EmBounded: Formally Bounded Embedded Systems

embound, *v.*
poet. arch.
trans. To set bounds to; to confine, contain, hem in.
Hence **embounded** *ppl. a.*
1595 SHAKESPEARE: *The Life and Death of King John* IV. iii 137
That sweete breath which was embounded in this beauteous clay.

"Embedding" has recently become a very familiar term, referring to situations where people or objects become firmly fixed into some larger setting. In Computer Science, however, the term has a slightly different meaning, referring to special-purpose (and often small-scale) computer systems rather than the general-purpose laptop or desktop systems we are more familiar with. Embedded systems are formed by linking tightly-coupled software and hardware components into a single dedicated system.

Among the first recognisable embedded systems were the guidance computers used in the Apollo lunar lander module. Loose descendents of these systems today fly many of the aircraft in the skies over the UK, including those built by BAe Systems, our research collaborators. With decreases in cost and increasing miniaturisation, embedded systems have become increasingly commonplace, however. Today, they comprise the *invisible* and *ubiquitous* computer-based systems which surround us in everyday life and which we largely take for granted: SmartPhones, PDAs, Digital TVs, ABS braking systems, Oyster Cards, intelligent lights, etc. etc. They also form the basis for many of the exciting technologically-based policies that government is pursuing in transport, security and other areas. This widespread use of embedded systems is reflected in their economic value: in 2004, the global market for embedded systems was worth £23bn, and this is projected to almost double by 2009. Significantly, Europe is the major player in many sectors that exploit embedded systems, including telecommunications, avionics, automotive systems and medical/industrial automation.

Despite serious limitations on the available system *resources* (processor speed, memory, power), embedded systems are expected to have reliability levels similar to that for consumer goods (7-9% failures p.a.) rather than for general-purpose computers (25% failures p.a.). This creates major challenges for system design. If it were possible to determine strong bounds on the use of system resources, however, then there would potentially be significant benefits in terms of reduced manufacturing cost, faster time to market, increased reliability and better system performance. Unfortunately, determining such bounds automatically is a hard problem, and doing so manually is becoming increasingly difficult as embedded software increases in complexity.

The €1.3M EU Framework VI **EmBounded** project, led by Dr Kevin Hammond at the University of St Andrews, aims to research this problem. The EmBounded project brings together top research teams from three European countries in a three-year pilot project, building on strong British expertise in formal methods for Computer Science. Our work is based around a new programming language notation, Hume, named after the C18th Scottish philosopher and sceptic, David Hume, and acting as a *virtual laboratory* for research into bounded resource usage. Our vision is one where *certificates* of the bounds on resource usage can be obtained from a source program through *automatic program analysis*. These certificates may then be verified using formal proof techniques based on a formal *program logic* that captures the meaning and behaviour of embedded software programs. In this way, we will be able to provide automatic guarantees that the required resource constraints will be met in practice.

EmBounded: **http://www.embounded.org**
Hume: **http://www.hume-lang.org**

ist information society technologies

HERIOT WATT UNIVERSITY

University of St Andrews