UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Introducción al uso de libmikmod para audio modular en el computador

Apellidos, nombre	**Agustí Melchor, Manuel** (magusti@disca.upv.es)
Departamento	**Departamento de Ingeniería de Sistemas y Computadores**
Centro	Escola Tècnica Superior d'Enginyeria Informàtica Universitat Politècnica de València

1 Resumen de las ideas clave

En este artículo vamos a presentar la librería **libmikmod** (*MikMod Sound Library*) que nos permitirá la reproducción de ficheros de audio modulares. Estos ficheros permiten con tamaños de entre 100 y 600 Kb, permiten mantener una secuencia sonora de alrededor de cuatro minutos con cuatro instrumentos sonando al mismo tiempo.

Los ficheros de audio modular almacenan muestras de sonido (notas de los instrumentos instrumentos) que van a sonar en las secuencias (también llamadas patrones o páginas) de conjuntos de datos (notas y otros valores que describen los instrumentos que suenan) que se repiten en el tiempo. Estas secuencias se subdividen en un número de pistas (*tracks*) que definen el número de notas (también llamadas voces) que van a sonar al mismo tiempo. Para su edición se emplean editores especializados denominados secuenciadores o *trackers*, de los que existen múltiples ejemplos y con especificaciones de formatos propios. Esto es, no hay un estándar en este campo.

El interés de esta librería se basa en:

- Su capacidad de reproducir formatos de ficheros de audio modular creados a partir de secuenciadores tan populares como: MOD (*Ultimate Soundtracker*), IT (*Impulse Tracker*), S3M (*Scream Tracker 3*), STM (*Scream Tracker*) o XM (*FastTracker 2*), entre otros.
- Ha sido portada a diferentes plataformas de computadores y a varias de videoconsolas (como la N3DS y la PSP[1]).
- Su licencia (LGPL[2]) que permite incorporarla a una desarrollo propio.
- Su pequeño tamaño, lo que permite incorporarla a una aplicación propia sin un gran impacto en el consumo de recursos.

2 Objetivos

Una vez que el lector haya se lea con detenimiento este documento, será capaz de:

- Instalar la librería *libmikmod* en la plataforma de computador de escritorio de su elección; está disponible para sistemas Linux (Unix), mac OS y Windows.
- Compilar algunos programas de ejemplo para experimentar con la librería.
- Será capaz de identificar las instrucciones básicas que permiten reproducir los ficheros modulares más populares.

Para abordarlos, en ese documento se está trabajando sobre la distribución de Linux Ubuntu 20.04, pero la explicación es trasladable a cualquier otra plataforma de computadores de escritorio y las referencias incluidas le llevarán a los materiales que puedan ser necesario y propios de una plataforma diferente.

---

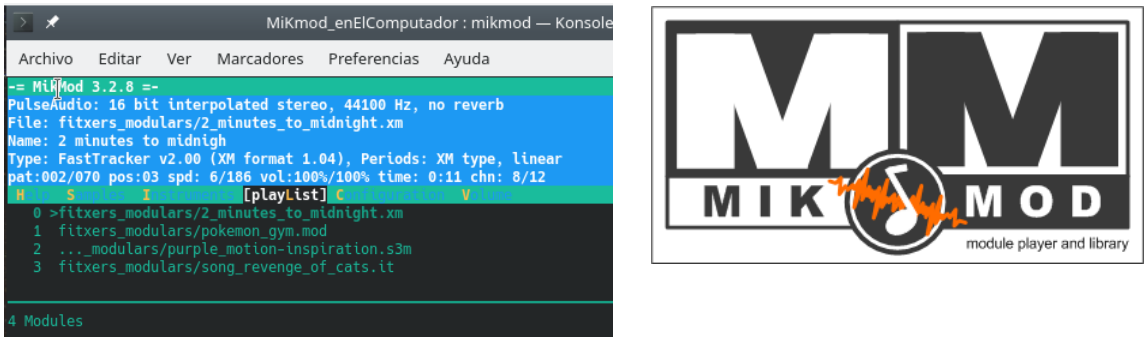
[1] Puede leer acerca de esto en <http://lukasz.dk/mirror/forums.ps2dev.org/viewtopica8e0.html?t=2540 >.

[2] Más acerca de la licencia LGPL en <http://www.gnu.org/licenses/lgpl-3.0.html>.

Se sugiere al lector que siga los comentarios de este documento comprobándolos en su propio equipo al tiempo que lee estos contenidos. Para ello se propondrá revisar algunos ejemplos que acompañan a la distribución de *libmikmod*.

3 Introducción

La primera versión de la libmikmod [1] se debe a Jean-Paul Mikkers (MikMak) y fue implementada sobre MS-DOS. Actualmente, ha sido portada a diferentes plataformas y , dentro de Unix, puede operar sobre diferente motores de audio como OSS, ALSA, SDL y PulseAudio, así como escribir en disco en formato *raw* (sin metadatos) o WAVE. El reproductor con interfaz de modo de texto MikMod véase la Figura 1) fue creado para mostrar las capacidades de la librería.



a) *b)*

Figura 1: Mikmod es un reproductor implementado con libmikmod: (a) captura de pantalla y (b) logotipo.

MikMod trabaja sobre dos pilares fundamentalmente: los ficheros de módulos de sonido (con varias voces, utilizados para ambientación musical) y los efectos de sonido (que solo utilizar una y se destinan a sonidos ocasionales y breves). El API[3] permite operar tanto a nivel global como al nivel de manipulación de cada voz por separado.

Mikmod proporciona un nivel donde realiza las operaciones que proporciona sobre los sonidos, así como la mezcla de todas las voces. Cuando se inicializa, comprueba la disponibilidad de alguno de los manejadores (*drivers*) que conoce para acceder al hardware de audio y, si es posible, utilizará la aceleración hardware que estos puedan ofrecer.

En el momento de redactar este documento, en Ubuntu 20.04 está disponible la versión 3.3. 11, de manera que es posible para desarrollar con *libmikmod*, instalar con rapidez los paquetes necesarios con las órdenes de la Figura 2.

[3] Disponible en <http://mikmod.sourceforge.net/doc/mikmod-3.1.21.html#Using-the-Library>.

```
sudo apt-get update
sudo apt-get install libmikmod-dev
```

Figura 2: Instalación de libmikmod.

Vamos a ver dos ejemplos que acompañan a la distribución de *libmikmod*, con alguna variación para comprobar cómo se usa esta librería. El detalle completo del API de la misma se puede encontrar en la documentación [2].

4 Desarrollo

Si ha seguido los pasos de instalación tendrá en su equipo instalada la librería junto a los ficheros de cabeceras necesario para incluir en código, sobre lenguaje C, las funciones del API de esta librería.

Puede comprobar que se ha instalado correctamente desde la línea con las órdenes y resultados similares a los mostrados en la Figura 3.

```
$ pkg-config libmikmod --modversion
3.3.11
$ pkg-config libmikmod --cflags
$ pkg-config libmikmod --libs
-lmikmod
```

Figura 3: Comprobaciones de la instalación de libmikmod.

Veamos ahora cómo es el esqueleto básico de una aplicación que utiliza libmikmod, cómo se compila y cómo puede utilizarse para reproducir audio modular.

4.1 Inicializaciones de libmikmod

La estructura básica de una aplicación que hace uso de *libmikmod* se pueden observar en el ejemplo ampliado de *skeleton.c* [2] que se muestra el Listado 1. En él, junto a los comentarios del código original, se pueden destacar las siguientes instrucciones:

- En la línea 5 se comprueba la disponibilidad de manejadores para acceder al hardware de audio.
  - Podremos ver el resultado de esta operación con las líneas 6 a la 8. La Figura 5 muestra hasta once opciones en el equipo donde se ha probado bajo Ubuntu 20.04.
- La línea 10, inicializa la librería, se podría aquí escoger uno de los manejadores encontrados en la línea 5 o modificar los valores globales (frecuencia de muestreo,

tamaño de muestra, número de voces, etc.) que permiten la librería y el hardware encontrados.

- La línea doce muestra cómo se puede preguntar por la lista de formatos modulares soportados.
  - Entre las líneas 13 y 15 podemos ver cómo averiguar que formatos de ficheros modulares reconoce la versión de *libmikmod* utilizada. En la Figura 5 se puede ver la lista de los veinte formatos soportados.
- De ahí en adelante vendría la lógica de la aplicación. Así que solo resta que para terminar la aplicación es importante liberar los recursos asignados y el uso de la tarjeta de sonido, lo que se hace en la línea diecisiete.

Para compilarlo se empleará la línea de órdenes que muestra la Figura 4.

```
$ gcc skeleton.c -o skeleton `pkg-config libmikmod --cflags –libs`
```

Figura 4: Línea de compilación para el ejemplo de skeleton.c.

```
1.  #include <mikmod.h>
2.  int main() {
3.    char *informacio;
4.
5.    MikMod_RegisterAllDrivers(); // register all the drivers
6.    informacio = MikMod_InfoDriver();
7.    printf("libMikMod registered drivers: \n%s\n", informacio);
8.    MikMod_free( informacio );
9.
10.   MikMod_Init(""); //   initialize the library
11.
12.   MikMod_RegisterAllLoaders(); //  registered module loaders
13.   informacio = MikMod_InfoLoader();
14.   printf("libMikMod registered loaders: \n%s\n", informacio);
15.   MikMod_free( informacio );
16.       /* we could play some sound here... */
17.   MikMod_Exit(); // give up
18. }
```

Listado 1: Listado de skeleton.c [1].

```
$ skeleton
libMikMod registered drivers:  1 PulseAudio driver v0.1
 2 SDL Driver v1.3
 3 OpenAL driver v0.4
 4 Advanced Linux Sound Architecture (ALSA) driver v1.11
 5 Open Sound System driver v1.7
 6 Nosound Driver v3.0
 7 Wav disk writer (music.wav) v1.3
 8 AIFF disk writer (music.aiff) v1.2
 9 Raw disk writer (music.raw) v1.1
10 Piped Output driver v0.2
11 Standard output driver v1.1
libMikMod registered loaders:
669 (Composer 669, Unis 669)
AMF (DSMI Advanced Module Format)
AMF (ASYLUM Music Format V1.0)
DSM (DSIK internal format)
FAR (Farandole Composer)
GDM (General DigiMusic)
IT (Impulse Tracker)
IMF (Imago Orpheus)
MOD (31 instruments)
MED (OctaMED)
MTM (MultiTracker Module editor)
OKT (Amiga Oktalyzer)
S3M (Scream Tracker 3)
STM (Scream Tracker)
STX (Scream Tracker Music Interface Kit)
ULT (UltraTracker)
UMX (Unreal UMX container)
APUN (APlayer) and UNI (MikMod)
XM (FastTracker 2)
MOD (15 instrument)
```

Figura 5: Resultado de la ejecución de skeleton.c.

4.2 Reproducción de archivos de audio modular

El ejemplo que se muestra el Listado 2 de *playingModules.c* es una versión ampliada del original [2] para producir sonido a partir de alguno de los formatos que se ha visto en la Figura 5 que son reconocidos por *libmikmod*.

En este ejemplo se han mantenido los comentarios del código original, se pueden destacar las siguientes instrucciones:

- Entre las líneas 14 a la 17 hemos incluido la lógica para que el fichero a reproducir se pase como argumento en la línea de órdenes.
- La línea 18 recoge la metainformación del fichero de audio modular cargado (nombre, tipo, comentarios, voces, instrumentos, muestras de audio, etc.), si ha sido posible hacerlo, lo que permitiría tomar decisiones al respecto en nuestra aplicación en tiempo de ejecución.
    - Los parámetros de esta orden especifican el nombre del fichero, el número máximo de canales del mezclador que utilizará este módulo y si se atenderá a ciertas propiedades particulares de algunos formatos que no están presentes en todos y es que no hay un estándar en este campo.
- Si se ha podido cargar, la línea 20 y 21 muestra cómo se pueden averiguar las propiedades del fichero como voces, canales, título, tipo o comentarios, etc.
- Si se ha podido cargar, la línea 22 muestra cómo se empieza a ejecutar el contenido sonoro del fichero.
- Entre las líneas 23 y 25 se comprueba que el fichero está siendo reproducido. Dejo a la curiosidad del lector comprobar si el valor que se le pasa como parámetro al "usleep" es posible reducirlo …
  Por cierto, en una aplicación compleja, este bucle debería ser llevado a cabo por un hilo diferente del de la ejecución de la aplicación para mantener el flujo del programa en paralelo con la reproducción del audio.
- La línea 26 muestra como parar la reproducción del sonido, también es posible pausarlo, alterar su tempo o silenciarlo, entre otras operaciones que se puede aplicar a cada recurso sonoro.
- La línea 27 muestra cómo se han de liberar los recursos específicos de un sonido modular que no se va a reproducir más. Esto permitiría cargar un número grande de los mismos y gestionar el consumo de recursos de la máquina.

La salida de este ejemplo solo muestra en pantalla un mensaje de qué fichero se está reproduciendo o que no ha podido hacerlo., Así que lo suyo es probarlo y escuchar el contenido de un ejemplo. Yo me he hecho un recopilatorio de los que tengo en el disco; pero, si no es tu caso, puedes acceder a *The Mod Archive*[4] por ejemplo.

[4] Disponible en la URL <https://audio-file.org/2019/11/20/the-mod-archive-mod-music-repository/ >.

```
1.  #include <unistd.h>
2.  #include <mikmod.h>
3.
4.  int main(int argc, char *argv[]) {
5.      MODULE *module;
6.      MikMod_RegisterAllDrivers();  // register all the drivers
7.      MikMod_RegisterAllLoaders(); // and all the module loaders
8.      md_mode |= DMODE_SOFT_MUSIC;   /* initialize the library */
9.      if (MikMod_Init("")) {
10.      fprintf(stderr, "Could not initialize sound, reason: %s\n",
11.              MikMod_strerror(MikMod_errno));
12.      return( 1 );
13.      }
14.    if ( argc == 1 ) {
15.        printf( "Uso: %s fichero\n", argv[0] );  exit( 0 );
16.    }
17.    else { /* load module */
18.     module = Player_Load( argv[1], 64, 0);
19.     if (module) {
20.        printf("Utilizando %s :\n %d veus, %d canals títul \"%s\"
tipo \"%s\" comentaris \"%s\" (elapsed-time %d mseg.).\n",
21.               argv[1], module->numvoices, module->numchn, module-
>songname, module->modtype, module->comment, module->sngtime>1024 );
22.          Player_Start(module);
23.          while (Player_Active()) {  // we're playing
24.              usleep(10000); MikMod_Update();
25.          }
26.          Player_Stop();
27.          Player_Free(module);
28.     } else
29.        fprintf(stderr, "Could not load module %s, reason: %s\n",
30.                argv[1], MikMod_strerror(MikMod_errno));
31.     MikMod_Exit();     /* give up */
32.    } // else de if ( argc == 1 )
33. }
```

Listado 2: Listado de playingModules.c [1].

Para compilarlo puede reutilizar la orden de la Figura 4, adaptándola al nombre de este fichero.

5 Conclusiones y cierre

El lector que ha seguido este documento habrá podido instalar la librería *libmikmod* en la plataforma de su elección y compilar algunos programas de ejemplo para experimentar con la librería. De manera que será capaz de reconocer la secuencia y las instrucciones básicas que permiten reproducir los ficheros de sonido modulares más populares, como MOD, IT, S3M o XM, por citar algunos.

Por brevedad de la exposición hemos dejado fuera cuestiones como la reproducción de audio en formato WAVE[5] (limitado a ficheros sin comprimir y monofónicos) o el uso de listas de reproducción.

Si se necesita realizar operaciones como la ecualización o sonido DSS (*Dolby Surround Sound*), otras alternativas como *libmodplug* y OpenMPT[6] (como secuenciador)r deberían ser valoradas. Quizás andas buscando el soporte para audio en formato MIDI, en ese caso sugiero encaminar la búsqueda hacia libtimidiy[7] … pero esa es otra historia.

Es el momento de buscar algunos ejemplos de ficheros de sonido modulares y comprobar que los ejemplos proporcionados y los indicados funcionan ¡¡ÁNIMO!!

6 Bibliografía

[1] Sito web de MikMod. Disponible en <http://mikmod.sourceforge.net/ >.

[2] Documentación de MikMod sound library. Disponible en <http://mikmod.sourceforge.net/doc/mikmod-3.3.11.html >.

[5] Ah, si se decide a probarlo (apartado “2.4 Playing Sound Effects” de []) es aconsejable bajar los “usleeo” a valores menores o iguales a 100. Compruébelo y verá por qué-

[6] Al respecto de OpenMPT se recomienda consultar <https://openmpt.org/>.

[7] Disponible en <http://libtimidity.sourceforge.net/>.