) &= A(e^{\ln b})^x \qquad \leftarrow b = e^{\ln b} \\ env(x) &= Ae^{x \ln b} \\ \frac{\partial env(x)}{\partial x} &= Ae^{x \ln b}(\ln b) \\ \frac{\partial env(x)}{\partial x} &= A(e^{\ln b})^x(\ln b) \\ \frac{\partial env(x)}{\partial x} &= Ab^x(\ln b) \end{aligned}$$

Luego se obtiene la derivada de la ecuación lineal, considerando que termina en atenuación total ($y_f = 0$)

$$
\begin{aligned}
env(x) &= \frac{y_f - y_i}{x_f - x_i}(x - x_i) + x_i, \qquad x_i \le x \le x_f \\
env(x) &= \frac{-y_i}{x_f - x_i}(x - x_i) + x_i \\
env(x) &= \frac{-y_i x}{x_f - x_i} + \frac{y_i x_i}{x_f - x_i} + x_i \\
\frac{\partial env(x)}{\partial x} &= \frac{\partial}{\partial x}\left(\frac{-y_i x}{x_f - x_i}\right) + \frac{\partial}{\partial x}\left(\frac{-y_i x_i}{x_f - x_i}\right) + \frac{\partial x_i}{\partial x} \\
\frac{\partial env(x)}{\partial x} &= \frac{-y_i}{x_f - x_i} \text{ (Las ultimas dos no dependen de x)}
\end{aligned}
$$

Para cumplir el requerimiento de pendiente uniforme, se igualan estas derivadas por la izquierda y derecha:

$$
\left.\frac{\partial env(x)}{\partial x}\right|_{x=x_i-} = \left.\frac{\partial env(x)}{\partial x}\right|_{x=x_i+}
$$

$$
\begin{aligned}
A b^{x_i}(\ln b) &= \frac{-y_i}{x_f - x_i} \\
y_i(\ln b) &= \frac{-y_i}{x_f - x_i} \quad \leftarrow \text{Sustituyendo la ec. anterior} \\
\ln b &= \frac{-1}{x_f - x_i} \\
b &= e^{\frac{-1}{x_f - x_i}}
\end{aligned}
$$

Como puede verse, la base no depende de la amplitud sino únicamente del punto de intersección y del punto final de la curva (que se corresponde directamente con la longitud de la tabla), luego, para una tabla de 10 bits o 1024 elementos y el punto de intersección a 3 cuartas partes del recorrido:

$$
\begin{aligned}
b &= e^{\frac{-1}{1023 - 768}} \\
b &= 0.99608611
\end{aligned}
$$

Ahora sólo resta encontrar la amplitud de la envolvente en el punto de intersección, para lo cual se emplea la expresión encontrada con anterioridad:

$$
\begin{aligned}
A b^{x_i} &= y_i \\
y_i &= 65535(0.99608611)^{768} \\
y_i &= 3224.6346
\end{aligned}
$$

Con ello, la ecuación de la tabla de envolvente exponencial queda completamente definida, quedando de la siguiente forma (usado resolución de punto fijo de 16 bits):

$$env(x)=\begin{cases}65535(0.99608611)^{x}, & 0\le x\le 768\\ \frac{0-3224.6346}{1023-768}(x-768)\ +\ 3224.6346, & 768\le x\le 1023\end{cases}$$

Simplificando:

$$env(x)=\begin{cases}65535(0.99608611)^{x}, & 0\le x\le 768\\ -12.645626x\ +\ 12936.475, & 768\le x\le 1023\end{cases}$$