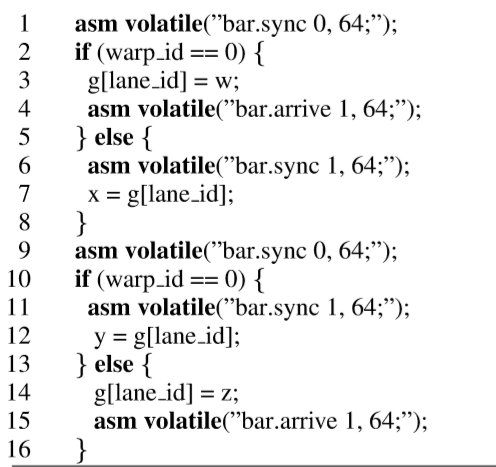
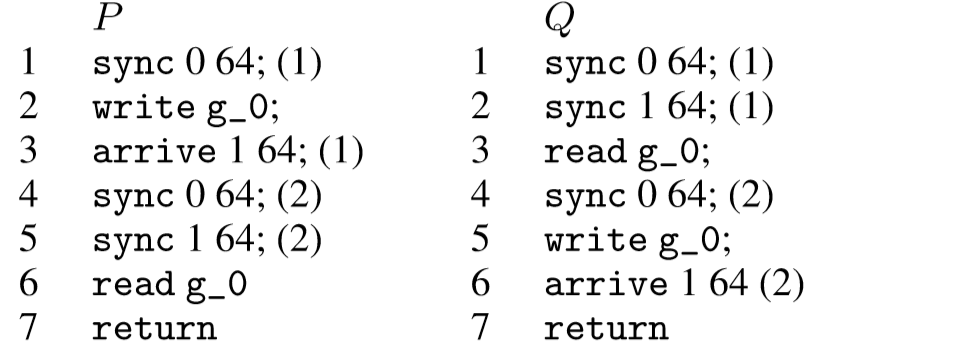
**Algorithm: Section 4:**



Consider the above kernel program which is to be verified for synchronization. The kernel consists of 2 warps. So a total of 64 threads.



P: Program for thread ID 0 Q: Program for thread ID 32

Consider a known constant j, Each location in shared memory is treated as a separate variable g[j]. Here j= lane\_id. Hence, there is a possibility of a data race to occur. To ensure there is no data race we need to establish a happens before relationship between the synchronisation commands.

We also need to ensure that the Generation Id’s of all the traces are same to have a well defined synchronization.

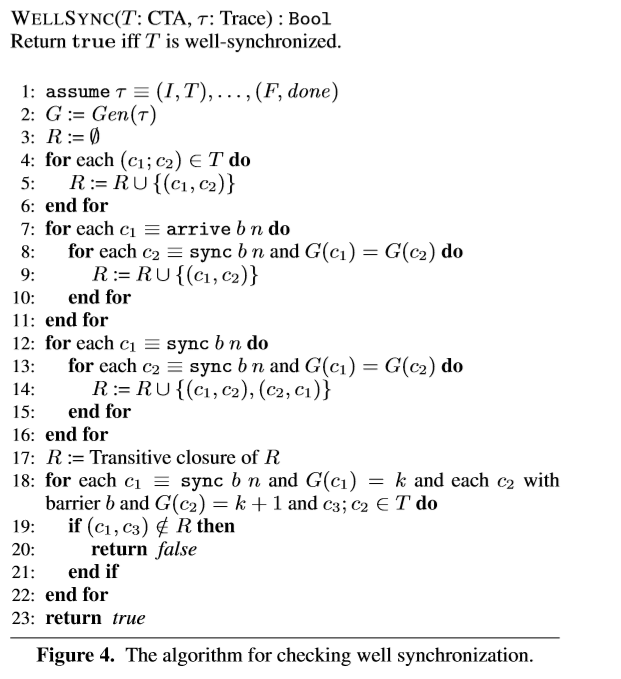
In the above example:

Both the threads synchronize at barrier 0. Then The barrier is recycled, giving the trace a generation ID of 1. After this the thread P writes at the shared memory location g[0] while the thread Q waits for this process to complete by establishing a sync operation on barrier 1. As soon as the write is complete, arrive is executed by thread P to signal that thread Q can now read the data at g[0].

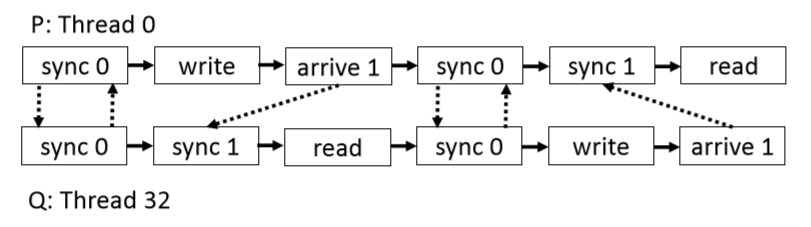
Finally, both the threads sync again at barrier 0, and the barrier is recycled, giving the sync command a generation id of 2.

The Order of operation is now reversed and Q acts as a Producer and P acts as a consumer. And the operations are inter-changed.

To ensure well synchronization, all traces are statically checked for same generations to commands as Gen(τ). The algorithm is shown below.



It is initially assumed that the Trace ends in a completion without any errors. The Generation of the trace is stored in G. G is used to construct a happens before relationship between the commands. The Relation is initialised to empty. Successive commands are added in order, as for GPUs the successive commands are guaranteed to execute in order.



The above graph gives a step by step establishment of happens before relationship establishment.

The graph starts with no initial edges. The lines 4-6 establish a happens before relationship among the successive commands in the program. They add all the solid lines representation from above graph.

The inter-thread happens before relationship is established using the lines 7-16.

If c1 is an arrive and c2 is a sync such that the two commands are in the same generation, then add (c1,c2) toR.Now,forc1 andc2 corresponding to sync in the same generation, add (c1,c2) and (c2,c1) to R.

Basics: A transitive closure of a binary relation R on a set X is the smallest relation on X that contains R and is transitive.

A transitive closure of the relation R is computed to have a full static happens before relationship.

lines 18 to 23 check that there exist happens-before relationships between successive generations of the same barrier. Happens before relationship is established between line 1 and 4 , 2 and 5 and so on. If for all (c1,c3) this satisfies then the thread execution is well synchronised else it is not well synchronised.