Title: METHOD AND SYSTEM FOR LOADING PROGRAMMABLE MEMORY DEVICES IN AN ELECTRONIC SYSTEM USING VOLATILE MEMORIES

Abstract: A method and system for storing the programming images of programmable memory devices is disclosed. In this method and system, a plurality of target boards each has at least one programmable memory device that uses a programming image for configuration and a board-ID to identify the target board, and a target board microprocessor subsystem. The method and system further includes a master controller board having a plurality of programming images for configuring the target boards, with each programming image corresponding to the board-ID of the target board. In the method and system disclosed herein, the programming images are transferred from the master controller board into the programmable memory devices when the target board power in the target boards is not providing power to the target board microprocessor subsystem.
For two-letter codes and other abbreviations, refer to the "Guidance Notes on Codes and Abbreviations" appearing at the beginning of each regular issue of the PCT Gazette.

Published:
— without international search report and to be republished upon receipt of that report.
METHOD AND SYSTEM FOR LOADING PROGRAMMABLE
MEMORY DEVICES IN AN ELECTRONIC SYSTEM USING
VOLATILE MEMORIES

FIELD OF INVENTION

This invention relates generally to digital electronics. More specifically, this
invention relates to methods of supplying programming image data to programmable
memory devices and/or programmable devices.

BACKGROUND OF THE INVENTION

An electronics system often consists of various components, such as digital
electronic devices, analog electronic devices, printed circuit (PC) boards, connectors,
power supplies, peripheral equipment, mechanical housings, input devices and display
devices. One example of such a system is a personal computer.

The electronics system may be partitioned into various subsystems, each which
may perform a specific sub-function contributing to the overall and complete func-
tionality of the electronics system as a whole. One of these subsystems may be a digital
electronics subsystem. This digital electronics subsystem may also further be portioned
into smaller subsystems. Often each of these smaller subsystems is located on a
separate PC board.

These PC boards are usually mechanically and electrically interconnected to a main
PC board. When the main circuit board itself has electronic subsystems on it, it is often
referred to as a 'motherboard.' The connected PC boards are then referred to as
'daughter-boards.' If the main circuit board contains no additional functionality, but
rather functions only to interconnect the PC boards, both electrically and mechanically,
it is often referred to as a 'backplane.'

Since collectively the PC boards operate cohesively to provide the overall function-
tality of the digital electronics subsystem, it follows that each PC board only
provides a fractional portion of the digital electronic system's functionality. Thus, the
PC boards must possess the capability to inter-operate and intercommunicate with each
other. This interoperation/intercommunication is necessary to provide for the exchange
of information between the PC board subsystems.

Generally, each PC board communicates to another via electrical signals.
Completely digital subsystems usually use only digital signals for communication.
These signals may be specific and unique between any two PC boards, or a common
set of signals may be used among all PC boards. If a common set of signals is used, it
is referred to as a bus architecture or system bus. The signals in the system bus
typically fall into 5 major categories of signal types: (1) Power Signals, that provide
direct current (DC) voltage and current, often referred to as power, and the current
return path and voltage reference or ground to the PC board; (2) Address Signals or the
address bus, that helps to select a particular device on a PC board or a particular
location within a device on a PC board to store \textit{i.e.}, write) and retrieve \textit{i.e.}, read) in-
formation to/from; (3) Data Signals or the data bus, that carries the information being
transferred to/from the location specified by the address bus; (4) Control Signals or the
Control bus, that specifies when and how \textit{i.e.}, read or write) the data is to be
transferred (collectively called a bus request); and (5) Interrupt Signals or interrupts,
that provide alert indications that a special event requiring service has been detected.

A common connector or slot is typically used on all PC boards. Corresponding
mating connectors are typically used on the motherboard/backplane. The bus ar-
chitectures signals are physically organized in the same order and orientation on all the
PC boards' and the motherboard/backplane's connectors. This provides the option of
placing any PC Board in any slot on the motherboard/backplane.

Every PC board and its associated devices residing on the system bus must be
located at a unique and different address, as viewed from the system bus, to prevent
two or more PC Boards from simultaneously responding to a bus request. The unique
system addresses for each PC Board may be: (i) Hardwired at the time of their
manufacture and are thereafter unchangeable; (ii) Hardware configured by the end user
prior to insertion into the motherboard/backplane, typically through switches or
jumpers \textit{i.e.}, small mechanical clips) located on the PC board; or (iii) Software
configured by the motherboard or a PC board designated as a master and the unique
address information loaded into a special \textit{i.e.}, specifically for the purpose of address
identification) volatile or non-volatile memory on each PC board using a special
algorithm, such as Plug and Play.

A PC board may contain a variety of digital electronic devices, some of which
include microprocessors, volatile \textit{i.e.}, cannot retain its stored contents after power is
removed from the device) Random Access Memories (RAMs), non-volatile \textit{i.e.}, can
retain its stored contents after power is removed from the device) Read Only Memories
(ROMs), non-volatile Programmable Read Only Memories (PROMs), Field Pro-
grammable Gate Arrays (FPGAs), logic gates, input buffers, output buffers, tri-state
buffers, Digital Signal Processors (DSPs), Programmable Logic Devices (PLDs), and
other devices.

A memory is a device that stores information or instructions. Such memory devices
are often categorized as volatile or non-volatile. Volatile memory devices can only
retain their stored information until power is removed from them. Typically, in-
formation, or data, is supplied to or written to \textit{i.e.}, loaded into) these devices by an
external source. Examples of such external sources include, but are not limited to, microprocessors, disk drives, and tape drives. These devices can retain the stored information until such time that they are powered off, either intentionally or unintentionally, and lose their stored information. In order to restore the information to these devices, the information must be re-supplied, or re-written, to these devices. In contrast, non-volatile memory devices are able to retain their information after being powered off and powered on again, thus alleviating the necessity of rewriting the information to them.

Programmable Logic Devices (PLDs) are those classes of devices that include both programmable memory devices and logic devices. The logic devices contain blocks of un-configured digital logic. The blocks are often referred to as logic blocks. Theses programmable logic devices require their logic blocks to be configured and organized prior to operation. This is typically done by providing the configuration and organization information to the programmable logic device in the form of programming data. Some devices can store this data within themselves, while others must have this data provided to them externally, typically stored in a non-volatile programmable memory device.

Certain digital hardware devices, such as ROMs, are hardwired with specific programming information at the time of their manufacture and can never be altered. Other devices, such as PROMs, are designed to accept their programming information subsequent to their manufacture. PROMs may only accept a one-time load of their programming information via a special programming apparatus. The PROM is physically placed in the programming apparatus. The programming information is downloaded into the PROM via the programming apparatus. Once programmed, the device can no longer be altered. Once a PROM is placed on a PC board, the only way to alter the information in the PROM is to replace the PROM with a new PROM. To replace a PROM, the PC board must manually be removed from the system, the PROM must manually be removed from the board and discarded, a new PROM must be programmed, the new PROM must be placed on the PC board, and the PC board must be returned to the system to provide for enhanced performance (i.e., feature upgrades, bug fixes, etc.).

Other devices exist that are similar in function and operation to PROMs, but provide for the device to be erased and reprogrammed many times. Non-volatile Erasable Programmable Read Only Memories (EPROMs) are one such device. EPROMs may be erased by removing them from their circuit and exposing them to strong ultraviolet light. They are reprogrammed in the same manner as previously described. Non-volatile Electronically Erasable Programmable Read Only Memories (EEPROMs) are another such device. These devices may typically be erased by
applying a special voltage to the device. They may be reprogrammed as previously
described. They may also be reprogrammed without removing them from the PC
board. In this case, the PC boards must make special connectors available to a
technician for the purpose of providing a connection to the programming apparatus.

Many other families of programmable memories and Programmable Logic Devices
(PLDs) also permit electronic reprogramming similar to an EEPROM. These include
Programmable Logic Arrays (PLAs), Programmable Array Logic (PALs), Field Pro-
grammable Gate Arrays (FPGAs), Non-Volatile Random Access Memories
(NVRAMS) and other devices. Once the memory devices or programmable logic
devices are programmed, they can retain their configuration without dependency on
external power; i.e., they have non-volatile memory properties.

Prior attempts to update and reprogram programmable logic devices have been
made. For example, U.S. Patent No. 6,622,246 (Biondi) describes a method to erase
and reprogram an EEPROM from a remote location. Similarly, U.S. Patent No.
6,459,297 (Smiley) also describes a method to program an EEPROM remotely over a
system bus. Both Biondi and Smiley, however, indicate that a PC board (i.e., a slave
PC board) with a resident EEPROM may only be reprogrammed by providing a special
reprogramming circuit locally on the slave PC board. Another PC board (i.e., a system
controller or master PC board) may then provide the programming information to the
local reprogramming circuit and command it over the system bus to reprogram the
EEPROM with the new information.

al.) describes a method to load software programming information containing in-
structions for Central Processing Units (CPUs) located on remote PC boards for boot
operation into RAM while the remote CPU is held in reset. Like Smiley and Biondi,
however, King also requires the remote PC board to have a special programming
circuit residing on it.

Unfortunately, the costs of the local special programming circuits may amount to a
significant portion of the PC board's cost. This increase in cost makes it impractical to
have replicas of such a circuit on every PC board in the system.

In a system, there may be many such programmable memories and programmable
logic devices located on many different PC boards, each requiring new programming
information. Since it is desirable to permit any PC board to reside in any slot, a method
of identifying each PC board, its associated programmable memories, its pro-
grammable logic devices and its current slot is required. For example, U.S. Patent No.
6,049,870 (Greaves) describes such a method as it applies to parametric software con-
figuration information (i.e., quantity of memory required, processor speed, etc.).
Greaves' system includes a 'configuration bus' in addition to the system bus. This con-
figuration bus need not be as complex as the system bus, but needs only to provide bidirectional communication between the slave PC boards and a master PC board. A serial bus is an example of one such simplified bidirectional bus. In Greaves' method, each slave PC board is configured with a unique non-volatile identification (ID). The master PC board may read this ID over the configuration bus. The master may then use the ID as an index into a table accessible to the master to correlate the PC board ID with a list of associated configuration information. According to Greaves, the master may then transfer this configuration to the slave PC board via the configuration bus.

The addition of a separate configuration bus, however, can only be provided for via custom, non-standard, system bus architectures. The development costs of such a bus can be extremely expensive, as custom components, usually manufactured by a single supplier, are often used. The manufacturing costs of systems using these custom architectures may also be significantly higher.

Certain types of non-volatile programmable memories and programmable logic devices may also provide a way to reprogram or update their programming information without removing the PC board from the system. Any failures during this reprogramming step \(i.e.,\) power failure, address bus errors, etc.), however, can cause the devices to receive an incorrect or corrupt program and render the devices and/or PC boards inoperable. Once inoperable, the malfunctioning devices and their PC boards must be physically removed and manually repaired \(i.e.,\) troubleshoot the PC boards and remove and repair the malfunctioning devices). This may cause a considerable delay in operation \(i.e.,\) system downtime), expense in material \(i.e.,\) new parts required), expense in labor charges \(i.e.,\) cost of repair), and lost revenue \(i.e.,\) diminished productivity).

Non-volatile reprogrammable devices are considerably more expensive than volatile reprogrammable devices. For large systems, with many copies of the same printed circuit boards, and thus many copies of the same programmable devices, it is costly to employ a large number of non-volatile reprogrammable logic devices in the system. These systems usually revert to using less expensive PROMS. Unfortunately, the savings in component costs by using these one-time programmable devices may be outweighed by repair/upgrade costs when a device needs to be updated.

Programmable logic devices are often found on PC boards along with other devices. Examples of such devices include, but are not limited to, microprocessors, digital signal processors, RAM and ROM. Because it is necessary for all of these devices to be ready for operation at the time of power-on, a way to configure the programmable logic devices quickly, at least as quickly as all other devices on the PC board, is needed. Typically, non-volatile memory storage devices are used. Although these devices accomplish the desired goal, they also have undesirable qualities, such as
the need for manual removal to receive upgraded memory image information, frequent corruption of data during in-circuit upgrade attempts, and other problems.

[23] The volume of parts requiring feature upgrades and bug fixes in large systems can be costly, time consuming, and tedious. Additionally, the upgrade process may temporarily render the equipment inoperable until all devices have been upgraded, resulting in significant loss of productivity due to system downtime.

[24] Therefore, a significant need exists to find a method to more easily, quickly, efficiently, and cost-effectively update programmable memory devices in various electronics systems.

SUMMARY OF THE INVENTION

[25] The present application describes a system for using programmable memory devices to store programming images. The system comprises a plurality of target boards, with each target board having at least one programmable memory device that uses a programming image for configuration, a board-ID to identify the target board, target board power for supplying power to the target board, an interboard bus control interface for controlling the target board power, and a target board microprocessor subsystem for exchanging data with the programmable memory device. The system further comprises a master controller board having a plurality of programming images for configuring the target boards, with each programming image corresponding to the board-ID of a target board. The system also comprises an interboard bus for transferring signals between the target boards and the master controller board, and a master power bus to provide power to the interboard bus control interface independent of the target board power. In the system of the present application, the programming images are transferred from the master controller board into the programmable memory devices when the target board power is not being supplied to the target board microprocessor subsystem.

[26] The present application also describes a method for using programmable memory devices to store programming images in a system having a master controller board and one or more target boards, each of the target boards including target board power, a programmable memory device, and a microprocessor subsystem. The method comprises supplying target board power to each of the target boards, obtaining a board-ID for each of the target boards, powering a master power bus via the master controller board, powering the interboard bus control interface on the target boards via the master power bus, and controlling the amount of power provided to the devices on the target board by the target board power via the interboard bus control interface. The method further comprises detecting a target board via the master controller board, communicating with the target board via the master controller board to determine data to be imaged based on the board-ID, correlating the board-IDs in the target boards to
programming images in the master controller board, configuring and downloading programming images to the programmable memory device via the master controller board, and acknowledging that images have been received via the target board. Finally, when all the images have been received, the target board power is restored to all devices on the target board via the master controller board.

**BRIEF DESCRIPTION OF THE DRAWINGS**

Preferred embodiments of the present inventions are described with reference to the following drawings, wherein:

- FIG. 1 is a block diagram depicting an exemplary system that can supply programming information to programmable memory devices;
- FIG. 2 is a block diagram of the target board shown in FIG. 1;
- FIG. 3 is a block diagram of the master controller board shown in FIG. 1; and
- FIGS. 4a-4c are flowcharts describing the method for supplying the programmable memory device images to the programmable memory device of FIG. 1.

**DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS**

FIG. 1 illustrates one embodiment of a system 10 for supplying programming information to programmable memory devices using a hardware controlled function. System 10 comprises a master controller board 6 and a group of target boards 12, which may include one or more target boards (such as target board 2 and target board 4), interconnected via an interboard bus 8. The master controller board 6 serves to identify a target board as a particular board-type (explained below) and to provide the corresponding programming image data to all programmable memory devices resident on the target boards 12. Each target board may use the programming image data to configure its programmable memory devices, which in turn determine the operation of each target board. In other words, the target boards 12 preferably operate in accordance with the programming image data loaded into their programmable memory devices.

FIG. 1 shows two target boards 2, 4. However, a system may actually contain as few as one or as many as an unlimited number of target boards. A system power bus 1 provides power to the master controller board 6, as well as the target boards 2 and 4. The interboard bus 8 serves to allow the transfer of data between all boards in the system 10.

The target boards 12 and the master controller board 6 are physically interconnected by a common set of system bus wires. These bus wires carry the interboard bus 8 signals. In this example, the bus wires are found on the system backplane. Other systems may provide these signals in other ways, such as, but not limited to, motherboards, midplanes, cables, wireless signals, radio frequencies, etc.

The interboard bus 8 described in this example is an 8-bit wide bi-directional
parallel bus, in which all data and address signals are provided concurrently. Other
buses, such as, but not limited to, a serial bus or a 16-bit wide bi-directional parallel
bus, may be employed as well. The interboard bus 8 directs its associated signals
between all the boards, such as the master controller board 6, and the target boards 12
in the system 10. The interboard bus 8 signals include: (i) the system control bus
signals, that are used to control when and in which direction the data is transferred; (ii)
the system address bus signals, that are used to control where the data being transferred
is being transferred to/from; and (iii) the system data bus signals, that contain the data
involved in the data transfer. Although only these signals are described in this
example, other signals may or may not be present.

Additionally, FIG. 1 shows a master power bus 14. The master power bus 14 is
controlled and driven by the master controller board 6 and received by an interboard
bus control interface 20 located on the target boards 12, which is shown in FIG. 2. The
master power bus 14 provides power from the master controller board 6 to the target
boards 12 while the target board power 18 of the target board 12 is switched off.
Therefore, the master controller board 6 can configure a device, such as a pro-
grammable memory device 26, on the target board 12 even though the target board
power 18 is switched off. When the target board power 18 is restored to all of the
devices on the target board 12, the internal hardware logic of the programmable
memory device 26 that was configured by the master controller board 6 will be
available for immediate use. Preferably, the master power bus 14 is a physical wire
present on the backplane that interconnects all the boards in the system, but as with the
system bus signals, it can be provided for by other functionally similar mechanisms.
Examples of such other functionally similar mechanisms are, but are not limited to,
wires on the motherboards, solder traces on the midplanes, cables, wireless signals,
radio frequencies, etc.

FIG. 2 shows an exemplary architecture of target board 2 in block-diagram form. It
should be understood that the architecture of target board 4 is preferably, but not
necessarily, identical to the architecture of target board 2, except that it may or may not
have a different board-ID value (explained below), and it may or may not have a
different primary function within system 10 than target board 2. Thus, for ease of
reference, only target board 2 will be discussed in detail herein. It should be
understood that the discussion of target board 2 applies equally to target board 4,
except for the possible exceptions previously noted.

The target board 2 generally comprises target board power 18, an interboard bus
control interface 20, a board-ID register 21 located on the interboard bus control
interface 20, a microprocessor subsystem 24, and a programmable memory device 26.
The programmable memory device 26 may be either a volatile or non-volatile memory
device. Examples of volatile memory devices include, but are not limited to, Static RAM (SRAM) or Dynamic RAM (DRAM) devices. Examples of non-volatile memory devices include, but are not limited to, hardwired values using jumpers, hardwired values using switches, hardwired values using solder traces, EPROMS, NVRAMs, and others devices.

In addition, the target board 2 preferably includes a first local target board bus 16 that connects the interboard bus control interface 20 to the programmable memory device 26. The first local target board bus 16 may also be connected to the microprocessor subsystem 24, but it is not necessary, as indicated by the dotted line in FIG. 2. The target board 2 also preferably comprises a target board power bus 23 that connects the target board power 18 to the microprocessor subsystem 24 and to the programmable memory device 26. Target board 2 preferably further includes a local data/control bus 25 for the exchange of data between the microprocessor subsystem 24 and the programmable memory device 26 during normal operation mode (i.e., when target board power provides power to all devices on the target board).

As shown in FIG. 2, power is supplied to the target board power 18 via the system power bus 1. Target board power 18 in turn supplies power to the components on each respective target board 12. More specifically, the target board power 18 supplies power to the microprocessor subsystem 24 and the programmable memory device 26, and any other devices that may be present on the target board 2. During program mode, the master controller board 6 may send a signal to disable the target board power 18 via the interboard bus 8. As a result, the microprocessor subsystem 24 and programmable memory device 26 will be disabled as well.

As previously mentioned, target board 2 also has an interboard bus control interface 20 including a board-ID register 21. The interboard bus control interface 20 is controlled by the master controller board 6 via the interboard bus 8, and is powered by the master power bus 14. The interboard bus control interface 20 contains a separate circuit from the remainder of the target board components, and is essentially an extension of the master controller board 6. The interboard bus control interface is connected to the target board power 18 via target board power control 22, and selectively powers the target board power 18 by turning on only small portions of the target board power at a time. In other words, the target board power 18 is not switched completely on, but just supplies enough power to program certain devices, such as programmable memory device 26.

The board-ID register 21, which is located on the interboard bus control interface 20, stores a value representing an identifier (i.e., the board-type) for the target board 2 and identifies the target board 2 as having a specific combination of functions and hardware configurations. Collectively, this specific combination of functions and
hardware configurations is referred to as the board-type of a target board, such as target boards 2 and 4. The board-type value contained in the board-ID register 21 permits the master controller board 6 to identify each target board 12 in the system 10. In this example, the board-ID register 21 is implemented as a non-volatile memory mapped register located within the address space of the interboard bus 8.

During normal operation mode, the programmable memory device 26 may accept data from the master controller board 6 across interboard bus 8 via the first local target board bus 16. Alternatively, the microprocessor subsystem 24 of target board 2 may accept data from the master controller board 6 across interboard bus 8 via the first local target board bus 16, as indicated by the dotted line in FIG. 2. The microprocessor subsystem 24 then transfers this data across the local data/control bus 25 to the programmable memory device 26 for processing. The processed data may or may not be returned to the microprocessor subsystem 24 via the local data/control bus 25 at some time in the future. The local data/control bus 25 is, in this example, a parallel bus comprised of control, address and data signals similar to the interboard bus 8 in function, but separate and distinct in connectivity. Other examples of busses that could be used for this function include, but are not limited to, an RS-232 serial bus and a set of two parallel unidirectional data busses.

FIG. 3 shows an exemplary architecture of the master controller board 6 in block diagram form. The master controller board 6 preferably comprises an interboard bus control interface 30, including a power supply 32, and a board power 37. The master controller board 6 further comprises a microprocessor subsystem 34, a memory 35, a programmable memory device (PMD) image/configuration storage database 36, and a board-ID register database 39, all interconnected via a first master controller board local bus 44.

Once the system power 1 is turned on, the power supply 32 in the interboard bus control interface 30 powers the master power bus 14 to all target boards 12. The master power bus 14 continuously provides power to the interboard bus control interface 20 of the target board 2.

The microprocessor subsystem 34 of the master controller board 6 controls the mode of operation (i.e., either normal operation mode or program mode) of the target board 2 via the interboard bus control interface 30. The microprocessor subsystem 34 may place the target board 2 in program mode by sending a command via the interboard bus 8 to shut off board power 18 in the target board 2 via a second master controller local bus 42 and interboard bus control interface 30. Likewise, the microprocessor subsystem 34 may place the target board 2 in normal operation mode by sending a command via the interboard bus 8 to restore full target board power 18 in the target board 2. Thus, full target board power 18 is restored to the target board 2.
The microprocessor subsystem 34 of the master controller board 6 also function to
retrieve the board-type value from the board-ID register 21 of the target board 2. The
microprocessor subsystem 34 first locates the address for the board-ID register (Reg.)
database 39 by reading the memory 35 via the first master controller local bus 44. The
microprocessor subsystem 34 then retrieves a board-ID register address from the
board-ID register database 39 (i.e., the board-ID register database 39 is a list of board-
ID register addresses, terminated by an end-of-database flag), that contains board-ID
register addresses for the target boards 12 in the system 10. In this example, the board-
ID register database 39 includes the address of the board-ID register 21 of each of the
target boards 2, 3, and 4, and is terminated with an end-of-database flag to indicate the
end of the database.

The microprocessor subsystem 34 selects a board-ID register address, such as that
of the board-ID register 21 of the target board 2, from the board-ID register database
39. The microprocessor subsystem 34 retrieves a board-type value from this board-ID
register 21 across the interboard bus 8 via the second master controller board local bus
42. The microprocessor subsystem 34 of the master controller board 6 may use this
board-type value to query the programmable memory device image storage database
36 and determine if any programmable memory devices exist on the target board 2.

The programmable memory device image storage database 36 contains the
parameters and programming image/configuration data, indexed by board-type, used
by the microprocessor subsystem 34 to program the programmable memory device 26
of the target board 2. The programmable memory device image storage database 36
functions as a data lookup table in this example. The microprocessor subsystem 34 of
the master controller 6 uses the board-type value to index into the programmable
memory device image storage database 36 and retrieve additional data related to a
particular board-type across the first master controller local bus 44.

In this example, the preferred related data parameters include the number of pro-
grammable memory devices that may be resident on a target board, the location of
these programmable memory devices within the system address space, the type of
device a particular programmable memory device may be, and the programmable
memory device programming image data needed to program the particular pro-
grammable memory device being referenced. Other preferred types of data may or may
not be stored in the programmable memory device image storage database 36, as well
as the possible exclusion of some of the listed types of data from the programmable
memory device image storage database 36. In this example, the programmable
memory device image storage database 36 is implemented as a PROM. Other
examples of capable devices include, but are not limited to, EEPROMS, EPROMS,
flash, magnetic disk systems, optical disk systems, magnetic tape systems, network
servers, local computer systems, modems, and wireless data servers.

It should be understood that, although system 10 is presented with two target boards 2 and 4, each with a single programmable memory device 26, a system may be extended to include as few as one and as many as an unlimited number of target boards 12, with as few as one and as many as an unlimited number of programmable memory devices 26. Likewise, a system may be extended to include as few as one and as many as an unlimited number of master controller boards 6.

The operation of system 10 and the methods for programming the programmable memory devices 26 will now be presented, with reference to the method 100 shown in FIGS. 4a-4c. For simplification of the descriptions, all references to a subsystem or board shall be understood to include the pertinent lower-level detailed functionality of that subsystem or board, as previously described. For example, when referring to the master controller board 6 writing to programmable memory device 26 of the target board 2, it should be understood from the previous descriptions of the master controller board 6, the target board 2, and the system 10, that the actual details involve the microprocessor subsystem 34 of the master controller board 6 transferring data across its second local master controller board bus 42 to the interboard bus 8. This data then travels onto the first local target board bus 16 of target board 2. The data is finally stored in the programmable memory device 26 of target board 2.

To begin the exemplary method 100 of operation, system power 1 is applied to all boards in system 10, including the master controller board 6 and the target boards 12, as shown in Step 110 of FIG. 4a. Also shown in Step 110, the master controller board power 37 is turned on, and the target board power 18 is turned off. Again, these boards are shown as an example for the system 10. A system supporting this method 100 is not limited to only these boards, as other boards could be present in other systems.

The master controller board 6 then detects the powered off target board presence, as shown in Step 120. The powered off target board presence is detected via the interboard bus 8, as shown in Step 122. If the target board detected by the master controller board does not have its target board power 18 off, then the master controller board advances to detect the status of the next target board, as shown in Step 124. If, however, the target board detected by the master controller board has its target board power 18 off, then the master controller board reads target board's board-ID register 21 to retrieve a board-ID value, as shown in Step 130. The master controller board 6 reads the target board board-ID register 21 via interboard bus 8.

Using the board-ID register 21, the master controller board 6 may then read and write the programming image and configuration data from the programmable memory device image storage database 36, as shown in Steps 140-160 in FIG. 4b. In Step 140, the master controller board reads the image or configuration from the database 36,
based on the target board ID. The master controller board 6 then writes the image or configuration into the target board 2 via the interboard bus 8, as shown in Step 150. As shown in Step 160, the master controller board then advances to the next target board image or configuration for the target board ID. When all of the image and configuration data has been downloaded to the programmable memory device 26 on the target board 2, processing continues for the next target board, as shown in Step 170. Once all of the target boards 12 have been processed, the method continues on to block 3.

Finally, the process of returning the target board 2 to normal operation mode is started (e.g., when the target board power is completely on), as shown in FIG. 4c. As shown in Step 180, the master controller board 6 sends a signal to the target board power 18 to turn on completely via the interboard bus 8. The master controller board 6 then waits for a power up acknowledgment from the target board 2 via the interboard bus 8 to return to normal operation mode in Step 190. Normal operation mode on target board 2 begins with the execution of these instructions. The programmable memory device 26 of target board 2 is then ready for its stored programming image and configuration data to be accessed by the microprocessor subsystem 24. The method now returns to block 1, where the master controller board 6 advances to the next target board, (i.e., target board 4), to perform the same functions described in FIGS. 4a-4c.

It should be recognized that the normal operating mode functions of the target boards 12 are not considered relevant to the implementation or understanding of this method and system. The target boards 12 could possess functions, such as, but not limited to, modems, Ethernet controllers, video processors, etc. Similarly, it should be understood that any combination or plurality of such functions on a single target board or master controller board are also possible implementations of this system and method.

The various embodiments described above offer several advantages over the prior art. For example, instead of storing the programming images on non-volatile memories of all target boards in a distributed manner, the programming data images are centrally located on the master controller board. Thus, when system target board upgrades or repairs are necessitated, only the programmable memory device image data storage database requires updating. The programmable memories of the target boards may be easily, systematically, quickly, and comprehensively upgraded without the need to manually remove any of the target boards, as is required by the prior art. Thus, the system and method of the present application may gain a considerable advantage, by substantially reducing system downtime due to its non-manual programming data image upgrade process, over manual upgrade methods found in systems employing the
The system and method of the present application also provide an extremely significant maintenance cost savings over systems employing the prior art. It reduces the need for expensive on-site technician visits for the individual and manual upgrade of potentially hundreds or thousands of non-volatile memory devices, as is necessitated in systems employing the prior art. It further reduces the need for costly on-site technician visits to repair any corrupt boards, due to mis-programming, as is also necessitated in systems employing the prior art.

Preferred embodiments have been described herein. It is to be understood, however, that changes and modifications can be made without departing from the true scope and spirit of the system. Other embodiments of the present invention, of course, will be apparent to those of ordinary skill in the art upon their review of the detailed description. No one embodiment should be deemed to be controlling, as all embodiments of the present invention are deemed to be covered by the appended claims. Certainly, the invention must be interpreted to encompass technological improvements not yet developed and/or available to the public. The following claims and their equivalents, which are to be interpreted in light of the foregoing specification, define the true scope and spirit of the invention.
Claims

[1] A system for storing programming images on programmable memory devices comprising:
one or more target boards, each target board having:
(i) at least one programmable memory device that stores a programming image for configuration;
(ii) a board-ID to identify the target board;
(iii) target board power for supplying power to the target board;
(iv) an interboard bus control interface for controlling the target board power; and
(v) a target board microprocessor subsystem for exchanging data with the programmable memory device;
a master controller board having a plurality of programming images for configuring the target boards, each programming image corresponding to the board-ID of a target board;
an interboard bus for transferring signals between each of the target boards and the master controller board; and
a master power bus to provide power to the interboard bus control interface independent of the target board power;
wherein the programming images are transferred from the master controller board into the programmable memory device when the target board power is not being supplied to the target board microprocessor subsystem.

[2] The system of claim 1 wherein the programming images are transferred from the master controller board into the programmable memory device when the target board power is being supplied to the programmable memory device.

[3] The system of claim 1 wherein the signals transferred through the interboard bus include control signals, address signals, and data signals.

[4] The system of claim 1 wherein the master controller board provides power to the master power bus.

[5] The system of claim 1 wherein the board-ID comprises a value readable by the master controller board.

[6] The system of claim 5 wherein the master controller board further comprises a board-ID database for storing the board-ID value of the target board.

[7] The system of claim 6 wherein the master controller board further comprises a master controller board microprocessor subsystem for retrieving a value from the board-ID database to determine if any programmable memory devices exist on the target board.
The system of claim 1 wherein the master controller board further comprises a programmable memory device image storage database for storing the plurality of programming images for configuring the target boards.

The system of claim 1 wherein the master controller board controls the amount of power provided by the target board power to the target board microprocessor subsystem via the interboard bus control interface and the interboard bus.

A system for storing programming images on programmable memory devices comprising:

one or more target boards, each target board having:

(i) at least one programmable memory device that stores a programming image for configuration;
(ii) board-ID to identify the target board;
(iii) target board power for supplying power to the target board;
(iv) an interboard bus control interface for controlling the target board power; and
(v) a target board microprocessor subsystem for exchanging data with the programmable memory device;

a master controller board having a plurality of programming images for configuring the target boards, each programming image corresponding to the board-ID of a target board, wherein the board-ID comprises a value readable by the master controller board;

an interboard bus for transferring signals between each of the target boards and the master controller board; and

a master power bus to provide power to the interboard bus control interface independent of the target board power;

wherein programming images are transferred from the master controller board into the programmable memory device when the target board power in each of the target boards is being supplied to the programmable memory device and is not being supplied to the target board microprocessor subsystem; and

wherein the master controller board controls the amount of power provided by the target board power to the target board microprocessor subsystem via the interboard bus control interface and the interboard bus.

The system of claim 10 wherein the master controller board further comprises a board-ID database for storing the board-ID value of the target board.

The system of claim 11 wherein the master controller board further comprises a master controller board microprocessor subsystem for retrieving a value from the board-ID database to determine if any programmable memory devices exist on the target board.
The system of claim 10 wherein the signals transferred through the interboard bus include control signals, address signals, and data signals.

The system of claim 10 wherein the master controller board provides power to the master power bus.

The system of claim 10 wherein the master controller board further comprises a programmable memory device image storage database for storing the plurality of programming images for configuring the target boards.

A method for storing programming images on programmable memory devices in a system comprising a master controller board and one or more target boards, each of the target boards including target board power, an interboard bus control interface, a programmable memory device, and a microprocessor subsystem, the method comprising:

- supplying target board power to each of the target boards;
- obtaining a board-ID for each of the target boards;
- powering a master power bus via the master controller board;
- powering the interboard bus control interface on the target boards via the master power bus;
- controlling the amount of power provided to the devices on the target board by the target board power via the interboard bus control interface;
- detecting a target board via the master controller board;
- communicating with the target board via the master controller board to determine data to be imaged based on the board-ID;
- correlating the board-IDs in the target boards to programming images in the master controller board;
- configuring and downloading the programming images to the programmable memory device via the master controller board;
- acknowledging that images have been received via the target board; and
- restoring target board power to all devices on the target board via the master controller board.

The method of claim 16 further comprising providing target board power to the programmable memory device and not providing target board power to the microprocessor subsystem when programming images are being configured and downloaded to the programmable memory device.

The method of claim 16 wherein the master controller board detects the target board via a board-ID.

The method of claim 16 further comprising operating the target boards in accordance with the programming images loaded into the programmable memory devices on the target boards.
The method of claim 16 further comprising providing a programmable memory image storage database on the master controller board.
FIG. 4a

1. Start
2. MCB Advances To Next TB
3. System Power On MCB Power On All TB Power Off
4. NO: MCB Detects Powered Off TB Presence
5. YES: MCB Reads TB ID
6. 100
7. 124
8. 120
9. 110
10. 122
11. 130
12. 21
13. TB's Board ID via Interboard bus
14. TB Presence and Power Status via Interboard bus
15. 2
FIG. 4b

1. MCB Reads Image/Configuration From Database based on TB ID
2. MCB Writes Image/Configuration Into TB via Interboard bus
3. MCB Advances To Next TB Image/Configuration For TB ID

Additional Entries For TB ID?

3. Additional Entries For TB ID?
FIG. 4c

3

180

MCB Powers TB Power

190

MCB Waits For Power Up ACK From TB Via Interboard Bus

1