so_ip_ecr_mv_s

Product Specification

Majority Voting Combination Rule Core – Serial Architecture

General Description

Machine learning is a scientific discipline that is concerned with the design and development of algorithms that allow computers to evolve behaviors based on empirical data, such as from sensor data or databases. A learner can take advantage of examples (data) to capture characteristics of interest of their unknown underlying probability distribution. Data can be seen as examples that illustrate relations between observed variables.

A major focus of machine learning research is to automatically learn to recognize complex patterns and make intelligent decisions based on data; the difficulty lies in the fact that the set of all possible behaviors given all possible inputs is too large to be covered by the set of observed examples (training data). Hence the learner must generalize from the given examples, so as to be able to produce a useful output in new cases.

There are many different predictive models (classifiers) in machine learning, including artificial neural networks (ANNs), decision trees (DTs) and recently introduced support vector machines (SVMs).

Recently a new way of making more accurate predictive models has emerged, ensemble learning.

Ensemble learning requires creation of a set of individually trained classifiers, typically decision trees (DTs) or neural networks, whose predictions have to be combined during the process of classification of previously unseen instances. Although simple, this idea has proved to be effective, producing systems that are more accurate then single a classifier.

All ensemble systems consist of two key components. First, a strategy is needed to build an ensemble that is as diverse as possible. A second strategy is needed to combine the outputs of individual classifiers that make up the ensemble in such a way that the correct decisions are amplified, and incorrect ones are cancelled out.

The main advantage of ensemble classifiers over single classifiers is in higher accuracy and greater robustness. The price to be paid is large amounts of memory to store the ensemble classifier and high computing power. This is because ensemble classifiers typically combine 30 or more individual classifiers, which means that 30+ times more memory and

computational power is required if we want to get the same performance in classification speed as with the single classifier.

Ensemble classifiers are typically implemented in software. But in applications that require rapid classification or ensemble creation, hardware implementation is the only solution.

So_ip_ecr_mv_s core can be used to implement the several variants of the Majority Voting combination rule to calculate the ensemble classification of the instance based on the classifications supplied by the ensemble members. Ensemble members whose classifications are being combined can be of any type, decision trees, neural networks, support vector machines, or some other predictive models. Even more, the ensemble can be even composed from a mixture of different predictive models.

So_ip_ecr_mv_s core can be used to implement the following Majority Voting combination rule variants: unanimous voting, simple majority voting, plurality voting and weighted majority voting.

So_ip_ecr_mv_s core should be used in conjunction with some ensemble evaluation module that is able to calculate the instance classifications for ensemble members sequentially, one at a time. Using these classifications, so_ip_ecr_mv_s core can calculate the combined classification of the current instance. Since the combination of the individual members classifications is done sequentially, the classification speed of this core is not so fast, but the core requires significantly less resources for the implementation.

So_ip_ecr_mv_s core is delivered with fully automated testbench and a compete set of tests allowing easy package validation at each stage of SoC design flow.

The so_ip_ecr_mv_s design is strictly synchronous with positive-edge clocking, no internal tri-states and a synchronous reset.

The so_ip_ecr_mv_s core can be evaluated using any evaluation platform available to the user before actual purchase. This is achieved by using a time-limited demonstration bit files for selected platform that allows the user to evaluate system performance under different usage scenarios.

Features

- Implements several variants of the Majority Voting combination rule that can be used to combine the classifications of individual ensemble members into one, collective classification
- Can be used to implement the following Majority Voting rules: unanimous voting, simple majority voting, plurality voting and weighted majority voting
- Combination of individual members classifications is done sequentially, resulting in area efficient design
- Ensemble members can be of any type, for example decision trees, neural networks, support vector machines, etc.
- Ensemble can be composed from a mixture of different predictive models
- No special IP blocks are needed to implement the core, only memory, adders and multipliers

Applications

- Speech and handwriting recognition
- Computer vision
- Machine perception
- Pattern recognition
- Medical diagnosis
- Robot locomotion

Deliverables

- Source code:
  - VHDL Source Code

- VHDL test bench environment
  - Tests with reference responses
- Technical documentation
  - Installation notes
  - HDL core specification
  - Datasheet
- Instantiation templates
- Example application
- Technical Support
  - IP Core implementation support
  - Variable length maintenance
    - Delivery of IP Core updates, minor and major changes
    - Delivery of documentation updates
    - Telephone & email support

Licensing

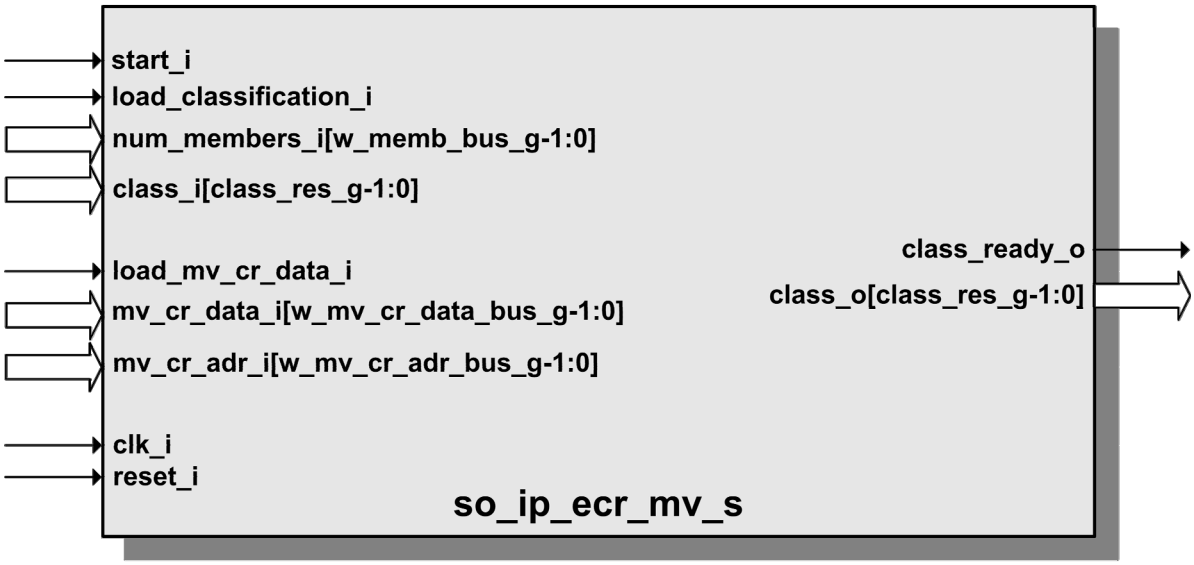
Netlist License

- Post-synthesis netlist
- Self checking testbench
- Test vectors for testing the core
- Place&Route scripts
- Constraints
- Instantiation templates
- Documentation

VHDL Source License

- VHDL RTL source code
- Complete verification plan together with testbenches needed to verify correct operation of the core
- Self checking testbench
- Vectors for testing the functionality of the core
- Simulation & synthesis scripts
- Documentation

Symbol



Pin Description

Name	Signal Direction	Description
Global Clocks and Reset Ports		
clk_i	Input	Main clock input
reset_i	Input	Main reset
Core Configuration Interface		
load_mv_cr_data_i	Input	Signal that is used to load the new configuration information into the MV core
mv_cr_data_i[w_mv_cr_data_bus_g-1:0]	Input	Data bus that is used to transfer the new data for the selected field in the MV core that should be modified
mv_cr_adr_i[w_mv_cr_adr_bus_g-1:0]	Input	Address bus that is used to specify field in the MV core should be accessed
Ensemble Member Classification Interface		
start_i	Input	Indication that a new cycle of combining ensemble member predictions into a collective classification should commence

load_classification_i	Input	Indication that there is a new classification from the next ensemble member that should be processed by the core
num_members_i[w_memb_bus_g-1:0]	Input	Information about the number of members in the ensemble
class_i[class_res_g-1:0]	Input	Classification of the current ensemble member that should be processed by the core
Output Class Interface		
class_ready_o	Output	Indication that the calculation of the collective classification by the core is finished and that the final classification is available at the *class_o* output port
class_o[class_res_g-1:0]	Output	Class values predicted by the ensemble for the current instance that has been classified

Verification Methods

Decision tree ensemble evaluation core was tested both using sophisticated verification environment and in dedicated hardware platform. Verification environment was used to extensively verify the so_ip_ecr_mv_s core's operation for sizes of ensemble classifiers. After reaching all verification goals, IP core was next tested using dedicated hardware platform. Using this platform so_ip_ecr_mv_s core was implemented in FPGA and tested in real applications to estimate the performance of the core. The details about the verification methodology that was used and performance results during hardware testing can be obtained from So-Logic upon request.

Device Utilization & Performance

So_ip_ecr_mv_s core has a very regular structure that allows an easy estimation of required hardware resources and classification speed.

The size of required memory, number of multipliers and adders and throughput of the so_ip_ecr_mv_s core are shown in the following table and expressed in terms of: number of DTs in the ensemble (N), number of classes (N_{cl}), number of bits for representation of coefficients values (N_c) and system clock period (CP).

Required Memory (bits)	Comparators	Adders	Multipliers	Throughput (number of classified instances per second)
$2 \cdot N_{cl} \cdot \lceil ld(N_{cl}) \rceil,$ $N \cdot N_w$	1	1	0	$\frac{1}{N \cdot CP}$

Contact Information

So-Logic
Lustkandlgasse 52/22
A-1090 Vienna
Austria/Europe
Phone: +43-1-3157777-11
Fax: +43-1-3157777-44
E-Mail: ip_ecr_mv_s@so-logic.net
URL: http://www.so-logic.net

Revision History

The following table shows the revision history for this document.

Date	Version	Revision
01/10/2009	1.0	Initial release.