

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ARM LIMITED
Petitioner,

v.

ICPILLAR LLC
Patent Owner.

U.S. Patent No. 9,367,657

Inter Partes Review No.: IPR2024-00566

PETITION FOR *INTER PARTES* REVIEW OF U.S. PATENT NO. 9,367,657

Mail Stop “PATENT BOARD”
Patent Trial and Appeal Board
U.S. Patent and Trademark Office
P.O. Box 1450
Alexandria, VA 22313-1450

TABLE OF CONTENTS

	Page
I. INTRODUCTION	1
II. OVERVIEW	1
III. GROUNDS FOR STANDING (37 C.F.R. § 42.104(A)).....	6
IV. RELIEF REQUESTED (37 C.F.R. § 42.22(A))	6
V. REASONS FOR THE REQUESTED RELIEF	6
A. ‘657 Patent Summary	6
B. Prosecution History	8
C. Claim Construction	8
D. Person of Ordinary Skill in the Art (“POSITA”).....	9
VI. IDENTIFICATION OF CHALLENGES.....	9
A. Challenged Claims	9
B. Statutory Grounds for Challenges	9
VII. IDENTIFICATION OF HOW THE CHALLENGED CLAIMS ARE UNPATENTABLE.....	10
A. The Challenged Claims’ Structure.	10
B. The State of the Art.	10
C. Ground 1: The Challenged Claims Are Obvious Over Banerjee.....	14
1. Banerjee Overview.....	14
2. Banerjee Discloses, And/ Or Renders Obvious, The Challenged Claims.	18
a. Claim 1 Preamble: “A method for designing an integrated circuit which controls a device when the	

integrated circuit is integrated into the device comprising:”	18
i. Element 1[a] “selecting a value for a [control [and] display] type parameter associated with an electronic device from a hierarchal classification database;.....	19
selecting a value for an [input [and] output] type parameter for the electronic device from the hierarchal classification database;.....	19
receiving the selected values for the control type, display type, input type and output type parameters into a memory of a processor programmed by software commands to design a hardware integrated circuit and associated operating software for controlling the hardware integrated circuit to control the electronic device”	19
ii. Element 1[b] “analyzing by the processor the selected values for the control type, display type, input type and output type parameters of the electronic device”	39
iii. Element 1[c] “outputting by the processor a hardware integrated circuit design based on the selection of input and output characteristics of the electronic device, the hardware integrated circuit including a communication bus for communicating with the electronic device; and”	43
iv. Element 1[d] “outputting by the processor software programming commands to be executed by the hardware integrated circuit for controlling the electronic device in accordance with the selection of categorized input and output characteristics of the electronic device through the	

	communication bus of the hardware integrated circuit using a communication protocol compatible with the electronic device and the hardware integrated circuit.”.....	44
b.	Claim 5 “The method of claim 1, wherein the hardware integrated circuit design is based on at least one predetermined hardware integrated circuit component which is able to control one of the selected values for the control type, display type, input type, and output type parameters.”.....	47
c.	Claim 6 “The method of claim 5, wherein the hardware integrated circuit design is provided as a software schematic, layout, or both.”	48
d.	Claim 8 “The method of claim 1, wherein the hierarchal classification database is a component of a fabless development kit.”	48
e.	Claim 9 Preamble “A system comprising:”.....	49
	Element 9[a] “one or more processors; a design user interface provided by programming commands executed by the one or more processors, the design user interface allowing a user to characterize an electronic device by selecting a value for a control type parameter, a value for a display type parameter, a value for an input type parameter and a value for an output type parameter, all of which are associated with the operation and control of the electronic device”	49
i.	Element 9[b] “logic encoded in one or more non-transitory computer-readable media that, when executed by the one or more processors, is operable to.”.....	51

ii.	Element 9[c] “receive the selected control type, display type, input type, and output type parameters selected by the user”	51
iii.	Element 9[d] “into a memory associated with the one or more processors, the logic executed by the one or more processors operable to design a hardware integrated circuit design based on the selected control type, display type, input type and output type, the hardware integrated circuit design including a communication bus, and”	51
iv.	Element 9[e] “output software programming commands to be executed by the hardware integrated circuit for controlling the device in accordance with the selected control type, display type, input type and output type parameter values selected by the user using a communication protocol compatible with the device and the hardware integrated circuit through the communication bus of the hardware integrated circuit.”	52
f.	Claim 15 “The system of claim 9, wherein the one or more non-transitory computer-readable media further comprises logic that, when executed by the one or more processors, is further operable to simulate the hardware integrated circuit’s functionality based on the hardware integrated circuit design.”	52
g.	Claim 16 “The system of claim 9, wherein the design user interface is in communication with a hierarchal classification database, and the user selects the selected control type, display type, input type and output type parameter values from the hierarchal classification database.”	53
D.	Ground 2: The Challenged Claims Are Obvious Over Rompaey Alone Or In Combination With CoWare I.	54

1.	Rompaey Overview	54
2.	Rompaey Discloses The Challenged Claims Alone Or in Combination With CoWare I	54
a.	Claim 1 Preamble:	54
i.	1[a].....	54
ii.	1[b]	76
iii.	1[c].....	81
iv.	1[d]	82
b.	Claim 5.....	84
c.	Claim 6.....	84
d.	Claim 8.....	85
e.	Claim 9 Preamble	85
i.	9[a].....	85
ii.	9[b]	86
iii.	9[c].....	87
iv.	9[d]	87
v.	9[e].....	87
f.	Claim 15.....	87
g.	Claim 16.....	88
VIII.	THE BOARD SHOULD NOT DENY INSTITUTION UNDER SECTION 314	89
A.	The Board Should Not Deny Institution Under 35 U.S.C. § 325(d)	89
B.	The Board Should Not Deny Institution Under 35 U.S.C. § 314(a).....	89

IX.	MANDATORY NOTICES UNDER 37 C.F.R. §42.8.....	92
A.	Real Party-in-Interest (37 C.F.R. § 42.8(b)(1)).....	92
B.	Related Matters (37 C.F.R. § 42.8(b)(2)).....	93
1.	Judicial Matters	93
2.	Administrative Matters:	93
3.	Related Patents.....	93
C.	Lead/Back-up Counsel (37 C.F.R. § 42.8(b)(3))	94
D.	Notice of Service Information (37 C.F.R. § 42.8(b)(4))	94
E.	Fees Under 37 C.F.R. § 42.103	95
X.	CONCLUSION.....	95
	CERTIFICATION OF SERVICE ON PATENT OWNER.....	1

Petitioner’s Exhibit List

<i>Exhibit #</i>	<i>Description</i>
1001	U.S. Patent No. 9,367,657 (“the ‘657 patent”)
1002	Prosecution History for U.S. Patent No. 9,367,657 (downloaded from PAIR)
1003	Declaration of Dr. Henry Houh.
1004	U.S. Patent No. 5,870,588 (“Rompaey”)
1005	U.S. Pat. Appl. No. 2008/0222581 (“Banerjee”)
1006	<i>CoWare—a design environment for heterogeneous hardware/software systems</i> , K. Van Rompaey, D. Verkest, I. Bolsens and H. De Man, Proceedings EURO-DAC ’96, Geneva, Switzerland, pp. 252-257 (1996).
1007	<i>CoWare—A design environment for heterogeneous hardware/software systems</i> , Verkest, D., Van Rompaey, K., Bolsens, I. et al. Des Autom Embed Syst 1, 357–386 (1996).
1008	ICPillar Preliminary Infringement Contentions, ‘657 Patent.
1009	U.S Pat. No. 5,197,016 (“Sugimoto”)
1010	U.S. Patent No. 4,922,432 (“Kobayashi”)
1011	Fletcher, William I., <i>An Engineering Approach to Digital Design</i> , Prentice-Hall, Inc., pp. 491-505 (1984) (excerpts)
1012	ICPillar Opening Claim Construction Brief
1013	Arm Opening Claim Construction Brief.
1014	Complaint in <i>ICPillar v. ARM Ltd. et al</i> , 1:23-cv-282 (WDTX)
1015	Scheduling Order in <i>ICPillar v. ARM Ltd. et al</i> , 1:23-cv-282 (WDTX)

<i>Exhibit #</i>	<i>Description</i>
1016	Pacer Docket in <i>Sable Networks, Inc. et al v. Cloudflare, Inc.</i> , case 6:21-cv-00261, WDTX.
1017	Pacer Docket in <i>Demaray LLC v. Samsung Electronics Co., Ltd.</i> , case 6:20-cv-00636, WDTX.
1018	Arm Reply claim construction brief in related Litigation
1019	ICPillar Sur-Reply claim construction brief in related Litigation.

I. INTRODUCTION

Pursuant to 35 U.S.C. §§ 311 *et seq.* and 37 C.F.R. §§ 42.1 *et seq.*, Arm Limited (“Petitioner”) hereby petitions for an *inter partes* review (“IPR”) of U.S. Patent No. 9,367,657 (“the ‘657 patent”) owned by ICPillar LLC (“PO”). Petitioner respectfully submits that Claims 1, 5, 6, 8, 9, 15 and 16 (the “Challenged Claims”) are unpatentable under 35 U.S.C. § 103. Petitioner respectfully requests that the Board institute IPR pursuant to 37 C.F.R. § 42.108.

In IPR2024-00476, Petitioner challenged claims from the ‘657 patent’s parent, U.S. Pat. No. 8,924,899 (‘the ‘899 patent’). The Challenged Claims herein received a double patenting rejection over the corresponding ‘899 claims. The prior art herein invalidates the Challenged Claims for similar reasons as discussed in IPR2024-00476. Because U.S. Pat. Appl. No. 2008/0222581 (“Banerjee”) teaches the claimed concepts using almost identical terms as the ‘657 patent uses, this Petition cites Banerjee as Ground 1 whereas it was Ground 2 in IPR2024-00476.

II. OVERVIEW

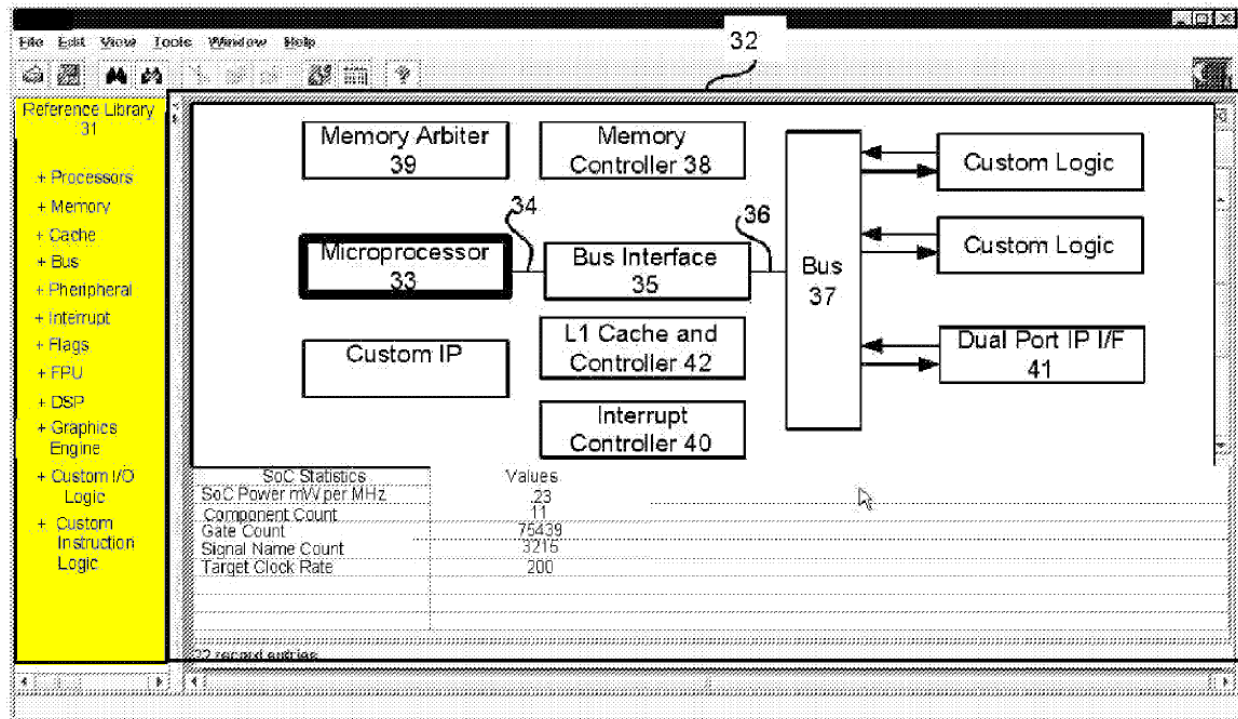
The Challenged Claims are invalid over decades of tools for designing integrated circuits. Challenged Claim 1 recites “[a] method for designing an integrated circuit which controls a device” Given that the ‘657 patent’s priority application was filed in May 2013 and that the electronics industry had been using

hardware / software design tools since the 1980's (*infra* at Section VII.B), it is unsurprising that many prior art references invalidate the Challenged Claims.

Challenged Claim 1 recites eight steps. Steps 1-5 relate to “selecting [] parameters” for control, display, input, and output and then receiving those selections into a memory. These selections relate to the functionality of the device. EX-1001, 4:35-45 (a “developer can select the functionality of their device”), 9:1-4, 14:34-50. Steps 6-8 recite “analyzing” the received selections and “outputting ... a hardware integrated circuit design” and “software” based on the selections. EX-1001, claim 1.

This Petition focuses on two similar references. First, Banerjee teaches a software tool for designing and creating both “application specific integrated circuits” (“ASICs”) and associated software to control devices.

Banerjee's designer selects design parameters such that “a functional description of the device that is generated through a series of menus presented as a graphical user interface (GUI) displaying a series of high level functional block components.” EX1005, ¶0033. Banerjee's Figure 3 exemplifies the selection of the designed device's functions / parameters.



EX-1005, Fig. 3. The designer selects from “classes” and “subclasses” of functions (such as control, display and input / output) implemented by pre-defined components to design the system. The designer continues to use hierarchal “sub-menus” to select from subclasses and to select “parameters,” “attributes,” and “characteristics” of the selected components. EX-1005, ¶¶0055-0058.

Banerjee’s designer tailors hardware and software for particular devices and their functionality (referred to by Banerjee as “applications”) through these selections. For example, Banerjee’s designer may tailor the system for devices with multimedia applications or with “real-time processing of video or audio packets.” EX-1005, ¶¶0069, 0081. Banerjee’s software tool then analyzes the designer’s

selections and outputs the hardware circuit design and the related software to implement the designed system.

The second invalidating reference is U.S. Pat. No. 5,870,588 (“Rompaey”). Rompaey teaches a software tool for co-developing hardware and software (“CoWare”). EX-1004, 10:53-11:35.

A CoWare designer selects functional blocks of the designed device from libraries / databases and refines those functions into different components by selecting further parameters. For example, Rompaey’s Fig. 10 depicts a diagram of a pager’s components that the designer selected by specifying functionality of the device and its components. EX-1004, 21:33-22:35.

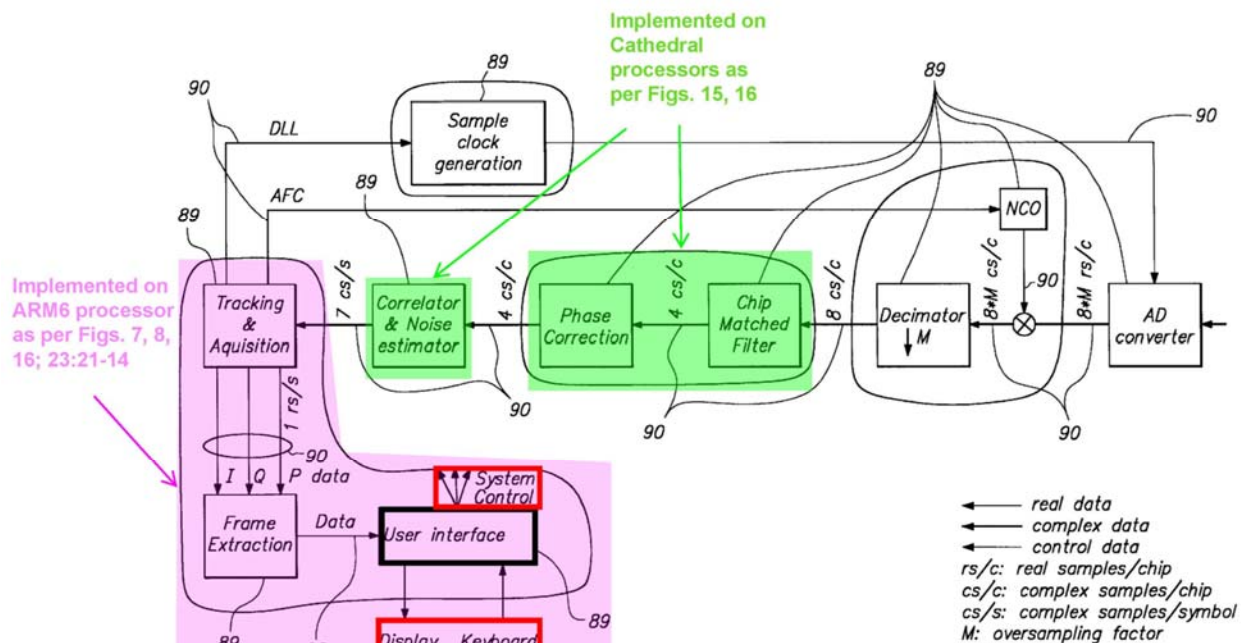
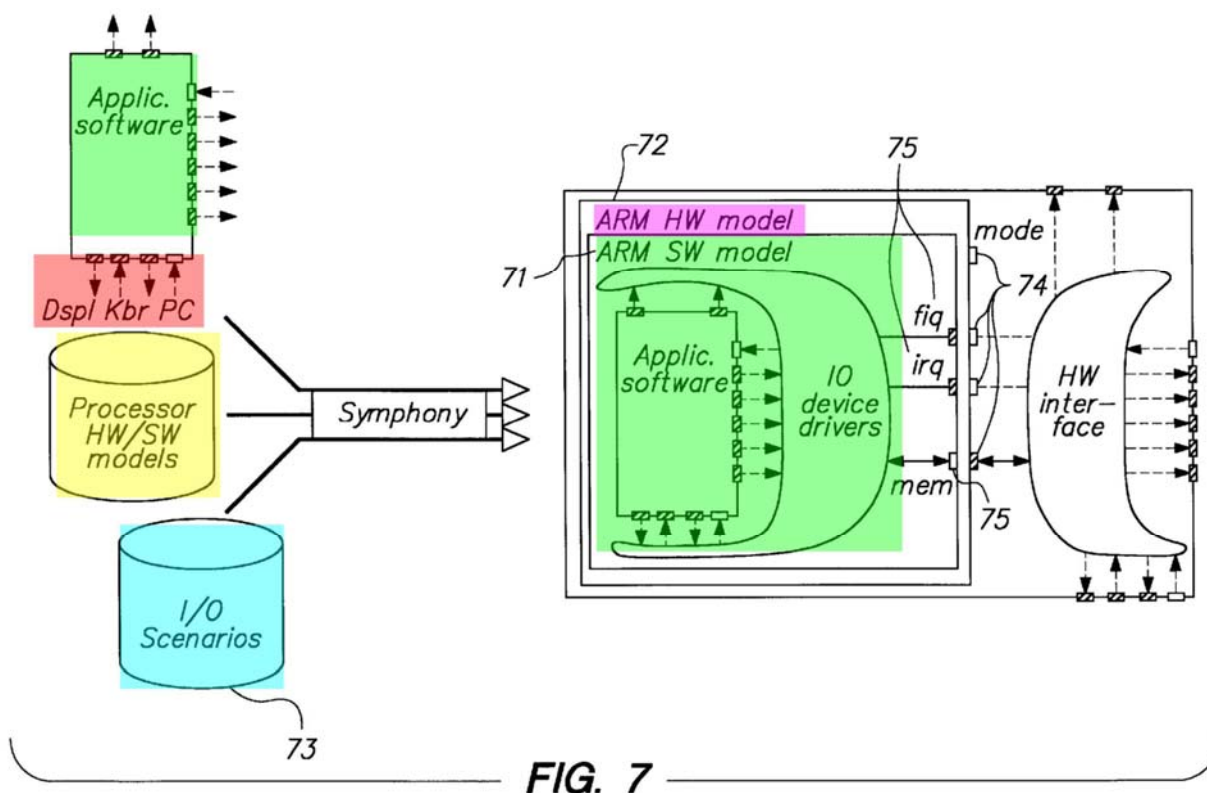


FIG. 10

EX-1004, Fig. 10.

The designer's selection includes selecting that the device will include functions such as outputting to a "Display" and inputting from a "Keyboard" (both as shown in red boxes above) and control by various processors. The designer selects from various hierarchal libraries the design components and parameters that implement the device functionality. EX-1004, Figs. 18, 6-8, 10, 15, 17; 21:35-22:67.

The CoWare system designer refines the device design by selecting further parameters. For example, Fig. 7 below shows an implementation of the ARM6 processor components in **purple** in Figure 10 above. The designer selects parameters related to pre-defined “**I/O Scenarios**” to implement the functionality using a sub-portion of CoWare called “SYMPHONY.”



EX-1004, Fig. 7. EX-1004, 17:65-18:12.

Ultimately, CoWare analyzes the selected system design and generates a hardware integrated circuit design and the associated software used to implement the designed device (*e.g.*, pager).

The Examiner did not consider the prior art cited herein. Petitioner respectfully requests that the Board institute trial and cancel the Challenged Claims.

III. GROUNDS FOR STANDING (37 C.F.R. § 42.104(A))

Petitioner certifies that the ‘657 patent is available for IPR and that Petitioner is not barred or estopped from requesting IPR of the Challenged Claims. 37 C.F.R. § 42.104(a). This Petition is filed pursuant to 37 C.F.R. § 42.106(a).

IV. RELIEF REQUESTED (37 C.F.R. § 42.22(A))

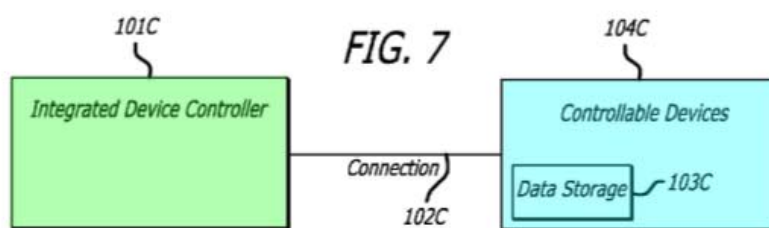
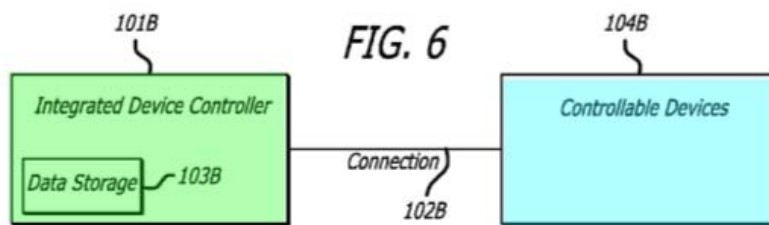
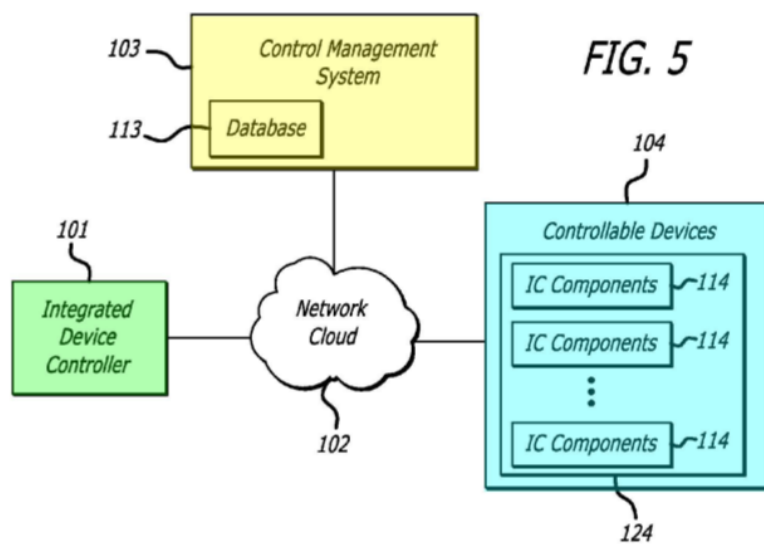
Petitioner respectfully requests institution of IPR pursuant to 37 C.F.R. § 42.108 and cancellation of the Challenged Claims.

V. REASONS FOR THE REQUESTED RELIEF

As explained below and in the Declaration of Dr. Henry Houh (EX-1003), the Challenged Claims are disclosed by, and obvious over, the prior art as understood by a person of ordinary skill in the art (“POSITA”) as of the invention date.

A. ‘657 Patent Summary

The ‘657 patent sought to solve a problem of different devices requiring different remote controls. EX-1001, 1:6-3:30. The ‘657 patent proposes a solution that operates over a connection as shown in Figures 5-7 below.



EX-1001, Figs. 5-7. According to the ‘657 patent, a “**controller**” receives information from a “**control management system**” to be able to “control” different

“**controllable devices**” across a connection. EX-1001, 10:40-11:10, 11:11-12:63 (describing Figs. 6, 7).¹

The Challenged Claims recite selecting and receiving parameters and then outputting hardware and software. The prior art included such functionality long before the ‘657 patent’s 2013 priority date.

B. Prosecution History

The ‘657 patent is a continuation of the ‘899 patent. The Examiner issued a double-patent rejection over the ‘899 claims. EX-1002, 000064-000068. PO filed a terminal disclaimer and the claims issued. EX-1002, 000010.

C. Claim Construction

Unless otherwise defined, the claims should be given “their ordinary and customary meaning.” *Phillips v. AWH Corp.*, 415 F.3d 1303 (Fed. Cir. 2005). In the Litigation, both ICPillar and Arm have submitted claim construction briefs to the Court with proposals for some terms. EX-1012, EX-1013.

For purposes of this Petition, the parties’ competing claim constructions do not need to be resolved by the Board because the prior art herein invalidates the claims under either parties’ constructions and/or under any reasonable constructions.

¹ Unless otherwise noted, all annotations and emphases are added.

D. Person of Ordinary Skill in the Art (“POSITA”)

A POSITA was familiar with automated circuit design. EX-1003, ¶¶46-47. A POSITA possesses this knowledge through training and work experience, including having a Bachelor’s or Master’s Degree in computer science, electrical engineering, or a related field, with at least three years of practical experience.

VI. IDENTIFICATION OF CHALLENGES

A. Challenged Claims

Petitioners challenge claims 1, 5, 6, 8, 9, 15, and 16 of the ‘657 patent.

B. Statutory Grounds for Challenges

The Challenges are set forth in detail below and summarized as follows:

Ground	Claims	Basis	Reference
1	1, 5, 6, 8, 9, 15, 16	§ 103	Banerjee
2	1, 5, 6, 8, 9, 15, 16	§ 103	Rompaey alone or in combination with CoWare I

U.S. Patent No. 5,870,588 (“Rompaey”) was filed on October 23, 1996 and issued on February 9, 1999. Rompaey is prior art under 35 U.S.C. § 102(a).

U.S. Patent Application No. 2008-0222581 (“Banerjee”) was filed on March 9, 2007 and published on September 11, 2008. Banerjee is prior art under Section 102(a).

CoWare—A design environment for heterogeneous hardware/software systems, Verkest, D., *et al.* Des Autom Embed Syst 1, 357–386 (1996) (“CoWare I”).

The following patents and publications present the state of the art:

CoWare-a design environment for heterogeneous hardware/software systems,
K. Van Rompaey *et al.*, Proceedings EURO-DAC '96, Geneva, Switzerland, 1996,
pp. 252-257 (“CoWare II”).

CoWare I and CoWare II published in 1996 and were submitted to the PTO during the prosecution of Rompaey (and were publicly available no later than 1999 when Rompaey issued in addition to publication noted above). CoWare I and CoWare II are prior art under 35 U.S.C. § 102(a).

VII. IDENTIFICATION OF HOW THE CHALLENGED CLAIMS ARE UNPATENTABLE

A. The Challenged Claims’ Structure.

The ‘657 patent has two independent claims: claim 1 recites a “method for designing an integrated circuit.” Claim 9 recites a corresponding “system.” The dependent Challenged Claims recite known components and functionalities.

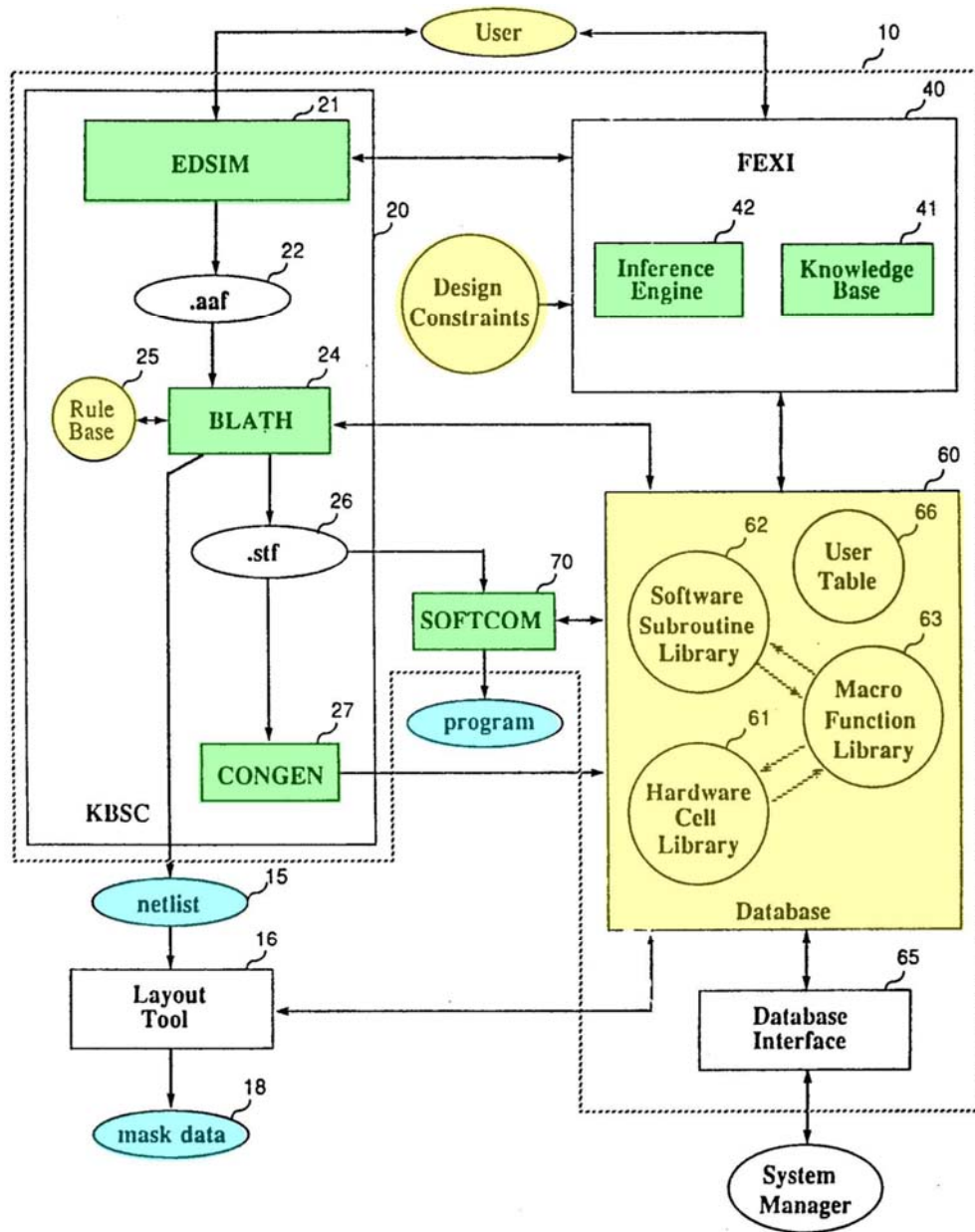
The discussion below focuses on claim 1 and its dependent claims. The analysis for claim 9 cites to the corresponding discussion for claim 1.

B. The State of the Art.

Tools to develop hardware circuits and corresponding software were well-developed long prior to the ‘657 priority date. The literature and commercially available products teach such tools from the 1980’s. EX-1003, ¶¶56-89. The ‘657 patent acknowledges that “[f]abless design of ICs and MCUs is well-known in the

art.” EX-1001, 10:9. Prior art included “computer aided design software developed by Synopsis, Inc.; Cadence Design Systems, Inc.; Magma Design Automation” EX-1001, 13:25-28.

U.S Pat. No. 5,197,016 (“Sugimoto”) was filed in 1989. Sugimoto teaches “[a] computer-aided system and method [] for designing an application specific integrated circuit (ASIC) whose intended function is implemented both by a hardware subsystem including hardware elements on the integrated circuit and by a software subsystem” EX-1009, Abstract. Sugimoto’s Fig. 2 depicts the “ISSC” system for designing an ASIC and its associated software. EX-1009, 4:26-29.



EX-1009, Fig. 2. ISSC receives inputs from **users**, **rule bases**, **design constraints**, and **libraries of software and hardware components**. ISSC analyzes to merge the components into a device design and outputs the hardware design (“**netlist**”) and software (“**program**”) to control the device. EX-1009, 3:58-5:30.

Sugimoto is a continuation-in-part of U.S. Patent No. 4,922,432 (“Kobayashi”) filed in 1988. Kobayashi presents its ASIC design for a specific device: “soft drink vending machine controller design example.” EX-1010, 12:37-14:30, Figs. 10-15.

Kobayashi’s “vending machine controller example is patterned after a textbook example used in teaching digital system controller design. *See* Fletcher, William I., *An Engineering Approach to Digital Design*” EX-1010, 12:51-55. *See also* EX-1011 (containing cited Fletcher passages).

By May 2013, a POSITA would have known that tools existed for designers to select parameters for functional components, to analyze those selections, and to generate and output the hardware integrated circuits and software to control a device. EX-1003, ¶89.

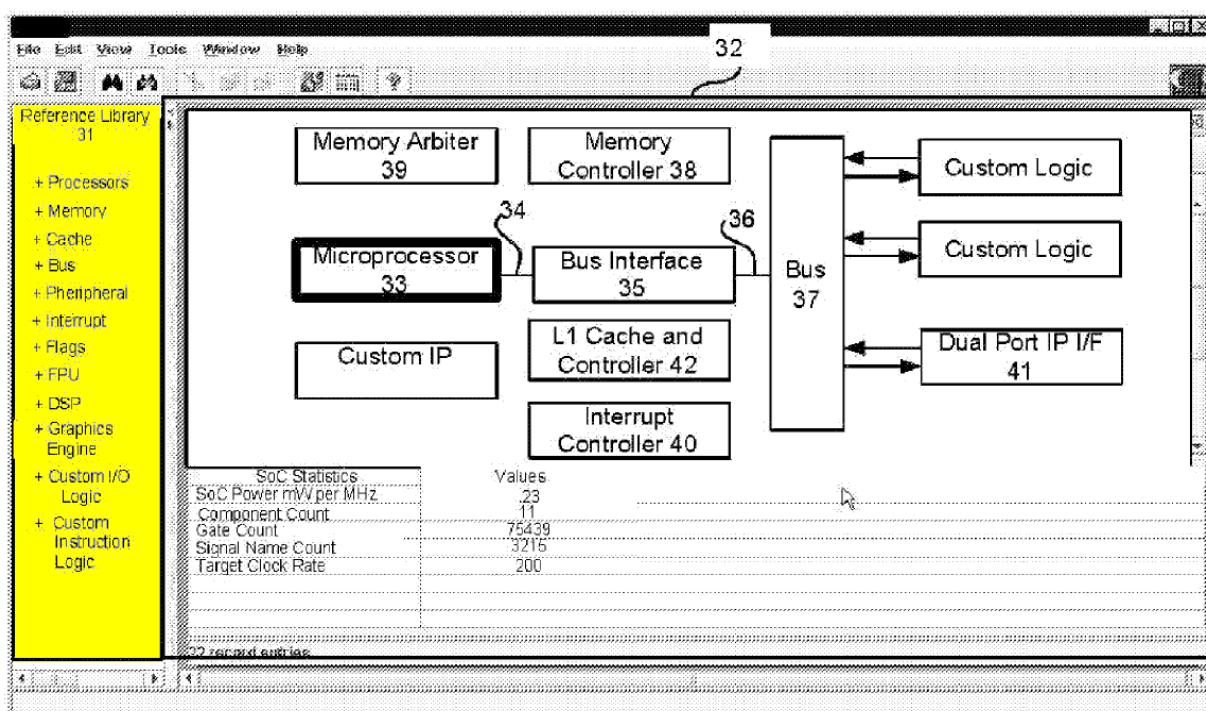
C. Ground 1: The Challenged Claims Are Obvious Over Banerjee.

1. Banerjee Overview

Banerjee discloses every element of the Challenged Claims.² EX-1003, ¶¶347-515. Banerjee teaches a software tool for designing systems / devices with hardware integrated circuits and software that executes on the hardware. The designer selects components and “parameters,” “characteristics,” and “attributes” to tailor the design for specific “applications.” Applications include devices that require “real-time processing of video or audio,” “media players,” “signal processing,” or “custom I/O logic.”

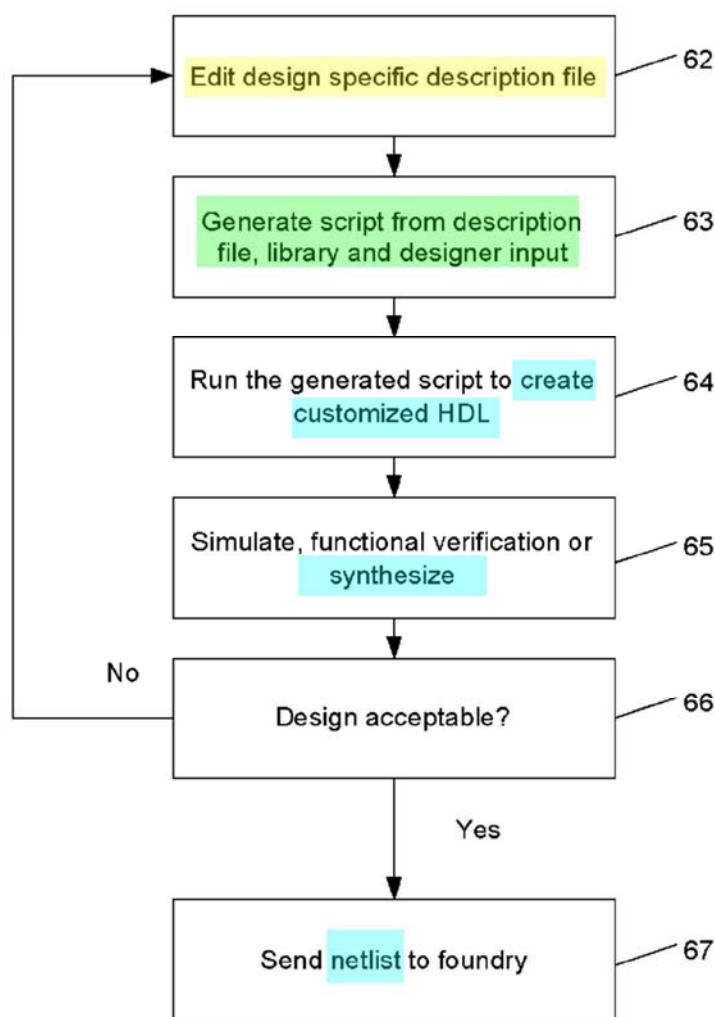
Banerjee’s designer uses a GUI with selectable classes – each with their own sub-menus of selectable components and parameters stored in a database / library.

² As used herein, “discloses” includes that the reference’s disclosure renders obvious a claim element and the claim as a whole.



EX-1005, Fig. 3. See EX-1005, 0065.

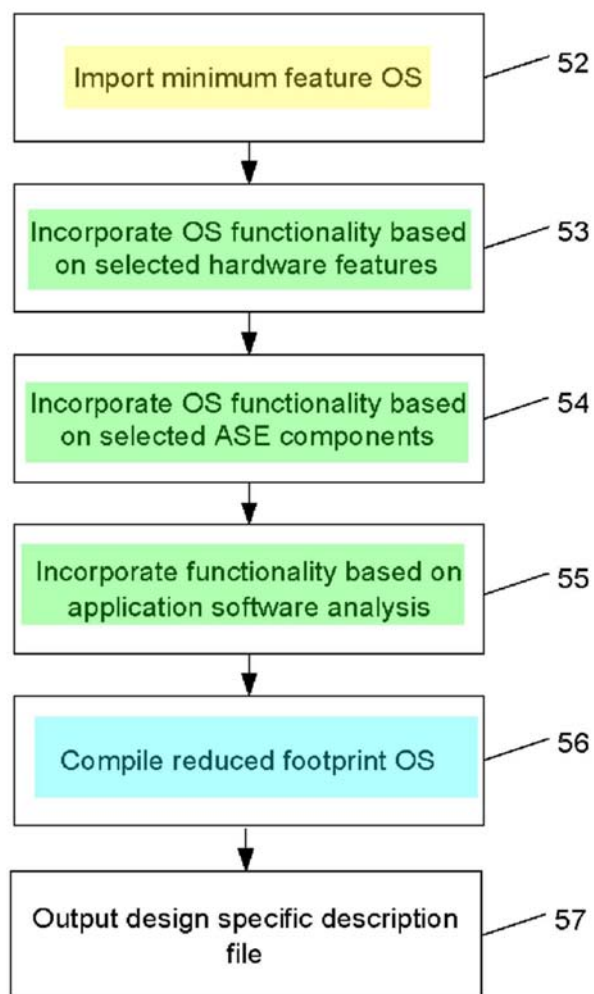
Banerjee’s Figure 6 exemplifies the hardware circuit design process. The Fig. 3 GUI is used to “**edit design specification description file.**” Then, Banerjee’s software tool receives parameters “**from description file, library, and designer input,**” and then “**creates customized HDL**” which can be “**synthesized**” into a “**netlist**” (hardware integrated circuit design).



EX-1005, Fig. 6.

Fig. 5 exemplifies how one software portion (“OS”) is generated and output. Banerjee’s tool selects a “**minimum feature OS**” from a library. The tool analyzes the system “hardware features” to select / “**incorporate OS functionality.**” Then, “the optimized operating system is **compiled into a reduced footprint operating system** that is instantiated in the device, preferably in a non-volatile computer readable storage medium.” EX-1005, ¶0083. Similar processes occur for Banerjee’s

“application software” and “middleware.” EX-1005, ¶¶0072, 0078.



EX-1005, Fig. 5.

Notably, Banerjee’s description of selecting types of functionality and related components mirrors that of the ‘657 patent.

<u>‘657 Patent</u>	<u>Banerjee</u>
“the invention includes a system and method for designing integrated circuits (IC) to	“the designer compil[e]s a set of system specifications based on the

integrate into an electronic device for control using IC components that are developed for generic <u>characteristics of electronic devices</u>. <u>A developer can select the functionality of their device</u> and an IC can be designed off the standard components.” EX-1001, 4:46-56.	desired functionality of the SoC.... These specifications can be used to define a SoC in terms of the <u>desired inputs and outputs</u>, as well as <u>desired functionality</u> ...” EX-1005, 0004.
---	---

In the ‘657 patent, the device’s “function” is “categorized again into control types, display types, input types and output types.” EX-1001, 9:1-4. The ‘657 patent uses “functions,” “parameters,” and “characteristics” interchangeably in referencing these function types. EX-1001, 9:1-4 (“function”), 8:31-33 (“parameters”), 14:47-50 (“characteristics”).

Banerjee discloses all features of the Challenged Claims and renders each invalid.

2. Banerjee Discloses, And/ Or Renders Obvious, The Challenged Claims.

a. Claim 1 Preamble: “A method for designing an integrated circuit which controls a device when the integrated circuit is integrated into the device comprising:”

Banerjee’s software tool (including elements 15 and 16 in Fig. 1) allows a designer to select the functionality of a desired ASICs or SoC for controlling specific applications (device’s functionality) and then output the integrated circuit design and

software. EX-1005, ¶¶0027-0028; claim 17 (“... selecting components... generating ... HDL ... file describing the integrated circuit.”); EX-1003, ¶¶351-354.

i. Element 1[a] “selecting a value for a [control [and] display] type parameter associated with an electronic device from a hierarchal classification database;

selecting a value for an [input [and] output] type parameter for the electronic device from the hierarchal classification database;

receiving the selected values for the control type, display type, input type and output type parameters into a memory of a processor programmed by software commands to design a hardware integrated circuit and associated operating software for controlling the hardware integrated circuit to control the electronic device”

Banerjee’s tool allows the user to select various “parameters” from a hierarchal database using almost verbatim language to the Challenged Claims:

Module 21 further includes a rules engine for determining configurable parameters that the designer or computer program may select. In one embodiment, the rules engine uses a database of descriptive elements that can be selectively incorporated into an RTL description of the component. In other embodiments, the rules engine may access a flat description file that includes one or more parameters that can be selectively customized by either inputting data or by selecting one of a

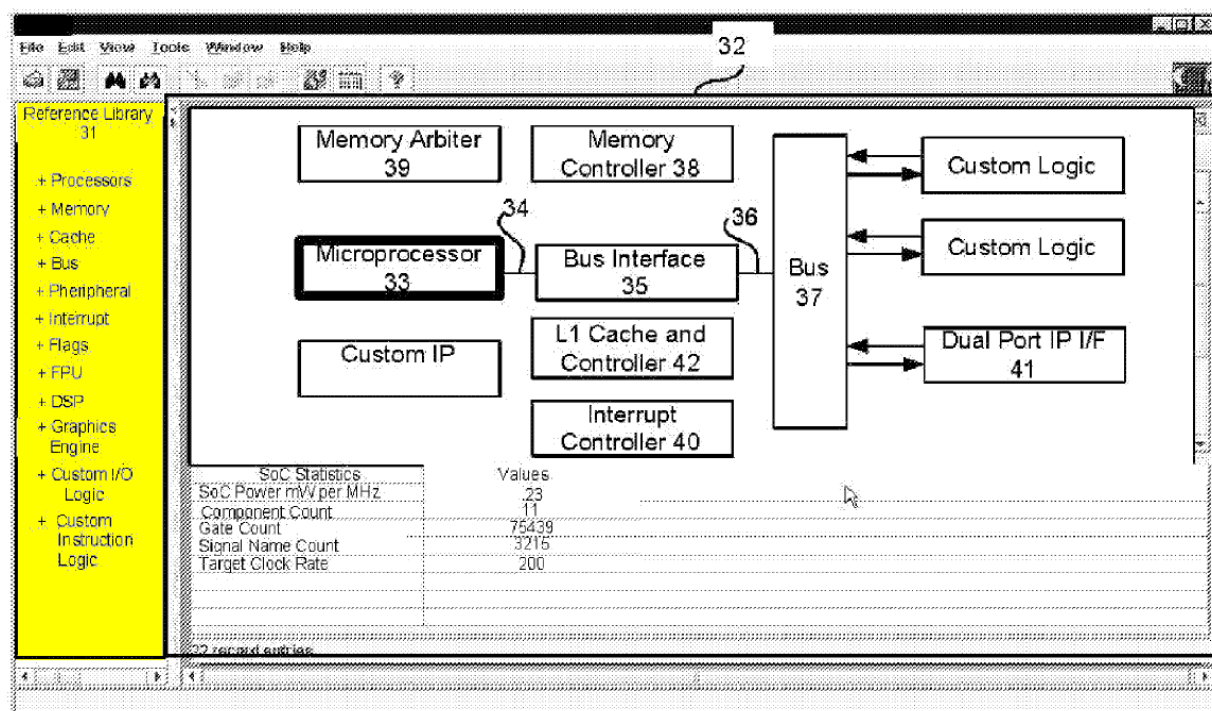
plurality of options for each component. In other embodiments, the description of customizable parameters are described in a relational database with each description of the component incorporating various combinations of the configurable options.

EX-1005, ¶0050.

Banerjee teaches many ways for a designer to “select” and “receive” the claimed “parameters” for the device.

First, Banerjee’s software allows the system designer to “select” the components implementing a design and the “parameters,” “attributes,” and “characteristics” thereof. These selections are received via user interfaces including a GUI, “design scripts,” or “in the form of an interactive software prompt, or ... [other identified inputs].” EX-1005, ¶¶0033-0034, 0047-0048, 0058-0061, 0088; Figs 2, 3. EX-1003, ¶¶355-370.

As shown in Figure 3 below, the GUI allows for selecting from different classes of components in a “reference library 31” database (an example of the claimed hierarchal classification database):



Second, Banerjee’s tool allows for selection and receives the claimed parameters by “access[ing] a flat description file that includes one or more parameters that can be selectively customized by either inputting data or by selecting one of a plurality of options for each component.” EX-1005, 0050. The designer modifies the design with selections as part of an iterative process after the tool has performed analysis of the design. EX-1005, ¶¶0009, 0028, 0030, 0040, 0045, 0047, 0067, 0069-0070 (“as the designer makes additional performance selections, the description file is updated with the RTL code to reflect the designer's designation.”). EX-1003, ¶¶371-372.

Third, the software will “select” additional parameters from the libraries and databases to perform, for example, optimization. EX-1005, 0064-0067 (describing

optimization process); 0078-0086 (describing selections available to system for optimization); 0096-0097; 0030, 0034-0038, 0045; 0055-0056. EX-1003, ¶¶373.

These forms of selections are from “libraries” and “repositories” and “databases” – which are all examples of the claimed hierarchal classification database. EX-1003, ¶¶374-379, 444-445. “FIG. 4 illustrates a graphical user interface and database structure.” EX-1005, 0016.

As illustrated in FIG. 4, selected performance characteristics, provided from module 22, are compared to a matrix of performance specifications associated with each component in a repository. In one embodiment, the description of the performance specifications for each component comprises a plurality of entries in a relational database that includes a basic RTL description of the component linked to other files that describe selectively customizable parameters

EX-1005, 0063. The selectable components are stored in a “database” as “class[es]” that have selectable “subclass[es]”:

[T]ables 46-48 preferably comprise an object-oriented programming database where each component is defined as an object that is a member of a component class (for example, a processor class) and each version or variation of a particular component is an individual instance that is

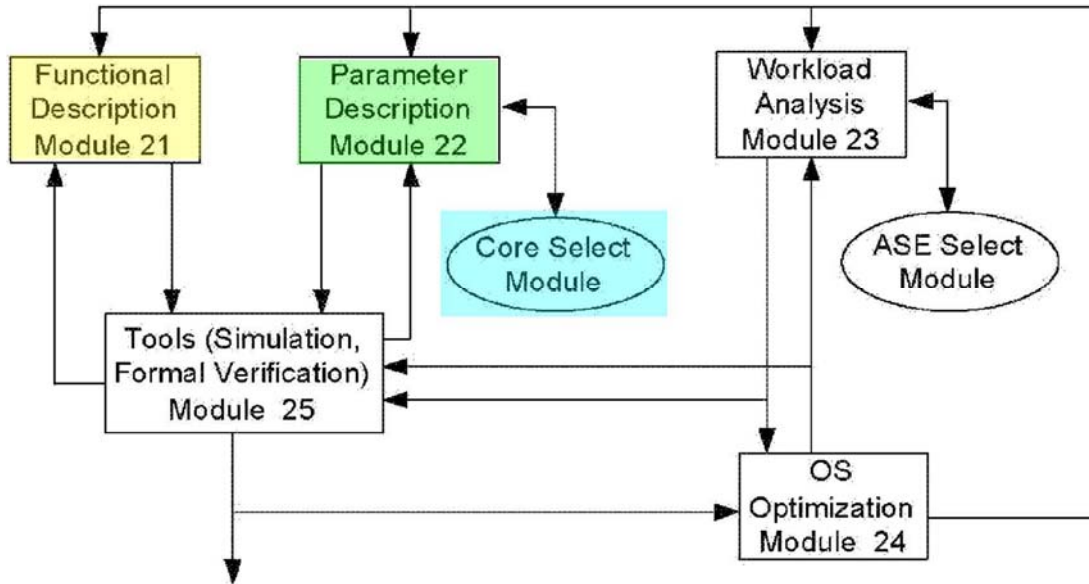
defined by a subclass that is a more specialized description of the processor class.

EX-1006, 0065. *See also*, EX-1005, 0050 (quoted above describing the “database of descriptive elements that can be selectively incorporated” and the “customizable parameters” in a “relational database...”: 0065 (the “selected characteristics” of the “processor class” “utilize a relational or multidimensional database...”; 0126 (“associated with each component listed in the reference library is a vendor source database file....”); EX-1005, Fig. 1 (“Repositories”); Fig. 3 (“Reference library”), Fig. 4 (“Processor library”), Fig. 6, Fig. 11 (“library files”).³

Banerjee’s Fig. 2 depicts an iterative, interactive design process in which the designer selects the functional parameters for the designed device:

³ The ‘657 patent similarly describes using a hierarchal classification database.

EX-1001, 9:45-48 (“designer will may use a series of cascading menus corresponding to the classification scheme of the hierarchal classification scheme....”)



EX-1005, Fig. 2. First, a designer makes selections for a system design:

Computer program 15 includes a **functional description module 21** **that guides the designer to develop a high-level or macro description of the device**. In one embodiment, the description includes a functional description of the device that is generated through a series of menus presented as a graphical user interface (GUI) displaying a series of high level functional block components. Computer program 15 also includes the capability to upload design scripts that describe customized elements created by the designer.

EX-1005, 0033. EX-1003, ¶¶380-381.

Second, Banerjee's selects further "parameters," "attributes," and "characteristics" of the design and its components in selected classes and subclasses (each being examples of claimed "parameters"). EX-1005, ¶¶0067-0071, 0074-

0092.

A second **module 22, the parameter description module, guides the developer in defining the performance requirements for the functional blocks**. Module 22 provides a GUI for selecting ... optimal clock operating frequencies, power levels and maximum allowed power dissipation, die size, and I/O requirements. Module 21 utilizes these parameters to optimize the selection of functional block components....

Module 22 enables the designer to select performance characteristics that are not otherwise determinable from the functional description...

EX-1005, 0034-0035.

The “**core select module**” works with “**parameter description module 22**” to allow designers to select from classes / subclasses of components (such as processors) and select their parameters, features, and attributes tailored to the specific desired application / device. EX-1005, ¶¶0064-0068, Figs. 4, 8. EX-1003, ¶¶382-383.

Banerjee’s selections tailor components and parameters thereof for specific “applications” (*e.g.*, devices controlled by the designed hardware / software) including:

- “application[s] [that] demands real-time processing of video or audio packets” with a “DSP application” (EX-1005, ¶0069);
- a “media player application” in which “the processor is configured to include media processing or DSP extension instructions for certain media applications” (EX-1005, ¶0081);
- “signal processing application” (EX-1005, ¶0086);
- Various “embedded applications” and “logical control” applications (EX-1005, ¶¶0067 0039, 0072, 0080, 0086);
- “a dual core SoC” application (EX-1005, ¶0082).

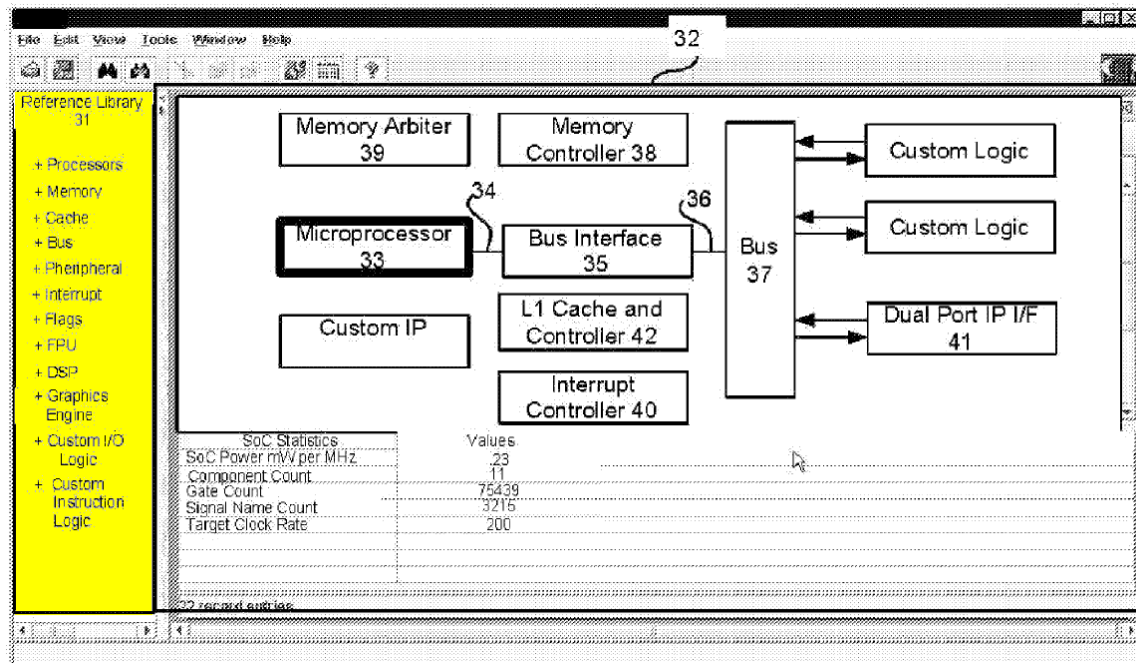
Banerjee describes a “cell phone” device that might have specific application requirements. EX-1005, ¶0077. Thus, Banerjee’s system allows the system designer to select the components and parameters of those components that are specific to the devices for which the hardware and software is being developed. EX-1003, ¶¶384-385.

The selections (from Banerjee’s hierarchal databases of “class” and “subclass”) allow Banerjee’s system to select the functions / parameters for a particular application device. For example, Banerjee allows for selection of different types of instructions. Because there is a “large number of possible applications for a SoC, the specific instruction set modifications can be numerous and are typically dependent on the specific application.” EX-1005, ¶0070. Using the “libraries,” “the

designer can easily mix and match ‘best of breed’ IP for a particular application.” EX-1005, ¶0124. “[A] designer may select the [component] that is best suited to the particular application...” EX-1005, ¶0125. EX-1003, ¶386.

Banerjee uses different terminology to describe the selection of the claimed “parameters,” such as “characteristics” (EX-1005, ¶¶0035, 0038, 0040, 0063-0065, 0069); “parameters” (EX-1005, ¶¶0034, 0041-0042, 0045, 0050, 0063-0066, 0078, 0097, 0116); and “attributes” (EX-1005, ¶¶0043, 0055-0057, 0066, 0072, 0088-0089). EX-1003, ¶387.

One invalidating example is that a designer may use the Fig. 3 GUI to specify an “application” device that includes: (a) a processor to control the device; (b) a “graphics engine” providing a graphic display; (c) a multimedia component providing “real-time processing of video or audio packets” and (d) “custom I/O logic” and/or “peripherals” that provide the inputs and outputs to the device. EX-1003, ¶388.



EX-1005, Fig. 3. Fig. 3 shows classes of different component types along the left column. EX-1005, ¶0048. As noted herein, these are “classes” with “subclasses” of devices with a “more specialized description” of the class. EX-1005, 0065.

These classes are “hierarchical” such that, when “selected,” each category displays further selectable options including subclasses:

[R]eference area 31 comprises a hierarchical index of various components. ... Component may be selected in reference area 31 and dragged into the work palette 32...

EX-1005, ¶0049. The designer then selects “configurable parameters” to customize the “class” or “subclass” of component. EX-1005, ¶0050 (quoted above).

The designer may specify additional parameters and “attributes” (claimed “parameters”) of the selected components. EX-1005, ¶¶0051-0054. Further

categorized and hierarchical selections can be made:

Further, once a component is selected, the designer may, if desired, also input related attributes when a selected component also requires the selection of hardware attributes. In one embodiment, the designer is prompted to select an appropriate value for the attribute from a popup menu. ... Instead of selecting a value, the designer can select an “optimize value” essentially reserving the attribute value to be determined at a later point in the design process or determined automatically via analysis of the workload by selecting the optimization function.

EX-1005, ¶0055. “[A] second menu, if necessary, enables selection” of additional parameters and “attributes.” EX-1005, ¶0056. EX-1003, ¶¶330-335. These additional selectable menus are examples of selecting parameters from a hierarchical classification database. Each relates to an additional selection based on the prior selection from a set of known parameters corresponding to the prior selection. EX-1003, ¶¶389-396.

Banerjee allows for selection of the claimed control, display, output and input parameters in selecting the “classes” and “subclasses” of components through the Fig. 3 GUI and subsequent selections. EX-1003, ¶¶398-436.

Components in reference area 31 may include processors, multiple processor cores, memory sub-systems, bus sub-systems, I/O interfaces, and custom logic by way of example. Indeed it is not unusual for a SoC design to include a processor, a co-processor, video processor, digital signal processing processor or a graphics processor coupled to other components such as memory, memory controllers, caches, a DSP, bus structures (such as a crossbar switch, hierarchical bus structures, networks on chip (NoCs) or direct point-to-point communications), bus controllers and I/O bridges all of which are obtained from the reference area 31 of the GUI. Advantageously, reference area 31 may provide the designer of a SoC with a plurality of configurable or configured processor cores so that the designer can implement a wide range of functions ranging from conventional DSP functions to media-processing functions while employing a consistent set of development tools.

EX-1005, 0048.

“Control Type Parameter”

Banerjee describes selection of many “control type parameters.” This includes the selection of any of the following “classes” (and then the “subclasses”) of components shown in Fig. 3 and described in 0048 (quoted above): processors,

multiple processor cores, memory controllers, DSP (digital signal processor), and bus controllers.⁴ *See also* EX-1005, 0027 (selecting “processor core, specialized processors, buses and controllers, and software components”), 0116 (listing types of selectable control parameters), 0053-0054, 0125. Similarly, the “applications that are essentially logical control oriented” show selection of “control type parameters.” EX-1005, ¶0086. EX-1003, ¶¶401-406.

In addition, selection of Banerjee’s software is also a “control type parameter.” EX-1003, ¶405. The ‘657 patent states that the designed device “contains Control Management Software.” EX-1001, 12:65-13:8, Fig. 7. The “Management software” may include “drivers or firmware necessary to operate the IC.” EX-1001, 15:64-67. This is exactly what Banerjee discloses in the ability to select the software that controls Banerjee’s designed device. EX-1005, 0075 (“module 24 configures the operating system to support application specific functions by incorporating only selected drivers”); 0072-0074, 0081, 0085, Fig. 5. Banerjee notes that “the design of a new device typically requires control and application software.” EX-1005, 0030. Such control software (including the drivers, firmware and operating system that may be selected) is a control type

⁴ The ‘657 uses the term “controller” to describe a device that is separate and apart from the claimed “device” that is being designed. *E.g.*, EX-1001, Figs. 6-7.

parameter as claimed.

For each selected “class,” Banerjee’s tool allows for further selections of “parameters,” “characteristics,” and “attributes” for that class and the subsequently selected subclasses as detailed above. These further selections also constitute selection of the claimed control type parameters.

“Display Type Parameter.”

Banerjee discloses the selection of several different display type parameters. EX-1003, ¶¶407-414. The “display type” includes visual communication such as “word or graphic.” EX-1001, 9:16, 14:53-57.

Banerjee discloses selecting the class of “processor” which then includes selecting subclasses such as “a graphics processor” or a “video processor” and components that can “implement... DSP [and] media-processing functions.” EX-1005, 0048; Fig. 3 (including “Graphics Engine” class). The designer can further select from associated other parameters / components such as memories and buses. EX-1005, 0048; Fig. 3. This class / subclass selection is then further refined by selecting further parameters, attributes, and characteristics (the claimed “parameter”) as described herein. Each selection corresponds to the claimed selection. EX-1003, ¶409.

The “graphics engine” (or “graphics processor”), “video processor” and “digital signal processor (‘DSP’)” are device output and display parameters (*e.g.*, a

device that outputs/displays graphics and/or video) that the designer selects. EX-1005, ¶0048.

The designer may “select” a subclass of graphics or video processor and attributes thereof as a parameter. EX-1005, ¶0066. Within that selection, the designer may select additional parameters. *E.g.*, EX-1005, ¶0068 (“...provide additional instructions targeted to a specific task. For example, one logic block may provide a set of instructions directed to handling graphics oriented tasks”). The designer can select “attributes” of such processors. EX-1003, ¶¶411-412.

For the video/DSP, “if the application demands real-time processing of video or audio packets, the processor core 33 selected by the designer may include specific instructions directed towards the digital signal processing (DSP) application.” EX-1005, ¶0069. These result in the selections of various selected additional “IP blocks” and “application specific instruction set modifications.” *Id.* EX-1003, ¶412.

Similarly, the designer may select / configure “media-processing functions” (a type of display, input, and/or output parameter for the device / application) and then the parameters of such a selection. EX-1005, ¶¶0048, 0081. “[I]f the processor is configured to include media processing or DSP extension instructions for certain media applications,” then the system allows for further configuration between hardware and software. EX-1005, ¶0081. EX-1003, ¶413.

The selection of additional instructions for graphics, video, or media-

processing is a claimed selection. EX-1003, ¶414.

“Output Type Parameters.”

Banerjee discloses the selection of several different output type parameters. EX-1003, ¶¶415-417. The ‘657 patent teaches that “input and output of the device may be audio, video, etc. ... the blender's display references an ‘output’” EX-1001, 14:59-61; EX-1001, 9:22-24. Thus, each “display type parameter” immediately above also corresponds to an “output type parameter.” For example, to meet the claim (which recites separate display / output parameters), Banerjee’s teaching would include the “graphics” as a “display” parameter and the “real-time video” as an “output” parameter.

Similarly, Banerjee teaches selecting components for “real-time processing of ... audio packets.” EX-1005, 0069. This exactly corresponds to the ‘657 statement that “output of the device may be audio.” EX-1001, 14:59. This further teaches the selection of output parameters in Banerjee as detailed for the “media” processing in the display parameters above. These selections include the selections of classes, subclasses, parameters, characteristics and attributes in Banerjee as described herein. For example, the designer “selects” parameters related to additional audio processing logic and instructions. EX-1005, 0069-0071. Furthermore, the “output type parameters” selection would include the “peripherals” [sic] and the “I/O custom logic” discussed below for the “input type parameters.” EX-1003, ¶417.

“Input type parameter”

Banerjee discloses the selection of several different input type parameters. EX-1003, ¶¶418-436. Banerjee notes that its specifications for the designed device including “desired inputs and outputs.” EX-1005, 0004.

Banerjee’s Fig. 3 specifically identifies at least the following selectable “classes” that correspond to the input type parameters: “pheripheral” [sic], “DSP,” “custom I/O logic,” “Custom Instruction Logic,” and “Interrupt.” EX-1003, ¶¶420-422. Each class includes its own set of selectable “subclasses” along with parameters, attributes and characteristics as described above. Each selection corresponds to the claimed selection of input type parameters.

For example, the “custom I/O logic” would include the components and parameters thereof to receive and send wireless signals for Banerjee’s disclosed cell phone. This would include the various RF and processing components to receive, demodulate, decode and process input RF signals (and the corresponding logic to transmit output signals). Rompaey gives examples of these component types. EX-1003, ¶¶421-426.

For example, the “custom I/O logic” allows for selection of categories of such input / outputs specific for the designed application (device). EX-1005, ¶0048 (selection of “I/O interfaces and custom logic”); ¶0034 (“parameter description module ... provides a GUI for selecting ... I/O requirements”). Such I/O

requirements disclose (or render obvious) the selection of input and output parameters. Banerjee specifically discloses a “cell phone” as a device. EX-1005, 0077. A POSITA would understand “cell phone” to include devices with input / output functionality such as touch screens or buttons – which exactly matches the ‘657 statement that “the input device may be push buttons or touch sensitive elements....” EX-1001, 9:7-8. EX-1003, ¶¶425-426. These are also forms of “custom I/O logic.”

Banerjee’s selection of interrupts and “peripherals” also disclose (or render obvious) forms of both input and output parameters to be selected. EX-1005, Fig. 3, ¶¶0053, 0076, 0097. A POSITA understood “peripherals” to include devices such as keyboards, buttons, and displays. Such peripherals would also include “buses” (Fig. 3) that provided for communications to device components. EX-1003, ¶¶427-433.

Banerjee’s inputs / outputs include the ability to send / receive data in Banerjee’s described cell phone through “custom I/O logic” such as network interface controllers.⁵ This would be part of the “desired inputs and outputs” selected by the user for the device. EX-1005, 0004.

⁵ *E.g.*, EX-1001, 17:56-61 (describing “network interface controller” as an input / output for the device).

Banerjee’s “digital signal processor” (DSP) selection also is a form of an input type parameter. The “signal” being “processed” would be understood to be an “input” – for example being “input” to the device from sensors, communication components, peripherals, or offboard storage. By defining the type of DSP and its associated specific instructions, the designer is selecting an input type parameter. EX-1005, 0069, 0081, 0048. EX-1003, ¶¶434-435.

Each selection above is “received” as claimed because Banerjee’s tool receives these selections into the memory of the tool used by the designer to select, tailor, and optimize the design and output the hardware and software for the design. EX-1003, ¶436.

If PO argues that Banerjee does not expressly disclose this element, Banerjee renders obvious every aspect of this element for the reasons above and, in addition for the following reasons. EX-1003, ¶¶437-446. For example, according to PO, the claimed parameters are met by the following:

control type parameter (e.g., system control signals), a display type parameter (e.g., multimedia or graphics aspects), an input type parameter (e.g., memory and other system inputs), and an output type parameter (e.g., system outputs) for an integrated circuit design.

EX-1008, 000025. As detailed above, Banerjee extensively details on selecting each type of parameters cited by PO including extensive details on “memory.” EX-1005,

Fig. 3; ¶¶0038, 0048, 0053-0054, 0073-0074, 0077-0078, 0088, 0096-0098, 0125.

At a minimum, Banerjee renders these elements obvious. EX-1003, ¶¶437-441.

Banerjee’s “ASICs” are tailored toward the “application” (the device) that the IC device will control including a controlling IC for its “application” such as multimedia, embedded, real-time audio and video. Such “applications” either are, or define, the device. Banerjee gives an example of tailoring the IC for a cell phone. At a minimum, Banerjee renders obvious receiving selected parameters for, or associated with, the device. EX-1003, ¶¶442-443.

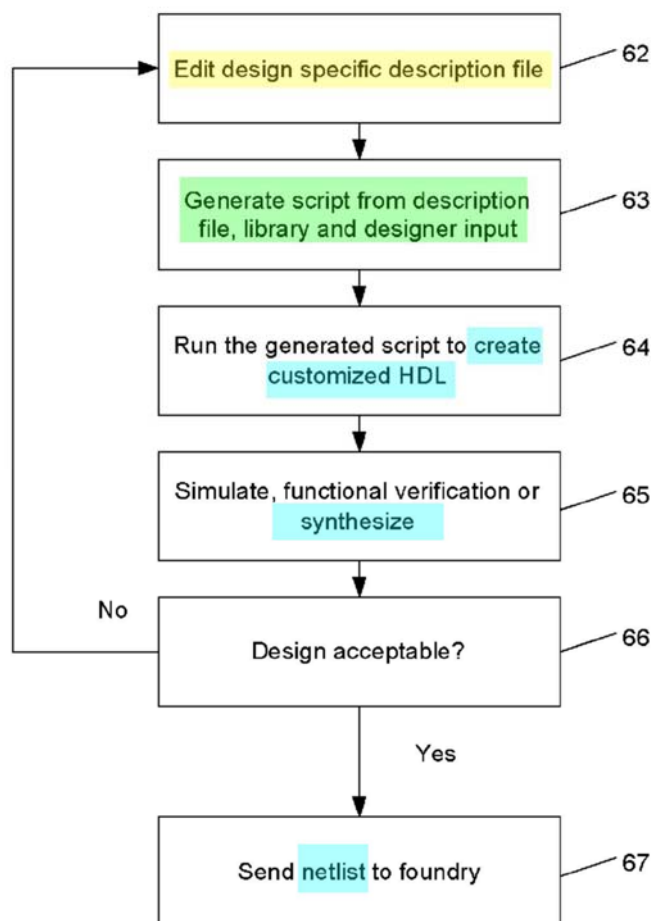
Moreover, PO asserts that the claimed parameter selection corresponds to “options and parameters for the integrated circuit that correspond to those of the electronic devices into which the integrated circuits will be integrated.” EX-1008, 000025. Similarly, PO asserts “[e]xamples of controlled devices include ...devices (and/or their subsystems, chips, or boards).” EX-1008, 000024-25. As detailed above, Banerjee discloses such configuration and attributes of both the designed integrated circuits and the associated device (with the tailored “applications”) that the circuits (and software) control including their “subsystems” and “chips” of those devices. EX-1003, ¶443.

Regarding “hierarchal classification database” Banerjee discloses a GUI of components and subcomponents identified specifically as “classes” and “subclasses” and their “parameters” and “characteristics” and “attributes” –which

can be progressively refined with further selections using the sub-menus – all from forms of databases such as a “relational database” as detailed above. At a minimum, this renders obvious the receipt of selections from a hierarchal classification database. EX-1003, ¶¶444-445.

ii. Element 1[b] “analyzing by the processor the selected values for the control type, display type, input type and output type parameters of the electronic device”

Banerjee’s tool performs different forms of the claimed “analyzing.” EX-1003, ¶¶448-462. Banerjee’s tool analyzes the selected parameters to create the hardware integrated circuit design for the device in formats such as netlists, HDL, VHDL, and RTL. Fig. 6 exemplifies this process in which the selections are received from **edited description files, the libraries/repositories and designer inputs** with the system ultimately “**creat[ing] the customized HDL**” in the form of a “**netlist**”:



EX-1005, Fig. 6. In Fig. 6, “a description file specific to a particular design is edited at 62 where the designer selects desired modules or functions for the design. In one embodiment, analysis of the workload is used to automatically generate a description file. Instructions relating to the design are added to or subtracted from the description file, or generated as necessary.” EX-1005, ¶0086. *See also* EX-1005, ¶0089 (“create a customized VHDL or Verilog model.”). EX-1003, ¶¶449-452.

Banerjee repeatedly teaches that the received selections are used to generate the integrated circuit. “[C]omputer program 15 comprises an interactive architecture

module that gathers information regarding the design parameters of the SoC and generates a merged HDL file descriptive of the design that complies with those design parameters.” EX-1005, ¶0045.

Such analysis includes Banerjee’s processes of “merging,” “compiling,” and “optimizing.” “Module 25 also performs logic synthesis to compile and optimize the high-level circuit description and generate a gate-level netlist representation of the device.” EX-1005, ¶0044. *See also* EX-1005, ¶¶0089-0102 (describing the compilation, merge, and optimization analysis occurring before the generation of final Verilog, VHDL or other HDL files); Figs. 7, 8 (“optimizing” the design); Fig. 9 (“configure the ASE [application specific instruction extensions]”; Figure 10 (“generate structural HDL files...”; ¶0038 (“optimization module...”). Similarly, Banerjee’s tools analyze the design for incompatibilities. EX-1005, ¶0041, ¶¶0082, 0125. EX-1003, ¶¶453-455.

As detailed in 1[a], Banerjee’s “modules 21, 22, and 23” allow the designer to generate the device components:

Using the descriptions generated by modules 21, 22 and 23, module 25 of computer program 15 assembles a description file that comprises a series of predefined VHDL, Verilog, System Verilog or other synthesizable constructs of an HDL....

EX-1005, ¶0040; *See also* EX-1005, ¶0043 (“computer program 15 initiates the

creation and testing of HDL-based synthesizable SoCs”), ¶¶0009 (“an interactive architecture module gathers workload information about the SoC and generates a merged HDL file descriptive of the design.”); ¶¶0047, 0057, 0071; *see also* EX-1005, claims 1, 17. EX-1003, ¶¶456-458.

Furthermore, Banerjee’s “module 23 performs a dynamic analysis of the workload to identify instructions and to select appropriate application specific instruction extensions ... Based on the analysis, module 23 recommends the addition of certain application specific instruction set modifications or the designer may specify a defined user instruction.” EX-1005, ¶0067. “[M]odule 24 selects operating functionality based on the workload analysis performed by module 23.” EX-1005, ¶0076. Similarly, a designer’s selection of particular memory “is analyzed (based on the designer’s technology choice), and the memory subsystem values are set accordingly.” EX-1005, ¶0097.

Banerjee’s tool analyzes the design (including parameters) to add “application specific extensions” (referenced as “ASE” in Figs. 2, 4-5, 9-11). EX-1005, ¶0097. The extensions include “application specific instruction extensions” (EX-1005, ¶0067) that include logic for additional instructions that the processor executes to perform tasks specific to the application. EX-1005, ¶0081; Fig. 3 (“custom instruction logic”); ¶¶0048, 0069.

Thus, Banerjee teaches many forms of analyzing the received selections to

generate and output the hardware circuit design and associated software. EX-1003, ¶¶459-460.

If PO argues that Banerjee does not disclose this element, Banerjee renders obvious every aspect of this element. EX-1003, ¶¶461-462. The ‘657 specification does not use any variant of “analyzing.” PO asserts that the claimed “*analyzing...*” occurs when a designer “uses the [] [d]esign [t]ools to define the desired parameters and configure the selected IP and create ... RTL, an IP microarchitecture, and executable files for the integrated circuit” EX-1008, 000029. As detailed above, Banerjee discloses such analyzing to create and output the RTL, netlist, or VHDL defining the IP microarchitecture and the executable files (the OS, application software, and middleware) for the integrated circuit appropriate for the device implementing the specified “applications.”

iii. Element 1[c] “outputting by the processor a hardware integrated circuit design based on the selection of input and output characteristics of the electronic device, the hardware integrated circuit including a communication bus for communicating with the electronic device; and”

As detailed in 1[b], Banerjee’s tool outputs the hardware integrated circuit designs based on the selections detailed in 1[a] in formats including Verilog, VHDL, System Verilog, HDL, netlist, GDSII (GDS2), and RTL. EX-1003, ¶¶464-470; *see* 1[b]; EX-1005, ¶0094 (“output of computer program 15 is an HDL synthesizable

model of the design.”); Figs. 5-7, 10-12; ¶¶0028, 0040, 0043-0045, 0050, 0066, 0071, 0089-0093, 0095-0099; 0102; 0107, 0116-0117, 0008-0012, claims 1, 17-18.

Banerjee teaches that the designed circuit includes communication buses as claimed. *See* Fig. 3 (“Bus” in element 31; “Bus 37”); ¶¶0116 (“the IC comprises a SoC comprising ... one or more bus structures ...”); 0027 (“...automated description of hardware components, such as ... buses and controllers...”), 0031, 0048 (“Components in reference area 31 may include ... bus sub-systems ... Indeed it is not unusual for a SoC design to include ... bus structures (such as a crossbar switch, hierarchical bus structures, networks on chip (NoCs) or direct point-to-point communications), bus controllers ... all of which are obtained from the reference area 31...”); 0051-0054 (details of buses), 0082, 0088, 0098, 0116, 0125. EX-1003, ¶¶468-469.

- iv. Element 1[d] “outputting by the processor software programming commands to be executed by the hardware integrated circuit for controlling the electronic device in accordance with the selection of categorized input and output characteristics of the electronic device through the communication bus of the hardware integrated circuit using a communication protocol compatible with the electronic device and the hardware integrated circuit.”*

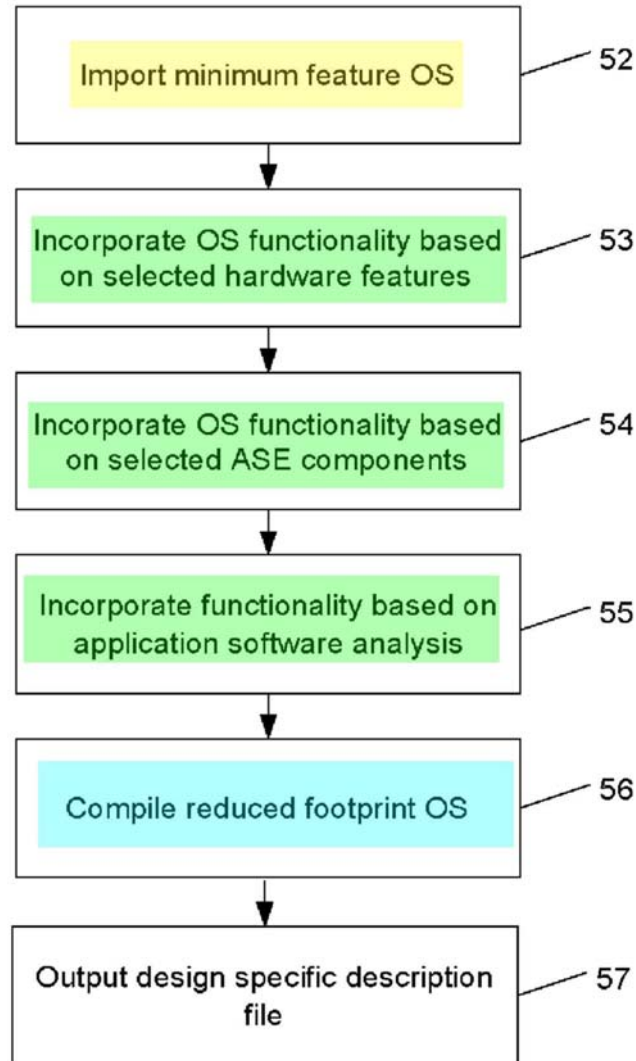
Banerjee’s tool outputs software based on the selections identified in 1[a]. EX-1003, ¶¶472-480. The output software includes: (1) the operating system; (2)

application software; (3) application specific instruction extensions; and (4) “middleware” (such as I/O device driver software).

[M]odule 24 **outputs** C or C++ code that configures the Windows, Linux or RTOS operating system by outputting header files, switches, and a modified makefile. Module 24 then determines the optimal configuration for the design by selecting the processor that most closely matches the specified parameters and providing any additional functionality in software either at the application, middleware or operating system level.

EX-1005, ¶0078; *See also* EX-1005, ¶¶0038-0039, 0045-0046, 0056, 0067, 0084.

Fig. 5 exemplifies how the operating system software is compiled and output:



EX-1005, Fig. 5; ¶¶0072-0082 (describing Fig. 5 and how OS software is generated and output), 0083 (“the optimized operating system is compiled into a reduced footprint operating system that is instantiated in the device, preferably in a non-volatile computer readable storage medium.”). For example, the OS and device drivers are specific to the applications and components selected. EX-1005, ¶¶0075, 0081. EX-1003, ¶¶472-476.

Similarly, Banerjee’s tool generates and outputs software instructions related

to the application specific instructions. EX-1005, ¶¶0086, 0038. EX-1003, ¶477.

The software is purposefully designed to use communications protocols compatible with the device and to operate through one or more communications buses on the device so that the software can control the device. For example, the software has to load from the persistent storage into the processor(s) controlling the device to be able to execute properly and the software must be able to communicate across the buses (with a compatible protocol) to access data in memory and control the device. EX-1003, ¶¶478-480; EX-1005, ¶0082 (“module 24 would compile a symmetric multiprocessing operating system that would support, by way of example, PCI Express, DDR2, bus width, description of on-chip buses, whether it is compatible with the Open Core Protocol, cache sizes and many features that can be configured in a typical SoC design.”)

Moreover, Banerjee teaches that compatibility (such as communication protocol compatibility for the bus(es)) among the components is tested and resolved during the development phase. EX-1005, ¶¶0041, 0125.

- b. **Claim 5 “The method of claim 1, wherein the hardware integrated circuit design is based on at least one predetermined hardware integrated circuit component which is able to control one of the selected values for the control type, display type, input type, and output type parameters.”**

Banerjee uses predetermined hardware integrated circuit designs that have been stored in, for example, the repositories and libraries. EX-1003, ¶¶482-483. *See*

EX-1005, ¶¶0040 (“module 25 of computer program 15 assembles a description file that comprises a series of **predefined** VHDL, Verilog, System Verilog or other synthesizable constructs of an HDL.”); 0057 (“Once a component is at least partially instantiated in the work palette 32, a corresponding HDL description script is obtained from repository 18 and added to the description file.”), 0107 (“generate a description of a SoC design using libraries of IP blocks (components) in a hardware description language”); 0028 (repositories); 0054 (“repositories can be defined and components can be readily added to or subtracted from each of the repositories”); 0063, 0065-0066, 0071; 0079-0080; 0088; 0101-0102, 0106 (“component libraries”), 0117; 0119, 0124.

c. **Claim 6 “The method of claim 5, wherein the hardware integrated circuit design is provided as a software schematic, layout, or both.”**

As detailed in 1[b], Banerjee’s tool outputs the hardware integrated circuit designs in formats including Verilog, VHDL, System Verilog, HDL, netlist, GDSII (GDS2), and RTL. *See also* EX-1005, ¶¶0091 (“a synthesis engine may create a specific physical layout of the design”). These disclose, or render obvious, this claim. EX-1003, ¶¶484-487.

d. **Claim 8 “The method of claim 1, wherein the hierarchal classification database is a component of a fabless development kit.”**

Element 1[a] details Banerjee’s tool and the database(s) used by it (including

the libraries and repositories). Banerjee’s tool is part of a “fabless development kit.”

Fabless design of ICs and MCUs is well known in the art. In general, the term fabless design is applied to process where the design of the IC is accomplished separately from the manufacture of the device itself.

EX-1001, 10:9-13. Thus, this claim recites an element that is “well known in the art” and does not impart independent patentability.

Banerjee repeatedly describes its software tool(s) as being separate from the manufacturing foundry. EX-1005, 0050 (after the design is completed, “the circuit description is sent to a selected foundry”); 0091 (exported to a foundry at conclusion of design process); 0108, 0116-0128 (can send design to different foundries). Thus, Banerjee discloses that its tool performs the design separately from the manufacture and discloses this claim.

Given that “fabless design ... is well known in the art,” the decision to use Banerjee’s tool either separately or integrated with a specific foundry is, at most, an obvious design choice. At a minimum, Banerjee renders this element obvious. EX-1003, ¶¶488-498.

e. **Claim 9 Preamble “A system comprising:”**

Banerjee teaches this “system.” See claim 1 above.

Element 9[a] “one or more processors; a design user interface provided by programming commands executed by the one or more processors, the design user interface allowing a

user to characterize an electronic device by selecting a value for a control type parameter, a value for a display type parameter, a value for an input type parameter and a value for an output type parameter, all of which are associated with the operation and control of the electronic device”

Banerjee discloses this element. Banerjee’s “**computer program**” executes on a “**server**” computer that discloses, or renders obvious, the claimed processor[s] as part of a “system” because a processor executes software. EX-1003, ¶¶500-506.

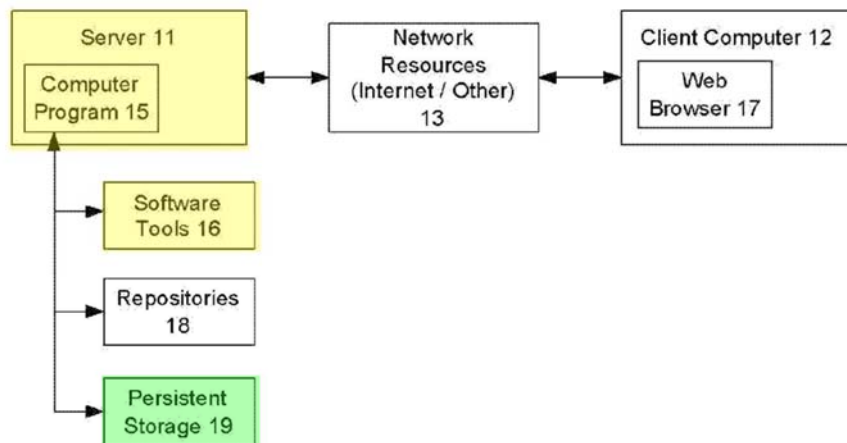


Figure 1

EX-1005, Fig. 1; *see also* EX-1005, Fig. 13; ¶¶0027-0028 (“the present invention ... takes ... the form of a server 11.... A silicon-on-chip (SoC) designer accesses **server-resident computer program 15 and software tools 16** ... server 11 may be one server in a data center environment Computer program 15, software tools 16 and repositories 18 are all retained on a **persistent storage device 19** that is capable of storing computer readable information. **Persistent storage device 19 may be**

either coupled to or is a component of server 12. Software tools 16 may further include synthesis and formal verification tools.”); 0059, 0062, 0085, 0091, 0103-0105, 0112-0116.

Element 1[a] details the “design user interface” provide by Banerjee for designing a system including characterizing an electronic device by selecting the claimed parameters.

- i. Element 9[b] “logic encoded in one or more non-transitory computer-readable media that, when executed by the one or more processors, is operable to.”*

As noted in 9[a], Banerjee’s “[c]omputer program 15, software tools 16 and repositories 18 are retained on a persistent storage device 19 that is capable of storing computer readable information,” which is part of the server and is non-transitory computer readable media that encodes logic to be executed by the processor(s). EX-1005, ¶¶0028, Figs. 1, 13. EX-1003, ¶¶507-508.

- ii. Element 9[c] “receive the selected control type, display type, input type, and output type parameters selected by the user”*

Banerjee’s program receives a selection of the claimed parameters as detailed in 1[a]. EX-1003, ¶¶509, 355-446.

- iii. Element 9[d] “into a memory associated with the one or more processors, the logic executed by the one or more processors operable to design a hardware integrated circuit design based on the*

selected control type, display type, input type and output type, the hardware integrated circuit design including a communication bus, and”

As detailed in 1[b] and 1[c], Banerjee’s program tools design a hardware integrated circuit design and output in many formats based on the selections in 9[c] / 1[a]. See 1[d], 5. EX-1003, ¶¶510, ¶¶447-471.

iv. *Element 9[e] “output software programming commands to be executed by the hardware integrated circuit for controlling the device in accordance with the selected control type, display type, input type and output type parameter values selected by the user using a communication protocol compatible with the device and the hardware integrated circuit through the communication bus of the hardware integrated circuit.”*

As detailed in 1[d], Banerjee discloses outputting software as claimed based on the selected parameters from 1[a].

f. **Claim 15 “The system of claim 9, wherein the one or more non-transitory computer-readable media further comprises logic that, when executed by the one or more processors, is further operable to simulate the hardware integrated circuit’s functionality based on the hardware integrated circuit design.”**

Banerjee’s tool simulates the hardware integrated circuit functionality. EX-1003, ¶¶512-513. See EX-1005, Figs. 2-3, 6-8, 10-13 (all reciting simulation); ¶¶0010 (“an apparatus for generating, simulating, and/or synthesizing a SoC design”), 0031, 0038, 0041-0042 (“During the simulation and verification, ... Timing

simulation is preferably done after device layout Full-system simulation of the design ensure that it will support the workload as analyzed and parameterized by modules 23 and 24....”), 0057 (“... module 21 builds the HDL file for simulation”); 0058, 0090-0093 (describing simulation process); 0099, 0103, 0115, 0127.

- g. **Claim 16 “The system of claim 9, wherein the design user interface is in communication with a hierarchal classification database, and the user selects the selected control type, display type, input type and output type parameter values from the hierarchal classification database.”**

As detailed in 1[a], Banerjee’s tool uses databases, including libraries and repositories, for a user to select classes and subclasses of components and then to select parameters, attributes, and characteristics for the components. Banerjee’s databases teach, or render obvious, the claimed hierarchal classification database. EX-1003, ¶514.

D. Ground 2: The Challenged Claims Are Obvious Over Rompaey Alone Or In Combination With CoWare I.

1. Rompaey Overview

Rompaey discloses or renders obvious the Challenged Claims. Rompaey's CoWare system outputs hardware and software based on a designer's selected specifications. EX-1004, Abstract; *see also* EX-1004, 6:6-18.

Using CoWare, a system designer selects a device specification and then selects “models” from libraries with categorized hardware and software pre-existing components. The designer refines the device design interactively by selecting the functions, parameters and characteristics to implement the designed system / device. CoWare analyzes the selections and converts the design into hardware circuit designs and software which it then outputs. Thus, Rompaey discloses the Challenged Claims.

2. Rompaey Discloses The Challenged Claims Alone Or in Combination With CoWare I

a. Claim 1 Preamble:

If the preamble is limiting, Rompaey discloses this element. CoWare designs integrated circuits – including the exemplary design of a “pager” device. *E.g.*, EX-1004 at 6:6-18; 8:4-48; 10:53-62; Figs. 1-2, 10-13; claim 6; EX-1003, ¶¶95-99.

i. 1[a]

Rompaey discloses this element. EX-1003, ¶¶100-252. Rompaey's designer selects the device's parameters including the functionality and standard components

(hardware and software) and parameters implementing that functionality.

Claim element 1[a] has two pieces: (a) actions showing selection and (b) the type of functionalities and parameters being selected.

(a) *Rompaey's Selection Actions From A Hierarchal Classification Database*

Rompaey discloses the claimed “selecting” and “receiving” in three ways: (1) a designer-selected system description; (2) selections from database library files of reusable components; and (3) selections with “tools” that refine functionality of the components. EX-1003, ¶¶101-200.

Figure 18 exemplifies CoWare's system design process.

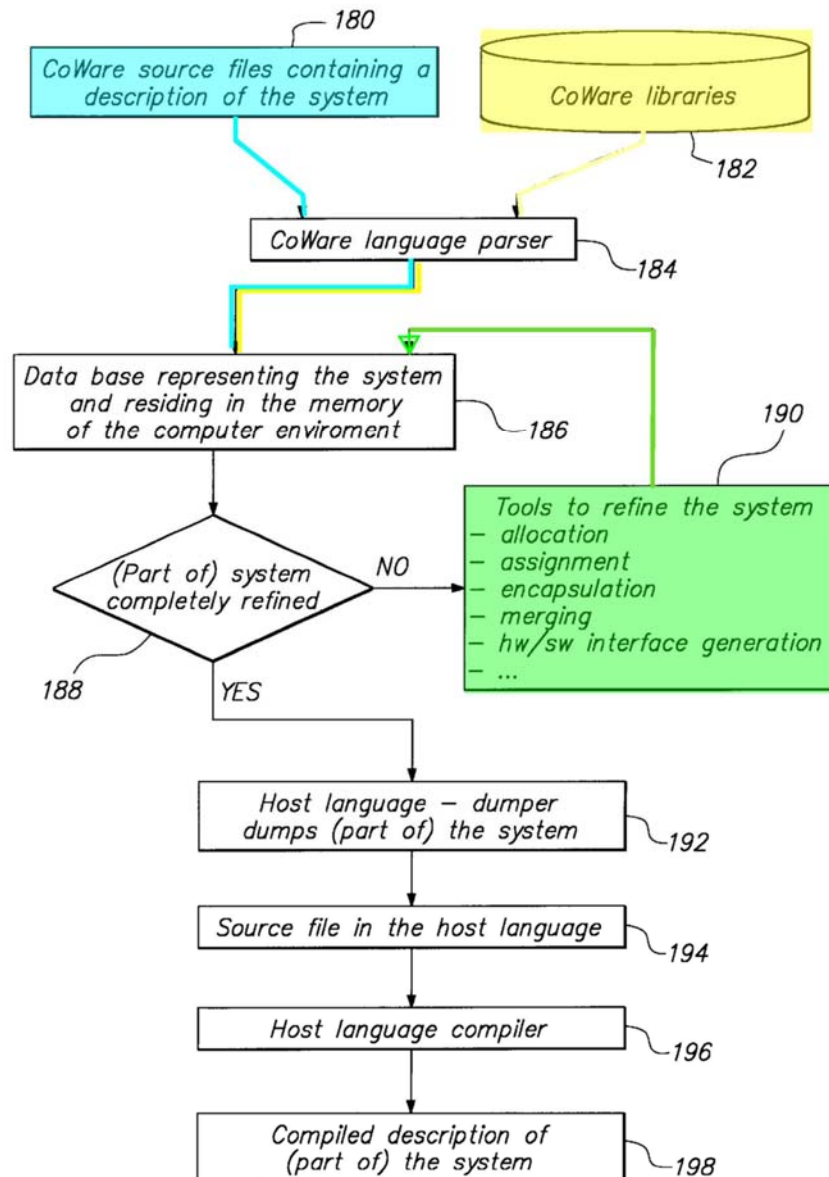


FIG. 18

EX-1004, FIG. 18.

FIG. 18 is an operational flowchart for a presently preferred method of specifying a hardware/software codesign At state 180, the **system designer creates software source files containing a specification of the system. This initial specification is parsed, in conjunction with libraries**

of predefined routines 182, to result in a database of objects which is stored in the computer memory as shown at states 184, 186. The CoWare libraries contain processor models, I/O scenario models, protocol definitions, process definitions for use as hierarchical port or hierarchical channel, and so forth. The database could be construed using the Standards Components Library ... This defines classes for each object in a graphical representation of the system undergoing codesign and has API functions that are used to build graphs, access elements of a graph, ... In addition, the database requires library functions or tools that implement graph algorithms which correspond to the described merger, flattener, and other tools of CoWare.

EX-1004, 26:50-27:3. The blue, green, and yellow blocks / passages exemplify distinct “selecting” and “receiving” actions and are described in detail below.

Fig. 2 also depicts the “co-specification” of the system, the “refinement and optimization” of the system, and the “IP Libraries” shown in Fig. 18 above.

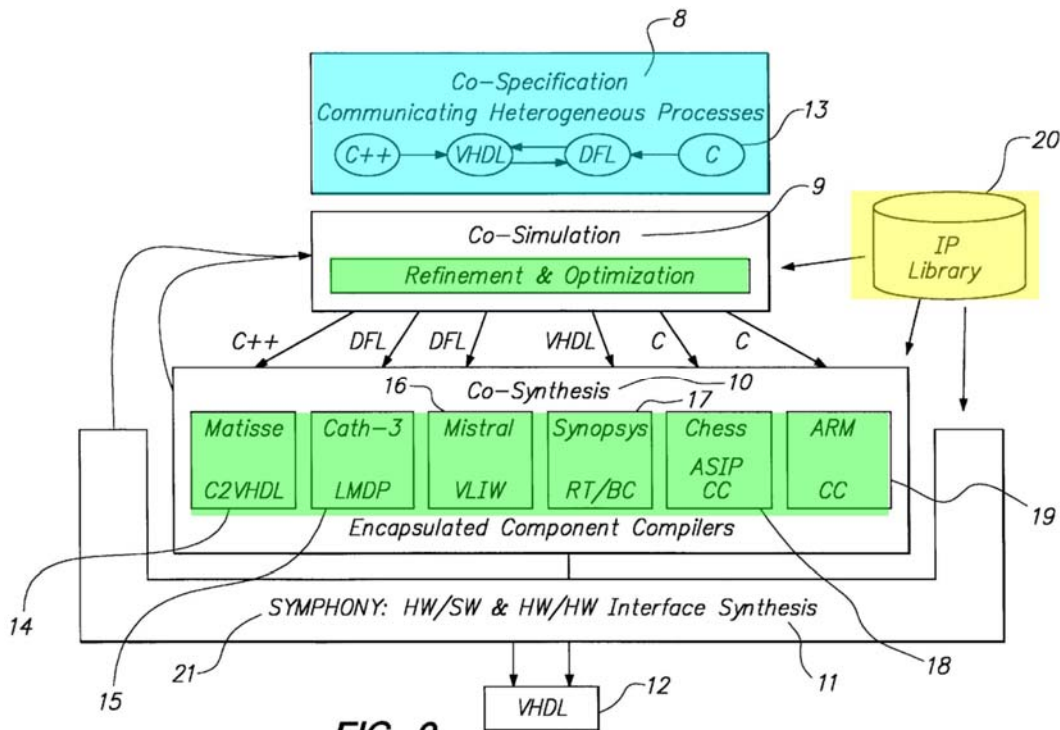


FIG. 2

EX-1004, Fig. 2, 11:12-12:30.

Selection During The Initial Specification.

Rompaey's designer uses a "data model" with classified hierarchical objects selected from a database with "processes," input/output "ports" and "channels" from a database. EX-1004, 11:21-29. Ports and channels define inputs and outputs of processes. Rompaey details selections of function types for components. EX-1004, Figs 3, 4, 5; 11:21-12:5, 12:51-15:49, claims 1, 6-12. The hierarchical categorized objects are used to select and define the device's parameters, including control, display, input and output parameters.

Fig. 10 exemplifies this initial selection of functionality and parameter types for a pager.

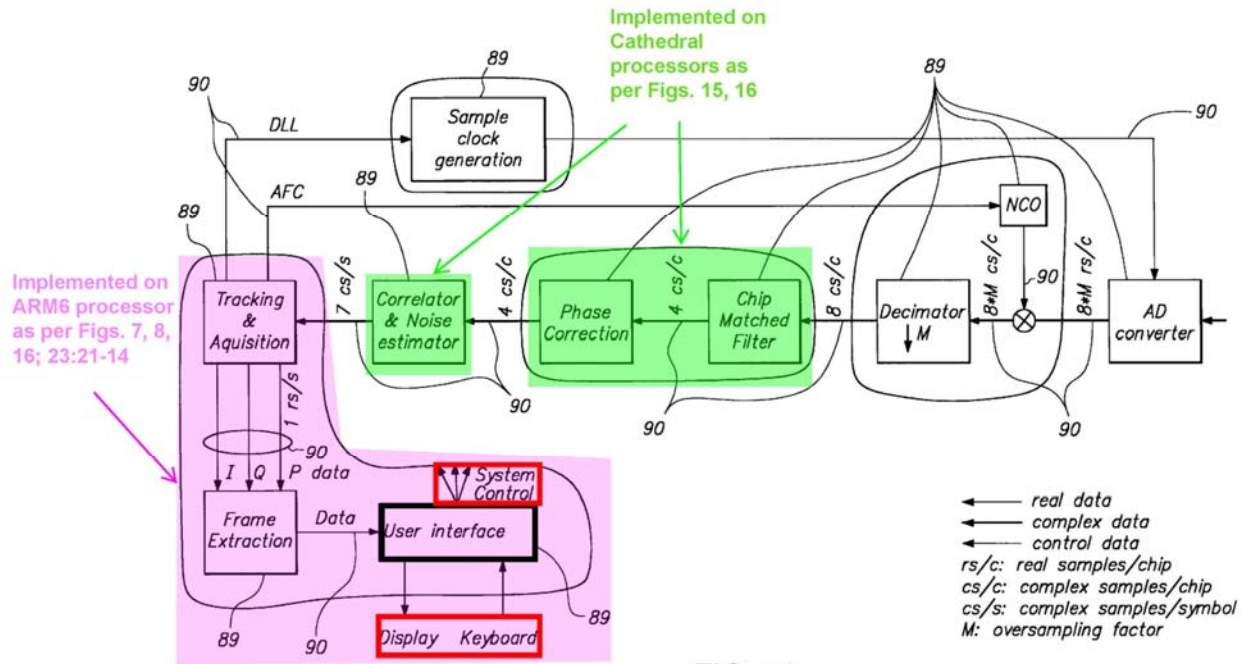
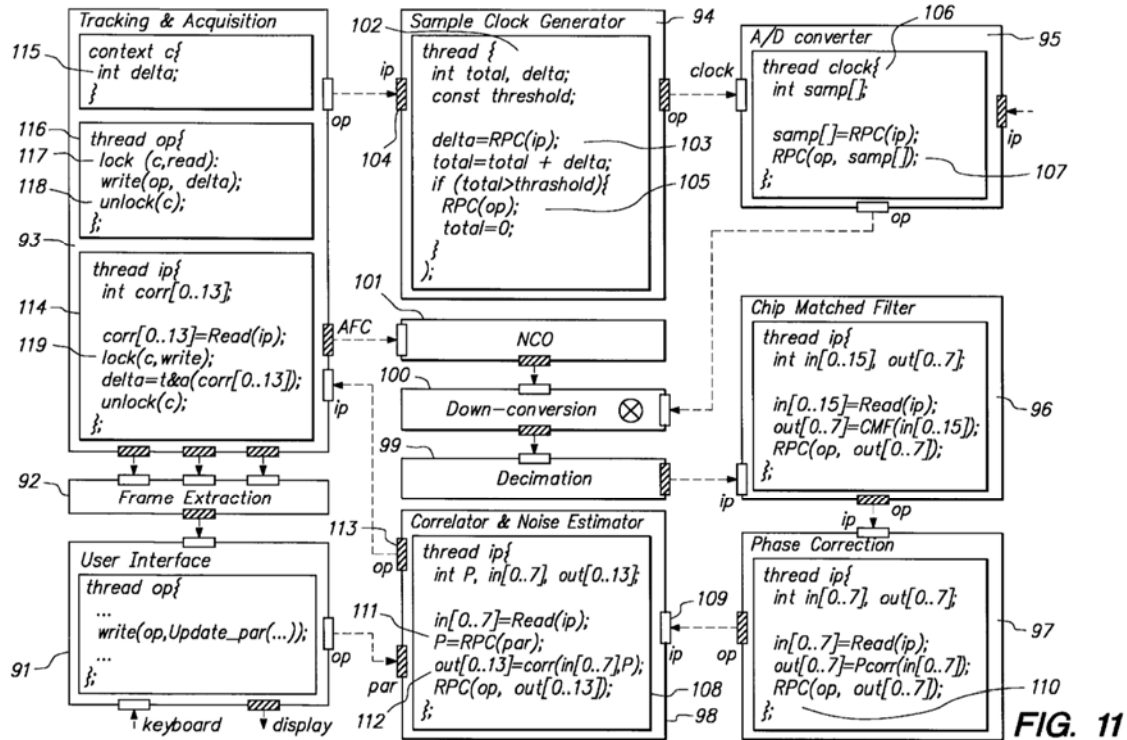


FIG. 10

EX-1004, FIG. 10. The designer selected that the pager has a “user interface” with selected functionality for a “display,” a “keyboard,” and “system control[s]” – which are examples of claimed selections. EX-1003, ¶¶153-154.

The designer selects specific functions and components implementing the block design. Fig. 11 represents Fig. 10. EX-1004, 10:18-20. In Fig. 11, the designer selected predefined blocks to implement display, keyboard, and components for receiving and processing wireless signals (blocks 95-110). EX-1003, ¶¶155-160.



EX-1004, FIG. 11.

Each block (89) in FIG. 10 corresponds to a process implementing a specific function of the pager.... blocks [91-101] in FIG. 11 correspond to the processes (89) from FIG. 10.

EX-1004, 21:34-41. A CoWare designer may select functionalities (parameters) by selecting “processes” in the specification. “Each process is described in the language that is best fit for the characteristics of the function it implements....” EX-1004, 22:26-33. EX-1003, ¶¶157-159.

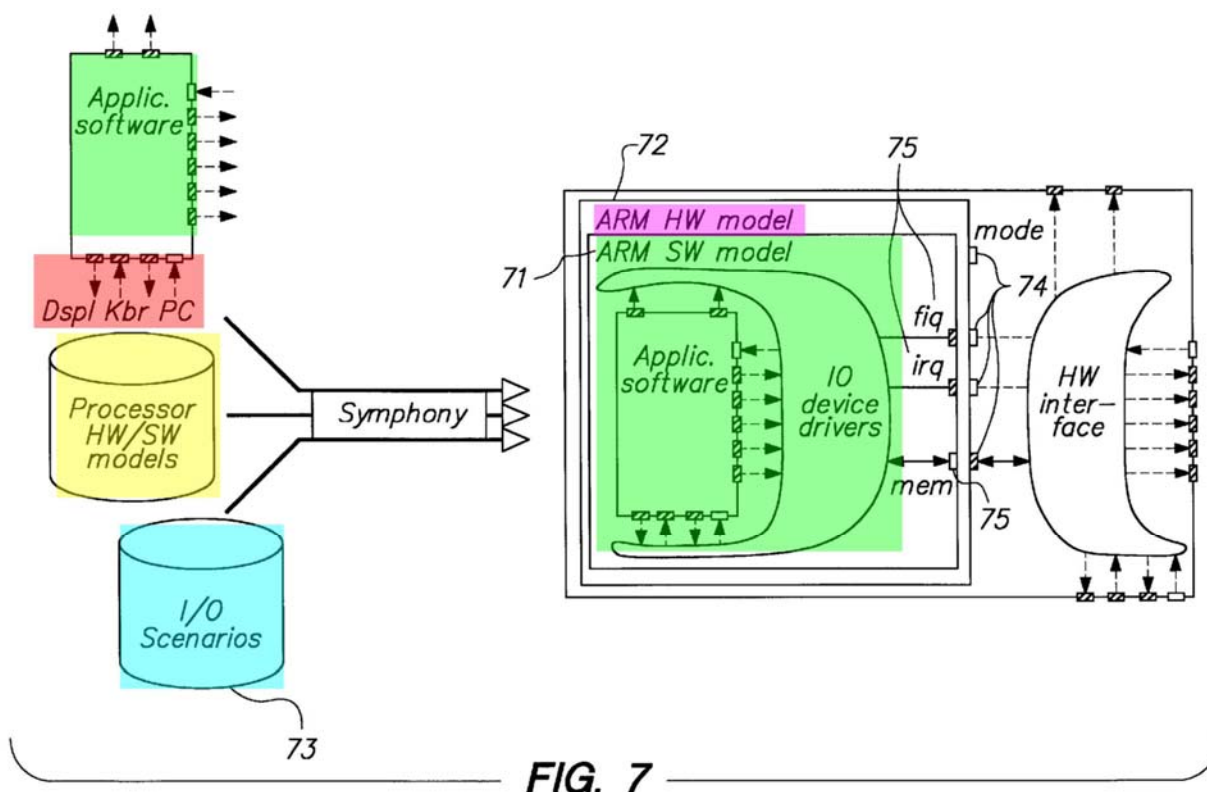
Rompaey’s designer also selects parameters associated with the processes as shown in Figs. 3 and 4. Fig. 3 shows an “encapsulation” that the designer builds by selecting the various subsystems to design the overall system (device) using

channels, ports, and protocols. EX-1004, 12:51-14:57, 9:61-63, Fig. 3.

Selecting The Implementation From Database Libraries.

Rompaey teaches “selecting” the parameters provided by libraries and “I/O scenarios” as shown in Figs. 2 (“IP library”); 6 (“I/O scenarios”); 7 (“I/O scenarios”) and 18 (“CoWare libraries”). EX-1003, ¶¶161-183. “The CoWare database libraries contain processor models, I/O scenario models, protocol definitions, process definitions for use as hierarchical port or hierarchical channel....” EX-1004, 26:56-27:3. These implement the design.

Rompaey’s Fig. 7 exemplifies selecting implementation components:



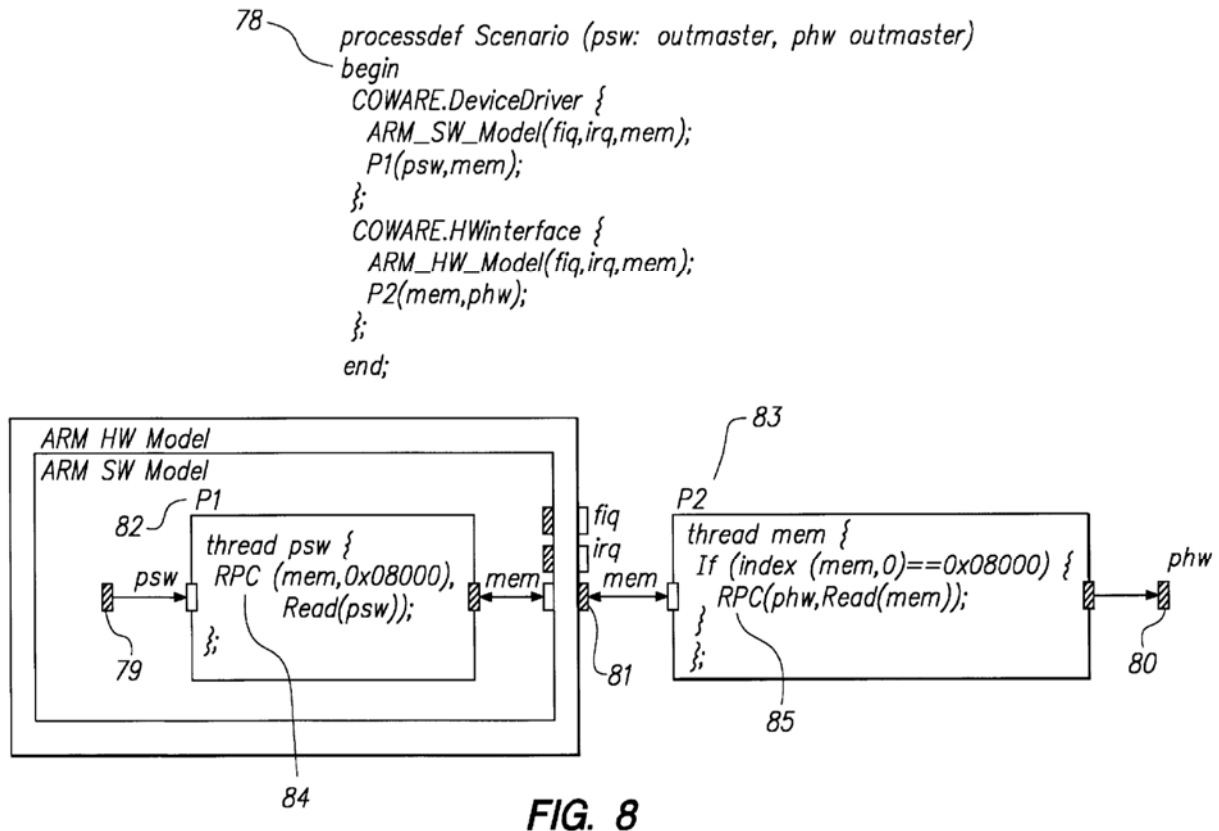
EX-1004, Fig. 7. “FIG. 7 illustrates the tool SYMPHONY. SYMPHONY makes

use of a software model (71) and hardware model (72) of the target processor and a library of I/O scenarios (73) for that processor.” EX-1004, 17:65-18:1. This “makes use” is SYMPHONY selecting the models from the library classification database. Rompaey’s “I/O scenario” example shows a designer’s selection of parameters:

FIG. 8 shows an example of an I/O scenario. It consists of a software host-language encapsulation and a hardware host-language encapsulation that describe a software I/O driver and the hardware counterpart, respectively. An I/O scenario is also tagged with some performance figures that will allow the designer or SYMPHONY to make a decision about what I/O scenario to use for which port.

EX-1004:9-15. This exemplifies how a designer (or CoWare) selects an I/O scenario (from the classes of different I/O scenarios identified below) that is within a hierarchical classification database (libraries that store the I/O scenarios). EX-1003, ¶¶161-172.

Fig. 8 shows selected hardware interfaces and associated software to control input / output hardware selected by the designer for a selected ARM6 processor. EX-1004, 19:10-28.



EX-1004, Fig. 8. In addition, the “fi” and “ir” are interrupts and the “mem” is a “memory-mapped I/O” – each being an input / output parameter along with specific sub-parameters. EX-1003, ¶¶172-173.

“The library of I/O scenario comprises:” (1) “memory-mapped I/O scenarios” that define parameters for an input / output via memory-mapped I/O ports; (2) “Instruction-Programmed I/O scenarios” defining parameters for “special communications ports of the processor ... connected to the external channels via the I/O unit”; and (3) “hardware interrupt controls” that define the parameters of the interrupts (inputs) of different processors. EX-1004, 19:29-20:19. This is three classifications of input / output parameters – each with their own sub-parameters

selected after the selection of the top-level category. EX-1003, ¶¶170-171.

The designer “creat[es]” and “select[s]” various processor models and I/O scenario models from objects in the library / database to implement the designed device using, for example, selected protocols based on libraries of re-usable components. EX-1004, 8:5-38; 12:36-40; 16:50-17:7; 20:10-14; 23:50-24:67 (section entitled “Communication Mechanism Selection”); 25:36-45 (asking designer to select I/O scenarios from categories); claim 31. EX-1003, ¶¶174-182.

Selecting During The Interactive Refinement Process.

Selecting occurs during the interactive refinement process. EX-1003, ¶¶184-186. The designer “refines” the design using “tools” such as “component compilers” that select implementation parameters. EX-1004, 12:11-34.

These “[t]ools, which are independent software functions selectively used by the designer, operate on the database to refine the system representation.” EX-1004, 27:4-7. These tools “access” the library to perform functions (and thus select from a hierarchal database). For example, one tool “input[s]” the “list of available processors” and “allocates” specific types of processors to tasks. EX-1004, 27:25-37. These tools include “allocation,” “encapsulation,” “flattener,” “hw/sw interfacing,” and “hw/hw interfacing” tools. EX-1004, 27:7-28:29. These tools take selected “inputs,” perform processing, and select additional parameters for each input – such as selecting / assigning processor types, or communication channels, or

software such as device drivers. EX-1003, ¶¶185-186.

The selections above are from a hierarchal classification database. EX-1003, ¶¶187-198. In the three “selecting” actions above, Rompaey’s designer (or CoWare) selects from existing functionality arranged in “libraries” (e.g., databases). The source files, libraries, database, and “classes for each object” are “hierarchical” and classified according to functionality. EX-1004, 26:50-27:3. Rompaey teaches different “classes” of objects defining components such as processors, ASICs, hardware accelerators, micro controllers, communications blocks and memory, and peripherals (DMA, UART...). EX-1004, 26:50-27:3; 4:36-49. Within each component category, Rompaey allows for further selection refinement to a particular subclassification of that component (from many) such as an “ARM6” or “Cathedral” processor or a “RS-232” communication block or categorized I/O Scenarios.

Rompaey’s selection uses a “hierarchical” set of different classes of objects and functions to implement different types / categories of device-level functionality. EX-1004, 6:19-34. The objects and functions “are refined into second hierarchical objects for adding implementation detail to the communication behavior...” EX-1004, 7:10-13, 9:10-13. Rompaey’s “data model is hierarchically structured and allows the refinement” of the selected components and functions. EX-1004, 11:31-44. The ‘657 patent used similar “hierarchal” language as Rompaey to describe its selections. EX-1004, Fig. 4, 4:20-23, 6:17-19. EX-1003, ¶¶189-191.

Rompaey's selections are from classified databases to effectuate Rompaey's "reuse" goal which is "key." EX-1004, 4:64-5:2. EX-1003, ¶¶185-188. "In modular designs, the complete system functionality is split into communicating components of manageable complexity. The advantage of this approach is that the components can be reused...." EX-1004, 5:18-21. Rompaey uses "hierarchical" protocols and systems to model "off-the-shelf components" using "a functional specification." EX-1004, 16:66-17:18. This allows for "off-the-shelf" components performing specific functions for the designer to classify, identify and select. EX-1004, 5:42-67, 9:49-51. EX-1003, ¶¶192-193.

Each selection above is received into the claimed "memory of a processor." Rompaey identifies that the entire CoWare tool can be implemented in a single workstation and in a memory. EX-1004, 15:50-61, 26:50-27:23, Fig. 18. EX-1003, ¶¶192. The selections have all been "received" into the "database representing the system and residing in the memory of the computer environment." EX-1004, Fig. 18, element 186. EX-1003, ¶199.

(b) Rompaey's "Parameter" Types

Rompaey teaches the claimed control, display, input, and output parameter types. As the '657 acknowledges, these parameter types are "generic characteristics" to almost every designed device. EX-1001, 4:46-56.

Rompaey describes classes of functional components that may be selected for

the device functionality including “programmable processors, application specific processors with hardwired controller, application specific processors with specialized instruction set, hardware accelerators, micro controllers, communication blocks and memory, peripherals (DMA, UART, and so forth).” EX-1004, 4:38-49. UARTs implement functions for display, input, and outputs for “peripherals” such as displays and keyboards. EX-1003, ¶¶189-191. This list exemplifies Rompaey’s selecting of parameters for control, display, input and output as claimed.

As noted above, the ‘657 describes “parameters” in terms of selection of components that implement different functionality. Rompaey describes the claimed parameters in at least the following manners. EX-1003, ¶¶201-232.

“Control Type Parameters”

Control type parameters include at least the first five components in Rompaey’s list above: programmable processors, application specific processors with hardwired controller, application specific processors with specialized instruction set, hardware accelerators, and micro controllers. EX-1004, 4:37-49. EX-1003, ¶¶207-219. Each component provides control of the device’s functionality – which is its intended purpose. These all represent different “classes” of processors that can be selected to provide different control.

Rompaey gives specific examples of selecting different sub-classes of processors to implement control of different functionality. In the pager example, an

“ARM6” “programmable processor” is selected to control certain pager aspects whereas two “Cathedral” “application specific DSP processors” are selected to control other aspects as shown in Fig. 10 below:

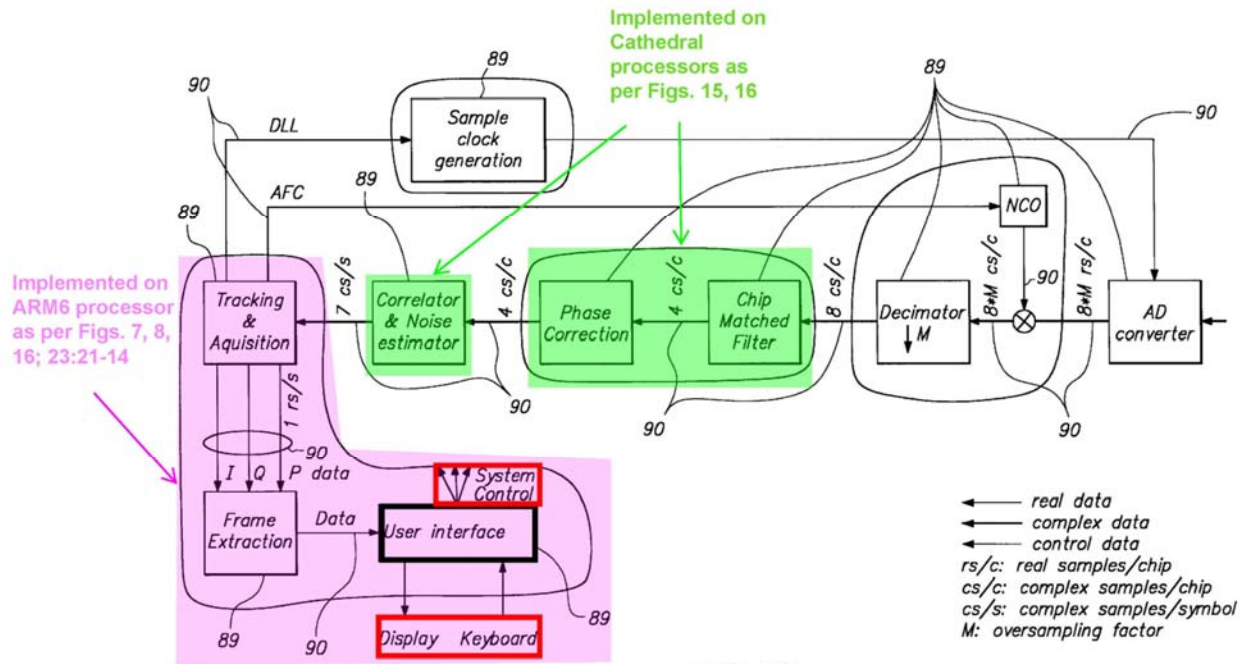


FIG. 10

EX-1004, Fig. 10, 21:34-22:35. Different processors are selected based on functional requirements. EX-1004, 22:41-23:45. Furthermore, for each processor, additional parameters are selected (by the designer or CoWare) to define the specific flow and control of data processing between the processors. EX-1004, 23:46-24:67. EX-1003, ¶¶209-218. For example, different processors may have a selected control parameter of different clock frequencies depending on the implemented functionality. EX-1004, 23:12-19.

Figures 14-16 describe the additional details of the parameters selected for the

Cathedral processors to implement the functionality for which they were selected. EX-1004, 24:27-59, 25:1-26:30.

Also, the description of the types of selections (above) includes Figs. 7-8 (and their associated descriptions) that exemplify the selection of the ARM6 processor type and the associated parameters within that processor type to implement control of the device. EX-1003, ¶¶219, 165-179, 269-270.

“Display Type Parameters”

Rompaey’s pager example shows selecting the functionality of having a “display,” which is a form of selecting of display type parameters associated with the device. Selecting the type of display and the specific implementation are also display type parameters as claimed. For example, in 1995, a POSITA would have known that a pager would typically have had an alphanumeric display using an LCD – both forms of parameters as claimed. EX-1003, ¶¶220-223.

Rompaey’s selectable components include “communication blocks and memory, peripherals (DMA, UART, and so forth).” EX-1004, 4:38-49. Rompaey’s disclosed “display” would be implemented as a selected “peripheral” component.

The “display” interfaces with the User Interface block. EX-1004, Figs. 10-13. Rompaey discloses both “UART” and RS-232 (see Input/Output below) which are forms of interfaces that a POSITA would have known to implement the functionality required for the disclosed “display.” EX-1003, ¶¶222-223, 197-198.

“Input / Output Type Parameters”

The selecting of input and output type parameters is largely detailed in the section describing the “selecting” actions above. EX-1003, ¶¶131-191. The description above includes: (1) selecting I/O scenarios, selecting the type of I/O scenario and selecting aspects / functionality of each I/O scenario and (2) the extensive details above for Figs. 7 and 8 describing the ARM6 functionality to implement the user interface for input and outputs. *See also* EX-1004, 8:5-38; 12:36-40; 16:50-17:7; 20:10-14; 23:50-24:67 (section entitled “Communication Mechanism Selection”); 25:36-45 (asking designer to select I/O scenarios from categories); claim 31. EX-1003, ¶¶173-182. Each is “selecting” the input and output type parameters. Moreover, the ‘657 patent identifies “display” type parameters (discussed immediately above) as also being “output” type parameters.

Using Fig. 4, Rompaey teaches selecting the parameters of an “RS232” port. EX-1004, 14:58-15:49. RS-232 is a well-known input / output component. EX-1003, ¶¶227-230.

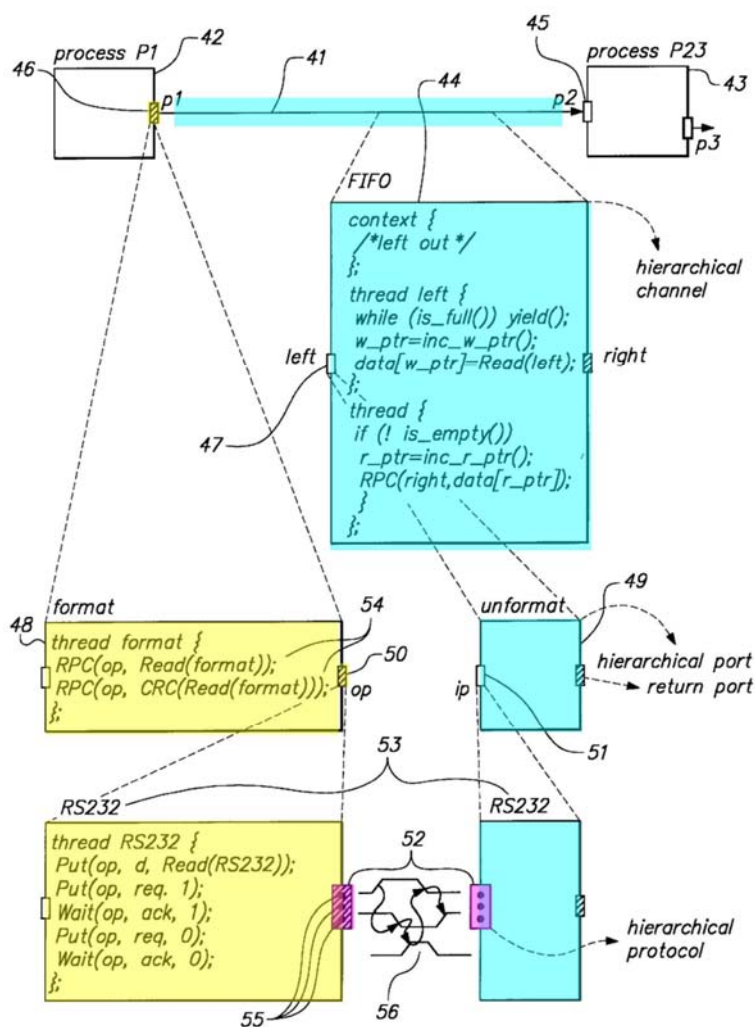


FIG. 4

EX-1004, Fig. 4. Figure 4 shows the hierarchy of the classified data model in which a process (42) includes an **input/output port** (p1) associated with a “**FIFO hierarchical channel (44)**” that selects sub-parameters such as the “protocol,” “data format, data direction and “data rate” implemented through “terminals ... according to a timing diagram.” EX-1004, 15:16-49; *see also* EX-1004, 9:64-67. “Primitive protocols provide a classification of all hierarchical protocols.” EX-1004, 15:29-31. “A primitive port is characterized by a protocol and a data type parameter.” EX-

1004, 12:58-59; *see also* EX-1004, 15:63-67.

Thus, the designer selected the claimed input and output parameters by selecting the port (RS-232) expressed through the “terminals” that perform the input and output according to the selected parameters for that RS-232 (such as protocol, timing and data type). As RS-232 ports implement input and output components such as displays, keyboards and controls such as control buttons on a pager, such designer specifications also are the input / output parameters of the device. EX-1003, ¶¶229-230. *See also* EX-1004, Fig. 1 (element 5); 3:19-25.

Furthermore, the “selection” of a particular processor type may also correspond to selecting an input type parameter because it may define the type of inputs that may be processed. For example, Rompaey specifically describes that a particular “application specific DSP processor” is selected for processing the input wireless signals received into the pager. EX-1004, 24:1-19, 21:43-22:34. *See also* EX-1004, 3:51-67 (describing functionality to process “incoming signals”). This selection can be performed by the designer or CoWare. The types of functions such as “down-conversion” and “decimation” relate to “frequency down- or up-conversion, bit to symbol modulation, data-compress and error correction encoding” (EX-1004, 3:28-32), which all relate to input data signals. Selecting these functionalities and a processor type that corresponds to these functionalities is the claimed selection of input type parameters. EX-1003, ¶¶231-232.

**(c) *Rompaey Renders This Element Obvious
Alone Or In Combination With CoWare I.***

If PO argues that Rompaey does not expressly disclose any aspect of this element, Rompaey renders obvious every aspect of this element. EX-1003, ¶¶234-245. As noted for Banerjee, PO asserts the claimed parameters are met by generically described functions of the IC, including “subsystems, chips, or boards.” Section VII.C.2.a.i, *supra* (quoting EX-1008, 000024-25). As detailed above, Rompaey discloses designing integrated circuits and the associated device. EX-1003, ¶¶235-239.

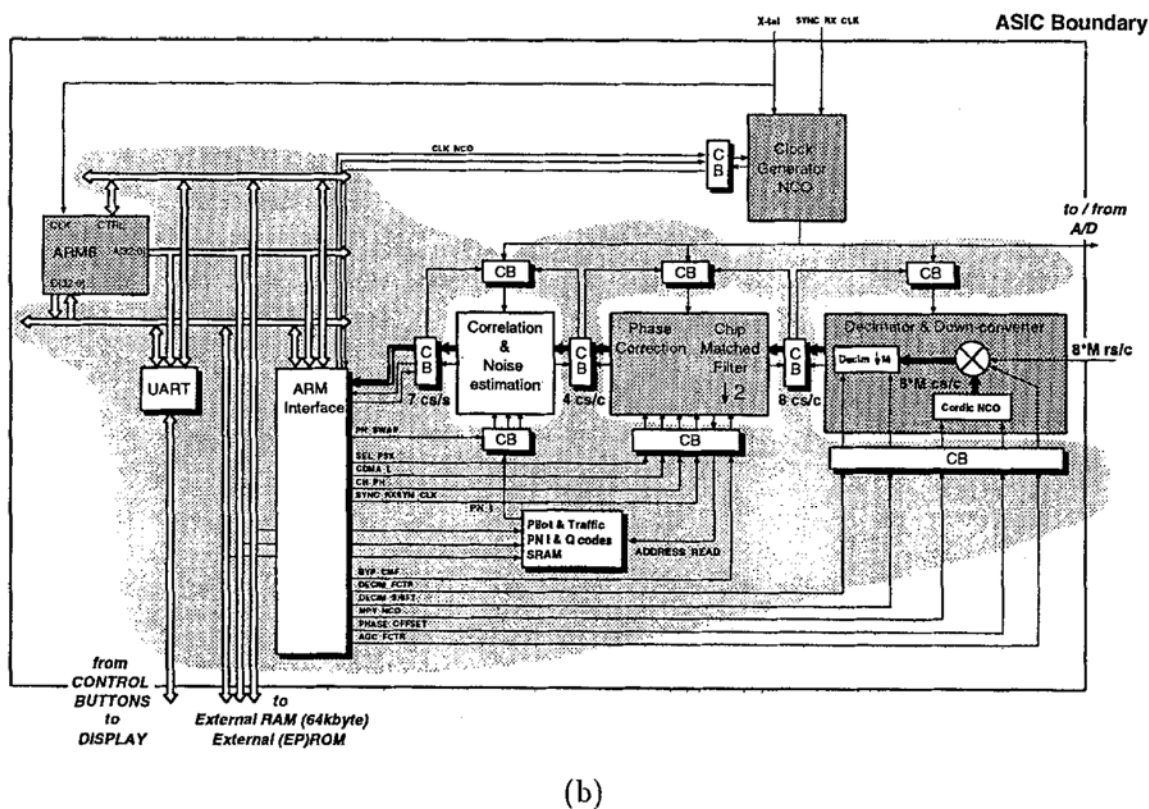
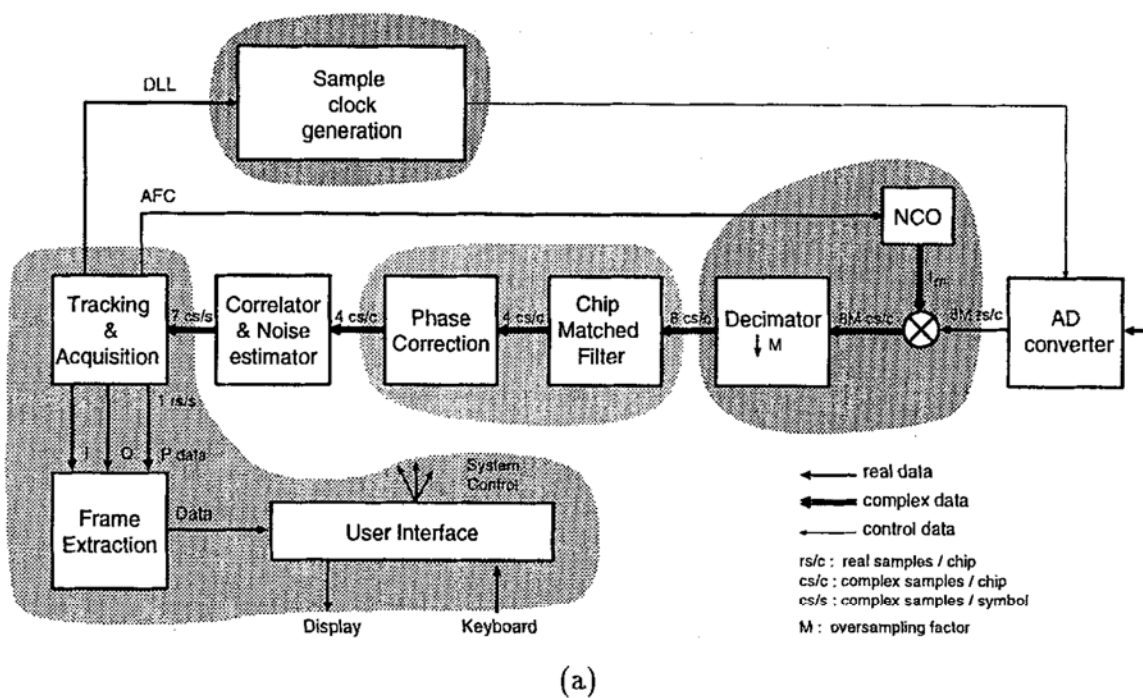
Also, Rompaey discloses (and claims) specifying parameters of “application specific integrated circuits” (and programmable processors). EX-1004, 4:36-49, claim 37. Such “application specific” integrated circuits (ASICs) operate as the name states – to be specific to a particular application – that defines the parameters of that device. *E.g.*, EX-1004, 1:18-45. For example, the “I/O scenarios” implement specific components used as the display, inputs and outputs of the device. EX-1003, ¶¶240-241.

Rompaey labels the pager “device” to be a “pager application.” EX-1004, 10:16-41, 21:29-31. There is no patentable distinction between selecting the parameters of Rompaey’s pager device and its associated “application specific integrated circuits,” and any selecting of “device” parameters. EX-1003, ¶242.

Similarly, if PO argues that Rompaey does not disclose selections from a

hierarchal classification database, Rompaey renders obvious such a database. As noted above, Rompaey stores components and the related parameters and functionality in a library (database) arranged in a fashion to be re-used. Such re-use would require classification and a hierarchal classification sorted by functionality (e.g., processors, I/O) is an obvious design choice. At a minimum, Rompaey's "hierarchical" classification renders obvious the claimed selecting and receiving of parameters. EX-1003, ¶¶243-244.

Rompaey combined with CoWare I (EX-1007) further discloses this element. EX-1003, ¶¶246-252. CoWare I is an IEEE paper that largely mirror's Rompaey's Figures and disclosure. CoWare I includes the following Figure:



EX-1007, Figs. 6(a), 6(b). Figure 6(a) is a copy of Rompaey's Fig. 10. Fig. 6(b)

adds relevant disclosure. First, Fig. 6(b) shows that the pager has “Control Buttons” that are part of the input / output for the device. These control buttons are essentially identical to the “buttons” identified in the ‘657 patent as corresponding to input parameters. Second, Fig. 6(b) shows that the “UART” is a component that is selected for the display, inputs and outputs (display and buttons). Rompaey discloses a “UART” as one of its categorized selectable functionalities. EX-1004, 4:38-49. The UART selection and refinement of its parameters and functionalities would be an example of a I/O scenario (and refinement of its specific functionalities) discussed above. EX-1003, ¶¶248, 197-198.

A POSITA would be motivated to combine Rompaey and CoWare I with a reasonable expectation of success. The references show largely the same content, have the same pager example, are authored by the same people, were filed / published contemporaneously, and the inventors submitted CoWare I to the PTO while prosecuting Rompaey. EX-1003, ¶¶249-251.

ii. 1[b]

Rompaey discloses this element. EX-1003, ¶¶254-274. CoWare analyzes the designer’s selections to generate hardware circuits and software. This is depicted in Figures 2, 6, 7, 11, 18 (showing the receiving of several different forms of input with the claimed parameters including input of libraries, scenarios, and models that include elements 180, 182, 186, and 190 in Figure 18).

Fig. 2 (*supra*) exemplifies the claimed analyzing in the blocks that perform “co-synthesis 10” as well as the “SYMPHONY” tool. The co-synthesis analyzes the selections to map the design onto hardware and software components. EX-1004, 12:12-43. EX-1003, ¶¶255-256.

Fig. 6 shows another example of “analyzing” in the “implementation process for hardware / software co-design according to ... FIG. 2.” EX-1004, 10:3-5.

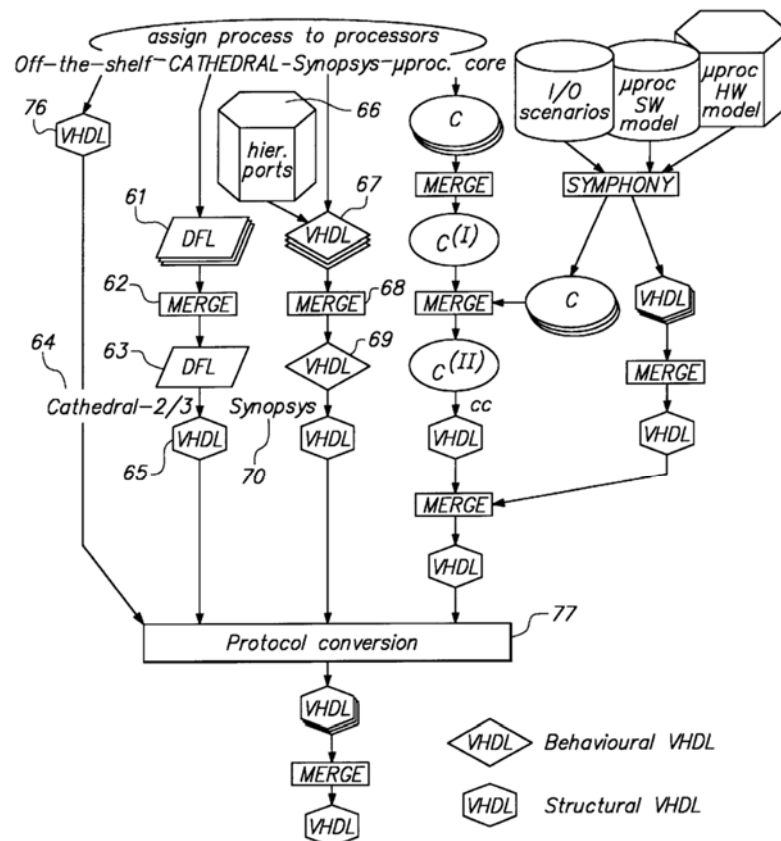


FIG. 6

EX-1004, Fig. 6. The “merge” and “assign process to processors” blocks are also the claimed “analyzing.” Rompaey’s “merge” process analyzes the design and

libraries to implement the designer's specification. EX-1004, 17:19-58; 22:51-24:63; 25:1-26 (and Fig. 15). EX-1003, ¶¶257-265.

“SYMPHONY” also analyzes the selections. SYMPHONY receives the selected “I/O scenarios” and selected hardware and software models.

The hardware interface and the software I/O device driver is generated automatically with SYMPHONY. To generate these interfaces, SYMPHONY analyses the ports of the software process. For each of these ports, SYMPHONY scans the library of I/O scenarios for an applicable scenario. The user is asked to select the most appropriate scenario amongst the applicable ones. SYMPHONY then combines the selected scenarios into a software I/O device driver and a hardware interface.

EX-1004, 25:36-44. *See also* EX-1004, 17:49-63; 12:37-43. EX-1003, ¶266.

“FIG. 17 is an illustration of the generation of an exemplary hardware/software interface between a software subsystem compiled on an ARM processor core and a hardware subsystem.” EX-1004, 10:43-46. Figure 17 shows two ports implemented by the SYMPHONY analysis (shown in Fig. 7). EX-1004, 25:45-26:49.

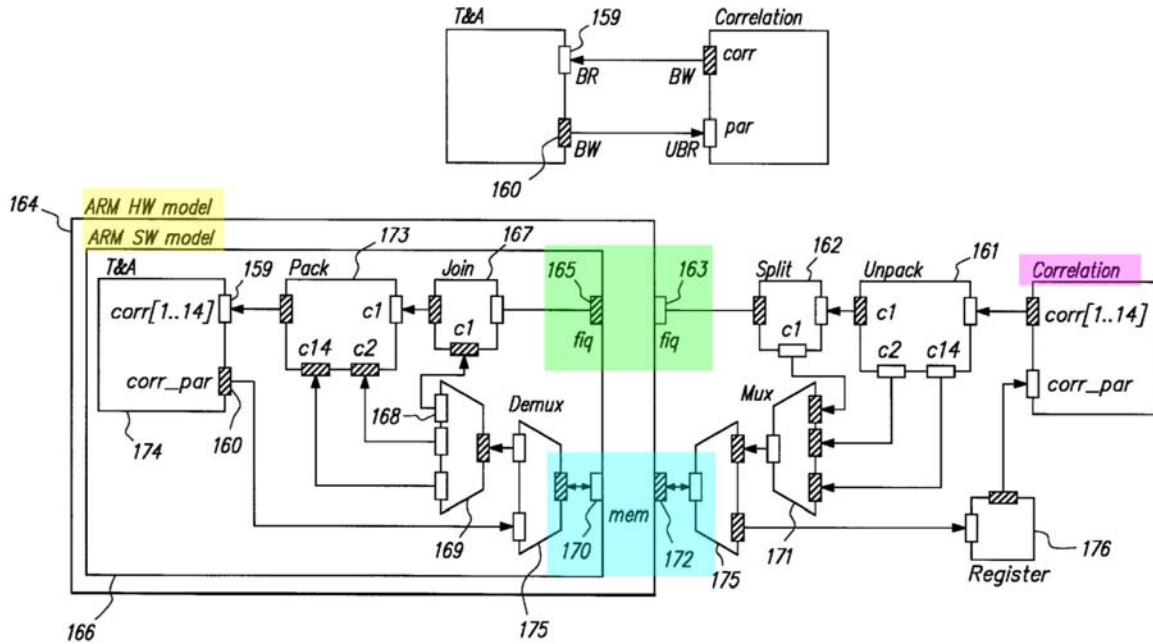
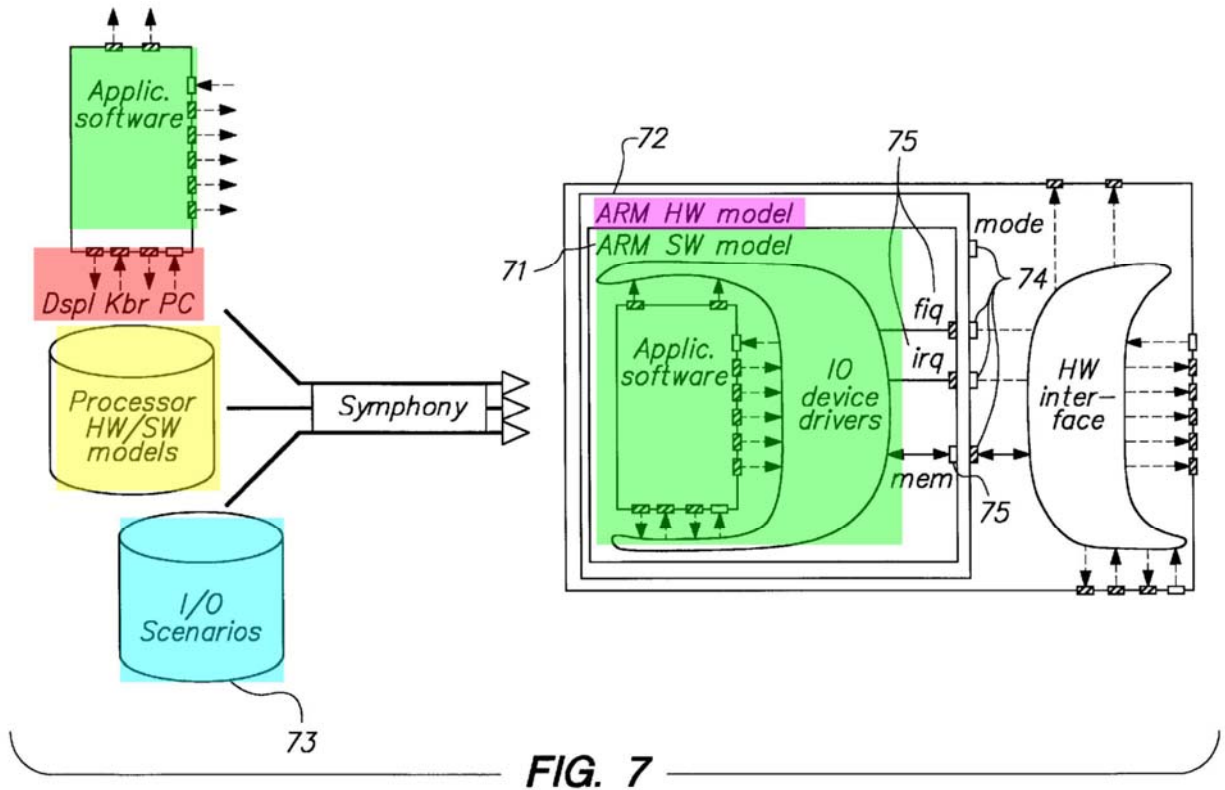


FIG. 17

Fig. 17 shows the input/output on two ports (**fiq** and **mem**) to the “**correlator**” which is used for the input / output of the wireless signals of the pager. EX-1004, 21:60-22:35, 25:45-25:49. These ports have various parameters such as protocols, data rates, and data directions. EX-1003, ¶¶267-268.

Fig. 7 shows SYMPHONY’s analysis of the selections to create a hardware integrated circuit design (“**ARM HW model**”), “**IO device drivers**” **software**; and “**Applic[ation] software**” for the device (pager) that includes the display, keyboard, and program controls.



EX-1004, Fig. 7.

Rompaey claims this process of analyzing the selected parameters to output hardware and software for the designed device. EX-1004, claims 1, 6-8, 10-25, 27-36. EX-1003, ¶¶269-270.

If PO argues that Rompaey does not expressly disclose this element, Rompaey renders obvious every aspect of this element. EX-1003, ¶¶271-274. The ‘657 patent does not use “analyzing.” As detailed above, Rompaey extensively describes linking functional parameter libraries of existing ICs implementing the functionality.

PO asserts that the “*analyzing*...” occurs when a designer “uses the [] [d]esign [t]ools to define the desired parameters and configure the selected IP and create ...

RTL, an IP microarchitecture, and executable files for the integrated circuit” EX-1008, 000029. As detailed above, Rompaey discloses analyzing to create and output the RTL or VHDL defining the IP microarchitecture and the executable files (the application software and device drivers) for the pager and thus renders this element at least obvious. EX-1003, ¶¶271-273.

iii. 1[c]

Rompaey outputs the generated hardware subsystem (integrated circuit design) in well-known formats including HDL, VHDL, and netlist. EX-1003, ¶¶277-288. EX-1004, 11:12-20 (“FIG. 2 shows the output 12 may include a net list for prior-art commercial tools for the generation of the implementation layout. This output preferably comprises structural VHDL or Verilog and machine code for the programmable processors.”). *See also* EX-1004, 17:46-48 (“The result is a VHDL net list (65) of the implementation....”); 20:32-38 (“VHDL net list” output); 28:22-30 (“the system designer can ... generate a compiled description of the system ...as shown in state 198. This compiled description may be ... a net list describing the final implementation of the system.”); 1:53-59; 4:19-29; 5:14-16, claims 38, 44; Figs. 1-2, 6-7, 9, and 18. The hardware output includes RTL code. EX-1004, 26:38-42; 3:35-39. EX-1003, ¶¶277-285.

Rompaey teaches generating software for execution on well-known “off-the-shelf” processor models, such as “Cathedral” and “ARM6.” The designs would

disclose, or render obvious, a communications bus because such buses are well-known and pervasive throughout processors and would be needed to effectively operate the software. EX-1003, ¶¶286-287.

Moreover, SYMPHONY generates the hardware and software to effectuate the inputs and outputs detailed in element 1[b]. *E.g.*, EX-1004, 19:1-20:25. These scenarios included use of “data that is residing on the memory data bus of the ARM [processor].” EX-1004, 19:1-29. EX-1003, ¶¶286-287.

Furthermore, “SYMPHONY adds RAM and ROM blocks to store the program code and data”). EX-1004, 20:22-25. The teaching of RAM and ROM to store the software and data teaches, or renders obvious, that the design includes a communication bus to facilitate communications between the processor and such memories. EX-1003, ¶¶286-287.

If PO argues that Rompaey does not expressly disclose this element, Rompaey renders obvious this element. As detailed above, Rompaey’s teachings are devoted to converting the design of an “application” (such as the “pager application”) into the hardware integrated circuits and software implementing that design. EX-1003, ¶288.

iv. 1[d]

Rompaey discloses that CoWare outputs software programming commands to be executed by the hardware circuit. EX-1003, ¶¶290-300. The discussion in

element 1[b] above regarding “SYMPHONY” highlights this output of software specific to the designed software. *See also* EX-1004, 20:22-25.

As noted above, Fig. 7 shows the hardware output and the associated “applic. Software” and “IO device drivers” to interface with the display, keyboard, and program controls. *See* EX-1004, 6:45-53 (“... The software subsystems comprise machine instructions for the hardware subsystems.”). *See also* EX-1004, at Figs. 2, 6-7, 9, 18, 25:29-26:49; 8:4-38; 26:14-30; claims 6-8, 27-37; TITLE; Abstract. EX-1003, ¶¶295-296.

Rompaey describes output of “I/O device driver” software. EX-1004, 12:35-43, 17:59-64, 18:31-46, 19:10-28; 26:38-49; 27:61-64. “The I/O device driver processes are merged with the original software application code” controlling the device. EX-1004, 26:42-43. EX-1003, ¶296.

CoWare’s outputted software used a communication protocol compatible with the designed device through the communication bus of the designed circuit based on the selections of parameters shown in element 1[a] to properly operate the software. EX-1003, ¶¶297-298.

As detailed above, part of SYMPHONY’s function was to determine the protocols for communications between different blocks and ensure compatibility of those protocols. For example, the software stored in RAM or ROM necessarily uses

the compatible communication protocols for the bus connecting the memories and processor execution units in order to execute properly.

If PO argues that Rompaey does not expressly disclose this element, Rompaey renders obvious this element. Rompaey's teachings are devoted to converting the design of an "application" (such as the "pager application") into the hardware integrated circuits and software implementing that design. EX-1003, ¶299.

b. Claim 5

Rompaey discloses using predetermined hardware integrated circuit components throughout its disclosure of libraries, databases, and models. EX-1003, ¶¶301-307. As detailed in 1[b] above, Rompaey emphasizes re-use with "off-the-shelf" components. EX-1003, ¶¶301-305.

Furthermore, Rompaey discloses predetermined components such as processors, including the "ARM6" and "CATHEDRAL" processors. EX-1004, Figs. 6-8, 15, 17; 23:12-45, 20:32-38; 25:14-15. *See also* EX-1004, 26:50-27:3. The "I/O scenario models" (*see* Figs. 4, 8) are also predetermined hardware integrated circuit components. EX-1003, ¶¶306-307.

c. Claim 6

As detailed for element 1[c], CoWare generates hardware integrated circuits in formats including HDL, VHDL, netlist, and RTL which discloses this element. EX-1003, ¶¶308-309. *E.g.*, EX-1004, 9:19-21 ("generating a net list comprising the

layout information of the implementation”); 9:44-51; 11:15-20; 12:48-50; claims 38, 46.

d. Claim 8

Element 1[a] details CoWare and the database(s) used by CoWare including the libraries. CoWare and its databases are not described as any part of a fabrication process. Rompaey does not link CoWare with any foundry or fabrication. CoWare is “fabless.”

The ‘657 patent notes that “Fabless design of ICs and MCUs is well known in the art....” EX-1001, 10:9-13. Thus, given this “well known in the art” element, the decision to use CoWare either separately or integrated with a specific foundry is, at most, an obvious design choice. At a minimum, Rompaey renders this element obvious. EX-1003, ¶¶311-317.

e. Claim 9 Preamble

Rompaey discloses this system. *See* 1[Preamble]. EX-1003, ¶¶318-337.

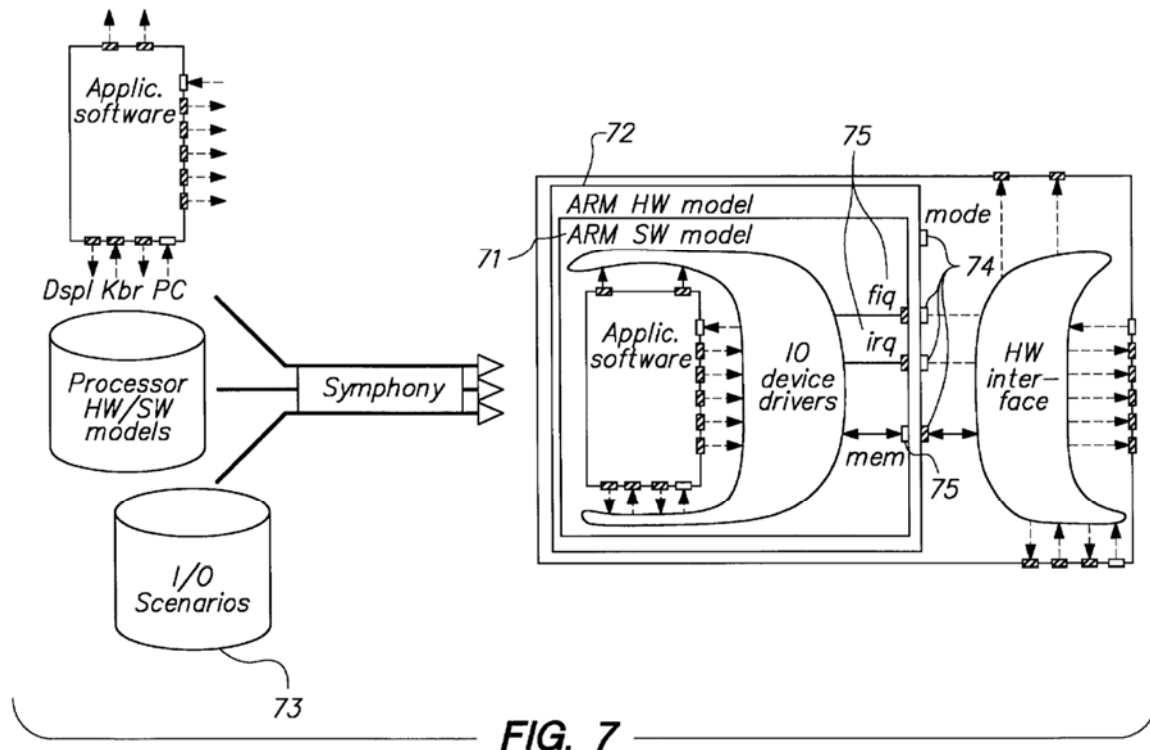
i. 9[a]

Rompaey’s CoWare design environment is implemented on a computer executing software, thus disclosing (including rendering obvious) a processor. EX-1004, 15:50-62; EX-1003, ¶¶319-324. *See also* EX-1004, 7:32-8:3, 26:50-27:26; Fig. 18; claims 1-33 (all claiming a tool in a “computer environment”).

Element 1[a] details the claimed “design user interface” provide by CoWare.

ii. 9[b]

As detailed above in 1[preamble], 1[a], and 9[a], CoWare was software that executed on processors in workstations. EX-1003, ¶¶325-330. Rompaey uses the well-known symbol for disk drives associated with its workstation executing the software.



EX-1004, Fig. 7. See also EX-1004, Figs. 2, 6, 18. A POSITA would have understood that CoWare software for the workstation would have been logic encoded / stored on non-transitory media such that the software could be used after the workstation is powered off / on. Disk drives provide the mechanism for software storage. EX-1003, ¶¶328-330.

This discloses, or renders obvious, this element.

iii. 9[c]

CoWare receives a selection of claimed parameters as detailed in 1[a]. EX-1003, ¶¶331-332, ¶¶101-253.

iv. 9[d]

As detailed in 1[a], 1[b], and 1[c] above, CoWare designs and outputs hardware integrated circuits based on the selections in element 1[a]. EX-1003, ¶¶333-335, ¶¶275-289; *see also* 1[b], 1[c], 5.

v. 9[e]

As detailed in 1[d], Rompaey discloses outputting software as claimed based on the selections in 1[a].

f. Claim 15

As detailed in 1[b], Rompaey discloses simulating the hardware design. EX-1003, ¶¶338-343; EX-1004, Abstract (“A hardware and software co-design environment and design methodology based on a data-model that allows one to specify, **simulate**, and synthesize heterogeneous hardware and software architectures...”). Figure 9 depicts simulation at phase (4):

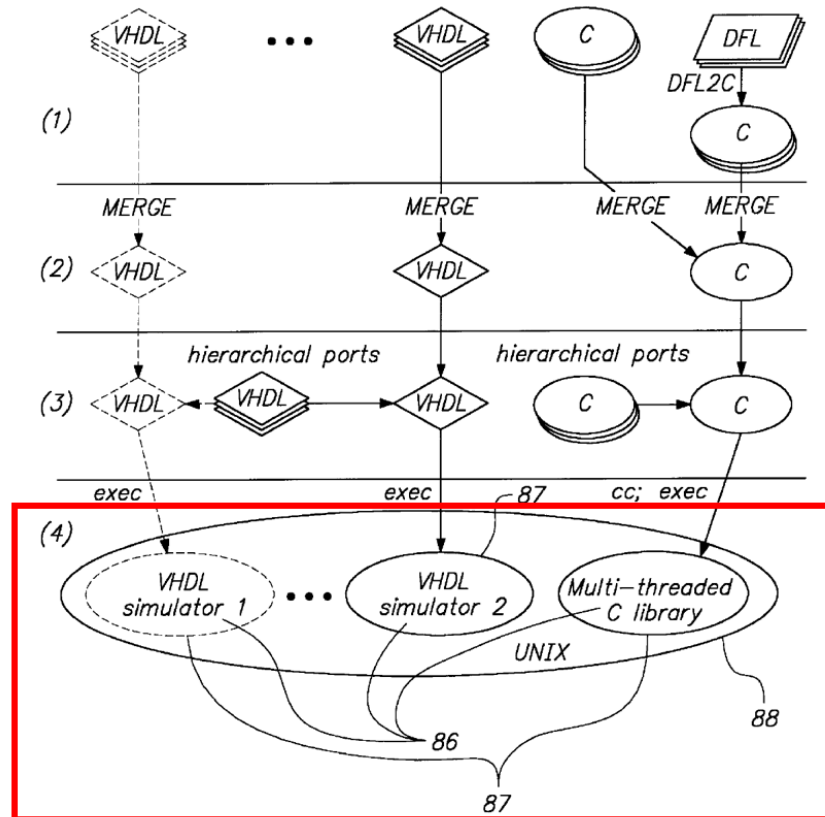


FIG. 9

EX-1004, Fig. 9; 10:13-15; 6:36-34; 7:32-8:3; 9:14-15; 11:12-20; 12:6-11; 18:63-67; 20:39-21:28; 22:38-40; 23:51-56; 22:65-67; 28:13-30; claims 2-5, 27, 35.

g. Claim 16

As detailed in 1[a], CoWare uses databases, including libraries and repositories, for a user to select classes and subclasses of components and then to select parameters, attributes, and characteristics for the components. CoWare teaches, or renders obvious, the claimed hierarchal classification database. EX-1003, ¶¶345-346.

VIII. THE BOARD SHOULD NOT DENY INSTITUTION UNDER SECTION 314

A. The Board Should Not Deny Institution Under 35 U.S.C. § 325(d)

The ‘657 patent has not been previously challenged in an IPR. The Examiner did not consider the references herein and the references present compelling merits. PO asserts that the claims’ scope encompasses well-known design tools going back to the 1980’s as shown above.

B. The Board Should Not Deny Institution Under 35 U.S.C. § 314(a)

Under the Director’s June 2022 interim guidance,⁶ this Petition is particularly strong in the underlying merits as it demonstrates that the exact functionality and structure recited in the Challenged Claims existed in the prior art. Moreover, the asserted prior art shows exactly the functionality that the PO asserted was patentable to the Examiner. Thus, under the Interim Guidance, the Board should not deny institution.

⁶ Interim Procedure For Discretionary Denials In AIA Post Grant Proceedings With Parallel District Court Litigation, PTO Director’s Memorandum (June 21, 2022).

Nevertheless, the Board's decision in *Sotera Wireless, Inc. v. Masimo Corp.*, IPR2020-01019, Paper 12 (Dec. 1, 2020) instructs that a holistic view of the remaining *Fintiv* factors also weighs in favor of institution.

Factor 1 is neutral because no request for stay has been filed.

Under factor 2, the scheduled Litigation trial date is March 3, 2025, but that is not a date certain. For example, the Judge assigned to the Litigation has two patent trials scheduled in February 2024. The first case (*Sable Networks*) was filed in March 2021 and its initial trial date (3/6/2023) was slipped to 6/12/2023, 11/6/2023, and finally 2/5/2024. EX-1016, 000015, 000019, 000025, 000041. The second case (*Demaray*) was filed in March 2020 and its initial trial date (12/27/2021) was slipped to 4/4/2022, 5/8/2023, 9/11/2023, 1/22/2024, and 2/12/2024. EX-1017, 000036, 000043, 000047, 000058, 000059. A Final Written Decision is expected in the present matter in August of 2025 – which would be before any trial if trial schedules similar to the currently scheduled trials (*Sable Networks* and *Demaray*) occur. Thus, this factor favors institution.

Factor 3 favors institution. The schedule (EX-1015) required binding Infringement Contentions on September 12, 2023. Thus, this Petition is filed approximately six months of Petitioners knowing the asserted claims. Merits discovery has not opened, and the *Markman* hearing is scheduled for February 27, 2024. Fact and expert discovery close September 12, 2024, and November 7, 2024,

respectively. EX-1015. Thus, the Litigation is in its early stages, and the investment has been minimal.

Under factor 4, there will not be complete overlap in the issues raised in the IPR and the Related Litigation. Petitioners challenge all claims asserted in the Litigation, but PO has asserted 13 claims from two patents in the Litigation – an amount that likely exceeds the number of claims allowed at trial. Given that the patentee will likely be required to reduce the asserted claims, it is highly unlikely the district court will address the validity of all Challenged Claims. Petitioner stipulates that it will not pursue invalidity against the asserted claims in the district court using the combination of prior art references in the grounds in this Petition for establishing obviousness. *Sand Revolution II LLC v. Continental Intermodal Group-Trucking LLC*, IPR2019-01393, Paper 24 at 11-12 (PTAB June 16, 2020) (holding a similar stipulation weighs against discretionary denial)

Under factor 5, Petitioner is a defendant in the Related Litigation. Therefore, this factor favors institution.

Under factor 6, other circumstances heavily weigh in favor of institution. Here, the Petition's merits are particularly strong. For example, each asserted reference directly and unambiguously teaches an identical architecture and the identical concepts disclosed and claimed in the '657 patent, demonstrating that the

Petition is particularly strong in the underlying merits. Each asserted reference discloses what PO asserted was patentable during prosecution of the parent.

When viewed holistically, the Petition's timing is reasonable, there has been relatively limited investment in the Litigation, and there is minimal overlap between the present IPR and the Litigation. Coupling these *Fintiv* considerations with the compelling evidence of unpatentability presented in the Petition, the efficiency and integrity of the IPR process is best served by instituting review.

IX. MANDATORY NOTICES UNDER 37 C.F.R. §42.8

A. Real Party-in-Interest (37 C.F.R. § 42.8(b)(1))

The real party-in-interest in this Petition is Arm Limited.

No entity other than Petitioner is funding, controlling, or otherwise has an opportunity to control or direct this Petition or Petitioner's participation in any resulting IPR. No entity other than Petitioner has reviewed this Petition. None of the entities identified above as potential parties-in-interest have agreed to be named as real parties-in-interest. In any event, because this Petition is being filed within one year of the filing of the Complaint in the Litigation (EX-1014), and this is the only petition that has ever been filed against the '657 patent, there is no time bar, estoppel or anything else material to the case relevant to the naming of real parties-in-interest, and thus the party-in-interest issue need not be addressed by the Board

Petition IPR2024-00566 for *Inter Partes* Review of U. S. Patent No. 9,367,657

as part of its institution analysis. *See Sharkninja Operating LLC, et al. v. iRobot Corp.*, IPR2020-00734, Paper No. 11 (PTAB Oct. 06, 2020).

Petitioner separately notes that Petitioner has other corporate subsidiary entities, including Arm Inc., each of which agrees to be estopped under the provisions of 35 U.S.C. § 315 to the same extent that Petitioner is estopped.

B. Related Matters (37 C.F.R. § 42.8(b)(2))

1. Judicial Matters

As of the filing date of this Petition and to the best knowledge of Petitioner, the ‘657 patent is involved in the following litigations:

- *ICPillar LLC v. ARM Ltd, et. al.*, Case No. 1:23-cv-00282 (WDTX) (the “Litigation”).

2. Administrative Matters:

As of the filing date of this Petition and to the best knowledge of Petitioner, the ‘657 patent has not been subject to any petitions for *Inter Partes* Review.

Petitioner has filed IPR2024-00476 which challenges the related U.S. Pat. No. 8,924,899 using the same prior art as herein.

3. Related Patents

To the best of Petitioner’s knowledge, the ‘657 patent is related to the following United States Patent Nos.: 8,924,899 and 10,216,174.

C. Lead/Back-up Counsel (37 C.F.R. § 42.8(b)(3))

Lead Counsel:

Kevin P. Anderson, Reg. No. 43,471

kpanderson@duanemorris.com

DUANE MORRIS LLP

901 New York Ave NW 700 E

Washington, D.C. 20001

T: (202) 776-5213

F: (202) 776-7801

Back-Up Counsel:

Patrick D. McPherson, Reg. No. 46,255

PDMcPherson@duanemorris.com

DUANE MORRIS LLP,

901 New York Ave NW 700 E

Washington, D.C. 20001

P: (202) 776-5214

F: (202) 776-7801

D. Notice of Service Information (37 C.F.R. § 42.8(b)(4))

Please direct all correspondence to lead and back-up counsel at the above addresses. Petitioner consents to electronic service at the email addresses above.

E. Fees Under 37 C.F.R. § 42.103

Petitioner submits fees of \$41,500. If more fees are necessary to accord this Petition a filing date, authorization is granted to charge the same to Deposit Account No. 04-1679.

X. CONCLUSION

Petitioner respectfully requests the Board institute IPR and cancel all Challenged Claims.

Respectfully submitted,

DUANE MORRIS LLP

BY: /Kevin P. Anderson/
Kevin P. Anderson, Reg. No. 43,471
kpanderson@duanemorris.com
DUANE MORRIS LLP
901 New York Ave NW 700 E
Washington, D.C. 20001
T:202-776-5213
F:202-776-7801

Dated: February 21, 2024

CERTIFICATION OF SERVICE ON PATENT OWNER

Pursuant to 37 C.F.R. §§ 42.6(e), 42.8(b)(4) and 42.105, the undersigned certifies that on February 21, 2024, a complete and entire copy of this Petition for *Inter Partes* Review of U.S. Patent No. 9,367,657 and all supporting exhibits were served, as per agreement between the parties, via email and/or electronic download to ICPillar LLC by serving the following counsel for ICPillar, LLC:

Jaime W. Marquart (Cal. Bar No. 200344)

WAYMAKER LLP

515 S. Flower Street, Suite 3500

Los Angeles, California 90071

Telephone: (424) 652-7800

jmarquart@waymakerlaw.com

John Jeffrey Eichmann (Cal. Bar No. 227472)

EICHMANN, a professional corporation

662 N. Sepulveda Blvd., Suite 300

Los Angeles, California 90049

310-237-9190 (tel.)

jeichmann@eichmann.com

Nate Lowenstein

Kenneth Weatherwax

Lowenstein & Weatherwax LLP

1016 Pico Boulevard

Santa Monica, California 90405

weatherwax@lowensteinweatherwax.com

lowenstein@lowensteinweatherwax.com

/Kevin P. Anderson/

Kevin P. Anderson, Reg. No. 43,471

kpanderson@duanemorris.com

DUANE MORRIS LLP

901 New York Ave. NW 700 E

Washington, D.C. 20001

T:202-776-5213

F:202-776-7801

ATTORNEY FOR PETITIONER

CERTIFICATE OF COMPLIANCE

Pursuant to 37 C.F.R. § 42.24 *et seq.*, the undersigned certifies that this document complies with the type-volume limitations. This document contains 13,941 words as calculated by the “Word Count” feature of Microsoft Word, the word processing program used to create it for the applicable sections.

Dated: February 21, 2024

/Kevin P. Anderson/
Kevin P. Anderson, Reg. No. 43,471
kpanderson@duanemorris.com
DUANE MORRIS LLP
901 New York Ave. NW 700 E
Washington, D.C. 20001
T:202-776-5213
F:202-776-7801

ATTORNEY FOR PETITIONER