

(51) International Patent Classification:
G06F 9/44 (2006.01)(21) International Application Number:
PCT/US2010/034103(22) International Filing Date:
7 May 2010 (07.05.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
61/176,272 7 May 2009 (07.05.2009) US(71) Applicant (for all designated States except US): **CY-
PRESS SEMICONDUCTOR CORPORATION**
[US/US]; 198 Champion Court, San Jose, CA 95134
(US).

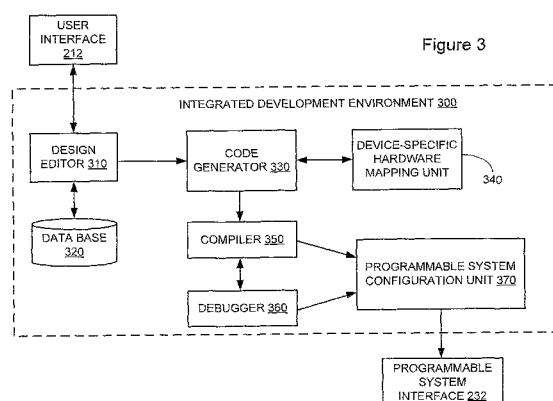
(72) Inventors; and

(75) Inventors/Applicants (for US only): **GRIFFIN, Jack**
[US/US]; 6500 Apollo Road, West Linn, OR 97068 (US).
MOHAMMED, Haneef [US/US]; 10844 Sw Nutcracker
Court, Beaverton, OR 97007 (US).(74) Agents: **RICHMOND, Jeffrey, J.** et al.; Stoloritz Ford
Cowger LLP, 621 SW Morrison Street, Ste 600, Portland,
OR 97205 (US).(81) Designated States (unless otherwise indicated, for every
kind of national protection available): AE, AG, AL, AM,
AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ,
CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO,
DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT,
HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP,
KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD,
ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI,
NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD,
SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR,
TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.(84) Designated States (unless otherwise indicated, for every
kind of regional protection available): ARIPO (BW, GH,
GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG,
ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ,
TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK,
EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU,
LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK,
SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ,
GW, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

(54) Title: DEVELOPMENT, PROGRAMMING, AND DEBUGGING ENVIRONMENT



(57) Abstract: A method includes receiving hardware description code that generically describes circuitry, and translating the hardware description code into one or more configuration files specific to a programmable system. The method further includes generating program code for a microcontroller of the programmable system based, at least in part, on the hardware description code, and configuring the programmable system to implement the circuitry according to the configuration files and the program code.

DEVELOPMENT, PROGRAMMING, AND DEBUGGING ENVIRONMENT**RELATED APPLICATION**

This patent application claims benefit of and priority to U.S. Provisional Patent
5 Application No. 61/176,272, filed May 7, 2009, which is incorporated by reference herein.

TECHNICAL FIELD

This disclosure relates generally to electronic systems, and, more particularly, to
developing, programming, and debugging environment for programmable systems.

BACKGROUND

10 Microcontroller manufacturers and vendors often supply their customers with
development tools that allow programmers to create software for the microcontrollers to
execute. Similarly, many configurable hardware manufacturers will provide their customers
with specialized hardware configuration tools that allow designers the ability to configure
their hardware devices.

15 Some electronic systems include both configurable hardware components and a
processing device, which can be programmed and configured to work together to implement
various functions. When configuring these electronic systems, designers often will utilize
software tools to program the processing device and utilize the specialized hardware
configuration tools to configure the hardware components. In other words, the system
20 designers manually manage multiple projects, e.g., the use of the multiple development tools,
with differing development methodologies when attempting to cohesively develop, program,
and debug these electronic systems.

SUMMARY

The patent application describes a method including receiving hardware description code that generically describes circuitry, and translating the hardware description code into one or more configuration files specific to a programmable system. The method also includes generating program code for a microcontroller of the programmable system based, at least in part, on the hardware description code, and configuring the programmable system to implement the circuitry according to the configuration files and the program code.

A system includes an interface device to receive hardware description code that describes hardware circuitry for a programmable system to implement, and to receive an indication to initiate automatic configuration and programming of the programmable system based on the hardware description code. The system further includes a processing system, responsive to the indication, to automatically generate one or more hardware configuration files and program code based, at least in part, on the hardware description code, and to automatically send the configuration files and the program code to the programmable system, wherein the programmable system is configured to implement the hardware circuitry according to the configuration files and the program code.

DESCRIPTION OF THE DRAWINGS

Figure 1 shows a programmable system configurable by a processing system implementing an integrated development environment according to embodiments of the invention.

Figure 2 illustrates an embodiment of a core architecture of a Programmable System-on-Chip (PSoCTM) shown in Figure 1.

Figure 3 illustrates an example embodiment of the processing system shown in Figure 1.

Figure 4 is an example operational flowchart for the processing device implementing the integrated development environment according to embodiments of the invention.

5 Figure 5 is an example operational flowchart for the integrated development environment according to embodiments of the invention.

DETAILED DESCRIPTION

A Programmable System-on-Chip (PSoCTM), such as that used in the PSoCTM family of products offered by Cypress Semiconductor Corporation (San Jose, California), or other
10 electronic system can include a microcontroller or other processing device and configurable hardware components, such as programmable analog and/or digital blocks. A processing system can implement a unified integrated development environment that allows designers to develop applications and program both the configurable hardware components and the microcontroller of the PSoCTM and/or electronic system. Embodiments are shown and
15 described below in greater detail.

Figure 1 shows a programmable system 100 configurable by a processing system 200 implementing an integrated development environment 300 according to embodiments of the invention. Referring to Figure 1, the programmable system 100 includes a microcontroller 102 and configurable hardware components, such as programmable digital blocks 132 and
20 programmable analog blocks 134. The microcontroller 102 can be programmed (and reprogrammed) and the programmable digital and analog blocks 132 and 134 can be configured (and reconfigured) to implement various applications and perform a variety of functions. Embodiments of the programmable system 100 will be described below in greater detail.

The processing system 200 can implement an integrated development environment 300, allowing unified hardware and software development and configuration of the programmable system 100 with hardware configuration files and software programming developed by the integrated development environment 300. The processing system 200 can
5 include one or more processors 202 to implement the integrated development environment 300, for example, by executing instructions stored in a memory system 204 or other computer readable medium.

After hardware configuration files and software programming is developed, the processing system 200 can program and/or configure the programmable system 100 with the
10 developed hardware configuration and software programming, for example, through a coupling device 230. In some embodiments, the coupling device 230 can be a wired device, such as a Universal Serial Bus (USB) cable, Ethernet cable, etc, or can represent a wireless link between the processing system 200 and the programmable system 100.

The processing system 200 can include system interface devices 206 that allow the
15 processing system 200 to communicate with external devices, such as the user input device 210, the display device 220, and the programmable system 100. For example, the processing system 200 can include a system interface 206 to communicate with the programmable system 100 over the coupling device 230. In some embodiments, the system interface devices 206 can receive inputs, for example, through the user input device 210, and present
20 information, for example, via the display device 220.

The processing system 200 can develop hardware and software applications for the programmable system 100 in response to user input, for example, from the user input device 210. The integrated development environment 300 can include various development tools that allow system designers to describe hardware circuitry for the programmable system 100

to implement and to provide software or firmware code for the microcontroller 102. In some embodiments, the integrated development environment 300 can receive hardware description code that describes this hardware circuitry in an abstracted or generic manner, and can convert the generic code into device-specific configuration files that are particular to the architecture and/or resources of the programmable system 100. The hardware description code provided by the system designers can include schematic circuit diagrams and/or hardware code written according to a hardware description language, such as Verilog or VHDL.

The processing system 200 can also generate application programming interfaces based at least in part on the hardware description code. These application programming interfaces, when provided to the programmable system 100, can program the microcontroller 102 to communicate with the programmable digital and/or analog blocks 132 and 134 configured according to the device-specific configuration files.

The processing system 200 can send the device-specific configuration files and the application programming interfaces to the programmable system 100. The programmable system 100 can utilize the configuration files to configure particular hardware components in the programmable digital and/or analog blocks 132 and 134 to implement the hardware circuitry described by the hardware description code. The programmable system 100 can utilize the application programming interfaces to program the microcontroller 102 to communicate with the programmable digital and/or analog blocks 132 and 134 configured according to the device-specific configuration files.

After the programmable system 100 has been programmed with the hardware configuration and software or firmware programming developed with the integrated development environment 300, the processing system 200 can include debug hardware 208 to

perform debugging operations on the programmable system 100. In some embodiments, the debug hardware 208 can be located externally from the processing system 200 and can communicate with the processing system 200 via the system interface devices 206.

Figure 2 illustrates an embodiment of a core architecture of a Programmable System-on-Chip (PSoCTM), such as that used in the PSoC3TM family of products offered by Cypress Semiconductor Corporation (San Jose, California). Referring to Figure 2, in one embodiment, the core architecture includes the microcontroller 102. The microcontroller 102 includes a CPU (central processing unit) core 104, flash program storage 106, DOC (debug on chip) 108, a prefetch buffer 110, a private SRAM (static random access memory) 112, and special functions registers 114. In an embodiment, the DOC 108, prefetch buffer 110, private SRAM 112, and special function registers 114 are coupled to the CPU core 104, while the flash program storage 106 is coupled to the prefetch buffer 110. The flash program storage 106 can be any type of program memory.

The core architecture may also include a CHub (core hub) 116, including a bridge 118, such as a single-level or multi-level Advanced High-Performance Bus Bridge, and optionally a DMA (direct memory access) controller 120, that is coupled to the microcontroller 102 via bus 122. The Chub 116 may provide the primary data and control interface between the microcontroller 102 and its peripherals and memory, and a programmable core 124. The DMA controller 120 may be programmed to transfer data between system elements without burdening the CPU core 104. In various embodiments, each of these subcomponents of the microcontroller 102 and CHub 116 may be different with each choice or type of CPU core 104. The Chub 116 may also be coupled to shared SRAM 126 and an SPC (system performance controller) 128. The private SRAM 112 is independent of the shared SRAM 126 that is accessed by the microcontroller 102 through the bridge 118.

The CPU core 104 accesses the private SRAM 112 without going through the bridge 118, thus allowing local register and RAM accesses to occur simultaneously with DMA access to shared SRAM 126. Although labeled here as SRAM, these memory modules may be any suitable type of a wide variety of (volatile or non-volatile) memory or data storage modules
5 in various other embodiments.

In various embodiments, the programmable core 124 may include various combinations of subcomponents (not shown), including, but not limited to, a digital logic array, digital peripherals, analog processing channels, global routing, analog peripherals, DMA controller(s), SRAM and other appropriate types of data storage, IO ports, and other
10 suitable types of subcomponents. In one embodiment, the programmable core 124 includes a GPIO (general purpose IO) and EMIF (extended memory interface) block 130 to provide a mechanism to extend the external off-chip access of the microcontroller 102, a programmable digital block 132, a programmable analog block 134, and a special functions block 136, each configured to implement one or more of the subcomponent functions. In various
15 embodiments, the special functions block 136 may include dedicated (non-programmable) functional blocks and/or include one or more interfaces to dedicated functional blocks, such as USB, a crystal oscillator drive, JTAG, and the like.

The programmable digital block 132 may include a digital logic array including an array of digital logic blocks and associated routing. In one embodiment, the digital block
20 architecture is comprised of UDBs (universal digital blocks). For example, each UDB may include an ALU together with CPLD functionality or other types of digital programmable logic functions.

In various embodiments, one or more UDBs of the programmable digital block 132 may be configured to perform various digital functions, including, but not limited to, one or

more of the following functions: a basic I2C slave; an I2C master; a SPI master or slave; a multi-wire (e.g., 3-wire) SPI master or slave (e.g., MISO/MOSI multiplexed on a single pin); timers and counters (e.g., a pair of 8-bit timers or counters, one 16 bit timer or counter, one 8-bit capture timer, or the like); PWMs (e.g., a pair of 8-bit PWMs, one 16-bit PWM, one 8-bit deadband PWM, or the like), a level sensitive I/O interrupt generator; a quadrature encoder, a UART (e.g., half-duplex); delay lines; and any other suitable type of digital function or combination of digital functions which can be implemented in a plurality of UDBs.

In other embodiments, additional functions may be implemented using a group of two or more UDBs. Merely for purposes of illustration and not limitation, the following functions can be implemented using multiple UDBs: an I2C slave that supports hardware address detection and the ability to handle a complete transaction without CPU core (e.g., CPU core 104) intervention and to help prevent the force clock stretching on any bit in the data stream; an I2C multi-master which may include a slave option in a single block; an arbitrary length PRS or CRC (up to 32 bits); SDIO; SGPIO; a digital correlator (e.g., having up to 32 bits with 4x over-sampling and supporting a configurable threshold); a LINbus interface; a delta-sigma modulator (e.g., for class D audio DAC having a differential output pair); an I2S (stereo); an LCD drive control (e.g., UDBs may be used to implement timing control of the LCD drive blocks and provide display RAM addressing); full-duplex UART (e.g., 7-, 8- or 9-bit with 1 or 2 stop bits and parity, and RTS/CTS support), an IRDA (transmit or receive); capture timer (e.g., 16-bit or the like); deadband PWM (e.g., 16-bit or the like); an SMBus (including formatting of SMBus packets with CRC in software); a brushless motor drive (e.g., to support 6/12 step commutation); auto BAUD rate detection and generation (e.g., automatically determine BAUD rate for standard rates from 1200 to 115200 BAUD and after detection to generate required clock to generate BAUD rate); and any other suitable type of

digital function or combination of digital functions which can be implemented in a plurality of UDBs.

The programmable analog block 134 may include analog resources including, but not limited to, comparators, mixers, PGAs (programmable gain amplifiers), TIAs (trans-impedance amplifiers), ADCs (analog-to-digital converters), DACs (digital-to-analog converters), voltage references, current sources, sample and hold circuits, and any other suitable type of analog resources. The programmable analog block 134 may support various analog functions including, but not limited to, analog routing, LCD drive IO support, capacitive sensing, voltage measurement, motor control, current to voltage conversion, voltage to frequency conversion, differential amplification, light measurement, inductive position monitoring, filtering, voice coil driving, magnetic card reading, acoustic doppler measurement, echo-ranging, modem transmission and receive encoding, or any other suitable type of analog function.

Figure 3 illustrates an example embodiment of the processing system 200 shown in Figure 1. Referring to Figure 3, the processing system 200 can implement the integrated development environment 300, for example, by executing instructions stored in the memory system 204 or other computer-readable medium. In some embodiments, the integrated development environment 300 can be at least partially implemented by a set of one or more discrete hardware components (not shown) in the processing system 200.

The integrated development environment 300 can include a design editor 310 to receive information describing hardware circuitry. This information describing hardware circuitry can be received from various sources and in various formats, for example, through a user interface 212. The design editor 310 can include various development tools that present a user or system designer options for inputting circuit designs or descriptions to the integrated

development environment 300. For instance, the design editor 310 can receive code written according to a hardware description language, such as Verilog or VHDL. The design editor 310 can also provide a graphics-based circuit design application, such as a Schematic Editor, a Symbol Editor, a GPIF (General Programmable Interface) editor, etc, which allows

5 designers to create schematic diagrams of the hardware circuitry to be implemented by the programmable system 100. In some embodiments, the design editor 310 can access a database 320 to help determine dependency, build rules, and debug rules for the received descriptions of the hardware circuitry.

The design editor 310 can also receive user-generated program code from the user

10 interface 222. The program code can utilize at least one application programming interface generated by the integrated development environment to communicate with the hardware components in the programmable system 100. This program code can also include at least one application programming interface to allow the microcontroller 102 in the programmable system 100, when programmed with the code, to communicate with hardware components in

15 the programmable system 100.

The integrated development environment 300 can include a code generator 330 to generate configuration files from the received descriptions of the hardware circuitry. In some embodiments, when the received descriptions of the hardware circuitry are in an abstracted or generic format, the code generator 330 can access a device-specific hardware mapping unit

20 340 to map the received descriptions of the hardware circuitry to the programmable digital and/or analog blocks 132 and 134 of the programmable system 100. In other words, the code generator 330 can determine where and how the programmable system 100 implements the generic circuitry provided by the user or system designer. This level of abstraction can allow users without specific knowledge of the programmable system 100 the ability to program and

configure the programmable system 100 to perform various applications through the use of generic circuit descriptions and diagrams. The code generator 330 can generate the configuration files from the device-specific version of the hardware circuitry descriptions.

5 The code generator 330 can also generate application programming interfaces from the received descriptions of the hardware circuitry. The application programming interface, when provided to the programmable system 100, can program the microcontroller 102 and allow it to communicate with hardware components of the programmable system 100.

The integrated development environment 300 can include a compiler 350 to compile the configuration files and the application programming interfaces and link them to the programmable system 100. Once the configuration files and the application programming
10 interfaces have been compiled and linked, the compiler 350 can provide them to a programmable system configuration unit 370 to send them to the programmable system 100, for example, via a programmable system interface 232. The programmable system 100 can configure its programmable digital and/or analog blocks 132 and 134 according to the
15 configuration files and program the microcontroller 102 according to the application programming interfaces in order to implement the hardware circuitry described by the user.

The compiler 350 can also provide the configuration files and the application programming interfaces to a debugger 360, such as the debug hardware 208. The debugger 360 can perform debugging operations on the programmable system 100 as configured with
20 the configuration files and the application programming interfaces. For instance, the debugger 360 can perform step over, step into, and step out operations, which allows users the ability to perform incremental evaluations that step through programming code.

Figure 4 is an example operational flowchart for the processing device implementing the integrated development environment 300 according to embodiments of the invention.

Referring to Figure 4, the integrated development environment 300 can receive hardware description code 401, such as hardware description language code 402, state diagrams 403, hardware schematics 404, and flowcharts 405, which can describe hardware circuitry. The hardware circuitry can include one or more circuits to perform various application or

5 functions and analog and/or digital signal routing associated with the circuits. The hardware description language code 402 can be written in Verilog, VHDL, or other similar hardware description language. The hardware schematics 404 can be schematic diagrams of the hardware circuitry created with a graphics-based circuit design application, such as a Schematic Editor, a Symbol Editor, a GPIF (General Programmable Interface) editor, etc.

10 The integrated development environment 300, in a block 410, can netlist the hardware description language code 402, the state diagrams 403, the hardware schematics 404, and/or the flowcharts 405 into a single representation of the hardware circuitry to be implemented by the programmable system 100. This netlisting of the hardware description language code 402, the state diagrams 403, the hardware schematics 404, and/or the flowcharts 405 can
15 combine and integrate the circuitry descriptions, which have various formats, into the single representation of the hardware circuitry.

The integrated development environment 300, in a block 420, can perform high-level synthesis on the netlisted hardware description code. The high-level synthesis can break-down or reduce the netlisted hardware description code into lower level primitives, logic
20 equations, and/or flip-flops. This reduction of the netlisted hardware description code allows the integrated development environment 300, in a block 430, to map the reduced hardware description code to the programmable system 100 through low-level synthesis. The integrated development environment 300 can determine which hardware resources or components within the program system 100, such as the programmable digital blocks 132 and

the programmable analog blocks 134, can implement the circuitry described by the reduced hardware description code according to a mapping.

The integrated development environment 300, in blocks 440 and 450, can perform placement and routing for both the programmable digital blocks 132 and the programmable analog blocks 134 of the programmable system 100. The placement and routing can determine where the hardware circuitry is to be placed in the programmable digital blocks 132 and the programmable analog blocks 134. The placement and routing can also allocate or set signal routing for the hardware circuitry placed in the programmable digital blocks 132 and the programmable analog blocks 134.

The integrated development environment 300, in a block 460, can generate perform hardware configuration files and application programming interfaces. The hardware configuration files can be based on the mapping of the reduced hardware description code and the place and routing analysis performed in blocks 430-450. The application programming interfaces can be based on the mapping of the reduced hardware description code and the place and routing performed in blocks 430-450, and can be based on software programming code 406 received from at least one system interface. The software programming code can include at least one application programming interface to allow the microcontroller 102 in the programmable system 100, when programmed with the software programming code, to communicate with hardware components in the programmable system 100.

The integrated development environment 300, in a block 470, can compile the hardware configuration files and the application programming interfaces, and link them to the programmable system 100. The integrated development environment 300, in a block 480, can send the compiled and linked hardware configuration files and the application

programming interfaces to the programmable system 100. The programmable system 100 can be configured to implement the hardware circuitry described in the hardware description language code 402, the state diagrams 403, the hardware schematics 404, and/or the flowcharts 405 responsive to the hardware configuration files and the application programming interfaces. The integrated development environment 300, in a block 490, can execute a debugging application to debug the programmable system 100 as configured with the hardware configuration files and the application programming interfaces.

In some embodiments, the integrated development environment 300 can receive an indication to initiate automatic configuration and programming of the programmable system 100 after receiving the input information 402, 404, and 406. The integrated development environment 300 can automatically perform operations associated with the blocks 410-490 in response to receiving the indication. In some embodiments, the indication can be received from a user via at least one of the system interfaces. Although Figure 4 shows blocks 410-490 being performed in a particular processing order, in some embodiments the integrated development environment 300 can perform the operations in different orders.

Figure 5 is an example operational flowchart of the integrated development environment 300 according to embodiments of the invention. Referring to Figure 5, in a first block 510, the integrated development environment 300 can receive hardware description code that generically describes circuitry. In some embodiments, the hardware description code can be code written in Verilog, VHDL, or other similar hardware description language, or schematic diagrams of the circuitry created with a graphics-based circuit design application, such as a Schematic Editor, a Symbol Editor, a GPIF (General Programmable Interface) editor, etc. The integrated development environment 300 can also receive program code for a microcontroller 102 in the programmable system 100

In a next block 520, the integrated development environment 300 can receive an indication to initiate automatic configuration and programming of the programmable system.

In some embodiments, the indication can be received from a user via at least one of the system interfaces. The integrated development environment 300 can automatically perform

5 the blocks 530-560 in response to receiving the indication.

In block 530, the integrated development environment 300 can translate the hardware description code into one or more configuration files specific to a programmable system 100.

The translation of the hardware description code into the configuration files can include multiple operations. For example, the hardware description code can be netlisted into a

10 single representation of the circuitry. The netlisted code can be reduced into lower-level primitives, logic expressions, and flip-flops. The reduced code can be mapped to the programmable device 100 to determine how the programmable system 100 can implement the circuitry. The mapped code can be analyzed to determine placement and routing of the circuitry implemented by the programmable system 100. The integrated development

15 environment 300 can translate the mapped code that has undergone placement and routing analysis into one or more configuration files that are specific to the programmable system 100.

In block 540, the integrated development environment 300 can generate program code for a microcontroller 102 of the programmable system 100 based, at least in part, on the

20 hardware description code. In some embodiments, the program code can be application programming interfaces for the microcontroller 102 to communicate with the hardware components of the programmable system 100.

In block 550, the integrated development environment 300 can configure the programmable system 100 to implement the circuitry according to the configuration files and

the program code. The integrated development environment 300 can provide the configuration files and the program code to the programmable system 100. The configuration files and the program code can prompt the programmable system 100 to implement the circuitry described by the hardware description code.

5 In block 560, the integrated development environment 300 can debug the programmable system 100 as programmed by the configuration files and the program code.

 One of skill in the art will recognize that the concepts taught herein can be tailored to a particular application in many other ways. In particular, those skilled in the art will recognize that the illustrated embodiments are but one of many alternative implementations
10 that will become apparent upon reading this disclosure.

 The preceding embodiments are examples. Although the specification may refer to “an”, “one”, “another”, or “some” embodiment(s) in several locations, this does not necessarily mean that each such reference is to the same embodiment(s), or that the feature only applies to a single embodiment.

15

CLAIMS

1. A method comprising:
receiving program code and hardware description code that generically describes
circuitry;
5 translating the hardware description code into one or more configuration files specific
to a programmable system;
generating additional program code for a microcontroller of the programmable system
based, at least in part, on the hardware description code; and
configuring the programmable system to implement the circuitry according to the
10 configuration files and the program code.
2. The method of claim 1, wherein the program code includes at least one application
programming interface to allow the microcontroller to communicate with programmable
hardware components in the programmable system.
15
3. The method of claim 1 further comprises debugging the programmable system
after programmable hardware components in the programmable system are configured to
implement the circuitry according to the configuration files and the program code.
- 20 4. The method of claim 1, wherein the hardware description code include at least one
of schematic circuit diagrams, flowchart, state diagram, or code written according to a
hardware description language.

5. The method of claim 1 wherein the translating of the hardware description code into configuration files further comprises:

synthesizing the hardware description code into a single representation of the circuitry, wherein the single representation of the circuitry includes a whole or a portion of a

5 design of the programmable system; and

mapping the single representation of the circuitry to specific programmable hardware components in the programmable system, wherein the specific programmable hardware components in the programmable system are configurable to implement the single representation of the circuitry.

10

6. The method of claim 5, wherein the synthesizing of the hardware description files into the single representation of the circuitry further comprises reducing the single representation of the circuitry the into at least one of lower level primitives, logic expressions, or flip-flops.

15

7. The method of claim 5, wherein the mapping of the single representation of the circuitry to the programmable system further comprises setting-up signal routing for the specific hardware components in the programmable system to implement the circuitry.

20

8. The method of claim 1 further comprises receiving an indication to initiate automatic configuration and programming of the programmable system, wherein the translating of the hardware description code into configuration files, the generating of the program code, and the configuring of the programmable system are automatically performed responsive to receiving the indication.

9. A system comprising:

an interface device to receive hardware description code that describes hardware circuitry for a programmable system to implement, and to receive an indication to initiate
5 automatic configuration and programming of the programmable system based on the hardware description code; and

a processing system, responsive to the indication, to automatically generate one or more hardware configuration files and program code based, at least in part, on the hardware description code, and to automatically send the configuration files and the program code to
10 the programmable system, wherein the programmable system is configured to implement the hardware circuitry according to the configuration files and the program code.

10. The system of claim 9, wherein the program code includes at least one application programming interface to allow a microcontroller in the programmable system to
15 communicate with programmable hardware components in the programmable system.

11. The system of claim 9, wherein the processing system is configured to synthesize the hardware description code into a single representation of the hardware circuitry, and to map the single representation of the hardware circuitry to specific programmable hardware
20 components in the programmable system, wherein the specific programmable hardware components in the programmable system are configurable to implement the single representation of the hardware circuitry.

12. The system of claim 9, wherein the processing system is configured to debug the programmable system after programmable hardware components in the programmable system are configured to implement the hardware circuitry according to the configuration files and the program code.

5

13. An apparatus having a computer-readable medium storing instructions configured to cause a processing system to perform operations comprising:

receiving program code and hardware description code that generically describes circuitry;

10 translating the hardware description code into one or more configuration files specific to a programmable system;

generating additional program code for a microcontroller of the programmable system based, at least in part, on the hardware description code; and

15 configuring the programmable system to implement the circuitry according to the configuration files and the program code.

14. The apparatus of claim 13, wherein the program code includes at least one application programming interface to allow the microcontroller to communicate with programmable hardware components in the programmable system.

20

15. The apparatus of claim 13, wherein the instructions are configured to cause the processing system to perform operations further comprising debugging the programmable system after programmable hardware components in the programmable system are

configured to implement the circuitry according to the configuration files and the program code.

16. The apparatus of claim 13, wherein the hardware description code include at least
5 one of schematic circuit diagrams, flowchart, state diagram, or code written according to a hardware description language.

17. The apparatus of claim 13, wherein the instructions are configured to cause the processing system to perform operations further comprising:
10 synthesizing the hardware description code into a single representation of the circuitry, wherein the single representation of the circuitry includes a whole or a portion of a design of the programmable system; and
mapping the single representation of the hardware circuitry to specific programmable hardware components in the programmable system, wherein the specific programmable
15 hardware components in the programmable system are configurable to implement the single representation of the circuitry.

18. The apparatus of claim 17, wherein the instructions are configured to cause the processing system to perform operations further comprising reducing the single
20 representation of the circuitry the into at least one of lower level primitives, logic expressions, or flip-flops.

19. The apparatus of claim 17, wherein the instructions are configured to cause the processing system to perform operations further comprising setting-up signal routing for the specific hardware components in the programmable system to implement the circuitry.

- 5 20. The apparatus of claim 13 wherein the instructions are configured to cause the processing system to perform operations further comprising receiving an indication to initiate automatic configuration and programming of the programmable system, wherein the translating of the hardware description code into configuration files, the generating of the program code, and the configuring of the programmable system are automatically performed
- 10 responsive to receiving the indication.

Figure 1

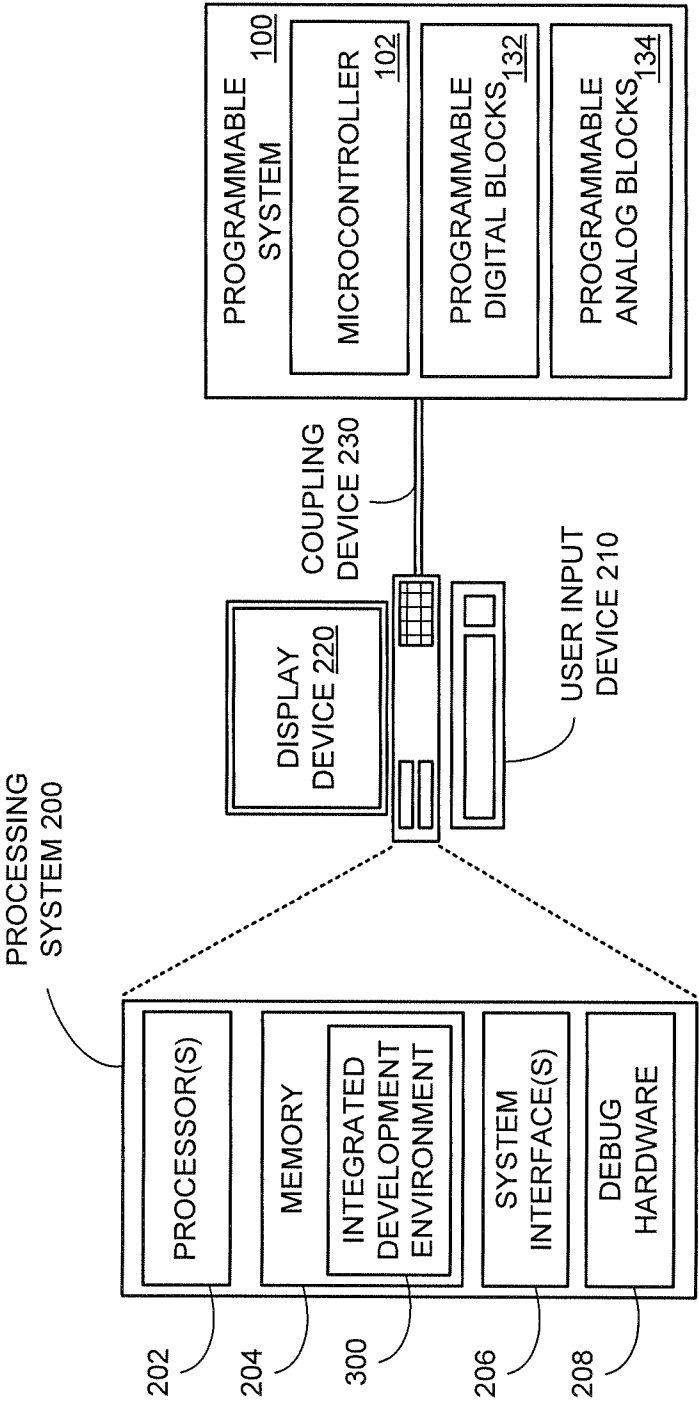


Figure 2

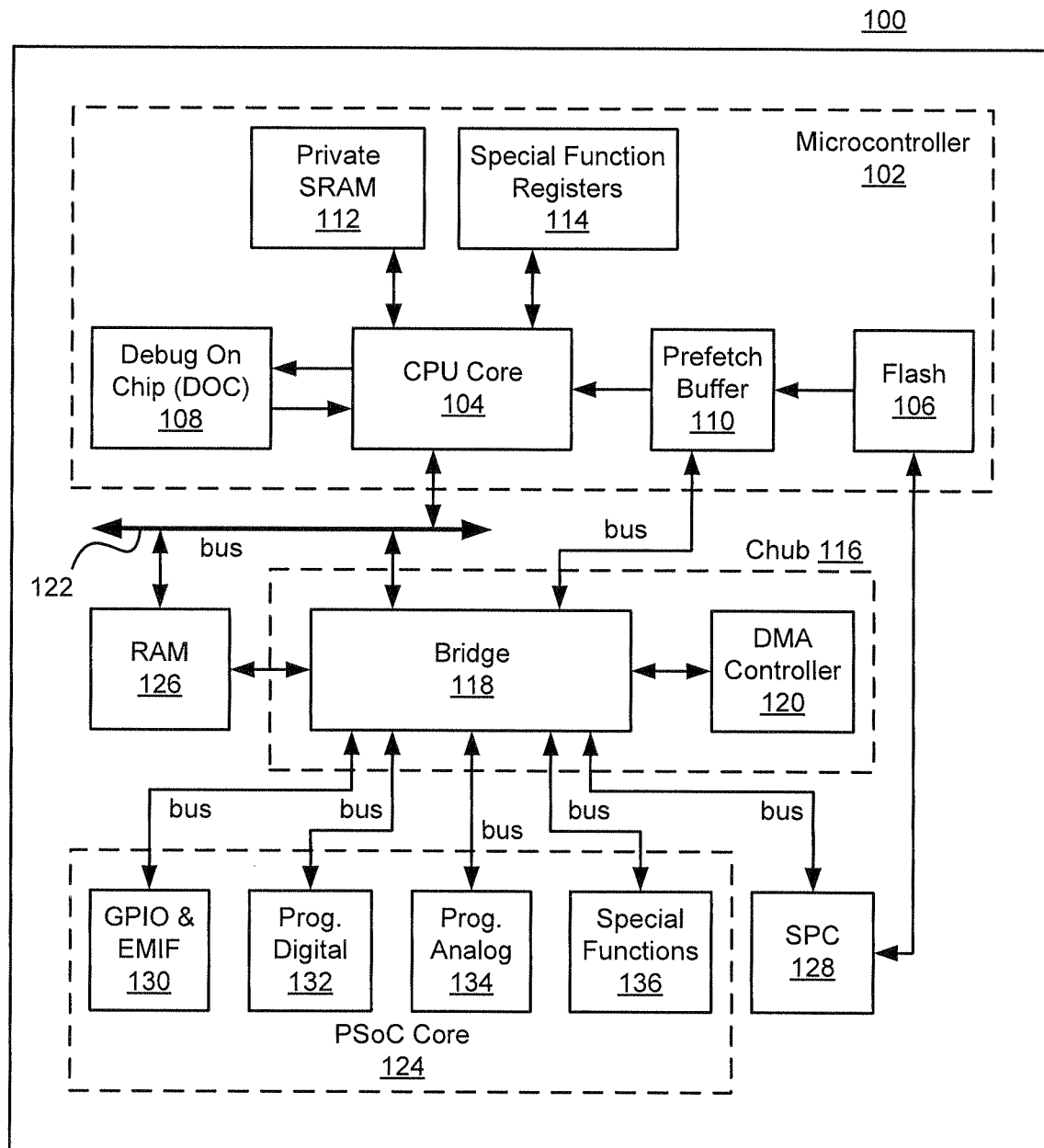


Figure 3

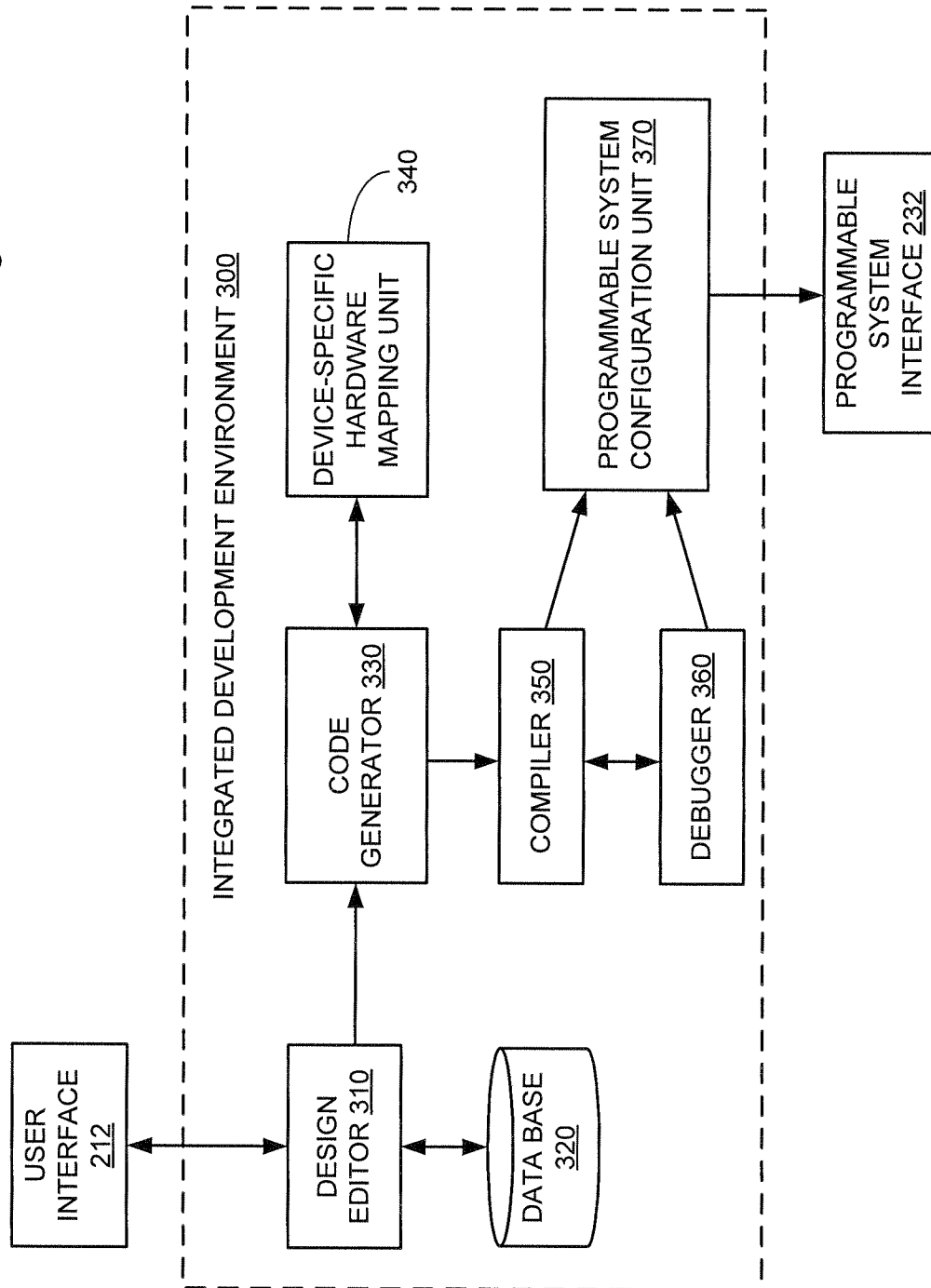


Figure 4

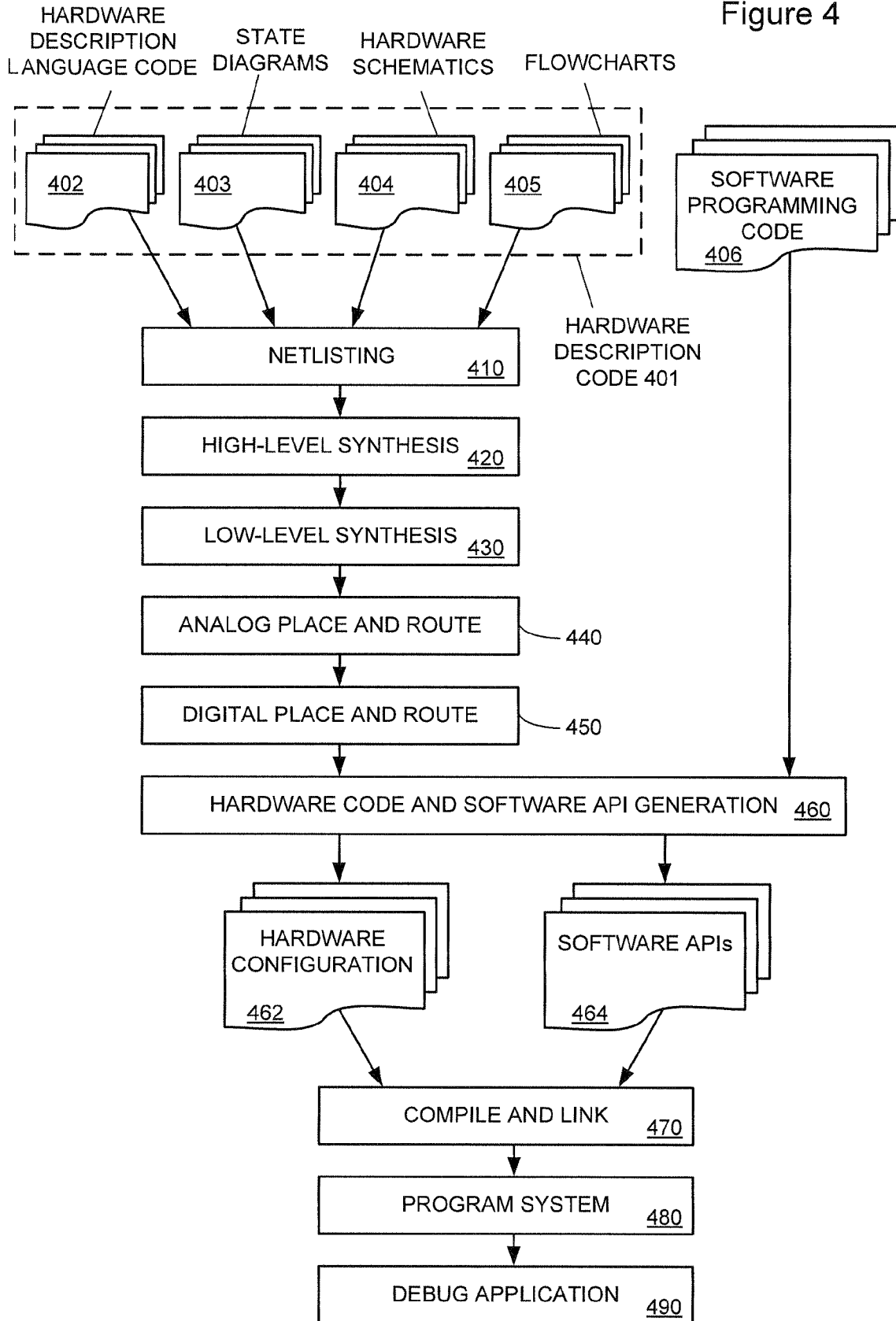
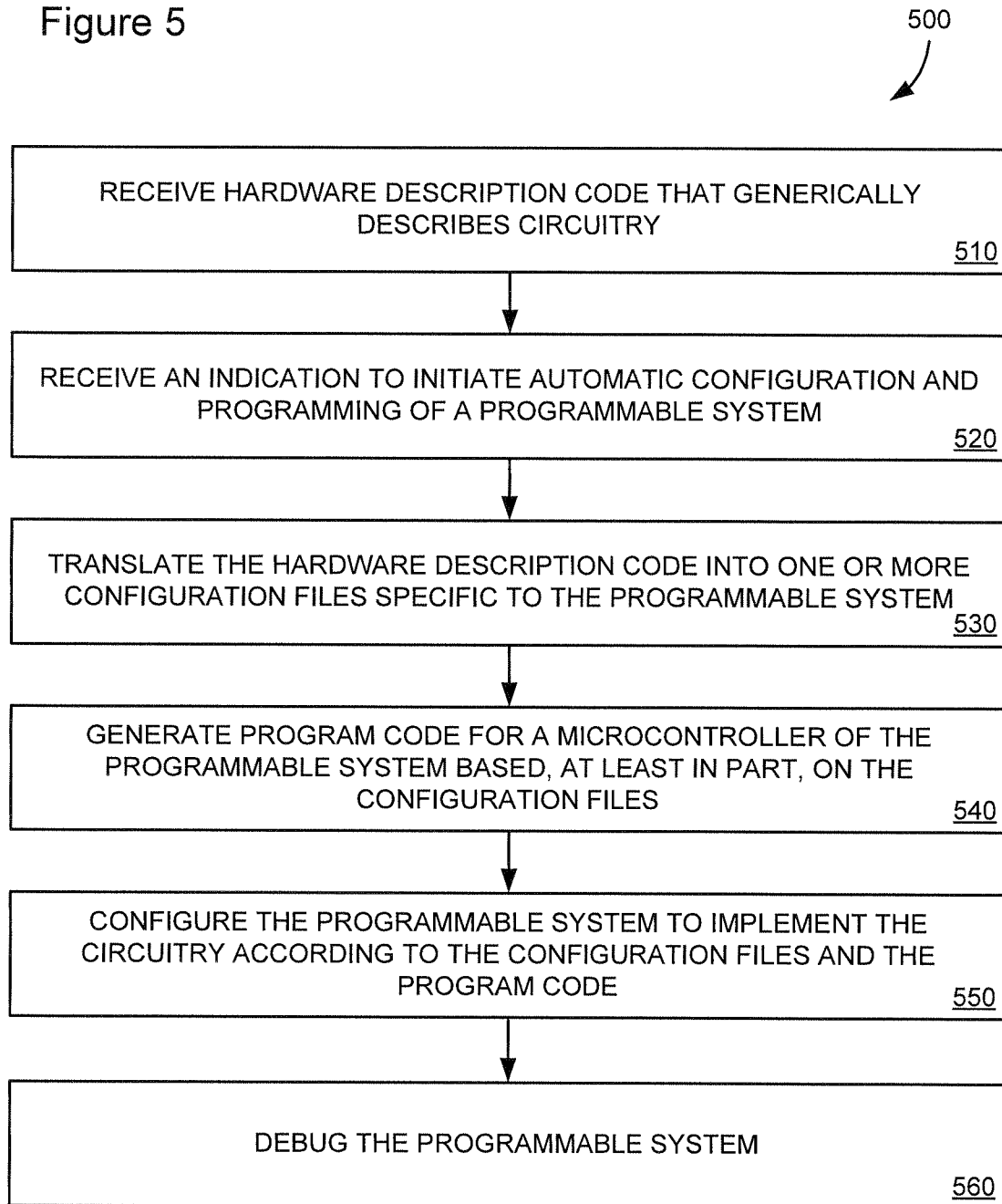


Figure 5



INTERNATIONAL SEARCH REPORT

International application No.

PCT/US 10/34103

A. CLASSIFICATION OF SUBJECT MATTER

IPC(8) - G06F 9/44 (2010.01)

USPC - 717/122

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC (8) - G06F 9/44 (2010.01)

USPC - 717/122

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

USPC - 717/100; 717/124; 717/140; 703/13 (See Keywords Below)

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

Pub WEST (USPT, PGPB, JPAB, EPAB), Google Scholar

Search Terms: Receive, input, enter, capture, program code, instruction, HDC, hardware description code, circuit, translate, convert, configuration file, configuration code, controller, microcontroller, processor, implementing, deploying, API, application program interface...

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2003/0192032 A1 (ANDRADE et al.), 09 October 2003 (09.10.2003), entire document, especially Para [0047]-[0048], [0182]-[0185], [0214]-[0217], [0223], [0227], [0294], [0356]	1-20
Y	US 2008/0195856 A1 (FORD et al.), 14 August 2008 (14.08.2008), entire document, especially Para [0021]-[0022], [0044], [0049]-[0050]	1-20
A	US 2003/0131325 A1 (SCHUBERT et al.), 10 July 2003 (10.07.2003), entire document	1-20
A	US 2006/0036988 A1 (ALLEN et al.), 16 February 2006 (16.02.2006), entire document	1-20



Further documents are listed in the continuation of Box C.



* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search

01 August 2010 (01.08.2010)

Date of mailing of the international search report

18 AUG 2010

Name and mailing address of the ISA/US

Mail Stop PCT, Attn: ISA/US, Commissioner for Patents
P.O. Box 1450, Alexandria, Virginia 22313-1450

Facsimile No. 571-273-3201

Authorized officer:

Lee W. Young

PCT Helpdesk: 571-272-4300

PCT OSP: 571-272-7774