Subject: Re: Real time signal processing using PV-WAVE
Posted by williamm on Mon, 15 Nov 1999 08:00:00 GMT
View Forum Message <> Reply to Message

In article <382f0f43.282258890@news.remarq.com>, dibanella@usa.net (Anna Maria di Banella) wrote:

> I am a new user of PV-WAVE and I'm in trouble because I have
> to process signals in real time but I have no Iidea about the way to
> do this. Does anyone know if it is possible to acquire data directly
> using PV-WAVE and process them, for example, every two or three
> seconds ?

Sure thing. There are several options depending on exactly what data gathering scenario you have in mind. I am assuming that you have some hardware-based data collection system that has an associated API for hardware control and data-stream output.

Given this assumption, the fastest and most direct way to feed data to PV-WAVE for processing is via the CWAVEC/OPI API. A second means would be to direct output from a control process to a pipe and have PV-WAVE read from that pipe.

CWAVEC is a PV-WAVE/C language command interface that allows you to directly call PV-WAVE from C for command execution. PV-WAVE itself is statically linked to your executable, which makes for a _large_ executable size, but affords the advantage of memory sharing: the C executable has direct access to the memory associated with variables in PV-WAVE via the OPI (Option Programming Interface) API. The OPI functions allow the programmer to get pointers or handles to PV-WAVE variables. Thus a programmer can create a FLTARR via a command call to the cwavec function (or more directly, the wave_execute function), then get a pointer to the PV-WAVE variable associated with that FLTARR via the wave_get_WVH function, and then make assignments to that PV-WAVE variable via the wave_assign_num function. The above OPI functions, and many more, are covered in detail in the hard-copy or on-line document: "GUI Application Developer's Guide". This memory-assignment process has the advantage of being much faster than pipe I/O.

Once you have made the relevant memory assignments, and thus have your data tucked in a PV-WAVE variable, subsequent calls to wave_execute can be initiated to call PV-WAVE functions to perform analysis on that variable or variables. Of course, if you are repeating this process

every two to three seconds, you would want to do some sort
of buffering before calling any analysis once your data
set gets very large. You certainly don't want to be calling
an FFT on a 1 x 10^6 element FLTARR every 3 seconds--you'll
certainly run out of memory and bog performance down immensely.

If you prefer pipes, check out the EXEC_ON_SELECT procedure.
It allows PV-WAVE to initiate a callback that can contain
I/O and analysis upon detecting I/O activity on a named
pipe. This process would be simpler to program, but slower
than the OPI/CWAVEC route.

But, I can tell you that it can and has been done. Following
essentially the steps outlined above with CWAVEC/OPI, I helped
develop an application that fed satellite telemetry data
gathered at approximately several data points every 1-2 seconds
into PV-WAVE to be plotted on a strip-chart type graphic for
display and analysis.

Hope this helps!

Regards,
M. Williams
