Thesis/Dissertation Sheet

Surname/Family Name	:	**Clifton-Everest**
Given Name/s	:	**Robert Leonard**
Abbreviation for degree as give in the University calendar	:	**PhD**
Faculty	:	**Faculty Of Engineering**
School	:	**School of Computer Science and Engineering**
Thesis Title	:	**Streaming irregular arrays**

Abstract 350 words maximum: (PLEASE TYPE)

It is well established that declarative array languages are a high-level and expressive means of writing programs for parallel architectures. However, they are not without their disadvantages. Flat data parallel array languages are limited in their modularity as they do not support nested arrays, and parallel functions cannot be called from within parallel contexts. Nested data parallel languages solve this problem, but they also assume irregularity of all nested structures. This places a cost on nesting, a cost that is needlessly paid for a certain class of programs, those where nesting is strictly regular.

A second problem is that arrays, by definition, allow for random access. This means that they must be loaded into memory in their entirety. If memory is limited, as is the case with GPUs, array languages offer no high-level means of expressing programs containing structures too large for that memory but do not require random access.

The research in this dissertation describes an extension to the Accelerate language: irregular array sequences. They allow for both a limited, but still useful, form of nesting and a streaming execution model that can work under limited memory.

Furthermore, in realising irregular array sequences, we describe a generalised program flattening (vectorisation) transform that does not introduce the unnecessary cost on nesting that is incurred for strictly regular (sub)programs. This transform applies to a much broader domain than just array sequences.

As a further complement to sequences, this dissertation also describes two other extensions to Accelerate, a foreign function interface (FFI) and GPU-aware garbage collection. The former is the first instance of an embedded domain specific language having an FFI, and both are of considerable practical value to programmers using Accelerate framework.