1.) The traditional architectures use von neumann architectures where compute and memory are separated whereas the AI test beds use spatial architectures where compute and memory are on the same chip.
2.)
    a. Cerebras main difference is the size where the wafer is very large, using the entire wafer and much bigger than GPU with more transistors. It uses a WSE-2 Architecture which appears as a 2D array of individual processing elements. Software: uses a variant of pytorch cstorch and decides how data is loaded.
    b. Graphcore has IPU using MIMD and uses a BSP model of execution which means one core can be executing and one core could be waiting. Each has its own core and local memory within each of the processing tiles. Has 8832 program threads executing in parallel. For software it uses various packages such as pytorch, tensoflow, and keras. Also has profiling tools.
    c. sambaNova uses an Reconfigurable Dataflow unit (RDU) which has compute and memory units connected by a mesh. It has a hybrid approach with high bandwidth memory. It has tiles with memory units inside of each of the 8 tiles. The execution of the task is dependent on the data. It has a spin on pytorch called samba flow. Write it in pytorch and then compile it to a pef file. This is then used in the samba runtime.
    d. Groq uses LPU (language processing unit) which is only targeted to the inference and training cannot be done. Woprks with pytorch, tensor flow, and keras. You train model io GPU using software and export it into onnx. Then it graphcore takes the onnx graph to graphflow to perform inference.
3.) Typically, these AI test beds have their own variants of existing software such as pytorch or tensorflow. For example cerebras uses cstorch. Graphcore uses popTorch in which extends the data loader object in pytorch. Instantiation is practically the same as in pytorch but the use must pass in and instance of “option”. For Samba, the user must specifically use sambaTensors for any tensor operations as it has a unique way of interfacing with the RDU. One way in that the flow is different is that we compile our pytorch code into a intermediary “pef” file.
4.) A project which I could imagine would be using AI to classify certain tracks of particles within a high energy particle detector. In fact, this is something I do on TACC supercomputers. So these Ai testbeds could help to speed up the training process. It would benefit because There are typically thousands of data points in each track. If we’re using a GNN, this becomes expensive very quickly.