CLAIMS

1. A computer-implemented method, comprising:

receiving source code expressed in a multi-threaded programming language, wherein the source code includes a branching statement that directs execution to one of a plurality of source code paths;

compiling the source code to a circuit description that includes a pipeline, wherein the pipeline includes a plurality of code paths associated with the plurality of source code paths, and wherein one or more pipeline stages are added to one or more of the plurality of code paths so that the plurality of code paths have a same number of pipeline stages; and

generating, based on the circuit description, a synchronous digital circuit comprising the circuit implementation, wherein the source code includes a reorder block construct that wraps a programming construct that does not maintain thread execution order, and wherein the reorder block construct maps to a circuit implementation that:

records an incoming thread execution order,

allows threads to execute the construct that does not maintain thread execution order, and

blocks threads from resuming until all lower-ordered threads have resumed.

2. The computer-implemented method of claim 1, wherein a plurality of threads enter the pipeline in a first order, and wherein the plurality of threads leave the pipeline in the first order.

3. The computer-implemented method of any of claims 1-2, wherein adding pipeline stages to one or more of the plurality of code paths includes determining a number of pipeline stages in a longest code path and adding a number of pipeline stages to each code path equal to the number of pipeline stages in the longest code path minus the number of pipeline stages in that code path.

4. The computer-implemented method of any of claims 1-3, wherein the pipeline comprises a first pipeline, wherein the circuit description includes a second pipeline, wherein a thread executing the first pipeline transfers execution to the second pipeline by pushing local variables into a first-in-first-out queue, and wherein the second pipeline maintains thread execution order across pipelines by reading local variables out of the first-in-first-out queue in the order they were pushed.

5. The computer-implemented method of any of claims 1-4, wherein a thread comprises a collection of local variables provided to the pipeline for execution.

6. The computer-implemented method of any of claims 1-5, wherein the pipeline

comprises stages executed in sequence, and wherein a plurality of threads maintain execution order by flowing through the stages in the sequence.