

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

WESTERN DIGITAL TECHNOLOGIES, INC.
Petitioner

v.

SHAMROCK INNOVATIONS, LLC,
Patent Owner

Case IPR2023-00842
Patent No. 9,535,454

DECLARATION OF MICHAEL CALOYANNIDES, PH.D.

TABLE OF CONTENTS

I.	INTRODUCTION	1
II.	QUALIFICATIONS	1
III.	SUMMARY OF OPINIONS	4
VI.	OVERVIEW OF THE PRIOR ART	9
A.	U.S. Patent No. 6,289,386 (“Vangemert”) (EX 1005).....	9
B.	U.S. Patent No. 5,179,661 (“Copeland”) (EX 1006)	13
C.	U.S. Patent No. 5,717,870 (“Dobson”) (EX 1007)	14
D.	U.S. Patent No. 6,005,414 (“Reay”) (EX 1008)	16
VII.	LEVEL OF ORDINARY SKILL IN THE ART	17
VIII.	CLAIM CONSTRUCTION	18
A.	“Decoder”	19
B.	“Generate a Clock Signal . . .”	19
IX.	UNPATENTABILITY OF THE ‘454 PATENT	21
A.	GROUND 1: Vangemert combined with Copeland teaches all elements of Claims 1, 2, 3, 4, 7, 15, 16, and 17	21
1.	Claim 1	21
2.	Claim 2	51
3.	Claim 3	56
4.	Claim 4	58
5.	Claim 7	59
6.	Claim 15	61

7.	Claim 16.....	62
8.	Claim 17.....	65
B.	GROUND 2: Vangemert combined with Copeland and Dobson teaches all elements of Claim 4.....	66
C.	GROUND 3: Vangemert combined with Copeland and Reay teaches all elements of Claim 6.....	70
D.	GROUND 4: Dobson combined with Copeland teaches all elements of Claims 1, 2, 3, 4, 7, and 16.....	73
1.	Claim 1.....	73
2.	Claim 2.....	106
3.	Claim 3.....	107
4.	Claim 4.....	107
5.	Claim 7.....	109
6.	Claim 16.....	111
E.	GROUND 5: Dobson combined with Copeland and Reay teaches all elements of Claim 6.....	113
F.	GROUND 6: Dobson, Copeland, and Vangemert teaches all elements of Claims 15 and 17.....	115
1.	Claim 15.....	115
2.	Claim 17.....	117
X.	CONCLUSION.....	119

I, Dr. Michael Caloyannides, declare as follows:

I. INTRODUCTION

1. I have been retained by Western Digital Technologies, Inc., (“Petitioner”) as an independent expert consultant in this proceeding before the United States Patent and Trademark Office regarding U.S. Patent No. 9,535,454 (“the ‘454 patent”), which I understand is labeled as EX. 1001. I have been asked to consider whether certain references disclose the features recited in claims 1-4, 6, 7, and 15-17 of the ‘454 patent. My opinions and the bases of my opinions are set forth below.

2. I am being compensated at my normal consulting rate for the time I spend on this matter. No part of my compensation is dependent on the outcome of this proceeding. I have no other interest in this proceeding.

II. QUALIFICATIONS

3. I am an expert in the field of electrical engineering and digital communication and protocols. My entire professional life lasting over fifty years thus far has concentrated on the overarching technologies underlying the ‘454 patent, namely signal processing. The Thesis for my Ph.D. degree in Electrical Engineering, awarded to me by Caltech, was on a theoretical and experimental investigation of receiver noise and its effect on telecommunications. I was also awarded a Ph.D. “subject minor” by Caltech in Applied Mathematics; this is

relevant because the subject matter of the '454 patent is mathematical in nature. Even before that, as a teenager, I had designed and built all the signal processing electronics for a high-power transceiver with which I communicated across the world.

4. Upon graduation I was employed for 14 years at top technical levels in Private Industry (e.g., Boeing Corp.) where I applied my education to advanced military telecommunications projects for the US Department of Defense and the Intelligence Community. During that time, I was also awarded a U.S. Patent in high-speed digital signal processing. I was subsequently hired by the CIA as senior scientist for its Office of Research and Development where, for 14 years, I led numerous classified projects in signal processing, both analog and digital; in that capacity I was awarded the highly competitive Scientist of the Year Award, as well as 5 separate Exceptional Accomplishment Awards and also the Meritorious Officer Award. One of my tasks was to design the architecture of a complete National Signals Interception ("SIGINT") system for a nation that was then a U.S. ally.

5. When the Cold War ended, I returned to Private Industry in technical leadership positions in signal processing as Chief Scientist (at Ideal Innovations Inc.) and later as Senior Fellow (at Mitretek Systems, a spinoff from the MITRE Corp.), and I also accepted appointments as Adjunct Faculty, first at George

Washington University and then at Johns Hopkins University where I taught digital signal processing to graduate school students studying for their advanced degrees. In that capacity I was awarded the Excellence in Teaching Award by Johns Hopkins University. At the same time, I authored two textbooks in digital signal processing, as well as countless peer-reviewed technical papers; In addition, I was a monthly columnist for the IEEE's Security and Privacy Magazine for its entire first year of publication.

6. Upon retiring from private industry, I applied my experience in serving as an expert witness in well over a hundred intellectual property cases, the majority of which were on signal processing algorithms, techniques, processing, and applications; these included the reception and processing of weak GPS signals, point-to-point wireless links, extraction of very weak signals from background noise, 3G, 4G, 5G cellular communications, Covert Communications, and other applications. Before any commercial cellular service became available in the U.S., I designed, fabricated, and operated my own small-scale cellular telephone system which allowed me to place and receive telephone calls over the Los Angeles Basin.

7. In light of all of the foregoing, I believe that I am well qualified to assess and opine authoritatively on the subject matter of the '454 patent which deals with a subset of signal processing.

8. A true and correct copy of my curriculum vitae is provided, and I understand that it will be labeled as Exhibit 1004.

III. SUMMARY OF OPINIONS

9. The opinions in this Declaration are based on the documents I have reviewed and my knowledge and professional judgment. In forming my opinions, I reviewed each of the publications that I discuss.

10. My opinions are based upon how a person having ordinary skill in the art (“PHOSITA”) would have understood the ‘454 patent, the prior art, and the state of the art at the time of the alleged invention, which I understand is assumed to be on or around August 6, 1998, for purposes of this analysis.

11. I have been informed that in determining whether a patent claim is invalid as being obvious, there are four considerations: (1) the level of ordinary skill in the art; (2) the scope and content of the prior art; and (3) any differences between the prior art and the claimed invention; and (4) any additional considerations that would suggest the claimed invention was obvious or nonobvious.

12. In my opinion, and explained in detail below, the combinations of prior art references I have been asked to consider teach all elements in claims 1-4, 6, 7, and 15-17 of the ‘454 patent.

13. It is also my opinion that a PHOSITA would have been motivated to and would have had reason to combine the teachings of the prior art references I have considered in combination, as discussed below.

14. In my opinion, it is more likely than not that a PHOSITA would have found claims 1-4, 6, 7, and 15-17 of the '454 patent obvious based on the prior art.

15. For reference, I attach to this declaration as Appendix A claim charts that support my opinions regarding the claims.

V. OVERVIEW OF THE TECHNOLOGY OF THE '454 PATENT AND THE STATE OF THE ART

16. The '454 patent generally discloses bridging (i.e., interfacing or connecting) devices for providing serial communication between two parallel data busses. *EX 1001*, at col. 5:13-17.

17. The '454 patent includes 45 claims, including independent claims 1, 18, 31, 32, and 38. I was asked to analyze the patentability of only claims 1-4, 6, 7, and 15-17.

18. For reference, I have reproduced below independent claim 1 of the '454 patent.

A system comprising:

a connector configured to be connectable to a serial link; and

a module coupled to the connector, wherein the module is operable to receive first serial data transmitted over the serial link through the connector and transmit second serial data to the serial link through the connector, and wherein the module includes;

a receiver operable to receive the first serial data form the serial link through the connector;

a first circuit configured to deserialize the first serial data to generate deserialized data;

a decoder configured to decode the deserialized data;

a first buffer operable to store data that has been deserialized and decoded by the first circuit and the decoder respectively;

a second buffer operable to receive data, wherein the second buffer is in data communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data; and

a transmitter operable to transmit the second serial data to the serial link through the connector;

wherein the module is configured to determine whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer; and

wherein the module is operable to generate a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount, and wherein the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer.

EX 1001, claim 1.

19. The claims and general disclosure of the '454 patent describe serial communication between devices. Digital serial communication can be traced back to Morse Code of the past, which was serial communication in that one data bit at a time (dots and dashes) was either transmitted or received. As data processing equipment advanced to where multiple bits could be processed simultaneously, such as a data symbol having multiple bits of information, some serial data busses

gave way to parallel busses that could handle multiples bits (e.g., 8, 16, 32, etc.) at a time to maximize bandwidth of a given device.

20. In its Abstract, the '454 patent states that it relates to extending the distance over which the reach of a parallel data bus can be stretched in distance and does so by using an asynchronous serial link as the conduit, because asynchronous serial data busses can cover much longer distances than parallel data busses. ("A computing module includes an interface to asynchronously, serially exchange parallel system bus data"). *EX. 1001*, Abstract.

21. This is further underscored in the '454 patent specification, which states:

"The PCI bus is pervasive in the industry, but as a parallel data bus is not easily extended over any distance or bridged to other remote PCI based devices" *EX. 1001*, at col. 1:38-40).

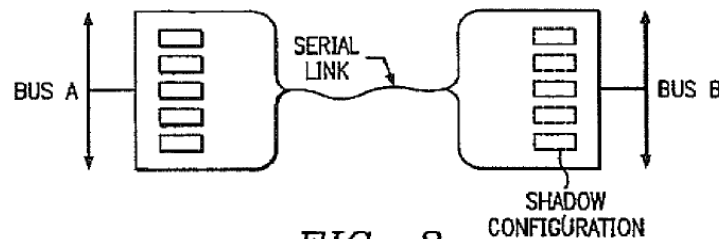
whereas, by contrast,

"Typical high speed serial bus operation on the other hand allows the possibility of all sizes of data transfers across the bus like PCI" *EX. 1001*, at col. 2:16-18.

22. To convert between parallel and serial busses, various parallel-to-serial, and serial-to-parallel, converters are required to transpose the data from one form to the other. These converters were well known in the art at the time of the earliest priority date of the '454 Patent, and in fact have been known since at least 1989. *EX. 1006*, at col. 2:22-45; col. 3:46-4:38; p. 1. Such devices are nothing

more than elementary circuits that handle the conceptually simple tasks of book-keeping to convert, say, an 8-bit symbol (such as an alphanumeric character) to the string of bits (zeroes and ones) that it represents.

23. As is known by a PHOSITA the process of implementing this conversion from a parallel PCI bus to a serial high-speed bus necessarily requires such converters, and the '454 patent discloses just that in FIG. 3 (reproduced below) where it shows two parallel busses A and B being interconnected through a serial link.



'454 Patent FIG. 3

24. Converters, such as the above, may include a variety of circuitry or components to perform the required conversions. Typically, the receiving device will store received data in a buffer after it has been converted from serial to parallel, where it is stored until retrieved by a host device. Flow control mechanisms may be used to prevent an overflow of the buffer, such as when the data is unable to be read out of the buffer quickly enough. See *e.g.*, EX. 1010, at col. 6:47-63. Prior to the buffer becoming full, a flow control action may be taken

to prevent overflow, such as instructing one or more devices to stop sending and/or receiving data, or alternatively instructing or otherwise interrupting a host device to read the data from the buffer before the overflow can occur. Utilizing flow control to prevent an overflow of a receive buffer has been known in the art as far back as at least 1989. *EX. 1006*, at p. 1, Abstract.

VI. OVERVIEW OF THE PRIOR ART

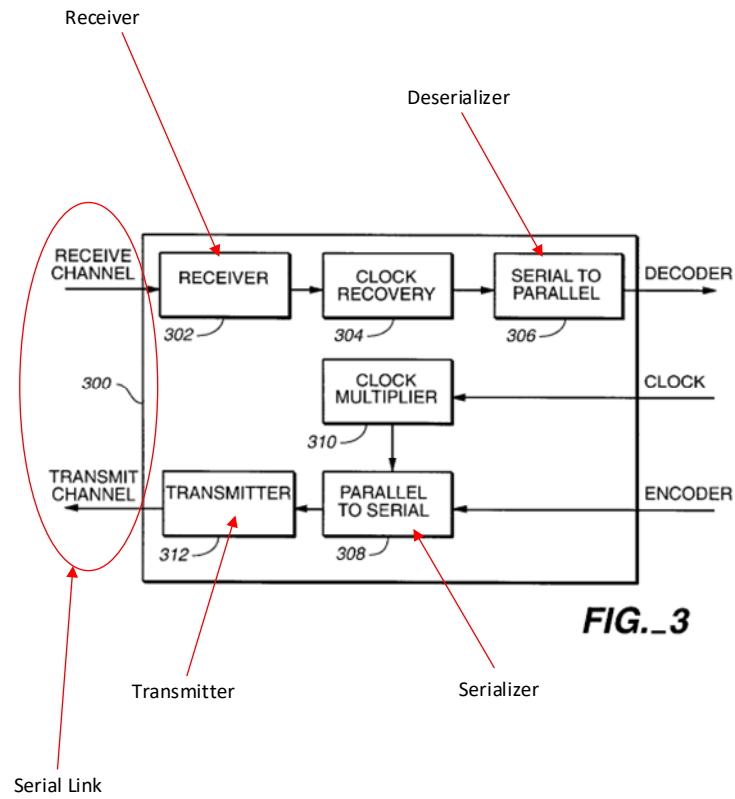
25. I briefly summarize below the prior art patents I was asked to review: Vangemert, Copeland, Dobson, and Reay. All statements regarding a prior art reference's disclosures are made from the perspective of a PHOSITA at the time of the alleged invention.

A. U.S. Patent No. 6,289,386 ("Vangemert") (EX 1005)

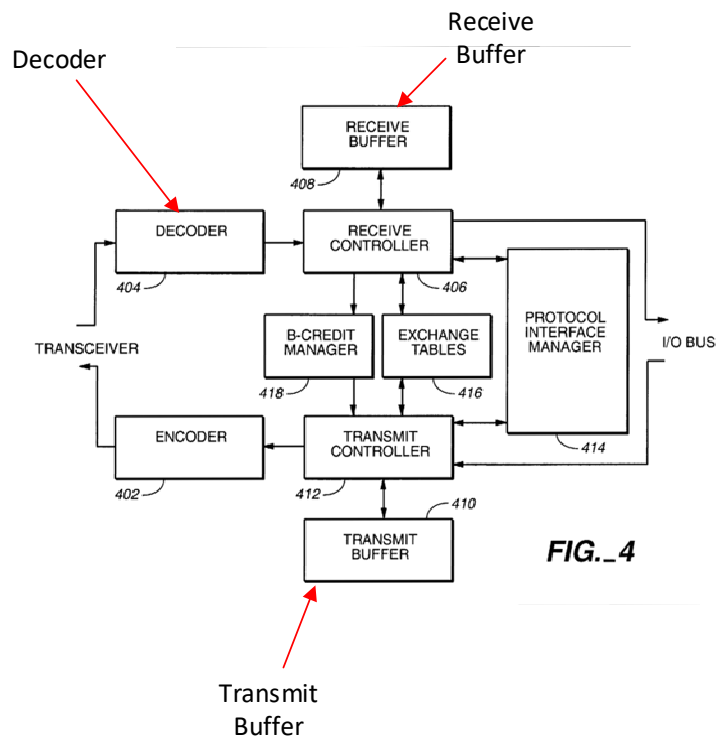
26. I understand that Vangemert is prior art to the '454 patent under pre-AIA 35 U.S.C. § 102(e) because it issued from a patent application filed on May 11, 1998, before the earliest possible priority date of the '454 patent. *EX. 1005*, at 1.

27. Vangemert is generally directed to a network interface unit coupled to one or more devices via a serial communication link and which implements a low latency algorithm for determining buffer credit (i.e., flow control). *EX. 1005*, at col. 1:47-63. Vangemert discloses a serial transceiver module within a network for coupling a node to a serial communication link. *EX. 1005*, at col. 3:19-21. The

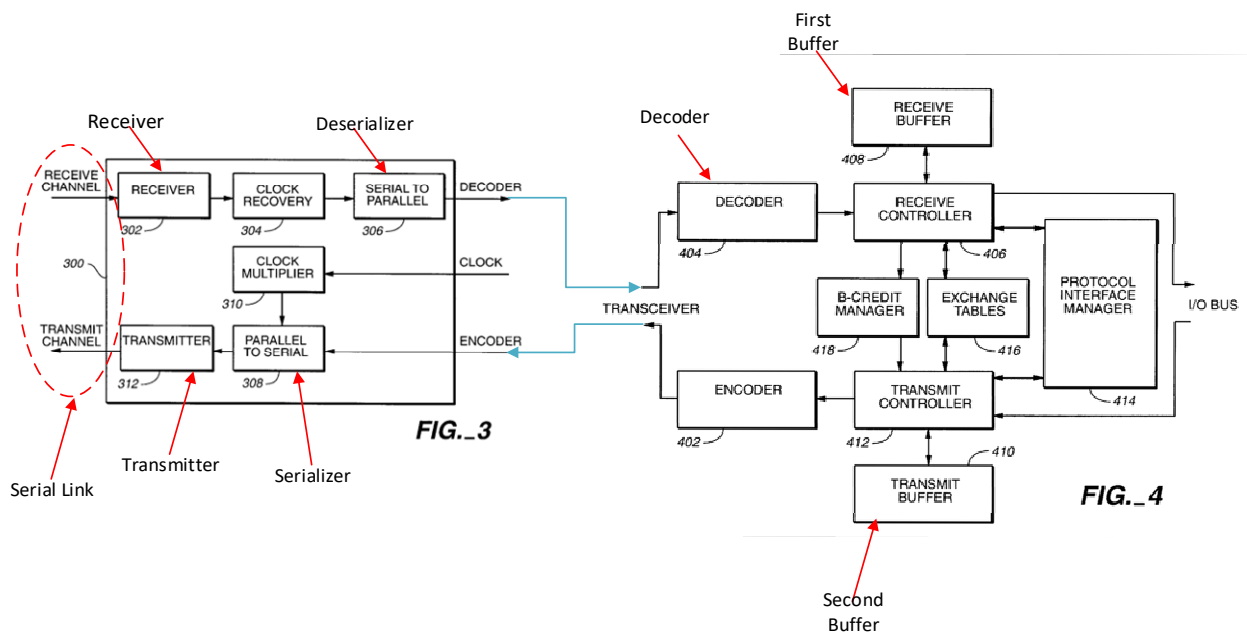
transceiver disclosed in Vangemert is further disclosed to include a receiver and a transmitter, as well as a deserializer (meaning a routine serial-to-parallel converter) in the form of serial to parallel converter 306. *EX. 1005*, at col. 3:21-28. The transceiver is coupled to an input-output (“I/O”) bus of a node device via network interface unit, which includes a decoder 404 for decoding deserialized data from the transceiver 300, as well as a receive buffer 408 and a transmit buffer 410. *EX. 1005*, at col. 3:41-61. Annotated figures 3 and 4 of Vangemert, below, highlight the main features of Vangemert as they relate to the relevant claims of the ‘454 patent. A modified combination of FIGs. 3 and 4 is also provided below, showing the interconnection between the transceiver 300 as well as the network interface unit of FIG. 4.



Vangemert FIG. 3 (Annotated)



Vangemert FIG. 4 (Annotated)

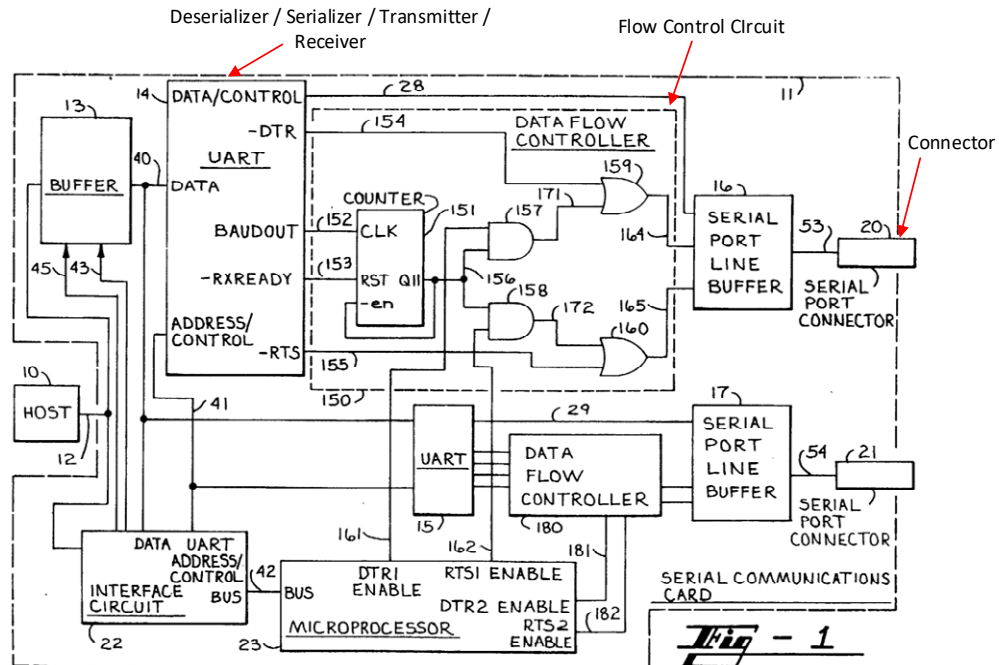


Vangemert Modified Combined FIGs. 3 and 4 (Annotated)

B. U.S. Patent No. 5,179,661 (“Copeland”) (EX 1006)

28. I understand that Copeland is prior art to the ‘454 patent under pre-AIA 35 U.S.C. § 102(b) because it issued as a patent on January 12, 1993, more than one year before the ‘454 patent’s earliest possible priority date. *EX. 1006*, at 1.

29. Copeland is generally directed to a data flow controller for monitoring and controlling serial communication between a remote transmitter and a host device. *EX. 1006*, at Abstract. Specifically, Copeland discloses a “dual channel asynchronous serial communications card 11” which provides a connection to a host computer 10. *EX. 1006*, at col. 3:49-52. The card 11 includes a “host data buffer 13, two parallel-serial converters such as universal asynchronous receiver transmitters (UARTs) 14 and 15 . . . two data flow controllers 150 and 180, two serial port line buffers [temporary data storage] 16 and 17, [and] two serial port connectors 20 and 21.” *EX. 1006*, at col. 3:53-60. Annotated Figure 1 illustrates the general architecture of the serial communications card of Copeland.



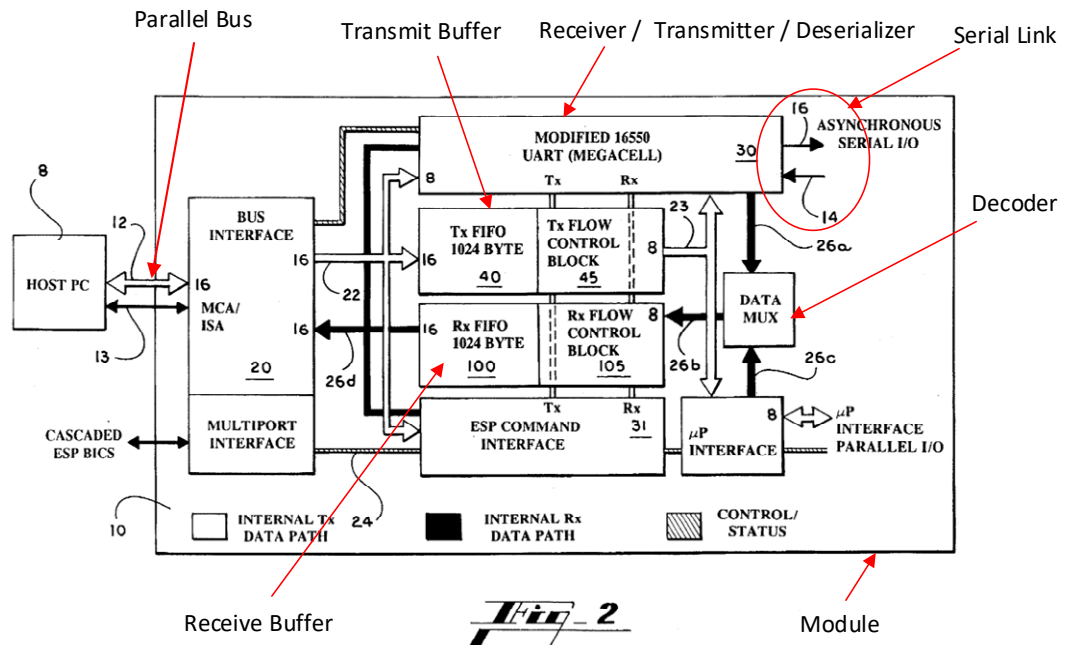
C. U.S. Patent No. 5,717,870 (“Dobson”) (EX 1007)

30. I understand that Dobson is prior art to the '454 patent under pre-AIA 35 U.S.C. § 102(a) because it was patented on February 10, 1998, before the invention of the '454 patent (i.e., earliest filing date of August 6, 1998). *EX. 1007*, at 1. I understand that Dobson is also prior art to the '454 patent under pre-AIA 35 U.S.C. § 102(e) because it issued from a patent application filed on October 26, 1994, before the earliest priority date of the '454 patent. *Id.*

31. Dobson is generally directed to an improved input/output FIFO (First In First Out, a common architecture for buffers) buffering device for use with a universal asynchronous receiver/transmitter (“UART”.) *EX. 1007*, at Abstract. The

FIFO buffering device is intended for use with serial ports associated with general purpose computers and for buffering input and output data from a high-speed serial port. *EX. 1007*, at col. 1:6-14. This allows for communication application programs to support high speed transfer rates while being backwards compatible with software written for conventional serial port devices. Dobson states, and I agree, that the “fundamental building block of a serial port for a computer or terminal is a universal asynchronous receiver-transmitter, commonly known by the acronym UART. This device accepts parallel words or bytes (units of digital information that usually consist of 8 bits each) from the data from the data terminal equipment, generates appropriate timing signals, depending on the bit rate, and transmits the bytes in bit serial form. On the receiver side of the UART, the centers of the bit times are determined, based on the approximate data rate, and serial data is clocked into a register to form a word or byte that is then available for reading out in parallel by the data terminal equipment.” *EX. 1007*, at col. 1:36-47. Thus, Dobson discloses serial communication of parallel data between devices over a high-speed serial link. Further, FIFO buffers serve to buffer outputs of serial to parallel and parallel to serial converters to provide buffering. *See EX. 1007*, at col. 2:27-32. Dobson further discloses outputting a flow control signal in response to the number of bytes written into a receive buffer equaling a predetermined value. *EX. 1007*, at col. 9:33-40.

32. The following is an annotated Figure 2 from Dobson that highlights the relevant portions of Dobson with respect to the claims of the '454 patent.



Dobson FIG. 2 (Annotated)

D. U.S. Patent No. 6,005,414 ("Reay") (EX 1008)

33. I understand that Reay is prior art to the '454 patent under 35 U.S.C. § 102(e) because it issued from a patent application filed on June 3, 1997, before the earliest possible priority date of the '454 patent. *EX 1008*, at 1.

34. Reay is generally directed to a mixed-mode multi-protocol serial interface driver. *EX. 1008*, at Abstract. Specifically, Reay discloses a “mixed-mode multi-protocol serial interface driver for conforming output signals to one of a plurality of selectable electrical interface standards. *EX. 1008*, at col. 4:3-6. The interface driver may allow for output signals to be conformed to various standards, such as “CCITT/EIA interface standards; V.35, V.11/RS-422, V.28/RS-232, and V.10/RS-423. Interface driver 100 includes circuitry that can operate in voltage-mode, current-mode, or both, and has input terminal 102, a mode-select input terminal 104, output terminals 106 and 108, and power supply terminals 110, 112, 114, and 116.” *EX. 1008*, at col. 4:7-14. Data output over terminals 106 and 108 “provide inverted and non-inverted output signals, respectively, of the input signal received at terminal 102. Thus a differential, or balanced-line, output can be provided for those interface standards that require such output.” *EX. 1008*, at col. 4:15-19.

VII. LEVEL OF ORDINARY SKILL IN THE ART

35. I have been asked to assume that the time of the alleged invention of the ‘454 patent is on or around August 6, 1998. I have applied that timeframe and understanding to my analysis and opinions in this declaration.

36. I have been asked to give my opinion regarding the level of ordinary skill in the art with respect to the ‘454 patent. To determine the level of ordinary

skill in the art, I considered the '454 patent, the technology, the educational level, and experience of persons in the field, the types of problems encountered in the art, prior solutions to those problems, the rapidity with which innovations are made, the sophistication of the technology in the field, observations of implementations in products for sale in the marketplace, and my own education and over half a century of experience in Industry, Government, and Academia summarized earlier in this declaration.

37. Based on these considerations, I believe a person of ordinary skill in the art at the time of the alleged invention of the '454 patent would have had at least a bachelor's degree in electrical engineering, or equivalent, with about three years of professional work experience in serial and parallel data communications within a computing device or system and between different computing devices, systems, or subsystems. More education can supplement practical experience and vice versa.

38. I understand this level of skill in the art and believe I am qualified to opine from the perspective of a person having ordinary skill in the art at the time of the claimed invention.

VIII. CLAIM CONSTRUCTION

39. I have been informed that claim terms are typically given their ordinary and customary meaning, as would have been understood by a PHOSITA,

at the time of the invention, taking into consideration the language of the claims, the specification and prosecution history. I address below two claim terms that do not appear in the '454 patent specification and whose meanings may be clarified for purposes of this analysis.

A. “Decoder”

40. Claim 1 recites “a decoder configured to decode the deserialized data.” The terms “decoder” and “decode” are not in the '454 patent specification. However, the term decoder is found in at least U.S. Patent No. 6,070,214 (“'214 patent”), which is incorporated by reference into the '454 patent. Specifically, the '214 patent states “[u]pon receipt, decoder 32 restores the frames into the original 38-bit format and loads the last two described cycles onto the stack of register 24.” *EX. 1008*, at col. 10:50-54. Based on the disclosure in the '214 patent, it is my understanding that the decoder is configured to restore received frames into their required bit format and load them as bits into one or more registers. This is consistent with my understanding of the plain and ordinary meaning of “decoder” in this context, as any component configured to restore or otherwise modify received data to place the received data in a desired format.

B. “Generate a Clock Signal . . .”

41. Dependent claim 17 recites “the module is further configured to *generate a clock signal*, for use by the module, using the first serial data.” EX

1001, col. 11:42-43. This term is not present in the '454 patent, and thus there is no disclosure to inform its meaning within the specification of the '454 Patent. Nor is the term present in the specifications of the '454 patent's related applications or patents. I also understand that the patent applicant identified certain portions of its related U.S. Patent Application No. 09/130,058 (now the '214 patent), at columns 6:33-46, 8:38-53, as support for the addition of a similar claim limitation ("generating a first clock signal . . .") during prosecution of the related '675 patent. I have reviewed these cited portions of the '214 patent, which do not include the claim limitation at issue, and I do not find them helpful to inform this claim limitation's meaning.

42. In this particular analysis, however, the language of claim 17 would have provided a PHOSITA information to generally understand the plain meaning of the above term. Specifically, claim 17 recites "generate a clock signal . . . **using the first serial data.**" Thus, the generated clock signal is not created out of thin air or in the absence of additional information, but rather, "using the first serial data." Thus, a PHOSITA would have understood that the generation of a clock signal using received serial data (i.e., first serial data) would include various ways of producing a clock signal, such as by extracting or recovering a clock signal **from** (i.e., using) the first serial data.

IX. UNPATENTABILITY OF THE ‘454 PATENT

43. I have been asked to evaluate the above-noted claims of the ‘454 patent and give my opinion as to whether certain combinations of references disclose the elements of the claims of the ‘454 patent. I have also been asked to provide my opinion as to whether a PHOSITA would have had reason to combine certain prior art references.

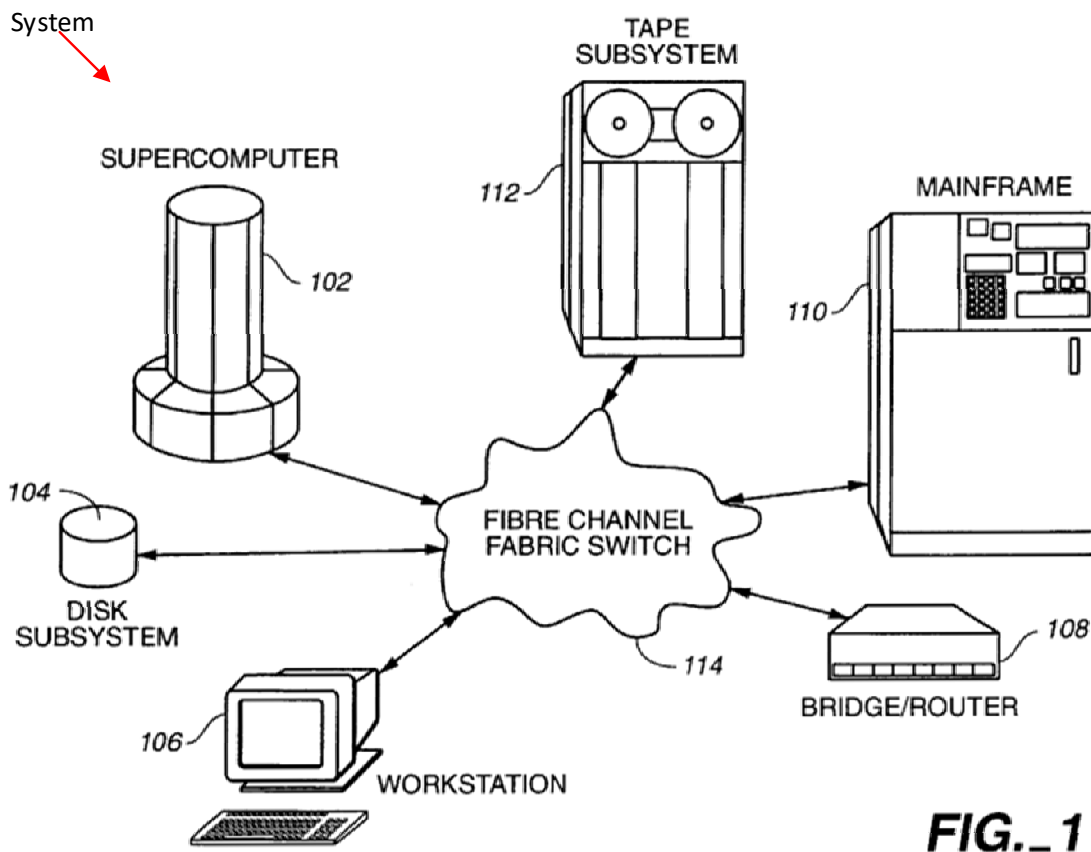
44. As explained below, it is my opinion that the combinations of references detailed below teach all elements of claims 1, 2-4, 6, 7, and 15-17 of the ‘454 patent, and a PHOSITA would have had reason to combine these references, thus rendering obvious those claims. I refer to my application of combination of references to the ‘454 patent claims at issue as a separate “Ground.”

A. GROUND 1: Vangemert combined with Copeland teaches all elements of Claims 1, 2, 3, 4, 7, 15, 16, and 17

1. Claim 1

a. A system comprising:

45. Vangemert discloses a system, such as the system network of FIG. 1 (below). *EX. 1005*, at col. 2:58-61.



Vangemert FIG. 1 (Annotated)

b. a connector configured to be connectable to a serial link;

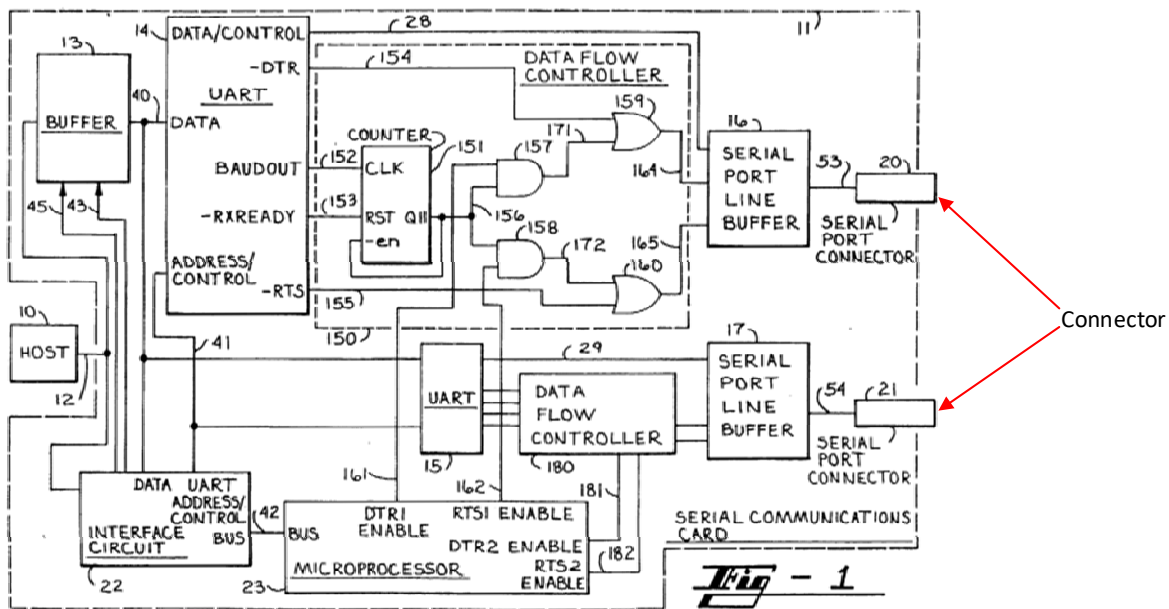
46. Vangemert explains that each of the nodes of the system of FIG. 1 are “coupled to a central ‘fabric switch’ 114 by a serial link having a receive channel and a transmit channel.” *EX. 1005*, at col. 2:61-63. Thus, the serial link connecting the nodes to the fabric switch 114 is a “serial link” as claimed. In reference to the example shown in FIG. 2, Vangemert discloses that “the nodes are coupled in an ‘arbitrated loop’ topology, in which the transmit channel from one node is directly

connected to the receive channel of the next node.” *EX. 1005*, at col. 3:5-7. The qualifier “arbitrary” preceding the word “topology” would be understood to mean “any” topology. From Vangemert, a PHOSITA would have appreciated that a connector would have been a required part of the system because connectors are generally known and widely used to efficiently create (implement) connections between components.

47. Copeland teaches a connector configured to be connectable to a serial link, as claimed. Copeland discloses a dual channel asynchronous serial communications card 11 that includes “two serial port connectors 20 and 21, an interface circuit 22, and a microprocessor 23.” *EX. 1006*, at col. 3:59-60. The serial port connectors 20 and 21 are further described as being “connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” *EX. 1006*, at col. 4:31-34. Thus, the serial port connectors 20 and 21 are configured to be coupled to a serial link, such as an RS-232-D serial interface. RS-232-D (as well as earlier generations of RS-232) uses a standard connector that has been known in the art at least as far back as Copeland’s filing in 1989. *See EX. 1006*, at 1. The RS-232 connector of Copeland is but one possible serial port connector, such as serial port connectors 20, 21, and is only discussed here to provide a general illustration of

but one possible serial port connector contemplated by Copeland. *EX. 1006*, at col. 4:33-36

48. As shown in annotated FIG. 1 of Copeland, below, the serial port connectors 20, 21 are coupled to UARTs 14 and 15, respectively. For example, serial port connector 20 is coupled to UART 14 via the data/control line 28, as shown below. Furthermore, Copeland teaches a “dual channel asynchronous serial communications card 11” for communicating with various devices, such as modems. *EX. 1007*, at col. 3:49-63. Thus, a PHOSITA would have understood that serial port connector 20 is associated with a first channel of the dual channel serial communications card 11, and serial port connector 21 is associated with a second channel of the dual channel serial communications card 11 and may therefore be used interchangeably. While references made herein are to serial port connector 20, it is understood that serial port connector 21 provides similar functionality. Accordingly, Copeland teaches “a connector configured to be coupled to a serial link.”



Copeland FIG. 1 (Annotated)

49. To the extent a connector is not disclosed in Vangemert, a PHOSITA would have looked to Copeland, which is directed to the same subject matter as Vangemert – serial interfaces for providing serial communication between devices. See EX. 1005, at col. 3:19-21; EX. 1006, at col. 2:35-47. Any electrical connection obviously and logically terminates with a connector. Accordingly, a PHOSITA would have found it obvious and been motivated to combine the connector 20 or 21 of Copeland with the serial link of Vangemert, recognizing the impracticality of hardwiring the disclosed link to the devices and the advantages of a connector, as a mechanism for connecting the serial link of the individual nodes 102, 104, 106, 108, 110, and 112 to one or more external serial devices, such as other nodes within the system.

50. The logical boundary of any communications interface ultimately terminates at a connection to the outside world; that connection can be wired, optical, wireless or any other kind but it is a connection none the less. Accordingly, Vangemert's serial link, too, would have to end at a mechanism for connecting the serial link to a device. A PHOSITA would have readily understood this logical truism and would have been motivated to combine connector 20 or 21 of Copeland with the devices of Vangemert (e.g., nodes 102-112) to connect them to the serial link. In addition, given the common use of connectors in serial communication systems and devices, a PHOSITA would have had a reasonable expectation of success in combining the connectors of Copeland (or connectors generally) with the devices and serial link of Vangemert.

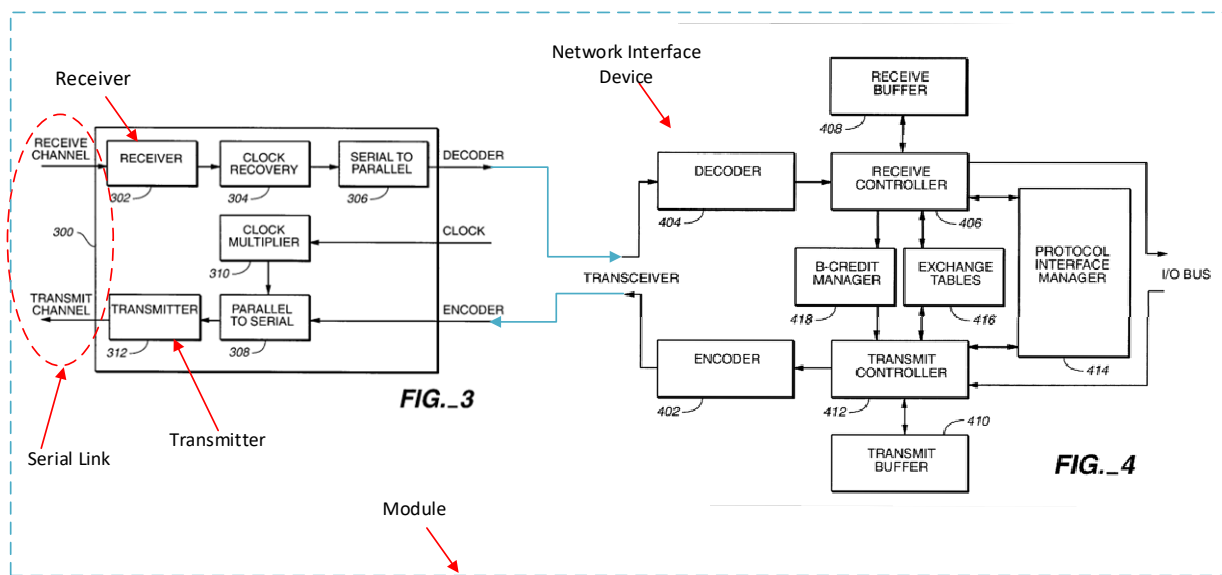
- c. *a module coupled to the connector, wherein the module is operable to receive first serial data transmitted over the serial link through the connector and transmit second serial data to the serial link through the connector, and wherein the module includes*

51. Vangemert FIG. 1 describes various nodes 102-112 coupled to a fabric switch 114 via a serial link. *EX. 1005*, at col. 2:61-63. The fabric switch facilitates connections between the transmit and receive channels of the individual nodes to allow for serial communication between nodes. *EX. 1005*, at col. 2:58 – 3:3. A PHOSITA would have known that the nodes, by virtue of their respective serial transmit and receive channels, would be operable to “receive first serial data

transmitted over the serial link . . . and transmit second serial data to the serial link,” as claimed.

52. Vangemert’s embodiment of FIG. 2 shows a communication topology in which the “nodes are coupled in an ‘arbitrated loop’ topology, in which the transmit channel from one node is directly connected to the receive channel of the next node.” *EX. 1005*, at col. 3:5-7. In this arrangement, control words are used to allow an initiating node to “get control of the loop, then notif[y] a responding node of its desire to transmit data frames.” *EX. 1005*, at col. 3:11-13. Thus, in both described topologies (Vangemert’s FIG. 1 and FIG. 2), the nodes transmit and receive serial data via the transmit and receive channels (serial link).

53. Vangemert further teaches that the nodes include the serial communication transceiver 300 of FIG. 3, and the network interface unit of FIG. 4, thereby further teaching the respective “modules” of claim 1 operable to transmit and receive data over a serial link. *See EX. 1005*, at col. 2:58 – 3:45. This combination of the transceiver 300 and network interface unit is illustrated in the modified combination of FIGs. 3 and 4 of Vangemert, shown below. Thus, any node 102-112 would have the module coupled to the connector and thereby the serial link.



Vangemert Modified Combined FIGs. 3 and 4 (Annotated)

54. FIGs. 3 and 4, and the associated description, are further illustrative on this point. As noted in Vangemert, “FIG. 3 illustrates a serial communication transceiver 300 which may be used by the nodes to couple to the serial communications link. Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming single levels. . . .” *EX. 1005*, at col. 3:19-23. The transceiver 300 further includes a transmitter 312 that “converts the serial bitstream into signals suitable for transport by the transmit channel.” *EX. 1005*, at col. 3:35-36. The above noted receiver 302 and transmitter 312 allow for reception of serial data and transmission of data over the serial link.

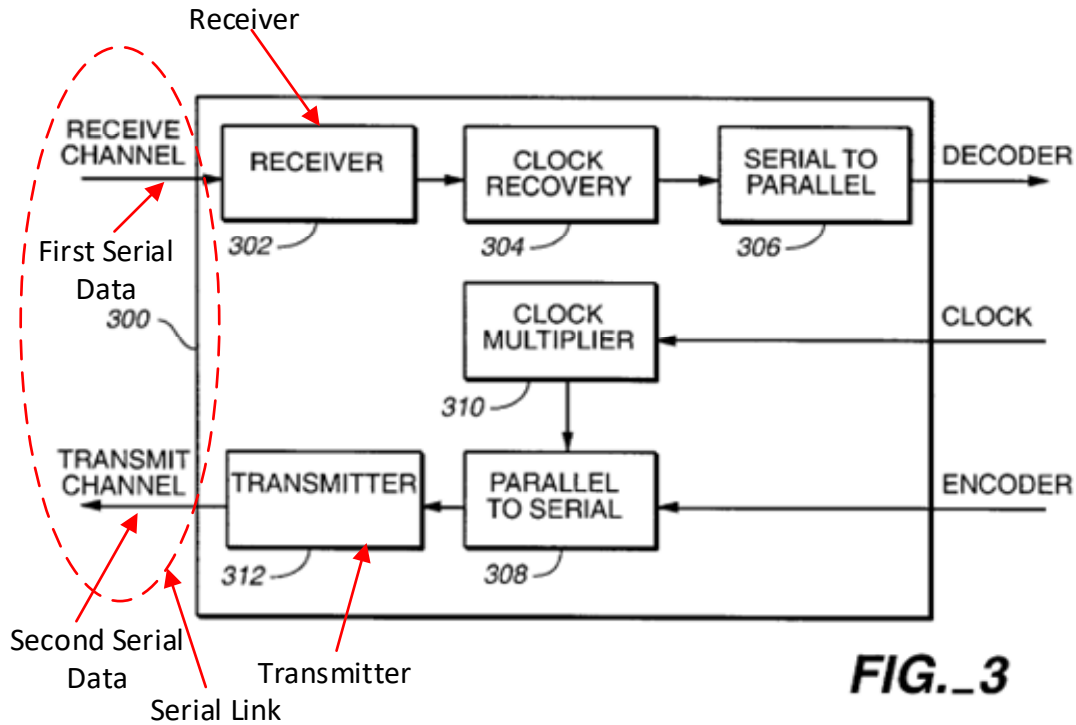


FIG. 3

Vangemert FIG. 3 (Annotated)

55. Furthermore, a PHOSITA would have understood that the various connections between nodes would require connectors to facilitate the connections as described above, and therefore would have been motivated to combine the connectors of Copeland with the module of Vangemert to provide connection of the nodes to the serial link.

56. Accordingly, Vangemert and Copeland disclose that the “module is operable to receive first serial data transmitted over the serial link through the connector and transmit second serial data to the serial link through the connector,” as recited in claim 1.

- d. *a receiver operable to receive the first serial data from the serial link through the connector;*

57. As shown in annotated FIG. 3, below, the transceiver 300 includes a “receiver 302 coupled to the receive channel to sense incoming signal levels. . . .” *EX. 1005*, at col. 3:21-23. The receive channel is part of the serial link coupled to the module. *EX. 1005*, at col. 2:61-63. Furthermore, a PHOSITA would have understood that the nodes, and therefore their associated receivers, would be connected to the serial link through a connector, and would therefore have been motivated to combine the connectors 20 or 21 of Copeland with the receiver of Vangemert, as discussed above. Accordingly, Vangemert discloses “a receiver operable to receive the first serial data from the serial link through the connector,” as recited in claim 1.

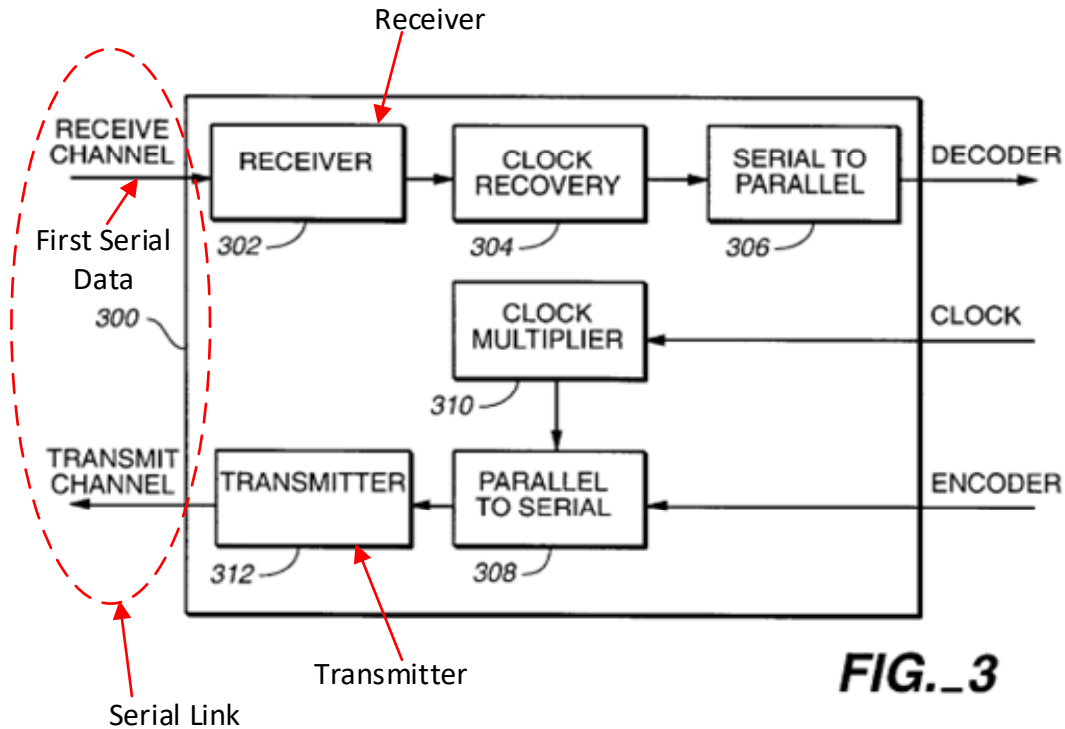


FIG._3

Vangemert FIG. 3 (Annotated)

- e. *a first circuit configured to deserialize the first serial data to generate deserialized data;*

58. Vangemert discloses the above element of claim 1. As illustrated in FIG. 3 (annotated below), the system of Vangemert includes a serial to parallel circuit 306 (i.e., first circuit) which is coupled to receive data from the receiver 302 and configured to deserialize the first serial data into parallel data (i.e., deserialized data). As explained in Vangemert, the “serial-to-parallel converter 306 uses the clock signal to convert the incoming signal [first serial data] from serial form to parallel form.” *EX. 1005*, at col. 3:26-28. Accordingly, Vangemert discloses “a

first circuit configured to deserialize the first serial data to generate deserialized data,” as recited in claim 1.

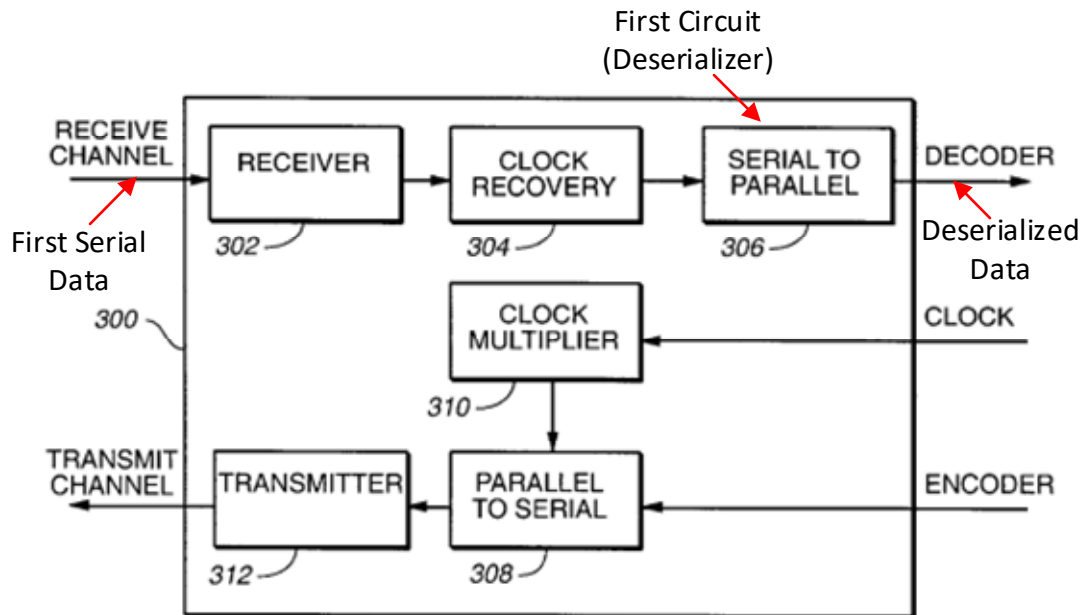
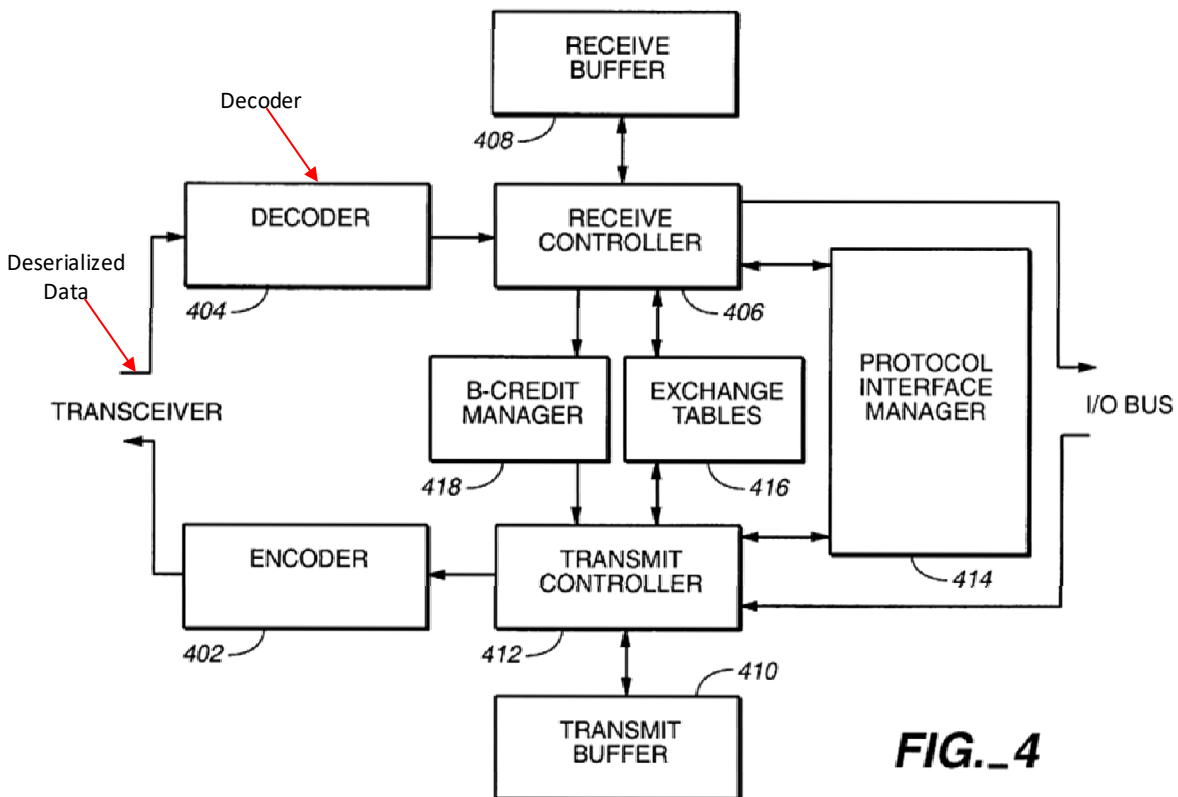


FIG._3

Vangemert FIG. 3 (Annotated)

f. *a decoder configured to decode the deserialized data;*

59. Vangemert discloses the above element of claim 1. As shown in FIG. 3, above, the output of the serial to parallel converter 306 is provided to a decoder 404, which is further shown in FIG. 4, as annotated below.



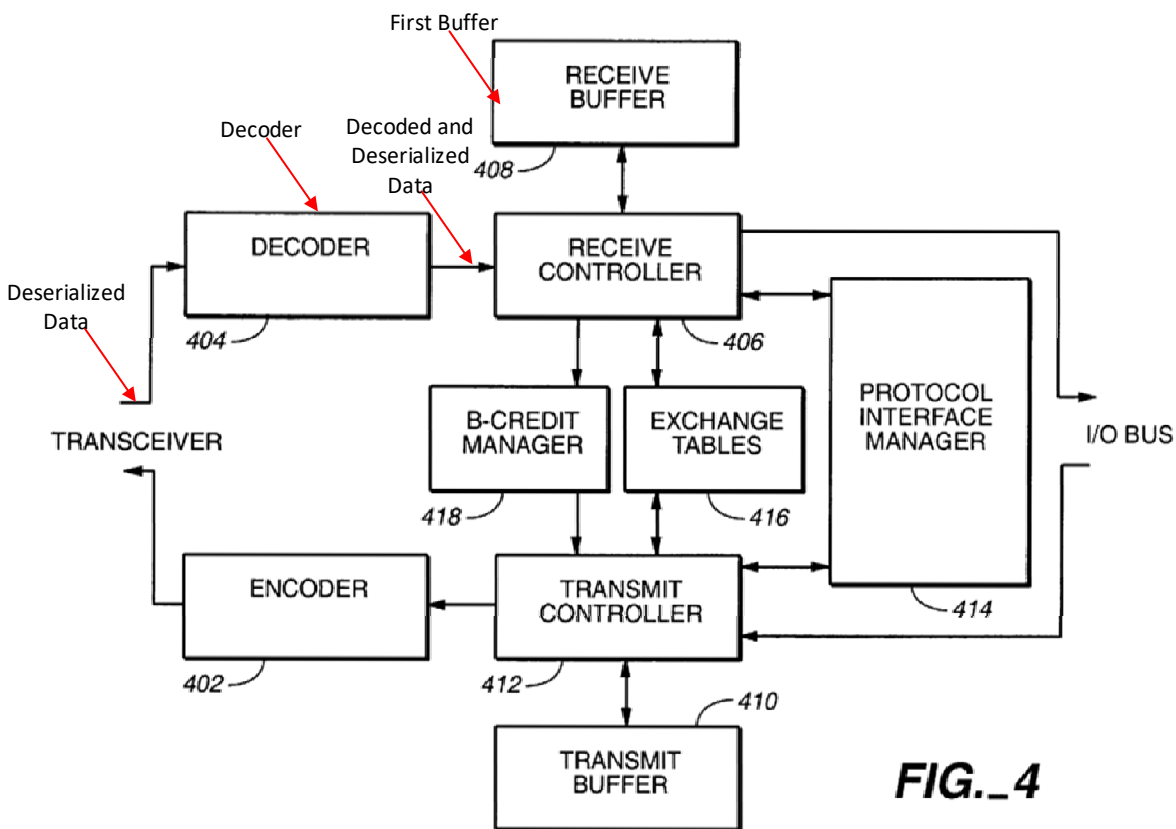
Vangemert FIG. 4 (Annotated)

60. Vangemert explains that the serial to parallel “converter 306 outputs one 10-bit channel codeword at a time. The output from the converter 306 is sent to a decoder.” *EX. 1005*, at col. 3:28-30. Vangemert further explains that “[e]ncoder 402 and decoder 404 provide error correction coding and decoding for information transmitted across the serial link.” *EX. 1005*, at col. 3:43-45. The decoder 404 discloses a decoder configured to decode deserialized data as the decoder 404 receives the output of the serial to parallel converter 306.

Accordingly, Vangemert discloses “a decoder configured to decode the deserialized data,” as recited in claim 1.

- g. *a first buffer operable to store data that has been deserialized and decoded by the first circuit and the decoder, respectively;*

61. As explained below, annotated FIG. 4 from Vangemert illustrates the claimed first buffer.



Vangemert FIG. 4 (Annotated)

62. Vangemert explains that “[d]ecoded data from the decoder 404 is provided to receive controller 406. Receive controller stores received control

words and data frames in receive buffer 408.” *EX. 1005*, at col. 3:45-48. As noted above, the data provided to the decoder is deserialized data, which is then decoded by the decoder. Thus, the decoded data provided to the receive buffer is deserialized and decoded. This decoded and deserialized data is then stored in the receive buffer 408 (shown in annotated FIG. 4, above) which corresponds to the first buffer of claim 1. Accordingly, Vangemert discloses a “first buffer operable to store data that has been deserialized and decoded by the first circuit and the decoder, respectively.

- h. *a second buffer operable to receive data, wherein the second buffer is in communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data; and*

63. Vangemert discloses the above element of claim 1. First, Vangemert discloses a transmit buffer 410. *EX. 1005*, at col. 3:49-52. Vangemert explains that a “transmit controller 412 retrieves control words and data frames from transmit buffer 410 and forwards them to encoder 402.” *Id.* Vangemert further states that the transmit controller 412 “creates or updates exchange information in exchange tables 416 in response to commands and data from node’s i/o bus, generates control words and data frames in transmit buffer 410, and forwards them to encoder 402.” *EX. 1005*, at col. 3:57-61. Thus, the transmit buffer 410 is “operable to receive data,” and therefore discloses the “second buffer” of claim 1. Annotated

FIG. 4, below, illustrates the transmit buffer 410 and its ability to receive or transmit data to the transmit controller 412.

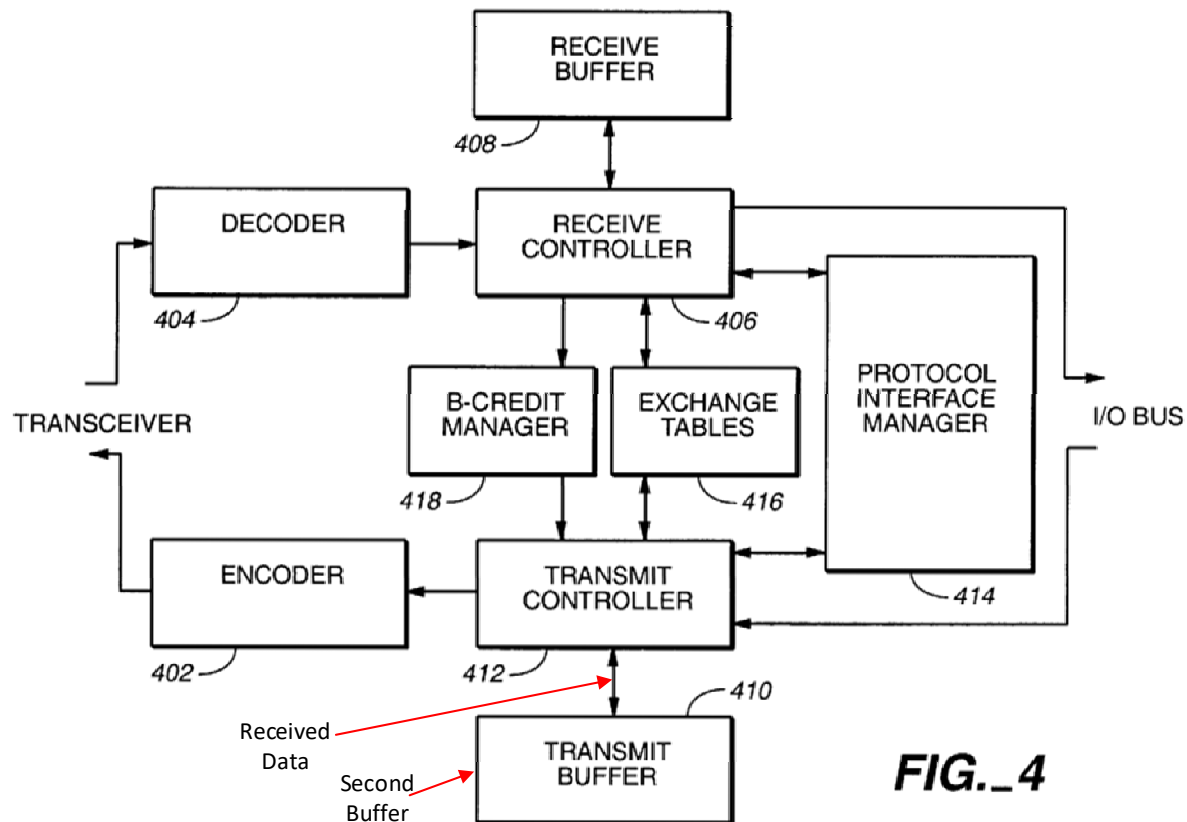
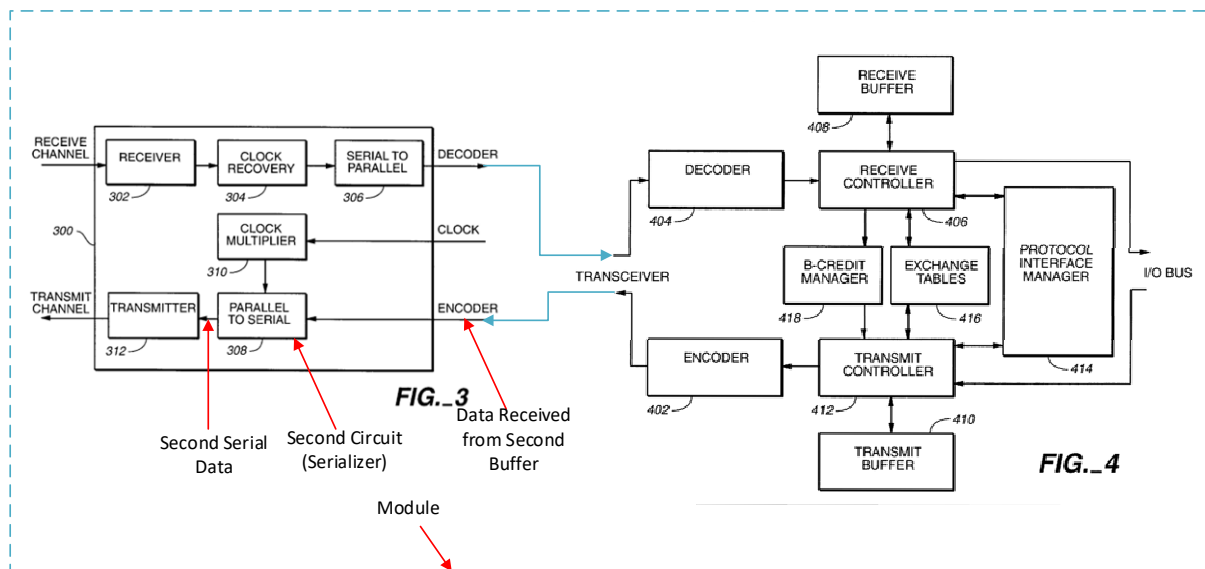


FIG._4

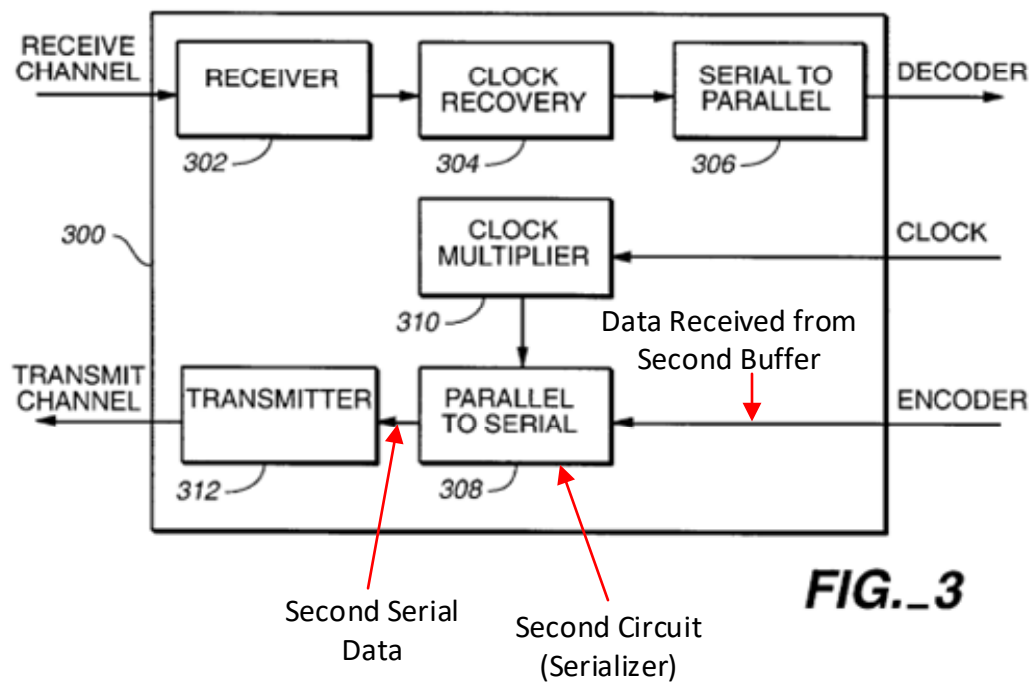
Vangemert FIG. 4 (Annotated)

64. Vangemert further discloses “wherein the second buffer is in data communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data,” as recited in claim 1. The transmit controller 412 is described as retrieving “control words and data frames from transmit buffer 410 and forwards them to encoder 402.” *EX. 1005*, at col.

4:49-52. The encoder 402 then provides the data to a parallel-to-serial converter 308. Specifically, Vangemert states that “[t]ransceiver 300 also includes a parallel-to-serial converter 308 which receives a stream of codewords from an encoder and uses a clock signal from clock multiplier 310 to convert the stream of codewords into a serial bitstream.” *EX. 1005*, at col. 3:31-34. This is shown in the below modified combination of FIG. 3 and FIG. 4 of Vangemert, as well as annotated FIG. 3, below. Accordingly, the parallel-to-serial converter 308 serializes data received from the encoder, which in turn receives data from the transmit buffer 410 [second buffer] via the transmit controller 412, and therefore discloses a “second circuit that is configured to serialize data received via the second buffer to generate the second serial data.”



Vangemert Modified Combined FIGs. 3 and 4 (Annotated)



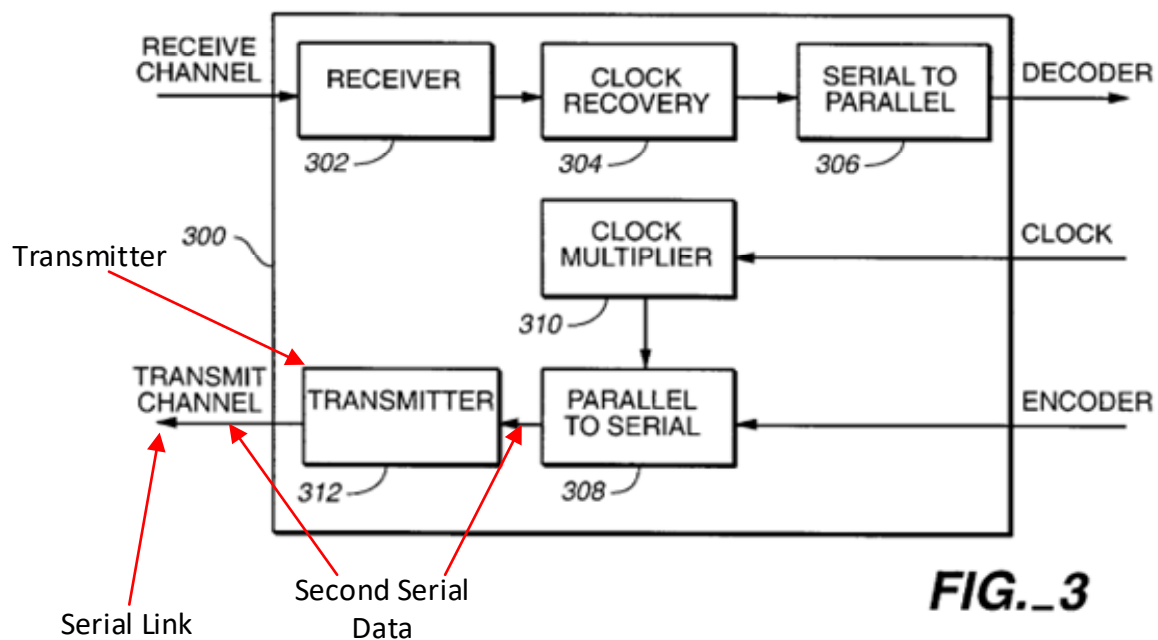
Vangemert FIG. 3 (Annotated)

65. As noted above, the parallel-to-serial converter 308 converts codewords from the encoder “into a serial bitstream,” which discloses generating the “second serial data.” Accordingly, Vangemert discloses “wherein the second buffer is in data communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data,” as claimed.

- i. *a transmitter operable to transmit the second serial data to the serial link through the connector;*

66. Vangemert discloses the above element of claim 1. Specifically, Vangemert discloses that a “[t]ransmitter 312 then converts the serial bitstream [second serial data] into signal suitable for transport by the transmit channel [serial

link].” *EX. 1005*, lines 35-36. Annotated FIG. 3, below, provides further illustration of the transmitter 312. Furthermore, as discussed above, PHOSITA would have understood that the nodes, and therefore their associated transmitters, would be connected to the serial link through a connector, and would therefore have been motivated to combine the connectors 20 or 21 of Copeland with the receiver of Vangemert, as noted above.



Vangemert FIG. 3 (Annotated)

- j. *wherein the module is configured to determine whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer; and*

67. Vangemert discloses this element of claim 1. Specifically, Vangemert states that the “[r]eceive controller 406 includes a counter or register which tracks the amount of available buffer space [of] the receive buffer 408.” *EX. 1005*, at col. 4:32-34. This available amount of buffer space teaches determining an amount of data stored in the first buffer. Vangemert also discloses a buffer credit manager which “includes a comparator 502 which asserts a credit signal when it determines that the additional committed buffer space (i.e., the buffer space committed but not yet occupied) is less than the available buffer capacity by at least the size of one data frame.” *EX. 1005*, at col. 4:24-29. Thus, the comparator evaluates the available buffer capacity as provided by the receive controller 406, against a fill amount represented by the capacity of the buffer, less the available buffer space in view of committed buffer space, as described above. The comparator 502 of Vangemert therefore discloses “determining whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer,” as claimed.

- k. *wherein the module is operable to generate a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount, and*

68. Vangemert teaches generating a flow control signal by transmitting a buffer credit signal to a remote device when the committed space is less than the available space in the buffer, thereby indicating that there is sufficient space in the buffer to accept the data word to be transmitted. Vangemert explains that “FIG. 5 shows one implementation of a buffer credit manager. It includes a comparator 502 which asserts a credit signal when it determines that the additional committed buffer space (i.e., the buffer space committed but not yet occupied) is less than the available buffer capacity by at least the size of one data frame.” *EX. 1005*, at col. 4:24-29. As described in the above passage of Vangemert, a credit signal is generated when the amount of available buffer capacity (fill amount) is sufficient to handle the committed buffer space as well as the data word to be transmitted. The required amount of space to handle the committed buffer space and the at least one data frame is, as noted above, equal to a fill amount, and in response to there being at least that much space (i.e., equaling the fill amount) the bit credit (flow control signal) is generated.

69. Vangemert further states, “before an initiating node can transmit a data frame to a responding node, it must first secure a buffer credit from that node.” *EX. 1005*, at col. 4:9-11. Thus, the credit signal described above teaches a

flow control signal, as the bit credit signal is required for data transfer (data flow) to occur between two devices. Accordingly, a PHOSITA would have understood that the credit signal is a “flow control signal,” as claimed. Accordingly, Vangemert discloses generating “a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount,” as claimed.

70. Copeland additionally teaches a module “operable to generate a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount,” as claimed. Copeland recites that a “data flow control is activated when the FIFO buffer is forty percent full.” *EX. 1006*, at col. 5:38-40. As explained in Copeland, the amount of data in the FIFO buffer (first buffer) of UART 14 is determined by monitoring the RXREADY signal generated by UART 14, “which is low whenever there is at least one byte of data, transmitted form serial data port 20 to the UART 14 FIFO buffer, which has not been read by host 10. UART 14 has another signal, BAUDOT, which is a clock at 16 times the baud rate. When -RXREADY goes low counter 151 counts the BAUDOUT cycles. When 1024 BAUDOUT cycles pass without the FIFO buffer being empty the Q11 output of the counter goes high. The count of 1024 BAUDOT cycles represents 64-bit times or, at 10 bits per character, 6.4 character times. This leaves at least 9.6 character times remaining for the remote transmitter connected to serial data port 20 to respond before the FIFO buffer overflow.” *EX. 1006*, at col. 5:24-38.

71. Data flow control is performed by sending a low signal for the Request to Send signal on the serial port connector 20, which “signals the remote transmitter to stop sending data, thus shutting down the flow of data from the serial data port 20 to the UART 14 FIFO buffer.” *EX. 1006*, col. 5:44-55. The low RTS signal (i.e., the flow control signal of claim 1) is therefore sent in response to the FIFO buffer of UART 14 reaching or exceeding a fill value, such as the 40% fill value disclosed above.

72. Copeland further states that “whenever host 10 is unable to keep up with the data transfer rate, the Q11 output of counter 151 will go high, thereby causing the DTR and/or RTS output signals on serial port connector 20 to go low and signal the remote transmitter to stop sending.” *EX. 1006*, at col. 5:65 – 6:2. This teaches that the RTS output signal is a flow control signal as the RTS signal controls the flow of data being transmitted by the remote transmitter. Accordingly, Copeland teaches “generating a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount.”

73. A PHOSITA would have found it obvious to combine the flow control signal generation of Copeland with the bit credit manager 418 of Vangemert as both Copeland and Vangemert teach interchangeable ways to prevent an overflow of the receive buffer 408. For instance, while Vangemert attempts to prevent overflow by preventing transmission of data unless and until there is sufficient

buffer space, Copeland provides a backstop in the event the buffer begins to fill beyond a predetermined limit by interrupting an existing transmission as described above. As an example, the predetermined flow control threshold of Copeland may be set much higher than normal when combined with the bit credit system of Vangemert such that, in general, the bit credit system of Vangemert will control the flow of data. However, the predetermined flow control threshold of Copeland may be set such that if there is miscalculation, or some sort of timing issue with the bit credit system of Vangemert such that the buffer begins to reach its maximum capacity, the flow control system of Copeland may instruct the transmitting device to stop transmitting to prevent a potential loss of data due to buffer overflow. A PHOSITA would have understood that this dual approach would further reduce the potential for an overflow condition to occur as the remote device is instructed to stop transmitting prior to the receive buffer reaching an overflow state in the event of a failure of the bit credit flow control methodology.

74. Alternatively, the flow control signal of Copeland may replace the bit credit signal of Vangemert to prevent overflow of the receive buffer. This may serve to improve latencies within the communication system of Vangemert by eliminating the need for each communication transaction to request, and then subsequently wait for, a bit credit signal in return, before the transmission can occur. For example, in systems where the host is able to retrieve data from the

receive buffer at, or exceeding, a rate in which data is stored in the buffer, the need for performing bit credit calculations is reduced. Implementing the flow control methodology of Copeland into the system of Vangemert would have provided a means for flow control by instructing the transmitting device to stop transmitting in the event that the receive buffer fill exceeds a predetermined threshold. This both reduces signal traffic within the communication network, and reduces any computational delays associated with determining and subsequently generating a bit credit.

75. Additionally, both Vangemert and Copeland are directed to serial interfaces for providing serial communication between devices. *See EX. 1005*, at col. 3:19-21; *EX. 1006*, at col. 2:35-47. As both Vangemert and Copeland are directed to serial interfaces and associated flow control schemes as described above, a PHOSITA would have recognized the advantages of implementing the flow control methodology of Copeland with Vangemert as described above and would have had a reasonable expectation of success in doing so.

76. As a further aspect of how Copeland and Vangemert complement each other, it is noteworthy that the bit credit manager 16 of Vangemert provides buffer credits to remote devices based on a capacity of the receive buffer 408. *See, EX. 1005*, at col. 4:24-29. Similarly, Copeland teaches generating a flow control output in response to the buffer being filled to a predetermined capacity, e.g., 40%.

See, *EX. 1006*, at col. 5:24-38. Thus, a PHOSITA at the time of the claimed invention would have been very motivated to combine the flow control signal of Copeland with the bit credit manager of Vangemert to avoid overflow of the receive buffer while reducing latency since both are proposing solutions to the problem of receive buffer overflow and would have had a reasonable expectation of success in doing so.

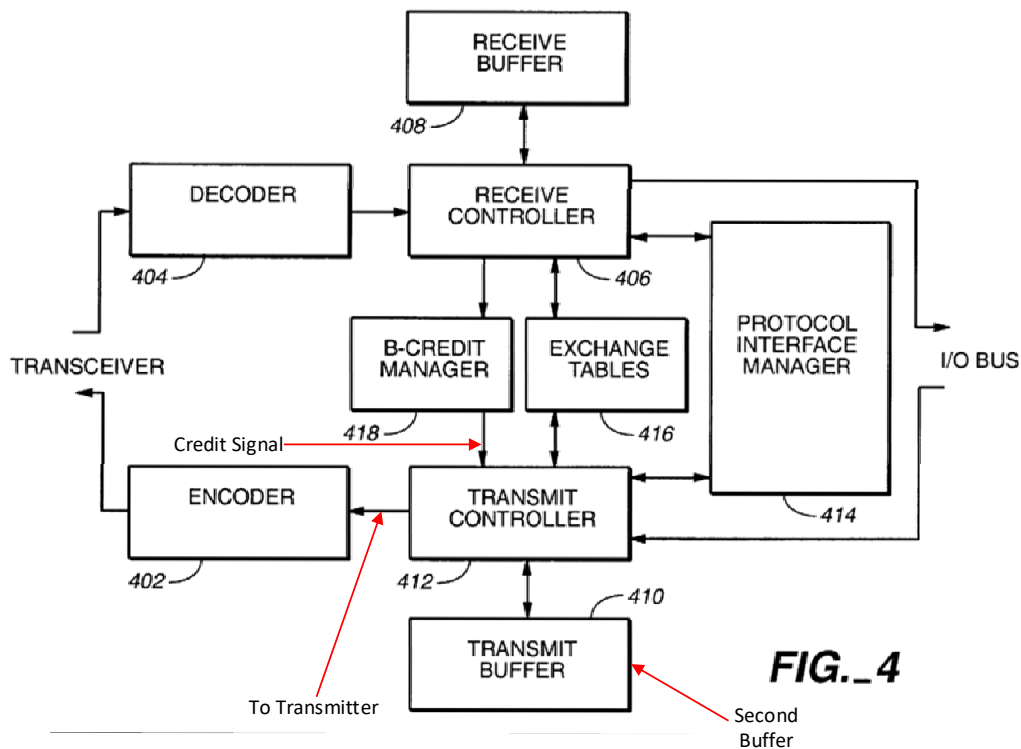
1. *wherein the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer.*

77. Vangemert discloses this element of claim 1. Specifically, Vangemert states that “when receive controller 406 receives a control word indicating that a remote node wishes to transmit data, the buffer credit manager 418 is enabled, and the transmit controller 412 begins transmitting buffer credit control words in response to a credit signal from buffer credit manager 418.” *EX. 1005*, at col. 4:11-16. The credit signal [flow control signal] is transmitted by the transmit controller 412, which is described above as providing control information to the encoder, and out to the serial link via the transmitter 312. See *EX. 1005*, at cols. 3:31-36, 3:57-61. FIG. 4, below, shows the buffer credit manager 418 coupled to the transmit controller 412, and not directly connected to the transmit buffer. The transmit controller 412 receives the credit signal directly from the buffer credit manager

418, and then can pass it directly to the encoder 402. There is no disclosure that the credit signal is first provided to the transmit buffer 410 by the transmit controller 412, nor would it be technically reasonable to do so. Rather, Vangemert teaches that when the buffer credit manager is enabled in response to a request to transmit data from a remote device, “the transmit controller 412 begins transmitting buffer credit control words in response to a credit signal from buffer credit manager,” thereby teaching that the transmit controller 412 outputs data directly upon receiving it from the bit credit manager 418. EX. 1005, at col. 4:11-16.

78. Accordingly, putting the credit signal in the buffer would make no sense to one of skill because the credit signal is not part of a data frame or packet to be sent but is rather a control signal that must be sent in response to a request to transmit data by a remote device. In fact, this would be an inefficient method of operation, and would therefore be opposite to a goal of Vangemert which is to reduce data transmission latency for high-speed data connections. *See EX. 1005*, at col. 1:54-55.

79. Accordingly, a PHOSITA would have understood to transmit the control signal, bypassing the transmit buffer, as the command did not need to be packaged with other data, but needed to be transmitted quickly in response to the request by the remote device to transmit data to minimize transmission latency for high-speed data connections.



Vangemert FIG. 4 (Annotated)

80. Therefore, Vangemert discloses “wherein the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer.”

81. Furthermore, in the event the teachings of Copeland were integrated with those of Vangemert, such as described above to improve the overall operation of Vangemert, a PHOSITA would have been motivated to do so to minimize latencies associated with flow control signals by routing any flow control signals directly to the transmitting device without passing through a transmit buffer. Such

routing would not only have been well within the skill of a PHOSITA but was also contemplated by Copeland. For example, as shown below in annotated FIG. 1 of Copeland, the data flow controller 150 is a separate circuit from the UART 14 and is configured to perform automatic flow control outside of control of the host 10.

EX. 1006, col. 5:65-6:11; 2:30-32. Specifically, upon a counter reaching a count indicating the receive buffer is approximately 40% full, “a low [logic level] for the RTS and DTR signals on serial port connector 20” is sent. *Id.* at col. 5:24-52.

Upon the counter reaching a value, such as 1024 (i.e., number of cycles indicating that the FIFO buffer is 40% full) a signal is generated by the counter to drive the outputs of the RTS and DTR signals low on the serial port connector 20. This signals the remote transmitter to stop sending data, thus shutting down the flow of data from the serial data port 20 to the UART 14 FIFO buffer. *Id.* at col. 5:38-55.

buffer of claim 1. The buffer disclosed in Copeland that holds parallel data to be serialized is bi-directional buffer 13, which holds parallel data sent from host 10 to be serialized in UART 14 before transmission through serial port connector 20. *See e.g., EX. 1006*, at col. 3:46-4:21. Thus, as the flow control signal of Copeland does not pass through the “second buffer,” Copeland teaches “transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer,” as recited in claim 1. Furthermore, as Vangemert and Copeland both teach transmitting flow control signals to remote devices directly, (i.e., without passing through the claimed second buffer), a PHOSITA would have had a reasonable expectation of success in combining the flow control architecture of both Vangemert and Copeland.

2. Claim 2

The system of claim 1, wherein: the system is configured to transmit the first serial data and the second serial data asynchronously over the serial link.

83. Vangemert combined with Copeland teaches all elements of claim 2. Vangemert, in general, teaches an asynchronous system, as there is no separate synchronized clock signal provided to the serial communication transceiver, and instead, any clock signal must be recovered from the data itself. Vangemert states that “[t]ransceiver 300 includes a receiver 302 coupled to the receive channel to

sense incoming signal levels [first serial data], clean them up as much as possible, and provide them to a clock recovery module 304. Clock recovery module 304 uses the incoming signal to reconstruct a clock signal synchronized to the incoming data, and a serial-to-parallel converter 306 uses the clock signal to convert the incoming signal from serial form to parallel form.” *EX. 1005*, at col. 3:21-28. Thus, the transceiver 300, via the receiver 302, is configured to receive serial data, i.e., first serial data, asynchronously as no common, separate clock signal is received by the receiver 302.

84. Furthermore, the transceiver 300 is configured to transmit serial data [second serial data] over the serial link asynchronously. Specifically, the transceiver 300 includes a transmitter 312 that transmits serial data [second serial data] as described above. As explained in Vangemert, the “[t]ransceiver 300 also includes a parallel-to-serial converter 308 which receives a stream of codewords from an encoder and uses a clock signal from clock multiplier 310 to convert the stream of codewords into a serial bitstream. Transmitter 312 then converts the serial bitstream into signals suitable for transport by the transport channel [serial link].” *EX. 1005*, col. 3:31-36. Thus, the transmitter of Vangemert does not transmit a separate, common clock signal as would be done in a synchronous communication scheme. Rather, the clock signal and clock multiplier of Vangemert serve only to convert the codewords received by the transmitter into a

serial bitstream, which is then transmitted. A PHOSITA would have understood that this is asynchronous transmission of serial data [second serial data].

85. Accordingly, while Vangemert does not expressly call its system “asynchronous,” a PHOSITA would have understood this system to teach asynchronous communication.

86. In addition, Copeland teaches that the communications card 11 of FIG. 1 is a “dual channel **asynchronous** serial communications card 11.” *EX. 1006*, at col. 3:49-52 (emphasis added). Thus, Copeland teaches that communication card 11 is configured to transmit and receive data asynchronously. Copeland further discloses “two parallel-serial converters such as universal *asynchronous* receiver transmitters (UARTs), (emphasis added)” which are coupled to the serial port connectors 20 and 21. *EX. 1006*, at col. 3:53-60, and FIG. 1. This is illustrated in annotated FIG. 1, below.

88. If Vangemert were not deemed to sufficiently disclose asynchronous communication, then a PHOSITA would have found it obvious and been motivated to combine the expressly asynchronous transmitter/receiver of Copeland with the transceiver 300 of Vangemert to configure the system to “transmit the first serial data and the second serial data asynchronously over the serial link.” As described above, both Vangemert and Copeland are directed to serial communication devices that are not synchronized to a common, separate clock signal. *EX. 1005*, at col. 3:19-4:7; *EX. 1006*, at col. 3:49-63.

89. For instance, a PHOSITA starting with Vangemert’s system would recognize that it is configured such that a clock signal is extracted from received data. A PHOSITA would also recognize that Vangemert’s system does not include a clock provided over a separate clock line, nor any synchronization to a common clock, as would be done in a synchronous system. A PHOSITA would have looked to Copeland’s disclosure of “asynchronous” communication and understood that the plain meaning of “asynchronous,” as used in Copeland’s system and elsewhere in the art, would refer to a system in which each end of the communication (e.g., host and target) are **not** synchronized to a common clock. Given that Vangemert’s system is configured to reconstruct a clock using the clock recover module and does not include a separate clock line, a PHOSITA would have been motivated to combine Copeland’s expressly asynchronous communication system with

Vangemert's communication system as the systems' communication configurations would mesh well together and ensure proper synchronization of data at each respective node 102-112 and would have had a reasonable expectation of success in doing so.

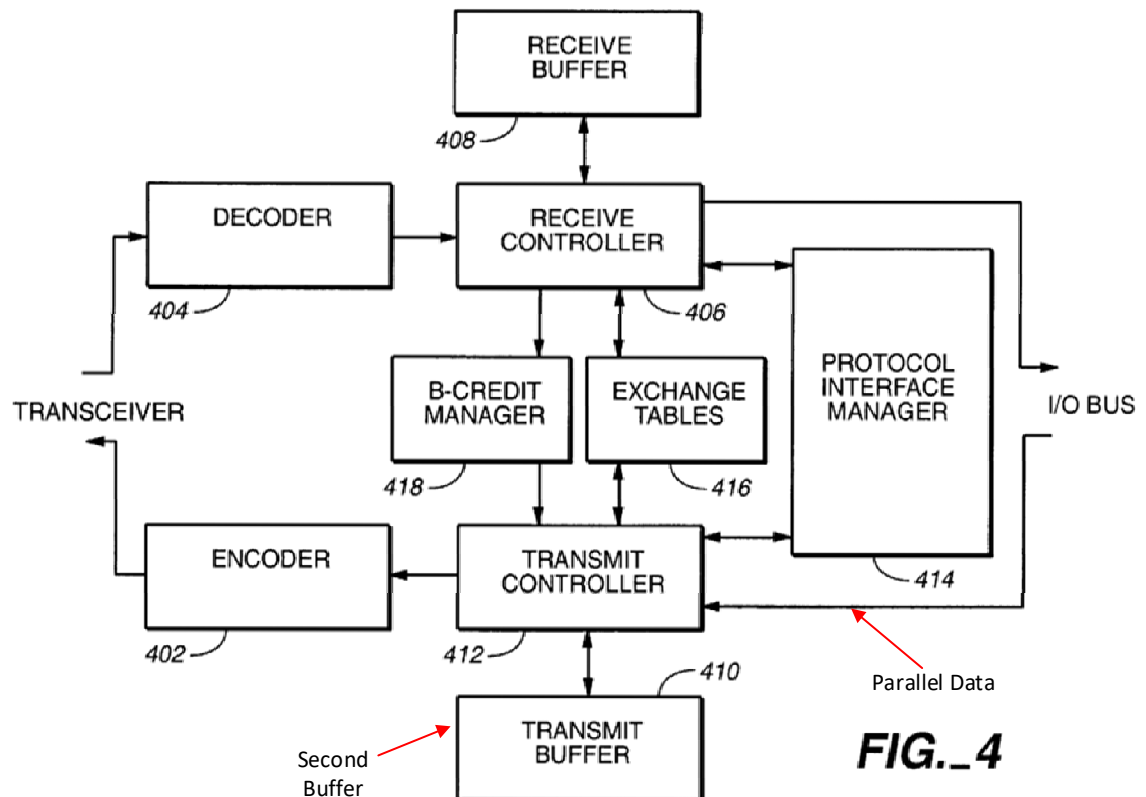
90. Furthermore, asynchronous communication is known in the art to be more cost effective and simple than synchronous serial communication. This is due to the difficulty and cost in installing and maintaining a synchronized clock signal over large distances to prevent loss in synchronization due to losses (e.g., inductive, resistive, capacitive), line noise, etc. In addition, a PHOSITA would understand that implementing a common clock or other required components within multiple devices, such as nodes 102-112 of Vangemert, would require substantial effort and complexity. Accordingly, a PHOSITA, for at least the above reasons, would have been motivated to combine the teachings of Vangemert and Copeland, and would have a reasonable expectation of success in doing so.

91. Accordingly, Vangemert combined with Copeland renders obvious claim 2.

3. Claim 3

The system of claim 1, wherein: the data received by the second buffer comprises parallel data.

92. Vangemert discloses all elements of claim 3 of the '454 patent. As noted above, Vangemert discloses a parallel-to-serial converter for converting data from parallel to serial, where the serial data is then provided to the transmitter.



Vangemert FIG. 4 (Annotated)

93. Specifically, Vangemert states that “[t]ransceiver 300 also includes a parallel-to-serial converter 308 which receives a stream of codewords from an encoder and uses a clock signal from clock multiplier 310 to convert the stream of codewords into a serial bitstream.” *EX. 1005*, at col.3:31-34. This discloses that the data provided to the parallel-to-serial converter 308 is parallel data. Accordingly,

Vangemert discloses that “the data received by the second buffer comprises parallel data,” as claimed. Accordingly, Vangemert combined with Copeland renders obvious claim 3.

4. Claim 4

The system of claim 1, wherein: the module is further operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission.

94. Vangemert teaches transmitting a data frame upon receiving a bit credit signal from a target device. *EX. 1005*, at col. 2:3-6. As already explained above, Vangemert’s “bit credit” term is merely another term for a flow control signal. As explained in Vangemert, the “transmit controller 416 creates or updates exchange information in exchange tables 416 in response to commands and data from the node’s i/o bus, generates control words and data frames in transmit buffer 410, and forwards them to encoder.” *EX. 1005*, at col. 3:57-61. Vangemert does not disclose, teach, or suggest that an acknowledgement signal is required to continue to transmit serial data. The ‘214 patent, upon which the ‘454 patent relies for support, recites similar operations. For example, the ‘214 patent states only that during a write transaction (transmission) the encoder 26 “will add at least two additional bits tagging this information as an addressing cycle. The information is then broken into frames that can carry flow control and other signals before being transmitted serially over link 40.” *EX 1010*, at col. 10:18-25. This disclosure from

the '214 patent is nearly identical to the data transmission operations described in Vangemert in that the data in Vangemert is transmitted, and no requirement for an acknowledgment during the transmission of a specific data packet is expressly recited.

95. A PHOSITA would have also understood that this communication of an entire data packet without requiring intervening acknowledgements is fundamental to serial communication via Fiber Channel as described in Vangemert, which transmits all data in a data frame once the bit credit is issued. Thus, claim 4 is obvious over Vangemert and Copeland.

5. Claim 7

The system of claim 1, wherein: the flow control signal is configured to be used to control transmission of the first serial data over the serial link to prevent overflow of the first buffer.

96. As described in the specification of Vangemert, the discussed protocols use a buffer-credit flow control methodology. Using the methodology, “before a first network node (initiating node) [device transmitting first serial data] can transmit a data frame to a second network node (responding node). The initiating node must first receive a buffer-credit from the responding node, *indicating that the responding node has available space in the receive buffer for a data frame.*” *EX. 1005*, at col. 1:49-54 (emphasis added). This discloses that the flow control signal, in the form of the buffer credit signal, controls transmission of

the first serial data by requiring that the transmitting node must first receive the buffer credit before transmitting. Further, the buffer credit signal, as noted above, is only sent when there is available space in the receive buffer. If there was not enough available space, receipt of the data would overflow the buffer. This is further supported later in the specification of Vangemert states that the buffer credit manager may include “a comparator 502 which asserts a credit signal when it determines that the additional committed buffer space (i.e., the buffer space committed by not yet occupied) is less than the available buffer capacity by at least the size of one data frame.” *EX. 1005*, at col 4:25-29. This further illustrates that the flow control signal is generated to prevent an overflow of the receive buffer.

97. Accordingly, Vangemert discloses that “the module is further operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement,” as claimed.

98. Copeland additionally teaches a flow control signal that is generated in response to a FIFO buffer (first buffer) reaching a predetermined threshold, such as 40%. *EX. 1006*, at col. 5:38-40. As discussed above in **paragraphs 70-72**, the flow control signal is generated by the data flow controller 150 and configured to instruct a remote transmitter to stop sending data (e.g., first serial data). *EX. 1006*, at col. 5:44-6.2. Accordingly, Copeland further teaches a flow control signal that is “configured to be used to control transmission of the first serial data over the

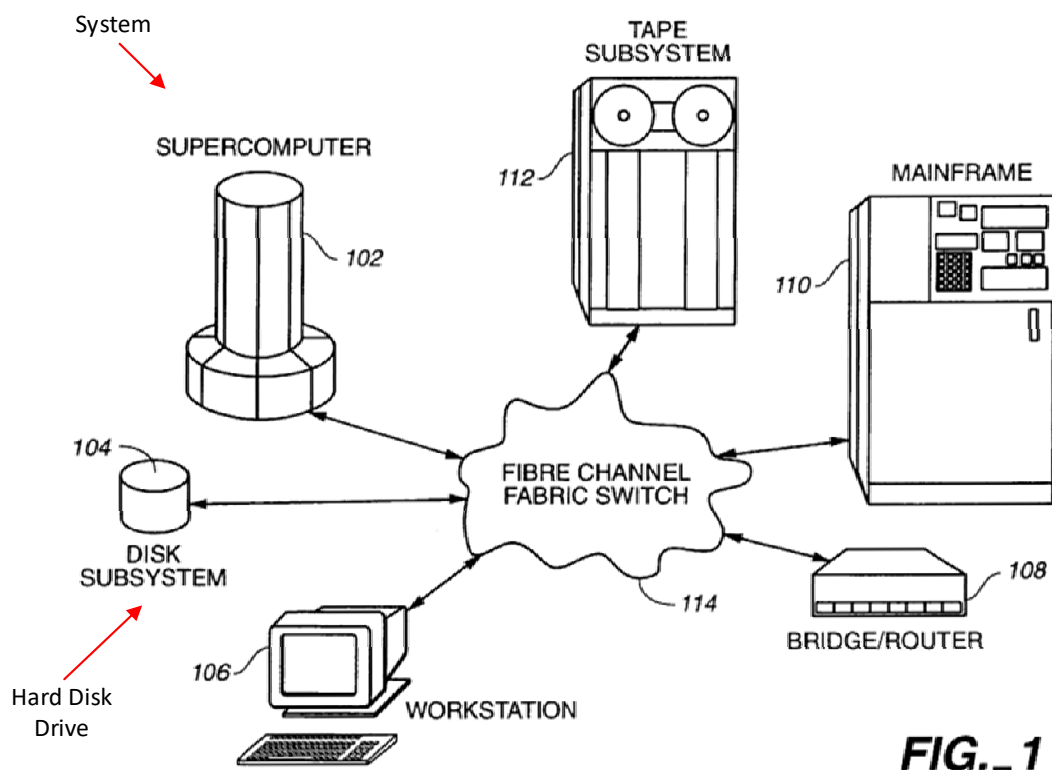
serial link to prevent overflow of the first buffer,” as claimed. Furthermore, as described in **paragraph 72** above, a PHOSITA would have been motivated to combine the teachings of Copeland with those of Vangemert, and would have had a reasonable expectation of success in doing so. Thus, to the extent Vangemert is deemed deficient, a PHOSITA would have been motivated to combine the teachings of Vangemert with those of Copeland, as discussed above, to teach all elements of claim 7.

99. Thus, Vangemert combined with Copeland renders obvious claim 7.

6. Claim 15

The system of claim 1, wherein the system comprises a hard disk drive.

100. Vangemert discloses a hard disk drive within the system. As shown in FIG. 1, the illustrated system is shown in including a disk subsystem 104. A PHOSITA would have understood that a disk subsystem includes a hard disk drive (or multiple hard disk drives) within the system of FIG. 1. Accordingly, Vangemert discloses “wherein the system comprises a hard disk drive,” as claimed, and Vangemert combined with Copeland renders obvious claim 15.



Vangemert FIG. 1 (Annotated)

7. Claim 16

The system of claim 1, further comprising:

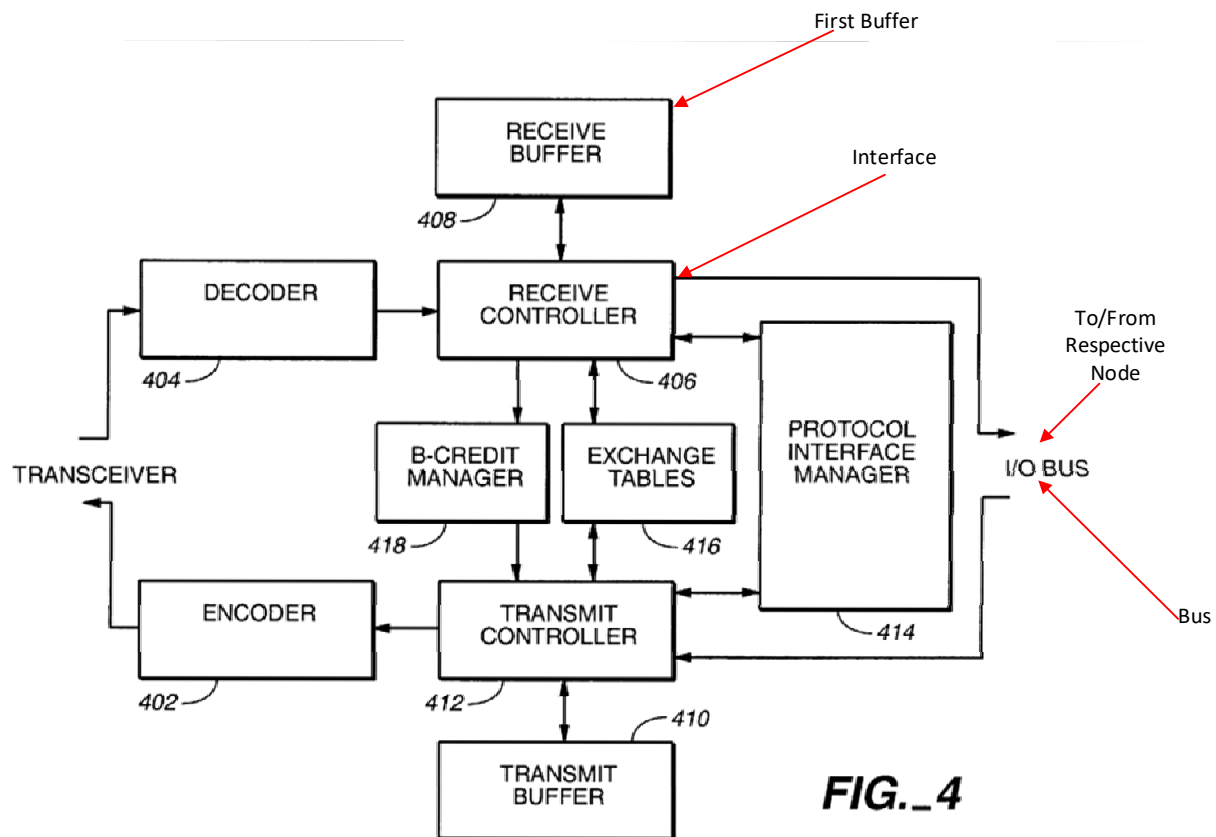
an interface included in the module coupled to the first buffer and operable to receive the data stored in the first buffer;

a bus coupled to the module and configured to receive the data from the interface; and

a processor coupled to the bus.

101. Vangemert discloses a receive controller 406 coupled to the receive buffer 408, and that is configured to store “received control words and data frames in the receive buffer 408.” *EX. 1005*, at col. 3:46-48. The receive controller also

“creates or updates exchange information in exchange tables 416 in response to received control words or data frames, and forwards data from the received data frames to the node’s i/o bus.” *EX 1005*, at col. 3:52-57. Thus, the receive controller 406 is coupled to the receive buffer 408 [first buffer], and is configured to receive data from the receive buffer, as noted above. Accordingly, the receive controller 406 discloses the interface of claim 16. The receive controller 406 is further disclosed to forward “data from the received data frames to the node’s i/o bus.” *Id.* Thus, Vangemert discloses “a bus coupled to the module and configured to receive the data from the interface” of claim 16. This is further illustrated in FIG. 4, which is annotated and reproduced below.



Vangemert FIG. 4 (Annotated)

102. Further, Vangemert discloses “a processor coupled to the bus,” as recited in claim 16. As noted above, the receive controller 406 “forwards data from received data frames to the node’s i/o bus.” *Id.* The referenced node includes the exemplary nodes of FIGs. 1 and 2, which may include a “supercomputer 102.” A PHOSITA would understand the supercomputer 102 includes at least one processor and the supercomputer node would communicate with other nodes through its i/o bus and associated network interface as shown in annotated FIG. 4, above. Thus, Vangemert discloses a “processor coupled to the bus.” Accordingly, Vangemert combined with Copeland renders obvious claim 16.

8. Claim 17

The system of claim 1, wherein: the module is further configured to generate a clock signal, for use by the module, using the first serial data.

103. As disclosed in Vangemert, “[t]ransceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signal levels (first serial data), clean them up as much as possible, and provide them to a clock recovery module 304. Clock recovery module 304 uses the incoming signal to reconstruct a clock signal synchronized to the incoming data, and a serial-to-parallel converter 306 uses the clock signal to convert the incoming signal from serial form to parallel form.” *EX. 1005*, at col. 3:21-28. The clock recovery module 304 therefore generates a clock signal using the first serial data (i.e., received data), which is then used by the transceiver 300 to convert the received data from serial to parallel form. The clock recover module is shown in FIG. 3, below.

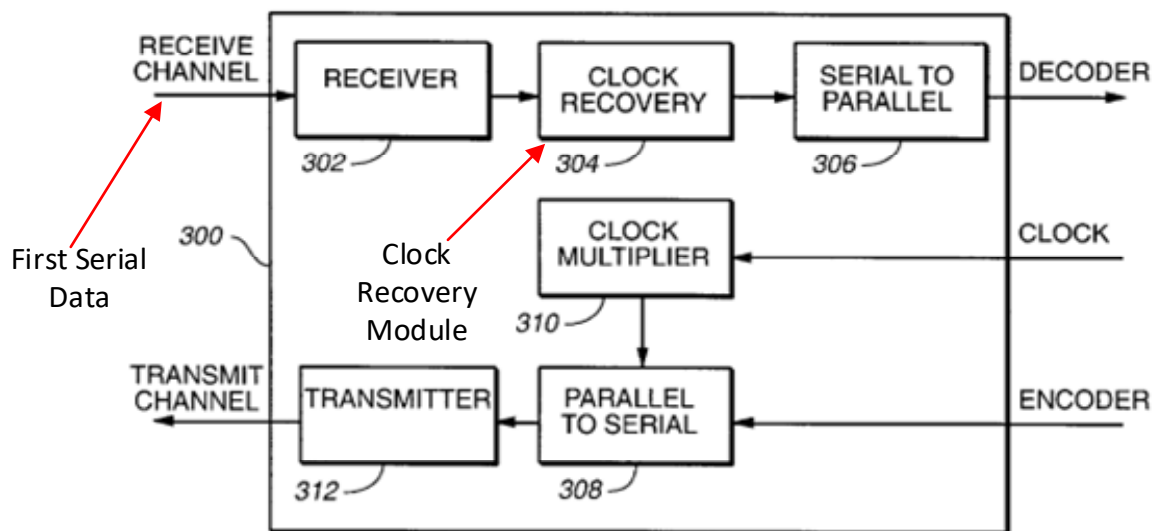


FIG._3

Vangemert FIG. 3 (Annotated)

104. Accordingly, Vangemert discloses “the module is further configured to generate a clock signal, for use by the module, using the first serial data.”

Accordingly, Vangemert combined with Copeland renders obvious claim 17.

B. GROUND 2: Vangemert combined with Copeland and Dobson teaches all elements of Claim 4.

The system of claim 1, wherein: the module is further operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission.

105. To the extent that Vangemert does not expressly state that no acknowledgement is required to transmit an entire serial data packet, Dobson teaches this claim element, as explained below. Dobson states the “fundamental

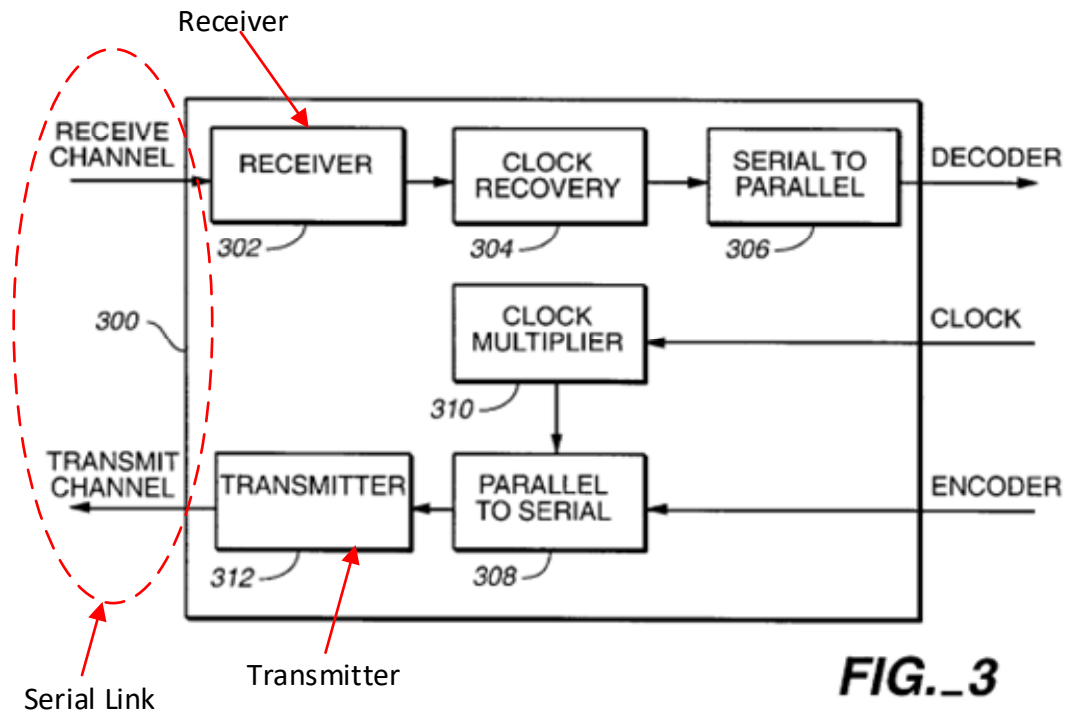
building block of a serial port for a computer or terminal is a universal asynchronous receiver-transmitter, commonly known by the acronym UART. This device accepts parallel words or bytes from the data terminal equipment, generates appropriate timing signals, depending on the bit rate, and transmits the bytes in bit serial form, together with appropriate start, stop and parity checking bits. On the receiver side of the UART, the center of the bit times are determined, based on the approximate data rate, and serial data is clocked into a register from a word or byte that is then available for reading out in parallel by the data terminal equipment.”

EX. 1007, at col. 1:36-47.

106. Dobson also describes the transmission process, which culminates with the data being “input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16 [serial link].” *EX. 1007*, at col. 6:10-12. No acknowledgement or other signal from the receiving device is required to transmit data by the UART 30. Data is simply serialized and transmitted as described above. Further review of the transmission of data by the enhanced serial portion controller 10 of Dobson illustrates that all the various functions of transmitting data do not rely on any acknowledgement (or other) signal being received from any device to transmit the required data. *See, EX. 1007*, at col. 5:52 – 6:16.

107. Thus, Dobson teaches that a module “operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission,” as claimed.

108. A PHOSITA would have been highly motivated to combine the transmission protocol of Dobson with that of Vangemert, to ensure that all data frames within a message stored in the exchange tables 416 can be transmitted within a specific exchange without having to wait for an acknowledgement after each portion of the data is transmitted to meet Vangemert’s goal of minimizing transmission latency. *See EX. 1005*, at col. 1:54-55, col. 3:49-63. Furthermore, as Vangemert, Copeland and Dobson are all directed to serial port controllers having flow control functionality to prevent overflow of a receive buffer, it would have been within the skill of a PHOSITA to combine the teachings of Dobson with that of Vangemert and Copeland, and the PHOSITA would have had a reasonable expectation of success in doing so.



Vangemert FIG. 3 (Annotated)

109. Furthermore, as discussed above, PHOSITA would have understood that the nodes, and therefore their associated transmitters, would be connected to the serial link through a connector, and would therefore have been motivated to combine the connectors 20 or 21 of Copeland with the transceiver of Vangemert, as explained above.

110. Accordingly, Vangemert combined with Copeland and Dobson renders obvious claim 4.

C. GROUND 3: Vangemert combined with Copeland and Reay teaches all elements of Claim 6.

The system of claim 1, wherein: the module is further operable to transmit the first serial data over the serial link and to transfer the second serial data over the serial link using differential signals.

111. As an initial matter, claim 6 appears to have a typographical error.

Claim 6 states that the module is operable to transmit the first serial data over the serial link. However, claim 1, from which claim 6 depends, states that the module is operable to “receive first serial data transmitted over the serial link.”

Accordingly, I assume that the data is received by the module (and transmitted by an external device), and that it is done over the serial link using differential signals.

112. As noted above, in Vangemert the “fabric switch 114 transports data frames and control words from the transmit channel of a source node to the receive channel of a destination node.” *EX. 1005*, at col. 2:65 – col. 3:1. Thus, Vangemert discloses that data is transmitted and received by each of the nodes (e.g., modules) shown in FIGS. 1 and 2. Further, as noted above, the nodes, (i.e., nodes 102, 104, 106, 108, 112, and/or 114) are coupled to “a serial link having a receive channel and a transmit channel.” *EX. 1005*, at col. 2:61-64. Accordingly, Vangemert discloses that serial data (e.g., data over the serial link) is transmitted and received at each module.

113. FIG. 3, and the associated description, are further illustrative on this point. As noted in Vangemert, “FIG. 3 illustrates a serial communication

transceiver 300 which may be used by the nodes to couple to the serial communications link. Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming single levels. . . .” *EX. 1005*, at col. 3:19-23. The transceiver 300 further includes a transmitter 312 that “converts the serial bitstream into signals suitable for transport by the transmit channel.” *EX. 1005*, at col. 3:35-36. The above noted receiver 302 and transmitter 312 allow for reception of serial data and transmission of data over the serial link.

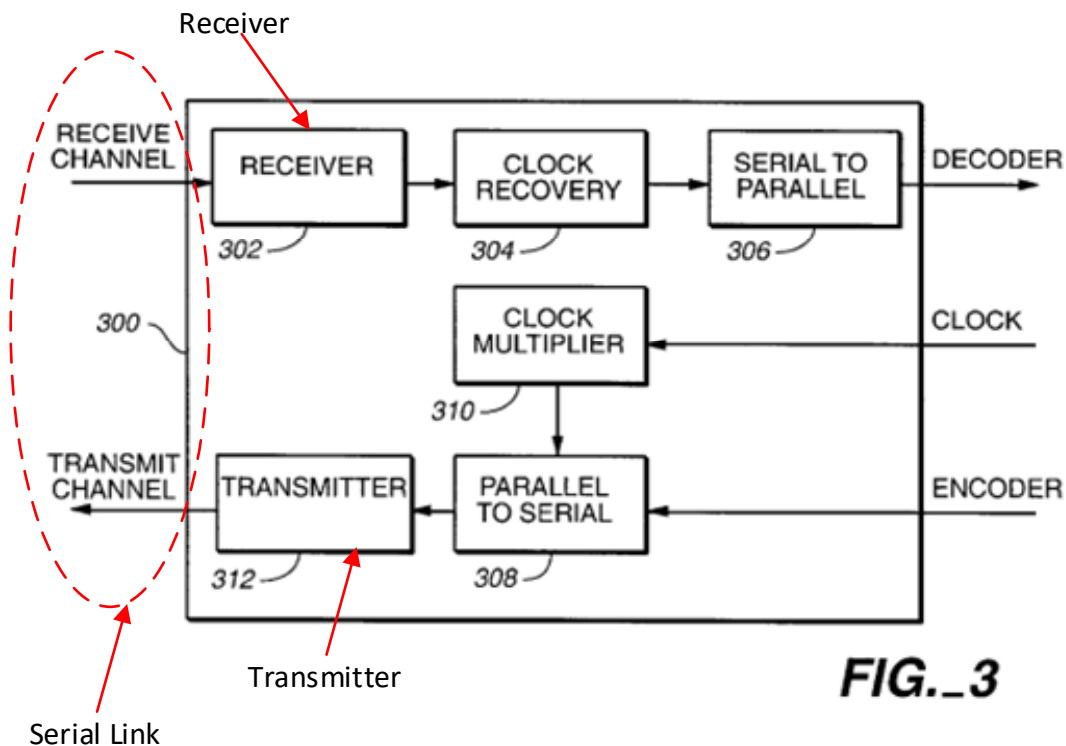


FIG._3

Vangemert FIG. 3 (Annotated)

114. Reay teaches transmitting serial data “using differential signals,” as recited in claim 6. Specifically, Reay explains that “[o]utput terminals 106 and 108 provide inverted and non-inverted output signals, respectively, of the input signal

received at terminal 102. Thus, a differential, or balanced-line, output can be provided for those interface standards that require such output.” *EX. 1008*, at col. 4:15-20. Reay goes on to explain that the “advantage to such outputs is noise immunity; any noise that appears on one of the two output lines is likely to appear on the other as well and is thus cancelled as the difference between the two output signal voltages is taken.” *EX. 1008*, at col. 4:32-35. Accordingly, a PHOSITA would have found it obvious and been motivated to combine the noise cancelling differential signals of Reay for use on the serial link to reduce errors due to noise with respect to the received and/or transmitted signals. Furthermore, a PHOSITA would have had a reasonable expectation of success in incorporating the differential signaling of Reay into the serial link of Vangemert, as the use of differential signaling for serial communication was well known in the art at the time of Vangemert and Reay.

115. Accordingly, Vangemert, Copeland, and Reay in combination teach “the module is further operable to transmit the first serial data over the serial link and to transfer the second serial data over the serial link using differential signals.”

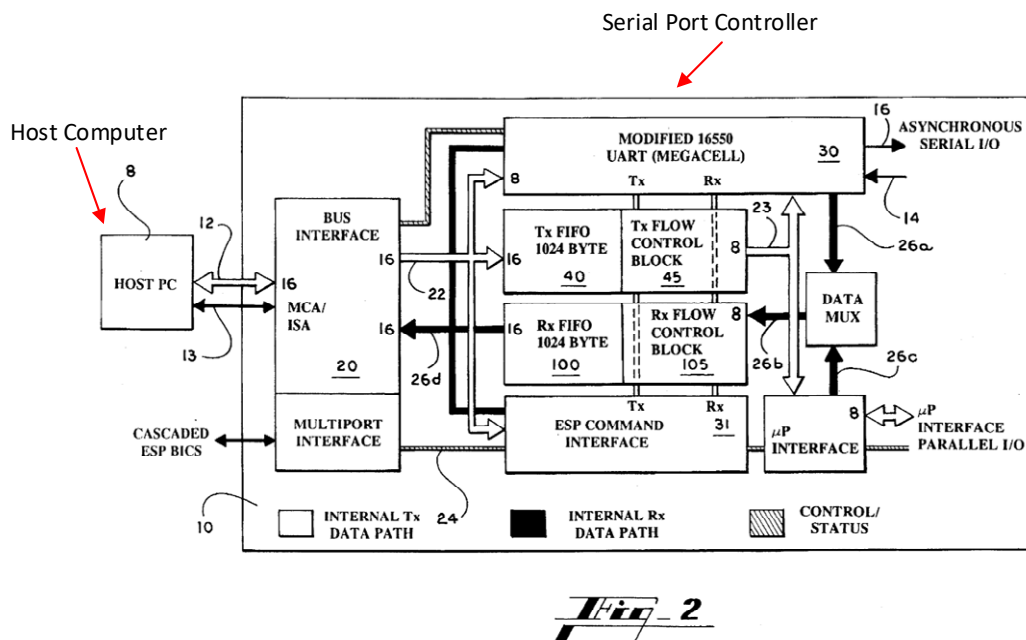
116. Thus, Vangemert, Copeland, and Reay in combination render obvious claim 6.

D. GROUND 4: Dobson combined with Copeland teaches all elements of Claims 1, 2, 3, 4, 7, and 16.

1. Claim 1

a. *A system comprising:*

117. Dobson discloses a system including an enhanced serial port controller 10 in communication with various devices, such as host computer 8. *EX. 1007*, at col. 5, lines 17-26. The system is illustrated in annotated FIG. 2, below.



Dobson FIG. 2 (Annotated)

b. *a connector configured to be connectable to a serial link;*

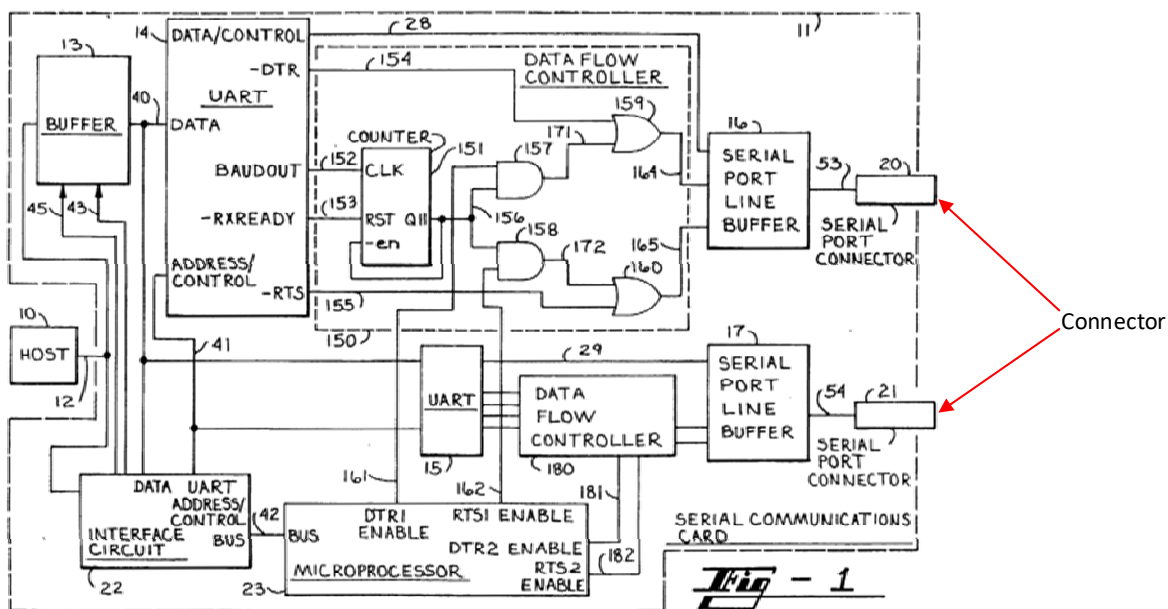
118. Dobson, as shown in annotated FIG. 1, below, discloses a serial link in the form of lines 14 (receive) and 16 (transmit). For example, Dobson states that

“[d]ata is input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16.” *EX. 1007*, at col. 6:10-12. With respect to the receive line 14, Dobson states that the “enhanced serial port controller 10 also accepts asynchronous, serial data on line 14.” *EX. 1007*, at col. 7:54-56. Thus, the combination of the transmit line 16 and the receive line 14 teaches a serial link, as the transmit and receive lines facilitate transmission and reception of serial data, as described above. A PHOSITA would have known that one or more connectors would have been required between the enhanced serial interface of Dobson and one or more other pieces of data communications equipment. *See EX. 1007*, at col. 1:23-26.

119. Copeland teaches a connector configured to be connectable to a serial link, as claimed. Copeland discloses a dual channel asynchronous serial communications card 11 that includes “two serial port connectors 20 and 21, an interface circuit 22, and a microprocessor 23.” *EX. 1006*, at col. 3:59-60. The serial port connectors 20 and 31 are further describes as being “connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” *EX. 1006*, at col. 4:31-34. As noted above, RS-232-D connectors were well known in the art as common connectors for interconnecting serial devices as the RS-232-D standard at least as far back as Copeland’s filing in 1989. *See EX. 1006*, at 1. Thus, the serial port

connectors 20 and 21 are configured to be coupled to a serial link, such as an RS-232-D serial interface.

120. Furthermore, as shown in annotated FIG. 1 of Copeland, below, the serial port connectors 20, 21 are coupled to UARTs 14 and 15, respectively. For example, serial port connector 20 is coupled to UART 14 via the data/control line 28, as shown below. Copeland teaches a “dual channel asynchronous serial communications card 11” for communicating with various devices, such as modems. *EX. 1007*, at col. 3:49-63. Thus, a PHOSITA would have understood that serial port connector 20 is associated with a first channel of the dual channel serial communications card 11, and serial port connector 21 is associated with a second channel of the dual channel serial communications card 11 and may therefore be used interchangeably. While references made herein are to serial port connector 20, it is understood that serial port connector 21 provides similar functionality. Accordingly, Copeland teaches “a connector configured to be coupled to a serial link.”



Copeland FIG. 1 (Annotated)

121. A PHOSITA would have found it obvious to combine the connectors 20 or 21 of Copeland with the serial lines 14 and 16 of Dobson to provide a required mechanism for connecting the serial lines 14 and 16 of the serial port controller 10 to one or more external serial devices. Both Dobson and Copeland have the same U.S. Classification, and sub-classification (395/250), and Dobson cited Copeland. *See EX. 1006*, at 1; *EX. 1007*, at 1. More specifically, both Dobson and Copeland are directed to serial interfaces utilizing 16550 UARTs to provide a serial interface for a host device, and the enhanced serial device 10 of Dobson would have required such a connector for interfacing with various external devices, such as modems. *See EX. 1007*, at col. 3:20-52; *EX. 1006*, at col. 2:35-47. In fact, Dobson states that one of its objectives is to “provide an improved serial

port device with expanded buffering capability that is backwardly compatible with apparatus and application programs written for use with conventional UARTs.”

EX. 1007, col. 4:21-25. Thus, a PHOSITA would have been motivated to combine the connectors of Copeland with the system of Dobson to ensure backwards compatibility with systems such as Copeland which include both a 16550 UART and the associated connectors.

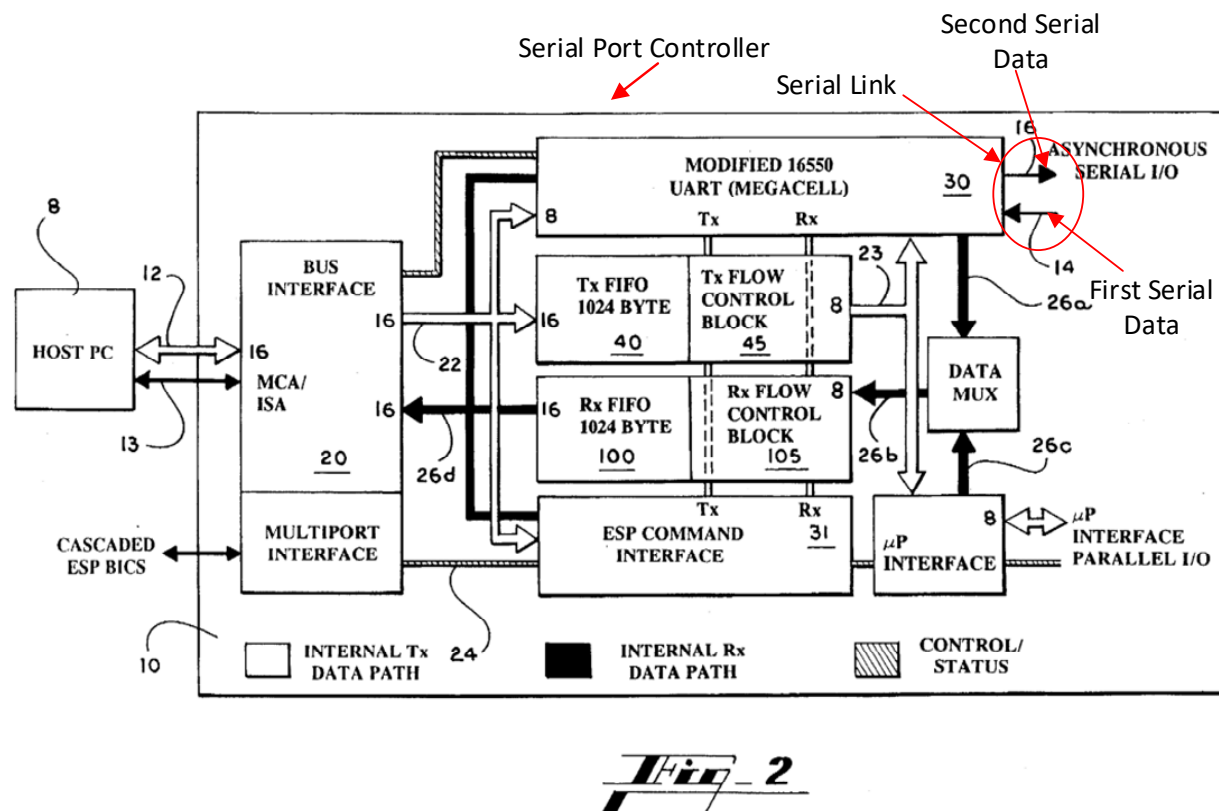
122. Additionally, because both Dobson and Copeland use similar 16550 UARTs, and Dobson was designed to be backwards compatible with systems using similar UARTs, such as a 16550, a PHOSITA would have had a reasonable expectation of success in modifying Dobson’s serial port controller with Copeland’s data flow controller to both prevent overflow of a receive buffer and reduce the host’s flow control workload. Additionally, as noted above, the use of connectors was well known in the art, and therefore provide more credence that a PHOSITA would have had a reasonable expectation of success in combining the connectors of Copeland with the serial link of Dobson.

- c. *a module coupled to the connector, wherein the module is operable to receive first serial data transmitted over the serial link through the connector and transmit second serial data to the serial link through the connector, and wherein the module includes*

123. As shown in the below annotated version of Dobson FIG. 2, the serial link (transmission line 16 and reception line 14), which communicates with other

I/O devices, provides serial communications to and from the serial port controller

10. Accordingly, the serial port controller 10 discloses the module of claim 1.



Dobson FIG. 2 (Annotated)

124. As noted above, serial line 14 is configured to act as a serial receive line. This is explained in Dobson, which states that “[t]he receive side of the enhanced serial port controller 10 also accepts asynchronous, serial data on line 14.” *EX. 1007*, at col. 7, lines 53-56. This serial data received on line 14 would be understood by a PHOSITA to be the first serial data received over the serial link.

125. As further explained in Dobson, data is “input to the UART 30 in parallel, [and] is converted by the UART 30 to serial and is output from the UART 30 on line 16.” *EX. 1007*, at col. 6:10-12. This serialized data that is output from the UART as serial data on line 16 discloses “transmit second serial data to the serial link through the connector.” Additionally, as discussed above, a PHOSITA would have understood that the serial link of Dobson would be connected to various external devices, such as a modem, through a connector such as the connectors 20 and 21 of Copeland.

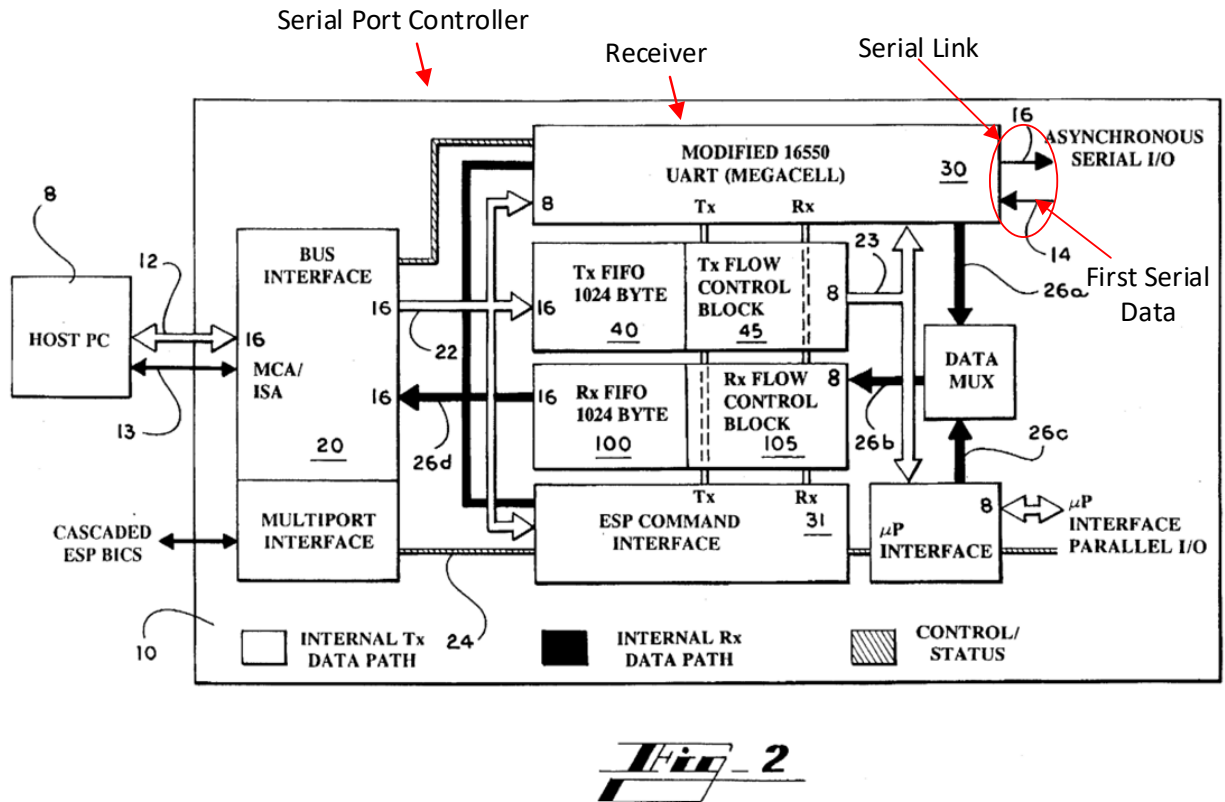
126. Copeland further teaches that the serial port connectors 20, 21 are coupled to a serial communications card 11, which provides serial communications between devices. *EX. 1006*, at col. 3:46-63. Thus, Copeland teaches coupling a module to the connectors 20, 21. A PHOSITA would have known to combine the connector 20 or 21 of Copeland with the serial port controller 10 of Dobson to facilitate connections between the serial port controller 10 of Dobson and one or more other devices, such as a modem, as discussed above.

d. *a receiver operable to receive the first serial data from the serial link through the connector;*

127. As shown in Dobson annotated FIG. 2, below, the enhanced serial port controller 10 includes a universal asynchronous receiver and transmitter (“UART”) 30. The UART 30, by definition, act as a receiver and/or a transmitter. The reception function of the UART 30 is described in Dobson, which states that

“[a]synchronous data received on line 14 is passed to the UART 50.” *EX. 1007*, at col. 7:56-57. Note that there appears to be a discrepancy of the numbering of the UART 30 as UART 50 in portions of Dobson. There is no UART 50 in the specification. Reference number 50 is used to refer to the write generator 50. Therefore, I assume that the three mentions of UART 50 were typographical mistakes and were intended to mean UART 30. Dobson further confirms the reception of serial data by the UART 30 stating that “[d]ata arriving at the enhanced serial port controller 10 on asynchronous serial line 14 (FIG. 2) and is fed through the serial to parallel converter 90 in the UART 30.” *EX. 1007*, at col. 7:67 – 8:3. In addition, a PHOSITA would have understood that the enhanced serial controller 10, and therefore the associated receiver (UART 30) would be connected to the serial link, including the receive line 14, through a connector, such as the connectors 20 and 21 of Copeland, as described above. Accordingly, a PHOSITA would have found it obvious, and been motivated, to combine the connector 20 of Copeland with Dobson’s enhanced serial port controller 10 to facilitate communication via the serial data lines 14 and 16. Furthermore, as connectors were well known in the art at the time of Dobson and Copeland, a PHOSITA would have had a reasonable expectation of success in combining the connector 20 of Copeland with the enhanced serial port controller 10 of Dobson.

128. Thus, Dobson in combination with Copeland discloses “a receiver operable to receive the first serial data from the serial link through the connector” in the form of UART 30.

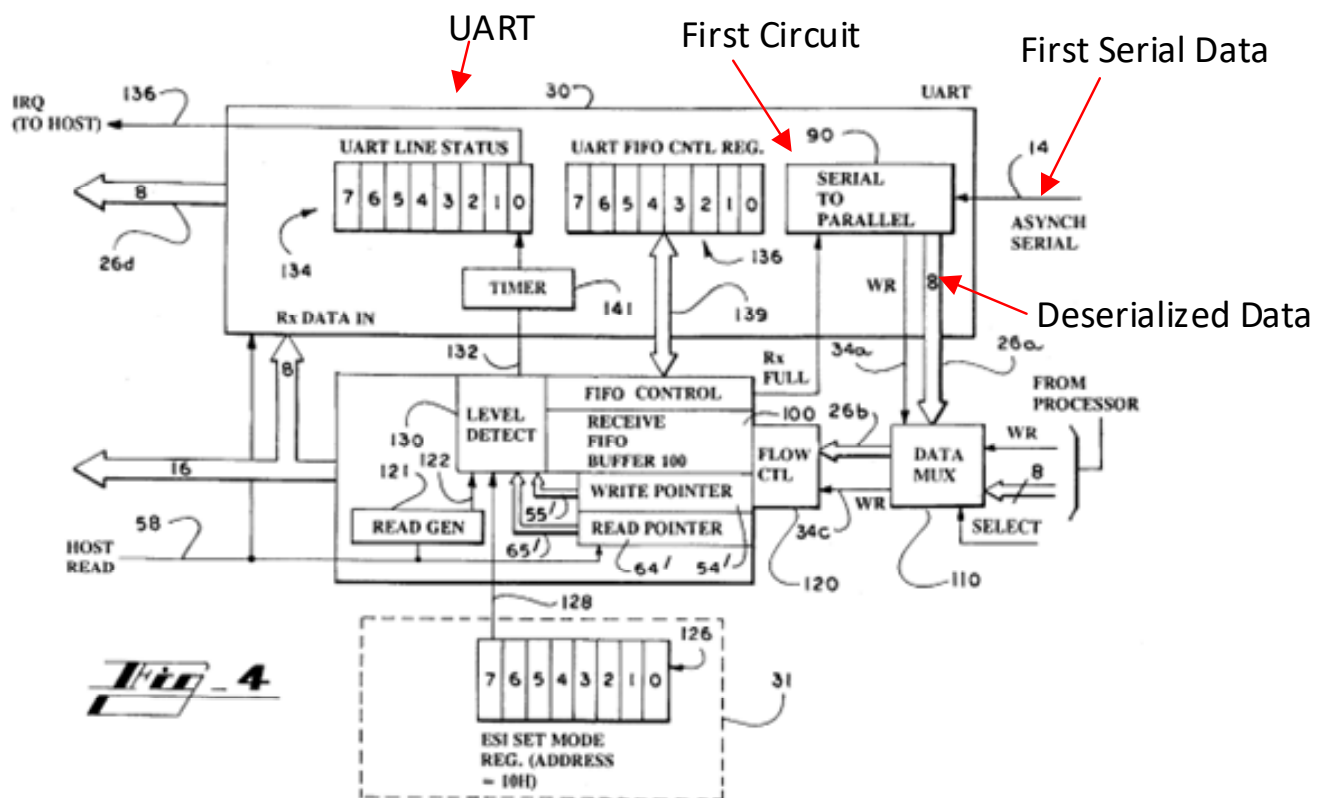


Dobson FIG. 2 (Annotated)

e. a first circuit configured to deserialize the first serial data to generate deserialized data;

129. Dobson discloses the above element of claim 1. The UART 30 of Dobson is described as having a serial to parallel circuit 90, which converts serial to parallel data (i.e., deserializes serial data.). Annotated FIG. 4, below, shows the serial to parallel converter 90. This is further supported by Dobson, which states

that “data arriving at the enhanced serial port controller 10 on asynchronous serial line 14 (FIG. 2) and is fed through the serial to parallel converter 90 in the UART 30. . . . The data is passed from the serial to parallel converter 90 is [sic] to the data multiplexor 110 over data bus 26a.” *EX. 1007*, at col. 7:67 – 8:6. As shown in annotated FIG. 4, the data provided to the data multiplexor 110 is 8-bit parallel data, which would disclose the deserialized data element of claim 1. Accordingly, a PHOSITA would understand that the serial to parallel converter of the UART 30 of Dobson discloses the “first circuit configured to deserialize the first serial data to generate deserialized data.”



Dobson FIG. 4 (Annotated)

f. *a decoder configured to decode the deserialized data;*

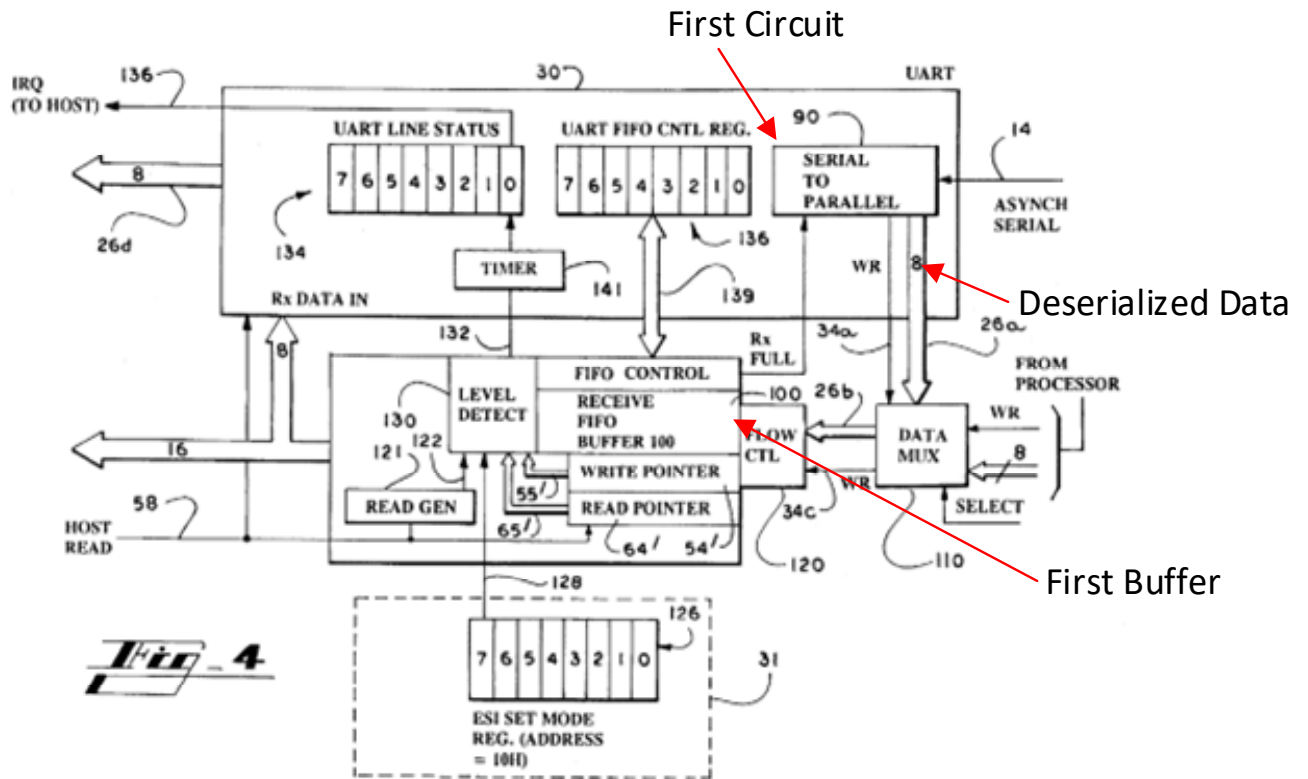
130. Dobson discloses the above element of claim 1. As shown in at least FIG. 4 of Dobson, serial data is received at the UART 30. As described above, the UART, via the serial to parallel converter 90 deserializes the received serial data. *See EX. 1007*, at col. 7:65 – col. 8:6. While not shown, Dobson discloses that a UART, upon deserializing received serial data, “responds to and strips start and stop bits and can give an indication of when a parity error is detected.” *EX. 1007*, at col. 1:47-49. This stripping out of start and stop bits decodes the serial data by isolating the data package from the associated control bits. This stripping of start of stop bits from the data package results in the original data packet being restored to its desired format by removing any control bits (i.e., start and stop bits), thereby decoding the received serial data.

131. Accordingly, Dobson discloses “a decoder configured to decode the deserialized data,” as claimed.

g. *a first buffer operable to store data that has been deserialized and decoded by the first circuit and the decoder, respectively;*

132. As described above, the received data, upon being converted to parallel data by serial to parallel converter 90 and subsequently decoded, is written into a receive buffer 100. *See EX. 1007*, col. 8:5-29. Specifically, Dobson discloses that the “data is clocked into the receive FIFO buffer 100 using the write signal on

line 34a.” *EX. 1007*, col. 8:24-25. The receive FIFO buffer 100 of Dobson therefore discloses “a first buffer operable to store data that has been deserialized and decoded by the first circuit and the decoder, respectively.” Annotated FIG. 2 of Dobson, shown below, illustrates the receive FIFO buffer (first buffer) which stores data that has been deserialized by the serial to parallel circuit 90 of the UART 30, and decoded by the read generator circuit 121, write pointer 55’ and read pointer 64’.



Dobson FIG. 4 (Annotated)

- h. a second buffer operable to receive data, wherein the second buffer is in communication with a second circuit

that is configured to serialize data received via the second buffer to generate the second serial data; and

133. Dobson discloses a transmit FIFO buffer 40 configured to receive data from the bus interface, which is in communication with the host PC 8. *See EX. 1007*, at col. 6:17-22, and FIG. 2. The transmit FIFO buffer 40 is configured to receive data from the transmit data bus 22 which is “provided to the UART 30, the transmit FIFO buffer 40, and the command interface 50.” *EX. 1007*, at col. 6:1-3. The data on transmit data bus 220 arrives from the host personal computer 8. *See EX. 1007*, at col. 5:59-60. Thus, the transmit FIFO buffer 40 discloses “a second buffer operable to receive data.”

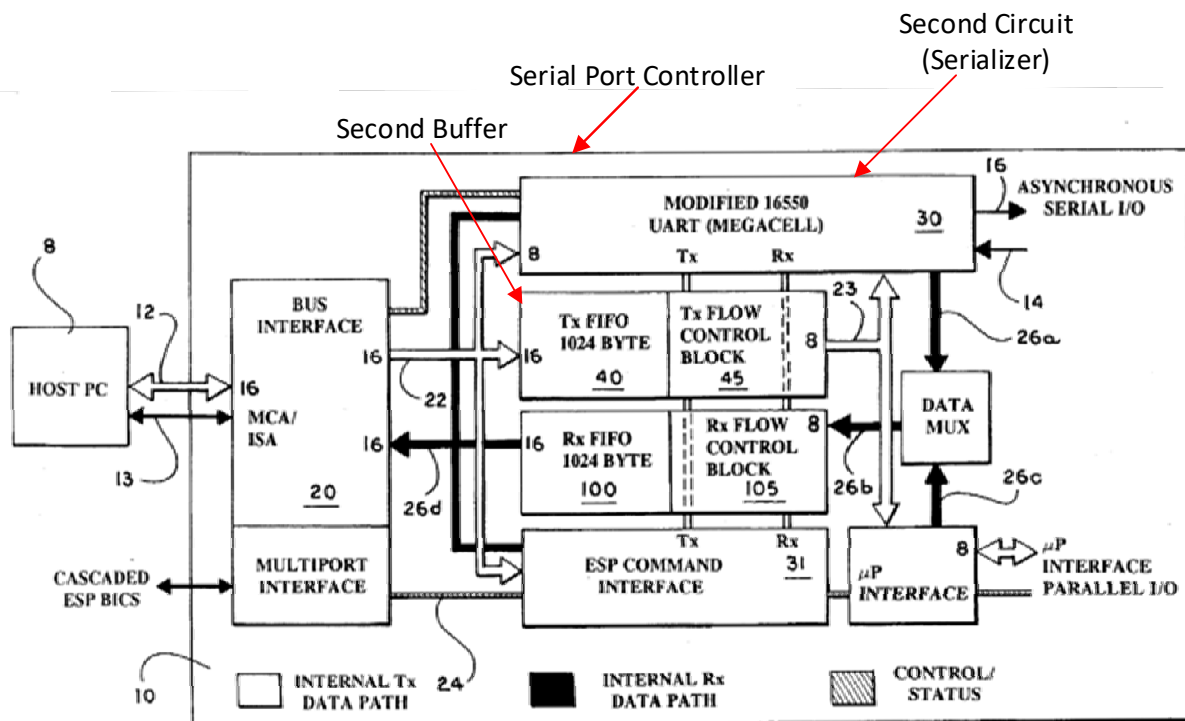


FIG. 2

Dobson FIG. 2 (Annotated)

134. Dobson further discloses that the second buffer is “in communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data,” as recited in claim 1. Dobson explains that the “data stored in the transmit FIFO buffer 40 [second buffer] is output on data bus 23 to the UART 30. As will [sic] familiar to those skilled in the art, the data is converted from parallel to serial in the UART 30 and output on line 16.” *EX. 1007*, at col. 7, lines 1-4. Thus, while not shown, the UART is disclosed to include a second circuit, similar to serial to parallel circuit 90 shown in FIG. 4 and

described above. This parallel to serial circuit in UART 30 therefore discloses the second circuit of claim 1. Further, as illustrated in the above passage of Dobson, this second circuit is in communication with the transmit FIFO buffer 40, which discloses the “second buffer” of claim 1. Accordingly, Dobson discloses “wherein the second buffer is in data communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data.”

- i. *a transmitter operable to transmit the second serial data to the serial link through the connector;*

135. As noted above, UART 30 is a UART, which stands for “Universal Asynchronous Receiver and Transmitter.” See *EX. 1007*, at col. 5:17-21. Thus, the UART, by definition, includes a transmitter. See, e.g., *EX. 1007*, at col. 5:18-22. This is further supported within Dobson, which explains that “[d]ata is input to the UART 30 in parallel, [and] is converted by the UART 30 to serial and is output from the UART 30 on line 16.” *Id.*, col. 6:10-12. As noted above, line 16 is the transmit portion of the serial link, with Dobson explaining that “[d]ata written into the FIFO buffer 40 is read from the buffer for transmission on the asynchronous serial data output line 16.” *EX. 1007*, at col. 6:49-51. Furthermore, as discussed above, a PHOSITA would have understood that the UART 30, and therefore its associated transmitter, would be connected to the serial link through a connector, and would therefore have been motivated to combine the connectors 20 or 21 of Copeland with the receiver of Dobson, as described above. Accordingly, a

PHOSITA would have found it obvious, and been motivated, to combine the connector 20 of Copeland with Dobson's enhanced serial port controller 10 to facilitate communication via the serial data lines 14 and 16. Furthermore, as connectors were well known in the art at the time of Dobson and Copeland, a PHOSITA would have had a reasonable expectation of success in combining the connector 20 of Copeland with the enhanced serial port controller 10 of Dobson.

136. Thus, Dobson discloses a transmitter operable to transmit the second serial data to the serial link through the connector.

- j. *wherein the module is configured to determine whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer;*

137. Dobson discloses this element of claim 1. Specifically, Dobson discloses an apparatus for "detecting the level of data in the receive FIFO buffer 100 [first buffer]." *EX. 1007*, at col. 8:53-57. Dobson goes on to explain that the "value on bus 147 is compared with the number of data bytes written into the receive FIFO buffer 100 . . . [w]hen the number of bytes written into the receive FIFO buffer 100 [first buffer] equals the selected scalable trigger level on bus 147, the comparator 138 outputs the trigger level reached signal on line 132." *EX. 1007*, at col. 9:32-39. By comparing the bytes in the first buffer (FIFO buffer 100) against a predetermined trigger level, Dobson discloses determining "whether an

amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer” via the comparator 138.

138. The selected scalable trigger level on bus 147 is associated with a storage capacity of the first buffer. This is described in at least the Abstract of Dobson, which states that “[s]tandard selectable trigger levels used in a type 16550 UART are provided as well as expanded scalable trigger levels to accommodate the larger buffers.” *EX. 1007*, at Abstract. Thus, the selectable trigger level is associated with a storage capacity of a buffer, such as the first buffer 100. Accordingly, Dobson discloses “wherein the module is configured to determine whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer,” as claimed.

139. Copeland also teaches a module “configured to determine whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer,” as recited in claim 1. Copeland teaches a flow control operation for the FIFO buffer (first buffer) of UART 14 which is “activated when the FIFO buffer [of UART 14] is forty percent full.” *EX. 1006*, at col. 5:38-40. The FIFO buffer of UART 14 is analogous to the “first buffer” of claim 1 as the FIFO buffer of UART 14 is configured to receive data provided by external devices and drives the flow control signal to be generated based on a fill level thereof,” as described in more detail below. *See EX. 1006*, at 3:3-8; 5:52-55.

140. Copeland's fill amount, such as the forty percent amount recited above, is determined based on the amount of data in the FIFO buffer of UART 14 (first buffer). Specifically, Copeland explains that "UART 14 has another signal, BAUDOUT, which is a clock at 16 times the baud rate. When -RXREADY goes low counter 151 counts the BAUDOUT cycles. When 1024 BAUDOUT cycles pass without the FIFO buffer being empty the Q11 output of the counter 151 goes high. The count of 1024 BAUDOUT cycles represents 64 bit times or, at 10 bits per character, 6.4 character times. This leaves at least 9.6-character times remaining for the remote transmitter connected to serial data port 20 to respond before the FIFO buffer overflow. Thus, in the preferred embodiment of the present invention, data flow control is activated when the FIFO buffer is forty percent full." *Id.* at col. 5:28-40. Thus, Copeland teaches determining, based on a predetermined number of BAUDOUT cycles passing without the FIFO buffer (first buffer) being emptied, "whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer," as recited in claim 1.

k. *wherein the module is operable to generate a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount.*

141. Dobson teaches generating a host interrupt signal in "response to the amount of data stored in the first buffer equaling or exceeding the fill amount," as

recited in claim 1. As explained in Dobson, the “value on bus 147 is compared with the number of data bytes written into the receive FIFO buffer 100 . . . [w]hen the number of bytes written into the receive FIFO buffer 100 [first buffer] equals the selected scalable trigger level on bus 147, the comparator 138 outputs the trigger level reached signal on line 132. Referring to FIG. 4, the trigger level reached signal on line 132 is fed to the UART 30 to cause the host computer to be interrupted.” *EX. 1007*, at col. 9:32-42. As described in the above passage of Dobson, in response to the comparator 138 determining that the number of bytes written into the receive buffer (i.e., first buffer) equals the predetermined selected scalable trigger level (i.e., fill amount), the trigger level signal is generated.

142. Copeland teaches “wherein the module is operable to generate a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount,” as recited in claim 1. As noted above, Copeland teaches a flow control operation for the FIFO buffer (first buffer) of UART 14 which is “activated when the FIFO buffer [of UART 14] is forty percent full.” *EX. 1006*, at col. 5:38-40. As also noted above, the FIFO buffer of UART 14 is analogous to the “first buffer” of claim 1. *See EX. 1006*, at 3:3-8; 5:52-55.

143. Copeland explains that the UART 14 includes a 16 byte FIFO buffer and outputs a signal “-RXREADY,” which is low “whenever there is at least one byte of data, transmitted from serial data port 20 to the UART 14 FIFO buffer,

which has not been read by host 10.” *Id.* at col. 5: 23-28. Thus, whenever the FIFO buffer of UART 14 is not completely empty, -RXREADY is low. Copeland explains that “UART 14 has another signal, BAUDOUT, which is a clock at 16 times the baud rate. When -RXREADY goes low counter 151 counts the BAUDOUT cycles. When 1024 BAUDOUT cycles pass without the FIFO buffer being empty the Q11 output of the counter 151 goes high. The count of 1024 BAUDOUT cycles represents 64 bit times or, at 10 bits per character, 6.4 character times. This leaves at least 9.6 character times remaining for the remote transmitter connected to serial data port 20 to respond before the FIFO buffer overflow. Thus, in the preferred embodiment of the present invention, data flow control is activated when the FIFO buffer is forty percent full.” *Id.* at col. 5:28-40.

144. Copeland goes on to explain that “[w]hen a count of 1024 is reached the Q11 output provides an enabling high signal to gates 157 and 158. If microprocessor 23 is providing a second enabling high signal to gates 157 and 158 over Data Terminal Ready 1 (DTR1) line 161 and Request To Send 1 (RTS1) line 162, the output of gates 157 and 158 provide high signals, respectively, to gates 159 and 160, respectively, which disables gates 159 and 160, thereby causing their outputs to be high, which sends a low for the RTS [shorthand for Request to Send] **[flow control signal]** and DTR [shorthand for Data Terminal Ready] signals on serial port connector 20. This signals the remote transmitter to stop sending data,

thus shutting down the flow of data from the serial data port 20 to the UART 14 [first buffer] FIFO buffer.” *EX. 1006*, at col. 5:43-55.

145. Copeland further explains that “[i]t will therefore be appreciated that whenever host 10 is unable to keep up with the data transfer rate, the Q11 output of counter 151 will go high, thereby causing the DTR and/or TRS output signals on serial port connector 20 to go low and signal the remote transmitter to stop sending.” *EX. 1006*, at col. 5:65 – col. 6:2. Thus, a PHOSITA would understand the RTS signal and/or the DTR signal is a flow control signal because it can instruct a remote device to stop the flow of data when driven low. Further, the flow control signal is generated in response to a fill level of the UART 14 FIFO reaching (i.e., equaling or exceeding) a predetermined value (e.g., 40%) [fill level] to prevent FIFO buffer overflow. *See, id.* Thus, Copeland teaches generating “a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount,” as recited in claim 1.

146. A PHOSITA would have found it obvious to combine Copeland’s flow control signal with Dobson’s host interrupt driven system for multiple reasons. First, and foremost, both Dobson and Copeland are directed to the same technology. Specifically, both are directed to serial port controllers utilizing UARTs, and specifically, 16550 UARTs, to control a serial interface of a host device, such as host PC 8 of Dobson or host 10 of Copeland. *See EX. 1007*, at

Abstract; *EX. 1006*, at col. 2:41-47. In fact, both Copeland and Dobson are in the same technical classification and sub-classification (395/250). *EX. 1006*, at 1; *EX. 1007*, at 1. To further illustrate the similarity in technologies, not only are both Copeland and Dobson directed to the same technology (serial port controllers), using the same componentry (16550 UARTs), in the same technical classification, but the technology described in both Copeland and Dobson was developed by the same applicant – Hayes Microcomputer Products, Inc. *EX. 1006*, at 1; *EX. 1007*, at 1.

147. Furthermore, both Dobson and Copeland are directed to similar, and complementary, goals. Dobson is directed to reducing the number of interrupts (e.g., read commands) to the host PC 8 by providing additional receive buffer capacity, thereby allowing more data to be stored before a read interrupt is required and reducing the strain on the processor to address said interrupts. *EX. 1007*, at col. 3:33-52. Dobson explains that an object of the invention is “to provide a serial port with expanded receive and/or transmit buffering with scalable trigger levels for service requests that can physically allow for greater amounts of data to be buffered [to] internally restrict[] the rate of its own generation of interrupt requests so as to prevent a thrashing condition in which serial port servicing occupies an inordinate amount of CPU processing time.” *EX. 1007*, at col. 4:27-37. This restriction in the rate of interrupt request generation “provides additional

data absorption capability that prevents the overwriting of received data while the generation of the new service request is inhibited. *Id.* at col. 4:38-43. Thus, Dobson not only reduces workload of the host with respect to flow control, but also attempts to prevent overwriting of data due to overflow conditions by providing additional buffering capacity along with limited interrupts that reduce the likelihood the buffer will overflow.

148. In a similar vein, Copeland describes a goal of relieving the host device of having to manage flow control by providing specific circuitry, such as the data flow controller 150 to handle flow control. For example, Copeland states that the described invention “automatically signals the remote transmitter to stop sending [data] when the FIFO buffer fills to a certain point and to resume sending once the FIFO buffer has been completely emptied. This relieves the host of the duty of managing the flow control.” *EX. 1006*, at col. 3:3-8. This flow control scheme of Copeland is further intended to signal “the remote transmitter to halt the flow of [] data in a timely manner such that data loss is prevented.” *Id.* at col. 2:25-28.

149. A PHOSITA would have been motivated to combine these complementary designs as they would work together to form a bulwark against data loss due to buffer overflow by increasing buffer capacity to reduce the amount of interrupt flow control signals provided to the host to reduce potential

overwriting of data due to host overload (i.e., thrashing) (Dobson), as well as providing an expedient and efficient flow control system, independent of the host, to instruct a remote device to stop sending data (Copeland) to reduce the burden on the host to handle flow control, as well as preventing data overwrite due to buffer overflow.

150. As an example, a PHOSITA would have been motivated to prevent the receive FIFO 100 (first buffer) of Dobson from overflowing by instructing the transmitting remote device to stop sending data by issuing a flow control signal, as noted above after an interrupt is sent to the host of Dobson. This would be useful as even with the additional buffering of Dobson, the host may sometimes be unable to immediately read the data in the receive FIFO 100, which could result in data overflow and subsequent data loss due to overwriting. For instance, where the host of Dobson is already strained, additional interrupts (even at a reduced frequency in view of the increased buffering) may still result in overload (thrashing) of the host, thereby preventing or impeding a timely reading of data in the FIFO 100. In addition, the predetermined threshold level of Copeland used to generate the flow control signal may be set higher than that of Dobson to allow for the host to first try to read the data from the receive buffer before the transmitting device is instructed by stop transmitting using the data flow controller of Copeland. This would allow the system to attempt to avoid any interruption of data while still

providing a failsafe in the event the host is not able to read out data prior to the upper threshold of Copeland being reached.

151. Thus, the combination of additional buffering in Dobson with the flow control signaling of Copeland would further their common goals by preventing the overwriting of received data without placing an undue burden on the respective host device.

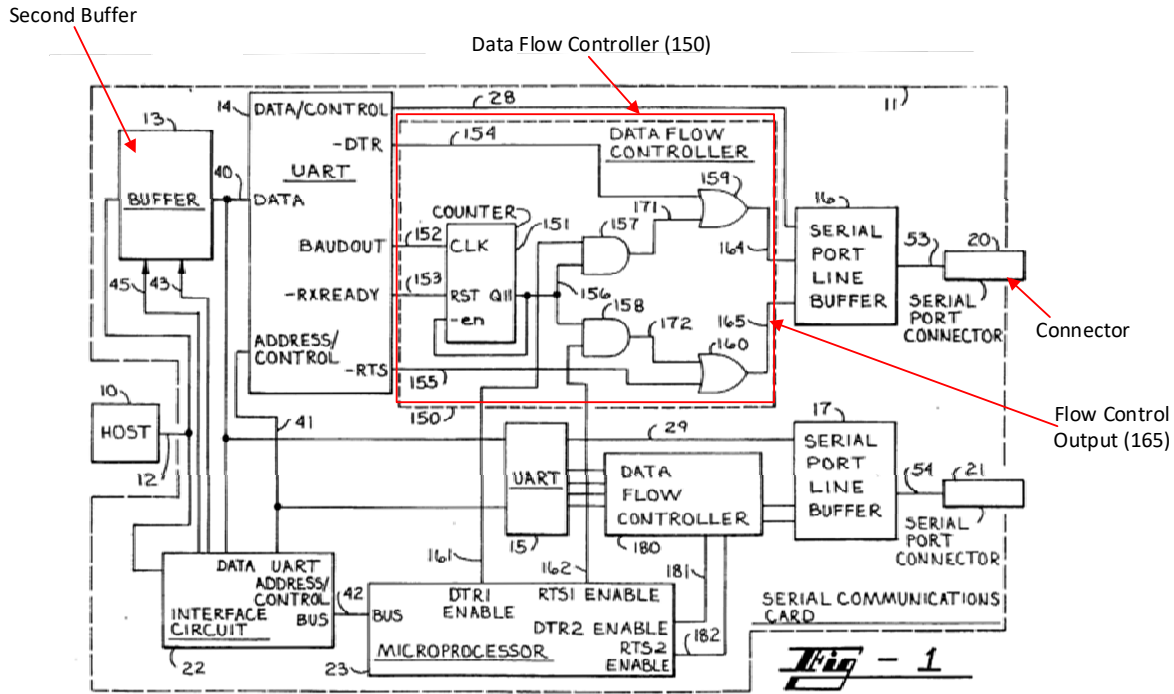
152. Additionally, a PHOSITA would have had a reasonable expectation of success in modifying Dobson's trigger level/comparator with the flow control signal of Copeland to aid in overflow prevention of a receive buffer as well as host workload reduction. Dobson explains that a further object of the disclosed invention was to "provide an improved serial port device with expanded buffering capability that is backwardly compatible with apparatus and application programs written for use with conventional UARTs." *EX. 1007*, at col. 4:21-25. As noted above, both Dobson and Copeland commonly utilize 16550 UARTs, and therefore a PHOSITA would have understood that the system of Dobson would interface well with the system of Copeland, due to the backwards compatibility of Dobson.

1. *wherein the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer.*

153. Copeland teaches that “the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer,” as recited in claim 1. As described in section IX.D.1.k, above, Copeland teaches generating a “flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount.”

154. Copeland further teaches that the flow control signal is transmitted “through the serial link and the connector without passing the flow control signal through the second buffer,” as claimed. As illustrated in FIG. 1 of Copeland (annotated below), the flow control signal is generated outside of the UART 14 and does not pass through the UART 14 or the FIFO buffer (second buffer). As shown in FIG. 1 of Copeland (below), the data flow controller 150 responsible for generating the flow control signal is separate from the UART 14. Furthermore, as noted above the counter 151 of data flow controller 150 is configured to modify the RTS signal and output by the UART 14 in response to the amount of data in the FIFO buffer of UART 14 exceeding a predetermined value. *See EX. 1006*, at col. 5:43-55. Similar modifications are done to the DTR signal outside of UART 14. *See id.* The modified RTS signal and/or DTR signal (collectively “flow control signal”) is output on line 165, which does not pass through the host data buffer 13 (which corresponds to the second buffer, as explained below) or UART 14, but

16, as shown.

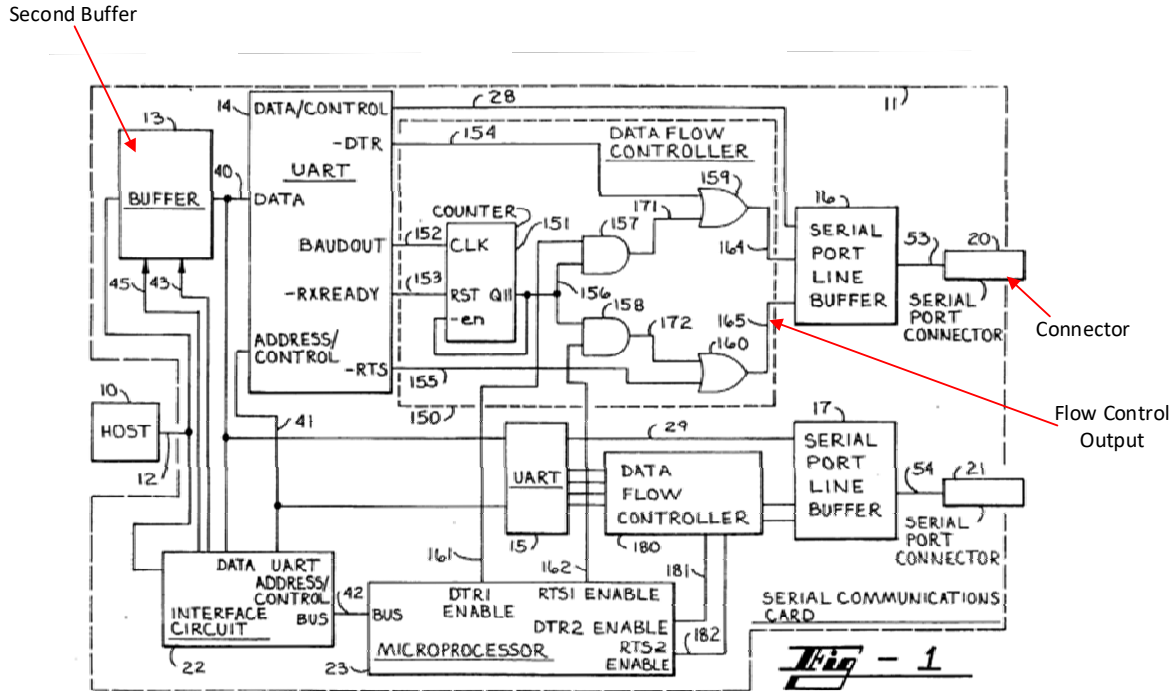


Copeland FIG. 1 (Annotated)

155. The flow control signal, as noted above, does pass through serial port line buffer 16 prior to being output on via the serial data port connector 20. The serial port line buffer 16 is stated to “perform standard line buffering and signal inverting functions.” *EX. 1006*, at col. 4:37-38. The serial port line buffer 16 is not “in data communication with a second circuit that is configured to serialize data received via the second buffer to generate second serial data,” and therefore is not analogous to the second buffer of claim 1. Rather, host buffer 13 is analogous to

the second buffer of claim 1. Specifically, host buffer 13 is described as a “bidirectional buffer which controls the flow of data to and from the host.” *Id.* at col. 4:7-9. Thus, host buffer 13 provides data from the host 10 to the UART 14 for subsequent serialization by the UART 14. For example, Copeland states that “UARTs 14 and 15 convert data between a serial data format on connectors 20 and 21 **and a parallel data format on bus 40.**” *Id.* at col. 4:35-38 (emphasis added). Bus 40 connects buffer 13 to UART 14 to be serialized—the data into and out of host buffer 13 on line 40 is parallel data. A PHOSITA would understand that the host buffer 13 of Copeland is the second buffer of claim 1, as the host buffer “is in data communication with a second circuit [parallel to serial converter of UART 14] that is configured to serialize data received via the second buffer to generate the second serial data.”

156. As described above, and shown in FIG. 1 of Copeland, the flow control signal (i.e., RTS and/or DTR signal) is not passed through the host buffer 13 (second buffer) or UART 14 (first buffer). Accordingly, Copeland teaches the claimed element of “wherein the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer.”



Copeland FIG. 1 (Annotated)

157. A PHOSITA would have found it obvious to combine Copeland's data flow controller 150 with Dobson's serial port controller 10 for multiple reasons. First, and most obvious, both Dobson and Copeland are directed to the same technology. Specifically, both are directed to serial port controllers utilizing UARTs, and specifically, 16550 UARTs, to control a serial interface of a host device, such as host PC 8 of Dobson or host 10 of Copeland. *See EX. 1007*, at Abstract; *EX. 1006*, at col. 2:46-47. To further illustrate the similarity in technologies, not only are both Copeland and Dobson directed to the same technology (serial port controllers), using the same componentry (16550 UARTs),

but the technology described in both Copeland and Dobson was developed by the same applicant – Hayes Microcomputer Products, Inc. *EX. 1006*, at 1; *EX. 1007*, at 1.

158. Furthermore, both Dobson and Copeland are directed to similar, and complementary, goals. Dobson is directed to reducing the number of interrupts (e.g., read commands) to the host PC 8 by providing additional receive buffer capacity, thereby allowing more data to be stored before a read interrupt is required and reducing the strain on the processor to address said interrupts. *EX. 1007*, at col. 3:33-52. Dobson explains that an object of the invention is “to provide a serial port with expanded receive and/or transmit buffering with scalable trigger levels for service requests that can physically allow for greater amounts of data to be buffered . . . [to] internally restrict[] the rate of its own generation of interrupt requests so as to prevent a thrashing condition in which serial port servicing occupies an inordinate amount of CPU processing time.” *EX. 1007*, at col. 4:27-37. This restriction in the rate of interrupt request generation “provides additional data absorption capability that prevents the overwriting of received data while the generation of the new service request is inhibited. *Id.* at col. 4:38-43. Thus, Dobson not only reduces workload of the host with respect to flow control, but also attempts to prevent overwriting of data due to overflow conditions by providing

additional buffering capacity along with limited interrupts that reduce the likelihood the buffer will overflow.

159. In a similar vein, Copeland describes a goal of relieving the host device of having to manage flow control by providing specific circuitry, such as the data flow controller 150 to handle flow control. For example, Copeland states that the described invention “automatically signals the remote transmitter to stop sending [data] when the FIFO buffer fills to a certain point and to resume sending once the FIFO buffer has been completely emptied. This relieves the host of the duty of managing the flow control.” *EX. 1006*, at col. 3:3-8. This flow control scheme of Copeland is further intended to signal “the remote transmitter to halt the flow of [] data in a timely manner such that data loss is prevented.” *Id.* at col. 2:25-29.

160. A PHOSITA would have been motivated to combine these complementary designs as they would work together to form a bulwark against data loss due to buffer overflow by increasing buffer capacity to reduce the amount of interrupt flow control signals provided to the host to reduce potential overwriting of data due to host overload (i.e., thrashing) (Dobson), as well as providing an expedient and efficient flow control system, independent of the host, to instruct a remote device to stop sending data (Copeland) to reduce the burden on

the host to handle flow control, as well as preventing data overwrite due to buffer overflow.

161. As an example, a PHOSITA would have been motivated to prevent the receive FIFO 100 (first buffer) of Dobson from overflowing by instructing the transmitting remote device to stop sending data by issuing a flow control signal, as noted above, after an interrupt is sent to the host of Dobson. This would be useful as even with the additional buffering of Dobson, the host may sometimes be unable to immediately read the data in the receive FIFO 100, which could result in data overflow and subsequent data loss due to overwriting. For instance, where the host of Dobson is already strained, additional interrupts (even at a reduced frequency in view of the increased buffering) may still result in overload (thrashing) of the host, thereby preventing or impeding a timely reading of data in the FIFO 100. In addition, Copeland's predetermined threshold level used to generate the flow control signal may be set higher than that of Dobson to allow for the host to first try to read the data from the receive buffer before the transmitting device is instructed by stop transmitting using Copeland's data flow controller. This would allow the system to attempt to avoid any interruption of data, but provide a failsafe in the event the host is not able to read out data prior to reaching Copeland's upper threshold.

162. Thus, the combination of additional buffering in Dobson with the flow control signaling of Copeland would further the common goals of both by preventing the overwriting of received data without placing an undue burden on the respective host device.

163. Finally, a PHOSITA would have had a reasonable expectation of success in modifying Dobson's serial port controller with the data flow controller of Copeland to aid in overflow prevention of a receive buffer as well as host workload reduction. Dobson explains that a further object of the disclosed invention was to "provide an improved serial port device with expanded buffering capability that is backwardly compatible with apparatus and application program written for use with conventional UARTs." *EX. 1007*, at col. 4:21-25. As noted above, both Dobson and Copeland commonly utilize 16550 UARTs, and therefore a PHOSITA would have understood that the system of Dobson would interface well with the system of Copeland, due to the backwards compatibility of Dobson.

164. In addition, as discussed above, a PHOSITA would have understood that the serial link of Dobson would be connected to various external devices, such as a modem, through a connector such as the connectors 20 and 21 of Copeland and would therefore have been motivated to combine Copeland's connector 20 with Dobson's serial port controller 10, as explained above.

165. Thus, Dobson combined with Copeland teaches all elements of, and render obvious, claim 1.

2. Claim 2

The system of claim 1, wherein the system is configured to transmit the first serial data and the second serial data asynchronously over the serial link.

166. As noted above, Dobson utilizes a Universal Asynchronous Receiver Transmitter for asynchronous serial communication. *See EX. 1007*, at col. 5:32-37. Specifically, Dobson discloses that the “enhanced serial port controller 10 provides asynchronous, serial digital data on line 16. As is familiar to those skilled in the art, the asynchronous data is typically sent to a modem for transmission over the public switch telephone network to other location.” *EX. 1007*, at col. 6:12-16. As noted above, the line 16 is the transmit line as the data serialized by the UART is “output from the UART 30 on line 16.” As the data provided to line 16 is disclosed as being “asynchronous data,” Dobson discloses transmitting the second serial data asynchronously over the serial link.

167. Similarly, Dobson discloses that the first data is received asynchronously. Specifically, Dobson states that the “enhanced serial port controller 10 also accepts asynchronous, serial data on line 14. Asynchronous data received on line 14 is passed to the UART.” *EX. 1007*, at col. 7:53-57. As noted

above, the serial data received on line 14 discloses first serial data received over the serial link element of claim 1.

168. Thus, Dobson combined with Copeland renders obvious claim 2.

3. Claim 3

The system of claim 1, wherein: the data received by the second buffer comprises parallel data.

169. As noted by Dobson, the “data stored in the transmit FIFO buffer 40 is output on data bus 23 to the UART 30. As will be familiar those skilled in the art, the data is converted from parallel to serial in the UART 30 and output on line 16.” *EX. 1007*, at col. 7:1-4. As noted above, the transmit FIFO buffer discloses the second buffer as recited in claim 1. Dobson explains that the data stored in the transmit FIFO buffer is later converted from parallel data to serial data, thereby disclosing that the data stored in the transmit FIFO buffer (i.e., second buffer of claim 1) is parallel data. *EX. 1007*, at col. 7:1-4.

170. Accordingly, Dobson discloses “wherein: the data received by the second buffer comprises parallel data,” as recited in claim 3.

171. Thus, Dobson combined with Copeland renders obvious claim 3.

4. Claim 4

The system of claim 1, wherein: the module is further operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission.

172. As explained in Dobson, the “fundamental building block of a serial port for a computer or terminal is a universal asynchronous receiver-transmitter, commonly known by the acronym UART. This device accepts parallel words or bytes from the data terminal equipment, generates appropriate timing signals, depending on the bit rate, and transmits the bytes in bit serial form, together with appropriate start, stop and parity checking bits. On the receiver side of the UART, the center of the bit times are determined, based on the approximate data rate, and serial data is clocked into a register from a word or byte that is then available for reading out in parallel by the data terminal equipment.” *EX. 1007*, at col. 1:36-47.

173. Dobson also describes the transmission process, which culminates with the data being “input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16 [serial link].” *EX. 1007*, at col. 6:10-12. No acknowledgement or other signal from the receiving device is required to transmit data by the UART 30. Data is simply serialized and transmitted as described above. Further review of the transmission of data by the enhanced serial portion controller 10 of Dobson illustrates that all the various functions of transmitting data do not rely on any acknowledgement (or other) signal being received from any device to transmit the required data. *See, EX. 1007*, at col. 5:52 – 6:16.

174. The '214 patent, upon which the '454 patent relies for support, recites similar operations. For example, the '214 patent states only that during a write transaction (transmission) the encoder 26 “will add at least two additional bits tagging this information as an addressing cycle. The information is then broken into frames that can carry flow control and other signals before being transmitted serially over link 40.” EX 1010, at col. 10:18-25. This disclosure from the '214 patent is nearly identical to the data transmission operations described in Dobson described above in that the data is transmitted and no requirement for an acknowledgment is required to start or maintain transmission of the data.

175. Additionally, as discussed above, a PHOSITA would have understood that the serial link of Dobson would be connected to various external devices, such as a modem, through a connector such as the connectors 20 and 21 of Copeland, as explained above.

176. Thus, Dobson in view of Copeland teaches that a module “operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission,” as claimed. Therefore, Dobson combined with Copeland render obvious claim 6.

5. Claim 7

The system of claim 1, wherein: the flow control signal is configured to be used to control transmission of the first serial data over the serial link to prevent overflow of the first buffer.

177. Copeland explains that the flow control is “activated when the FIFO buffer is forty percent full.” *EX. 1006*, at col. 5:38-40. Thus, the flow control is used to prevent an overflow of the FIFO buffer of UART 14 as the flow control is generated upon the FIFO buffer reaching a threshold. Copeland goes on to state that “[i]t will therefore be appreciated that whenever host 10 is unable to keep up with the data transfer rate, the Q11 output of counter 151 will go high, thereby causing the DTR and/or RTS output signals [flow control signal] on serial port connector 20 to go low and **signal the remote transmitter to stop sending.**” *EX. 1006*, at col. 5:65 – 6:2 (emphasis added).

178. Stopping the remote transmitter from sending in the event the host cannot read data from the FIFO buffer of UART 14 [first buffer] fast enough teaches a flow control signal “configured to be used to control transmission of the first serial data over the serial link to prevent overflow of the first buffer,” as recited in claim 7.

179. A PHOSITA would have found it obvious to combine the data flow controller of Copeland with the serial port controller 10 of Dobson to provide a mechanism for the Receive Flow Control Block 105 to transmit a flow control signal to an external device over serial link 16 to stop sending data in response to an overflow condition on receive FIFO 100. It would have been understood that to prevent the receive FIFO 100 from overflowing, a flow control signal would need

to be sent to the external device transmitting the data to stop transmitting, as described in Copeland, above. As both Dobson and Copeland are directed to serial interfaces utilizing 16550 UARTs to provide a serial interface for a host device, a PHOSITA would have been motivated to combine the receive flow control functionality of Copeland with the flow control functionality of Dobson to prevent the target device from sending data until the host PC could read the data from the receive FIFO buffer, as explained above. *See EX. 1007*, at col. 3:20-52; *EX. 1006*, at col. 2:35-47.

180. Furthermore, the common use of 16550 UARTs described above in both Dobson and Copeland, combined with the desired backwards compatibility of Dobson with devices using UARTs such as the 16550 would have ensured that a PHOSITA would have had a reasonable expectation of success in combining the data flow controller of Copeland with the serial port controller 10 of Dobson as described above.

181. Thus, Dobson combined with Copeland render obvious claim 7.

6. Claim 16

The system of claim 1, further comprising: an interface included in the module coupled to the first buffer and operable to receive the data stored in the first buffer;

a bus coupled to the module and configured to receive the data from the interface; and

a processor coupled to the bus.

183. As shown in annotated FIG. 2, above, the interface 20 is coupled to the first buffer via data line 26d. The interface 20 is coupled to the host PC 8 via “the host personal computer data bus 12 and control bus 13, both of which are familiar to those skilled in the art.” *EX. 1007*, at col. 5:52-58. Thus, the host personal computer data bus 12 teaches the bus of claim 16 as it provides data from the bus interface 20 to the host PC, and therefore is “configured to receive the data from the interface.” As shown in annotated FIG. 2, above, the host PC 20 is coupled to the bus 12. A PHOSITA would understand that a host PC, by virtue of being a Personal **Computer**, would include a processor. Accordingly, Dobson teaches a “processor coupled to the bus.”

184. Accordingly, Dobson combined with Copeland renders obvious claim 16.

E. GROUND 5: Dobson combined with Copeland and Reay teaches all elements of Claim 6.

The system of claim 1, wherein: the module is further operable to transmit the first serial data over the serial link and to transfer the second serial data over the serial link using differential signals.

185. As an initial matter, as discussed above, claim 6 appears to have a typographical error. Claim 6 states that “the module is further operable to transmit the first serial data over the serial link.” However, independent claim 1, from which claim 6 depends, states that the module is operable to “receive first serial data transmitted over the serial link.” Accordingly, based on the language of

independent claim 1, I assume for my analysis that the data is received by the module (and transmitted by an external device), and that it takes place over the serial link using differential signals.

186. As noted above, Dobson teaches receiving “first serial data transmitted over the serial link through the connector and transmit second serial data to the serial link through the connector,” as claimed in claim 1. This is explained in Dobson, which states that “[t]he receive side of the enhanced serial port controller 10 also accepts asynchronous, serial data on line 14.” *EX. 1007*, at col. 7, lines 53-56. This serial data received on line 14 would be understood by a PHOSITA to be the first serial data received over the serial link. As further explained in Dobson, data is “input to the UART 30 in parallel, [and] is converted by the UART 30 to serial and is output from the UART 30 on line 16.” *EX. 1007*, at col. 6:10-12. This serialized data that is output from the UART as serial data on line 16 discloses “transmit second serial data to the serial link through the connector.”

187. Reay teaches transmitting serial data “using differential signals,” as recited in claim 6. Specifically, Reay explains that “[o]utput terminals 106 and 108 provide inverted and non-inverted output signals, respectively, of the input signal received at terminal 102. Thus, a differential, or balanced-line, output can be provided for those interface standards that require such output.” *EX. 1008*, at col.

4:15-20. Reay goes on to explain that the “advantage to such outputs is noise immunity; any noise that appears on one of the two output lines is likely to appear on the other as well, and is thus cancelled as the difference between the two output signal voltages is taken.” *EX. 1008*, at col. 4:32-35. Accordingly, a PHOSITA would have found it obvious and been motivated to combine the noise cancelling differential signals of Reay for use on the serial link lines 14 and 16 to reduce errors due to noise with respect to the received and/or transmitted signals.

188. Furthermore, a PHOSITA would have had a reasonable expectation of success in incorporating the differential signaling of Reay into the serial link of Dobson and Copeland, as the use of differential signaling for serial communication was well known in the art at the time of Dobson, Copeland, and Reay.

189. Accordingly, Dobson combined with Copeland and Reay teach “the module is further operable to transmit the first serial data over the serial link and to transfer the second serial data over the serial link using differential signals.”

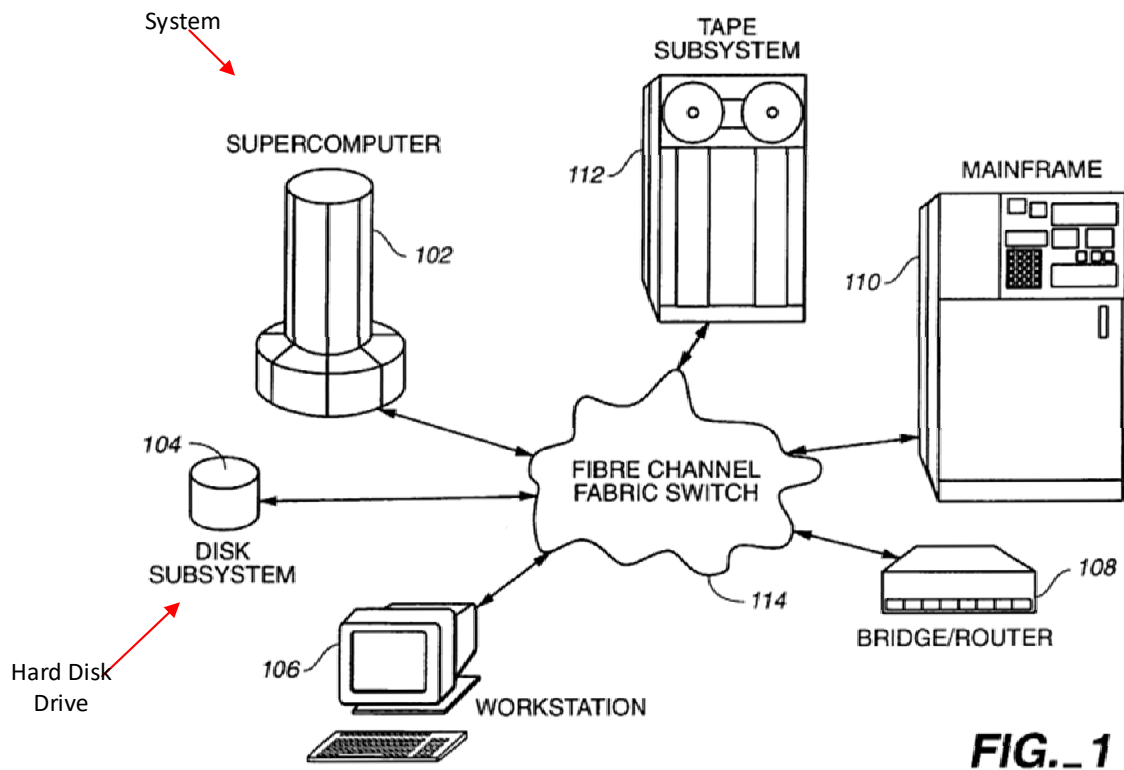
190. Therefore, the combination of Dobson, Copeland and Reay renders obvious claim 6.

F. GROUND 6: Dobson, Copeland, and Vangemert teaches all elements of Claims 15 and 17.

1. Claim 15

The system of claim 1, wherein: the system comprises a hard disk drive.

191. Dobson discloses a host PC 8, which a PHOSITA would have understodd to include at least one hard disk drive. To the extent Dobson is deficient, Vangemert teaches a hard disk drive within the system. As shown in Vangemert's FIG. 1, the illustrated system is shown in including a disk subsystem 104. A disk subsystem would be understood by a PHOSITA to disclose a hard disk drive (or multiple hard disk drives) within the system of FIG. 1. Accordingly, Vangemert discloses "wherein the system comprises a hard disk drive," as claimed.



Vangemert FIG. 1 (Annotated)

192. A PHOSITA would have understood to combine the hard disk drive of Vangemert with the system of Dobson to allow for the high-speed serial port of

Dobson to communicate with the data storage devices of Vangemert. A PHOSITA would be motivated to use the high-speed serial port controller 8 of Dobson with other devices to facilitate high speed data transfer, such as the disk subsystem of Vangemert. In fact, as shown in FIG. 1 of Vangemert, above, Vangemert contemplates connections to devices such as workstations, mainframes, supercomputers, etc., which would be similar to the host PC of Dobson. Thus, a PHOSITA would have understood the advantage of using a high-speed serial port connection with more than just a host PC, such as the various subsystems shown in FIG. 1 of Vangemert. Furthermore, Dobson is configured to couple to at least one external device, such as a modem. *EX. 1007*, at col. 6:10-16. Accordingly, a PHOSITA would have had a reasonable expectation of success in combining the teachings of Dobson with those of Vangemert to allow for connection of the serial port device 10 to the disk subsystem 104.

193. Accordingly, the combination of Dobson, Copeland, and Vangemert renders obvious claim 15.

2. Claim 17

The system of claim 1, wherein: the module is further configured to generate a clock signal, for use by the module, using the first serial data.

194. Dobson and Copeland both disclose asynchronous communication systems and, therefore, a PHOSITA would have understood them to generate clock

signals using incoming data. This concept would have been generally known by a PHOSITA and was not even disclosed in the '454 patent. As disclosed in Vangemert, "[t]ransceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signal levels (first serial data), clean them up as much as possible, and provide them to a clock recovery module 304. Clock recovery module 304 uses the incoming signal to reconstruct a clock signal synchronized to the incoming data, and a serial-to-parallel converter 306 uses the clock signal to convert the incoming signal from serial form to parallel form." EX. 1005, at col. 3:19-28. The clock recovery module 304 therefore generates a clock signal using the first serial data (i.e., received data), which is then used by the transceiver 300 to convert the received data from serial to parallel form. Accordingly, Vangemert discloses "the module is further configured to generate a clock signal, for use by the module, using the first serial data."

195. A PHOSITA would have understood to combine the clock recovery functionality of Vangemert with the high-speed serial port of Dobson to allow for data to be synced when clocking in to the receive FIFO buffer 100. As noted in Dobson, the received "data is clocked into the receive FIFO buffer 100 using the write signal on line 34a." EX. 1007, at col. 8:24-25. Extracting the clock signal from the incoming serial data is an efficient and cost effective method of creating the clock signal used in Dobson. Thus, it would be advantageous to generate the

clocking signal based on the received data to ensure proper data interpretation. Additionally, as Dobson, Copeland, and Vangemert all describe utilizing asynchronous communication, which often embeds a clock signal or allows for one to be reconstructed based on the data, a PHOSITA would have had a reasonable expectation of success in combining the asynchronous clock reconstruction of Vangemert with the teachings of Dobson and Copeland.

196. Accordingly, the combination of Dobson, Copeland, and Vangemert renders obvious claim 17.

X. CONCLUSION

197. As explained above, it is my conclusion that the prior art combinations discussed above disclose all elements of claims 1, 2-4, 6-7, and 15-17 of the '454 patent, and PHOSITA would have been motivated to and had reason to make such combinations with a reasonable expectation of success.

I declare under penalty of perjury that the foregoing is true and correct to the best of my knowledge.

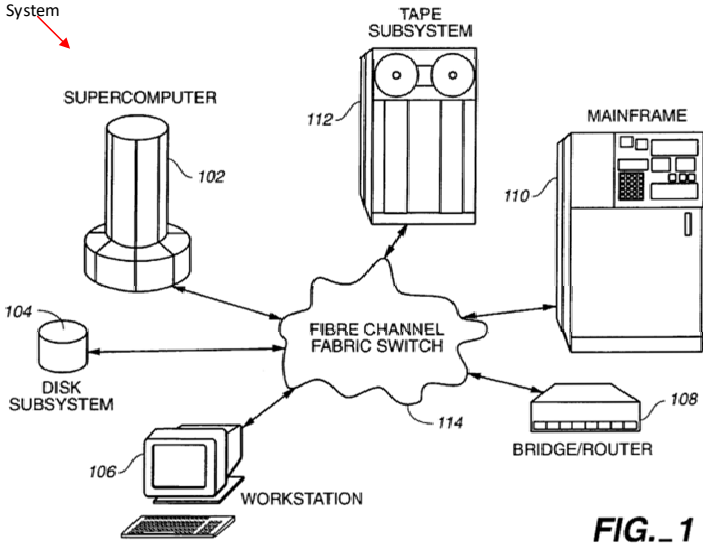
Executed on: April 12, 2023



Michael Caloyannides, Ph.D.

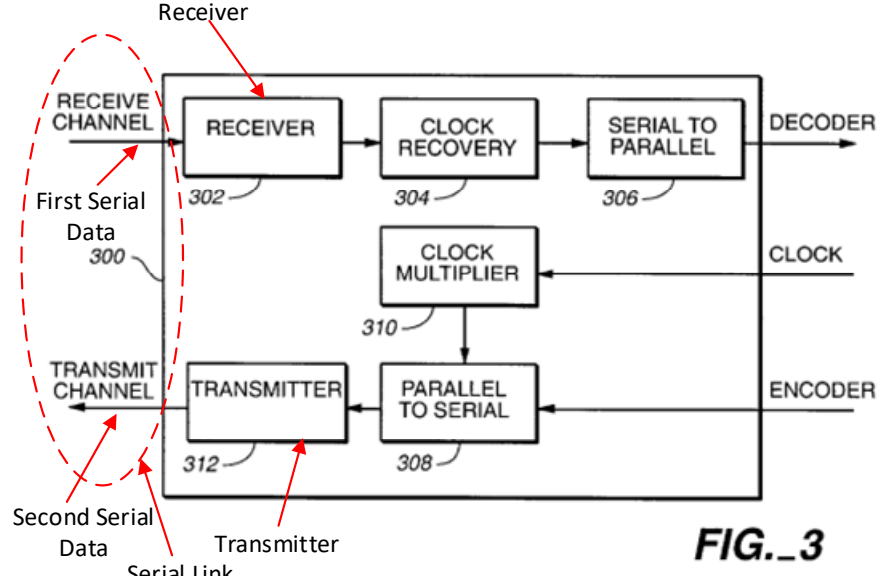
Claim Charts Regarding U.S. Patent No. 9,535,454, Claims 1-4, 6-7, and 15-17
in view of Prior Art Vangemert, Copeland, Dobson, and Reay,
Grounds 1-6

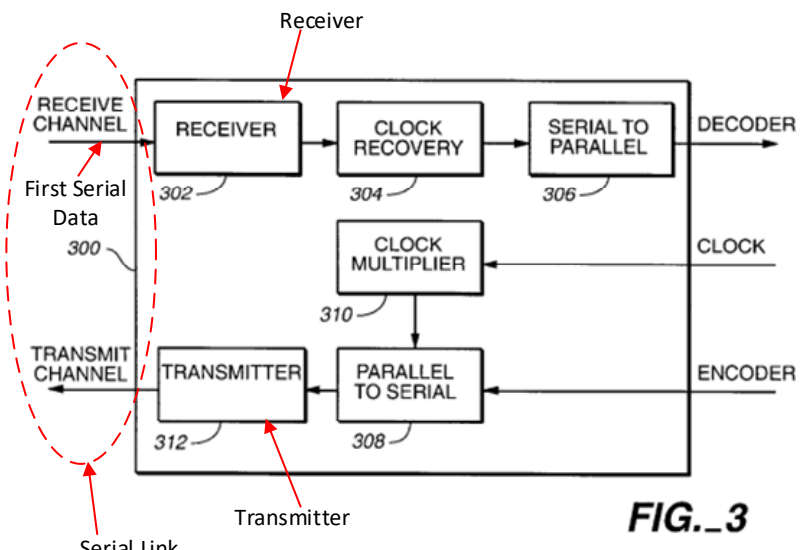
A. Ground 1: Claims 1, 2, 3, 4, 7, 15, 16, and 17 in view of Vangemert and Copeland

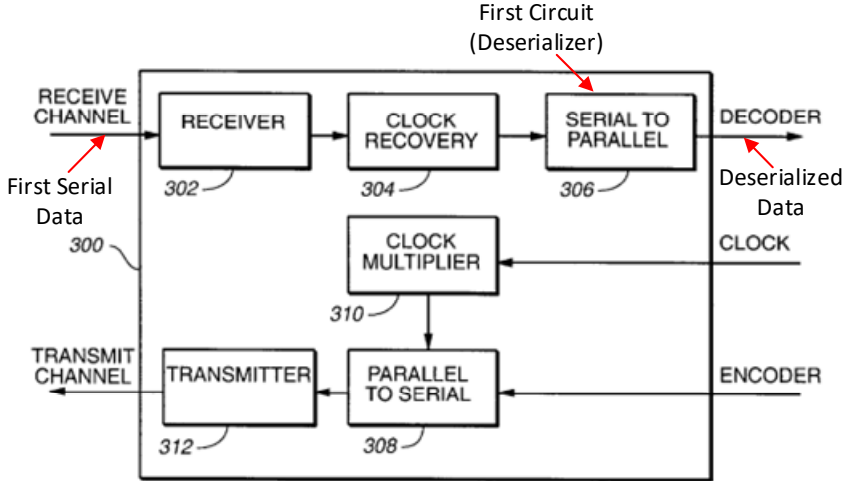
US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
[a] 1. A system, comprising:	<u>VANGEMERT</u> Vangemert discloses a system, such as the system network of FIG. 1 (below). Col. 2:58-61.  FIG. 1 <i>Vangemert FIG. 1 (Annotated)</i>
[b] a connector configured to be connectable to a serial link; and	<u>VANGEMERT</u> “Each of the nodes is coupled to a central ‘fabric switch’ 114 by a serial link having a receive channel and a transmit channel.” Col. 2:61-63. “The nodes are coupled in an ‘arbitrated loop’ topology, in which the transmit channel from one node is directly connected to the receive channel of the next node.” Col. 3:5-7. “Serial Link” = Serial Link “Connector” = Vangemert system would have required a connector <u>COPELAND</u>

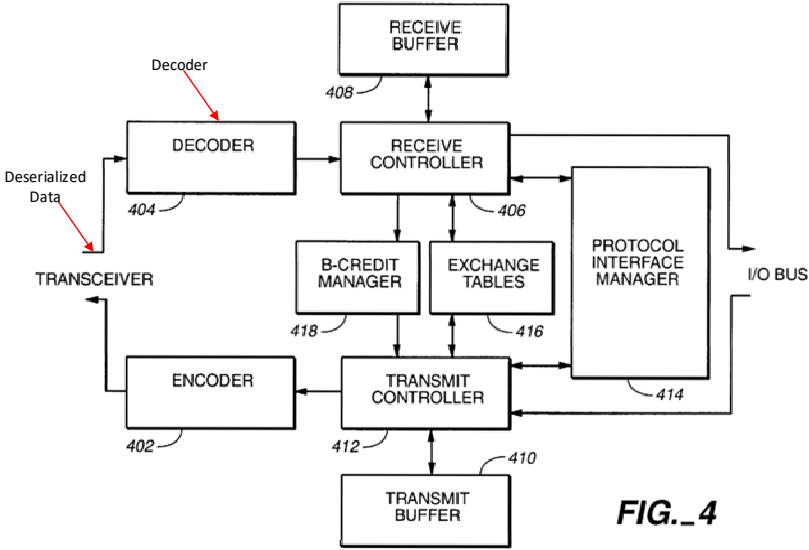
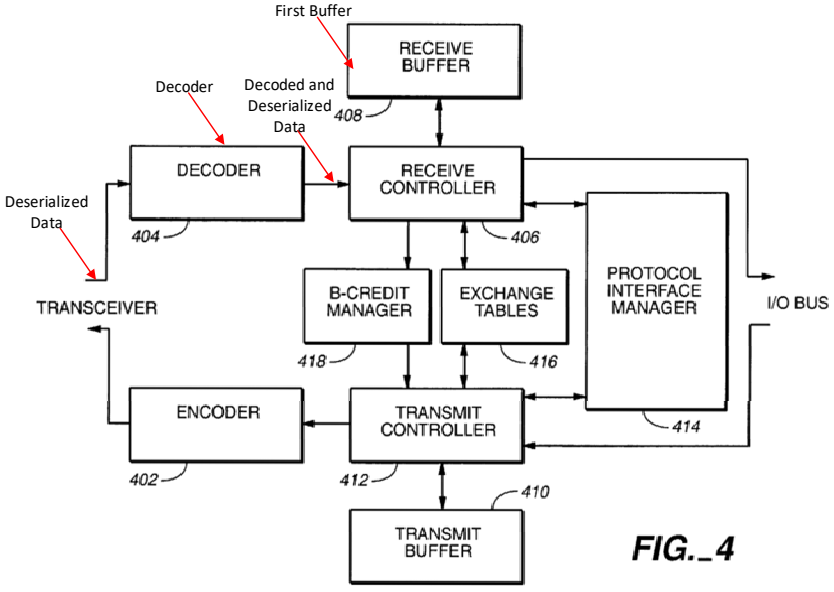
US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	"Host computer 10 is connected to a dual channel asynchronous serial communications card 11 [c]ard 11 contains . . . two serial port connectors 20 and 21." Col 3:49-60. "[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface." Col. 4:31-34. A PHOSITA would have understood that serial port connector 20 is associated with a first channel of the dual channel serial communications card 11, and serial port connector 21 is associated with a second channel of the dual channel serial communications card 11 and may therefore be used interchangeably. "Connector" = Serial Port Connector 20 or 21 Copeland FIG. 1 (Annotated) <u>RATIONALE TO COMBINE</u> To the extent a connector is not disclosed in Vangemert, a PHOSITA would have looked to Copeland, which is directed to the same subject matter as Vangemert – serial interfaces for providing serial communication between devices. See, EX. 1005, at col. 3:19-21; EX. 1006, at col. 2:35-47. A PHOSITA would have found it obvious and been motivated to combine the connector 20 or 21 of Copeland with the serial link of Vangemert, recognizing the impracticality of hardwiring the disclosed link to the devices and the advantages of a connector, as a mechanism for connecting the serial link of the individual nodes 1-2, 104, 106, 108, 110,

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	and 112 to one or more external serial devices, such as other nodes within the system.
[c] a module coupled to the connector, wherein the module is operable to receive first serial data transmitted over the serial link through the connector and transmit second serial data to the serial link through the connector, and wherein the module includes:	<u>VANGEMERT</u> “Each of the nodes is coupled to a central ‘fabric switch’ 114 by a serial link having a receive channel and a transmit channel.” Col. 2:61-63. “The nodes are coupled in an ‘arbitrated loop’ topology, in which the transmit channel from one mode is directly connected to the receive channel of the next node.” Col. 3:5-7. “FIG. 3 illustrates a serial communication transceiver 300 which may be used by the nodes to couple to the serial communications link. Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signal levels.” Col. 3:19-23. “Transmitter 312 then converts the serial bitstream into signals suitable for transport by the transmit channel.” Col. 3:35-36. “Using control words, an initiating node first arbitrates to get control of the loop, then notifies a responding node of its desire to transmit data frames.” Col. 3:10-13. “Module” = The network interface of any of nodes 102-112 “First serial data” = data received on receive channel “second serial data” = data transmitted on transmit channel The diagram illustrates the internal components of a serial communication transceiver and its connection to a network interface device. FIG. 3 (left) shows the transceiver 300, which includes a receiver 302, clock recovery 304, serial-to-parallel 306, clock multiplier 310, parallel-to-serial 308, and transmitter 312. FIG. 4 (right) shows the network interface device, which includes a receive buffer 408, receive controller 405, de-coder 404, encoder 402, transmit controller 412, and transmit buffer 410. It also includes a de-coder 415, exchange tables 416, and a protocol interface manager 414 connected to an I/O bus 414. The entire system is annotated as a 'Module' connected to a 'Serial Link' and a 'Network Interface Device'. <i>Vangemert Combined FIGs. 3 and 4 (Annotated)</i>

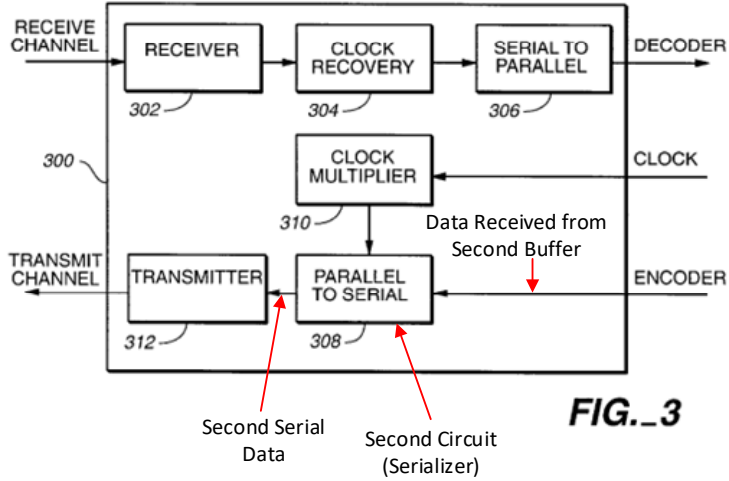
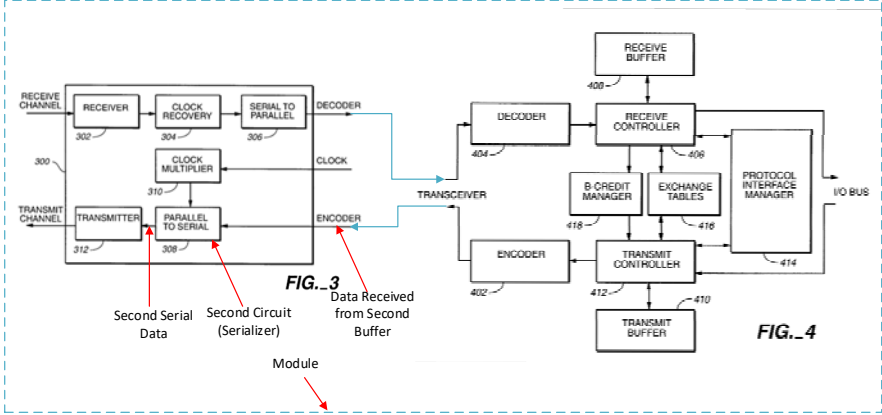
US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	 FIG. 3 Vangemert FIG. 3 (Annotated) COPELAND “[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” Col. 4:31-34. RATIONALE TO COMBINE A PHOSITA would have understood that the various connections between the nodes would require connectors to facilitate the connections, as described above, and therefore would have been motivated to combine the connectors 20 and 21 of Copeland with the module of Vangemert to provide connection of the nodes to the serial link.
[d] a receiver operable to receive the first serial data from the serial link through the connector;	VANGEMERT “FIG. 3 illustrates a serial communication transceiver 300 which may be used by the nodes to couple to the serial communications link. Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signal levels.” Col. 3:19-22. The transceiver 300 also includes a transmitter 312 that “converts the serial bitstream into signals suitable for transport by the transmit channel.” Col. 3:35-36. “Receiver operable to receive first serial data” = Receiver 302

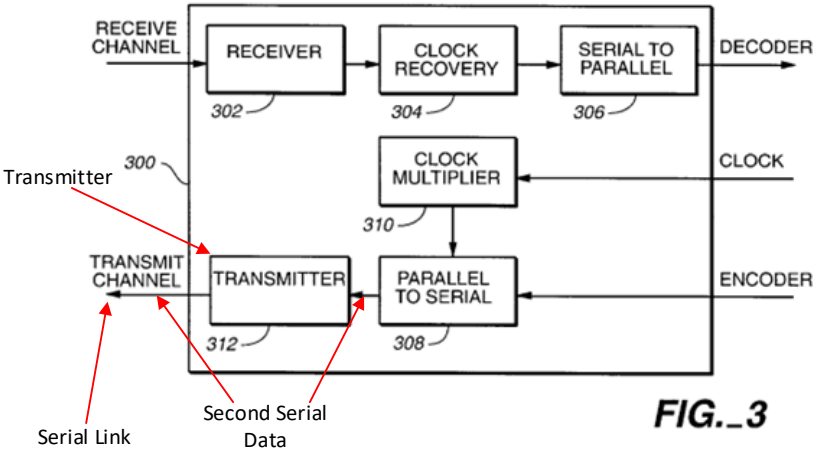
US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	“First Serial Data” = Incoming Signal “Serial Link” = Serial Link (receive channel and transmit channel)  FIG._3 Vangemert FIG. 3 (Annotated) <u>COPELAND</u> “[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” Col. 4:31-34. <u>RATIONALE TO COMBINE</u> A PHOSITA would have understood that the various connections between the nodes would require connectors to facilitate the connections, as described above, and therefore would have been motivated to combine the connectors 20 and 21 of Copeland with the module of Vangemert to provide connection of the nodes to the serial link.
[e] a first circuit configured to deserialize the first serial data to generate deserialized data;	<u>VANGEMERT</u> “[A] serial-to-parallel converter 306 uses the clock signal to convert the incoming signal from serial form to parallel form.” Col. 3:26-28. “First Circuit” = Serial-to-parallel converter 306 “Deserialized Data” = Data that has been converted from serial to parallel form

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	 FIG. 3 Vangemert FIG. 3 (Annotated)
[f] a decoder configured to decode the deserialized data;	<u>VANGEMERT</u> “[Serial-to-parallel] converter 306 outputs one 10-bit channel codeword at a time. The output from the converter 306 is sent to a decoder.” Col. 3:29-30. “Encoder 402 and decoder 404 provide error correction coding and decoding for information transmitted across the serial link.” Col. 3:43-45. “Decoder” = Decoder 404

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	 FIG._4 Vangemert FIG. 4 (Annotated)
[g] a first buffer operable to store data that has been deserialized and decoded by the first circuit and the decoder, respectively;	<u>VANGEMERT</u> “Decoded data from decoder 404 is provided to receive controller 406. Receive controller stores received control words and data frames in receive buffer 408.” Col. 3:45-48. “First Buffer” = Receive buffer 408  FIG._4 Vangemert FIG. 4 (Annotated)

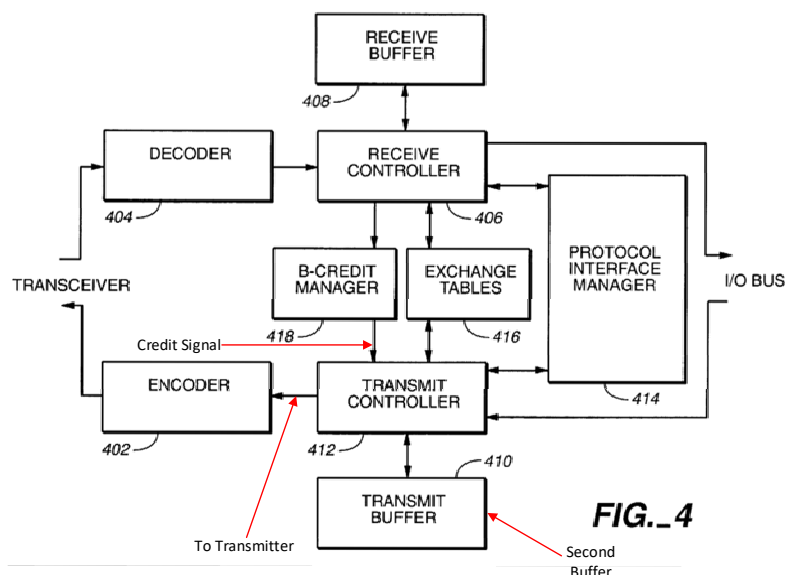
US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
[h] a second buffer operable to receive data, wherein the second buffer is in communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data; and	<u>VANGEMERT</u> “A transmit controller 412 retrieves control words and data from transmit buffer 410 and forwards them to encoder 402.” Col.3:49-52. “Transmit controller [412] creates or updates exchange information in exchange tables 416 in response to commands and data from the node's i/o bus, generates control words and [generates] data frames in transmit buffer 410, and forwards them to encoder 402.” Col. 3:57-61. “Transceiver 300 also includes a parallel-to-serial converter 308 which receives a stream of codewords from an encoder and uses a clock signal from clock multiplier 310 to convert the stream of codewords into a serial bitstream.” Col. 3:31-34. “Second Buffer” = Transmit buffer 410 “Second Circuit” = Parallel-to-Serial Converter 308 “Second Serial Data” = Output of Parallel-to-Serial Converter 308 FIG. 4 Vangemert FIG. 4 (Annotated)

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	 Vangemert FIG. 3 (Annotated)  Vangemert Combined FIGS. 3 and 4 (Annotated)
[i] a transmitter operable to transmit the second serial data to the serial link through the connector;	<u>VANGEMERT</u> “Transmitter 312 then converts the serial bitstream into signal suitable for transport by the transmit channel.” Col. 3:35-36. “Transmitter” = Transmitter 312

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	 FIG. 3 Vangemert FIG. 3 (Annotated) <u>COPELAND</u> "[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface." Col. 4:31-34. <u>RATIONALE TO COMBINE</u> A PHOSITA would have understood that the various connections would require connectors to facilitate the connections, as described above, and therefore would have been motivated to combine the connectors 20 and 21 of Copeland with the module of Vangemert to provide connection of the nodes to the serial link.
[j] wherein the module is configured to determine whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage	<u>VANGEMERT</u> "Receive controller 406 includes a counter or register which tracks the amount of available buffer space [of] the receive buffer 408." Col. 4:32-34. "[A] buffer credit manager . . . includes a comparator 502 which asserts a credit signal when it determines that the additional committed buffer space (i.e., the buffer space committed but not yet occupied) is less than the available buffer capacity by at least the size of one data frame." Col. 4:24-29.

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
capacity of the first buffer; and	“Determine whether an amount of data ...equals or exceeds a fill amount” = Comparator 502 “First Buffer” = Receive Buffer 408
[k] wherein the module is operable to generate a flow control signal in response to the amount of data stored in the first buffer equaling or exceeding the fill amount; and	<u>VANGEMERT</u> “FIG. 5 shows one implementation of a buffer credit manager. It includes a comparator 502 which asserts a credit signal when it determines that the additional committed buffer space (i.e., the buffer space committed but not yet occupied) is less than the available buffer capacity by at least the size of one data frame.” Col. 4:24-29. “[B]efore an initiating node can transmit a data frame to a responding node, it must first secure a buffer credit from that node.” Col. 4:9-11. “Flow Control Signal” = Credit Signal <u>COPELAND</u> “[D]ata flow control is activated when the FIFO buffer is forty percent full.” Col. 5:38-40. “[A]utomatic data flow control occurs as follows. UART 14 contains a 16 byte First In First Out (FIFO) buffer. UART 14 has a signal, -RXREADY, which is low whenever there is at least one byte of data, transmitted from serial data port 20 to the UART 14 FIFO buffer, which has not been read by host 10. UART 14 has another signal, BAUDOUT, which is a clock at 16 times the baud rate. When -RXREADY goes low counter 151 counts the BAUDOUT cycles. When 1024 BAUDOUT cycles pass without the FIFO buffer being empty the Q11 output of the counter 151 goes high. The count of the 1024 BAUDOUT cycles represents 64 bit times or, at 10 bits per character, 6.4 character times. This leaves at least 9.6 character times remaining for the remote transmitter connected to serial data port 20 to respond before the FIFO buffer overflow.” Col. 5:22-38. “When a count of 1024 is reached the Q11 output provides an enabling high signal to gates 157 and 158. If microprocessor 23 is providing a second enabling high signal to gates 157 and 158 over Data Terminal Ready 1 (DTR1) line 161 and Request to Send 1 (RTS1) line 162, the output of gates 157 and 158 provide high signals, respectively, to gates 159 and 160, respectively, which disables gates 159 and 160, thereby causing their outputs to be high, which sends a low for the RTS and DTR signals on serial port connector 20. This signals the remote transmitter to stop sending data, thus shutting down the flow of data from the serial data port 20 to the UART 14 FIFO buffer.” Col. 5:44-55. “[W]henver host 10 is unable to keep up with the data transfer rate, the Q11 output of counter 151 will go high, thereby causing the DTR and/or

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	RTS output signals on serial port connector 20 to go low and signal the remote transmitter to stop sending." Col. 5:65-6:2. Flow Control Signal = RTS / DTR signals <u>RATIONALE TO COMBINE</u> A PHOSITA would have found it obvious to combine the flow control signal of Copeland with the bit credit manager 418 of Vangemert as both Copeland and Vangemert teach interchangeable ways to prevent an overflow of the receive buffer 408.
[I] wherein the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer.	<u>VANGEMERT</u> "[W]hen receive controller 406 receives a control word indicating that a remote node wishes to transmit data, the buffer credit manager 418 is enabled, and the transmit controller 412 begins transmitting buffer credit control words in response to a credit signal from buffer credit manager 418." Col. 4:11-16. "Transmit controller 416 . . . forwards [control words] to encoder 402." Col. 3:57-61. FIG. 4, below shows the buffer credit manager 418 coupled to the transmit controller 412, and not directly connected to the transmit buffer. See FIG. 4. "The transmit controller 412 begins transmitting buffer credit control words in response to a credit signal from buffer credit manager." Col. 4:14-16. A PHOSITA would have understood to transmit the control signal, bypassing the transmit buffer, as the command did not need to be packaged with other data, but needed to be transmitted quickly in response to the request by the remote device to transmit data. See Annotated FIG. 4, below. "Flow Control Signal" = Credit Signal "Second Buffer" = Transmit Buffer 410



Vangemert FIG. 4 (Annotated)

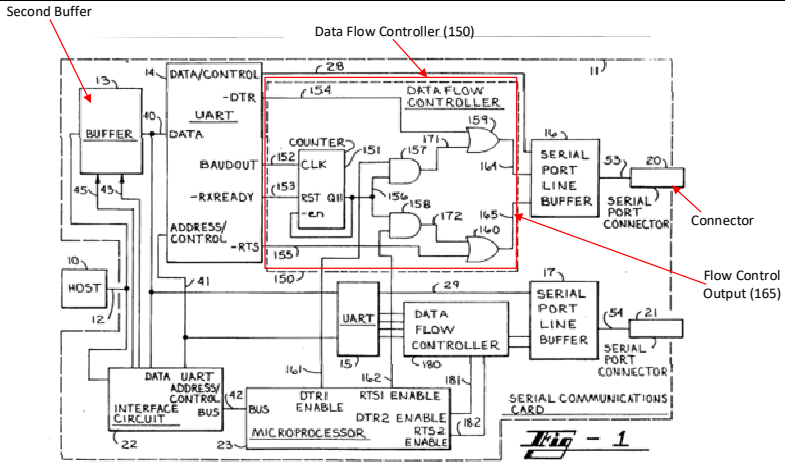
COPELAND

FIG. 1 of Copeland shows the data flow controller 150 is a separate circuit from the UART 14 and is configured to perform automatic flow control outside of control of the host 10. Col. 5:65-6:11; 2:30-32.

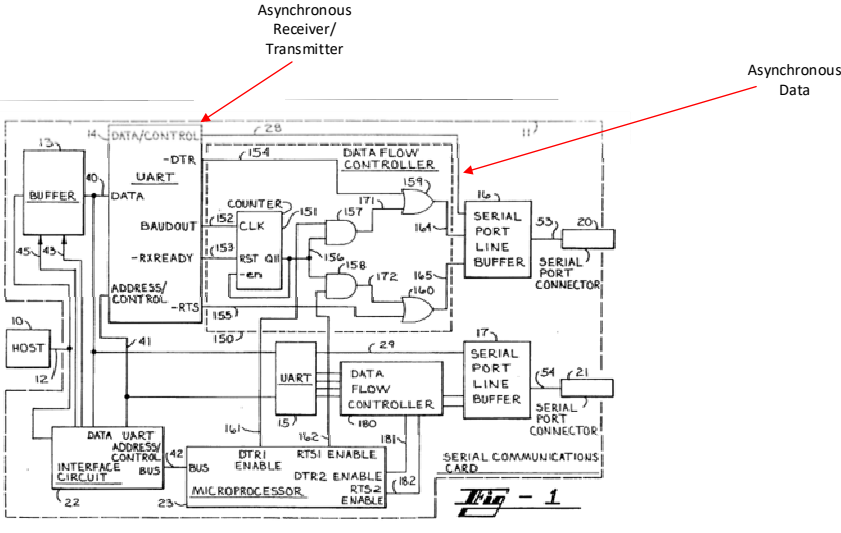
Upon a counter reaching a count indicating the receive buffer is approximately 40% full, "a low [logic level] for the RTS and DTR signals on serial port connector 20" is sent. Col. 5:24-52.

The serial port line buffer 16 is not "in data communication with a second circuit that is configured to serialize data received via the second buffer to generate the second serial data" and is therefore not analogous to the second buffer of claim 1.

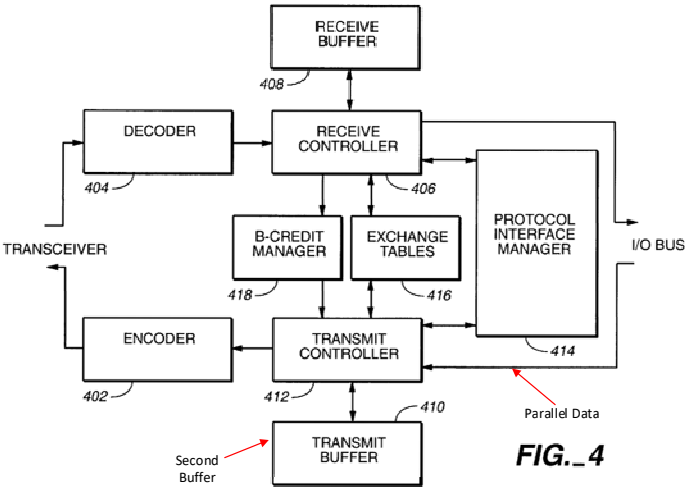
"[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface." Col. 4:31-34.

US 9,535,454, Claim 1	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	 Copeland FIG. 1 (Annotated) Flow Control Signal = RTS output signal. <u>RATIONALE TO COMBINE</u> A PHOSITA would have been motivated to integrate the teachings of Copeland with those of Vangemert to improve the overall operation of Vangemert by minimizing latencies associate with flow control signals by routing any flow control signals directly to the transmitting device without passing through a transmit buffer. A PHOSITA would have also understood that the various connections would require connectors to facilitate the connections, as described above, and therefore would have been motivated to combine the connectors 20 and 21 of Copeland with the module of Vangemert to provide connection of the nodes to the serial link.

US 9,535,454, Claim 2	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
2. The system of claim 1, wherein the system is configured to transmit the first serial data and the second serial data asynchronously over the serial link.	<u>VANGEMERT</u> “Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signals levels, clean them up as much as possible, and provide them to a clock recover module 304. Clock recovery module 304 uses the incoming signal to reconstruct a clock signal synchronized to the incoming data, and a serial-to-parallel converter 306 uses the clock signal to concert the incoming signal from serial form to parallel form.” Col. 3:21-28.

US 9,535,454, Claim 2	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	“Transceiver 300 also includes a parallel-serial converter 308 which receives a stream of codewords from an encoder and uses a clock signal from clock multiplier 310 to convert the stream of codewords into a serial bitstream. Transmitter 312 then converts the serial bitstream into signals suitable for transport by the transport channel.” Col. 3:31-36. Vangemert explains the preferred protocol used with the described devices and methods is Fibre Channel communication protocol. A PHOSITA would have understood that Fiber Channel is an asynchronous communication protocol as it: (a) does not require a common, synchronized clock signal as described above; and (b) requires that all data frames include start and stop delimiters. <u>COPELAND</u> “Host computer 10 is connected to a dual channel asynchronous serial communications card 11.” Col. 3:49-52. “Card 11 contains . . . two parallel-serial converters such as universal asynchronous receiver transmitters (UARTs) 14 and 15.” Col. 3:53-60.  <i>Copeland FIG. 1 (Annotated)</i> <u>RATIONALE TO COMBINE</u> A PHOSITA would have found it obvious and been motivated to combine the expressly asynchronous transmitter/receiver of Copeland with the transceiver 300 of Vangemert to configure the system to “transmit the first serial data and the second serial data

US 9,535,454, Claim 2	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	asynchronously over the serial link.” As discussed above, both Vangemert and Copeland are directed to serial communication devices that are not synchronized to a common, separate clock signal. <i>EX 1005</i>, Col. 3:19-4:7; <i>EX. 1006</i> at Abstract. Furthermore, asynchronous communication is known in the art to be used in applications where distances between devices exceeds short distances (e.g., several feet), such as would be the case with respect to the various nodes 102-112 of Vangemert. In addition, a PHOSITA would understand that implementing a common clock or other required components within multiple devices, such as nodes 102-112 of Vangemert, would require substantial effort to accomplish.

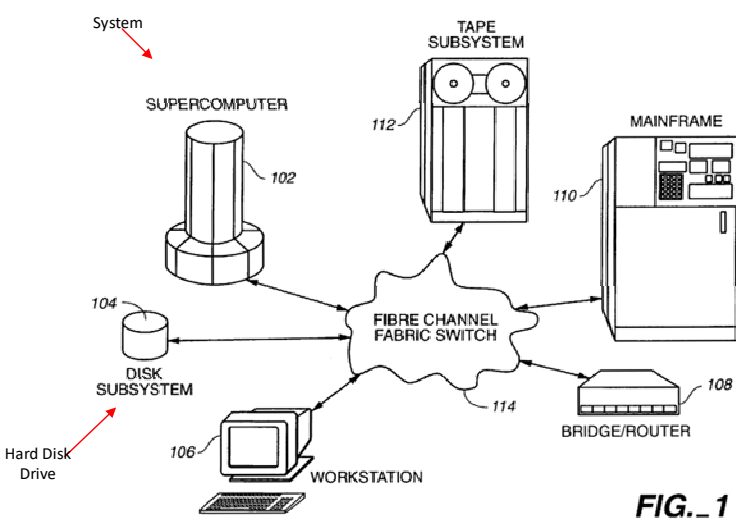
US 9,535,454, Claim 3	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
3. The system of claim 1, wherein: the data received by the second buffer comprises parallel data.	<u>VANGEMERT</u> “Transceiver 300 also includes parallel-to-serial converter 308 which receives a stream of codewords from an encoder and uses a clock signal from clock multiplier 310 to convert the stream of codewords into a serial bitstream.” Col. 3:31-34.  FIG. 4 Vangemert FIG. 4 (Annotated)

US 9,535,454, Claim 4	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
4. The system of claim 1, wherein: the module	<u>VANGEMERT</u>

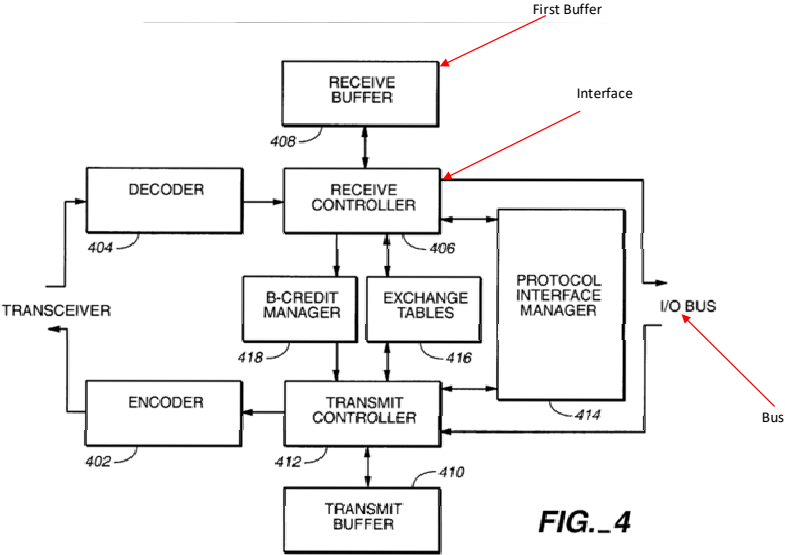
US 9,535,454, Claim 4	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
is further operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission.	“The remote node is configured to transmit a data frame to the local node only if the remote node receives a buffer credit which indicates that the local node has available receive buffer space for a data frame.” Col. 2:3-6. “Transmit controller 416 creates or updates exchange information in exchange tables 416 in response to commands and data from the node’s i/o bus, generates control words and data frames in transmit buffer 410, and forward them to encoder 402.” Col. 3:57-61. A PHOSITA would have understood that communication of an entire data packet without requiring intervening acknowledgements is fundamental to serial communication via Fiber Channel as described in Vangemert, which transmits all data in a data frame once the bit credit is issued.

US 9,535,454, Claim 7	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
7. The system of claim 1, wherein: the flow control signal is configured to be used to control transmission of the first serial data over the serial link to prevent overflow of the first buffer.	<u>VANGEMERT</u> “[B]efore a first network node (initiating node) can transmit a data frame to a second network node (responding node), the initiating node must first receive a buffer-credit from the responding node, indicating that the responding node has available space in its receive buffer for a data frame.” Col. 1:49-54. “FIG. 5 shows one implementation of a buffer credit manager. It includes a comparator 502 which asserts a credit signal when it determines that the additional committed buffer space (i.e., the buffer space committed but not yet occupied) is less than the available buffer capacity by at least the size of one data frame.” Col.4:24-29. “Flow Control Signal” = Credit Signal <u>COPELAND</u> “[D]ata flow control is activated when the FIFO buffer is forty percent full. Col. 5:39-40. “It will therefore be appreciated that whenever host 10 is unable to keep up with the data transfer rate, the Q11 output of counter 151 will go high, thereby causing the DTR and/or RTS output signals on serial port connector 20 to go low and signal the remote transmitter to stop sending. Col. 5:44-6:2.

US 9,535,454, Claim 7	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	“Flow Control Signal” = RTS / DTR Signals <u>RATIONALE TO COMBINE</u> As discussed above, a PHOSITA would have been motivated to combine the flow control teachings of Copeland with those of Vangemert, and would have had a reasonable expectation of success in doing so.

US 9,535,454, Claim 15	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
15. The system of claim 1, wherein the system comprises a hard disk drive.	<u>VANGEMERT</u> Vangemert discloses in FIG. 1 a disk subsystem 104. A disk subsystem discloses a hard disk drive (or multiple hard disk drives). “Hard Disk Drive” = Disk Subsystem 104  FIG. 1 Vangemert FIG. 4 (Annotated)

US 9,535,454, Claim 16	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
16. The system of claim 1, further comprising: an interface included in the module coupled to	<u>VANGEMERT</u> “Receive controller stores received control words and data frames in receive buffer 408.” Col. 3:46-48.

US 9,535,454, Claim 16	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
the first buffer and operable to receive the data stored in the first buffer; a bus coupled to the module and configured to receive the data from the interface; and a processor coupled to the bus.	“[R]eceive controller 406 creates or updates exchange information in exchange tables 416 in response to received control words or data frame, and forwards data from received data frame to the node’s i/o bus.” Col. 3:53-57. The nodes are described as the nodes of FIG. 1, which may include a “supercomputer 102.” Interface = Receive Controller 406 First Buffer = Receive Buffer 408 Bus = I/O Bus (see FIG. 4) Processor = Supercomputer 102  FIG. 4 Vangemert FIG. 4 (Annotated)

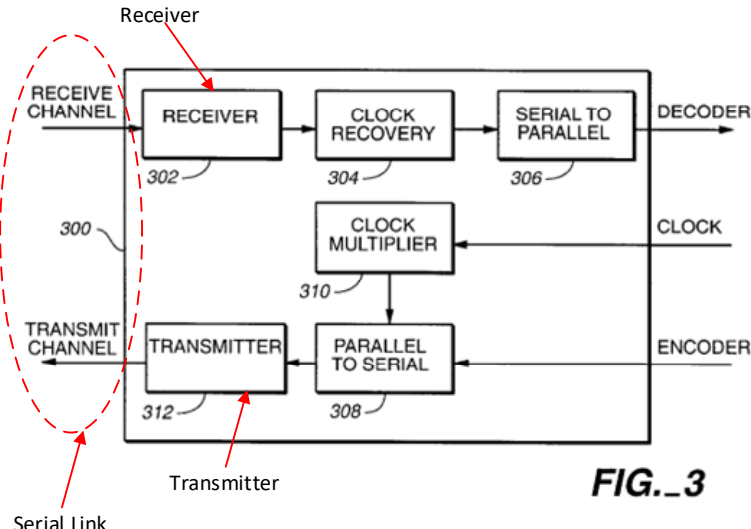
US 9,535,454, Claim 17	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
17. The system of claim 1, wherein the module is further configured to generate a clock signal, for use by the module, using the first serial data.	<u>VANGEMERT</u> “Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signal levels, clean them up as much as possible, and provide them to a clock recovery module 304. Clock recovery module 304 uses the incoming signal to reconstruct a clock signal synchronized to the incoming data, and a serial-to-parallel converter 306 uses the clock signal to convert the incoming signal from serial form to parallel form.” Col. 3:21-28.

US 9,535,454, Claim 17	US 6,289,386 Vangemert (EX 1005) and US 5,179,661 Copeland (EX 1006)
	“Generate a clock signal, for use by the module, using the first serial data” = Reconstructed clock signal, for use by the serial-to-parallel converter 306, to convert incoming signal from serial form to parallel form FIG._3 Vangemert FIG. 3 (Annotated)

B. Ground 2: Vangemert combined with Copeland and Dobson teaches all elements of Claim 4.

US 9,535,454, Claim 4	US 6,289,386 Vangemert (EX 1005); US 5,179,661 Copeland (EX 1006); and US 5,717,870 Dobson (EX 1007).
4. The system of claim 1, wherein: the module is further operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission.	<u>DOBSON</u> “[F]undamental building block of a serial port for a computer or terminal is a universal asynchronous receiver-transmitter, commonly known by the acronym UART. This device accepts parallel words or bytes from the data terminal equipment, generates appropriate timing signals, depending on the bit rate, and transmits the bytes in bit serial form, together with appropriate start and parity checking bits. On the receiver side of the UART, the center of the bit times are determined, based on the approximate data rate, and serial data is clocked into a register to form a word or byte that is then available for reading out in parallel by the data terminal equipment.” Col. 1:35-46. “Data is input into the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16.” Col. 6:10-12. Dobson illustrates that the various functions of transmitting data do not rely on any acknowledgement (or other) signal being received from any device to transmit the required data. See <i>generally</i> col. 5:52-6:16. <u>RATIONALE TO COMBINE</u> A PHOSITA would have been motivated to combine the transmission protocol of Dobson with that of Vangemert to ensure that all data frames within a message stored in the exchange tables 416 can be transmitted within a specific exchange without having to wait for an acknowledgement after each portion of the data is transmitted to meet Vangemert’s goal of minimizing transmission latency. See EX. 1005, at col. 1:54-55, col. 3:49-63.

C. Ground 3: Vangemert in view of Copeland and further in view of Reay teaches all elements of Claim 6.

US 9,535,454, Claim 6	US 6,289,386 Vangemert (EX 1005); US 5,179,661 Copeland (EX 1006); and US 6,005,414 Reay (EX 1008).
6. The system of claim 1, wherein: the module is further operable to transmit [sic] the first serial data over the serial link and to transfer the second serial data over the serial link using different signals.	<u>VANGEMERT</u> Claim 6 depends from claim 1, which recites that the module is operable to “receive” first serial data. “[F]abric switch 114 transports data frames and control words from the transmit channel of a source node to the receive channel of a destination node.” Col. 2:66-3:1. “Each of the nodes is coupled to a central “fabric switch” 114 by a serial link having a receive channel and a transmit channel.” Col. 2:61-63. “FIG. 3 illustrates a serial communication transceiver 300 which may be used by the nodes to couple to the serial communications link. Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signal levels.” Col. 3:19-23. The transceiver 300 also includes a transmitter 312 that “converts the serial bitstream into signals suitable for transport by the transmit channel.” Col. 3:35-36.  FIG._3 <i>Vangemert FIG. 3 (Annotated)</i> <u>REAY</u> “Output terminals 106 and 108 provide inverted and non-inverted output signals, respectively, of the input signal received at terminal 102. Thus, a differential, or balanced-line, output can be provided

US 9,535,454, Claim 6	US 6,289,386 Vangemert (EX 1005); US 5,179,661 Copeland (EX 1006); and US 6,005,414 Reay (EX 1008).
	for those interface standards that require such output.” Col. 4:15-29. “[An] advantage to such outputs is noise immunity; any noise that appears on one of the two output lines is likely to appear on the other as well and is thus cancelled as the difference between the two output signal voltages is taken.” Col. 4:32-36. “Differential signals” = differential output signals <u>RATIONALE TO COMBINE</u> A PHOSITA would have known and been motivated to combine the noise cancelling differential signals of Reay for use on the serial link to reduce errors due to noise with respect to the received and/or transmitted signals. Furthermore, a PHOSITA would have had a reasonable expectation of success in incorporating the differential signaling of Reay into the serial link of Vangemert, as the use of differential signaling for serial communication was well known in the art at the time of Vangemert and Reay.

D. Ground 4: Claims 1, 2, 3, 4, 7, and 16 in view of Dobson and Copeland

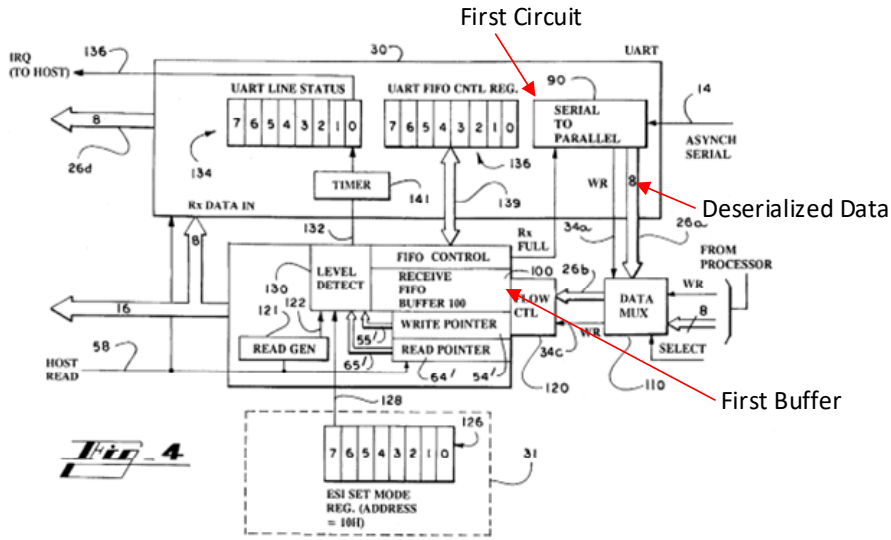
US 9,535,454, Claim 1 [a] 1. A system, comprising:	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006) <u>DOBSON</u> Dobson discloses a system including an enhanced serial port controller 10 in communication with various devices, such as a host computer 8. Col. 5:17-26. “System” = system Dobson FIG. 2 (Annotated)
[b] a connector configured to be connectable to a serial link;	<u>DOBSON</u> “Data is input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16.” Col. 6:10-12. “The enhanced serial port controller 10 also accepts asynchronous, serial data on line 14.” Col. 7:54-56. A PHOSITA would have known that one or more connectors would have been required between the enhanced serial interface of Dobson and one or more other pieces of data communications equipment such as a modem. See Col. 1:23-26. “Serial Link” = Lines 14 and 16 <u>COPELAND</u>

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
	"Host computer 10 is connected to a dual channel asynchronous serial communications card 11 . . . [c]ard 11 contains . . . two serial port connectors 20 and 21." Col. 3:49-60. "[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface." Col. 4:31-34. A PHOSITA would have understood that serial port connector 20 is associated with a first channel of the dual channel serial communications card 11, and serial port connector 21 is associated with a second channel of the dual channel serial communications card 11 and therefore may be used interchangeably. "Connector" = Serial Port Connector 20 or 21 Copeland FIG. 1 (Annotated) <u>RATIONALE FOR COMBINATION</u> A PHOSITA would have known to combine the connector 20 or 21 of Copeland with the serial lines 14 and 16 of Dobson to provide a required mechanism for connecting the serial lines 14 and 16 of the serial port controller 10 to one or more external serial devices. Both Dobson and Copeland are directed to serial interfaces utilizing 16550 UARTs to provide a serial interface for a host device and therefore it would have been obvious to a PHOSITA for the connectors 20 and 21 to be used in both situations. See EX. 1007, at col. 3:20-52; EX. 1006, at col. 2:35-47.
[c] a module coupled to the connector,	<u>DOBSON</u>

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
wherein the module is operable to receive first serial data transmitted over the serial link through the connector and transmit second serial data to the serial link through the connector, and wherein the module includes	Annotated FIG. 2, below, illustrates that the serial link (transmission line 16 and reception line 14), which communicate with other I/O devices, provides serial communications to and from the serial port controller 10. “Data is input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16.” Col. 6:10-12. “[T]he receive side of the enhanced serial port controller 10 is now described. The enhanced serial port controller 10 also accepts asynchronous, serial data on line 14.” Col. 7:53-56. “Module” = Enhanced Serial Port Controller 10 “Second Serial Data” = Serial Data Output on Line 16 “First Serial Data” = Serial Data Received on Line 14 Fig - 2 Dobson FIG. 2 (Annotated) COPELAND “[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” Col. 4:31-34. “Connector” = Serial Port Connectors 20 or 21 RATIONALE TO COMBINE

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
	A PHOSITA would have understood that the serial link of Dobson would be connected to various external devices through a connector such as the connector 20 or 21 of Copeland.
[d] a receiver operable to receive the first serial data from the serial link through the connector;	<u>DOBSON</u> UART stands for “Universal Asynchronous <u>Receiver</u> and.” Col. 1:36-38. “Asynchronous data received on line 14 [and] is passed to the UART [30].” Col. 7:56-57. “Data arriving at the enhanced serial port controller 10 on asynchronous serial line 14 (FIG. 2) and is fed through the serial to parallel converter 90 in the UART 30.” Cols. 7:67-8:3. “Receiver” = Receiver in UART 30 “Serial Link” = Receive lines 14, transmit line 16 <u>Fig. 2</u> Dobson FIG. 2 (Annotated) <u>COPELAND</u> “[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” Col. 4:31-34. “Connector” = Serial Port Connector 20 <u>RATIONALE TO COMBINE</u>

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006) A PHOSITA would have understood that the serial link (line 14) of Dobson would be connected to various external devices through a connector such as the connector 20 of Copeland. [e] a first circuit configured to deserialize the first serial data to generate deserialized data; DOBSON "Data arriving at the enhanced serial port controller 10 on asynchronous serial line 14 (FIG. 2) and is fed through the serial to parallel converter 90 in the UART 30. . . . The data is passed from the serial to parallel converter 90 . . . to the data multiplexor 110 over data bus 26a." Col. 7:67-8:6. "First Circuit" = Serial to Parallel Converter 90 "Deserialized Data" = Data output by UART 30 over data bus 26a Dobson FIG. 4 (Annotated)
[f] a decoder configured to decode the deserialized data;	DOBSON Upon deserializing the received serial data, "[t]he UART responds to and strips start and stop bits and can give an indication of when a parity error is detected." Col. 1:47-49. "Decoder" = UART strips start and stop bits from the data package to restore it to its desired format

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
[g] a first buffer operable to store data that has been deserialized and decoded by the first circuit and the decoder, respectively;	<u>DOBSON</u> Received data, upon being converted to parallel data by serial to parallel converter 90 and subsequently decoded, is written into a receive buffer 100. See col. 8:5-29. “The data is clocked into the receive FIFO buffer 100 using the write signal on line 34a.” Col. 8:24-25. “First Buffer” = Receive FIFO buffer 100 “First Circuit” = Serial to Parallel Converter 90  Dobson FIG. 4 (Annotated)
[h] a second buffer operable to receive data, wherein the second buffer is in communication with a second circuit that has configured to serialize data received via the second buffer to generate the	<u>DOBSON</u> “The data on transmit data bus 22 is provided to the UART 30, the transmit [FIFO] buffer 40, and the command interface 50.” Col. 6:1-3. “The data stored in the transmit FIFO buffer 40 is output on data bus 23 to the UART 30. As will [sic] familiar to those skilled in the art, the data is converted from parallel to serial in the UART 30 and output on line 16.” Col. 7:1-4. “Second Buffer” = Transmit FIFO Buffer 40 “Second Circuit” = Parallel to Serial Converter (serializer) within UART 30 “Second Serial Data” = Serial Data output by UART 30 on line 16

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
second serial data; and	Fig. 2 Dobson FIG. 2 (Annotated)
[i] a transmitter operable to transmit the second serial data to the serial link through the connector;	<u>DOBSON</u> UART stands for “Universal Asynchronous Receiver and <u>Transmitter</u>.” Col. 1:36-38. “Data is input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16.” Col. 6:10-12. “Data written into the FIFO buffer 40 is read from the buffer for transmission on the asynchronous serial data output line 16.” Col. 6:49-51. “Transmitter” = Transmitter in UART 30 “Second Serial Data” = Serial Data output by UART 30 on line 16 “Serial Link” = Lines 14, 16 <u>COPELAND</u> “[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” Col. 4:31-34. “Connector” = Serial Port Connector 20

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
	<u>RATIONALE TO COMBINE</u> A PHOSITA would have understood that the serial data output line 16 of Dobson would be connected to various external devices through a connector such as the connector 20 of Copeland.
[j] wherein the module is configured to determine whether an amount of data stored in the first buffer equals or exceeds a fill amount associated with a storage capacity of the first buffer;	<u>DOBSON</u> “Referring to FIGS. 4 and 5, the apparatus and method for detecting the level of data in the receive FIFO buffer 100 are detailed.” Col. 8:53-55 “The value on bus 147 is compared with the number of data bytes written into the receive FIFO buffer 100, as provided by the adder circuit 150. The number of bytes written into the receive FIFO buffer 100 is output on bus 152, and received in comparator 138. When the number of bytes written into the receive FIFO buffer 100 equals the selected scalable trigger level on bus 147, the comparator 138 outputs the trigger level reached signal on line 132.” Col. 9:33-40. “Standard selectable trigger levels used in a type 16550 UART are provided as well as expanded scalable trigger levels to accommodate the larger buffers.” Abstract. “First Buffer” = Receive FIFO Buffer 100 <u>COPELAND</u> Data flow control is “activated with the FIFO buffer [of UART 14] is forty percent full.” Col. 5:38-40. “UART 14 has another signal, BAUDOUT, which is a clock at 16 times the baud rate. When -RXREADY goes low counter 151 counts the BAUDOUT cycles. When 1024 BAUDOUT cycles pass without the FIFO buffer being empty the Q11 output of the counter 151 goes high. The count of 1024 BAUDOUT cycles represents 64 bit times or, at 10 bits per character, 6.4 character times. This leaves at least 9.6-character times remaining for the remote transmitter connected to serial data port 20 to respond before the FIFO buffer overflow. Thus, in the preferred embodiment of the present invention, data flow control is activated when the FIFO buffer is forty percent full.” Col. 5:28-40. “First Buffer” = FIFO Buffer OF UART 14
[k] wherein the module is operable to generate a flow control signal in	<u>DOBSON</u> “The value on bus 147 is compared with the number of data bytes written into the receive FIFO buffer 100, as provided by the adder circuit 150. The number of bytes written into the receive FIFO buffer 100 is output on bus 152, and received in comparator 138. When the number of bytes written into

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
response to the amount of data stored in the first buffer equaling or exceeding the fill amount; and	the receive FIFO buffer 100 equals the selected scalable trigger level on bus 147, the comparator 138 outputs the trigger level reached signal on line 132. Referring to FIG. 4, the trigger level reached signal on line 132 is fed to the UART 30 to cause the host computer to be interrupted” Col. 9:33-42. “First Buffer” = Receive FIFO Buffer 100 <u>COPELAND</u> Data flow control is “activated with the FIFO buffer [of UART 14] is forty percent full.” Col. 5:38-40. UART 14 includes a 16 byte FIFO buffer and outputs a signal “RXREADY,” which is low “whenever there is at least one byte of data, transmitted from serial data port 20 to the UART 14 FIFO buffer, which has not been read by host 10.” Col. 5: 12-28. “UART 14 has another signal, BAUDOUT, which is a clock at 16 times the baud rate. When -RXREADY goes low counter 151 counts the BAUDOUT cycles. When 1024 BAUDOUT cycles pass without the FIFO buffer being empty the Q11 output of the counter 151 goes high. The count of 1024 BAUDOUT cycles represents 64 bit times or, at 10 bits per character, 6.4 character times. This leaves at least 9.6-character times remaining for the remote transmitter connected to serial data port 20 to respond before the FIFO buffer overflow. Thus, in the preferred embodiment of the present invention, data flow control is activated when the FIFO buffer is forty percent full.” Col. 5:28-40. “When a count of 1024 is reached the q11 output provides an enabling high signal to gates 157 and 158. If microprocessor 23 is providing a second enabling high signal to gates 157 and 158 over Data Terminal Ready 1 (DRT1) line 161 and Request To Send 1 (RTS1) line 162, the output of gates 157 and 158 provide high signals, respectively, to gates 159 and 160, respectively, which disables gates 159 and 160, thereby causing their outputs to be high, which sends a low for the RTS [shorthand for Request to Send] [flow control signal], and DTR [shorthand for Data Terminal Ready] signals on serial port connector 20. This signals the remote transmitter to stop sending data, thus shutting down the flow of data from the serial data port 20 to the UART 14 [first buffer] FIFO buffer.” Col. 5:43-55. “It will be appreciated that whenever host 10 is unable to keep up with the data transfer rate, the Q11 output of counter 151 will go high, thereby causing the DTR and/or RTS output signals on serial port connector 20 to go low and signal the remote transmitter to stop sending.” Cols. 5:65 –6:2. “First Buffer” = FIFO Buffer OF UART 14 “Flow Control Signal” = DTR / RTS output signals <u>RATIONALE TO COMBINE</u>

US 9,535,454, Claim 1	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
	A PHOSITA would have been motivated to combine Copeland's flow control signal with Dobson's system because they are directed to the same technology using 16550 UARTs, Dobson is intended to be backward compatible, and they have similar, complementary goals.
[I] wherein the module is further configured to transmit the flow control signal through the serial link and the connector without passing the flow control signal through the second buffer.	<u>COPELAND</u> FIG. 1 of Copeland illustrates that the Data Flow Controller 150 provides the RTS output on line 165, which does not pass through the host data buffer 13. See Annotated FIG. 1, below. The RTS output does pass through the serial port line buffer 16 but the serial port line buffer 16 is not "in data communication with a second circuit that is configured to serialize data received via the second buffer to generate second serial data" and cannot be the second buffer. Buffer 13 corresponds to the claimed second buffer as it stores parallel data to be converted by UART 14 into serial data. "Flow Control Signal" = DTR / RTS output signals "Second Buffer" = Buffer 13
	Copeland FIG. 1 (Annotated)
	<u>RATIONALE TO COMBINE</u>
	A PHOSITA would have found it obvious to combine the data flow controller of Copeland with the serial port controller 10 of Dobson as they are directed to the same technology using the same UART, as discussed above.

US 9,535,454, Claim 2	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
2. The system of claim 1, wherein the system is configured to transmit the first serial data and the second serial data asynchronously over the serial link.	<u>DOBSON</u> “The enhanced serial port controller 10 provides asynchronous, serial digital data on line 16. As is familiar to those skilled in the art, the asynchronous data is typically sent to a modem for transmission over the public switch telephone network to other locations.” Col. 6:12-16. “The enhanced serial port controller 10 also accepts asynchronous, serial data on line 14. Asynchronous data received on line 14 is passed to the UART [30].” Col. 7:54-57. “Asynchronous” communication = Universal Asynchronous Receiver Transmitter “Serial Link” = Line 16 and Line 14
US 9,535,454, Claim 3	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
3. The system of claim 1, wherein: the data received by the second buffer comprises parallel data.	<u>DOBSON</u> “Data stored in the transmit FIFO buffer 40 is output on data bus 23 to the UART 30. As will [be] familiar to those skilled in the art, the data is converted from parallel to serial in the UART 30 and output on line 16.” Col 7:1-4. “Second Buffer” = Transmit FIFO Buffer 40
US 9,535,454, Claim 4	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
4. The system of claim 1, wherein: the module is further operable to repeatedly transmit the second serial data through the serial link and the connector without receiving acknowledgement during transmission.	<u>DOBSON</u> “The fundamental building block of a serial port for a computer or terminal is a universal asynchronous receiver-transmitter, commonly known by the acronym UART. This device accepts parallel word or bytes from the data terminal equipment, generates appropriate timing signals, depending on the bit rate, and transmits the bytes in bit serial form, together with appropriate start, stop and parity checking bits. On the receiver side of the UART, the center of the bit times are determined, based on the approximate data rate, and serial data is clocked into a register form a word or byte that is then available for reading out in parallel by the data terminal equipment.” Col. 1:36-47.

US 9,535,454, Claim 4	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
	“Data is input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16.” Col. 6:10-12. No acknowledgement or other signal from the receiving device is required to transmit data by the UART 30. Data is simply serialized and transmitted as described above. Further review of the transmission of data by the enhanced serial portion controller 10 of Dobson illustrates that all the various functions of transmitting data do not rely on any acknowledgement (or other) signal being received from any device to transmit the required data. See Col. 5:52-6:16. “Serial Link” = Transmit line 16 <u>COPELAND</u> “[S]erial port connectors 20 and 21 are each connected to a communication device (not shown), such as a modem. Serial port connectors 20 and 21 preferably define an EIA RS-232-D serial interface.” Col. 4:31-34. “Connector” = Serial Port Connector 20 <u>RATIONALE TO COMBINE</u> A PHOSITA would have understood that the serial link of Dobson would be connected to various external devices, such as a modem, through a connector such as the connector 20 of Copeland.

US 9,535,454, Claim 7	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
7. The system of claim 1, wherein: the flow control signal is configured to be used to control transmission of the first serial data over the serial link to prevent overflow of the first buffer.	<u>COPELAND</u> “[D]ata flow control is activated when the FIFO buffer is forty percent full.” Col. 5:39-40. “It will therefore be appreciated that whenever host 10 is unable to keep up with the data transfer rate, the Q11 output of counter 151 will go high, thereby causing the DTR and/or RTS output signals on serial port connector 20 to go low and signal the remote transmitter to stop sending.” Col. 5:65-6:2. “Flow Control Signal” = DTR and/or RTS Output Signal “First Buffer” = FIFO Buffer of UART 14 <u>RATIONALE TO COMBINE</u>

US 9,535,454, Claim 7	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
	A PHOSITA would have found it obvious to combine the data flow controller of Copeland with the serial port controller 10 of Dobson to provide a mechanism for the Receive Flow Control Block 105 to transmit a flow control signal to an external device over serial link 16 to stop sending data in response to an overflow condition on receive FIFO 100. It would have been understood that to prevent the receive FIFO 100 from overflowing, a flow control signal would need to be sent to the external device transmitting the data to stop transmitting, as described in Copeland, above. As both Dobson and Copeland are directed to serial interfaces utilizing 16550 UARTs to provide a serial interface for a host device, a PHOSITA would have been motivated to combine the receive flow control functionality of Copeland with the flow control functionality of Dobson to prevent the target device from sending data until the host PC could read the data from the receive FIFO buffer.

US 9,535,454, Claim 16	US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)
16. The system of claim 1, further comprising: an interface included in the module coupled to the first buffer and operable to receive the data stored in the first buffer; a bus coupled to the module and configured to receive the data from the interface; and a processor coupled to the bus.	<u>DOBSON</u> “Received data is passed from the receive FIFO buffer 100 to the host personal computer 8 via the receive data bus 26d, [and] bus interface circuit 20.” Col. 7:60-63. Annotated FIG. 2 of Dobson, below, discloses all elements of claim 16 arranged as claimed. More specifically, the interface 20 is coupled to the first buffer via data line 26d, the interface 20 is coupled to the host PC 8 via “the host personal computer data bus 12 and control bus 13, both of which are familiar to those skilled in the art.” Col. 5:52-55. A PHOSITA would understand that a host PC, by virtue of being a personal computer, would include a processor.

US 9,535,454,
Claim 16

US 5,717,870 Dobson (EX 1007) and US 5,179,661 Copeland (EX 1006)

Processor

Bus

Interface

Module

HOST PC

MCA/ISA

MULTIPOINT INTERFACE

CASCADED ESP BICS

MODIFIED 16550 UART (MEGACELL)

Tx

Rx

Tx FLOW CONTROL BLOCK

Rx FLOW CONTROL BLOCK

ESP COMMAND INTERFACE

DATA MUX

μP INTERFACE

ASYNCHRONOUS SERIAL I/O

INTERNAL Tx DATA PATH

INTERNAL Rx DATA PATH

CONTROL/STATUS

First Buffer

FIG. 2

Dobson FIG. 2 (Annotated)

“Interface” = Bus Interface Circuit 20

“First Buffer” = Receive FIFO Buffer 100

“Bus” = Personal Data Bus 12

“Processor” = Host PC 8

E. Ground 5: Claim 6 in view of Dobson, Copeland, and Reay

US 9,535,454, Claim 6	US 5,717,870 Dobson (EX 1007), US 5,179,661 Copeland (EX 1006), and US 6,005,414 Reay (EX 1008)
6. The system of claim 1, wherein: the module is further operable to transmit [receive] the first serial data over the serial link and to transfer the second serial data over the serial link using differential signals.	<u>DOBSON</u> “Referring back to FIG. 2, the receive side of the enhanced serial port controller 10 is now described. The enhanced serial port controller 10 also accepts asynchronous, serial data on line 14.” Col. 7:53-56. “Data is input to the UART 30 in parallel, is converted by the UART 30 to serial, and is output from the UART 30 on line 16.” Col. 6:10-12. “Serial Link” = Line 14 and Line 16 <u>REAY</u> “Output terminals 106 and 108 provide inverted and non-inverted output signals, respectively, of the input signal received at terminal 102. Thus, a differential, or balanced-line, output can be provided for those interface standards that require such outputs.” Col. 4:15-19. “The advantage of such outputs is noise immunity; any noise that appears on one of the two output lines is likely to appear on the other as well, and is thus canceled as the difference between the two output signal voltages is taken.” Col. 4:32-36. “Differential Signals” = “differential output” <u>RATIONALE TO COMBINE</u> A PHOSITA would have found it obvious and been motivated to combine the noise cancelling differential signals of Reay for use on the serial link lines 14 and 16 to reduce errors due to noise with respect to the received and/or transmitted signals.

F. Ground 6: Claims 15 and 17 in view of Dobson, Copeland, and Vangemert

US 9,535,454, Claim 15	US 5,717,870 Dobson (EX 1007), US 5,179,661 Copeland (EX 1006), and US 6,289,386 Vangemert (EX 1005)
15. The system of claim 1, wherein: the system comprises a hard disk drive.	<u>DOBSON</u> Dobson teaches that the enhanced serial port controller 8 is coupled to an external device, such as host PC 8, which would include a hard disk drive. <u>VANGEMERT</u> As shown in FIG. 1, below, the illustrated system is shown including a disk subsystem 104. A disk subsystem would be understood by a PHOSITA to disclose a hard disk drive (or multiple hard disk drives). “Hard Disk Drive” = Disk Subsystem 104 FIG. 1 Vangemert FIG. 1 (Annotated) <u>RATIONALE TO COMBINE</u> A PHOSITA would have understood to combine the hard disk drive of Vangemert with the system of Dobson to allow for the high-speed serial port of Dobson to communicate with the data storage devices of Vangemert.
US 9,535,454, Claim 17	US 5,717,870 Dobson (EX 1007), US 5,179,661 Copeland (EX 1006), and US 6,289,386 Vangemert (EX 1005)
17. The system of claim 1, wherein: the	<u>DOBSON AND COPELAND</u>

US 9,535,454, Claim 17	US 5,717,870 Dobson (EX 1007), US 5,179,661 Copeland (EX 1006), and US 6,289,386 Vangemert (EX 1005)
module is further configured to generate a clock signal, for use by the module, using the first serial data.	Dobson and Copeland disclose asynchronous communication and would have been understood to generate clock signals using incoming data. <u>VANGEMERT</u> “Transceiver 300 includes a receiver 302 coupled to the receive channel to sense incoming signal levels, clean them up as much as possible, and provide them to a clock recovery module 304. Clock recovery module 304 uses the incoming signal to reconstruct a clock signal synchronized to the incoming data, and a serial-to-parallel converter 306 uses the clock signal to convert the incoming signal from serial form to parallel form.” Col. 3:21-28. “Clock Signal” = Clock Signal Reconstructed by Clock Recovery Module 304 “First Serial Data” = Incoming Data <u>RATIONALE TO COMBINE</u> A PHOSITA would have understood to combine the clock recovery functionality of Vangemert with the asynchronous, high-speed serial port of Dobson to facilitate data synchronization as Dobson’s received “data is clocked into the receive FIFO buffer 100 using the write signal on line 34a.” <i>EX. 1007</i>, at col. 8:24-25.