Title: IMPROVED EMBEDDED SYSTEM PERFORMANCE

Abstract: A method of generating an embedded system (4999) from an original computer program (996) which embedded system (4999) provides a parallelized hardware (4598) and software (4599) implementation of the original computer program (996), which parallelized implementation (4598, 4599) satisfies one or more criteria regarding hardware constraints of the embedded system (4999). The system provides partitioning of functionality from the original computer program (996) using structural and behavioral program models and detects streaming and memory dependencies to improve the partitioning, relying on added indications of source lines and variables in said original computer program to relate partitions and dependencies in the program model with locations in the original program source code.
1. A computer-implemented method of generating an embedded system (4999) comprising a parallelized hardware and/or software implementation of an original computer program (996), which parallelized implementation satisfies one or more criteria regarding hardware constraints of the embedded system (4999), the method comprising the steps of

1. Subjecting (1000) the original computer program (996) to static and dynamic program analysis to obtain structural and behavioral program models,

2. Deriving (2000) a preferred design (3999) that satisfies the one or more criteria regarding hardware constraints using the obtained structural and behavioral program models, and

3. Building (4000) the embedded system (4999) by transforming the design (3999) to a parallel implementation that comprises a plurality of parallel processes that execute in software and/or hardware characterized in that

4. The step of subjecting (1000) the original computer program to static and dynamic program analysis is performed on a version of the original computer program (996) in which load and store instructions are annotated with information that links these instructions back to a specific part of a control-dataflow graph (1157),

5. The step of subjecting (1000) comprises adding to the structural and behavioral program models indications of memory dependencies and indications of streaming dependencies,

6. The step of deriving (2000) the design (3999) comprises providing information about mutually parallel implementation of each part of the preferred design (3999), which step comprises

7. a) Introducing a separation into different partitions of load and store instructions that have a detected streaming dependency between them, using the added indications of streaming dependencies from the models,

8. b) Introducing a separation into different partitions of load and store instructions that have a detected memory dependency with synchronization protection between them,
using the added indications of memory dependencies from the models.

c) retaining in a single partition load and store instructions that have another detected memory dependency between them, using the added indications of memory dependencies from the models.

the step of deriving (2000) the design (3999) further comprising a step of adding to the structural and behavioral program models indications of source lines and variables in said original computer program to relate the introduced partitions and the detected dependencies in the program model with locations in the original program source code, and

the step of building (4000) the embedded system (4999) comprising specifying the preferred design (3999) by means of a series of changes to be made to the original computer program (996) based at least in part on the added indications, applying the series of changes to the original computer program (996), and using the thus-changed original computer program (996) in the building of the embedded system (4999).

2. The computer-implemented method of claim 1, in which the step of subjecting (1000) comprises additionally adding to the structural and behavioral program models indications of at least the size of a buffer used for communication between tasks, and

the the step of building (4000) further comprises providing a function to bijectively map any address in a memory address range of a size corresponding to the size of the buffer to a virtual address range such that any address range \([aa .. aa+size-1]\) is mapped bijectively on the offset range \([0 .. size-1]\), where 'size' represents a number that is at least the size of the buffer.

3. The computer-implemented method of claim 1, in which the step of deriving (2000) derives a plurality of design alternatives that each satisfy the one or more criteria, and is followed by a step of selecting (3000) the preferred design (3999) from the plurality of design alternatives.

4. The computer-implemented method of claim 3, in which the step of selecting (3000) involves calculating an impact of at least one aspect of each of the design
alternatives and selecting the preferred design as the one of the design alternatives whose impact best fits a predetermined criterion.

5. The computer-implemented method of claim 4, where the impact relates to the aspect of an estimated execution speed of each of the design alternatives.

6. The computer-implemented method of claim 1, in which the step of subjecting (1000) the original computer program (996) to static and dynamic program analysis comprises adding to the structural and behavioral program models at least one of: the run time of every function invocation, the run time of every loop body invocation, the number of times each loop body is executed.

7. The computer-implemented method of claim 6, in which the step of deriving (2000) involves presenting said structural and behavioral models as a seed design (1999) to an operator using a graphical user interface and receiving an indication of a preferred invocation for deriving the preferred design (3999), in which the graphical user interface presents the invocations in said seed design as boxes and the data dependencies as lines, where the width of each box is dependent on the execution delay of the corresponding invocation as derived from the added run times and number of times each loop body is executed.

8. A computer program product comprising executable code for causing a processor to execute the method of claim 1.

9. A system configured for executing the method of claim 1.

10. A computer program product as produced by the method of claim 1.

11. A computer-readable storage medium comprising a computer program product of claim 10.

AMENDED SHEET (ARTICLE 19)