IN THE

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

LZLABS GMBH

Petitioner

v.

INTERNATIONAL BUSINESS MACHINES CORPORATION

Patent Owner

IPR2023-00274

Patent No. 8,713,289

PETITIONER'S NOTICE OF APPEAL

Notice is hereby given, pursuant to 35 U.S.C. §§ 141(c), 142, and 319; 37 C.F.R. §§ 90.2(a) and 90.3(a); Rules 4(a) and 15 of the Federal Rules of Appellate Procedure; Federal Circuit Rule 15; and 28 U.S.C. § 1295(a)(4)(A), that Petitioner LzLabs GmbH ("LzLabs") appeals to the United States Court of Appeals for the Federal Circuit from the Final Written Decision and Judgment entered May 23, 2024 by the Patent Trial and Appeal Board (Paper 21, Attachment A) (the "Final Written Decision"), and all underlying and related findings, orders, decisions, rulings, opinions, or other determinations merged into those orders. This notice is timely filed within 63 days of the Final Written Decision. 37 C.F.R. § 90.3(a)(1).

In accordance with 37 C.F.R. § 90.2(a)(3)(ii), LzLabs further indicates that the issues on appeal include, but are not limited to, the Patent Trial and Appeal Board's determination of the scope of patented subject matter; its application of the claim language, including implied construction of the term "target machine condition code;" its failure to provide a reasoned basis for its findings and conclusions; and any Board or Director finding, determination, judgment, or order supporting or related to the Final Written Decision and decided adversely to LzLabs.

LzLabs is concurrently filing true and correct copies of this Notice of Appeal, along with any required fees, with the United States Court of Appeals for the Federal Circuit, the Patent Trial and Appeal Board, and the Director.

July 19, 2024
Date

/s/ *Joseph D. Gray*

Joseph D. Gray (Reg. 61,803)
Tecuan Flores (Reg. 77,668)
Slayden Grubert Beard PLLC
401 Congress Avenue, Suite 1650
Austin, Texas 78701

Christopher V. Ryan (Reg. No. 54,759)
RyanIPLaw, PC
1508 Bay Hill Drive
Austin, TX 78701-4078

Counsel for Petitioner

CERTIFICATE OF FILING

I hereby certify that, in addition to being filed electronically through the Patent Trial and Appeal Board's P-TACTS system, a true and correct copy of the above-captioned Petitioner's Notice of Appeal of Final Written Decision is also being filed by Electronic Mail with the Director of the United States Patent and Trademark Office, on July 19, 2024, at the following address pursuant to 37 C.F.R. §§ 90.2(a), 104.2(a):

efileSO@uspto.gov

I also hereby certify that a true and correct copy of the above-captioned Petitioner's Notice of Appeal of Final Written Decision, along with the filing fee, is being filed via CM/ECF with the Clerk's Office of the United States Court of Appeals for the Federal Circuit on July 19, 2024.

Respectfully submitted,

July 19, 2024
Date

/s/ *Joseph D. Gray*

Joseph D. Gray (Reg. 61,803)
Tecuan Flores (Reg. 77,668)
Slayden Grubert Beard PLLC
401 Congress Avenue, Suite 1650
Austin, Texas 78701

Christopher V. Ryan (Reg. No. 54,759)
RyanIPLaw, PC
1508 Bay Hill Drive
Austin, TX 78701-4078

Counsel for Petitioner

CERTIFICATE OF SERVICE

In accordance with 37 C.F.R. §§ 42.6(e), the undersigned certifies that a complete copy of the foregoing was served on counsel of record for Patent Owner on July 19, 2024 via Electronic Mail to the following addresses:

tkonstantakopoulos@desmaraisllp.com

yha@desmaraisllp.com

jwilcox@desmaraisllp.com

ibmlzlabsservice@desmaraisllp.com

Respectfully submitted,

July 19, 2024
Date

/s/ *Joseph D. Gray*

Joseph D. Gray (Reg. 61,803)
Tecuan Flores (Reg. 77,668)
Slayden Grubert Beard PLLC
401 Congress Avenue, Suite 1650
Austin, Texas 78701

Christopher V. Ryan (Reg. No. 54,759)
RyanIPLaw, PC
1508 Bay Hill Drive
Austin, TX 78701-4078

Counsel for Petitioner

Attachment A

Trials@uspto.gov
571-272-7822

Paper 21
Entered: May 23, 2024

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

LZLABS GMBH,
Petitioner,

v.

INTERNATIONAL BUSINESS MACHINES CORPORATION,
Patent Owner.

IPR2023-00274
Patent 8,713,289 B2

Before KEN B. BARRETT, MICHELLE N. WORMMEESTER, and ARTHUR M. PESLAK, *Administrative Patent Judges*.

WORMMEESTER, *Administrative Patent Judge*.

JUDGMENT
Final Written Decision
Determining No Challenged Claims Unpatentable
35 U.S.C. § 318(a)

I. INTRODUCTION

LzLabs GmbH ("Petitioner") filed a Petition (Paper 1, "Pet.") requesting *inter partes* review of claims 1–4, 6, 10–12, 16, and 17 of U.S. Patent No. 8,713,289 B2 (Ex. 1001, "the '289 patent"). International Business Machines Corporation ("Patent Owner" or "IBM") did not file a preliminary response. Pursuant to 35 U.S.C. § 314, we instituted an *inter partes* review of all the challenged claims based on all the grounds presented in the Petition. Paper 7 ("Inst. Dec."). Thereafter, Patent Owner filed a Response (Paper 12, "PO Resp.") to the Petition, Petitioner filed a Reply (Paper 13, "Pet. Reply"), and Patent Owner filed a Sur-reply (Paper 14, "PO Sur-reply"). On March 21, 2024, we conducted an oral hearing. A copy of the transcript (Paper 20, "Tr.") is in the record.

We have jurisdiction under 35 U.S.C. § 6(b). This Final Written Decision is issued pursuant to 35 U.S.C. § 318(a). For the reasons that follow, we determine that Petitioner has not shown by a preponderance of the evidence that claims 1–4, 6, 10–12, 16, and 17 of the '289 patent are unpatentable.

II. BACKGROUND

A. Related Proceedings

The parties identify one federal district court case: *International Business Machines Corp. v. LzLabs GmbH*, No. 6:22-cv-00299 (W.D. Tex.). Pet. 1; Paper 3, 1 (Patent Owner's Mandatory Notices). Petitioner also identifies one *inter partes* review proceeding. Pet. 1.

B. The '289 Patent

The '289 patent describes emulating computer system architectures, and focuses on "providing sequences of instructions that produce valid condition code settings without the use of branching instructions from the target architecture." Ex. 1001, 1:8–14. To illustrate an emulation environment, Figure 5 of the '289 patent is reproduced below.

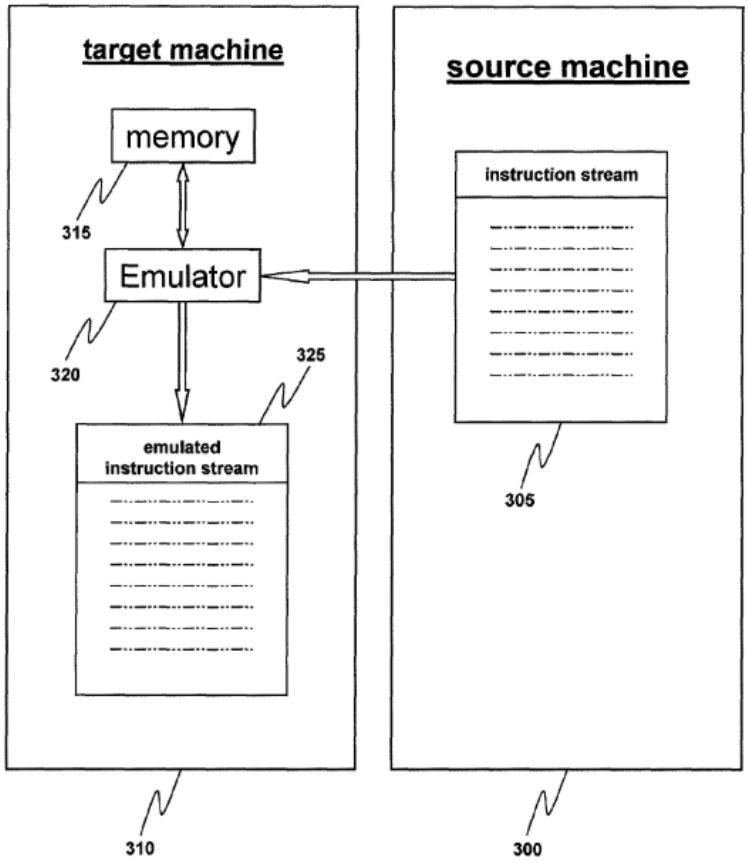

Fig. 5

Figure 5 is a block diagram of an emulation environment that includes source machine 300 and target machine 310. *Id.* at 4:16–17, 8:64–65. Emulator 320 receives a stream of instructions 305, which represent machine or assembly language instructions designed to operate on source machine 300. *Id.* at 8:65–9:1. Emulator 320 uses memory 315 in target

machine 310 to produce a stream of instructions that are capable of executing on target machine 310. *Id.* at 9:1–4.

The '289 patent explains that the execution of various operations, such as arithmetic, logical, and data transfer operations, "may result in the generation of several bits of data to indicate the outcome status of instruction execution," where "[t]hese bits are typically referred to as condition codes." Ex. 1001, 1:18–23. As an "example of an instruction which produces condition code changes upon execution," the '289 patent identifies "the compare instruction which sets a condition code to 'zero' if the operands are equal, to 'one' if the first operand is strictly less than the second operand and to 'two' if the first operand is strictly greater than the second operand." *Id.* at 1:29–34. The '289 patent notes that, "[i]n conventional approaches to the emulation of a compare instruction, the result is the construction of a sequence of instructions, which include three branch instructions." *Id.* at 2:7–10; *see also id.* at Fig. 1. According to the '289 patent, branch instructions are not desirable because they may impede overall processing speed and waste computer resources. *Id.* at 2:13–32.

The '289 patent thus provides for the generation of "a sequence of target machine instructions which together operate to directly calculate target machine condition codes from carry, sign and overflow codes without the use of branch instructions from the target machine." Ex. 1001, 3:12–18. To illustrate, Table II of the '289 patent is reproduced below.

TABLE II

[1]	sub__set__carry	rC, rA, rB
[2]	set__bit__on__zero	rX, rC
[3]	set__bit__on__not__zero	rC, rC
[4]	add__to__carry__immed	rC, rC, 0
[5]	sub	rC, rC, rX

	rA > rB Register Contents			rA = rB Register Contents			rA < rB Register Contents		
Instruction	rC	rX	Carry	rC	rX	Carry	rC	rX	Carry
[1]	rC > 0	N/A	1	rC = 0	N/A	1	rC < 0	N/A	0
[2]	rC > 0	0	1	rC = 0	1	1	rC < 0	0	0
[3]	1	0	1	0	1	1	1	0	0
[4]	2	0	N/A	1	1	N/A	1	0	N/A
[5]	2	0	N/A	0	1	N/A	1	0	N/A

Table II shows an example of a sequence of instructions for emulating condition code settings of a source instruction, namely, the z/Architecture Instruction called the Compare Logical operation. *Id.* at 5:28–35. In this example, the z/Architecture is the architecture of the source machine. *See id.* at 2:62–67. The designations rA and rB represent the first and second operands of the z/Architecture instruction evaluated into a register. *Id.* at 5:20–21. The designation rX represents a temporary register used to hold intermediate results. *Id.* at 5:24–25. The designation rC represents the register that will hold the condition code value at the end of the sequence. *Id.* at 5:26–27. The designation "Carry" represents the carry bit flag in the target machine. *Id.* at 5:59–66.

As shown in Table II, after executing instruction [1] (sub__set__carry) in the target machine, the CPU[1] carry bit is set to "1" where rA $\geq$ rB or to "0" where rA < rB. Ex. 1001, 5:59–61, 6:21–23. The

[1] CPU stands for "central processing unit." Ex. 1001, 8:26–28.

value of rC accounts for the result of the subtraction, where the entries rC > 0, rC = 0, and rC < 0 indicate the resulting condition. *Id.* at 6:23–26.

After executing instruction [2] (set_bit_on_zero), rC and the carry bit remain the same, but rX is set to "1" if rA and rB are the same based on the contents of rC. Ex. 1001, 6:27–32.

After executing instruction [3] (set_bit_on_not_zero), the carry bit remains the same, but rC is set to "1" where rC is not zero. Ex. 1001, 6:33–37. The '289 patent explains that, "at this point, rX is set up to provide discrimination information distinguishing equality from inequality," and "this occurs outside of (that is, apart from) both rC and the CPU carry bit." *Id.* at 6:37–58.

Executing instruction [4] (add_to_carry_immed) involves carrying out the operation rC + CPU carry bit + 0, where rC is set to "2" if rA > rB or to "1" if rA = rB or rA < rB. Ex. 1001, 6:59–67. The '289 patent notes that, at this point, "there is provided an indication in rC for which the case rA>rB is distinguished from the other two cases (rA=rB and rA<rB)." *Id.* at 6:67–7:2.

Instruction [5] (sub) is the last step, where rX is subtracted from rC to distinguish the case of rA = rB from the case of rA < rB. Ex. 1001, 7:3–8. At the end of the instruction sequence, rC = 2 if rA > rB, rC = 1 if rA < rB, and rC = 0 if rA = rB. *Id.* at 7:8–10.

C. Illustrative Claim

As noted above, Petitioner challenges claims 1–4, 6, 10–12, 16, and 17 of the '289 patent. Claims 1, 11, and 16 are independent. Claim 1, reproduced below, is illustrative of the claims under challenge.

> 1. A computer program product for emulating computer instructions from a source machine to produce sequences of instructions on a target machine, said computer program product comprising:
>
> a non-transitory computer readable storage medium readable by a processor and storing instructions for execution by the processor for performing a method comprising:
>
> obtaining by the target machine a computer instruction from the source machine, said source machine having a different architecture from said target machine; and
>
> generating a sequence of target machine instructions which together operate to derive an encoding for a target machine condition code for the computer instruction, wherein the sequence of target machine instructions provides distinguishing information to distinguish between a plurality of possible outcomes for the target machine condition code, and directly calculates the target machine condition code without using branch instructions, the directly calculating comprising:
>
> determining an intermediate condition code value, the intermediate condition code value being a provisional value for the target machine condition code and subject to change based on the distinguishing information; and
>
> determining, based on the intermediate condition code value and based on the distinguishing information, the target machine condition code, wherein at least part of said distinguishing information is separate from the intermediate condition code value.

D. Asserted Grounds of Unpatentability

Petitioner challenges claims 1–4, 6, 10–12, 16, and 17 of the '289 patent on the following two grounds. Pet. 10, 18–76. We instituted *inter partes* review on every ground presented in the Petition. Inst. Dec. 22.

Claims Challenged	35 U.S.C. §[2]	References
1, 4, 6, 10, 11, 16	103	Loderer[3]
1–4, 6, 10–12, 16, 17	103	Loderer, PoO,[4] Kane,[5] knowledge of a POSITA[6]

In support of its asserted grounds, Petitioner relies on a Declaration of Ian Jestice (Ex. 1003). Patent Owner relies on a Declaration of Richard M. Goodin (Ex. 2001). A transcript of the deposition of Mr. Jestice (Ex. 2003) is also in the record.

III. DISCUSSION

A. Claim Construction

In an *inter partes* review proceeding, we construe a claim of a patent "using the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. 282(b)." *See* 37 C.F.R. § 42.100(b) (2021). Applying that standard, we construe a claim in accordance with its ordinary and customary meaning as would have been understood by one of ordinary skill in the art, taking into account the specification and the prosecution history pertaining to the patent. *See id.*; *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–17 (Fed. Cir. 2005) (en banc).

---

[2] The Leahy-Smith America Invents Act ("AIA"), Pub. L. No. 112-29, 125 Stat. 284 (2011), amended 35 U.S.C. § 103, effective March 16, 2013. Because the application from which the '289 patent issued was filed before this date, the pre-AIA version of § 103 applies.
[3] Loderer, EP Publ'n No. 0 843 256 A2, published May 20, 1998 (translated by Sun IP, May 11, 2022) (Ex. 1005).
[4] IBM, *Enterprise Systems Architecture/390 Principles of Operation, SA22-7201-08* (9th ed., June 2003) (Ex. 1008).
[5] Gerry Kane, MIPS RISC ARCHITECTURE (1989) (Ex. 1009).
[6] POSITA stands for "person of ordinary skill in the art."

Petitioner asserts in its Petition that "terms should be given 'ordinary and customary meaning' to those of skill in the art." Pet. 17. In its Reply, Petitioner adds,

> To the extent the Board determines a construction is necessary to resolve the parties' dispute, the Board should interpret "target machine condition code" consistent with the intrinsic record and [Patent Owner]'s litigation position to include bits of data stored on the target machine that indicate the outcome status of executing the source machine instruction on the source machine (*i.e.*, "an IBM mainframe condition code representation on the target machine").

Pet. Reply 26. Patent Owner responds. PO Sur-reply 2–6.

For purposes of this Decision, we determine that no claim term requires express interpretation to resolve any controversy in this proceeding. *See Vivid Techs., Inc. v. Am. Sci. & Eng'g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999).

B. Alleged Obviousness over Loderer

We turn now to Petitioner's asserted grounds, starting with obviousness over Loderer. Petitioner asserts that claims 1, 4, 6, 10, 11, and 16 of the '289 patent would have been obvious over Loderer. Pet. 18–48. Patent Owner disputes certain aspects of Petitioner's analysis. PO Resp. 26–34. For the reasons explained below, we determine that Petitioner has not demonstrated by a preponderance of the evidence that any of claims 1, 4, 6, 10, 11, and 16 of the '289 patent would have been obvious over Loderer.

Before addressing the parties' arguments, we provide an overview of Loderer.

1. Overview of Loderer

Loderer (Ex. 1005) is an English-language translation of European Patent Office Publication No. 0 843 256 A2 (Ex. 1006), which was published in the German language. Ex. 1005, 8 (page providing certification of accuracy of translation).

Loderer describes "a method of simulating a condition code (CC) when porting hardware-specific program code (PCU) from a source hardware (M1) to a target hardware (M2) where no hardware support of a condition code (CC) is provided." Ex. 1005, code (57). Figure 1 of Loderer is reproduced below.

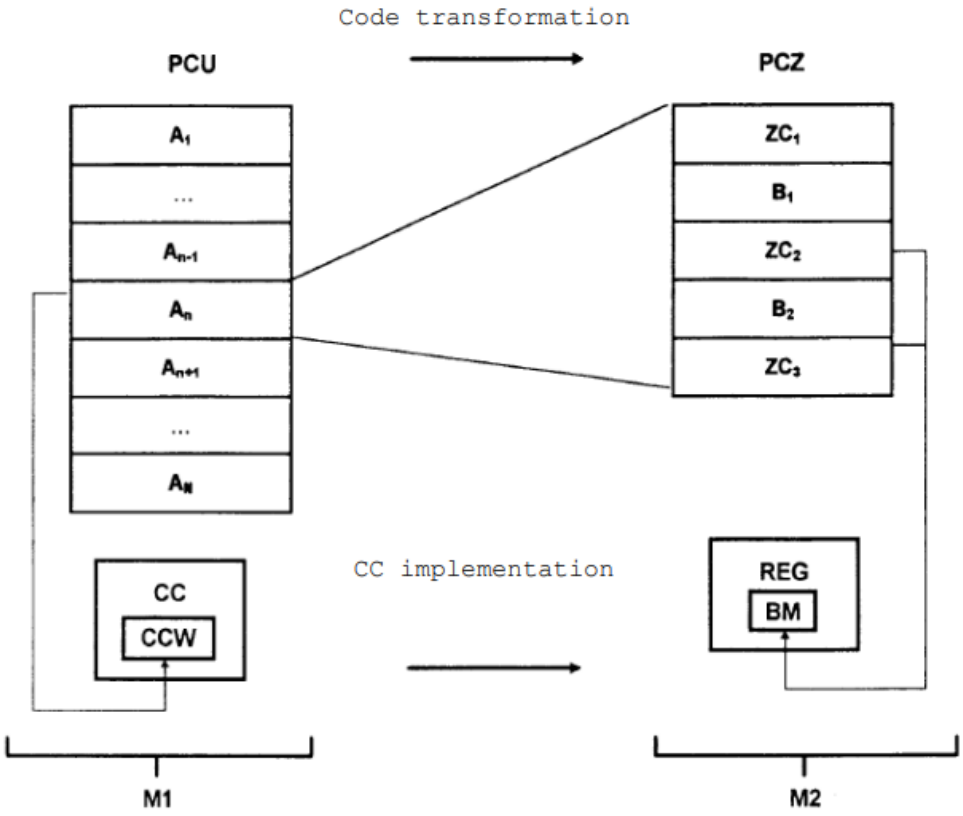

FIG. 1

Figure 1 of Loderer is a schematic diagram showing the basic principle of Loderer's method. *Id.* at col. 4, para. 7. Program code PCU, which includes

individual instructions A_1 through A_N, is to be ported from source hardware M1 to target hardware M2. *Id.* In a code transformation, program code PCU is converted into program code PCZ for target hardware M2, and each of instructions A_1 through A_N is implemented as "a command (or a semantically equivalent command sequence)" in program code PCZ. *Id.* at col. 4, para. 8. For example, instruction A_n in program code PCU is implemented as a command sequence in program code PCZ. *Id.* The command sequence includes two commands B_1 and B_2 for the semantic representation of instruction A_n and additional code having three subsections ZC_1, ZC_2, and ZC_3. *Id.*

Various comparison instructions are available on source hardware M1. Ex. 1005, col. 5, para. 3. These instructions compare first operand OP1 with second operand OP2, and, depending on the result of the comparison, they set value CCW of condition code CC on source hardware M1. *Id.* To illustrate, Figure 2 of Loderer is reproduced below.

Comparison result	CCW	BM
OP1 equal to OP2	0	1000
OP1 < OP2	1	0100
OP1 > OP2	2	0010
Overflow	3	0001

FIG. 2

Figure 2 of Loderer shows the value assignment for the condition code in a comparison command. *Id.* at col. 4, para. 4.

When simulating the condition code on target hardware M2, bit values BM that are on target hardware M2 are uniquely associated with values CCW of condition code CC that are on source hardware M1. Ex. 1005, col. 2, paras. 1–2; *id.* at col. 5, para. 3. In the example of Figure 2,

bit pattern BM is set to 1000 when operands OP1 and OP2 are equal, to 0100 when operand OP1 is less than operand OP2, to 0010 when operand OP1 is greater than operand OP2, and to 0001 when there is a numeric overflow. *Id.*

To further illustrate, Figure 4 of Loderer is reproduced below.

Command sequence in PCZ		a = 0	a > 0	a < 0
ZC_1	Set BM to '0100'	BM = 0100	BM = 0100	BM = 0100
B1	If a = 0, set t_1 = 1, otherwise t_1 = 0	t_1 = 1	t_1 = 0	t_1 = 0
ZC_2	BM << t1	BM = 1000	BM = 0100	BM = 0100
B2	If a > 0, set t_2 = 1, otherwise t_2 = 0	t_2 = 0	t_2 = 1	t_2 = 0
ZC_3	BM >> t_2	BM = 1000	BM = 0010	BM = 0100

FIG. 4

Figure 4 of Loderer shows a specific example of simulating an instruction for a source hardware on a target hardware. Ex. 1005, col. 4, para. 6. In particular, Figure 4 shows a command sequence generated in program code PCZ for target hardware M2. *Id.* at col. 6, para. 2. At the first step in subsection ZC_1, bit pattern BM is set to 0100, where OP1 < OP2 (i.e., a < 0), and therefore is associated with condition code value CCW = 1. *Id.* at col. 6, para. 3. Command B1 checks whether both operands OP1 and OP2 are equal, and then sets register variable t_1 to 1 for equality or to 0 for inequality. *Id.* at col. 6, para. 4. At the next step in subsection ZC_2, shift operator << shifts bit pattern BM to the left by the number of bit positions (0 or 1) stored in register variable t_1. *Id.* Command B2 checks whether operand OP1 is greater than operand OP2, and then sets register variable t_2 to 1 if a > 0 or to 0 if a < 0. *Id.* at col. 6, para. 5. At the final step, in subsection ZC_3, shift operator >> shifts bit pattern BM to the right by the number of bit positions (0 or 1) stored in register variable t_2. *Id.* Loderer explains,

> The bit pattern BM resulting in this step is uniquely associated with the value CCW of the condition code CC occurring after execution of the original comparison instruction on the source hardware M1 and can be evaluated by subsequent instructions in the program code PCZ instead of the original condition code CC of source hardware M1.

Id.

Loderer also notes that its method does not use branching commands. Ex. 1005, code (57), col. 2, para. 1.

2. Claims 1, 4, 6, 10, 11, and 16

We begin with independent claims 1, 11, and 16. Claim 1 recites "obtaining by the target machine a computer instruction from the source machine" and "generating a sequence of target machine instructions which together operate to derive an encoding for a target machine condition code for the computer instruction." The sequence of instructions "provides distinguishing information to distinguish between a plurality of possible outcomes for the target machine condition code." And, the sequence of instructions also "directly calculates the target machine condition code without using branch instructions." Claims 11 and 16 recite these limitations, as well.

The parties dispute whether Loderer teaches the recited "target machine condition code" in particular. Petitioner relies primarily on its discussion of claim 1 for claims 11 and 16. Pet. 44–48. Similarly, Patent Owner relies on the same discussion for claims 1, 11, and 16. PO Resp. 26–34. Accordingly, our analysis applies to claims 1, 11, and 16.

a. Petitioner's Mapping

Petitioner provides an annotated version of Figure 1 of Loderer, which is reproduced below. Pet. 28.

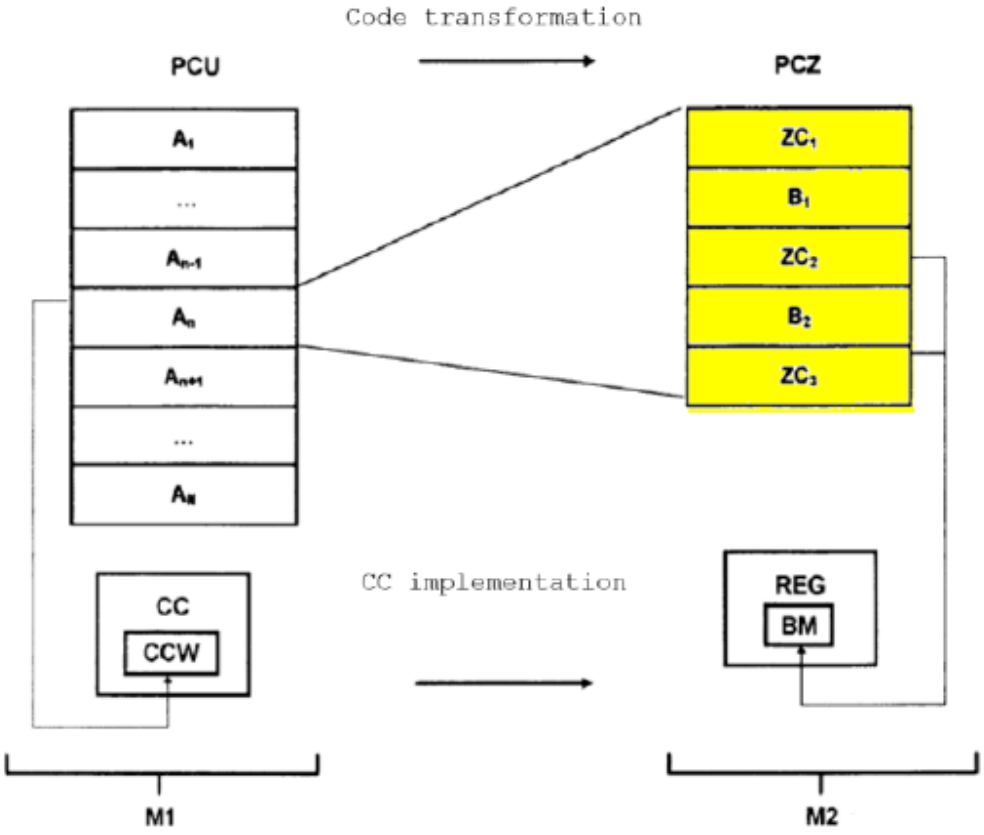

FIG. 1

Figure 1 of Loderer, as annotated by Petitioner, is a schematic diagram showing the basic principle of Loderer's method. *See* Ex. 1005, col. 4, para. 7. Petitioner identifies Loderer's source hardware M1 as the recited source machine and Loderer's target hardware M2 as the recited target machine. Pet. 25–26. Petitioner asserts that "Loderer's Fig. 1 depicts a sequence of target machine instructions (yellow) that derive an encoding for a target machine condition code in register BM for the source instruction A_n." *Id.* at 28 (citing Ex. 1005, col. 4, para. 8).

Petitioner further asserts that the same sequence of instructions is shown in Figure 4 of Loderer. Pet. 28. To illustrate, Petitioner provides an annotated version of Loderer's Figure 4, reproduced below. *Id.* at 29.

	Command sequence in PCZ	a = 0	a > 0	a < 0
ZC_1	Set BM to '0100'	BM = 0100	BM = 0100	BM = 0100
B1	If a = 0, set $t_1 = 1$, otherwise $t_1 = 0$	$t_1 = 1$	$t_1 = 0$	$t_1 = 0$
ZC_2	BM << t1	BM = 1000	BM = 0100	BM = 0100
B2	If a > 0, set $t_2 = 1$, otherwise $t_2 = 0$	$t_2 = 0$	$t_2 = 1$	$t_2 = 0$
ZC_3	BM >> t_2	BM = 1000	BM = 0010	BM = 0100

FIG. 4

Loderer's Figure 4, as annotated by Petitioner, shows a specific example of simulating an instruction for a source hardware on a target hardware. *See* Ex. 1005, col. 4, para. 6. According to Petitioner, Loderer's "target machine instructions illustrated in Figs. 1 and 4 (labeled 'Command sequence in PCZ' [in Figure 4]) operate together to derive a representation for the target machine condition code that is stored in register BM." Pet. 29.

Petitioner asserts that "[t]he values of the BM register are a target encoding of the condition codes because they are a machine representation of the source condition codes that directly map to the corresponding values on the IBM/390 computer (i.e., the source machine)." Pet. 29. Petitioner directs us to where Loderer describes "uniquely associat[ing] each possible value (CCW) of the condition code (CC) on the source hardware (M1) with a bit pattern (BM) on the target hardware (M2)," and providing "a memory area (REG) . . . on the target hardware (M2)" in order "[t]o store the bit pattern (BM) associated with the respective current value (CCW)." Ex. 1005, code (57) (cited by Pet. 29); *see also id.* at Fig. 2 (cited by Pet. 30). Petitioner relies on the declaration testimony of its declarant Mr. Jestice. Pet. 29–30 (citing Ex. 1003 ¶¶ 99–100, 102).

b. The Parties' Dispute

Patent Owner argues that "Petitioner fails to demonstrate how the prior art teaches a 'target machine condition code.'" PO Resp. 26 (capitalization and emphasis omitted). Patent Owner asserts that "Loderer ports program code from a source architecture that supports condition codes (e.g., ESA/390 system[]) to a target architecture 'in which ***no hardware support of a condition code is provided***.'" *Id.* at 29 (quoting Ex. 1005, col. 1, para. 1). Patent Owner points to Loderer's abstract, which says "***[t]he invention*** relates to a method of simulating a condition code (CC) when porting hardware-specific program code (PCU) from a source hardware (M1) ***to a target hardware (M2) where no hardware support of a condition code (CC) is provided***." *Id.* at 28 (quoting Ex. 1005, code 57)). Patent Owner also points to Loderer's teaching in the specification that "***[t]he object of the invention*** is . . . to specify a method for simulating a condition code when porting hardware-specific program code from a source hardware ***to a target hardware that is not providing for condition code***." *Id.* at 27 (quoting Ex. 1005, col. 1, para. 5). Additionally, Patent Owner points to Loderer's claim 1, which recites "[a] method for simulating a condition code (CC), used for the execution of hardware-specific program code (PCU) for a source hardware (M1), in the program code (PCZ), obtained following transformation, for a differing target hardware (M2) ***with a machine architecture which operates without a condition code (CC)***." *Id.* at 28 (quoting Ex. 1005, claim 1).

Patent Owner further asserts that "[b]ecause the target architecture disclosed in Loderer provides no support for a condition code, Loderer teaches instead a 'method for ***simulating*** a condition code," where "a

'mapping is defined which uniquely associates each possible value of the condition code on the source hardware with a bit pattern on the target hardware.'" PO Resp. 32 (quoting Ex. 1005, col. 2, para. 1). According to Patent Owner, Loderer's "source machine condition code[] . . . is distinguished from Loderer's target machine bit pattern." *Id.* Patent Owner adds that even Petitioner's declarant "Mr. Jestice admitted[] Loderer's bit pattern is not a condition code" during his deposition:

> Q. Loderer uses the term "condition code on the source hardware," correct?
> A. Yes.
> Q. Loderer also uses the term "a bit pattern on the target hardware," correct?
> A. Yes.
> . . .
> Q. So you're saying because the bit pattern in the target architecture uses a single bit for its condition, Loderer chooses to call it a bit pattern instead of a condition code?
> A. ***It isn't a condition code. It's a bit pattern.***

Id. at 32–33 (quoting Ex. 2003, 86:5–87:2).

Petitioner replies that "the parties' dispute appears to be whether a 'target machine condition code' includes bits of data stored in a register[7] at

---

[7] The parties argue about the types of registers disclosed in both the '289 patent and Loderer. *See, e.g.*, PO Resp. 29–33; Pet. Reply 4, 8, 10–12, 18–23, 25–26, 28. In its Sur-reply, Patent Owner clarifies that its argument "focuses on the calculation of 'target machine condition codes'—not where they are stored." Sur-reply 18; *see also id.* at 19 ("[Patent Owner's] argument relates to whether the target machine provides a 'condition code;' not whether condition code values are, after being generated, stored in a particular location."). During oral argument, Petitioner's counsel "agree[d] . . . wholeheartedly" that "it doesn't really matter[] . . . [w]hat kind of register there is" because "the claims don't require [a] register." Tr. 27:7–13. Thus, we do not address the parties' arguments regarding registers.

the target machine that represent the outcome status of executing the source machine instruction on the source machine." Pet. Reply 17–18. Petitioner argues that the recited "'[t]arget machine condition code' . . . includes emulated condition code settings on the target machine that would have resulted from execution of the source machine instruction on the source machine." *Id.* at 9. Petitioner asserts, "Though the '289 patent identifies PowerPC and Intel as exemplary target machine architectures . . . , neither the patent nor the [Patent Owner Response] identifies condition codes used by PowerPC or Intel processors for any instructions (including 'Compare Logical' or 'Add Logical')." *Id.* at 9–10 n.4. Petitioner adds that "[t]he '289 patent . . . explicitly states that the disclosed principles for generating the target machine condition code apply to 'any target architecture.'" *Id.* at 27 (citing Ex. 1001, 1:47–49, 3:29–32); *see also id.* at 5 ("The teachings disclosed in the '289 patent are expressly applicable to ***any*** target architecture."). Petitioner appears to take the position that the '289 patent contemplates target architectures that do not support condition codes.

Petitioner further asserts that "[Patent Owner] offers no explanation why Loderer's BM does not satisfy the '289 patent's broad description of condition codes as 'several bits of data to indicate the outcome status of instruction execution.'" Pet. Reply 23 n.11 (quoting Ex. 1001, 1:18–23); *see also id.* at 27 (quoting same). Petitioner points to the prosecution history for the '289 patent, where the applicant explained that "examples provided in the specification . . . describe how to derive an encoding of 0, 1 or 2 (with binary bit patterns of 00, 01, 10, respectively)." *Id.* at 23 n.11 (quoting Ex. 1004, 212 (Response to Office Action)). Petitioner reasons that "[t]he challenged claims recite deriving '***an encoding for a target machine***

condition code,' and—like Loderer—the '289 specification discloses bit patterns stored in a register representing emulated IBM architecture condition codes." *Id.*; *see also id.* at 15 ("Like the '289 patent, Loderer's sequence of instructions generates target machine condition codes corresponding to condition codes as if the instruction had been executed on the source machine—*i.e.*, BM on the target machine representing the IBM/390 condition code CCW in Figure 2 of Loderer.").

Petitioner also points to Patent Owner's positions in parallel litigation. According to Petitioner, "[Patent Owner]'s March 2022 complaint against Petitioner alleges the accused 'target machine condition code' is **a representation of an IBM mainframe condition code** (*i.e.*, a representation of the source machine condition code) generated **on a target machine**." Pet. Reply 24. For example, the complaint says, "Once it receives this [source machine] instruction, the SDM [accused product] generates a sequence of target instructions which together operate to derive **an IBM mainframe condition code representation on the target machine, the 'target machine condition code,'** for a particular IBM mainframe source instruction." *Id.* (quoting Ex. 1027 ¶ 143 (Complaint)); *see also id.* at 24–25 (citing Ex. 1028, 4 (Preliminary Infringement Contentions)). The complaint also says that "the procedure for generating a condition code without branches involves successive updates **to an IBM mainframe condition code representation on the target machine** based on other values such as sign bits and the like." *Id.* at 24 (quoting Ex. 1027 ¶ 144). Referring to the preliminary infringement contentions, Petitioner adds that "[Patent Owner] expressly states it is the 'representation of the

source machine ISA[8] condition code that is referred to here as the "target machine condition code."'" *Id.* at 25 (quoting Ex. 1028, 5). Petitioner asserts that "[Patent Owner] thus alleged infringement of the '289 patent based on generating a representation of IBM mainframe condition codes on the target machine—precisely what the '289 patent discloses as 'target machine condition codes,'" which is "consistent with Petitioner's mapping of 'target machine condition code' in Loderer." *Id.*

In response, Patent Owner reiterates its "argument . . . that Loderer does not satisfy the claimed 'target machine condition code,' because it discloses a target machine that does not provide condition codes." PO Sur-reply 10; *see also id.* at 1–2. Patent Owner again points to testimony from the deposition of Petitioner's declarant Mr. Jestice:

> Q. . . . The invention described in Loderer relates to ***a target hardware that is <u>not</u> providing a condition code, correct?***
> A. ***That's correct.***
> . . .
> Q. So it is your opinion that ***the target machine condition code in Loderer is the bit pattern BM***, correct?
> A. Right.
> . . .
> Q. Loderer uses the term "condition code on the source hardware," correct?
> A. Yes.
> Q. Loderer also uses the term "a bit pattern on the target hardware," correct?
> A. Yes.
> …
> Q. So you're saying because the bit pattern in the target architecture uses a single bit for its condition, Loderer chooses to call it a bit pattern instead of a condition code?
> A. ***It isn't a condition code. It's a bit pattern.***

[8] ISA stands for "Instruction Set Architecture." Ex. 1027 ¶ 34.

PO Sur-reply 7–8 (quoting Ex. 2003, 86:5–87:2, 89:4–7, 99:19–22). Patent Owner acknowledges that "[c]ondition codes . . . can be several bits of data that indicate the outcome status of instruction execution," but contends that "not ***any*** several bits of data that indicate the outcome status of instruction execution are condition codes." *Id.* at 20–21. As support, Patent Owner emphasizes Mr. Jestice's deposition testimony that Loderer's bit pattern is not a condition code. *Id.* at 21. Patent Owner also notes the language in Loderer's specification, which distinguishes a "condition code on the source hardware" from a "bit pattern on the target hardware." *Id.* (citing Ex. 1005, col. 2, para. 2).

Patent Owner additionally points to the '289 patent's teaching that "[i]t is the job of emulation software to accept, as input, strings of source architecture instructions and to ***generate therefrom strings of instructions that, when run on the target architecture, produce the same results***," where "[t]hese results include the ***setting of various condition codes, such as sign, carry, overflow*** and various others indicating exceptions and machine states." PO Sur-reply 13 (quoting Ex. 1001, 1:51–57). According to Patent Owner, this teaching means "the target machine condition codes are set as a result of instructions that—when run in the target machine—emulate, e.g., produce the same results as instructions that would run on the source machine." *Id.*

As to Petitioner's assertion that the '289 patent disclosures are "applicable to ***any*** target architecture" (*see* Pet. Reply 5, 27), Patent Owner counters that "[Petitioner] takes these disclosures out of context." PO Sur-reply 10–11. For example, Patent Owner directs us to where the '289 patent teaches,

> While the systems and methods of the present invention are widely applicable to any emulation method ***where condition codes are present***, it is particularly applicable to the emulation of the z/Architecture. However, the principles set forth herein are applicable to any source architecture and to any target architecture.

Id. at 11 (quoting Ex. 1001, 1:44–49). Patent Owner asserts that "the last sentence, which allegedly forms the basis for [Petitioner's] argument, merely discloses that the invention principles are not limited to the emulation of the *z/Architecture* but are applicable . . . 'where condition codes are present.'" *Id.* (emphasis added).

Patent Owner also directs us to where the '289 patent teaches,

> The present invention provides specific guiding techniques . . . to efficiently ***set conditions codes*** . . . in an emulated binary translation environment for the ***z/Architecture***. These techniques are specifically directed to situations in which the ***PowerPC architecture and the Intel IA32 architectures*** are employed to emulate the z/Architecture. . . . However, it is noted that the princip[le]s, techniques and methods of the present invention are not limited to any particular target machine architecture. The two exemplar architectures discussed herein are merely the most currently ones anticipated to be of the greatest value.

PO Sur-reply 11 (quoting Ex. 1001, 3:20–34). Patent Owner asserts that "the Specification merely discloses that the invention is not limited to the ***PowerPC architecture and the Intel IA32 architectures***." *Id.* at 12. Patent Owner further maintains that "the claims do not cover ***any*** architecture; only those architectures that provide 'target machine condition codes.'" *Id.*

With respect to its litigation positions, Patent Owner explains that "the 'target machine' in the accused products is a target machine that provides condition codes." PO Sur-reply 23. Patent Owner directs us to where its

complaint says "target instructions would provide information distinguishing between different IBM mainframe condition codes to ensure that ***the correct condition code is generated***." *Id.* at 23–24 (quoting Ex. 1027 ¶ 143).

In sum, Petitioner argues that "a target machine condition code is a representation of the source machine ISA [instruction set architecture] or source machine architecture condition code on the target machine," and Patent Owner responds that "whatever the meaning of the target machine condition code is, . . . Loderer does not have it." Tr. 11:5–8, 44:13–17.

c. Analysis

We agree with Patent Owner. On this record, we find that Loderer does not teach or suggest the recited "target machine condition code." Loderer says that "[t]he object of [its] invention is . . . to specify a method for simulating a condition code when porting hardware-specific program code from a source hardware to a *target hardware that is not providing for condition code*." Ex. 1005, col. 1, para. 5 (emphasis added). Loderer's claims are directed to "[a] method for simulating a condition code (CC), used for the execution of hardware-specific program code (PCU) for a source hardware (M1), in the program code (PCZ), obtained following transformation, for a differing *target hardware (M2) with a machine architecture which operates without a condition code (CC).*" *Id.*, claim 1. In other words, Loderer's target machine does not support condition codes, let alone a "target machine condition code." Accordingly, Loderer does not teach or suggest a "target machine condition code."

We note Petitioner's contention that, "like Loderer[,] the '289 specification discloses bit patterns stored in a register representing emulated

IBM architecture condition codes." Pet. Reply 23 n.11. Petitioner relies on the '289 specification's teaching that condition codes include "several bits of data to indicate the outcome status of instruction execution." *Id.* at 23 n.11, 27 (citing Ex. 1001, 1:18–22). Petitioner also relies on the applicant's explanation during prosecution that "examples provided in the ['289] specification . . . describe how to derive an encoding of 0, 1 or 2 (with binary bit patterns of 00, 01, 10, respectively)." *Id.* at 23 n.11 (citing Ex. 1004, 212).

Petitioner's contention is not persuasive. Although Loderer may teach "a bit pattern on the target hardware," Loderer makes clear that "the target hardware . . . is not providing for condition code" and "operates without a condition code." Ex. 1005, col. 1, para. 5; *id.* at col. 2, para. 1; *id.*, claim 1. Loderer's bit pattern on the target hardware therefore is not a condition code, much less a "target machine condition code." *See also* Ex. 2003, 86:5–87:2 (Petitioner's declarant Mr. Jestice testifying during deposition that Loderer's bit pattern is not a condition code).

We further note Petitioner's contention that Patent Owner "alleged infringement of the '289 patent [in parallel litigation] based on generating a representation of IBM mainframe condition codes on the target machine." Pet. Reply 25. This contention also is not persuasive. Patent Owner explains that "the 'target machine' in the accused products is a target machine that provides condition codes." PO Sur-reply 23. In particular, as Petitioner acknowledges, "the procedure for generating a condition code . . . involves successive updates to an IBM mainframe condition code representation on the target machine," where "[v]alues stored in the mainframe condition code representation before the final update are an

'intermediate' condition code value subject to change based on 'distinguishing information' used for final update and determination of the condition code." Ex. 1027 ¶ 144 (cited by Pet. Reply 24). It is our understanding that Patent Owner's position in litigation is that the mainframe condition code representation on the target machine in the accused products therefore is a "target machine condition code." By contrast, Patent Owner's position in this proceeding is that Loderer's target hardware does not support condition codes, and that Loderer's bit pattern on the target hardware therefore is not a "target machine condition code." Patent Owner's litigation position is not inconsistent with its position here. Nor is it inconsistent with our finding that Loderer does not teach or suggest a "target machine condition code."

Lastly, we note Petitioner's contention that the teachings in the '289 patent apply to *any* target architecture, including target architectures that do not support condition codes. Pet. Reply 5, 9–10 n.4, 27. Even if that were true, Petitioner's showing for the claims is still insufficient. *See Baran v. Med. Device Techs., Inc.*, 616 F.3d 1309, 1316 (Fed. Cir. 2010) ("It is not necessary that each claim read on every embodiment."); *TIP Sys., LLC v. Phillips & Brooks/Gladwin, Inc.*, 529 F.3d 1364, 1373 (Fed. Cir. 2008) ("Our precedent is replete with examples of subject matter that is included in the specification, but is not claimed."). The claims recite a "target machine *condition code*" (emphasis added). As discussed above, we find that Loderer's target machine does not support condition codes, let alone a "target machine condition code." Thus, Loderer does not teach or suggest a "target machine condition code."

On the record before us, we are not persuaded that Loderer teaches or suggests the recited "target machine condition code" of claims 1, 11, and 16.

3. Summary

In view of the foregoing, we determine that Petitioner has not demonstrated by a preponderance of the evidence that claims 1, 11, and 16 would have been obvious over Loderer. Because claims 4, 6, and 10 depend from claim 1, we determine that Petitioner also has not demonstrated by a preponderance of the evidence that these dependent claims would have been obvious over Loderer.

C. Alleged Obviousness over Loderer, PoO, and Kane

Petitioner asserts that claims 1–4, 6, 10–12, 16, and 17 of the '289 patent would have been obvious over Loderer, PoO, Kane, and the knowledge of a person of ordinary skill in the art. Pet. 48–76. Each of these claims requires a "target machine condition code." As discussed above with respect to obviousness over Loderer, we are not persuaded that Loderer teaches or suggests a "target machine condition code." Petitioner does not rely on PoO, Kane, or the knowledge of a person of ordinary skill in the art to cure this deficiency. *See id.* at 48–68. Accordingly, we determine that Petitioner has not demonstrated by a preponderance of the evidence that claims 1–4, 6, 10–12, 16, 17 would have been obvious over Loderer, PoO, Kane, and the knowledge of a person of ordinary skill in the art.

IV. CONCLUSION

For the reasons given, Petitioner has not demonstrated by a preponderance of the evidence that claims 1–4, 6, 10–12, 16, and 17 of the '289 patent are unpatentable as follows.

Claims	35 U.S.C. §	References/ Basis	Claims Shown Unpatentable	Claims Not Shown Unpatentable
1, 4, 6, 10, 11, 16	103	Loderer		1, 4, 6, 10, 11, 16
1–4, 6, 10–12, 16, 17	103	Loderer, PoO, Kane, knowledge of a POSITA		1–4, 6, 10–12, 16, 17
Overall Outcome				1–4, 6, 10–12, 16, 17

V. ORDER

In consideration of the foregoing, it is hereby

ORDERED that claims 1–4, 6, 10–12, 16, and 17 of the '289 patent have not been shown to be unpatentable; and

FURTHER ORDERED that, because this is a Final Written Decision, parties to the proceeding seeking judicial review of the decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.

For PETITIONER:

Joseph Gray
Tecuan Flores
SLAYDEN GRUBERT BEARD PLLC
jgray@sgbfirm.com
tflores@sgbfirm.com

Christopher Ryan
RYANIPLAW, PC
chris.ryan@ryaniplaw.com

For PATENT OWNER:

Theodoros Konstantakopoulos
Yung-Hoon Ha
DESMARAIS LLP
tkonstantakopoulos@desmaraisllp.com
yha@desmaraisllp.com