Analyzing Schedulability of Energy-oriented Distributed Real-time Embedded Software

Liqiong Chen¹, Guisheng Fan¹² and Yunxiang Liu¹+ 
¹Department of Computer Science and Information Engineering, Shanghai Institute of Technology, Shanghai, China 
Email: lqchen@sit.edu.cn
²Department of Computer Science and Engineering, East China University of Science and Technology, Shanghai, China
Email: gsfan@ecust.edu.cn, yxliu@sit.edu.cn
Corresponding author: yxliu@sit.edu.cn

Abstract—As computer systems become increasingly inter-networked, most of critical systems are distributed real-time embedded (DRE) system. A challenging problem faced by researchers and developers of DRE system is devising and implementing an effective method that can analyze requirements in varying operational conditions. In this paper, we analyze the requirements of DRE software and construct the corresponding energy consumption model, which is divided into fork module and leaf module based on its characteristics, and an energy consumption schema with time constrains is proposed for DRE software. The concept of critical task is presented according to the different position of task in the module, then constructing divide task set for module based on the characteristics of module and its critical task’s position, the idle time allocation strategy and DVS adjustment method of sub task set are also advanced. Finally, a specific example is given to simulate the analysis process, the results show that the method can be a good solution to analyze DRE software.

Index Terms—Distributed real-time and embedded system; Petri net; energy consumption; DVS; critical task

I. INTRODUCTION

Distributed real-time embedded (DRE) systems are becoming increasingly widespread and important. Most of critical application is embedded systems that control physical, biological, or defense processes and devices [1]. For example, a typical networked DRE system will consist of multiple subsystems, which may involve combinations of both local and distributed deployment. In this environment, developers require an effective method that can help identify requirements and design defects before a commitment is made to a particular design strategy. In these early development phases, the cost effectiveness and ease of use of validation tools is significant, as well as the level of rigor supplied by the modeling language and environment. Besides meeting timing constraints, energy consumption has become a major consideration in DRE design. Even in an energy-rich platform, energy consumption has raised a serious of concerns with respect to reliability and cost [2].

Power consumption is one of the critical design considerations for embedded systems. Reducing power consumption can extend battery lifetime of portable systems, decrease chip cooling costs, as well as increase system reliability. Dynamic voltage scaling (DVS) [3] is a popular technique for reducing energy consumption, especially in DRE systems where each component could take hundreds of cycles to execute. Based on the DVS technique, energy management schemes in real-time application have been extensively explored. Fewer works, however, have focused on energy management for DRE systems. Therefore, energy consumption has become a hot research in DRE software not only to reduce energy consumption but also to reduce associated cost.

To address the schedulability of energy-oriented DRE systems, this paper makes three main contributions to the state of the art in DRE. First, it uses formal techniques in an accessible and cost-effective manner to support optimizing energy consumption of DRE systems. The approach is based on Petri nets, an established formal method which has been widely used to model and analyze concurrent and distributed systems. Second, we extend for Place Timed Petri net and propose an Hierarchical Distributed Real-time Embedded net (HDRE-net) model. A optimizing energy consumption schema by using DVS capability is advanced. Third, We propose the concept of critical task and divide task set for module based on critical task. The DVS adjustment method of task set and optimizing energy consumption steps of whole application are advanced, and its enforcement algorithm is also given. Finally, we explain the effectiveness and feasibility of method by using ELC sub system.

II. COMPUTATION MODEL

A. Definition of HDRE-net

Timed Petri nets (TPN) is a mathematical formalism, which allows to model for the features present in most concurrent and real-time systems[4], such as concurrent, asynchronism and distribution, etc. Some recent researches indicate that TPN is powerful enough to describe behavioral features of DRE software. The basic concepts of it can refer to [5]. In this paper, we extend for TPN and establish a model for analyzing DRE software.
Definition 1: A tuple $\Sigma=(TPN, I, \gamma, \mu)$ is called Distributed Real-time Embedded net (DRE-net) iff:

1. $TPN=(P, T, \Sigma, M_0)$ is a Timed Place Petri net;
2. $I \cap P$ is a special place, which is called the interface of $\Sigma$ and denoted by the dotted circle;
3. $\gamma$ is the priority function of transition. $\gamma(i)=(\alpha_i, \beta_i)$, where $\alpha_i, \beta_i$ are called the primary and secondary priority of transition $i$;
4. $\mu: T \to N$ is the unit energy consumption of transition, the default value is 0.

The distribution of token in each place at time 0 is called the marking of DRE-net model, denoted by $M$. The marking $M(p)$ denotes the number of tokens in the place $p$. $M=M^P \cup M^T$, where $M^P$ is the available tokens of $M$, $M^T$ is the unavailable tokens of $M$. For any $x \in (P \cup T)$, we denote the pre-set of $x$ as $x^*=\{y|\exists (P \cup T)(\gamma(y), x) \in E\}$ and the post-set of $x$ as $x^+=\{y|\exists (P \cup T)(\gamma(x), y) \in E\}$. $M(t_i)=(\alpha_i, \beta_i)$ is called the initial priority of transition $i$.

Definition 2: A six tuple $\Omega=(\Sigma, I, \Gamma, T, \alpha, \beta)$ is called Hierarchical Distributed Real-time Embedded Net (HDRE-net) model, where:

1. $\Sigma$ is a DRE-net model, which describes the basic structure of $\Omega$;
2. $\Gamma=\{|\gamma|\in Z^*\}$ is the finite set of DRE-net and HDRE-net, each element is called a page of $\Omega$;
3. $I \cap T$ is the set of substituted operation, each page of HDRE-net corresponds to a substituted node and denoted by the double rectangle;
4. $T \cap I$ is the page allocation function, whose function is to allocate the page to the substituted node;
5. $P \cap \Gamma$ is the set of interface node, which describes the input and output of substituted node, and denoted by double circle;
6. $\Gamma \cap P$ is the mapping function of interface, which maps the interface node into the input and output of the operation.

From the definition, we can get that DRE-net is a special case of HDRE-net, that is, $\Gamma$ of HDRE-net model is empty. We will analyze the operation mechanism of HDRE-net model in the following.

B. Operation mechanism of HDRE-net

Because the tokens in HDRE-net model include time factor, therefore, we will introduce the concept of wait time in this paper.

Definition 3: Let $\Omega$ be a HDRE-net model, which reaches marking $M$ at time 0, $\forall p \in P$, place $P_i$ has $j$ tokens in marking $M$, $P_i^j$ is the $k$th token of $P_i$. Vector: $TS(P_i)=(TS_1, TS_2, \ldots, TS_i)$ is the wait time of place $P_i$, where $TS_i$ is the earliest firing delay under state $S$. $TS_i$ is called the wait time of $P_i$, $P_i^k$ is the wait time of $P_i$.

TS $\forall P_i \in P$: the wait time of $P_i$ is $P_i^k$, while $P_i^k=0$ represents the token is available. Recorded $TS(M, 0)$ as the wait time set of places under marking $M$. A triple $S=(M, TE, TR)$ is called a state of $\Omega$ at time $\theta$. Where $M$ is marking, which describes the distribution of resources; $TS(M, 0)$ is the time stamp of marking $M$, which depicts time properties of system.; $TE$ is the energy consumption of reaching state $S$. Initial state $S_0=(M_0, TS_0, TE_0)$, where $TS_0$ is a zero vector, that is, all tokens are available in the initial state, $TE_0=0$, which means the energy consumption of initial state is 0.

Definition 4: Let $\Omega$ be a HDRE-net model, $S$ is a state of $\Omega$ at time 0, for transition $t_i \in T$, iff:

1. $\forall p \in P: p \notin \alpha_i, t_i \to M(p) \geq W(p_i, t_i)$, then transition $t_i$ is strong enabled under marking $S$, denoted by $S(t_i \geq t)$, all strong enabled transitions under state $S$ are denoted by set $SET(S)$.
2. $\forall p \in P: p \in \alpha_i, t_i \to M(p) \geq F(p_i, t_i)$, then transition $t_i$ is weak enabled under state $S$, denoted by $S(t_i \geq t)$, all weak enabled transitions under state $S$ are denoted by set $WET(S)$.

The set $ET(S)=SET(S) \cup WET(S)$. If transition $t_i$ has weak enabled under state $S$ and at least pass through $\omega$ time units to be strong enabled, then $\omega$ is called firing delay of transition $t_i$ under state $S$, denoted by $FD(S,t_i)$.

From the definition, we can get that the firing delay of strong enabled transition is 0.

Definition 5: Let $\Omega$ be a HDRE-net model, $S$ is a state of $\Omega$ at time 0, $\forall t_i \in ET(S), \alpha_i \in \alpha^\Omega$, the firing of transition $t_i$ is effective if it meets one of the following conditions:

1. $t_i \in SET(S): \alpha_i \leq \min(\alpha_i, \beta_i)$, where $t_i \in SET(S), U(t_i) \in U(t_i)$
2. $t_i \in WET(S): SET(M)=\emptyset \land FD(S,t_i) \leq \min((FD(S,t_i)), t_i \in WET(S))$

The set $U(t_i)=\{t_i \in SET(S) \land \alpha_i = \alpha_i\}$. All the effective firing transitions under state $S$ are denoted by $FT(S)$.

Definition 6: Let $\Omega$ be a HDRE-net model, $S$ is a state of $\Omega$ at time 0, the model will reach a new state $S'$ by effectively firing enabled transition $t_i$ at time $0+\omega$, denoted by $S(t_i, \omega)=S', S'$ is called the reachable state of $S$, the computation of $M^*$, $TS'$, $TE'$ are based on the following rules:

1. Computing marking:
   $\forall P_i \in I$: $M(P_i)=M(P_i)^{\alpha}+W(P_i, t_i)^{\beta}+W(t_i, P_i)^{\mu}$
2. Computing wait time:
   First, adding wait time to the new generated marking: $TS(P_i)=\omega_i$, $P_i^k$ is generated when firing transition $t_i$.
   Second, modifying the wait time of tokens which are generated before the firing of transition $t_i$:
   $TS(P_i')=\max\{TS(P_i)^{k}, 0\}$, $TS(P_i') \geq 0$.
3. Computing energy consumption:
   $TE=TE'+FD(S, t_i) \times \mu$

III. MODELING DRE SOFTWARE

A. Requirements of DRE software

DRE software can be regarded as a number of modules; each module also contains a number of partially ordered, serial or parallel implemented sub tasks [9, 10]. The function of DRE systems will be distributed to a number of interrelated embedded devices, each device is responsible for certain functions, and has certain autonomy, but relies on the computation of other embedded devices. Among them, DRE system has $n$ tasks; each task is composed by a series of interrelated sub tasks set and a bus controller. The effective and
reliable communication between tasks is done by bus and bus controller. In this paper, we assume the communication between tasks is done by Controller Area Network (CAN).

As DRE software has strong performance requirements such as predictability, efficiency, reliability and security, et al. Therefore, it is necessary to consider above characteristics when describe the requirements of DRE software.

Definition 7: DRE software requirement model is a 9-tuple \( \Xi = \{ \text{TK, NT, RS, RL, CP, RT, D, En, MPS} \} \):

(1) TK, NT, RS are the finite tasks set, module set and resource set, the \( j \)th task of module \( N_i \) is denoted by \( \text{TK}_{i,j} \);
(2) RL is the relation between tasks, which may be sequence(\( > \)), choice(\( + \)), parallel(\( || \)) and exclusive(\( \ominus \));
(3) CP: TK \( \rightarrow \) (\( N_i \times N_i \times N_i \)) is the attribution function of task, which describes the running time and priority of task;
(4) RT: TK \( \rightarrow \) RS* is the resource function of task, whose function is to assign necessary resources to each task, RS* represents the multiple set of resource, that is, a task can use multiple sources;
(5) D is the deadline of whole application;
(6) En: TK \( \rightarrow \) N\( _i \) is the unit energy consumption of task, the unit energy consumption of task \( \text{TK}_{i,j} \) is denoted by \( e_i \);
(7) MPS: N\( _i \rightarrow \) (\( N_i \times N_i \)) is the max and min supply voltage of module.

B. Modeling DRE software

The model of task \( \text{TK}_{i,j} \) is shown in Fig.1, where place \( p_{\text{abp}}^{i,j} \) describes the state of task \( \text{TK}_{i,k} \) and its delay time is equal to the running time of task \( \text{TC}_{i,j} \). While transition \( t_{\text{abt}}^{i,j} \), \( t_{\text{cap}}^{i,j} \) describe the beginning and termination operation of task, place \( p_{\text{abt}}^{i,j}, p_{\text{cap}}^{i,j} \) describe the input and output parameters of task. And energy consumption of task \( \text{TK}_{i,j} \) is regarded as firing energy consumption of transition \( t_{\text{pap}}^{i,j} \), that is \( \mu(t_{\text{pap}}^{i,j}) = e_i \); transition \( t_{\text{pou}}^{i,j}, t_{\text{pou}}^{j,i} \) represent the operation of module has overtime. We introduce place \( p_{\text{en}}^{i,j} \) to control the running process of task.

Operator \( > \) represents the sequence relationship: If the firing of task \( \text{TK}_{i,j} \) can lead to the firing of task \( \text{TK}_{i,k} \), then the relationships between task \( \text{TK}_{i,j} \) and \( \text{TK}_{i,k} \) is sequence. \( \text{TK}_{i,j} \) is the forward task of \( \text{TK}_{i,k} \), while \( \text{TK}_{i,k} \) is the afterward task of \( \text{TK}_{i,j} \). The set \( \text{Forw}(\text{TK}_{i,j}) \), \( \text{Back}(\text{TK}_{i,j}) \subset \text{TK} \) are the forward and afterward task set of task \( \text{TK}_{i,j} \). The HDRE-net model of \( \text{TK}_{i,j} \rightarrow \text{TK}_{i,k} \) is shown in Figure 2(a), the substituted node \( \text{TK}_{i,j} \) and \( \text{TK}_{i,k} \) corresponds to the page of task \( \text{TK}_{i,j} \) and \( \text{TK}_{i,k} \), while interface node \( p_{\text{pou}}^{i,j} \), \( p_{\text{pou}}^{j,i} \) represent the input and output of substituted node \( \text{TK}_{i,j} \), which are mapped into the interface \( p_{\text{pou}}^{i,j} \), \( p_{\text{pou}}^{j,i} \) of task \( \text{TK}_{i,j} \). Because the relationship between task and substituted node is one by one, the substituted node is also called task in the following. In order to describe the sequence relationship, we introduce transition \( t_{\text{ou}} \) to transfer the result of task \( \text{TK}_{i,j} \) to the input interface of task \( \text{TK}_{i,k} \).

We can construct the model of \( \text{TK}_{i,j} + \text{TK}_{i,k} \) \( \text{TK}_{i,j} \mid \text{TK}_{i,k} \) and \( \text{TK}_{i,j} \bigtriangledown \text{TK}_{i,k} \) in the similar way, which are shown in Fig.2(b)-(d).

We will construct HDRE-net model of each module from bottom to up based on the relationships between task, as shown in Fig.3.

![Fig. 1 HDRE-net Model of Task](image1)

![Fig. 2 HDRE-net Model of basic Relation](image2)

![Fig. 3 HDRE-net Model of Module Ni](image3)
According to the communication principle of CAN bus, the communication process of task TK_{ij} sending message to task TK_{gf} is abstracted as a communication task TK_{ij-gf}. The HDRE-net model of task TK_{ij-gf} is shown in Fig.4. Where place p_{in}^{ij-gf} and p_{out}^{ij-gf} are the input and output interface of task, while place p_{i} and p_{o} describe the idle buffer and bus token resource. The process of communication task TK_{ij-gf} is: getting data packet after beginning to operate(t_{in}^{ij-gf}), and being in waiting for idle buffer (p_{ip}^{ij-gf}); the system will release buffer and bus token (t_{in}^{ij-gf}) after finishing sending data packet. The energy consumption of task TK_{ij-gf} is regarded as firing energy consumption of transition t_{in}^{ij-gf}, that is \mu(t_{in}^{ij-gf}) = e_{ij-gf}.

According to the requirement model and relationships between module, we can construct HDRE-net model of whole application from bottom to up, which is shown in Fig.5. Similarly, place p_{in}, P_{m} are introduced to describe the initial and termination state of whole application, transition p_{in}, P_{m} are introduced to describe the initial and termination operation of whole application. Place P_{ip}, P_{o}, P_{N}, P_{NT} are the input and output interface of module, interface node P_{in}^{i}, P_{out}^{i}, ..., P_{m}^{NNT} are the failure output of module. Transition t_{in}, t_{ex}, ..., t_{m}^{NNT} are used to transfer the overtime info of each module to the coordination place p_{in}. If any one module has overtime, the system will invoke transition t_{in} to do overtime handling for all modules.

Based on the constructed HDRE-net model, we will allocate priority to transition: in the HDRE-net model of task, the primary priority of transitions are equal to the priority of the corresponding task, while the priority of transition that introduced to describe process is (0,0), and the priority of substituted node is (0,0); the secondary priority of transition is divided into 5 grades according to its importance, the priority of transition t_{in} is 5, t_{ex} = t_{m} is 4, t_{m} = 3, t_{in} is 2, while the secondary priority of transition t_{in} and t_{ex} is the highest, which is used to describe overtime handling, for example, if the task is in the initial position, then aborting it, if the task is in the active position, then canceling it.

A. Offline DVS Scheduling of DRE Software

In this section, we will analyze the characteristics of task and module, then proposing the concept of communication task and critical task.

Let \Omega be a HRDE-net model, \forall TK_{ij}, TK_{gf} \in TK, i \neq g, if there exists communication task TK_{ij-gf} between task TK_{ij} and TK_{gf}, then task TK_{ij} is the communication forward task of TK_{gf}, while TK_{gf} is the communication afterward task of TK_{ij}; the communication forward and afterward task set are denoted by ConF(TK_{ij}) and ConB(TK_{ij}); meanwhile, module N_{j} is the directly afterward module of N_{i}, the set Next(N_{i}) is denoted as the directly afterward module set of module N_{j}; if the inner tasks of module don't output parameters to other module, then the module is leaf module, otherwise it is the fork module; the module that need interact with external module (other modules) is called critical tasks; the moment that module receive the outputs of other module's task is called critical point.

We will divide the task set of leaf module N_{i} according to the critical point set of module. We may set task TK_{ij} and TK_{ij} of module N_{j} need the output of other module, and task TK_{ij} is the forward task of TK_{ij}. Then the task set NT of module N_{j} is divided into three sub set: the task set before task TK_{ij} is denoted by Fbeg(TK_{ij}), the total running time is Start_{ij}; the task set in the middle of task...
$TK_{i,j}$ and $TK_{g,k}$ are denoted by $Bet(TK_{i,j}, TK_{g,k})$, and running time is $Start_{i,j} - Start_{g,k}$; the task set after task $Start_{i,j}$ is denoted by $Bbeg(TK_{i,j})$, the total running time is $D-Start_{i,j}$.

Fork module is divided into communication forward task and non-communication forward task according to whether the fork module need the parameters. We will divide fork task set into two type of fork module in the following.

We will firstly model for the fork module of non-communication forward module. Let task $TK_{g,k}$ of module $N_g$ need output parameters to the task $TK_{i,j}$ of module $N_i$, while task $TK_{p,q}$ need output parameters to the task $TK_{i,j}$ of module $N_p$. And the dynamic voltage of task in module $N_i$ and $N_p$ have finished adjusting, $TK_{g,k}$ is operated before the beginning of $TK_{p,q}$, we can divide task set $NT_{g,p}$ of module $N_g$ into three sub set, which is shown in Fig.6(a). The task set before the termination of task $TK_{g,k}$ is denoted by $Fend(TK_{g,k})$, and the total running time is $End_{g,k}$, where $End_{g,k} = Start_{i,j} - m_{g,k-a_{i,j}}$; the task set between the termination of task $TK_{g,k}$ and $TK_{p,q}$ is denoted by $Fbet(TK_{g,k}, TK_{p,q})$, and running time is $End_{g,r} - End_{g,k}$, where $End_{g,r} = Start_{p,q} - m_{g,r-a_{p,q}}$; the task set after the termination of task $TK_{g,r}$ is denoted by $Bbeg(TK_{i,j})$, and running time is $D-End_{g,k}$.

Second, we will model for fork module of communication forward module. The corresponding model is shown in Fig.6(b), we can divide task set $NT_{g,p}$ of module $N_g$ into three sub set: The task set before the termination of task $TK_{g,k}$ is $Fbeg(TK_{g,k})$, and total running time is $Start_{g,k}$, where $Start_{g,k} = Start_{i,j} - m_{g,k-a_{i,j}}$; the task set between the termination of task $TK_{g,k}$ and $TK_{p,q}$ is $Fbet(TK_{g,k}, TK_{p,q})$, and running time is $End_{g,r} - Start_{g,k}$, where $End_{g,r} = Start_{p,q} - m_{g,r-a_{p,q}}$; the task set after the termination of task $TK_{g,r}$ is $Bbeg(TK_{i,j})$, and running time is $D-End_{g,k}$.

![Fig. 6 Cross Module Task Set Division Example of Fork Module](image)

**B. DVS adjust for task set**

We will allocate idle time to task according to its characteristics, let the deadline of all modules be $D$, the steps of allocating idle time are as following:

1. **Calculating running time of the longest path by combining time reachability graph, we may set it be**: $TC_{max}$;
2. **Computing idle time**: $D_{rem} = D - TC_{max}$;
3. **Adjusting running time of the longest path, assuming that the path is composed by task $TK_{i,q1}$, $TK_{i,q2}$, ..., $TK_{i,qm}$, then:**
   
   $$TC_{i,j} = TC_{i,j} + D_{rem} \cdot e_{i,j} / \sum_{j=q}^{m} e_{i,j}$$

4. **Combining the process of diving task set, we can get running time of other tasks.**

Let the total running time of task set $SubT = \{TK_{i,j}, TK_{i,q2}, ..., TK_{i,qm}\} \subseteq NT$; be $D_{SubT}$, we will dynamically adjust supply voltage of task in $SubT$, the specific steps:

1. **Computing idle time**: $D_{rem} = D_{SubT} - \sum_{j=q}^{m} TC_{i,j}$;
2. **Recomputing running time of each task**: according to the relationship between unit energy consumption and voltage, we will allocate running time based on the proportion of energy consumption. The adjusted running time of task $TK_{i,j}$ is:
   
   $$TC_{i,j}' = TC_{i,j} + D_{rem} \cdot e_{i,j} / \sum_{j=q}^{m} e_{i,j}$$

3. **Computing the adjusted probability of task's running time** $T_{c,j}$: $ETC_{i,j} = TC_{i,j}'$;
4. **Computing new supply voltage of task $TK_{i,j}$:**
   
   $$V_{dd}' = V_{min} + \frac{V_{i}^0}{2ETC_{i,j}} + \sqrt{(V_{i}^0 + \frac{V_{i}^0}{2ETC_{i,j}})^2 - (V_{i}^0)^2}$$
   
   where $V_{i}^0 = \frac{(V_{max}^i - V_{min}^i)^2}{V_{max}^i}$;
5. **Computing new unit energy consumption of task $TK_{i,j}$** from the formula:
Step 5, otherwise doing Step 2; we can get new energy consumption of task is:

\[ e_{i,j} = \frac{1}{E_{TC_{i,j}}^*} * V_{dd,j}^* \]

(6) Computing the beginning and termination time of task \( T_{K_i,j} \):

\[ \text{Start}_{i,j} = \sum_{k=1}^{j} T_{C_{i,k}} \]  
\[ \text{End}_{i,j} = \text{Start}_{i,j} + T_{C_{i,j}} \]

B. Optimizing energy consumption and enforcement

Based on the state space of \( HDRE\)-net model, the energy consumption optimization of whole application is:

(1) Constructing the state space of each module; (2) Dividing task set for each module; (3) Computing running time of task set; (4) Adjusting \( DVS \) for task set; (5) Computing energy consumption of whole application:

\[ \sum_{i=1}^{\text{NT}} \sum_{j=1}^{\text{NT}} e_{i,j} \cdot * \text{TC}_{i,j} \]

For the bounded \( HDRE\)-net, because its reachability state set \( R(S_0) \) is a finite set, therefore, \( R(S_0) \) is viewed as a vertex set, and the direct reachability relationship between states is viewed as arc set, and the transition is marked in the corresponding arc, which constructs a directed arc. The directed graph is called Timed Reachable Graph in \( HDRE\)-net model. We can analyze state change, transition firing sequence and execution time of system by using Timed Reachable Graph, thus getting the related properties of \( HDRE\)-net model.

According to the construction algorithm of Petri net reachability graph, we can construct \( HDRE\)-net model. Based on the state space, we will use a heuristic algorithm energy consumption for \( DRE \) software. The algorithm will firstly compute running time of the longest path by combining with the requirement model and state space, then gradually optimizing energy consumption of other tasks, we will do following operation for module set \( TNT \):

Step 1: Computing the largest leaf module set \( NOB = NOB(TNT) \), then doing Step 2;

Step 2: If \( NOB(TNT) \neq \emptyset \), then arbitrary choosing module \( N_i \in NOB, NOB = NOB - N_i \), and doing Step 3, otherwise doing Step 6;

Step 3: Computing critical task set \( Crut(N_i) \) of module \( N_i \);

Step 4: If \( Sub(N) \neq \emptyset \), then arbitrary choosing task set \( Sub_j \in Sub(N) \), \( Sub(N) = Sub(N) - Sub_j \); and doing Step 5, otherwise doing Step 2;

Step 5: Computing critical task's beginning time of \( Sub_j \) by combining with the path of state space, then computing running time of whole task set \( Sub_j \) and dynamically adjusting each task in task set \( Sub_j \) according to \( DVS \) adjustment method of task set, then doing Step 4;

Step 6: Let \( TNT = TNT - NOB(TNT) \), then doing next operation.

V. EXPERIMENTS

In order to better describe the above modeling process and explain the correctness of analysis process, we use an actual case - Electronic Toll Collection (ETC) as an example. ETC system is an advanced system which consists of high-tech equipment and software such as electronics technology, computer technology, communications and network technology, and can achieve the function of automatically charging the cost of road without stopping vehicle. Strictly speaking, ETC application is the typical \( DRE \) software.

The workflow of ELC system is: the system will display traffic light once starting to operate, then informing lane computer to send start-up instructions to antenna controller when trigger coil detects the passage of vehicles. The read information from \( OBE \) will send to data processing center for data processing and charging. If success, then charge information screen will display “charge successfully, the amount of consumption”, the system mainly adopt camera and capture on the spot, and traffic police department will do the corresponding handling. The general distance of charging is 30m. The design speed of ETC lanes is 40km/h, then handling time of whole applications is 0.09s.

According to the actual requirements, we can divide the whole application into four function modules. Module 1 responses for controlling auxiliary equipment, including traffic lights, display screen, lever and coil; Module 2 responses for reading \( OBE \) data; Module 3 responses for processing charged data; And module 4 responses for capturing peccancy vehicles. Because module 3 need handle a large number of data, \( ARM9 \) is used in this module, the rest modules use 8051 Single-Chip. All modules use \( SJA100 \) as bus controller. According to the structure of ELC sub system, and combining with the functions of each module, we can divide task set for each function module, the attributes of task including: deadline, the ways of accessing resource, required resource, running time. The required preemptive task and deadline are shown in table I, the resource mainly includes buffer, communication buffer and bus token(where time unit TTU is 2ms, the unit energy consumption is mw). The energy consumption of whole ELC sub system is 740J, because the total running time of ELC sub system is 34.5TTU, the rest time for optimizing energy consumption is 10.5TTU, let the increased unit of running time be 0.1TTU. The max and min supply voltage of Module 1, 2, 3, 4 is 3.3V and 0.8V, while module 3 is 5V and 1.2V.

We can model for task, module and communication process, and construct the \( HDRE\)-net model of ELC sub system by merging the corresponding interface. Because the task has less conflict, the using of priority can reduce the corresponding level, and the communication buffer is set to 3.
Using the method proposed by Schmitz[7] and Yan[8], the optimized energy consumption is 577 and 618, which are higher than the method proposed in this paper, the comparison results are shown in Fig.7. The application results of Schmitz's method show that the proposed method can not only describes the characteristics of DRE software, but also can effectively reduce energy consumption.

Using the method proposed by Schmitz[7] and Yan[8], the optimized energy consumption is 525.07319, the optimization probability is 0.70955835.

VI. Related Works

The method that used to analyze energy consumption of DRE system are given in [9-11]. In[9], the authors addresses the problem of static and dynamic variable voltage scheduling of multi-rate periodic task graphs and aperiodic tasks in heterogeneous DRE systems. Reference[10] uses a comprehensive traffic description function at nodes and provides adequate information about the worst-case traffic behavior anywhere in the distributed network, thereby enhancing the system power management capabilities. A similar approach is given in [11], the authors incorporate a fault tolerance mechanism into DRE systems in order to utilize the available dynamic slack to maintain checkpoints and provide for rollbacks on faults. Previous approaches have relied on scaling down the supply voltage to reduce power consumption. However, supply voltage scaling can affect the performance. P.Paul et al[12, 13] used task graph to describe tasks and relations between tasks in DRE systems. More specifically, it discussed the schedulability analysis of DRE systems, and introduced several design optimization problems characteristic of this class of systems. The proposed method in this paper is easier to use because of the high abstraction level offered by using Petri nets. In addition, using the related tools of Petri net is also more suitable for DRE system.

Although the non-formal methods has made some corresponding results in the design of DRE systems, but they may cause some of the semantic ambiguity, in order to solve this problem, there has been some formal methods: G. Madl et al[14] used Time Automata (TA) to model various components of the non-preemptive real-time distributed embedded systems and converted the system scheduling problem into TA state reachability. As the TA model implied the existence of global clock, it unfit for modeling the Distributed Systems. V. Marcel et al[15] used the time extended of Vienna Development Method (VDM++) to stipulate DRE systems, and used VDM verification tools to verify the properties of system. However, comparing with other formal methods, the VDM may be more difficult to understand and grasp for

Table I Requirement of Task in ELC Sub-System

<table>
<thead>
<tr>
<th>TK</th>
<th>TC</th>
<th>UE</th>
<th>TK</th>
<th>TC</th>
<th>UE</th>
<th>TK</th>
<th>TC</th>
<th>UE</th>
<th>TK</th>
</tr>
</thead>
<tbody>
<tr>
<td>TK1,1</td>
<td>1</td>
<td>2</td>
<td>TK1,2</td>
<td>1</td>
<td>3</td>
<td>TK1,3</td>
<td>4</td>
<td>TK1,4</td>
<td>5</td>
</tr>
<tr>
<td>TK1,1</td>
<td>1</td>
<td>3</td>
<td>TK1,2</td>
<td>1</td>
<td>5</td>
<td>TK1,3</td>
<td>4</td>
<td>TK1,4</td>
<td>5</td>
</tr>
<tr>
<td>TK1,1</td>
<td>1</td>
<td>4</td>
<td>TK1,2</td>
<td>1</td>
<td>5</td>
<td>TK1,3</td>
<td>4</td>
<td>TK1,4</td>
<td>5</td>
</tr>
<tr>
<td>TK1,1</td>
<td>1</td>
<td>2</td>
<td>TK1,2,4</td>
<td>0.5</td>
<td>TK1,3</td>
<td>15</td>
<td>TK1,4</td>
<td>2</td>
<td>TK1,5</td>
</tr>
<tr>
<td>TK1,1</td>
<td>1</td>
<td>2</td>
<td>TK1,2,5</td>
<td>0.5</td>
<td>TK1,3</td>
<td>15</td>
<td>TK1,4</td>
<td>2</td>
<td>TK1,5</td>
</tr>
<tr>
<td>TK1,1</td>
<td>1</td>
<td>2</td>
<td>TK1,2,6</td>
<td>0.5</td>
<td>TK1,3</td>
<td>15</td>
<td>TK1,4</td>
<td>2</td>
<td>TK1,5</td>
</tr>
</tbody>
</table>

According to the division method of module's task set, we can divide task set for each module, and computing new running time of each task by computing increase time of each task set, the computed running time, supply voltage and unit energy consumption are shown in Table II. The optimized energy consumption is 525.07319, the optimization probability is 0.70955835.

Table II Energy Consumption Properties of Task

<table>
<thead>
<tr>
<th>Task</th>
<th>TC'</th>
<th>V_dd'</th>
<th>e'</th>
<th>Task</th>
<th>TC'</th>
<th>V_dd'</th>
<th>e'</th>
</tr>
</thead>
<tbody>
<tr>
<td>TK1,1</td>
<td>1.18</td>
<td>2.99</td>
<td>1.54</td>
<td>TK1,2</td>
<td>1.46</td>
<td>2.6</td>
<td>1.93</td>
</tr>
<tr>
<td>TK1,2</td>
<td>0.80</td>
<td>2.13</td>
<td>1.89</td>
<td>TK1,3</td>
<td>1.46</td>
<td>2.6</td>
<td>1.93</td>
</tr>
<tr>
<td>TK1,3</td>
<td>0.80</td>
<td>2.13</td>
<td>1.89</td>
<td>TK1,4</td>
<td>1.46</td>
<td>2.6</td>
<td>1.93</td>
</tr>
<tr>
<td>TK1,4</td>
<td>0.80</td>
<td>2.13</td>
<td>1.89</td>
<td>TK1,5</td>
<td>1.46</td>
<td>2.6</td>
<td>1.93</td>
</tr>
<tr>
<td>TK1,5</td>
<td>0.80</td>
<td>2.13</td>
<td>1.89</td>
<td>TK1,6</td>
<td>1.46</td>
<td>2.6</td>
<td>1.93</td>
</tr>
</tbody>
</table>

Fig. 8 Comparison Results of Optimization Method
developers. A resource-based time Petri Net is proposed in [16] to model the DRE systems and analyzed the corresponding semantic, properties. In our work, it allows user to describe and model on the non-function of DRE system by using Petri net, which can be helpful in analyzing its performance, and they didn't describe the communication between modules which is the key issue of DRE systems.

VII. Conclusion
In this paper, we have proposed a HDRE-net to model and analyze energy consumption of DRE software. This approach is based on a formal model, Petri net, that allows to consider different components of DRE software. According to the characteristics of module, we divide it into leaf module and fork module, then dividing task set for module based on the concept of critical task; the DVS adjustment method of task set and energy consumption steps of whole application are advanced, its enforcement algorithm is also given. Finally, we explain the effectiveness and feasibility of the method by ELC sub system. Comparing with other related works, the advantages of this paper are: Constructing energy consumption model of DRE software; proposing new idle time allocation method and offline DVS scheduling of DRE model.

The study of DRE software is still underway at present. Our current research is focused on exploring formal method as means to improve its mapping into DRE’s architecture. The following two aspects are the main work in the next phase: (1) further improves this method, consider the fault-tolerant of each task to assurance system's schedulability; (2) developing the corresponding tools to support the modeling.

ACKNOWLEDGMENT
This paper is supported by key Foundation of Shanghai Educational Committee (07ZZ164, 06OZ016), Foundation of Shanghai Institute of Technology (YJ2004-05) and key subject Educational Committee (07ZZ164, 06OZ016), Foundation of Institute of Technology of China under Grant No. YJ2009-17.

REFERENCES


Liqiong Chen. She was born in 1982, Ph. D. candidate. Her research interests include distributed computing, embedded systems and formal methods.

Guisheng Fan. He was born in 1980, Ph. D. candidate. His research interests include service oriented computing, distributed computing and formal methods.

Yunxiang Liu. He was born in 1967, professor, Ph. D. supervisor, IEEE senior member. His research interests include software engineering, information security and formal methods.