Switch Matrix Controller

This gives a quick introduction to the LabWindows controller for the switch matrix. To make things as portable as possible the controller program is contained in one “c” file (SwitchMatrixControl.c” and its associated “h” file. Be sure to include the .h file in your program that will access the switch matrix. This defines the Struct that contains the switch matrix configuration data, and the user functions. A simple LabWindows application (Tester) has been created to illustrate use of the switch matrix. As with most homemade software, it works, but there is little error checking so passing bad values to routine might cause undignified crashes.

To use the switch matrix there are two data files in csv format that must be in the same directory as the application: the file “SwitchMatrixConfig.csv” that contains the configuration of the switch matrix hardware, and a file that contains the configuration data for the probe card (csv format but user definable file name). These files are best created and edited in Excel, then saved in .csv format. The first line of the switch matrix file defines the connections between input connectors and board addresses. The first cell is the label “Boards”, and subsequent cells have the addresses of the boards connected to the input connectors, in order, starting at input 1. The second line is labels and subsequent lines define how relays are connected to the card edge connect (essentially defining the output cable). The probe card file simply defines the connections of probe card pins to card edge connector. In this way the user can define which probe is “Pin 1”. In the file, the first line contains labels, and subsequent lines give pin numbers (in order starting at 1) and the corresponding card edge connection.

Now to an explanation of the control software. The measurement software should make a global instance of the c struct “SwitchMatrixConfig\_type”, and allocate memory for it. This struct will be passed to functions using the switch matrix. There are four user functions to address the switch matrix:

//Function prototypes

int initSwitchMatrix (struct SwitchMatrixConfig\_type \*SwitchMatrixConfig,

char ProbeCardConfig[]);

int switchMatrix(int inputNumber, int pinNumber, int action, struct SwitchMatrixConfig\_type \*SwitchMatrixConfig);

int resetUsedRelays(struct SwitchMatrixConfig\_type \*SwitchMatrixConfig);

int resetAllRelays(struct SwitchMatrixConfig\_type \*SwitchMatrixConfig);

The file tester.c shows simple sample of using the switch matrix. The first step is to call the “initSwitchMatrix” function, passing in the filename of the probecard definition file (include the .csv extension). This routine sets up the translation from probe card pin to relay ID, and from input to appropriate board. The other routines are used to manipulate the relays. The primary routine is “switchMatrix” which has the arguments inputNumber, pinNumber, action, and SwitchMatrixConfig. Action should be set to “Connect” or “DisConnect” (defined in the .h file), while InputNumber defines the input and pinNumber refers to the probecard pin. The function matches the input and probecard pin to the correct board and relay and opens or closes the relay as appropriate. The function “resetAllRelays” opens all relays on all boards. This routine takes about 30 seconds per board to complete. The routine “resetUsedRelays” resets only those relays are currently in use. This is a much faster reset routine, but is not a fool-proof as it only tracks the last relays connected to pins. For instance, if the user connects input 1 to pin 1, fails to disconnect the input, then connects input 4 to pin1, both inputs 1 and 4 will be connected to pin 1. This is bad, since inputs 1 and 4 are shorted together, and a call to resetUsedRelays will only disconnect input 4. Tracking past actions is complicated and therefore not currently implemented.