Domínguez Montagud, CP.; Hassan Mohamed, H.; Crespo, A.; Albaladejo Meroño, J. (2015). Multicore and FPGA implementations of emotional-based agent architectures. Journal of Supercomputing. 71(2):479-507. doi:10.1007/s11227-014-1307-6.

Multicore and FPGA Implementations of Emotional Based Agent Architectures

Carlos Domínguez, Houcine Hassan[1], Alfons Crespo, José Albaladejo

Department of Computer Engineering (DISCA)
Universitat Politecnica de Valencia (UPV) - Spain
Tel:(+34) 96-3877578, Fax:(+34) 96-3877579,
E-mail: {carlosd, husein, jalba}@disca.upv.es

Abstract Control architectures based on Emotions are becoming promising solutions for the implementation of future robotic agents. The basic controllers of the architecture are the emotional processes that decide which behaviors of the robot must activate to fulfill the objectives. The number of emotional processes increases (hundreds of millions/s) with the complexity level of the application, limiting the processing capacity of the main processor to solve complex problems (millions of decisions in a given instant). However, the potential parallelism of the emotional processes permits their execution in parallel on FPGAs or Multicores, thus enabling slack computing in the main processor to tackle more complex dynamic problems. In this paper, an emotional architecture for mobile robotic agents is presented. The workload of the emotional processes is evaluated. Then, the main processor is extended with FPGA co-processors, which are in charged of the parallel execution of the emotional processes. Different Stratix FPGAs are compared to analyze their suitability to cope with the proposed mobile robotic agent applications. The applications are set-up taking into account different environmental conditions, robot dynamics and emotional states. Moreover, the applications are run also on Multicore processors to compare their performance in relation to the FPGAs. Experimental results show that, Stratix IV FPGA increases the performance in about one order of magnitude over the main processor and solves all the considered problems. Quad-Core increases the performance in 3.64 times, allowing to tackle about 89% of the considered problems. Stratix III could be applied to solve problems with around the double of the requirements that the main processor could support and a Dual-Core provides slightly better performance than stratix III and it is relatively cheaper.

Keywords: Multicore processors, Field-Programmable Gate Array (FPGA), Emotional-based Architectures, Real-time Systems, Intelligent Robotic systems

[1] Corresponding author.

1 Introduction

Many research works [1, 2, 3, 4] predict a growth of the number of intelligent robots in the industry and in our lives in the two next decades. They state that advanced robots capable of making decisions on their own as humans do are still under development and the first prototypes will not start to appear until 2030. Some researches [2, 6] state that we are seeing the emergence of the first generation of robots such as the demining robot Warrior manufactured by iRobot [7], which are only able to solve simple tasks with little ability to adapt to the changing environment, and running their program code on a single-core processor. However, more intelligent features that robots would include, such as decision-making, are not yet developed in real robotic agents. It is expected that by 2050, these agents will be implemented in advanced computers capable of running hundreds of billions of instructions (i.e., 4th. generation). These robots would rival human intelligence and would be able to perform operations of abstraction and generalization, medical diagnostics, planning and decision-making [5, 24, 25].

These kinds of applications involve high complexity, such as the proposed multi-purpose mobile robotic agent performing transportation, diagnosis, cleaning, and surveillance services simultaneously in uncertain and unpredictable environments. Each service problem is decomposed in a set of sub-problems and possible alternatives to be assessed (e. g., observation, path planning and object handling sub-problems for the transportation service). In the same way, each of the sub-problems is decomposed in simpler tasks (e. g., the path planning sub-problem generates a full tree of path alternatives that must be assessed to select one path). As this decomposition is performed for each of the sub-problems of all the simultaneous services, it arises that in a given instant the total number of decision alternatives that the agent has to manage are significantly high (e. g., 1M decisions).

On the other hand, control architectures based on emotions are inspired on emotional natural agents. They are becoming promising solutions for the implementation of advanced robotic systems [8, 9, 10] because they facilitate the process of decision-making [17, 18]. They use the mechanism of emotions to organize the behaviors, which has the following advantages: (i) allow the robot to be autonomous to focus its attention on the most promising behavior, (ii) provide a bounded response time which helps organizing the deliberative processes, (iii) sort the problems based on the expectations of success, (iv) autonomously adapt the computational load to the available processor capacity allowing solving problems of increasing complexity, (v) separate the decision from the action processes and the use of subjective appraisal of the situation permit find always an alternative solution. In this paper, an emotional robotic architecture for the control of complex mobile robot applications is used. In this programming model, there coexist two main types of processes: behavior and emotion processes. The formers solve the application problems (e. g., surveillance) while the latter use an emotional mechanism to motivate the robot behaviors.

Originally, all the processes of the emotional architecture, including the behaviors and the emotions were executed on a single-core computer (e. g., AMD 3.3 GHz). The emotion processes must be applied to all problems/subproblems of the agent agenda at every cycle of attention (e. g., 0.1 s) to decide which behaviours to execute. As the number of problems grows in the agenda, the emotional workload increases significantly as well. For instance, the proposed multipurpose robotic agent, to tackle complex problems, will need a high computational workload of about 200 *Millions emotion Operations per Second* (MOPS), as will be shown in the experiments section. Where an *Operation* is a Dot-Product function involving: a hyperbolic tangent function, a multiplication by a weight and a sum of up to 6 other functions computed in parallel. However, the single-core computer cannot support this workload because it can only execute 25 MOPS. Moreover, the implementation of the emotion processes on an MCU or low-medium performance DSP is discarded also because these devices provides even less power computation (i. e., between 10 MOPS and 20 MOPS). Therefore, the most suitable solutions that provide the performance required by the above-mentioned applications are high performance FPGA and Multicore processors. To this end, we transfer the execution of the emotion processes to the FPGA, thus the single-core computer gets slack time to solve more complex applications by: (i) improving the productivity of the simultaneously number of problems, or (ii) tackling more time critical dynamic problems (e. g., solve the problems at highest speed).

Regarding the FPGA processors, the communication between the FPGA and the Control Computer (CC) is performed through Ethernet. The CC sends, at each attention cycle, the parameters of the emotions through an Ethernet connection. The FPGA performs the calculation

and sends back the results of the behaviors' motivation to the CC. As will be presented in section 3.2, the emotional process has an accumulative phase. Thus, in order to optimize the use of the processing resources of the FPGA-based coprocessor, the CC arranges the parameters of the emotions over the data flow in a specific order. The latencies of each of the phases in the processing pipeline of the FPGA require this optimal order. Different FPGA models have been analyzed. Low performance FPGA are not sufficient to tackle the execution of the emotion processes due to their limited computational power (e.g., Stratix EP1S20F484I6 performs only 15 MOPS). Therefore, high performance FPGA Stratix III and Stratix IV [12, 13] are selected. To undertake the implementation, the potential parallelism of the emotion processes is identified and characterized. The Dot-Product functions based on hyperbolic tangent that are computed in parallel are optimized based on the A^3 methodology [19] to use the optimal number of emotion operators. Finally, the emotional processor is designed in VHDL for both FPGA models.

A second implementation of the emotional architecture is carried out using Multicore processors in order to compare the performance of the FPGA's and the Multicore, depending on the number of dedicated cores. In this case, a six-Core Intel i7 processor is used [14]. Four of the cores are dedicated to the emotion processes, one for the application processes and one for the attention system. The emotion processes are allocated to the different cores using the Worst Fit algorithm, which allows balancing the total workload among the cores. Each core implements a local Rate Monotonic Scheduler to support the execution of the processes [20].

In the experimental evaluations, different application problems of varying complexity levels (simple, normal, complex), under distinct environmental conditions and robot agent dynamics (safe, normal, risky), and different emotional states (relaxed, normal, stressed), are implemented using the FPGAs. The obtained performance results are compared with the execution of the same applications on the multicore processor. Experimental results show that, Stratix IV FPGA increases the performance in about one order of magnitude over the main processor and solves all the considered problems. Quad-Core increases the performance in 3.64 times, allowing to tackle about 89% of the considered problems. Stratix III could be applied to solve problems with around the double of the requirements that the main processor could support and a Dual-Core provides slightly better performance than stratix III and it is relatively cheaper.The rest of the paper is organized as follows: section 2 reviews the state of the art of emotional architectures and their implementations; section 3 describes the general characteristics of the emotional control architecture and details the emotional system; section 4 describes the multipurpose robotic agent application and the FPGA and Multicore based emotional processor designs; experimental evaluation and results are discussed in section 5; finally, conclusions are sum-up in section 6.

2 Related work

Different control architectures based on emotions are found in the bibliography. Arkin et al. [17] develop algorithms based on the emotion of deception to control robotic agents. The authors are inspired on the behavior of deception observed in animals or humans. In the simulations they show that robots including this emotion are more effective. Salichs [18] proposes a decision-making system based on drives and motivations, but also emotions and self-learning. The agent's goal is to learn to behave through interaction with the environment, using reinforcement learning, and maximizing their welfare. Lee-Johnson et al. [8] develop a hybrid architecture reactive/deliberative that incorporates artificial emotions to improve decision-making and actions of mobile robotic agents. Emotions are active at different levels of the architecture and serve to modulate the decisions and actions of the agent. Damiano [2] presents a model where decision-making is based on a motivational system. The motivations are dependent on the value of the need that have to be satisfied and a stimulus incentive. Once they calculate all the values, the highest motivation activates and organizes the behavior so as to satisfy the most urgent need. On the other hand, intelligent agents have been implemented using different SoC technologies. In [11] a neuro–fuzzy agent for ambient-intelligence environments is proposed. The agent has been implemented as a system-on-chip (SoC) on a FPGA around a MicroBlaze processor and a set of parallel intellectual property cores for neuro–fuzzy modeling. The SoC is an autonomous electronic device able to perform real-time control of the environment in a personalized and adaptive way, anticipating the desires and needs of its inhabitants. In [15] authors present a parallel genetic programming (PGP) Boolean synthesis implementation based on a cluster of Virtex5 FPGAs using parallel programming and hardware/software co-design techniques. The performance of the cluster of

FPGAs implementation has been compared with an HPC implementation resulting in an improvement of the speed up and in terms of solving the scalability problems of this algorithm. A practical implementation of a neural network based estimator of the load machine speed for a drive system with elastic coupling, using an FPGA placed inside the NI CompactRIO controller is presented in [16]. The algorithm code for the neural estimator implemented in C-RIO was performed using the LabVIEW software. The focus is on the minimization of the used programmable blocks of the FPGA matrix. Tests of the load machine estimator implementation are performed and results show high-quality state variable estimation of the two-mass drive system.
The aforementioned papers present very interesting implementations of complex control applications, however, regarding the implementation of emotional models, this is for the best of our knowledge, the first proposal of a parallel implementation of emotional architectures on FPGA and Multicores. Moreover, this paper differs from the above ones in the sense that it tackles the NP-hard problem of decision making in multi-objective intelligent agent applications, where different high-level complexity problems are simultaneously solved using an emotional system. In the proposed system, the computational power of the applications is higher compared to the previously commented papers, due to the high number of problems and decision alternatives to undertake. These types of problems are very different from the above ones, which are more focused in a specific well-defined task (i. e., robot manipulator) where the number of solution alternatives is small. Likewise, the paper uses high performance FPGAs: Stratix III and IV to tackle the implementation of the emotional robotic architecture and provides a discussion of the experimental results regarding the convenience of implementing the emotional model whether on an FPGA or on a Multicore processors.

Some ASICs have been developed to give processing support in different areas of artificial intelligence. A remarkable example is the SyNAPSE project [26, 27] lead by IBM, where thousands of neural cores integrated in a single chip offer a processing layer for the emulation of natural neural processes. The model of the emotion process in our agent architecture, however, differs from the artificial neural network model; therefore, the emotions have been implemented on a specific processor. The development of an ASIC for emotional purposes would require considerable financing resources. The initial design phases of a new processing approach typically benefits of the availability and flexibility of the FPGA technology, which is an affordable platform for prototypes and small production series.

3 Emotional control architecture

In this section, the complexity of the emotional control architecture is described and the computational requirements of the emotional system and the exhibited potential parallelization are discussed.

3.1 Real-time emotional control architecture overview

An emotional control architecture has been developed in the group of Industrial Informatics at Universitat Politecnica de Valencia. This architecture (see Fig. 1) is composed of five modules: Belief, Behavior, Emotion, Attention and Relation sub-systems. The Belief and Behavior systems represent the application processes, while Emotion and Attention systems are in charged of the execution of the operational processes.

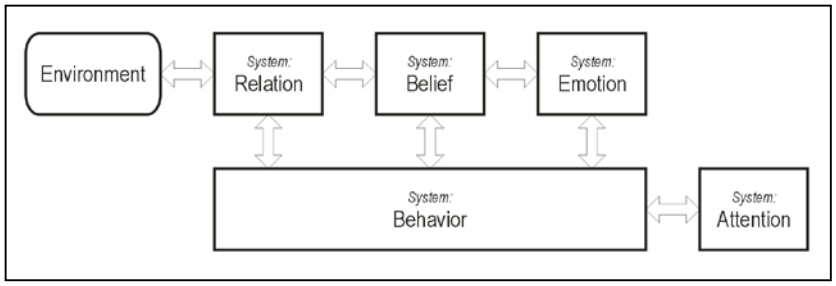

Fig. 1 Emotional control architecture block diagram

A behavior of the emotional control architecture is based on a problem resolution process. A problem appears when the agent desires a new situation. Every new desire starts an associated behavior, which defines a context for the execution of observation, decision and action processes related to the problem to be solved. These problem-domain processes are the application processes. On the other hand, the system implements operational processes that use an emotional mechanism to motivate the application processes. Fig. 2 shows the information flow in the emotional control architecture. Ellipses represent concepts and squares represent processes. The bold arrows represent the main paths. The (a) path connects sensors and motors devices, from the perception to the action. The application processes in (a) flow through two subways, the deliberative-way (b) and the reactive-way (c). Reactive processes have a guaranteed response time, which is usually short. The response time of deliberative processes however is usually longer. The rest of the paths are used by the operational emotional processes, which generate new desire-behavior (d) and execute motivated behaviors (e).

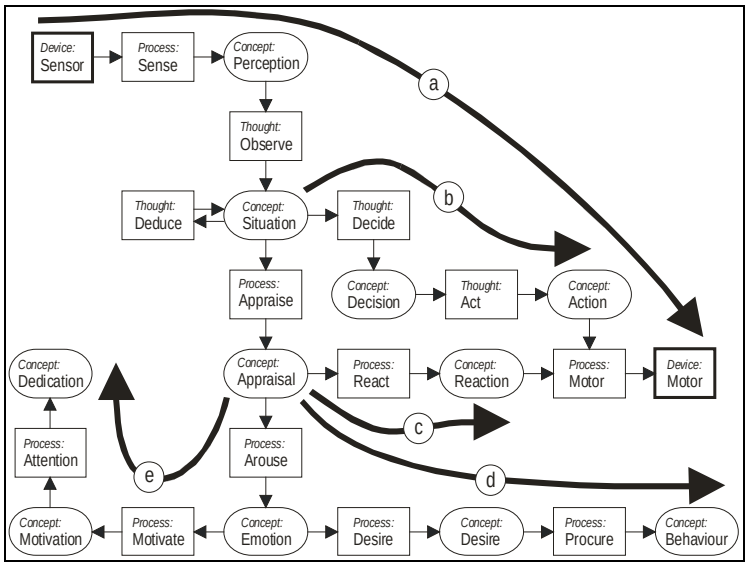

Fig. 2 Information flow in the emotional control architecture

Initially, the architecture was running on a main computer (single-core) as shown in Fig. 3, where the Input/Output is managed by a DAQ. However, as the complexity of the applications increases the emotional workload raises significantly, reducing the capacity of the single-core to solve the problems. This paper proposes the implementation of the emotional and attention processes on FPGA/Multicore processors to allow the single-core focus in solving more complex problems with high dynamic requirements.

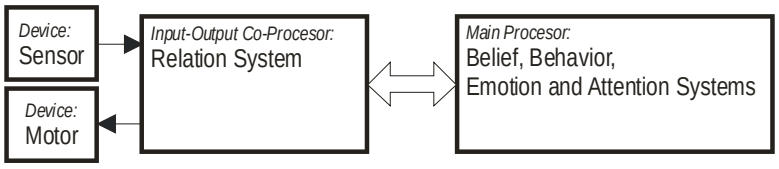

Fig. 3 Original system

To show how to perform the distribution of the different types of processes among the single-core and the FPGA/Multicore processors, the structure of the emotional architecture is described.

3.2 Emotional processes specification

An emotion is the process of appraising an observed situation and motivating a robotic agent behavior to undertake this situation (see Fig. 4).

Fig. 4 Emotional process

During its attention cycle, the agent evaluates the set of active emotions. This set of emotions grows and decreases dynamically as new problems are registered/unregistered in the agent agenda. Two subsets comprise the emotions set: (1) the set of intrinsic emotions and (2) the set of non-intrinsic emotions.

(1) Intrinsic Emotions - application independent
One intrinsic emotion is associated to each of the problem instances in the agenda, and its emotional response consists in the motivation of the process in charged of resolving the problem.
The structure of the intrinsic emotions (the number, nature and weight of their emotional contributions) is the same for each instance; that is, it is independent of the problem. The agent builder however, defines this general structure accordingly with the situational factors that are relevant to motivate the problem resolution's process, no matter the specific nature of the problem. These intrinsic emotion structure definitions define and name different agent characters.
- Example of Intrinsic Emotions
In the case of the mobile robot's application presented in paragraph 4.1, the Importance, Confidence, Urgency and Opportunity situation appraisals, contribute to the intrinsic emotions. The Importance emotional contribution comes from the appraisal of the benefits that would be obtained if the problem is resolved, meanwhile the remainder emotional contributions come from the appraisal of the success' expectative on the problem being solved. The Confidence emotional contribution has different dimensions: Confidence on the situational observations, Confidence on the problem solving method, and Confidence on the processor availability. The Urgency and Opportunity contributions distortion the importance of the problem, and contribute to the motivation of each problem, at least partially, in an inter-problem basis, causing motivation inversions; the Urgency dealing with the deadlines and the adverse effects of not meeting them; and the Opportunity dealing with the benefit-effort balance at each time.

(2) Non-intrinsic Emotions - application dependent
The methods that the problem-solving processes apply follow a sub-problem decomposition strategy with a sequence-and-alternative schema.
Every new sub-problem built and registered in the agenda is the result of an emotional decision. When a method is defined, either during the agent initial building or during a later learning process, non-intrinsic emotions are defined and linked to key emotional decision points in the method. These non-intrinsic emotions are application dependent, so their structure (the situation appraisals that contribute to the emotion activation, the contribution functions, and their weights) are specific for the type (not at instance level) of problem. The agent builds and registers the non-intrinsic emotions when the method reaches their specific emotional decision points. After that, the emotions are evaluated every attention cycle, and their response consists of building (or destructing) new sub-problem resolution processes, giving them an importance level (applying an importance appraisal propagation mechanism), and building and registering their intrinsic emotions.
A key design decision of the agent builder is the level of granularity of the decomposition of the problem. The agent builder explicitly expose this decomposition to the emotional motivation mechanism. Beyond it, the methods applied in the final steps of the problem decomposition look like black boxes to the emotional system.
- Example of Non-Intrinsic Emotions
The mobile robot application presented in paragraph 4.1 defines a wide set of problem types, resolution methods and emotional decisions. As an illustrative example, consider the Robot Displacement Problem. Since most of the services require the robot to get to some spatial locations, the displacement problem arises often, and several, or even many, instances of the Robot Displacement Problem usually populate the agent agenda at a time. The Displacement Problem decomposes in several sub-problems: path-planning, physical trajectory-planning and motor action. For a path-planning problem, the agent could apply an algorithm to obtain the optimum path on a spaces-accesses' graph representing the robot navigation environment. However, since the navigation space is not fully observable from any current robot location, the displacement plans would usually need to be recalculated while the robot applies them and gets new

environmental observations. The emotional approach instead, decomposes the spaces-accesses graph in levels of confidence, e.g. by considering spaces-accesses defined by objects with different location volatility: static objects (e.g. architectural elements), moveable objects (e.g. furniture and machines), and mobile objects (e.g. people and other robots). Then it applies the path finding for each of the subspaces and keeps open different path alternatives while the robot is already moving. The agent searches the spaces-accesses' graph hierarchically. The path-planning problems generate new path-planning sub-problems while their respective sub-plans can be refined. A non-intrinsic emotion (Generate New Path) in the problem resolution method is in charged of this process. When the path's refinement has gotten at an end, a different non-intrinsic emotion (Generate Trajectory) creates a new trajectory-planning problem. Furthermore, the trajectory-planning problems create motor-action-planning problems.
The full set of problems in the agenda is arranged on a tree-structure.

Although the agent currently attends only the most motivated problems (until the processing resources get to saturation) the agent must periodically evaluate the full set of emotions to motivate all the problems in the agenda. Thus, the emotional processes cause an extra workload in this architecture. The size of the agenda however, is partially auto-controlled, because in order to create new sub-problems, the parent problems need to get the attention of the processor to reach the emotional decision points, where the new sub-problems are created. Additionally, a proper definition of the agent character permits the control of the emotional sensitivity and limits the emotional state parameter $\varepsilon = t_e / T_a$ (See paragraph 3.3)

Despite the extra workload, this emotional approach presents some important benefits. The benefits consist of the explicit separation of the attention process, and the use of an emotional motivation mechanism, which explicitly shows its decision criteria and is configured in a centralized way (defining the agent character). To minimize the impact of the emotional workload the architecture defines a simple reactive model for the emotional motivation processes. This model permits its sequential or parallel execution.

Fig. 5 and Fig. 6 detail the emotional motivation process. Fig. 5 shows the appraisal of the observed situation.

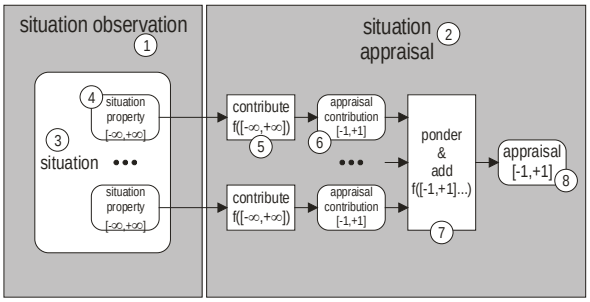

Fig. 5 Situation appraisal process

Situations (3) are generated by observation processes (1) and are represented as real properties (4). The appraisal process (2) depends on the agent character. The character dynamically adjusts the parameters of this process. To calculate the appraisal of the situation (8) the agent ponders and adds (7) a set of appraisal contributions (6), which are evaluated using contribution functions (5). The number of situation appraisals for the type of applications considered in the experiments on average is 2M. Equation 1 represents the ith situation appraisal

$$a_i = \sum_{k=1}^{l} w_{ak} \cdot f_{ak}(p_k) \tag{1}$$

Where: p_k is the k^{th} property of the situation, f_{ak} is the k^{th} contribution function, w_{ak} is the weight of the function and l is the number of appraisal contribution in the range [1, 6].

Fig. 6 shows the emotional motivation process. This process has two phases: first, an emotional activation (9) sets an emotional state (10), and second, an emotional response builds and motivates a behavior (11). The emotional contributions (13), evaluated with contribution functions (12), are

pondered and added (14) to finally give an emotional state (15). The emotional contributions functions are defined in the real range [-1,+1] and the emotional state in [0, +1]. Every emotional state is labeled in the robotic agent navigation problem (e.g., “fear of collision”), a 0 level would mean “no fear”, while a 1 level would be “afraid”. The emotional response generates new desires (16, 17), and motivates the behavior to accomplish the desire (18, 19). The number of emotions involved in the proposed robotic applications on average is 0.5 M.

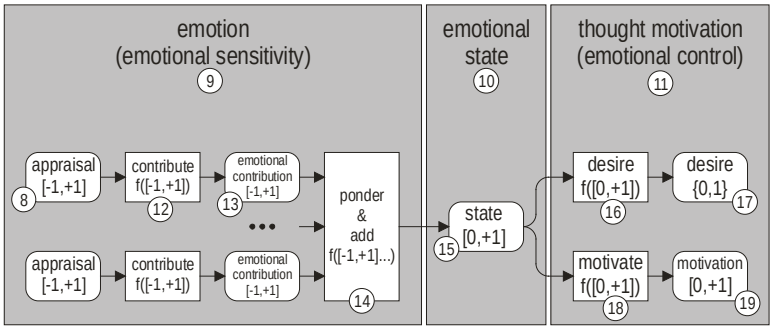

Fig. 6 Emotional motivation process

The emotion is expressed in equation 2.

$$s_j = \sum_{i=1}^{I} w_{ci} \cdot f_{ci}(a_i) \tag{2}$$

Where: s_j is the state of the j^{th} emotion, f_{ci} is the i^{th} contribution function, w_{ci} is the weight of the function.

The emotion contribution functions, f_{ci}, have properties such as slight variations at the ends of the range that tend to asymptotic values and abrupt variations around an inflection point in the center of the range. These properties are found in the hyperbolic tangent functions, which are used to represent the contribution functions as shown in equation 3.

$$th(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{3}$$

Where x is the appraisal value a_i when calculating the emotion. To allow adjusting the hyperbolic function, equation 3 is transformed in the function shown in equation 4, where the parameters x_0, y_0, δx *and* δy permit to translate and scale the contribution function.

$$th^*(x) = \left(\frac{e^{2(x-x_0)\delta x} - 1}{e^{2(x-x_0)\delta x} + 1} - y_0\right)\delta y \tag{4}$$

These emotions are grouped in the emotional system shown in Fig. 7. The emotional system gets, in a given instant, inputs from a set of N situation appraisals (e. g., 2M) and produces a set of K motivations (e. g., 0.5M). Each emotion can be composed of up to 6 different contributions functions. These contributions have the structure of the Dot-Product functions, each of them consists of: a hyperbolic tangent, a multiplication by the weight, a sum of the different contributions and the identity function. The total number of these Dot-Product functions in the emotional system depends on the complexity of the problem, the environment conditions and the robotic agent dynamics. In the experiments section, applying the multi-objective robotic agent, this number reaches a value of about 200 Million contribution Operations per Second (MOPS).

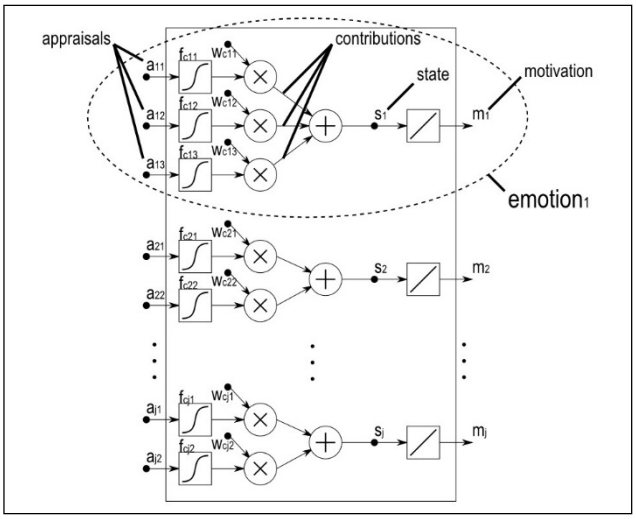

Fig. 7 Emotional system structure

These Dot-Product functions can be computed in parallel since they are independent in the sense that they perceive the situation and have to evaluate it and generate an emotional state. However, in the initial implementation they were executed sequentially in a main computer but due to their highly computational requirements, they surpass the capacity of the main processor, leading to the impossibility of executing the rest of the process applications and hence unfulfilling the robotic agent objectives. This situation is analyzed in the next subsection to propose a parallel implementation of the emotional system.

3.3 Emotional system computational requirements

The robotic agent executes observation, decision, and action application processes periodically. A period is called the attention cycle and is represented by T_a (see Fig. 8). T_a depends on the problem dynamics (e. g., robot speed). Besides of the application processes, the system must execute the operational processes of the emotional system of Fig. 7. The temporal cost, t_e, of executing the emotion processes is represented in grey color in Fig. 8 while the application processes are represented in white color. The agent needs to balance between the costs of the execution of the application processes and the emotional processes.

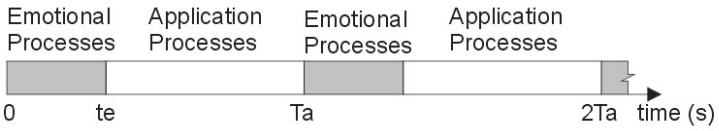

Fig. 8 Attention cycle

The workload of the emotional processes strongly depends on the application problem: the definition of the type of services, the complexity of the navigation and operation environment, and the amount of allowed simultaneous service requests.

Firstly, the emotional workload depends on the number of problems in the agenda, which is variable over time. Since the type of the problem defines the number of emotional decision points in the method that resolves the problem, the nature of those problems also affects the workload. The model of the emotional motivation process lets a variable number of emotional contributions, so, the Number of Emotional Contributions, identified as O (for Operations), or MO (for Millions of Operations), is a better choice for estimating the workload than just the Number of Emotions.

Equation 5 estimates the workload of the emotional processes in Number of Operations.

$$O = O\left(n_p\left(n_{ic} + n_e \cdot n_{nic}\right)\right) \quad (5)$$

being:

- n_p is the number of problems in the agenda.
- n_e is the mean of the number of non-intrinsic emotions per problem.
- n_{ic} is the number of emotional contributions of the intrinsic emotions, which is a constant value specified by the agent character.
- n_{nic} is the mean of the number of emotional contributions of non-intrinsic emotions, which is variable with the type of problems and the distribution over type)

In order to estimate the necessary processor throughput however, it is necessary to calculate the Number of Operations per second. Equation 6 estimates the workload of the emotional processes in OPS, being Ta the Attention Cycle Period in seconds.

$$OPS = \frac{O}{T_a} \quad (6)$$

Arbitrarily, we have defined three complexity problem levels: Simple, Normal and Complex, related to the number of problems to be processed, which stablish different service performance of the robotic platform, and three risk of robot collision levels: Safe, Normal and Risky, related to the robot navigation speed.

The computational costs of the emotional contributions vary depending on the complexity of the application problems. In the worst case, the number of millions of contributions functions per second is about 200. This growth of the emotional workload can put in danger the accomplishment of the objectives of the application. That is, if the time t_e dedicated to the emotional computation is too high, then the remaining time (T_a - t_e) will not be enough to execute the applications processes. Therefore, the goal is to minimize t_e as much as possible, hence the robotic agent can dedicate more time to solve practical problems and less time to the emotional processing. To this end, this paper proposes the design of the emotional system in hardware processors.

The selection of the hardware architecture to implement the emotional system will depend on the nature and the volume of the emotional processes. As shown before, the emotions have the Dot-Product functions structure computed iteratively in each attention cycle. This structure is very suitable for its implementation on FPGA processors. On the other hand, the volume of these operations is significantly high (200 MOPS) and cannot be tackled by low-medium performance FPGA. For instance, Stratix EP1S20F484I6 provides only 15 MOPS, which is not enough to cope with the application requirements. These low cost FPGA are used to implement reactive controllers (e. g., manipulators) where the required computational power is low, but they are not adequate in deliberative decision-making processes where millions of alternatives have to be computed in a given instant. Therefore, in this work, Stratix III and IV are used for the implementation of the emotional system. The available resources of Stratix FPGAs permit the synthesis of the communication IP cores necessary to communicate the emotional coprocessor with the main processor, which has been an additional reason for selecting this FPGA family.

A second implementation alternative is the use of Multicore processors. This architecture is also adequate because it permits the distribution of the emotional processes among the cores and their execution in parallel. The high throughput of the multicore will allow reducing substantially the t_e time. A six-core i-7 processor is used to implement the whole architecture. Both implementations are compared to show their performance when undertaking the different robotic agent application problems.

4 Emotional Processor Architecture Design

4.1 Robotic agent application

The FPGA/Multicore emotional processor is designed to tackle robotic agent applications as shown in Fig. 9. The multipurpose robotic agent performs activities such as diagnosis, transportation, cleaning, and surveillance simultaneously. Initially, the single-core (control computer) was executing the whole workload of the robotic agent (i. e., application processes and emotional processes). In the proposed design, the emotional processes are transferred to the FPGA/Multicore to allow the single core solve more complex problems.

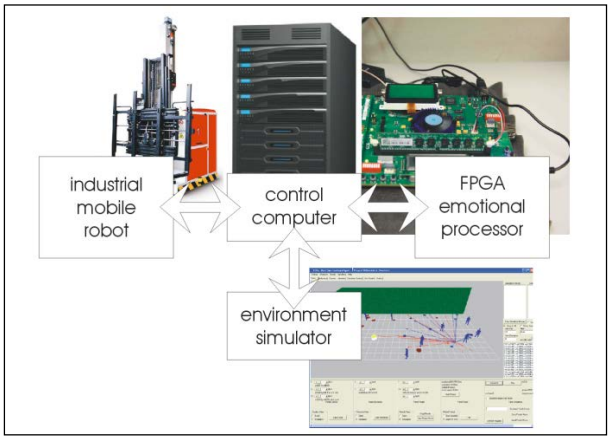

Fig. 9 FPGA Stratix III based robotic agent architecture

To define the emotional computational workload of the applications, a simulator of the agent environment is used (see Fig. 10). This simulator allows defining different scenarios where the aforementioned activities are tackled (e. g., surveillance).

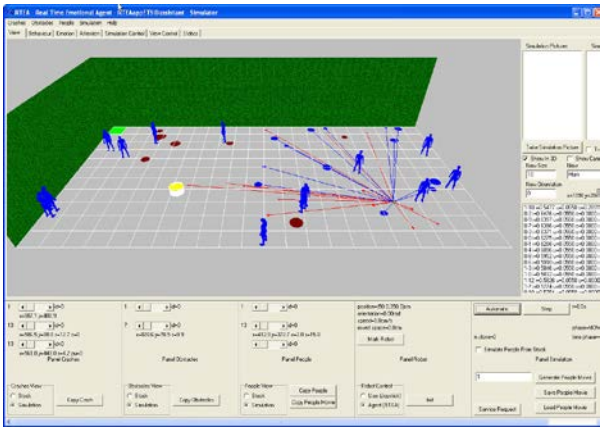

Fig. 10 Agent solving a crash in the operating area

The robot speed, the number of objects in the operation area, and the collision risk factors are used to define the attention cycle of the robotic agent, Ta, which is defined in the range of [0.1s , 0.5s]. Table 1 shows the robotic agent speed values used in the experiments.

Table 1 Robot speed

Safe	Normal	Risky
0.1 m/s	1 m/s	2 m/s

Table 2 shows the values of the complexity of three types of applications, measured in millions of emotion operations per attention cycle (Mopc). To obtain these values, applications of different

complexities are run in the simulator environment and the number of emotions involved in each of the applications is calculated. A simple problem requires the execution of about 0.5Mopc, while a complex application involves the execution of 2Mopc.

Table 2 Complexity of application problems

Number of Emotional Operations per Attention Cycle		
Simple	Normal	Complex
0.5 Mopc	1 Mopc	2 Mopc

The emotional state of the robotic agent represents the ratio between the time spent to execute the emotional processes and the time of the attention cycle as shown in equation (7).

$$\varepsilon = t_e / T_a \tag{7}$$

Where, t_e is the processing time of the emotions.

Three robotic agent emotional states are considered in the experiments (i. e., relaxed, normal and stressed). In the ideal situation, the emotional computational time, t_e, in the relaxed mode is less than 10% of T_a, in the normal mode it is between 10% and 25%, and in the stressed mode it is between 25% and 40% as shown in Fig. 11. A workload higher than 40% is not be acceptable because the process applications are stalled and the robotic agent cannot fulfill the objectives.

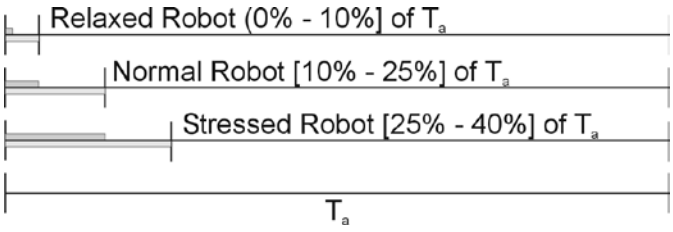

Fig. 11 Emotional workload limits

In the relaxed mode, the robot dedicates less time to the emotional processing and more time to solve application problems. That is, using the FPGA, this device will have less than 10% of T_a to compute the emotional processes. In the opposite, for the stressed mode, the robotic agent dedicates more time to the emotional processing and hence the FPGA will have between 25% and 40% of T_a to process the emotions. This means that the throughput of the FPGA in the relaxed mode is much higher than in the stressed mode. Table 3 shows the three emotional states used in the experiments, when the robotic agent undertakes the resolution of the different problems.

Table 3 Agent global emotional state ε

Relaxed	Normal	Stressed
< 0.1	[0.1, 0.25]	[0.25, 0.4]

The environment simulator (see Fig. 10) has been programmed with different robotic applications where the different combinations of the complexities of the problems, the robot speeds, and the emotional states are applied in order to calculate the emotional computational costs. Table 4 summarizes the obtained emotional costs, measured in MOPS, for each of the robotic scenarios.

Table 4 Emotional processing cost for robotic agent applications (MOPS)

	Robot speed

Problem	Robotic agent state	*0.1 m/s*	*1m/s*	*2m/s*
Simple	Stressed	3	5	11
	Normal	4	8	17
	Relaxed	13	25	51
Normal	Stressed	6	11	19
	Normal	8	17	33
	Relaxed	26	49	99
Complex	Stressed	9	21	39
	Normal	17	33	68
	Relaxed	51	99	200

The emotional computational requirements shown in Table 4 have to be fulfilled by the FPGA processor in order to allow tackling the corresponding type of problem. For instance, a simple problem using a relaxed robot running at the maximum speed (see Table 4) will require an FPGA that can process at least 51MOPS. If the FPGA is not able to support this throughput the robot will fail to solve this problem.

4.2 FPGA based emotional system design

In this section, the implementation of the emotional system presented in section 3 is developed using Stratix III and IV FPGA's.

4.2.1 Emotional processor design

Fig. 12 shows a block diagram of the FPGA based agent control system. The single-core executes the application processes of the behavior system, while the FPGA implements the operational processes of the emotional system.

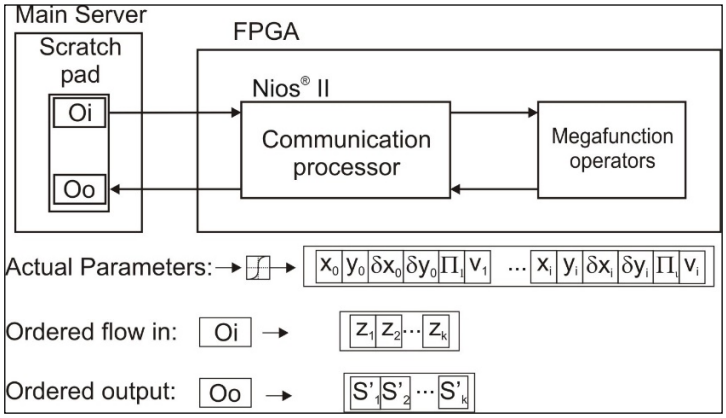

Fig. 12 Emotional processor

The Megafunction operators block implements the emotional processes and contribution functions. The Nios II is a software processor (IP by Altera) used to communicate the FPGA emotional processor with the main server of the robotic agent controller. Operators of the Altera Megafunction floating point library work synchronously and are implemented on a segmented basis. This makes it possible to process a continuous stream of data (pipeline) at the operating frequency of the FPGA device. The latency of each operator is due to its internal structure and the algorithm in which it is based. FIFO buffers are used to synchronize the operations according to their latencies.

The proposed design is based on A3 methodology to exploit the parallelism of the emotional processes so as to find the potential factorization with the aim of using the minimum number of operators to process the maximum number of operations. The emotional system is based on Dot-Product functions (contributions), which are applied to the appraisal data. Since emotions can have a variable number of contributions, the emotional processor is designed around an emotion operator, which processes a sequence of contributions and adds them to the emotional state.

The Data Flow Graph of an emotion operator is shown in Fig. 13, where an example of the

execution sequence of two emotions is shown. The data input is performed through the FIFO-in and the emotional states are obtained through the FIFO-out. The flow rate of the obtained emotional states at the output is lower than the flow rate of the input parameters (6 parameters * number of emotional contributions). The first emotion, with two contributions, completes its evaluation at t4. The second one has 4 contributions and finishes at t8. Emotional contributions are based on hyperbolic tangent functions (th) and each contribution has a weight (w). Although Fig. 13 shows the temporal separation between the processing of each contribution on the same operator in a simplified way, in the pipeline each operator processes several contributions overlapped in time (as many as the size of the pipeline).

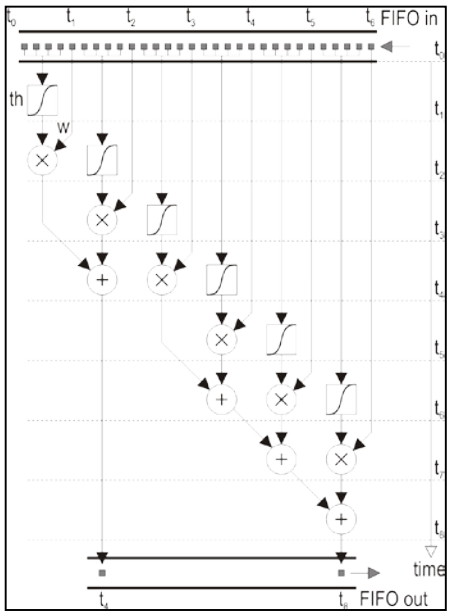

Fig. 13 Data flow graph for two emotions

The pipelined-processing of the emotional contributions can reach a frequency in between 200MHz and 300MHz depending on the design synthesized on the FPGA. However, the accumulation phase of the contributions in an emotional state must be performed with a feedback adder, so that the processing rate must be reduced depending on the latency of the addition operator (14 cycles). To avoid this latency, the main processor, when it sends the data to the emotional processor, it interleaves the contribution parameters of groups of emotions (14 emotions) in order to accommodate the data flow to the latency of the accumulation phase.

Using ordered IN and OUT data flows a performance of 265 millions of emotional contributions per second (MOPS) can be achieved. In addition to the pipelined processing, the emotion operator has been also replicated to process in parallel. FPGA devices used in the experiment have sufficient resources to implement multiple emotion operator replicas (7 in Stratix III and 8 in Stratix IV). In this case, multiplexers are used to distribute the data to the emotion operator replicas as shown in Fig. 14.

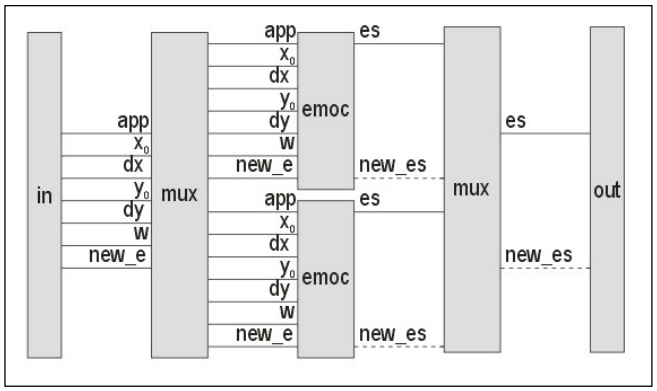

Fig. 14 Two replicas of the emotion operator

The input stream receives the parameters: *x0*, *dx*, *y0*, *dy*, *w*, *app* (appraisal) and *new_e* (start new emotion), then the multiplexers send the parameters to the emotion operators. Finally, the output stream sends the emotional state, *es*, and the *new_es* signal, indicating its availability.

To implement the emotion, a modular design is followed. First, the hyperbolic tangent is implemented using the available megafunctions. The latency of this function is 56 clock cycles. Fig. 15 shows the implementation of the parameterized hyperbolic tangent. FIFO queues are used to adjust the latencies and synchronize the parameters. The total latency of this process is 106 clock cycles.

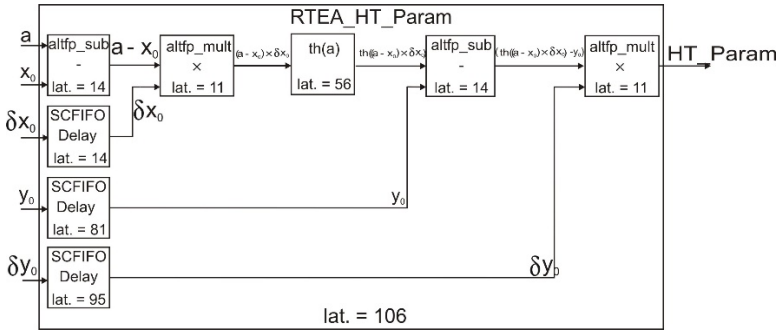

Fig. 15 Parameterized hyperbolic tangent

The parameterized hyperbolic tangent is multiplied by the weight *w* to obtain an emotional contribution; this process has a latency of 117 clock cycles. Finally, an accumulation phase completes the emotion operator with a latency of 131 clock cycles (see Fig. 16).

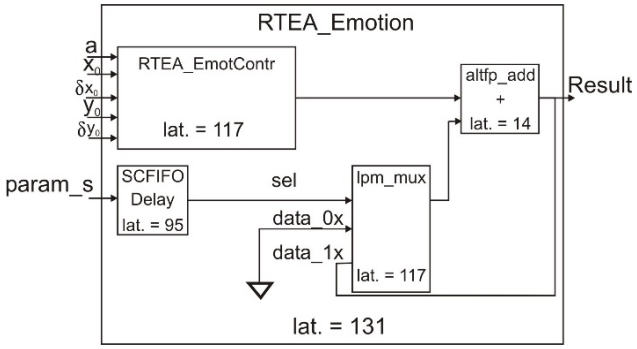

Fig. 16 Emotion operator

The emotion operator (see Fig. 16) is replicated in the selected FPGAs (Stratix III and IV) to process emotions in parallel. This replication will allow increase the throughput to calculate the millions of emotions composing the emotional system.

4.2.2 Design synthesis of the emotional system

The emotional system is simulated on Stratix III and Stratix IV and synthesized on Stratix III. The EP3SL150F1152C2N device is implemented on a Stratix III EP3SL150N development board, which is the available system in our laboratory. Likewise, the processor is simulated on a Stratix IV, implementing the EP4SGX230KF40C2 device, since it is a FPGA family widely available in the industry. The type of resource that has limited the design is the number of integrated DSPs (Stratix III: 112, Stratix IV: 161).

Quartus II software, which allows structural design, is used together with Megafunction library. This library defines arithmetic operators for real floating point numbers IEEE-754, with simple (32-bits) or double (64-bits) precisions. The developer can define other formats, since the exponent and mantissa fields are configurable. Instead of codifying the basic processing resources directly in VHDL the Altera Megafunction library has been chosen because its functions are optimized for Altera devices and most of them let the developer to choose the optimization criteria (speed or area). In the proposed design, floating point real numbers are used as simple precision. Integer or fixed point real number representations were discarded in front of the floating point, because this representation has the following advantages for our design:

(i) provides a direct interface with the processes in the main processor avoiding the time spent in the process of transformation of the real numbers represented in main processor and FPGA.

(ii) facilitates the performance analysis between FPGA and Multicore since the same representation in both systems is used.

(iii) and allows flexibility to adjust the bit-widths of the operators which allows us to modify the range and accuracy of the values to suit different types of emotional sensitivity. These are important features of the research, and the use of this representation is not an excessive penalty, as we have enough resources in the considered Stratix FPGA devices.

The Megafunctions listed in Table 5 are used, and the option to optimize the area is selected. The latency, when there are different function versions is set to maximize the clock frequency allowed for a specific function. Table 5 shows the parameters for the Stratix III device (ALUT – Adaptive Look-Up Table, DLR – Dedicated Logic Register, ALM – Adaptive Logic Module and DSP is an 18-bit DSP).

Table 5 Megafunctions used in the emotional processor design

ALTFP Function	Latency (clock cycles)	f_{MAX} (MHz)	ALUT	DLR	ALM	DSP
ADD_SUB	14 (high latency)	416	599	603	427	-
DIV	14 (low)	296	295	331	262	16
MULT	11 (high)	466	195	301	206	4
EXP	17 (low)	275	631	521	445	19

Table 6 shows the resource utilizations of Stratix III and Stratix IV, summarized for the designed process emotion block. The values of the Stratix IV are represented in brackets. The utilization is represented as a percentage of the available resources, which permits to have an initial estimation of the number of replicas of every synthesizable emotional operator on each device.

Table 6 Resources utilization using Stratix III and Stratix IV

Type of Resources	Total Resources	**Emotion** 5% (4%)
Combinational ALUTs	*113,600 (182,400)*	4,091-4% (3,699 - 2%)
Memory ALUTs	*56,800 (91,200)*	662-1% (0 - 0%)
Dedicated Logic Registers	*113,600 (182,400)*	5,640-5% (6,867 - 4%)
Total Block Memory Bits	*5,630,976 (14,625,792)*	17,536 -< 1% (17,536 -< 1%)
DSP Block 18-bit Elements	*384 (1288)*	51-13% (51-4%)

Table 7 shows the maximum operation clock frequency for the emotional processor depending on the selected device. TimeQuest Static Timing Analyzer is used to obtain these values.

Table 7 FPGA frequency

Device	clock - F_{Max} (MHz)
Stratix III	265.67
Stratix IV	311.53

The maximum frequency, Fmax, is shown for a single instance of the emotional process. So, it is possible to maintain an in-flow (situation appraisals) at Fmax frequency, then the emotional processor would be able to evaluate 265MOPS and 311MOPS, respectively. This means that after a latency of 131 clock cycles, a constant output flow (emotional states) at 265MOPS or 311MOPS, depending on the device, can be obtained.

Table 8 shows the effect on the Fmax when the emotion operator is replicated into the emotional processor. This table shows the maximum number of replicas implemented in the FPGA devices due to the limitation of their available resources (the number of DSPs in the proposed design).

Table 8 Fmax with emotion replication

Device	Number of replicas	clock - FMax (MHz)
Stratix III	4	253.42 (x0.95)
	7	209.64 (x0.79)
Stratix IV	4	269.54 (x0.86)
	8	230.47 (x0.74)

When dealing with the emotion operator replication in the FPGA, a key point is the bottleneck of the communication between the FPGA and the single-core. In the proposed design, 24 bytes (six simple floating point parameters x0i, y0i, δxi, δyi, wi, ai) per contribution function are sent over the input flow. In a complex problem (highest speed and relaxed robotic agent) where 2M emotional contributions are executed per attention cycle, the number of bytes transmitted is 24 bytes x 200 Millions functions/s, which means 38.4GB/s. To transmit this data flow, a PCIe 3.0 interface at 40Gb/s is suitable.

Table 9 shows the performance of the communication IP cores currently available for Altera FPGA devices. The selected communication interfaces of each device are selected to maximize the throughput: Ethernet for Straix III and PCIexpress for Stratix IV. As shown in Table 9, the performance of the emotional processor in this case is limited to 40MOPS using Stratix III, and to 208MOPS with Stratix IV.

Table 9 FPGA performance for the available IP communication cores

Bus performance	(Gb/s)	Processor performance	(Mops)
PCIe 3.0	40	Stratix IV	208
USB 3.0	4.8	S. III-E, S. IV-GX, S. IV-GT	25
HyperTransport 3.1	10	S. IV	40
Ethernet	10	S. III, S. IV	40
	40	S. IV-GT	160

4.3 Partitioned Multicore based emotional system design

Regarding the implementation on a Multicore, the robotic agent architecture including the belief, behavior, attention and emotional sub-systems is implemented as a partitioned system [22] on a six-Core processor; the Intel Core i7-980X at 3.33GHz per core. The i7 based computer has 8MB cache memory, 12 GB DDR3 RAM. One of the cores is dedicated to the attention system, a second core is used for the behavior-belief systems and the remaining 4 cores implement the emotional system. The number of active cores for the emotional system can be configured by the operating system, using the *sched_setaffinity()* system call in Linux. For evaluations purposes, the emotional processor is run as Single-core mode, Dual-core and Quad-core in order to assess which is the sufficient number of cores to tackle the robotic applications. In the Single-core mode, all the emotional processes are assigned to one processor. In the Dual and Quad core modes, the processes are distributed evenly among the different cores. To this end, the Worst Fit algorithm is

used to refill the cores [20]. The system model implemented is shown in Fig. 17.

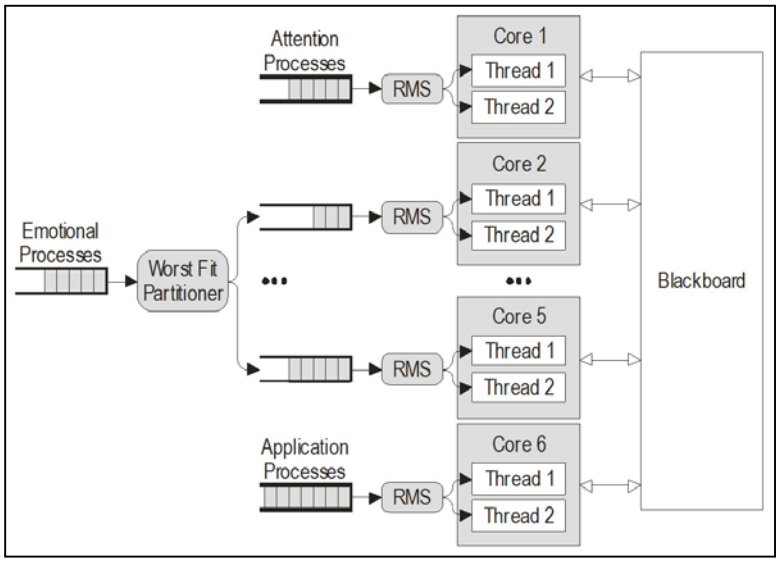

Fig. 17 Multicore architecture

Initially, the WF replenishes the core with the emotional processes. The policy of this strategy is to assign the process to the less loaded core until all the processes are assigned to their corresponding core. Each core implements a Rate Monotonic Scheduler to execute the processes belonging to that core [21]. That is, the process with the shortest period has the highest priority for execution in the core. The schedulers and the processes are implemented as real-time tasks in the rt-linux kernel. The real-time processes are executed using the main memory without accessing the disk device.

The cores share a memory structure called blackboard [23]. Through this memory the processes read and write the important data as the appraisals, emotional states and motivations. Also it allows, using shared variables to synchronize the different processes. The attention core, at each attention cycle, activates the Belief processes that write in the blackboard the appraisal information. The emotional processes then read the appraisals and calculate the emotions to update the motivations in the memory. Finally, the Behavior processes read the motivations and execute the actions prioritizing the behaviors with highest motivations. These operations are repeated periodically at each attention cycle.

5 Experimental evaluation

In this section, the implementations of the emotional architecture in FPGAs Altera Stratix III, IV and Multicore processors are compared. The comparison is performed by executing on each of the platforms different complexity levels applications of the robotic agent. The evaluation is focused on the analysis of the performance, measured on MOPS, that FPGAs Stratix III and IV, and multicore provide to solve the agent problems.

Figures 18 to 26 show the results of evaluating the different implementation alternatives, considering 3 complexity problems (Simple, Normal and Complex) and 3 robotic agent emotional state levels (Stressed, Normal and Relaxed) for 3 different values of the robot speeds (0.1m/s, 1m/s and 2m/s). In each Figure, the bars represent the maximum computation capacity in MOPS that each processor or FPGA can provide (e. g., Fig. 18 shows that a Stratix IV allows 208MOPS, a Quad-core performs 91MOPS while a Dual-Core reaches 46MOPS).

For each pair (complexity problem, robotic agent emotional state), the speed of the agent limits the computational capacity (horizontal lines) required for each processor to solve a specific problem. For instance, in the case of a (simple problem, relaxed robot), if the speed is 2m/s the minimum required computation capacity to solve the problem is 51MOPS, while at 0.1m/s it is

13MOPS (see Fig. 18). This is because the attention cycle increases as the speed is reduced and hence more time is available to solve the same problem, then less computation MOPS are required. At the former speed, only Quad-Core and Stratix IV can solve this kind of problems, while at the latter speed all the considered processors can tackle the problem. In general, for the same type of problems, at higher speeds, the computational requirements of the processors increase.

On the other hand, as the complexity of the problem increases, the processor computation requirements also increase. For instance, for a normal problem and relaxed robot at 1m/s the required MOPS are 49 (see Fig. 21), while a complex problem with the same relaxed robot at the same speed, requires 99 MOPS (see Fig. 24). Moreover, for the same kind of problems, if the robot emotional state is becoming more stressed, then the FPGA/Multicore computational requirements decrease because the execution time of the emotions is higher, and hence less computing power is required from the FPGA-Multicore processors. For instance, Fig. 24 shows that for a complex problem and relaxed robot at 2m/s, 200MOPS are required, while Fig 26 shows that the same problem with a stressed robot at the same speed requires only 39MOPS. This is due to the fact that if a relaxed robot is desired then more throughput is required from the FPGA/Multicore.

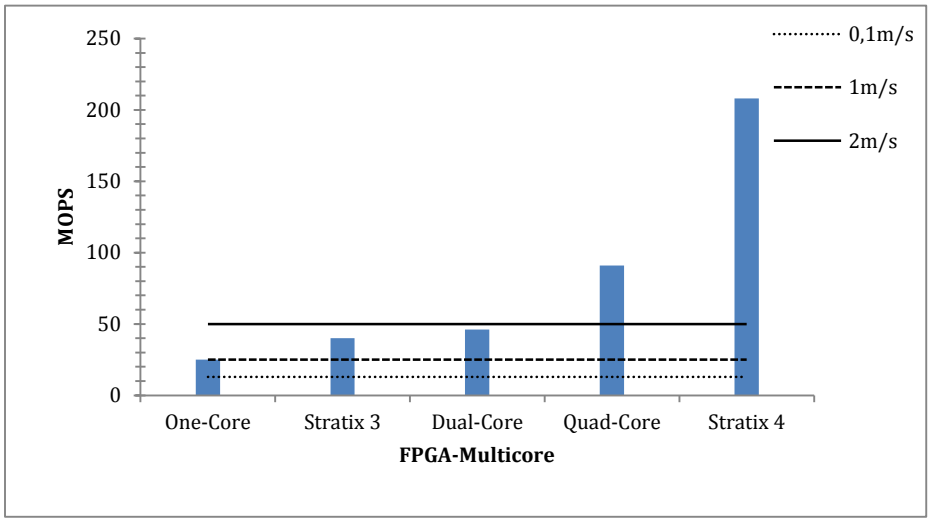

Fig. 18 Simple problem – Relaxed robotic agent

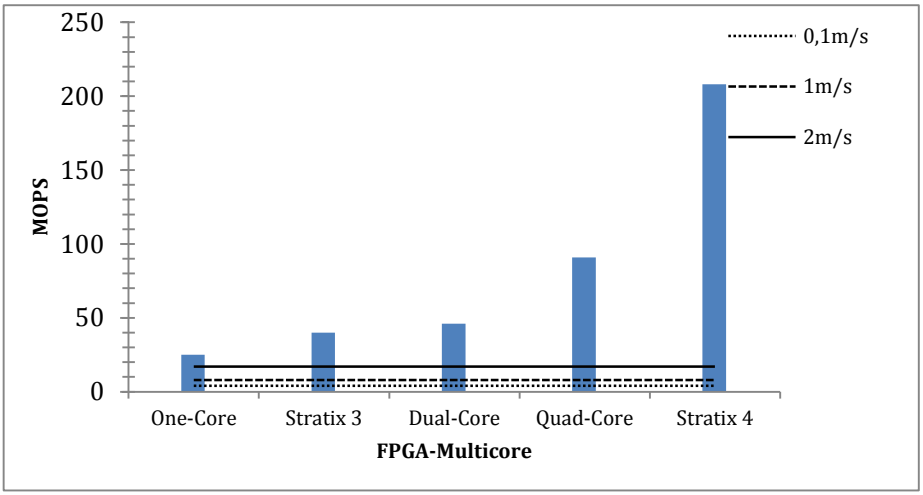

Fig. 19 Simple problem – Normal robotic agent

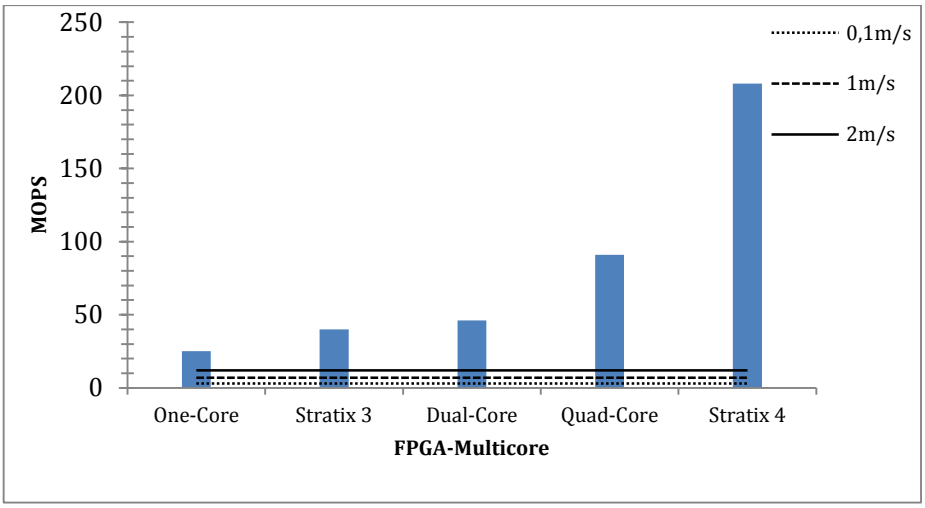

Fig. 20 Simple problem – Stressed robotic agent

Analyzing the results in more details, it is noted that all the platforms (single-core, Stratix III, Dual-Core, Quad-Core, Stratix IV) can solve simple problems with a stressed or normal agent at any speed. However, with a relaxed agent, single-core can only support the application if the agent runs at the lowest speed (0.1m/s), Stratix III and Dual-Core solve the problem at low and medium speeds, while Stratix IV can afford it even at the maximum speed.

For the normal defined problems, when the agent is stressed all the evaluated processors can solve the applications at any speed. Using a normal agent, only the single-core fails to solve the problems at the maximum speed. A relaxed agent can solve the applications by using Stratix III, Dual-Core, Quad-Core and Stratix IV at 0.1 m/s. If the spped is increased at 1 m/s only Quad-Core and Stratix 4 can solve the problems. Finally, at 2m/s only Stratix IV can tackle the situation.

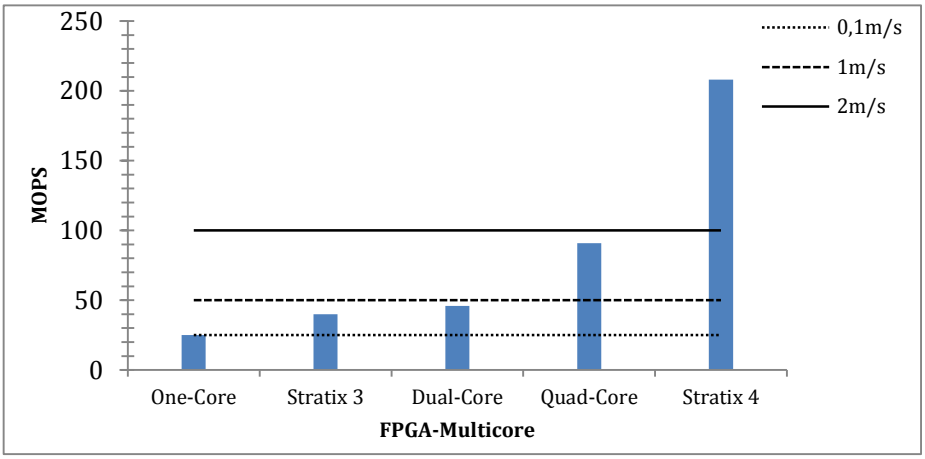

Fig. 21 Normal problem – Relaxed robotic agent

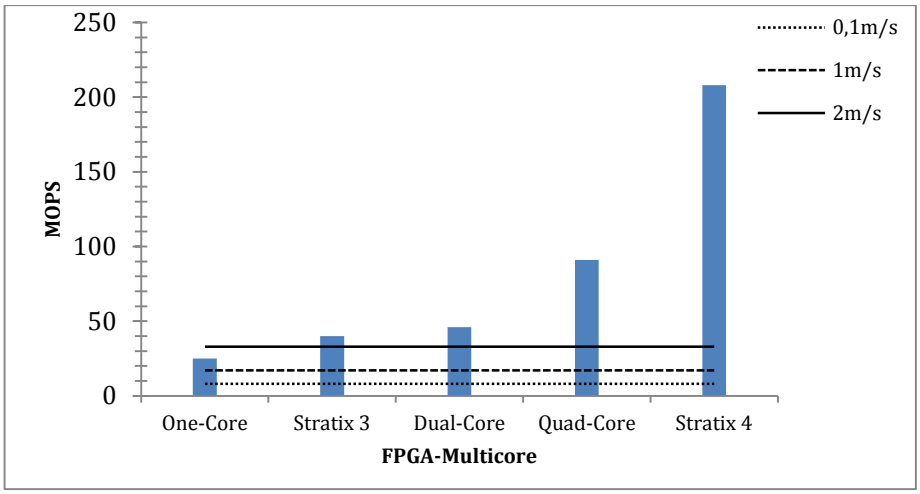

Fig. 22 Normal problem – Normal robotic agent

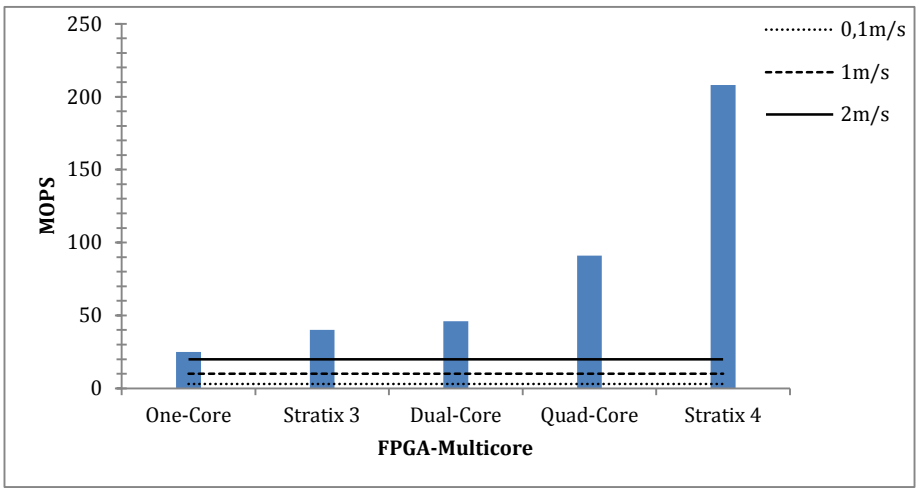

Fig. 23 Normal problem – Stressed robotic agent

Regarding complex problems, using a stressed agent at low and medium speeds, all solutions solve the applications. At the maximum speed, only Dual-core, Quad-core and Stratix IV are applicable. For a normal agent, at the minimum speed all solutions are valid. At 1m/s a single-core cannot solve the situation. Increasing the speed at 2m/s only Quad-Core and Stratix IV can solve the applications. If the agent is relaxed, at the minimum speed, only Quad-core and Stratix IV are adequate solutions. Increasing the speed more than 1m/s causes that only Stratix IV is a valid solution to solve the application.

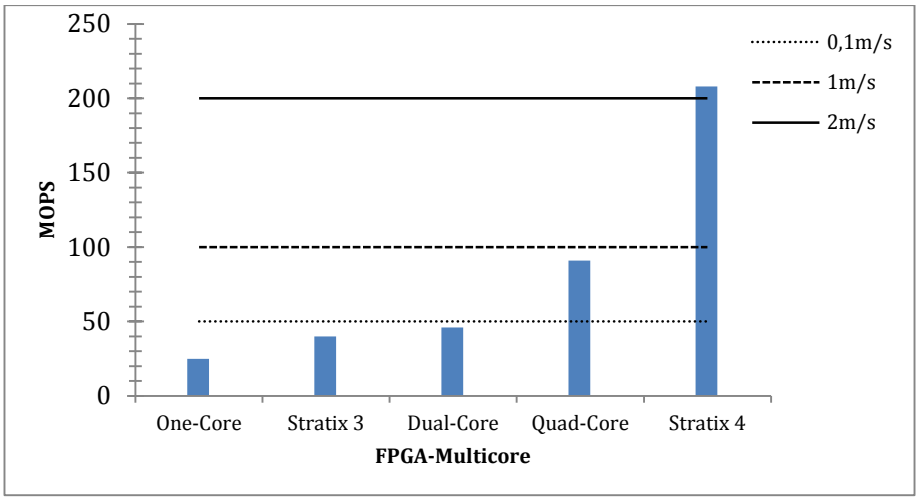

Fig. 24 Complex problem – Relaxed robotic agent

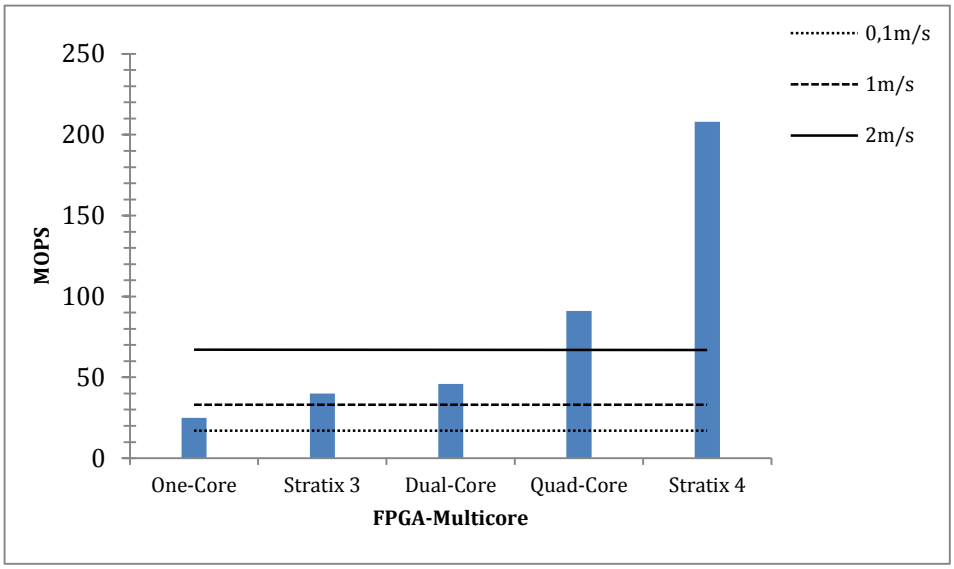

Fig. 25 Complex problem – Normal robotic agent

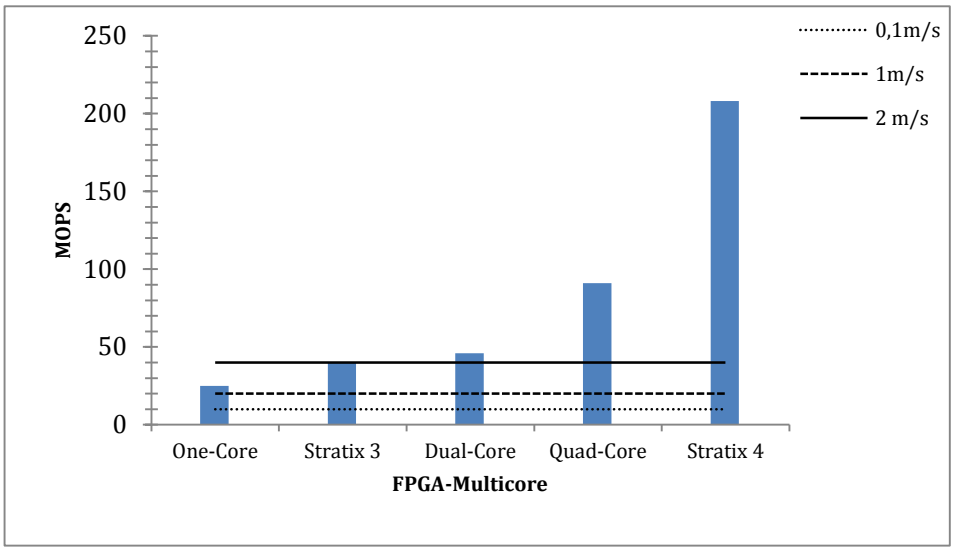

Fig. 26 Complex problem – Stressed robotic agent

As a summary of the evaluation, it can be pointed out that the obtained performance of the Stratix IV implementation of the emotional processor increases the performance of the initial implementation of the architecture in about one order of magnitude. As a consequence, all the complex applications that could never be executed using the initial version using a single-core (max. 25MOPS) now they can be undertaken (e.g., complex problem and relaxed robot -min. 50MOPS).

The study shows also that other less expensive solutions using Stratix III could be applied to solve problems (e.g., complex problem and normal agent at 1m/s -38MOPS) with around the double of the requirements that a single-core could support. Dual-Core provides slightly better performance than stratix III, so it can be used to solve some problems that Stratix III cannot solve, such as the complex problem and normal agent at 1m/s (33MOPS). Moreover, Dual-core is relatively cheaper so it is a better choice than Stratix III.

Using Quad-Core, the performance of the architecture is increased in 3.64 times in relation to the first implementation. Thus, from the 27 proposed applications about 89% can be solved. However, using the original implementation only 55 % can be tackled. Furthermore, Quad-Core has a lower cost than a Stratix IV, so more adequate solution but only if the type of applications to carry out is not the most complex one.

6 Conclusions

An FPGA based emotional control architecture to implement future robotic agents has been presented. The emotional processes of the architecture have high computational requirements, which consumes the computational power of the main processor. To reduce this consumption, the parallel capabilities of the emotional processes of the architecture have been exploited and the implementation of the emotional processes on high performance FPGA processors has been tackled. An industrial mobile robotic agent application (under different environmental, dynamic and emotional robot state conditions), implementing the emotional based FPGA architecture has been proposed. The performances have been evaluated for FPGAs Altera Stratix III and IV, and the results are compared with the implemented emotional system in a Single-Core, Dual-Core and Quad-Core. Results show that Stratix IV implementation of the emotional processor increases the performance of the initial implementation of the architecture in about one order of magnitude. Stratix III could be applied to solve problems with around the double of the requirements that a single-core could support. Dual-Core provides slightly better performance than stratix III and it is relatively cheaper so it is a better choice than Stratix III. Using Quad-Core, the performance of the architecture is increased in 3.64 times in relation to the first implementation. Thus, about 89% of the proposed applications can be solved. Quad-Core has a lower cost than a Stratix IV, so more adequate solution but only if the type of applications to carry out is not the most complex one.

Acknowledgements

This work was supported in part under Spanish Grant PAID/2012/325 of "Programa de Apoyo a la Investigación y Desarrollo. Proyectos multidisciplinares", Universitat Politécnica de Valencia, Spain.

References

1. Malfaz M, Salichs MA (2010) Using MUDs as an experimental platform for testing a decision making system for self-motivated autonomous agents. Artificial Intelligence and the Simulation of Behaviour Journal. 2(1): 21-44
2. Damiano L, Cañamero L (2010) Constructing Emotions. Epistemological groundings and applications in robotics for a synthetic approach to emotions. Proceedings of AI-Inspired Biology Symposium (AIIB'2010): 20-28
3. Hawes N, Wyatt J, Sloman A (2009) Exploring design space for an integrated intelligent system. Knowledge Based Systems 22(7): 509-515
4. Sloman A (2009) Some Requirements for Human-Like Robots: Why the Recent Over-Emphasis on Embodiment Has Held Up Progress. Creating Brain-Like Intelligence 2009: 248-277
5. Moravec H (2009) Rise of the Robots. The Future of Artificial Intelligence. Scientific American. 23 March 2009.
6. Moshkina L, Arkin RC (2009) Beyond Humanoid Emotions: Incorporating Traits, Attitudes and Moods. IEEE International Conference on Robotics and Automation (ICRA'2009)
7. iRobot industrial robots website. http://www.irobot.com/gi/ground/. Access 22 September 2014
8. Lee-Johnson CP, Carnegie DA (2010) Mobile Robot Navigation Modulated by Artificial Emotions. IEEE Transactions On Systems, Man, And Cybernetics - Part B. 40(2): 469-480

9. Daglarli E, Temeltas H, Yesilogly M (2009) Behavioral task processing for cognitive robots using artificial emotions. Neurocomputing 2009
10. Ventura R, Pinto-Ferreira C (2009) Responding efficiently to relevant stimuli using an emotion-based agent architecture. Neurocomputing 2009
11. Del Campo I, Basterretxea K, Echanobe J, Bosque G, Doctor F (2012) A System-on-Chip Development of a Neuro–Fuzzy Embedded Agent for Ambient-Intelligence Environments. IEEE Transactions on Systems, Man, and Cybernetics - Part B. 42(2): 501-512
12. Altera Corporation (2010) Stratix III Device Handbook vol. 1
13. Altera Corporation (2012) Stratix IV Device Handbook vol. 1
14. Intel Corporation (2014) Desktop 4th Generation Intel Core Processor Family, Desktop Intel Pentium Processor Family, and Desktop Intel Celeron Processor Family – Datasheet vol. 1 and 2
15. Pedraza C, Castillo J, Martínez JI, Huerta P, Bosque JL, Cano J (2011) Genetic Algorithm for Boolean minimization in an FPGA cluster. The Journal of Supercomputing. 58(2): 244-252
16. Orlowska-Kowalska T, Kaminski M (2011) FPGA Implementation of the Multilayer Neural Network for the Speed Estimation of the Two-Mass Drive System. IEEE Transactions on Industrial Informatics. 7(3): 436-445
17. Arkin RC, Ulam P, Wagner AR (2012) Moral Decision-making in Autonomous Systems: Enforcement, Moral Emotions, Dignity, Trust and Deception. Proceedings of the IEEE. 100(3): 571-589
18. Salichs MA, Malfaz M (2012) A new approach to modelling emotions and their use on a decision making system for artificial agent. IEEE Transactions on affective computing. In Press. 3(1): 56-68
19. Naouar MW, Monmasson E, Naassani AA, Slama-Belkhodja I, Patin N (2007) FPGA-based current controllers for AC machine drives - A review. IEEE Transactions on Industrial Electronics. 54(4): 1907–1925
20. March JL, Sahuquillo J, Hassan H, Petit S, Duato J (2011) A New Energy-Aware Dynamic Task Set Partitioning Algorithm for Soft and Hard Embedded Real-Time Systems. The Computer Journal. 54(8):1282-1294
21. Lehoczky J, Sha, L, Ding Y (1989) The rate monotonic scheduling algorithm: exact characterization and average case behavior. IEEE Real-Time Systems Symposium, Santa Monica, Dec. 1989: 166–171
22. Seo E, Jeong J, Park S, Lee J (2008) Energy efficient scheduling of real-time tasks on multicore processors. IEEE Transactions on Parallel and Distributed Systems. 19(11): 1540–1552.
23. Ng-Thow-Hing V, Lim J, Wormer J, Sarvadevabhatla RK, Rocha C, Sakagami Y (2008) The memory game: Creating a human robot interactive scenario for ASIMO. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'2008): 779–786.
24. Thu Bui L, Abbass HA, Barlow M, Bender A (2012) Robustness Against the Decision-Maker's Attitude to Risk in Problems With Conflicting Objectives. IEEE Transactions on Evolutionary Computation. 16(1): 1-19
25. Pedrycz W, Song M (2011) Analytic Hierarchy Process (AHP) in Group Decision Making and its Optimization with an Allocation of Information Granularity. IEEE Transactions on Fuzzy Systems. 19(3): 527-539
26. Cassidy AS, Merolla P, Arthur JV, Esser SK, Jackson B, Alvarez-icaza R, Datta P, Sawada J, Wong TM, Feldman V, Amir A, Ben-dayan D, Mcquinn E, Risk WP, Modha DS (2013) Cognitive computing building block: A versatile and efficient digital neuron model for neurosynaptic cores. IEEE International Joint Conference on Neural Networks (IJCNN'2013).
27. IBM Cognitive Computing and Neurosynaptic chips website. http://www.research.ibm.com/cognitive-computing/neurosynaptic-chips.shtml. Access 22 September 2014