United States District Court,
D. Massachusetts.

DATA GENERAL CORPORATION,
Plaintiff.
v.
INTERNATIONAL BUSINESS MACHINES CORPORATION,
Defendant.

No. Civ.A. 94-12213-NMG

April 27, 2000.

Owner of patents for intermediate level computer language and method to be used by computer to resolve unresolved pointers sued competitor for infringement. The District Court, Gorton, J., construed disputed claims.

Claims construed.

4,455,603, 4,575,797, 4,761,785. Cited.

Douglas E. Whitney, Morris, Nichols, Arsht & Tunnell, Wilmington, DE, for Plaintiff.

Donald J. Rosenberg, Vice President and Asst. Gen. Counsel, Jennifer Daniels, Sr. Counsel, I.B.M. Corp., White Plains, NY, Evan R. Chesler, Thomas G. Rafferty and Richard J. Stark, Cravath, Swaine & Moore, New York, NY, Christopher A. Hughes, Morgan & Finnegan, L.L.P. New York, NY, Stephen D. Poss P.C., Goodwin Proctor & Hoar, Boston, MA, John O. Mirick, Mirick O'Connell DeMalle & Lougee, Worcester, MA, Morris Waisbrot and William F. Haigney; Davis Weber & Edwards, New York, NY, for Defendant IBM.

MEMORANDUM OF DECISION

GORTON, **District Judge.**

On November 7, 1994, Plaintiff, Data General Corporation ("Data General"), brought this action against Defendant International Business Machines Corporation ("IBM") alleging infringement of U.S.Patent No. 4,455,603 ("the '603 Patent") and U.S.Patent No. 4,575,797 ("the '797 Patent"). IBM timely answered and filed a counterclaim against Data General alleging infringement of U.S.Patent 4,761,785 ("the '785 Patent"). On September 23, 1999, this Court entered a Memorandum and Order setting forth its claim construction of the contested terms of the '785 patent. Pending before this Court is the parties' dispute over the proper construction of several terms found in the asserted claims of the '603 and '797 Patents.

On January 25, 1999, this Court conducted a four-day hearing ("the *Markman* hearing") to construe the disputed language pursuant to Markman v. Westview Instruments, Inc., 52 F.3d 967 (Fed.Cir.1995) (en banc). Upon close examination of the relevant evidence, this Court recognized that it lacked an adequate foundation and knowledge of the relevant art with respect to the technology underlying the three patents sufficient to draw definitive conclusions. Accordingly, it directed the parties to agree upon the selection of a technical advisor sufficiently conversant with and able to teach the subject technology of the patents at issue in this case for consultation with the Court. As a result of that process, this Court appointed Douglas W. Clark of Princeton, New Jersey to act as the Court's technical advisor in this action, consistent with the terms contained in his Affidavit of Engagement.

In November 1999, Data General merged into EMC Corporation ("EMC"), having previously assigned all right, title and interest to its patents to DG Patent Holdings, LLC ("DG"). DG was substituted for Data General as the plaintiff in this action and EMC was substituted for Data General as counterclaim defendant.

This Memorandum sets forth the claim construction of the disputed terms in the asserted claims of the '603 and '797 Patents.

I. Background

A. The '603 Patent

The '603 invention is a method performed by a computer to resolve unresolved pointers. A "pointer" is a data item used in a computer system to "point" to a location in a computer's memory, where information to be used is stored. A "resolved pointer" contains the target address, whereas an "unresolved pointer" does not, but instead contains data from which the address may be determined. The claimed method is one aspect of a concept known as dynamic linking, which enables a running program to access information "on the fly," for programming and other efficiencies.

Claim 10, the claim at issue in the '603 Patent, reads as follows:

10. In a digital computer system including

memory means for storing and providing data items in response to memory commands, each said memory command including an address specifying a location in said *memory means,* and

processor means connected to said *memory means* for providing said memory commands and providing and receiving said data items in response to sequences of instructions of said data items executed by said *processor means,* and wherein

said data items include *ordinary unresolved pointers,* each one of said *ordinary unresolved pointers* representing a *represented address of said addresses* and containing said data items from which said represented addresses may be derived and

said processor includes *call-return operation execution means* responsive to operation of said processor means for causing said processor means to commence and terminate an execution of any said sequence,

the method of resolving said *ordinary unresolved pointers* into said represented addresses comprising the steps of:

(1) receiving any said *ordinary unresolved pointer* in said *processor means;*

(2) providing said received *ordinary unresolved pointer* to said *call-return operation execution means;*

(3) causing said *call-return operation execution means* to commence execution of a pointer resolution sequence of said sequences, said pointer resolution sequence serving to derive said represented address for said received *ordinary unresolved pointer* using said contained data items in said received *ordinary unresolved pointer* and to return said represented address; and

(4) receiving said represented address from said *call-return operation execution means*.

'603 Patent, col. 7, ll. 4-42. (emphasis on disputed terms).

B. The '797 Patent

The '797 Patent discloses, among other things, an intermediate level computer language, into which commands of a higher level language, such as FORTRAN or COBOL which are understandable to a human computer programmer, are translated. The intermediate language commands are themselves interpreted by microcode routines. Claim 1 of the '797 Patent reads as follows:

A digital computer system comprising:

(1) *memory means* for storing and providing items of data, said items of data including *instructions,* each one of said *instructions* containing an operation code of a plurality of operation codes, said operation codes belonging to a plurality of functionally different operation code sets, said operation codes in a given one of said operation code sets being definable solely with reference to said given operation code set, and said instructions having *common formats;* and

(2) *processor means* connected to said *memory means* for receiving said items including said *instructions* and responding to each said received *instruction* by performing said operation defined for said operation code in said received instruction in said operation code set to which said operation code in said received *instruction* belongs.

'797 Patent, col. 4, ll. 27-44 (emphasis on disputed terms).

II. Analysis

A. Legal Principles of Claim Construction

[1] The resolution of a patent infringement claim requires a two-step analysis. First, the court must construe the asserted claims to determine their meaning and scope. *See* Texas Instruments Inc. v. Cypress Semiconductor Corp., 90 F.3d 1558, 1563 (Fed.Cir.1996). Only then may the trier of fact determine whether the properly interpreted claims encompass the accused structure. *Id*.

[2] [3] [4] [5] Claim construction is a question of law for the Court. *See* Markman, 52 F.3d at 979. The objective of claim construction is to ascertain the meaning that a person of ordinary skill in the art would give to the terms in dispute at the time of the application for the patent. Wiener v. NEC Electronics, Inc.,

102 F.3d 534, 539 (Fed.Cir.1996) (abrogated on other grounds); *see* Haynes International, Inc. v. Jessop Steel Co., 8 F.3d 1573, 1578 n. 4 (Fed.Cir.1993). To ascertain the meaning of claim language, the Court looks initially to the three sources of intrinsic evidence of record: 1) the claims themselves, 2) the specification, to the extent that it defines terms in a manner inconsistent with their ordinary meaning, and 3) the prosecution history of the patent, to the extent that it contains the patentee's express representations with respect to the scope of the claims. Vitronics Corp. v. Conceptronic, Inc., 90 F.3d 1576, 1582 (Fed.Cir.1996). Drawings from the specification may be used in interpreting the scope of a claim. *See* 5 Donald S. Chisum, *Patents* s. 18.03 [2] [c][iii] at 18-108 (1998) (citing cases).

Claim construction begins with the wording of the claims, asserted and non-asserted, which are to be examined in their entirety. Bell Communications Research, Inc. v. Vitalink Communications Corp., 55 F.3d 615, 620 (Fed.Cir.1995). Unless the specification or the prosecution history indicates that the inventor expressly intended otherwise, a claim term will be accorded its ordinary and accustomed meaning. *Id.;* Markman, 52 F.3d at 979 (stating that "a patentee is free to be his own lexicographer" but that "any special definition given to a word must be clearly defined in the specification").

[6] [7] The Court may look to extrinsic evidence, such as expert testimony, dictionaries and learned treatises, to understand the technology and to construe the claims, only if the claims are ambiguous and only after consideration of all available intrinsic evidence. *See* Vitronics, 90 F.3d at 1584. Opinion testimony on claim construction "is no better than opinion testimony on the meaning of statutory terms." Id. at 1585. Extrinsic evidence, however, may not be used to vary or contradict the terms of the claims. Markman, 52 F.3d at 981.

[8] While claims are to be interpreted in light of the specification and with a view to ascertaining the invention, *see* Unique Concepts, Inc. v. Brown, 939 F.2d 1558, 1561-62 (Fed.Cir.1991), it does not follow that limitations from the specification should be read into the claims. Sjolund v. Musland, 847 F.2d 1573, 1581 (Fed.Cir.1988). Therefore, it is generally improper to limit the scope of the claim to examples used in the specifications. *See* Electro Medical Systems, S.A. v. Cooper Life Sciences, Inc., 34 F.3d 1048, 1054 (Fed.Cir.1994).

Claims drafted in purely functional terms present a special case, however. Pursuant to 35 U.S.C. s. 112(6) (" s. 112(6)"), claim elements described in the form of a "means" for performing a function without recitation in the claim itself of the definite structure for performing the function "shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof." FN1

FN1. 35 U.S.C. s. 112(6) provides that:
An element in a claim for a combination may be expressed as a means or step for performing a specified function without the recital of structure, material, or acts in support thereof, and such claim shall be construed to cover the corresponding structure, material, or acts described in the specification and equivalents thereof.

A determination of the claimed function, being a matter of construction of specific terms in the claim, is a question of law for the Court. Chiuminatta Concrete Concepts Inc. v. Cardinal Industries, Inc., 145 F.3d 1303, 1308 (Fed.Cir.1998). Similarly, the identification of the structure described in the specification that corresponds to a means-plus-function limitation also presents a question of law for the Court. *See* B. Braun

Medical, Inc. v. Abbott Laboratories, 124 F.3d 1419, 1424-25 (Fed.Cir.1997).

The Federal Circuit has held that structure disclosed in the specification is "corresponding" structure only if "the specification or prosecution history clearly links or associates that structure to the function recited in the claim." B. Braun, 124 F.3d at 1424. "This duty to link or associate structure to function is the *quid pro quo* for the convenience of employing s. 112, para. 6." *Id*. However, where the specification elaborates on the details of the preferred embodiment,

more particularly defining the structure in ways unrelated to the recited function, ... [those] additional structural aspects are not what the statute contemplates as structure corresponding to the recited function.

Chiuminatta, 145 F.3d at 1308.

B. Discussion-'603 Patent

IBM and Data General dispute the meaning of the following terms in claim 10 of the '603 Patent: "memory means", "processor means", "call-return operation execution means", "ordinary unresolved pointer", and "represented address of said addresses". Because the construction of "memory means" depends on the construction of "represented address of said addresses", this Court will construe that term first.

The parties do not disagree about the interpretation of the method steps of claim 10, except for their dispute regarding the meaning of "call-return operation execution means" and "ordinary unresolved pointer".

1. "Represented Address of Said Addresses"

[9] Data General maintains that "represented address of said addresses" refers simply to one of the addresses represented by an ordinary unresolved pointer. IBM contends that the phrase refers to two different kinds of logical addresses, of which the second kind of address is used to represent the first.

Claim 10 recites in the third paragraph:

said data items include ordinary unresolved pointers, each one of said ordinary unresolved pointers representing a *represented address of said addresses*....

'603 Patent, col., 7, 11. 14-16. (emphasis added). "Said addresses", described in the first paragraph of Claim 10, are components of memory commands which refer to locations in the memory means. *See* '603 Patent, col. 7, 11.5-8.

An ordinary unresolved pointer does not itself contain one of "said addresses", which specifies a location in the memory means, but rather *represents* a "said addresses", which is subsequently to be determined. The address for which the pointer is a proxy is the "represented address." Thus, "represented address" is the address into which the pointer will eventually be resolved and "said addresses" refers to the set of addresses specifying locations in the memory means of which the "represented address" is one.

Data General argues that Claim 10 does not expressly claim any specific kind of address-physical, logical or otherwise. IBM responds that "represented address of said addresses" refers to *logical* addresses and, more specifically, that "said addresses" found in memory commands are a particular kind of logical address, AON-offset addresses, which are resolved into a second kind of logical address, UID-offset addresses.

The Common Specification describes the use of *logical* addresses in the computer system at issue as follows:

CS 10110's addressing mechanisms are based, first upon *UID addressing* of objects. That is, all information generated for use in or by operation of a CS 10110, for example, data and procedures, is structured into objects and *each object is assigned a permanent UID*.

Effectively, UID addressing means that the address (or memory) space of a particular CS 10110 includes the address space of all systems, for example, disc drives.... FN2

FN2. "UIDs" stands for Unique Identifiers, one aspect of the described logical addressing system.

U.S.Patent No. 4,455,602 ("the '602 Specification" or "Common Specification"), col. 72, 11. 13-17, 11. 24-26 (emphasis added).FN3

FN3. U.S.Patent No. 4,455,602 consists primarily of a specification, which is incorporated by reference in numerous patents, including both the '603 and '797 Patents.

The Common Specification clearly describes the use of logical addresses in connection with the resolution of unresolved pointers:

The Pointer Resolution System translates pointers, i.e., data items which contain location information, into UID-offset addresses.

'602 Specification, col. 327, 11. 15-17.

Although the Common Specification describes the use of logical addresses in its addressing system, limitations from the specification should not necessarily be read into the claims. *See* Sjolund v. Musland, 847 F.2d 1573, 1581-82 (Fed.Cir.1988). While "represented address of said addresses" may be logical addresses, they are not limited to logical addresses.

This Court concludes that "said addresses" refer to the addresses in memory commands specifying a location in memory means and "represented address" is the address derived from the ordinary unresolved pointer.

2. "Memory Means"

[10] As an initial matter, it is necessary to determine whether s. 112(6) applies to the term "memory means", thereby restricting claim limitations to those structures, materials or acts disclosed in the specification that perform the claimed function.

The use of the word "means" creates a presumption that s. 112(6) applies. However, "where a claim recites a function, but then goes on to elaborate sufficient structure, material, or acts within the claim itself to perform entirely the recited function, the claim is not in means-plus-function format." Sage Prods. v. Devon Indus., Inc., 126 F.3d 1420, 1427-28 (Fed.Cir.1997). Data General argues that "memory" describes, to one skilled in the art, the memory and processor of any general purpose computer system, a sufficiently definite structure to remove the term from the strictures of s. 112(6).

The Court must determine whether the claim elaborates sufficient structure *to perform the recited function,* not simply whether the claim contains a term that has a commonly accepted meaning to those of ordinary skill in the art. In Claim 10, the memory means performs the function of:

storing and providing data in response to memory commands, each said memory command including an address specifying a location in said memory means, and ... said data items include ordinary unresolved pointers....

'603 Patent, col. 7, 11. 5-8, 14.

Although the memory of a general purpose computer system is a sufficiently described structure to perform the function of "storing and providing data, " the '603 Patent language is not sufficiently descriptive where the data includes logical addresses. It becomes necessary to examine whether physical or logical memory is used when discussing the use of logical addresses. A physical memory system, in the absence of a mechanism to convert logical addresses into physical addresses, could not perform the function stated in Claim 10. Thus, s. 112(6) applies and it becomes necessary to look to the Common Specification to determine the construction of "memory means."

The Court must first identify the function performed by the means and then identify the structure described in the specification which performs that function, the "corresponding structure." Data General contends that the corresponding structure is the physical main memory of a general purpose computer system, i.e., Main Store Bank 1810, whereas IBM argues that it is the logical memory system described in the Common Specification.FN4

FN4. Main Store Bank 1810 is one of five major units of the main memory (MEM 10112). According to IBM, the logical memory system consists of the structures referred to as MEM 10112, IOS 10116, ED 10124, Memory Reference Unit 27017 and Protection Unit 27019.

Again, the function performed by the memory means is:

... storing and providing data items in response to memory commands, each said memory command including an address specifying a location in said memory means.

'603 Patent, col. 7, 11. 5-8.

Data General maintains that the term "memory" is commonly used in computer engineering to refer to the physical main memory of a computer system. Accordingly, it argues, Main Store Bank is the structure that stores and provides data, using DRAM (Dynamic Random Access Memory). Data General contends that the four other devices in MEM 10112 do not "store and provide data" but rather, are part of the interface between the processor and memory means.FN5

FN5. The four other devices in MEM 10112 are Bank Controller, Field Interface Unit, Memory Cache, and

Memory Interface Controller.

Data General's construction does not, however, encompass the use of *logical* addresses, as described in the Common Specification and discussed in the preceding section of this Memorandum. Although *physical* memory generally stores and provides data, if that data includes *logical* addresses, physical memory *alone* does not store and provide data. Rather, a mechanism is required to convert the logical addresses into physical addresses, compatible with physical memory. The use of logical addresses in the Common Specification indicates a logical, as opposed to a physical, memory system.

Data General's assertion that "memory means" does not refer to logical memory because logical memory is a *scheme* for organizing the memory system, rather than a physical device that "stores" data is unpersuasive. Although logical memory is not a physical device, it is composed of and implemented by physical devices, microcode and software.

This Court concludes that the term "memory means" refers to the logical memory system of a general purpose computer system and the corresponding structures that perform the function of storing and providing data which contain logical addresses are MEM 10112, IOS 10116, ED 10124, MRU 27017 and PU 27019.

3. "Processor Means"

[11] Data General maintains that "processor means" refers to the central processor of any general purpose computer system, while IBM argues that the term refers to FU 10120, an unconventional processor described in the Common Specification, not including MRU 27017 or PU 27019.

Claim 10 refers to "processor means" as follows: ... *processor means* connected to said memory means for providing said memory commands and providing and receiving said data items in response to sequences of instructions of said data items executed by said processor means....

'603 Patent, col. 7, 11. 9-13 (emphasis added).

Data General contends that s. 112(6) does not apply to the construction of the term "processor means" because the claim recites a structure which performs the functions associated with the "means", while IBM responds that there is no detailedand specific structure set forth in the claim.

The functions performed by the "processor means" set forth in Claim 10 are 1) providing memory commands, 2) providing and receiving data items which are sequences of instructions and 3) executing those sequences of instructions. *See* '603 Patent, col. 7, 11. 10-13.

This Court is persuaded by the testimony of Data General's expert, Professor Finkel, that, to one of ordinary skill in the art, "processor" refers to a central processing unit which provides memory commands and provides and receives data items. (Finkel T. 1/103-04). Because Claim 10 refers to a structure, i.e., a "processor" which a person of ordinary skill in the art would consider to be a structure that performs the same functions as the "means", s. 112(6) does not apply to the construction of that term. Thus, this Court concludes that the term "processor means" in the '603 Patent refers to the central processing unit of a general purpose computer system.

4. "*Call-Return Operation Execution Means*"

[12] Data General claims that call-return operation execution mechanism refers to the processor of a general purpose computer system, together with instructions implementing conventions for calling and returning from any sequence of instructions. IBM, on the other hand, maintains that the term should be construed as the microcode-to-software call-return mechanism described in the Common Specification. Their dispute devolves to whether the claim includes a software-to-software call-return mechanism or simply a microcode-to-software call-return mechanism.

Claim 10 sets forth in relevant part:

... said processor includes *call-return operation execution means* responsive to operation of said processor means for causing said processor means to commence and terminate an execution of any said sequence, [for the purpose of resolving ordinary unresolved pointers where the resolution consists of] the steps of ... (3) causing said *call-return operation execution means* to commence execution of a pointer resolution sequence of said sequences....

'603 Patent, col. 7, 11. 19-27, 33-35 (emphasis added).

Section 112(6) presumptively applies because the term is written in "means-plus-function" language. Data General asserts that the claim recites supporting structure to perform the function claimed. Specifically, Data General claims that every processor of any general purpose computer system includes instructions capable of implementing conventions for calling and returning from any sequence of instructions.

Although Data General is correct that the typical processor is capable of calling to and returning from procedures, this Court is persuaded by the testimony of IBM's expert, Professor Graham, that processors have different ways of calling and returning from procedures. (Graham T. 3/27). Thus, even though the claim refers to a "processor" and processors contain instructions that enable them to perform call-return operations, because such operations may be performed differently by processors, the claim does not recite a sufficiently specific structure and s. 112(6) applies.

Here, the function of the call-return operation execution means is to cause the processor means to commence and to terminate an execution of any said sequence. IBM claims that only one structure, FU 10120 (an unconventional processor executing microcode-to-software call microcode) is described in the intrinsic evidence as performing that function.

The Common Specification specifically describes an alternate embodiment, however, that uses a software-to-software call to invoke the pointer resolution sequence:

In alternate embodiments of CS 10110, KOS subsystems may be embodied entirelyin microcode, or in high-level language routines.

'602 Specification, col. 326, 11. 1-3.FN6

FN6. According to Professor Graham, in this context, "high-level language routines" means software. (Graham T. 3/37).

Although the prosecution history cites FU 10120 executing microcode-to-software call microcode as an example of supporting matter for the call-return operation execution means element of Claim 10, the prosecution history also states that:

Examples of supporting matter in the figures, specification and original claims are provided below; however, additional support may be found elsewhere in the specification and original claims, and the new claims are to be understood in the light of the entire specification and original claims.

August, 1983 Amendment, p. 25.

IBM responds that a software-to-software call-return mechanism is inconsistent with the claim language, "said processor *includes* a call-return operation execution means". That response is unpersuasive given Professor Graham's testimony that, to one skilled in the art, a processor "includes" instructions that can implement conventions for calling and returning. (Graham T. 3/70-71).

This Court concludes that "call-return operation execution means" refers to the processor of a general purpose computer system, together with instructions implementing conventions for calling and returning from any sequence of instructions, including both microcode-to-software call-return mechanisms and software-to-software call-return mechanisms.

5. "Ordinary Unresolved Pointer"

[13] While the parties agree that "ordinary unresolved pointers" are always resolved by high-level language routines, IBM reads into the term the further requirement that an "ordinary unresolved pointer" is an unresolved pointer that is not converted into a resolved pointer after the address of its target is determined. Data General, on the other hand, maintains that an ordinary unresolved pointer could be (but is not necessarily) resolved once by a high-level language routine and then converted into a resolved pointer.

Proper claim construction of the term "ordinary unresolved pointer" requires an understanding of several related terms, of which the definitions are undisputed. The parties agree that a "pointer" is a data item that represents the address of a target data item. A "resolved pointer" contains the target's address. An "unresolved" pointer does not contain the target's address, but instead, contains data from which that address may be determined.

Claim 10 recites in relevant part:

... said data items include *ordinary unresolved pointers,* each one of said *ordinary unresolved pointers* representing a represented address of said addresses and containing said data items from which said represented address may be derived....

'603 Patent, col. 7, 11. 14-18 (emphasis added).

The term "ordinary unresolved pointer" is not a term commonly used in computer engineering, and thus, this Court must look beyond Claim 10 to the Common Specification to construe the term. The Common Specification distinguishes between two kinds of unresolved pointers, "ordinary unresolved pointers" and "associative pointers":

The subclasses of unresolved pointers are ordinary unresolved pointers and associative pointers. The difference between the two kinds of unresolved pointers is the manner in which they are resolved. Ordinary unresolved pointers are always resolved by high-level language routines, while associative pointers are resolved the first time they are used in a Process 610 and a domain by high-level language routines, but are subsequently resolved by means of a table called the Associated Address Table (AAT). This table is accessible to microcode, and associative pointers may therefore be more quickly resolved than ordinary unresolved pointers.

'602 Specification, col. 327, 1.60 - col. 328, 1. 4. The parties agree that, as the Common Specification unambiguously states, ordinary unresolved pointers are always resolved by "high-level language routines."

Data General maintains that an ordinary unresolved pointer could be resolved once by a high-level language routine and then be converted into a resolved pointer. According to Data General, Claim 10 simply requires that the pointer resolution sequence return the target's address but is silent as to whether or not the unresolved pointer is overwritten with the resolved pointer.

This Court was persuaded by Professor Graham's testimony that the replacement of an unresolved pointer with a resolved pointer is conceptually distinct from determining the target of the unresolved pointer. (Graham T. 3/19). The computer can access the target data item without overwriting the unresolved pointer, but at the cost of having to determine again the target address should it encounter the pointer again in the same program. For reasons of efficiency, a computer might overwrite a pointer so that it can go directly to the target data item if it encounters the pointer again in the same run, without the need to resolve the pointer again. Claim 10, Data General argues, does not speak to whether the computer performs this optional step.

In IBM's view, however, the language of the Common Specification supports its proposed construction, namely, that an ordinary unresolved pointer is never converted into a resolved pointer upon the determination of the address of its target. IBM further argues that Data General's construction is inconsistent with the language in the Common Specification.

First, IBM asserts that Data General's contention that an ordinary unresolved pointer could be resolved once and then converted into a resolved pointer contradicts the statement in the Common Specification that "ordinary unresolved pointers are always resolved by high-level language routines." IBM contends that if Data General's position were correct, then an ordinary unresolved pointer would be "resolved by high-level language routines" only the first time it was used. Thereafter, on subsequent uses of the pointer, no resolution would be required. Thus, such a pointer would not "always" be "resolved by high-level language routines."

IBM's argument is without merit. If a computer replaced an unresolved pointer with a resolved pointer, after determining the target's address, that pointer will no longer be unresolved when encountered again in the same program. Thus, replacing an ordinary unresolved pointer with a resolved pointer is not inconsistent with the statement in the Common Specification that an ordinary unresolved pointer is an unresolved pointer that is "always resolved by high-level language routines" because a particular ordinary unresolved pointer disappears when it is replaced with a resolved pointer.

Second, IBM argues that Data General's proposed construction contradicts the statement in the Common

Specification that "associative pointers may therefore be more quickly resolved than ordinary unresolved pointers." IBM contends that if, according to the system Data General envisions, a particular ordinary unresolved pointer were overwritten with a resolved pointer, the next time the processor encountered that pointer it would be "resolved" more quickly than an associative unresolved pointer, which would still have to be looked up in the Associated Address Table.

This argument is similarly unavailing. Again, the critical weakness in IBM's argumentlies in the fact that an ordinary unresolved pointer would disappear were it replaced by a resolved pointer. If a particular ordinary unresolved pointer were overwritten with a resolved pointer, the next time the processor encountered that pointer, it would no longer be an ordinary *unresolved* pointer. The notion that a resolved pointer is "resolved" faster than an associative pointer in no way contradicts the Common Specification language that "associative pointers may therefore be more quickly resolved than ordinary unresolved pointers."

Thus, as outlined above, Data General's interpretation does not appear to be inconsistent with the language in the Common Specification. Data General's expert, Professor Finkel, highlights an example in the Common Specification where an unresolved pointer is resolved and then overwritten. (Finkel T. 1/109-10). That portion of the Common Specification which describes the resolution and overwriting of unresolved pointers by software lends credence to Data General's proposed construction while undermining that of IBM:

The Dynamic Linker then obtains the UID of Procedure Object 608 from EOS, uses the UID and offset information contained in the Binder Area location to locate Procedure 602's gate, places the pointer to the Gate at the location occupied by the unresolved pointer in the Static Data Block, and returns the pointer to the pointer-to-descriptor microcode. The microcode is now able to convert the pointer to a descriptor....

'602 Specification, col. 331, 1. 66 - col. 332, 1. 5.

C. Discussion-'797 Patent

IBM and Data General dispute the meaning of the following terms in Claim 1 of the '797 Patent: "memory means", "instructions", "common formats", and "processor means".

1. "Memory Means"

[14] IBM proposes the same construction of "memory means" as in the '603 Patent, i.e., that "memory means" refers to the logical memory system described in the Common Specification, which, IBM argues, consists of the structures referred to as MEM 10112, IOS 10116, ED 10124, MRU 27017 and PU 27019. Data General likewise proposes the same construction it proffered in connection with the '603 Patent, i.e., "memory means" refers to the physical memory of a general purpose computer system, which is Main Store Bank 1810 in the Common Specification.

Although Data General is correct that "memory" has a generally understood meaning in computer science, its contention that "memory," when read in light of the claim function, denotes a device with that specific meaning is suspect.

The '797 Patent provides that:

The memory system is organized into objects containing data items. Each object is identified by an object identifier. Locations of data items in the memory system are specified by means of the object identifier for the object containing the data item and an offset specifying the location at which the data item begins in the object. The memory system responds to a memory operation specifier consisting of a memory command specifying a memory operation such as read data or write data, a logical address containing a representation of an object identifier and an offset.

'797 Patent, col. 3, 11. 3-13. Physical memory, without supporting mechanisms to translate logical addresses into physical addresses, would not be able to perform the functions set forth in Claim 1. Thus, it is clear that "memory means" in the '797 Patent refers to logical memory.

2. "Instructions"

[15] Data General contends that "instructions" means S-Instructions which are a level below high-level language instructions and a level above conventional machine language instructions. IBM argues that "instructions" means conventional machine instructions into which programs written in high-level user languages are compiled for execution by the processor.

Claim 1 provides in relevant part:

... *instructions,* each one of said *instructions* containing an operation code of a plurality of operations codes, said operation codes belonging to a plurality of functionally different operation code sets, said operation codes in a given one of said operation code sets being definable solely with reference to said given operation code set....

'797 Patent, col. 4, 11. 29-36. Claim 1 also provides that the processor means receives and responds to said instructions by performing the operation in the instruction. *See* '797 Patent, col. 4, 11. 38-44.

IBM offers extrinsic evidence that "instructions," to persons skilled in the art, means executable machine language instructions into which high-level language programs are compiled. Before relying on extrinsic evidence, however, this Court must look to the Common Specification to understand what the drafter of the patent meant by "instructions."

The Common Specification provides that:

CS 101 is both an S-Language machine and a Name-space machine. That is, operations to be executed by CS are expressed as S-Language Operations (SOPs) while operands are identified by Names. *SOPs are of a lower, more detailed, level than user language instructions, for example FORTRAN and COBOL, but of a higher level than conventional machine language instructions*. SOPs are specific to particular user languages rather than a particular embodiment of CS 101, while conventional machine language instructions are specific to particular machines. SOPs are in turn interpreted and executed by microcode. There will be a S-Language Dialect, a set of SOPs, for each user languages (sic). CS 101, for example, may have SOP Dialects for COBOL, FORTRAN, and SPL.

'602 Specification, col. 21, 1. 57 - col. 22, 1. 3. (emphasis added). Thus, the Common Specification makes it clear that the claim language referring to functionally different operation code sets refers to the S-Instructions (or SOPs) of S-Languages.

IBM's contention that Data General would have used the word "S-Instructions" rather than "instructions" if it had meant the former is baseless. Because the concept of "S-Instructions" and a mechanism to compile high-level languages into such instructions which are then converted into machine instructions by S-Language interpreters was a novel concept when the patent was issued, it stands to reason that the patent drafters would explain S-Instructions in a detailed manner.

Furthermore, it is not necessarily true, as IBM implies, that the processor cannot execute S-Instructions, and that the instructions which the processor receives from memory and responds to must therefore be conventional machine instructions. Its implication is inaccurate because the processor receives S-Instructions and uses an S-Language interpreter to convert them to machine instructions. Thus "instructions" received by the processor may be S-Instructions.

IBM argues that "instructions" could not mean S-Instructions because it would render superfluous all of the limitations set forth in the claim. That argument is nonsensical. The Court does not construe "instructions" as S-Instructions, thereby rendering the surrounding language superfluous, but rather, interprets the *entire* description of "instructions," including the descriptive characteristics which follow it, as S-Instructions.

Nor does construing "instructions" to mean S-Instructions "disregard" the plain meaning of the word "said," as IBM argues. Claim 1 first establishes that "said" instructions refer to instructions which are part of the data provided by the memory means and then describes the characteristics of such instructions.

This Court construes "instructions" to mean "S-Instructions" which are a level below high-level language instructions and a level above conventional machine language instructions.

3. "Common Formats"

[16] IBM argues that the phrase "common formats" requires that all of the instructions stored and provided by memory means have a "fixed" or "uniform" format, while Data General contends the phrase allows for more than one format of instructions but requires commonality.

Claim 1 provides that "instructions", now construed as S-Instructions which are a level below high-level language instructions and a level above conventional machine language instructions, have "common formats." *See* '797 Patent, col. 4, 11. 30-37.

Instruction format refers to the layout of the constituent parts of the instruction. An instruction typically begins with an operation code and is followed by operation code syllables known as Names. The length of operation codes and operation code syllables are measured in "bits." Data General suggest that two sets of instructions consist of instructions having common formats if the formats of one set are the same as, or a subset of, the formats of the other set.

IBM disagrees and asserts that the term "common formats" requires that all instructions share a "common" format in that every instruction has an operation code with a fixed length and operand syllables that have a fixed length within a procedure. In support of its proposed construction, IBM cites the following portion of the Common Specification which describes instructions as having a fixed format:

... the I-Stream has a fixed format: in the present embodiment, SOPs are always 8 bits long, and Names may be 8, 12, or 16 bits long. The length of Names used in a given procedure is fixed....

'602 Specification, col. 345, 11. 60-63. Based upon that and similar provisions of the Common Specification, IBM argues that all instructions must have a fixed format. It contends that because "common formats" does not have a commonly understood meaning in the art of computer science, IBM argues that it is necessary to depend upon the Common Specification to glean the meaning of that phrase.

Again, this Court is unpersuaded by IBM's argument for three reasons:

1. Although the phrase "common formats" does not have a generally accepted meaning in the art, the word "formats" does have such a meaning and the word "common" is not a term of art in computer science. "Common" has the same generally accepted meaning to one skilled in the art of computer science and to laypersons.

2. Even if it were necessary to rely solely on the Common Specification, it does not follow that limitations from the specification should be read into the claims. *See* Sjolund v. Musland, 847 F.2d 1573, 1581-82 (Fed.Cir.1988).

3. To construe the phrase "common formats" to mean that all instructions have a single, fixed format contradicts the plain meaning of the plural "formats". The use of a plural indicates that instructions may have more than one format.

The phrase "said instructions" which are to have common formats refers to instructions containing operation codes belonging to more than one functionally different operation code sets. Instructions which are part of two separate sets of instructions must have identical formats. Although the formats in instruction sets of different S-Languages must be the same, Claim 1 does not require that only one fixed format may be used. This Court construes "common formats" as allowing for instruction sets to have more than one format, but requiring commonality.

4. "Processor Means"

[17] Finally, IBM and Data General dispute the meaning of the term "processor means". IBM argues that it refers to FU 10120, the hardware implementation of the processor in the preferred embodiment in the Common Specification, which includes numerous specialized components.FN7 Data General responds that "processor means" also includes a conventional CPU of a computer system, a software interpreter and an operating system.

FN7. IBM claims that the corresponding structure is FU 10120, *except* for Memory Reference Unit 27017 and Protection Unit 27019.

The parties agree that s. 112(6) applies to the term "processor means" in the '797 Patent and thus, the construction of that term is limited to the corresponding structures in the specification which perform the function associated with the "processor means." FN8

FN8. Contrary to IBM's contention, it is not inconsistent for this Court to apply s. 112(6) to the term "processor" in the '797 Patent while declining to apply that statutory provision to the identical term used in the '603 Patent. While "processor" refers to a definite structure sufficient to perform the function set forth in

Claim 10 of the '603 Patent, it is not apparent from the claim language in the '797 Patent what kind of "processor" is necessary to perform that stated function. Although identical claim terms used in different claims must be interpreted consistently when they are used in different claims *of the same patent, See* American Permahedge, Inc. v. Barcana, Inc., 105 F.3d 1441, 1446, (Fed.Cir.1997) (emphasis added), the meaning may vary between identical claim terms used in different patents.

The function of the processor means of Claim 1 is:

... receiving said items including said instructions and responding to each said received instruction by performing said operation defined for said operation code in said received instruction in said operation code set to which said operation code in said received instruction belongs.

'797 Patent, col. 4, 11. 39-44. The Court has previously found that "said instructions" refers to S-Instructions which are a level below high-level language instructions and a level above conventional machine language instructions.

The Common Specification provides, in pertinent part, that:

Principal functions of FU 10120 include: (1) Fetching and interpreting instructions, that is SINs comprising SOPs and Names, and data from MEM 10112 for use by FU 10120 and EU 10122; (2) Organizing and controlling flow and execution of user programs; (3) Initiating EU 10122 operations; (4) Performing arithmetic and logic operations on data; (5) Controlling transfer of data from FU 10120 and EU 10122 to MEM 10122; and, (6) Maintaining certain stack register mechanisms.

'602 Specification, col. 172, ll. 20-29. It is clear that FU 10120 performs the function performed by the "processor means" in Claim 1.

In addition, Data General contends an alternative corresponding structure may also perform the function of the "processor means." In Micro Chemical, Inc. v. Great Plains Chemical Co., 194 F.3d 1250, 1258-59 (Fed.Cir.1999), the specification disclosed three kinds of structures: (1) the preferred embodiment, (2) several alternative embodiments and (3) a suggested embodiment. The Federal Circuit Court held that the pertinent means element includes each of the three embodiments or an equivalent structure.

Data General argues that a conventional CPU, together with a software interpreter and an operating system, could perform the function of the "processor means." The Common Specification states that FU 10120 may be "replaced by a conventional CPU, such as Data General Corporation's Eclipse(R)." '602 Specification, col. 73, ll. 33-36.

IBM concedes that a conventional CPU is *mentioned* in the Common Specification, but asserts that 1) it is not sufficiently *described* to be an alternative corresponding structure and 2) the Common Specification never states that a conventional CPU can perform the claimed processor means function. When the Common Specification notes, however, that FU 10120 may be replaced by a conventional CPU, it implies that a conventional CPU can perform the same functions that FU 10120 performs and thus be considered an alternative corresponding structure.

Furthermore, IBM's contention that a software interpreter is not disclosed conflicts with the intrinsic evidence. The Common Specification discloses a software interpreter:

In the present embodiment, the interpreters consist of dispatch tables and microcode, but in other embodiments, the interpreters may themselves be written in high-level languages.

'602 Specification, col. 360, ll. 27-31. As Data General's expert witness, Professor Gruner, testified, interpreters written in high-level languages are software interpreters. (Gruner T. 2/42-43).

This Court finds that "processor means" may refer to either FU 10120 (except for Memory Reference Unit 27017 and Protection Unit 27019) or a conventional CPU, together with a software interpreter and an operating system.

III. Conclusion

Based upon the preceding analysis, with respect to the contested terms as they are used in Claim 10 of the '603 Patent, this Court concludes that:

1) "represented address" refers to an address into which an ordinary unresolved pointer is resolved;

2) "said addresses" refers to addresses that are part of memory commands which specify a location in the memory means;

3) "memory means" refers to the logical memory system of a general purpose computer system; the structures implementing logical memory are MEM 10112, IOS 10116, ED 10124, MRU 27017 and PU 27019;

4) "processor means" refers to the central processing unit of any general purpose computer system;

5) the structure that corresponds to "call-return operation execution means" is the processor of a general purpose computer system, together with instructions implementing conventions for calling and returning from any sequence of instructions, including both microcode-to-software and software-to-software call-return mechanisms; and

6) an "ordinary unresolved pointer" may be resolved once by a high-level language routine after which it is deemed converted into a resolved pointer.

With respect to the contested terms as they are used in Claim 1 of the '797 Patent, this Court concludes that:

1) "memory means" refers to the logical memory system and includes the same structures found to be part of memory means in connection with the '603 Patent;

2) "instructions" refers to "S-instructions" which are a level below high-level language instructions and a level above conventional machine language instructions;

3) "common formats" allows for more than one format of instructions, but commonality is required amongst them; and

4) "Processor means" may refer to either FU 10120 (except for Memory Reference Unit 27017 and Protection Unit 27019) or a conventional CPU, together with a software interpreter and an operating system.

D.Mass.,2000.
Data General Corp. v. International Business Machines Corp.