

# Concurrent start and completion of persistent requests (#858)

Joachim Jenke

**Joseph Schuchart**

```
MPI_Request req[2];
int buf0 = 0, value = 42;
MPI_Recv_init(&buf0, ..., 42, MPI_COMM_SELF, &reqs[0]);
MPI_Irecv(..., 1, MPI_COMM_SELF, &reqs[1]);
```

Thread 0:

```
MPI_Waitall(2, reqs, MPI_STATUSES_IGNORE);
```

```
/* after return from waitall */
```

```
MPI_Test(&reqs[0], &flag, MPI_STATUSES_IGNORE);
```

```
assert(flag); // succeeds
```

```
assert(buf0 == 0); // succeeds
```

```
MPI_Send(&value, ..., 42, MPI_COMM_SELF);
```

```
assert(buf0 == 42); // succeeds
```

Thread 2:

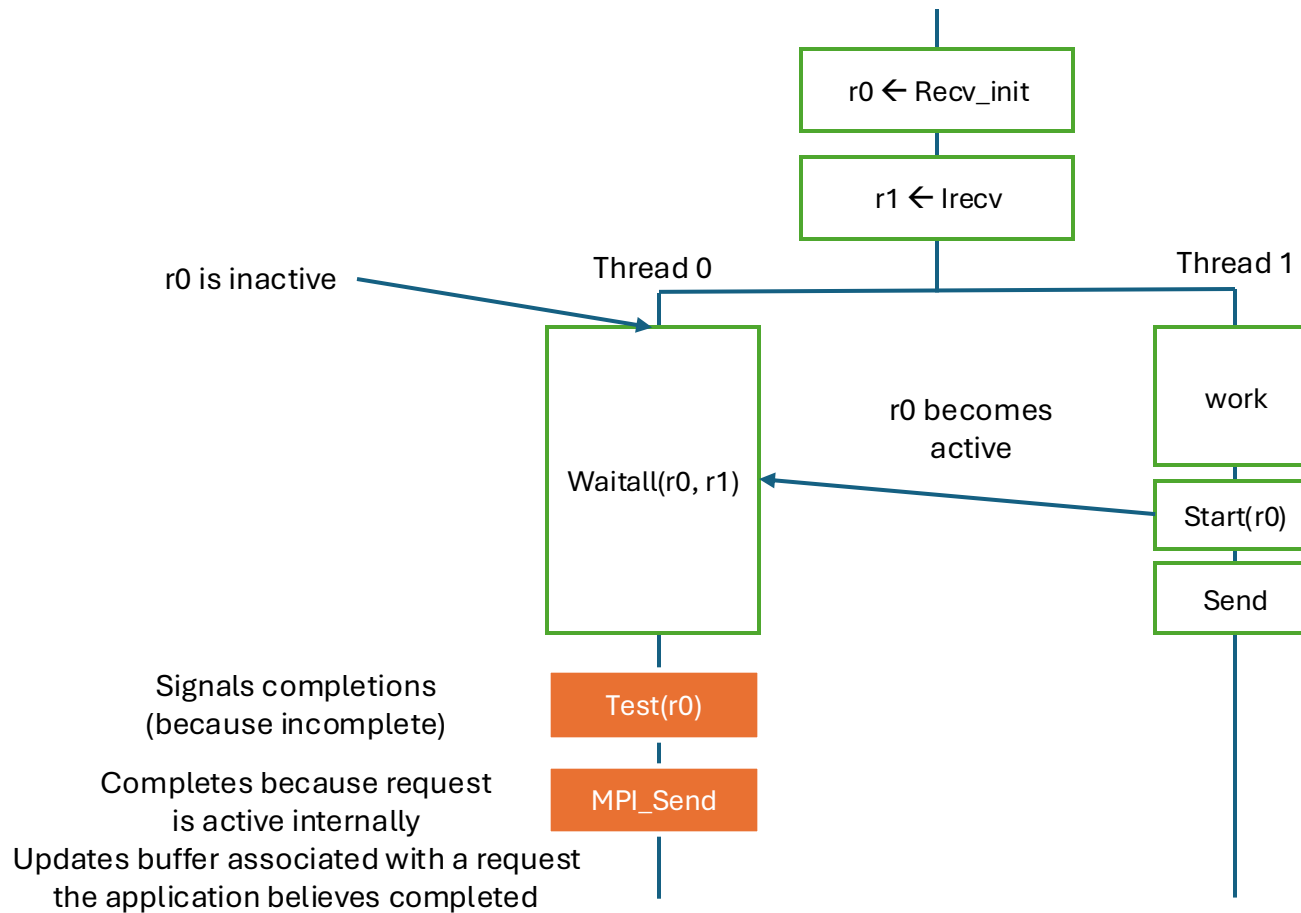
```
/* ensure Thread 0 enters waitall */
```

```
usleep(100000);
```

```
MPI_Start(&reqs[0]); // start recv
```

```
MPI_Send(..., 1, MPI_COMM_SELF); // complete irecv
```

Modifies the buffer of a completed request!



- Operations should be executed in some order
- Open MPI and MPICH mark request as inactive but mutate buffer once the send with tag 42 is posted

# Discussion

- What is the expected behavior?
- Is it valid to start a request another thread is waiting on?
  - *“When a thread is executing one of these routines, if another concurrently running thread also makes an MPI call, the outcome will be as if the calls executed in some order.” [MPI 4.1, 11.6]*
  - *“A program in which two threads block, waiting on the same request, is erroneous. Similarly, the same request cannot appear in the array of requests of two concurrent MPI\_{WAIT|TEST}{ANY|SOME|ALL} calls. In MPI, a request can only be completed once. Any combination of wait or test that violates this rule is erroneous.” [MPI 4.1, 11.6.2]*
- <https://github.com/mpi-forum/mpi-issues/issues/846>