Twelfth Quarterly Progress Report

NIH-NO1-DC-6-0002

Open Architecture Research Interface for Cochlear Implants

Nageswara R Gunupudi, Arthur Lobo[†], Douglas Kim, Rohith Ramachandran,
Vani Gopalakrishna, Nasser Kehtarnavaz and Philipos C. Loizou

Department of Electrical Engineering
University of Texas-Dallas
Richardson, TX 75080
Email: loizou@utdallas.edu

[†] Signals and Sensors Research, Inc, McKinney, TX

January 1, 2009 – March 31, 2009

1. Introduction

The main aim of this project is to develop a research interface platform which can be used by researchers interested in exploring new ideas to improve cochlear implant devices. This research platform includes a stimulator unit which can be used for electrical stimulation in animal studies, a recording unit for collecting evoked potentials from human subjects and a portable processor for implementing and evaluating novel speech processing algorithms after long-term use. The research platform chosen for this project is the personal digital assistant (PDA).

2. Summary of activities for the quarter

Work in this quarter focused on development of a MATLAB-to-PDA Winsock Interface. Such an interface would allow for offline processing of speech stimuli, as well as the design of sophisticated psychophysical experiments that would not necessarily require real-time processing. A graphical user interface was also developed to allow the user to import, save and modify patient parameter files (MAPs). Ongoing work is focused on completing the hardware design of the SDIO interface board to be used for the 8-channel animal stimulator.

2.1 Development of a MATLAB-to-PDA Winsock Interface

A MATLAB-to-PDA communication interface was implemented to transfer stimulus parameters and amplitudes of speech stimuli continuously from the desktop to the PDA for offline testing of subjects. The interface is based on Winsock (Windows Sockets API) which is a technical specification that defines how Windows network software should access network services, especially TCP/IP. It defines a standard interface between a Windows TCP/IP client application (such as an FTP client or a Gopher client) and the underlying TCP/IP protocol stack.

The Winsock interface implementation consists of three software components:

1) Winsock server running on the PDA,
2) MATLAB Winsock client .mexw32 (dll) called from a MATLAB command script,
3) MATLAB command script running on desktop.

Figure 1 shows the transfer of parameters and amplitudes from MATLAB to the PDA and status returned from the PDA to MATLAB.

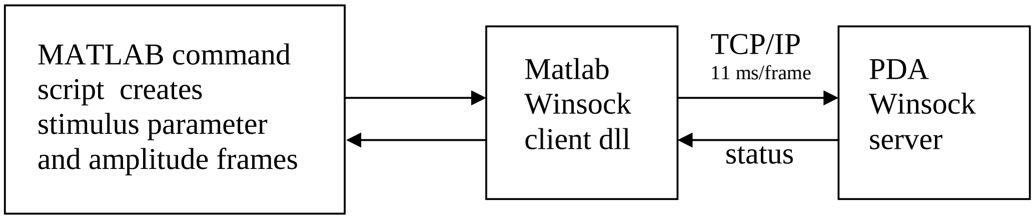

MATLAB command script creates stimulus parameter and amplitude frames
Matlab Winsock client dll
TCP/IP
11 ms/frame
status
PDA Winsock server

Figure 1. MATLAB/PDA Winsock Interface.

The PDA component initializes Winsock, creates a socket, binds the socket, “listens” on the socket, accepts incoming connections, and performs blocking receives to receive the parameter and stimulus amplitude data from the client. The “receive” is performed within a thread in two steps. In the first step, information about the total number of 11 ms frames, denoted as *nframes*, to be transmitted and the number of pulses per frame is received. The server then performs *nframes* receives, each time sending the data to the SDIO board. After *nframes* receives and sends, the server closes the socket and the connection. The PDA component is built as a Windows Mobile 5.0 executable using Visual Studio 2005 Professional. The executable is deployed on the PDA and run from the desktop remotely using the Windows Remote API (RAPI) application *prun*. The server needs to be started on the PDA before the client (discussed below) is run. Furthermore, the connection between the PDA and the desktop is switched to RNDIS Sync Mode from the default USB Serial Sync Mode to allow network connectivity.

The MATLAB client dll initializes Winsock, creates a socket, connects to the server and transmits parameter and stimulus amplitude frames. It does this in two steps to match the receive function on the server: first the number of 11 ms frames, *nframes*, and the number of pulses per frame are transmitted. Second, *nframes* frames are transmitted continuously with the time interval between frames set to 11 ms. The dll is compiled from the C source using the MATLAB 7.7 MEX compiler.

The third component is the MATLAB script. The following MATLAB code shows as an example how the parameter and stimulus frames are built and how the call to the Winsock client dll (called socclient) is made to transfer the data to the PDA:

```
% Parameters rate = 1100 pps, pw = 35 uS are kept constant for all frames
% for illustration
% First pulse in each frame every channel is 127
% Rest of the pulses are 255

nchannels = 12;

nframes = 364;      % 4 seconds @ 11 ms per frame

% Create parameter frames
parm = zeros(nframes*2,1);
j=1;
for i=1:nframes,
  parm(j) = 1100;       % rate (pps)
  parm(j+1) = 35;       % pulse width (μSecs)
  j = j+2;
end

% Create stimulus frames
L_amp=[];
x = 127.*ones(nchannels,1);
L_amp = x;
x = 255.*ones(nchannels,1);
pulses_per_frame_per_ch = 12;

for i=1:pulses_per_frame_per_ch-1
 L_amp = [L_amp; x];
end

x = L_amp;
```

```
for i=1:nframes-1,    % 4 seconds of stimulation
 L_amp = [L_amp; x];
end

R_amp = L_amp;        % For illustration purposes, the right ear stimulation
                      % is set to be the same as  left ear stimulation


pulses_per_frame = nchannels * pulses_per_frame_per_ch;

frames_data = [nframes pulses_per_frame];

socclient(parm, L_amp, R_amp, frames_data);
```

Before calling the `socclient(.)` again with a new set of stimulus data, a minimum pause of 1.3 ms needs to be inserted. The PDA server will automatically initialize a new connection for the next incoming stimulus and wait for the client to transfer the corresponding next set of parameter and amplitude frames. In this way the transfer of amplitudes takes place on demand i.e. the transfer is made under the complete control of the user. As illustrated in Figure 1 communication from the server back to the client and MATLAB has been implemented so as to confirm to the subject that a complete speech stimulus has been played.

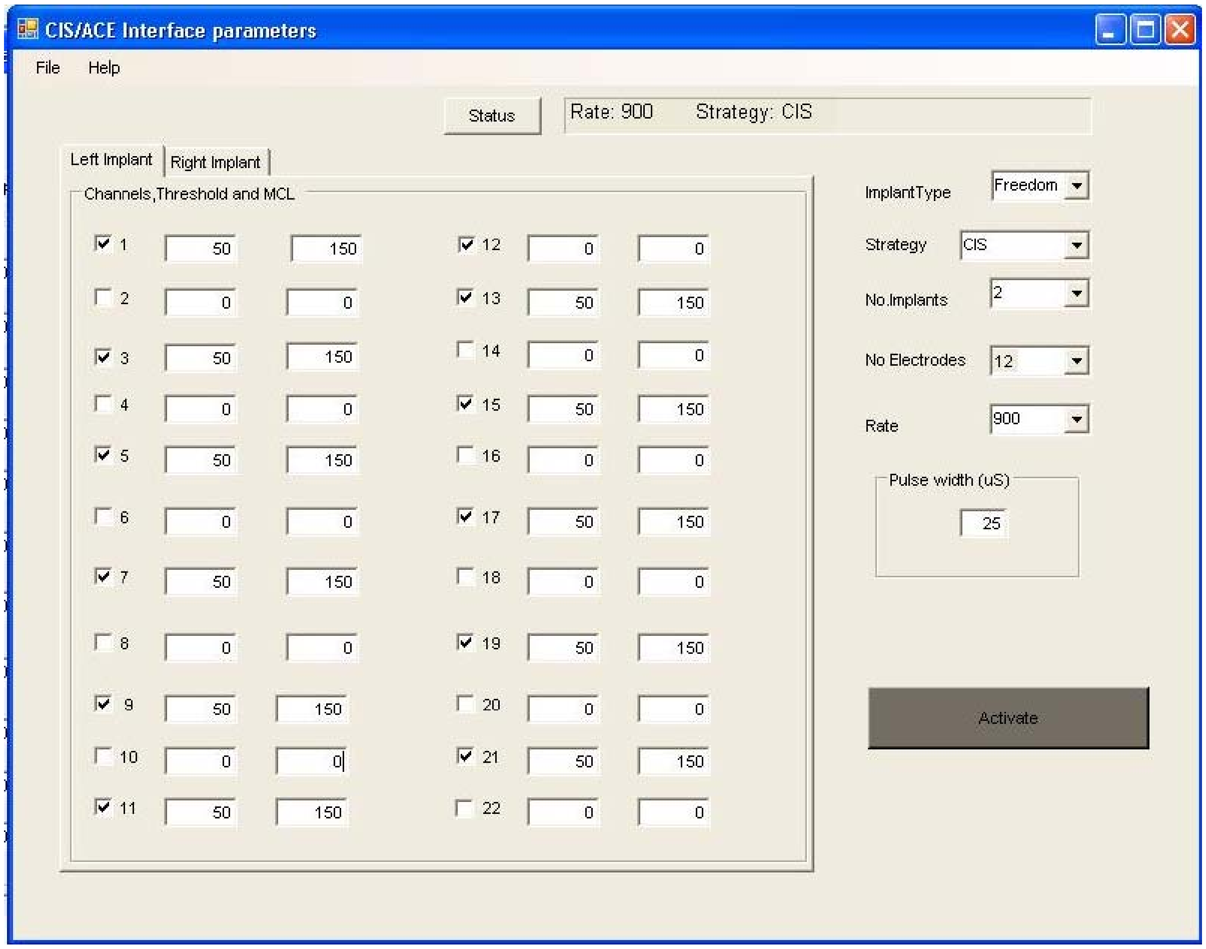
2.2 Design of a CIS/ACE Parameter Interface

A graphical user interface (GUI) was built (using VisualBasic) to allow the user to import, save and modify patient parameter files (MAP) to be used in the PDA code. The GUI allows the user to create a new patient file, as well as load existing patient files. Figure 2 shows an example snapshot of the developed GUI.

The patient file has a pre-defined format, and includes, among others, the following parameters:

-Strategy, Left Implant: [ACE, CIS]
-Strategy, Right Implant [ACE, CIS]
- Number of maxima, Left (only for ACE)
- Number of maxima, Right (only for ACE)
-Electrode configuration, Left [MP1, MP2, MP1+2]
-Electrode configuration, Right [MP1, MP2, MP1+2]
-Stimulation rate per channel, Left
-Stimulation rate per channel, Right
- Pulse width- Left
- Pulse width- Right
-Threshold values, 22, Left
- Most comfortable levels, 22, Left
-Threshold values, 22, Right
- Most comfortable levels, 22, Right

To accommodate bilateral users, the GUI has two tabs (see Figure 2), one for the left implant and one for the right implant. The user has the flexibility to turn on/off individual channels, by checking the appropriate box located left to each channel (see example in Fig. 2).



Rate: 900 Strategy: CIS

Channels,Threshold and MCL

✓	1	50	150	✓	12	0	0
☐	2	0	0	✓	13	50	150
✓	3	50	150	☐	14	0	0
☐	4	0	0	✓	15	50	150
✓	5	50	150	☐	16	0	0
☐	6	0	0	✓	17	50	150
✓	7	50	150	☐	18	0	0
☐	8	0	0	✓	19	50	150
✓	9	50	150	☐	20	0	0
☐	10	0	0	✓	21	50	150
✓	11	50	150	☐	22	0	0

ImplantType	Freedom
Strategy	CIS
No.Implants	2
No Electrodes	12
Rate	900
Pulse width (uS)	25

Figure 2. GUI developed for importing and creating patient's MAP files.

The GUI interface allows the user to edit parameters such as pulse width, stimulation rate, etc. If an erroneous value or an out-of-range value is entered, an error message is issued. Error checking is performed for all parameters entered to ensure that the parameters fall within the range supported by the commercially available Nucleus Freedom (or older generation) processor. Finally, the `activate` button is used for downloading the patient file to the PDA