Embedded Software Engineer - Coding Challenge

This is a coding challenge intended to showcase your coding and problem-solving abilities. There are two parts.

*** Use of external knowledge sources is permitted, however posting the question itself or parts of it to public media sites is strictly prohibited. ***

Write an application in C that compiles on Linux.

The application reads from an input file (e.g. “samples.bin”) exactly one sample every 50ms. Each sample is a 16-bit signed integer (simulated audio stream). The application tries to detect “loud noises” in the stream.

When a noise is detected, the application registers in a log file: the triggering sample number (from the beginning of the file), the exact timestamp (millisecond resolution) of the noise, and a sequence of 11 samples - such that the triggering sample is the 6th sample (so in total we should get 5 samples before trigger, the trigger sample, and 5 samples after trigger).

Once the application reaches the end of the input file - it prints out “Done” and terminates.

The output file should be a CSV file in the following format:
<trigger sample num>,<timestamp>,<sample1>,<sample2>,....,<sample11>
<trigger sample num>,<timestamp>,<sample1>,<sample2>,....,<sample11>
<trigger sample num>,<timestamp>,<sample1>,<sample2>,....,<sample11>
<trigger sample num>,<timestamp>,<sample1>,<sample2>,....,<sample11>
...

Notes

1 - Writing to file must not delay the reading of samples!
2 - Make any needed assumptions which do not violate the above instructions
3 - You are allowed to use threading
4 - Include a makefile with “all” and “clean” targets

We will review your submission as if it were production-level code. We will test for functional correctness and also inspect the code for style and overall design. We will test your submission on JSLinux. For example:

```
# make all
# ./boom samples.bin noises.log
# make clean
```

Part 2

A program has been created to generate a samples.bin file to help test your answer to Part 1. However, the program does not run as expected due to coding errors. Please inspect and test the following code and make any improvements and/or corrections that you identify. When executed, the program should produce a file called “samples.bin” containing exactly 40 samples of a sine wave at max amplitude.

The program can be compiled with the command: gcc -o samples main.c -l m

```
#include <math.h>
#include <stdio.h>

#define NUM_SAMPLES 40

static short samples[NUM_SAMPLES];

int main(int argc, char *argv[])
{
    /*
        We want to scale the output of sin() below to the max
        range that will fit into a 16-bit signed integer;
        sin() returns values in the range [-1,1], so let's
        multiply by the max short int value.
    */
    const int scale_factor = (2^15)-1; /* 32767 */

    /* Calculate the samples*/
    for (int i = 0; i < sizeof(samples); i++) {
        samples[i] = (short)(scale_factor * sin(i));
        printf("%d\n", samples[i]);
    }

    /* write the output file */
    FILE *f = fopen("samples.bin", "w");
    fwrite(samples, sizeof(short), sizeof(samples), f);

    return 0;
}
```