Knowledge Based Automatic Testing Of Microprocessor Based Systems (KBAT)

K.C. Yarlagadda, P.K. Lala
Department of Electrical Engineering
North Carolina A&T State University
Greensboro, North Carolina

Keywords: Microprocessor based systems, Artificial intelligence, PC based Applications.

Abstract
With the increasing use of microprocessors in both commercial and military applications, testing of microprocessor based systems has become a critical issue. Due to the increasing need and wide area of applications, many different microprocessors and peripherals are coming into market. The systems based on these components (Microprocessor based systems) have different organization and configuration, though the basic concept of CPU, memory and peripherals remains the same. Testing of such systems need adequate knowledge about the components used in the system. However, it is difficult to keep track of all newly introduced microprocessors and peripherals in detail. This paper explains a knowledge based system based on an IBM PC compatibles, designed to assist engineers without detailed knowledge about microprocessors and associated peripherals, to debug microprocessor based systems.

1. Introduction
The basic faults or problems in the microprocessor based systems can be divided into two types from the testing point of view:
Kernel is the hardware that consists of the microprocessor and the decoding logic. If that part is working satisfactorily, it is possible to use the same microprocessor to run test routines to debug the problem further. Such faults are called non-kernel faults. If the kernel is not working properly then it is necessary to generate patterns externally to debug the kernel
associated problems. Such faults are called kernel faults. To debug the non-kernel faults a ROM emulator is designed which makes use of the system ROM (may be monitor) space to run the test routines using the target CPU itself. To debug the kernel faults a KDM (Kernel Debugging Module) is designed. These two hardware modules, ROM emulator and Kernel Debugging Module (KDM), are designed to be employed as add-on cards in the IBM PC compatibles.

Initially, the Expert Systems learns about the configuration of the system by posing different standard set of questions to the user. The user is asked to prompt the symptoms of the problem like display not working, keyboard is not responding etc. The Expert system decides which module to be used for troubleshooting the problem and prompts the user to connect that module. Then the system runs the particular test for the testing the fault and takes the feedback of the test from the user or directly from the target system. After taking in the feedback about the result of the test, the system infers the probable cause of the problem. Depending upon user's advice it decides further course of action.

2. ROM Emulator
The ROM emulator is used to run the test programs to check various peripherals of the system. It supports different types ROM capacities 2K, 4K, upto 64 K. After knowing the target system's configuration, KBSAT suggests appropriate ROM emulator pod to be connected to the target system. The ROM emulator basically consists of a dual port RAM, accessible to both PC and the target system's CPU. The system loads the appropriate test routines and passes control to the target CPU to run those programs. The ROM emulator can be used only when the target system's hardware kernel i.e., the CPU and the address decoding section, is working properly.
3. Kernel Debugging Module (KDM)
The Kernel Debugging Module is used to debug the kernel problems like shorts in the buses as well as faults in the decoding circuitry latches etc. The KDM is different for different CPU packages and their pin counts. It is implemented using the fast static RAMS. This module does not fully emulate the CPU of the target system. Patterns for basic bus cycles like memory read, memory write etc., are emulated by outputting the patterns of 1's and 0's from the memory. These patterns are transferred by the Expert System to the KDM depending upon the problem. Feedback from the user is taken for the test patterns and a decision is made about the probable fault in the kernel part of the target system. The patterns need not exactly follow the same timing as that of the CPU (more specifically the speed of the clock) as we are more concerned about the shorts in the PCB track as far as the kernel problems are concerned.

4. Software
The knowledge base is designed using a LEONARDO expert shell and PROLOG. Knowledge is represented using both rules and frames. Programs controlling the hardware of the system have been designed using C. Test routines for various peripherals have been developed for various microprocessors using the respective assembly languages. LEONARDO is an expert system shell supporting both rule and frame based knowledge base. The characteristics of various elements such as different pins of the chips could be stored using frame representation. The rules for taking various decisions such as

   IF the kernel is good
   THEN use ROM emulator.

are represented using the rule database. This shell also has a good user graphical interface for asking questions and suggesting solutions. Another important feature is the flexibility to import the programs of other
languages. The basic test routines are written in the assembly languages of the corresponding CPUs. These programs are the skeleton test routines without the address reference components. This is because of the fact that these components will depend upon the address decoding logic of the target, and hence varies for each system. Programs are written to fill these address reference components of the skeleton programs before loading the test routines into the memory of the target system. These programs are written in C language.

5. Conclusion
Test modules for the popular 8 bit and 16 bit CPUs have been developed. As the knowledge base can be increased without disturbing the original design, support of other CPUs could be added. Hardware can be designed to reduce the operator feedback. Knowledge about the testing instruments could be added to the database and rules could be added to tailor the system for a particular setup also. Programs could be developed for converting the test programs for the same peripherals for target systems with different CPUs to reduce the development time.

6. References