IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF COLORADO
Judge Philip A. Brimmer

Civil Action No. 09-cv-01257-PAB-MEH

BIAX CORPORATION,

Plaintiff,

v.

NVIDIA CORPORATION,
SONY COMPUTER ENTERTAINMENT AMERICA, INC., and
SONY ELECTRONICS, INC.,

Defendants.

ORDER REGARDING CLAIM CONSTRUCTION

This matter is before the Court for the construction of U.S. Patent No. 5,517,628 (the "'628 Patent") [Docket No. 146-1] and U.S. Patent No. 6,253,313 (the "'313 Patent") [Docket No. 146-2] (collectively, the "patents"), both assigned to plaintiff BIAX Corporation ("BIAX"). BIAX has brought suit against defendants NVIDIA Corporation, Sony Computer Entertainment America, Inc., and Sony Electronics Inc., alleging that defendants have infringed the patents. At this stage of the case, the Court is called upon to construe certain disputed terms in the patents.

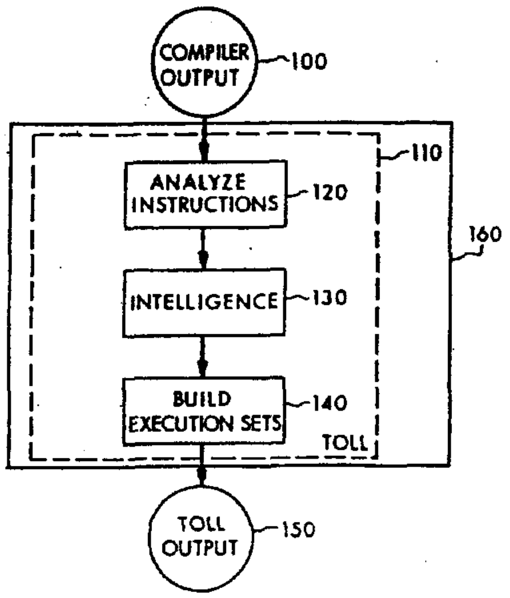
I. BACKGROUND

The parties' disagreement regarding claim construction involves language appearing in many claims of the '628 Patent and the '313 Patent. *See generally* Defs.' Br. [Docket No. 144]; Pl.'s Br. [Docket No. 149]. The patents have the same specification, as both patents derive from U.S. Patent No. 4,847,755 (the "'755 Patent").

See Defs.' Br. at 9; *see also* '628 Patent [Docket No. 146-1 at 49] ("Related U.S. Application Data"); '313 Patent [Docket No. 146-2 at 2] ("Related U.S. Application Data"). As stated in their shared specification, the "invention generally relates to parallel processor computer systems and, more particularly, to parallel processor computer systems having software for detecting natural concurrencies in instruction streams and having a plurality of processor elements for processing the detected natural concurrencies." '313 Patent col. 1 ll. 18-23.

The patents describe both software and hardware. The software, which is referred to as "TOLL" software, acts upon object code produced by the computer system's compiler. This object code constitutes a series of instructions, upon which the software "performs three basic determining functions." '313 Patent col. 7 l. 25. The software (1) "analyze[s] the resource usage of the instructions," (2) "extend[s] intelligence for each instruction in each basic block," and (3) "build[s] execution sets composed of one or more basic blocks." '313 Patent col. 7 ll. 26-29. These functions are depicted in Figure 1:

FIG. 1



The software is capable of identifying "natural concurrencies" in the instructions and assigning hardware components, including processor elements, accordingly.

Instructions that are naturally concurrent do not depend on each other and can, therefore, be executed by processor elements simultaneously. Dependent instructions can be placed into sets of instructions that depend on each other, and, "unlike the teachings of the prior art, the present invention teaches that it is not necessary for dependent instructions to execute on the same processor element." '313 Patent col. 9 ll. 34-38. Rather, "[t]he determination of dependencies is needed only to determine condition code sets and to determine instruction firing times. . . ." '313 Patent col. 9 ll. 37-39. The invention relies on processor elements that access register files containing relevant information for subsequent processing of instructions. Included within the teachings of the inventions are condition code register files containing multiple, addressable condition code registers, each of which contain condition codes indicating whether the results of executed instructions meet certain conditions. The condition codes can include indications of whether the next instruction in a series should be executed or whether a branch must be taken, requiring a jump to an earlier or later instruction. The provision of multiple, addressable condition code registers is a central feature of the claimed invention. As a consulting engineer to BIAX has declared,

> [w]ith the BIAX technology of the '313 patent, the condition setting instruction was no longer needed to be placed immediately before the conditional branch instruction. By providing multiple, addressable condition code registers, there was no longer a concern that a second condition-setting instruction would overwrite the results of a first condition-setting instruction and the branch instructions would test the wrong condition code value. That is because the first condition-setting instruction and the second condition-setting instruction could store their respective condition codes in different addressable registers.

Docket No. 146-13 at 4. These qualities are depicted in Figure 6:

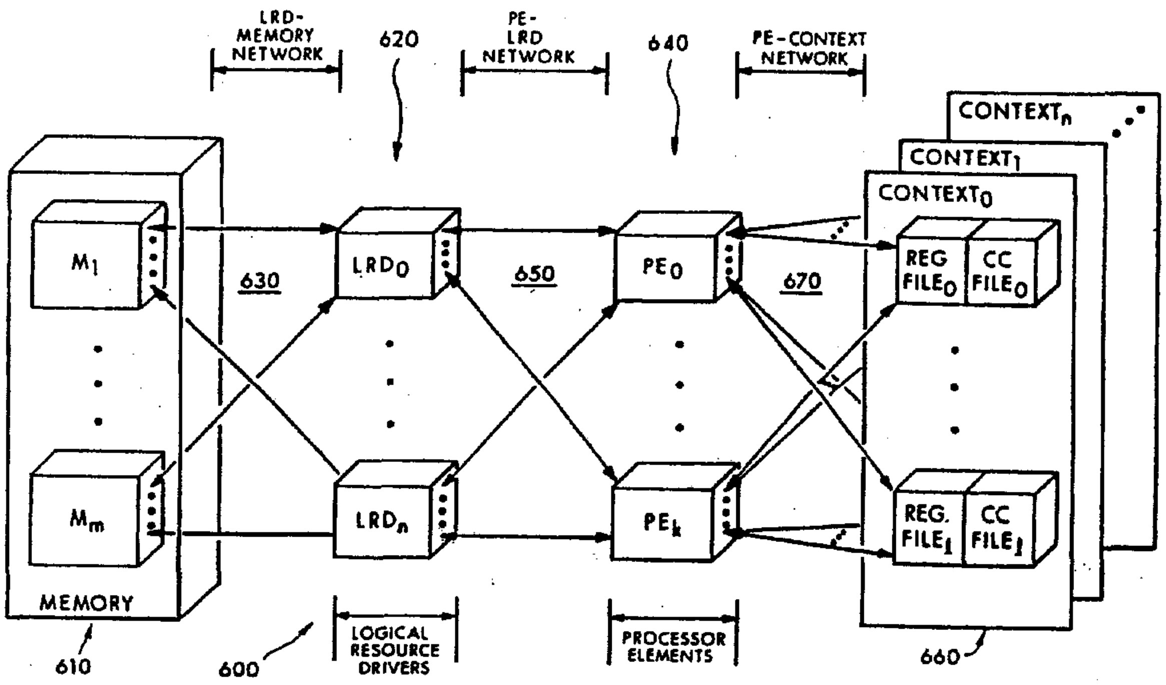


FIG. 6

Additional features described in the patents will be discussed below as relevant to the construction of particular claim terms.

As an overview, Claim 1 of the '628 Patent and Claim 3 of the '313 Patent contain many of the disputed terms found elsewhere in the patents' claims. Claim 1 of the '628 Patent describes:

> A computer comprising:
>
> a general purpose register file comprising at least two general purpose registers;
>
> a condition code register file distinct from said general purpose register file, having a plurality of addressable condition code registers, each condition code register for representing a condition code value as a small number of

> bits summarizing the execution or result of a previously-executed instruction;
>
> a processor element configured to execute instructions, including condition-setting instructions that each produce a condition code value for storage in one of said condition code registers;
>
> a branch execution unit configured to execute conditional branch instructions that each determine a target instruction for execution based on analysis of a condition code value from one of said condition code registers; and
>
> a condition code access unit configured to act in response to condition-selecting instructions, at least one of said condition-selecting instructions being one of either said condition-setting instructions or said conditional branch instructions, said condition-selecting instructions for selecting from said condition code register file a condition code register for at least one of:
>
>> storing into said selected condition code register a condition code value produced by one of said condition-setting instructions, and
>>
>> fetching from said selected condition code register a condition code value for analysis by one of said conditional branch instructions;
>>
>> said selecting being by direct addressing on a condition code address field of the condition-selecting instruction.

’628 Patent col. 45 l. 63 - col. 46 l. 28. Claim 3 of the ’313 Patent provides for the following:

> In an instruction processing apparatus, a system for using multiple sets of condition values within a single context, the system comprising:
>
> an opcode storage configured to buffer a plurality of opcodes corresponding to at least some of said instructions to be processed from said context;
>
> a first circuit coupled to said opcode storage, said first circuit configured to receive opcodes of a first type of instruction and addresses for storing sets of at least one condition value, and to generate for each said opcode of said first type a set of at least one condition value associated with one of the addresses;
>
> a condition storage coupled to said first circuit, said condition storage configured to store a plurality of said sets of condition values at storage locations based upon addresses received from the first circuit;

> a second circuit coupled to said opcode storage, said second circuit configured to receive opcodes of a second type of instruction and condition storage addresses, and to generate a set of at least one output value, said output value for each said second type of instruction depending on a particular one of said stored sets of condition values associated with the condition storage address; and
>
> an access circuit coupled between said condition storage and said second circuit, said access circuit configured to access by the condition storage address said particular one of said stored sets of condition values for each said opcode of said second type.

'313 Patent col. 46 l. 48 - col. 47 l. 11. Along with these claims, Claims 9 through 14, 16, 17, 20, 23, 25, 26 and 29 of the '628 Patent and Claims 4, 5, 8, 9, 12, 14 through 16, 19 through 21, and 24 of the '313 Patent contain terms in dispute. Before construing those terms, the Court will first discuss the legal standards applicable to claim construction.

II. LEGAL STANDARDS FOR PATENT CLAIM CONSTRUCTION

Claim construction is a question of law for the court. *See, e.g.*, *Cybor Corp. v. FAS Techs., Inc.*, 138 F.3d 1448, 1454 (Fed. Cir. 1998) (en banc). In construing patent claims, courts are guided by the precedent of the Federal Circuit. *See SunTiger, Inc. v. Scientific Research Funding Group*, 189 F.3d 1327, 1333 (Fed. Cir. 1999). As that court has explained, "there is no magic formula or catechism for conducting claim construction." *Phillips v. AWH Corp.*, 415 F.3d 1303, 1324 (Fed. Cir. 2005) (en banc). Even so, the *Phillips* decision outlined several key sources and doctrines that should be consulted and applied, all the while making clear that "[t]he sequence of steps used by the judge in consulting various sources is not important; what matters is for the court to attach the appropriate weight to be assigned to those sources in light of the statutes and policies that inform patent law." *Id.*

Courts begin with the "bedrock principle" that "'the claims of the patent define the invention to which the patentee is entitled the right to exclude.'" *Id.* at 1312 (quoting *Innova/Pure Water, Inc. v. Safari Water Filtration Systems, Inc.*, 381 F.3d 1111, 1115 (Fed. Cir. 2004)). The words of the claims "'are generally given their ordinary and customary meaning,'" *id.* (quoting *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1582 (Fed. Cir. 1996)), which is "the meaning that the term would have to a person of ordinary skill in the art in question at the time of the invention," *id.* at 1313; *see CCS Fitness, Inc. v. Brunswick Corp.*, 288 F.3d 1359, 1366 (Fed. Cir. 2002) ("Generally speaking, [courts] indulge a 'heavy presumption' that a claim term carries its ordinary and customary meaning."). Sometimes, when the claim language "involves little more than the application of the widely accepted meaning of commonly understood words," construction is relatively straightforward and "the ordinary meaning . . . may be readily apparent even to lay judges." *Phillips*, 415 F.3d at 1314. However, when the claim terms have a particular meaning in the field, courts "look[ ] to 'those sources available to the public that show what a person of skill in the art would have understood disputed claim language to mean.'" *Id.* (quoting *Innova*, 381 F.3d at 1116). These sources include "'the words of the claims themselves, the remainder of the specification, the prosecution history, and extrinsic evidence concerning relevant scientific principles, the meaning of technical terms, and the state of the art.'" *Id.*

Importantly, claim terms are not read in a vacuum. *Id.* at 1313. The context in which a term is used, both in the asserted claim as well as in other claims of the patent, can be valuable and instructive. *Id.* at 1314. In addition, the patent specification – the text and figures of the patent that precede the claims – "'is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single best guide to the meaning of a disputed term.'" *Id.* at 1315 (quoting *Vitronics*, 90 F.3d at 1582). Courts also consider the patent's prosecution history – the official record of the patent application and subsequent process before the U.S. Patent and Trademark Office. *Id.* at 1317. That history "provides evidence of how the PTO and the inventor understood the patent." *Id.* However, "because the prosecution history represents an ongoing negotiation between the PTO and the applicant, . . . it often lacks the clarity of the specification and thus is less useful for claim construction purposes." *Id.*

Courts may consult extrinsic evidence such as "expert and inventor testimony, dictionaries, and learned treatises." *Id.* However, this evidence is "'less significant than the intrinsic record,'" i.e., the specification and prosecution history, *id.* (quoting *C.R. Bard, Inc. v. U.S. Surgical Corp.*, 388 F.3d 858, 862 (Fed. Cir. 2004)), and courts must be wary not to use extrinsic evidence to override the meaning of the claim terms demonstrated by the intrinsic evidence. *Id.* at 1318-19. That is, "extrinsic evidence may be useful to the court, but it is unlikely to result in a reliable interpretation of patent claim scope unless considered in the context of the intrinsic evidence." *Id.* at 1319.

In sum, a court's basic role is to construe the claim terms as they would be viewed by "the ordinary artisan after reading the entire patent." *Id.* at 1321. This is crucial in order to respect the public notice function of patents:

> The patent system is based on the proposition that claims cover only the invented subject matter. As the Supreme Court has stated, "[i]t seems to us

that nothing can be more just and fair, both to the patentee and the public, than that the former should understand, and correctly describe, just what he has invented, and for what he claims a patent.

Id. at 1321 (quoting *Merrill v. Yeomans*, 94 U.S. 568, 573-74 (1876)). With these principles in mind, I turn to the claim construction issues presented in this case.

III. ANALYSIS

Before construing the specific disputed terms, the Court addresses two preliminary issues raised by the parties: (1) new matter and (2) the effect of prior judicial construction of the claims.

A. New Matter

While the original '755 Patent application was pending, BIAX submitted a substitute specification to the Patent Office. Defendants contend that the substitute specification improperly introduced "new matter" to the patent. *See* 35 U.S.C. § 132 ("No amendment shall introduce new matter into the disclosure of the invention."). Therefore, defendants are requesting that, when construing certain claim terms, the Court rely upon the original specification, instead of the substitute specification that became part of the filed patents. The Court agrees with plaintiff that the issue of "new matter" is not appropriately addressed at the claim construction stage. *See Commonwealth Scientific and Indus. Research Organisation v. Buffalo Tech. (USA), Inc.*, 542 F.3d 1363, 1380 (Fed. Cir. 2008) ("The question whether new matter has been added to an application is a question of fact.") (citing *Brooktree Corp. v. Advanced Micro Devices, Inc.*, 977 F.2d 1555, 1574 (Fed. Cir. 1992)); *Reiffin v. Microsoft Corp.*, No. C-98-0266, 2002 WL 31268220, at *6 (N.D. Cal. April 30, 2002) ("Courts do not make a determination whether a patent contains new matter at the claim construction stage.").

No motion for summary judgment, with corresponding briefing from the parties, is before the Court on the issue. *See Advanced Technology Incubator, Inc. v. Sharp Corp.*, No. 2:07-CV-468, 2009 WL 4403314, at *20 (E.D. Tex. June 26, 2009) ("[I]nvalidity for 'new matter' should be addressed in the context of summary judgment or trial."). If the Court were to rely upon the original specification, it would effectively be resolving the new matter issue in defendants' favor. That issue, however, is not ripe for disposition. *Cf. Pliant Corp. v. MSC Marketing and Technology, Inc.*, 416 F. Supp. 2d 632, 643 n.5 (N.D. Ill. 2006) ("Defendants make a decent case for the fact that claim 9 is invalid because it adds impermissible 'new matter' in violation of 35 U.S.C. § 132. While this argument may ultimately prevail, it is not a matter for claim construction."). Therefore, the Court will construe the claims in light of the current specification. With that said, the original specification remains a part of the prosecution history of the patents at issue.

B. <u>Prior Cases</u>

At least two courts have already construed the present patents, including many of the claim terms at issue here. *See BIAX Corp. v. Sun Microsystems, Inc.*, No. 2:06-CV-364, 2008 WL 2811304 (E.D. Tex. July 18, 2008); *BIAX Corp. v. Intel Corp.*, No. 2:05-CV-184, 2007 WL 677132 (E.D. Tex. March 1, 2007). Plaintiff does not assert that these decisions should have collateral estoppel effect, but does argue, on prudential grounds, that the Court adopt the prior constructions of the claim terms. *See*

Docket No. 149 at 7-8 (arguing that parties in other litigation have relied – and members of the public will rely – on those constructions, that the prior rulings were "well-reasoned and persuasive," and that deference to those constructions will streamline the present litigation). To the extent the Court agrees with the conclusions reached by the prior courts, those prudential concerns will largely be addressed. But, in the absence of any contention that those decisions have binding collateral estoppel effect, the Court must independently address the terms at issue in the matter presently before it.[1] The Court therefore will now construe the claim terms in dispute.

C. <u>'628 Patent</u>

The parties dispute the proper constructions of the following terms appearing in the '628 Patent: processor element, processor, instruction, conditional branch instruction, condition code access unit, condition code register, general purpose register, addressable, condition code address field, and field directly addressing one of said condition code registers.

1. Processor element

Plaintiff suggests the following construction of "processor element": "A device that is capable of interpreting and executing instructions." Docket No. 149 at 14.[2] Defendants seek the following definition: "A context-free device that is capable of

[1]*Cf. Lamps Plus, Inc. v. Dolan*, No. 3:01-CV-1537, 2003 WL 22435702, at *2 (N.D. Tex. Aug. 26, 2003) ("While this Court agrees that neither collateral estoppel nor *stare decisis* dictate the adoption of the [previous] claim construction on the patent at issue in this case, this Court finds the [previous] court's determination instructive.").

[2]All page numbers following references to docket numbers refer to the page indicated at the top of the docket entry and not necessarily any page numbers supplied by the filer at the bottom of any respective page.

interpreting and executing instructions. All processor elements are identical." Docket No. 144 at 31. Both parties agree that a processor element is capable of interpreting and executing instructions. The question is whether the processor elements referenced in the '628 Patent are "context-free" and identical to all other processor elements. Claim 1 of the '628 Patent claims "a processor element configured to execute instructions, including condition setting instructions that each produce a condition code value for storage in one of said condition code registers." '628 Patent col. 46 ll. 5-8.

By arguing that processor elements need not be context free, plaintiff essentially contends that a processor element can retain information regarding the execution of prior instructions. *See* '313 Patent col. 4 ll. 18-21 (where being "context free" is defined as "contain[ing] no execution state information from the execution of previous instructions"). That, however, is not clearly claimed in the language of Claim 1 and the evidence in the record belies the notion that the claimed processor element can retain information regarding prior instructions.

For example, the specification[3] provides:

> The processor elements, in this illustrated embodiment, contain no execution state information from the execution of previous instructions, that is, they are context free. In addition, a plurality of context files, one for each user, are provided wherein the plurality of processor elements can access any storage resource contained in any context file through total coupling of the processor element to the shared resource during the processing of an instruction. In a preferred aspect of the present invention, no condition code or results registers are found on the individual processor elements.

[3]The '313 Patent and the '628 Patent share the same specification. Because the '628 Patent was originally issued with the original specification, and later reissued with the substitute specification, BIAX cites the '313 Patent specification throughout to avoid any confusion, and so will the Court.

'313 Patent col. 4 ll. 18-28. In support of its argument that the processor elements need not be context free, BIAX points out that the specification goes on to provide that "[i]n another preferred particular embodiment of the invention wherein pipelined processor elements are employed, the processor elements are not *strictly* context free as was described previously." '313 Patent col. 15 ll. 30-34 (emphasis added). This passage from the specification, however, does not clearly indicate that there is not a context free quality to the processor elements. Rather, and as additional intrinsic evidence supports, the processor elements described in the patents are either context free or "substantially" so, as explained below.

The claimed invention differentiates itself from the prior art, at least in part, based on the context free quality of the processor elements. For example, the specification provides that

> unlike the teachings of the prior art, the present invention teaches that it is not necessary for dependent instructions to execute on the same processor element. The determination of dependencies is needed only to determine condition code sets and to determine instruction firing times, as will be described later. The present invention can execute dependent instructions on different processor elements, in one illustrated embodiment, because of the context free nature of the processor elements and the total coupling of the processor elements to the shared resources, such as the register files, as will also be described below.

'313 Patent col. 9 ll. 34-45. The specification, as noted above, does make clear that the *strictly* context free processor elements are a preferred embodiment and that the "processor elements *can be* context free and thereby data, condition, or information relating to past processing is not required, nor does it exist, internally to the processor element." '313 Patent col. 16 ll. 61-64 (emphasis added); *see* '313 Patent col. 16 ll. 64-67 ("The context free processor elements react only to the requirements of each individual instruction and are interconnected by the hardware to the necessary shared registers."). That does not, however, clearly support the plaintiff's position that the construction of "processor element" need not contain any reference to a context free quality. *See* '313 Patent col. 17 ll. 33-40 ("Because the timing and the identity of the shared resources and the processor elements are all contained within the extended intelligence added to the instructions by the TOLL software, each processor element . . . can be completely (or in some instances substantially) context free and, in fact, from instruction firing time to instruction firing time can process individual instructions of different users as delivered by the various logical resource drivers."); *see also* '313 Patent col. 4 ll. 65-66 ("It should be noted that many alternative implementations of context free processor elements are possible."). Indeed, as noted already, the patent makes clear that being context free, at least after execution of an instruction or set of instructions, is a distinguishing feature of the invention. *See* '313 Patent col. 17 ll. 44-48 ("It is the context free nature of the processor elements which allows the independent access by any processor element of the results of data generation/manipulation from any other processor element following the completion of each instruction execution."); '313 Patent col. 17 ll. 54-56 ("It is also the context free nature of the processor elements that permits the true sharing of the processor elements by multiple LRDs."); '313 Patent col. 17 ll. 56-64 ("This sharing can be as fine-grained as a single instruction cycle. No programming or special processor operations are needed to save the state of one context (assigned to one LRD), which has control of one or more processor elements, in order to permit control by another context (assigned to a second LRD). In processors which are not context free, which is the case for the prior art, specific programming and special machine operations are required in such state-saving as part of the process of context switching.").

During prosecution of the original '755 Patent, plaintiff distinguished prior art, in part, by asserting that a particular reference "simply does not provide for the complete interconnection of the processors with *plural* shared resources when properly combined with the McDowell system reference, which by its very nature does not operate in a context-free mode nor with provision for plural resources with plural processors." Docket No. 182-8 at 6-7 (emphasis in original); *see Bradford Co. v. Conteyor North America, Inc.*, 603 F.3d 1262, 1270 (Fed. Cir. 2010) ("'[A] court should also consider the patent's prosecution history, if it is in evidence . . . . Like the specification, the prosecution history provides evidence of how the [Patent Office] and the inventor understood the patent.'") (quoting *Phillips v. AWH Corp.*, 415 F.3d 1303, 1317)) (alterations in original). In a Petition Pursuant to 37 C.F.R. § 1.102(d) dated October 31, 1985 requesting Advancement of Examination in the prosecution of the '755 Patent, plaintiff represented that the

> present invention further executes the basic blocks containing the added intelligence on a system using a plurality of identical processor elements each of which is incapable of retaining execution state information from prior operations. Hence, all processor elements are context free and this feature is not found in any of the above references and is specifically claimed in the following independent claims: 1, 2, 3, 6, 11, 13, 60, 68, and 73. A given processor element in the present invention is capable of executing an instruction from one user followed from [sic] an instruction from another user or followed by an instruction from another independent concurrent stream of the same user. All context information necessary for processing a given instruction being contained elsewhere in the system.

Docket No. 146-28 at 23. Moreover, during a reexamination of the patents filed with the Patent and Trademark Office ("PTO") in 2006, as an appendix to a Response to Office

Action, plaintiff filed an expert report of Professor William J. Dally. *See* Docket No. 149-10.[4] Professor Dally wrote that "[t]he specification of the '628 and '313 patents describe a novel computing system that incorporates several distinct inventions including . . . context free processor elements . . . ." *Id.* at 11; *see id.* at 12 ("A second distinct invention disclosed in the specification involves sharing a pool of processor elements (execution units) by multiple contexts or threads of control ('313, 15:15-34). Each context includes its own register file. Thus, for the processor elements to be shared, the register files must be logically separated from the processor elements making them context free."); *id.* at 14 ("One of the inventions described in the specification of the 628 and 313 patents, sharing context-free processor elements (execution units) across multiple threads of control – what is today called simultaneous multithreading, depends on explicit parallelism."); *id.* at 16 ("inventions of . . . context-free processor elements. . ."); *but see id.* at 17 ("As I stated above in Section 4, it is my opinion that a person of ordinary skill in the art would have recognized that the 628 and 313 patents disclosed multiple independent inventions . . . . Additionally, the originally filed claims make clear that, while the detailed description discloses a single embodiment that included several inventions, these inventions were independent of each other. For example, original claim 73 specifies context-free processor elements,

[4]Plaintiff submitted this report to the PTO in its effort to survive the reexamination process. Therefore, it is part of the prosecution history of the patents and does not constitute extrinsic evidence in the form of testimony by an "after-the-fact 'expert[].'" *Bell & Howell Document Management Prods. Co. v. Altek Systems*, 132 F.3d 701, 706 (Fed. Cir 1997) ("Patents should be interpreted on the basis of their intrinsic record, not on the testimony of such after-the-fact 'experts' that *played no part in . . . prosecution of the patent.*") (emphasis added).

whereas other claims, such as claim 53, does not.").[5] The claims, read in the light of the specification and the prosecution history,[6] provide for a processor element that does not generally store context information.

Plaintiff correctly points out that the specification contemplates "pipeline processors." During the *Markman* hearing, plaintiff argued that such pipeline processors "are not context free." In support of that argument, plaintiff contended that a pipeline processor was a hardware component that is "inherently not context free." These arguments are directly contradicted by the specification. The specification "note[s] that many alternative implementations of context free processor elements are possible." '313 Patent col. 4 ll. 65-66. For instance, "each processor element" can be "monolithic and execute[] a single instruction to its completion prior to accepting another instruction." '313 Patent col. 4. l. 67 - col. 5 l. 2. This is a "non-pipelined implementation." '313 Patent col. 4 ll. 66-67. The specification goes on to describe a pipelined implementation:

In another aspect of the invention, the *context free* processor is a pipelined

---

[5]Plaintiff filed a Declaration Under 37 C.F.R. § 1.131 relating to the '313 Patent wherein the co-inventors stated that "[n]one of these resources is dedicated to any specific processor element; rather, each processor element has equal access to all of context storage." Docket No. 146-14 at 5; *see* 37 C.F.R. § 1.131(a) ("When any claim of an application or a patent under reexamination is rejected, the inventor of the subject matter of the rejected claim, the owner of the patent under reexamination, or the party qualified under §§ 1.42, 1.43, or 1.47, may submit an appropriate oath or declaration to establish invention of the subject matter of the rejected claim prior to the effective date of the reference or activity on which the rejection is based. . . .").

[6]*Cf. Arlington Industries, Inc. v. Bridgeport Fittings, Inc.*, 345 F.3d 1318, 1332 (Fed. Cir. 2003) ("Arguments made in a reexamination proceeding will constitute a disclaimer of claim scope if they are 'clear and unmistakable statements of disavowal.'") (quoting *Cordis Corp. v. Medtronic Ave, Inc.*, 339 F.3d 1352, 1358 (Fed. Cir. 2003)).

> processor element, in which each instruction requires several machine instruction clock cycles to complete. In general, during each clock cycle, a new instruction enters the pipeline and a completed instruction exists [sic] the pipeline, giving an effective instruction execution time of a single instruction clock cycle. However, it is also possible to microcode some instructions to perform complicated functions requiring many machine instruction cycles. In such cases the entry of new instructions is suspended until the complex instruction completes, after which the normal instruction entry and exit sequence in each clock cycle continues. Pipelining is a standard processor implementation technique and is discussed in more detail later [in the specification].

'313 Patent col. 5 ll. 3-16 (emphasis added). Moreover, whatever unique hardware features a particular processor element might have, the TOLL software "must handle all hardware idiosyncrasies and their effects on execution of the program instructions stream," "accommodat[ing], when necessary, processor elements which are either monolithic single cycle or pipelined." '313 Patent col. 6 ll. 47 - 51. When the invention uses "pipelined" processors, "the processors are not *strictly* context free as was described previously." '313 Patent col. 15 ll. 32-34 (emphasis added). That is because "there exist data dependencies between the instructions from the basic block just executed, and the instructions from the basic block to be executed." '313 Patent col. 40 ll. 2-4. Yet, contrary to plaintiff's contention that this renders a pipelined processor element not context free, the specification makes clear that "[i]mportantly, the additional data (chaining) paths do not change the fundamental context free nature of the processor elements of the present invention." '313 Patent col. 40 ll. 59-61. "That is, at any given time (for example, the completion of any given instruction cycle), the entire process state associated with a given program (that is, context) is captured completely external to the processor elements." '313 Patent col. 40 ll. 61- 65. Any "replication of . . . data" implicit in pipelined processing is only "transitory." '313 Patent col. 40 ll. 65-66.

The specification also describes another embodiment of the processor element that is not strictly context free, yet still "can be considered a *substantially* context free processor element implementation." '313 Patent col. 18 ll. 16-18 (emphasis added). The specification provides:

> There is one additional alternative in implementing the processor elements of the present invention, which is a modification to the context free concept: an implementation which provides the physically total interconnection discussed above, but which permits, under program control, a restriction upon the transmission of generated data to the register file following completion of certain instructions.
>
> In a fully context free implementation, at the completion of each instruction which enters the processor element, the state of the context is entirely captured in the context storage file. In the alternative case, transmission to the register file is precluded and the data is retained within the processor and made available (for example, through data chaining) to succeeding instructions which further manipulate the data. Ultimately, data is transmitted to the register file after some finite sequence of instructions completes; however, it is only the final data that is transmitted.
>
> This can be viewed as a generalization of the case of a microcoded complex instruction as described above, and can be considered a substantially context free processor element implementation. In such an implementation, the TOLL software would be required to ensure that dependent instructions execute on the same processor element until such time as data is ultimately transmitted to the context register file. As with pipelined processor elements, this does not change the overall functionality and architecture of the TOLL software, but mainly affects the efficient scheduling of instructions among processor elements to make optimal use of each instruction cycle on all processor elements.

'313 Patent col. 17 l. 65 - col. 18 l. 26.

In light of the foregoing, it appears that the specification describes three variants of "context free processor elements." First, it refers to *fully* or *strictly* context free processor elements. *See* '313 Patent col. 35 ll. 59-64 ("If the processor elements are context free, the processor elements can be assigned in any manner whatsoever.").

When the processor element is fully context free in this way, "an instruction is issued to the processor element and the processor element completely executes the instruction before proceeding to the next instruction." '313 Patent col. 40 ll. 18-21.

Second, the patents also describe the "employ[ment of] pipelined processor elements," wherein the "assignment of the processor elements is more complex . . . ." '313 Patent col. 40 ll. 22-25; *see* '313 Patent col. 39 l. 65 - col. 40 l. 4 ("Note that the delay field is used to guarantee the execution of all instructions in the basic block governed by this branch in single cycle context free PE's. A small complexity is encountered when the PE's are pipelined. In this case, there exist data dependencies between the instructions from the basic block just executed, and the instructions from the basic block to be executed."). Yet, as noted above, that variant does "not change the fundamental context free nature of the processor elements of the present invention." '313 Patent col. 40 ll. 59-65; *see* '313 Patent col. 41 ll. 4-11 ("It is to be expressly understood, that any prior art type of processor element such as a micro-processor or other pipeline architecture could not be used under the teachings of the present invention, because such processors retain substantial state information of the program they are processing."); *cf.* '313 Patent col. 29 ll. 7-10 ("As the system contains multiple, *generally independent*, processor elements . . ., requests to the instruction cache are for a group of concurrently executable instructions.") (emphasis added). Plaintiff does not account for why, upon reading the claims in the light of the specification, one schooled in the art would not understand the processor elements to revert to their fully context free quality after execution of a pipelined instruction cycle.[7] In other words, while the processor elements may temporarily retain certain context information regarding dependencies between instructions, they are consistently described in the patent as context free.

Third, the specification describes a variant whereby "data is retained within the processor and made available (for example, through data chaining) to succeeding instructions which further manipulate the data." '313 Patent col. 18 ll. 9-11. Again, however, the retention of data is only temporary. "Ultimately, data is transmitted to the register file after some finite sequence of instructions completes; however, it is only the final data that is transmitted." '313 Patent col. 18 ll. 12-14. This "can be considered a substantially context free processor element." '313 Patent col. 18 ll. 16-17. Because of that temporary retention of data required for later execution of instructions, this variant is only *substantially* context free. But, in light of the fact that the data is ultimately transmitted to a register file "after some finite sequence of instructions completes," it is still context free.

[7]Claim 1 of the '313 Patent explicitly provides that the processor elements are "free of any context information." '313 Patent col. 46 ll. 1-3. Plaintiff relies on the doctrine of claim differentiation to contend that, therefore, when processor element is used in the '628 Patent, it can not be so limited. Claim 1 of the '313 Patent, however, is not a dependent claim to any claims in the '628 Patent. *See Enzo Biochem, Inc. v. Applera Corp.*, 599 F.3d 1325, 1342 (Fed. Cir. 2010) ("Under [the doctrine of claim differentiation], 'the presence of a dependent claim that adds a particular limitation gives rise to a presumption that the limitation in question is not present in the independent claim.'") (quoting *Phillips v. AWH Corp.*, 415 F.3d 1303, 1315 (Fed. Cir. 2005) (en banc)). Plaintiff identifies nothing in the shared specification supporting the conclusion that, because the processor elements of Claim 1 of the '313 Patent are explicitly defined as "free of *any* context information" (emphasis added), one schooled in the art would understand the '628 Patent to be claiming a processor element that was not context free at all.

That leaves the question of whether the term "processor element" must also be construed to require that all processor elements be identical. The specification provides that, "[i]n one embodiment of the invention, the processors are identical." '313 Patent col. 4 ll. 17-18. Plaintiff, in arguing that this limitation should not be read into the claim, confusingly cites the following line from the specification: "In the illustrated embodiment shown in FIG. 21, each processor element pipeline operates autonomously of the other processor elements in the system. Each processor element is *homogeneous* and is capable, by itself, of executing all computational and data memory accessing instructions." '313 Patent col. 41 ll. 14-19 (emphasis added by plaintiff). Figure 21, depicting this particular embodiment, is set forth below:

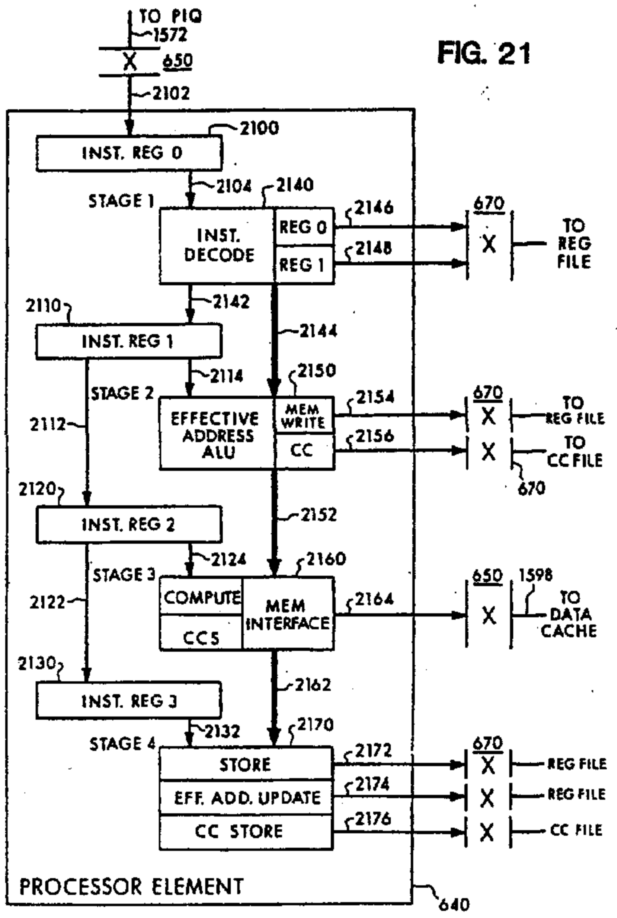


FIG. 21

To the extent the processor elements are homogeneous, however, that would seem to support the defendants' proposed construction.

Moreover, plaintiff emphasizes that the line in the specification referencing an embodiment where the processor elements are "homogeneous" appeared in the original specification. This, in plaintiff's view, supports the contention that, "[s]ince the very first application filed in 1985 . . ., the inventors explicitly recited this aspect as but *a feature* of the preferred embodiment." Docket No. 182 at 22 (emphasis in original). The record belies that claim. The original specification provided, in the Background of the Invention, that the processor elements were "identical." *See, e.g.*, Docket No. 182-4 at 1 (deleting, in the substitute specification, the word "identical" that appeared before "processor elements" in the Background of the Invention section of the specification). Furthermore, during the substitution process, the specification was changed from "are executed on a system containing a plurality of *identical* processor elements. These processor elements are characterized by the fact that they contain no execution state information from the execution of previous instructions, i.e., they are context free" to "are executed on a system containing a plurality of processor elements. In one embodiment of the invention, the processors are identical. The processor elements, in this illustrated embodiment, contain no execution state information from the execution of previous instructions, that is, they are context free." Docket No. 182-4 at 9 (emphasis added). Moreover, during prosecution of the '755 Patent, in a Petition Pursuant to 37 C.F.R. § 1.102(d) requesting Advancement of Examination, plaintiff represented that the "present invention further executes the basic blocks containing the added intelligence on a system using a plurality of identical processor elements . . . ." Docket No. 146-28 at 23. In other words, during the initial prosecution and in the original specification, the processor elements were defined as being identical.[8]

As plaintiff points out, the operative specification only refers to a particular embodiment of the invention. Review of Claim 1 of the '628 Patent, in light of the specification, however, supports a construction whereby the processor elements possess certain identical functional capacities. For instance, as discussed above, processor elements are able to interpret and execute "machine-level" instructions.[9] All of them share this identical quality. The fact that different "idiosyncracies," '313 Patent col. 6 l. 48, in processor element hardware may exist does not change this. The specification makes clear that the TOLL software can "handle" these potential differences in hardware to ensure that the processor elements all share the same functionality within the invention. *See* '313 Patent col. 6 ll. 47 - 51. What the processor elements must share is the ability, "under program control," to execute instructions. '313 Patent col. 18 l. 2; *see* '313 Patent col. 18 ll. 18-21 (noting that, in the "substantially context free" variant of the invention, "the TOLL software would be required to ensure that dependent instructions execute on the same processor element until such time as

---

[8]The '313 Patent, which contains the same specification as the '628 Patent, includes references to the processor elements being identical. *See, e.g.*, '313 Patent (Abstract) ("A computer processing system containing a plurality of identical processor elements each of which does not retain execution state information from prior operations."); '313 Patent col. 41 ll. 2-4 ("All processor elements . . . according to the illustrated embodiment are identical.").

[9]The term "instructions" will be construed below so as to clarify that it refers to "machine-level" instructions. *See infra* Section III.C.3.

data is ultimately transmitted to the context register file").

In sum, the Court construes "processor element" as a device that is capable of interpreting and executing instructions, a quality it shares with all other processor elements, and which does not retain context information after the execution of an instruction or a set of instructions.

2. Processor

Defendants contend that "processor" should be given the same construction as "processor element." Plaintiff argues that it has a distinct meaning, i.e., "[a] device that is capable of interpreting and executing one or more of arithmetic, logic, and branch instructions." Docket No. 149 at 14. The Court agrees that the terms are used synonymously in the claims. First, plaintiff's proposed construction appears to simply replace "instructions" with a more specific definition that makes clear that the processor interprets and executes "machine-level" instructions. The Court, when it construes "instruction" below, will clarify that the term refers to such "machine-level" instructions. Therefore, there is no need to do so here. And, to that extent, it does not present a true alternative to defendants' construction.

Furthermore, to the extent plaintiff is attempting simply to incorporate language that appears in the claims that specifies that the processor's ability to interpret and execute instructions includes certain types of machine-level instructions, such as arithmetic or logic instructions, the claims are clear on this point. *See* '628 Patent col. 48 ll. 5-9 (Claim 16) ("a processor configured to execute instructions, the instructions including: arithmetic or logical instructions that each produce an arithmetic or logical result . . . ."); *cf.* '628 Patent col. 47 ll. 61-63 (Claim 15) ("said branch unit has means for executing said conditional branch instructions concurrently with execution of arithmetic instructions by said processor *element*") (emphasis added).

Finally, plaintiff contends that its representations to the PTO during reexamination of the patents support a differentiation between "processor element" and "processor." Yet, it asserted to the PTO during the reexamination of the patents that the "'628 patent and claim 18 uses 'processor' as shorthand for the term 'processor element.'" Docket No. 149-8 at 24. It is also worthy of note that, in its opening claim construction brief in *BIAX v. Intel Corp.*, No. 02:05-CV-184, 2007 WL 677132 (E.D. Tex. March 1, 2007), plaintiff asserted that "the terms 'processor' and 'processor element' are used interchangeably in the patents." Docket No. 146-29 at 24 (citing support in the specification); *see also* Docket No. 146-31 (*Markman* Hearing Transcript in *BIAX v. Intel Corp.*) at 3-4 ("[I]f you look at when processor is used, it is not used a separate term, but rather [as] a shorthand term for processor element."). Because both parties essentially agree that a "processor" is capable of interpreting and executing instructions (which, as will be discussed below, include the types of instructions plaintiff seeks to have included in the construction of "processor"), I conclude that "processor" is used as shorthand for "processor element." I will, therefore, give it the same construction.

3. Instruction

Both parties agree that the definition of "instruction" should include a "machine language or assembly language construct that specifies an operation and identifies its operands." *See* Docket No. 144 at 26; Docket No. 149 at 21. Defendants, however, seek to have the following phrase added to the end of the definition: "as opposed to microoperations, microinstructions or microcode constructs." Docket No. 144 at 26. Plaintiff interprets "instruction" to consist of "machine or assembly language," and plaintiff has consistently represented that such machine-level instructions are different from micro-instructions. *See* Docket No. 149-8 at 28-31.

During the reexamination of the '628 Patent, plaintiff disavowed any construction of "instruction" that would include microoperations, microinstructions or microcode constructs. *See Arlington Industries, Inc. v. Bridgeport Fittings, Inc.*, 345 F.3d 1318, 1332 (Fed. Cir. 2003) ("Arguments made in a reexamination proceeding will constitute a disclaimer of claim scope if they are 'clear and unmistakable statements of disavowal.'") (quoting *Cordis Corp. v. Medtronic Ave, Inc.*, 339 F.3d 1352, 1358 (Fed. Cir. 2003)). For example, plaintiff argued in a Response to Office Action that "there are fundamental differences between microinstructions and machine instructions . . . . Because of these fundamental differences, a person of ordinary skill in the art at the time of invention would not have been motivated to apply the teachings from the microcode domain disclosed by Rosocha to the machine code, or instruction-level, domain." Docket No. 149-8 at 30; *see* Docket No. 149-8 at 70 (". . . microoperations and instructions are not equivalent, and the proposed change from microoperations, as disclosed by Rosocha, to instructions, as recited in the present patent, is not an obvious modification."). Furthermore, in an expert report filed by plaintiff as an appendix to its Response to Office Action, Professor William J. Dally outlined "several fundamental differences between microinstructions and machine instructions." Docket No. 149-10 at 7-9; *see*

Docket No. 149-10 at 19 ("[M]icrocode instructions are quite different than machine-language instructions and similarly the conditions on which microcode branches are based are quite different than the condition code values generated and tested by machine-language instructions. . . . The microcode prior art references . . . fail to disclose any of the machine instructions (arithmetic or logical, condition setting, or conditional branch) . . . ."); Docket No. 149-10 at 39 ("[B]oth Maccianti and Rosocha describe microcoded controllers that deal with micro-operations and micro-instructions rather than machine-language instructions. . . .").

During the reexamination process, PTO examiners interviewed representatives of the plaintiffs who "discussed that there is a significant divide between Rosocha's microinstructions/microoperations versus machine-level instructions." Docket No. 146-11 at 5 ("It was also noted that the specification of the '628 patent clearly distinguishes microcode from the claimed instructions, as illustrated by col. 5, lines 10-12."). Plaintiff filed a Declaration Under 37 C.F.R. § 1.132[10] by a consulting engineer as part of the reexamination of the '313 Patent. *See* Docket No. 146-13. Plaintiff's consultant "agree[d] with Prof. Dally's explanation of the substantive differences between Machine instructions and Microcode" and "also agree[d] with the Examiner[']s statement that 'instructions and microcode are not equivalents.'" Docket No. 146-13 at 4; *see* Docket No. 146-13 at 5 ("It is clear that the term 'instruction' in the claims of the '313 patent refers to a machine-language instruction. Throughout the specification of the '313

[10] 37 C.F.R. § 1.132 ("When any claim of an application or a patent under reexamination is rejected or objected to, any evidence submitted to traverse the rejection or objection on a basis not otherwise provided for must be by way of an oath or declaration under this section.").

patent, the term instruction is used to refer to a machine-language instruction (*See*, e.g., '313 at 6: 39-60) and consistently address machine-level registers, such as general purpose registers. Rosocha fails to disclose any of the machine instructions (arithmetic or logical, condition setting, or conditional branch) required by the asserted claims.").

Plaintiff argued at the *Markman* hearing that these statements distinguishing machine instructions from micro-instructions focused on distinguishing only a certain type of micro-instructions described in the Rosocha prior art. While the context in which the statements were made was distinguishing that prior art, it was not limited to only that purpose. Plaintiff filed an expert report by Professor William J. Dally as an appendix to a Response to Office action during reexamination, which also distinguished Maccianti on the grounds that it "describe[d] microcoded controllers that deal with micro-operations and micro-instructions rather than machine-level instructions. . . ." Docket No. 149-10 at 39. Plaintiff's expert asserted that "[o]ther prior art references use micro-condition code registers as part of a microcoded control unit," and that such "microcode instructions are quite different than machine-language instructions" as described in a section of the report not limited to discussing the Rosocha reference. Docket No. 149-10 at 19. Furthermore, in distinguishing "instructions" as described in the specification and "microcode," it was noted during a reexamination interview with the PTO "that the specification of the '628 patent clearly distinguishes microcode from the claimed instructions, as illustrated by col. 5, lines 10-12." Docket No. 146-11 at 5. There is no indication that "microcode," when used in the specification, is limited to the type disclosed in Rosocha.

In sum, microoperations, microinstructions, and microcode constructs are not included in "instruction" as used in the patents. Contrary to plaintiff's argument, I conclude that making this clear in the construction of the term "instruction" is more accurate. Therefore, I adopt defendants' proposed construction.

4. Conditional branch instruction

Plaintiff argues that this is a commonly-used term of art that needs no construction. If the Court does construe the term, plaintiff proposes: "[A]n instruction that determines a target instruction to be executed based on a condition code value." Docket No. 149 at 24. Defendants seek the following construction: "A branch instruction that already contains the full target address of the instruction to branch to if a condition is met." Docket No. 144 at 40. The parties agree that a "branch instruction" or "branch" is "an instruction that determines a target instruction to be executed." *See* Docket No. 149 at 25; *see* Docket No. 144-1 at 1.

Plaintiff's proposal is essentially a restatement of the claim language itself. Claim 1 of the '628 Patent provides for "a branch execution unit configured to execute conditional branch instructions that each determine a target instruction for execution based on analysis of a condition code value from one of said condition code registers." '628 Patent col. 46 ll. 9-12. Claim 1 also provides for a "condition code access unit," a term that will be construed below, which is

> configured to act in response to condition-selecting instructions, at least one of said condition-selecting instructions being one of either said condition-setting instructions or said *conditional branch instructions*, said condition-selecting instructions for selecting from said condition code register file a condition code register for at least one of:
>
> storing into said selected condition code register a condition code

> value produced by one of said condition-setting instructions, and
>
> fetching from said selected condition code register a condition code value for analysis by one of said *conditional branch instructions*;
>
> said selecting being by direct addressing on a condition code address field of the condition-selecting instruction.

'628 Patent col. 46 ll. 13-28 (emphasis added). The claim language clearly indicates that a "conditional branch instruction" analyzes condition code values. The purpose of doing so is to determine a target instruction to be executed. Defendants do not dispute this function of the conditional branch instruction. Rather, they contend that the conditional branch instruction must contain the full target address of the instruction the program will branch to based on the analysis of the condition code value. This argument, however, is based on a reading of the original specification. As stated above, the Court will interpret the claim language in light of the current operative specification.

Plaintiff points out that, in the substitute specification, the target address need *not reside in the conditional branch instruction.* Rather, it "can be (a) taken directly from the instruction, (b) an offset from the current contents of the next instruction fetch register, or (c) an offset of a general purpose register of the context register file." '313 Patent col. 36 ll. 35-41. Defendants do not dispute that the substitute specification so provides, but rather urge the Court to "consult the original specification to avoid the consideration of new matter." Docket No. 183 at 21.[11] In the absence of any argument

[11]Defendants argue that plaintiff engaged in inequitable conduct in claiming to the PTO that changes to the specification did not introduce new matter. While the Court focuses on the current specification for purposes of construing the claim term, it is potentially relevant to the new matter issue that plaintiff does not dispute defendants'

that the '628 Patent as it currently reads requires that "conditional branch instruction" contain the full target address, the Court will not include that requirement in the construction. All that remains, essentially, is the language of the claims themselves, which clearly state that conditional branch instructions "determine a target instruction for execution based on analysis of a condition code value from one of said condition code registers." '628 Patent col. 46 ll. 10-12. There is no additional construction needed of the term.

5. Condition code access unit

Plaintiff proposes "[a] unit that accesses a condition code register based on an analysis of an instruction field" as the construction for "condition code access unit." Docket No. 149 at 26. Defendants propose: "[a] unit that determines whether a condition code value must be fetched or will be delivered based on an analysis of an instruction field." Docket No. 144 at 44. The parties dispute the scope and descriptions of the condition code access unit's functionality. Plaintiff is correct that the claims themselves specify what the condition code access unit does within the invention. A review of the claims, however, shows that plaintiff's proposed definition is incomplete. Specifically, Claim 1 of the '628 Patent makes clear that a "condition code access unit" is

configured to act in response to condition-selecting instructions, at least one

---

interpretation of the original specification, but rather focuses exclusively on the language of the substitute specification. *Cf.* '628 Patent (Orig. Spec.) [Docket No. 146-1 at 41] col. 42 ll. 6-9 ("[T]he target address is fully contained within the branch, i.e. all branch addresses are known at program preparation time in the TOLL output and, as a result, are directed to absolute addresses only."). That, however, is an issue for another day.

> of said condition-selecting instructions being one of either said condition-setting instructions or said conditional branch instructions, said condition-selecting instructions for selecting from said condition code register file a condition code register for at least one of:
>
> > storing into said selected condition code register a condition code value produced by one of said condition-setting instructions, and
> >
> > fetching from said selected condition code register a condition code value for analysis by one of said conditional branch instructions;
> >
> > said selecting being by direct addressing on a condition code address field of the condition-selecting instruction.

'628 Patent col. 46 ll. 13-28. Claim 13 of the '628 Patent, which is dependent on Claim 1, provides that

> said condition code access unit selects from said condition code register file a condition code register for both:
>
> > storing into said selected condition code register a condition code value produced by one of said condition-setting instructions, and
> >
> > fetching from said selected condition code register a condition code value for analysis by one of said conditional branch instructions.

'628 Patent col. 47 ll. 47-54. These claims describe a unit that contains qualities found in both proposed definitions. It seems clear that the condition code access unit does "determine[] whether a condition code value must be fetched or will be delivered," as defendants' definition provides. But the claims also specify that the unit makes this determination by accessing a condition code register within the condition code register file.

These specific qualities claimed in the patent are supported by the description of the condition code access unit found in the specification. For example, the specification provides that the "condition code access unit . . . determines whether or not the condition codes must be fetched by the instruction, or whether the instruction that determines the branch condition delivers them." '313 Patent col. 37 ll. 57-63.

The specification is clear regarding the unit's ability to access condition code storage locations. *See, e.g.*, '313 Patent col. 38 l. 58 - col. 39 l. 8 ("If a FETCH is requested, the [condition code access] unit 1920 accesses the register file-PE network 670 (see FIG. 6) to access the condition code storage 2000 which is shown in FIG. 20. . . . A set of registers . . . are provided for storing condition codes for procedural level y. Hence, the condition code storage 2000 is accessed and addressed by the unit 1920 to retrieve, pursuant to a FETCH request, the necessary condition code. The actual condition code and an indication that the condition code is received by the unit 1920 is delivered over lines 1922 to the evaluation unit 1930. The OPCODE field, delivered to the evaluation unit 1930, in conjunction with the received condition code, functions to deliver a branch taken signal over line 1932 to the next instruction interface 1950."). These components are depicted in Figures 19 and 20:

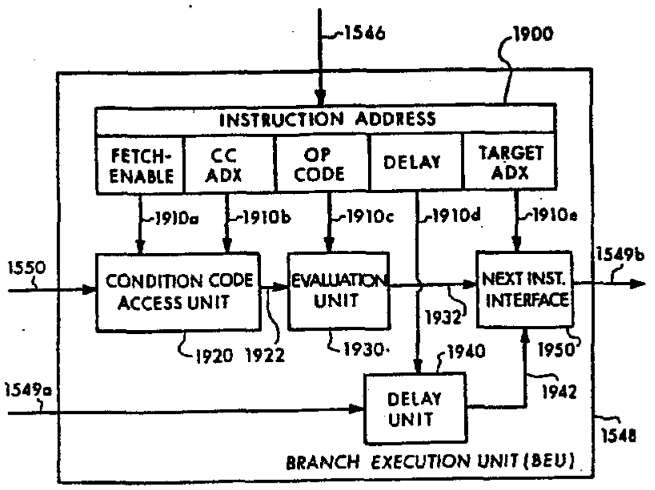


FIG. 19

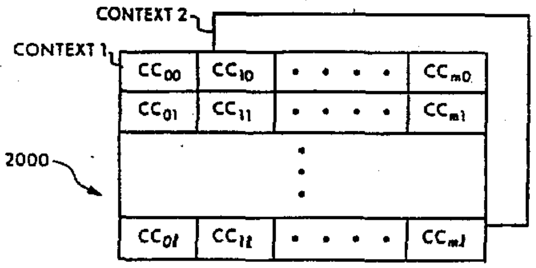


FIG. 20

The Court shares the view of the court in *BIAX v. Sun Microsystems, Inc.* that the "plain language of the claim . . . coupled with the description of the unit in the specifications, reveals that [the] proposed 'active' construction is closer to correct." 2008 WL 2811304, at *9. But, with that said, the Court finds that the unit can access a condition code register or some other form of condition code storage location and that it responds to condition-selecting instructions in particular. Therefore, in light of the claim language, as supported by the specification, the Court concludes that the appropriate construction for "condition code access unit" is a unit that accesses a condition code storage location and, based upon an analysis of condition-selecting instructions, determines whether a condition code value must be fetched or will be delivered.

6. Condition code register / general purpose register

The parties agree that the definition of "condition code register" includes a "special purpose register for storing a condition code" and that the definition of "general purpose register" includes a "memory location for holding a condition value." Defendants, however, propose adding the following to each: "which is shared by all processor elements." Docket No. 144 at 20.

Claim 1 of the '628 Patent describes "a general purpose register file comprising at least two general purpose registers" and "a condition code register file distinct from said general purpose register file, having a plurality of addressable condition code registers, each condition code register for representing a condition code value as a small number of bits summarizing the execution or result of a previously-executed instruction." '628 Patent col. 45 l. 64 - col. 46 l. 4. Plaintiff argues that, because Claim 2 specifically provides that "said condition code register file [is] shared by said processor elements," the limitation should not be read into the term as used in Claim 1. Moreover, plaintiff argues that defendants' proposal incorrectly assumes multiple processor elements whereas Claim 1 uses the term "a processor element."

In regard to the provision of multiple processor elements, it is true that Claim 1 describes "*a* processor element." Defendants are correct that the "words 'a' or 'an' in a patent claim carry the meaning of 'one or more.'" *TiVo, Inc. v. EchoStar Communications Corp.*, 516 F.3d 1290, 1303 (Fed. Cir. 2008). "That is particularly true when" – like here – "those words are used in combination with the open-ended antecedent 'comprising.'" *Id.*; *see Baldwin Graphics Sys., Inc. v. Siebert, Inc.*, 512 F.3d 1338, 1342 (Fed. Cir. 2008) (stating that the Federal Circuit "has repeatedly emphasized that an indefinite article 'a' or 'an' in patent parlance carries the meaning of 'one or more' in open-ended claims containing the transitional phrase 'comprising.'") (citation omitted). Although "the question whether 'a' or 'an' is treated as singular or plural depends heavily on the context of its use," *TiVo, Inc.*, 516 F.3d at 1303 ("The general rule does not apply when the context clearly evidences that the usage is limited to the singular."), "[t]hat 'a' or 'an' can mean 'one or more' is best described as a rule, rather than merely as a presumption or even a convention," exceptions to which are found in only "extremely limited" circumstances. *Baldwin Graphics*, 512 F.3d at 1342.

Here, Claim 1 provides for multiple "condition code registers"; but use of "a processor element" in the singular in Claim 1 potentially identifies a single processor element. *See TiVo, Inc.*, 516 F.3d at 1303 (noting that, in *Baldwin Graphics*, "the claims and the written description could be read to encompass either a singular or plural interpretation of 'a' or 'an'"). Furthermore, the patent makes clear that, upon introduction of "at least one additional processor element," the "condition code register file [is] *shared* by said processor elements." '628 Patent col. 46 ll. 37-39 (Claim 2) (emphasis added); *see Comark Commc'ns, Inc. v. Harris Corp.*, 156 F.3d 1182, 1187 (Fed.Cir.1998) ("[T]he doctrine of claim differentiation . . . create[s] a presumption that each claim in a patent has a different scope.").

But it is unnecessary to determine whether "a processor element" is *limited* to the singular, as it is, at a minimum, clear that the language of Claim 1 provides for the possibility of a single processor element. In the event there is only one processor element, that processor element is capable of accessing any of the condition code registers for storing condition code values. In that event, the register is not shared by all processor elements since there would just be one. Thus, while defendants are correct that any processor element is able to access any condition code register, their argument to include within the definition of "condition code register" a reference to multiple processor elements sharing the condition code register is a more limited definition than Claim 1 requires.[12] Because that feature is separately provided for in the claim language, it need not be included within the definition of the component of the system called a "condition code register."[13] Similarly, Claim 1 does not clearly provide

[12]Similarly, Claim 1 does not require *multiple* processor elements that would *share* the "general purpose register file."

[13]In the '313 Patent, this aspect of the invention is specifically provided for in Claim 1. *See* '313 Patent col. 46 ll. 14-17 ("each of said processor elements being capable of accessing said plurality of registers and condition code storages in a program's context during the processing of the program's instruction").

for multiple processor elements that would share the "general purpose register file."

7. Addressable / condition code address field / field directly addressing one of said condition code registers

The parties largely agree on the appropriate definitions of these terms. Defendants, however, seek to include a specific reference to "address" in each construction. *See* Docket No. 144 at 43 ("Defendants would accept BIAX's constructions if a reference to 'address' is added to each . . . ."). For example, for "addressable," plaintiff proposes "[c]apable of being specifically identified," while defendants propose adding "by an address" to the end of the construction. Docket No. 149 at 32; Docket No. 144 at 43.

The Court finds defendants' request to add "by an address" to be unnecessary. If the parties are seeking construction of the term "addressable," for example, it is not clear – absent some argument that "address" has a different meaning when it is used in the claim terms than when it is used in defendants' proposed definitions – what is clarified or added by introducing the term "address" to the definitions. As it stands, plaintiff's proposal includes a definition of the relevant "address" terms, e.g., in the case of "addressable," that it is capable of being identified or, in the example of "condition code address field," a "portion of the instruction used *to identify* a specific condition code register." In other words, the address portion of each disputed claim term contains a definition of "address," i.e., to identify a location. An address field is "used to identify" and something addressable is "capable of being specifically identified." About this, the parties agree. Therefore, the Court rejects the addition of "address" to the construction of each term.

There also is a dispute over whether "condition code address field" and "field directly addressing one of said condition code registers" are synonymous. *See* Docket No. 149 at 35; Docket No. 144 at 43 n.9. Plaintiff argues that when it "intended to limit the 'condition code address field' to one 'directly addressing one of said condition code registers,' it specifically drafted the claims to include such a limitation." Docket No. 149 at 35. It is true that, after "condition code address field," certain claims specify that it can directly address one of the condition code registers. This, however, appears to be less a limitation than a description of what the condition code address field does. Plaintiff does not suggest what else it might do.

The varied wording used to describe the direct addressing and the condition code address field further supports their shared meaning. For example, Claim 1 describes "selecting being by direct addressing on a condition code address field," '628 Patent col. 46 ll. 26-28, and Claim 8 describes that "condition-setting and conditional branch instructions each comprise a condition code address field" and an association of "values of said condition code address field to condition code registers." '628 Patent col. 47 ll. 11-12, 14-16. The "condition code address field of each said conditional branch instruction . . . directly addresses a condition code register." '628 Patent col. 47 ll. 31-34 (Claim 11). Claims 10 and 16 provide for "a field directly addressing one of said condition code registers" as well as "a condition code address field for directly addressing one of said condition code registers," '628 Patent col. 47 ll. 24-27, col. 48 ll. 20-22, while Claim 23 describes "a condition code address field directly addressing one of said condition code registers." '628 Patent col. 49 ll. 6-8. These examples demonstrate that a "condition code address field" is a "field directly addressing one of said condition code registers." While it is true that the latter phrase is sometimes combined with the former, the Court believes that is done to define "condition code address field" rather than to introduce a new and distinct limitation. Therefore, the terms "condition code address field" and "field directly addressing one of said condition code registers" mean the same thing.

D. '313 Patent

The parties' dispute over claim construction of the '313 Patent centers on the following claim terms: first circuit, second circuit, condition storage, address for storing, address input, address selection circuit, condition storage address, and coupled.[14]

1. First circuit

Plaintiff argues that this term need not be construed, but, if it is to be construed, proposes "a circuit that receives opcodes and generates a set of at least one condition value." Docket No. 149 at 14. Defendants seek to have the Court require that a "first circuit" be a "context-free unit" and that "[a]ll such units are identical." Docket No. 144 at 38.

The term "first circuit" appears in the following context: "a first circuit coupled to said opcode storage, said first circuit configured to receive opcodes of a first type of instruction and addresses for storing sets of at least one condition value, and to generate for each said opcode of said first type a set of at least one condition value associated with one of the addresses." '313 Patent col. 46 ll. 55-60. The parties agree that an "opcode" is "the portion of an instruction that specifies an operation" and that a

[14]The term "instruction" appears in both patents, and the parties agree that it should be given the same construction in both.

"condition code value" is "the summary of an executed instruction represented as a plurality of bits." Docket No. 144-1 at 2.[15] Furthermore, the parties agree that "circuit" means "an assemblage of electronic elements." Docket No. 144-1 at 1. The only apparent disputes are whether the "first circuit" is context free and whether "[a]ll such units are identical."

Defendants argue that the "first circuit" is "functionally equivalent to the processor element" and, therefore, the "patentee's requirement of context-free and identical is therefore applicable to the first circuit as well." Docket No. 144 at 38. Defendants, however, have not persuasively explained why Claim 2 of the '313 Patent would use "first circuit" instead of "processor element" if they are necessarily and always the same, in light of "processor element" appearing in Claim 1.[16] As for what a "first circuit" does, the claim describes what it is configured to do in relation to a "first type of instruction." Because the parties otherwise agree on what "circuit" means, the Court concludes that a "first circuit" is "an assemblage of electronic elements" that receives an opcode of a first type of instruction and generates a set of at least one condition value.

---

[15]The parties agree that "condition code value" and "condition value" are synonymous, both meaning "the summary of an executed instruction represented as a plurality of bits." Docket No. 144-1 at 1.

[16]Moreover, defendants' citation to statements made by one of plaintiff's experts during reexamination does not reflect a clear assertion by plaintiff that a "first circuit" and a "processor element" are synonymous. *See, e.g.*, Docket No. 144 at 39 ("BIAX's expert . . . stated in his declaration that '[i]n the "first circuit" of the . . . '313 patent, a Processor Element (PE) stores sets of condition values at locations of the PE-Context Network per the claim."). While the two terms may overlap, the claim language does not require they be construed as coterminous.

2. Second circuit

Plaintiff does not believe this term needs to be construed. If the Court is to construe it, however, plaintiff requests the following definition: "[A] circuit that receives opcodes and generates a set of at least one output value." Docket No. 149 at 14. Defendants propose the following: "A unit distinct from the first circuit." Docket No. 144 at 28. For the reasons outlined above, the Court concludes that the "second circuit" is "an assemblage of electronic components" configured to engage in the tasks described in Claim 2 of the '313 Patent. No further description of its capabilities is required.

Furthermore, defendants are correct that the claims describe the "second circuit" as being distinct from the "first." All plaintiff argues in response is that "neither the claims nor the specification require that the 'second circuit' be 'distinct from the first circuit,'" Docket No. 149 at 17, but plaintiff fails to address what, if any, effect the use of "second" has on the appropriate construction and does not explain how the "second circuit" functions as described if it were the same unit as the "first circuit." On its face, the use of "first" and "second" distinguishes the two "assemblages of electronic elements." Furthermore, the claim language describes distinct functions. The "first circuit" receives "an opcode of a *first type* of instruction and . . . generate[s] a set of at least one condition value." '313 Patent col. 46 ll. 32-34. Those condition values are stored in condition storage. The "second circuit" receives "an opcode of a *second type* of instruction and . . . generate[s] a set of at least one output value, said output value for each second said type of instruction *depending on a particular one of said stored sets of condition values.*" '313 Patent col. 46 ll. 39-43. These are distinct circuits providing different functions in the claimed invention. *Cf.* Docket No. 146-13 at 11 (where plaintiff's consulting engineer represented during the reexamination that a particular reference did not "disclose first and second circuits that are *both* coupled to 'said opcode storage'") (emphasis added). The Court, therefore, construes the "second circuit" as a distinct unit from the "first circuit" which receives opcodes of a second type of instruction and generates a set of at least one output value.

3. Condition storage

For the term "condition storage," plaintiff proposes "[a] memory location for holding a condition value." Docket No. 149 at 27. Defendants request the following construction: "[s]pecial purpose memory locations for holding condition values, which are shared by all processor elements." Docket No. 144 at 20. Defendants' addition of "special purpose" apparently is designed to make clear that the storage location is not a general purpose location insofar as it is used to store condition values. However, both parties' proposed constructions already include that special purpose within the definition. What is clear from the patents is that "condition storage" appears to refer to any memory location designed to store condition code values and is, therefore, potentially broader than other "registers," as described in Claim 1 of the '313 Patent. *See* '313 Patent col. 46 ll. 14-24 (". . . each of said processor elements being capable of accessing said plurality of registers and condition code storages in a program's context during the processing of the program's instruction, a plurality of memory locations . . ., and second means . . . for connecting each of said processor elements with any one of said plurality of memory locations, each said processor element being capable of accessing said memory locations during said processing of each said instruction.).

Moreover, the claim language clearly provides that the "condition code storages" are accessible by "each of said processor elements." Therefore, it is unnecessary to import that quality into the definition of the term. Even if the Court were to adopt plaintiff's construction, the claim would still provide for "processor elements . . . capable of accessing a plurality of [memory locations for holding a condition value]." But, to the extent there is any dispute over whether, despite the foregoing, condition code storage locations are shared by all processor elements, the Court rules that they are.

4. Address for storing / address input / address selection circuit / condition storage address

For the same reasons stated in Section III.C.7., *supra*, the Court adopts the agreed-upon constructions proposed by the parties without inclusion of the word "address" within those definitions, as suggested by defendants.

5. Coupled

The term "coupled" appears numerous times throughout the '313 Patent claims. For example, Claim 2 provides for "a first circuit coupled to said opcode storage, said first circuit configured to receive an opcode of a first type of instruction and to generate a set of at least one condition value . . ." '313 Patent col. 46 ll. 31-34. In other words, as will be discussed more fully below, the claim describes a "first circuit" being connected to "opcode storage" in such a manner so that it can "receive an opcode . . . ." Consistent with this plain reading of the claim language, plaintiff defines "coupled" as "[c]onnected, directly or indirectly," and defendants propose "[t]otally coupled" as the definition of "coupled." Docket No. 149 at 36; Docket No. 144 at 45.

As an initial note, it does little to elucidate the meaning of "coupled" to include the word "coupled" in the proposed definition. Moreover, it is not clear, on its face, what defendants mean by "totally." One possibility is that by "totally coupled" defendants are seeking a "direct" coupling, i.e., a connection with no intermediary steps or components. When defendants use "totally" in this case, they mean that "the hardware components – such as the processor elements and the registers –" constitute "shared resources," Docket No. 144 at 45, i.e., they are able to access each other "totally." Because defendants have not argued that "coupled" means "to directly connect," instead introducing and defining "totally" to import the notion of sharing resources more generally, the Court finds that they take no issue with the fact that the word "coupled" means "connected, directly or indirectly." Therefore, the defendants' proposal can be read as "totally connected, directly or indirectly."

The question then becomes whether the notion of "all hardware components" being able to access each other "totally" should be added to "connected, directly or indirectly." The specification does use the term "totally coupled" in the way proposed by defendants. *See* '313 Patent col. 3 ll. 24-28 ("The present invention, in one aspect, teaches a method of hardware and software based upon totally coupling the hardware resources thereby significantly simplifying the communication problems inherent in systems that perform multiprocessing."); '313 Patent col. 9 ll. 39-45 ("The present invention can execute dependent instructions on different processor elements, in one illustrated embodiment, because of the context free nature of the processor elements and the total coupling of the processor elements to the shared resources, such as the register files, as will also be described below."); '313 Patent col. 16 ll. 18-24 ("Under the teachings of the present invention, each processor element (PE) is 'totally coupled' to the necessary registers in the register file 660 during any particular instruction firing time (IFT) and, therefore, there is no need to move the data out of the register file for delivery to another resource; e.g. in another processor's register as in some conventional approaches."); '313 Patent col. 17 ll. 20-23 (". . . the memory, the logical resource drivers, the processor elements, and the context shared resources are totally coupled through their respective networks in a pure, non-blocking fashion.").

Yet, the word coupled, when used in the claims of the '313 Patent, specifically indicates which components are coupled, or connected, to each other, i.e., which are sharing information.[17] Therefore, there is no need to import the existence of other potentially connected resources into the understanding of the word "coupled" as used in the specific claims at issue. That "coupled" means "connected, directly or indirectly" in the claims and that certain shared resources can be described as "totally coupled" in the specification are perfectly consistent. When the '313 Patent claims describe specific components being "coupled," they are describing connections between them. Likewise, when total couplings are described in the specification, they refer to the fact that connections are made between various components of the system, either directly or indirectly, so that all the described hardware components are able to share resources through those connections. In sum, when "coupled" is used in the '313 Patent claims, it means "connected, directly or indirectly." Adding "totally" would not alter that definition

[17]*Cf. Neomagic Corp. v. Trident Microsystems, Inc.*, No. Civ.A. 98-699-KAJ, 2003 WL 21076779, at *6 (D. Del. May 9, 2003) ("The ordinary meaning of the terms of the claims of the patent at issue are controlling. Therefore, as already elaborated in detail in previous decisions of this court and the Federal Circuit, the term 'coupled to' means an electrical communication between the two specified components, here being the memory portion and the logic portion of the integrated circuit.").

and has the potential to cause confusion.

IV. CONCLUSION

Accordingly, the patent claim terms presently at issue shall be construed as follows:

A. **’628 Patent Claim Terms**

• Processor Element: a device that is capable of interpreting and executing instructions, a quality it shares with all other processor elements, and which does not retain context information after the execution of an instruction or a set of instructions

• Processor: same meaning as term “processor element”

• Instruction: machine language or assembly language construct that specifies an operation and identifies its operands, as opposed to microoperations, microinstructions or microcode constructs

• Conditional Branch Instruction: an instruction that determines a target instruction for execution based on analysis of a condition code value from a condition code register

• Condition Code Access Unit: a unit that accesses a condition code storage location and, based upon an analysis of condition-selecting instructions, determines whether a condition code value must be fetched or will be delivered

• Condition Code Register: a special purpose register for storing a condition code

• General Purpose Register: a memory location for holding a condition value

• “Address” Terms: agreed-upon constructions proposed by the parties without inclusion of the word “address” within those definitions

• Field Directly Addressing One of Said Condition Code Registers: synonymous with condition code address field

B. ’313 Patent Claim Terms

• First Circuit: an assemblage of electronic elements that receives an opcode of a first type of instruction and generates a set of at least one condition value

• Second Circuit: a distinct unit from the “first circuit” which receives opcodes of a second type of instruction and generates a set of at least one output value

• Condition Storage: a memory location designed to store condition code values

• “Address” Terms: agreed-upon constructions proposed by the parties without inclusion of the word “address” within those definitions

• Coupled: connected, directly or indirectly

It is so **ORDERED.**

DATED June 21, 2010.

BY THE COURT:

[illegible]

PHILIP A. BRIMMER
United States District Judge