

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

Record of Oral Hearing
Held: March 21, 2024

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

APPEARANCES:

ON BEHALF OF THE PETITIONER:

JOSEPH GRAY, ESQ.
Slayden, Grubert, Beard, PLLC
401 Congress Avenue
Suite 1650
Austin, TX 78701
jgray@sgbfirm.com
(512) 402-3561

CHRIS RYAN, ESQ.
RyanIPLaw, PC
1250 S. Capital of Texas Highway
Bldg. 3, Suite 400
Austin, TX 78746
chris.ryan@ryaniplaw.com
(512) 970-7331

ON BEHALF OF THE PATENT OWNER:

THEODOROS KONSTANTAKOPOULOS, ESQ.
Desmarais, LLP
230 Park Avenue
26th Floor
New York, NY 10169
tkonstantakopoulos@desmaraisllp.com
(212) 351-3400

The above-entitled matter came on for hearing on Thursday, March 21, 2024, commencing at 1:00 p.m., via video teleconference.

P R O C E E D I N G S

- - - - -

JUDGE WORMMEESTER: Good afternoon, everyone. We have our final hearing in IPR2023-00274, LzLabs v. International Business Machines, which concerns U.S. Patent Number 8,713,289. I'm Judge Wormmeester, and with me are Judges Barrett and Peslak.

Thank you for being here today. We want to start off by clarifying a few items. First, our primary concern is your right to be heard. If at any time during the proceeding you encounter technical or other difficulties that undermine your ability to adequately represent your client, please let us know immediately, for example, by contacting the team members who provided you with the connection information.

Second, for the benefit of the Judges, opposing counsel, and court reporter, please identify yourself each time you speak. When not speaking, please mute yourself. Third, we have the entire record, including demonstratives. When referring to demonstratives, papers, or exhibits, please be explicit and identify any slide numbers or page numbers. And finally, please note that members of the public may be listening to this oral hearing. Does anyone have concerns about that?

MR. KONSTANTAKOPOULOS: No, not from Patent Owner, Your Honor.

JUDGE WORMMEESTER: Great. Thank you. All right. I don't hear anyone expressing concerns, so I'll move on. Okay. Let's get the parties' appearances, please. Who do we have for Petitioner?

SPEAKER: This is Charle from the VTC team. It looks like Mr. Joseph Gray's audio dropped.

1 JUDGE WORMMEESTER: Oh, okay.

2 SPEAKER: I'm trying to reach out.

3 JUDGE WORMMEESTER: All right. Why don't we go off the
4 record until counsel gets audio back?

5 SPEAKER: I just advised the counsel to disconnect and reconnect.
6 Thank you. This is Charle from the VTC, still waiting on the counsel to
7 reconnect.

8 JUDGE WORMMEESTER: Okay. Thank you.

9 SPEAKER: We have Mr. Ryan just connected. Thank you.

10 MR. RYAN: Good morning, Your Honor. Mr. Gray will be --

11 JUDGE WORMMEESTER: We're back on the record. Thank
12 you.

13 MR. RYAN: Good morning, Your Honor. Mr. Gray will be
14 arguing, and if he hasn't solved this technical problem in a minute, I will
15 take my computer down to him and let him operate from my machine, which
16 seems to be working.

17 JUDGE WORMMEESTER: Okay. Before we move on, I just
18 wanted to double check. I had noted earlier that members of the public may
19 be listening to this oral hearing. Does counsel for Petitioner have any
20 concerns about that?

21 MR. RYAN: No concerns, Your Honor.

22 JUDGE WORMMEESTER: Okay, great. And so let's formally
23 get the appearances, please. Who do we have for Petitioner?

24 MR. RYAN: For Petitioner, we will have in just a moment Joey
25 Gray, and with him but not arguing is Chris Ryan.

26 JUDGE WORMMEESTER: So we're still waiting.

1 MR. RYAN: Yes. I'm going to take my machine down to Joey
2 right now and let him use it so that we aren't wasting anyone's time.

3 MR. KONSTANTAKOPOULOS: And on behalf of Patent Owner
4 IBM, this is Theodoros Konstantakopoulos, Your Honor.

5 JUDGE WORMMEESTER: Thank you. And is there anyone else
6 on the line for Patent Owner, or is it just one--?

7 MR. KONSTANTAKOPOULOS: I will be arguing, Your Honor.

8 MR. RYAN: Can we mute?

9 MR. GRAY: Hello, guys. I'm here. It's hard to hear us.

10 JUDGE WORMMEESTER: So it looks like we have everyone
11 here. All right. Thank you for introducing yourselves. Welcome. We have
12 set forth the procedure for today's hearing. But just to remind everyone the
13 way this will work, each party will have 60 minutes to present arguments.
14 Petitioner will go first and may reserve rebuttal time. Patent Owner will
15 then present its response and may reserve sur-rebuttal time. Please
16 remember that the demonstratives you submitted are not part of the record.
17 The record of the hearing will be the transcript. We will maintain a clock
18 and try to give you a warning when you're reaching the end of your
19 argument time. Are there any questions before we proceed?

20 MR. GRAY: No, Your Honor.

21 MR. KONSTANTAKOPOULOS: No, Your Honor.

22 JUDGE WORMMEESTER: Okay, counsel. Mr. Gray, thank you.
23 Will you be reserving any time?

24 MR. GRAY: Yes, Your Honor. I'd like to reserve 15 minutes.

25 JUDGE WORMMEESTER: Fifteen minutes? Okay. Let me just
26 set the clock real quick. All right. You may begin when ready.

1 MR. GRAY: Thank you, Your Honor. Do you mind if I share my
2 screen?

3 JUDGE WORMMEESTER: That will be fine. Thank you.

4 MR. GRAY: Thank you. Is everyone able to see that?

5 JUDGE WORMMEESTER: Yes, thanks.

6 MR. GRAY: Yes, thank you. Okay. Good afternoon. My name
7 is Joseph Gray, and I'll be presenting on behalf of Petitioner LzLabs. The
8 Petition in this case raises two grounds. Both grounds involve a reference
9 named as Loderer, the primary reference at issue here. And the difference
10 between the two grounds is that Ground 2 challenges four additional claims.
11 The same claims as Ground 1, plus four more, for a total of 10 challenged
12 claims out of 20 claims of the '289 Patent.

13 IBM raises a single dispute across all the claims in both grounds,
14 and that is whether Loderer discloses a target machine condition code. I just
15 note at the outset that IBM does not dispute the Petition's mapping of the
16 prior art to any other limitation of any of the challenged claims, any other
17 limitation of the independent claim and/or any of the dependent claims. So
18 one dispute here, whether Loderer discloses a target machine condition code.
19 What I'd like to do today is walk through first the Patent and the challenged
20 claims, then discuss Loderer's target machine condition code, and then I'll
21 address IBM's contentions as time permits.

22 So the title of the '289 Patent, the challenged Patent here, is
23 efficiently emulating computer architecture condition code settings without
24 executing branch instructions. It's quite a mouthful, but in general, this
25 Patent relates to emulation, which is, and the Patent refers to a machine for
26 which instructions are designed to be run as the source machine, and the

1 machine running the emulator as the target machine. And the Patent teaches
2 that an important consideration in emulation is how to handle condition code
3 settings. And that's important because condition code generation and
4 functionality is different for different computer architectures. So the
5 machine running the emulator has to have a way to emulate the condition
6 code settings that would have been generated on the source machine that it is
7 emulating. And the '289 Patent purports to invent a way to eliminate -- to
8 emulate condition code settings without branch instructions. And there's no
9 dispute in this case that Loderer does not execute branch instructions, so
10 Loderer avoids branch instructions during emulation like the '289 Patent.
11 And I'll discuss these ideas of emulation, condition code settings, and branch
12 instructions briefly in just a minute. But what we'll see is that Loderer is
13 directed to the same problem as the '289 Patent, avoiding branch instructions
14 when emulating condition code settings, and it gives the same solution to
15 that same problem.

16 Now, the subject matter of this IPR is complicated, to me at least,
17 but the dispute between the parties is not. And what we'll show is that
18 Loderer's target machine condition code is just like the target machine
19 condition code disclosed in the '289 Patent. And for that reason, all
20 challenged claims should be held unpatentable.

21 I'm looking now at slide 4, at the text on the right side of the slide
22 from column 1, lines 42 through 49. We know the invention relates to the
23 handling of condition code settings in the context of the emulation, and it is
24 particularly applicable to the emulation of the z/Architecture. That's the
25 IBM mainframe computer. So we're emulating the IBM mainframe on a
26 computer having a different architecture. So a computer that wouldn't

1 otherwise be able to execute instructions designed for the IBM mainframe
2 can be executed on the target machine in emulation. And as we see at the
3 bottom of this quote, it says-- the Patent says "the principles set forth herein
4 are applicable to any source architecture and to any target architecture."
5 And I'll talk more about that quote and others like it later in this
6 presentation.

7 The '289 Patent describes emulation generally in column 8 and 9
8 and in Figure 5. And what we see in Figure 5 we have a source machine on
9 the right labeled 300 and the target machine on the left labeled 310. The
10 source machine as I mentioned, the example in the specification is the IBM
11 mainframe. So we're talking about the IBM mainframe source machine
12 being emulated on some other type of machine with a different architecture.
13 And the term target machine in this Patent refers to item 310, which includes
14 the emulator program, 320, and the memory. And what we see is that there's
15 an instruction string that would have been executed on the IBM mainframe
16 or the source machine. Those instructions go to the emulator and the
17 emulator outputs its own instruction stream of instructions that can execute
18 on the target machine architecture. And as part of that emulated instruction
19 stream, the emulator is emulating the condition code settings that would
20 have been generated on the IBM mainframe.

21 JUDGE PESLAK: Counsel, this is Judge Peslak. What does
22 emulation mean in the context of this Patent?

23 MR. GRAY: Emulation means attempting to produce the same
24 results on the target machine that would have been obtained on the source
25 machine. So you run a program on the source machine where it's designed,
26 and if you run that same program in emulation on the target machine, the

1 program will operate the same way.

2 JUDGE PESLAK: So it's a result oriented rather than just a
3 translation from one computer language to another?

4 MR. GRAY: That's correct, because in some architectures you
5 won't have the same machine instructions. So let's say there's an IBM
6 machine instruction that doesn't exist on the target machine architecture. So
7 there may be a series of instructions generated through the emulator that
8 would essentially produce the same results as the one IBM instruction would
9 be one example. So it's not a direct translation. The emulator attempts to
10 produce the same results by using instructions of the target architecture. If
11 we look at what a condition code setting is. The Patent uses an example of
12 the compare instruction. So you're comparing the values in two different
13 registers. So register A, register B, you want to compare those two values
14 and the results of that comparison can result in three different what are
15 called conditions.

16 And we see here at the bottom of slide 8, on the compare logical
17 instruction, this is from the IBM mainframe principles of operation. And it
18 defines the condition codes on the IBM mainframe for this compare
19 instruction. And it says you can have a zero, a one, or a two. And zero
20 signifies the operands are equal, one and two are greater than or less than.
21 And we see in column 1, lines 29 through 34, the Patent discusses this
22 example, and it says the, you know, classic example of an instruction that
23 produces condition code changes is a compare instruction which sets a
24 condition to zero if the operands are equal or one or two. And these
25 condition code settings that the Patent is attempting to emulate are the IBM
26 mainframe condition codes we see here at the bottom from Exhibit 1026.

1 Going back to slide 6, I'll just note now I will go out of sequence
2 during this presentation. I apologize for anyone following along on paper,
3 but I will try to say the slide number each time, and I'll also navigate to the
4 correct slide on the screen. On slide 6, so the Patent says the designer of
5 emulation systems is faced with the paradoxical choice of needing branch
6 instructions to successfully emulate the generation of condition code settings
7 on target architectures while at the same time desiring to avoid branch
8 instructions. So we're emulating the generation of condition code settings
9 and avoiding branch instructions. And essentially what branch instructions
10 are is if you imagine a sequence of computer instructions, you know,
11 executing one after the other, a branch instruction causes instructions to be
12 executed out of sequence.

13 So there's a jump to somewhere else in the program code. And the
14 Patent explains that this is inefficient during emulation, so the Patent is
15 trying to avoid those jumps or those branches. And so why do we need to
16 emulate the condition code settings of the source machine, or why do we
17 need to emulate the IBM mainframe condition code settings on the target
18 machine? Well, IBM has told us -- this is from Exhibit 1028, and that's
19 IBM's infringement contentions in parallel litigation. This is on slide 7.
20 And IBM basically says, look, you have to have a representation of the
21 source machine condition code on the target machine so that later target
22 machine instructions -- sorry, so that later instructions from the source
23 machine can utilize those condition codes. Those later instructions may rely
24 on those exact settings, those numbers in this instance, 0, 1, and 2, and the
25 meanings of those numbers. And those condition codes may be different on
26 the target architecture than they were on the source architecture.

1 So what you need is to emulate what would have been generated
2 on that mainframe so that the emulator has access to that so it can faithfully
3 execute the remaining instructions. And this representation of the source
4 machine architecture condition code is what IBM is referring to as the target
5 machine condition code in the litigation. That's precisely our argument in
6 this IPR, that a target machine condition code is a representation of the
7 source machine ISA or source machine architecture condition code on the
8 target machine. So how does the Patent do it? I'm turning to slide 9 now,
9 and there's a lot going on -- on this slide, but I'll start with the box on the
10 left. This is the same box we saw earlier from the principles of operation.
11 These are the IBM mainframe condition codes for the compare instruction,
12 0, 1, 2. And if you look in the table in the middle, we have 0, 1, and 2 in red
13 boxes. That is the --after the sequence of instructions that this Patent teaches
14 -- the result is a zero, a one, or a two, it's held in register rC at the top.

15 You can see that's in a red box as well. So ultimately the goal
16 and the conclusion of the sequence of instructions, they build to the value
17 that's in register rC, which is 0, 1, or 2. In the text above and below this
18 table, you know, it explains that there's a sequence of instructions for
19 emulating condition code settings. And at the end of that sequence, the
20 following results are obtained. That's the 0, 1, 2. And just sort of for
21 orientation of the -- for this figure, you can see that the three conditions
22 shown in the blue box on the left -- equal, greater than, less than -- those are
23 labeled at the very top of the snippet we have here. You have rA, register A,
24 is greater than register B. They're equal, or the first is less than the second.
25 And what this table shows us is that as you execute each instruction, it --
26 well what this table illustrates is the value in each register, if any, as a result

1 of each instruction. So you can follow the sequence down to see how the
2 encoding for the target machine condition code is derived. And I'll talk
3 more about this later, but for now that I would -- the emulated condition
4 code settings in the '289 Patent are the IBM mainframe condition codes.
5 That's what we're talking about is emulating the mainframe condition codes.

6 So if we turn to the claims on slide 12, we'd use Claim 1 as
7 exemplary here. There are three independent claims, 1, 11, and 16. We've
8 mapped in our Petition the correspondence among the different limitations
9 of each of those claims. They're substantially similar. There's no dispute
10 that, you know, that they are substantially similar, so we're just using 1 as
11 exemplary here today. And if we look at the bottom of the claim, we see
12 that what we're ultimately working up to is determining the target machine
13 condition code. And the claim requires that you determine the target
14 machine condition code based on what's known as an intermediate condition
15 code value, and that's in green on slide 12, and based on the distinguishing
16 information, which we've shown here in blue. So distinguishing information
17 and intermediate condition code value are used to generate the target
18 machine condition code. And the Petition maps these claim elements to the
19 embodiment in the Patent, and that's here on slide 15. This mapping is
20 undisputed. IBM does not take issue with our identification of these claim
21 elements, distinguishing information in register rX, an intermediate
22 condition code value in register rC, or the target machine condition code also
23 in rC at the end of the sequence.

24 So these are the claim elements. This is the preferred embodiment.
25 I don't believe there's a dispute about this mapping. And the reason that I
26 don't think there's a dispute is this Petition, sorry, IBM also stated that as

1 much during prosecution. Here on slide 14, from the prosecution history,
2 Exhibit 1004 at 83 to 84, IBM said an intermediate condition code value is
3 determined by performing those first four instructions and it's placed in rC.
4 So you have an intermediate condition code value in rC after instruction
5 four. You have distinguishing information stored in register rX, and then the
6 target machine condition code is determined and stored in rC after the final
7 instruction. So the takeaways from the preferred embodiment here, I think,
8 are at least two, and they're both undisputed. One is the target machine
9 condition code is held in register rC at the end of the sequence. It's stated
10 expressly in prosecution, and it's clear from the specification itself. That's
11 what we're talking about, 0, 1, or 2. And it's held in register rC.

12 And the second point is that what is held in rC is the emulated
13 condition code setting. So that's the condition code from the IBM
14 mainframe that's being emulated on the target machine. So these
15 instructions one through five execute on the target machine, and what they
16 do is they derive an encoding for the IBM mainframe condition code. That's
17 what this Patent is talking about. I'll pause for any questions about the
18 disclosure before I turn to Loderer. Thank you, Your Honors.

19 So Loderer is very similar to the '289 Patent. It uses the IBM
20 mainframe as the source machine that's being emulated. Loderer uses a
21 sequence of instructions to derive the emulated condition code setting
22 without branch instructions, just like the Patent, and we see that on slide 20.
23 We have here it's -- it's Figures 1 and 2 from Loderer. And if we look at the
24 top right, Figure 1, what we see is a target machine M2 on the right. So the
25 target machine is on the right, source machine is on the left. So here we
26 have IBM mainframe on the left, and there's, for example, an instruction

1 labeled An, sort of in the middle of that sequence, and it blows out to the
2 yellow boxes.

3 Those yellow boxes are the sequence of instructions that Loderer
4 uses to generate or to derive the encoding for the IBM mainframe condition
5 code, which it stores in register BM. And it does that without branch
6 instructions. So at the end of that sequence, what we have in register BM is
7 a -- is our bit patterns that correspond to the same condition codes that were
8 being emulated in '289 Patent 0, 1, or 2. So we have an encoding of the
9 target machine condition code in register BM in Loderer.

10 JUDGE WORMMEESTER: Counsel, I'm sorry, did you just say
11 we have an encoding? Can you repeat that, we have an encoding of what?

12 MR. GRAY: Yes. So we have the -- so register BM contains
13 those bits, the bit patterns we see in BM and those bits are encoded to mean
14 the condition codes 0, 1 or 2 that we were talking about emulating.

15 JUDGE WORMMEESTER: Are they condition codes?

16 MR. GRAY: Are the bit patterns?

17 JUDGE WORMMEESTER: The BM, yes.

18 MR. GRAY: They're, well, they're an encoding of it. They
19 represent the number 0, 1 and 2. So you can't store the number 0, 1 or 2 in
20 the register. So Loderer maps a bit pattern encoding of those numbers. So
21 for example --

22 JUDGE WORMMEESTER: That bit pattern is what you're
23 calling, you're mapping to the recited target machine condition code.

24 MR. GRAY: Well, I mean, no, not exactly. So the bit pattern, just
25 like in the in the Patent at 0, 1 or 2. And the prosecution history, which I'll
26 get to here in a few minutes, it's -- or I could jump now if you'd like, but in

1 the prosecution history, IBM explained that the values 0, 1, or 2 are encoded
2 as 2 bit binary bit patterns, or binary bit patterns. So they use a binary bit
3 pattern, 2 bits, to represent the number of 0, 1, and 2. So Loderer has a
4 different encoding of the numbers 0, 1, and 2, but it is an encoding. It's a bit
5 pattern that maps directly to these condition code values that we're
6 attempting to emulate. So the bit pattern in BM is how the target machine in
7 Loderer understands the numbers 0, 1, and 2, or it would interpret the bit
8 pattern to mean 0, 1, or 2.

9 JUDGE WORMMEESTER: Thank you.

10 MR. GRAY: And we explained this is along the same lines of the
11 answer I just gave. At slide 23 in the Petition, we explained that the values
12 of the BM register are a target encoding of the condition codes because they
13 are a machine representation of the source condition codes that directly map
14 to the corresponding values on the IBM mainframe. This is what I was
15 attempting to say maybe less articulate a second ago. But it's a target
16 encoding of the condition codes because it's the encoding the target machine
17 will recognize as representing the mainframe condition codes at 0, 1 and 2.
18 In turn, this is from the Petition on page 29. And our expert, Mr. Ian Jestice,
19 on slide 24, also explained the same. This is from Exhibit 1003, paragraphs
20 76, 98 and 102 that we are describing. Sorry, let me make sure I'm not
21 misquoting (INDISCERNIBLE).

22 The claim says, “generating a sequence of target machine
23 instructions which together operate to derive an encoding for a target
24 machine condition code for the computer instruction.” So Loderer's
25 sequence of instructions together operate to derive an encoding for a target
26 machine condition code. And the target machine condition code here is the

1 representation on the target machine of the condition code that would have
2 been generated on the IBM mainframe. And this is what I mentioned a
3 second ago. And so this slide 27, we knew the claim requires, as I said,
4 generating a sequence of instructions, which together operate to derive an
5 encoding for a target machine condition code. And from the prosecution
6 history, IBM said the examples in the specification describe how to derive
7 an encoding of 0, 1 or 2 with binary bit patterns and it gives those bit
8 patterns, as you'd expect, 00.

9 JUDGE WORMMEESTER: What is the target machine condition
10 code in Loderer?

11 MR. GRAY: It's the exact same target machine condition code as
12 the '289 Patent. It is the emulated IBM mainframe condition codes. That's
13 the purpose of the Patent is to generate on the target machine a
14 representation of the IBM mainframe condition code. And I'll just note, I
15 think where this argument has ended up after reading IBM's sur-reply is that
16 IBM appears to be equating target or is equating target machine condition
17 code with target architecture condition code. Target architecture condition
18 code is something different from an IBM mainframe condition code. But
19 what the Patent teaches, it's not about generating a native target architecture
20 condition code without branch instructions. You're trying to emulate the
21 condition code setting that would have been generated on the IBM
22 mainframe.

23 So the target machine condition code is the target machine
24 representation of the IBM mainframe condition code. And IBM is arguing
25 that the target machine condition code is not the IBM mainframe condition
26 code, it's something native to hardware elements of the target processor, and

1 that is simply divorced from the specification. The example we walked
2 through with the five instructions would make no sense in the context of
3 trying to derive a PowerPC architecture condition code. In fact, the Patent
4 doesn't even talk about the PowerPC condition codes for the compare
5 instruction. It only talks about the IBM mainframe condition codes that are
6 being emulated through the sequence of instructions.

7 JUDGE WORMMEESTER: It does talk about a condition code in
8 the target architecture though, the specification. You had brought us to, I'm
9 on your slide 6, okay, and right around, I guess you have it as column 2. So
10 on your slide 6 at line 38, it says, "This problem is especially severe when
11 condition code generation and functionality in the target architecture are
12 quite different from that found in the architecture of the source machine."
13 So I don't know if it's as divorced from the specification as you're saying.
14 What is your explanation of this here?

15 MR. GRAY: So here it's basically saying, look, if you have
16 different condition codes in the target versus source, you can't just rely on
17 the native hardware condition codes of, sorry, the target, you can't rely on
18 the native target condition codes. So the problem with branching is
19 especially severe because if you do have, you know, hardware elements for
20 condition codes in the target machine, then those might be getting set while
21 you're trying to emulate the source machine. And so what the Patent does is
22 it abstracts this problem away from the hardware, and it makes very clear in
23 several instances that you don't need condition code registers, you don't need
24 target CPU flag bits to be set, that while it's preferable that you use this
25 method to set hardware condition code elements in the target hardware, you
26 don't have to. And IBM's argument is that you do in fact have to set

1 condition codes in the target architecture, in the target hardware, and that in
2 the Patent is it says the opposite. And so --

3 JUDGE WORMMEESTER: But the claim says target machine
4 condition code. So it sounds like it's the condition code in the target
5 machine.

6 MR. GRAY: Let me go to slide 12. It is, Your Honor. It is a
7 condition code in the target machine, on the target machine. But what you're
8 doing -- what -- target machine doesn't mean target architecture. If we look
9 at the obtaining step in this claim on slide 12, this is Claim 1. It's obtaining
10 by the target machine, a computer instruction from the source machine, the
11 source machine having a different architecture from said target machine. So
12 machine does not equal architecture. Machine in this Patent, as we saw in
13 slide 5, the machine includes the emulator and the memory, in the emulated
14 instruction stream. So what Loderer is doing, it does on the target machine,
15 it is emulating the condition code settings of the IBM mainframe. And it's
16 doing that without branch instructions. So there is a target machine
17 condition code because there's an emulated condition code being generated
18 on the target machine without branch instructions.

19 And if we look at slide 9 again, this is the only embodiment that
20 either party has pointed to as representing these challenged claims. And
21 essentially the target architecture condition code argument that IBM is
22 making would mean that the target machine condition code, which we know
23 and it's undisputed, is held in rC, register rC. That that value of two would
24 somehow need to be set in the condition code register of the PowerPC. And
25 to get there, these instructions, rC holds the result of a subtraction, so A
26 minus B, hold the result of the subtraction in the register rC. IBM never

1 addresses what kind of register rC is. They don't take a position. But we
2 know it's separate from the carry bit. The Carry bit's right here in the
3 column 2. So you have rC, register rC, register rX, and the Carry bit. So the
4 Carry bit is there. There is a target architecture condition code in the
5 preferred embodiment. But if we look at the mapping to the claim, which is
6 slide 15, that Carry bit does not correspond to any of the claim elements.

7 What we're calculating here is the IBM mainframe condition code.
8 It's the representation of the IBM mainframe condition code on the target
9 machine, target machine condition code. And frankly, I don't know how
10 with one bit for each condition, so Carry bit or overflow bit, et cetera, in,
11 say, the PowerPC condition code register, which is what the Patent discusses
12 using in this embodiment, how would you encode the number two in those
13 bits in the condition code register? IBM doesn't tell us. And as far as we
14 know, well, let me take a step back. So because the Patent doesn't even tell
15 us what the PowerPC condition codes are for a compare instruction, there's
16 simply no basis for IBM to assume without directly arguing that these values
17 of 0, 1, or 2 are the target architecture condition codes. They're simply not.
18 Because it's clear, as shown here, the bottom of slide 15 from the principles
19 of operation, that these are the IBM mainframe condition codes represented
20 on the target machine. And going back to what IBM is saying in
21 communication.

22 JUDGE PESLAK: Counsel, this is Judge Peslak. How do you
23 reconcile this with your experts' testimony that Loderer doesn't provide
24 condition codes?

25 MR. GRAY: I'll start by saying, I don't believe he said Loderer
26 doesn't provide condition codes. I believe the questions were directed to the

1 hardware or the MIPS processor. So does the MIPS processor provide
2 hardware support for condition codes? No. There's no dispute about that.
3 Or does the MIPS architecture, I'm sorry, does the MIPS processor contain a
4 condition code register? No. There's no dispute about that. But that doesn't
5 matter in the context of these claims. And I'll note that there are claims, 9,
6 15, and I forget the third. We have this in our reply. There are claims in this
7 Patent that require setting CPU target or CPU flag bits. Claim 9 is an
8 example of one of those. And then I'll walk through more of his testimony
9 because it is taken out of context.

10 If we look at slide 47, I'll start with 46. So in the sur-reply, IBM
11 mentions for the first time setting one or more target CPU flag bits. As we
12 understand it, IBM is saying, look, you have to set a target CPU flag bit, or
13 you have to generate a condition code in a condition code register. Either
14 way, we're talking about specialized hardware applications. And if we look
15 at some of the other claims of this Patent, they track the exact language IBM
16 has been trying to rely on for the target machine condition code in Claim 1.
17 Claim 9 requires mimicking an instruction present in a source instruction
18 string in a manner which sets one or more target central processing unit flag
19 bits.

20 So, there are claims that are limited to a target architecture that has
21 flag bits. But the term target machine condition code in the context of this
22 specification cannot mean anything other than a representation of the IBM
23 mainframe condition code on the target machine. That's the only
24 interpretation that to us makes any sense. And what makes this additionally
25 difficult is that IBM doesn't engage with the reasoning of the Petition or our
26 explanation. They simply assume that target machine means target

1 architecture. But they don't try to square that with the disclosed
2 embodiment, because the five instructions we saw are not building up to the
3 setting of flag bits. They're building up to an encoding of a value of 0, 1, or
4 2. And those are IBM mainframe condition codes.

5 So you can use this invention and also set flag bits along the way.
6 But you do not need it. It's not a requirement because what we're talking
7 about in Claim 1 is deriving and encoding for a target machine condition
8 code by using distinguishing information and an intermediate condition code
9 value to generate that target machine condition code. And the target
10 machine condition code is the representation on the target machine of what
11 would have been generated on the IBM mainframe. And I would like to
12 address --

13 JUDGE WORMMEESTER: How is the condition code an
14 emulation of a condition code? Like, how does that work? When you see
15 the word condition code, you say, oh, that's an emulated condition code.

16 MR. GRAY: Well, in the context of this Patent, that's all they're
17 talking about. It's how to -- how to obtain on the target machine a
18 representation of what would have happened on the source machine.
19 Because the instruction --

20 JUDGE WORMMEESTER: And you're getting "representation"
21 from the litigation positions, or from --

22 MR. GRAY: Or encoding. That's what I mean by representation:
23 encoding. I mean, just like if you're going to store the value 2 in a register,
24 typically, you know, it would be one zero. So you don't actually store the
25 number two, you're going to store it in bits and encoded bits. So that's what
26 I'm talking about.

1 JUDGE WORMMEESTER: Okay. When I asked earlier if the bit
2 pattern in Loderer is the target machine condition code, you said no. But it
3 almost sounds like your argument is saying that's what it is.

4 MR. GRAY: Well, I have difficulty with that. And I think this is
5 the sort of ambiguity that IBM is leveraging. Because the target machine
6 condition code, it's bits. I mean, condition code is bits. The Patent tells us
7 its bits of data representing the outcome status of instruction execution. So
8 it's bits. And those bits have to mean something. And what they mean is the
9 IBM mainframe condition code value that's 0, 1, 2. So the bit pattern itself
10 is literally the encoding of those values. So it's the encoded bits that the
11 target machine will recognize as representing 0, 1 or 2. I hope I answered
12 your question, because I feel like this is the encoding or system machine
13 condition code. It can be confusing.

14 JUDGE WORMMEESTER: So if the bit pattern speaks to the
15 encoding part of the claim language, what is the target machine condition
16 code?

17 MR. GRAY: It's the emulated condition code setting. It's the
18 emulated IBM mainframe condition code represented in the bit patterns that
19 Loderer discloses. Just like in the Patent, the encoding is the 0110, the
20 equivalent of the bit pattern in Loderer. And what's being encoded, those
21 bit patterns represent 0, 1 or 2. So Loderer just uses a different manner of
22 encoding the values 0, 1, 2. And this Patent, the challenged claims do not
23 speak to how those values are encoded. It just says --

24 JUDGE WORMMEESTER: Did you explain that in your
25 Petition? Sorry.

26 MR. GRAY: Explain which part? I'm sorry.

1 JUDGE WORMMEESTER: That what the target machine
2 condition code is. Could you show me where that is?

3 MR. GRAY: Yes, look at slide 51. Well, I'll directly answer your
4 question and I'll walk through the testimony explaining the same. So here
5 on slide 51, we have paragraph 102 of Exhibit 1003, and this was cited in the
6 institution decision as well. "The values of the BM register are a target
7 encoding of the condition codes because they are a machine representation
8 of the source condition codes that directly map to the corresponding values."
9 So it's an encoding of 0, 1 or 2.

10 JUDGE WORMMEESTER: The condition codes here – sorry,
11 "the values of the BM register are a target encoding of the condition codes."
12 The condition codes on the source machine or on the target machine?

13 MR. GRAY: I'm sorry, could you repeat that, Your Honor?

14 JUDGE WORMMEESTER: When it says here, paragraph 102,
15 "the values of the BM register are a target encoding of the condition codes."
16 This term here, "condition codes," is that referring to condition codes on the
17 source machine or on the target machine?

18 MR. GRAY: That is referring to a target encoding of the condition
19 codes that would have been generated on the source machine because you
20 see in Figure 3 --

21 JUDGE WORMMEESTER: But so are these condition codes, are
22 they on the source machine or on the target machine?

23 MR. GRAY: Both. So they are in hardware on the source
24 machine, and they are emulated condition code settings on the target
25 machine. So the target machine condition code is the representation of the
26 IBM mainframe condition code on the target machine. But again, we're not

1 talking about a target architecture condition code. If that were the case,
2 there would be no dispute. What we're talking about is a representation on
3 the target machine of what would have happened on the IBM mainframe.
4 That's the whole point of the Patent, is to achieve a representation on the
5 target machine of the IBM mainframe condition codes, the source machine.
6 And I'd like to go back to the claim if I can.

7 JUDGE WORMMEESTER: Okay. I just want to let you know
8 you have about five more minutes before you go into your rebuttal time.

9 MR. GRAY: Okay. Thank you. And, Your Honor, the question a
10 few minutes ago about our expert's testimony about a bit pattern is not a
11 condition code. What our expert actually said, and this is on slide 49, we did
12 this quote at the bottom of the screen on slide 49 in IBM's briefing. But
13 here's the question and answer immediately preceding that. It says, "How is
14 the condition code on the source hardware different from the bit pattern on
15 the target hardware? The bit pattern on the target hardware has a single bit
16 for each condition. The condition code on the source architecture, if it were
17 run, has a condition code.

18 Loderer is describing it in coding the bit pattern, one bit for each
19 condition, into a condition code that . . . the source architecture would have
20 recognized." He corrected himself there, but. So you're encoding a bit
21 pattern into a condition code that the source architecture would have
22 recognized. This has been our position from day one in the Petition. This is
23 how we've mapped the claim. The question is, so you're saying because the
24 bit pattern in the target architecture uses a single bit, it's a bit pattern instead
25 of a condition code, he says it isn't a condition code, it's a bit pattern.

26 And if we look at slide 50, which again, these are the questions and

1 answers immediately before page 99:19 through 22, I believe, which is what
2 IBM cites. This is Mr. Jestice answering this question: “Well, is it the
3 target machine condition code BM? It's a representation of it, yes.” That's
4 the encoding. “What do you mean representation? The target machine
5 condition code will end up in BM, so that is the target condition code.” So
6 we're talking about bits representing something, and the something here is
7 the 0, 1 or 2, the IBM mainframe condition codes. So there's literally bit
8 patterns, those bit patterns correspond to the condition codes we're trying to
9 emulate. And that is the target machine condition code, the representation
10 on the target machine of what the condition code you're trying to emulate.

11 And before I sit down, I'll just go back to the claim. So if we start
12 at the top and just kind of walking down, you obtain a computer instruction
13 from the source machine. And then on the target machine, you generate
14 instructions that derive an encoding for a target machine condition code for
15 the source machine computer instruction. That's what the NSE phases tells
16 us. It's the source machine instruction that you're trying to derive an
17 encoding for a target machine condition code. So you're emulating the
18 condition code that would have been generated by that instruction on the
19 source machine. And so then you get distinguishing information, and then
20 you directly calculate the condition code. It's not a side effect of an
21 instruction like a condition code as IBM defines it. IBM says, look,
22 condition codes are these side effects that get automatically set in hardware
23 as a result of instructions. This Patent is saying, no, no, no, that's not what
24 we're talking about. We're talking about directly calculating an emulated
25 condition code value. And the way you directly calculate it is defined by the
26 claim. You determine an intermediate condition code value. It's a

1 provisional value for the target machine condition code. I'm not aware of
2 any concept of its intermediate --

3 JUDGE WORMMEESTER: It doesn't claim calculating an
4 emulated condition code. It says calculating a condition code.

5 MR. GRAY: That's correct. Sorry. You're right. In the context of
6 this Patent, we don't believe there's any other interpretation. And IBM
7 doesn't dispute what we say about the specification. In fact, IBM's expert
8 doesn't even -- he ignores the intrinsic evidence. He doesn't address the
9 preferred embodiment. He doesn't address register rC. He just says,
10 Loderer doesn't have a condition code register. So there's no possible way it
11 could disclose a target machine condition code. But what he's assuming
12 without establishing is that target machine condition code means target
13 architecture condition code. And that's where the rubber meets the road.
14 That's where IBM is absolutely wrong. In the context of this Patent, we're
15 not directly calculating an intermediate value of a flag bit. There's no such
16 thing as an intermediate value of a flag bit.

17 We're talking about an intermediate condition code value of 0, 1,
18 or 2. And those are the IBM mainframe condition codes. And at the end of
19 the sequence, you've now generated those values of 0, 1, or 2, which are the
20 emulated condition code settings of the source machine that are represented
21 on the target machine binding coding. And I just note that, as I said, IBM
22 and its expert in the POR do not address the preferred embodiment at all.
23 They run from it. They have to run from it because, and they don't contend
24 register rC as a condition code register because they can't. It holds the result
25 of a subtraction. It holds an intermediate value, and it holds the calculated
26 value at the end. None of those things can be done in a condition code

1 register. But IBM, and that's why IBM takes no position on what kind of
2 register it is.

3 JUDGE WORMMEESTER: Okay. Counsel, you're running into
4 your rebuttal time just so you know, but you can keep going.

5 MR. GRAY: Thank you, Your Honor. I'll reserve the rest of my
6 time.

7 JUDGE WORMMEESTER: Just one more thing, actually,
8 though. But you mentioned something about register just now. But the
9 claims don't require register, right? So it doesn't really matter, does it?
10 What kind of register there is or isn't?

11 MR. GRAY: We agree with that wholeheartedly, but IBM's
12 position presumes that register rC is comprised of flag bits or register rC is a
13 condition code register. Because if it's not, we know that the target machine
14 condition code of these claims is not being calculated in flag -- or not being
15 set in flag bits, and it's not being set in a condition code register. And if
16 that's the case, IBM's argument falls apart completely.

17 And so I would invite Your Honors to walk those five instructions,
18 which we've done in our Petition, and ask if it makes any sense that what's
19 being calculated in register rC through those five instructions is a flag bit in
20 a condition code register. It makes no sense.

21 JUDGE WORMMEESTER: Okay. Thank you. You have about
22 14 minutes left in rebuttal.

23 MR. GRAY: Thank you, Your Honor. I'll reserve that time.

24 JUDGE WORMMEESTER: All right. Patent Owner when you're
25 ready.

26 MR. KONSTANTAKOPOULOS: Yeah, I'd also like to reserve 15

1 minutes, Your Honor.

2 JUDGE WORMMEESTER: Okay, great.

3 MR. KONSTANTAKOPOULOS: And may I also share my
4 screen?

5 JUDGE WORMMEESTER: Yes, please.

6 MR. KONSTANTAKOPOULOS: Good afternoon, Your Honors.
7 And again, this is Theodoros Konstantakopoulos on behalf of Patent Owner
8 IBM. Let's jump to slide 3. Now, you heard a lot about the '289 Patent from
9 Petitioner. So I will skip the overview. I just want to address the question
10 about emulation from Judge Peslak. So let's just go to slide 4. So emulation
11 is imitating the behavior of another architecture. So the goal is indeed to get
12 the same results when you execute a program in a target machine. But how
13 the emulation process generates the instructions to get the same results is
14 specific to the particular target architecture. The emulation process will
15 generate different instructions. If, for example, the target machine
16 implements condition codes compared to a target machine that does not
17 implement condition codes.

18 So let's jump to slide 8 and Petitioner's grounds. The Petition
19 presents two grounds, and the primary reference in both is Loderer. The
20 other two references in Ground 2 is an IBM Principles of Operation manual,
21 and Kane, which is a manual of the MIPS RISC Architecture. Slide 9.
22 Loderer relates to a simulation of a condition code during code
23 transformations. And in both grounds, LzLabs uses Loderer for the target
24 machine condition code, and it will be the focus of our discussion.

25 Now let's turn to the disputed issues on slide 11. The disputed
26 issues involve the claimed term target machine condition code and LzLabs'

1 construction introduced in the reply. And in the reply, LzLabs also raised
2 additional arguments that we will also address.

3 Slide 12. Let's turn to the first issue. In the next slide, we will
4 show how Loderer fails to teach or suggest the claimed target machine
5 condition code. Now, let me bring up Claim 1. Target machine condition
6 code is a term recited in the claims, and that's what we have highlighted.

7 Slide 14. Loderer expressly discloses that each method relates to a target
8 hardware where no condition code is provided. For example, you see the
9 specification of the bottom left excerpt discloses a target hardware that is not
10 providing for condition code, and also specifically in Claim 1 at the bottom
11 excerpt on the right, a target hardware with a machine architecture which
12 operates without the condition code. So the architecture in Loderer operates
13 without the condition code. It does not provide condition code in the target
14 architecture. And Mr. Gray tried to distinguish hardware from a machine
15 and then from architecture. So let me try to put things into context.

16 There is a source computer that is a source machine that includes
17 hardware, right, there's hardware components in that machine, and that
18 hardware implements an architecture. The source machine architecture, that
19 computer architecture provides condition codes in both the '289 Patent and
20 in Loderer. But there is also a target computer, that's a target machine that
21 includes hardware that also implements an architecture, a computer
22 architecture. The target machine architecture of the '289 Patent provides
23 condition codes. A target machine architecture of Loderer does not provide
24 condition codes, as we saw expressly discussed in Loderer.

25 And moving on to slide 15. Even Mr. Jestice, LzLabs' expert,
26 admitted that Loderer provides no condition codes, right? We asked him,

1 “The invention described in Loderer relates to a target hardware that is not
2 providing a condition code, correct?” Answer, “That's correct.” This
3 question didn't have anything about MIPS architecture that Mr. Gray
4 mentioned. And on slide 16, our expert confirmed, Loderer's target
5 hardware, the target machine, does not provide a condition code. So both
6 experts admit, and Loderer expressly discloses that the target machine in
7 Loderer does not provide condition codes, but there is also additional
8 corroborating evidence. And that's from the Petition on page 18. LzLabs
9 writes, “Loderer discloses emulation of condition codes for IBM/390
10 instructions on a computer having a MIPS processor.” Now on slide 18, we
11 show an excerpt from Exhibit 2004. Exhibit 2004 includes pages from a
12 seminal textbook on computer architecture. And on the first row of the
13 table, exemplary architectures that provide condition codes include Intel's
14 80x86 architecture, the PowerPC architecture, and also some SPARC
15 machines. On the second row, the MIPS architecture, which is used as the
16 target architecture in Loderer, is an example of a computer architecture that
17 does not provide the condition code.

18 JUDGE WORMMEESTER: I'm sorry, counsel. Can you go back
19 to your slide there on 18? The figure caption says “the major methods for
20 evaluating branch conditions”-- I guess, I know in both the '289 Patent and
21 in Loderer, they're avoiding branch instructions. So how does this play into
22 that? Because this seems to be relevant to branch conditions. Does that
23 have anything to do with--?

24 MR. KONSTANTAKOPOULOS: It does, Your Honor, and that's
25 a great question. So these architectures in the first row, if they're used to
26 emulate another architecture, if the '289 invention is not implemented on

1 these architectures, these architectures will create condition codes that
2 generate branch instructions. So this is consistent with the '289 Patent
3 specifically because the '289 Patent uses the x86 and the PowerPC as
4 examples of target architectures. So, these two target architectures, if you
5 don't implement the '289 invention, will provide condition codes, but also
6 these condition codes will generate branch instructions. What the '289
7 Patent tries to avoid is to calculate the condition codes and therefore
8 avoiding the use of branch instructions in the instructions stream in the
9 source machine, excuse me, in the target machine.

10 JUDGE WORMMEESTER: Thank you.

11 MR. KONSTANTAKOPOULOS: Slide 19, again, this is Kane.
12 This is the reference used in Ground 2. This is Petitioner's Exhibit 1009.
13 Kane is a manual of the MIPS RISC Architecture, and we highlighted on the
14 slide that the MIPS RISC Architecture provides no condition code register.
15 It provides no condition codes like Carry or Overflow. And this is
16 consistent with Exhibit 2004 and with Loderer that discloses a target
17 architecture with no condition codes. Slide 20. On the other hand, IBM's
18 390 architecture is an example of an architecture which includes condition
19 codes. Specifically, IBM's 390 architecture provides a program status word
20 that includes two condition code bits. And the condition codes are shown
21 here and highlighted on slide 20, Your Honors.

22 Now, slide 21, LzLabs' expert, Mr. Jestice, could not map the
23 claimed target condition code to a condition code in Loderer because, as we
24 discussed, there are no target machine condition codes in Loderer. So what
25 did he do? Well, he tried to map it to something else that is not a condition
26 code. And here he admits that he had to map the target machine condition

1 code to a bit pattern. And he actually admitted that the bit pattern is not a
2 condition code. That's what we saw in slide 22. At his deposition, Mr.
3 Jestice confirmed that when Loderer refers to the source hardware, it uses
4 the term condition code. But when Loderer refers to the target architecture,
5 Loderer uses the term bit pattern. And we wanted to explore the reason for
6 this distinction.

7 We wanted to know why Loderer distinguishes condition codes
8 from bit patterns. So we asked Mr. Jestice. And that's what we saw on slide
9 22 that's his deposition testimony. And he says it right there in the last
10 sentence. "It isn't a condition code. It's a bit pattern." He admitted that the
11 bit pattern is not a condition code. And let me actually put up on the screen,
12 Petitioner's slide 50 because I want to address a point that Mr. Gray made.
13 First, a procedural point. There is no citation to a paper in the record on this
14 slide. And it's not that LzLabs had no opportunity to point to this testimony.
15 We filed this deposition transcript with our POR and actually Mr. Gray
16 defended Mr. Jestice. So Mr. Gray was aware of this testimony and could
17 have used this in the reply, but he didn't. In fact, LzLabs did not comment
18 on anything Mr. Jestice said in his declaration or his testimony in their reply.
19 But second, and most importantly on the merits, this testimony actually
20 supports our position. Look at what Mr. Jestice admits. BM, a bit pattern, is
21 a representation of the target machine condition code. Again, a
22 representation of a target machine condition code. The claims, however, do
23 not recite the representation, right?

24 So let's jump to our slide 48. And I'm sorry, I need to go a little
25 out of order to address what Mr. Gray mentioned. Now, even if we're still --

26 JUDGE PESLAK: Counsel, --

1 MR. KONSTANTAKOPOULOS: Yes, Your Honor.

2 JUDGE PESLAK: We have a question. In column 1 of the '289
3 Patent, it's discussing condition codes, and it says, "logical operations and
4 even data transfer operations may result in the generation of several bits of
5 data to indicate the outcome status of instruction execution. These bits are
6 typically referred to as condition codes." So why aren't -- why isn't the bit
7 pattern that Petitioner is relying on in Loderer a condition code?

8 MR. KONSTANTAKOPOULOS: Your Honor, the fact that bit
9 patterns are a representation, right, of a status of an instruction execution
10 doesn't mean that all representation of an instruction execution are bit
11 patterns, are the condition codes. There's a logical fallacy in Mr. Gray's and
12 LzLabs' argument. So there are a lot of things that define a condition code
13 and the plain and ordinary meaning of what someone of skill in the art
14 would understand. And this is something that our expert provided in the
15 technology background and Mr. Jestice did not dispute. The fact that two
16 things can be a representation, a condition code can be a representation,
17 doesn't mean that all representations are condition codes, Your Honor.

18 JUDGE PESLAK: Can you point us to where your expert
19 explained the plain and ordinary meaning of condition code in his
20 declaration?

21 MR. KONSTANTAKOPOULOS: Yes, Your Honor.

22 JUDGE PESLAK: If you don't have it at your fingertips right now
23 just let us know during reply.

24 MR. KONSTANTAKOPOULOS: I can actually use slide 71 of
25 Petitioner's demonstratives. So, our expert provided a technology
26 background section with a discussion about condition codes, and that's what

1 Petitioner's slide 71 shows, paragraph 30 of Exhibit 2001. Mr. Goodin
2 explains that condition codes are special bits that represent hardware state
3 information to hold information about the results of various hardware
4 operations. So, paragraph actually 30 of Mr. Goodin's declaration points to
5 Mr. Jestice's deposition where Mr. Jestice also testified that condition codes
6 are typically implemented in special hardware. That's the last line that you
7 see on the excerpt on slide 71.

8 So Mr. Jestice agrees about the plain and ordinary meaning, or
9 how someone skilled in the art would understand condition code to be the
10 special bit. So again, not every representation would satisfy the plain and
11 ordinary meaning of condition codes, Your Honor, or someone skilled
12 would understand. And that's also relevant to what we show on slide 48,
13 right? Even if we assume, let me go back to our slides, even if we assume
14 that Loderer's bit pattern is a representation of a condition code, a
15 representation or an indication of a condition code is not actually the
16 condition code that the claims require.

17 Let's see what happened during prosecution of the '289 Patent.
18 The excerpt on slide 48 shows a portion of an (INDISCERNIBLE) response.
19 The examiner had rejected the claims over Liu, that's the reference used by
20 the examiner. And IBM explained that among others, Claim 9, that was an
21 independent claim, directly calculates a condition code and distinguished
22 Liu specifically because Liu places indications of carry, sign, overflow, in a
23 register. So IBM distinguished placing indications of a condition in a
24 register, which is exactly what, Your Honors, Loderer is doing. So unless
25 that you need further clarification, I would like to move on to slide 23. Did I
26 answer your question?

1 JUDGE PESLAK: Yes.

2 MR. KONSTANTAKOPOULOS: Thank you. Because another
3 important point about the distinction between a condition code and a bit
4 pattern is the fact that architectures that implement condition codes are
5 fundamentally different from architectures that do not use them like the
6 MIPS architecture. Architectures that use condition codes and architectures
7 that do not or use bit patterns are two incompatible architectures. For
8 example, you cannot modify an architecture that does not implement
9 condition codes to start using condition codes, because a condition code is
10 not something that you can add to a MIPS architecture, you would need to
11 redesign the entire architecture from scraps.

12 Now on slide 24, let me summarize the most important points in
13 the record since institution. Both parties' experts agree that Loderer
14 discloses a target machine that does not provide condition codes. LzLabs'
15 expert confirmed that the Petition mapped the claimed target machine
16 condition code to Loderer's bit pattern. And both parties' experts also agree
17 that Loderer's bit pattern is not a condition code. But in their reply, LzLabs
18 ignores these points. They did not rebut any of these points, Your Honors,
19 and they considered everything that we list on this slide. And this would be
20 fatal to their Petition because that means that their mapping does not work.

21 Now on slide 25, instead of rebutting the expert's testimony of
22 record, in the reply, LzLabs advanced a new argument about the meaning of
23 a target machine condition code. So everything in LzLabs reply is just
24 attorney argument. There is no rebutting in any argument with expert
25 testimony. LzLabs had the opportunity to submit an expert reply declaration
26 to show their expert support their argument, but they did not. Indeed,

1 LzLabs was so keen to keep expert testimony out of the record that they did
2 not even depose IBM's expert. So regardless, as we will explain, these
3 LzLabs new arguments fail as well.

4 Slide 26. Now let's see LzLabs new claim construction proposal
5 we have highlighted at the top. The reply defines target machine condition
6 codes as encoded values stored as bits in a register corresponding to
7 condition codes that would have been generated from executing instructions
8 on the source machine. So essentially, they're trying to expand the meaning
9 of a condition code to be a bit pattern like the one disclosed in Loderer.

10 Slide 27. Before I address the merits, Your Honor, of the
11 construction, I will explain why this construction is untimely. So LzLabs
12 argued in the Petition that terms should be given their ordinary and
13 customary meaning. And you can see that on the top excerpt. And LzLabs
14 actually understood they could only respond to constructions offered by
15 IBM or adopted by the Board. We've highlighted this. And these are the
16 rules in an IPR proceeding. But as you see in the middle excerpt, Your
17 Honors, you concluded that no claim term requires express interpretation.
18 You did not adopt or propose any construction. And in the POR, IBM also
19 applied the plain and ordinary meaning of the claim terms.

20 Slide 28. But in an IPR, the petitioner has the burden from the
21 onset to show with particularity why the patent it challenges is unpatentable,
22 and this is from the Harmonic case. In the reply brief, the Petitioner is
23 permitted to address issues discussed in the institution decision as the trial
24 practice guide notes, but the construction of target machine condition code
25 was not an issue discussed in the institution decision, nor was it an issue in
26 the POR. And as the trial practice guide also notes, Petitioner may not

1 submit new evidence or argument in reply, but it could have presented
2 earlier. And the LzLabs, Your Honors, could have and should have
3 presented this construction in their Petition.

4 So LzLabs' construction and corresponding arguments, they are
5 untimely and therefore improper. But we will show on slide 29 that what
6 they did is they didn't follow the rules, Your Honors. They presented this
7 belated claim construction argument to try to save their case. Respectfully,
8 Your Honors, this isn't something that you should consider because what
9 LzLabs did was not proper, but to the extent you consider the construction, it
10 should also be rejected on the merits.

11 On slide 30, LzLabs' construction is circular. The term condition
12 code is used in the proposed construction of target machine condition code.
13 This is what we underlined in red on slide 30, and in our brief, in our sur-
14 reply, we provided case law and explained that courts have repeatedly
15 criticized circular claim constructions because they do not help to clarify the
16 meaning of the term. And actually courts have found that circular
17 constructions are actually proof that the claim term does not require
18 construction and should be given its plain and ordinary meaning.
19 Additional, on slide 31, we show that both parties' experts agree that target
20 machine condition code requires no construction.

21 Slide 32, that's an important point, so I want to spend some time on
22 this. LzLabs argues that the '289 Patent generates and stores condition codes
23 in register rC. And LzLabs argues that rC is a general purpose register.
24 LzLabs is mistaken, and you can tell, tellingly, the reply does not point to
25 any disclosure in the specification that rC is a general purpose register. The
26 term general purpose register does not appear in the specification, and there

1 is no basis for LzLabs to make this argument. And in fact, the only
2 disclosure and evidence in the specification is that rC holds condition code
3 values. And this is what we underlined in the evidence that LzLabs
4 identified in their reply. rC is specific register for condition codes. So this
5 disclosure in the specification actually undercuts the claim construction
6 argument because this shows that there is a specific register to hold these
7 values, not a general purpose register. And this is a complete distinction
8 from Loderer, where all that is described and discussed is a general purpose
9 register that holds bit patterns.

10 Now, it is clear from the specification that there is no discussion
11 about the general purpose register, LzLabs is trying to make it sound like it
12 is, but they have zero evidentiary support. This is just an argument they
13 have conjured up in the reply to try to save their Petition because, well, they
14 could have their expert, right. Their expert could have testified that rC is a
15 general purpose register -- he did not. And LzLabs could have also asked
16 our expert whether rC is a general purpose register, but they did not take his
17 deposition. They actually did not give you a chance to hear from either
18 expert whether their argument is technically correct. And again, this is a
19 new argument. This is another new argument not presented in the Petition.
20 The Petition does not even include the term general purpose register, let
21 alone an argument.

22 JUDGE WORMMEESTER: Sorry --

23 MR. KONSTANTAKOPOULOS: Yes, Your Honor.

24 JUDGE WORMMEESTER: What is it if rC is a general purpose
25 register, what does that mean for Petitioner's position? Does it mean a bit
26 pattern is a condition code?

1 MR. KONSTANTAKOPOULOS: Well, our position is correct,
2 Your Honors, is that rC cannot be a general purpose register because it holds
3 condition code values. If, yes, general purpose registers, like the ones that
4 are implemented in Loderer hold bit patterns, they do not hold condition
5 codes value. So if rC would, again, there's no evidentiary support, was a
6 general purpose register, it could not hold target machine condition codes.
7 But that's not the case here. The only specific reason that rC is discussed in
8 the specification is to hold condition code values.

9 JUDGE WORMMEESTER: Okay. Thank you.

10 MR. KONSTANTAKOPOULOS: Slide 33. And that's further
11 confirmation from the specification. But rC is a specific register for
12 condition codes. The specification discloses that an instruction executed in a
13 target machine sets the CPU flag bits, and then these results, the condition
14 code results, are placed in register rC. And I want to make two points here.
15 First, a POSITA would have understood there is a distinction between a
16 general purpose register and the condition code register, and the trade-offs of
17 using one over the other. For example, as we discussed with Mr. Jestice
18 during his deposition, Exhibit 1009 describes the benefits of using the
19 general purpose register in the MIPS architecture over condition code
20 registers. But slide 34, second, the second point, and more importantly,
21 IBM's argument is not -- does not rely on whether the conditional codes are
22 stored. Our argument relies on the fact that condition -- that target machine
23 condition codes are generated in the first place. The target machine
24 implements an architecture that provides condition codes. It's not relevant or
25 where these condition codes are stored after being generated.

26 JUDGE WORMMEESTER: Is a flag bit the same thing as a

1 condition code or is it a type of condition code? What's the difference
2 between flag bit and condition code?

3 MR. KONSTANTAKOPOULOS: Flag bits are another term of,
4 excuse me, Your Honor, flag bits are another term in the art used for
5 condition codes. So when you see flag bits, these are condition code bits
6 that are generated and hold some values, and these values can be stored in
7 condition code registers.

8 JUDGE WORMMEESTER: Okay. Thank you.

9 MR. KONSTANTAKOPOULOS: So slide 35. Now, with their
10 proposed construction, LzLabs is trying to equate a condition code with a bit
11 pattern. But, Your Honors, this is not proper. They cannot expand the
12 meaning of a claim term to include something that the claim term is not. A
13 condition code is not a bit pattern, their expert confirmed. So any attempt to
14 expand the meaning of a condition code to merely include a bit pattern
15 should fail.

16 JUDGE WORMMEESTER: I'm sorry, I just want to ask you one
17 thing. If you go back to, exactly, thank you. I know you're pointing to
18 Petitioner's reply at page one, but I think Petitioner's -- if -- to the extent,
19 you know, that the Board determines a construction is necessary -- according
20 to Petitioner -- I'm on page 26 of its reply, and I think Petitioner had a
21 different definition, and it reads a little differently from the page one of the
22 reply. And I think it says "bits of data stored on the target machine that
23 indicate the outcome status of executing the source machine instruction on
24 the source machine, i.e., an IBM mainframe condition code representation
25 on the target machine." Is that different from what you're -- I mean that
26 looks a little different from what you're seeing going on.

1 MR. KONSTANTAKOPOULOS: It does look different, sorry,
2 Your Honor. It does look different, but essentially the same meaning. They
3 want to expand, and first of all this is a definition that you see, i.e., this is
4 how they defined the condition codes, target machine condition codes in
5 page one. You are right, they use a different meaning at the end of page 26,
6 but it's essentially the same thing. They're trying to expand the meaning of a
7 condition code to be bits stored in a register that represents something.
8 That's the main gist. And I mean what clearly they're doing is they're trying
9 to expand the meaning to cover a bit pattern stored in a register. Both
10 definitions or both constructions have this goal and have equivalent
11 meaning.

12 JUDGE WORMMEESTER: Okay. Thank you.

13 MR. KONSTANTAKOPOULOS: I just want to spend some time
14 on some additional arguments that they provided in the reply. So I'll just
15 move to slide 37, and Mr. Gray touched on this, that the fact that they argue
16 that the Patent, the '289 Patent is applicable to any target architecture. Their
17 basis is just three sentences in the specification taken out of context, and
18 that's a screenshot from LzLabs' reply. These are the three sentences.

19 On slide 38, let's see the first sentence in the proper context. You
20 see on the left what LzLabs used? The sentence reads, "the principles set
21 forth herein are applicable to any source architecture" and in highlight "any
22 target architecture." But on the right, we also show the previous sentence in
23 the paragraph. In the proper context, the specification would inform
24 someone skilled in the art that the invention principles are not limited to the
25 emulation of the z/Architecture but are applicable to other architectures
26 where condition codes are present. So condition codes in the emulated

1 method need to be present. And you see “however” links these two
2 sentences, and on slide 39 the second and third sentences are also put in the
3 proper context. In the proper context, the specification merely discloses that
4 the invention is not limited to the PowerPC architecture and the Intel IA32
5 architecture. But the main point and the most important point, even
6 assuming for the sake of argument, that LzLabs' arguments have any merit,
7 it is still inconsistent with the claim language that requires target machine
8 condition codes. That is, the claims do not cover any architecture. Only
9 those architectures that provide target machine condition codes.

10 Another related argument relates to the target machine condition
11 code can be stored in locations other than the hardware or condition code
12 elements. We've touched on this in our brief, Your Honor, and Mr. Gray
13 didn't discuss it.

14 So I will move on to slide 41. LzLabs criticizes our response for
15 using extrinsic evidence, specifically our evidence, including Exhibits 2004
16 and 5. 2004, as we discussed, is a textbook on computer architecture.
17 Exhibit 2005 is a user manual of the Intel architecture processor, which is
18 explicitly referred to as an exemplary target architecture in the '289
19 specification. But most importantly, LzLabs does not dispute the actual
20 evidence in those exhibits.

21 LzLabs does not argue that these exhibits are inconsistent with the
22 intrinsic record because they are not. And the POR also relies on extrinsic
23 evidence that LzLabs introduced into the record. These are Exhibits 2005, 8,
24 9, and 24. So LzLabs relies on those exhibits to support their arguments, but
25 now at the same time complains when disclosures from these exhibits
26 contradict their positions. And that is a double standard, Your Honors. But

1 more importantly. Again, LzLabs does not dispute the actual evidence in
2 those exhibits. They only complain that we use it, but there is nothing that
3 prevents us from relying on extrinsic evidence. Nothing in the extrinsic
4 evidence is inconsistent with the intrinsic record. And their expert has not
5 taken any issue with the way we have used the evidence. Again, this is just
6 another attorney argument.

7 So with regard to the encoding argument, that's another red
8 herring. All encoding does is making sure that the bit, the condition codes
9 are stored in a more efficient manner, and we explain that in our papers. So
10 I just want to move on to the argument about the inconsistency or the alleged
11 inconsistency with our litigation position.

12 Now, again, taking things out of context, LzLabs points to two or
13 three sentences taken out of context from IBM's complaint and preliminary
14 infringement contentions. And LzLabs argues that these documents refer to
15 the target machine condition code as a representation of the source
16 instruction set architecture condition code. And LzLabs now concludes that
17 this for some reason is inconsistent, excuse me, that this interpretation is
18 consistent with Loderer's teaching. Slide 44. But again, there's no basis or
19 support for this conclusion. For example, LzLabs does not provide any
20 evidence that the representation of the source ISA condition code in the
21 target machine is a bit pattern stored in a general purpose register. They
22 don't make that claim. They cannot make that claim. Neither does LzLabs
23 argue that the target machine in IBM's complaint is a target machine that
24 does not provide condition codes. Rather, you see highlighted in the
25 complaint, to perform this emulation, the SDM first obtains by the target
26 machine with an x86 architecture a completed instruction. And as we see on

1 the right, the x86 architecture is an architecture that provides condition
2 codes. So the target machine in the accused products is a target machine that
3 provides condition codes.

4 So I would like to discuss some of the points that Mr. Gray did, so
5 -- raised or included in his presentation. So let me point you to his slide 32.
6 So the first bullet point that we didn't file a preliminary response is true. At
7 the institution decision, Your Honors, you should only hear LzLabs'
8 arguments. What we presented since institution in our POR and sur-reply
9 are arguments that you had not considered for your preliminary
10 determination. But now you have heard our arguments and most
11 importantly, you have heard from Mr. Jestice and his admissions. Now on
12 point number one that the POR did not discuss the specification and register
13 rC, there was no need to discuss the specification or rC. We got clean
14 admissions from Mr. Jestice that Loderer does not disclose a target machine
15 condition code and that a bit pattern is not a condition code. So whatever
16 the meaning of the target machine condition code is, their expert admitted
17 that Loderer does not have it. There's no reason for us to go into the
18 preferred embodiment in the specification or discuss the register rC.

19 Now on the second point about emulating IBM's condition code.
20 Now, when instructions are executed in a target machine that implements
21 condition codes, the execution of the instructions will produce condition
22 codes in the target machine. On the other hand, when instructions are
23 executed in the target machine that does not implement condition codes, the
24 execution of those instructions will not produce condition codes. It is as
25 simple as that. That the instructions emulate the behavior of a source
26 machine is irrelevant.

1 Again, on the third point, about the special bits, as I mentioned, our
2 expert provided an explanation of the plain and ordinary meaning, and Mr.
3 Jestice or LzLabs does not dispute it. And on point number four, LzLabs
4 again ignores their expert's admissions. Loderer does not provide condition
5 codes.

6 So let me move on to slide 10 of their slides, Your Honor. Now on
7 slide 10, LzLabs argues that bits of data that indicate the outcome status of
8 instruction execution are condition codes. And this goes to Judge Peslak's
9 question about this logical fallacy that I explained. That condition codes are
10 bits that indicate the outcome status of instruction execution doesn't mean
11 that all bits that indicate the outcome of instructions are condition codes.
12 For example, the result of an addition instruction indicates that the addition
13 has been completed, but that does not mean that the result of the addition is a
14 condition code. And this point will become more clear with the next slide,
15 slide 42 of Petitioner's demonstratives. And on slide 42, Your Honors,
16 LzLabs accuses us about pivoting in the sur-reply, but it is their reply that
17 introduced all these new arguments, right? Our sur-reply responded to the
18 new arguments about the first point on special bits. Nothing was disputed
19 about special bits. There's no need for us to discuss this. But the second and
20 third point on slide 42 illustrates the problem with LzLabs' arguments which
21 they mapped the target machine condition code in rC as a representation of
22 an IBM mainframe condition code. And they also say that Loderer's BM is
23 a representation of an IBM mainframe condition. And I'll give you an
24 analogy. But Mary is a doctor, and Peter is a doctor, does not mean that
25 Mary is Peter. This is the problem with illogical fallacy. There's so much
26 more that defines a condition code. So --

1 JUDGE WORMMEESTER: What other things define a condition
2 code? If, you know, we have that start on the previous slide that says it's
3 the bits, several bits that indicate, you know, the state of -- the status of an
4 instruction execution.

5 MR. KONSTANTAKOPOULOS: So --

6 JUDGE WORMMEESTER: What else needs to come into play
7 for a code to be a condition code?

8 MR. KONSTANTAKOPOULOS: There's a discussion again in
9 the technology background. These are special bits and what does it mean?
10 Also from Exhibit 2004, which nobody disputes, condition codes are part of
11 the CPU's state. So these are integral part of what a CPU is and understands
12 is of its state. Meaning that whenever execution is -- an instruction is
13 executed in a CPU that implements condition codes, the CPU needs to have
14 these condition codes, also being aware of them so that next instructions can
15 follow these condition codes. So it's part of the state in a non-condition code
16 architecture, the CPU has no idea what the condition code is if they're being
17 set up in execution of an instruction. So that's the most important distinction
18 from Exhibit 2004. So I would direct you to the technology background that
19 again is not being disputed and gives an overview of what condition codes
20 are. Did that answer your question, Your Honor?

21 JUDGE WORMMEESTER: It does. Thank you. And you have
22 about two more minutes before you go into your sur-rebuttal time.

23 MR. KONSTANTAKOPOULOS: Thank you. And then let me
24 just -- let me just go to, and it's fine if I go into my rebuttal time, I just want
25 to hit --

26 JUDGE WORMMEESTER: Yes. Okay.

1 MR. KONSTANTAKOPOULOS: -- a few points. Slide 39, I
2 think this is something that we touched. Slide 39 of Petitioner's exhibits.
3 Again, respectfully, they're playing word games. They're trying to make a
4 distinction between the words hardware and machine. Let me show again
5 our slides -- our slide 14. And let's look, right, at the claim, Claim 1, there is
6 a different target architecture with a machine architecture, which operates
7 without a condition code. So the architecture in the target machine operates
8 without condition codes. So if the target hardware or machine or
9 architecture, however they want to define it in Loderer, implemented
10 condition codes, why would the Loderer need to disclose bit patterns for the
11 target site? Why did the Loderer need to make the distinction about
12 condition code and bit patterns? And on slide 32, Your Honors, that's
13 exactly what Mr. Gray mentioned when you asked him about target
14 architecture.

15 So I just want to direct you to the specification of the '289 Patent,
16 and on column 7, lines 15, which I'm going to pull up, apologies, right. So
17 the specification uses the terms target machine condition code and target
18 architecture condition code interchangeably, right, reading from column 7,
19 lines 15 through 18. "While the above example is specifically directed to
20 the setting of a carry bit, it is equally applicable to other target architecture
21 condition code bits, such as the sign and overflow bits." And, Your Honors,
22 unless you have, sorry, one last point, and that's on slide 64 of Petitioner's
23 demonstratives. Because they actually accuse us, right, of failing to offer a
24 construction that excludes Loderer. As you know, and you noted in the
25 institution decision, a claim term requires construction, construction only to
26 the extent it would resolve a controversy. There is no reason to construe the

1 term target machine condition code. Whatever the meaning, Mr. Jestice
2 admitted that Loderer does not provide it. And in addition, he admitted that
3 the bit pattern is not a target machine condition code. These two admissions
4 should be enough to resolve the controversy.

5 Besides, LzLabs twist things again and inappropriately shifts the
6 burden. To the extent LzLabs believed a claim construction was necessary
7 for their mapping of Loderer's bit pattern to the target machine condition
8 code, they should have done it in their Petition. Now, by introducing the
9 construction in the reply, they did not give us a chance to have our expert
10 opine on this. And more egregiously, they see their expert, right? And by
11 doing so, all they are left is with unsupported attorney argument, Your
12 Honors.

13 And I'll reserve the rest of my time for rebuttal.

14 JUDGE WORMMEESTER: Okay. Thank you. So you'll have
15 about 12 minutes in sur-rebuttal. All right. So, let's see, I'm going to reset it
16 for, I think I said 14 minutes earlier, right? So you can start when you're
17 ready.

18 MR. GRAY: Thank you, Your Honor. I'll share my screen again.
19 I'm going to start here on slide 9 because I think this is the dispositive issue
20 in this IPR. IBM has never once in its papers and/or here today identify
21 what it contends the target machine condition code is in the '289 Patent.
22 We've been clear from day one in the Petition that it is the values of 0, 1, and
23 2 in the register rC. That is the target machine condition code. There can be
24 no dispute. IBM has never disputed, has never addressed it. And IBM's
25 expert doesn't try to square his opinion about target architecture condition
26 codes with what the Patent actually discloses as the target machine condition

1 code. These codes 0, 1, and 2 are the emulated IBM mainframe condition
2 codes. That is how this Patent uses the term, the four words strung together,
3 target machine condition code. It has to be an emulated condition code.
4 That's the only disclosure. So the question to IBM is they agree that 0, 1,
5 and 2 are target machine condition codes, like they said in the prosecution
6 history. Because if that's the case, we're not talking about a target
7 architecture condition code. There is no description in this Patent about
8 taking the value 2 and somehow encoding that into flag bits, like a carry bit
9 or an overflow bit. It encodes the values. And we know that, as in
10 prosecution, they said they're encoding 0, 1 and 2. Zero, 1 and 2 are not
11 PowerPC condition codes. There's no allegation that those are. PowerPC
12 uses CPU flag bits. So

13 there's an enormous disconnect due to IBM's ignoring the
14 specification. IBM takes us to task for not deposing their expert, but it's a
15 waste of time. He ignored the specification. We, from day one, have said
16 ordinary meaning in view of the specification of prosecution history. That's
17 what ordinary meaning is. What IBM has done is taking target machine
18 condition code, taken it out of the specification and said that must mean
19 target architecture condition code. That's how their expert would interpret
20 that in a vacuum. But he doesn't attempt to square it with the only disclosed
21 embodiment of the target machine condition code. That's why we didn't take
22 his deposition. There's no need. His testimony is irrelevant because he
23 doesn't address the most critical evidence, and nor does IBM in its papers,
24 nor did counsel today. Zero, 1, and 2 are the target machine condition
25 codes, and if that's true, we win.

26 Now I'll go through a couple of quotes from the Patent that counsel

1 brought up one of them and tried to say we were taking things out of
2 context, but let's look at what this -- what the Patent actually says. On slide
3 59, column 1, 57 through 62, the full sentence says, it is noted that while an
4 emulation environment preferably results in the setting of hardware or
5 condition code elements in the target architecture, the present invention also
6 contemplates the situation in which condition codes are generated and stored
7 in locations other than the machine code register. You don't have to set
8 hardware elements. You don't have to set condition code elements.
9 Because the Patent is talking about a method of directly calculating an
10 emulated condition code such that you get to the result of 0, 1, or 2. Those
11 are IBM mainframe condition codes, not PowerPC condition codes. Here
12 from column 3, lines 24 through 32, the Patent says these techniques are
13 specifically directed to situations in which PowerPC and Intel architectures
14 emulate the IBM mainframe. However, it's noted that the principles,
15 techniques, and methods of the present invention are not limited to any
16 particular target machine architecture. IBM doesn't address this quote in the
17 papers or today, as far as I know.

18 We've talked about the litigation position. Counsel talked about
19 the x86 processor, how that's what's used and that does have its own
20 condition codes. But those are not relevant. So the point we're making,
21 which is, yeah, the x86 has flag bits. But IBM said, I'm not talking about
22 flag bits. I'm talking about the representation of the source machine, the
23 IBM mainframe condition code. That's what the target machine condition
24 code is. And we are not asking for a construction. We're not proposing new
25 construction, and we're not disputing the term condition code in isolation.
26 The dispute is, is a target machine condition code a hardware condition code

1 of the target architecture, or is it an emulated IBM mainframe condition code
2 represented on the target architecture? And we submit the intrinsic record
3 dictates the latter.

4 Another point IBM made is that we have no support for the fact
5 that register rC is a general purpose register. What you'll note from the
6 briefing is that IBM takes no position on what kind of register r is, register
7 rC. Today he said it's a special purpose register of some sort, presumably
8 limited to storing condition codes or generating condition codes. But if we
9 look at slides 57 and 58, and these are a little bit -- it's complicated because
10 this Patent is complicated. But the very first instruction proves that Register
11 rC is not a dedicated condition code register. Instruction one, you subtract
12 rB from rA, as we see in blue, instruction one at the top, you place that result
13 in rC. That instruction additionally sets the carry bit, which we see in row
14 one of carry, carry label. So after instruction one, you have a value. It could
15 be 10 or 20 or 35. It's a result of the subtraction. So --

16 JUDGE WORMMEESTER: Doesn't -- sorry, counsel. Doesn't the
17 carry imply the rC--? I mean, isn't the carry related to the condition code?
18 But doesn't that imply the rC relates to condition codes?

19 MR. GRAY: Well, here it sets a carry bit, but rC is separate from
20 the carry bit. rC is holding the value of the subtraction, the result. So 10
21 minus 5, rC would hold 5. It's not going to hold a carry bit. It's not going to
22 encode 5 in carry bits. At least the Patent doesn't say so. So in our papers,
23 when we called rC a general purpose register, it's because it's used for
24 general purposes. It holds the results of a subtraction. It then holds the
25 results of an addition in instruction four, and then it holds the results of
26 another subtraction in instruction five. These are calculated values.

1 JUDGE WORMMEESTER: And the Patent never says it's a
2 general purpose register?

3 MR. GRAY: It does not use those words.

4 JUDGE WORMMEESTER: Okay.

5 MR. GRAY: But our position is that the description is undisputed,
6 that these instructions are the variety of subtracts and adds. And there's no
7 intention that those could be done in a condition code register. It's used for
8 general purposes. That's the point we're making. But we don't have to
9 prevail on the fact that it's a general purpose register, however you might
10 define that. All we have to show is that what IBM is saying means that rC
11 has to be a condition code, right? And IBM cannot show that, has not shown
12 that, and their expert doesn't even address this.

13 So I'd like to go back to the claim one last time. So I'm on slide
14 12. And the claim itself tells us that we're not talking about a traditional
15 condition code. We're not trying to induct setting condition code bits in a
16 condition code register. You're trying to derive an encoding for a target
17 machine condition code for the computer instruction, okay? And then it
18 goes on to say you directly calculate that target machine condition code by
19 first determining an intermediate condition code value. There's nothing in
20 the record to suggest that the condition codes that are set in bits as a result of
21 other instructions that there's even a concept of an intermediate condition
22 code value if you're talking about target architecture condition codes. This
23 is talking about an intermediate condition code value that you're trying to
24 calculate from the IBM mainframe. And then at the end, you determine
25 based on those two inputs what the target machine condition code is. You
26 don't determine the target machine condition code as a side effect of

1 executing certain instructions. You calculate it. These instructions work
2 together to ultimately result in a value of zero, one, or two. That's the target
3 machine condition code. If the Board finds that the target machine condition
4 code in this Patent is zero, one, or two, that's exactly what Loderer discloses.
5 And that's what we've been arguing since the beginning. I'll pause for any
6 questions. Okay.

7 I would just say at the end of the day, the dispute is not over what
8 the meaning of a condition code is. IBM has tried to steer the dispute that
9 way, but that's not at all what it is. The dispute is whether target machine
10 equals target -- target machine condition code equals target architecture
11 condition code. And if you look at the Patent, what's in rC at the end of the
12 sequence, it does not hold a target architecture condition code. It holds an
13 emulated IBM mainframe condition code. That's dispositive. Thank you,
14 Your Honors.

15 JUDGE WORMMEESTER: Thank you.

16 MR. KONSTANTAKOPOULOS: So let me just start with what
17 Mr. Gray concluded.

18 JUDGE WORMMEESTER: Let me set the timer real quick.
19 Sorry about that. So you'll have 12 minutes, and you can start when ready.

20 MR. KONSTANTAKOPOULOS: Thank you. And as I said, I'm
21 going to start with how Mr. Gray concluded. He said that it's not a question
22 of what a condition code is and what it means. We agree. Whatever it
23 means, whatever the plain and ordinary meaning is, whatever the special
24 meaning is, their expert admitted, the target machine condition code is
25 something that he could not find in Loderer, right? It's as plain as that. See,
26 Mr. Gray also said that the issue is whether there's a distinction between

1 target machine condition code and target architecture condition code. Well,
2 I just showed you in the specification, right? Column 7, line 15. Target
3 architecture condition code is used in the specification, and it has the same
4 meaning as target machine condition code. But can I please share my screen
5 again, because I want to make one final point --

6 JUDGE WORMMEESTER: Of course.

7 MR. KONSTANTAKOPOULOS: -- about the encoding that Mr.
8 Gray discussed both in his opening and now in his rebuttal. And I'm
9 showing slide 20 of Petitioner's demonstratives, because I believe Mr. Gray
10 misspoke, and I would like to clarify the record. When discussing slide 20,
11 Mr. Gray stated that BM is an encoding of the target machine condition
12 code. And, Your Honor, you asked him, and I think then he changed his
13 testimony, and he said that BM is both. BM is both an encoding of the
14 target machine condition code, and it's also an encoding of the condition
15 code of the source machine. Well, it is certainly not an encoding of the
16 target machine condition code.

17 You see CCW is the condition code in the source architecture. At
18 best, BM is an encoding of the source machine condition code. It's not an
19 encoding of the target machine condition code and Claim 1 requires an
20 encoding of a target machine condition code. That's what Claim 1 requires.
21 Claim 1 does not require an encoding of a source machine condition code
22 which Loderer describes, and this goes back to our argument. There's no
23 such thing as a target machine condition code, and therefore there cannot be
24 an encoding of something that is not present in the target machine in
25 Loderer. And, Your Honors, we heard about the register that Mr. Gray
26 mentioned. Your Honors, and specifically, Judge Wormmeester, mentioned

1 that the claims do not recite a register, let alone a general purpose register.
2 That's not even in the specification. But what LzLabs is struggling to do is
3 they're struggling to incorporate the register limitation into the claims. And
4 in fact, they're trying to incorporate a specific type of register.

5 Well, a general purpose register that is not discussed anywhere,
6 and we think that this is just an attempt to distract you from the fact that
7 whatever they argued in the Petition has failed. Anything that you've heard
8 about registers and general purpose registers at rC is what they argued in
9 their reply, right? And again, if there was any merit in LzLabs' argument
10 that bit pattern is the same as a target machine condition code, neither
11 Loderer is disclosed, nor Mr. Jestice would need to distinguish that to a bit
12 pattern stored in a general purpose register is not a target machine condition
13 code. And Mr. Gray said that we didn't take a position about rC in our
14 papers, but that's clearly not true. It was their job, I mean, they made the
15 contention that rC is a general purpose register in the reply, but it was their
16 burden to prove this point.

17 All we did in our sur-reply is we explained how they failed to
18 provide any evidentiary support. They could have had their expert state and
19 be cross-examined about what type of register is rC. Their expert did not
20 file a reply declaration. They could have asked our expert, is rC a general
21 purpose register? They said -- Mr. Gray said there was no purpose, because
22 his declaration had nothing to do with the point that he wanted to bring up.
23 Well, he could still ask this question, right? And he also accused our expert,
24 but he didn't address whether rC is a general purpose register or not.

25 Your Honors, they made this argument in the reply. We had no
26 opportunity to file a sur-reply declaration, right? They didn't make this

1 argument in the Petition where our expert could actually address it. They
2 cannot now accuse us that our expert did not address rC and whether it is a
3 general purpose register or a register that only holds condition code values
4 when our expert could not file a reply declaration.

5 And, Your Honors, unless you have questions, I have no further
6 comments.

7 JUDGE WORMMEESTER: Let me double check. Does anyone
8 else have any questions? Well, I don't have any other questions. Thank you
9 to both parties for your time. That was very helpful. The case is submitted,
10 and the hearing is adjourned.

11 (Whereupon, the above-entitled matter went off the record at 3:03
12 p.m.)

IPR2023-00274
Patent 8,713,289

PETITIONER:

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

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

PATENT OWNER:

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