AN INTELLIGENT MULTI- OBJECTIVE EVOLUTIONARY SCHEDULERS TO SCHEDULE REALTIME TASKS FOR MULTICORE ARCHITECTURE BASED AUTOMOTIVE ELECTRONIC CONTROL UNITS

M. Lordwin Cecil Prabhaker[1] and K. Manivannan[2]

[1] Department of Electronics and Communication Engineering, Anna University – University College of Engineering – Dindigul, Dindigul, Tamilnadu, India – 624 622.

[2] Department of Computer Science and Engineering, PSNA College of Engineering and Technology, Dindigul, Tamilnadu, India – 624 622.

E-Mail: cecillord@gmail.com

Abstract

In an automotive electronics applications there are approximately 230 electronic control units ECU's are used to provide intelligent driving assistance. So, there is an effective multiple objective real time task scheduling techniques are required to provide better solution in this domain. This paper describes novel multiobjective evolutionary algorithmic techniques such as Multi - Objective Genetic Algorithm (MOGA), Non-dominated Sorting Genetic Algorithm (NSGA) and Multi - Objective Messy Genetic Algorithm (MOMGA) for scheduling real time tasks to a multicore processor based ECU. These techniques improve the performance upon earlier reported of an ECU's by considering multiple objectives such as, low power consumption (P), maximizing core utilization (U) and minimizing deadline missrate (δ). This work also analysis the schedulability of realtime tasks by computing the converging value of a series of task parameters such as execution time, release time, workload and arrival time. Finally, we investigated the performance parameters such as power consumption (P), deadline missrate (δ), and core utilization for the given architecture. The evaluation results show that the power consumption is reduced to about 5 - 8%, utilization of the core is increased about 10 % to 40% and deadline missrate is comparatively minimized with other scheduling approaches.

Key words: *Automotive Electronics, Multicore Architecture, Multiobjective evolutionary Algorithms, Scheduling Realtime Tasks.*

1. INTRODUCTION

In recent days, unmanned intelligent driving, safety and telematics are major technological improvement in an automobile industry. These features can be achieved through electronic control units (ECUs). There are approximately 230 ECUs are used in automotive functional domains such as, power train, chassis, body, telematics and safety. The ECUs will generate almost 650 million realtime tasks to provide better intelligent - safety – comfort in automotive applications (Dr. Preeti Bajaj et.al , 2011). The ECUs can be designed through advanced multicore processor, sensors, actuators, automation controllers, and infotainment / telematic devices (Simon Schliecker et.al, 2009). In such ECUs, the scheduling of realtime tasks is a major constraint in which power consumption, deadline missrate, core utilization and temperature are major performance parameters. To achieve, an optimum scheduling results in an automotive functional domains the following objectives has to be considered. 1. Low power consumption, 2. Maximum core (processor) utilization and 3. Minimum deadline missrate. The Multi – Objective Evolutionary Algorithms (MOEAs) address this problem and provides optimum scheduling results (Antonio J et.al, 2009). The multicore architecture based embedded controller for an automotive appliances consists of multiple processing cores with interconnecting networks. Each core on the controller may act as master or slave and it has adequate processing unit with its local input/output buses. Depending upon the architecture characteristics the cores are classified as homogeneous or heterogeneous. For inter and intra vehicular communication separate slave core are there (XieYong et.al, 2017). According to the application nature the core may work in various clock speeds. The following are the list of Communication protocol used in ECUs; 1.Standard communication protocol (eg. CAN, LIN, TTP, FlexRay), 2.User communication protocols (eg. RS232, USB) and 3. RF communication (eg. Bluetooth). The shared bus management authority manages the allocation bus for different ECUs. In electric car (M.Krügera et.al, 2017) presents a novel procedure which will be used for future design of ECUs. They also compared their findings with the expected distribution of temperature. Based on their results, they have discussed how to incorporate the existing ECUs prototype into electric cars. (Aurelian et.al ,2012) considers a set of homogeneous cores and also addresses the numerous entry of sequencing software module problem. To overcome this problem they have provided two solutions such as partitioning by using low complexity heuristic techniques and executing runnable tasks on each core by building the sequencer of tasks. A real time energy management approach was proposed by (Bindhu et.al, 2017). They have proposed a high power convertor to high voltage and continuous conduction. (K. MERINI et al 2016) proposed an optimal location finding algorithms for SATCOM. They are Particle Swarm Optimization (PSO) and Harmony Search (HS).

This work organized as follows; chapter 2 presents the system model, which describes the automotive input realtime task model, power model and electronic control unit based on multicore architecture. The chapter 3 discusses the three types of multiobjective evolutionary algorithm which is used to schedule the realtime task. The evaluation results are discussed in chapter 4. The Chapter 5 discussed about conclusion and future scope.

2.SYSTEM MODEL

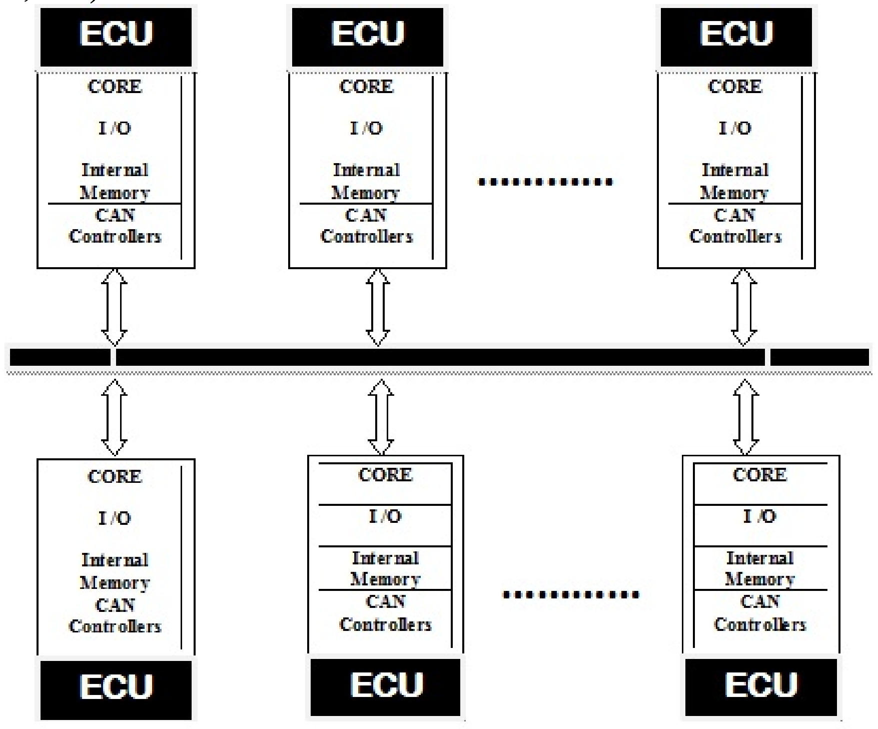

Figure 1. Electronic System Model used in automotive applications

The (Figure 1) shows the overall electronic system model used in automotive appliances. The EUCs are designed based on multicore architecture (ALU operations and Program execution), CAN controllers (to pass message between ECUs), internal memory and input/ output devices (designed with sensors and actuators) [8, 9, 11]. The (Figure 2) shows the multicore architecture based ECU in which real time periodic tasks $\tau_1, \tau_2, \tau_3 \dots \dots \dots \dots \tau_k$ are given as the input to the system. The tasks are arriving with an arrival rate (λ = 0.0 to 1.0) to the performance aware scheduler.

The performance aware scheduler will optimizes the performance of the system by considering power consumption, core utilization and deadline missed tasks. The Earliest deadline first (EDF) algorithm performs the initial schedule, in which tasks having earliest deadline are scheduled first. Then any one of the proposed optimization algorithm will be implemented to allocate the task to the core. The Dynamic voltage Frequency scaling algorithm determines the core voltage $(V_{i,core_j})$ and frequency (f_{new}). The communication between the core is established by interconnect network model. After execution of the task on core the core performs any one of the following operation:

1. Sends the output to the storage device
2. Sends the output to an optimization algorithm for further performance improvement
3. The task will exit to do input – output operation.

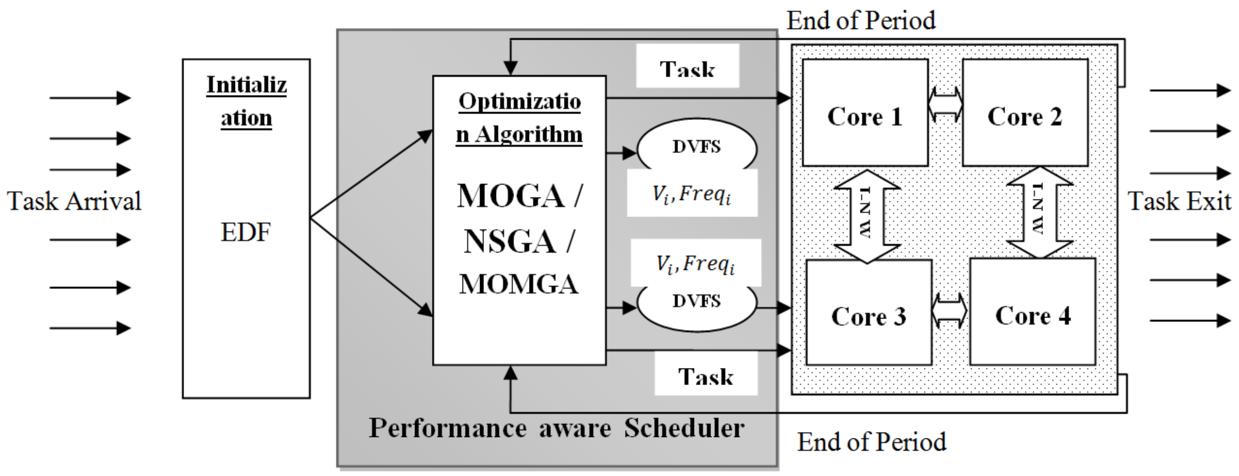

Figure 2. Multicore Architecture based Electronic Control Unit.

2.1 Automotive realtime task and power model

Several typical tasks are carried out by gateway ECU. At first the collection of data received from the buses. Here, the data's are detected using sensors and it is sent by using end node ECU. Second, it inspects the status of each end ECU on the bus and makes the desired reaction based on the dynamic changes. Third, it exchanges the data between two bus systems or more than two systems which is connected to the gateway ECU and also it performs message exchange. At fourth, it sends messages to the end ECU based on the behaviour of in-car components if necessary.

The important automobile functional domain is power train, which controls engine, transmission and gear system. The primary task of engine control is to observe the amount of fuel and exact injection moment. These tasks are depends on drivers pedal, temperature and engine load. The crank shaft and vale position are checked by using various sensors and actuator. To achieve maximum goal in engine control there are upto 100 executing tasks needed to be operate in closed synchronization. Thus, efficient smooth running engine is possible with minimum output of pollution.

To provide optimum solution in chassis, Antilock Braking System (ABS), Electronic Stability Program (ESP), Automatic Stability Control (ASC), and Adaptive Cruise Control (ACC) are composed with the following electronic system namely, 1.Sensor (detects that the wheel will lock), 2.Actuator (release and repeat the pressure on the discs) and 3.Controller (requires an ECU) [2, 5, 6, 7]. To provide body comfort the components such as, 1.air conditioning, 2.climate control devices, 3.dash board, 4.wipers, 5.cruise control and 6.park distance control are used. The telematics/wireless technology is provided by using the following components. 1. Multimedia, 2.infotainment, 3.GPS and in-vehicle navigation systems, 4.CD/DVD players and 5.rear-seat entertainment. The below table provides the properties of different ECUs used in an automotive electronic system. The table 1 shows the detailed properties of automotive functional domains.

Table 1. Properties of automotive functional domains.

	Powertrain	**Chassis**	**Body**	**Telematics**	**Passive safety**
Program size	2 MegaBytes	4.5 MegaBytes	2.5 MegaBytes	100 MegaBytes	1.5 MegaBytes
Number of ECUs	3 to 6	6 to 10	14 to 30	4 to 12	11 to 12
Number of messages	36	180	300	660	20
Bus topology	Bus	Bus	Bus	Ring	star
Bandwidth	500 Kb/secs	500 Kb/secs	100 Kb/secs	22 Mb/ secs	10 Mb/ secs
Time Cycle	10 ms to 10 s	10 ms to 10 s	50 ms to 2 s	20 ms to 5 s	~50 ms
Safety requirements	High	High	Low	Low	Very high

In periodic real time task the events occur at a constant rate and it depends on execution time, release time and deadline. All periodic tasks $\{\tau_i: i = 1, . . . , n\}$ have hard deadlines and their schedulability must be guaranteed. Each input task has the following properties,

$$\tau_i = \{R_i, e_i, d_i\} \quad (1)$$

Where,

R_i → Release time, which can be derived from the Poisson distribution with an arrival rate λ.
e_i → Worst case execution time is randomly chosen [16].

The workload of each task is defined as,

$$WL_i = \frac{e_i}{d_i - R_i} \quad (2)$$

The value of WL_i is generated by the Gaussian distribution. The task workload is randomly generated with the range (0, 1) with a mean of ρ and a standard deviation of ρ × 0.1using Gaussian distribution [5, 6, 7, 8].

Power consumption in Multicore architecture has two components such as leakage and dynamic power consumption [1, 4, 9, 15]. Leakage power essentially consists of the power used when the transistor is not in the process of switching and it is essentially determined by the formula,

$$P_{Leakage} = I_{Leakage} V_{DD} \quad (3)$$

The dynamic power consumption is occurring due to the non ideal characteristics of PMOS and NMOS networks. It is composed of load capacitances charging and discharging and short circuit current while both networks are partially ON.

It is formulated as,

$$E_{task} \quad (4)$$

$$= \sum_{k_i \in K} \sum_{Core_j \in Core} \sum_{i=1}^{M} \sum_{S_i=R_i}^{D_i} \sum_{f_i=S_i}^{D_i} \alpha C v_i^2$$

$$* \ freq_j * [(f_i - S_i). X_{K_i, core_j, v_i, s_i, f_i}]$$

The time interval $[S_i, f_i]$ can be estimated by,

$$(f_i - S_i) = s * e_i \quad (5)$$

Where, $e_i$is the worst case execution time of task τ_i, and s is the ratio of the extended execution time when performing dynamic voltage scaling. $X_{K_i, core_j, v_i, s_i, f_i}$ is a variable and is equal to 1, if the task τ_i is scheduled to core $core_j$ between starting time (S_i) and finishing time (f_i). Otherwise, this variable is equal to 0. Transaction power consumption is also to be considered in Multicore processor. It is defined as,

$$G_{tran} = \sum_{i,j} (1 - \beta . C) \left| v_j^2 - v_i^2 \right| \quad (6)$$

So, the total power consumption is,

$$P_{Total} = P_{Leakage} + E_{Task} + G_{Tran} \quad (7)$$

2.2 Multicore architecture

This architecture has more than two cores and each core has identical performance. In this architecture network, flexible, series and parallel IO can be used. Each core consists of process engines (PE), data memory, floating point multiple access (FPMAC) and router. Interconnecting bus (NIC) is used to share the resources between cores [15, 16, 17, 18] (Fig 2). The process engine consists of a single cycle Instruction Memory (IMEM) and data memory (DMEM). The capacity of total memory was enough to implement blocked the execution of selected kernels. To achieve high performance in realtime applications a highly efficient multiply and accumulate (MAC) unit is always required. For example a Floating point number can be given by the equation (8),

$$z = (-1^{\ a}) * 2^{(\exp - bias)} * (1 * M) \quad (8)$$

A n x n 2D mesh Multicore architecture contains n^2 routers. Each router has an address$\{x, y\}$, where x and y belongs to $\{0,1,2, \ldots \ldots \ldots, n-1\}$ a n x n mesh.

3. MULTIOBJECTIVE EVOLUTIONARY ALGORITHM

Exact and approximation techniques will provide better optimization results. For a particular groups, choosing the best solutions for multiobjective optimization problems are preferred by the subjective information preference, which can be provided by the Decision Maker (DM) [5,10]. This decision maker is mathematically equivalent to Pareto optimal solutions. The multiobjective optimization problem can be solved by the following two main approaches, they are:

i) Multi- Criteria Decision Making approach (MCDM)
ii) Evolutionary Multiobjective optimization approach (EMO).

The preference of choosing Multi-Objective Evolutionary Algorithms (MOEAs) is addressed by considerable researchers in recent years. The following equation mathematically defines the multiobjective optimization problem (MOP):

$$min \{f(x)\} = \{f_1(x),\ f_2(x), f_3(x) \ldots\ldots\ldots\ldots.. f_n(x)\} \quad (9)$$

subject to **x ε χ**

The ‘n’ decision making variables forms a vector (X ε R^n) and it is chosen for the optimization problem. The vector function *f :* $R^n \rightarrow R^k$, R^n - decision variable space and R^k objective function space. And the vector function *f* is composed by f_i: $R^n \rightarrow R$ (i = 1, … , k; k ≤ 2). The feasible set(χ) is implicitly found by equality and inequality constraints. The image of χ under the function *f* is a subset of the objective function space denoted by Z = *f (χ)* and referred to as the feasible set in the objective function space.

The following objectives are addressed to achieve better performance in Multicore processor,

i. Minimize the power consumption *{min (Power)}*
ii. Minimum voltage transaction delay, *{min (Tran_time)}*
iii. Maximum utilization of core, *{Max (Ucore)}*
iv. Shut off unused cores
v. Parallel task scheduler. *{min (Task_Sched_Time)}*

3.1 Multi-Objective Genetic Algorithm (MOGA)

The Multi-Objective Genetic Algorithm (MOGA) solves $\boldsymbol{f_k(x)}$ objective function by evolving ‘g’ generations for N-members. The procedure of MOGA is represented in algorithm1 in which the population P was initialized based on EDF approach. In this algorithm, the geno functions were computed in a normal procedure but the rank is assigned based on the pareto dominance. The individual rank is assigned by the following rule: *rank*($\boldsymbol{x_i}$, *t*) = 1+*p*(*t*), where *p*(*t*) is the dominance in the current population [12, 20].

Algorithm1: Multi-Objective Genetic Algorithm

```
Procedure:- MOGA(N, g, $f_k(x)$) : N-members evolved g generations to solve $f_k(x)$
        Initialize the Population (P)
        Rank Assignment - Pareto Dominance
        Assign Fitness (Linearly Scaled)
        Share Fitness
        for i=1 to g do
            Selection : Stochastic Universal Sample Technique
            Single Point Crossover
            Mutation
            Evaluate: Total utilization of Core value ($U_{core} < 1$)
            Assign: Rank (Pareto Dominance), Fitness, Shared Fitness
        end for
end procedure
```

3.2 Non-dominated Sorting Genetic Algorithm (NSGA)

The NSGA algorithm is used to solve the $\boldsymbol{f_k(x)}$ objective functions in which the population’s rank is compute on the basis of non – dominance. Then, the non-dominated individuals are grouped into one catageroy with the following characteristics; 1. Dummy fitness value, 2. Which one is propotinate to the population size, and 3.to provides an equal reproductive potential for such individual. These groups are shared with the dummy fitness value to maintain the population diversity. After sharing, the current group will be terminated and a new group of non-dominated individuals considered. To perform this technique a stochastic remainder proportionate selection is adopted. The first front individual will get more copies than the remaining set of population, because it is having maximum fitness value. So that, a better search of pareto-front is allowable for the known regions and also the population is converged towards such region. As a result, one might think that this MOEA converges rather quickly; however, a computational bottleneck occurs with the fitness sharing mechanism [13,21].

Algorithm 2: Non-dominated Sorting Genetic Algorithm

```
Procedure:- NSGA-I(N, g, $f_k(x)$)
        Initialize Population (P)
        Generate random population - size (N)
        Assign Rank : dominance - sort
        Generate : Child Population
        Selection : Binary Tournament
        Recombination and Mutation
        for i = 1 to g do
        for Population : Parent and Child do
        Assign Rank : Pareto - sort
        Generate : non-dominated vectors
        end for
        Select points: lower pareto-front
        Create next generation
        Binary Tournament Selection
        Recombination and Mutation
        end for
end procedure
```

3.3 Multiobjective Messy Genetic Algorithm (MOMGA)

MOMGA consists of three phases: (1) Initialization Phase, (2) Primordial Phase, and (3) Juxtapositional Phase. In the first phase, a partially enumerative initialization process is specified to produce MOMGA blocks. And also the population of each members fitness value is evaluated with respect to 'k' templates. In the second phase, the population size will be reduced by performing tournament selection technique. In the last phase, the cut and splice recombination operator is used to build up the population for the MOMGA approach [20].

Algorithm 3: Multiobjective Messy Genetic Algorithm

```
procedure MOMGA($N$, $g$, $\boldsymbol{f_k}(\boldsymbol{x})$)
        for $i$ = 1 to epoch do
          Initialization:
          Partially Enumerative Initialization
          Evaluate (based on templeates)
          Primordial Phase:
          for $i$ = 1 to Max Primordial Generations do
          Tournament Thresholding Selection
          if generations accomplished then
          Reduce Population Size
          end if
          end for
          Juxtapositional Phase:
          for $i$ = 1 to Max Juxtapositional Generations do
          Cut-and-Slice
          Evaluate fitness (templates)
          Tournament Thresholding Selection
          and Fitness Sharing
            $P_{known(t)} = P_{current(t)} \cup P_{known(t-1)}$
          end for
          Update $k$ templates - Using best known value in each objective
        end for
end procedure
```

4.EVALUATION

For evaluation 1000 random tasks are generated based on equation 1 and 2 with different arrival rate ranges from 0.1 – 0.9λ. To obtain an average result the simulation runs about 100,000 times and tasks which cannot be scheduled are skipped. The following Intel ATOM –C - processor voltage, frequency and power consumption is considered to investigate the parameters of Multicore architecture [19, 20, 21] (Table 2).

Table 2. Frequency, Voltage and Power consumption of Intel ATOM - C- Processor

Parameters	***Dynamic Level's***				
	1	*2*	*3*	*4*	*5*
Frequency (f_k) in *MHz*	150	400	600	800	1000
Voltage (V)	0.75	1.0	1.3	1.6	1.8
Power Consumption (P_k) in *mW*	80	170	400	900	1600

Here, we have shown the execution time of forty sample tasks and the scheduling results were computed based on the formulas and algorithms discussed in chapter 2 and chapter 3 are depicted in (Table 3.a, Table 3.b and Table 3.c).

e_i = {0.018, 0.174, 0.95, 1.097, 0.317, 0.8, 0.836, 0.376, 0.56, 0.83, 0.116, 0.128, 0.072, 0.88,0.33, 0.81, 1.12, 1.4, 0.88,0.5, 0.133, 0.185, 0.883, 0.307, 0.241, 0.51, 0.59, 0.614,0.552,1.05,0.13,0.101, 0.331, 0.152, 0.646, 0.92, 0.939, 0.398, 0.957, 0.825}

Table 3. Scheduling Result of different scheduling algorithms a) MOGA b) NSGA c) MOMGA algorithm

ECU - Core (Avg. Power Consumed)	Tasks [τ(1) - τ(40)]									
C1 (0.872)	34	23	17	15	9	8	36	34	33	7
C2(0.957)	27	18	16	22	10	25	38	30	4	31
C3(0.821)	36	19	6	14	2	13	37	29	32	3
C4(0.712)	12	20	11	1	21	28	35	39	40	5

(a)

ECU - Core (Avg. Power Consumed)	Tasks [τ(1) - τ(40)]									
C1 (0.6772)	10	27	16	8	35	22	19	38	36	4
C2(0.833)	14	6	1	30	15	40	18	32	21	13
C3(0.9211)	5	23	11	34	37	9	2	24	39	33
C4(0.8112)	20	7	25	17	3	31	26	19	12	28

(b)

ECU - Core (Avg. Power Consumed)	Tasks [τ(1) - τ(40)]									
C1 (0.9172)	18	8	11	3	31	17	36	28	13	20
C2(0.8257)	4	2	23	15	35	6	21	39	34	40
C3(0.6621)	24	16	26	33	22	37	12	1	38	29
C4(0.6933)	9	30	5	32	10	27	19	14	25	7

(c)

4.1 Power Consumption

The power consumption (P) is computed for different arrival rate (λ) is computed by using equation 6 is shown in (Fig 3) for 1000 random task configurations. The power consumption of EDF algorithm is taken as reference value and we consider maximum power consumption. When the arrival rate (λ) is less than 0.5 the MOGA and NSGA algorithms consumes almost the same power. For large arrival rate ($\lambda > 0.6$) the Non – dominated Sorting Genetic algorithms (NSGA) consumes more power. With the help of MOMGA we have reduced the voltage transaction power, but for higher arrival rate ($\lambda > 0.8$) the execution power is large. When $\lambda = 0.6$ the power consumed by MOMGA is minimized by 20% as the tasks are executed on all available cores.

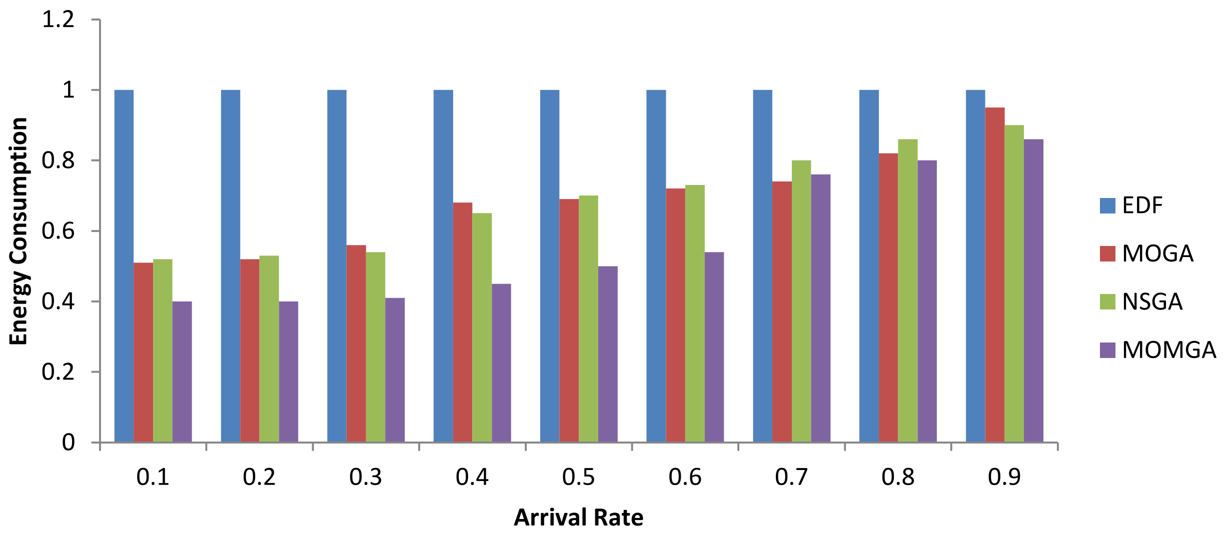

Figure 3. Power consumption for different scheduling techniques by considering voltage transaction power (G) and task execution power (E). Statistics collected for 1000 random configurations

4.2 Deadline Missrate (δ)

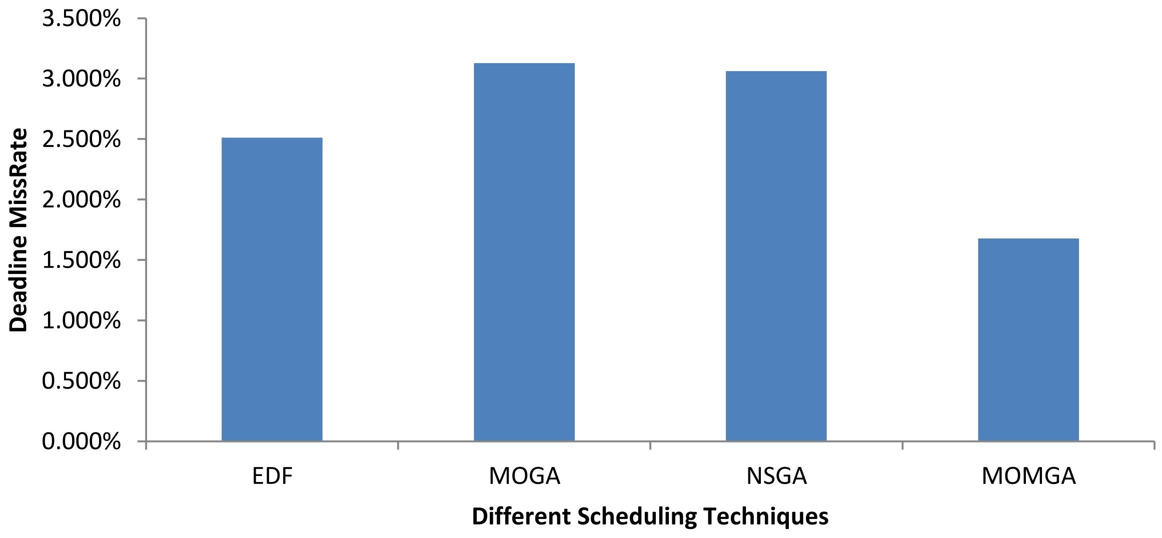

Figure 4. Deadline missrate for different scheduling techniques. Statistics collected for 1000 random configurations

The dead line missrate is also an important performance parameters used to find how the scheduler serves the task to the core within deadline. It can be computed by finding the ratio between the number of tasks that misses deadline to the input task given to the system. The deadline missrate for different scheduling scheme is depicted in (Fig 4).

$$\delta = \frac{\sum \tau_{deadline_{miss}}}{\sum_{i=1}^{n} \tau_i} \quad (10)$$

4.3 Core Utilization

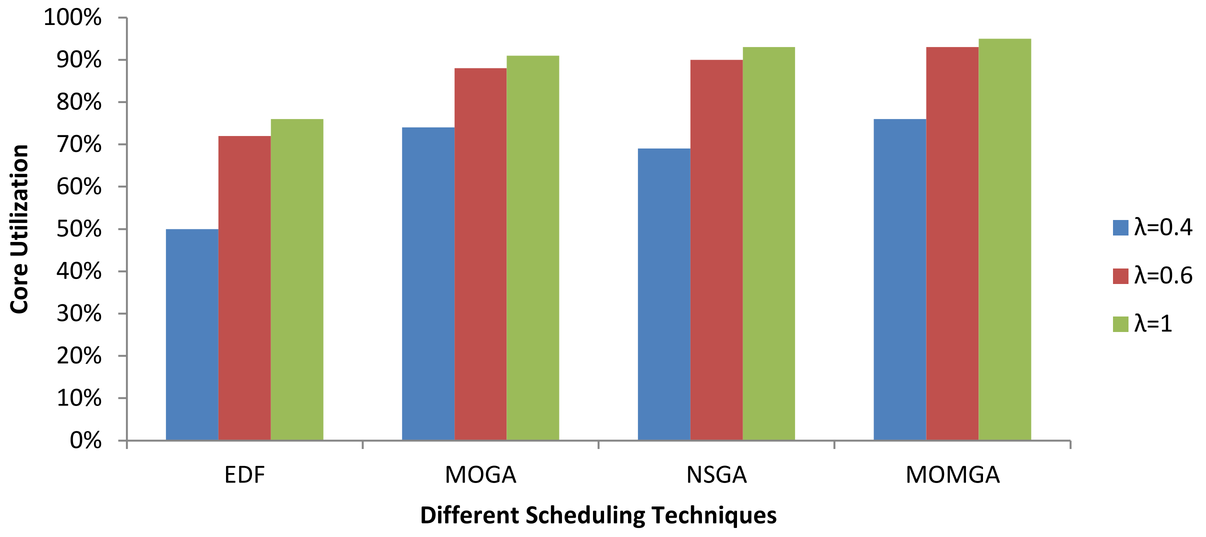

Figure 5. Core Utilization for different scheduling techniques. Statistics collected for 1000 random configurations

Utilization of core is a proportion u_i of the total number of cycles of a core will be dedicated to executing τ_n is formulated by,

$$u_i = \frac{W_n}{P_n} \quad (11)$$

Then, Total Utilization of task is,

$$U = \sum_{\tau_n \xi T} u_i \quad (12)$$

In multicore architecture more utilization provides significant improvement in performance. The task is schedulable when the utilization (U) should be less than one. The maximum utilization can be achieved through a dynamic task repartitioning mechanism. EDF approach assigns the task to the cores in static fashion and the proposed MOEA approaches assign it dynamically. If maximum utilization achieved the leakage power consumption can be minimized by implementing shut off techniques. The (Fig 5) shows the utilization different scheduling algorithm with arrival rate λ = 0.4 and λ = 0.8 and also we can identify that EDF approach utilized 40% - 50% of core but MOEA approaches utilize 60% -95% of core.

5.CONCLUSION AND FUTURE SCOPE

The performance of a ECU is an important factor for automotive electronic appliances. This can be achieved significantly by maximizing the utilization, minimizing deadline missrate and optimizing power consumption. In this work, we have proposed three task scheduling approaches to optimize the performance at runtime. They are, Multiobjective

Genetic Algorithm (MOGA), Non – dominated Sorting Genetic Algorithm (NSGA) and Multiobjective Messy Genetic Algorithm (MOMGA). The proposed approach assigns the tasks at runtime, so that the utilization of core increased when compared to EDF approach is from 10 % to 40 %. The power consumption is reduced to about 5 % - 8% and deadline missrate is comparatively minimized with the earlier studies. In future further, we can optimize the performance by introducing multi objective parallel task scheduling mechanism.

REFERENCES

[1] M.Krügera.Straubea.Middendorfb.HahnbT.Dobsb and Langa , "Requirements for the application of ECUs in e-mobility originally qualified for gasoline cars" , J.Microelectronics Reliability, (2016) ,Volume No. 64, Pages 140-144.

[2] Jaeyong Rho, TakuyaAzumic, MayoNakagaw, KenyaSatod and Nobuhik Nishi, "Scheduling parallel and distributed processing for automotive data stream management system", Journal of Parallel and Distributed Computing (2017) Volume 109, , Pages 286-300.

[3] XieYong, ZengGang, RyoKurachi, XieGuoqi, DouYong and ZhouZhili, "An optimized design of CAN FD for automotive cyber-physical systems" , Journal of Systems Architecture, (2017) Volume 81, Pages 101-111.

[4] Aurelian, Nicolas Navet, Bernard Bavoux and Françoise Simonot-Lion, "Multisource Software on Multicore Automotive ECUs—Combining Runnable Sequencing With Task Scheduling", IEEE Transactions on Industrial Electronics, (2012) Volume: 59, Issue: 10, Pages: 3934 – 3942.

[5] Simon Schliecker, Mircea Negrean and Rolf Ernst, "Response Time Analysis on Multicore ECUs With Shared Resources", IEEE Transactions on Industrial Informatics, , (2009) Volume: 5, Issue: 4 Pages: 402 - 413

[6] S. Abinesh, M. Kathiresh and R. Neelavenik, "Analysis of multi-core architecture for automotive applications", International Conference on Embedded Systems (ICES), (2014) Pages: 76 – 79

[7] Dr. Preeti Bajaj and Dinesh Padole, "Design of an Embedded Controller for Some Applications of an Automotives" Publisher: InTech, Chapters, Edited by Marcello Chiaberge, , (2011) ISBN 978-953-307-517-4, 678 pages.

[8] Abusayeed Saifullah, Kunal Agrawal, Chenyang Lu & Christopher Gill, "Multi-core Real-Time Scheduling for Generalized Parallel Task Models", All Computer Science and Engineering Research, Report no.WUCSE-2011-45, (2011) (2011) Available from: http://openscholarship.wustl.edu/cse_research/58 .

[9] Anant Balakrishnan & Azad Naeemi (2011). "Interconnect Network Analysis of Many-Core Chips", IEEE Transactions on Electron Devices, (2011) vol. 58, no. 9, pp. 2831-2837.

[10] Antonio J. Nebro, Juan J. Durillo, Francisco Luna, Bernab´e Dorronsoro, & Enrique Alba "A Cellular Genetic Algorithm for Multiobjective Optimization", International Journal of Intelligent Systems, (2009) vol.24, no.7, pp. 726-746

[11] Buttazzo & Giorgio "Hard Real-Time Computing Systems", Real-Time Systems Series, Available from: Springer books (2011).

[12] C.A. Barros, L.F.Q. Silveira, C.A. Valderrama & S. Xavier-de-Souza, "Optimal processor dynamic-energy reduction for parallel workloads on heterogeneous multi-core architectures" Journal of Microprocessors and Microsystems, (2015) vol.39, pp.418–425.

[13] C.H. Papadimitriou & K. Steiglitz "Combinatorial Optimization: Algorithms and Complexity", Dover Publications, (1998) Inc., pp. 307–312.

[14] Carlos M. Fonseca & Peter J. Fleming "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization" in Genetic Algorithms: Proceedings of the Fifth International Conference, ed.S. Forrest, San Mateo, CA: Morgan Kaufmann, (1993) pp 12-20.

[15] Da He & Wolfgang Mueller "Heuristic energy-aware approach for hard real-time systems on multi-core platforms", Journal of Microprocessors and Microsystems, (2013) vol. 37, no. 8 Part A, pp. 858-870.

[16] David Rhodes & Robert Dick, "Task Graphs for Free (TGFF v3.0)", (2008). Available from: http://ziyang.eecs.umich.edu/~dickrp/tgff/.

[17] David Wentzlaff "On-Chip Interconnection Architecture of the Tile Processor" IEEE Computer Society, (2007) vol.27,no.5, pp.15-3

[18] E. Seo, J. Jeong, S. Park & J. Lee "Energy efficient scheduling of real-time tasks on Multicore processors" IEEE Trans. Parall. Distrib. Syst. (2008) vol.19, no.11 pp.1540–1552.

[19] Geoffrey Blake, Ronald G. Dreslinski, & Trevor "A Survey of Multicore Processor" IEEE Signal Processing Magazine, (2009) pp.26 -37

[20] Kalyanmoy Deb "Multi-Objective Optimization Using Evolutionary Algorithms: An Introduction" (2011) Indian Institute of Technology

[21] Kanpur, KanGAL Report no. 2011003,Available from: http://www.iitk.ac.in/kangal/deb.html

[22] Thomas Weise (eds) "Global Optimization Algorithms Theory and Application" (2008). Available from: http://www.it-weise.de/

[23] A.Bindu , M.Carolin Mabel, and C.Bharatiraja "A Real-Time Energy Management Approach And Its Power Converter For PV Powered Plug-In Electric Vehicles" J.Electrical Engineering, 2017, volume 17.no 1.8 pp. 52-61.

[24] Kamel Merini & Abdelkader Rami, "Optimal Allocation and Voltage stability in Electical network by using a STATCOM" J.Electrical Engineering, 2016, 16.4.1, pp.1-9