UDC 681.324

Grytsenko A., Malcheva R., Barkalov A.
Computer Engineering Department,
Donetsk National Technical University,
Artema Str., 58, Donetsk, 83000, Ukraine,
E-mail: redscop@gmail.com

RUN-TIME RECONFIGURABLE SYSTEM FOR DISTRIBUTED ALGORITHM EXECUTION

Run-time reconfigurable systems (RTR systems) allow configure hardware computing resources during computation process, without it interruption. This paper describes model of RTR system and model of complex algorithm distribution which designed to implement complex computation algorithm using optimal hardware resources. Main optimization factor for proposed system is minimization quantity of hardware resources without large productivity loss.

Keywords – run-time reconfigurable system, distributed algorithm, control unit, computation, logical block

1. Introduction

RTR systems could by assigned to implement complex algorithms with dynamic behavior using hardware. These systems allow reconfigure whole algorithm or part of them without interrupting computation process and also transparent for hardware environment [1]. RTR systems have several variants of using which reviewed in [2]. One of the variant is implementation of complex computation algorithm using minimal hardware resources. This means algorithm distribution to several parts – computational logical blocks, and separate configuration/execution of each part. Computational logical blocks configures and later executes sequentially (sequentially managed execution) or using special internal algorithm (algorithmically managed execution). Each computation logical block concerned with other blocks by dataflow and operation unit. Computational logical block (CLB) – part of algorithm which takes some data as input, process it and put some data as output. So CLB is logical structure with one input and one output. All CLBs could use shared operation unit, which allows reduce quantity of reconfigurable hardware resources and reduce time of each reconfiguration, if partially reconfiguration allowed. CLBs used shared external memory where they stores results of data processing.

2. Complex algorithm distribution

Complex algorithm distribution is determined by several factors. These factors define way to distribute base algorithm to set of CLB and order if CLBs configuring/execution.

Algorithm managing – define type of RTR system configuration/execution manager (RTR system control unit, RTR system CU). There are could be two main managing algorithms which determine different variants of algorithmically dependencies between separate CLBs. Computation logical blocks always have dataflow dependencies which couldn't be reduced anyway.

1. Simple manager of configuration/execution or sequentially manager of configuration/execution (sequentially CU) – performs sequential reconfiguration independently of previous CLB execution results. Next configured CLB is provident, excluding erroneous and exceptional situation. Algorithm distributes to several computation logical blocks which are fully independent concerning algorithm execution. Sequentially CU uses static planning of algorithm execution;
2. Algorithmically manager of configuration/execution (algorithmically CU) – performs reconfiguration dependently of current algorithm execution state. Algorithm execution state defines by current CLB execution results. Each CLB contains special function, which manages, for example, set of system flags. This managing system demand to use special control block which contain implementation of analyzing algorithms. Algorithmically CU activates after each CLB computations termination and solves which computational logical block will be next to configuration/execution.

Parallelism of CLBs execution – define method of CLB configuration/execution, characterize number of CLB executes simultaneously. There could be following main variants:

1. Single threaded system (ST system) – in each moment of time only one CLB is configured and executes (Fig. 1). Main drawback of this variant: time of CLB reconfiguration and CU working conforms to RTR system downtime. Main advantage of this system is simplification of managing system with static planning algorithm, absence of databases competitions, and possibility to use reconfigurable hardware which doesn't allow partial reconfiguration, fully hardware resources using between system downtimes. Figure 1 shows common presentation of ST system reconfiguration modeling. From t1 till t2 system is downtime, because current CLB execution terminated, but there no another tasks on reconfigurable hardware€. To reduce drawbacks of system downtimes could be used: decreasing of reconfiguration and analyzing time, increase size of CLB tasks to make downtime well less then CLB task execution time;

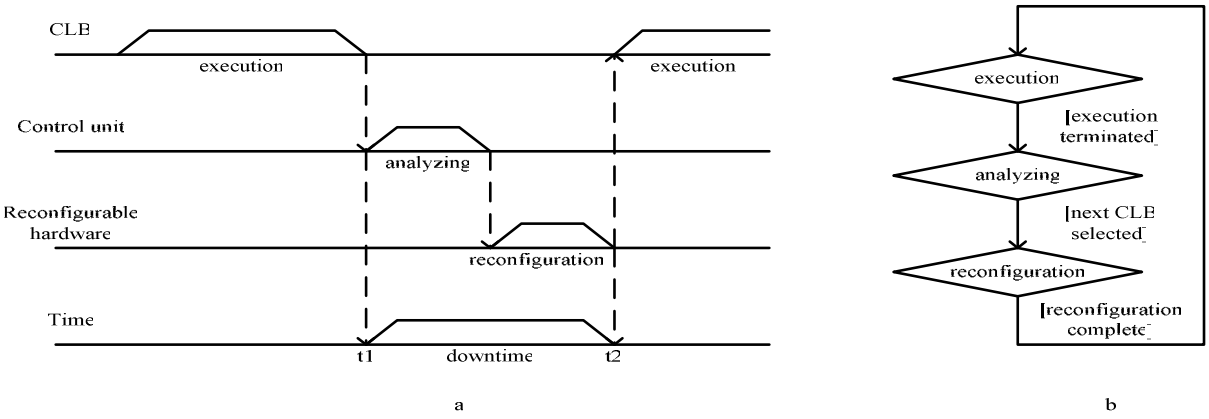

Fig.1 Single threaded RTR system working common diagram:
a) time-diagram form;
b) algorithmically form.

2. Multi threaded system (MT system) – in each moment of time several CLBs are configured and executes simultaneously. CU supply transparent CLB reconfiguration without interrupting other CLB execution algorithm, dynamically planning algorithm, conflicts solving algorithm. This type of RTR system needs supporting of partial reconfiguration from reconfigurable hardware. MT system hasn't drawback with system downtime when one CLB reconfigures, but has drawback with productivity lost. In this case downtime drawback compensate with other CLBs executing, so RTR system only has partial loss of productivity when one CLB reconfigures (Fig. 2). Features of MT system are: concurrently working CLBs shouldn't have dataflow dependencies, in other case one CLB will wait for data output from another so productivity loss will rise up; CU should support complex algorithm of multi threading execution and reconfiguration control; data bases sharing between several executes CLB, because each of them need access to input data, but number of ports is constraint.

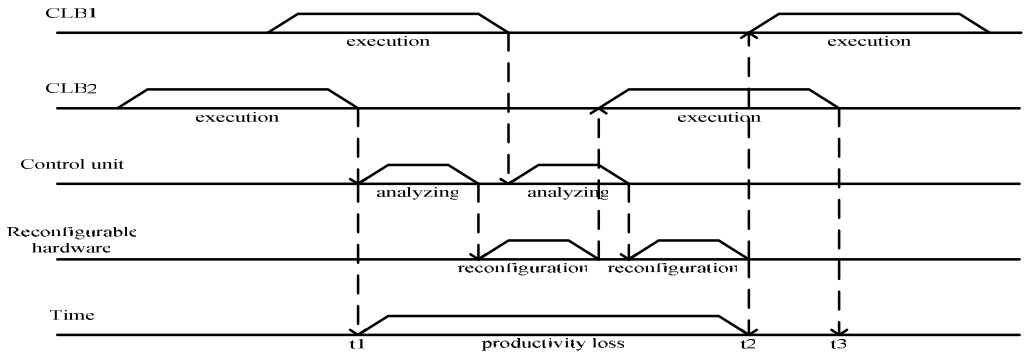

Fig. 2 Multi threaded RTR system working common diagram.

Type of algorithm distribution becomes formed from algorithm features and RTR system demands. Parallelism for algorithm distribution could be used only in case if algorithm has several independent branches of execution, where each branch could be distributed to one or more CLB and tasks of these CLBs could work simultaneously. This means functional decomposition of algorithm which peculiar to all algorithms but parallelism could be used only if result of functional decomposition is set of dataflow independent functions.

Implementation of algorithmically CU has sense also if we have some independent branches, but they could be dataflow dependent. Algorithmically CU used to increase system productivity by configured “just needed” CLBs.

For each type of distribution better to use CLB for large tasks, has execution time commensurable with reconfiguration time (including CU analyzing time). If CLB task execution time well greater then reconfiguration time then reconfiguration will be fully transparent for hardware environment.

3. RTR system for distributed algorithms execution design

System contains set of blocks which ensuring algorithm configuration, execution, data exchange etc. Offered system contains following blocks (Figure 3):

3. Executive subsystem – ensure configuring and reconfiguring CLBs and they execution. Contains: operation unit – non reconfigurable subsystems which implements set of low-level operations (functions) shared by most of CLBs, using of operation unit reduces reconfiguration time by making shared operations with more performance non reconfigurable hardware; reconfigurable kernel – hardware reconfigurable resources which purposed for CLBs configuration and execution; driver – control subsystem for reconfigurable kernel which ensure kernel encapsulation by special hardware interface with hardware environment.
4. Control unit – ensure execution of managing algorithms (algorithm managing and parallelism support) and interface with external memory where CLB configurations stored (flash memory, for example). Provides user interface for algorithm execution control, data processing etc.
5. Memory manager – ensure supply of external dynamically memory for executive subsystem and configurations memory for control unit. Implements set of caching or/and swap algorithms etc.

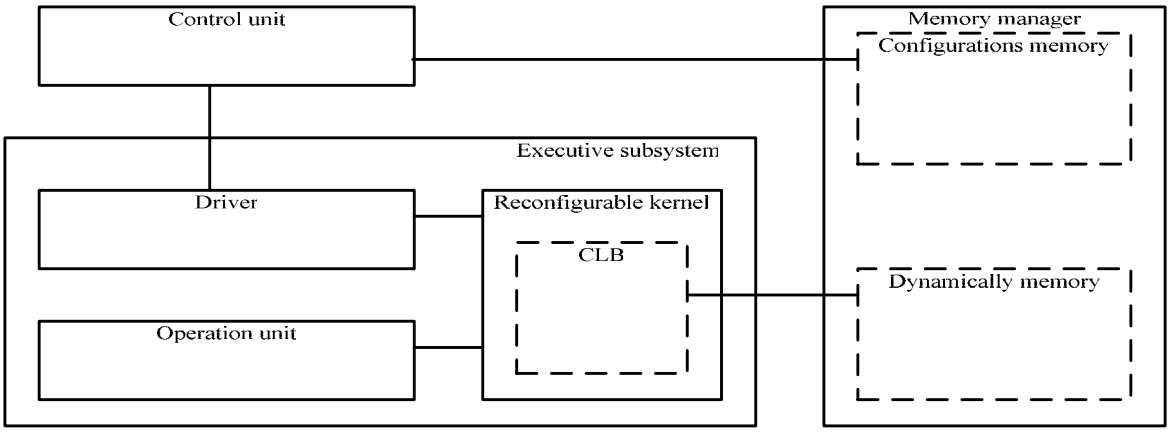

Fig. 3 RTR system design.

4. Implications and future directions

For this moment RTR system modeled using Hardware Description Language Verilog. At this moment investigates advantages and drawbacks of using complex control unit (means advantages of using algorithmically CU of sequentially CU) and parallelism (also dependencies between CLB task size and RTR system productivity). Also investigates advantages and drawbacks of using system fully implemented using hardware between systems which uses microcontrollers and software.

5. Summary

In this paper described design of RTR system for distributed algorithms execution and algorithms distribution features. In paper showed advantages and drawbacks of complex algorithm distribution algorithms, parallelism etc, given common diagrams for ST and MT RTR systems working.

References

[1] Nollet V., Mignolet J-Y. Hierarchical Run-time Reconfiguration Managed by an Operation System for Reconfigurable Systems – Leuven, Belgium 2003. – 7 pp.

[2] Гриценко А.А., Мальчева Р.В., Баркалов А.А. Классификация реконфигурируемых вычилсительных систем – Донецк, ДонНТУю – 2006 (CD-version).