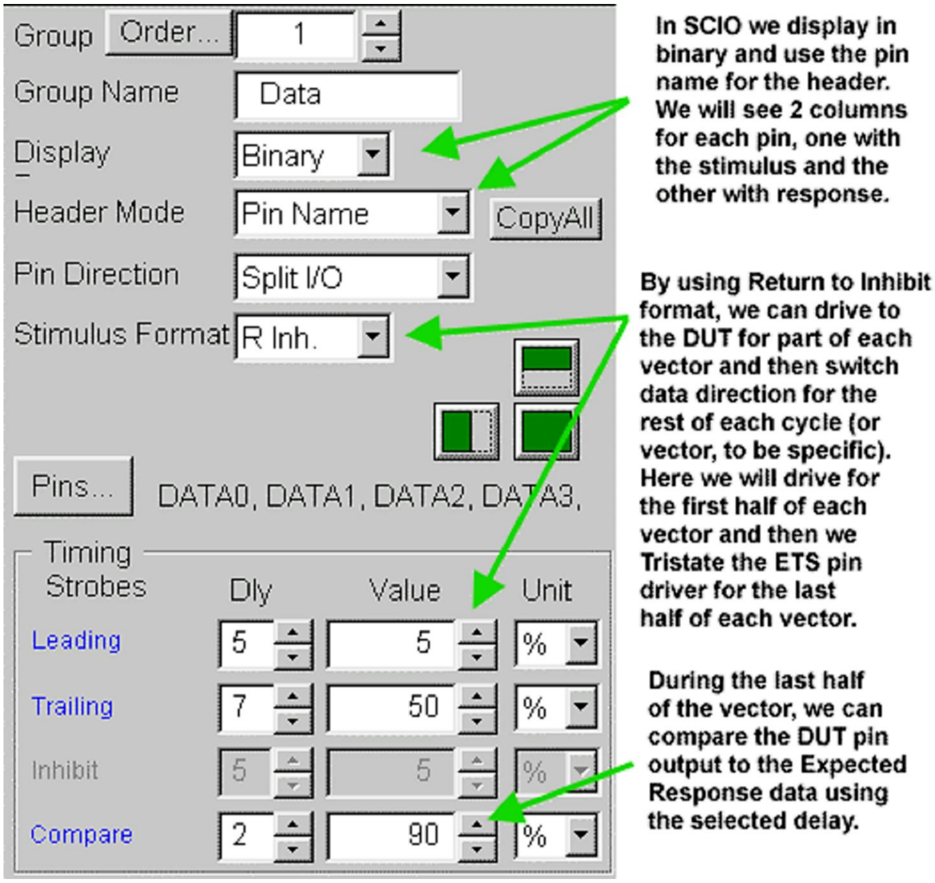
HILEVEL Technology, Inc. Q'n Apps #E42

Q: What is Split I/O?

Split Cycle I/O Data Format

The Split Cycle I/O mode (SCIO) allows you to switch data direction on every vector; that is *within* every vector. The typical application is to drive data to the DUT during the first part of each cycle and then compare DUT output data during the second part of each cycle. This mode also allows us to view the content of both the Stimulus RAM and the Expected Response RAM behind each tester channel.

Setup of the Pin Group is represented in this example:



Group: 1
Group Name: Data
Display: Binary
Header Mode: Pin Name
Pin Direction: Split I/O
Stimulus Format: R Inh.
Pins: DATA0, DATA1, DATA2, DATA3,

Timing Strobes	Dly	Value	Unit
Leading	5	5	%
Trailing	7	50	%
Inhibit	5	5	%
Compare	2	90	%

In SCIO we display in binary and use the pin name for the header. We will see 2 columns for each pin, one with the stimulus and the other with response.

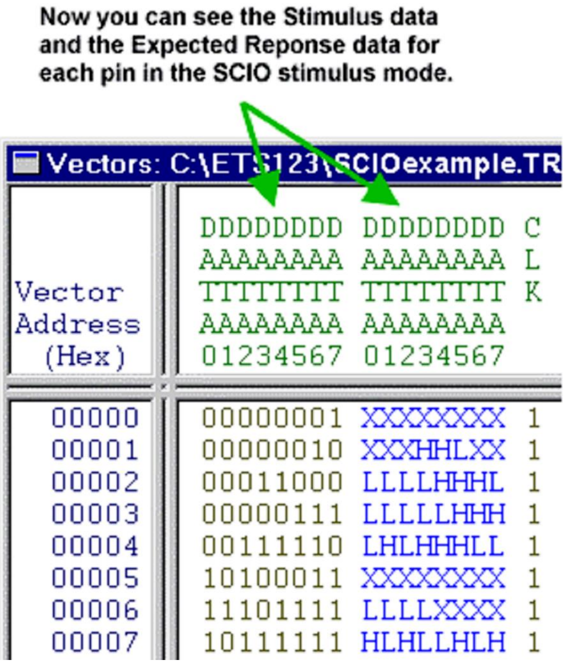
By using Return to Inhibit format, we can drive to the DUT for part of each vector and then switch data direction for the rest of each cycle (or vector, to be specific). Here we will drive for the first half of each vector and then we Tristate the ETS pin driver for the last half of each vector.

During the last half of the vector, we can compare the DUT pin output to the Expected Response data using the selected delay.

Setting up for SCIO

It is important to insure that you are not comparing Output data during the Stimulus portion of the cycle, otherwise you will be simply ñmonitoringò the tester pin driver, because your DUT output probably cannot overcome the 50 mA of drive current that the driver is providing.

When you now look at the Vector Edit window you can see two columns of data for each pin. This data also shows Tristate and Mask information. Look at the following example:



Now you can see the Stimulus data and the Expected Reponse data for each pin in the SCIO stimulus mode.

```
          DDDDDDDD DDDDDDDD C
          AAAAAAAA AAAAAAAA L
Vector    TTTTTTTT TTTTTTTT K
Address   AAAAAAAA AAAAAAAA
 (Hex)    01234567 01234567

 00000    00000001 XXXXXXXX 1
 00001    00000010 XXXHHLXX 1
 00002    00011000 LLLLHHHL 1
 00003    00000111 LLLLLHHH 1
 00004    00111110 LHLHHHLL 1
 00005    10100011 XXXXXXXX 1
 00006    11101111 LLLLXXXX 1
 00007    10111111 HLHLLHLH 1
```

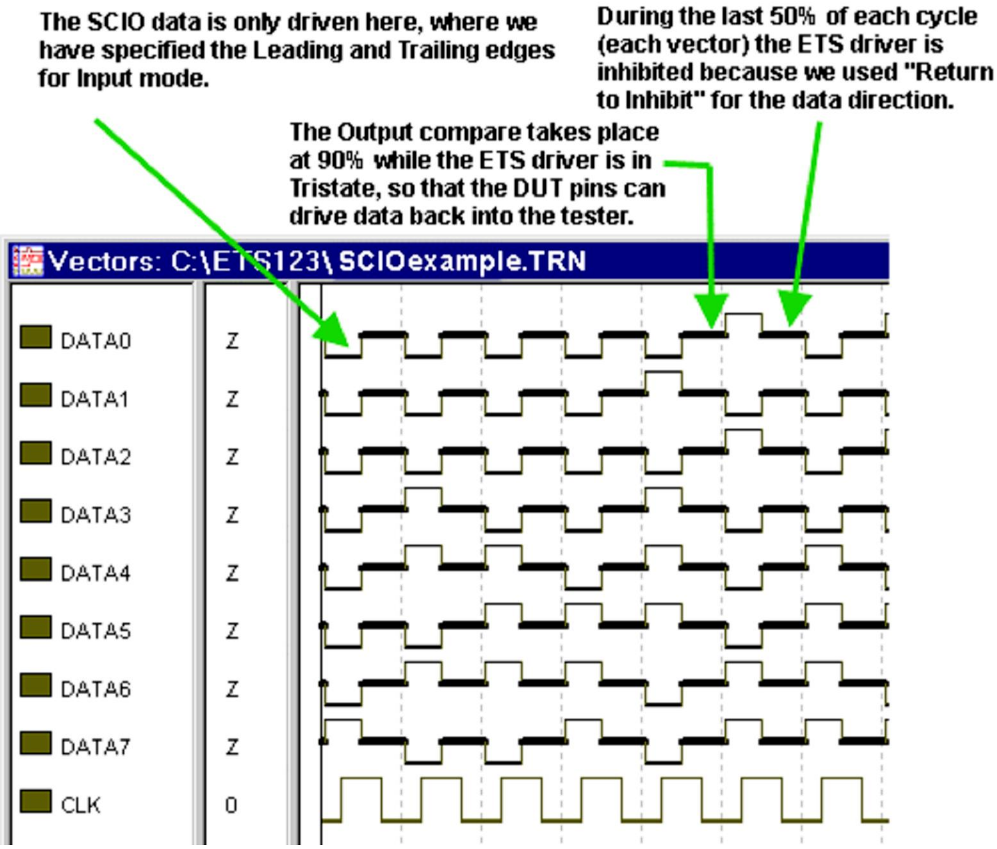
Vectors Window using SCIO

For a better understanding of SCIO, it can be helpful to know more about the pin electronics memory. There are actually 5 RAMs behind each tester pin:

Stimulus RAM	(amber colored drive data)
Response RAM	(blue Expected Response data)
Tristate RAM	(an amber ñZò)
Mask RAM	(a blue ñXò)
Acquisition RAM	(captures the DUT output data)

When a Tristate or Mask bit is set at any given vector, it does not mean that the Stimulus or Response RAM is empty. Data is still present in the Stimulus and Response RAM at every vector, we just cannot see that data when a Z or X is displayed unless we are using SCIO. To see this data in a ñnon-SCIOò mode, simply place the cursor over the Z or X and type a ñUò. You can also ñunmaskò or ñuntristateò an entire range of vectors by using a U with the Fill command.

If we look at the Vector Graph while the Vectors window is open, we can see more about how the tester performs. Vector Graph can only show Stimulus data in SCIO mode, but look at the following example to learn more.



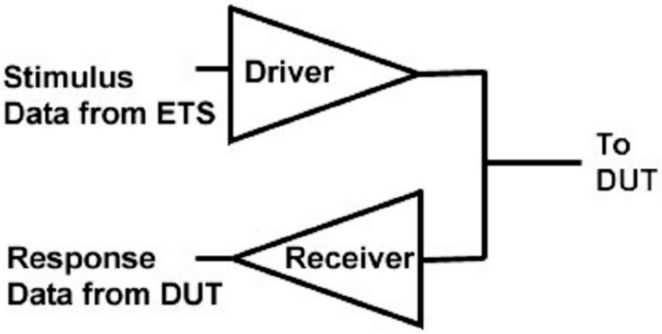
The SCIO data is only driven here, where we have specified the Leading and Trailing edges for Input mode.

During the last 50% of each cycle (each vector) the ETS driver is inhibited because we used "Return to Inhibit" for the data direction.

The Output compare takes place at 90% while the ETS driver is in Tristate, so that the DUT pins can drive data back into the tester.

Vector Graph showing SCIO vectors

At HiLevel, we use SCIO to help verify tester functionality by driving data on each pin directly into the pinsôown receiver. This can be done entirely in software because of the design of the driver/receiver circuit:



Simplified drawing of ETS Driver/Receiver circuit

Also See:
Q'nApp #E17: Stimulus formats
Q'nApp #E23: Inhibit Delay
Q'nApp #E24: Data vs. Clock
Q'nApp #E25: Drivers and receivers
Q'nApp #E53: Loopback