Addressing Embedded Programming Needs within an ECE Curriculum

Kenneth G. Ricks, David J. Jackson, William A. Stapleton

Workshop on Embedded Systems Education

October 26, 2006

Agenda

- **Introduction**
- **Typical ECE Approach to Addressing Programming Skills**
- **Embedded Programming Needs: Areas of Concern**
- **Curricula Reform Options to Address These Areas**
- **The University of Alabama Experience**
- **Lessons Learned and Recommendations**

Introduction

- As embedded systems become more pervasive in our society, the need for embedded systems engineers is well documented.
- As academia tries to address this need, it must overcome the **breadth problem.**
  - The breadth problem refers to the difficulty of incorporating the broad spectrum of embedded systems topics into the curriculum.
  - How broad is the field? The IEEE/ACM model computer engineering curriculum has an embedded systems component consisting of:
    - 11 knowledge units
    - Covering a total of 59 topics
    - Addressing 34 different learning outcomes [8].

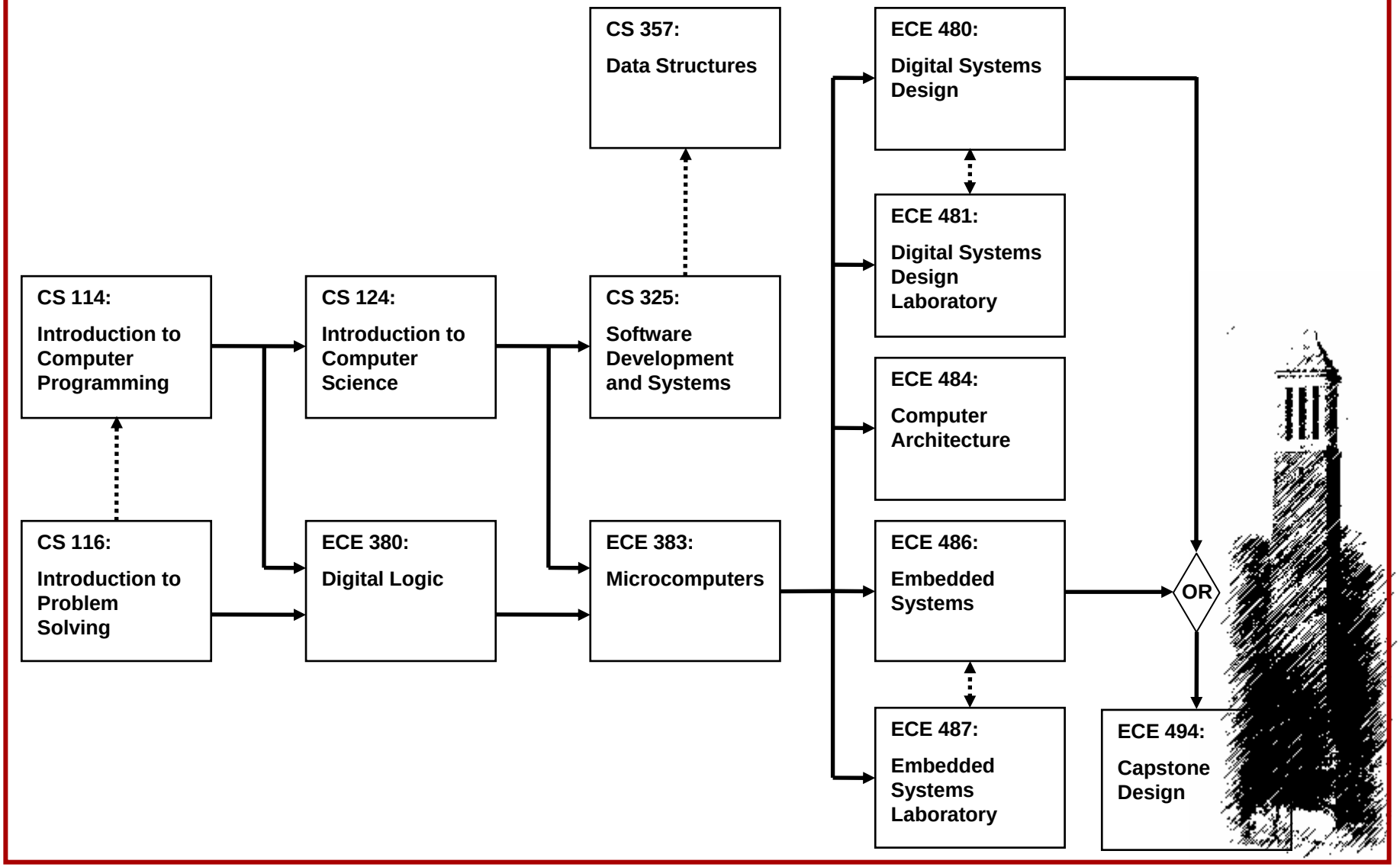
Introduction (continued)

- **In this paper, we limit the discussion to embedded programming skills of ECE students.**
- **Embedded programming skills are of vital importance.**
  - **It is estimated that the amount of embedded software doubles every 10 months and will account for 90% of all software being written by the year 2010 [5].**

Typical ECE Approach

- **Two-level approach**
  - **LEVEL-1: General programming offered very early in the curriculum**
    - **Usually a high-level language such as C or C++**
  - **LEVEL-2: Assembly programming offered later in the curriculum usually associated with a microprocessors or microcontrollers course**

The UA Computer Engineering Course Sequence



Areas of Concern: Embedded Programming Skills

- **Proficiency with a High-Level Language (HLL) applicable to embedded systems**
- **Using registers to interface with peripherals**
- **Program structure**
- **Programming impacts on resource constraints**

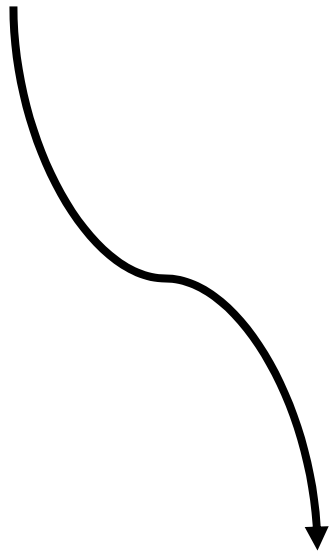
Drawbacks of the Two-Level Approach

- **Introductory high-level programming courses typically incorporate many different concepts in addition to teaching a HLL:**
  - **General problem solving**
  - **Language syntax**
  - **Teaming**
  - **Communication skills**
  - **Algorithm design**
  - **Object-oriented programming techniques.**

Drawbacks of the Two-Level Approach (continued)

- **Microprocessors/microcontrollers assembly language courses typically cover MUCH material including:**
  - **Language syntax and basic assembly program structure**
  - **Interfacing to specific hardware and I/O devices**
  - **System integration**
  - **Program debugging**
  - **Pulse width modulation, A/D and D/A conversion.**
  - **Assembly language is NOT the most popular language for embedded applications.**
    - **In 2000, 80% of embedded applications were written in the C programming language [16].**

HLL Programming (too general)



?

Low-level Programming (limited to assembly language)

Areas of Concern: Examples

- **Choice of High-Level language**
  - **ANSI C programming language**
    - **Prevalence of C for embedded applications**
    - **ANCI C is portable promoting code reuse and shortening time-to-market**
    - **Software development tools are ranked as the number 1 most important factor for microprocessor choice for embedded systems [17].**

Areas of Concern: Examples (continued)

- **Peripheral Interfacing Using Registers**
  - **Access to memory-mapped registers using pointers**

```
int * CSR_ptr = 0xFFAA;
*CSR_ptr = 1;
```

  - **Bitwise operators**

&	bitwise AND operator
|	bitwise OR operator
~	logical NOT operator
:	bitfield structure operator

Areas of Concern: Examples (continued)

- **Peripheral Interfacing Using Registers**
  - **Register scope**

```
int * CSR_ptr = 0xFFAA;
*CSR_ptr = 1;      /* initialize register contents */
... (During this part of the program, the device's status changes
and the register value is overwritten by the device hardware.)
*CSR_ptr = 1;     /* re-initialize register contents */
```

Unsafe code fragment subject to erroneous compiler optimization.

```
volatile int * CSR_ptr = 0xFFAA;
*CSR_ptr = 1;      /* initialize register contents */
... (During this part of the program, the device's status changes and
the register value is overwritten by the device hardware.)
*CSR_ptr = 1;     /* re-initialize register contents */
```

Safe code fragment not subject to erroneous compiler optimization.

Areas of Concern: Examples (continued)

- **Program structure (motivations for specific structures)**
  - **Subroutines vs. in-line code**
    - **The use of subroutines promotes modular code development, code reuse, and facilitates debugging.**
    - **In-line code can execute faster, but it can require more memory.**
    - **Pass-by-reference can minimize subroutine overhead.**
  - **Global vs. local variables**
    - **Global variables can reduce storage requirements.**
    - **Globally shared data must be protected to ensure data coherency.**
  - **Program translation to executable**

Areas of Concern: Examples (continued)

- **Resource restrictions**
  - **Memory limitations**
    - **Memory may be too small to support linked lists and arrays of structures.**
    - **Single bit values should use bitfields instead of int types.**
    - **Bitwise operators require some understanding of bit allocation within a word (big endian vs. little endian).**
  - **Time limitations (real-time constraints)**
    - **Program efficiency must be considered.**
  - **Power limitations**
    - **Power budgets might require slower clocks.**

UA Assessment Data

1) "Describe the difference between the ANSI C bitwise operators (e.g. "&") and the logical-test operators (e.g. "&&"). When is each appropriate?"
   Results = 19%

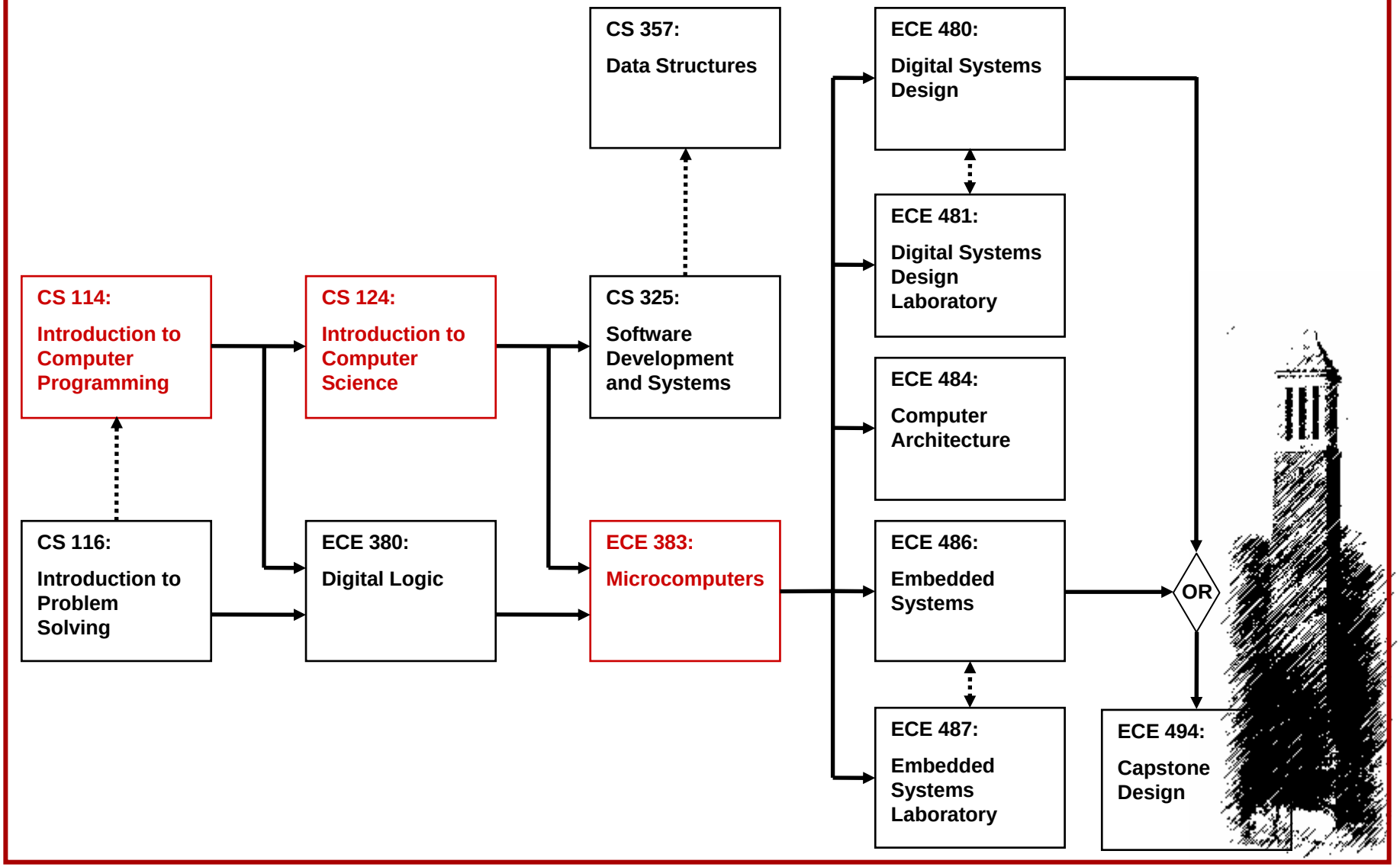
2) "Describe the difference between the ANSI C bitwise-AND operator "&" used like "X = Y & 0x1f" and the ANSI C address operator "&" used like "ptr = &var;". How are these two uses for the same character ("&") distinguished? "
   Results = 22%

3) "In ANSI C, what is a pointer? How is a pointer specified? How is a pointer used? How is the pointer value related to the physical memory system or computer?"
   Results = 31%

4) "How are parameters passed between C subroutines? (i.e. in what format and in what order)"
   Results = 9.4%

5) "What is the difference between global and local variables in ANSI C? How are each type specified? What are the advantages of each type?"
   Results = 16%

The UA Computer Engineering Course Sequence



Conclusions from Assessment Data

- **ECE students show a general lack of overall programming skills.**
- **Upperclassmen demonstrate little retention of the HLL programming skills presented in the introductory programming courses.**
- **Students demonstrate a complete lack of understanding of the HLL constructs necessary for embedded systems programming .**

Curriculum Reform

- **Two options:**
  - **Integrate needed programming skills into existing courses.**
  - **Add courses to the curriculum to address embedded programming needs.**

Curriculum Reform

- **Integrate needed programming skills into introductory programming course(s)?**
- **Integrate needed programming skills into assembly language course?**
  - **These changes may be difficult if the course is taught by another department.**
  - **Requires coordination with existing course(s).**

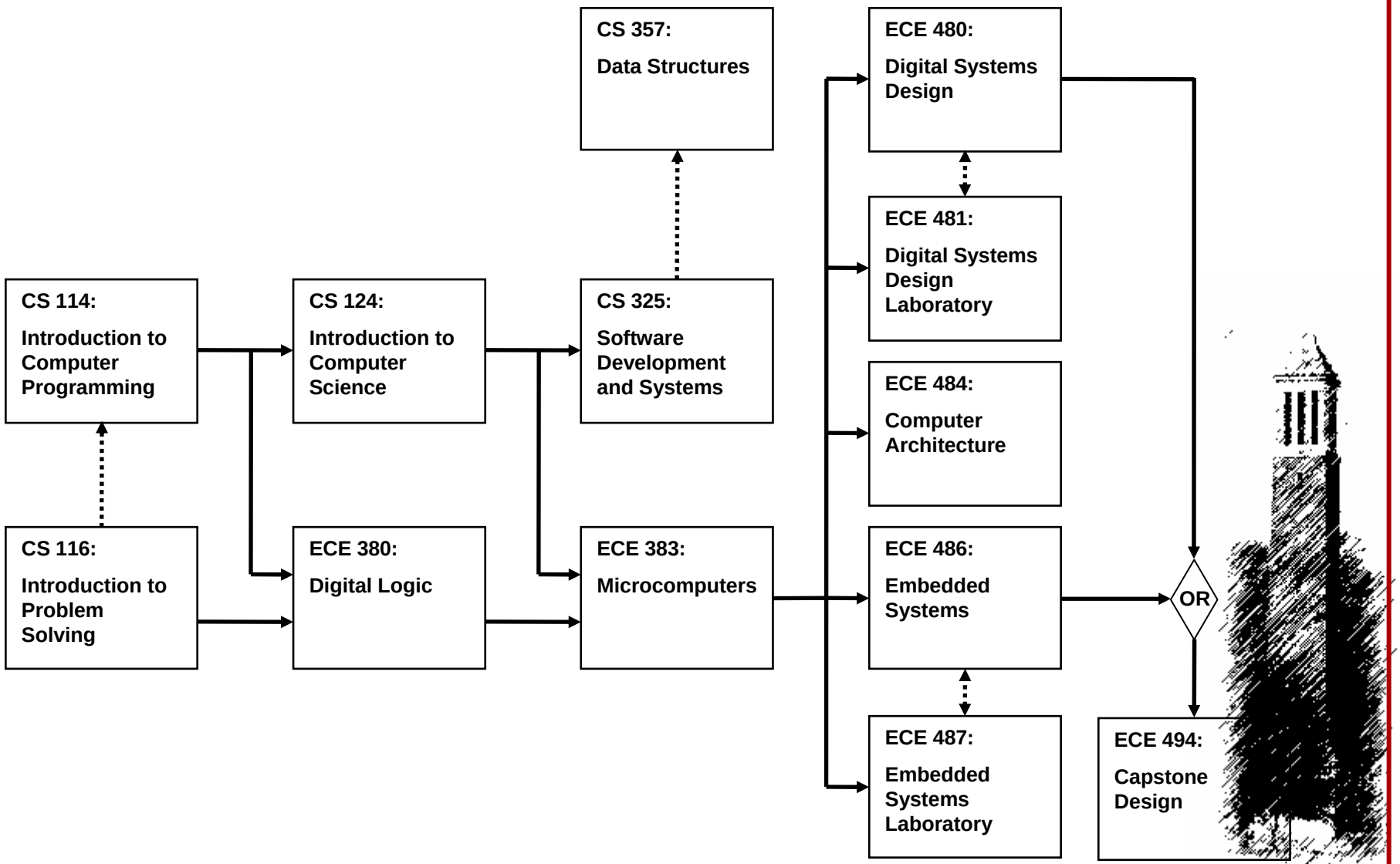
Curriculum Reform (continued)

- **Adding new courses to the curriculum?**
  - **ANSI C introductory programming course could be taught by the ECE department.**
    - **This would provide more control over course content and require less coordination among existing courses.**
    - **Administration may view this as unnecessary redundancy.**

Curriculum Reform (continued)

- **Adding new courses to the curriculum?**
  - **Add a higher-level "Embedded Systems" course to the ECE curriculum?**
    - **This course could cover many aspects of embedded systems including programming skills and could require the assembly course as a prerequisite.**
    - **Fundamental programming skills do not belong in a 3rd – 4th year course.**
    - **Adding hours to any curriculum is difficult if courses cannot be identified for replacement.**

The UA Computer Engineering Course Sequence



Lessons Learned at UA

- **Adding any software component to the ECE curriculum is viewed as a positive.**
  - **Addresses poor performance on the assessment questions.**
  - **Breaks down stereotypes associated with "hardware" and "software" engineers.**
- **Introductory programming skills really do not belong in a 3rd – 4th year "Embedded Systems" course.**
- **Programming skills should take priority over the hardware platform used in an embedded systems course.**

Recommendations

- **Adding embedded programming concepts to introductory programming course(s) does not appear to be a good approach.**
  - **The retention problem still persists.**
  - **At this point, students have no clear understanding of the skill set needed for embedded software development.**
- **An advanced "Embedded Systems" course should be dedicated to more advanced concepts.**
- **Embedded programming concepts should be integrated into the assembly language course.**
  - **Interfacing aspects are already present.**
  - **Programming concepts are already present.**
  - **Natural fit for presentation of HLL and its relation to assembly language.**

References

[1] Bjedov, G., Andersen, P.K., "Should Freshman Engineering Students Be Taught a Programming language?", Proceedings of the 26th Annual Frontiers in Education Conference, Volume 1, Nov. 6-9, 1996, pp. 90-92.

[2] Bramer, B., Bramer, S., *C for Engineers, 2nd Edition*, John Wiley & Sons, New York, New York, 1997.

[3] Budny, D., Lund, L., Vipperman, J., Patzer, J.L.I.I.I., "Four Steps to Teaching C Programming", Proceedings of the 32nd Annual Frontiers in Education Conference, Volume 2, November 6-9, 2002, pp. F1G-18 - F1G-22.

[4] Davenport, D., "Experience Using a Project-Based Approach in an Introductory Programming Course", *IEEE Transactions on Education*, Volume 43, Issue 4, November 2000, pp. 443 – 448.

[5] Ganssle, J., "The Demise of the Embedded Generalist", Embedded.com, Available: http://www.embedded.com/showArticle.jhtml?articleID=51202213, November 2, 2004.

[6] Haberman, B., Trakhtenbrot, M., "An Undergraduate Program in Embedded Systems Engineering", Proceedings of the 18th Conference of Software Engineering Education and Training (CSEET'05), April 18-20, 2005, pp. 103-110.

[7] Harbison III, S. P., Steele Jr., G. L., *C: A Reference Manual, 5th Edition*, Prentice Hall, Upper Saddle River, New Jersey, 2002.

[8] Joint Task Force on Computer Engineering Curricula, IEEE Computer Society, Association for Computing Machinery, "Computer Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering", December 12, 2004, pp. A.43 – A.45, Available: http://www.computer.org/education/cc2001/CCCE-FinalReport-2004Dec12-Final.pdf.

[9] Kernighan, B. W., Ritchie, D. M., *The C Programming Language, 2nd Edition*, Prentice Hall, 1988.

[10] Kochan, S. G., *Programming in ANSI C*, Prentice Hall, Indianapolis, Indiana, 1994.

[11] Nagurney, L.S., "Teaching Introductory Programming for Engineers in an Interactive Classroom", Proceedings of the 31st Annual Frontiers in Education Conference, Volume 3, October 10-13, 2001, Reno, Nevada, pp. S2C - 1-5.

[12] Parrish, A., Borie, R., Cordes, D., Dixon, B., Jackson, J., Pimmel, R., "An Integrated Introductory Course for Computer Science and Engineering", Proceedings of the 29th Annual Frontiers in Education Conference, Volume 1, November 10-13, 1999, pp. 11A3/12 - 11A3/17.

[13] Ricks, K. G., Stapleton, W. A., Jackson, D. J., "An Embedded Systems Course and Course Sequence", in Proc. of the 2005 Workshop on Computer Architecture Education (WCAE), Madison Wisconsin, June 5, 2005, pp. 46-52.

[14] Saks, D., "Representing and Manipulating Hardware in Standard C and C++", Embedded Systems Conference, Session ESC-243, San Francisco, California, March 6-10, 2005, Available: http://newit.gsu.unibel.by/resources/conferences%5Cesc_2004%5CSan_Francisco%5Cesc_243.pdf.

[15] Stapleton, W. A., Ricks, K. G., Jackson, D. J., "Implementation of an Embedded Systems Curriculum", in Proc. of the 20th International Conference on Computers and Their Applications (CATA'05), New Orleans, Louisiana, March 16-18, 2005, pp. 302-307.

[16] 1999/2000 TRON Association Survey, Available: http://www.ncsu.edu/wcae/ISCA2005/submissions/ricks.ppt.

[17] Turley, J., "Survey says: Software Tools More Important Than Chips", Embedded Systems Design, Available: http://www.embedded.com/showArticle.jhtml?articleID=160700620, April 11, 2005.