**Systolic Array Traceback Functionality**

Work Completed

As it stands, we have a largely untested traceback functionality for query lengths up to 128. Traceback directionalities are reported from the hardware and are assembled by the software into a matrix that can be traced through. Once the subject starts being passed through the systolic array, traceback data from each individual cell will be sent for each clock cycle until the subject has completed. If the query is wide enough to require it, two output streams will be used for the traceback data. Once the software has received the traceback information it is combined into a matrix format that is traced through for the final result.

Firmware Changes (from initial SmithWaterman):

* SmWaCell.v - Added comparison to set traceback direction (2 bits) in unison with all comparisons that are done to find the next score. Direction value is flopped before being output.
* SystolicArray.v - Combines outputs of all cells into a single bus with all traceback info. Added logic to keep track of when scores are actually being calculated to supply the data valid signal output.
* SmWa.v - Added a cross clock fifo for the traceback data.
* SmWaWrapper.v - Added traceback signals, no logic just pass signals through.
* PicoSmithWaterman.v - added signals from SmWaWrapper to output to streams that were used to transfer the traceback data.

Possible Improvements

* BLAST integration - Our most important goal remains the integration of the function with the actual NCBI Blast software which would have been possible, given more time.
* Add streams - As the design currently stands we believe that we have not maxed out the PCIe bandwidth. Our current design should be flexible enough that very little would need to be changed in software and output streams could be added from hardware for each extra 64 bases wide that the max query length is made. This can only get us so far as the PCIe bandwidth limits.
* Filter data - A method could be found that would allow the module to determine which regions of data are actually of use. Data bandwidths are the largest hurdle in the design. If the module were to send only the areas of data there were of interest, a large amount of the data that is being transferred would be thrown away instead. BLAST uses several drop-off values that could be written to a register on the PicoBus and which could be compared to the current scoring evaluation.
* Use DRAM - Using DRAM would allow for higher bandwidths for the traceback data being generated. If the traceback data were stored in DRAM, larger amounts could be generated each cycle commensurate with the higher memory bandwidth speeds as opposed to PCIe.
* If data were stored locally, the actual traceback could then also be done in hardware leaving only the actual trace result to be sent back to software instead of the entirety of the traceback data. However, this will pose a restriction on the size of subject sequence as well, since only limited amount of memory available locally.