Integrating Field-Programmable Gate Array (FPGA) Devices Across the Computer Science Curriculum

William M. Jones and D. Brian Larkins
Department of Computer Science
Coastal Carolina University
Conway, SC 29528

**Background**
- Field Programmable Gate Arrays (FPGA) are hardware devices that may be configured through the use of hardware design software.
- An FPGA may be reprogrammed multiple times to accommodate multiple hardware designs.
- A single FPGA device can replace a large variety of dedicated hardware components as well as implement custom hardware configurations.

**Curriculum Integration**

Current courses:
- CSC1210: Computer Organization
- CSC1310: Computer Architecture
- CSC450: Compiler Construction

Future courses:
- CSC410: Operating Systems
- CSC415: Computer Architecture
- CSC450: Compiler Construction
- CSC460: Advanced Computer Architecture

**Motivation**

Conventional Approach
- Requires manual debugging and wire tracing
- Inflexible to rapid changes in circuit layout
- Inventory control and maintenance is complex

FPGA Approach
- Altera DE2 FPGA development board is designed for educational use
- Hardware design is done with a free cross-platform development environment
- The DE2 retains the kinesthetic experience of the conventional breadboard approach

**Logic Design**

- Schematic entry allows the definition of low-level combinational and sequential logic
- Component libraries provide collections of frequently used devices
- FPGA can implement a modern RISC CPU which can be programmed in assembly language
- Students can work with gate structures or modern microprocessors on the same platform

**Computer Architecture**

- Students leverage prior FPGA experience in the computer architecture course
- Exposes students to modern architecture design tools such as VHDL and Verilog
- DE2 supports variety of peripherals: USB, LCD, LED, video, ethernet, audio
- Nios II FPGA CPU supports multiple instruction and cache configurations
- Students are able to perform tradeoff analysis between hardware and software solutions to the same problem

**Sample Tiger Program: Mergesort Algorithm**

```tiger
function sort(a: list) : list =
  if length(a) <= 1 then a
  else if a.first < b.first then
    a := merge(a, b.rest)
  else
    b := nil
    if length(a) <= 1 then a
    else
      b := a
      a := nil
      if length(b) <= 1 then b
      else
        a := merge(a, b.rest)
        b := nil
      end
    end
  end
end
```

- Tiger is a high-level programming language based on the ML functional programming language
- This mergesort implementation illustrates the use of complex data structures by using arrays and linked lists
- The compiler runtime interfaces between the FPGA hardware and built-in programming language functions for allocation and I/O