Prelab 3 - Comparator V_{OS}/ V_{HYST} Test

Introduction

In this lab exercise, you will write and test the code to perform the input offset voltage test. You have all of the fundamentals to perform this test. However, this prelab is also intended to teach you how to present your testing strategies in a standard format.

When someone is developing a set of production tests, they generally create a test plan that shows how each test should be performed, expected results, and the final test order. This test plan shows the what and why for your final test program. It is intended to be referenced later by a product engineer who may consider changing something about the test. If choices are randomly chosen, it should be noted as such. If choices are made for a reason, that too should be noted so that the product engineer does not mess something up trying to improve something else.

The typical sections in a test plan include:

1. Device background/description: describe the major features, general operation of the device, test modes that may not appear on a final datasheet, etc.
2. A list of each test performed. Within each test, the following information should be included:
   - purpose of the test
   - hardware schematic showing test set-up (simplify as much as possible)
   - operating conditions for each test (the hardware schematic shows some of these conditions, but some tests require the conditions to change to get the full operating range. These changes should be noted.)
   - any mathematics that is required to obtain the datasheet specification from the measurements- datasheet limits
3. A list of the test order with an explanation for the test order
4. Any tester specific information, such as clock speed, waveform file names, or DIB information. (placing this information in its own section simplifies the transfer of a test program from one tester to another.)

An example of a test plan for the LF147 general purpose op-amp is given for your reference.

Prelab Assignment

(a)Print out a copy of the DIB schematic for each site. Show all paths necessary to perform this test. In this manner, you will determine which relays must be closed and which resources to use.

(b)Will you need to add any more groups of resources? Explain your answer.

(c)Write the test plan for the continuity test and the V_{OS} / V_{HYST} test. Be sure to include the method of finding your trip point, and the calculations necessary to find V_{OS} and V_{HYST}. To find the trip points, please do a linear sweep of input voltages. While successive approximation may be faster, due to the hysteresis and necessity of resetting the comparator, the linear sweep will actually be faster given the resolution we can meet. Find the trip points to the nearest 0.5mV. The resolution for the SPU is 80uV. However, the accuracy is only 0.8mV. Therefore, you can distinguish between a 0.2mV and 0.3mV offset voltage between 2 chips, but the instrument may introduce a 0.8mV offset voltage to both measurements.

Lab 3 – Comparator V_{OS}/ V_{HYST} Test

Write the test code for the V_{OS} /V_{HYST} test

Reminders of the key coding elements are shown below. Please refer to labs 1 and 2 for details.

1. Group any resources together as necessary.
2. Create a new function.
3. Set up the datasheet file.
4. Write the V_{OS}/V_{HYST} test. Recall that you must save the data into a datastructure immediately after the measurement or the measurement will be lost. Hint: when you look for the V_{IN} that causes V_O to flip high, you will need to set a boolean variable (an integer is sufficient) that will pop you out of the loop when the output has flipped. Please do this to minimize test time. However, this must be done for each site, and you cannot leave the loop until both sites have flipped.) In this test, you will have to calculate Vos and Vhyst from the upper trip point and lower trip point. Pull your data from the data structure, perform the math, and store the results into a new array. Using this procedure, you can look at your measurements and your mathematical results at the same time to aide in debugging.
5. Don't forget to modify TestCompletion and FailSite if necessary.
6. Compile, build, and run your program. Fix any errors.
7. Backup your program onto your memory stick and let the professor know that you are ready to go to the tester.
8. First, run your test empty socket and debug any final errors. Then, test both sites simultaneously. Debug any errors. Once both tests pass, repeat the test a few times. How much variation does your measurement show? Do you ever get different measurements on site1 and site 2? Swap your chips. How consistent are your measurements from site to site? Do differences in sites move with the chip or remain with the site? Record your results.
9. Remember to record your test time.
10. If you made any necessary changes to your code, make sure you back up the new version and restore it onto your computer for next week. Save your data to your memory stick.

Write-up your results

Turn in your final code, test results, and a discussion of site-to-site reproducibility (change switch your chips between sites and see how much your measurement varies).