Paper 135, ENT 207

Role of Simulation Software in Enhancing Student Learning in Computer Organization and Microcontroller Courses

Nikunja K. Swain, James A. Anderson
South Carolina State University
nkswain2001@yahoo.com, jaanderson@scsu.edu

Raghu B. Korrapati
Walden University
raghu.korrapati@waldenu.edu

Abstract

Simulation exercises are an integral part of electrical engineering technology (EET) and computer science (CS) programs. Simulation exercises provide verification of the basic theory; reinforcement of the underlying principles; greater attention to the theoretical limitations; and application of logical analysis to solve real-world problems. Studies show that students who used simulation prior to conducting actual experiments performed better than the students who conducted the laboratory experiments without conducting simulation first. There is no doubt that simulation cannot replace the physical hands-on experience, but simulation can enhance the teaching and learning experience. The objective of this paper is to discuss simulation software packages used in various EET and CS courses with special emphasis on Design Works and EDSIM51 software packages and its effect on student learning in computer organization and microcontroller courses, respectively.

Introduction

The engineering, science, and technology field, at present, is very dynamic due to recent advances in computer and other technologies. These advances have resulted in numerous computer programs to solve traditional and novel problems. These programs use the computer's increased computational capabilities and assist in the design, development, and control of complex systems in a matter of minutes. Automation is becoming part and parcel of every industry, and industries need a trained workforce to manage this new development. Engineering and technology graduates must have a comprehensive background covering a wider range of technical subjects. The graduates must be proficient in the use of engineering and scientific equipment, conducting experiments, collecting data, and effectively presenting the results [1, 2, 3, 4]. This is especially true for the graduates of engineering, technology, and science. Furthermore, these graduates must be well-trained in courses and laboratories such as electric and electronic circuits; digital systems and microprocessors; computer programming; computer aided design; computer organization and architecture; electronic and data communications; networking; control and robotics; electric machines and power systems; PLC and virtual instrumentation; and others. One cost-effective way of achieving this is through the use of simulation software programs, and a number of simulation software packages are available for these purposes. These software packages play an important role in education and are used to deliver training for all kinds of activities, from piloting sophisticated aircraft or ships to operating nuclear power plants or complex chemical processing facilities.

There are numerous uses of simulation, starting from simulation of simple electric circuits to complex tasks such as electromagnetic fields, heat transfer through materials, networking, computer circuits, game programming, electron flow in semiconductors, or beam loading with the ultimate objective of providing illustrations of concepts that are not easily visualized and difficult to understand. Simulators are also used as an adjunct to and, in some cases such as distance learning courses, as a substitute for actual laboratory experiments. In many instances, students are required to verify their theoretical design through simulation before building and testing the circuit in the laboratory. Studies show that students who used simulation prior to conducting actual experiments performed better than the students who conducted the laboratory experiments without conducting simulation first. Also, simulation is used to model large and complex systems. There is no doubt that simulation cannot replace the physical hands-on experience, but simulation can enhance the teaching and learning experience.

The objective of this paper is to discuss Design Works and EDSIM51 software packages and its effect on student learning in computer organization and microcontroller courses, respectively.

A Partial List of Simulation Software Used in EET and CS Programs

A partial listing of simulation software programs used in EET and CS programs are shown in Table 1 [5, 6]. These programs are either used as stand-alone teaching tools or in conjunction with other tools. For example, a student may use one package to get the experimental data and another a spreadsheet package, such as Excel, for plotting and data analysis. Table 1 lists a few of the most widely used EET and CS simulation software packages.

Table 1 – List of Widely Used Software Packages in EET and CS

Name of Software	**Primary Application Areas**
PSPICE	Electric and Electronic Circuits (Analog and Digital)
Electronics Workbench (Multisim)	Electric and Electronic Circuits (Analog and Digital), Communication
VisSim	Electric and Electronic Circuits (Analog and Digital), Communication
Logic Works	Digital/Microprocessor Design
Design Works	Digital/Microprocessor Design/Computer Organization
MatLab	Mathematics, Control Systems, Power Systems
Mathematica	Mathematics

MathCad	Mathematics
AutoCad	Computer Aided Drafting (CAD)
Simulink	Control and Power Systems
LabVIEW	Control, Signal Processing, Mathematical Simulation
Excel	Spreadsheet for Multipurpose Activities
UMPS	Microprocessors and Microcontrollers
UV151	Microprocessor and Microcontrollers
MASM	Microprocessors
DEBUG	Microprocessors
RSLOGIX	Programmable Logic Controller

Many of the software packages listed above are used in various electrical engineering technology courses at SCSU to assist the faculty and students in teaching and learning as shown in Table 2.

Table 2 – List of Simulation Software Package Used in EET and CS Programs at SCSU.

Name of Software	**Primary Application Areas**
PSPICE, Electronic Workbench (MultiSim)	Electric and Electronic Circuits (Analog and Digital), Electronic Communication
Logic Works	Digital/Microprocessor Design
Design Works	Digital/Microprocessor Design/Computer Organization
MatLab & Simulink	Mathematics, Control Systems, Power Systems
LabVIEW	Control, Signal Processing, Mathematical Simulation, Power Systems, Electric Circuits, and Electronics
Excel	Spreadsheet for Multipurpose Activities
MASM	Microprocessors
DEBUG	Microprocessors
EDSIM 51	8051 Microcontroller
RSLOGIX	Programmable Logic Controller

Examples of Application of Simulation Software to EET and CS Programs at SCSU

As presented in Table 2, various simulation software packages are currently being used by the EET and CS programs at SCSU to enhance teaching and learning. The faculty at SCSU have developed a number of modules for course and laboratory use. Packages like PSPICE, Multisim, MatLab, Simulink, LogicWorks, RSLogix, Debug, MASM, and LabVIEW are widely used by engineering and technology programs at other institutions, and there is sufficient information on these in textbooks and on the Web. Packages such as Design Works and EDSIM 51 are not that well-known and may not be widely used, but both of these packages have tremendous potential for enhancing student learning in computer organization and microcontroller courses. We will discuss these software packages and the instructional modules developed using these packages below.

Examples of DesignWorks Instructional Modules

DesignWorks[7] is a logic schematic creation and simulation program. It comes with many circuit symbols and models that can be used to design and simulate various types of analog and digital circuits. DesignWorks comes with libraries of various types of components required to construct and simulate various types of circuits and systems. For understanding basic circuit and logic designs, one need not to use the component libraries that model real components used on a circuit board. Instead, one should be using the components in the following four libraries: Pseudo Devices.clf, Simulation IO.clf, Simulation Gates.clf, and Simulation Logic.clf.

Combinational and Sequential Design Modules

The objectives of these modules are to assist the student in validating the theoretical design process to have a better understanding of the concept. The student completes the design theoretically and derives the logic circuit. For the combinational design, the student completes steps such as Truth Table, K-MAP simplification, and minimized logic diagram. For the sequential design, the student completes steps such as State Table, Flip-Flop Excitation Table, Excitation Table for the design problem, K-MAP simplification of the Flip-Flop inputs, and minimized diagram. The student then builds the circuit in the DesignWorks simulator, simulates the circuit using appropriate inputs, and verifies the output to validate the theoretical design. Following this, the circuit can be built in the laboratory using physical components and tested using actual physical signals. The following are examples of two such modules.

Design and Simulation of a Three-bit Synchronous UP Counter Using J-K Flip-Flops

This counter consists of three states—A, B, and C (output of three JK Flip-Flops)—and one input X. When X is zero, the state of the counter doesn't change. When X = 1, the counter goes through a repeated sequence of 000, 001, 010, 011, 100, 101, 110, and 111. Table 3 through Table 6 and Figure 1 through Figure 3 present the theoretical solution steps, and Figure 4 presents the corresponding DesignWorks simulation results.

Theoretical Design

Table 3 – State Table

Present State (PS)			Input	Next State (NS)		
Y3	y2	y1	X	Y3	Y2	Y1
0	0	0	0	0	0	0
0	0	0	1	0	0	1
0	0	1	0	0	0	1
0	0	1	1	0	1	0
0	1	0	0	0	1	0
0	1	0	1	0	1	1
0	1	1	0	0	1	1

0	1	1	1	1	0	0
1	0	0	0	1	0	0
1	0	0	1	1	0	1
1	0	1	0	1	0	1
1	0	1	1	1	1	0
1	1	0	0	1	1	0
1	1	0	1	1	1	1
1	1	1	0	1	1	1
1	1	1	1	0	0	0

Table 4 – J-K Flip-Flop Excitation Table

PS (y)	NS (Y)	J	K
0	0	0	D
0	1	1	D
1	0	d	1
1	1	d	0

d = Don’t care

Table 5 – Counter Excitation Table

PS			Input	NS			FF3		FF2		FF1	
y3	y2	y1	x	Y3	Y2	Y1	J3	K3	J2	K2	J1	K1
0	0	0	0	0	0	0	0	d	0	d	0	d
0	0	0	1	0	0	1	0	d	0	d	1	d
0	0	1	0	0	0	1	0	d	0	d	d	0
0	0	1	1	0	1	0	0	d	1	d	d	1
0	1	0	0	0	1	0	0	d	d	0	0	d
0	1	0	1	0	1	1	0	d	d	0	1	d
0	1	1	0	0	1	1	0	d	d	0	d	0
0	1	1	1	1	0	0	1	d	d	1	d	1
1	0	0	0	1	0	0	d	0	0	d	0	d
1	0	0	1	1	0	1	d	0	0	d	1	d
1	0	1	0	1	0	1	d	0	0	d	d	1
1	0	1	1	1	1	0	d	0	1	d	d	0
1	1	0	0	1	1	0	d	0	d	0	1	d
1	1	0	1	1	1	1	d		d	0	1	d
1	1	1	0	1	1	1	D	0	d	0	d	0
1	1	1	1	0	0	0	D	1	d	0	d	1

Table 6 – Modified Counter Excitation Table with PS, Input X, and J-K inputs

PS			Input	FF3		FF2		FF1	
y3	y2	y1	x	J3	K3	J2	K2	J1	K1
0	0	0	0	0	d	0	D	0	d
0	0	0	1	0	d	0	D	1	d
0	0	1	0	0	d	0	D	d	0
0	0	1	1	0	d	1	D	d	1
0	1	0	0	0	d	D	0	0	d
0	1	0	1	0	d	D	0	1	d
0	1	1	0	0	d	D	0	d	0
0	1	1	1	1	d	D	1	d	1
1	0	0	0	d	0	0	D	0	d
1	0	0	1	d	0	0	D	1	d
1	0	1	0	d	0	0	D	d	1
1	0	1	1	d	0	1	D	d	0
1	1	0	0	d	0	D	0	1	d
1	1	0	1	d		D	0	1	d
1	1	1	0	D	0	D	0	d	0
1	1	1	1	D	1	D	0	d	1

K-MAPs and Logic Equations for J and K Inputs of the FFs Using Table 6

y1x

y3y2	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	d	d	d	d
10	d	d	d	d

J3 = xy1y2

y1x

y3y2	00	01	11	10
00	d	D	d	d
01	d	D	d	d
11	0	0	1	0
10	0	0	0	0

K3 = xy1y2

Figure 1 – KMAP for J3 and K3

y1x

y3y2	00	01	11	10
00	0	0	1	0
01	D	D	d	d
11	D	D	d	d
10	0	0	1	0

J2 = xy1

y1x

y3y2	00	01	11	10
00	d	D	d	d
01	0	0	1	0
11	0	0	1	0
10	d	D	d	d

K2 = xy1

Figure 2 – KMAP for J2 and K2

	y1x			
y3y2	00	01	11	10
00	0	1	d	d
01	0	1	d	d
11	0	1	d	d
10	0	1	1	d

J1= x

	y1x			
y3y2	00	01	11	10
00	d	D	1	0
01	d	D	1	0
11	d	D	1	0
10	d	D	d	0

K1 = x

Figure 3 – KMAP for J1 and K1

DesignWorks Simulation Results

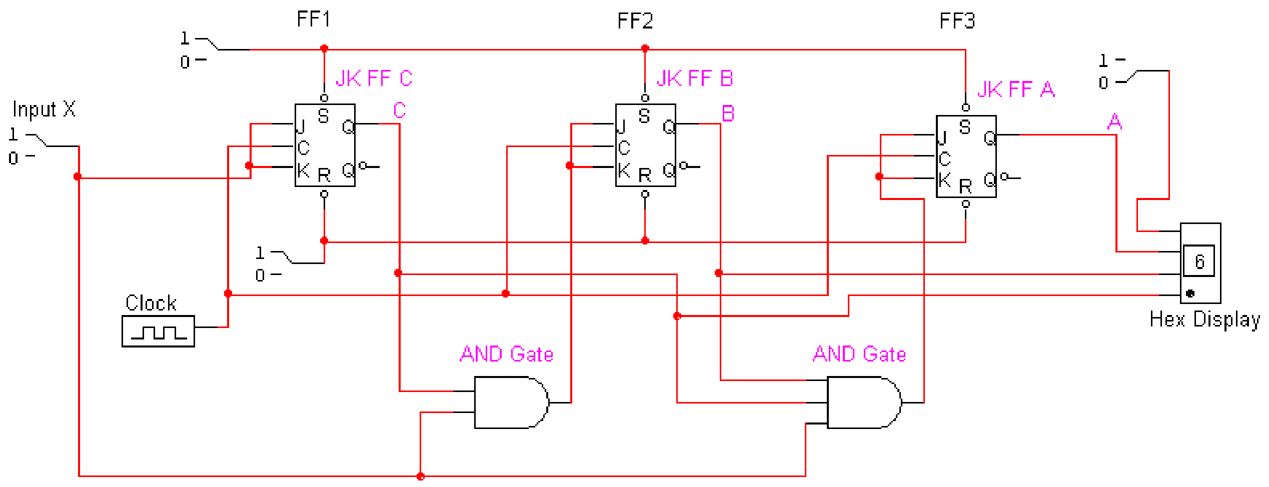


Figure 4 – Three-bit Synchronous Counter

Bus System for Four Registers [8]

A typical digital computer has many registers, and paths must be provided to transfer information from one register to another. The number of wires will be excessive if separate lines are used between each register and all other registers in the system. A more efficient scheme for transferring information between registers in a multiple-register configuration is a common bus system. One way of constructing a common bus system is with multiplexer. The module shown in Figure 5 is a bus system for four registers—A, B, C, and D—with four bits. The control signals S1, S0 selects the register whose content will be available on the bus. The four possible binary S1, S0 values and the register selected are as follows:

S1 S0 = 00, Register Selected = A; S1 S0 = 01, Register Selected = B; S1 S0 = 10, Register Selected = C; S1 S0 = 11, Register Selected = D.

Table 7 presents this function table for this bus system.

Table 7 – Function Table for Bus System

S1	S0	Register Selected
0	0	A
0	1	B
1	0	C
1	1	D

In Figure 4, S1, S0 = 00; therefore, the register selected = A. Since the A register is connected to 1011, the Hex displays 1011.

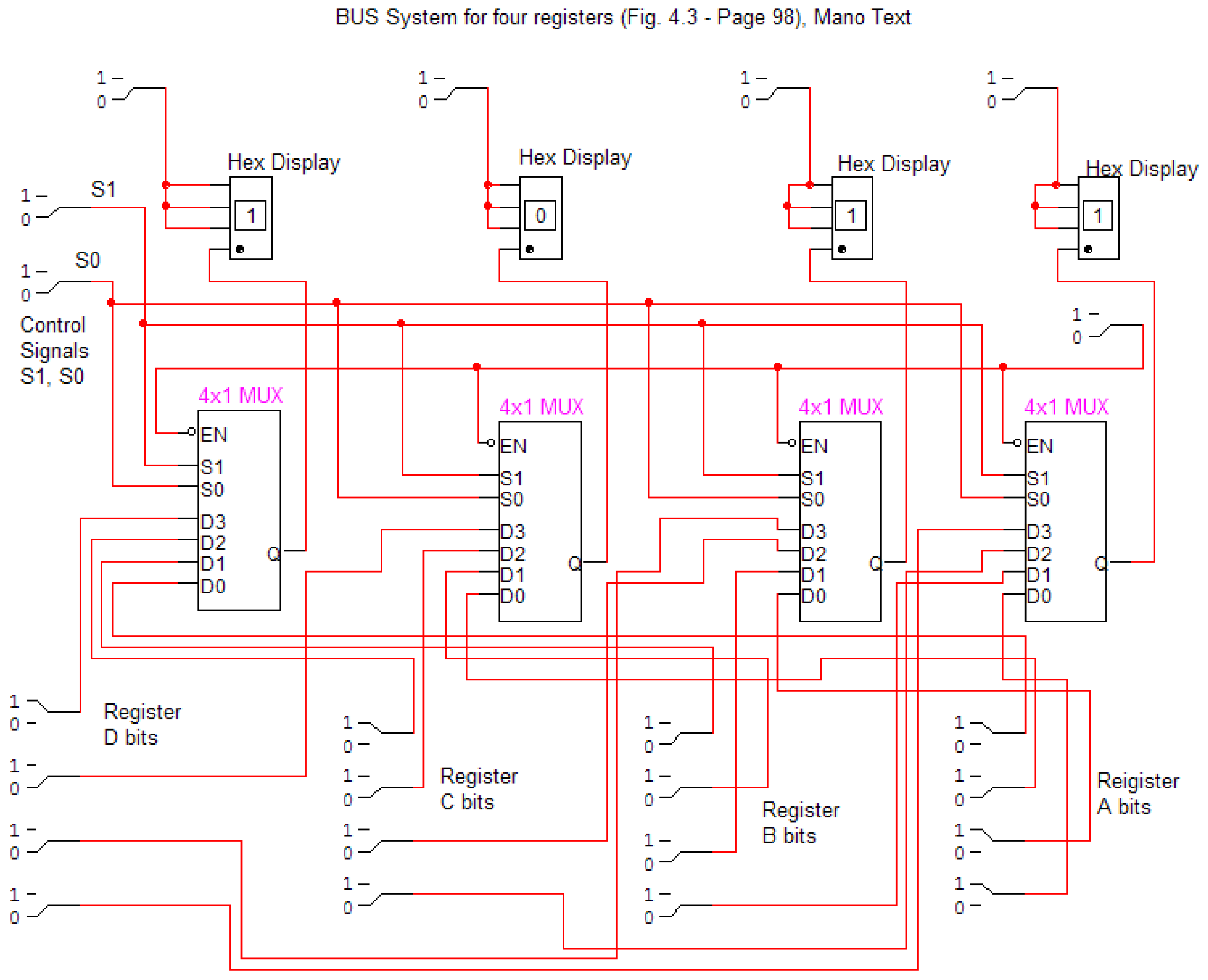


Figure 5 – Bus System for Four Registers

EDSIM51 Instructional Module

The EdSim51 [9] Simulator is a free simulator for the popular 8051 microcontroller. In EDSIM51, a virtual 8051 is interfaced with virtual peripherals, such as analog-to-digital converter (ADC); comparator; UART; four multiplexed seven-segment displays; liquid crystal display; four X 3 keypads; eight LEDs; DC motor; eight switches; and digital-to-analog converter (DAC). Students write 8051 assembly code, load it to the simulator, assemble it, and execute it to understand the concept better. Students also use the simulator to complete the laboratory experiments in traditional and online environments. Various types of 8051 course exercises are tested using this simulator, and an example of one such laboratory exercise, along with parts of the student laboratory experiment, is presented below.

Problem Statement

(a) Find the sum of the values 79_{16}, $F5_{16}$, and $E2_{16}$. Put the sum in register R0 (low byte) and R5 (high byte).

(b) Write a program to toggle all bits of port 1 by sending to it the values 55H and AAH continuously. Put a time delay in between each issuing of data to port 1.

Solutions

1. Expected Results

The expected results of this program are listed below.

(a) 16-Bit Addition

After executing the program to sum $E2_{16}$, 79_{16}, and $F5_{16}$, the following values are expected to be stored in the indicated registers:

Register	Value (Hex)
ACC	50 (1001 0000 in binary)
R5	02
R0	50
R1	79
R2	F5

(b) Port 1 Bit Toggling

During program execution, the sequence of operations is as follows:

i. The accumulator will be loaded with the initial bit pattern $AA_{16.}$

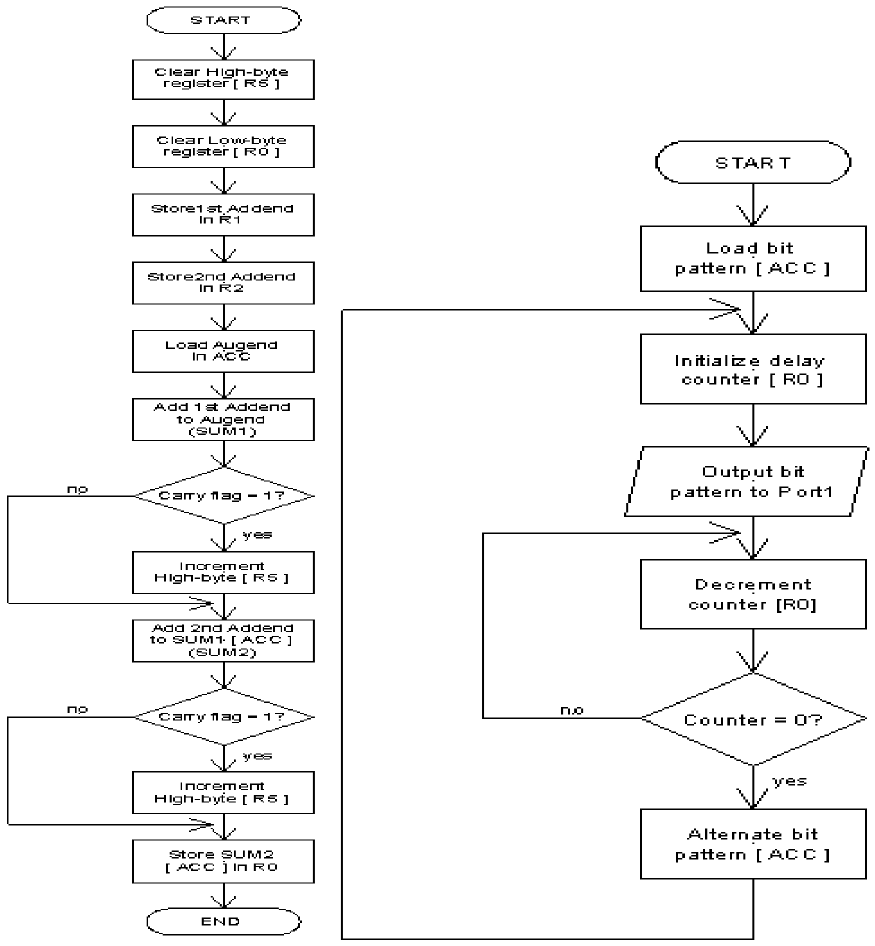
ii. The delay counter [R0] will be initialized (10_{16}).

iii. The bit pattern stored in the accumulator will be output to port.

iv. A delay will be executed, wherein the delay counter [R0] will decrement by one down to zero.
v. Once the delay counter reaches zero, the bit pattern [ACC] will be rotated right, resulting in a bit value shift from zero to one and one to zero.
vi. The delay counter will be reloaded and the routine repeated.

The port 1 value will be 55_{16} and $AA_{16,}$ alternately, following the values stored in the accumulator.

2. Flow Chart


START
Clear High-byte register [R5]
Clear Low-byte register [R0]
Store1st Addend in R1
Store2nd Addend in R2
Load Augend in ACC
Add 1st Addend to Augend (SUM1)
Carry flag = 1?
no
yes
Increment High-byte [R5]
Add 2nd Addend to SUM1 [ACC] (SUM2)
Carry flag = 1?
no
yes
Increment High-byte [R5]
Store SUM2 [ACC] in R0
END
START
Load bit pattern [ACC]
Initialize delay counter [R0]
Output bit pattern to Port1
Decrement counter [R0]
Counter = 0?
no
yes
Alternate bit pattern [ACC]

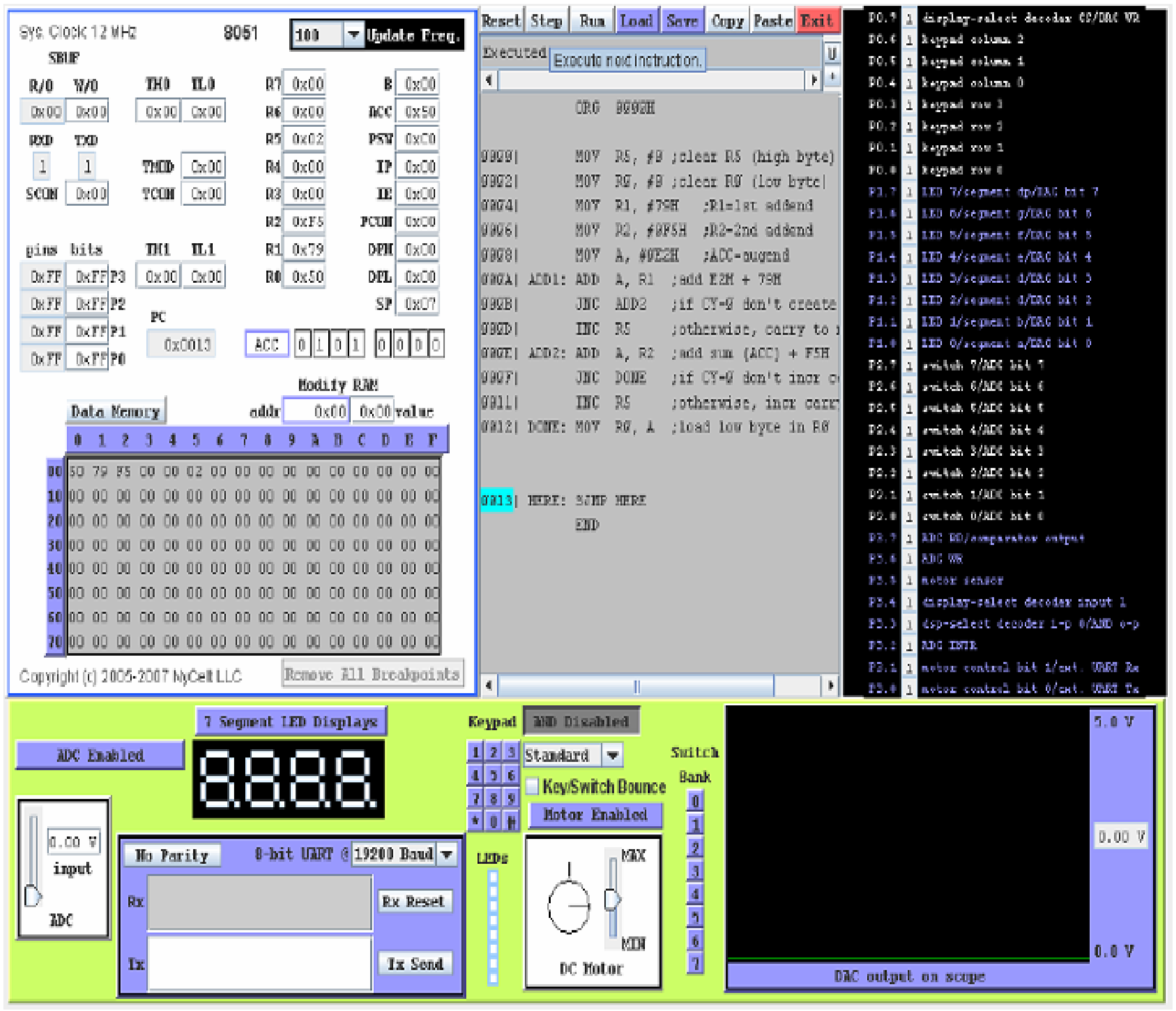
3. Assembly Language Program Listing

Table 8 – Assembly Language Program Listings for Parts A and B

(a) Program for 16-bit Addition	(b) Port1 Bit Toggling Listing
ORG 0000H	ORG 0000H
MOV R5, #0 ;clear R5 (high byte)	MOV A, #0AAH ;init bit pattern
MOV R0, #0 ;clear R0 (low byte)	LOOP: MOV R0, #16 ;initialize delay counter
MOV R1, #79H ; R1=1st addend	MOV P1, A ;send bit pattern to Port1
MOV R2, #0F5H; R2=2nd addend	DELAY: DJNZ R0, $;delay loop
MOV A, #0E2H; ACC=augend	RR A ;alternate bit pattern
ADD1: ADD A, R1; add E2H + 79H	SJMP LOOP ;repeat
JNC ADD2; if CY=0 don't create carry	
INC R5;otherwise, carry to next col	END
ADD2: ADD A, R2; add sum (ACC) + F5H	
JNC DONE; if CY=0 don't incr carry	
INC R5; otherwise, incr carry	
DONE: MOV R0, A; load low byte in R0	
HERE: SJMP HERE	
END	

4. EDSIM Simulation Results

The corresponding EDSIM51 simulation results are shown in Figures 6 and 7. The simulation results agree with the expected results.



R/O 0x00	W/O 0x00	TH0 0x00	TL0 0x00
RXD 1	TXD 1	TMOD 0x00	TCON 0x00
SCON 0x00		TH1 0x00	TL1 0x00
R7 0x00	R6 0x00	R5 0x02	R4 0x00
R3 0x00	R2 0xF5	R1 0x79	R0 0x50
B 0x00	ACC 0x50	PSW 0xC0	IP 0x00
IE 0x00	PCON 0x00	DPH 0x00	DPL 0x00
SP 0x07	PC 0x0013	ACC 0 1 0 1 0 0 0 0	
P3 0xFF 0xFF	P2 0xFF 0xFF	P1 0xFF 0xFF	P0 0xFF 0xFF

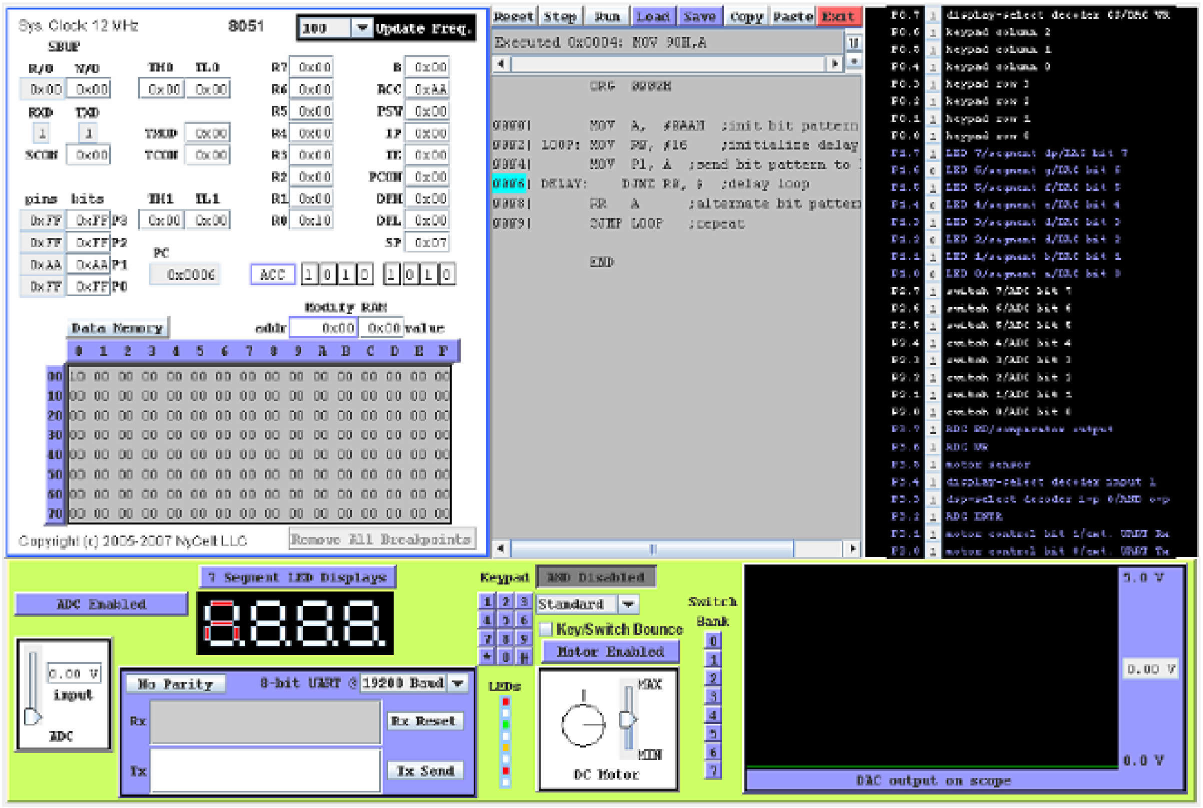
```
   0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
00 50 79 F5 00 00 02 00 00 00 00 00 00 00 00 00 00
```

```
             ORG   0000H

0000|        MOV   R5, #0 ;clear R5 (high byte)
0002|        MOV   R0, #0 ;clear R0 (low byte)
0004|        MOV   R1, #79H   ;R1=1st addend
0006|        MOV   R2, #0F5H  ;R2=2nd addend
0008|        MOV   A, #0E2H   ;ACC=augend
000A| ADD1:  ADD   A, R1   ;add E2H + 79H
000B|        JNC   ADD2    ;if CY=0 don't create
000D|        INC   R5      ;otherwise, carry to
000E| ADD2:  ADD   A, R2   ;add sum (ACC) + F5H
000F|        JNC   DONE    ;if CY=0 don't incr c
0011|        INC   R5      ;otherwise, incr carr
0012| DONE:  MOV   R0, A   ;load low byte in R0


0013| HERE:  SJMP HERE
             END
```

Figure 6 – EDSIM51 Output for 16-bit Addition (Part A)



```
Executed 0x0004: MOV 90H,A

                ORG   0000H

0000|           MOV   A,  #0AAH  ;init bit pattern
0002|   LOOP:   MOV   R0, #16    ;initialize delay
0004|           MOV   P1, A   ;send bit pattern to
0006|   DELAY:      DJNZ R0, $   ;delay loop
0008|           RR    A          ;alternate bit pattern
0009|           SJMP  LOOP    ;repeat

                END
```

```
P0.7  1  display-select decoder CS/DAC WR
P0.6  1  keypad column 2
P0.5  1  keypad column 1
P0.4  1  keypad column 0
P0.3  1  keypad row 3
P0.2  1  keypad row 2
P0.1  1  keypad row 1
P0.0  1  keypad row 0
P1.7  1  LED 7/segment dp/DAC bit 7
P1.6  0  LED 6/segment g/DAC bit 6
P1.5  1  LED 5/segment f/DAC bit 5
P1.4  0  LED 4/segment e/DAC bit 4
P1.3  1  LED 3/segment d/DAC bit 3
P1.2  0  LED 2/segment c/DAC bit 2
P1.1  1  LED 1/segment b/DAC bit 1
P1.0  0  LED 0/segment a/DAC bit 0
P2.7  1  switch 7/ADC bit 7
P2.6  1  switch 6/ADC bit 6
P2.5  1  switch 5/ADC bit 5
P2.4  1  switch 4/ADC bit 4
P2.3  1  switch 3/ADC bit 3
P2.2  1  switch 2/ADC bit 2
P2.1  1  switch 1/ADC bit 1
P2.0  1  switch 0/ADC bit 0
P3.7  1  ADC RD/comparator output
P3.6  1  ADC WR
P3.5  1  motor sensor
P3.4  1  display-select decoder input 1
P3.3  1  disp-select decoder inp 0/AND o/p
P3.2  1  ADC INTR
P3.1  1  motor control bit 1/ext. UART Rx
P3.0  1  motor control bit 0/ext. UART Tx
```

Figure 7 – EDSIM51 Output for Port Toggling Program (Part B)

Conclusions

The sample modules presented above are user-friendly and performed satisfactorily under various input conditions. These and other modules helped the students understand the concepts in more detail. Because of the space limitations of this paper, we were not able to present other modules. These modules can be used in conjunction with other teaching aids to enhance student learning in various courses and will provide a truly modern environment in which students and faculty members can study engineering, technology, and sciences at a level of detail. The EDSIM51 simulation module is currently being used by one of the authors to teach an online 8051 Microcontroller course, and it has helped the students to understand the concept better.

Acknowledgement

This work was funded in part by a grant from the NSF-HBCU-UP/RISC grant. We are thankful to the NSF and SCSU for providing us with this help. We are also thankful to Kenneth Black, an EET student at Excelsior University, for his help in the EDSIM51 example.

References

[1] Swain, N. K., Korrapati, R., Anderson, J. A, "Revitalizing Undergraduate Engineering, Technology, and Science Education Through Virtual Instrumentation," NI Week Conference, Austin, TX, 1999.

[2] Elaine L., Mack, Lynn G, "Developing and Implementing an Integrated Problem-based Engineering Technology Curriculum in an American Technical College System," *Community College Journal of Research and Practice*, Vol. 25, No. 5–6, pp. 425–439, 2000.

[3] Buniyamin, N, Mohamad, Z., "Engineering Curriculum Development: Balancing Employer Needs and National Interest—A Case Study," Retrieved from ERIC database, 2000.

[4] Kellie, Andrew C., et al., "Experience with Computer-Assisted Instruction in Engineering Technology," *Engineering Education*, Vol. 74, No. 8, pp 712–715, 1984.

[5] URL: http://www.idsia.ch/~andrea/simtools.html#libraries

[6] URL: http://thelearningpit.com

[7] URL: http://www.capilano.com

[8] Mano, Morris M. *Computer System Architecture*, Prentice Hall, 1993.

[9] URL: http://www.EDSIM51.com

Biography

Nikunja K. Swain is a professor at South Carolina State University. Dr. Swain has more than 25 years of experience as an engineer and educator. He has more than 50 publications in journals and conference proceedings; has procured research and development grants from the NSF, NASA, DOT, DOD, and DOE; and reviewed multiple books on computer-related subjects. He is also a reviewer for ACM Computing Reviews, IJAMT, CIT, ASEE, and other conferences and journals. He is a registered Professional Engineer in South Carolina.

James A. Anderson's areas of specialization are in electro-optics, solid-state devices/microelectronics, and microwave and optical communications. He has performed research and design at various industries, worked as a consultant and professional engineer, and has been a university professor and Dean of the School of Engineering Technology and Sciences (SETS) at South Carolina State University. Currently, he serves as Professor and Manager of HBCU/UP grant at South Carolina State University.

Raghu Korrapati is a faculty member in the Applied Management and Decision Sciences program in the School of Management at Walden University, Minneapolis, MN. Dr. Korrapati has published numerous papers and acted as reviewer for multiple journals and conferences. Currently, he is the editor in chief of the *International Journal on Applied Technology* (IJAMT).