Event-driven computing

(I of IV) 5 February 2019

Andrew Brown
Department of Electronics and Computer Science, University of Southampton, SO17 1BJ, UK

Abstract:

All modern computers are based on an idea first published by Alan Turing as a thought experiment to support his proof of key results about the fundamentals of computability. His concept of the Universal Machine forms the theoretical foundation for the stored-program computer, conceived in practical terms by von Neumann, Eckert and Mauchly, and which first became an engineering reality in the Manchester Baby machine and soon thereafter as a practical computing service supported by Maurice Wilke's EDSAC in Cambridge. At the heart of this idea is the concept of sequential execution: each "instruction" starts with the state of the machine when the preceding instruction has completed, and leaves the machine in a well-defined state for its successor. High-speed implementations bend the actual timing of instruction execution as far as it will go without breaking semantics, but still emulate the sequential model.

The history of advances in computing has revolved around making this very simple execution model go faster, partly through bending the timing of instruction execution as noted above, but mainly through making transistors smaller, faster, and more energy-efficient, all thanks to Moore's Law. This approach has delivered spectacular progress for over 50 years, but then hit a brick wall – the power wall is much vaunted, but in reality the wall is multidimensional and complex, but solid nevertheless. Since then largely illusory (marketing) advances in performance have been delivered through multi-core and then many-core parallelism – putting a modest number of sequential execution engines on the same chip, whose potential (marketing) performance can rarely be realized due to the difficulty inherent in trying to make sequential programs work together in parallel.

The time has come to look again at the fundamentals of computation: to abandon sequential instruction execution as the only model of a computation. Alternatives to sequential execution are all around us, including the massive parallel computing resource on a modern FPGA, and the vast complex of biological neurons inside each of our brains. Minor difficulties remain: we do not yet have any general theory of computing on such huge, distributed, networked resources, and we can't build biological computers anyway. It's time that changed. Sequential computing is not natural, it's not efficient and, in fact, the only thing it has going for it is that it's easy.

In this seminar, we present a little bit of history, and then suggest an alternative way of approaching real-world engineering computing problems - principally simulation, although this is a very broad term. This alternative approach is that of ***event-based computation*** - where we allow our model of a system to "relax" into some reasonable solution configuration corresponding to reality, in exactly the same way as occurs in nature. We generalise the issue almost beyond recognition, and suggest ways in which we might realise this alternative approach. Finally, we offer some hint - in preparation for the subsequent seminars - about the properties of these solutions.