Algoritmos y Programación II
Trabajo Práctico 0

Carlos Germán Carreño Romano, Sebastián Sampayo, Rodrigo Bourbon

April 23, 2015

Contents

1 Enunciado

2 Introducción

2.1 Radio definida por software (SDR)

El concepto de Radio Definida por Software se le atribuye a Joseph Mitola, 1990. Se refiere a un dispositivo que permite reducir al mínimo el hardware necesario para la recepción de señales de radio. Dicho equipo captura la señal analógica (ya sea mediante un cable o una antena), la digitaliza (mediante un conversor A/D) para luego realizar por software toda la etapa de procesamiento de señal requerido en la decodificación. Esto ha logrado que la recepción de cierto rango de telecomunicaciones sea mucho más accesible en términos económicos y prácticos (ya que el mismo dispositivo físico se puede utilizar para distintos fines con solo re-programar el software). Un ejemplo de este dispositivo se puede ver a continuación:

2.2 Transmisión de TV por cable

En telecomunicaciones, la televisión analógica se transmite mediante el método de la Multiplexión por División en Frecuencia (FDM). Esta técnica consiste en

transmitir varias señales simultáneamente modulando cada una con una portadora diferente, en el rango de VHF/UHF, de forma tal que los anchos de banda de cada señal no se superpongan significativamente. El canal destinado para la transmisión de una emisora tiene un ancho de banda de aproximadamente 6 Mhz, donde los 5.45 Mhz más bajos corresponden al espectro de la señal de video y los últimos 0.55Mhz (aproximadamente) se reservan para el espectro de la señal de audio. Este modelo de comunicación se puede ver representado en el siguiente gráfico:

2.3 Aplicación del Trabajo Práctico

Sabiendo que el audio de la televisión está modulado en frecuencia (FM), si se toma la porción del canal adecuada es posible demodular dicha señal y escuchar algún canal de televisión.

En este caso particular, el SDR se utilizó para capturar un ancho de banda de 2.4Mhz y centrado en 181.238 Mhz. A través del aplicativo desarrollado se pudo escuchar efectivamente el programa emitido.

3 Introducción

Adoptamos la convención de code styling de Google para C++, salvando las siguientes excepciones:

- streams: utilizamos flujos de entrada y salida
- sobrecarga de operadores
-

4 Desarrollo

5 Códigos

main.cc

Los códigos no deben tener **texto acentuado**.

```
1 //
    ------------------------------------------------------------------------------
    //
2 // Facultad de Ingenieria de la Universidad de Buenos
    Aires
3 // Algoritmos y Programacion II
4 // 1er Cuatrimestre de 2015
5 // Trabajo Practico 0: Programacion en C++
6 // Demodulacion de senal FM
7 //
8 // main.cc
9 // Archivo principal donde se ejecuta el main.
```

```
10 //
       ------------------------------------------------------------------------
       //
11
12 #include <iostream>
13 #include <fstream>
14 #include <sstream>
15 #include <cstdlib>
16 #include <cstring>
17
18 #include "main.h"
19 #include "complex.h"
20 #include "cmdline.h"
21 #include "arguments.h"
22 #include "utilities.h"
23 #include "types.h"
24
25 using namespace std;
26
27
28 // Coleccion de funciones para imprimir en formatos
       distintos
29 void (*print_phase[])(double) = {
30
31   print_phase_text,
32   print_phase_U8
33
34 };
35
36 // Opciones de argumentos de invocacion
37 static option_t options[] = {
38   {1, "i", "input", "-", opt_input, OPT_SEEN},
39   {1, "o", "output", "-", opt_output, OPT_SEEN},
40   {1, "f", "format", "txt", opt_format, OPT_SEEN},
41   {0, },
42 };
43
44 extern istream *iss;
45 extern format_option_t format_option;
46
47 int
48 main(int argc, char * const argv[])
49 {
50   size_t i = 0;
51   size_t j = 0;
52   Complex buffer1 = 0;
53   double buffer2 = 0;
54   Complex x, x_prev;
55   Complex aux;
56   double output_phase;
```

```
57
58    // Parsear argumentos de invocacion
59    cmdline cmdl(options);
60    cmdl.parse(argc, argv);
61
62    // Inicializar el complejo con el fomato especificado
63    Complex input_complex(format_option);
64    // cout<<int(input_complex.format_option())<<endl;
65
66    // ( -- Condiciones Iniciales Nulas -- )
67    x_prev = 0;
68
69    // Mientras haya complejos en la entrada
70    while (*iss >> input_complex)
71    {
72      // ( -- Promediar 11 elementos -- )
73      buffer1 += input_complex;
74      if (i < DECIMATOR1_SIZE-1)
75      {
76        i++;
77        continue;
78      }
79      x = buffer1/DECIMATOR1_SIZE;
80      buffer1 = 0;
81      i = 0;
82
83      // ( -- Obtener la derivada de la fase -- )
84      aux = x * x_prev.conjugated();
85      output_phase = aux.phase();
86
87      // ( -- Avanzar una muestra -- )
88      x_prev = x;
89
90      // ( -- Promediar 4 elementos -- )
91      buffer2 += output_phase;
92      if (j < DECIMATOR2_SIZE-1)
93      {
94        j++;
95        continue;
96      }
97      output_phase = buffer2/DECIMATOR2_SIZE;
98      buffer2 = 0;
99      j = 0;
100
101     // ( -- Imprimir en el formato especificado -- )
102     (print_phase[format_option])(output_phase);
103   }
104   return 0;
105 } // main
```

Figure 1: Sintonizador de radio digital.

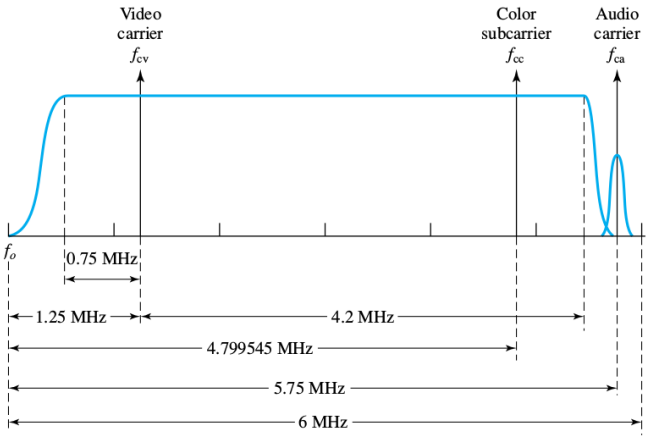

Figure 2: Señal de TV transmitida.