/\*

 \* This example plays every .WAV file it finds on the SD card in a loop

 \*/

#include <WaveHC.h>

#include <WaveUtil.h>

SdReader card; // This object holds the information for the card

FatVolume vol; // This holds the information for the partition on the card

FatReader root; // This holds the information for the volumes root directory

WaveHC wave; // This is the only wave (audio) object, since we will only play one at a time

uint8\_t dirLevel; // indent level for file/dir names (for prettyprinting)

dir\_t dirBuf; // buffer for directory reads

/\*

 \* Define macro to put error messages in flash memory

 \*/

#define error(msg) error\_P(PSTR(msg))

// Function definitions (we define them here, but the code is below)

void play(FatReader &dir);

//////////////////////////////////// SETUP

void setup() {

 Serial.begin(9600); // set up Serial library at 9600 bps for debugging

 putstring\_nl("\nWave test!"); // say we woke up!

 putstring("Free RAM: "); // This can help with debugging, running out of RAM is bad

 Serial.println(FreeRam());

 // if (!card.init(true)) { //play with 4 MHz spi if 8MHz isn't working for you

 if (!card.init()) { //play with 8 MHz spi (default faster!)

 error("Card init. failed!"); // Something went wrong, lets print out why

 }

 // enable optimize read - some cards may timeout. Disable if you're having problems

 card.partialBlockRead(true);

 // Now we will look for a FAT partition!

 uint8\_t part;

 for (part = 0; part < 5; part++) { // we have up to 5 slots to look in

 if (vol.init(card, part))

 break; // we found one, lets bail

 }

 if (part == 5) { // if we ended up not finding one :(

 error("No valid FAT partition!"); // Something went wrong, lets print out why

 }

 // Lets tell the user about what we found

 putstring("Using partition ");

 Serial.print(part, DEC);

 putstring(", type is FAT");

 Serial.println(vol.fatType(), DEC); // FAT16 or FAT32?

 // Try to open the root directory

 if (!root.openRoot(vol)) {

 error("Can't open root dir!"); // Something went wrong,

 }

 // Whew! We got past the tough parts.

 putstring\_nl("Files found (\* = fragmented):");

 // Print out all of the files in all the directories.

 root.ls(LS\_R | LS\_FLAG\_FRAGMENTED);

}

//////////////////////////////////// LOOP

void loop() {

 root.rewind();

 play(root);

}

/////////////////////////////////// HELPERS

/\*

 \* print error message and halt

 \*/

void error\_P(const char \*str) {

 PgmPrint("Error: ");

 SerialPrint\_P(str);

 sdErrorCheck();

 while(1);

}

/\*

 \* print error message and halt if SD I/O error, great for debugging!

 \*/

void sdErrorCheck(void) {

 if (!card.errorCode()) return;

 PgmPrint("\r\nSD I/O error: ");

 Serial.print(card.errorCode(), HEX);

 PgmPrint(", ");

 Serial.println(card.errorData(), HEX);

 while(1);

}

/\*

 \* play recursively - possible stack overflow if subdirectories too nested

 \*/

void play(FatReader &dir) {

 FatReader file;

 while (dir.readDir(dirBuf) > 0) { // Read every file in the directory one at a time

 // Skip it if not a subdirectory and not a .WAV file

 if (!DIR\_IS\_SUBDIR(dirBuf)

 && strncmp\_P((char \*)&dirBuf.name[8], PSTR("WAV"), 3)) {

 continue;

 }

 Serial.println(); // clear out a new line

 for (uint8\_t i = 0; i < dirLevel; i++) {

 Serial.write(' '); // this is for prettyprinting, put spaces in front

 }

 if (!file.open(vol, dirBuf)) { // open the file in the directory

 error("file.open failed"); // something went wrong

 }

 if (file.isDir()) { // check if we opened a new directory

 putstring("Subdir: ");

 printEntryName(dirBuf);

 Serial.println();

 dirLevel += 2; // add more spaces

 // play files in subdirectory

 play(file); // recursive!

 dirLevel -= 2;

 }

 else {

 // Aha! we found a file that isnt a directory

 putstring("Playing ");

 printEntryName(dirBuf); // print it out

 if (!wave.create(file)) { // Figure out, is it a WAV proper?

 putstring(" Not a valid WAV"); // ok skip it

 } else {

 Serial.println(); // Hooray it IS a WAV proper!

 wave.play(); // make some noise!

 uint8\_t n = 0;

 while (wave.isplaying) {// playing occurs in interrupts, so we print dots in realtime

 putstring(".");

 if (!(++n % 32))Serial.println();

 delay(100);

 }

 sdErrorCheck(); // everything OK?

 // if (wave.errors)Serial.println(wave.errors); // wave decoding errors

 }

 }

 }

}