

# SR192 Tutorial

## Using the SR192 VXI Plug&Play Instrument Driver A24/A32 Utilities to Load and Dump I/O Module Memory (via a Disk File)

**\*\* Note:** Compare this Tutorial with the one entitled “Using the VISA “vi” Functions to Write and Read SR192 I/O Module Memory (via an Array)”, which performs the same operations only using VISA library functions and data arrays (vs. disk files).

### Overview

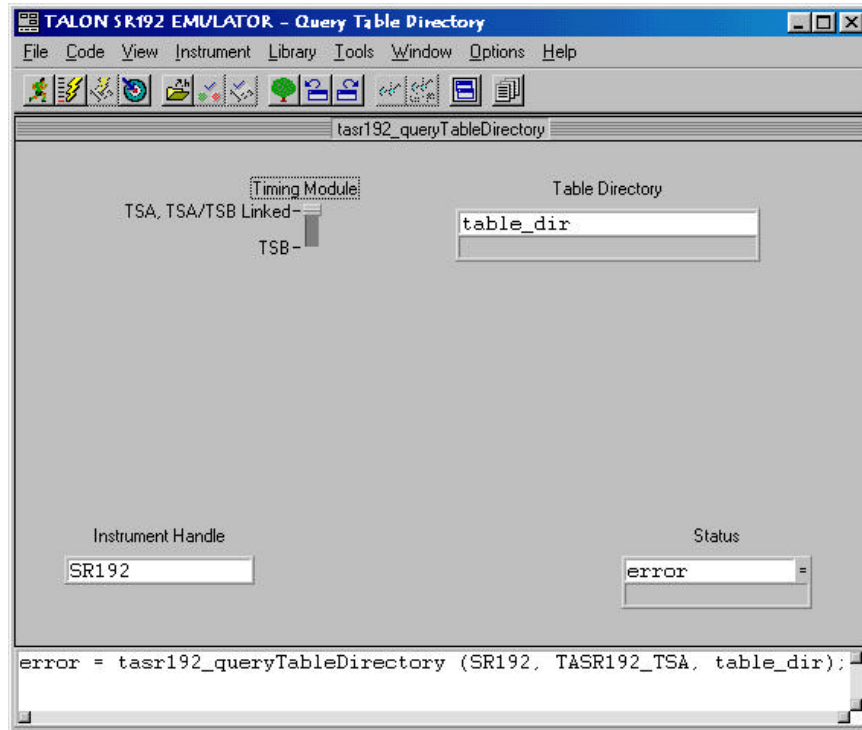
The following tutorial outlines the steps involved in using the A24/A32 register-based utility functions of the SR192 VXI Plug&Play Instrument Driver to access I/O module memory. In both the LOAD and the DUMP operations of these A24/A32 utilities, data is passed via a disk file.

In this example, an SR105 16-channel TTL I/O module is located in SR192 slot DRA3 and 3 data tables have been defined as follows: DUMMY (54321 words deep), TABLE1 (15 words deep) and TABLE2 (20 words deep). TABLE1 and TABLE2 are the targeted data tables, with DUMMY1 serving only to force non-zero offsets for TABLE1 and TABLE2.

The first step will be to retrieve the offsets of the two tables within the 128K I/O memory space of the SR192. After these offsets are known, direct A24/A32 register access is used to either load data to or dump data from the I/O module. A hex-ASCII-based \*.txt file supplies the source data to be written to the SR105's Output Memory, while another \*.txt file is used to store the data retrieved from the SR105's Response Memory.

# Step 1: Targeting the Data Tables within the I/O Memory Space

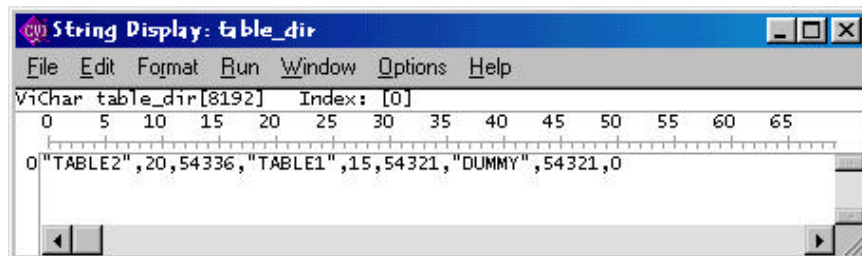
Before data can be transferred to or from a data table using A24/A32 access, the offset of that table within the 128K I/O memory space must be known. The *tasr192\_queryTableDirectory* function (shown below) returns a SCPI string into a character buffer (*table\_dir*, 1024 elements deep) that contains the **<name>**, **<size>**, **<beginning word offset>** of all defined tables. The **<beginning word offset>** values for TABLE1 and TABLE2 will be used to provide the **Module Offset** parameter for the *tasr192\_loadMemoryFromFile* and *tasr192\_dumpMemoryToFile* functions in the following Steps 2 and 3.



```
error = tasr192_queryTableDirectory (SR192, TAsR192_TSA, table_dir);
```

```
if (error) {
    tasr192_error_message (SR192, error, buf);
    printf("Error returned from SR192: %s\n", buf);
    exit(0);
}
```

The query contents of the *table\_dir* buffer are shown below. DUMMY, the first defined table, has been assigned a **<beginning word offset>** of 0, while TABLE1 and TABLE2 have offsets of **54321** and **54336**, respectively. All offsets are returned in **decimal format**.



# Step 1 (cont'd):

## ***An important note:***

For **16-channel dynamic I/O modules**, such as the SR105 used in this example, the **<beginning word offset>** value returned from the ***tasr192\_queryTableDirectory*** function must be **multiplied by 2** before being passed to the **Module Offset** parameter in the ***tasr192\_loadMemoryFromFile*** and ***tasr192\_dumpMemoryToFile*** functions.

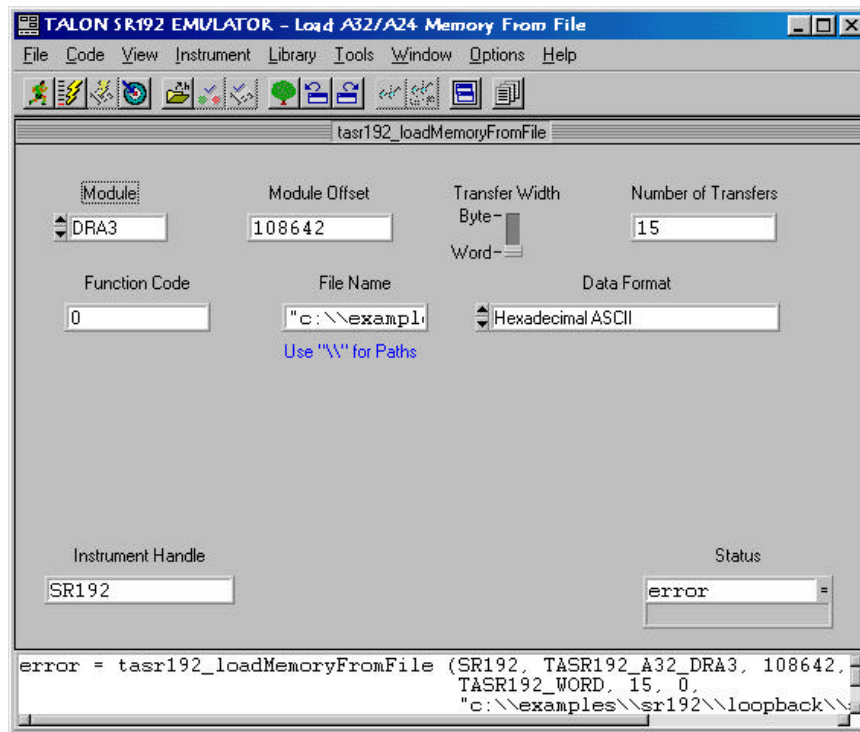
For **8-channel dynamic I/O modules**, such as the SR104 and SR122, the **<beginning word offset>** value returned from the ***tasr192\_queryTableDirectory*** function is used **verbatim** as the **Module Offset** parameter in the ***tasr192\_loadMemoryFromFile*** and ***tasr192\_dumpMemoryToFile*** functions.

Therefore, the offsets needed to program the SR105's I/O memories for TABLE1 and TABLE2 are 108642 and 108672, respectively (i.e., 54321 and 54336 both multiplied by 2).

## Step 2: Writing to the Output Memory of “TABLE1”

To write data from a disk file into I/O module memory using A24/A32 access, the *tasr192\_loadMemoryFromFile* function (shown below) is used. This function requires the following parameters:

- 1) the **Module** slot (**DRA3** in this case)
- 2) the **Module Offset** (the calculated **<beginning word offset>** for TABLE1, from Step 1 above, is **108642**)
- 3) the **Transfer Width** (**Word**, since the SR105 is a 16-channel, or 16-bit, module)
- 4) the **Number of Transfers** (**15** for TABLE1)
- 5) the **Function Code** for the target I/O memory (**0** for the Output Memory)
- 6) the **Filename** (and full path) of the source data file ("**c:\\examples\\sr192\\loopback\\sr105\\105a\_out.txt**")
- 7) the **Data Format** (**Hexadecimal ASCII** for the “105a\_out.txt” file)



```
error = tasr192_loadMemoryFromFile (SR192, TASR192_A32_DRA3, 108642, TASR192_WORD, 15, 0,  
"c:\\examples\\sr192\\loopback\\sr105\\105a_out.txt", 1);
```

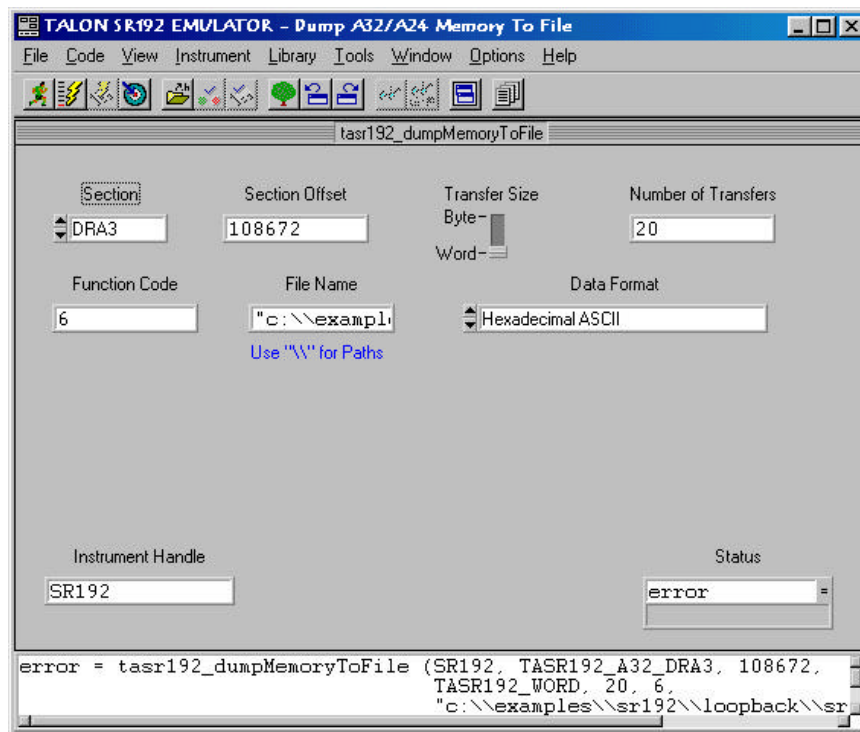
```
if (error) {  
    tasr192_error_message (SR192, error, buf);  
    printf("Error returned from SR192: %s\n", buf);  
    exit(0);  
}
```

The *tasr192\_loadMemoryFromFile* function executes a block data transfer to the Output Memory of the DRA3 I/O module (using the hex-ASCII data from the “105a\_out.txt” file), beginning at offset 108642 and continuing for a total of 15 16-bit transfers.

# Step 3: Reading from the Response Memory of "TABLE2"

To read data from I/O module memory into a disk file using A24/A32 access, the *tasr192\_dumpMemoryToFile* function (shown below) is used. This function requires the following parameters:

- 1) the **Section** slot (**DRA3** in this case)
- 2) the **Section Offset** (the calculated **<beginning word offset>** for TABLE2, from Step 1 above, is **108672**)
- 3) the **Transfer Size** (**Word**, since the SR105 is a 16-channel, or 16-bit, module)
- 4) the **Number of Transfers** (**20** for TABLE2)
- 5) the **Function Code** for the target I/O memory (**6** for the Response Memory)
- 6) the **Filename** (and full path) of the destination file ("**c:\examples\sr192\loopback\sr105\resp\_mem.txt**")
- 7) the **Data Format** (**Hexadecimal ASCII** for the "resp\_mem.txt" file)



```
error = tasr192_dumpMemoryToFile (SR192, TASR192_A32_DRA3, 108672, TASR192_WORD, 20, 6,
    "c:\examples\sr192\loopback\sr105\resp_mem.txt", TASR192_ASCII);
```

```
if (error) {
    tasr192_error_message (SR192, error, buf);
    printf("Error returned from SR192: %s\n", buf);
    exit(0);
}
```

The *tasr192\_dumpMemoryToFile* function executes a block data transfer from the Response Memory of the DRA3 I/O module, beginning at offset 108672 and continuing for a total of 20 16-bit transfers. The resulting hex-ASCII data is stored in the "resp\_mem.txt" file.

### Note:

In this example, data was read from the Response Memory of the SR105. Although all input data is physically stored in the I/O module's Record Memory, it is stored in the form of "unmasked error" data. From a logical derivation of the contents of the Expect Memory and the Record Memory, the SR105 provides a virtual memory source known as the Response Memory, which returns the actual data captured at the I/O channels. A query of the Response Memory contents works in the same manner as a query to any of the physical I/O module memory sources (i.e., Output, Tristate, Expect, Mask and Record).