

**UNITED STATES PATENT AND TRADEMARK OFFICE**

---

**BEFORE THE BOARD OF PATENT APPEALS  
AND INTERFERENCES**

---

*Ex parte* KENT G. FIELDEN

---

Appeal No. 2002-0490  
Application No. 09/152,751

---

ON BRIEF

---

Before THOMAS, BARRETT, and BARRY, *Administrative Patent Judges*.  
BARRY, *Administrative Patent Judge*.

DECISION ON APPEAL

A patent examiner rejected claims 16-27. The appellant appeals therefrom under 35 U.S.C. § 134(a). We reverse.

BACKGROUND

The invention at issue on appeal executes "load instructions." (Spec. at 1.) A microprocessor executes a load instruction to retrieve data from a memory. Such an instruction specifies an address at which data of interest are stored and how much data are to be retrieved. (*Id.* at 2.)

To execute a computer program more quickly, an instruction may be moved "up" to a place earlier in the program's execution. In particular, the appellant explains, "it is useful to move instructions to a location prior to branch instructions, the execution of which determine whether the moved instructions would have been executed." (*Id.* at 1.) Once moved, such an instruction will be executed "speculatively," i.e., before the branch instruction is executed to determine whether the moved instruction should be executed. (*Id.*)

An exception<sup>1</sup> may arise when executing a load instruction in a computer program. (*Id.* at 2.) Although executing instructions speculatively can be desirable, the appellant cautions, "it is important not to take exceptions on such instructions where an instruction potentially may not actually be executed." (*Id.* at 1-2.)

Accordingly, the appellant's invention replaces a load instruction in a computer program with a "dismissible load instruction" and a "check instruction." The dismissible load instruction, which, "by definition, does not generate an exception," (*id.* at 6), is positioned in the program to be executed speculatively. (Appeal Br. at 3.) The check

---

<sup>1</sup>An exception is "a problem or change in conditions that causes a computer's microprocessor to stop what it is doing and then find and carry out . . . instructions in a separate routine designed to handle the situation." *Microsoft Press Computer Dictionary* 153 (2d. ed. 1994).

instruction is positioned in the program at the original location of the (replaced) load instruction. (*Id.*) At the location in the program where the load instruction was originally placed, a check is made to determine whether an exception should have occurred. If so, the exception is taken. (Spec. at 6.) A further understanding of the invention can be achieved by reading the following claim.

16. A method of compiling a series of instructions having a load, said method comprising:

converting the load into a dismissible load instruction;

positioning the dismissible load instruction in a stream of executable instructions so that the load would be executed speculatively;  
and

positioning a check instruction after the dismissible load instruction in the stream of executable instructions to determine if an exception should have occurred on the load.

Claims 16-18, 20-22, 24-25 and 27 stand rejected under 35 U.S.C. § 102(b) as anticipated by Todd C. Mowry, *Tolerating Latency Through Software-Controlled Data Prefetching* (Mar. 1994) ("Mowry"). Claims 19 and 23 stand rejected under 35 U.S.C. § 103(a) as obvious over Mowry and Anne Rogers and Kai Li, *Software Support for Speculative Loads*, Proceedings of 5th Int'l. Conf. Architectural Support for Programming Languages and Operating Sys., pp. 38-50 (Oct. 1992) ("Rogers").

Claim 26 stands rejected under § 103(a) as obvious over Mowry and the appellant's admitted prior art ("AAPA").

## OPINION

Our opinion addresses the rejections in the following order:

- anticipation rejection of claims 16-18, 20-22, 24, 25 and 27
- obviousness rejection of claims 19, 23, and 26.

### *Anticipation Rejection of Claims 16-18, 20-22, 24, 25 and 27*

Rather than reiterate the positions of the examiner or the appellant *in toto*, we address the main point of contention therebetween. The examiner asserts, "Mowry taught . . . placing a check instruction after the dismissible load (page 8, 3rd full para.; page 7, figure 1.4; page 129, section titled Summary)." (Examiner's Answer at 3.) The appellant "submits that the unmodified load instruction of Mowry does not perform the same function as the check instruction put in place of the converted load instruction in Appellant's claimed invention." (Appeal Br. at 6.) The examiner answers, "Appellant arguments are directed unclaimed elements, nowhere in the claims are a separate check instruction recited or checking the validity of the data." (Examiner's Answer at 7.)

"Analysis begins with a key legal question -- *what is the invention claimed?*"

*Panduit Corp. v. Dennison Mfg. Co.*, 810 F.2d 1561, 1567, 1 USPQ2d 1593, 1597 (Fed.

Cir. 1987). "In construing claims, the analytical focus must begin and remain centered on the language of the claims themselves. . . ." *Interactive Gift Express, Inc. v. Compuserve, Inc.*, 256 F.3d 1323, 1331, 59 USPQ2d 1401, 1406 (Fed. Cir. 2001) (citing 35 U.S.C. § 112, ¶2).

Here, independent claim 16 specifies in pertinent part the following limitations: "positioning a check instruction after the dismissible load instruction in the stream of executable instructions to determine if an exception should have occurred on the load." Independent claims 20 and 24 specify in pertinent part similar limitations. Focussing on the language of the claims, we join the appellant in being "puzzled at the assertion in the Examiner's Answer that the check instruction of Appellant's invention as claimed is actually an unclaimed element." (Reply Br. at 4.) To the contrary, the limitations require positioning a check instruction after a dismissible load instruction in a stream of executable instructions and using the check instruction to determine if an exception should have occurred on an associated load.

"Having construed the claim limitations at issue, we now compare the claims to the prior art to determine if the prior art anticipates those claims." *In re Cruciferous Sprout Litig.*, 301 F.3d 1343, 1349, 64 USPQ2d 1202, 1206 (Fed. Cir. 2002). "A claim is anticipated only if each and every element as set forth in the claim is found, either

expressly or inherently described, in a single prior art reference." *Verdegaal Bros., Inc. v. Union Oil Co.*, 814 F.2d 628, 631, 2 USPQ2d 1051, 1053 (Fed. Cir. 1987) (citing *Structural Rubber Prods. Co. v. Park Rubber Co.*, 749 F.2d 707, 715, 223 USPQ 1264, 1270 (Fed. Cir. 1984); *Connell v. Sears, Roebuck & Co.*, 722 F.2d 1542, 1548, 220 USPQ 193, 198 (Fed. Cir. 1983); *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 771, 218 USPQ 781, 789 (Fed. Cir. 1983)). "[A]bsence from the reference of any claimed element negates anticipation." *Kloster Speedsteel AB v. Crucible, Inc.*, 793 F.2d 1565, 1571, 230 USPQ 81, 84 (Fed. Cir. 1986).

Here, we find no teaching of positioning a check instruction after a dismissible load instruction in a stream of executable instructions let alone using the check instruction to determine if an exception should have occurred on an associated load in the three sections of Mowry cited by the examiner. Figure 1.4 of the reference merely shows "how prefetching improves performance." P. 7. Its third full paragraph discusses "how far loads can be moved ahead of their uses." P. 8. For its part, the Summary section of Mowry recapitulates that "deciding when to drop prefetches is a complex issue." P. 129.

The absence of positioning a check instruction after a dismissible load instruction in a stream of executable instructions and using the check instruction to determine if an

exception should have occurred on an associated load negates anticipation. Therefore, we reverse the anticipation rejection of claim 16; of claims 17 and 18, which depend therefrom; of claim 20; of claims 21 and 22, which depend therefrom; of claim 24; and of claims 25 and 27, which depend therefrom.

*Obviousness Rejection of Claims 19, 23, and 26*

"In rejecting claims under 35 U.S.C. Section 103, the examiner bears the initial burden of presenting a *prima facie* case of obviousness." *In re Rijckaert*, 9 F.3d 1531, 1532, 28 USPQ2d 1955, 1956 (Fed. Cir. 1993) (citing *In re Oetiker*, 977 F.2d 1443, 1445, 24 USPQ2d 1443, 1444 (Fed. Cir. 1992)). "A *prima facie* case of obviousness is established when the teachings from the prior art itself would . . . have suggested the claimed subject matter to a person of ordinary skill in the art." *In re Bell*, 991 F.2d 781, 783, 26 USPQ2d 1529, 1531 (Fed. Cir. 1993) (quoting *In re Rinehart*, 531 F.2d 1048, 1051, 189 USPQ 143, 147 (CCPA 1976)).

Here, the examiner fails to allege, let alone show, that the addition of Rogers or AAPA cures the aforementioned deficiency of Mowry. Absent a teaching or suggestion of positioning a check instruction after a dismissible load instruction in a stream of executable instructions and using the check instruction to determine if an exception should have occurred on an associated load negates anticipation, we are unpersuaded

of a *prima facie* case of obviousness. Therefore, we reverse the obviousness rejection of claims 19, 23, and 26.

#### CONCLUSION

In summary, the rejection of claims 16-18, 20-22, 24-25 and 27 under § 102(b) is reversed. The rejection of claims 19, 23, and 26 under § 103(a) is likewise reversed.

REVERSED

JAMES D. THOMAS  
Administrative Patent Judge

LEE E. BARRETT  
Administrative Patent Judge

LANCE LEONARD BARRY  
Administrative Patent Judge

)  
)  
)  
)  
)  
) BOARD OF PATENT  
) APPEALS  
) AND  
) INTERFERENCES  
)  
)  
)  
)

Appeal No. 2002-0490  
Application No. 09/152,751

Page 10

BLAKELY SOKOLOFF TAYLOR ZAFMAN  
12400 WILSHIRE BOULEVARD 7TH FLOOR  
LOS ANGELES, CA90025