CHAPTER 2

LITERATURE REVIEW

This chapter discusses the related work to this study. The literature review covers three major sections: *low power design techniques, error control schemes* and *wireless data link issues.* In each section, a few selected issues are discussed.

2.1 Low Power Design Techniques

In a mobile devices, such as cellular phones and portable computers, the goal to keep the battery lifetime reasonable make the study for low power design techniques most important. The next section presents low power consumption concepts in term of wireless devices and their strategy, and ends with a discussion on low power consumption in wireless links.

2.1.1 The Advance of Technology

The rapid advance in technology can be applied for several goals, such as to increase performance, to add functionality, and also to reduce energy consumption. Today's battery technology, processors and displays are rapidly improving continuously in terms of power consumption, battery life and battery weight. These issues will influence on how portable devices can be used.

Over the past two decades the semiconductor technology has been continuously improved. This technology has led to ever-smaller dimensions of transistors, higher packaging density, faster circuits, and lower power dissipation. The trend is expected

to continue beyond the year 2000. Over the past five years, feature sizes have dropped from about $f = 0.8$ to $f = 0.35$, where f is feature sizes. Semiconductor Industry Associates (SIA) has developed a road map for the next few years as depicted in Figure 2-1 (SIA, 1997). It is expected that a feature size of $f = 0.1$ will be reached in year 2007 within the context of current CMOS technology. Such advances provide an effective area increase of about an order of magnitude. To avoid the effect of high electric fields which would be present in very small devices, and to avoid the overheating of the devices, power supply must be scaled down. The power supply voltage is expected to be as low as 1.2 V in 2007.

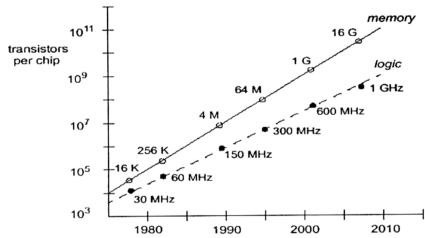

Figure 2-1: 1994 SIA Road Map Summary (SIA, 1997)

Meanwhile, power consumption has become a major concern because of the ever-increasing density of solid-state electronic devices, coupled with an increasing use of mobile computers and portable communication devices. The technology has thus far helped to build low-power systems. The speed-power efficiency has indeed gone up since 1990 by 10 times each 2.5 years for general-purpose processors and digital signal processors (DSPs). Table 2-1 shows the performance and power consumption of a few

recent processors[1]. However, this help will slow down because physical limits will be reached soon.

Processor	MHz	Year	SPECint-95	Watts	Watts/SPEC
P54VRT(Mobile)	150	1996	4.6	3.8	0.83
P55VRT(Mobile MMX)	233	1997	7.1	3.9	0.55
PowerPC 603e	300	1997	7.4	3.5	0.47
PowerPC 740 (G3)	300	1998	12.2	3.4	0.28
Mobile Celeron	333	1999	13.1	8.6	0.65

Table 2-1: Speed and power characteristics of recent processors [1] (Srivastava, 1999)

Design for low-energy consumption is certainly not a new research field, and yet remains one of the most difficult as future mobile system designers attempt to pack more capabilities such as multimedia processing and high bandwidth radios into battery operated portable miniature packages. Playing time of only a few hours for personal audio, notebooks, and cordless phones are clearly not very consumer friendly. Also, the required batteries are voluminous and heavy, often leading to bulky and unappealing products.

The primary problem is that in the case of battery technology, there is no equivalent of Moore's Law which forecasts a doubling of the complexity of microelectronic chips every 18 months, and Gilder's Law, which theorises a similar exponential growth in communication bandwidth. In contrast, battery technology has improved very slowly, and only a 20% improvement in capacity is expected over the next 10 years (Sheng, 1992). These trends are showed as in Figure 2-2 (Srivastava, 1999).

[1] Srivastava M.: *'Designing energy efficient mobile systems'*, Tutorial during MobiCom'99,

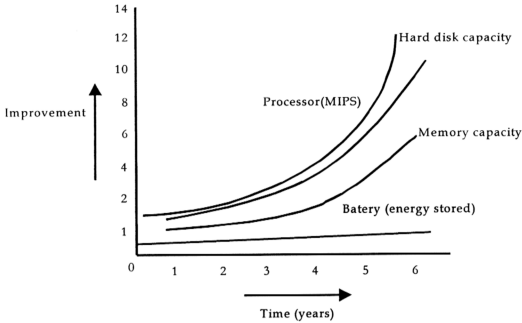

Figure 2-2: Improvement in technology[1] (Srivastava, 1999)

With increasing computation and communication functions desired for wireless mobile systems, the energy density of existing battery technologies are far from what is needed. Table 2-2 shows the energetic potentials of current battery technology.

The most recent advances in laptop batteries are in the form of better 'fuel gauging' of the battery, to give a more precise measure of the charge level and to estimate the time left before a recharge is needed (Truman, 1998). Although this is a useful technique, it does not extend battery life.

---

http://www.janet.ucla.edu/~mbs/tutorials/mobicom99.

Battery	Rechargeable	Wh/kg	Wk/litre
Alkaline MnO2	no	130	347
Li/MnO2	no	210	550
Zinc Air	no	280	1150
Lead acid	yes	30	80
Nickel-Cadmium NiCd	yes	40	130
Nickel-metal hybride NiMH	yes	60	200
Lithium-ion	yes	60	200
Methanol fuel cell	yes	6200	4900

Table 2-2: The energetic potentials of batteries[1] (Srivastava, 1999)

A promising technique might be *fuel cells*. A fuel cell is an electrochemical device that converts the chemical energy of a fuel directly to usable energy, electricity and heat, without combustion. The energetic potential is very high, a fuel cell running on methanol could provide power for more than 20 times longer than traditional nickel cadmium batteries in a comparably sized package (Dyer, 1999). They are theoretically quiet and clean like normal batteries. Another benefit is that fuel cells do not require lengthy recharging; instead they can be replenished quickly, simply by adding more fuel. Fuel cells were once prohibitively expensive, but sophisticated engineering has recently driven cost down considerably. However, designing a miniature fuel cell that can be mass-produced cheaply is a formidable task. Although initial prototypes have been built, no one has yet demonstrated a compact device that could be mass-produced at a cost lower than that of comparable rechargeable batteries.

Several researchers have studied the power consumption pattern of mobile computers. Laptops use several techniques to reduce energy consumption, primarily by turning them off after a period of no use, or by lowering the clock frequency. However, because they studied different platforms, their results are not always in agreement, and sometimes even conflicting. Lorch reported that the energy use of a typical laptop

computer is dominated by the backlight of the display, the disk and the processor (Lee, 1995). Stemm et al. (1996) concluded that the network interface consumes at least the same amount of energy as the rest of the system (i.e. a Newton PDA). If the computer can receive messages from the network even when it is 'off', the energy consumption increases dramatically. Ikeda et al. (1994) observed that the contribution of the CPU and memory to power consumption has been on the rise the last few years[2]. Even though it is difficult to compare these results because the measurements are made for different architectures, operating systems, communication interfaces, and benchmarks, there is a common pattern: there is no primary source of energy consumption. The energy spent is distributed over several devices and for several operations. The conclusion is that implementing an energy efficient system involves looking at all the functions in the system, and not just a single function such as, for example, networks protocol processing.

2.1.2 Low Power Design for Wireless Devices

In Figure 2-3, Lettieri et al. (1998b) have shown a systematic approach on how to design low power systems for wireless devices. At the lowest level, individual circuits and components are designed for low power. For example, Chandrakasan et al. (1995) discusses many special techniques for minimizing power consumption in CMOS circuits, applicable across all of the digital hardware shown in the figure, including DSPs, modems, and processors. Shrinking threshold voltages along with the use of techniques such as parallelism and pipelining continue to reduce power consumption

[2] The IBM Thinkpad laptops have shown an increase in the fraction of power consumption in the memory and CPU from18% to 40% from 1992 to 1995.

while maintaining performance. New display technologies and improved means of mass storage have brought additional benefits.

Lettieri et al. (1998b) named the next level of the hierarchy as intranode power management. Within a single wireless device, improved energy efficiency may be achieved for conventional components such as the CPU (Weiser, 1994) and the disk (Douglis, 1994) by turning them off or slowing them down when not needed. Othman (1999) has discussed this in more detail under the 'Power Management Strategies' section in her Phd thesis. She categorized power management strategies to three: Power Management Strategies for Hard Disks, Power Management Strategies for CPUs and Power Efficient Communication. The last one is more related to this project.

In her study, she has successfully adopted a well-known concept in distributed systems, load sharing, as an effective power management strategy for use in a mobile computing environment (Othman, 1999). On average, battery lifetime was extended by about 20% when load sharing was performed. In her experiment which combined load sharing with another power management strategy, battery lifetime was extended by an average of 33%.

Research also has shown that a significant fraction of the wireless node power is consumed by the wireless network interface cards that are responsible for transporting the data packets. This has been found to be true for both commercial wireless LAN products such as Lucent's WaveLAN card or Metricom's Metricom modem using ethernet-like frames to carry IP traffic, see (Stemm, 1996). Further, the fraction of power consumed by the wireless network interfaces is only likely to increase as these

nodes evolve towards a thin client network computer like mode with no disks and minimal or no CPUs, such as advocated by Bell Lab's PSA (Asthana, 1995). Increasingly, therefore, designing low power wireless computing nodes will be less of a disk or CPU power management problem and more of a wireless link interface power management problem.

The final level of the hierarchy is known as internode power management. To make the wireless interfaces more energy efficient, algorithms embodying power efficient protocols must be distributed across two or more wireless end points. This implies that we must examine the layers of the network stack through which the nodes interact. At the physical layer, the focus has been on providing "control knobs", such as variable processing gain (Chien, 1997) or variable data rates, for higher layer algorithms to make use of.

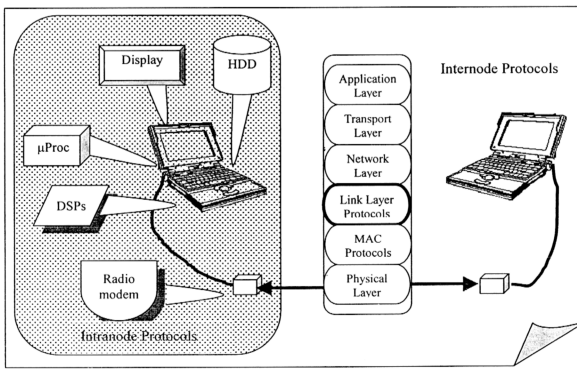

Figure 2-3: The Power Efficiency Hierarchy (Lettieri, 1998b)

At the MAC layer is perhaps the most developed of the protocols in terms of energy efficiency (Sivalingam, 1997), techniques such as scheduled access and power control reduce collisions, allow powering down of key components of the wireless interface hardware, and minimize transmit power, based on knowledge of the location of neighboring nodes and their transmission schedules.

2.1.3 Low Power Consumption

Wireless devices must be easy to use, must have good operating time, must be portable and be able to provide fast, efficient access to the network. There will be a proliferation of devices from simple voice phones to communicators that encompass the capabilities of a PC and phone. Power consumption is extremely important to have long operating time. Low power consumption can be achieved by transmitting a low power, higher degree of integration and by using low voltage IC technology.

Batteries are the largest single source of weight in a portable computer and cellular phones. While reducing battery weight is important, too small a battery can undermine portability - users may have to recharge frequently, carry spare batteries, or use their mobile computer less. Minimizing power consumption can improve portability by reducing battery weight and lengthening the life of a charge.

Power consumption of dynamic components is proportional to CV^2F, where C is the capacitance of the wires, V is the voltage swing, and F is the clock frequency. This function suggests three ways to save power.

1. Capacitance can be reduced by greater levels of VLSI integration and multichip module technology.
2. Voltage can be reduced by redesigning chips to operate at lower voltages. Historically, chips operate at five volts, but to save power, the Apple MessagePad operates at three volts. Manufacturers are rapidly developing a line of low-power chip sets for 2.5 and 3.3 volt operation.
3. Clock frequency can be reduced by trading off computational speed for power savings. In some notebook computers, the clock frequency can be changed dynamically, providing a flexible tradeoff. In order to retain more computational power at lower frequencies, processors are being designed that perform more work on each clock cycle (Chandrakasan, 1992).

Power can be conserved not only by the design, but also by efficient operation. Power management software can power down individual components when they are idle, for example, spinning down the internal disk or turning off screen lighting. Li et al. (1993) determined that for today's notebook computing it is worthwhile to spin down the internal disk drive after it has been idle for just a few seconds. Applications may conserve power by reducing their appetite for computation, communication and memory, and by performing their periodic operations infrequently to amortize the startup overhead. Since cellular telephone transmission typically requires about ten times as much power as reception, trading talking for more listening may also save power. Although screen lighting consumes a large amount of power, it has been found to greatly improve readability. Nevertheless, PDA's products have elected to omit screen lighting in favor of longer battery life.

Viterbi Codes

Another class of error correcting code operates on an arbitrary stream of bits, rather than fixed-sized blocks. These are the convolutional codes, sometimes called 'tree' codes. In a more general form, they are known as 'trellis' codes and are found in modern dialup telephone modems (e.g., V.32 and V.32bis). Convolutional codes are easily generated in hardware with a shift register, two or more exclusive-OR 'parity trees' and a multiplexor. These operations are also easily implemented in software. The length of the shift register defines an important parameter called the 'constraint length' K of the code. The larger K is, the better the code will perform. Because a convolutional code operates on a continuous stream, they are specified by the ratio of the input data and output symbol rates. For example, an encoder that produces two encoded symbols for each user data bit is known as a 'rate ½' coder, often abbreviated to 'r = ½'.

Convolutional codes have several interesting properties. They are especially easy to generate and provide excellent performance for a given amount of complexity. They are well suited to varying amounts of data, e.g. variable length data packets. They are also readily adapted to soft decision decoding. This uses symbol quality indications from the modem to aid the decoding process. For example, instead of slicing the demodulator output to binary 0 or 1 with a comparator, one uses an A/D converter to indicate the relative quality of each 0 and 1.

Viterbi decoding is one of two types of decoding algorithms used with convolutional encoding. The other type is sequential decoding. Sequential decoding has the

- **Latency**

  Both encoders at the transmitter and the receiver introduce latency; their processing delays the availability of the information to the receiver application. The amount of latency that can be tolerated depends strongly on the application. For interactive multimedia application such as video conferencing, keeping the delay small is critical.

- **Complexity**

  The number of computations per unit time or the chip area required to implement the encoder and decoder are common measures of complexity. In general, increased complexity implies increased power consumption and increased product cost. For battery-powered mobile transceivers, keeping complexity low helps maintain a tolerable battery lifetime.

In any communication system, there have always been errors and the need to deal with them. Basically there are two methods of dealing with errors:

1. **Retransmission**

   Retransmission methods, such as ARQ (Automatic Repeat reQuest) and variants such as Stop-and-Wait, Go-Back-N or Selective-Repeat recover errors by re-transmitting the erroneously received packet (Lin, 1984). The missing packets are retransmitted upon timeouts, or after explicit requests from the receiver.

2. **Error correction**

   Error corrective coding can be used to preserve the integrity of received data. With this technique, redundant information is sent with the message, such that the

receiver is able to detect errors in the received message, and correct the errors, thus, eliminating the need for retransmission. This addition of redundancy is called Forward Error Correction (FEC).

Computer communication generally implements a reliable data transfer using either methods, or a combination of them at different levels in the protocol stack. Error correcting codes like FEC are mainly used at the data link layer to reduce the impact of errors in the wireless connection. In most cases, these codes provide less than perfect protection and some amount of residual errors pass through. The upper level protocol layers employ various block error detection and retransmission schemes (e.g. (Huitema, 1996)). The explanation of two methods are discusses more details as follows.

2.2.1 Retransmission

Retransmission techniques for error control are called Automatic Repeat Request (ARQ) techniques. In practice, not every errored block can be recognized as such. However, proper design of the error detection code can make undetectable errors extremely rare. Details of block codes of error detection and correction are discussed in section 2.2.2. ARQ schemes assume the existence of a back channel from the receiver to the transmitter over which positive and negative acknowledgements (ACKs and NACKs) from the receiver inform the transmitter of blocks that have been received correctly or incorrectly. The three basic ARQ schemes are Stop-and-Wait, Go-Back-N, and Selective-Repeat.

a. Stop-and-Wait

In the stop-and-wait scheme, the transmitter stops transmitting after each block and waits for an ACK from the receiver indicating that the block has been correctly received. When the ACK is received by the transmitter, it sends the next block. If a NACK is received instead, the transmitter resends the block and waits again for an ACK. If neither an ACK nor a NACK is received after a sufficiently long time, the transmitter assumes that either the acknowledgement or the block itself has been lost, and acts as if it has received a NACK.

A disadvantage of the stop-and-wait ARQ is the transmitter time spent waiting for acknowledgements from the receiver. The transmission time lost by the transmitter is significant when the round-trip transmission is a significant fraction of the transmission time of a block. This wasted time prevents stop-and-wait from achieving the erasure channel capacity.

b. Go-Back-N

With the Go-Back-N scheme, the receiver transmits ACKs and NACKs as before, but the transmitter continues to transmit without waiting for acknowledgements, sending N blocks before it receives the acknowledgement of the first one.

When the receiver sends a NAK for block j (or the transmitter sees the acknowledgement for message j+1 without having seen an ACK or NACK for block j), the transmitter must resend block j. For simplicity, the go-back-N scheme also resends

every block after the lost block j. Thus, NACK is essentially a message to the transmitter to 'Go-Back-N blocks' and start retransmitting from there.

The transmitter must store N blocks for possible later retransmission, but the receiver does not need to store more than a single block since it passes block in order to the higher layers until there is corrupted block, at which point the receiver will disregard the next N-1 blocks (waiting for the retransmission of the corrupted block). Thus, the go-back-N requires more memory in the transmitter than stop-and-wait, but no additional memory is required at the receiver.

The problem with the go-back-N scheme is that it requires retransmission of N blocks every time a single block is corrupted. This inefficiency prevents go-back-N from achieving the erasure channel capacity. As with stop-and-wait, the performance loss is dramatic when the time required for one round trip transmission is sufficient for many blocks to be transmitted by the receiver. With Stop-and-Wait, the transmitter sends only one block during the round- trip time, but must retransmit every block if the first block transmitted is corrupted.

c. Selective-Repeat

The Selective-Repeat scheme avoids these inefficiencies by having the transmitter resends only blocks for which a NAK is received (or neither ACK nor NAK is received). Selective-Repeat does achieve the erasure channel capacity. Every block is either erased or it provides new information to the receiver. However, for Selective-Repeat the receiver must buffer blocks as it waits for the transmitter to fill in the gaps

caused by erasures. The ability to use feedback allows the erasure capacity to be achieved with relative ease. However, it does require significant memory for buffering at both the transmitter and receiver.

2.2.2 Forward Error Correction

The purpose of Forward Error Correction (FEC) is to improve the capacity of a channel by adding some carefully designed redundant information to the data being transmitted through the channel. The process of adding this redundant information is known as channel coding. Convolutional coding and block coding are the two major forms of channel coding. Convolutional codes operate on serial data, one or a few bits at a time. Block codes operate on relatively large, typically up to a couple of hundred bytes of message blocks.

FEC has been around for quite some time. It essentially began when Claude Shannon's revolutionary 1948 paper launched the field of information theory (Shannon, 1948). Error correction is a very deep and often highly mathematical subject. Shannon proved that it is possible to send data over even a noisy channel with an arbitrarily low error rate, as long as the data rate is less than the channel capacity, defined by the famous formula,

$$C = B\log_2\left(1+\frac{S}{N}\right)$$

Unfortunately, Shannon did not show how one could achieve this capacity, only that it was theoretically possible. Practical systems have only begun to approach the Shannon limit, although some have come remarkably close (the Voyager spacecraft down link

operated at about 50% of the Shannon capacity of its channel). The basic idea behind FEC is to add redundancy to the data being sent so that some number of channel errors can be corrected, up to a limit. The encoded data are referred to as symbols, to distinguish them from the original user data bits. Compare this with the more familiar ARQ scheme, where redundancy (e.g., a CRC) is added to the data only to detect errors; if even one error occurs, the receiver discards the entire block of received data and waits for a retransmission. If this happens often, a significant amount of channel capacity and transmitter energy is wasted.

With FEC, a receiver can use all of the received signal energy; it does not have to throw any of it away. This actually allows one to reduce the transmitter power required for a given user data throughput. This reduction is called the coding gain of the code. The following is a literature review of two types of FEC: block codes and convolutional codes.

Block Codes

The earliest error-correcting codes inspired by Shannon's work were Hamming's block codes (Hamming, 1950). As the name implies, a block code operates on a fixed amount of data that depends on the particular code. Hamming code has been used on several amateur spacecraft to correct radiotion-induced memory errors; the memory word makes a natural code 'block'. This code can generally correct one error in each 8-bit byte of memory. Another popular block code, the Golay code, adds 11 redundant bits to 12 data bits to produce 23 bits for transmission. This is referred to as a block code

(Petit, 1990). This code is able to correct any combination of three or fewer errors in its transmitted block of 23 bits (Lin, 1983).

Reed-Solomon Codes

The most important block code is the Reed-Solomon code, used in compact discs and elsewhere (Lin, 1983) (Pohlmann, 1990). Reed and Solomon actually defined a whole family of codes. Reed-Solomon can operate on multibit symbols (as opposed to binary symbols) and can be easily constructed with relatively large block sizes. These two characteristics together provide an excellent burst-error-correcting capability, which is especially useful when combined or concatenated with other error detecting and correcting codes.

The Reed-Solomon encoder takes a block of digital data and adds extra 'redundant' bits. Errors occur during transmission or storage for a number of reasons (for example noise or interference, etc). The Reed-Solomon decoder processes each block and attempts to correct errors and recover the original data. The number and type of errors that can be corrected depends on the characteristics of the Reed-Solomon code.
This form of coding divides the data stream into blocks that are encoded by adding parity bits that relate only to the information contained in the block. In Reed-Solomon coding, the data is first divided into blocks of k symbols. Each block is then coded by appending a string of (n - k) parity symbols, resulting in an (n,k) Reed-Solomon code. The code's error-correcting capability is given by:

$$t = \frac{(n-k)}{2}$$

where t is the number of symbol errors the code is capable of correcting.

Viterbi Codes

Another class of error correcting code operates on an arbitrary stream of bits, rather than fixed-sized blocks. These are the convolutional codes, sometimes called 'tree' codes. In a more general form, they are known as 'trellis' codes and are found in modern dialup telephone modems (e.g., V.32 and V.32bis). Convolutional codes are easily generated in hardware with a shift register, two or more exclusive-OR 'parity trees' and a multiplexor. These operations are also easily implemented in software. The length of the shift register defines an important parameter called the 'constraint length' K of the code. The larger K is, the better the code will perform. Because a convolutional code operates on a continuous stream, they are specified by the ratio of the input data and output symbol rates. For example, an encoder that produces two encoded symbols for each user data bit is known as a 'rate ½' coder, often abbreviated to 'r = ½'.

Convolutional codes have several interesting properties. They are especially easy to generate and provide excellent performance for a given amount of complexity. They are well suited to varying amounts of data, e.g. variable length data packets. They are also readily adapted to soft decision decoding. This uses symbol quality indications from the modem to aid the decoding process. For example, instead of slicing the demodulator output to binary 0 or 1 with a comparator, one uses an A/D converter to indicate the relative quality of each 0 and 1.

Viterbi decoding is one of two types of decoding algorithms used with convolutional encoding. The other type is sequential decoding. Sequential decoding has the

advantage that it can perform very well with long-constraint-length convolutional codes, but it has a variable decoding time.

2.3 Wireless Data Link Layer Issues

The wireless medium is inherently less reliable which causes frequent temporary outages. In addition, the effective date rate (or bandwidth) may be changing dynamically. The bit-error behavior of the wireless channel is also subject to temporal changes causing variable and dynamic bit-error rate patterns. These impacts, along with other impairments due to multiple access problems associated with potentially a large number of users sharing the same wireless channel could cause significant impairments on the overall performance of existing data link and transport layer protocols which have typically been designed and optimized for the more reliable link characteristics. For example, many of the error recovery mechanisms of the standard transport (e.g. TCP) and data link (e.g. HDLC) layer protocols perform poorly over wireless links. These impairments significantly impacts the applications that require reliable transport. Therefore, a new class of data link protocols are needed to improve the end-to-end reliability of the connections that include a wireless segment. These protocols must also be adaptive and capable of dynamically adjusting to the wireless link behavior and status (link layer QoS management). Similarly, one can envision new signaling interface to incorporate link status information to the end-points of wireless links (for example, wireless access point) which can intelligently adapt to the connection behavior changes and to control the traffic and flow fairness.

2.3.1 Low Power in Wireless Link

One of the function of the wireless link interface is to transport packets received from the network layer protocol entities on the transmit side over the air to the receive side while participating in a suitable medium access control(MAC) protocol and doing processing such as encryption/ decryption, error control, mapping network layer packets to frames going over the air, radio transmission and reception of frame etc. From the perspective of battery power consumption due to the wireless link interface, the MAC function has been the subject of recent attention, with various low power scheduled access MAC protocols being proposed by various researchers, e.g. (Mangione, 1996)(Sivalingam, 1997).

Another wireless link interface function that has a direct and substantial effect on battery power consumption is the error control strategy. Error control refers to schemes used to increase the reliability of a communication link, and is typically done using FEC (where error-correcting codes are used) and ARQ (where error-detecting codes are used in conjunction with positive or negative ACK packets from a receiver to elicit data packet retransmission). Total reliability is an end-to-end issue, and is best left to transport protocols such as TCP. Instead, the goal of error control in wireless link is to merely provide sufficient reliability for end-to-end transport layer quality of service (QoS) requirements to be met. For example, link layer error control in the form of FEC or ARQ with a bounded number of retransmissions can be used to mitigate the adverse effect of wireless link noise and fading on performance of transport protocols (Balakrishnan, 1995) such as TCP. However, the right error control scheme to use in a wireless link - FEC or ARQ or some combination of the two - depends on the

requirements of the service being transported as well as the time varying radio channel.

Error control has traditionally been studied either by channel coding researchers from the perspective of selecting FEC codes to achieve a desired level of radio channel performance, or by protocol researchers to select ARQ protocols with desired throughput characteristics. Although channel coding research literature does characterize FEC codes by their coding gains and equivalent reduction in transmitted radio signal power(as opposed to battery power consumed, of which transmitted RF power is only a component), there is no prior work on what we consider to be the right metric of interest: the amount of battery energy consumed to transmit bits across a wireless link while meeting QoS constraints. Further, the relative trade-off between FEC and ARQ from the battery power consumption perspective under different QoS requirements, channel conditions, packet sizes etc. is totally unaddressed.

2.3.2 Related Work

Error control is an area in which much research has been performed. Books on error control, such as (Lin, 1983), cover the basic FEC and ARQ schemes well. More recently, much work has focussed on error control in wireless channels. Some error control scheme alternatives and their implications have been discussed in previously. Adaptive error-control is mainly used to improve the throughput on a wireless link (Eckhardt, 1996) (Elaoud, 1998). Schuler presents in (Schuler, 1998) some considerations on the optimization and adaptation of FEC and ARQ algorithms with focus on wireless ATM developments. The optimization, with respect to the target bit

error rate and the mapping of the wireless connection quality to the ATM QoS concept, is discussed in detail. Eckhardt et al. argue and demonstrate that protocol-independent link-level local error control can achieve high communication efficiency even in a highly variable error environment, that adaptation is important to achieve this efficiency, and that inter-layer coexistence is achievable (Eckhardt, 1998).

Meanwhile, Zorzi et al. (1997) have been shown that classic ARQ strategies could lead to a considerable waste of energy (due to several reasons: more communication overhead, more transitions, longer communication time, etc.). They propose an adaptive scheme, which slows down the transmission rate when the channel is impaired. This scheme saves energy without a significant loss in throughput.

Classic ARQ protocols overcome errors by re-transmitting the erroneously received packet, regardless of the state of the channel. Although in this way these retransmission schemes maximize performance – as soon as the channel is good again, packets are received with minimal delay – the consequence is that they expend energy (Zorzi, 1997). When the tolerable delay is large enough, ARQ outperforms error correction mechanisms, since the residual error probability tends to zero in ARQ with a much better energy efficiency than error correction methods (Zorzi, 1998).

Most relevant work that relates the error coding strategy to energy consumption is by Zorzi and Lettieri. Zorzi describes in (Zorzi, 1997) and (Zorzi, 1998) an adaptive probing ARQ strategy that slows down the transmission rate when the channel is impaired without a significant loss in throughput. A modified scheme is also analyzed, which yields slightly better performance, but requires some additional

complexity. Lettieri et al. (1998a) describes how energy efficiency in the wireless data link can be enhanced via adaptive frame length control in concert with adaptive error control based on hybrid FEC and ARQ. The length and error coding of the frame going over the air and the retransmission protocol are selected for each application stream based on QoS requirements and continually adapted as a function of varying radio channel conditions.

Spragins et al. (1991) discuss optimizing frame size for a variety of factors, including number of hops in a multihop link, total memory consumed for buffering, and error in the channel. They discuss calculating this optimal in the presence of several ARQ schemes based on formulae for packet error rates given bit error rates. This optimization is to be done only once at the time the network is designed, however, and is fixed across the entire network.

Meanwhile, Hara et al. (1996) presented a simulation based study of throughput improvements with a stop-and-wait ARQ protocol with adaptive packet length chosen according to channel condition estimate. They, however, do not address crucial issues such as the adaptation mechanism itself and how packet length adaptation is incorporated in the protocol stack, how channel measurements are done, the impact on energy efficiency, relationship with error control, and overall implementation architecture.

Finally, also of considerable interest and relevance is the ongoing work on power management in MAC protocols (Mangione, 1996). We believe that in the case of multiple packet streams with potentially different QoS requirements sharing a time

varying wireless link, link layer control really ought to be treated in close conjunction with MAC (e.g. (Gang, 1995)), and packet multiplexing/scheduling, with a joint design from a low power perspective.

2.4 Conclusion

In this chapter, the advanced technology in mobile applications and low power design techniques for wireless devices were illustrated in order to support low power error control strategies are discussed. We identified, power efficient consumption as the focus of this dissertation. Several low power design strategies for wireless devices have been discussed, most of which focused on designing algorithms to minimizing power in hardware components which consume a substantial amount of power.

The error control strategy proposed in this dissertation takes a different approach, and is based on the concept of optimizing battery energy efficiently. It is, to some extent, similar to the strategy proposed by Lettieri et al. (1997), but tries to make it more efficient, and it also takes into consideration several additional issues, such as minimum delay, maximum throughput and energy efficient which are discussed in details in the next chapter.

In summary, while studies have shown that it is possible to reduce power consumption by applying the low power design techniques, which includes optimizing the operating system, it is possible to further reduce power consumption by minimizing the amount of battery energy taken to transport bits across the link. In the next chapter, adaptive error control strategies and the proposed algorithms are discussed.