Internal distribution code:

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

Datasheet for the decision of 16 May 2014

Case Number: T 1903/09 - 3.5.06

Application Number: 03776468.5

Publication Number: 1573461

IPC: G06F9/45, G06F17/50

Language of the proceedings: EN

Title of invention:
MAP COMPILER PIPELINED LOOP STRUCTURE

Applicant:
SRC Computers, LLC

Headword:
Pipelined loop structure/SRC

Relevant legal provisions:
EPC Art. 84, 123(2), 56

Keyword:
Claims - support in the description (no) - main request - clarity (no) - first auxiliary request
Amendments - added subject-matter (yes) - main request
Inventive step - (no) - second auxiliary request

Decisions cited:

Catchword:

Europäisches Patentamt
European Patent Office
Office européen des brevets

Beschwerdekammern
Boards of Appeal
Chambres de recours

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

Case Number: T 1903/09 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 16 May 2014

Appellant: (Applicant)	SRC Computers, LLC 4240 North Nevada Avenue Colorado Springs, CO 80907 (US)
Representative:	Moore, Barry Hanna Moore & Curley 13 Lower Lad Lane Dublin 2 (IE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 15 May 2009 refusing European patent application No. 03776468.5 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairwoman	M.-B. Tardo-Dino
Members:	A. Teale
	M. Müller

Summary of Facts and Submissions

I. This is an appeal against the decision, dispatched on 15 May 2009, by the examining division to refuse European patent application No. 03 776 468.5 on the basis that the subject-matter of claim 1 of the then main request did not involve an inventive step, Article 56 EPC 1973, in view of the following document:

D1: Rinker R. et al., "An Automated Process for Compiling Dataflow Graphs into Reconfigurable Hardware", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, IEEE Service Center, Piscataway, NJ, US, vol. 9, no. 1, February 2001, XP011063138, ISSN: 1063-8210.

Claim 1 according to the then auxiliary request was found not to comply with Article 123(2) EPC regarding added subject-matter.

II. A notice of appeal was received and the appeal fee paid on 22 July 2009. The appellant requested that the appealed decision be set aside in its entirety and that the application be granted or remitted to the examining division to appoint oral proceedings.

III. With a statement of grounds of appeal, received on 1 September 2009, the appellant filed amended claims according to a main and six auxiliary requests. The appellant also made an auxiliary request for oral proceedings prior to any adverse decision by the board.

IV. In an annex to a summons to oral proceedings the board expressed doubts as to whether the application satisfied *inter alia* Article 84 EPC 1973, regarding

clarity and support, and Article 123(2) EPC, regarding added subject-matter.

V. With a letter received on 15 April 2014 the appellant filed amended claims according to a main and a first auxiliary request to replace all requests then on file, together with amended pages of the description and drawings.

VI. At the oral proceedings on 16 May 2014 the appellant filed amended claims according to a second auxiliary request and requested that the decision under appeal be set aside and a patent be granted on the basis of the main request or, alternatively, of the first auxiliary request, both filed with the letter of 15 April 2014, or of the second auxiliary request filed during the oral proceedings, description page 1 as submitted with the letter of 15 April 2014, pages 4 to 12 as published on 21 May 2004, and pages 2 and 3 as received on 3 September 2008; figures 5 and 7 as submitted with the letter of 15 April 2014, and figures 1 to 4, 6 and 8 as published on 21 May 2004.

VII. At the end of the oral proceedings the board announced its decision.

VIII. Claim 1 according to the main request reads as follows:

"A pipelined loop structure apparatus (100) comprising: a first unit (116) that processes an input value to generate an output value in successive iterations of the first unit (116), wherein the output value is captured by a second unit (110, 112, 114) coupled to the first unit (116); a third unit (118) coupled to the first unit (116) that determines a final loop iteration; a fourth unit (122, 124) coupled to the

second unit (110, 112, 114), wherein the fourth unit (122, 124) stores a final output value, is latched for distribution and subsequently ignores output values generated after the third (118) unit determines the final loop iteration has occurred, and a fifth unit (108) for adjusting the period for each iteration of the first unit (116) such that one or more clock cycles pass between values being input to the first unit."

IX. Claim 1 according to the first auxiliary request reads as follows:

"A reconfigurable hardware system including control-flow dataflow pipelined loop structures (100) that iterate after a final condition is met, the pipelined loop structure comprising; a first part (116) configured to process an input value to generate an output value in successive iterations of the first part (116), wherein the output value is captured by a second part (110, 112, 114) coupled to the first part (116); a third part (118) coupled to the first part (116) and is configured to determine a final loop iteration; a fourth part (122, 124) coupled to the second part (110, 112, 114), wherein the fourth part (122, 124) is configured to store a final output value, is latched for distribution and subsequently ignores output values generated after the third part (118) determines the final loop iteration has occurred, and a fifth part (108) configured for adjusting the period for each iteration of the first part such that one or more clock cycles pass between values being input to the first part."

X. Claim 1 according to the second auxiliary request reads as follows:

"The use of a reconfigurable hardware compiler, to compile a control-flow dataflow graph pipelined loop structure that is executed as a logic routine on a FPGA wherein the loop structure comprises: a loop body that processes an input value to generate an output value in successive iterations of the loop body, wherein the output value is captured by a circulate node coupled to the loop body; a loop valid node coupled to the loop body that determines a final loop iteration; an output value storage node coupled to the circulate node, wherein the output value storage node stores a final output value, is latched for distribution and subsequently ignores output values generated after the loop valid node determines the final loop iteration has occurred, and a loop driver node for adjusting the period for each iteration of the loop body such that one or more clock cycles pass between values being input to the loop body."

Reasons for the Decision

1. The admissibility of the appeal

In view of the facts set out at points I to III above, the appeal fulfills the admissibility criteria under the EPC and is consequently admissible.

2. The context of the invention

2.1 The application relates to "reconfigurable" computing. In this context, "reconfigurable" means that, when software is compiled, part of the computing hardware, for example one or more FPGAs (Field Programmable Gate Arrays) is also reconfigured to form circuitry specifically adapted to the computing task in hand. As a result, significant performance improvements can be

achieved when carrying out computationally intensive tasks such as image processing and hydrodynamic simulations.

2.2 The application focuses on reconfiguring hardware to produce circuitry adapted to execute loops, meaning blocks of code (termed the "loop body" or first unit/part in the claims) executed repeatedly, and in particular, "pipelined" loops. The term "pipelined" refers to the fact that, by appropriately grouping the tasks executed in the loop body, several instances of the loop can be overlapped and thus executed in parallel, allowing throughput to be increased by fully utilizing the available hardware resources.

2.3 The invention concerns the execution of loops which do not terminate after a predetermined number of iterations. Put another way, when the loop is started, the total number of iterations is unknown. The invention further considers what are called "iterative" loops, the intended meaning being that the result of one loop iteration is required by the following one.

2.4 Loop structures can be represented by control-dataflow graphs, as shown in figures 1, 3 to 5 and 7. These are also referred to as "loop structures". The "iterative" nature of the loops is represented in the loop structure by what is termed in the claims a "circulate node" or second unit/part to the input of the next loop instance. As to the termination of the loop, a "loop valid node" or third unit/part evaluates a loop termination criterion every time the loop is repeated. Only when the loop termination criterion is met does the loop terminate. The loop output value is passed by the "output value storage node" or fourth unit/part via a latch to the loop output. The clocking of the various

nodes/units/parts of the loop is controlled by the "loop driver node" or fifth/unit/part.

3. The main request

According to the application as originally filed, the pipelined loop structures are implemented using reconfigurable hardware, for instance FPGAs; see page 2, lines 4 to 5. There is no suggestion that the hardware might also be "non-reconfigurable", meaning that its circuitry cannot be repeatedly changed. However claim 1 has been amended to set out a "pipelined loop structure apparatus" followed by the features of the loop structure, the original claims having all set out control-flow dataflow graph pipelined loop structures. Hence claim 1 has been amended in such a way that the application contains subject-matter, namely the possibility that the apparatus might be non-reconfigurable, which extends beyond the content of the application as filed, contrary to Article 123(2) EPC. Moreover, insofar as claim 1 now covers non-reconfigurable apparatuses, it is also not supported by the description, contrary to Article 84 EPC 1973. Thus the main request cannot be allowed.

4. The first auxiliary request

4.1 Claim 1 sets out "A reconfigurable hardware system **including** control-flow dataflow pipelined loop structures ..." (emphasis by the board). In the oral proceedings it was put to the appellant that the effect of the term "including" seemed unclear in this context, since a "control-flow dataflow pipelined loop structure" could be understood as an abstract model used for specifying the intended reconfigurable

hardware but different from it (see also description, page 2, lines 17 to 19). The "inclusion" of such a model in a reconfigurable hardware system constituted a "clash of categories" which created uncertainty as to the relationship between the reconfigurable hardware system and the control-flow dataflow pipelined loop structures. Consequently the technical meaning of the term "including" would be unclear to the skilled person in this context. The appellant disputed this view, arguing that claim 1 was clear because the skilled person would know what "including" meant.

4.2 The board takes the view that there is indeed uncertainty as to whether "including" in this context can be understood, for instance, to cover systems which merely simulate the execution of loops based on the control-flow dataflow pipelined loop structures rather than implementing them in reconfigurable hardware. Consequently the board finds that the appellant has not succeeded in overcoming the board's doubts as to the clarity of the term "including" and concludes that claim 1 is unclear, contrary to Article 84 EPC 1973.

4.3 Hence the first auxiliary request is also unallowable.

5. The second auxiliary request

5.1 Prior art document D1

5.1.1 D1 concerns compiling programs written in the SA-C subset of C into dataflow graphs (DFGs) which are then translated into "VHDL" (VHSIC (Very High Speed Integrated Circuit) Hardware Description Language). The VHDL code is then mapped onto reconfigurable hardware, such as FPGAs, the process from SA-C code via a DFG to hardware being summarized in figure 2. Figure 4 shows

the structure of the code generated by the DFG to VHDL translator comprising an inner loop body (ILB) fed with data by a loop generator and feeding data to a data collector. According to section V on page 133, the DFG to VHDL translator classifies dataflow graph nodes into four types: run-time input nodes, generator nodes, loop body nodes and reduction nodes. The ILB is converted to VHDL by replacing the operations encountered when traversing the dataflow graph by corresponding VHDL components. According to section C on page 134, the data collectors and generators are created from VHDL components in a module library using parameters extracted from the DFG. D1 discusses a program written in SA-C, shown in figure 1(a) and having the dataflow graph shown in figure 1(b), being mapped via a VHDL description onto hardware, namely a reconfigurable computing board comprising five FPGAs (CPEO and PE1-PE4). The five FPGAs are connected by a bus, termed the "crossbar", and the reconfigurable computing board is connected to a "host" computer system which sends data to and retrieves data from the board.

5.1.2 The hardware implementation, shown in figure 6, includes a window generator which extracts a small subarray from a larger array. If the ILB produces several output values for each input, then the window generator places NULL values on the crossbar to reduce the window generation rate; see page 135, right column, lines 19 to 25. The translator achieves this by finding the maximum number of cycles required by all the components to process a frame of data and sends NULLs after each frame to avoid the data generation rate from exceeding the output data rate. If, due to the added input NULLs, the ILB does not produce a valid output in the current cycle then a "DontStore" signal determines that the output value is not to be stored. The output

values from the ILB are stored by the collector function in a buffer and, when the buffer is full, its contents are sent to the "MemArb" component for storage in memory.

5.1.3 Section VI relates to the example of convoluting an image with two constant 3x3 masks using the "Prewitt" algorithm to derive X and Y gradients, respectively. The corresponding SA-C code is compiled into a DFG and then translated to VHDL and used to produce FPGA configuration codes which are mapped onto the reconfigurable hardware comprising FPGAs. Section A (Preliminary Performance Results) states on page 137 (see lines 11 to 9 from the bottom) that improvements in processing speed were achieved by, amongst other measures, pipelining the ILB.

5.1.4 The board regards the "DontStore" signal (see page 135, right column, lines 32 to 41) in conjunction with the "MemArb" function (see page 136, right column, lines 22 to 32) as performing a latching function at the VHDL/ hardware level.

5.2 Inventive step, Article 56 EPC 1973

5.2.1 In view of the compilation process from SA-C code via a DFG and VHDL to the RCS (reconfigurable computer system) shown in figure 2 (see page 132) and the example of the implementation of the Prewitt algorithm disclosed in section VI, D1 discloses, in the terms of claim 1, the use of a reconfigurable hardware compiler, to compile a control-flow dataflow graph pipelined loop structure that is executed as a logic routine on a FPGA wherein the loop structure comprises a loop body that processes an input value to generate an output value

and an output value storage node for storing a final output value.

5.2.2 Hence the subject-matter of claim 1 differs from the disclosure of D1 in that:

(a) the loop body allows successive iterations, wherein the output value is captured by a circulate node coupled to the loop body and the output value storage node;

(b) the loop structure comprises a loop valid node coupled to the loop body that determines a final loop iteration;

(c) the output from the output value storage node is latched for distribution, output values generated after the loop valid node determines a final loop iteration being ignored, and

(d) a loop driver node adjusts the period for each iteration of the loop body such that one or more clock cycles pass between values being input to the loop body.

5.2.3 In the oral proceedings it was put to the appellant that the difference features between the subject-matter of claim 1 and the disclosure of D1 related to the consequences of the desire to evaluate iterative algorithms which terminated unpredictably. The appellant argued that at the priority date reconfigurable processors could only implement a restricted set of loops. For instance, in D1 none of the input data to the ILB shown in figure 4 derived from the output data. Also in the Prewitt algorithm, discussed in section VI, the application of the two

convolution masks did not change the input data to the ILB. That is, D1 did not disclose "iterative" loops in the explained sense. The effect of the difference features was that iterative loop functions could now be run on the reconfigurable hardware.

5.2.4 The board takes the view that the objective technical problem starting from D1 is to extend the reconfigurable hardware compiler to deal with more general loops (see page 2, lines 29 to 25, of the description). The consideration of such a problem would have been a usual matter of design for the person skilled in the art of reconfigurable hardware compilers at the priority date.

5.2.5 Difference features "a" and "b" set out straightforward changes to the dataflow graphs which the reconfigurable hardware compiler must be adapted to compile in order to, firstly, implement iterative loops and, secondly, repeatedly reassess whether the condition for loop termination has been met, and thus solve the stated problem.

5.2.6 Turning to difference feature "c", the reasons for the appealed decision stated that it was inherent in loop pipelining that a pipelined loop continued to execute after a final condition was met, since pipeline stages were usually not empty when the final condition was met. The appellant has challenged this statement, arguing that a pipelined loop could be implemented which did not start an iteration until it had been determined that the iteration was needed. However the appellant did not produce a corresponding example when invited by the board during oral proceedings. At any rate, in the board's view, it would have been obvious for the skilled person starting from D1 to implement a

pipelined loop which, in addition to reassessing the termination criterion, could carry out other functions spread across several loop iterations. In this case the pipelined loop would continue to execute after a final condition had been met. As stated above, the board understands D1 to disclose performing an output value latching function at the VHDL/hardware level. The skilled person reading this disclosure would have been aware that an obvious way of achieving this functionality would be to include the representation of such a latching function in the DFG used to generate the VHDL. The effect of such a latching function in the DFG would be that output values generated after the loop valid node determined a final loop iteration would be ignored. Hence the board finds that difference feature "c" sets out an obvious measure to solve the stated problem for a common class of pipelined ILBs.

5.2.7 Regarding difference feature "d", the board regards the window generator slowing down the window generation rate by adding NULLs (see page 135, right column, lines 19 to 31) constitutes a loop driver function at the VHDL/hardware level. As with difference feature "c", the skilled person reading this disclosure would have been aware that an obvious way of achieving this functionality would be to include the representation of such a loop driver function in the DFG used to generate the VHDL. Hence the board finds that difference feature "d" sets out an obvious measure to solve the stated problem.

5.2.8 Hence, starting from D1 and making usual changes necessary to solve the stated problem, the skilled person would have arrived at the subject-matter of claim 1 in an obvious manner. Consequently the subject-

matter of claim 1 does not involve an inventive step, Article 56 EPC 1973.

5.2.9 Hence the second auxiliary request is not allowable either.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairwoman:

B. Atienza Vivancos M.-B. Tardo-Dino

Decision electronically authenticated