EDITOR: Joshua J. Yi, josh@joshuayipatentlaw.com

DEPARTMENT: MICRO LAW

A Review of *Wisconsin Alumni Research Foundation v. Apple*—Part IX

Joshua J. Yi, *The Law Office of Joshua J. Yi, PLLC, Austin, TX, 78750, USA*

Part I of this series introduced the *Wisconsin Alumni Research Foundation v. Apple* cases and described the asserted patent (U.S. Patent No. 5,781,752). That article also summarized some recent large verdicts for patents asserted by academic institutions and provided several reasons why this series may be of interest to the readership of *IEEE Micro*, most notably because the inventors are well known and several well-known computer architects worked as experts on this case. Part II described the complaints, namely, it described the plaintiff, Wisconsin Alumni Research Foundation ("WARF"), the inventors, and WARF's allegations as to how the Apple's products infringed WARF's patent. Part III described Apple's answer to the allegations in WARF's complaint, Apple's counterclaims, and WARF's response to those counterclaims. Part IV examined Apple's allegation of inequitable conduct by the inventors, a technical analysis of that allegation, and Judge William M. Conley's legal analysis of the sufficiency of Apple's allegations. Parts V, VI, and VII reviewed three of WARF's motions to compel Apple to produce 1) executable versions of the simulators for its A6, A7, and A8 processors (hereinafter "accused products"), 2) documentation regarding future (unreleased) processors, and 3) a version of each accused processor that had the accused feature disabled, and the hearings that the court had on those motions. Part VIII described what claim construction is and what claim terms were disputed both in these cases and in the prior *Intel* case.

DISPUTED CLAIM TERMS IN THE *WARF* v. *INTEL* CASE

The purpose of the claim construction is to interpret the meaning of claim terms, which helps define the scope of the asserted claims. Based on that scope, the parties can prepare their arguments so the jury (or the court in some cases) can determine if the accused products infringe the asserted claims and/or are invalid.

Prior to suing Apple, WARF sued Intel for alleged infringement of the '752 Patent. While different judges presided over each case—Judge Barbara B. Crabb presided over the *Intel* case and both Judge Crabb and Judge Conley presiding over the *Apple* case—Judge Crabb's claim construction decisions from the *Intel* case are still relevant to the *Apple* case as the parties largely adopted her claim constructions for the Apple case, with only one exception. Furthermore, the parties' arguments regarding the meaning of the disputed claim terms in the *Intel* case are interesting from a technical perspective as it describes the parties' technical arguments for each term. Table 1 lists the claim terms that were at-issue in the *Intel* case.

Of the nine terms in Table 1, Judge Crabb decided that, based on "the parties' arguments at the hearing, their pre-hearing briefs, the patent claims, patent specification and prosecution history," that she only needed to rule on five of the nine terms in order to "resolve the parties' disputes" (see pages 1–2 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[1]). In particular, she only construed Terms #1 ("in fact executed"), #2 ("data speculation circuit"), #3 ("mis-speculation"), #5 ("predictor"), and #7 ("prediction"). In the subsequent *Apple* case, the parties adopted the *Intel* court's constructions for all but "prediction." For that term, the parties proposed slightly different constructions from what Judge Crabb decided in the *Intel* case. This article series will focus only on the terms that Judge Crabb ruled on.

ORDER AND TIMING OF THE CLAIM CONSTRUCTION BRIEFS

The parties simultaneously filed their opening claim construction briefs in the *Intel* case on 26–27 June 2008 and their responsive claim construction briefs

Digital Object Identifier 10.1109/MM.2026.3676479
Date of current version 23 April 2026.

TABLE 1. List of disputed claim terms in the *Intel* Case, and the parties' competing constructions.

Term	WARF's Proposed Construction	Intel's Proposed Construction
#1: in fact executed	a LOAD instruction is "in fact executed" before the STORE instruction when the LOAD instruction has actually accessed or was certain to access data that has not yet been updated by the STORE instruction	a load instruction is "in fact executed" when the load instruction actually has loaded data from a memory location
#2: data speculation circuit	a circuit that detects data dependence between LOAD and STORE instructions and tracks execution of such instructions in order to detect any mis-speculations arising from the data speculative execution of LOAD instructions	a circuit that detects data dependence between load/store pairs and detects a mis-speculation by a load instruction that is in fact executed
#3: mis-speculation	where a LOAD instruction, that is dependent for its data on a STORE instruction appearing earlier in program order, is in fact executed before the STORE instruction	where an instruction has been in fact executed prematurely and erroneously
#4: where a data consuming instruction … is in fact executed before the data producing instruction	where a LOAD instruction, that is dependent for its data on a STORE instruction appearing earlier in program order, has actually accessed or was certain to access data that has not yet been updated by the STORE instruction	where a load instruction that depends on a store instruction has actually loaded data from a memory location before the store instruction has put data into that same memory location
#5: predictor	a circuit that receives a mis-speculation indication from the data speculation circuit to produce a prediction	a circuit that receives a mis-speculation indication from the data speculation circuit to produce a prediction based on historical mis-speculation indications
#6: mis-speculation indication	an indication that the data speculative execution for a LOAD instruction was incorrect.	an indication that an instruction has been executed prematurely and erroneously
#7: prediction	a dynamic multibit value which indicates the likelihood that the data speculative execution of a LOAD instruction will result in a mis-speculation	a value indicating the likelihood that data speculative execution of a load/store pair will result in a mis-speculation
#8: prediction associated with the particular data consuming instruction	a dynamic multibit value associated with the particular LOAD instruction which indicates the likelihood that the data speculative execution of that LOAD instruction will result in a mis-speculation, and which is based on historical mis-speculation indications	a value associated with the particular load instruction that indicates the likelihood that data speculative execution of a load/store pair will result in a mis-speculation, and which is based on mis-speculation indications
#9: a prediction threshold detector	a circuit that prevents data speculation for a particular LOAD instruction when the prediction associated with said LOAD instruction is in a predetermined range	a circuit that prevents data speculation of a load/store pair if the prediction value for the pair is within a predetermined series of multiple values

on 24 July 2008. When parties file their briefs simultaneously, that means both sides file their briefs on the same day, without knowing what the other side will file. As such, while the parties can explain why they think their proposed construction for each term is correct and why the other side's proposed construction is wrong, they have to guess what arguments the other side will make. This "guessing" may lead to parties making counter-arguments to arguments that the other side did not actually make, which is not a good use of space in their briefs for the parties and is not helpful for the court. An alternative to simultaneously filed briefs is nonsimultaneously filed briefs where the party that is arguing against the default construction goes first—usually the defendant—and then the parties alternate briefs for as many rounds as the court wants (usually two more rounds but sometimes three more).

Furthermore, only having two rounds of simultaneous briefs also may not be helpful for the court. For

example, because each round of briefs were simultaneously filed, the parties may make new arguments only in the second, i.e., last, round of briefs, which means that there is no counter-arguments to those arguments.

BACKGROUND SECTIONS OF WARF'S AND INTEL'S OPENING CLAIM CONSTRUCTION BRIEFS

Both WARF and Intel provided a lengthy background of the underlying technology—computer architecture—at the beginning of their briefs before providing a shorter description of the asserted patent. Each background section was approximately 10–15 pages, which is extremely long given that briefs usually have page limits and that each party's opening brief was around 40 pages. One potential reason why the background sections were so long was because the parties did not appear to submit a technical tutorial. Tutorials are usually relatively short, narrated videos with slides and animation to explain the underlying technology. In this case, Judge Crabb did not appear to ask for technology tutorials although WARF offered to present one.

The technical background sections for both sides' briefs were relatively similar. Namely, the parties' briefs described basic concepts such as the hardware/software divide, the division of hardware into processor and memory, fetching and executing instructions, speculative out-of-order execution, register-based dependences between instructions, memory dependences between earlier stores and later loads, and so on.

After the technical background, both parties provided a description of the asserted patent. A detailed description of the asserted patent can be found in the November/December 2024 issue of *IEEE Micro*.[2] Unusually, Intel also included a description a prior art patent that was listed on the asserted patent. While a description of prior art patents may be helpful as background information to understand the asserted patent, in this case, the technical background section appears to provide more than enough of a background. Intel likely included this description in order to plant the idea in the court's mind that the asserted patent is invalid in light of the prior art patent.

Both sides retained experts to support their claim construction positions, and each expert filed two declarations, one in each round of briefs, that provided their technical opinions regarding the disputed claim terms. WARF retained Prof. William J. Dally of Stanford (see *Wis. Alumni Rsch. Found. v. Intel Corp.*[3]) while Intel retained Prof. Douglas Clark of Princeton (see *Wis. Alumni Rsch. Found. v. Intel Corp.*[4]). Parties in patent cases retain experts for a variety of reasons including claim construction and to testify at trial about infringement/noninfringement, validity/invalidity, and damages. Parties often retain engineering professors to be their experts as they likely do not have any conflict of interest issues that could impact their credibility before the jury and/or disqualify them, as compared to an expert who currently works in industry may have. While an expert's technical opinions can be very helpful for a court to make claim construction decisions, under U.S. patent law, the weight a court gives to an expert's opinion is very low.[5]

INTEL LIKELY INCLUDED THIS DESCRIPTION IN ORDER TO PLANT THE IDEA IN THE COURT'S MIND THAT THE ASSERTED PATENT IS INVALID IN LIGHT OF THE PRIOR ART PATENT.

FIRST DISPUTED CLAIM TERM: "IN FACT EXECUTED"

All disputed claim terms that Judge Crabb construed appear in independent Claim 1, which is as follows, with the claim term bolded and underlined:

1. In a processor capable of executing program instructions in an execution order differing from their program order, the processor further having a data speculation circuit for detecting data dependence between instructions and detecting a **mis-speculation** where a data consuming instruction dependent for its data on a data producing instruction of earlier program order, is **in fact executed** before the data producing instruction, a data speculation decision circuit comprising:
   a) **predictor** receiving a mis-speculation indication from the **data speculation circuit** to produce a **prediction** associated with the particular data consuming instruction and based on the mis-speculation indication; and
   b) prediction threshold detector preventing data speculation for instructions having a **prediction** within a predetermined range.

For the claim term "in fact executed," the claim language ("detecting a mis-speculation where a data

consuming instruction dependent … on a data producing instruction … is in fact executed before the data producing instruction"), describes that a mis-speculation occurs when a load that depends on a store is "in fact executed" before the store.

Both sides heavily rely on what the asserted patent discloses, as required under U.S. patent law as "the specification [of the patent] is always highly relevant to the claim construction analysis. Usually, it is dispositive; it is the single best guide to the meaning of a disputed term."[6] Intel argues that its proposed construction is correct because the asserted patent describes how a sequence of load/store instructions is executed:

INTEL ARGUES THAT ITS PROPOSED CONSTRUCTION IS CORRECT BECAUSE THE ASSERTED PATENT DESCRIBES HOW A SEQUENCE OF LOAD/STORE INSTRUCTIONS IS EXECUTED.

> The STORE instruction 10.1 then stores a value in memory location A[2]. The LOAD instruction 8.2 then loads a value from the same memory location A[2]. This LOAD instruction 8.2 is thus unambiguously dependent on instruction 10.1, the data dependency 36 as indicated by an arrow. If instruction 8.2 were to be executed prior to instruction 10.1, it would operating on erroneous data.

(See pages 24–25 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[7]). Intel also argues that "[t]he specification also refers to an instruction as 'executed' when it has 'executed as far as possible'" (see page 25 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[7]). And that when an instruction has been "executed" as far as possible, the only thing left is for the load instruction to be 1) retired, if the prediction that there was no store/load dependence was correct, or 2) squashed, if the prediction was incorrect (see page 25 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[7]). In other words, in order to execute as far as possible, Intel argues that a load instruction needs to load data from memory.

Part of WARF's proposed construction is similar to Intel's proposed construction in that it states that a load instruction has "in fact executed" before a store instruction "when the LOAD instruction has actually accessed data." But WARF's proposed construction expands what it means to have "in fact executed" to include a load instruction that "was certain to access data that has not yet been updated by the STORE instruction." This part of WARF's proposed construction covers long-latency loads that are in-flight when the hardware detects the dependence between the load and store (see pages 36–38 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[8]). To justify why long-latency loads should be included, WARF argues that the asserted patent never restricts itself to only Intel's proposed construction (see page 34 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[8]). Rather, WARF argues that the asserted patent encompasses LOAD instructions that are certain to access incorrect data and that a failure to squash these loads will lead to incorrect program execution (see page 36–37 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[8]).

Intel's counter-argument regarding long-latency loads is that WARF is trying to include "partly executed" load instructions within the scope of "in fact executed" (see page 4 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[9]). Intel argues that the claims and specification of the asserted patent do not use the words "was certain to access data" that is in WARF's proposed construction or describe partial execution (see page 4 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[9]). Intel further argues that even if it partial execution "makes sense," the specification and Claim 1 do not disclose that scenario; they only disclose "in fact executed," i.e., complete execution up to the point of squash or retirement (see page 4 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[9]).

WARF's counter-argument regarding the long-latency loads is that the asserted patent describes that the data speculation circuit can track a load instruction before it accesses memory (see page 51 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[10]). WARF also argues that the actual issue is "whether there has been a mis-speculation between a LOAD and a STORE" (see page 52 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[10]). WARF asserts that the asserted patent clearly teaches that the data mis-speculation can be detected before the load retrieves the data from memory, i.e., once the memory addresses of the load and store are known (see page 52 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[10]). WARF contends that "[i]f a comparison with a STORE indicates that the LOAD instruction is certain to access data that has not yet been updated by the STORE instruction (i.e., an error is going to occur), then common sense dictates that corrective action should be initiated as soon as possible" (see page 52 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[10]).

With respect to Intel's argument that "in fact executed" requires that the instruction "executed" as far as possible, WARF argues that "[i]n an out-of-order

processor, a[n] instruction may be 'executed' and squashed prior to completion[,]" so "executed" "does not necessarily imply completion" (see page 53 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[10]). WARF further argues that from the perspective of the data speculation circuit or the LSQ, "the meaning of 'in fact executed' should be interpreted from the perspective of the data speculation circuit (or the load-store queue [LSQ]), which is whether the execution of a LOAD has progressed to a point that the circuit can declare a data mis-speculation" (see page 55 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[11]) (emphasis in WARF's brief). WARF also argues that there were patents that disclosed tracking data mis-speculations between load/store pairs at the time the patent was filed (see pages 55 to 56 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[10]).

JUDGE CRABB'S CLAIM CONSTRUCTION FOR "IN FACT EXECUTED"

After holding a hearing, Judge Crabb entered a claim construction order where she decided that the meaning of "in fact executed" meant "when a load instruction has actually accessed a memory address that has not yet been updated by a store instruction appearing earlier in the program order" (see page 10 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[1]). Judge Crabb concluded "in fact executed" cannot mean "completed" because if there is a mis-speculation, then the load is squashed and never completes (see page 11 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[1]). Judge Crabb concluded that the specification described an example where the data speculation circuit checks if a subsequent load accessed the same memory location as an earlier store, which "establishes that it is enough that the load instruction actually accessed a memory address for the instruction to have in fact executed" (see page 12 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[1]). Accordingly, Judge Crabb concluded that Intel's proposed construction, which required that the load instruction has to "actually has loaded data from a memory location" is incorrect (see page 12 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[1]).

Judge Crabb also concluded that WARF's argument that "'in fact executed' should be construed to include 'or was certain to access' because a load instruction 'is in fact executed as soon as you can determine that it was mis-speculated,' misses the mark as well" (see page 12 of *Wis. Alumni Rsch. Found. v. Intel Corp.*). Judge Crabb concluded that "[u]nder [WARF's] proposed circumstances, that is, detecting a mis-speculation before a memory address is actually accessed, 'in fact executed' would mean that the load instruction had begun to execute prematurely and erroneously or that the load instruction would execute prematurely and erroneously. To begin to execute or be executing is not the same as to have 'executed'" (see page 13 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[1]). Finally, Judge Crabb explained that "[WARF's] addition of 'was certain to access' adds a future action to what the claim describes as something occurring in the past, and therefore, such a construction is incorrect in light of the claim language" (see page 13 of *Wis. Alumni Rsch. Found. v. Intel Corp.*[1]).

WARF ALSO ARGUES THAT THERE WERE PATENTS THAT DISCLOSED TRACKING DATA MIS-SPECULATIONS BETWEEN LOAD/STORE PAIRS AT THE TIME THE PATENT WAS FILED.

ANALYSIS OF THE PARTIES' PROPOSED CONSTRUCTIONS AND JUDGE CRABB'S ORDER

I agree with Judge Crabb that Intel's proposed construction is overly narrow as it excludes long-latency loads are that in-flight and have not yet "actually loaded data from a memory location." Arguably, a load "executes" once it is sent to the memory hierarchy as the memory hierarchy will return a value for that load regardless if it is mis-speculated (rather, the LSQ will just ignore that data if the load has been squashed). In other words, the processor could have "executed" the load as soon as the processor sends the load to the memory hierarchy as the processor does not need to take any further action in order for that load to receive the data at that memory location. As such, Judge Crabb is arguably incorrect that "[t]o begin to execute or be executing is not the same as to have 'executed'" as loads may be considered to have been "executed" the cycle they access the memory hierarchy.

Furthermore, speculatively sending loads to the memory hierarchy involves one assumption and one action, namely, it 1) assumes that the load does not depend on a prior store, and 2) as such, the load can access the memory hierarchy (as soon as possible). A mis-speculation occurs when the assumption is wrong and when the load accesses the memory

hierarchy. In other words, if the assumption that the load does not depend on a prior store is true, then there is no issue with letting the load speculatively access the memory hierarchy. On the other hand, if the load does depend on a prior store, but the load has not accessed the memory hierarchy, the load did not incorrectly access the memory hierarchy. Therefore, this explanation provides additional support for the conclusion that a load has "executed" once it accesses the memory hierarchy.

Furthermore, Judge Crabb's final construction may inadvertently introduced a race condition that could lead to noninfringement depending on the timing of when the load and store accesses the memory hierarchy. More specifically, a person of ordinary skill in the art (which the standard that U.S. patent law uses to understand the meaning of a claim) might understand that "when a load instruction has actually accessed a memory address" to require that the load instruction needs to actually have physically accessed the particular memory in the memory hierarchy (but not necessarily have returned the stored data back to the LSQ). If so, when there is an L1 d-cache hit, the load would be "in fact executed" within a few cycles as L1 d-cache hits are very fast. But if the load misses the LLC, then the latency for the load to access the main memory might require hundreds of cycles, during which time the store instruction may have executed and stored its value to L1 d-cache. In that situation, an in-flight load would be "in fact executed" if it physically accesses main memory before the store stores its value in the L1 d-cache, but not if it store stored its value in the L1 d-cache before the load physically accesses main memory.

The final question is why did WARF and Intel propose the constructions that they did? WARF most likely proposed their construction in order to capture in-flight loads in an out-of-order processor. While that makes technical sense, the claim term is "in fact executed," which is in the past tense, and thus WARF's proposed construction reasonably might not cover in-flight loads.

Why Intel proposed their construction is harder to determine. A defendant in a patent case usually tries to propose a construction that, if adopted, means that their accused products will not infringe. It is unclear how Intel's proposed construction would lead to noninfringement. For example, if the load is an L1 d-cache hit, then the number of cycles to actually load that data from a memory location may be a few cycles, which may be before the store's address is computed. In this situation, Intel's processors, assuming all the other claim elements are met, would infringe Claim 1 because the load is "in fact executed." On the other hand, if there is an L1 d-cache miss, then the load may not actually have loaded the value from a memory location before the mis-speculation is detected, so Intel's processors would not infringe in those situations.

On the other hand, Intel may have intended to argue that "actually load[ing] data from a memory hierarchy" requires loading the value into the reorder buffer, and not only that it has been retrieved from memory and stored into a temporary queue. Obviously, if the processor mis-speculates that a subsequent load does not depend on a prior store, that load will be squashed and its value will never be loaded into the reorder buffer. Consequently, those loads under Intel's proposed construction will not "in fact [have] executed."

The next article in this series will continue describe the parties' arguments for each term in Table 1, the courts' analyses and reasoning, and my analysis.

REFERENCES

1. Wis. Alumni Res. Found. v. Intel Corp., No. 3:08-cv-00078, ECF No. 65, Sep. 18, 2008 (W.D. Wis.).
2. J. J. Yi, "A review of Wisconsin Alumni Research Foundation v. Apple—Part I," *IEEE Micro*, vol. 44, no. 6, pp. 92–96, Nov./Dec. 2024, doi: 10.1109/MM.2024.3504881.
3. Wis. Alumni Res. Found. v. Intel Corp., No. 3:08-cv-00078, ECF No. 31, Jun. 26, 2008 (W.D. Wis.).
4. Wis. Alumni Res. Found. v. Intel Corp., No. 3:08-cv-00078, ECF No. 34, Jun. 27, 2008 (W.D. Wis.).
5. Phillips v. AWH Corp., 415 F.3d 1303, 1318, 2005 (Fed. Cir.).
6. Phillips v. AWH Corp., 415 F.3d 1303, 1316, 2005 (Fed. Cir.).
7. Wis. Alumni Res. Found. v. Intel Corp., No. 3:08-cv-00078, ECF No. 33, Jun. 27, 2008 (W.D. Wis.).
8. Wis. Alumni Res. Found. v. Intel Corp., No. 3:08-cv-00078, ECF No. 29, Jun. 26, 2008 (W.D. Wis.).
9. Wis. Alumni Res. Found. v. Intel Corp., No. 3:08-cv-00078, ECF No. 51, Jul. 24, 2008 (W.D. Wis.).
10. Wis. Alumni Res. Found. v. Intel Corp., No. 3:08-cv-00078, ECF No. 54, Jul. 24, 2008 (W.D. Wis.).

JOSHUA J. YI is a solo practitioner at The Law Office of Joshua J. Yi, PLLC, Austin, TX, 78750, USA. Contact him at josh@joshuayipatentlaw.com.