Interactive Learning **Aid for Microcontroller Architecture and Assembly Language Programming in an Introductory Microprocessor Course**

Jeffrey W. Honchell
Purdue University

ABSTRACT

Assembly language programming and its relationship to the microcomputer architecture poses a significant new challenge to the Electrical Engineering Technology student. Although the hardware and software concepts in an introductory microprocessor course are usually straightforward, the development of the skills required to achieve a solid understanding of them are not. The personal computer is an ideal instrument that can be used to deliver a training tool that would be very effective in developing their understanding of microcomputer architecture and its relationship to assembly language programming. The intent of this project was to develop a "windows" based "point and click" learning tool that utilizes graphics, animation and text. Then integrate the tool into introductory microprocessor course where it would function as a "personal" instructor for each student.

Alternate approaches to integrating the learning aid into this course, and courses at other universities will be explored. Factors to be evaluated include effectiveness in helping the student with microcomputer fundamentals and programming, and acceptance by both the faculty and students.

THE PROBLEM

Assembly language programming and its relationship to the microcomputer architecture is a basic skill that must be developed by the student to succeed in an introductory microprocessor course. Electrical Engineering Technology (EET) students receive thorough instruction in both the hardware and software facets of microcomputers in the introductory microprocessor course (EET 205). However, the only way to master the skill of programming is by achieving a thorough understanding of the architecture and a considerable amount of programming practice. The hypothesis that forms the basis for the need, for this interactive learning aid, is that a student's level of success in developing this required skill would be greatly enhanced. This would allow the student to concentrate on his or her specific weaknesses, at his/her own individual pace.

Programs in an introductory microprocessors course are conceptually simple. However, the logical step-by-step thought process and the strong connection between the microcomputer architecture and the instruction set are new ideas for most students. Most of the difficulty that the student encounters lies in the lack of understanding the connection between the architecture and the instruction set. Since students are required to write assembly language programs that function correctly in the course laboratory, they will be motivated to use the tool provided -- if they believe that the tool will help them in understanding the methods necessary to successfully complete their assigned programs.

SIGNIFICANCE OF THE PROBLEM

Microcomputer programming is part science and part art learned only through considerable practice coupled with guidance from an experienced instructor. This learning is an interactive process in which the student must examine the problem, decide what features of the microcomputer hardware are applicable and, after significant interaction with the instructor, select an appropriate approach. The student then attempts to break the problem down into small manageable tasks and then learn which assembly language instructions are available to accomplish these tasks. This process usually requires frequent consultation with the instructor. It is observed in the laboratory that, without periodic and frequent assistance from the instructor, students typically become frustrated, lose confidence and abandon the systematic procedures they have been taught for a usually futile trial and error approach and, this frequently creates a reliance on other student's course files. The development of this skill is usually a slow and frustrating one, since instructors are unable to provide the personal, focused and sustained support that each student may require for success.

THE LEARNING AID

An interactive Personal Computer (PC) based software system acting as a personal instructor has the potential to provide personalized instruction to each student in EET 205. Over 150 students register for this course each year at the West Lafayette campus and the Purdue Statewide Technology sites. With this tool, each student can proceed at his or her own pace and focus on his/her specific areas of instructional need. The tool will help students in learning the microcomputer architecture and its relationship to writing assembly language programs required in the EET 205 laboratory.

The PC based software learning tool, running in the Microsoft Windows environment, provides the students with a tool that will allow them to select the specific items of interest through a series of "pull-down" menus or button-bar selections, as compared to the standard tutorial format. The learning aid is initially supporting the Signetics 8051 and 80C552 microcontrollers, with the Motorola 68HC 11 to be added in the future. Items such as processor operating frequency, amount of external memory, etc. are set by the user, to provide greater flexibility in supporting numerous implementations using these microcontrollers.

The following may be selected from the initial window:

- Microcontroller architecture
- Timing
- Instruction set
- Program edit
- Program assembly
- Program simulation
- Program execution

For each of the functions listed there are a series of graphics and animation coupled with "pop-up" text windows providing the desired information in a way that is easy to visualize and comprehend. Now, let's look at the function of each of the items listed above in more detail.

Microcontroller Architecture

For example, if the 80C552 microcontroller were selected the graphical representation of the internal

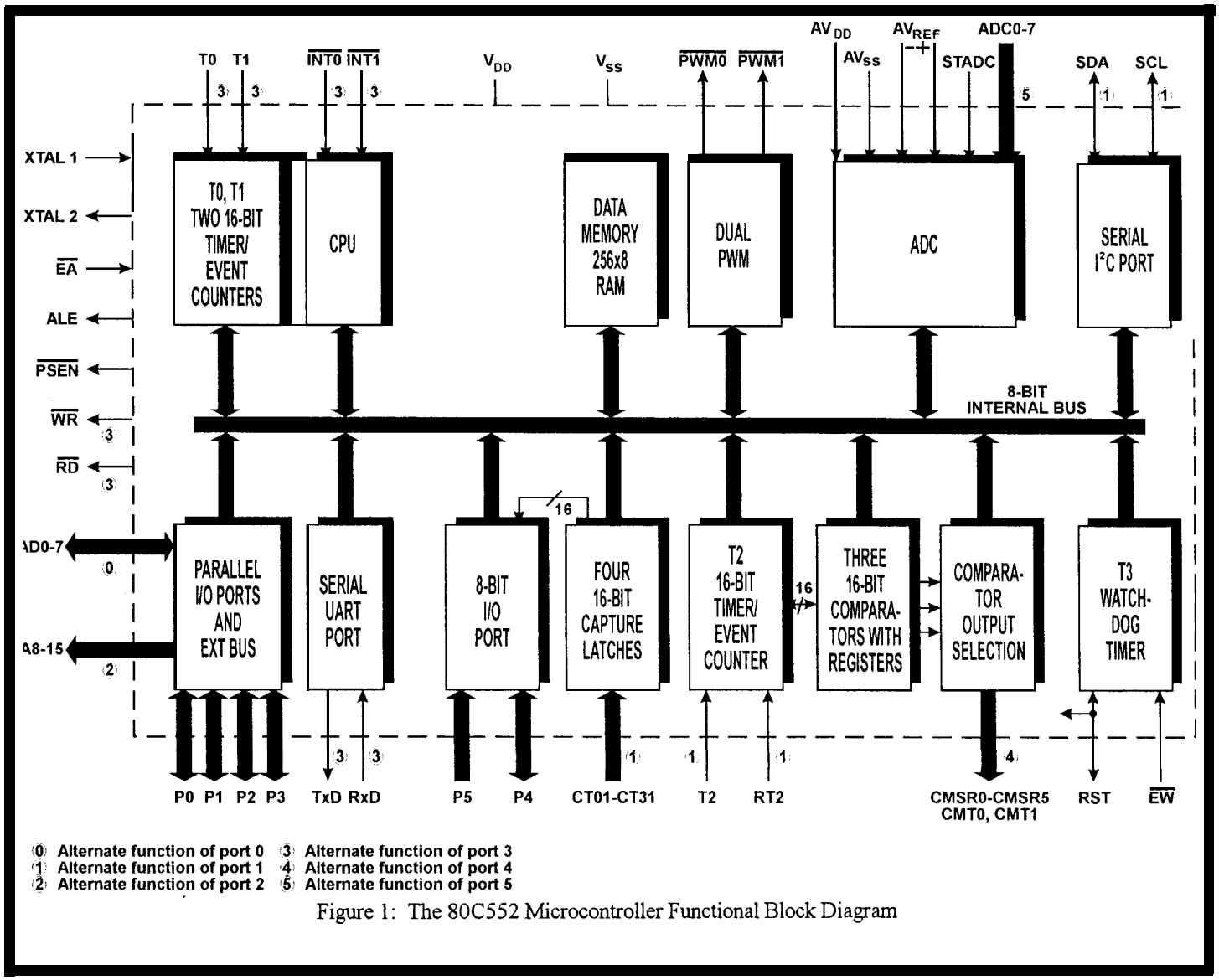

Figure 1: The 80C552 Microcontroller Functional Block Diagram

architecture, shown in Figure 1, would be displayed. In reality, each of the blocks and busses are color coded by function. At this point the student may receive additional information on any of the blocks or busses simply by "clicking" on them. If the student were to select the “interrupts” block, the diagram of Figure 2 would be displayed. This shows a more detailed model of the interrupt system, where the student may again select any part of the diagram to receive an explanation of that section including the role of the individual Special Function Registers (SFR) used by this microcontroller subsystem In this case it would be possible to select information on the following SFRs:

- Interrupt ENable registers (IEN0 and IEN1)
- Interrupt Priority registers (IP0 and IP1)

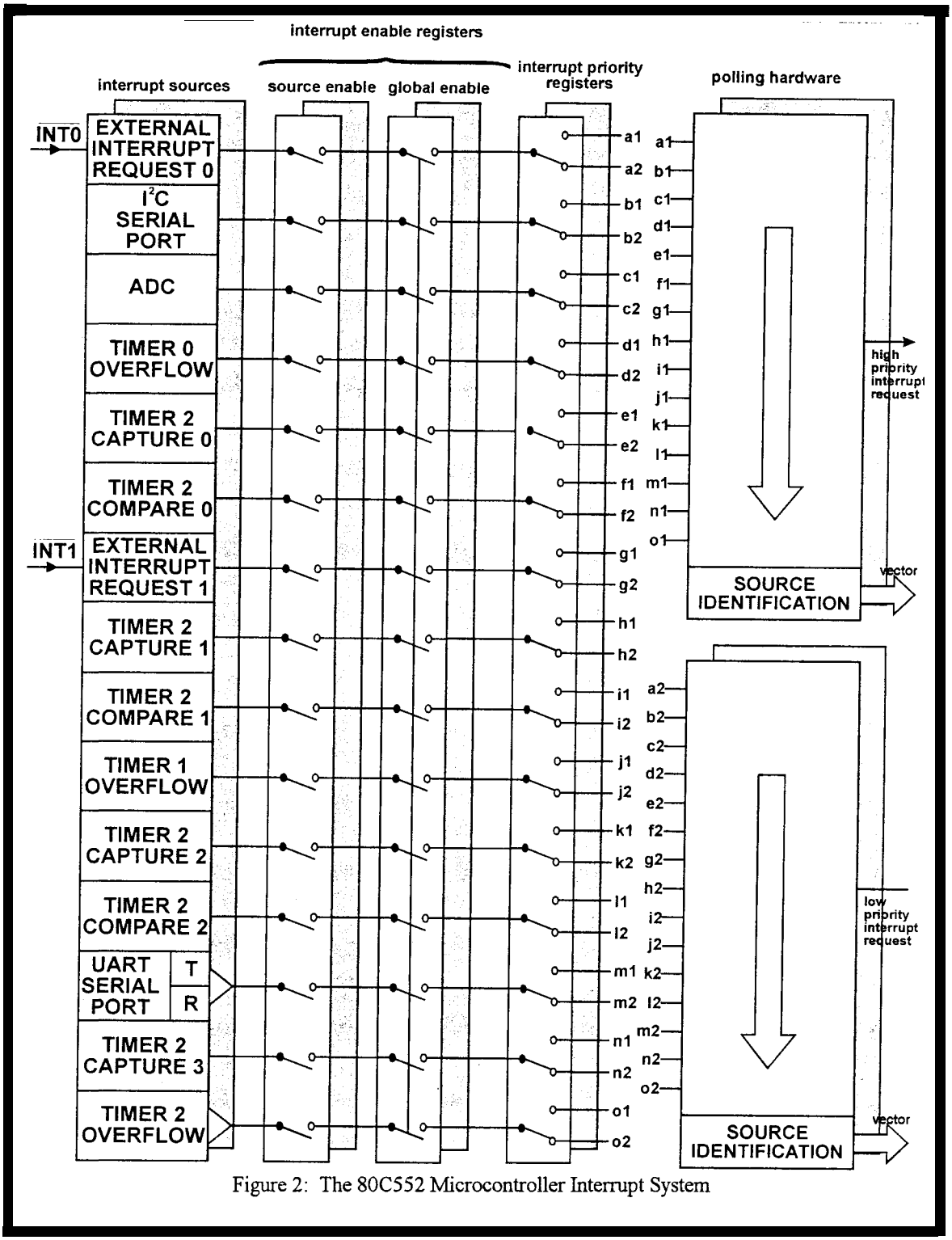

Figure 2: The 80C552 Microcontroller Interrupt System

Therefore utilizing the fill capability of the interrupt subsystem the student would be able to visualize and better understand how interrupts are enabled, how their priority is set, how external interrupts are configured, and the use and function of all applicable SFRS.

Similar results can be obtained by selecting any item from the original block diagram, then selecting the items for which more information is desired.

Timing

The timing selection may be used in a stand-alone mode or in conjunction with the architecture option. In the stand-alone mode the student may select one of the following:

- External code memory read
- External data memory read
- External data memory write

When one of the three is selected, a timing diagram of the operation will be displayed. The student may then

select to receive additional information on the control signals involved, or to view an animation of the selected operation, or both.

When used in conjunction with the architecture option, the student may observe the timing requirements of any of the applicable microcontroller subsystems. This is accomplished by "clicking" the timing button on the button bar with an architecture block selected, and the corresponding timing related information will be presented in the form of text, diagrams, and/or animations.

Instruction Set

This portion of the learning aid provides the student with an explanation of the operation of any instruction in the instruction set along with a block diagram animation illustrating the execution of that instruction. The student is initially provided with a list of all instructions in the instruction set, with the option to:

- group the instructions by function
- alphabetize the instructions
- group the instruction by addressing mode
- select an individual instruction

When an instruction is selected, general information such as, number of bytes in the instruction, number of bus cycles required to execute the instruction, instruction syntax, addressing modes available, and a description of the instruction's operation is provided. Then the student may view an animation illustrating the execution of the instruction utilizing the specified addressing mode.

For example, if *MOVX A, @DPTR* were selected the following information would be displayed:

Syntax: MOVX A, @DPTR
Addressing modes: Register, Indirect
Number of bytes: 1
Number of cycles: 2

Operation: (A) ← ((DPTR))
Transfers a data byte from the external data memory location specified by the Data Pointer to the Accumulator. [1]

At this point an animation of the instruction operation could be selected.

Had the student simply requested *MOVX* then the available addressing modes and instruction formats would have been displayed. The student could then select the desired operation mode to receive specific information, like that shown above.

Program Development and Execution

The learning tool also gives the students the ability to edit and assemble programs which they have written. The system allows the students to select their own editor, assembler and monitor programs.

The default editor is the same as the windows default and the assembler and monitor must be provided by the user. For example, at Purdue we use the Franklin Assembler and a monitor program written by R.H. Barnett, an EET faculty member.

The monitor allows the student to download programs, manipulate memory, SFRS, I/O peripherals, and to execute programs. In addition it allows setting breakpoints, running programs from the breakpoints, and singly executing program loops. [2]

Each of the students in EET 205 receives a free copy of each of the programs, for his/her use during the course. It is then a very straight-forward procedure for setting the path to your specific editor, assembler and monitor programs.

Once this is accomplished, the student can create, assemble, download and execute programs and obtain on-line information without ever leaving the development environment. This proves to add a significant element of convenience to the student's program development cycle.

Program Simulation

A detailed description of the simulator is beyond the scope of this paper, however, a brief explanation of its capability is provided. The simulator provides all the basic functions of any simulation program. This program simulates the ASCII source file rather than the object file, therefore the user is not required to assemble the program during each iteration of the development cycle. The simulator supports both internal and external memory, all built-in features of the microcontroller (A/D converter, ports, etc.), and various execution modes (single step, breakpoints, etc.). This provides the students with a valid method of doing preliminary testing, of their programs, prior to arriving in lab.

COURSE INTEGRATION

The learning aid is being integrated using select volunteer students from the introductory microprocessor course at Purdue's South Bend campus. The integration is being accomplished in phases as more of the project is implemented. The students are asked to provide a substantial amount of feedback on numerous aspects of the tool, ranging from user friendliness to technical completeness. Subjective information such as, does the student feel the learning aid facilitated the learning of microcontroller architecture and assembly language programming, is also solicited.

After substantial testing, the learning aid will be provided to the other Purdue campuses to receive additional feedback, and then if all goes well the tool will be made available to all students at all universities.

CONCLUSION

The PC based interactive learning aid/microcontroller software development system should prove to be a very useful tool to students enrolled in an introductory microprocessor courses. Through the use of text, graphics and animation, the tool should be able to break down some of the barriers, inherent to traditional instruction, and allow the student the opportunity to learn conceptually difficult material more easily.

REFERENCES

[1] *Signetics MicrocontroIler Users' Guide,* Signetics Company (a division of Philips Corporation), 1989,

[2] R. H. Barnett, *Program Development Software for an Educational Environment,* Illinois/Indiana ASEE Conference Proceedings, 1996.

Jeffrey W. Honchell is an Assistant Professor of Electrical Engineering Technology at Purdue University at South Bend. He has a B.S.E.T. from Purdue University and a M.S.C.S. from the State University of New York at Binghamton. Prior to starting his teaching career in 1993, Mr. Honchell served as an Engineer for IBM and The Johns Hopkins University Applied Physics Laboratory. He is a member of the IEEE and ASEE. He is also the Secretary for the IL/IN Section of the ASEE.