Internal distribution code:

(A) [-] Publication in OJ
(B) [-] To Chairmen and Members
(C) [-] To Chairmen
(D) [X] No distribution

Datasheet for the decision of 23 June 2016

Case Number:	T 2374/11 - 3.5.06
Application Number:	04002677.5
Publication Number:	1455273
IPC:	G06F9/455
Language of the proceedings:	EN

Title of invention:
Emulation of microprocessor instructions

Applicant:
Northrop Grumman Systems Corporation

Headword:
Instruction emulation/NORTHROP

Relevant legal provisions:
EPC 1973 Art. 56
EPC Art. 52(2)(c)

Keyword:
Inventive step - (no)

Decisions cited:

Catchword:

European Patent Office
D-80298 MUNICH
GERMANY
Tel. +49 (0) 89 2399-0
Fax +49 (0) 89 2399-4465

Case Number: T 2374/11 - 3.5.06

D E C I S I O N
# of Technical Board of Appeal 3.5.06
of 23 June 2016

Appellant: (Applicant)	Northrop Grumman Systems Corporation 1840 Century Park East Los Angeles, CA 90067-2199 (US)
Representative:	Lieck, Hans-Peter BOETERS & LIECK Oberanger 32 80331 München (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 21 June 2011 refusing European patent application No. 04002677.5 pursuant to Article 97(2) EPC.**

Summary of Facts and Submissions

I. The appeal is directed against the decision of the examining division, dated 21 June 2011, to refuse application 04002677 for lacking an essential feature (Article 84 EPC 1973). As obiter dicta, further reasons concerning Article 84 EPC 1973 are given and lack of inventive step over the following document is objected to:

D1 Lang et al.: "Threaded Code Interpreter for Object Code", IBM TDB, 1 March 1986, pages 4238-4241; reproduced in IP.com PriorArtDatabase, XP013049537, ISSN: 1533-0001.

II. A notice of appeal was received on 31 August 2011. The appeal fee was received on 30 August 2011. A statement of the grounds of appeal was received on 31 October 2011, including a sole claim set. Oral proceedings were requested.

III. In its summons to oral proceedings, the board argued that the claims lacked an inventive step over D1.

IV. In a letter dated 23 May 2016, the appellant filed an auxiliary request.

V. In a letter dated 15 June 2016, the appellant announced that it would not attend the oral proceedings.

VI. Oral proceedings were held on 23 June 2016 in the absence of the appellant's representative, as announced. At the end of the oral proceedings the board announced its decision.

VII. The appellant requests that the decision be set aside and a patent be granted based on the main request filed with the grounds of appeal or the auxiliary request filed on 23 May 2016.

The further application documents are: description pages 4, 6 and 8-25 as originally filed; pages 2 and 2a filed on 12 April 2011; pages 1, 2b, 3, 5 and 7 filed on 23 May 2016; drawing sheets 1-11 as originally filed.

VIII. Claim 1 of the main request reads as follows:

"1. A method for emulating instructions (22) of a legacy microprocessor comprising the steps of:

(a) providing a host processor with host processor instructions (34);

(b) running emulated legacy instructions (22) on said host processor, wherein the instruction sets of the legacy processor and the host processor are different from each other;

(c) categorizing each of said legacy instructions (22) into one of four categories as Ring 0, Ring 1, Ring 2 and Ring 3 instructions, wherein

- a legacy instruction (22) falling into the Ring 0 category can be replaced with a single host instruction that performs the same function;
- a legacy instruction (22) falling into the Ring 1 category requires utilizing multiple host instructions (34) which together perform the same function as a single legacy instruction (22);
- a legacy instruction (22) falling into the Ring 2 category requires utilizing multiple host instructions (34) and one or more scratch pad registers;

- a legacy instruction (22) falling into the Ring 3 category is replaced with compiled higher-order language instructions which together perform the same function as the legacy instruction (22);

(d) converting each Ring 0 legacy instruction by directly mapping said Ring 0 instruction to a single host instruction (34); and

(e) providing instruction handlers (36) and directing said Ring 1, Ring 2 and Ring 3 instructions to said instruction handlers (36) for further processing, wherein an instruction stream (72) containing emulated Ring 0 instructions and unconditional branch instructions to said instruction handlers (36) for said Ring 1, Ring 2 and Ring 3 instructions is formed."

IX. Claim 1 of the auxiliary request reads as follows:

"1. A method for emulating legacy instructions (22) of a legacy processor comprising the steps of:

(a) providing a host processor with host processor instructions (34);

(b) providing four categories Ring 0, Ring 1, Ring 2, Ring 3 for categorizing each of said legacy instructions (22) into one of said categories, wherein

- said Ring 0 category relates to a legacy instruction (22) which is replaced with a single host instruction that performs a same function;
- said Ring 1 category relates to a legacy instruction (22) which requires utilizing multiple host instructions (34) which together perform the same function as a single legacy instruction (22);

- said Ring 2 category relates to a legacy instruction (22) which requires utilizing multiple host instructions (34) and one or more scratch pad registers;
- said Ring 3 category relates to a legacy instruction (22) which is replaced with compiled higher-order language instructions which together perform a same function as said legacy instruction (22);

(c) mapping each Ring 0 instruction directly to a single host instruction of said host processor instructions (34);
(d) mapping each Ring 1, Ring 2 and Ring 3 instruction to respective instruction handlers (36) for further processing;
(e) forming an instruction stream (72) containing said mapping to each Ring 0 instruction and unconditional branch instructions to said instruction handlers (36) for said Ring 1, Ring 2 and Ring 3 instructions; and
(f) executing said instruction stream (72) to emulate said legacy instructions (22) on said host processor, wherein a legacy processor has a different instruction set than said host processor."

Reasons for the Decision

1 *Overview of the invention*

The application relates to what is usually called "static binary translation" of legacy processor object code to host processor object code (see for example https://en.wikipedia.org/wiki/Binary_translation). Assuming that the instruction set of the host processor is sufficiently similar to that of the legacy processor, some instructions can be mapped one to one (called "Ring 0 category" in the application; see description sections [28] and [29]). If a legacy instruction cannot be mapped to one instruction of the host processor (Rings 1-3), it is replaced by an unconditional branch instruction (i.e. a jump) to a so-called "instruction handler". The latter is a subprogram in host processor object code to emulate one specific instruction of the legacy object code (figures 8 and 13A). In this way, the distance between instructions is not changed by the translation, so that the offset values of relative branch instructions remain valid and do not have to be adapted ([40], [41]). If the host processor is capable of "branch folding" ([29], sentence 6, and [42]), the insertion of unconditional branch instructions does not impair the host processor's branch prediction capabilities.

2 *Overview of the decision*

Starting from the prior art acknowledged in the application ([1]-[6]), the subject-matter of claim 1 of both requests lacks an inventive step (Article 56 EPC 1973), because its central features relate to schemes, rules and methods for performing mental acts (Article 52(2)(c) EPC) and are therefore not considered technical.

3 *Inventiveness of claim 1 of the main request*

3.1 The claim and the application mix steps from *different phases* in *preparing* and *performing* the emulation.

3.2 The first phase is the *preparation of the translation*. This comprises steps (a) and (c)-(e). During this phase, a human being first has to study the description of the two processors and their respective instruction sets in order to find equivalent or similar instructions of the host processor to emulate each instruction of the legacy processor. For some of the instructions, he may find 1:1 translations, notes these pairs of instructions (legacy instruction and equivalent host instruction) in a table similar to table 1 in the description (pages 20-23) and marks them with the category "Ring 0". He might come to the conclusion that a binary reorganisation (i.e. translation of the object code) is necessary ([45]), that sub op code fields must also be translated ([47], second sentence) and that registers and immediate value fields have to be moved "as necessary" (third sentence). He may also decide to map legacy registers (last sentence).

3.3 Then the person preparing the translation has to program small object code subroutines - perhaps using scratch pad registers (which are not provided by many host processors, e.g. mainstream desktop processors, see https://en.wikipedia.org/wiki/Scratchpad_memory) - in order to build Ring 1 and Ring 2 translation schemes for legacy instructions. He adds these translation schemes to his translation table.

3.4 The person may then realise that some instructions are "not easily emulated" using host instructions ([60]) - which raises the question of what "not easily" means in this context - and decide to program the translation of a specific instruction in a high-level programming language, such as C or C++. He then enters these programs into his computer and starts a compiler program to translate the programs into host instructions. He either notes these host instructions in its translation table on paper, or stores it electronically.

3.5 To summarise, the person has to understand the exact semantics of all instructions of both processors and to take many decisions. He further has to write object code or high-level programs to do (more or less) the same on the host processor as the legacy instruction on the legacy processor.

3.6 The second phase is the *translation* of the legacy object code into host object code (steps (d) and (e)). This can be done either using pencil and paper or with the help of a computer. The claim leaves this open. If the preparing person wants to let a computer (e.g. a computer with the host processor or a computer with a another processor) perform the translation, there is another phase, not explicitly set out as a step in the claims, namely a *programming phase* during which he has to program translation software, called the crack map object in the description ([34] and figure 1: 32). In doing so, he has to enter and to formalise the translation schemes and host object codes from its translation table and to write the program text of the "instruction cracking" program ([10]) in some high-level programming language, so that it performs the steps of figure 2 (see also [35]-[39]).

3.7 The board considers the tasks performed during the *preparation and programming phases* to be *mental acts*, in particular the *categorisation into Rings 0-3*.

3.8 After the translation phase, the execution phase can start during which the host object code (translated from the legacy object code) is executed on a host processor (step (b)). This can happen separately from the translation phase, possibly on a different computer (which can be termed the target computer containing a host processor) and triggered by a start command from (possibly) another person, the user of that target computer.

3.9 In view of the above, the only technical aspects of claim 1 would occur when performing the method for emulating instructions of a legacy microprocessor. As this is known from D1 (see page 1, first sentence), claim 1 does not involve an intentive step, Article 56 EPC 1973, in view of D1.

3.10 To summarise, the steps of the central phases (i.e. preparing the translation schemes, including categorising the instructions into Rings 0-3, and (possibly) programming the translation software), can only be performed by a human being. Thus, they lack technical character and cannot contribute to the presence of an inventive step.

3.11 Therefore, the subject-matter of claim 1 of the main request is not inventive (Article 56 EPC 1973).

4 *Inventiveness of claim 1 of the auxiliary request*

4.1 The same arguments apply to claim 1 of the auxiliary request, since the steps of the main request correspond to steps of the auxiliary request in the following manner:

main request -> auxiliary request:

(a) -> (a)
(b) -> (f)
(c) -> (b)+(c)+(d)
(d) -> (c)
(e) -> (e).

4.2 Therefore, the subject-matter of claim 1 of the auxiliary request is not inventive either (Article 56 EPC 1973).

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

Beschwerdekammern des Europäischen Patentamts
Boards of Appeal of the European Patent Office
Chambres de recours de l'Office européen des brevets

A. Vottner W. Sekretaruk

Decision electronically authenticated