

How to define and use a distributable function

version	1.1.1
scope	Example. This code is provided as example code for a user to base their code on.
description	How to define and use a distributable function
boards	Unless otherwise specified, this example runs on the SliceKIT Core Board, but can easily be run on any XMOS device by using a different XN file.

If a task is a never-ending loop containing a single select (like a combinable function) that *only has cases responding to interface messages*, the function can be marked as *distributable*. For example:

```
[[distributable]]
void port_wiggler(server interface wiggle_if c, port p)
{
    // This task waits for a message on the interface c and
    // wiggles the port p the required number of times.
    while (1) {
        select {
            case c.wiggle(int n):
                printstrln("Wiggling port.");
                for (int i=0;i<n;i++) {
                    p <: 1;
                    p <: 0;
                }
                break;
            case c.finish():
                return;
        }
    }
}
```

A distributable task can be distributed within a `par`. This means that the task will not run on any particular core but will be run on the core of the task that calls to it.

```
int main() {
    interface wiggle_if c;
    par {
        task1(c);
        [[distribute]] port_wiggler(c, p);
    }
    return 0;
}
```

A distributed task must be on the same tile as the tasks it is connected to.



Copyright © 2013, All Rights Reserved.

Xmos Ltd. is the owner or licensee of this design, code, or Information (collectively, the "Information") and is providing it to you "AS IS" with no warranty of any kind, express or implied and shall have no liability in relation to its use. Xmos Ltd. makes no representation that the Information, or any particular implementation thereof, is or will be free from any claims of infringement and again, shall have no liability in relation to any such claims.