

CLAIMS

1. A method of controlling execution of a transaction in a computing environment, the method comprising steps of:

initiating, by a processor, execution of a transaction in a transaction mode via a transaction begin instruction, the transaction begin instruction to begin any one of a first type of transaction or a second type of transaction, the transaction comprising a plurality of instructions to be executed, and the transaction begin instruction explicitly indicating by a field of the transaction begin instruction the first type of transaction or the second type of transaction being initiated;

based on being in transaction mode, effectively delaying committing transactional stores to main memory until completion of a selected transaction;

based on the transaction aborting, determining the type of transaction initiated, the type of the transaction being the first type of transaction or the second type of transaction, and the determining being based on the field of the transaction begin instruction;

based on the transaction aborting and the transaction being the first type of transaction, resuming execution at the transaction begin instruction; and

based on the transaction aborting and the transaction being the second type of transaction, resuming execution at a next sequential instruction directly following the transaction begin instruction in program order.

2. The method of claim 1, wherein the first type of transaction is a constrained transaction having one or more restrictions associated therewith, and the transaction begin instruction is a transaction begin constrained instruction, and wherein the second type of transaction is a nonconstrained transaction.

3. The method of claim 1, wherein the transaction is the second type of transaction, the second type being a nonconstrained transaction, and wherein the resuming execution comprises:

checking by the next sequential instruction following the transaction begin instruction a condition code of the transaction to determine whether the instruction is receiving control from an abort; and

based on the instruction receiving control from the abort, performing processing to handle the abort.

4. The method of claim 3, wherein the performing processing comprises taking action based on the condition code, wherein based on a condition code of one value, the action includes re-executing the transaction, and based on a condition code of another value, the action includes branching to a fall-back path.

5. The method of claim 1, wherein the method further comprises:
executing the transaction begin instruction, the executing setting a condition code to a first value; and
performing an abort of the transaction, the performing setting the condition code to a second value, wherein the second value indicates the next sequential instruction following the transaction begin instruction is receiving control from the abort.

6. The method of claim 1, wherein the transaction is the second type of transaction, the second type being a nonconstrained transaction, and wherein the method further comprises based on the transaction aborting, storing diagnostic information in a transaction diagnostic block specified by the transaction begin instruction.

7. The method of claim 1, wherein the method further comprises based on the aborting, restoring one or more registers designated in the transaction begin instruction and discarding transactional stores.

8. The method of claim 1, wherein the method further comprises:
determining that the transaction has aborted;
based on determining that the transaction has aborted, determining whether the abort was due to an interception or interruption; and
based on the abort being due to other than an interception or interruption, performing the resuming execution at the transaction begin instruction or resuming execution at the next sequential instruction depending on the transaction being the first type or the second type, respectively.

9. The method of claim 1, wherein the transaction is part of a nest of transactions, and wherein the method further comprises executing by the processor an extract nesting depth instruction to obtain an indication of a nesting depth of the nest of transactions.
10. The method of claim 9, wherein the transaction is an outermost transaction of the nest of transactions.
11. A system comprising means adapted for carrying out all the steps of the method according to any preceding method claim.
12. A computer program comprising instructions for carrying out all the steps of the method according to any preceding method claim, when said computer program is executed on a computer system.

CLAIMS

1. A method of controlling execution of a transaction in a computing environment, the method comprising steps of:

initiating, by a processor, execution of a transaction in a transaction mode via a transaction begin instruction, the transaction begin instruction to begin any one of a first type of transaction or a second type of transaction, the transaction comprising a plurality of instructions to be executed, and the transaction begin instruction explicitly indicating by a field of the transaction begin instruction the first type of transaction or the second type of transaction being initiated;

based on the transaction being the second type of transaction, the execution setting a condition code to a first value;

based on being in transaction mode, effectively delaying committing transactional stores to main memory until completion of a selected transaction;

based on the transaction aborting, determining the type of transaction initiated, the type of the transaction being the first type of transaction or the second type of transaction, and the determining being based on the field of the transaction begin instruction;

based on the transaction aborting and the transaction being the first type of transaction, resuming execution at the transaction begin instruction; and

based on the transaction aborting and the transaction being the second type of transaction, setting the condition code to a second value, wherein the second value indicates a next sequential instruction following the transaction begin instruction is receiving control from the abort, and resuming execution at a next sequential instruction directly following the transaction begin instruction in program order.

2. The method of claim 1, wherein the first type of transaction is a constrained transaction having one or more restrictions associated therewith, and the transaction begin instruction is a transaction begin constrained instruction, and wherein the second type of transaction is a nonconstrained transaction.

3. The method of claim 1, wherein the transaction is the second type of transaction, the second type being a nonconstrained transaction, and wherein the resuming execution comprises: checking by the next sequential instruction following the transaction begin instruction a

condition code of the transaction to determine whether the instruction is receiving control from an abort; and

based on the instruction receiving control from the abort, performing processing to handle the abort.

4. The method of claim 3, wherein the performing processing comprises taking action based on the condition code, wherein based on a condition code of one value, the action includes re-executing the transaction, and based on a condition code of another value, the action includes branching to a fall-back path.

5. The method of claim 1, wherein the method further comprises:

executing the transaction begin instruction, the executing setting a condition code to a first value; and

performing an abort of the transaction, the performing setting the condition code to a second value, wherein the second value indicates the next sequential instruction following the transaction begin instruction is receiving control from the abort.

6. The method of claim 1, wherein the transaction is the second type of transaction, the second type being a nonconstrained transaction, and wherein the method further comprises based on the transaction aborting, storing diagnostic information in a transaction diagnostic block specified by the transaction begin instruction.

7. The method of claim 1, wherein the method further comprises based on the aborting, restoring one or more registers designated in the transaction begin instruction and discarding transactional stores.

8. The method of claim 1, wherein the method further comprises:

determining that the transaction has aborted;

based on determining that the transaction has aborted, determining whether the abort was due to an interception or interruption; and

based on the abort being due to other than an interception or interruption, performing the resuming execution at the transaction begin instruction or resuming execution at the next sequential instruction depending on the transaction being the first type or the second type, respectively.

9. The method of claim 1, wherein the transaction is part of a nest of transactions, and wherein the method further comprises executing by the processor an extract nesting depth instruction to obtain an indication of a nesting depth of the nest of transactions.
10. The method of claim 9, wherein the transaction is an outermost transaction of the nest of transactions.
11. The method of claim 1, wherein the resuming execution at the transaction begin instruction based on the first type of transaction aborting further comprises automatically re-driving the constrained transaction, wherein in an absence of constraint violations, the constrained transaction is assured eventual completion.
12. The method of claim 11, wherein to ensure eventual completion, the processor takes certain actions, including at least one of: inhibiting out-of-order execution; inhibiting other processors from accessing conflicting storage locations; or inducing random delays in abort processing.
13. A system comprising means adapted for carrying out all the steps of the method according to any preceding method claim.
14. A computer program comprising instructions for carrying out all the steps of the method according to any preceding method claim, when said computer program is executed on a computer system.