program that runs on a host computer system (not shown) and that emulates a design for a hardware circuit **102**. For example, the design can be for an integrated circuit, such as a microprocessor, a controller, a logic device or a finite state machine. The virtual environment **200** is a software program that also runs on the host computer system and that emulates a design for a hardware environment **202** for the hardware circuit **102**. For example, if the design is for an integrated circuit microprocessor, the virtual environment **200** will generally emulate memory devices, their associated memory maps, input/output devices, input/output port addresses, various registers and their associated register definitions, but could emulate a wide variety of devices and parameters associated with the anticipated environment external to the circuit under design. Together, the software model **100** and the virtual environment **200** form a virtual system **300**.

A step in the design of the hardware circuit **102** is to generate the software model **100** for the design and the virtual environment **200** by appropriately programming the host computer system. Test code **400** is developed which provides stimulus for the software model **100** to execute instructions within the virtual environment **200** and to provide an indication of whether the operations were performed as expected. For example, a test can include the software model **100** loading a sequence of bits into a virtual memory device located in the virtual environment **200** according to a particular algorithm, retrieving the bits, and then examining the bits to determine if the values were loaded and retrieved correctly. Thus, the virtual environment **200** is necessary to test functions of the software model **100** that involve interaction of the design for the hardware circuit with its environment.

Preferably, the test code **400** includes a wide variety of diagnostic tests that can be selectively performed so that the design for the hardware circuit can be thoroughly analyzed and verified. The design can be modified by making corresponding changes to the software model **100** if the results are not satisfactory.

Once the software model **100** is completely debugged, the software model **100** is transformed into an equivalent hardware circuit **102** by any known means. The equivalent hardware circuit **102** can be constructed using programmable logic, interconnected logic modules, discrete components or can be a completely integrated hardware implementation of the design. It is important, however, that the hardware circuit **102** is equivalent to the software model **100** in that the hardware circuit **102** provides functionality that is equivalent to the functionality of the software model **100**. In addition, the virtual environment **200** is transformed into an equivalent hardware environment **202** for the hardware circuit **102**. The hardware environment **202** can include various types of memory devices, registers, input/output ports and devices, co-processors, power supplies, and the like. It is important that the hardware environment provides functionality that is equivalent to the functionality of the virtual environment **200**. Together, the hardware circuit **102** and the hardware environment **202** form a hardware system **302**.

It is important that a logical distinction can be drawn between the software model **100** and the virtual environment **200** so that when they are transformed into the equivalent hardware circuit **102** and into the equivalent hardware environment **202**, respectively, a physical distinction can be drawn between the hardware circuit **102** and its hardware environment **202**. Maintaining such logical and physical distinctions will facilitate the process of isolating and correcting any faults located during the process of analyzing and verifying the design. In addition, maintaining such a physical distinction will enable the design for the hardware circuit **102** to be separated from the design for the hardware environment **202** for manufacturing the hardware circuit **102**.

FIG. **2** contrasts the test code **400** utilized for verifying the software model **100** with the test code **402** utilized for verifying the hardware circuit **102**. Because the hardware circuit **102** and hardware environment **202** are functionally equivalent to the software model **100** and the virtual environment **200**, respectively, the test code **400** used to verify and analyze the software model **100** can be utilized to verify the hardware circuit **102** with few changes. As can be seen from FIG. **2**, the test code **400** differs from the test code **402** in that the set up program "A" which includes syntax or specific commands required to initialize the host computer system and to instruct the host computer system to execute appropriate test suites will likely differ from the set up program "B" which includes syntax or specific commands required to initialize the hardware circuit **102** and the hardware environment **202** to execute the identical test suites. This is because any software overhead ("glue code") associated with the host computer system will no longer be required, but different overhead ("glue code") associated with the hardware circuit **102** and hardware environment **202** will be needed. Therefore, the test code **400** has a different set up file than the test code **402**.

The diagnostic test suites contained within the test code **402** which provide stimulus to the hardware circuit **102** and which examine results obtained, however, can be essentially identical to the diagnostic test suites contained within the test code **400**. This is advantageous because the diagnostic test suites provide a consistent and uniform means for verifying both the software model **100** and the hardware circuit **102**. In other words, diagnostic test suites provide a uniform "yardstick" by which the performance of the software model **100** and the hardware circuit **102** can both be measured.

Preferably, the diagnostic test suites of the test code **400** and **402** can be run in a batch mode or in an interactive mode. In batch mode, a predefined set of diagnostic tests are performed and the results obtained. In interactive mode, a test can be halted at selected points in its operation and register activity, internal signal activity or external signal activity, can be observed and analyzed. Batch mode is generally used to ensure that the design performs properly after a change is made, while interactive mode is generally used to isolate faults. In addition, a monitor mode is preferably provided which allows specific addresses to be monitored, such as program execution addresses, to determine whether a test was failed, passed, or resulted in a hang state. Once a test has been completed, the results are obtained, compared to expected results and a pass/fail determination is made. The results will then be logged to a display and stored in a designated memory file.

In addition, specific parameters of the virtual environment **200** and of the hardware environment **202** can preferably be selectively configured before running a diagnostic test suite. The virtual environment **200** can be appropriately initialized through its set up file and the hardware environment **202** can also be configured via a similar setup file or by appropriately setting dip switches. Preferably, the virtual environment **200** and the hardware environment **202** are configured in a like manner to maintain their functional equivalence, but could be configured differently for testing purposes. The selective configuration parameters can include: assigning memory segments for particular operations, controlling a number of