電気電子情報第一（前期）実験

I1. 情報: 第1部

考察レポート

03-160441　土屋潤一郎（工学部 電子情報工学科）
共同実験者: 井上友貴、田中大幹、坂口達彦（第 28 班）

2016 年 6 月 8 日

1.　実験の概要

第 1 日 Linux コマンド及び Emacs の使用法の復習、C 言語プログラムのデバッグを通してのデバッガの使用法の復習
第 2 日 低水準入出力関数を用いたファイルの入出力、raw 形式を主とした音声データの取り扱い
第 3 日 実習時間
第 4 日 ダウンサンプリングによるナイキスト周波数の実感、高速フーリエ変換を用いたデジタルフィルタの作成

2.　考察

2.1　低水準入出力関数

第 2 日に、低水準入出力関数を用いてファイルの読み書きを行った。低水準入出力関数では、OS のシステムコールを直接呼び出すことでファイルを取り扱う。重要なのは、C 言語に標準的に用意されている高水準入出力関数と異なり、メモリにバッファを用意しない。このため、これらの命令が何度も繰り返されるような場面では、何度もディスク（の目的のファイル領域）へのアクセスが発生し低速になるが、一方で、複数のプロセスが一つのファイルを同時に扱うような場合、入出力が意図した時系列順に行われる。従って、一連のメモリ領域に一度にデータを溜め込みたい場合や、逆に一連のメモリ領域から一度にデータを吐き出したい場合には、低水準入出力関数を用いて可能な限り意図した時系列で動作を行わせることができる。また、低水準入出力関数では、そのサイズ指定をバイト単位で行うことから、よりバイナリを意識することが必要である。これらのうち、入出力メモリバッファの有無を確認するために、簡単な比較実験を以下のように行った。

特定のファイルをオープンし、そのファイルに（配列に格納した）10^6byte の 0 を書き込む。これを、

1. 低水準（write）・1byte ずつ（つまり、write を 10^6 回呼び出す）
2. 低水準（write）・一度に 10^6byte まとめて
3. 高水準（fwrite）・1byte ずつ
4. 高水準（fwrite）・一度に 10^6byte まとめて

という条件で行い、C 言語標準の<time.h>を用いて時間を計測する。その結果が以下の通りである。

1. 1127.59 ms
2. 9.45 ms
3. 31.29 ms
4. 9.46 ms

やはり低水準出力関数を複数回実行するとその分だけ時間がかかっていることがわかる。高水準出力関数を用いてもこの傾向は見られるが、これは出力のメモリバッファがプログラム実行途中でいっぱいになってしまい、複数回書き込みをしているためだと考えられるが、その回数は低水準出力関数を用いた時よりも明らかに少ないことがわかる。

2.2　ダウンサンプリングとナイキスト周波数

第 4 日に、正弦波をサンプリングした値列から、n 個に一個の割合で標本を抜き出してきて新たに標本値列を作成した。これは、標本化周波数を n^{-1} 倍するのに等しい。この際、標本化周波数の 1/2 の値を持つナイキスト周波数以上の周波数は正しく復元できない。

そこで今回は、基本の A の 2 オクターブ上の A、すなわち 1760 Hz の正弦波の 2 種のダウンサンプリングを行った。標本化周波数を $\frac{44100}{10} = 4410$[Hz] にする、標本値列から 10 個に 1 個の割合で値を抜き出してくる方法と、標本化周波数を $\frac{44100}{15} = 2940$[Hz] にする、標本値列から 15 個に 1 個の割合で値を抜き出してくる方法とである。以下図 1 がその様子である。

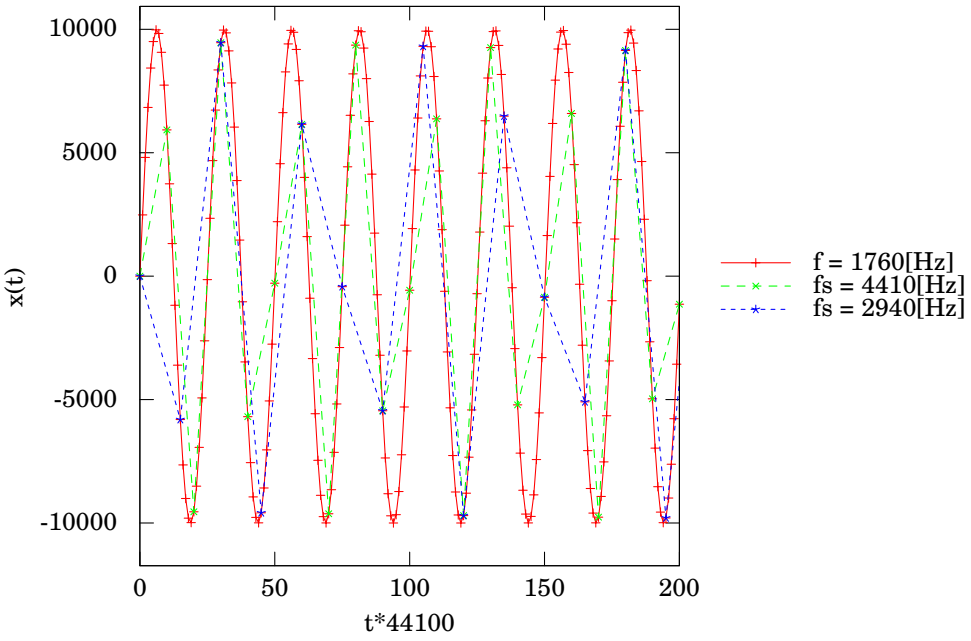

図 1　ダウンサンプリングの結果

緑の破線が示す、標本化周波数 4410 Hz のグラフは、もとの赤い線が示す正弦波の山と谷を捉えきれている、と言える。一方、青の破線が示す、標本化周波数 2940 Hz のグラフは、もとの赤い線が示す正弦波の山と谷を明らかに捉えきれていない。1760 Hz をナイキスト周波数として復元可能域に含めるためには、標本化周波数は 3520 Hz でなければならないことを、（簡易的にであるが）確認することができた。また、人間の可聴域の上限がおおよそ 20.000 Hz なので、CD などの標本化周波数は 44.100 Hz が用いられるが、今後音声電話を作成していく際には、これにとらわれず、ある程度（周波数フィルタによって）高音域を諦めることで標本値列のデータ量を減らせることがわかる。

3.　参考資料

1. 東京大学工学部電子情報工学科・電気電子工学科（2016）『電気電子情報第一（前期）実験テキスト』