



(51) International Patent Classification:

G01R 31/3181 (2006.01) G06F 11/22 (2006.01)
G01R 31/3177 (2006.01)

(21) International Application Number:

PCT/US2016/066736

(22) International Filing Date:

14 December 2016 (14.12.2016)

(25) Filing Language:

English

(26) Publication Language:

English

(30) Priority Data:

14/975,685 18 December 2015 (18.12.2015) US

(71) Applicant: INTEL CORPORATION [US/US]; 2200 Mission College Blvd., Santa Clara, California 95054 (US).

(72) Inventors: MENON, Sankaran; 110 Whitley Drive, Austin, Texas 78738 (US). SMITH, Bradley; 418 NE 154th Street, Vancouver, Washington 98685 (US). HUANG, Jinshi; 183 Indian Hill Place, Fremont, California 94539 (US). KUEHNIS, Rolf; 3027 NW Spencer Street, Portland, Oregon 97229 (US).

(74) Agent: PATTY, Herbert; Suite 407, 1625 The Alameda, San Jose, California 95126 (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, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, 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, SA, 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, ST, 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).

Published:

- with international search report (Art. 21(3))
- before the expiration of the time limit for amending the claims and to be republished in the event of receipt of amendments (Rule 48.2(h))

(54) Title: INTERFACES FOR WIRELESS DEBUGGING

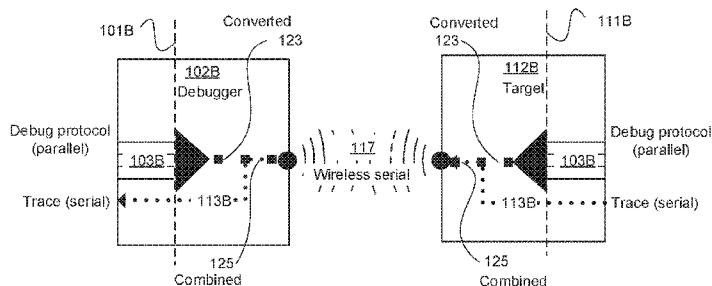


FIG. 1B

(57) Abstract: Existing multi-wire debugging protocols, such as 4-wire JTAG, 2-wire cJTAG, or ARM SWD, are run through a serial wireless link by providing the debugger and the target device with hardware interfaces that include UARTs and conversion bridges. The debugger interface serializes outgoing control signals and de-serializes returning data. The target interface de-serializes incoming control signals and serializes outgoing data. The actions of the interfaces are transparent to the inner workings of the devices, allowing re-use of existing debugging software. Compression, signal combining, and other optional enhancements increase debugging speed and flexibility while wirelessly accessing target devices that may be too small, too difficult to reach, or too seal-dependent for a wired connection.



INTERFACES FOR WIRELESS DEBUGGING

invented by

SANKARAN M. MENON

BRADLEY H. SMITH

JINSHI HUANG

ROLF H. KUEHNIS

FIELD

[0001] Related fields include debugging, particularly debugging of platforms, systems and devices over a wireless link.

BACKGROUND

[0002] Debugging conventional systems which incorporate system-on-chip and integrated circuits typically require them to be connected to a debugger device using one or more cables. The form-factors of these systems and platforms are becoming smaller for each generation. Likewise, the number of wired-ports or pins on these systems have drastically reduced such that many of these devices do not have wired ports or pins for debugging purposes. Due to this trend, a new debugging solution is needed. The present disclosure addresses this need.

BRIEF DESCRIPTION OF DRAWINGS

[0003] To facilitate understanding, identical reference numerals have been used, where possible, to designate identical elements that are common to the drawings. The present disclosure may readily be understood by considering the following detailed description with the accompanying drawings which are not necessarily drawn to scale, in which:

[0004] FIGs. 1A and 1B schematically illustrate wired and wireless debugging arrangements.

[0005] FIGs. 2A-2E illustrate examples of target devices that could benefit from wireless debugging.

[0006] FIG. 3 is a flowchart for a debugger implementing wireless debugging.

[0007] FIG. 4 is a flowchart for a target device undergoing wireless debugging.

[0008] FIG. 5 is a block diagram of a debugger and target configured for wireless debugging.

[0009] FIG. 6 is a block diagram of inputs and outputs in a conversion bridge.

[0010] FIG. 7 is a state diagram of a UART JTAG tunneling protocol.

DETAILED DESCRIPTION

[0011] The description of the different advantageous embodiments has been presented for purposes of illustration, and is not intended to be exhaustive or limited to the embodiments in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. Further, different advantageous embodiments may provide different advantages as compared to other advantageous embodiments. The embodiment or embodiments selected are chosen and described in order to best explain the principles of the embodiments, the practical application, and to enable others of ordinary skill in the art to understand the disclosure for various embodiments with various modifications as are suited to the particular use contemplated.

[0012] In the following description, numerous specific details are set forth, such as examples of specific types of processors and system configurations, specific hardware structures, specific architectural and micro architectural details, specific register configurations, specific instruction types, specific system components, specific measurements/heights, specific processor pipeline stages and operation etc. in order to provide a thorough understanding of the present disclosure. It will be apparent, however, to one skilled in the art that these specific details need not be employed to practice the present disclosure.

[0013] In other instances, well known components or methods, such as specific and alternative processor architectures, specific logic circuits/code for described algorithms, specific firmware code, specific interconnect operation, specific logic configurations, specific manufacturing techniques and materials, specific compiler implementations, specific expression of algorithms in code, specific power down and gating techniques/logic and other specific operational details of computer system haven't been described in detail in order to avoid unnecessarily obscuring the present disclosure.

[0014] In the following description and claims, the terms “coupled” and “connected,” along with their derivatives, may be used. It should be understood that these terms are not intended as synonyms for each other. Rather, in particular embodiments, “connected” may be used to indicate that two or more elements are in direct physical or electrical contact with each other. “Coupled” may mean that two or more elements are in direct physical or electrical contact. However, “coupled” may also mean that two or more elements are not in direct contact with each other, but yet still co-operate or interact with each other.

[0015] FIG. 1A illustrates a conventional wired debugging configuration. By way of non-limiting example, a debugger device 102A establishes a multi-wire connection with a target device (device under test) 112A. Debugger 102A formulates control signals in a multi-wire debug protocol 103A and sends them over the multi-wire connection. For example, debug protocol 103A may be Joint Test Action Group (JTAG), which uses 4 wires, or compact Joint Test Action Group (cJTAG), which uses 2 wires.

[0016] Target 112A understands and reacts to debug protocol 103A commands as sent over the multi-wire connection. For example, target 112A may respond with readouts of the values presently in the requested registers. Short responses, such as a few bytes, may be sent back to debugger 102A over the multi-wire connection. However, lengthy responses such as a data dump from a large memory array may be sent back to the debugger on a separate channel as a serial bit stream such as trace signal 113A.

[0017] FIG. 1B illustrates a wireless debugging configuration. Debugger 102B and target 112B, no longer connected by a multi-wire connection, cannot send or receive debug protocol 103A signals as generated. Instead, Bluetooth® (which is intended to include Bluetooth Low-Energy, known as BTLE) and similar wireless technologies send and receive serial bit streams such as wireless serial signal 117. Providing a separate wireless link to replace each of the parallel wires

in FIG. 1A may add to production cost and be cumbersome to pair correctly and protect from crosstalk. Alternatively, serial control protocols exist, such as those formerly sent over wired RS-232 connections. However, their speed, versatility, and pervasiveness of adoption may fall short of the achievements of multi-wire debugging protocols. In addition, in many scenarios where the devices are of small form factor or implanted inside a patient's body, the debug targets may not be physically accessible by wired connectors. A wireless solution compatible with existing multi-wire debugging protocols may be desirable. A way to debug target devices through asynchronous interfaces as well as the usual serial-synchronous interfaces may also be beneficial.

[0018] Debugger 102B generates commands in multi-wire debug protocol 103B, but then converts them to serial-converted signal 123. Serial-converted signal 123 is transmitted to target 112B as wireless serial signal 117. Target 112B receives wireless serial signal 117 and recovers serial-converted signal 123. Then, the conversion process is reversed to yield the original debug protocol 103B commands and distribute them among a plurality of compatible wires.

[0019] From there, individual registers, groups of registers, and memory arrays on target 112B are read. Some of the values are carried back along the wires distributing debug protocol 103B. Other values coming from locations referred to as "trace sources" (memory arrays and the like) are routed along a data pipe as a serial trace signal 113B. Trace signal 113B is combined (e.g., multiplexed) with the returning converted serial signal 123 to form combined serial signal 125. Combined serial signal 125 is transmitted back to debugger 102B as wireless serial signal 117.

[0020] Debugger 102B receives returning wireless serial signal 117 and recovers combined serial signal 125. Combined serial signal 125 is separated (e.g., de-multiplexed) into serial-converted signal 123 and trace signal 113B. Finally, serial-converted signal 123 is reverse-converted to debug protocol 103B data, which is distributed among a plurality of compatible wires.

[0021] The section of debugger 102B to the left of line 101B has the same types of inputs and outputs as wired debugger 102A. Moreover, the part of target 112B to the right of line 111B has the same inputs and outputs as wired target 112A. In some embodiments, the interfaces between line 101B and line 111B may be added to existing debuggers and targets originally designed for wired use which may be faster and less expensive to implement than a fully redesigned SoC or IC. Also, the new interface may be made transparent to software that presently runs on the processors of debugger 102 and target 122 so that much, or even all, existing software may be re-used to reduce development cost.

[0022] FIGs. 2A-2E illustrate examples of target devices that could benefit from wireless debugging. In general, an increasing number of miniature and embedded devices are difficult to reach with wired connectors. In-situ diagnostics and debugging in the field could be particularly aided by wireless processes.

[0023] In FIG. 2A, chip 202A is built into architectural material such as frame 204 of a smart-glass window for a building 206. A wired port may function as an unwanted collector of moisture or dust. Large household appliances with smart or connected features may also have difficult-to-reach electronics.

[0024] In FIG. 2B, smart glasses 206 have chip 202B in a very small control/power module that would have to be larger and heavier to accommodate a wired connection.

[0025] In FIG. 2C, chip 202C is in a sealed module 212. Where sealing is function-critical, such as in liquid environment 214, a wired connector that has to remain accessible is a likely point of failure.

[0026] In FIG. 2D, chip 202D is part of a medical implant. Devices positioned inside a body 216 may be safer and easier to live with if they do not require repeated through-the-skin access.

[0027] In FIG. 2E, chip 202E is built into drone 218. With wireless debugging, chip 202E could potentially be debugged from a remote debugger while drone 218 is in flight performing a task or landed in a difficult-to-access location. Some drones also may be too small for a conventional wired connector or, like chip 202E in FIG. 2C, may need to be sealed for reliability or safety.

[0028] Wireless debugging may also be done at a distance; for instance, by transmitting the debugging signals first over a cellular phone network from the support office to the customer site, then from the customer's phone to the device over Bluetooth, BTLE, WiFi, or some other wireless technology common to the device and the customer's phone. In this case, the debugger may be running at the support office and the serial signals will be further transmitted, as packetized data, from the customer's phone to the support office before converting back to the debug protocol and trace. Desirable characteristics of a suitable wireless technology for wireless debugging include (1) staying on indefinitely after being activated, (2) low power consumption or overhead in case of a target device or debugger with a low battery, and (3) activation as soon as possible after the target device is powered to detect boot-disabling bugs.

[0029] FIG. 3 is a flowchart for a debugger implementing wireless debugging. At block 302, the debugger has started and scans for a target's wireless presence. If a target is not detected at decision 310, the scanning continues at block 311 until a target is detected. When a target is detected, the debugger pairs with the target or connects with it through a common network at block 312 so communication can begin. For example, some current WiFi links may depend on a common network, and in some situations such as enterprise IT or a network provider servicing its own equipment the common network is already in place. As another example, some current Bluetooth links are network-independent and use ad-hoc pairing to create connections.

[0030] At block 313, link training occurs if needed to negotiate the data rate for communication between the debugger and target. If the debugger and target communicate at a common fixed

rate, link training is not needed. However, a variable data rate may advantageously enable debugging and testing at a lower data rate than normal high-speed operation. At block 314, the debugger and target establish communication at a mutually acceptable data rate, whether fixed or trained.

[0031] At block 316, the debugger generates or retrieves debug-protocol control signals for the desired debugging process. The process may be retrieved from storage or defined by real-time input from an operator. The debug protocol may be a multi-wire protocol such as JTAG or cJTAG. At block 318, the control signals are converted to a wireless protocol (e.g., a serial protocol such as Bluetooth/BTLE). Optionally, the control signals may be compressed at block 317 after, before, or concurrently with the serial conversion of block 318. At block 322, the wireless control signals are transmitted from the debugger to the target.

[0032] The next task for the debugger is to receive the data being returned from the target at block 324. Serial trace data from data-intensive sub-processes such as memory dumps may arrive through the same port as serialized multi-wire data returning along the control-signal path, or through a different port. If the data is compressed, the debugger may decompress it at block 325. At decision 330, incoming trace data may be directly compared with stored known-good or expected values at block 332, and non-trace data may be converted back to debug protocol at block 331 before analysis. When enough data has been evaluated, the debugger may output a diagnosis of the target at block 334.

[0033] FIG. 4 is a flowchart for a target device undergoing wireless debugging.

[0034] Blocks 402-12 provide examples of stages in the target's boot or start-up process. Power comes on at block 402, the basic input/output system (BIOS) loads at block 404, a boot loader is activated at block 406, the operating system (OS) loads at block 408, and applications load at block 412.

[0035] At block 422, activation of the wireless transmitter/receiver that will communicate with the debugger can occur between or during any of these stages to allow the debugging of later processes. However, because errors can occur at any point in start-up, it may be preferable to activate the wireless transmitter/receiver as early as possible to enable debugging of a maximum range of operational stages. For example, a dedicated embedded controller may activate immediately upon power-up to start the wireless transmitter/receiver. Some systems already activate a wireless transmitter/receiver very early to take advantage of wireless battery charging even if the battery is largely drained; this capability can be leveraged for low-level and low-power debugging.

[0036] At block 424, the wireless transmitter/receiver senses the wireless presence of the debugger. At block 426, the wireless transmitter/receiver connects with the debugger by ad-hoc pairing or over a shared network. Pairing is a more versatile approach, but there are situations where a debugger and its targets can almost always access a shared network. At block 428, the target wireless transmitter / receiver receives serialized wireless control signals from the debugger.

[0037] At block 432, the target converts the serialized wireless control signals into debug-protocol control signals. In some embodiments, the target debug protocol may be JTAG, cJTAG, Universal Asynchronous Receiver/Transmitter (UART), Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I2C), or Serial Wire Debug (SWD). At block 434, the target reacts to the debug-protocol control signals, e.g., by reading register values to transmit back to the debugger for evaluation. At block 436, the register values are converted to serial wireless signal(s) and as such they are transmitted to debugger at block 438. At decision 440, if the process finishes or if the target hangs or freezes before the process is finished, the process stops at block 442. If not, the debugging process continues at block 441 as the target proceeds to the next stage of its start-up cycle.

[0038] FIG. 5 is a block diagram of a debugger and target configured for wireless debugging.

Individual blocks in the diagram may represent individual components, groups of components, or parts of components in various alternative embodiments.

[0039] In debugger 502, debugger controller 506 may receive instructions from an operator through debugger input/output interface 508. Alternatively, debugger controller 506 may automatically load a program from storage 504. The instructions or program may use a multi-wire protocol such as JTAG or cJTAG. Other suitable protocols may include Device Communications Interface (DCI), Virtualization of Internal Signals Architecture (VISA), Mobile Industry Processor Interface (MIPI) STP or TWP protocols, or other serial or multi-wire protocols.

[0040] Debugger controller 506 sends the debug-protocol control signals to debugger conversion bridge 512 to be serialized in preparation for wireless transmission. The resulting serial-converted signal is routed through a debugger UART 514 that controls debugger 502's serial communication with target 522. Debugger de-multiplexer 517 may be present if the incoming serial-converted data stream and the incoming trace data stream 513 share a single port and would otherwise become so intermixed as to obfuscate the trace-source registers on the target that produced each value or set of values. For the outgoing control signals, debugger de-multiplexer 517 would act as a pass-through if there is no other outgoing signal.

[0041] In some embodiments, debugger compressor/decompressor 515 may be present, compressing outgoing signals and decompressing compressed incoming signals. In models and experiments, gains in debugging speed are in direct proportion to the amount of compression, at least up to 50X compression. This indicates that, at least in this range, the time taken by the extra operations of compressing the outgoing control signal at debugger 502, decompressing it when it arrives at target 522, compressing the returning data at target 522, and decompressing it when it

arrives at debugger 502 is insignificant compared to the time saved by sending a smaller number of bits across the wireless link.

[0042] Both the debugger and the target preferably are configured to use the same algorithm for compression and decompression. For example, LZ4, an open-source compression algorithm usually used in databases or for increasing hard-drive space or bandwidth without changing hardware, may be re-purposed for use in wireless debugging. Other candidate algorithms, like LZ4, are not computationally intensive; although firmware-based, their overhead is very small. For example, the entire algorithm may occupy 1KB of read-only memory (ROM) or less.

[0043] Compressing all the data exchanged over the link in both directions has been shown to produce a net gain of 10x-50X in debugging speed. This may make up for some of the speed disadvantages of some of the ad-hoc-pairing wireless technologies such as Bluetooth, which may be 100-1000x slower than technologies requiring a common network such as WiFi.

[0044] The serialized control signals are converted to wireless serial signals transmitted by debugger wireless transmitter / receiver 516 and received by target wireless transmitter / receiver 526. Target wireless transmitter / receiver 526 may be located on target wireless communication block 524 along with other blocks 528.1-528.N. For example, target wireless communication block 524 may include transmitters and receivers for Bluetooth (BT), Wireless Fidelity (WiFi), Global Positioning System (GPS), Near Field Communication (NFC), other RF signals, and/or other free-space signals such as infrared.

[0045] Target embedded controller 532 may be provided in some embodiments to activate target wireless transmitter / receiver 526 independently of target processor 538. If target embedded controller 532 activates target wireless transmitter / receiver 526 very early (or even first) as the target starts up, giving debugger 502 access to register errors that may occur early in the boot

cycle so that target 522 fails before target processor 538 can load the operating system or applications.

[0046] After reception at the target, the control signals revert to their wired serialized form. Target compressor / decompressor 525 may be present to accommodate compressed incoming control signals and to compress outgoing data streams. Target UART 534 may be located either upstream or downstream of first target multiplexer / de-multiplexer 536, as part of target wireless communication block 524 or of target processor 538. In some embodiments, target wireless communication block 524 may be integrated with target processor 538. Alternatively, target wireless communication block 524 and target processor 538 may be on different parts of the same chip, or on two different chips.

[0047] A first target multiplexer / de-multiplexer 536 is a pass-through for received control signals. Target conversion bridge 552 recovers the control signals in their original multi-wire form. A second target multiplexer / de-multiplexer 556 mixes the recovered control signals with other signal channels going into target processor core 558, allowing the recovered control signals to bypass any adapters 554, if present, that convert input and output of target processor core 558 to different protocols. The recovered control signals cause target processor core 558 to read its own internal registers, if any, and then to read registers in other parts 542 of target processor 538 and other parts 562 of the target SoC or IC 522.

[0048] Data from smaller groups of registers, such as status registers, may be sent in a reverse direction along the path of the control signals. Larger groups of registers, such as memory arrays, act as trace sources 546 and 556 and output trace data streams 343 and 353, which may travel in one or more paths that, at least in places, do not coincide with the control-signal path. The returning data paths meet at first target multiplexer / de-multiplexer 536, which mixes them into a single returning data stream.

[0049] The combined data stream goes through target UART 534 in embodiments where target UART 534 is closer than first target multiplexer / de-multiplexer 536 to target wireless transmitter / receiver 526. If target compressor / decompressor 525 is present, it compresses the returning combined data stream. The returning combined data stream is then converted to a wireless data stream and transmitted from target wireless transmitter / receiver 526 back to debugger wireless transmitter / receiver 516 to be analyzed by debugger 502.

[0050] Alternatively, the constituent returning data streams may continue on separate paths and be transmitted and received through separate ports.

[0051] The returning data received at debugger wireless transmitter / receiver 516 is decompressed, if needed, by debugger compressor/decompressor 515, if present. Combined returning data streams may be separated, if needed, by debugger de-multiplexer 517, if present. The trace signal 513, which was serial at its sources, goes directly to debugger controller 506 while the signal returning along the control-signal path goes through debugger UART 514 to be converted to its original form by debugger conversion bridge 512 before being routed to debugger controller 506. Debugger controller 506 may analyze the data and provide the result to the operator through debugger input/output interface 508. Additionally or alternatively, debugger controller 506 may store the data in storage 504 for later reference.

[0052] FIG. 6 is a block diagram of inputs and outputs in a conversion bridge. This diagram could apply to the debugger or the target. Wireless transmitter / receiver 626 provides serial data in (EDI) 623 to conversion bridge 652. In response, conversion bridge 652 provides clock-edge (CK) 659 with converted data (DI(N-1:0)) 653 to processor 638. Each EDI symbol sent is converted to a clock event and an associated data set 1 or more bits wide. For each data set sent from conversion bridge 652 to processor 638, processor 638 returns a data set with the same

number of bits. This logic enables an ordinary "data pipe" link to control virtually any type of debug hardware.

[0053] When target processor 638 outputs data (DO(N-1:0))655 to conversion bridge 652, conversion bridge 652 outputs corresponding serial data (EDO) 625. Additionally processor 638 provides an indicator of its state (TS) 651 to conversion bridge 652, which uses selected TS values to control the output-enable function (OE) 621 on wireless transmitter / receiver 626. In some embodiments, wireless transmitter / receiver 626 is only allowed to transmit data when the processor is in certain states.

[0054] FIG. 7 is a state diagram of a UART JTAG tunneling protocol. This non-limiting example describes one way to use a hardware state machine to establish and manage a wireless link data pipe between a debugger and a target. Attention to a process for establishing the link may be important; if the debugger begins sending commands before the target is ready to receive them, the commands may be ignored or data may be lost.

[0055] The states, other than Reset, may be divided into three groups. Training state group 710 includes the sequence for establishing communication between the target and the debugger at a mutually compatible frequency. Connecting state group 720 includes the exchange of debugging information over the established communication link. Releasing or "un-training" section 730 includes breaking the connection, releasing the link, and recovering the link if needed.

[0056] The debugger and the target may need to be ready to communicate at the same frequency. In older RS232 links, the problem was addressed by operating both devices at a single fixed frequency. Alternatively, if the communicated data from the debugger to the target (or vice versa) were to be accompanied by a clock signal, the clock might be ramped to any frequency. However, many current debuggers and targets can operate at two or more frequencies and many widely used wireless technologies do not include passing a clock signal from one device to the

other. Link training is one approach to negotiating a frequency for the debugging link. One advantage of not using a fixed frequency is that debugging and other kinds of testing can be done at a lower frequency (i.e., slower speed) than normal operation.

[0057] After Reset 701, the system is in Untrained state 702. From Untrained state 702, receiving the "EDI=Train" (Incoming Request for Training) signal 703 may trigger a change to Locked state 704, in which the debugger and target have locked onto a mutually selected frequency. In some embodiments, link quality testing and/or handshake processes may also be part of this stage.

[0058] From Locked state 704, receiving the "TS=Trained" (Target System Trained) signal 705 may trigger a change to Trained state 706, in which the signal integrity is established via a loopback stage. Alternatively, from Locked state 704, receiving the "EDI=Break" (Incoming Request for Disconnection) signal 713 may trigger a change back to Untrained state 702 to restart the training process.

[0059] From Trained state 706, receiving the "EDI=Connect" (Incoming Request for Connection) signal 707 may trigger a change to Connecting state 708, in which the debugger and target exchange preliminary messages to begin the debugging process. Alternatively, from Trained state 706, receiving the "EDI=Break" signal 713 may trigger a change back to Untrained state 702 to restart the training process.

[0060] From Connecting state 708, receiving the "TS=Active" (Target System Active) signal 709 may trigger a change to Connected state 712 in which quanta of information are exchanged between the UART and JTAG-responsive parts of the target, each including a clock-edge and a predetermined number of bits. Alternatively, from Connecting state 708, receiving the "EDI=Break" signal 713 may trigger a change to Break state 714, in which the link is disconnected. Alternatively, from Connecting state 708, receiving the "TS=!Active" (Target

System Not Active") signal 715 may trigger a change back to Untrained state 702 to restart the training process.

[0061] From Connected state 712, receiving the "EDI=Break" signal 713 may trigger a change to Break state 714, in which the link is disconnected. Alternatively, from Connected state 712, receiving the "TS=!Active" signal 715 may trigger a change back to Untrained state 702 to restart the training process.

[0062] From Break state 714, receiving the "TS=Active" signal 709 may trigger a change to Recovery state 716, from which the connection may be resumed. Alternatively, from Break state 714, receiving the "TS=!Active" signal 715 may trigger a change back to Untrained state 702 to restart the training process.

[0063] From Recovery state 716, receiving either the "Timeout" (maximum allowed recovery time elapsed) signal 717 or the "TS=!Active" signal 715 may trigger a change back to Untrained state 702 to restart the training process.

[0064] Numerous variations on the wireless-debugging concept may fall within the protected scope. For example, different types of wireless link technologies may be used such as Bluetooth(TM), Bluetooth Low Energy™, WiFi™, NFC, cell-phone networks, or combinations for multiple-link wireless debugging. Diagnostic or debugging signals may include trace protocols JTAG, cJTAG, DCI, serialized VISA (Virtualization of Internal Signals Architecture), or serialized MIPI protocols such as STP or TWP. For compressed wireless debugging, compression algorithms may include LZ4 or Debug Port Profile (DPP).

[0065] The preceding Description and accompanying Drawings describe examples of embodiments in some detail to aid understanding. However, the scope of protection may also include equivalents, permutations, and combinations that are not explicitly described herein. Only the

claims appended here (along with those of parent, child, or divisional patents, if any) define the limits of the protected intellectual-property rights.

CLAIMS

We claim:

1. A debugger, comprising:
 - a wireless communication block;
 - a controller; and
 - a conversion bridge;wherein the conversion bridge is to convert a first parallel signal from the controller into a first serial signal for transmission by the wireless communication block; and
 - wherein the conversion bridge is to convert a second serial signal received by the wireless communication block to a second parallel signal for analysis or storage by the controller.
2. The debugger of claim 1, wherein the analysis comprises a comparison of the second parallel signal with a set of expected values.
3. The debugger of claim 1, further comprising a de-multiplexer between the wireless communication block and the conversion bridge;
 - wherein the wireless communication block is to receive a third serial signal from the target device; and
 - wherein the de-multiplexer is to separate the third serial signal from the second serial signal.
4. The debugger of claim 1 further comprising a compressor and a decompressor between the wireless communication block and the conversion bridge;
 - wherein the compressor is to compress the first serial signal and the decompressor is to decompress the second serial signal.

5. The debugger of claim 1, wherein the conversion bridge is to convert the first parallel signal from the controller into the first serial signal for transmission by the wireless communication block.
6. A system, comprising:
 - a first wireless communication block on a first device;
 - a second wireless communication block on a second device; and
 - a wireless link between the first wireless communication block and the second wireless communication block,
 - wherein the first wireless communication block is to transmit a control signal to the second wireless communication block,
 - wherein the second wireless communication block is to transmit a trace signal to the first wireless communication block in response to the control signal,
 - wherein the first wireless communication block and the second wireless communication block are to remain activated until the later of an end-of-communication signal from the first wireless communication block or an end of the last trace signal,
 - wherein the second device is to begin receiving and reacting to the control signal after being supplied with power and before completing a boot cycle.
7. The system of claim 6, wherein the second device comprises a wireless block and an applications processor.
8. The system of claim 6, wherein the wireless block or the applications processor comprises a Universal Asynchronous Receiver/Transmitter (UART) and a component to translate a UART output to multi-wire diagnostic or debugging signals.
9. The system of claim 6, wherein the wireless link comprises Bluetooth technology.

10. The system of claim 6, wherein the wireless link comprises WiFi, WiGig, cellular phone, or NFC technology.
11. The system of claim 6, wherein the wireless link comprises a peer-to-peer link.
12. The system of claim 6, wherein the first device and the second device are to be connected to a common network.
13. The system of claim 6, wherein the first device and the second device are to be paired to create the wireless link.
14. A non-transitory machine readable storage medium containing instructions that when executed, cause a machine to:
 - connect a debugger with a target device through a network, wherein the target device is multi-wire debug compatible;
 - generate debug-protocol control signals;
 - convert the debug-protocol control signals to wireless control signals according to a wireless protocol; and
 - transmit the wireless control signals to the target device.
15. The non-transitory machine readable storage medium of claim 14 containing instructions that when executed, cause a machine to further train a link to negotiate a data rate for communication between the debugger and the target device.
16. The non-transitory machine readable storage medium of claim 14 containing instructions that when executed, cause a machine to further compress the wireless control signals.

17. The non-transitory machine readable storage medium of claim 14 containing instructions that when executed, cause a machine to further decompress the wireless returning data.

18. The non-transitory machine readable storage medium of claim 14 containing instructions that when executed, cause a machine to convert non-trace data of the wireless returning data back to a debug protocol.

19. The non-transitory machine readable storage medium of claim 14 containing instructions that when executed, cause a machine to compare the wireless returning data with a set of values.

20. The non-transitory machine readable storage medium of claim 14 containing instructions that when executed, cause a machine to output a diagnosis of the target device based on the comparison.

1/7

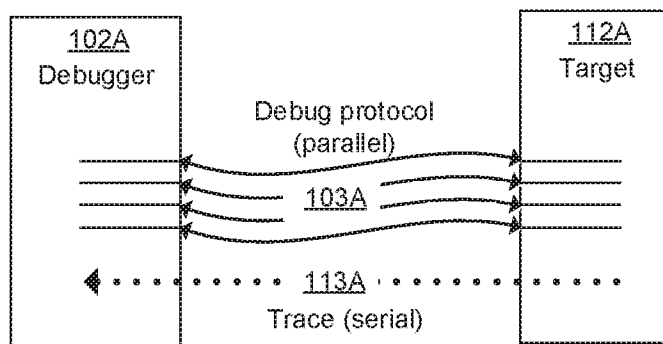


FIG. 1A
(Prior Art)

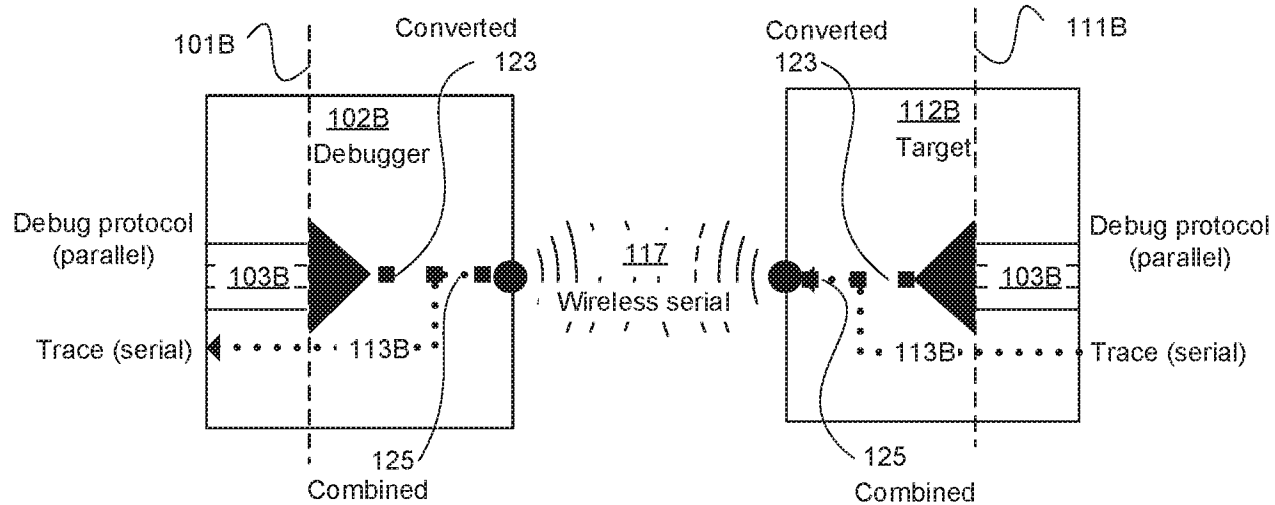


FIG. 1B

2/7

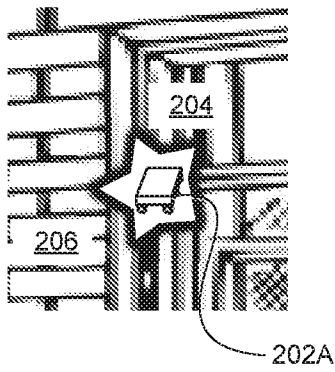


FIG. 2A

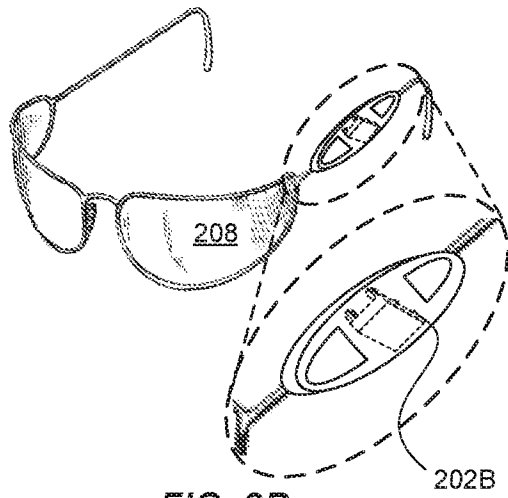


FIG. 2B

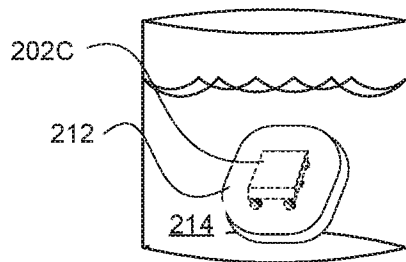


FIG. 2C

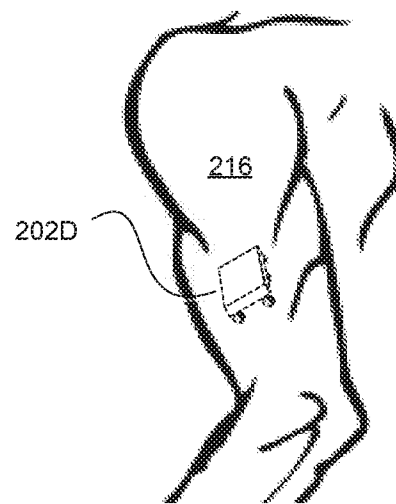


FIG. 2D

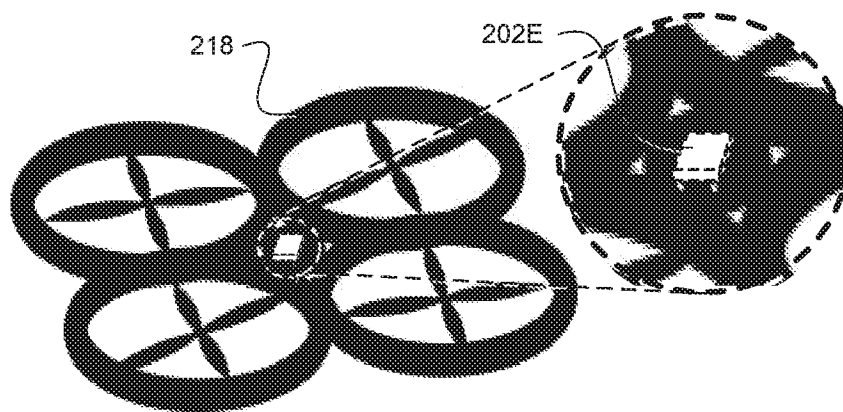


FIG. 2E

3/7

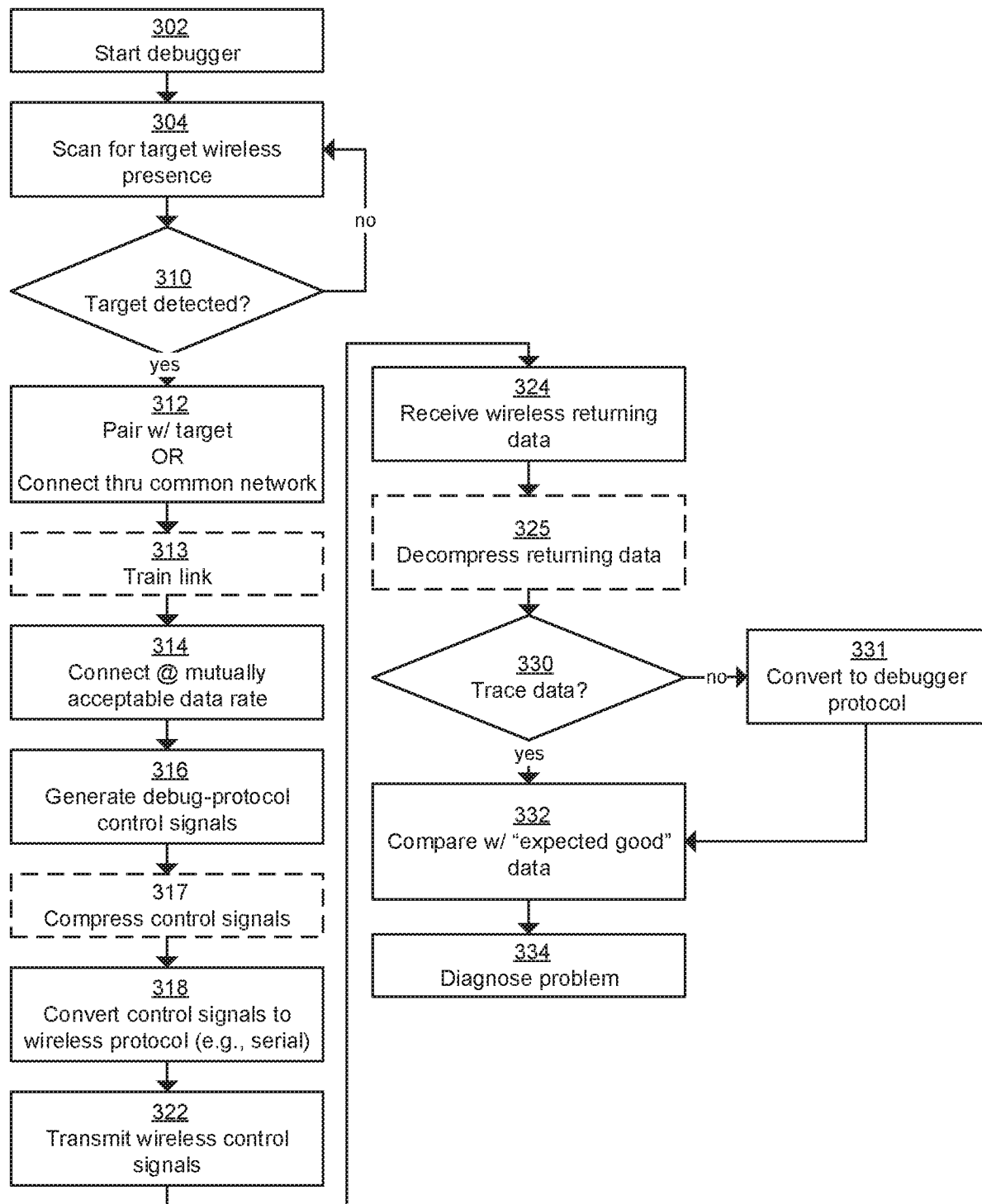


FIG. 3

4/7

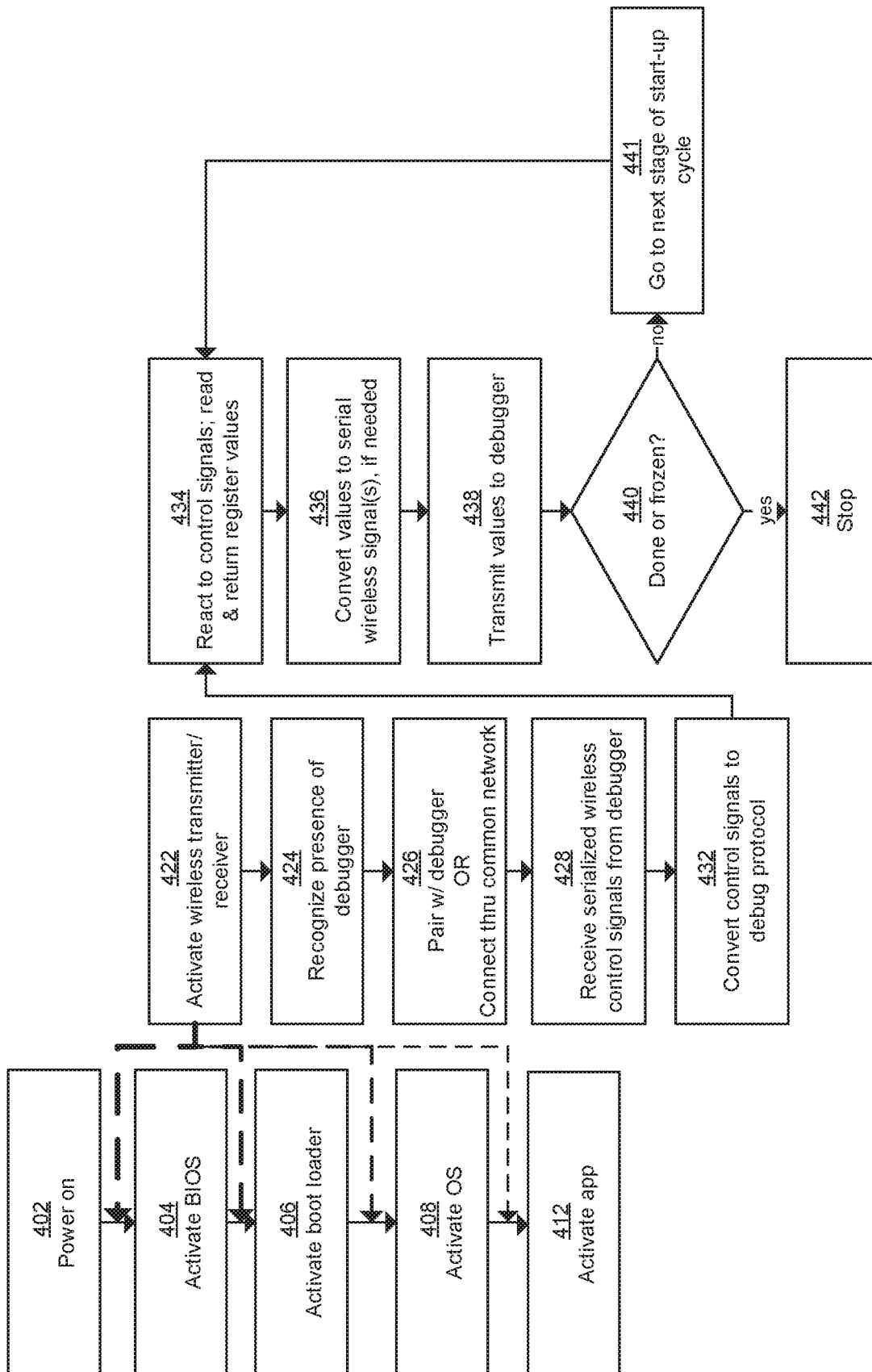


FIG. 4

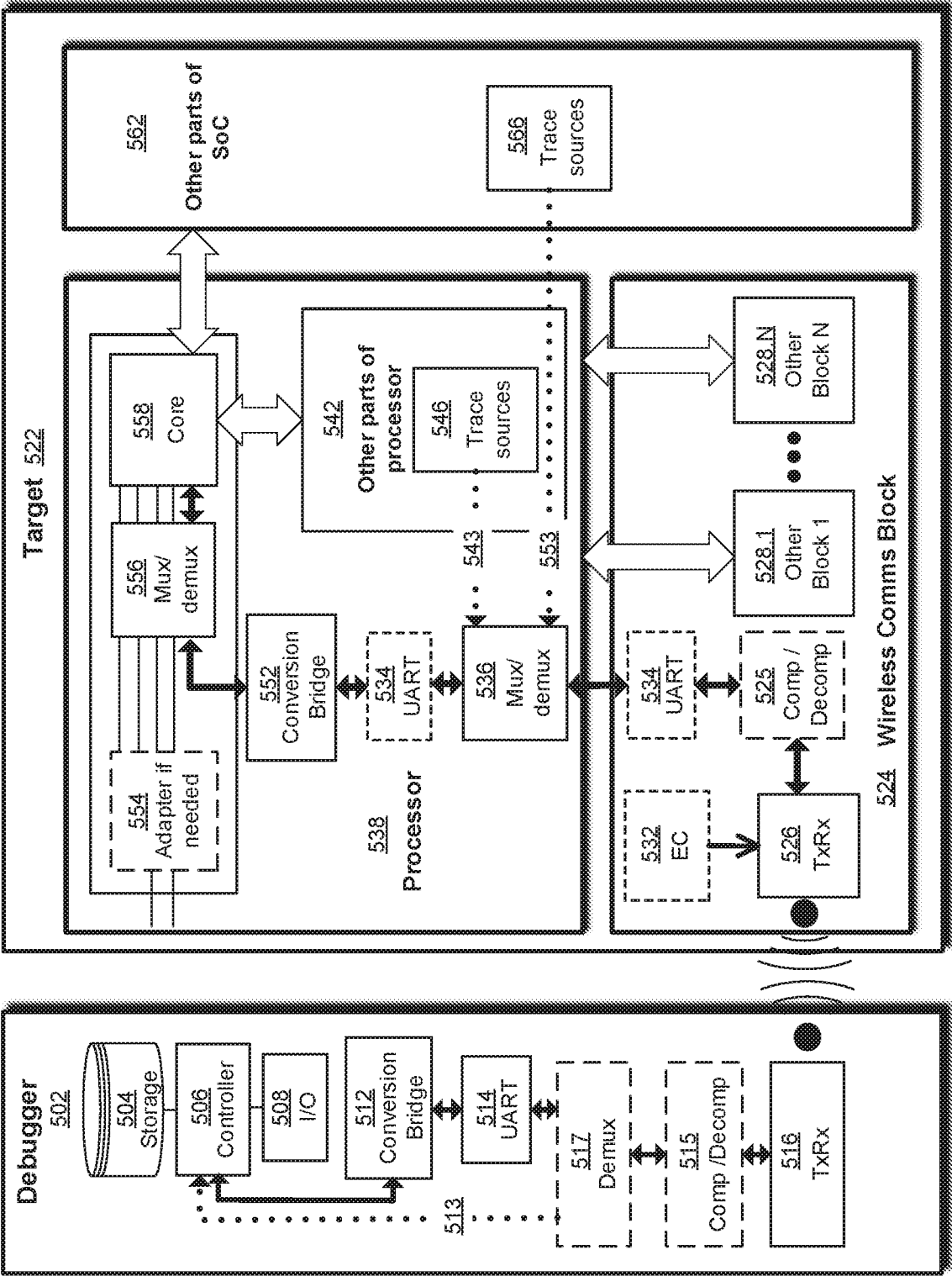


FIG. 5

6/7

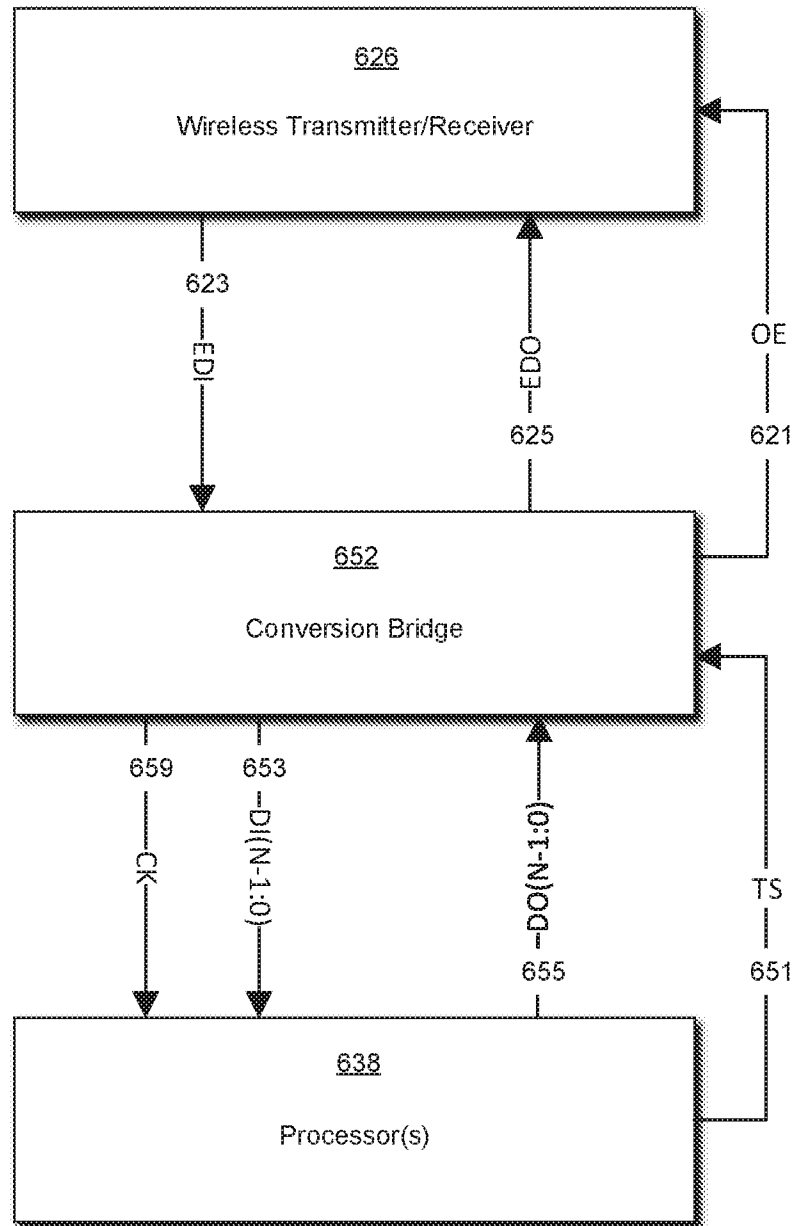


FIG. 6

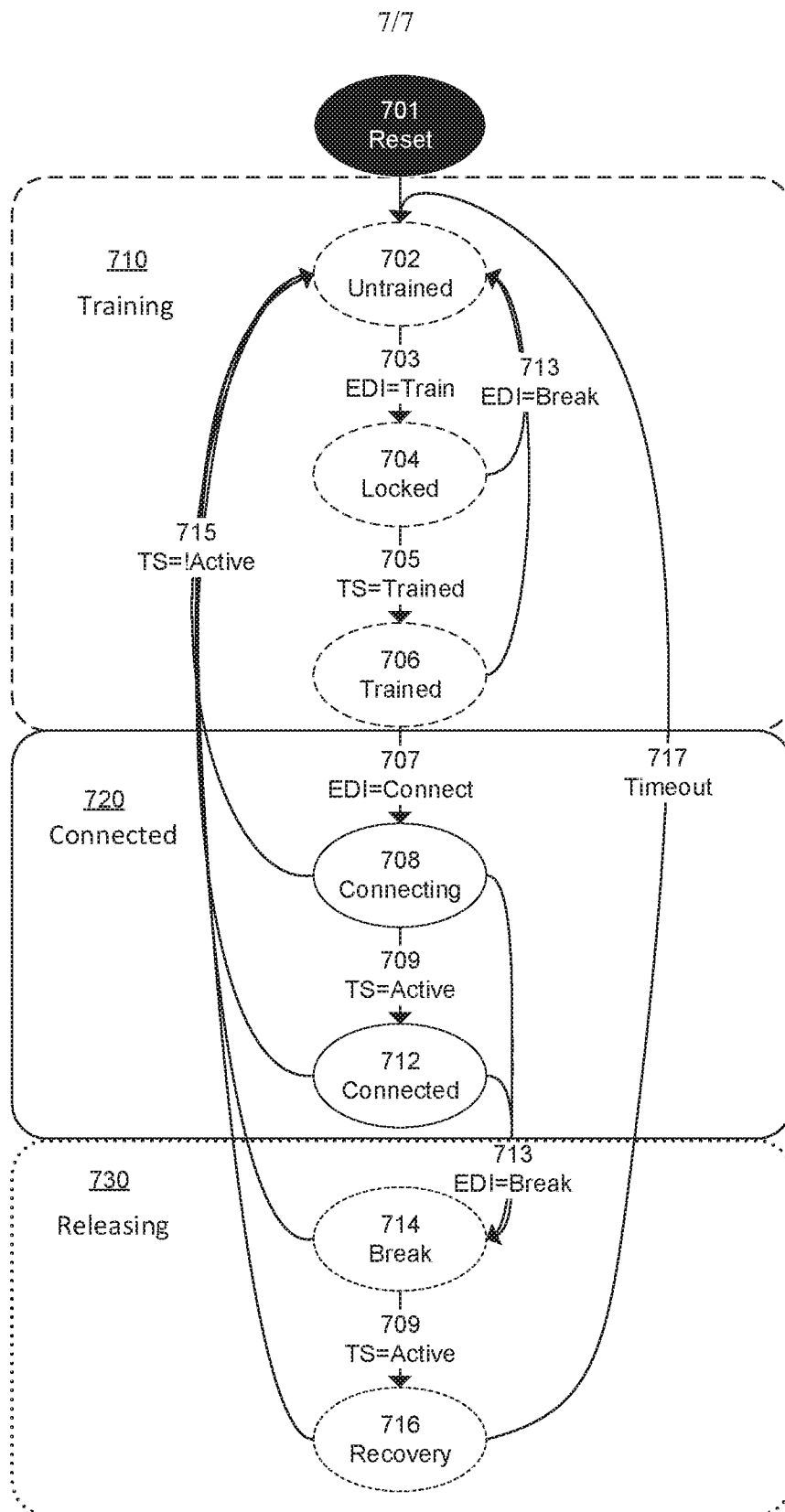


FIG. 7

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US16/66736

A. CLASSIFICATION OF SUBJECT MATTER

IPC - G01R31/3181; G01R31/3177; G06F11/22 (2017.01)

CPC - G01R31/31705; G01R31/318572; G01R31/318533; G01R31/31935; G06F11/2733

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)

See Search History document

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

See Search History document

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

See Search History document

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 2002/0133794 A1 (KANAPATHIPPILLAI, R et al.) 19 September 2002, paragraphs [0021], [0050], [0053]-[0054]	1-5
Y	US 2012/0239339 A1 (KANEKO, S) 20 September 2012, fig. 2, paragraph [0034]-[0035]	1-5
A	US 8,548,767 B2 (HOLZMANN, G et al.) 01 October 2013, entire document	1-5

☐ Further documents are listed in the continuation of Box C.☐ See patent family annex.

* 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

11 April 2017 (11.04.2017)

Date of mailing of the international search report

05 MAY 2017

Name and mailing address of the ISA/

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

Facsimile No. 571-273-8300

Authorized officer

Shane Thomas

PCT Helpdesk: 571-272-4300
PCT OSP: 571-272-7774

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US16/66736

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. ☐ Claims Nos.:
because they relate to subject matter not required to be searched by this Authority, namely:

2. ☐ Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. ☐ Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

*** Continued in extra sheet ***

1. ☐ As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.
2. ☐ As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.
3. ☐ As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:
4. ☒ No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:
1-5

Remark on Protest

- ☐ The additional search fees were accompanied by the applicant's protest and, where applicable, the payment of a protest fee.
- ☐ The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.
- ☐ No protest accompanied the payment of additional search fees.

INTERNATIONAL SEARCH REPORT

International application No.

PCT/US16/66736

*** Continued from Box No. III Observations where unity of invention is lacking ***

This application contains the following inventions or groups of inventions which are not so linked as to form a single general inventive concept under PCT Rule 13.1. In order for all inventions to be examined, the appropriate additional examination fee must be paid.

Group I: Claims 1-5 are directed towards a debugger.

Group II: Claims 6-13 are directed towards a system comprising a trace signal.

Group III: Claims 14-20 are directed towards instructions for a multi-wire debug compatible target device.

The inventions listed as Groups I-III do not relate to a single general inventive concept under PCT Rule 13.1 because, under PCT Rule 13.2, they lack the same or corresponding special technical features for the following reasons:

The special technical features of Group I include at least a controller; and a conversion bridge; wherein the conversion bridge is to convert a first parallel signal from the controller into a first serial signal for transmission by the wireless communication block; and wherein the conversion bridge is to convert a second serial signal received by the wireless communication block to a second parallel signal for analysis or storage by the controller, which are not present in Groups II-III.

The special technical features of Group II include at least wherein the second wireless communication block is to transmit a trace signal to the first wireless communication block in response to the control signal, wherein the first wireless communication block and the second wireless communication block are to remain activated until the later of an end-of-communication signal from the first wireless communication block or an end of the last trace signal, wherein the second device is to begin receiving and reacting to the control signal after being supplied with power and before completing a boot cycle, which are not present in Groups I and III.

The special technical features of Group III include at least wherein the target device is multi-wire debug compatible; generate debug-protocol control signals; convert the debug-protocol control signals to wireless control signals according to a wireless protocol, which are not present in Groups I-II.

The common technical features shared by Groups I-III are a debugger; a wireless communication block; and transmitting a control signal to a target device.

However, these common features are previously disclosed by US 2012/0239339 A1 (KANEKO). Kaneko discloses a debugger (an external verifier or debugger 220; Abstract and Fig. 2); a wireless communication block (wireless transceiver 224 includes components such as one or more modulators, demodulators, data formatters, encoders/decoders, upconverters, downconverters, amplifiers, filters antennas and the like; Fig. 2 and para [0034]); and transmitting a control signal to a target device (debugger 220 transmits high level commands over wireless link 226 to probe card electronics 200 (target device); Fig. 2 and para [0036]).

Since the common technical features are previously disclosed by the Kaneko reference, these common features are not special and so Groups I-III lack unity.