Single-Value Analog Input

The following code fragments illustrate the steps required to perform a single-value analog input operation. Refer to the example program svadc.c in the directory C:\Program Files\Data Translation\Win32\SDK\Examples\ for the entire program.

This program calls a user-defined function called GetDriver(), which enumerates the devices installed in the system.

Declare Variables and User Functions

This code fragment defines the variables used and the user-defined GetDriver() function; note that this program uses the device’s default values for channel type, resolution, data encoding, range, and channel filter.

typedef struct tag_board {
    HDRVR hdrvr;        /* driver handle */
    HDASS hdass;        /* subsystem handle */
    ECODE status;       /* board error status */
    char name[STRLEN];  /* string for board name */
    char entry[STRLEN]; /* string for board name */
} BOARD;

typedef BOARD FAR* LPBOARD;

static BOARD board;

BOOL __export FAR PASCAL GetDriver(lpstrName,lpstrEntry,lpParam)

    LPSTR  lpstrName;     /* board name */
    LPSTR  lpstrEntry;    /* system.ini entry */
    LPARAM lpParam;       /* optional user data */
    UINT channel = 0;
    DBL gain = 1.0;
DBL min, max;
float volts;
long value;
UINT encoding, resolution;

Initialize the Driver

The following code fragment, in WinMain(), calls the
CHECKERROR error handler macro and the olDaEnumBoards
function, which initializes the first available DT-Open Layers device.
olDaEnumBoards calls GetDriver(), which lists the name of the
device:

board.hdrv = NULL;
CHECKERROR(olDaEnumBoards(GetDriver,
   (LPARAM) (LPBOARD) &board));

This code fragment is in GetDriver() and gets the device name:

{  
LPBOARD lpboard = (LPBOARD) (LPVOID)lParam;
/* fill in board strings */
lstrcpyn(lpboard->name, lpszName, STRLEN);
lstrcpyn(lpboard->entry, lpszEntry, STRLEN);

This code is in WinMain() and checks for errors within the callback
function:

CHECKERROR (board.status);
/* check for NULL driver handle - means no boards */
if (board.hdrv == NULL){
   MessageBox(HWND_DESKTOP, "No DT-Open Layer
   boards!!!", "Error",
   MB_ICONEXCLAMATION | MB_OK);
   return ((UINT)NULL);
}
This code fragment is in `WinMain()` and initializes the device:

```c
lpboard->status = olDaInitialize(lpszName, &lpboard->hdrvr);
if (lpboard->hdrvr != NULL)
    return FALSE;
/* false to stop enumerating */
else
    return TRUE;         /* true to continue */
```

**Get a Handle to the Subsystem**

The following code fragment gets a handle to the A/D subsystem and checks for errors:

```c
CHECKERROR(olDaGetDASS(board.hdrvr, OLSS_AD, 0, &board.hdass));
```

**Set the DataFlow to Single Value**

The following code fragment sets the dataflow mode of the A/D subsystem to single value and checks for errors.

```c
CHECKERROR (olDaSetDataFlow(board.hdass, OL_DF_SINGLEVALUE));
```

**Configure the Subsystem**

The following code fragment configures the A/D subsystem and checks for errors.

```c
CHECKERROR (olDaConfig(board.hdass));
```
Acquire a Single Value

The following code fragment acquires a single analog input value from channel 0 of the A/D subsystem (using a gain of 1) and checks for errors.

```c
CHECKERROR (olDaGetSingleValue(board.hdass, &value, channel, gain));
```

Convert the Value to Voltage

The following code fragment uses the default range, encoding, and resolution of the A/D subsystem to convert the acquired value into voltage and to check for errors. Note that this step is optional.

```c
CHECKERROR (olDaGetRange(board.hdass,&max,&min));
CHECKERROR (olDaGetEncoding(board.hdass, &encoding));
CHECKERROR (olDaGetResolution(board.hdass, &resolution));

/* Convert value to volts */
if (encoding != OL_ENC_BINARY)
{
/* convert to offset binary by inverting the */
/* sign bit */
    value ^= 1L << (resolution-1);

    /* zero upper bits */
    value &= (1L << resolution) - 1;
}
volts=(float)max-(float)min)/(1L<<resolution)*
    value+float)min;

/* display value with message box */
sprintf(str, "Single Value AD Op.\nADC Input = 
%.3f V", volts);
MessageBox(HWND_DESKTOP, str, board.name, 
          MB_ICONINFORMATION | MB_OK);

Release the Subsystem and Terminate the Session

The following code fragment releases the A/D subsystem, terminates the session, and checks for errors:

CHECKERROR (olDaReleaseDASS(board.hdass));
CHECKERROR (olDaTerminate(board.hdrv));

Handle Errors

The following code fragment handles the errors from the DataAcq SDK and displays the error codes. Note that this step is optional but recommended.

#define STRLEN 80 /* String size for general text*/
        /* manipulation. */
char str[STRLEN]; /* Global string for general */
        /* text manipulation */

#define SHOW_ERROR(ecode)
MessageBox(HWND_DESKTOP,olDaGetErrorString(ecode, 
           str,STRLEN),"Error",
          MB_ICONEXCLAMATION|MB_OK);

#define CHECKERROR(ecode) \
if ((board.status = (ecode)) != OLNOERROR) \
{
    SHOW_ERROR(board.status);\
    olDaReleaseDASS(board.hdass);\
    olDaTerminate(board.hdrv);\
    return ((UINT)NULL);
}

192