Algorithm To Optimize Code Size And Energy Consumption In Real Time Embedded System

Santosh Chede and Kishore Kulat
Department of Electronics and Computer Science Engineering, Visvesvaraya National Institute of Technology, Nagpur, India
Email:santoshchede@rediffmail.com

Abstract— **Processor is an important computing element in portable battery operated real time embedded system and it consumes most of the battery energy. Energy consumption, processor memory space are considered as basic design constraints in ARM based system and heuristic algorithm is developed for energy consumption as well as memory space management. This Algorithm includes rate monotonic fixed priority task scheduling scheme, DVS, ARM's Normal /Thumb mode, execution time, and number of execution cycles, which are analytically related with energy consumption. Mathematical modeling and simulation of Heuristic algorithm is done using MATLAB. This gives optimized code size, execution time, energy consumption of each task / whole system and proves a novel strategy in the field of software related energy optimization and real time embedded system design.**

Index Terms— **Dynamic Voltage Scaling (DVS), code size, energy consumption, ARM processor, embedded system, task scheduling.**

I. INTRODUCTION

Design engineers must be aware of various critical issues while designing the complex high performance low power real time embedded system. As most of the real time embedded systems such as PDAs, and other handheld dedicated appliances are battery operated, energy savings is a great challenge. Processor is a main computing element in such systems. It consumes most of the battery energy and improper code is responsible for that. [2]. Processor selection depends on the type of application and processor has always tight constraints like memory space, energy consumption and instruction /program execution time i.e. desired performance. Code size is responsible for memory space management. Voltage levels and operating frequency sets processor energy consumption. Processor performance is related with real time constraints like critical and non critical tasks execution. Embedded processors are built with CMOS architecture. DVS is power reduction technique which can be used to set voltage levels and operating frequency depending on the real time application, with desired performance. DVS is basically used to carry out compromise between voltage and frequency with reference to power consumption and execution time. By varying processor speed to meet task deadlines, energy saving is achieved. This concept has been employed on commercial variable voltage processors such as ARM, Intel's Xscale, AMD's K6-2+ and Transmeta's Crusoe processors.

To use processor memory space efficiently, code size optimization is important. Reduced code size can be obtained in ARM processor at the expense of a significant increase in the number of instructions executed by the program. ARM processor has Normal 32-bit and 16-bit Thumb mode. For ARM processor in Normal mode, code size is more while execution time and energy consumption is less. Exactly reverse situation is realized when ARM processor operates in Thumb mode. Mixed ARM and Thumb code simultaneously gives compact code size, low energy and good performance with respect to program execution. In order to run critical task Normal mode is used, otherwise Thumb mode is used [5]. Normal 32-bit mode and Thumb 16-bit mode access all the 16 and 8 general purpose registers respectively. Hence, execution time for the code written in Normal mode is lesser than that with Thumb mode [3, 4] i.e. code size is inversely proportional to the number of execution cycles, execution time.

Real time application deals with critical and non critical task execution. For critical and non critical tasks Normal and Thumb modes are used respectively. Task priority decides task execution sequence i.e. critical or non critical. For example, considering Rate Monotonic (RM) fixed priority task scheduling algorithm, task with less period executes first and to execute it faster Normal mode is used. Embedded application may consist of many numbers of tasks, subtasks. Hence task priority scheduling is a primary factor in deciding an execution time and code size.

Real time embedded system uses hardware/software codesign strategy. Hardware as well as software consumes energy. Such system consists of number of tasks and each task can be written in number of ways with different code sizes. Execution time and energy consumption for each task can be estimated on the basis

This paper is based on "Hardware/software codesign techniques in low power embedded system," by S. D. Chede and K. D. Kulat, which is appeared in proceedings of International MultiConference of Engineers and Computer Scientists IMECS, 2007, held at Hong Kong, during 21-23 March, 2007.

of real time constraints. In order to design energy efficient real time system, code size, execution time and energy consumption optimization has major importance. Three issues like code size, energy consumption and execution time, are discussed to minimize code size for energy constrained real time embedded systems in [3]. But there is no previous work or algorithm which will address optimization problem in detail.

In this paper, heuristic algorithm is developed for ARM processor based systems. It includes rate monotonic fixed priority task scheduling scheme, DVS, Normal /Thumb mode, execution time, and number of execution cycles, which are analytically related with energy consumption. Mathematical modeling of this algorithm is realized using Matlab. This algorithm addresses novel and simple strategy for code size based energy consumption optimization in real time embedded system. Optimized code size, execution time and energy consumption are the final outcome of algorithm.

II. Related Parameters

Dynamic voltage scaling, code size and task scheduling are three basic parameters to constitute heuristic algorithm. Task scheduling priority is assigned according to RM algorithm. Minimization of the energy consumption issues are based on two distinct approaches i.e. task schedulability and voltage scaling [1].

A. Dynamic Voltage Scaling

This technique is implemented for energy management of embedded real time systems. In this, devices dynamically change their speed increasing the energy operation efficiency. The reduction of energy consumption in systems can be achieved without affecting the performance. DVS algorithms are able to make energy savings while providing the necessary peak computation power in general purpose systems [6,7,8,9,10,12,13,15,16,18].
For CMOS technology,

$$P_D \ \alpha \ C_L \times V_{DD}^2 f \tag{1}$$

$$T_D \ \alpha \frac{C_L \times V_{DD}}{(V_{DD} - V_T)^{\beta}} \tag{2}$$

$$f = \frac{1}{T_D} \tag{3}$$

$$E = P_D \times t \tag{4}$$

where,

P_D -Dynamic power,
C_L -Load capacitance,
f -CPU clock frequency,
V_{DD} -supply voltage,
V_T -threshold voltage,
T_D -delay,
t -task execution time
β- Technology dependent constant (varies between 1&2).
E- Energy consumption.

Lowering the supply voltage $V_{DD,}$ will also decrease the clock speed f. Relation between dynamic power P_D and V_{DD} is quadratic in nature. DVS which involves dynamical adjustment of supply voltage and the CPU clock speed is a basic strategy to reduce the energy consumption for the embedded system.

B. Code size (bytes)

Specific real time embedded application needs dedicated code. As embedded processor has limited memory space, it is essential to optimize code size for better utilization of memory. Code density and energy consumption are critical efficiency metrics for embedded processors. Code size determines the amount and cost of on chip or off chip memory necessary for program storage. Normal and Thumb modes are used to get compact code size. A reduction in code size is encoded from subset of the 32-bit normal mode into 16-bit Thumb mode. Reduction in code size is possible with minimal hardware additions also. As Thumb mode can access only 8 general-purpose registers out of 16 general purpose registers, number of execution cycles is inversely proportional to code size [3].

C. Rate Monotonic Fixed Priority Scheduling Algorithm

The rate-monotonic (RM) scheduling is one of the most widely studied and used in practice [11, 14, 17]. It is uniprocessor static priority preemptive scheme. The priority of the task is inversely related to its period. In this paper, tasks priority is set with respect to RM and energy consumption for the task is estimated. For example, if system consists of number of tasks (T_1, T_2, T_3) with different periods P_1=3, P_2=4, P_3=5 and execution times in µsec. are t_1=0.5,t_2=2.0,t_3=1.73. Since $P_1 < P_2 < P_3$, task T_1 has higher priority and task T_3 cannot execute when either task T_1 or T_2 is unfinished. Let system energy consumption is equal to 1000 nJ and execution time is equal to 0.5+2.0+1.73 µsec; DVS is implemented to dedicate highest clock frequency and maximum energy consumption to complete T_1 than that for T_2, T_3. Frequency is set to get execution time in µsec. of 0.5, 2.0, 1.73, with energy consumption ≤ 1000nJ.

III. Problem Description And Mathematical Analysis

A. Problem Description

Real time embedded application consists of number of tasks and denoted by

$$Tsys = \{T_1, T_2, T_3, ..., T_n\} \tag{5}$$

Assuming each task has various time intervals, generally known as Periods P_1, P_2-------P_{n-1} respectively. Here P_1 is the time interval between the two consecutive tasks T_1 & T_2. Similarly P_2----P_{n-1} can be defined as represented in figure 1. The scheduling theory deals with schedulability of concurrent tasks with deadlines and priorities.

Priority-3	Priority-1	Priority-2	Priority-4	---- Priority-n
T_1	T_2	T_3	T_4	---- T_n
Period $P_{1=100}$	Period $P_{2=300}$	Period $P_{3=250}$	Period $P_{4=80}$	---- Period P_{n-1}

Figure 1. Real time task scheduling in priority sequence

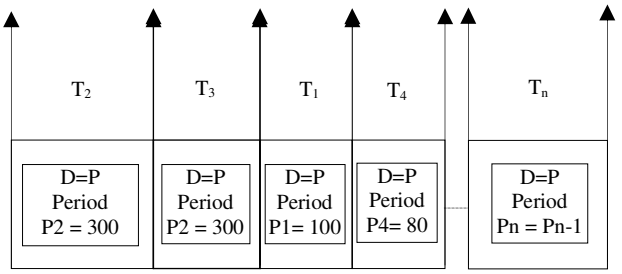

Figure 2. Illustration of task execution with D=P

Different tasks though they run concurrently are differentiated based on their priorities. A real time task has the following parameters

- Period P
- Deadline D
- Number of execution cycles: n_i
- CPU clock frequency f
- Execution time for i^{th} task t_i
- Energy consumption E

All the tasks, subtasks in the system must be scheduled to meet deadlines. Figure 2 depicts the process of task execution with D=P. If periodic task takes more time to complete than its own period, it cannot be scheduled. If this is the case then the processor frequency can be increased to reduce t_i as

$$ti = \frac{ni}{f} \tag{6}$$

Therefore, frequency (f) should be increased such that $t_i \le P_i$, which results in increase of power consumption. From "(1)," it is clear that P_D is proportional to CPU clock frequency and square of V_{DD}. Figure 3 shows the variation of clock frequency F and V_{DD} for various values of β. Pi denotes period for i_{th} task. Code size optimization is implemented using ARM Normal and Thumb mode. Decrease in code size, increases number of execution cycles and execution time. Task may consist of 32 and 16 bit instructions to manage the energy consumption, execution time and code size. For critical deadline task, instructions could be written into 32-bit format instead of 16-bit. This reduces execution time with overhead of higher clock frequency and power dissipation as given in "(1)." But for non-critical deadline task, 32 bit instructions can split into 16 bits. This needs less clock frequency and power dissipation. In the same manner each task may consists of various 32 and 16 bit instructions to manage the code size, execution time and energy consumption. Considering instruction code size within a task and code size of various tasks within the system, program can be optimized with DVS approach i.e. code size optimization needs various frequency levels, and frequency is varied by changing V_{DD}, which will be responsible for optimized power dissipation, energy consumption.

B. Problem Formulation

The problem of finding minimum system code size satisfying the system's real time and energy constraints is crucial. Here heuristic algorithm is used to solve this problem. Here it is assumed that each real time task has a trade off relationship between code size and execution time. System may consist of number of tasks with various code sizes.

Let us define the system code size $S_{sys,}$ as the sum of code sizes S_i of the task in the system.

$$S_{sys} = \sum_{T_i \in T_{sys}} S_i \tag{7}$$

System energy consumption $E_{sys,}$ is the sum of energy consumptions of the entire task in the system during operations.

$$E_{sys} = \sum_{T_i \in T_{sys}} \frac{P_{LCM}}{P_i} e_i \tag{8}$$

where, P_{LCM} is least count multiple of period.

An inverse relationship should be considered for code size and number of execution cycles.

$$n_i = ST_i(f) \tag{9}$$

where,

ST_i is the step function.

Again assume that, real time embedded system has a trade off relationship between voltage and CPU clock speed from figure 4.

$$V = FV(f) \tag{10}$$

For given n_i & f_i for task T_i, execution time can be obtained as

$$t_i = \frac{n_i}{f_i} \tag{11}$$

And energy consumption can be obtained from "(1)" as

$$e_i = t_i V^2 f_i \tag{12}$$

From "(10)" and "(11),"

$$e_i = t_i [FV(f)]^2 f_i \tag{13}$$

$$e_i = n_i [FV(f)]^2 \tag{14}$$

Hence, code size optimization problem can be solved as

$$S_{sys} = \sum_{T_i \in T_{sys}} S_i \quad (15)$$

subject to

$$t_i \le P_i \quad (16)$$

$$\sum_{T_i \in T_{sys}} \frac{t_i}{P_i} \le 1 \quad (17)$$

$$E_{sys} \le E_{sys}^{limit} \quad (18)$$

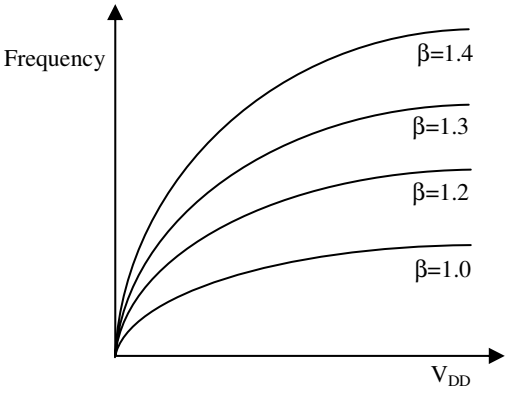

Figure 3. Variation of V_{DD} and frequency F

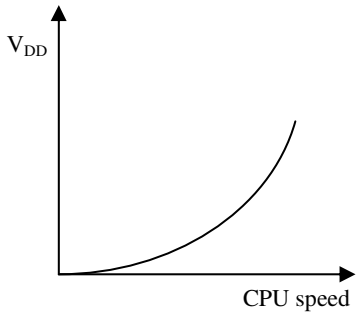

Figure 4. Trade off relation of CPU Speed and V_{DD}

IV. HEURISTIC ALGORITHM

Consider heuristic algorithm, that gradually reduce the system code size by increasing number of execution cycles of a task and adjusting the clock speed for the task as long as the systems real time and energy constraints are satisfied. The schematic procedure to obtain sub-optimal solution is explained below step by step.

Let, S_i - set of code sizes for i^{th} tasks.

$$S_i = \left\{ S_{i_1}, S_{i_2}, ..., S_{i_{m(i)}} \right\} \quad (19)$$

n_i - set of number of execution cycles for each code size per task

$$n_i = \left\{ n_{i_1}, n_{i_2}, ..., n_{i_{m(i)}} \right\} \quad (20)$$

$$f = \left\{ f_1, f_2, ..., f_j \right\} \quad (21)$$

where

f_1 is f_{min} and f_j is f_{max}.

f - set of CPU clock frequency ranging from f_{min} to f_{max}.

j - number of CPU clock frequencies.

Therefore,

$$V = \left\{ V_1, V_2, ..., V_K \right\} \quad (22)$$

where

$V_1=V_{min}$ and $V_K=V_{max}$,

V- Set of applied V_{DD} to obtain frequencies as well as energy consumption.

K - Number of voltage levels

The schematic procedure to obtain sub-optimal solution is explained below step by step.

Step 1: Obtain possible code size and number of execution cycles for each task.

$$S_i = \left\{ S_{i_1}, S_{i_2}, ..., S_{i_{m(i)}} \right\}$$

$$n_i = \left\{ n_{i_1}, n_{i_2}, ..., n_{i_{m(i)}} \right\}$$

Step 2: Obtain the Cartesian products of two sets n_i and f_i for each task. For i^{th} task the Cartesian product is developed as below.

$$\begin{aligned} &\langle n_{i_1}, f_1 \rangle, \langle n_{i_1}, f_2 \rangle, ---- \langle n_{i1}, f_j \rangle, --- \\ &\langle n_{i_2}, f_1 \rangle, \langle n_{i_2}, f_2 \rangle, ---- \langle n_{i_2}, f_j \rangle, --- \\ &\langle n_{i_{m(i)}}, f_1 \rangle, \langle n_{i_{m(i)}}, f_2 \rangle, ---- \langle n_{i_{m(i)}}, f_j \rangle, --- \end{aligned} \quad (23)$$

Step 3: Obtain energy consumptions for each combinations of <n_i, f> using

$$e_{i_p} = n_{i_p} V_p^2 \quad (24)$$

where,

i=1,2,….,η & p=1,2,….C

C= maximum value of voltages i.e V1, V2-----VC.

Therefore,

Number of e_i obtain are

$$C = {}^{m(i)}\eta_j \quad (25)$$

Step 4: Develop triple tradeoff tuples for each task

$$X_i \left\{ S_{i_r}, t_{i_p} = \frac{n_{i_p}}{f_p}, e_{i_p} \right\} \quad (26)$$

where, p =1,2,….,C & r =1,2,…., η

Step 5: Create triple trade off tuples X = {X_i}, using various combinations, such that X includes exactly one triple trade-off tuples X_i for each task Ti. Figure 5 shows the process of combinations in detail.

Step 6: Obtain the optimal solution as follows

$$\text{Minimum } S_{sys} = \sum_{T_i \in T_{sys}} S_i \quad (27)$$

subject to

$$t_i \le P_i \quad (28)$$

$$\sum_{T_i \in T_{sys}} \frac{t_i}{P_i} \le 1 \quad (29)$$

$$E_{sys} \le E_{sys}^{limit} \quad (30)$$

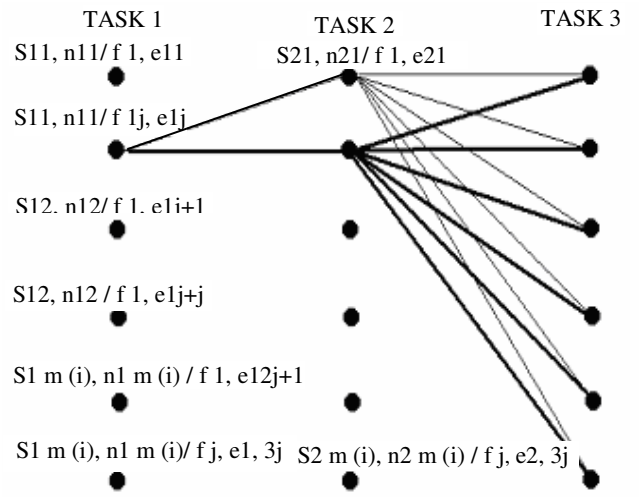

Figure 5. Schematic procedure to obtain combinations of triple trade off tuples

V. Result and Discussion

To realize Heuristic algorithm, three tasks having execution period of $P_{i\,=}$ 30, 40 and 50 µsec. respectively is considered. For simulation, V_{DDmin}= 2 volts, V_{DDmax}= 4 volts, V_T =1.2 Volts and β =1.2 is taken. System energy consumption limit of 1000 nJ is considered. Heuristic algorithm mentioned in step 1–step 6 is used to obtain the optimized code size of each task, satisfying real time requirements such as execution time,energy consumption.

TABLE I
SAMPLE PROGRAM SIMULATION USING IAR WORKBENCH

PROGRAM 1	PROGRAM 2	PROGRAM 3

LDREQ r0,[r1]
LDRNE r0,[r2]
ADDEQ r 0,r3,r0
ADDNE r0,r4,r0
	BNEL1
	LDR r0,[r1]
	ADD r0,r3,r0
	ADD r0,r3,r0
	BL2
	L1
	LDR r0,[r2]
	ADD r0,r4,r0
	L2
		LDR r0,[r1]
		LDR r0,[r2]
		ADD r0,r3,r0
		ADD r0,r4,r0

Codesize = 16 bytes Number of clock cycles = 04	Codesize=12 bytes Number of clock cycles = 05	Codesize = 10 bytes Number of clock cycles = 07

Various programs for the same application with different code sizes and number of clock cycles are illustrated in table I. Clock cycles are estimated using IAR workbench for ARM. Hence appropriate code can be written on the basis of available memory space, and battery energy budget. For the ARM based systems, code size and number of clock cycles are used. Various programs are written and code size, clock cycles are estimated as given in table II.

The results are shown in table (III) and figure 6. From table (III) for the task T_1, T_2, T_3 optimized code sizes are 29 bytes, 30 bytes, 27 bytes respectively with system execution time of 37.63 µsec, energy consumption of 940 nJ< 1000nJ and system code size of 86 bytes. Also execution time for each task T_1, T_2, T_3 is less than that of 30, 40 and 50µsec. respectively. Table (III) shows, all the possible combinations of trade off tuples X_i, which satisfies the energy consumption and execution time constraints. The combinations of code sizes for each task that results in optimized solution of (27) are joined by line. Use of Heuristic algorithm results in large number of combinations of different code sizes, along with possible optimized solution.

TABLE II.
POSSIBLE CODE SIZES AND NUMBER OF EXECUTION CYCLES

TASK 1		TASK 2		TASK 3	
Si bytes	ni	Si bytes	ni	Si bytes	ni
24	64	26	56	24	48
25	48	27	36	25	40
27	40	28	32	26	32
29	24	29	20	27	28
31	16	30	12	28	24
--	--	31	08	29	16
--	--	--	--	30	08

TABLE III.
OPTIMAL SOLUTION OBTAINED USING HEURISTIC ALGORITHM

Task	Code Size (bytes)	Execution Time (µsec)	Energy Consumption (nJ)
T_1	29	12.24	384
T_2	30	11.11	108
T_3	27	14.28	448
Total	86	37.63	940

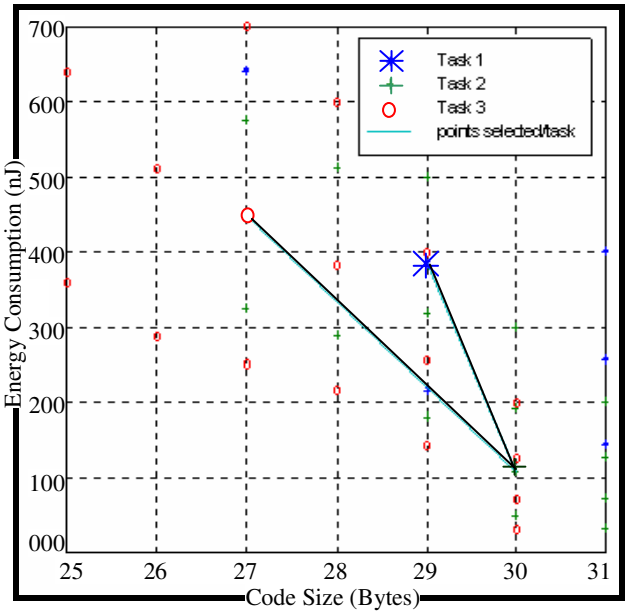

Figure 6. Code Size Vs Energy Consumption (Matlab simulation)

VI. CONCLUSION

In this paper, a new strategy for code size based energy consumption optimization in real time embedded system is discussed. Mathematical modeling of heuristic algorithm is implemented using Matlab and simulation results are presented. Optimization methodology is dedicated to ARM processor based systems. Algorithm seems to be powerful tool, which provides optimized code size, execution time and energy consumption, for low power battery-operated ARM based real time embedded system.

REFERENCES

[1] Bita Gorji Ara, Pai Chou, Nader Bagherazadeh, Mehrdad Reshadi, David Jensen, "Fast and efficient voltage scheduling by evolutionary slack distribution," in proc. of the conference Asia Pacific Design Automation, Electronics design solution fair ASP-DAC-2004, pp.659-662.

[2] S.Nikolaidis,N.Kavvadias,P.Neofotistos,K.Kosmatopoulos ,T.Laopoulos,L.Bisdounis. "Instrumentation set up for instruction level power modeling," in Int. Workshop on Power and Timing Modeling, Optimization and Simulation, Spain, Sept.2002, pp. 71-80.

[3] Insik Shin, Insup Lee ,Sang Lyul Min, "A Design approach for real time embedded systems with energy and code size constraints", in proc. of conference on Real Time and Embedded systems and Applications,RTCSA-2004.

[4] Ashok Halambi ,Aviral Shrivastava, Partha Biswas, Nikil Dutt, Alex Nicolau, "An efficient compiler technique for code size reduction using reduced bit width ISAs," in proc. of Design Automation and Test in Europe Conference,DATE-02,pp. 402-409.

[5] Ahmad Zmily, Christos Kozyrakis , "Simultaneously improving code size, performance and energy in embedded processors," in proc. of Design Automation and Test in Europe Conference,DATE-06,pp. 224-229.

[6] C.M.Krishna, Yann-Hang Lee, "Voltage clock scaling adaptive scheduling techniques for low power in hard real time systems," IEEE transaction, Computers, Vol.52, No. 12, Dec.2003, pp. 1586-1593.

[7] Ramesh Mishra, Namrata Rastogi, Dakai Zhu, "Energy aware scheduling for distributed real time systems," in proc. of Int. symposium on Parallel and Distributed Processing,IPDP-03,Nice,France.

[8] Osman S.Unsal, Israel Koren, "System level power aware design techniques in real time systems," IEEE transaction Vol. 91, No.7, July 2003, pp. 1055-1069.

[9] Johan Pouwelse, Koen Langendoen, Henk Sips "Dynamic voltage scaling on a low power microprocessor", in proc. of 7th annual International Conference on Mobile Computing and Networking. Rome, Itali, July 2001, pp.251-259.

[10] Padmanabhan Pillai, Kang G. Shin, " Real time dynamic voltage scaling for low power embedded operating system," in proc. of 18 th ACM symposium on Operating Systems Principles, SOSP,Banff,Canada,2001, pp.89-102.

[11] Flavius Gruian, "Hard real time scheduling for low energy using stochastic data and DVS processors," in proc. of Int. symposium on Low Power Electronics and Design California, USA, 2001,pp. 46-51.

[12] Flavius Gruian, "System level design methods for low energy architectures containing variable voltage processors," in proc. of conference on Power Aware Computer System PACS, Cambridge, 2000, pp.1-12.

[13] Trevor Pering, Tom Burd, Robert Brodersen, "The simulation and evaluation of Dynamic Voltage Scaling algorithms," in proc. of ISLPED, Monterey, USA, 1998, pp. 76-81.

[14] C. M .Krishna, Kang G. Shin, "Real time systems," Mc Graw Hill international edition 1997,pp. 47-52.

[15] Ying Zhang and Krishnendu Chakrabarty, " Energy aware adaptive check pointing in embedded real time systems," in proc. of conference on Design Automation and Test , Europe, IEEE computer society, DATE -03,pp 918-925.

[16] Willan Fornaciari, Paolo Gubean, Donatella Sciuto, Cristina Silvano, "Power estimation of embedded systems: A hardware/software codesign approach," IEEE transaction, Very Large Scale Integration (VLSI) systems, Vol.6, No.2, 1998, pp. 266-275.

[17] Youngsoo Shin, Kiyoung Choi, Takayasu Sakurai, "Power optimization of real time embedded systems on variable speed processors," in proc. of IEEE ACM international conference on Computer Aided Design,2000,pp. 365-368.

[18] S. D. Chede, Dr.K.D.Kulat, "Hardware/software codesign techniques in low power embedded system," in proc. of Int. MultiConference of Engineers and Computer scientists IMECS,2007,Hongkong,pp. 1716-1721.

Santosh D. Chede was born in 1968. He received B.E. degree in Industrial Electronics and M.E. degree in Electronics Engineering from Sant Gadge Baba Amravati University, Amravati, Maharashtra, India in 1990 and 2000 respectively. He is currently pursuing PhD degree in Electronics Engineering from Visvesvaraya National Institute of Technology, Nagpur, India.

His current research interests include low power hardware/software codesign strategy in real time embedded system, especially low power implantable pacemaker design.

Santosh D. Chede is life member of Indian Society for Technical Education (ISTE).

Kishore D. Kulat was born in 1958. He received B.E. degree in Electrical Engineering from Visvesvaraya Regional College of Engineering (VRCE) Nagpur and M. Tech. degree in Electronics Engineering from VJTI, Mumbai, India in 1980 and 1984 respectively. He received PhD degree from VNIT, Nagpur in 2003.

Currently he is professor in Department of Electronics and Computer science Engineering, VNIT, Nagpur. He has authored /coauthored over 55 papers in International/National journals and conferences. His current research interests include wireless (wi-fi, wi max) communication systems, networking and real time embedded system design.

Dr. Kishore D. Kulat is life member of Indian Society for technical Education ISTE, Fellow member of Institute of Electronics and Telecommunication Engineers (IETE) and member of Institution of Engineers (IE).