



(43) International Publication Date
23 January 2014 (23.01.2014)

- (51) International Patent Classification:
G06F 13/42 (2006.01) *G06F 9/44* (2006.01)
- (21) International Application Number:
PCT/US2013/050946
- (22) International Filing Date:
17 July 2013 (17.07.2013)
- (25) Filing Language: English
- (26) Publication Language: English
- (30) Priority Data:
61/672,530 17 July 2012 (17.07.2012) US
- (71) Applicant: MILWAUKEE ELECTRIC TOOLS CORPORATION [US/US]; 13135 West Lisbon Road, Brookfield, WI 53005 (US).
- (72) Inventor: MERGENER, Matthew, J.; W149 N9935 Rimrock Road, Germantown, WI 53022 (US).
- (74) Agents: RIZZUTO, Kevin, P. et al.; Michael Best & Friedrich LLP, 100 East Wisconsin Avenue, Suite 3300, Milwaukee, WI 53202-4108 (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, BN, 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, 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, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, 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, RW, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, 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, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

[Continued on next page]

(54) Title: UNIVERSAL PROTOCOL FOR POWER TOOLS

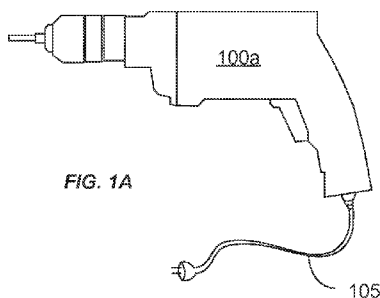


FIG. 1A

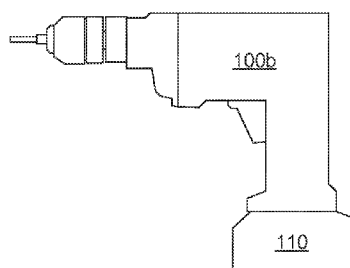


FIG. 1B

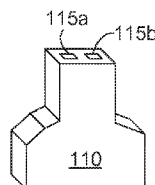


FIG. 1C

(57) Abstract: A system and method for communicating with power tools using a universal protocol. The universal protocol may be implemented using a universal core module that is installed across a variety of power tools and other devices to enable communications therewith. Communications to and from the power tools are translated to a universal protocol once received. The translated communications are handled by the universal core module of a particular tool according to a set of rules. In response, the universal core module outputs communications according to the universal protocol and the set of rules, which may be translated to another protocol for receipt by components of the tool or an external device. The communications may be used, for example, to obtain tool performance data from the tools and to provide firmware updates.



Published:

- *without international search report and to be republished
upon receipt of that report (Rule 48.2(g))*

UNIVERSAL PROTOCOL FOR POWER TOOLS

RELATED APPLICATIONS

[0001] This application claims priority to U.S. Provisional Patent Application No. 61/672,530, filed July 17, 2012, the entire contents of which are incorporated herein by reference.

BACKGROUND

[0002] The present invention relates to communication protocols for power tools and other devices.

SUMMARY

[0003] A protocol is a set of rules defining how communications are to occur between two components. For instance, a protocol may define one or more of syntax, semantics, and synchronization rules for communications between components, including rules relating to data formats for data exchange, address formats for data exchange, address mapping, routing rules, detection/correction of transmission errors, acknowledgment messages, loss of information handling, direction and timing of information flow, sequence control, and flow control. The complexities of protocols selected for use in particular applications may vary.

[0004] Embodiments of the invention relate to a universal protocol for communicating with power tools to enable, for instance, the retrieval of performance data or the installation of firmware. Embodiments of the invention also relate to an interface module implementing a universal protocol that is used across a variety of power tools and other devices to enable communications with the power tools and other devices.

[0005] In one embodiment, the invention provides a method of implementing a universal protocol on a plurality of tools, the plurality of tools including a variety of power tool types. The method includes generating an interfacing module including a universal core module and an interface module, and then installing the interfacing module on a memory of each of the plurality of tools. The universal core module is used to implement the universal protocol and is generic to the plurality of tools. The universal core module includes a set of rules enabling interpretation

and handling of communications that are in accordance with the universal protocol. The interface module is particular to each of the power tool types and is configured to translate communications on route to the universal core module to the universal protocol and to translate communications output by the universal core module from the universal protocol to another protocol.

[0006] In another embodiment, the invention provides a universal protocol system including a plurality of tools and an interfacing module. The plurality of tools includes a variety of power tool types, each tool including a memory. The interfacing module is installed on the memory of each tool and includes a universal core module and an interface module. The universal core module is used to implement a universal protocol and is generic to the plurality of tools. The universal core module also includes a set of rules enabling interpretation and handling of communications that are in accordance with the universal protocol. The interface module is particular to each of the power tool types. The interface module is configured to translate communications on route to the universal core module to the universal protocol and to translate communications output by the universal core module from the universal protocol to another protocol.

[0007] In some embodiments of the method and system, the variety of power tool types include tools having various power platforms. For instance, in some embodiments, the variety of power tool types includes tools powered by at least an AC power source and a DC power source. Additionally, in some embodiments, the variety of power tool types includes tools powered by at least two of a 4 volt battery pack, a 12 volt battery pack, an 18 volt battery pack, a 28 volt battery pack, and a 36 volt battery pack. Thus, the universal core module is generic to the plurality of tools having a plurality of power platforms.

[0008] Other aspects of the invention will become apparent by consideration of the detailed description and accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

[0009] Figs. 1A-B illustrate tools incorporating a universal protocol according to embodiments of the invention.

[0010] Fig. 1C illustrates a battery for powering the tool of Fig. 1B.

[0011] Fig. 2 illustrates a controller of a tool incorporating the universal protocol.

[0012] Fig. 3 illustrates a memory of a tool incorporating the universal protocol.

[0013] Fig. 4 illustrates interface software, including universal core software for implementing the universal protocol.

[0014] Figs. 5A-B illustrate exemplary communication flows between an external device and a tool component according to the universal protocol.

[0015] Fig. 6 illustrates a method of implementing a universal core module on a plurality of various tools and accessories.

[0016] Fig. 7 illustrates a method for implementing an exemplary rule set of the universal core module.

DETAILED DESCRIPTION

[0017] Before any embodiments of the invention are explained in detail, it is to be understood that the invention is not limited in its application to the details of construction and the arrangement of components set forth in the following description or illustrated in the following drawings. The invention is capable of other embodiments and of being practiced or of being carried out in various ways.

[0018] It should also be noted that a plurality of hardware and software based devices, as well as a plurality of different structural components, may be used to implement the invention. In addition, it should be understood that embodiments of the invention may include hardware, software, and electronic components or modules that, for purposes of discussion, may be illustrated and described as if the majority of the components were implemented solely in hardware. However, one of ordinary skill in the art, and based on a reading of this detailed description, would recognize that, in at least one embodiment, the electronic based aspects of the invention may be implemented in software (e.g., stored on non-transitory computer-readable medium) executable by one or more processors. As such, it should be noted that a plurality of

hardware and software based devices, as well as a plurality of different structural components may be utilized to implement the invention. Furthermore, and as described in subsequent paragraphs, the specific mechanical configurations illustrated in the drawings are intended to exemplify embodiments of the invention and that other alternative mechanical configurations are possible. For example, “controllers” described in the specification can include standard processing components, such as one or more processors, one or more computer-readable medium modules, one or more input/output interfaces, and various connections (e.g., a system bus) connecting the components. In some instances, the controllers described in the specification may be implemented in one of or a combination of a general processor, an application specific integrated circuit (ASIC), a digital signal processor (DSP), a field programmable gate array (FPGA), or the like.

[0019] Figs. 1A and 1B each illustrate a power tool 100 incorporating a universal protocol according to embodiments of the invention. The power tool 100a of Fig. 1A is an AC powered tool that receives power from an AC source (not shown), such as a standard 120 V/60 Hz source, via power cord 105. Power tool 100b of Fig. 1B is a DC powered tool that receives power from a DC battery 110. As shown in Fig. 1C, the battery 110 includes two terminals 115, a positive terminal 115a, and a negative terminal 115b, for interfacing with the power tool 100b. Although not shown, the power tool 100b has complimentary terminals for forming a connection with the terminals 115 when the battery 110 is inserted into the power tool 100b. In some instances, additional terminals are provided for duplicating connection points and/or for providing separate communication paths for the power tool 100b to communicate with internal components of the battery 110. The battery 110 is a tower-style battery that may include a latching mechanism to ensure that the battery 110 remains coupled to the power tool 100b during operation. In some embodiments, the battery 110 does not include a vertical tower but, rather, has an alternate battery configuration, such as for slidably engaging the power tool 100b via a horizontal rail and groove arrangement.

[0020] Fig. 2 illustrates a block diagram of the power tool 100. The power tool 100 includes a controller 125 in communication with and receiving power from a power block 130. In the battery-powered power tool 100b, the power block 130 is coupled to the battery 110 via the terminals 115 of the battery 110 and matching terminals (not shown) of the power block 130. In

the AC-powered power tool 100a, an AC source 135 is coupled to the power block 130, rather than the DC battery 110. The power block 130 may further include power regulating and conversion circuitry to ensure that the power provided to various components of the power tool 100a and 100b is at the appropriate level(s).

[0021] The controller 125 is further coupled to a wireless antenna 140 and an input/output (I/O) port 145. As will be described in greater detail below, the power block 130, wireless antenna 140, and I/O port 145 enable the tool 100b to communicate with external devices and are collectively referred to as the physical interface 150.

[0022] The controller 125 is also coupled to user input 155, which may include one or more of an actuating device (e.g., a trigger, button, etc.), one or more selectors (e.g., a forward/reverse selector, light control, etc.), and other input elements (speed level, torque, level, etc.) to provide user commands to the controller to indicate how the user desires the power tool 100 to operate.

[0023] The controller 125 is coupled to a memory 160. The memory 160 includes, among other elements, software that is executed by a processing unit of the controller 125 to control the functions of the tool 100 described herein. The controller 125 is further coupled to a motor drive 165, sensors 170, and user output 175. The controller 125 provides control signals to the motor drive 165 such that the motor drive 165 selectively enables power from one of the AC power source 135 and the battery 110 to drive the motor 180. The control signals to operate the motor drive 165 are generated based on (a) signals received by the controller 125 from the user input 155 and the sensors 170 and (b) software and parameters received and executed by the controller 125 from the memory 160. The motor 180 may be one of a brushless motor and brushed motor.

[0024] The sensors 170 include one or more sensors for detecting various operating conditions of the tool 100, such as the temperature within the tool housing, the rotational position, speed, acceleration, and/or direction of the motor 180, vibrations, acceleration of the tool, the location of the tool, etc. For instance, the sensors 170 may include one or more hall sensors for monitoring the position, speed, acceleration, and/or direction of the motor 180. The controller 125 may receive the outputs of the hall sensors to properly time commutation control signals of the motor drive 165. In some instances, the motor drive 165 may directly receive the hall sensors' outputs to control timing of the commutation of the motor 180. The user output 175

includes one or more of LEDs, a speaker, a vibrating element, etc. to inform the user of the status of the tool 100. For instance, if an error occurs, such as low battery power, the tool 100 may output an audible alert, an LED may flash, and/or the vibrating element may provide tactile feedback to the user. Additionally, the controller 125 may selectively activate and deactivate one or more LEDs that provide illumination of a working area. In some instances, the sensors 170 include a global positioning satellite (GPS) module to provide a location of the tool 100 or a strength-of-signal sensor in wireless communication with another wireless device, such as a fixed wireless unit at a known position, to determine a relative location of the tool with respect to the wireless device.

[0025] Fig. 3 illustrates the memory 160 in greater detail. The memory 160 is illustrated as word-addressable and contains interface software 200, application software 205, parameters 210, and performance data log 215. The application software 205 generally includes the instructions executed by the controller 125 to operate the power tool 100, for instance, to drive the motor in a selected direction based on depression of a trigger. The parameters 210 include stored values that effect the operation of the power tool 100. For instance, various torque, speed, and limit parameters may be stored in the parameters 210. During operation, the controller 125, based on instructions in the application software 205, may retrieve and use values stored in the parameters 210, which affect tool performance. The performance data log 215 stores performance data for the power tool 100. For instance, the performance data log 215 may include usage data for the motor 180, temperature of the power tool 100, error data, battery charge levels, battery current output levels, among other information and sensor data obtained by the controller 125. Usage data for the motor 180 may include motor rotational position, speed, acceleration, direction, temperature, current, etc.

[0026] Although the memory 160 is shown as a singular unit, the memory 160 may be made up of various memories individually coupled to the controller 125 or coupled to the controller 125 via a bus. Additionally, portions of the memory 160 may be embedded within the controller 125. For instance, the parameters 210 may be stored within a memory of the controller 125.

[0027] As shown in Fig. 4, the interface software 200 includes three main components: physical interface software 250, universal core software 255, and tool component interface

software 260. The interface software 200 includes a set of instructions that enables the controller 125 to carry out commands provided by external devices. For example, the interface software 200 enables external devices to communicate with components of the tool 100, such as with tool components 265, including the memory 160, the sensors 170, the user output 175, and the motor drive 165 (see Fig. 2).

[0028] The physical interface software 250 includes translation software that enables the controller 125 to interpret signals received via the physical interface 150. For instance, the wireless antenna 140 may receive and send signals to the controller 125 according to a wireless protocol, and the I/O port 145 may receive and send signals to the controller 125 according to a different protocol. The physical interface software 250, however, enables the controller 125 to translate the signals received from the wireless antenna 140 and I/O port 145 to a universal protocol. The translated signals may include a combination of commands and data (collectively, the “universal protocol input”), which are handled by the controller 125 according to the universal core software 255.

[0029] The universal core software 255 includes a set of rules that enables the controller 125 to carry out the commands provided via the physical interface software 250 and tool component interface software 260. The universal core software 255 may also include a boot loader for the controller 125. Based on the universal protocol input and the rules of the universal core software 255, the universal core software 255 provides universal protocol output, i.e., commands and data according to the universal protocol, to the physical interface software 250 and to the tool component interface software 260, as appropriate for a particular situation. The tool component interface software 260 translates the universal protocol output to one or more protocols used to interact with the various tool components 265.

[0030] Figs. 5A-B illustrate exemplary communication flows between an external device 275 and one of the tool components 265, such as the memory 160. The example of Figs. 5A-B will be described as a read of the memory 160 of the tool 100 by the external device 275; however, the communications from the external device 275 may be for other reasons as well, such as memory writes, sensor reads, firmware updates, troubleshooting, etc.

[0031] As shown in Fig. 5A, the external device 275, such as a smart phone, laptop, handheld fob device, another power tool, or an accessory, communicates a read request with the goal of receiving the performance data log 215 of the memory 160. The read request is communicated in a first protocol, such as a wireless protocol, via the physical interface 150, to the controller 125. The controller 125, via the physical interface software 250, translates the read request from the first protocol to the universal protocol. After the translation, the controller 125, via the universal core software 255, interprets and handles the translated read request according to a set of rules of the universal core software 255. For instance, the read request may include an identifier and security element (e.g., a password) of the requesting external device 275 and the controller 125 may verify that the external device 275 is authorized to request the read. The controller 125 may also respond to the external device 275 with an acknowledge message, which is translated by the physical interface software 250 from the universal protocol to the first protocol on route to the external device 275. After verification, the controller 125 translates the read request in the universal protocol to a second protocol via the tool component interface software 260. The second protocol is the protocol used to communicate with the memory 160.

[0032] Turning to Fig. 5B, the memory 160 has received the read request in the second protocol and responds by outputting the requested data back to the controller 125 according to the second protocol. The controller 125, via tool component interface software 260, translates the data to the universal protocol. The controller 125, via the universal core software 255, handles the data according to rules, and provides the data in the universal protocol to the physical interface software 250. The controller 125 then translates, via the physical interface software 250, the data to the first protocol. The controller 125 outputs the data in the first protocol, via the physical interface 150, to the external device 275 that originally provided the read request.

[0033] Although the physical interface software 250 and tool component interface software 260 are described as software stored in memory 160 and executed by the controller 125, each may be implemented in part or wholly in hardware of the controller 125 or outside of the controller 125. For example, the physical interface software 250 and/or the tool component interface software 260 may be implemented by one or more separate digital signal processors (DSPs) or general purpose processors executing the physical interface software 250 and/or tool component interface software 260, or by one or more field programmable gate arrays (FPGAs)

or application specific integrated circuits (ASICs). Use of the term “module” rather than “software,” (e.g., the physical interface module), refers to implementations in which software is executed by the controller 125 as described above, implementations in which the software is executed by another processing device (e.g., a DSP) separate from the controller 125, and implementations in which the module is implemented in whole or in part by hardware (e.g., with an FPGA or ASIC).

[0034] Although the interface software 200 is described above as being executed on a power tool 100, embodiments of the invention include versions of the interface software 200 loaded on other types of tools and accessories. For instance, versions of the interface software 200 may be loaded and executed on battery packs, battery chargers, other power tools, test and measurement equipment, vacuum cleaners, work site radios, outdoor power equipment, and vehicles. Other power tools can include drills, circular saws, jig saws, band saws, reciprocating saws, screw drivers, angle grinders, straight grinders, hammers, multi-tools, impact wrenches, rotary hammers, impact drivers, angle drills, pipe cutters, grease guns, and the like. Battery chargers can include wall chargers, multi-port chargers, travel chargers, and the like. Test and measurement equipment can include digital multimeters, clamp meters, fork meters, wall scanners, IR thermometers, laser distance meters, laser levels, remote displays, insulation testers, moisture meters, thermal imagers, inspection cameras, and the like. Vacuum cleaners can include stick vacuums, hand vacuums, upright vacuums, carpet cleaners, hard surface cleaners, canister vacuums, broom vacuums, and the like. Outdoor power equipment can include blowers, chain saws, edgers, hedge trimmers, lawn mowers, trimmers, and the like. The battery pack can also be attachable to and detachable from devices such as electronic key boxes, calculators, cellular phones, head phones, cameras, motion sensing alarms, flashlights, worklights, weather information display devices, a portable power source, a digital camera, a digital music player, a radio, and multi-purpose cutters.

[0035] Additionally, the power tool 100 and other types of tools and accessories may be powered by different power platforms. For instance, the various tools and accessories may be powered by one of an AC source (e.g., 120 V, 60 Hz) via a power cord, or a DC source (e.g., 4-V battery pack, 12-V battery pack, 18-V battery pack, 28-V battery pack, 36-V battery pack, 40-V battery pack, etc.). The DC battery packs include a housing within which are one or more

battery cells, which may be lithium ion (“Li-ion”) cells, Nickel-Cadium (“Ni-Cad”) cells, or cells of another chemistry type.

[0036] Across the various tools and accessories having various power platforms, the interface software 200 includes the same universal core software 255. For instance, a hammer drill powered by a 28-V battery pack, an impact driver powered by an 18-V battery pack, a thermal imager powered by a 12-V battery pack, and screwdriver powered by a 4-V battery pack all include the same universal core software 255. However, the physical interface software 250 and tool component interface software 260 are customized for a particular tool or accessory. For example, if the tool 100 includes the wireless antenna 140, but not the I/O port 145, the physical interface software 250 may include only the software for translating communications between a first protocol used with the wireless antenna 140 and the universal protocol, but not the software for translating a second protocol used with the I/O port 145 and the universal protocol. Moreover, if the tool 100 includes the memory 160 and a particular temperature sensor as one of the sensors 170, the tool component interface software 260 can be customized to translate communications between the universal protocol and those particular tool components 265. Having the universal core software 255 consistent across various tools and accessories reduces development costs and simplifies communications across the various tools and accessories, troubleshooting, and data logging, and provides other benefits. Rather than development and maintaining software for each separate tool and accessory, a single universal core module is installed on the various tools and accessories, and (generally) the only customization is for the translation modules used to interface with the universal core.

[0037] Additionally, in some embodiments, generic physical interface modules 250 and/or tool component interface modules 260 are generated and installed. The generic modules have the ability to translate between various protocols and the universal protocol, such that the generic modules are able to be installed on various tool types, including those that have different protocol needs. For instance, a generic physical interface module 250 may include the ability to translate communications for a wireless protocol and an I/O port protocol, and this generic physical interface module 250 may be installed on (a) a first tool that uses the wireless protocol, but not the I/O port protocol, (b) a second tool that uses the I/O port protocol, but not the

wireless protocol, and (c) a third tool that uses both the wireless protocol and the I/O port protocol.

[0038] Fig. 6 illustrates a method 300 of implementing a universal core on a plurality of various tools and accessories. In step 305, the universal core module 255, including the set of rules for handling communications between tool components and external devices, is generated. Generation may include writing code in a first programming language and compiling the code into machine code executable by the controller 125. In some instances, the generation may include a combination of hardware and software design, or may include solely hardware design.

[0039] In step 310, the physical interface of a particular tool or accessory is determined, including the particular protocols used by the physical interface. For instance, with respect to the power tool 100 as shown in Fig. 2, in step 310, the physical interface 150 is determined to include the power block 130, the wireless antenna 140, and the I/O port 145. Additionally, the particular protocols of the power block 130, the wireless antenna 140, and the I/O port 145 are determined. In step 315, the physical interface module 250 is generated based on the determinations of step 310. For example, the physical interface module 250 is generated to enable the translation of the particular protocols used by the physical interface 150 to the universal core protocol, and the translation of the universal core protocol to the particular protocols used by the physical interface 150. The generation of the physical interface module 250 may include hardware design, software design, or a combination thereof, similar to the generation of the universal core module 255.

[0040] In step 320, the components of the particular tool or accessory, including the particular protocols used by the components, are determined. For instance, with respect to the power tool 100 as shown in Fig. 2, in step 320, the tool components 265 are determined to include the memory 160, the motor drive 165, the sensors 170, and the user output 175. Additionally, the particular protocols of the memory 160, the motor drive 165, the sensors 170, and the user output 175 are determined in step 320. In step 325, the tool component interface module 260 is generated based on the determinations of step 320. For example, the tool component interface module 260 is generated to enable the translation of the particular protocols used by the tool components 265 to the universal core protocol, and the translation of the

universal core protocol to the particular protocols used by the tool components 265. The generation of the tool component interface module 260 may include hardware design, software design, or a combination thereof, similar to the generation of the universal core module 255.

[0041] In step 330, the generated universal core module 255, physical interface module 250, and tool component interface module 260 are installed in the particular tool. For instance, in the case of the tool 100 and wherein the generated universal core module 255, physical interface module 250, and the tool component interface module 260 are software, installation includes loading the modules into the memory 160. For instance, as shown in Fig. 2, the memory 160 may include a power port and a read/write (r/w) port for reading and writing to the memory 160 (e.g., to install software). The r/w port of the memory 160 and the connection shown coupling the memory 160 and controller 125, although shown as separate ports in Fig. 2, may be implemented as a combined communication port. To install software initially into the memory 160, while power (e.g., 3.3 VDC) is applied to the power port, a write command and the software to be installed may be transmitted to the memory 160 over the r/w port. At the end of step 330, the memory 160 of the power tool 100 has stored thereon the interface software 200 as shown in Fig. 3.

[0042] Thereafter, the interface software 200 may be used to install the application software 205 and/or parameters 210 on the memory 160. For instance, an external device may communicate the application software 205 and/or parameters 210 to the universal core software 255 along with commands to initially load the application software 205 and/or parameters 210 onto the memory 160. The installation of the application software 205 and/or parameters 210 may occur at a different stage of manufacture of the tool 100 and/or at a different facility. For instance, the tool may be purchased without the application software 205 and/or parameters 210, and then the purchasing entity may install the application software 205 and parameters 210, which may be custom to the purchasing entity. Alternatively, in step 330, the application software 205 and/or parameters 210 may also be installed in the memory 160.

[0043] In the case of one or more of the universal core module 255, physical interface module 250, and tool component interface module 260 being implemented at least in part in hardware, the installation of such modules may also include mounting the various hardware

components to a PCB that resides within the particular tool. Mounting may include making conductive connections between the hardware and the other components of the particular tool.

[0044] If in step 335, an additional tool or accessory is determined to require installation of the interface module 200, the method 300 proceeds to step 340. In step 340, it is determined whether the additional tool or accessory has the same physical interface and tool components as a tool for which a physical interface module 250 and tool component interface module 260 was previously generated. If so, the method proceeds to step 330 and the previously generated universal core module 255, physical interface module 250, and tool component interface module 260 are installed. Additionally, although not explicitly shown in Fig. 6, if the additional tool or accessory has the same physical interface, but unique tool component protocols (or vice versa), relative to a particular tool previously handled according to steps 305-325, the previously generated physical interface module 25 (or tool component interface module 260, as the case may be) may be re-used and installed on the additional tool. The other of the tool component interface module 260 and the physical interface module 250 may then be generated and installed on the additional tool or accessory.

[0045] If in step 340, however, the additional tool or accessory is determined to not have had the physical interface module 250 and tool component interface module 260 previously generated, the method returns to step 310. The steps 310 through 330 are repeated for the additional tool or accessory to produce a customized physical interface module 250 and tool component interface module 260; however, since the universal core module 255 is consistent across various tools, the universal core module 255 does not need to be re-generated or customized for the additional tool or accessory. Rather, the same universal core module 255 generated for the first particular tool is re-used with the additional tool.

[0046] The steps in the method 300 may be executed in an order different than the order shown in Fig. 6. For instance, the physical interface modules and tool component interface modules may be generated for various tools (steps 305-325), and then installed at a later point. Additionally, the universal core module, physical interface module, and tool component interface module may be generated in a different order.

[0047] Fig. 7 illustrates a method 350 that implements an exemplary rule set of the universal core module 255. Method 350 is entered into upon an interrupt 355. The interrupt, which may be a software interrupt or a hardware interrupt, causes the controller 125 to jump to a particular line of code in the memory 160, such as a line in the interface software 200, that includes instructions for handling the interrupt (i.e., an interrupt handler). As illustrated, the interrupt 355 is assumed to be an interrupt generated due to receipt of a command from the external device 275 (see Figs. 5A-B). For instance, the external device 275 may send an interrupt signal, or a command without an interrupt signal, which causes the physical interface 150 to activate (e.g., pull up or pull down) an interrupt pin of the controller 125. Once the interrupt pin is activated, the controller 125 jumps to the particular line of code in the memory 160 to handle the interrupt. Alternatively, a software interrupt may be generated at regular intervals during execution of the application software 205, or a hardware interrupt may be generated at regular intervals through activation of the interrupt pin of the controller 125, at which point the method 350 is initiated.

[0048] In step 360, the universal core module 255 determines whether the command received was a read or write command. In the example, the command has been translated before step 360, either before or after the interrupt 355 occurred, depending on the particular interface module 200 design. Upon determining that a read command was received, the universal core module 255 proceeds to step 365 to determine whether the command is authorized. For instance, the read command may include an identifier and security element (e.g., a password) of the requesting external device 275 and the universal core module 255 may verify that the external device 275 is authorized to request the read. After verification, in step 370, the universal core module 255 outputs the read command to the tool component interface module 260 according to the universal protocol. The tool component interface module 260 then proceeds to translate the read command to the appropriate protocol and outputs the request to the appropriate tool component. In some instances, the controller 125 is coupled to the tool components 265 via one or more busses, and the read command includes an address that specifies a particular tool component, and a potentially a particular portion thereof, to be read. For instance, the motor drive 165, sensors 170, and user output 175 may be memory-mapped such that a read command with an address may specify any of the motor drive 165, a sensor of the sensors 170, a user output 175, or a section of the memory 160.

[0049] The addressed tool component provides data back to the controller 125 in response to the read command, and, in step 375, the universal core module 255 receives the data, which was translated to the universal protocol by the tool component interface module 260. In step 380, the universal core module 255 outputs the data obtained in step 375 to the physical interface module 250. Thereafter, the physical interface module 250 outputs the data to the external device 275 according to the protocol of the external device 275. After step 380, the universal core module 255 exits the interrupt handler (i.e., returns from the interrupt) in step 385.

[0050] Upon determining that a write command was received, the universal core module 255 proceeds to step 390 to determine whether the command is authorized. For instance, the write command may include an identifier and security element (e.g., a password) of the requesting external device 275 and the universal core module 255 may verify that the external device 275 is authorized to request the write. If the command cannot be authorized, the universal core module 255 proceeds to provide a denial message to the requesting device in step 395 and exit the interrupt handler in step 385. After verification, in step 400, the universal core software 255 determines whether the command necessitates that the tool 100 enter a safe mode. In a safe mode, various functions of the tool 100 are disabled, such as the ability for the motor 180 to rotate. For example, if the command directs the controller 125 to install, reinstall, or update the application software 205 and/or parameters 210 (i.e., a firmware install/update), the universal core software 255 determines in step 400 to enter the safe mode (step 405). Accordingly, in the safe mode, depression of the trigger of the tool 100 will not result in motor 180 rotating. Without a safe mode, operation of the tool 100 may be unpredictable, as the software for controlling the tool 100 may be temporarily corrupted while it is being overwritten. Although not illustrated in Fig. 7, a safe mode may also be entered when handling read commands or other commands.

[0051] After entering the safe mode, or after determining that the safe mode is not required, the universal core module 255 proceeds to step 410 to output the write command and data to the tool component interface module 260. The tool component interface module 260 then proceeds to translate the write command from the universal protocol to the appropriate protocol and outputs the request to the appropriate tool component. In some instances, the controller 125 is coupled to the tool components 265 via one or more busses, and the write command includes an

address that specifies a particular tool component or, potentially, a particular portion thereof, for which the write command is intended. In the case of a firmware update, the write command may specify an address of the memory 160 where the new firmware associated with the write command is to be stored. Such a firmware update may include several write commands to overwrite or add the firmware to the memory 160. Accordingly, step 410 may loop several times until the firmware has been stored in the memory 160. Similarly, the steps 370, 375, and 380 for handling a read command may be looped to complete a read operation, particular when larger sections of data are being requested. After completing the write command, if the safe mode was previously entered, the universal core module 255 exits the safe mode in step 415 and proceeds to step 385 to return from the interrupt. If the safe mode was not entered, the universal core module 255 may bypass step 415 and proceed to step 385 to return from the interrupt.

[0052] As noted above, the motor drive 165, sensors 170, and user output 175 may be memory-mapped such that a write command with an address may specify any of the motor drive 165, a sensor of the sensors 170, a user output 175, or a section of the memory 160. Accordingly, such a write command may be used to control a tool component 265 to operate in a particular way, rather than to store data or instructions for later retrieval. For instance, a particular write command or several write commands to the motor drive 165 may cause activation of the motor drive 165 such that the motor 180 rotates, or may cause deactivation of the motor drive 165 such that the motor 108 is disabled or placed in a sub-optimal performance mode. Additionally, a write command to a light of the user output 175 may cause the light to be illuminated or disabled. Alternatively, rather than memory-mapped components addressable over a bus, the controller 125 may have dedicated or semi-dedicated pins and connections to communicate with one or more of the various tool components 265.

[0053] The method 350 presupposes one of two commands, a read or write. However, in some embodiments, the universal core module 255 handles additional commands (“other commands”), including more complex commands, such as conditional instructions, multiple-read or multiple write commands, illumination commands (e.g., to enable or disable a light of the user output 175), audible commands (e.g., to cause a speaker of the user output 175 to generate sound), or motor drive commands (e.g., to cause the motor 180 to operate), etc. Similar to the

read and write commands, the other commands are handled by the universal core module 255, potentially in a safe mode.

[0054] Additionally, although not illustrated in Fig. 7, in instances where the interrupt 355 occurs and no command is received (e.g., after a periodic interrupt), the method 350 exits the interrupt handler in step 385 and returns to normal operation.

[0055] In some instances, internal tool components may generate an interrupt and provide one or more commands to the universal core module 255. For instance, at regularly timed intervals, or upon an error condition, the motor drive 165 or sensors 170 may generate a hardware interrupt and provide a command to the universal core module 255. In response, the controller 125 may write performance data for the sensors 170 or motor drive 165 to the data log 215, may initiate communication with the external device 275 to report conditions of the sensors 170 or motor drive 165, or take another action.

[0056] In some embodiments, a software loop executed by the controller 125 during normal operation of the tool 100 includes a non-interrupt-based, periodic check to determine whether any commands have been received from the physical interface 150 or tool components 265, rather than or in addition to an interrupt-based approach as described above. The code for performing the check resides, for instance, in a software loop of the interface software 200, of the application software 205, or a combination thereof.

[0057] Thus, the invention provides, among other things, a universal protocol for use with power tools and other devices. Various features and advantages of the invention are set forth in the following claims.

CLAIMS

What is claimed is:

1. A method of implementing a universal protocol on a plurality of tools, the plurality of tools including a variety of power tool types, the method comprising:
 - generating an interfacing module including
 - universal core module for implementing the universal protocol, the universal core module being generic to the plurality of tools and including a set of rules enabling interpretation and handling of communications that are in accordance with the universal protocol, and
 - an interface module particular to each of the power tool types, the interface module configured
 - to translate communications on route to the universal core module to the universal protocol and
 - to translate communications output by the universal core module from the universal protocol to another protocol; and
 - installing the interfacing module on a memory of each of the plurality of tools.
2. The method of claim 1, wherein the interface module includes:
 - physical interface software enabling translation of communications between the universal core module and external devices; and
 - tool component interface software enabling translation of communications between the universal core module and internal components of the plurality of tools.
3. The method of claim 2, wherein generating the physical interface software comprises determining physical interface components and associated protocols for each of the tool types.
4. The method of claim 1, further comprising:
 - providing, at a physical interface of a tool of the plurality of tools, commands according to the another protocol;

translating, by the interface module of the tool, the commands from the another protocol to the universal protocol to form translated commands;

providing the translated commands to the universal core module; and

executing, by the universal core module, the translated commands.

5. The method of claim 1, further comprising:

generating application software and parameters particular to each tool;

using the interfacing module of each tool to install on each tool the application software and parameters particular to that tool.

6. The method of claim 5, further comprising:

generating an update for at least one of the application software and parameters for at least one tool of the plurality of tools; and

installing the update on the at least one tool of the plurality of tools using the interfacing module.

7. The method of claim 1, further comprising

capturing and storing performance data in the memory of each tool of the plurality of tools; and

reading the performance data from the memory of each tool using an external device communicating with the interfacing module via a physical interface of each tool.

8. The method of claim 1, further comprising:

receiving a write command by the universal core module of a tool of the plurality of tools, the write command specifying a write to the memory of the tool; and

entering a safe mode after receipt of the write command and before commencing the write to the memory, wherein, in the safe mode, a motor of the tool is disabled until the write to the memory of the tool is completed.

9. The method of claim 1, wherein the variety of power tool types includes at least two of: a drill, a circular saw, a jig saw, a band saw, a reciprocating saw, a screw driver, a grinder, an impact wrench, a rotary hammer, and an impact driver.

10. A universal protocol system comprising:
a plurality of tools including a variety of power tool types, each tool including a memory;
interfacing module installed on the memory of each tool, the interfacing module
including

a universal core module for implementing a universal protocol, the universal core module being generic to the plurality of tools and including a set of rules enabling interpretation and handling of communications that are in accordance with the universal protocol, and

an interface module particular to each of the power tool types, the interface module configured to translate communications on route to the universal core module to the universal protocol and to translate communications output by the universal core module from the universal protocol to another protocol.

11. The universal protocol system of claim 10, wherein the interface module includes:
physical interface software enabling translation of communications between the universal core module and external devices; and

tool component interface software enabling translation of communications between the universal core module and internal components of the plurality of tools.

12. The universal protocol system of claim 11, wherein the physical interface software is generated based on determining physical interface components and associated protocols for each of the tools types.

13. The universal protocol system of claim 10, wherein each tool of the plurality of tools includes a physical interface that is communicatively coupled to an external device and that is communicatively coupled to the universal core module via a physical interface module, the

physical interface module translating commands from the external device to the universal protocol to form translated commands.

14. The universal protocol system of claim 13, wherein the universal core module of at least one tool of the plurality of tools receives and executes the translated commands from the physical interface module of the at least one tool.

15. The universal protocol system of claim 14, wherein the translated commands include a write of a software application and parameters to the memory of the at least one tool.

16. The universal protocol system of claim 14, further comprising:
sensors in each tool that generate performance data; and
performance data stored in the memory of each tool,
wherein the translated commands include a read of the performance data stored in the memory of the at least one tool.

17. The universal protocol system of claim 10, further comprising application software and parameters particular to each tool, wherein the application software and parameters particular to each tool are installed on the memory of each tool, respectively, using the interfacing module of each tool, respectively.

18. The universal protocol system of claim 17, further comprising an update for at least one of the application software and parameters for at least one tool of the plurality of tools, wherein the update is installed on the memory of the at least one tool using the interfacing module of the at least one tool.

19. The universal protocol system of claim 10, further comprising a safe mode on each tool of the plurality of tools, the safe mode disabling operation of a motor of an associated tool until execution of a write to the memory of the associated tool is completed.

20. The universal protocol system of claim 10, wherein the variety of power tool types includes at least two of: a drill, a circular saw, a jig saw, a band saw, a reciprocating saw, a screw driver, a grinder, an impact wrench, a rotary hammer, and an impact driver.

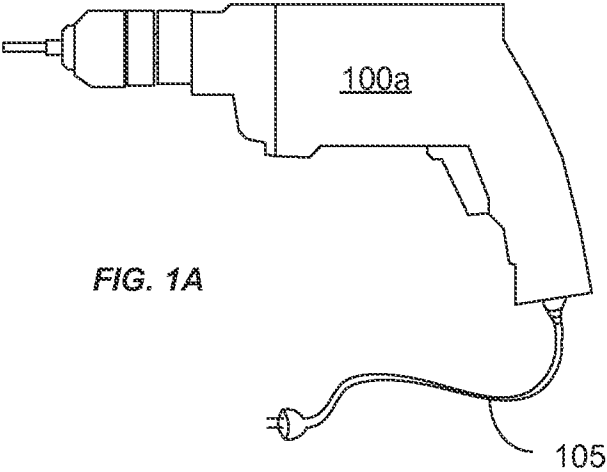


FIG. 1A

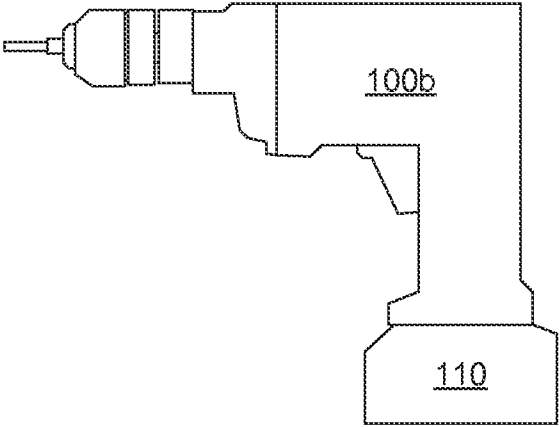


FIG. 1B

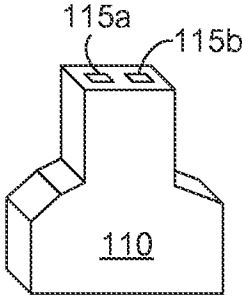


FIG. 1C

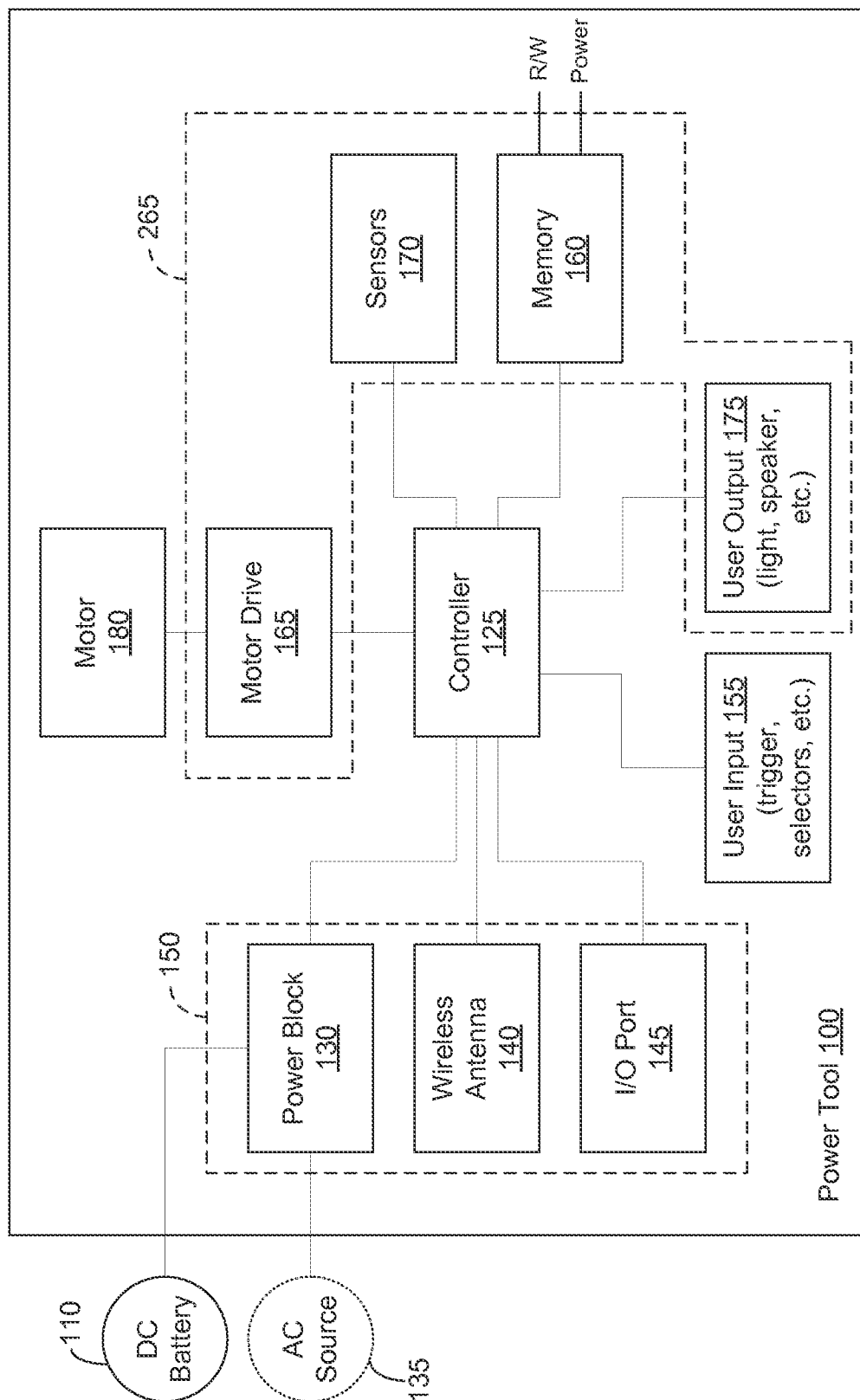


FIG. 2

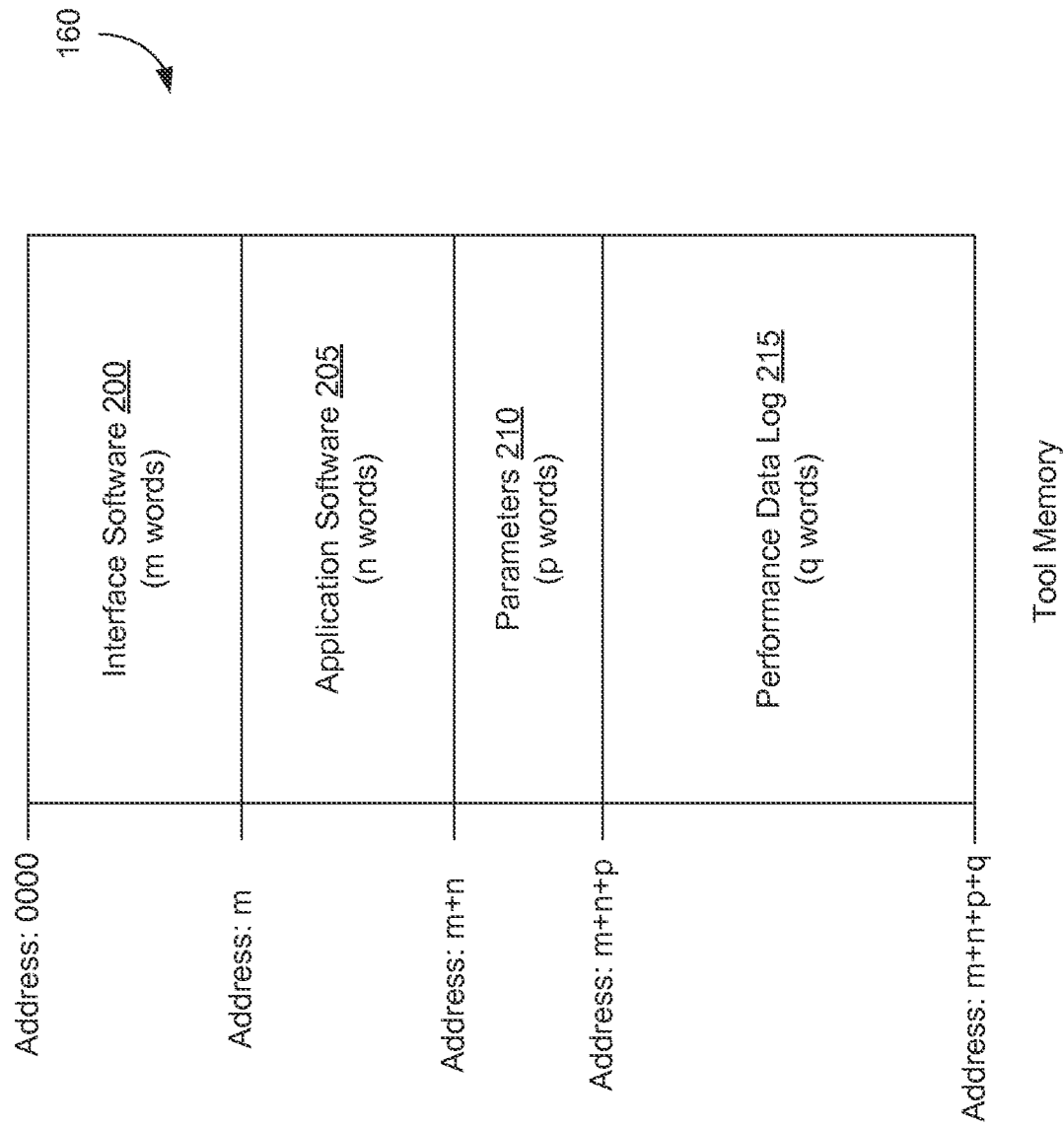


FIG. 3

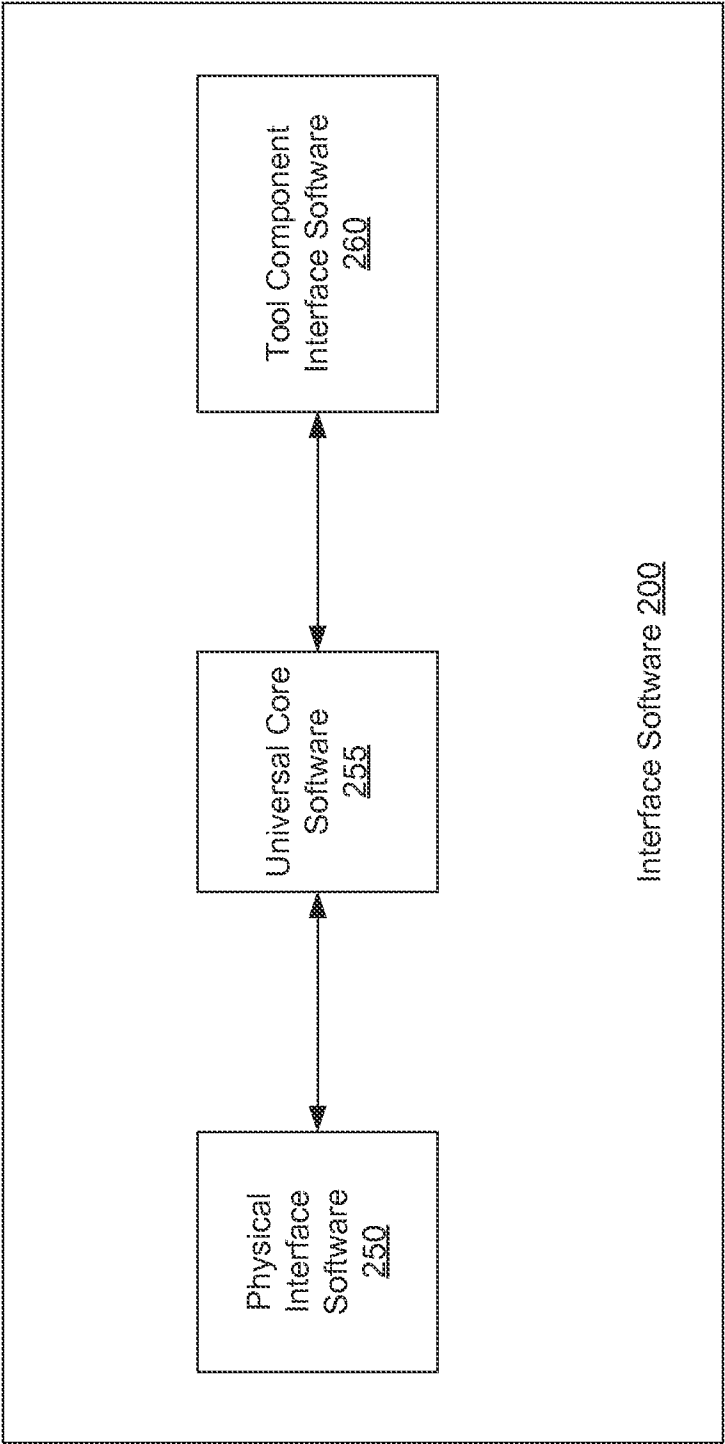


FIG. 4

5/8

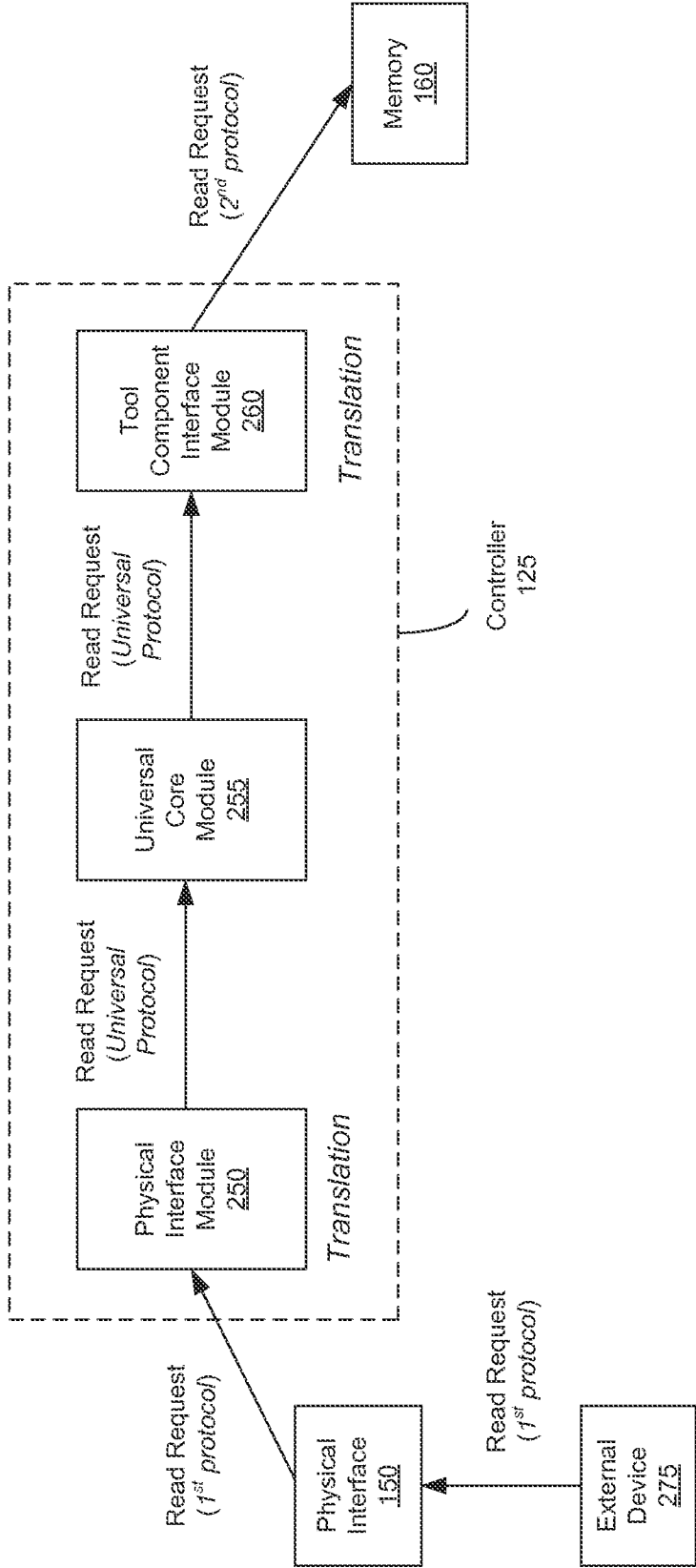


FIG. 5A

6/8

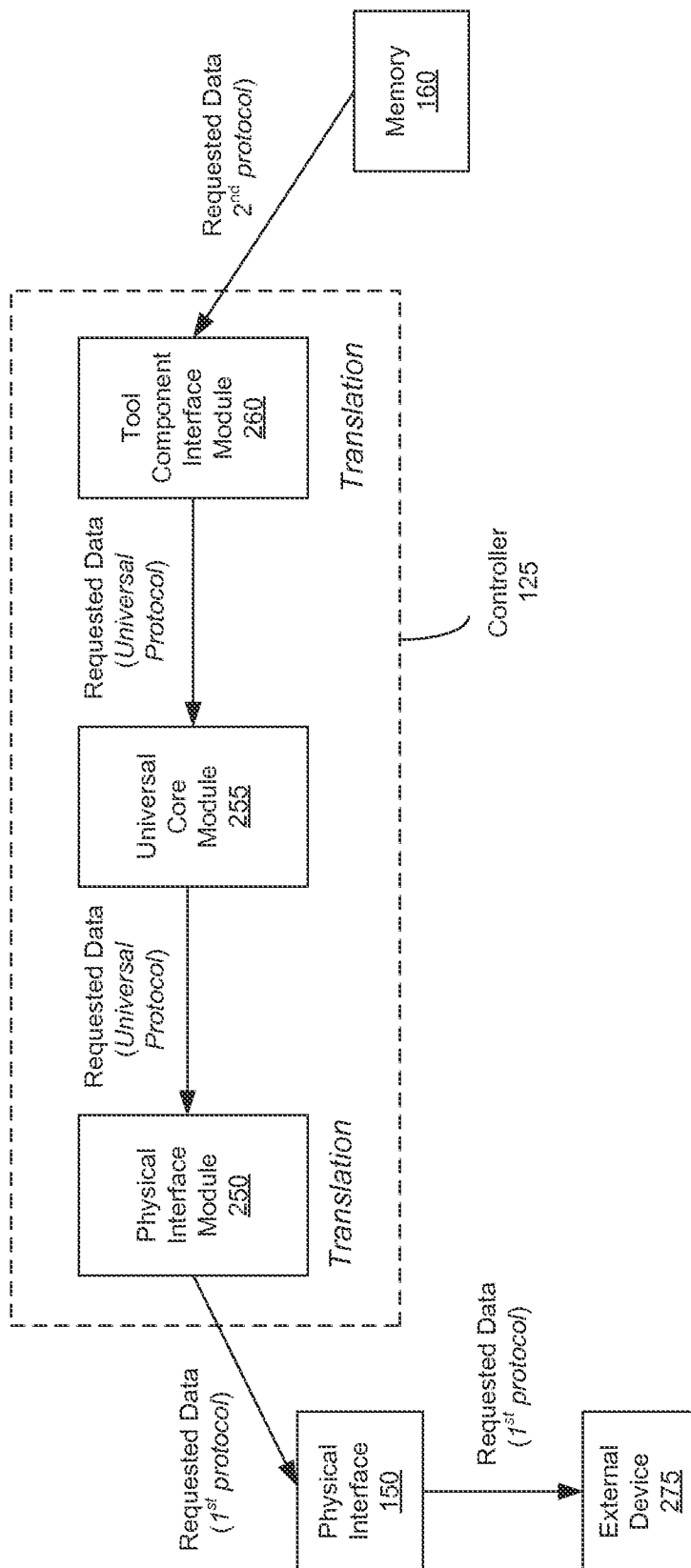


FIG. 5B

7/8

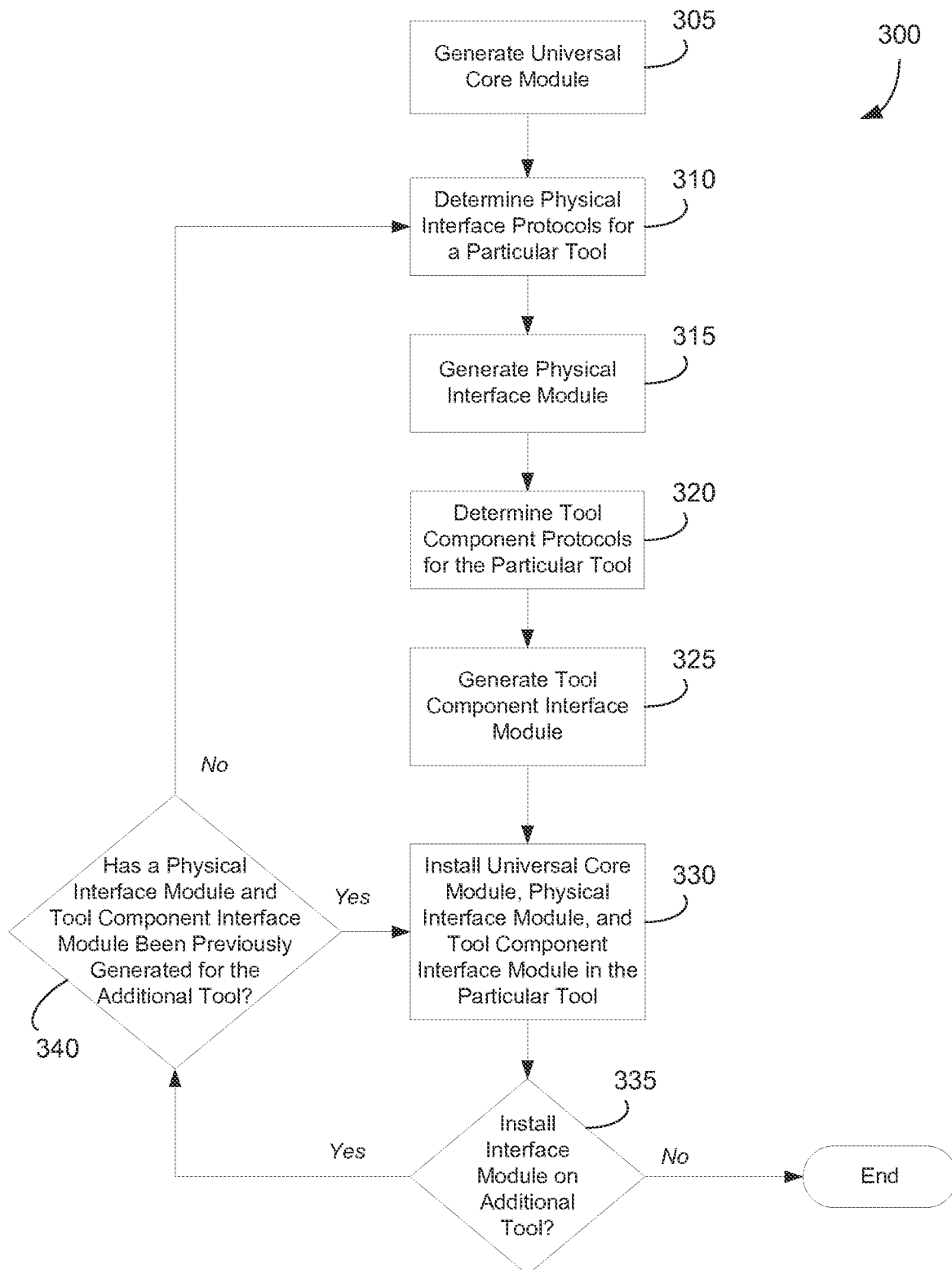


FIG. 6

