EE442/EE592 Real-Time Digital Signal Processing
Project #1: Sound Field Simulator
Due: 5:00pm, Friday, February 25

Assignment

The goal of this project is to process an audio signal with a linear, shift-invariant (LSI) system whose impulse response resembles that of a large concert hall. This will be implemented by combining the output of a tapped delay line (TDL) (which simulates the direct sound and early reflections) and the output of bank of parallel comb filters (CFs) which feed a serial pair of all pass filters (APFs) (which simulates reverberation). The complete theory and algorithm of the project is described in the *Real-Time Digital Signal Processing Using the Motorola DSP5630xEVM*. Students registered for EE442 will code the TDL and partial reverberator (single CF) while those registered for EE592 will code the complete reverberator; the standard assignment will only filter the right channel. Code segments listed in this description may be downloaded from

`http://www.ece.nmsu.edu/~pdeleon/EE592/Freescale_DSP56K_Code.html`

Tapped Delay Line (EE442/EE592)

The tapped delay line is shown in Fig. 2. The data for tap gains, g_k and tap spacings in samples, τ_k are adapted from J. Moorer, "About this Reverberation Business" and provided in **`sfs442.dat`** or **`sfs592.dat`**. The pass-through gain, g_0 is often called the "dry gain." Mathematically, the output of the TDL is given by the convolution

$$y[n] = \sum_{k=0}^{N} g_k x[n - \tau_k].$$

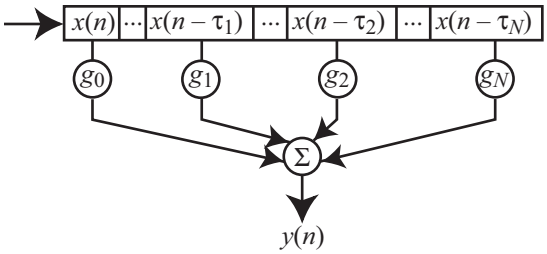

Figure 2: Tapped delay line

Partial Reverberator (EE442 only)

By adding a single CF (described below) to the TDL, we can partially simulate reverberation (see below).

Reverberator (EE592 only)

A parallel bank of four CFs followed by a series pair of APFs can produce reasonably good reverberation. With internal filter states stored as in Fig. 3, the CF can be realized in DFII (Fig. 4) with the state equations

$$\begin{aligned} w_0(n) &= x(n) + g w_m(n) \\ y(n) &= w_m(n) \\ w_k(n+1) &= w_{k-1}(n),\ k = m,\ m-1, \cdots, 1 \end{aligned}$$

while the APF can be realized in DFII (Fig. 5) with the state equations

$$w_0(n) = x(n) - gw_m(n)$$
$$y(n) = gw_0(n) + w_m(n)$$
$$w_k(n+1) = w_{k-1}(n),\ k = m,\ m-1,\cdots,1$$

(a) (b)

Figure 3: Internal filter states at time (a) n and (b) $n + 1$.

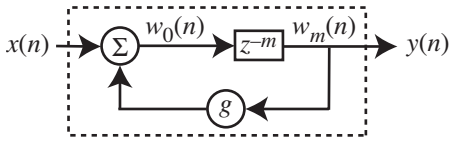

Figure 4: Comb filter (z-transform notation)

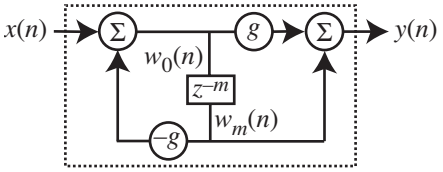

Figure 5: Allpass filter (z-transform notation)

The complete reverberator is given in Fig. 6. All filter parameters are taken from J. Moorer, "About this Reverberation Business" and provided in **`sfs592.dat`**. Note that due to memory restrictions on the DSP56302EVM, we will only implement four CFs instead of the six described in the paper.

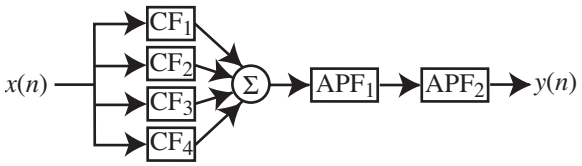

Figure 6: Reverberator: parallel bank of four CFs feeding serial pair of APFs.

Sound Field Simulator

By combining basic building blocks (the tapped delay line and the reverberator), we can produce electronic concert hall simulation with little spectral coloration. For the full simulator, feed the (partial or complete) reverberator with the last element of the delay line (as shown in Fig. 7a and 7b). The final output is the sum of the outputs of the TDL and the reverberator. You will control the mix of these two components with g_{early} and g_{late}.

Algorithm

The complete algorithm for the sound field simulator is given in the text.

Testing

One way to rigorously test is to capture the impulse response using the sound card and examine in MATLAB. You may compare your captured impulse response with the simulated response in the text. Download the **`impulse.asm`** from the web page. The code will replace the incoming input samples with an internally-generated

unit pulse train. Directions are included in the comment section for how to include it in your program. During official evaluation and grading, we will measure the delays and gains in your actual impulse response and compare with the original specifications.

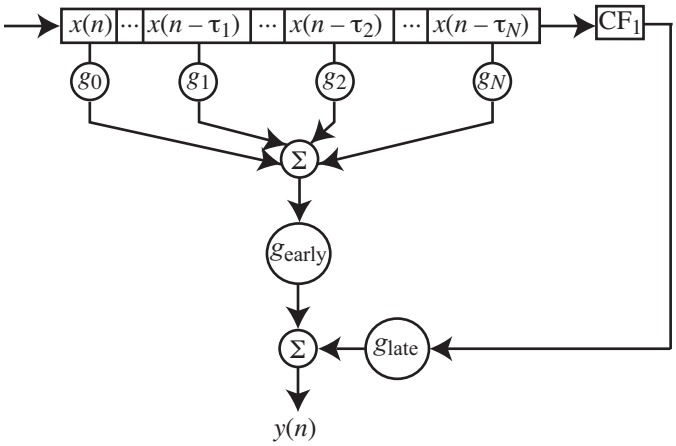

Figure 7a: Sound field simulator with partial reverberator (EE442)

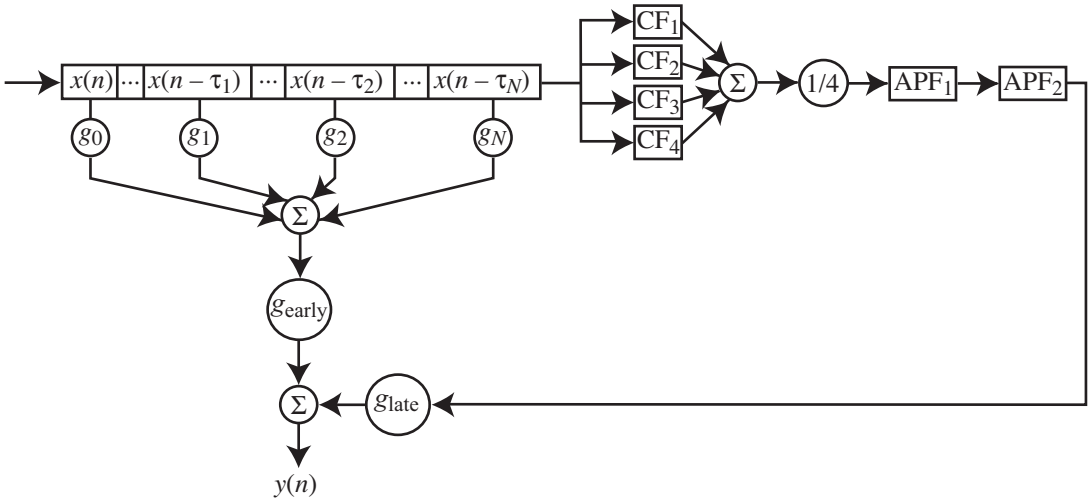

Figure 7b: Sound field simulator simulator with complete reverberator (EE592)

Submitted Items

You will be required to submit several items for this project, described below.

Code Printout

Please turn in a printout of your codes **`project1.asm`**, **`project1.dat, proginit.asm, procster.asm`**, and other programmer-created files (exclude **`ada_equ.asm`**, **`ada_init.asm`**, **`intequ.asm`**, **`ioequ.asm`**, and **`vectors.asm`**). Your codes should be fully documented in the header and completely commented.

Code in Electronic Form

Turn in a USB flash drive (with your name clearly labeled on it) with two directories: one which contains code to process audio samples and one which processes the impulse. Each directory should contain **ALL** code necessary to assemble the program—we will not struggle to get your code to assemble. Be sure to *include* the files **ada_equ.asm**, **ada_init.asm**, **intequ.asm**, **ioequ.asm**, and **vectors.asm** in each directory. Do not provide **.CLD**, **.LST**, or backup files.

Grading

You will be scheduled to demonstrate your code the week following the due date. During this time, we will copy the contents of the submitted code to a lab PC and assemble. You will have a short period of time to establish your setup and to demonstrate the impulse and audio responses. A working code base which gives correct responses will yield +95 points; incorrect responses will scale the point total accordingly. In addition, readability will be evaluated and scored up to an additional +5 points (see below). In the event your code does not function properly, you will have a short period of time to demonstrate what is working through listening tests and/or stepping us through your code listing.

A clearly defined DAT file, excellent commenting, and a well-structured (good use of subroutines) code will further increase score by up to +5.

Addresses are not to be hard-coded (–1 point deduction), i.e.

```
   move x0,x:MYVAR   ;acceptable: readable and portable
   move x0,x:$123    ;not acceptable: not readable and not portable
```

If an array or variable is declared in X-memory space it may not be used for Y-memory access and vice versa, i.e.

project.dat

```
 org x:$100
MYQ   dsm   10
```

project.asm

```
move   x0,x:MYQ   ;acceptable: readable code
move   x0,y:MYQ   ;not acceptable: not readable (based on DAT file)
```

Bonuses

You may wish to earn bonus points with the following enhancements to the project. Note that we will only grade one program be it the basic project or enhanced project. It is far better (point-wise) to have a working basic project than a non-working enhanced project.

EE442

1) Submit your code 1 week early (+5 points)
2) Add the complete reverberator to your right tapped delay line. (+20 points)
3) Add the tapped delay line and complete reverberator for the left channel. (+10 points).

Maximum grade for EE442, Project #1 is 135 points.

EE592

1) Submit your code 1 week early (+5 points)
2) Add the tapped delay line and reverberator for the left channel. (+5 points).

Maximum grade for EE592, Project #1 is 110 points.

Evaluation

EE442

Baseline: One TDL and one CF for right channel +95 points. Good commenting adds +1 to + 5 additional points for a maximum of +100 points. Average commenting and documentation will be worth +3 points.

Deductions:

–2	No initial clearing of buffers
–5	On impulse response, TDL missing last tap
–5	Tap gains in TDL are incorrect
–10	Tap spacings in TDL are incorrect
–10	Single CF does not have proper response
–10	Metallic or other noise distortion
–40	No Assemble

EE592

Baseline: TDL and Reverberator right channel +95 points. Good commenting adds +1 to + 5 additional points for a maximum of +100 points. Average commenting and documentation will be worth +3 points.

Deductions:

–3	No initial clearing of buffers
–5	On impulse response, TDL missing last tap
–5	Tap gains in TDL are incorrect
–10	Tap spacings in TDL are incorrect
–10	Single CF does not have proper response
–5	Accumulation of CF outputs is not handled correctly
–10	Metallic or other noise distortion
–40	No Assemble