Cognitive Neuroscience of Language Laboratory
Department of Psychology, University of Connecticut

Methods Report 2006.01
Original version, 2006

Sending signals from E-Prime to the BioSemi EEG system

Dan Mirman
daniel.mirman@uconn.edu

Abstract
This briefly describes how to use InLine objects to send signals from E-Prime to the BioSemi EEG system. The signals are integers that are sent over the parallel port. In addition to basic parallel port communication, a few tips/hints are included.

Sending signals from E-Prime to the BioSemi EEG system

Communicating using the parallel port

1. **Define convenient names for ports.** In the "User" section of the script (you can get there by hitting Alt+5 and selecting the User tab) add the following lines:

   ```
   Const BASE_ADDRESS = 888
   Const OUT_PORT_ADDRESS = BASE_ADDRESS
   Const IN_PORT_ADDRESS = BASE_ADDRESS + 1
   ```

2. **Send your signal.** The signal must be an integer (*val*). Use the following command in an InLine:

   WritePort OUT_PORT_ADDRESS, *val*

Tips and hints

1. **Use pulses.** BioSemi records a parallel input until a new input is sent, which makes e.g., overlapping experimental events difficult to record. The recommended to solution (and recommended practice) is to use pulses of information to signal stops and starts of events. To do this send a signal, pause, then turn the signal off. Note that "off" is a signal in itself (e.g., 0). Here is the code to do this (using a 10ms pulse):

   ```
   WritePort OUT_PORT_ADDRESS, val
   Sleep 10
   WritePort OUT_PORT_ADDRESS, 0
   ```

2. **Send information that will be useful for analysis.** Since the EEG analysis will take place on the BioSemi control computer, it will be useful to have the key experimental data encoded alongside the EEG recording. For example, to analyze only correct-response trials and compare across conditions, you could send condition-identifying information at the start of a trial and accuracy information at the end of a trial. Your trial procedure should look like this:

   startSignal (an InLine that sends the trial start signal)
   trial procedure (for the example below, the trial list should include an attribute called *Condition* and the procedure should include an object called *target* which is the target presentation and which records accuracy)
   stopSignal (an Inline that send the trial stop signal)

   startSignal should look like this:

   ```
   WritePort OUT_PORT_ADDRESS, c.GetAttrib("Condition")
   Sleep 10
   WritePort OUT_PORT_ADDRESS, 0
   ```

   stopSignal should look like this:

   ```
   WritePort OUT_PORT_ADDRESS,  target.ACC+1
   Sleep 10
   WritePort OUT_PORT_ADDRESS, 0
   ```

Note that the "off" signal is the same as the default "incorrect response" signal (ACC=0), to avoid this problem accuracy has been re-coded from [0,1] to [1,2]. However, note that conditions numbered 1 or 2 will (I think) be confused during analysis with the stop signals. The upshot here is that you should be careful about how you code your information and what you send to BioSemi. Integers from 0 to 255 are legal (I think).

Known issues

1. **Synchrony between trial onset/offset signals and actual trial onsets/offsets.** If the trial procedure is structured according to the simple schematic above, after the TrialStartSignal is sent, E-Prime will still need to load stimuli, refresh the screen, etc. before the trial actually begins.
    a. It is not clear to me how bad this problem is likely to be, so one recommended step is to test it by creating a dummy experiment in which a Start and Stop signal are separated by a fixed duration stimulus and seeing the duration between the signals on the BioSemi side.
    b. One way to resolve this problem is to send the actual stimulus onset time (it is recorded by E-Prime and available at the end of the trial). However, since E-Prime will record the time in milliseconds with respect to some internal starting point, this value is going to be greater than 255 and thus tricky to send and inelegant (perhaps impossible) to analyze from BioSemi.