Updating devGpib instrument support to ASYN

Table of Contents

Updating devGpib instrument support to ASYN

May 17, 2004

This tutorial provides basic instructions on converting 'devGpib' device support from R3.13 to R3.14/ASYN. If possible, R3.14/ASYN device support modules should be placed in the EPICS CVS repository in a directory within the modules/instrument directory. Files placed in the modules/instrument directory should be as flexible as possible to ease their use by others. For example, device support database file GPIB link numbers and addresses should be macros and all PV names should be prefixed with a macro.

1. Create instrument support development area

1. Make a directory in modules/instrument/ for your instrument. For example:
   mkdir <*epicsTop*>/modules/instrument/keithley196
2. 'cd' to the directory created in the previous step and create a set of support files by running:
   <*asynTop*>/bin/<*EpicsHostArch*>/makeSupport.pl -t devGpib <*InstName*>
   For example:
   /usr/local/EPICS/modules/soft/asyn/bin/linux-x86/makeSupport.pl -t devGpib Keithley196
3. Check configure/RELEASE and confirm that the ASYN and EPICS_BASE values specify the location of these packages on your system.
4. Replace <*InstName*>Sup/dev<*InstName*>.c with your existing instrument support.
5. Replace <*InstName*>Sup/dev<*InstName*>.dbd with your existing instrument support.
6. Replace <*InstName*>Sup/dev<*InstName*>.db with your existing instrument support, if any.

2. Modify instrument support C source file

A side-by-side display of the results of applying the following changes to a support file for a simple instrument can be found in devKeithley196.c.diff.html.

1. Remove all #include lines except #include <devCommonGpib.h>.
2. Ensure that there is a definition for DSET_AI, even if the device has no analog-in records.
3. Add #include <devGpib.h> after the #include <devCommonGpib.h> line and after all the DSET_xxx definitions.
4. Remove any DSET_xxx structure initializations. Instead of using devGpib.h, older device support created it's own DSET_xxx definitions. Remove all of these and use the devGpib.h technique
5. Remove any xxxDebug definitions.
6. Change the value of the TIME_WINDOW macro from 1/60th of a second units to seconds.
7. Change the DMA_TIME macro to TIMEOUT and its value from 1/60th of a second units to seconds.
8. Modify the command table entries:
   - Change the end-of-string value (the last element) of each command from -1 to NULL or from a single character constant to a string constant:

Convert	
From	To
-1	NULL
'\r'	"\r"

   - Remove any GPIBEOS from the command type (the second element) of each command. Input end-of-string checking is now enabled by the presence of a non-NULL end-of-string value.

- ♦ If a device support module used GPIBIOCTL it will have to be changed to use one of the new GPIBxxx command types.

9. Change the name of the device-support initialization function to init_ai.
10. Remove from the device-support initialization function any calls to devGpibLib_init(). Simply return 0 from the device-support initialization function.
11. The fields in devGpibParmBlock have been reordered and some have gone away.

    - ♦ It must be initialized in init_ai. For example:

    ```
    static long init_ai(int parm)
    {
        if (parm == 0) {
            devSupParms.name = "your device's name";
            devSupParms.gpibCmds = gpibCmds;
            devSupParms.numparams = NUMPARAMS;
            devSupParms.timeWindow = TIME_WINDOW;
            devSupParms.timeout = TIMEOUT;
            devSupParms.respond2Writes = -1;
        }
        return 0;
    }
    ```

      The `name` and `respond2Writes` values are unique to your implementation. All the rest should appear exactly as shown above.

      Older device support initialized devGpibParmBlock with a C initializer statement. This is no longer allowed.
    - ♦ The following fields no longer exist:

      debugFlag
      : This is an obsolete field since devGpib uses the asynTrace facility.

      magicSrq
      : This is an obsolete field.

      srqHandler
      : SRQs are handled differently than before. It is no longer possible for device support code to provide it's own SRQ handler for ALL gpib devices attached to the gpib interface. It is possible, as described above to register a handler for a specific gpibAddr.

      wrConversion
      : The secondary conversion routine is no longer supported. It was used if respond2Writes is >=0.
12. Remove any report function.
13. If the instrument device support contains custom conversion routines ensure that they return proper success/failure (0 or <0) values.
14. If old support makes calls to the old devGpibLib_xxx routines, they must be replaced by calls to devSupportGpib routines.
15. (Optional) Update copyright/license comments.

3. Modify instrument support database definition file

1. Check your instrument support database definition file and see if it contains an entry for analog-in (ai) record device support. Add such an entry if the file does not already contain one.
2. Add the line: `include "asyn.dbd"`

4. (Optional) Create an example database

1. To work well with the ASYN tools the example database should abide by the following conventions:
    - ♦ All record names should begin with (P)(R).
    - ♦ All GPIB INP/OUT fields should specify the link value as L$(L).
    - ♦ All GPIB INP/OUT fields should specify the address value as A$(A).

   If your instrument already has an associated database file (or files) it may be useful to adapt it to follow these conventions.