1

Unravelling `XC` concepts `[[combine]]`,`[[combinable]]`, `[[distribute]]`, `[[distributable]]` and `[[distributed(..)]]` plus `par` and `on`..

In search of understanding how some attributes, as present in XC code, by the compiler are treated weaker and more general and ultimately may be ignored altogether - in the runnable code for xCore multicore microcontrollers

Øyvind TEIG[1]
Coder and blogger

The XC programming language is designed to make runnable multitask programs for XMOS' xCore multicore microcontrollers, spread on *tiles* and *cores*. This presentation addresses only a few aspects of XC. What is a *combinable* and what is a *distributable* task, syntactically and semantically, but also resource wise and geographically on the chip? The xCore compiler handles the lowering of interfaces onto statically and dynamically allocated channel resources. The microcode and built-in scheduler also reflect them. Of the rather limited amount of resources, one must make it with 32 *chanends* per tile, with seemingly loose coupling between the XC code and the final number of chanends. Fiddling around, rather overwhelmed, with *[[combine]]*, *[[combinable]]*, *[[distribute]]*, *[[distributable]]* and *[[distributed(..)]]* plus *par* and *on* reveals a lot of how unchanged functionality spreads differently over cores or across tiles. Thus the same functional code may require anything from, like, six to *zero* chanends! The goal of this fringe presentation is to trigger interest, not provide answers. Or rather, only those answers that the author might presently have acquired by surrendering to this can of reserved words and observe what might possibly happen. Like, there must be an electric motor inside, because the sound of it is so pleasant.

Keywords. code lowering, placement, attribute decay, conversion, substitution, optimisation, channel, interface, critical region, lock, multicore, scheduler, scheduling.

---

[1] http://www.teigfam.net/oyvind/home/ and http://www.teigfam.net/oyvind/work/work.html, embedded and safety-critical coder, retired since June 2017