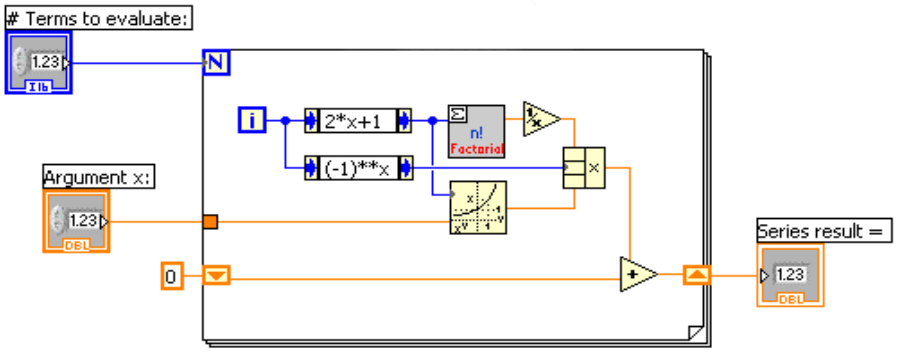
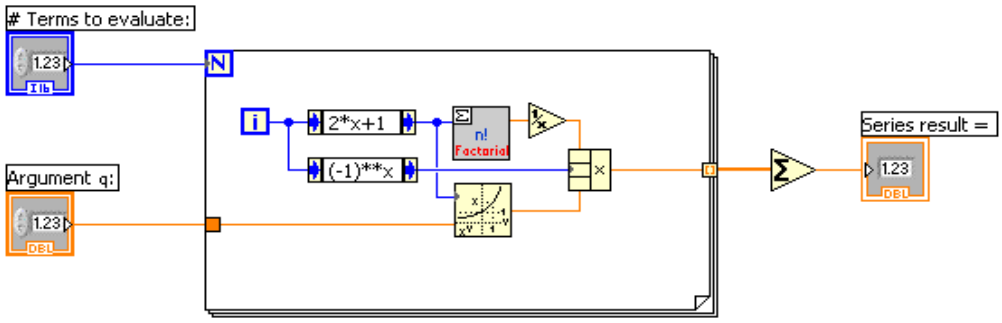
Worked Example: Sine Series by Array

Consider the previous example which evaluated the infinite series representation of the sine function:



The implementation used a shift register to keep a running total of the terms as they were produced in each iteration of a loop. The shift register was necessary because each term was overwritten in the next iteration, so the summation had to be done continuously. However, what if we were to store **all** of the term values instead? Then we could simply add all of the values together **outside** of the loop, and so the shift register would not be needed. Consider wiring the term from each loop to the border, and using *array indexing*:



In this implementation, a single array function is used to add up all of the array elements **after** the loop, yielding a simpler program. However, note the following tradeoffs:

- The array implementation offers no means of an intermediate result, if desired. In other words, in the first diagram, we could move the *Series Result* indicator into the loop, and get continuous feedback about the summation progress.
- The array implementation requires more memory space, because the shift register stores only one value at a time rather than all values generated. This is a trivial issue here, but in the context of data acquisition—where we could realistically be dealing with millions or billions of data values in loops—the tradeoff is much more important.