Energy Management in Ad Hoc Mobile Wireless Networks

Sridhar G and Sridhar V*
Applied Research Group
Satyam Computer Services Ltd., Bangalore 560 025, INDIA
{sridhar_gangadharpalli, sridhar}@satyam.com
*Corresponding author

Abstract

Energy management issues are very important in the context of ad hoc mobile wireless networks in general and sensor networks in particular. Energy needs to be optimally utilized so that the nodes can perform their functionality satisfactorily. It is known that energy can be managed at various levels. In this paper, we propose energy management at three different levels: component, system, and network levels by suggesting an approach to conserve energy at each of these three levels. We illustrate the approach by considering network traffic with differing QoS requirements. We briefly sketch an approach for energy-centric system design.

1. Introduction

The current developments in mobile sensor devices indicate a future trend where we will have devices performing diverse functions in a fast changing environmental conditions. Also, these devices (a) need to be part of both infrastructure based and ad hoc mobile networks, and (b) need to be dynamically reconfigurable so as to provide adequate flexibility to adapt to changing applications and processing environment. Different applications running on a device have diverse resource requirements and this diversity presents a challenge during system design to provide efficient performance across different applications over a wide range of environmental conditions.

Three important aspects of ad hoc mobile wireless networks in general and sensor networks in particular are (a) Location management; (b) Energy management; and (c) Topology management. Location is an important attribute of senor or ad hoc mobile wireless nodes. In a recent paper [8], we have described an approach for location management in a sensor network wherein not all the nodes are GPS-enabled. In another paper [9], we have addressed the issues related to topology management and suggest an approach for maintaining the desired topology even under uncontrolled mobility of the involved sensor nodes. In the present paper, we address the issues related to energy management and describe how to achieve energy management at various levels, namely, component, system, and network levels.

The power requirements of a mobile system will progressively increase in spite of impressive developments in terms of the power efficient designs and compact battery resources. Traditional energy management is based on schemes that make decisions on when to activate or shut the system so as to minimize the energy consumption within the system. Task sequencing has also been proposed as a means for effective energy management in mobile devices [7]. In traditional design approaches, a one-time selection of an energy management scheme, from different alternative schemes for energy management, is made. However, an efficient energy management to fulfill the QoS requirements would require an effective energy policy optimization [3].

Reconfigurable systems offer a huge scope for adaptability and flexibility for managing different tasks with varying energy requirements and QoS requirements. Although reconfigurability can be used effectively to optimize the energy utilization, the cost of frequent reconfiguration itself needs a careful consideration [6]. A trade-off needs to be arrived in terms of the energy savings made against the cost incurred due to reconfiguration. Reconfigurability can be effected at different levels. For example, FPGAs can be used to achieve gate level dynamic reconfiguration to dynamically change the functionality or the performance of a component. Chujo [1] describes an FPGA implementation where a component that performs the role of a fault detector in a set of ECUs in normal conditions is dynamically reconfigured to perform the role of an ECU when a failure is detected in any of the ECUs. Havinga *et al* [3] discuss the aspects of system level reconfiguration in terms of effective partitioning of system level tasks for energy management.

2. Energy Management in Ad hoc Networks

The energy management in ad hoc networks is a very important aspect of the overall management of ad hoc networks. The mobile wireless sensor nodes in the field need to conserve energy and use it optimally in order to play the assigned role in an ad hoc network for a longer period of time [10]. Energy can be managed at various levels: Component level [2,4,11], system level [5], and network level.

Component level energy management

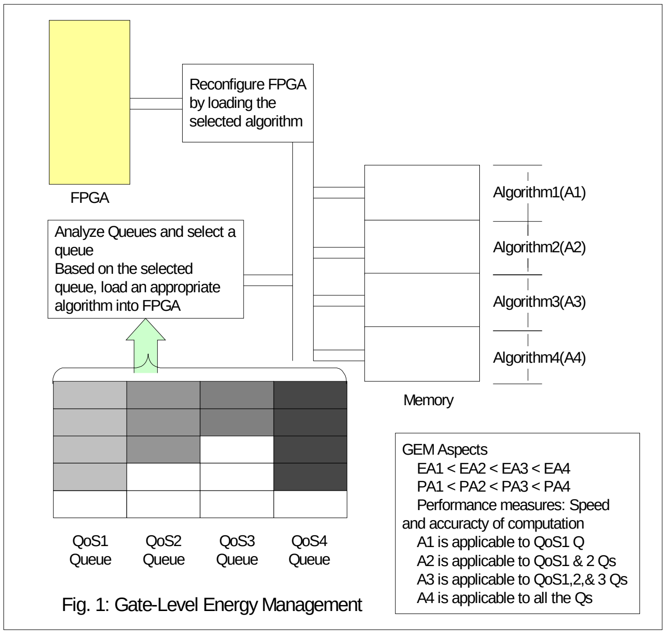


Fig. 1: Gate-Level Energy Management

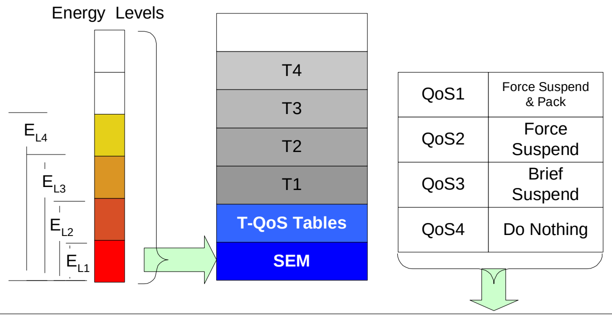
Component level energy management (CEM) gives an opportunity to control the energy utilization by various components of a system. There are several components that are part of a system that get used during initialization, and other components that get used at irregular intervals. By a suitable design, if the energy consumed by these components during idle time can be reduced close to zero, the main operation of the system can be better sustained. For example, at the completion of initialization, the hardware-software design can be such that the components used only in the initialization can be isolated to conserve energy. Let us consider a component, say, an FPGA. FPGA provides an opportunity to reconfigure the gate array and energy efficient algorithms to be used in the context of FPGAs can be designed. Fig. 1 describes an approach for gate-level energy management (GEM). As shown, FPGA can be reconfigured during run-time and for a given task, multiple algorithms can be designed that vary in their (a) energy consumption; (b) speed; and (c) accuracy. As shown in Fig. 1, we consider a scenario involving QoS. Ad hoc sensor networks carry different kinds of traffic with different QoS requirements. For example, real-time traffic such as voice puts the greatest demand on QoS while traffic related to FTP puts the least demand. In between are streaming media traffic and traffic due to interactive applications such as browser. With this observation, let us consider a task such as coding or decoding. Depending on the QoS, different codec algorithms can be developed that produce the same result. The possible trade-offs are amount of compression, the time it takes for compression, quality of compression (while compressing data, audio, or video), and the number of CLBs. As an illustration, we have considered the four different kinds of algorithms with the performance and energy consumption characteristics as shown in Fig. 1. There are four different queues to handle traffic requiring distinct QoS requirements. Note that A4 (Algorithm 4) can be used to handle traffic from the all the four queues while A1 can be used to handle traffic from QoS1 queue only. At any point in time, number of packets waiting in different queues vary and will, in general, be non-uniformly distributed. Keeping the primary objective as to conserve energy, we need to analyze the four queues to determine which is the best algorithm (of the four) that can be deployed and for how long. The decision of using an algorithm for a time period has to ensure that QoS of packets in various queues do not get affected. Yet another dimension that we need to keep in mind is the time it takes to reconfigure an FPGA and hence, it is better to avoid frequent reconfigurations. A first-cut solution is provided in Fig. 1A.

```
GEM Algorithm;
//  A1, A2,...,An are algorithms; Q1,Q2,...,Qn are the queues;
//  EAi < EAj for all i < j; Ai can be used on queues Qj for all j <= i;

{  Let FPGA reconfiguration time be Rt;
   Define FPGA recycle time Ft as f*Rt where f can be around 10;
   {  Repeat at Ft intervals
      Starting from Qn down to Q1, determine Qi  such that there is a packet in Qi whose TTL is within current time + Ft;
      Load Ai; Schedule packets in queus that can be processed by Ai;
   }
}
```

Fig. 1A: GEM Algorithm

System level energy management

We continue our discussion of ad hoc mobile wireless networks carrying different kinds of traffic. A node in such a network is the system under consideration and consider a scenario in which multiple tasks handle traffic of different kinds. As mentioned earlier, there are four different QoS requirements: QoS1 through QoS4. The proposed approach is to utilize the available energy in a most appropriate way. For example, if the energy falls below a lower watermark, there is a contention among these multiple tasks leading to QoS violations. However, under this condition, it may be better to trade-off low priority traffic so as to be able to handle high priority traffic. In order to achieve this, we define multiple energy levels and take appropriate actions as the available energy falls below these levels. The details are provided in Fig 2. Note that we have defined four different actions: Force Suspend & Pack: in this case, the tasks handling QoS1 traffic are forcefully suspended, packed, and sent to a remote energy-rich, resource-rich node for archiving purposes; Force Suspend: in this case, the tasks handling QoS2 traffic are forcefully suspended and archived onto a local store; Brief Suspend: in this case, the tasks handling QoS3 traffic are briefly suspended and the task status is retained in the memory for a quick resumption; and Do Nothing: if a node is



QoS1	Force Suspend & Pack
QoS2	Force Suspend
QoS3	Brief Suspend
QoS4	Do Nothing

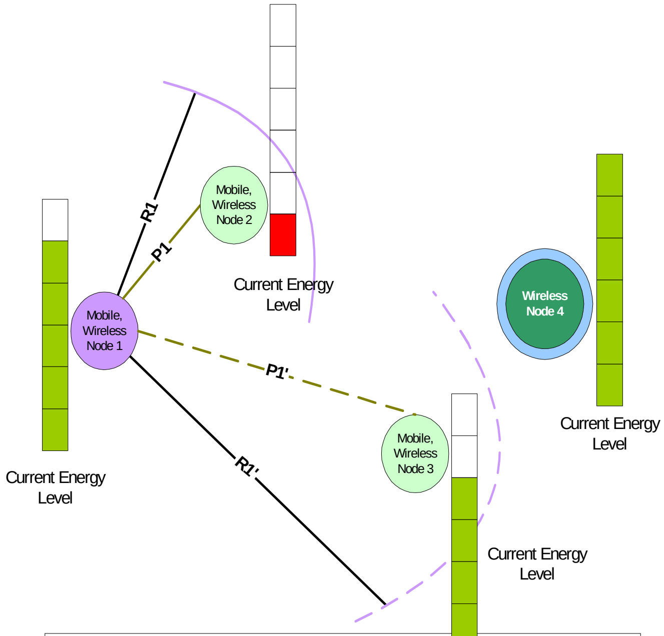
```
SEM

//  Remedial action: wheneven energy level falls below a lower watermark
On energy level < $E_{L3}$:
{  Check T-QoS table and locate QoS1 tasks;      Select as many number of QoS1 tasks to improve energy level;      Brief Suspend these tasks;      }
On energy level < $E_{L2}$ :
{  Check T-QoS table and locate QoS1 tasks;      Select as many number of QoS1 tasks and force suspend them;
   If not adequate { Check T-QoS table and locate QoS2 tasks;      Select as many number of QoS2 tasks and Brief suspend them; }
}
On energy level between $E_{L1}$:
{  Check T-QoS table and locate QoS1 tasks;      Select as many number of QoS1 tasks and Force suspend (if not already) and pack them;
   If not adequate { Check T-QoS table and locate QoS2 tasks;      Select as many number of QoS2 tasks and Force suspend them;   }
   If not adequate { Check T-QoS table and locate QoS3 tasks;      Select as many number of QoS3 tasks and Brief suspend them; }
}

// Recovery action: whenever energy level raises above an upper watermark
On energe level > $E_{L1}$ + epsilon
{  Check Bried Suspension list; Resume as many QoS3 tasks as possible;   }
On energe level > $E_{L2}$ + epsilon
{  Check  Force Suspension list; Resume as many as QoS2 tasks as possible ;
}
On energy level > $E_{L1}$ + epsilon
{  Check Force Suspension and Pack list; Get as many tasks as possible from remote systems; Resume as many tasks as possibe;
}
```

Fig. 2: System-Level Energy Management

operational, all effort is made to handle QoS1 traffic. When available energy goes above any of these energy levels, the recovery process begins first resuming tasks related to QoS3 traffic, resuming tasks related to QoS2 traffic next, and finally, resuming tasks related to QoS1 traffic.

Network level energy management

At the third level, we consider the issues related to energy management at network level. The objective is to conserve energy at network level by a cooperative loading of neighboring nodes. Consider a scenario in which paths are established from a node to another via multiple, intermediate nodes. An intermediate node carrying traffic expends energy and under the condition when its energy level is red, this energy outflow has to be avoided. This situation and an approach for managing the energy under this situation is described in Fig. 3. Note that a node in such an energy-sensitive network, at regular intervals, exchanges energy level information with its neighbors enabling them to react in a cooperative way. Further, such an exchange can be piggy-backed along with data packets, wherever possible, to reduce the overhead due to such transmissions. It also has been suggested in Fig. 3 that the paths need not have to be bi-directional. If energy levels are insufficient for setting up bi-directional paths or in order to normalize the energy expenditure across all the nodes of a network, two distinct uni-directional paths could be set up resulting in bi-directional communication (as described in Fig.3).



```
NEM;
// Node1, Node2 and Node3 are mobile wireless nodes;
// Node4 is an energy rich, resource rich node and acts as a co-host for nodes 1,2, and 3;
Processing in a node (say, N1)
{  Monitor energy level of neighbors;
   If (a neighboring node (N2)  has energy below a lower watermark and N1 uses N2 as an intermediate node for communication purposes (path P1))
   {  Determine if there are any other nodes within its range (R1);
      If there is none, increase the signal strength so as to increse range (R1') and reach a nearest neighbor (N3)
      If N3 has sufficient energy, use N3 as an intermediate node (P1')
      Else try to repeat the above steps until a suitable node is found;
   }
   Establish return path from N3 to N1;
   If N3 doesn't have sufficient energy to reach N1, retain the original return path from N2 to N1;
}
```

Fig. 3: Network-Level Energy Management

Energy-aware system design

In this subsection, we provide a brief sketch of an approach for an energy-centric system design. As described in the previous subsections, it is important to consider the energy-related issues at component / subsystem level. For each component / subsystem, during design time, it is required to propose alternatives and typical trade-offs should be in terms of cost, component life, and energy consumption. The related aspects would be size and weight. Considering these choices, select a best possible alternative for each of the components / subsystems. Note that, as described above, both process-level and algorithm-level analyses is required to design an energy-sensitive component. An example of algorithm-level analysis and exploiting the same in the context of FPGA has been provided in an earlier section. Another example is in the selection of reusable components and off-the-shelf components. Many times, these components are designed keeping generic requirements in mind and as a consequence, when used as a part of a particular system design, one has to evaluate the same from the point of view of energy overloading.

At system level, we need to evaluate the energy loading due to the design decisions such as (a) real-time OS; (b) system drivers; (c) protocol stacks; and (d) user applications. User interfaces with differing graphical content can be designed so that the interface can be switched on the fly from graphical to text based one whenever there is a need to conserve energy. Similarly, for each of the I/O interfaces, layered interfaces can be supported so that a right interface can be selected based on the need. This suggests a dynamic reconfiguration of I/O interfaces. In the case of protocol stacks, some of the run-time reconfigurable aspects include (a) switching on/off of logging; (b) switching on/off of certain features (TCP/UDP); (c) switching on/off codec features; and (d) switching on/off security features. In the case of system drivers, support for multiple devices, backward compatibility, and multiple speeds of operation could be traded-off dynamically based on the energy availability. Real-time OS could be hand-analyzed to identify a set of ordered set of features so that elements of this set could be traded off under low-energy situations.

3. Current Work

Computational needs in mobile sensor networks are growing rapidly. In spite of impressive developments in the battery technologies and diminishing power requirements for displays and other similar power intensive tasks, the energy management in adhoc sensor networks remains a challenge. We have suggested an approach in which the energy management is addressed at different levels of an ad hoc sensor network so as to achieve the best possible fulfillment of performance expectations.

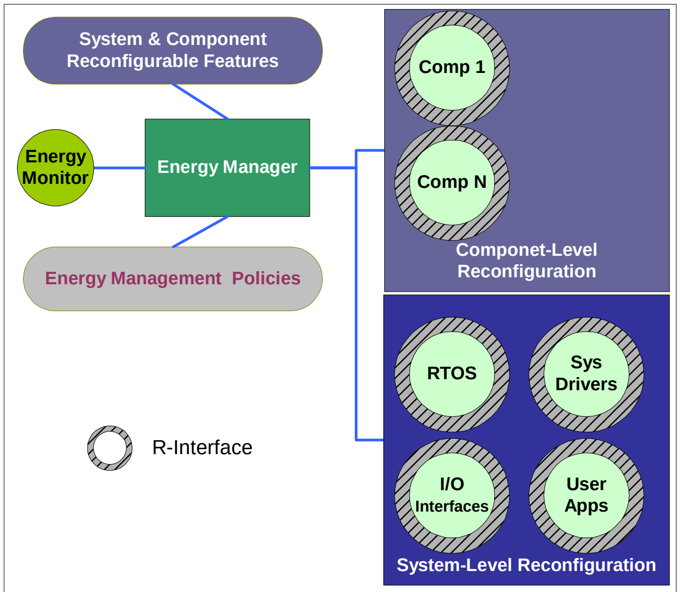


Fig. 4: Energy Management System

Our ongoing work is aimed at realizing an architecture for achieving an effective energy management across different levels of adhoc wireless sensor network. An initial sketch of such an architecture is depicted in Fig. 4. In order to achieve both component-level reconfiguration, it is essential that the components are designed with suitable R-Interface for interacting with the same for reconfiguration purposes. This suggests that components are designed to be self-reconfigurable and the reconfiguration itself is effected by Energy Manager. Similarly, it is necessary for system-level components to possess such R-interfaces. Continuing the example of System Drivers, a particular driver would have a following R-interface: *sd_CMode(mode)* with *mode* being “single” or “multiple”; *sd_Speed(speed)* with *speed* being a list of supported speeds; *sd_BackwardCompatibility(status)* with *status* being on or off. Note that, for example, when *sd_speed({300,600,1200})* is invoked, the modulation techniques required for handling higher speeds are off-loaded (specifically, the related code and static buffers).

References

[1] Chujo, N., “Fail-safe ECU System Using Dynamic Reconfiguration of FPGA,” *R&D Review of Toyota CRDL*, Vol. 37, No. 2, April 2002.

[2] Flinn, J. and M. Satyanarayan, “Energy-aware adaptation for mobile applications,” *Proceedings of the seventeenth ACM symposium on Operating systems principles*, South Carolina,1999.

[3] Havinga, J.M., L.T. Smit, G.J.M. Smit, M. Bos, P.M..Heysters, “Energy Management for Dynamically Reconfigurable Heterogeneous Mobile Systems,” *Proceedings of the fifteenth international Parallel and Distributed Processing Symposium*, San Francisco, California, USA, April 2001.

[4] Lee, J. and X. Yang, “Energy-aware routing algorithms in Ad hoc networks”, *CS395T Project Report*, Spring, 2001.

[5] Lettieri P. and M.B. Srivastava, “Advances in wireless Terminals,” *IEEE Personal Communications*, February 1999.

[6] Rakhmatov, D. and S. Vrudhula, “Hardware-software bi-partitioning for dynamically reconfigurable systems,” *ACM/IEEE International Symposium on Hardware/Software Codesign*, 2002.

[7] Rakhmatov, D. S. Vrudhula, C. Chakrabarti, “Battery-conscious task sequencing for portable devices including voltage/clock scaling,” *ACM/IEEE Design Automation Conference*, 2002.

[8] Sridhar, G., G. Uday, and V. Sridhar, “A Topology based Localization in Ad hoc Mobile Sensor Networks.” Submitted to *First working conference on Wireless On-demand Network Systems (WONS 2004)*, January 2004.

[9] Srinidhi, T., G. Sridhar, and V. Sridhar, “Topology Management in Ad Hoc Mobile Wireless Networks.” Submitted to *WIP workshop of Real time systems symposium (RTSS 2003)*, December 2003.

[10] Warrior, J. "Smart Sensor Networks of the Future," *Sensors Magazine*, March 1997.

[11] Yuen, W.H. and C.W. Sung, “On Network Connectivity and Energy Efficiency of Mobile Ad Hoc Networks,” *Proceedings of IEEE ICDCS* 2003.