IPS Lida notes

(Phil Hoole; last update 4.9.2010)

Unless indicated otherwise the following matlab functions are designed to be run on the control server of the AG500 system.

GETRAW4SHOWZ Record rawdata as complex amplitudes on AG500 linux
  function getraw4showz(mytrial,mychans,calpath)

Basic use: Test that sensors are intact before starting calibration.

Calls the two BC programs/shell scripts getrawdata and sincosdemod to store complex demodulated amplitudes in a file:

```
mystat=system(['getrawdata | sincosdemod ' calstring ' > ' trialstring
'.camp'])
```

Then calls the matlab function showz. This shows the data in the z-plane (cosine and sine component of the amplitude) for each combination of sensor and transmitter.
Rationale: May be easier to tell whether a sensor is starting to misbehave in this kind of display than in the final amplitude signal.

CHECKAMP: Continuous display of amplitudes for Linux AG500 system
 function checkamp(totaldur,chunkdur,chanlist,myfunc,calpath)

Basic use: Simply make sure that sensors are connected and receiving a signal

Uses essentially the same BC programs as getrawdata4showz, but illustrates how the existing BC shell script getrawdata can be easily modified to record for a fixed length of time, rather than waiting until the user hits the 'enter' key.
(In fact, there is now a BC script called getrawdatatime that probably has basically the same functionality. The listing of getrawdatawait is given below.)

```
    [mystat,myresult]=system(['getrawdatawait ' num2str(chunkdur) ' |
sincosdemod ' calstring ' > tmp.camp']);
```

LIDA_RTMON RT AG500 display; copy trials from lida to control server
 function lida_rtmon(recpath,displaychans)

The matlab program opens a socket to a specific TCP/IP port on the Lida host.
Via this connection it has access to the same kind of real-time data on which the BC CS5VIEW program is based (this matlab program can only be run when the BC CS5RECORDER program is also running).

Based on this real-time data stream it has the following two main functions

(1) Real-time display of positions
This can be used as a platform for experimenting with alternative ways of displaying the real-

time data compared to the BC CS5VIEW program.
Two examples:
1. Unlike the BC program it also shows the RMS signal amplitude of each sensor (so gives a quick indication if a sensor is broken).
2. When a sweep is in progress, the time in seconds since the start of the sweep is shown (could be useful when recording long sweeps in order to know when the maximum sweep length will be reached)

If used just for display the matlab function can be run on any machine on the network that has access to the control server.

(2) Automatic copying of files from Lida
The real-time data stream includes a flag indicating whether a sweep is currently being recorded, and what the number of that sweep is.
If the matlab program detects that a sweep has been completed, then it copies the .amp file of that sweep from Lida (where it is initially stored) to any subdirectory on the control server (below /srv/ftp/data/).
This means that it is not necessary to wait until the end of the session and to use the BC shell script getsessionfromlida in order to have access to the data.

One reason why this could be important is that it allows much more rigorous checks on the quality of the raw amplitude data.
For example, when a sensor is getting near to the end of its life (but not yet completely broken) then occasional dropouts can occur. In other words, the amplitude may drop to nearly zero for a few samples and then come back again. This may well not be noticeable in the real-time display.
At the end of each sweep lida_rtmon loads the amp file and determines the maximum and minimum amplitude for each sensor, and displays it on the terminal.

Notes on the implementation:
To do the file copy lida_rtmon passes to the operating system a slight variation on some of the code in the BC shell script getsessionfromlida: The original version does a wild-card copy whereas lida_rtmon copies a specific file (determined by the current sweep number):

```
    [mystat,myresult]=system(['echo "/srv/data/' trials '.* ' amppath '" | rcplidaop']);
```

PROMPTER_PARALLEL_TRIGAG
This is a matlab program that is used in a number of different variants (not just for experiments with the AG500) for giving prompts to the subject (usually text, but can also be images or audio), and for keeping a log of the course of the experiment.

This program will be run on a separate machine from the control server. It actually has to be a Windows machine because it uses matlab’s Data Acquisition Toolbox to access the Sybox status signals via the parallel port (the DAQ toolbox is only available for Windows).

When used with the earlier Windows version of the AG500 it monitored the status signals from the Sybox in order to determine when to give the subject the signal to start speaking and when to switch to the next prompt, but the AG500 recorder program still needed to be started

by hand.
The newer version now accesses the same real-time data stream used by lida_rtmon (in order to determine the number of the current sweep in the AG500 system) and uses one small additional function to start and stop the BC CS5RECORDER remotely (again using the TCP/IP connection).

Thus running of the experiment can be controlled from a single machine.

Further information
(1)
The main function used by lida_rtmon and prompter_parallel_trigag for accessing the real-time data stream is getemaall.m

(2)
The modified shell script getrawdatawait is currently not part of the subversions system and is not located on one of the standard matlab paths.
Its location on the control server is currently home/csop/bin.
This is its listing:

```
#!/bin/bash
if [ -z $1 ]; then RECTIME=1
else
  RECTIME=$1
fi
echo "recording rawdata for $RECTIME s." >&2
netcat $LIDAHOST 50006 & pid=$!
sleep $RECTIME
kill $pid
```