SIMULATION OF MOBILE WIRELESS NETWORKS WITH ACCURATE MODELLING OF NON-LINEAR BATTERY EFFECTS

M. Handy, D. Timmermann
Institute of Applied Microelectronics and Computer Science
University of Rostock, Richard-Wagner-Str. 31, 18119 Rostock, Germany

Abstract

For the simulation of protocols and algorithms of mobile devices, an ideal energy source, i.e. a battery with linear charge and discharge characteristics, is often assumed. However, real batteries like lithium-ion cells show non-linear behavior, taking Rate Capacity Effect and Recovery Effect into account. The battery model presented by Rakhmatov and Vrudhula models non-linear battery behavior and can be utilized for lifetime optimization strategies since it provides a formal cost metric. A drawback of this lifetime optimization algorithm is that the load profile of the battery has to be known completely in advance. An estimation of battery lifetime at simulation runtime is not feasible. We present an algorithm for a runtime estimation of battery lifetime of lithium-ion cells. Our algorithm allows the integration of a non-linear battery model into network simulation environments for mobile devices. As an example, we describe the integration of our algorithm for battery lifetime estimation into a popular network simulation tool.

Keywords
Modelling of Energy Sources, Simulation Optimization

1. Introduction

For the design and implementation of communication protocols and algorithms, the use of simulation tools means a substantial productivity increase. Using simulations, protocols do not need to be implemented in explicit detail; in most cases, simulation of one or more protocol layer provides significant and sufficient results. Additionally, unlike for TCP/IP-based networks, the developer of communication protocols for mobile devices cannot fall back on existing infrastructure like intranet and therefore is dependent on other evaluation methods. Network simulation tools like NS2 [1] support developers through the integration of protocols for mobile networking. NS2 is a discrete event simulator and provides support for wired and wireless networks.

A key challenge for developers of mobile networking protocols is the unavailability of a durable energy source. Mobile devices are usually battery-powered. However, the lifetime of mobile devices is extremely important for market success. Therefore, primary design objective for portable systems is to reduce power consumption to increase the lifetime of a mobile device. Present network simulation tools provide simple energy models. For example, NS2 equips each node of a mobile network with an initial amount of energy and decreases this value every time a packet is sent or received. In NS2, the power consumed by a packet does not depend on the distance between sender and receiver and is default set to 281.8 mW for both sending and receiving.

Besides an accurate modelling of the energy consumption of mobile devices, precise knowledge of battery behavior is important for the reliability of simulation results. For the simulation of communication protocols, an ideal energy source, i.e. a battery with linear charge and discharge characteristics, is often assumed. However, real batteries like lithium-ion cells show non-linear behavior. The lifetime of a battery and its delivered capacity mainly depend on the current discharge profile. If the discharge current is higher than the rated current of a battery, then the energy efficiency of the battery decreases. Hence, the real capacity of the battery is lower than the nominal capacity. This effect is termed as *Rate Capacity Effect* and results from the fact that at high discharge rates, electrochemical reductions only occur at the outer surface of the cathode. At pulsed discharge, i.e. the alternating occurrence of discharge and idle periods, the lifetime of a battery can be increased. During idle periods, also called *Relaxation Times*, the battery can partially recover the capacity lost in previous discharge periods. This effect is known as *Recovery Effect*.

We present an algorithm for a runtime estimation of battery lifetime of lithium-ion cells. Our algorithm allows the integration of a high-level battery model into network simulation environments for mobile devices.

The remainder of the paper is organized as follows. Section 2 describes related work and the characteristics of the battery model proposed by Rakhmatov. Section 3 describes our algorithm for runtime estimation of battery lifetime. In this section, we compare our algorithm to the lifetime estimation algorithm proposed by Rakhmatov based on different battery load profiles. Additionally, simulation results are presented that show the impact of particular function parameters on the recovery capabilities of a lithium-ion cell. The integration of our lifetime estimation algorithm into existing simulation environments is presented in section 4. Section 5 gives some concluding remarks.

2. Preliminaries

2.1 Related Work

Lahiri et al. give a good survey of present battery models discussing their pros and cons [2]. They categorize nonlinear battery models into analytical, electrical-circuit, stochastic and electrochemical models. We discuss three of them here briefly. In [3], a battery model for lithium-ion cells is presented which is computationally intensive but the most accurate for this battery type. This model looks at the electro-chemical phenomena underlying the cell discharge. A large number of parameters are needed to model battery behavior, including electrode geometrics, concentration of the electrolyte, diffusion coefficients, reaction rate constants etc. With these parameters, a set of partial differential equations (PDE) is constructed for a specific battery. This model is not applicable for high-level battery simulation because of its computational intensity. Panigrahi et al. introduce a stochastic model of a battery, which models the *Rate Capacity Effect* and the *Recovery Effect* [4]. This model represents the cell behavior as a discrete time transient stochastic process, which tracks the cell state of charge. Compared to the PDE model, the stochastic model is significantly faster with an average error <3%. In [5], a discrete time model for the power supply sub-system is introduced that closely approximates the behavior of its circuit-level, continuous-time counterpart. This model is based on constructing a SPICE model of coupled network to represent the battery and is capable of modeling the *Rate Capacity Effect* but not the *Recovery Effect*. Rakhmatov et al. present in [6] an analytical battery model for a lithium-ion cell. This model is nearly as accurate as the PDE-model, however not so computational intensive. Furthermore, this model can be applied for lifetime optimization strategies since it provides a formal cost metric.

2.2 The Rakhmatov Battery Model

In [6], Rakhmatov et al. describe electrical and chemical processes inside a lithium-ion cell with a set of mathematical equations. The authors derive equations for both constant and variable battery load to compute the lifetime of a lithium-ion battery. For the estimation of battery lifetime, only two battery-specific parameters are needed. The α-parameter represents the capacity of the battery, β describes the non-linear battery behavior during charge and discharge periods. A parameter estimation technique is given in [7].

For variable load, the time-varying discharge rate is approximated by a piece-wise constant load. The variable load *i(t)* can be expressed as an n-step staircase function *U(t)*:

$$i(t)=\sum_{k=1}^{n} I_{k-1}\left[U\left(t-t_{k-1}\right)-U\left(t-t_k\right)\right] \qquad (1)$$

Equation 2 describes the impact of load profile on battery lifetime. As described above, α and β are battery-specific parameters, I_{k-1} denotes battery load during period k-1. The *A*-function computes the impact of nonlinear battery behavior where L is the battery lifetime, t_k the duration of period k, and t_{k-1} the duration of period k-1. See [6] for the detailed A-function.

$$\alpha=\sum_{k=1}^{n} 2I_{k-1}A\left(L,t_k,t_{k-1},\beta\right) \qquad (2)$$

Since equation 2 is hard to solve for L, Rakhmatov et al. present an algorithm to compute battery lifetime for a specific load profile. The input parameters of the lifetime estimation algorithm are a set of load values S_l and their timing S_t and the battery-specific parameters α and β. The output of the function is a two-element set where the first element is the time to failure. The second element denotes the differences between the n-term sum and α in equation 2. Figure 1 shows the described algorithm.

---

lifetime estimation(S_l, S_t, α, β)

IF $\alpha < 2I_0A(t_1,t_1,0,\beta)$ **THEN**

Find the smallest $t \in [0, t_1]$ such that:

$$\alpha < 2I_0A(t,t,0,\beta)$$

RETURN {*t*, 0}

Find the smallest integer $u \in \{2, 3, \ldots, n\}$ such that:

$$\alpha < \sum_{k=1}^{u} 2I_{k-1}A(t_u,t_k,t_{k-1},\beta)$$

IF *u* is not found **THEN**

RETURN

$$\{\infty,\ \alpha - \sum_{k=1}^{n} 2I_{k-1}A(t_n,t_k,t_{k-1},\beta)\}$$

Find the smallest $t \in [t_{u-1}, t_u]$ such that:

$$\alpha < \sum_{k=1}^{u-1} 2I_{k-1}A(t_u,t_k,t_{k-1},\beta) + 2I_{u-1}A(t,t,t_{u-1},\beta)$$

RETURN {t, 0}

END *lifetime estimation*

Fig. 1: Rakhmatov Algorithm for Battery Lifetime Estimation.

This algorithm allows a lifetime estimation of a lithium-ion cell for any load profile. A drawback of the algorithm is that the load profile of the battery has to be known completely in advance. A runtime estimation of battery lifetime is not feasible. However, for the use in simulation environments, such a premise is not handy. In simulations, usually the load profile is not known a priori, but a result of the simulation. We present an algorithm for a runtime estimation of battery lifetime for lithium-ion cells. Our algorithm allows the integration of the Rakhmatov-battery model into network simulation environments for mobile devices.

3. Runtime Battery Simulation

3.1 The Algorithm

Using the algorithm presented by Rakhmatov et al., a network simulation for mobile devices in general has the following order:

(a) Simulation without battery model and recording of the load profile.

(b) Running the lifetime estimation algorithm with the recorded load profile

The complete load profile has to be known in advance in order to estimate battery lifetime. Simulation effort could be decreased if present battery level could be estimated during simulation runtime. For a runtime algorithm, other parameters are needed than those described in the *lifetime estimation*-algorithm in figure 1. According to our assumptions, neither load profile S_l nor its timing S_t are known. We define a function *discharge*(l_t, t) that computes the battery level from its input parameters l_t as the present load, t as the timestamp, and some global variables. Note that the battery level is not equivalent to the energy dissipated up to the present timestamp since battery level is not linear dependent on energy consumption.

We sample the energy consumption of a simulated protocol or device with a frequency f. Hence, a simulation environment has to provide a value for energy consumption (or load current) at any time of the simulation. A single step of the staircase function of the variable battery load has the length Δ or $1/f$. If battery load changes during a simulation run, a new element containing the new load value is appended to the array L_I. Additionally, a new element is stored in the array of timings S_t that contains the timestamp of the load change. Thus, the arrays L_I and S_t record the load profile during a simulation run. Both are needed to compute the battery level. The global variable p is an element counter for both arrays. Figure 2 shows our algorithm.

```
PROGRAM Runtime_BatterySim
        GLOBALS: L_I, S_t, p, Δ, α, β
        LOOP WHILE simulation is running
          IF discharge(l, t) > α THEN
                  battery_lifetime = simulation time
                  EXIT
          END IF
          battery_level = discharge(l,t) / α
        END LOOP
END PROGRAM

discharge( l_t, t )
        IF l ≠ L_p-1 THEN
          L_p=l
          IF t ≠ 0 THEN S_p=t-Δ
          ELSE S_p=0
          increment p
        END IF
        S_p=t
        IF p=1 THEN RETURN
```

$$2 \times L_0 \times A(t,t,0,\beta)$$

ELSE RETURN

$$\sum_{k=1}^{p} 2 \times L_{k-1} \times A(t,S_k,S_{k-1},\beta)$$

END *discharge*

Fig. 2: Algorithm for Runtime Battery Simulation

The program *Runtime_BatterySim* defines the global variables L_I, S_t, p, Δ, α, β and calls the core function *discharge*(l_t,t). *Runtime_BatterySim* runs parallel to the simulation and continuously samples the load current. The load current together with a timestamp is passed to *discharge*(l_t,t). If the output of the *discharge*-function is higher than the battery parameter α, then the capacity of the battery is exhausted. Battery_lifetime then has the value of the present simulation time. Additionally, *Runtime_BatterySim* provides the battery level at any time of the simulation.

The *discharge*-function first checks whether a load change occurred during the preceding period. If yes, a new element is appended to both L_I and S_t. Let S_p be the last element of S_t. S_p is set to the present timestamp each time the function is called end denotes the end of the present period. Finally, discharge is computed for the interval [0,t] and returned to *Runtime_BatterySim*.

3.2 Experimental Results

For the validation of our algorithm, we estimated the lifetime of a battery for different load profiles. Therefore, we follow the experiments described in [6]. The authors validate their battery model by comparing experimental results to results of the DUALFOIL battery simulator. DUALFOIL uses partial differential equations to model discharge behavior of a lithium-ion cell and has proved its high accuracy [8]. From the load profiles described in [6] we chose five with different characteristics and compared the results to the lifetime values computed by our algorithm (Table 1, see end of paper). The first profile is a constant load experiment with 5 A/ m². Profile 2 contains two load changes. The timing is given in the third column of table 1. In profile 3, load is decreased stepwise from 20 to 5 A/m² and finally kept constant at 9.6 A/m². Profile 4 consists of ten 12.5-minute long periods followed by a constant load of 9.6 A/m². Each of the ten periods contains a 20 A/m² load for the first 1.5 minutes, followed by a 15 A/m² load for the next 2 minutes, followed by a 10 A/m² load for the next 3 minutes, and is concluded by a 5 A/m² load for 6 minutes. Profile 5 is similar to Profile 4, except for the reverse order of load values within the ten 12.5-minutes periods. The results of Table 1 prove the correctness and accuracy of our algorithm.

Furthermore, we present selected experiments performed with our algorithm. We make use of the advantage that our algorithm allows to query battery level at any time. For the battery-specific parameters, we adopt the values from [6] with α=271.47 and β=10.39. The first experiment records the battery level during the first 13 minutes of profile P4 and P5 (Table 1, see end of paper). We sample the battery level every 0.1 minutes. The results of this experiment are given in Figure 3. Both profiles have the same battery level at $t \approx 9.35$ min. For profile P5, during the first 12.5 minutes no recovery effects can be noticed; the battery level is strictly monotonic decreasing until t=12.5 min. Only at the load change from 20 A/m² to 5 A/m² at t=12.5 min the *Recovery Effect* is noticeable (increasing battery level). In contrast, profile P4 shows the *Recovery Effect* explicitly. At the load changes t=1.5, t=3.5, and t=6.5 min the battery level recovers. Figure 4 shows the first 2.5 minutes of profile P4 with one load change at t=1.5 min in detail. After the load change, a recovery of the battery level can be noticed. The *Recovery Effect* lasts for roughly 0.25 min, and then battery discharge reaches its normal level.

Notice that the *Recovery Effect* interacts with the underlying regular discharge. The *Recovery Effect* decreases as we move away from the time of load change. Furthermore, note that profile P4 has a 7.5 min longer lifetime than profile P5, as expected due to the preceding considerations (see Table 1).

The recovery effect occurs when battery load changes from a high to a lower value. Hardware and software developers can take advantage of this effect to increase

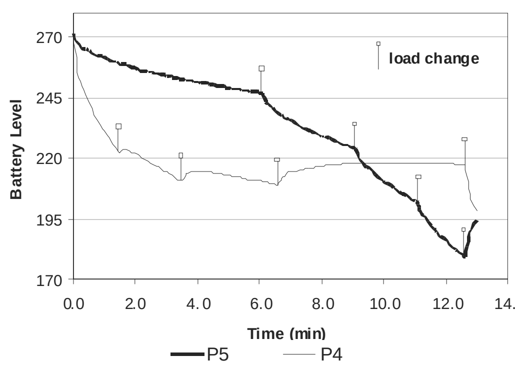

Fig. 3: Battery Levels of Profiles P4 and P5 during the first 12.5 minutes.

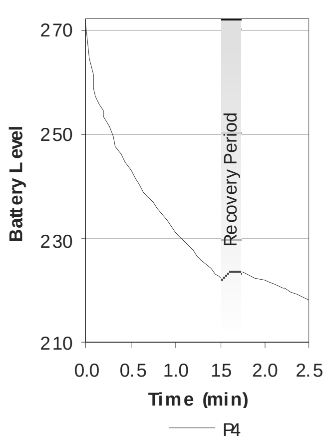

Fig. 4: Recovery Effect

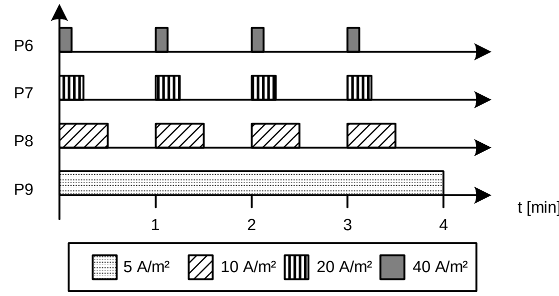

Fig. 5: Discharge Profiles for Battery Lifetime Estimation.

Profile	Lifetime [min]
P6	270.2
P7	303.3
P8	274.6
P9	280.1

Table 2: Battery Lifetime for Three Pulsed Discharge Profiles (P6, P7, P8) and a Constant Load Profile (P9)

the lifetime of mobile devices. For example, task scheduling algorithms could insert relaxation periods into the normal task processing time allowing the battery to recover. In another experiment, we compared discharge profiles with constant load to pulsed discharge profiles to estimate the increase of lifetime due to the *Recovery Effect*. As constant load profile, we used P9 with 5 A/m². We compared P9 to three pulsed discharge profiles P6, P7 and P8. Profile P6 discharges every minute 40 A/m² for 0.125 min. Profile P7 discharges 20 A/m² for 0.25 minutes at the same rate. Profile P8 discharges every minute 10 A/m² for 0.5 min. Figure 5 depicts the described load profiles. Notice that energy consumption for every profile P6-P9 is exactly the same. Hence, if an ideal energy source is assumed, battery lifetimes would be the same for each profile. In reality, a non-ideal energy source causes different lifetimes as shown in Table 2.

As expected, the results show different lifetimes for all profiles. Profile P7 has the maximum lifetime of all four profiles. Furthermore, P7 is the only pulsed discharge profile that exceeds the lifetime of the constant load profile. Profile P6 has the shortest lifetime since the *Rate Capacity Effect* dominates the *Recovery Effect* for the high discharge current. Table 2 shows the results of our experiments. The lifetime gain of pulsed discharge compared to constant load is 8% (P7/P9).

4. Application Example

4.1 Possible Applications

A precise simulation of a finite energy source is essential for any development process of software or hardware for mobile devices. For accelerated development, simulations are a common alternative to a full implementation. On the one hand, battery simulation can be utilized for software simulations, such as testing of network protocols. On the other hand, a precise battery model is important for a realistic evaluation of the efficiency of integrated circuits for mobile devices. Moreover, our lifetime estimation algorithm can be implemented in real hard- or software to enhance efficiency of power management functions in terms of intelligent idle-period insertions. In this chapter, we exemplarily describe the integration of our runtime algorithm into the network simulation tool NS2 [1].

4.2 Integration into NS2

The battery model does not replace the energy model available with recent NS2 releases but is plugged into the simulator as an additional module. To illustrate the differences between energy model and battery model, we will first briefly describe the energy model that comes with NS2. Then we will explain the necessary modifications to NS2 for the integration of the battery model. Finally, we give hints for the handling of the battery module.

The NS2 energy model equips each mobile node with an ideal energy source. For each packet sent or received by a mobile node, a certain amount of energy is subtracted from the node. Furthermore, the user can specify a node's idle energy consumption. When all energy of a node is consumed, the energy model declares this node as dead. From the OTcl-interface, a user can specify the values of transmission, reception, and idle power consumption.

Each mobile node in NS2 can be equipped with an energy model. Optionally, our battery model can be added to the mobile node. Notice that the energy model is a prerequisite for the battery model since the latter uses functions of the former. Figure 6 shows the block diagram of a mobile node in NS2 after the integration of the battery model.

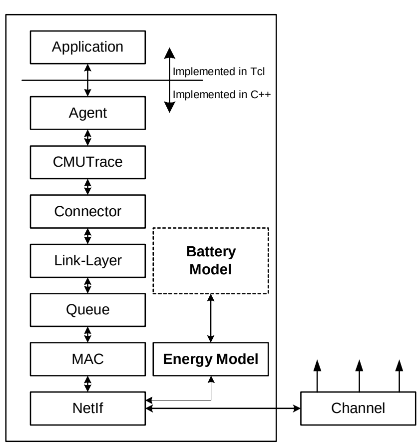

Fig. 6: Block Diagram of an NS2 Mobile Node with Battery Model

The battery model periodically polls the energy level of its node and computes the energy consumption of each period. Using the energy consumption of the last period and a timestamp, the battery model calculates the battery discharge up to this time. If the cumulative discharge exceeds the capacity of the battery, the battery is fully discharged and dies. It should be noted that after the integration of the battery model the energy model is no longer responsible for the termination of a node. This task is now performed by the battery model.

Similar to the energy model, the battery model can be attached to a mobile node from the OTcl-interface of NS2. Figure 7 shows a fragment of OTcl-code which is used for the configuration of a mobile node. The latter four lines of code specify the parameters of the battery model. The parameter *batteryModel* denotes the name of the battery model and can be used for the future integration of other battery models. The parameters *alpha* and *beta* are battery specific parameters as described in section 3. Remember that *alpha* represents the battery capacity and *beta* stands for the non-linear discharge behavior of the battery. The parameter *voltage* is necessary for the computation of battery load. The energy model provides the current energy level in Joule. However, the battery model needs as input parameter a discharge current in Ampere. The parameter *voltage* has to be converted from Joule into Ampere. We assume that the battery voltage is constant for the whole battery lifetime. A possible value for the *voltage* parameter can be the average of the open-circuit voltage V_o and the cutoff-voltage V_{cutoff} of the battery, as described in [6].

```
$ns_ node-config \
   -adhocRouting DSDV \
   -llType $opt(ll) \
    .
   -energyModel $opt(em) \
   -rxPower 0.3 \
   -txPower 0.6 \
   -initialEnergy $opt(ie) \
   -batteryModel RTBattery \
   -alpha 35220 \
   -beta 0.637 \
   -voltage 4.1
```

Fig. 7: Configuration of a Mobile Node with Battery Model in NS2

4.3 Simulation Example

As simulation example we use a scenario with 50 mobile nodes spread over an area of (670x670) m. This is a typical application in mobile ad hoc sensor networks. We use AODV (Ad Hoc On-Demand Distance Vector Routing) and DSDV (Destination-Sequenced Distance-Vector Routing) as routing protocols. The simulation is stopped after 100 seconds. For the battery specific parameters, we set *alpha*= 35220, *beta*=0.637, *voltage*=4.1 V. These values are estimations for a lithium-ion cell adopted from [7]. For our simulations, AODV shows lower energy consumption than DSDV. This is not surprising since AODV is a reactive routing protocol where the nodes only discover new routes on demand, whereas DSDV proactively tries to discover routes between all nodes in the network.

The purpose of the simulation example is to compare the dynamics of the energy level produced by the energy model to the battery level provided by the battery model. Figure 8 shows similar curve shapes for the energy and the battery level. Notice that, in contrast to the energy model, the curve of the battery model is not strictly monotonic decreasing. The short increase periods are due to the *Recovery Effect.*

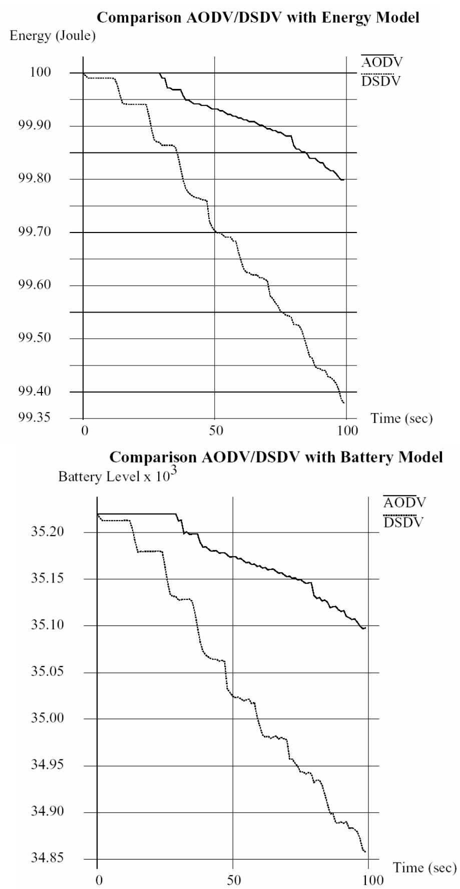

Fig. 8: Simulation of Ad-Hoc Routing Protocols AODV and DSDV with Energy Model (above) and Battery Model (below).

5. Conclusion

We presented an algorithm for the simple integration of the Rakhmatov analytical battery model into simulation environments for wireless networks and mobile devices. Hence, lifetime estimation of battery powered devices simplifies since we don't need to know the complete load profile of the battery in advance. With this algorithm, the handling of the Rakhmatov battery model considerably improves. Furthermore, we integrated the battery model into the network simulation environment NS2. Thus, non-linear battery effects like the *Recovery Effect* and the *Rate Capacity Effect* can be taken into account for simulations allowing the optimization of communication protocols and algorithms for maximum battery lifetime.

6. Acknowledgments

This work is supported in part by Gottlieb Daimler- and Carl Benz-Foundation, Ladenburg.

References

[1] The Network Simulator - ns-2, URL: http://www.isi.edu/nsnam/ns/

[2] K. Lahiri, A. Raghunathan, S. Dey, D. Panigrahi, Battery-driven system design: a new frontier in low power design, *Proc. of the 15th International Conference on VLSI Design (VLSID'02)*, 2002.

[3] M. Doyle, T. Fuller, J. Newman, Modeling of the galvanostatic charge and discharge of the lithium / polymer/insertion cell, *Journal Electrochem. Soc.*, 140(6), 1993.

[4] D. Panigrahi, C. Chiasserini, et al., Battery life estimation of mobile embedded systems, *Proc. of the 14th International Conference on VLSI Design (VLSID'01)*, 2001.

[5] L. Benini, G. Castelli, et al., A discrete-time battery model for high-level power estimation, *Proc. of the International Conference on Design, Automation and Test (DATE'00)*, 2000.

[6] D. Rakhmatov, S. Vrudhula, An analytical high-level battery model for use in energy management of portable electronic systems, *Proc. of International Conference on Computer Aided Design (ICCAD'01)*, 2001.

[7] D. Rakhmatov, S. Vrudhula, D. Wallach, Battery lifetime prediction for energy-aware computing, *Proc. of the International Symposium on Low Power Electronics and Design (ISLPED'02)*, 2002.

[8] P. Arora, M. Doyle, A. Gozdz, R. White, J. Newman, Comparison between computer simulations and experimental data for high-rate discharges of plastic lithium-ion batteries, *J. Power Sources*, 88, 2000.

#	Profile [A/m^2]	Timing [min]	Rakhmatov-Algorithm [min]	Our Algorithm [min]
P1	5	[0,∞)	280.1	280.2
P2	5-0-5	[0,206.5)-[206.5,275.3)-[275.3, ∞)	349	348.9
P3	20-15-10-5-9.6	[0,15)-[15,35)-[35,65)-[65,125)-[125,∞)	133.7	133.7
P4	{20-15-10-5}-9.6	{[0,1.5)-[1.5,3.5)-[3.5,6.5)-[6.5,12.5)}-[125,∞)	131.2	131.2
P5	{5-10-15-20}-9.6	{[0,6)-[6,9)-[9,11)-[11,12.5)}-[125, ∞)	123.6	123.7

Table 1: Load Profiles and Battery Lifetimes