TEST
Automation & Controls

SCADAWARE ®

RTU / SCADA SYSTEMS

SIMPLE RELAY CARD I/O SUBSYSTEM

Document No. 1050-02
January 2001

Box 10, 1036 Destrehan Ave
Harvey (New Orleans), LA 70058
(504) 371-3000
www.test-us.com

INTRODUCTION

This document describes the use of simple relay type I/O subsystem cards in the TEST RTU/SCADA system. These cards typically fit directly into the PC expansion bus and provide a number of 8 bit inputs or outputs. Each group of 8 inputs occupies one byte in the PC's I/O address space. Thus, the software can directly access each byte, and therefore each bit, without any special command code operations as is required in more complex systems like the Metrabus.

The main SCADA program contains a driver that can handle a number of relay interface cards. Each card must be configured with a control file that tells the software about the basic layout and address of the cards.

SIMPLE RELAY SYSTEM DESIGN

As mentioned, this design assumes a simple byte wide interface for all input and output locations. The driver software is able to handle variations that will occur in the typical PC products currently available. These variations are:

1. BASE ADDRESS

2. GROUP INCREMENT

3. OUTPUT OFFSET

4. BIT MASK

RELAY SYSTEM DRIVER PROGRAM

The relay system is scanned by a "driver" program built into the main RTU program. This driver takes care of scanning all input and output types and placing the values in the proper position in the RTU's data table. From this point, the RTU processes the information without regard to where it came from. Therefore, the relay driver only performs low-level operations for input and output. All high level functions like engineering units conversion, alarm checks, etc. are done in the main program no matter where the data came from.

The driver works at the system tick level, meaning that inputs and outputs are updated about 36 times per second in the standard configuration.

CONTROL FILE PROGRAMMING

The relay card driver program is setup during the loading of the main RTU program via information contained in a text control file. The default name of the file is RTU.RLY, for RTU RELAY I/O data. The RTU program is informed that it needs to use the relay driver by using a standard DRIVER command line in it's main configuration file. The driver name must be RELAY, which will signal the requirement for local relay points. In addition to the normal task keywords and parameters, and additional file name can be supplied that tells the relay driver where to get it's control input. The default name is RTU.RLY, although any valid DOS filename can be used.

Note that this is a two step process. First, the DRIVER keyword must be present as a task name (with an optional file name) in the MAIN RTU CONTROL file (normally called RTU.DAT) to tell the system that the RELAY is being used. Second, a separate control file (normally called RTU.RLY) must be used to set up the particular relay configuration for each system.

The information contained in the relay control file tells the driver how many PC interface board setups are present, their addresses, and information about the relay cards connected to each PC interface board. In most cases, default information will be used if not supplied by the user. However, it is always better to completely understand what is going on and provide as much information as possible directly in the file.

Each line in the relay control file begins with a keyword followed by optional parameters. Each keyword provides a specific type of information to the driver, and the driver will build it's internal control tables from the data in this file. Changes in the relay setup only require simple text editing of this file in order to modify the hardware or software setup of the relay system.

The keywords used in the relay system are:

BOARD	Define a PC interface board with it's address.
STATUS	Define input points and mapping.
OUTPUT	Define output points and mapping
MSG	Display text on CRT during processing.

Each interface board should have at least 3 parameters following it. These specify the PC I/O address that is the base address for the board. Another specifies the I/O space between each board in a set. The third parameter specifies the offset between input and output points on the same board.

Say the first board has 16 inputs and 16 outputs. Its configuration file setup would look like this:

```
BOARD 16 $280 ; base address for simple board
status 16 1 ; first group of status points at $280-$281
output 16 1 ; first group of outputs at $280-$281
```

Note the use of comments on the command line following the semicolon. THESE COMMENTS ARE A GOOD IDEA so you and others can understand the contents of the file

KEYWORD

The BOARD command just sets up another group of relay points to be scanned. All points in each group must be in sequential addresses, and the status points must begin on an even 8 boundary.

The BOARD keyword takes up to 5 parameters, of which the first three are required. They are:

1. BASE ADDRESS: This is the base I/O address of the relay card group. The address is assumed to be the address of the first input port in the subsystem. This address must be unique and not conflict with any other boards in the computer. Typical values are in the $280 to $340 range ($ denotes Hex addressing).

2. GROUP INCREMENT: This is the about of addresses to skip between successive bytes. Some boards occupy more than 1 or 2 I/O address due to incomplete decoding of the address lines. For example, each board may occupy 4 addresses in a row although only 1 is used. This value is added to the base address as the driver moves from one board to another.

3. OUTPUT OFFSET: Sometimes the output ports are not at the same address as the input ports but are offset by a fixed amount. This value allows the a signed integer to be added to the base address when referencing outputs rather than inputs.

   EXAMPLE: `BOARD $284 4 2`

   The example above sets up a board that starts at hex address $284. Each board occupies 4 I/O address locations, so there are 3 wasted addresses out of each group of 4 that the board uses. Outputs are located at +2 from the inputs, so the first outputs would be located at $286.

4. BIT MASK: Some boards only provide one bit of input and output per PC address port, while others provide 8 bits. This parameter, if non zero, tells the system to use this value as a "mask" that will be logically anded with the byte input from each input port. With this scheme, each input port is

assumed to supply only one input point, not 8. The 8 inputs for the board are assumed to be in successive memory locations.

EXAMPLE: `BOARD $280 8 0 2`

This sets up a "bit" type board, where each I/O location only provides one input point. There are 8 I/O addresses between the base of each board. The outputs are not offset relative to the inputs (noted with the 0 entry). The software will use the "mask" of 2, or 00000010 when testing for inputs (and setting outputs).

STATUS INPUTS

Each input point (or output point) represents one bit of information, and points are grouped into 8-bit bytes for addressing purposes. Therefore, each relay address normally provides 8 bits of I/O information. No channel programming or other addressing tricks are needed to access these bytes. To access a byte, the program simply reads or writes to the specific PC port address.

Normally, all 8 points per address are used in the RTU data table, although this is not specifically required. It is possible to only use a portion of an address, although the others can normally be used as spares without any complications.

The standard relay status card uses 1 address in the PC I/O space. Other types of relay status input cards can also be used although they have differing address requirements.

STATUS OUTPUTS

Output points are very similar to status points, with one bit per point and 8 bits per I/O address. Normally, outputs are grouped by themselves into 8 bit bytes separately from any input points. This restricts the system to groups of 8 for input and output. This provides the most efficient processing by the driver program.

PRINTER PORTS AS OUTPUTS

Note that it is possible to use a standard PC parallel printer port as a simple output driver. Low power solid state relays can be used as horn drivers and other simple output devices without the need to install a special board. The computer's BIOS will establish the addressing and initial setting of the port during system boot. Once established, the port can be used to drive outputs as described in this document.

-end-