It would seem that programming an artificial neural network (ANN) onto an FPGA is not a new concept, due mainly to the fact that the parallel nature of FPGA’s lend itself so well to ANN’s. One of the main advantages of using an FPGA, which many neural network models haven’t taken advantage of, is using the reconfigurability of FPGAs to implement an ANN algorithm.

**Approaches**

One approach to implementing an ANN is to use “FPGA run-time reconfiguration” (1). This involves dividing an algorithm into sequential stages and having the FPGA implement one stage at a time, reconfiguring itself after each stage for the next stage. This would reduce the space complexity needed to run the ANN, but I feel we would need to look more into what exactly this method would entail.

**Pit-Falls**

We will have to be careful about our use of multiplication units, as these are one of the most space-consuming features of a digital neural network (1). There are a few approaches we could take to limit the impact that these multipliers will have. These methods include time-division multiplexing (TDM) and bit-serial stochastic computing techniques.

1. <http://hackbbs.org/article/reds/form_de_base/SBI/doc_introANN.pdf>