CONNECTION ESTABLISHMENT IN THE BLUETOOTH SYSTEM

BY IVANA MARIC

A thesis submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Master of Science

Graduate Program in Electrical and Computer Engineering

Written under the direction of

Professor Roy D. Yates

and approved by

____

____

____

____

New Brunswick, New Jersey

October, 2000

ABSTRACT OF THE THESIS

Connection Establishment in the Bluetooth System

by Ivana Maric

Thesis Director: Professor Roy D. Yates

Any communication between hosts in the Bluetooth system is preceded by a connection establishment procedure comprising two steps. The first step, referred to as Inquiry, enables hosts to learn about other hosts in their range. Inquiry can be omitted when not needed. The second step in which a host synchronizes and starts communication with the intended host is called Page. The complexity of the connection establishment procedure stems from the fact that hosts use different hopping sequences before a connection is established. The set-up delay is also affected by the requirement of low-power consumption and the possible connections that hosts may already have. The delay introduced by the procedure may impose constraints on applications such as ad hoc networking and LAN access.

We investigate the connection establishment when different paging modes are used. Analytical and simulation results show the range of values for the average page duration between two hosts. In a channel with packet errors, we derive analytical results for the average page duration as a function of the time interval a host allots to scanning for

page messages. The most efficient connection establishment procedure takes up to 20 ms and comes at the expense of high power consumption since a host needs to scan for page messages continuously. In a second paging mode, a host wakes up to scan for page messages for a short time interval resulting in significantly higher delays. In such a situation, the average page duration shows a linear dependence on the time a host spends in *page scan* in an error-free environment. In the presence of errors, such a behavior is observed only after the scan interval reaches a certain value depending on the channel error probability. For smaller values of the scan interval, the duration changes more rapidly.

Simulation results on the connection establishment procedure that consists of both inquiry and page illustrate the additional delay the inquiry procedure adds to the connection establishment.

Acknowledgements

I am grateful to Prof. Roy Yates for his invaluable guidance throughout my thesis project. I would like to express my thanks to my committee members, Prof. Narayan Mandayam and Prof. Christopher Rose. I would especially like to thank Prof. David Goodman for being the external examiner in the committe as well as for his continued support throughout my graduate studies, his advice and interest in my research. I thank to everyone at WINLAB for a wonderful atmosphere to work in. A special word of thanks to Ivan Seskar for all his encouragement and help and to David Pandian for his suggestions during proofreading.

Table of Contents

List of Figures

Chapter 1

Introduction

Bluetooth [1] is a short-range radio link intended for cable replacement between electronic devices, such as computers, cell phones, PDAs and printers. The technology is an open specification for wireless communication of data and voice.

Apart from being just a cable replacement, Bluetooth enables Bluetooth-equipped hosts to form an ad hoc network. A large number of ad hoc Bluetooth connections will coexist in the same area without mutual coordination. A large amount of overlap between ad hoc links distinguishes such an environment from a conventional ad hoc environment and is referred to as scatter ad hoc environment [2]. Routing protocols that have been devised for routing in conventional ad hoc networks [3], [4], [5] are feasible solutions for the routing in the Bluetooth scatternet; however, it is anticipated [6] that scatternets will differ from classical ad hoc networks in terms of the applications, traffic characteristics, mobility patterns and scaling requirements. Further, scatternets and wireless LANs, for example, differ in terms of their communication capabilities; in contrast to the wireless LAN, it is not enough for two bluetooth nodes to be within each other's transmission range in order to hear each other, as will be described later. Therefore, more efficient solutions require routing algorithms that are well suited for a Bluetooth system.

Any communication between hosts using Bluetooth is preceded by a connection establishment phase comprising two steps. The first step, referred to as Inquiry, enables hosts to learn about other hosts within their range. Inquiry can be omitted when not needed. The second step is Page where a host initiates the communication with another host and enables the synchronization between them. For a successful connection establishment, hosts have to dedicate time to scan for inquiry and page messages.

There has been a recently proposed algorithm [6] for routing in a Bluetooth system. The objective was to design a simple, bandwidth efficient protocol. The algorithm is based on source routing and hence, it is not scalable. The basic assumption in the analysis was that the scatternet had been established previously and a path could be found between any two hosts. Therefore, no new connections needed to be set up while establishing the routes. The changing topology of an ad hoc network will, however, need new routes to be established. Further, in some situations it might be beneficial in terms of throughput to establish a new, direct connection between hosts rather than communicating via multiple hops. Establishment of new connections between hosts during the route discovery would have an impact on routing performance by increasing the overall route discovery delay and influencing the system throughput.

In this work, we analyze the influence of Bluetooth parameters (primarily timeouts) on the time required for a connection establishment between two Bluetooth hosts within range of each other. Hosts may be meeting for the first time or may have had a previous connection that was terminated sometime in the past.

Our results illustrate the change in the average connection establishment time as the parameters are varied and hence, suggest how the parameters should be set. We present the range of average set-up duration values for different scenarios. The motivation behind our work was to characterize Bluetooth connection establishment in order to be able to devise routing algorithms that are well suited for such an environment.

Morris considered fixing timeout intervals for a protocol that resembles the page protocol in the Bluetooth system by considering the mean delay and mean number of retransmissions [7].

The literature on Bluetooth is scarce at the moment. However, Bluetooth has gained significant attention in the past two years since the first version of the Bluetooth Specification was released. One of the earliest Bluetooth overviews was given in [8]. A more detailed Bluetooth overview is presented by the same author in [2]. Since Bluetooth operates in the unlicensed ISM band along with other systems, a major concern is the amount of interference that these systems introduce to each other and the impact on their performance. The effect of interference caused by a Bluetooth system on IEEE 802.11 wireless LAN is studied in [9] while the opposite case is studied in [10]. The simulation results presented in [10] showed that, in an office environment, a Bluetooth voice link was disturbed in less than 1% of the time when the distance between Bluetooth devices was less than 2 m but this increased to 8% for distances between 2 m and 10 m. In the case of a Bluetooth data link, for distances of 10 m, a 10% throughput reduction was observed with probability of 24%.

Another short range wireless technology that was intended for cable replacement is Infrared Data Association (IrDA). Many of the applications defined for IrDA devices are also defined for Bluetooth [11]. The higher layer protocol Object Exchange Protocol (OBEX), developed and used by IrDA may also be used by Bluetooth usage models since it is a part of Bluetooth protocol stack [12]. Hence, it is possible for an application to run over both IrDA and Bluetooth. The better choice between the two technologies depends on a particular application. A comparison is presented in [11].

Scheduling policies for Bluetooth were proposed in [13] to increase the achievable throughput. The policies were based on the size of head-of-the line packet of the hosts.

This thesis is organized as follows. We briefly describe Bluetooth system characteristics in Chapter 2. The procedure to start a new connection with all the modes that a Bluetooth unit can operate in (i.e *inquiry*, *inquiry scan*, *page*, *page scan*, *inquiry*, *slave* and *master response*) are described in the same chapter. Analysis of the average page duration for two different paging modes is presented in Chapter 3. Simulation results are given in Chapter 4. Chapter 5 presents the summary and future work.

Chapter 2

Bluetooth System Characteristics

Bluetooth operates in the unlicensed ISM band at 2.4GHz. Frequency hopping is used with 1MHz wide RF channels. In the US and most of Europe, there are 79 RF channels available. In Japan, Spain and France, only 23 RF channels are available. Gaussian, binary FSK modulation is employed. The dwell time in each RF channel is 625μs slots. From slot to slot, a terminal employs a pseudorandom hopping sequence. Thus the nominal hop rate is 1600 hops/s. A packet nominally covers one slot but can be extended to cover three or five slots in which case, the used frequency stays the same for the duration of the packet rather than changed on per-slot basis. A Time-Division Duplex (TDD) scheme is used for full-duplex transmission.

Bluetooth can support an asynchronous data channel (ACL link), up to three synchronous voice channels (SCO links) or a channel which simultaneously supports asynchronous data and synchronous voice. Different packet types involving different overhead have been defined by the Specification and the rate on the ACL link depends on the type used. Each voice channel supports a 64 kb/s synchronous link. The asynchronous channel can support a maximal 721 kb/s asymmetric link (and 57.6 kb/s in other direction), or a 432.6 kb/s symmetric link. For packets on the ACL link, an automatic repeat request (ARQ) scheme [14] is used. Packets on the SCO link are never retransmitted.

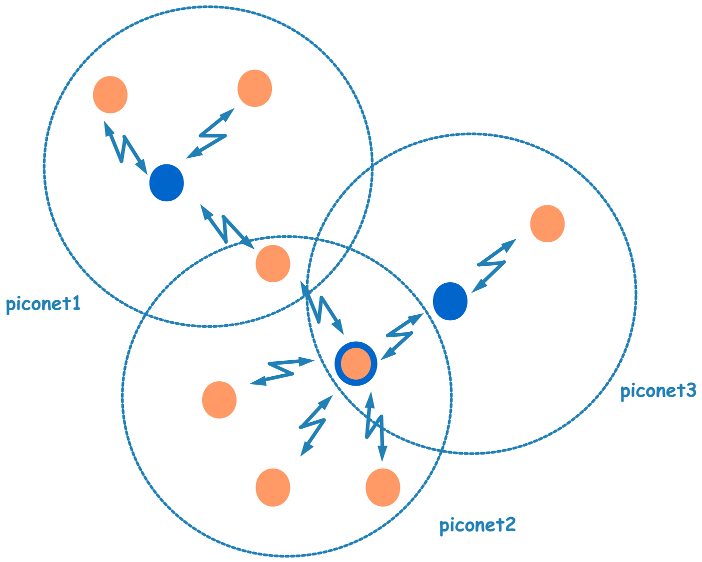

Figure 2.1: Scatternet. Masters are shown in blue and slaves are shown in orange. The host shown in both colors has a master role in piconet 2 and a slave role in piconet 3.

The Bluetooth specification [1] introduces the *piconet* - a collection of units connected via Bluetooth links in a master - slave configuration. Each piconet has one *master* whose Bluetooth device address defines the piconet channel *(channel hopping sequence)* and who synchronizes and coordinates packet transmission in that piconet. All other units in the piconet are referred to as *slaves*. They are allowed to exchange packets only with their master. One unit can participate in more than one piconet, but can have only one master role. A group of piconets with overlapping coverage areas and with connections between them is called a *scatternet*. A scatternet is shown in Figure 2.1. As the number of piconets in the same area increases, each Bluetooth link suffers more interference due to occasional collisions between packets from different piconets. The benefits of using frequency hopping in such an environment as well as its impact on the overall design of a packet radio network is presented in [15].

The Bluetooth specification defines two major states (modes of operation) for a unit:

- *Standby*
- *Connection*

Each unit that communicates with other Bluetooth units is a member of a piconet and is in the *connection* state. Establishment of a piconet is initiated by any unit that needs to communicate with other ones and after the set-up procedure has successfully finished and a piconet has been established, that unit becomes a master of that piconet.

Each Bluetooth host is assigned a unique 48-bit Bluetooth device address (BD_ADDR) derived from the IEEE 802 standard. Also, each host has an internal system clock derived from a free running native clock that ticks every $312.5\mu s$. The clock has a cycle of about a day. If the clock is implemented with a counter, a 28-bit counter is required that wraps around at $2^{28} - 1$. Prior to establishing a connection, a host uses BD_ADDR to determine its frequency hopping sequence and uses the current value of its clock to determine the phase in the sequence for each slot/half a slot. Therefore, in the set-up procedure, hosts are hopping independently and they have to meet on a same frequency to be able to transmit and receive data necessary for synchronization and forming a piconet (Bluetooth device addresses and native clock values). The set-up procedure consists of two steps: Inquiry and Page. There are seven substates that are used in these procedures. For the future master, possible states are *inquiry*, *page* and *master page response*. Future slave may use: *inquiry scan*, *inquiry response*, *page scan* (future slave) and *slave page response*. Figure 2.2 shows all the states with transitions between

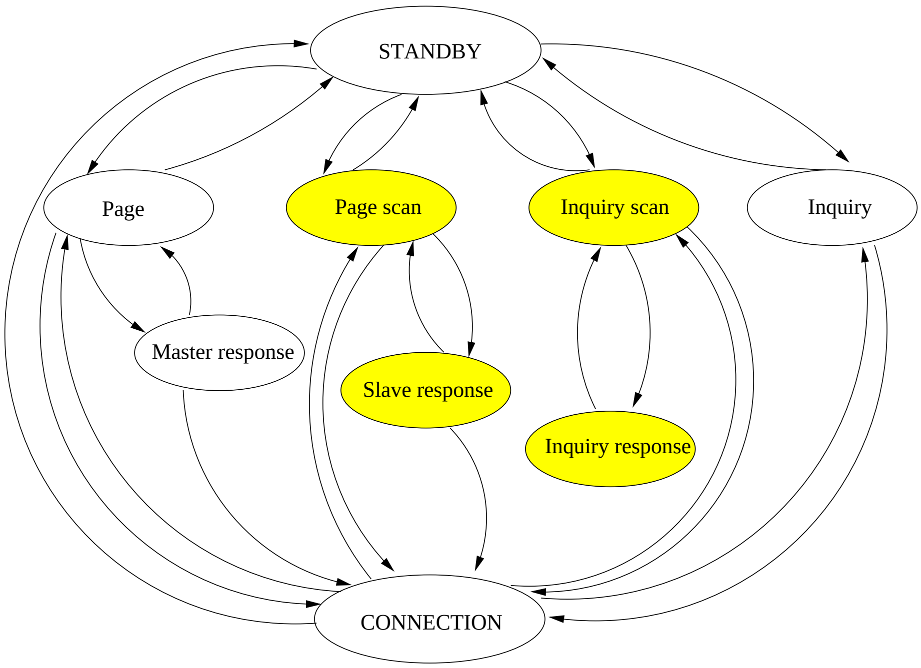

Figure 2.2: Bluetooth unit states. Two major states are *Standby* and *Connection*. The additional seven states are used in the connection establishment procedure to add new slaves to a piconet. Shaded states represent states used by a future slave. The rest of the states are used by a future master.

them as given in [1].

In the next section, we present the details of the Bluetooth connection establishment procedure through description of the states and the transitions shown in Figure 2.2.

2.1 Overview of States

2.1.1 Standby

Standby is the default state of a Bluetooth unit. It is a low-power mode in which only the native clock is running. A unit might leave *standby* to go to *inquiry*, *page*, *inquiry scan* or *page scan*.

2.1.2 Inquiry

The inquiry procedure is executed by a unit (future master) to discover other units in range. After a successful inquiry, a master will have the Bluetooth device addresses (BD_ADDR) and the value of the native clocks of units that responded, in addition to other information. The procedure is as follows: In each Tx slot, a future master sequentially transmits two inquiry messages (broadcast ID packets) on two different frequencies, therefore increasing the hopping rate to 3200 hops/s. In each Rx slot, it listens for a response sequentially on two corresponding frequencies. As mentioned before, it continuously alternates between Tx and Rx slots. A response is a Frequency Hopping Synchronization (FHS) packet defined by the Specification. The FHS packet contains the Bluetooth device addresses (BD_ADDR) and native clock of the responding unit, among other information sent in the packet.

For inquiry messages, an *inquiry sequence* containing 32 unique, pre-specified frequencies distributed uniformly over 79 MHz is used. To select a frequency from the inquiry sequence, a future master uses an *inquiry access code (IAC)* given by the Specification, instead of its BD_ADDR. The sequence is split into two trains A and B, each with 16 frequencies. Hence, one train lasts for 8 Tx slots and there are 8 Rx slots interleaved giving the overall duration of a train to be 16 slots * 625 μs = 10 ms. Each train has to be used repeatedly for at least $N_{\text{inquiry}} = 256$ times before switching to the other one, that is, for 2.56 s. The procedure is shown in Figure 2.3. Exact constraints on N_{inquiry} are given in the Table 2.3. For inquiry message responses sent in the Rx slots, the *inquiry response sequence* is used, covering 32 frequencies that are in one-to-one correspondence to the frequencies in the inquiry sequence.

No exact requirements are given about when and how often *inquiry* should be done.

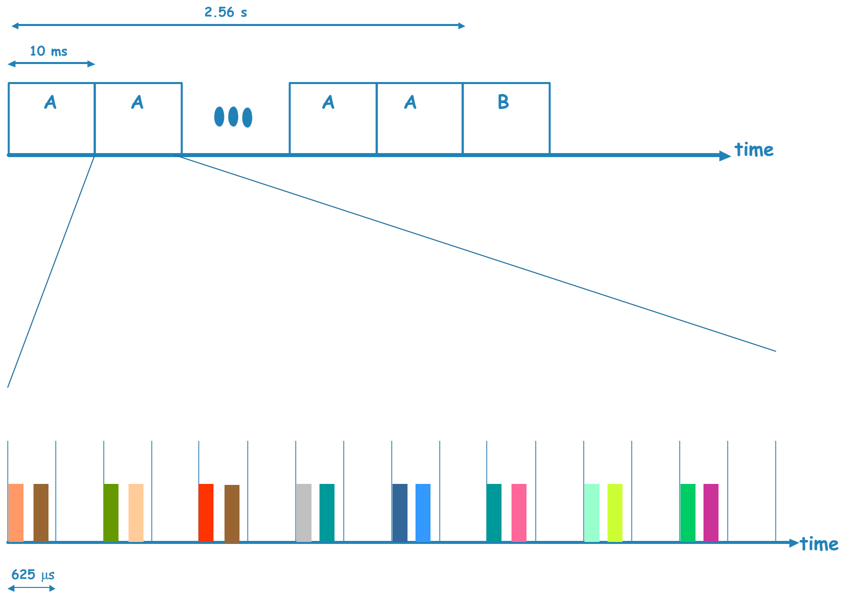

Figure 2.3: Frequency selection during *inquiry*. An inquiry sequence is split into two trains A and B, each with 16 frequencies. Each train has to be used repeatedly for at least 2.56 s before switching to the other one. Different colors show different hopping frequencies from a train A used to send inquiry messages during Tx slots.

Both periodic and one-time *inquiry* can be performed. In our simulation, *inquiry* is initiated when there is data to be sent to a unit with an unknown address, for example, a public printer. A unit leaves *inquiry* based on two criteria:

1. After the timer *inquiryTO* expires.

2. After a *required number of responses* are collected.

2.1.3 Inquiry Scan

Inquiry scan is the substate that enables a unit to be discovered by other units. In this substate, a unit listens for inquiry messages at a single frequency chosen from the inquiry sequence and based on the unit's native clock at a start of the *inquiry scan*.

The window between two consecutive *inquiry scans* ($T_{\text{inquiryscan}}$) is constrained by the Bluetooth specification to be no greater than 2.56 s. This is exactly the same time interval as the time it takes for a master to do inquiry using the frequencies from one train.

In our simulation, *inquiry scan* is finished when:

1. The timer T_{winqscan} expires. If the unit is in *inquiry scan* at this time, it switches to *page scan*.

2. An inquiry message is received. The unit enters the *inquiry response* state.

2.1.4 Inquiry Response

A future slave terminal enters the *inquiry response* state to send a response to an inquiry message that has been received during the *inquiry scan*. In order to avoid collisions of packets sent as the response to the same inquiry message and on the same frequency, units are not allowed to respond immediately after receiving the inquiry message. Instead, a slave generates a random number W between $[0, 1023]$ and returns to its previous state (the one prior to entering the *inquiry scan*), for W time slots. Only after W slots, does a slave return to the *inquiry response* state. If an inquiry message is received, a slave will send a response and go back to *inquiry scan*. If an inquiry message is not received for $inqrespTO$ time, the slave will return to *standby* or *connection* depending on what was the state the slave was in before starting the *inquiry scan*. The response is a FHS packet containing the unit BD_ADDR and value of the unit's native clock. It is sent on a frequency from the inquiry response sequence corresponding to the frequency on which an inquiry message was received.

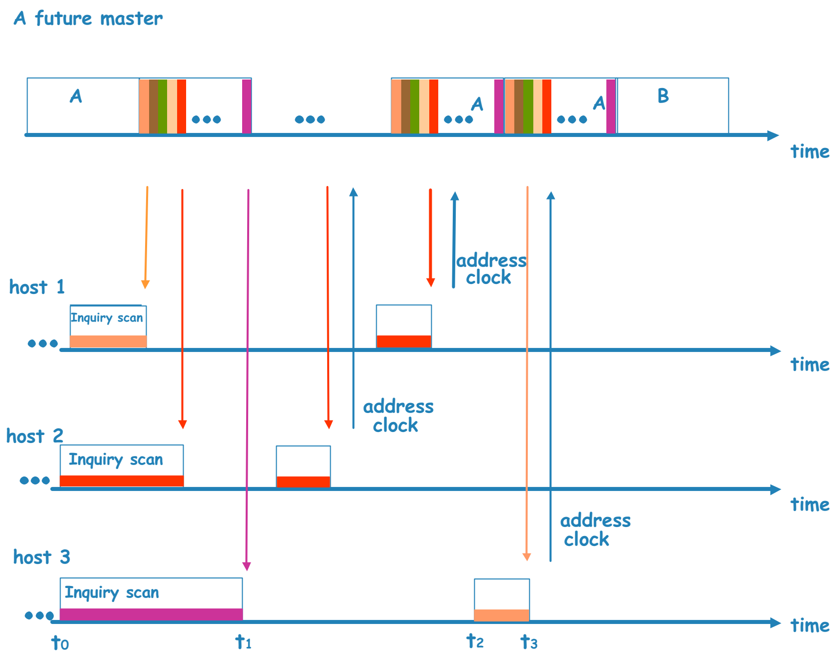

Figure 2.4: Inquiry procedure. For a future master, colors show frequencies used to send inquiry messages. For other hosts, colors show current scan frequencies. A host performing an *inquiry scan* receives an inquiry message sent on its current scan frequency. Host 3, for example, receives an inquiry message at $t = t_1$. The reception of an inquiry message is followed by a random backoff. For host 3, a backoff duration is $t_2 - t_1$ and after that, an *inquiry scan* starts again. After an inquiry message is received for the second time, a host responds by sending its address and the value of its clock. For host 3, an inquiry message is sent on its scan frequency at $t = t_3$.

2.1.5 Page

The *page* state is used by a future master to set up a connection with a particular unit. After a successful *page*, a connection is established between a master and the paged slave. A paging scheme can be mandatory or optional. A mandatory paging scheme has to be supported by each Bluetooth host. In addition, each host can support some of the optional paging schemes. At the moment, only one out of three optional paging schemes has been defined. In the following we describe mandatory paging.

The procedure is essentially the same as the *inquiry procedure*; however, instead of sending broadcast ID packets, a master sends ID packets to the slave to which he wants to connect. The ID packet is 68 bits long and uniquely identifies the future slave since its content is derived from the slave's address.

A Page hopping sequence containing 32 frequencies assigned to a paged unit is used for paging. To obtain this sequence, a future master uses the clock and the address of the unit he wants to page. There is a corresponding *page response sequence*. Unlike the case of *inquiry* where there is a dedicated inquiry sequence, a page sequence is different for each unit and is determined by the address of that unit (BD_ADDR). As in *inquiry*, 32 available frequencies are divided into trains A and B, each of them being probed for N_{page} times before switching to the other one. N_{page} can take different values for each page and is determined by the paging unit as described in Section 4.2 and given in the Table 2.4. A unit will leave *page* if either:

1. Timer $pageTO$ expires and no response has been received, or

2. A response has been received. In this case, the paging unit enters *master response* state.

2.1.6 Page Scan

In *page scan*, a unit listens for the ID packets that were unicast to it. Similarly to *inquiry scan*, the terminal listens on one frequency chosen from its page hopping sequence and based on its native clock at the start of the *page scan*.. When an ID packet is received, the host enters the *slave response* state. If no ID packets are received for time $T_{\text{wpagescan}}$, the *inquiry scan* substate is entered.

The interval between the start of two consecutive *page scans* is T_{pagescan}. The intervals

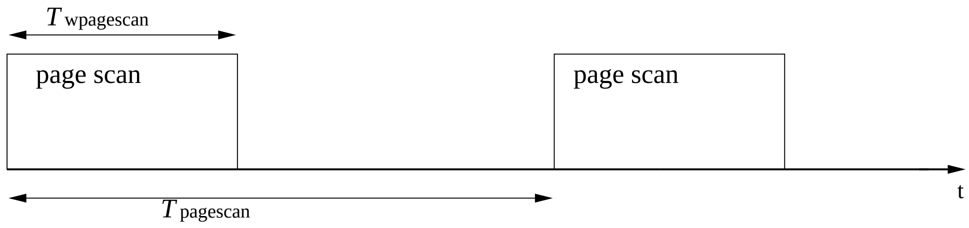

Figure 2.5: Time intervals $T_{\text{wpagescan}}$ and T_{pagescan}. $T_{\text{wpagescan}}$ is a *page scan* window. T_{pagescan} represents the time interval between two consecutive *page scans*.

$T_{\text{wpagescan}}$ and T_{pagescan} are shown in Figure 2.5.

2.1.7 Master Response

The *master response* state is entered by a unit after a successful *page*. The unit sends FHS packets to the previously paged unit until an acknowledgment is received or until the timer $pagerespTO$ expires. The page hopping sequence and page response sequence are still used.

After an acknowledgment has been received, the master enters the *connection* state. From now on, the channel hopping sequence will be used. If the timer expires before an acknowledgment has been received, the unit returns to the *page* state. If the FHS packet and an acknowledgment for it are not lost, the unit will stay in this state only for three slots (1.875ms).

2.1.8 Slave Response

A unit enters the *slave response* state from *page scan* when a page (unicast ID packet) has been received. The unit sends a response (same ID packet) to the page until a response (FHS packet) is received or until timer $pagerespTO$ expires. In this process, the page hopping sequence and page response sequence are used.

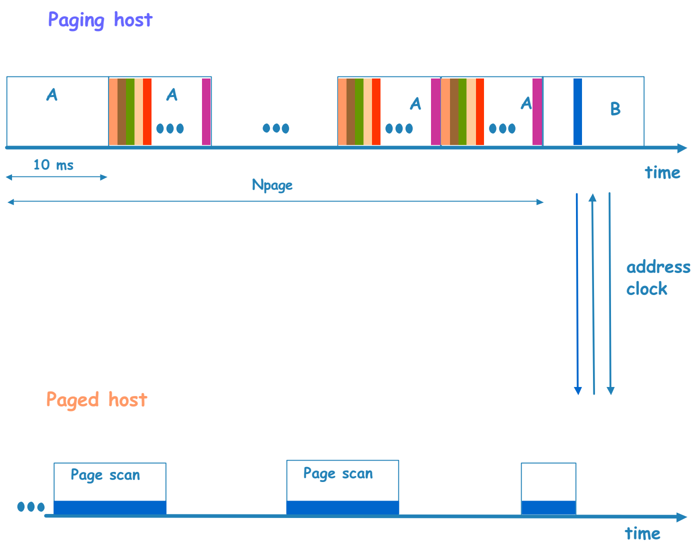

Figure 2.6: Page procedure. For a paging host, colors show different hopping frequencies used to send page messages. For a paged hosts, color shows current scan frequency. A host performing a *page scan* will receive a page message sent on the its current scan frequency. In the situation shown, the scan frequency is one of the frequencies from paging host's train B which delays a successful page for time interval in which a train A is being used. After the reception of the page message has been acknowledged, the paging host will send its address and the value of its clock to the paged host.

After an acknowledgment has been received from the future master, the unit sends an acknowledgment (the same ID packet again) to the future master, switches to the channel hopping sequence and enters the *connection* state. If no FHS packets are received until the timer expires, the unit returns to *page scan*.

The *page*, the *page scan* and the *page response* are shown in Figure 2.6.

The connection establishment procedure is summarized in Figure 2.7. The flow of messages during the connection establishment procedure is summarized in Figure 2.8.

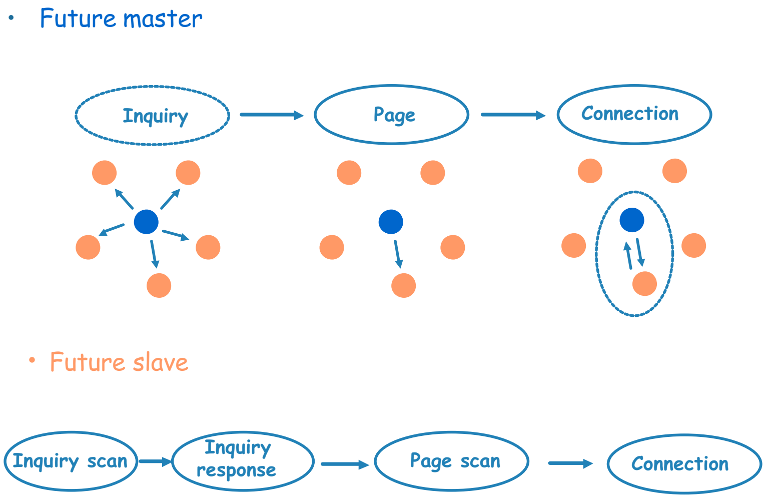

Figure 2.7: Connection establishment. A host initiating a connection establishment is a future master of a piconet which is going to be formed and is shown in blue. Other hosts are shown in orange. During the *inquiry*, a future master broadcasts inquiry messages that will be heard by hosts performing an *inquiry scan*. During the *page*, a future master unicasts page messages to its future slave. After a successful *page*, a piconet is established.

2.1.9 Connection

A unit in a *connection* state is a part of a piconet. To transmit or receive data, the channel hopping sequence, as determined by the master's address and clock, is used. Four possible modes of the *connection* state have been defined:

Active mode When in active mode, a unit actively participates in packet exchange in a piconet. A master schedules the transmission based on traffic demands to and from slaves. A master uses polling [14] to schedule transmission from different slaves. An active slave listens on the channel during Rx slots and sends packets in Tx slots assigned to it by the future master. There can be up to seven active

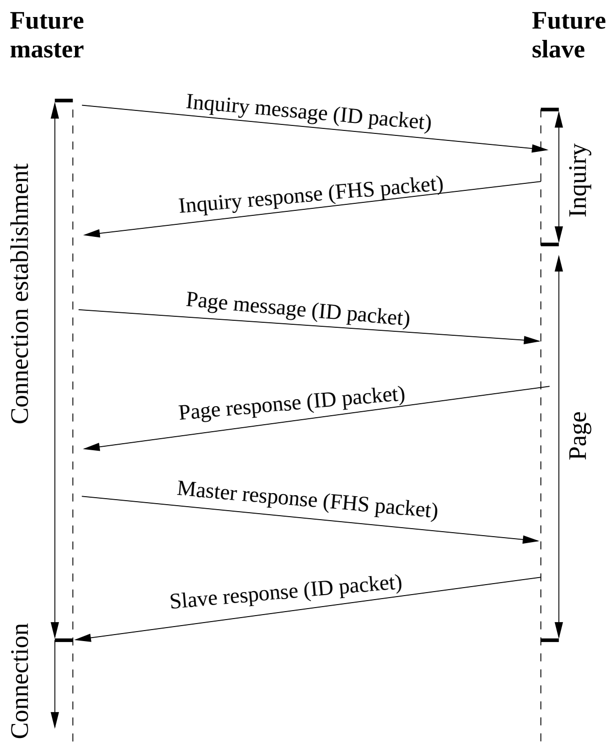

Figure 2.8: Flow of messages in a connection establishment procedure. Arrows show the direction of messages exchanged between a future master and a future slave.

slaves in a piconet.

Sniff mode This is a low-power mode in which a slaves's duty-cycle has been reduced. When a slave is in *sniff* mode, its master can only transmit to that slave in specified time slots (called *sniff* slots) that are regularly spaced. A slave can enter the *sniff* mode upon request made by it or its master after the sniff mode parameters are negotiated or it can be forced to *sniff* mode by the master.

Hold mode While in hold mode, a slave doesn't support packet exchange, but it stays a member of a piconet. Prior to entering the hold mode, master and slave agree on the time duration the slave stays in the hold mode.

Park mode In the park mode a slave doesn't participate in the packet exchange, but stays synchronized and wakes up occasionally to listen to the channel. In addition to using it for low power consumption, the park mode is used to connect more than seven slaves to a single master.

The Specification defines two commands for leaving the *connection* state:

- *detach*, where the link has been disconnected in the normal way when no longer needed and
- *reset*, to handle a connection breakdown.

To supervise a link, a timer $T_{supervision}$ has been introduced. The timer is reset when a correct packet is received. If at any time the timer expires, the *connection* is reset.

2.1.10 Baseband Timers and Parameters

The parameters of the connection establishment procedure are summarized in Table 2.1. The constraints on those parameters are given in Tables 2.2 and 2.3.

We notice that, for the ranges of parameter values given by the Specification, a connection set-up might still fail for the following reasons:

- An inquiry times out (1.28 s) before a slave starts an *inquiry scan* (2.56 s window). If $inquiryTO$ is at least 2.56 s, this problem would not happen.
- If during one-time inquiry, a master only probes one train. If a slave is tuned to a frequency from the other train, the inquiry will fail. This happens only when $inquiryTO \leq 2.56$ s, since it takes 2.56 s to probe one train. To avoid this, the constraint should be imposed that $inquiryTO > 2.56$ s.

Parameter	Definition/Purpose
inquiryTO	maximum number of slots the inquiry state will last
required number of responses	number of received responses by a future master after which it leaves the inquiry state
$T_{\text{inquiryscan}}$	time between beginnings of two consecutive *inquiry scans*
T_{winqscan}	maximum number of slots the *inquiry scan* state can last
inqrespTO	maximum number of slots the *inquiry scan* state can last after the RAND wait period
N_{inquiry}	number of times one train (A or B) is used repeatedly to send inquiry messages before using the other train
pageTO	maximum number of slots the page state can last
T_{pagescan}	time between beginnings of two consecutive *page scans*
$T_{\text{wpagescan}}$	maximum number of slots the *page scan* state can last
pagerespTO	if a slave: number of slots it waits for the master's FHS packet in the *slave response* state if a master: number of slots it waits for the FHS packet acknowledgment in the *master response* state
N_{page}	number of times one train (A or B) is used repeatedly to send page messages before using the other train

Table 2.1: Baseband Timers and Parameters.

- A page times out before the slave starts a *page scan*. To avoid this, the *pageTO* should be constrained to satisfy $pageTO > (T_{\text{pagescan}} - T_{\text{wpagescan}})$.

2.1.11 Scenarios

Different scenarios are possible for paging depending on the paging scheme, the paging mode and the number of connections of a master and a slave. Some possible cases are listed below.

- A paging scheme can be mandatory or optional. We assume mandatory paging.
- A slave can already have:

  1. no SCO links requiring $T_{\text{wpagescan}} \geq 11.25\text{m}s$ (18 slots)
  2. one SCO link requiring $T_{\text{wpagescan}} \geq 22.5\text{m}s$ (36 slots)
  3. two SCO links requiring $T_{\text{wpagescan}} \geq 33.75\text{m}s$ (54 slots)

Parameter	Range given by the Specification	Values used in simulation
inquiryTO	$1.28 - 61.44s$ in steps of 1.28 s	$1.28, 2.56, 3.84s$
required number of responses	$1 - 255$	1
$T_{\text{inquiryscan}}$	≤ 2.56 s	$\leq 2.56s$
T_{winqscan}	depends on number of existing SCO links	$[0.005, 0.5s]$
inqrespTO	no constraints default value: $0.08s(128slots)$	0.08s (128 slots)
N_{inquiry}	depends on number of existing SCO links	256
pageTO	$0.625 - 40.9s$	$1.28, 40.9s$
$T_{\text{wpagescan}}$	depends on number of existing SCO links	$[0.01, 0.5s]$
T_{pagescan}	$\leq 2.56s$	$\leq 1.28s$
pagerespTO	no constraints default value: 5 ms(8 slots)	$5ms(8slots)$
N_{page}	depends on paging mode and number of SCO links	$1, 128$

Table 2.2: Constraints on Baseband Timers and Parameters.

	no SCO links	one link	two links
N_{inquiry}	≥ 256	≥ 512	≥ 768

Table 2.3: Constraints on N_{inquiry}.

- Paging mode depends on slave's T_{pagescan} and is defined by the Specification to be:

  1. R0 if $T_{\text{pagescan}} = T_{\text{wpagescan}}$ (continuous scan)
  2. R1 if $T_{\text{pagescan}} \leq 1.28s$
  3. R2 if $T_{\text{pagescan}} \leq 2.56s$

- A master starting a new connection can already have:

  1. no SCO links
  2. one SCO link
  3. two SCO links

	no SCO link	one SCO link	two SCO links
R_0	$N_{\text{page}} \geq 1$	$N_{\text{page}} \geq 2$	$N_{\text{page}} \geq 3$
R_1	$N_{\text{page}} \geq 128$	$N_{\text{page}} \geq 256$	$N_{\text{page}} \geq 384$
R_2	$N_{\text{page}} \geq 256$	$N_{\text{page}} \geq 512$	$N_{\text{page}} \geq 768$

Table 2.4: Constraints on N_{page}.

Based on its paging mode and a number of existing connections, a future master determines N_{page}, that is, the number of times one train (A or B) is used repeatedly to send page messages to a future slave as given in Table 2.4.

Chapter 3

Average Page Duration

The procedure to start a new connection between two Bluetooth hosts may require that either both inquiry and page are performed or, when the address of the future slave is known to a future master, page only. Hence the set-up time may or may not include the time needed for inquiry.

In this chapter, we present an analysis of connection establishment that consists of page only. We restrict ourselves to a scenario where a paging unit has no other SCO links established. We assume that the pageTO timer is large enough not to expire during the set up procedure. The round-trip propagation delay is not considered as it is negligible. We consider the case where a paging unit has no knowledge of the clock of a unit it tries to page. Different paging modes lead themselves to different analyses and are presented separately.

In Section 3.1 we analyze the case where a paged host performs a continuous *page scan* and therefore a paging mode R0 is used by a paging host. In Section 3.2 we present the analysis on R1 paging mode.

3.1 Paging in R0 Mode

In this section we analyze the page duration when R0 mode is used for paging. A future master uses R0 mode when a future slave is performing continuous *page scan*.

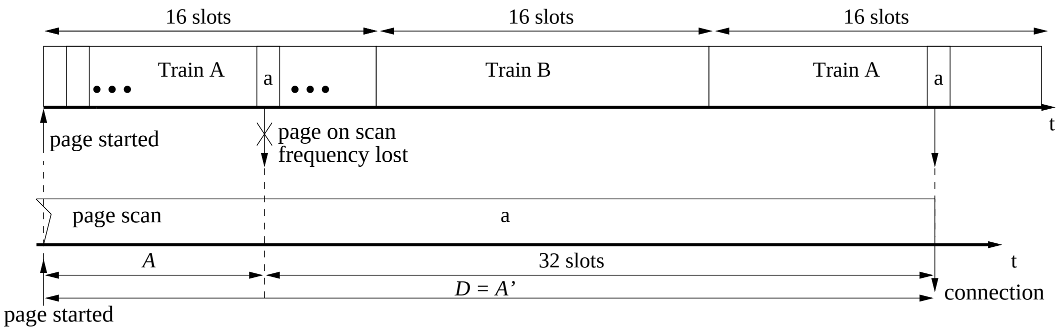

Figure 3.1: Paging in R0 Mode. A future slave is performing a continuous *page scan*. In the figure, a denotes the slave's scan frequency. Paging host will send page messages on 32 page frequencies in 32 slots. After A slots, the page message is sent on slave's scan frequency. When that page message is lost, the same frequency will be paged after 32 slots. In this scenario, $D = A'$.

For this case $N_{\text{page}} \geq 1$ as shown in Table 2.4. In our analysis, we assume $N_{\text{page}} = 1$ which results in the shortest page procedure on average. The reason is that $N_{\text{page}} = 1$ implies that a paging unit needs 32 slots to send a page on all 32 page frequencies. In the general case where $N_{\text{page}} \geq 1$, it takes $16(N_{\text{page}} + 1)$ slots to send a page on all 32 frequencies which increases the duration of the establishment procedure when a paged unit is scanning the train which is not currently used for paging.

To calculate the average page duration we define random variables,

- D - *page duration*: The time from the instant a master starts a page until the connection is established.

- A - The time interval from the instant at which a paging host started page and a paged host is simultaneously in a *page scan* until a page message is sent on the slave's scan frequency.

- A' - The time interval from the instant at which a paging host is in a *page* and at the same time a paged host is in a *page scan* until the page is successful.

In the case where the channel is error-free, $A' = A$. Situation is shown in Figure 3.1.

Since a paging unit has no knowledge of the clock of a unit it is trying to page, any of the 32 frequencies can be a slave's scan frequency and we can model the random variable A with the discrete uniform PMF:

$$P_A(a) = \begin{cases} 1/32 & a = 1, 2, \ldots, 32 \\ 0 & otherwise \end{cases} \tag{3.1}$$

This implies,

$$E[A] = 16.5 \text{ slots} \tag{3.2}$$

In the case when a master knows the clock of a host it wants to page, it can determine the frequency $f(k)$ on which that host is scanning. However, instead of sending a page on $f(k)$, the procedure given by the Specification is such that master starts and continues paging at $f(k-8), f(k-7), \ldots, f(k),, \ldots$ Once the page message is received by the slave, the Specification requires that three additional packets be exchanged in a *slave* and *master response* states before units can go to a *connection*.

Taking the next step, we assume errors in the channel can occur and let L equal the number of page packets sent on slave's scan frequency or page responses from the slave that have been lost. Let p be probability that an arbitrary packet is lost.

We model L as a geometric random variable with PMF,

$$P_L(l) = \begin{cases} p^l(1-p) & l = 0, 1, 2, \ldots \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

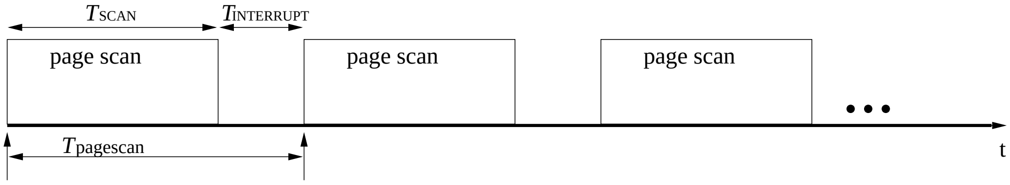

Figure 3.2: *Page scan.* In each T_{pagescan} interval, a host spends T_{SCAN} time in a *page scan* and $T_{\text{INTERRUPT}}$ time performing other tasks.

This implies,

$$E[L] = \frac{p}{1-p} \tag{3.4}$$

When either a packet that was sent on the slave's scan frequency or the corresponding acknowledgment is lost, the set-up procedure is prolonged for another 32 slots since it takes 32 slots for the master to send a page again on the slave's scan frequency. Therefore,

$$A' = A + 32L \tag{3.5}$$

We obtain,

$$E[A'] = E[A] + 32E[L] = 16.5 + \frac{32p}{1-p} \text{ slots} \tag{3.6}$$

A unit in a continuous *page scan* is allowed to interrupt scanning to perform other tasks [1], for example to support a SCO link. We analyze the page duration in the situation where scanning is periodically interrupted for $T_{\text{INTERRUPT}}$ in each T_{pagescan} interval, as shown in Figure 3.2. T_{SCAN} denotes time devoted only to scanning, in a single T_{pagescan} interval. Two cases can be distinguished depending on the time instant a future master starts a page:

1. A future slave is in a *page scan* when a future master starts a page. Let C_1 denote

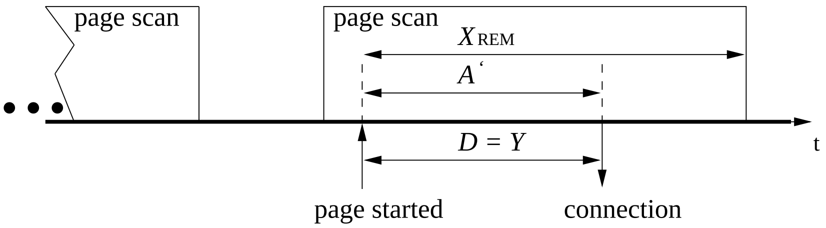

Figure 3.3: Page duration when C_1 and B_1 occur. A paging unit starts a page while a future slave is in a *page scan*. X_{REM} is the time left for that *page scan* from the instant the page started. In this case X_{REM} is longer than time A' needed for a page message to be sent and received on the slave's scan frequency and therefore a connection is established before the slave finishes the *page scan*.

this event. The event C_1 happens with probability

$$P[C_1] = \frac{T_{\text{SCAN}}}{T_{\text{pagescan}}} \tag{3.7}$$

2. A future slave is not in a *page scan* when a future master starts a page. Denoting this event as C_2 we have,

$$P[C_2] = \frac{T_{\text{INTERRUPT}}}{T_{\text{pagescan}}} \tag{3.8}$$

In both cases, a page can successfully finish during the first *page scan* interval (i.e. the first *page scan* since the page started). Let B_1 denote that event. The situation is shown in Figures 3.3 and 3.4.

Otherwise, the first *page scan* ends before a page message has been received. Let B_2 denote that event. From that point, the train of *page scan* intervals repeats until a successful page occurs, as shown in Figures 3.5 and 3.6.

This situation yields a delayed renewal process with interarrival times $X_0, X_1, \ldots$ where X_i is the time from the start of the i^{th} interrupt period plus the time of the

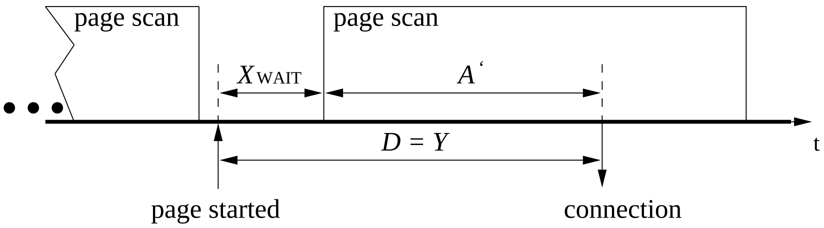

Figure 3.4: Page duration when C_2 and B_1 occur. A paging unit starts a page while a future slave is not in a *page scan*. X_{WAIT} is the time it passes from the beginning of a page until next *page scan* starts. After the *page scan* starts, it takes A' time for a slave to receive a page message. In this case, T_{SCAN} is longer than A' and the connection is established during the first *page scan* since a page started. Hence, a page duration D lasts for $X_{\text{WAIT}} + A'$

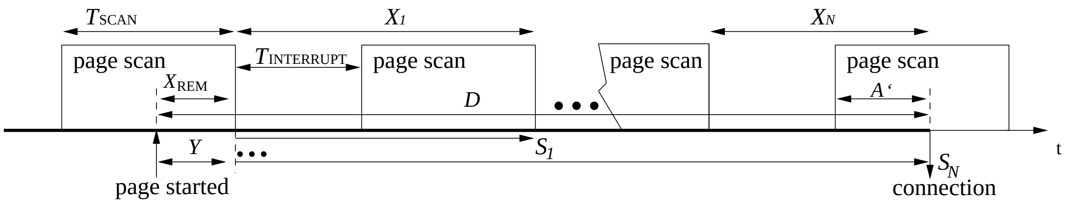

Figure 3.5: Page duration when C_1 and B_2 occur. A page starts while a future slave is in a *page scan*, but remaining time X_{REM} is shorter than time A' needed and the slave leaves the page scan before being successfully paged. A connection will be established in one of the following *page scans*.

i^{th} page scan interval period (not counting the first interval before the first renewal). This renewal process stops when the unit is successfully paged. Thus we define N as a stopping time such that the page is successful during the Nth renewal period. Random variable N represents the number of *page scans* that happen before a successful page (including the interval in which it happened, but not counting the first *page scan* before the first renewal). N is a stopping rule for the experiment [16]. The beginning of each new interrupt is a renewal point of the process. For convenience, arrival times $S_1, \ldots, S_N$ are labeled such that S_n is the instant of arrival $n + 1$.

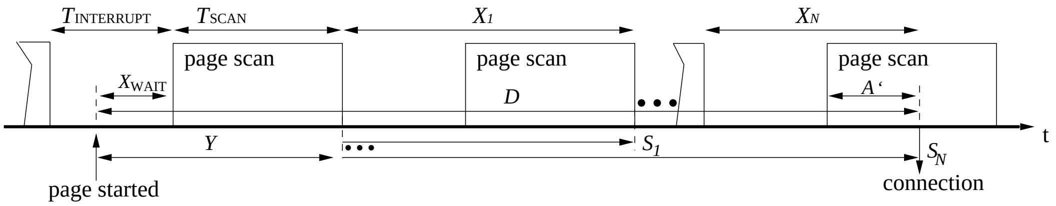

Figure 3.6: Page duration when C_2 and B_2 occur. A page starts while a future slave is not in a *page scan*. During the first following *page scan*, the slave leaves the *page scan* before being paged. A connection will be established in one of the following *page scans*.

We can calculate the average page duration as,

$$E[D] = E[Y] + P[B_2]E[S_N] \tag{3.9}$$

where random variable Y is the time from the beginning of a page until either a connection is established or an interrupt starts, whichever happens first.

In the case of event B_1, Y equals the page duration time. Otherwise, Y is the first interarrival time before the first renewal happens, i.e. $Y = X_0$. Note that $P[B_2]$, the probability that a successful page doesn't happen in the first *page scan*, is also the probability that the first renewal happens.

Let $E[X]$ denote the mean of each of the random variables $\{X_n; n \geq 1\}$. We calculate the expectation $E[S_N]$ using Wald's equality [16] as,

$$E[S_N] = E[X]E[N] \tag{3.10}$$

We reason that N is a geometric random variable with the PMF,

$$P_N(n) = \begin{cases} p_{suc}(1 - p_{suc})^{n-1} & n = 1, 2, ... \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

where p_{suc} is the probability of a successful page in a *page scan* interval. A successful page happens when $A' \leq T_{\mathrm{SCAN}}$. To find p_{suc} we represent T_{SCAN} as:

$$T_{\mathrm{SCAN}} = 32n_f + m_f \tag{3.12}$$

where,

$$n_f = \left\lfloor \frac{T_{\mathrm{SCAN}} - 1}{32} \right\rfloor \tag{3.13}$$

and

$$m_f = T_{\mathrm{SCAN}} - 32n_f \tag{3.14}$$

A future master will send a page on all the frequencies for n_f times during one *page scan*. A page will fail only if all n_f packets sent on a frequency a future slave is scanning on are lost. This occurs with probability p^{n_f}. Otherwise, a successful page happens. Also, if page is not successful after the right frequency has been used for n_f times, a future master sends additional pages on m_f frequencies. With probability $m_f/32$, one of the m_f frequencies that are used again is the right frequency, and with probability $(1-p)$ a page packet is not lost. Thus, a successful page occurs in the final m_f pages with probability $\frac{m_f}{32}p^{n_f}(1-p)$. Hence,

$$p_{suc} = 1 - p^{n_f} + \frac{m_f}{32}p^{n_f}(1-p) \tag{3.15}$$

The expected value of N is,

$$E[N] = \frac{1}{p_{suc}} \tag{3.16}$$

Next we find the mean interarrival time $E[X]$. Each interarrival time X_i can be represented as:

$$X_i = T_{\text{INTERRUPT}} + \min\{A', T_{\text{SCAN}}\} \tag{3.17}$$

From (3.17) we can compute the expected value $E[X]$ using the conditional expected values [17] to be,

$$E[X] = T_{\text{INTERRUPT}} + p_{suc}E[A'|A' \leq T_{\text{SCAN}}] + (1 - p_{suc})T_{\text{SCAN}} \tag{3.18}$$

Since $T_{\text{INTERRUPT}} + T_{\text{SCAN}} = T_{\text{pagescan}}$, (3.18) becomes,

$$E[X] = T_{\text{pagescan}} + p_{suc}(E[A'|A' \leq T_{\text{SCAN}}] - T_{\text{SCAN}}) \tag{3.19}$$

Substituting (3.16) and (3.19) into Equation (3.10) we find $E[S_N]$ to be,

$$E[S_N] = \frac{T_{\text{pagescan}}}{p_{suc}} - T_{\text{SCAN}} + E[A'|A' \leq T_{\text{SCAN}}] \tag{3.20}$$

The conditional mean $E[A'|A' \leq T_{\text{SCAN}}]$ is derived in Appendix and given by Equation (**??**).

To complete the derivation of the expected page duration $E[D]$, we must find $E[Y]$ from Equation (3.9). From Figures 3.3, 3.4, 3.5 and 3.6 we see that Y is different for each of the events $\{C_i, B_j\}$, $i, j = 1, 2$:

$$Y = \begin{cases} \min\{A', X_{\text{REM}}\} & \{C_1\} \\ X_{\text{WAIT}} + \min\{A', T_{\text{SCAN}}\} & \{C_2\} \end{cases} \tag{3.21}$$

Note that X_{REM} is the time left for a *page scan* from the moment the future master starts the *page*, for the event C_1, as shown in Figures 3.3 and 3.5. Since T_{SCAN} is a constant, X_{REM} is a discrete uniform random variable. PMF is given as:

$$P_{X_{\text{REM}}}(x) = \begin{cases} 1/T_{\text{SCAN}} & x = 1, 2, \ldots, T_{\text{SCAN}} \\ 0 & otherwise \end{cases} \tag{3.22}$$

The random variable X_{WAIT} is the time interval the master spends paging a slave that is not in *page scan*, for the event C_2, as shown in Figures 3.4 and 3.6. The PMF of X_{WAIT} is,

$$P_{X_{\text{WAIT}}}(x) = \begin{cases} 1/T_{\text{INTERRUPT}} & x = 1, 2, \ldots, T_{\text{INTERRUPT}} \\ 0 & otherwise \end{cases} \tag{3.23}$$

This implies,

$$E[X_{\text{WAIT}}] = \frac{T_{\text{INTERRUPT}} + 1}{2} \text{ slots} \tag{3.24}$$

Again, using the conditional expected values, we can write the expected value $E[Y]$ as,

$$\begin{aligned} E[Y] &= E[A'|A' \leq X_{\text{REM}}]P[C_1, B_1] + E[X_{\text{WAIT}}]P[C_2] \\ &\quad + E[A'|A' \leq T_{\text{SCAN}}]P[C_2, B_1] + T_{\text{SCAN}}P[C_2, B_2] \\ &\quad + E[X_{\text{REM}}|A' > X_{\text{REM}}]P[C_1, B_2] \end{aligned} \tag{3.25}$$

Conditional expectations for A' and X_{REM} are derived in Appendix and given by Equations (A.3), (A.11) and (**??**). $E[X_{\text{WAIT}}]$ is given by Equation (3.24).

We calculate probabilities needed as $P[C_j, B_i] = P[B_i|C_j]P[C_j]$ and $P[B_1|C_j] = 1 - P[B_2|C_j]$ where $i, j = 1, 2$. Probabilities $P[C_1]$ and $P[C_2]$ are given by Equations (3.7) and (3.8), respectively. The probability of event B_2 needed in Equation (3.9) for the average page duration is then found as $P[B_2] = P[B_2, C_1] + P[B_2, C_2]$.

Next we calculate conditional probabilities $P[B_2|C_j]$.

3.1.1 Case C_1: Slave is in a *Page Scan* When Master Starts a Page

To find the probabilities of events B_1 and B_2 for this case, we reason from Figure 3.3 that the event B_1 happens when a residual time left for a *page scan* from the moment a page started, X_{REM}, is longer than the time A' needed for a page message to be sent on the right frequency without a loss, i.e. the event B_1 happens when $A' \leq X_{\mathrm{REM}}$. Otherwise, event B_2 happens. So, conditioned on event C_1,

$$B_1 = \{A' \leq X_{\mathrm{REM}}\} \tag{3.26}$$

and

$$B_2 = \{A' > X_{\mathrm{REM}}\} \tag{3.27}$$

To find $P[B_2|C_1]$, we represent X_{REM} as a function of two random variables N and M:

$$X_{\mathrm{REM}} = 32N + M \tag{3.28}$$

where,

$$N = \left\lfloor \frac{X_{\mathrm{REM}} - 1}{32} \right\rfloor \tag{3.29}$$

$$M = X_{\mathrm{REM}} - 32N \tag{3.30}$$

Recall that X_{REM} is a uniform random variable over $\{1, \ldots, T_{\mathrm{SCAN}}\}$ where $T_{\mathrm{SCAN}} = 32n_f + m_f$. Thus, $N \in \{0, \ldots, n_f\}$ and $M \in \{1, \ldots, 32\}$ when $N \in \{0, \ldots, n_f - 1\}$ and $M \in \{1, \ldots, m_f\}$ when $N = n_f$. From the PMF of X_{REM} given by Equation (3.22) we conclude that the joint PMF of M and N is,

$$P_{M,N}(m,n) = \begin{cases} 1/T_{\mathrm{SCAN}} & 0 \le n < n_f, 1 \le m \le 32 \\ 1/T_{\mathrm{SCAN}} & n = n_f, 1 \le m \le m_f \\ 0 & otherwise \end{cases} \tag{3.31}$$

Using Equation (3.5),

$$P[B_2|C_1] = P[X_{\mathrm{REM}} < A'] = P[X_{\mathrm{REM}} < A + 32L] \tag{3.32}$$

Substituting for X_{REM} from Equation (3.28) and summing over all possible values of M and N, we obtain,

$$P[B_2|C_1] = \sum_{n=0}^{n_f} \sum_{m=1}^{32} P[A + 32L > 32n + m] P_{M,N}(m,n) \tag{3.33}$$

Since the largest value m can take is $(m)_{max} = 32$ and the smallest value of A is $(A)_{min} = 1$, the inequality $A + 32L > 32n + m$ holds always when $L > n$. In the case when $L = n$ this inequality becomes $A + 32n > 32n + m$ and hence $A > m$ must hold. Therefore,

$$P[A + 32L > 32n + m] = P[L > n] + P[L = n]P[A > m] \tag{3.34}$$

Substituting (3.31) and (3.34) into Equation (3.33), we get

$$\begin{aligned} P[B_2|C_1] \quad &= \quad \frac{1}{T_{\mathrm{SCAN}}} \sum_{n=0}^{n_f-1} \sum_{m=1}^{32} (P[L > n] + P[L = n]P[A > m]) \\ &\quad + \frac{1}{T_{\mathrm{SCAN}}} \sum_{m=1}^{m_f} (P[L > n_f] + P[L = n_f]P[A > m]) \end{aligned} \tag{3.35}$$

From the PMFs of L and A we find the probabilities needed,

$$P[L > n] = p^{n+1} \tag{3.36}$$

$$P[A > m] = 1 - \frac{m}{32} \tag{3.37}$$

Substituting (3.36) and (3.37) into Equation (3.35) we find the probability $P[B_2|C_1]$ as,

$$P[B_2|C_1] = \frac{1 - p^{n_f}}{1 - p}\frac{33p + 31}{2T_{\text{SCAN}}} + \frac{m_f}{T_{\text{SCAN}}}p^{n_f}[1 - \frac{(1 - p)(m_f + 1)}{64}] \tag{3.38}$$

For the special case where $T_{\text{SCAN}} \leq 32$ in an error-free environment (i.e. $p = 0$) term p^{n_f} is of the form 0^0. In the above equation and the remainder of the text we follow the convention that 0^0 evaluates to 1.

3.1.2 Case C_2: Slave is not in a *Page Scan* when Master Starts a Page

As in the previous case, we distinguish between events B_1 or B_2 indicating whether a page was successful during the first *page scan* interval (i.e. the first *page scan* since the page started) or not. We calculate the probability of event B_2 given event C_2. This situation happens only if a master fails to page slave during the full duration of a *page scan*, that is if $A' > T_{\text{SCAN}}$. Therefore $P[B_2|C_2]$ is ,

$$P[B_2|C_2] = 1 - p_{suc} \tag{3.39}$$

Hence,

$$P[B_2|C_2] = p^{n_f}[1 - \frac{m_f}{32}(1 - p)] \tag{3.40}$$

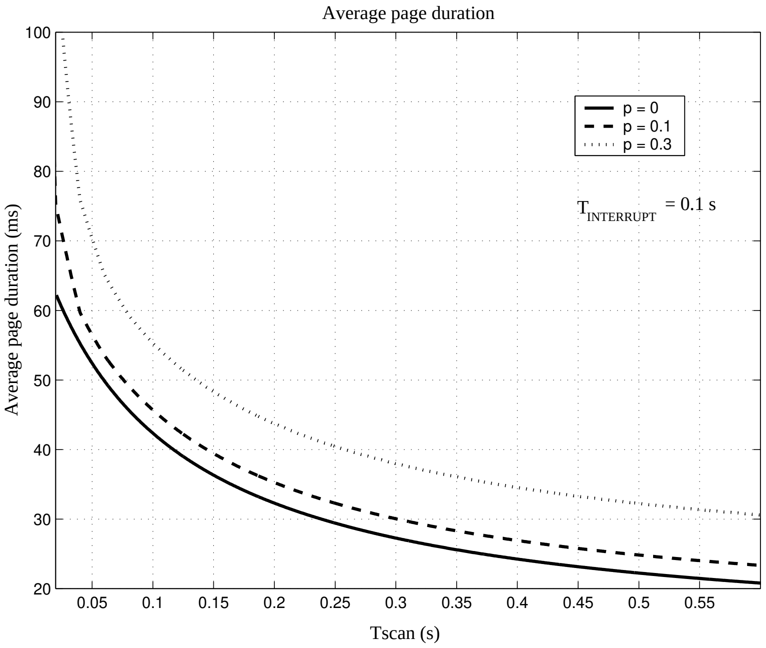

Figure 3.7: Average page duration as a function of the *page scan* interval.

At this point, all the terms in Equation (3.25) have been calculated enabling us to find $E[Y]$. Substituting that result and the expression for $E[S_N]$ given by (3.20) into Equation (3.9) we derive the average page duration.

The average page duration $E[D]$ as a function of T_{SCAN} for three different values of p and for fixed value of $T_{\text{INTERRUPT}}$ is shown in Figure 3.7.

The average page duration as a function of the ratio of the parameters $T_{\text{SCAN}}/T_{\text{INTERRUPT}}$ is shown in Figure 3.8 for the error-free case. We observe the improvement in the connection establishment time when the *INTERRUPT* and *SCAN* intervals are made proportionally smaller while keeping the ratio of the parameters the same. When these intervals are shorter, a paging unit that started a page while a paged unit is not scanning, will have to wait less time for a new scanning interval which will accelerate the set-up procedure.

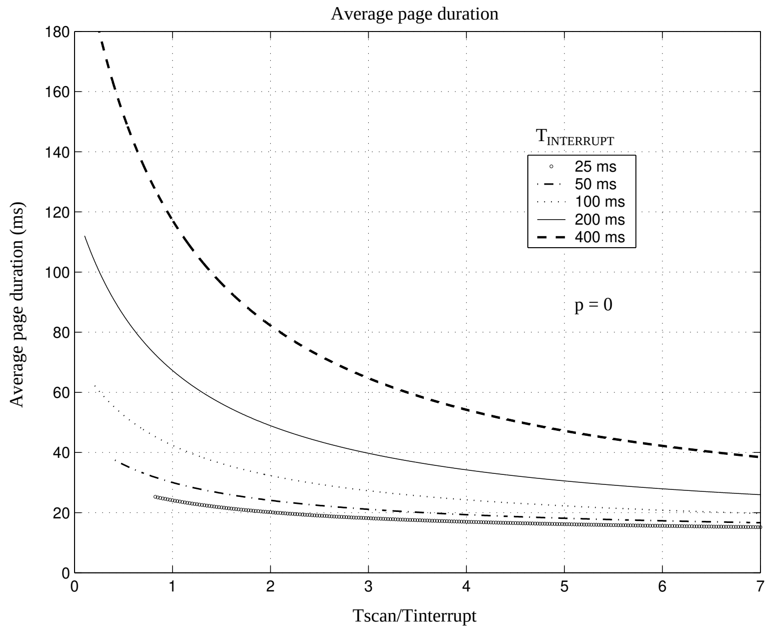

Figure 3.8: Average page duration as a function of the *page scan* interval.

3.2 Paging in R1 mode

In the paging mode R1, a paging unit is paging a host that performs *page scan* every $T_{\text{pagescan}} \leq 1.28$ s as explained in Section 2.1.11. In this mode, it is required by the Specification that a paging unit uses $N_{\text{page}} \geq 128$, as given in Table 2.4. We assume that $N_{\text{page}} = 128$. This implies that a master will send page messages repeatedly on one train 128 times (1.28 s) before switching to the other one. As seen previously, the master uses train A first. The time span of 1.28 s is chosen such that it equals the maximum T_{pagescan} so that a future slave wakes up at least once while one train is used (that is, every 1.28 s) to perform a *page scan*. In the following we assume that $T_{\text{pagescan}} = 1.28$ s [2] which results in the lowest power consumption possible in R1 mode. Again, we look into the case where a paging unit has no knowledge of the clock of a unit it tries to page. The page procedure is shown in Figure 3.9. where a and b denote scan frequencies that belong to trains A and B, respectively.

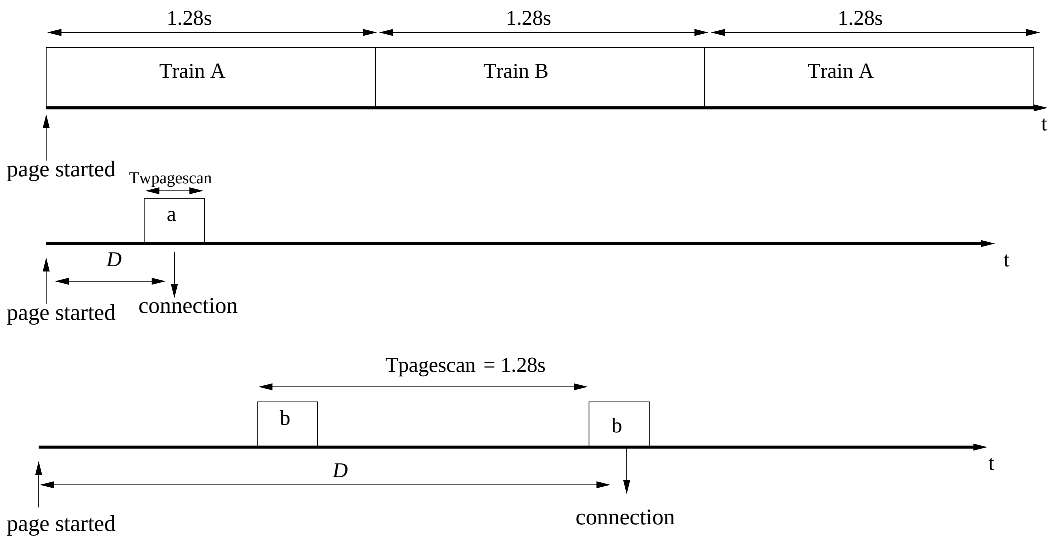

Figure 3.9: Paging in R1 mode when $T_{\text{wpagescan}} = (T_{\text{wpagescan}})_{\text{min}}$. A paging host uses each train for 1.28 s repeatedly. A future slave starts a *page scan* at a random time in the first 1.28 s of paging and repeats a page scan every 1.28 s until a connection is established. The scan frequency could be any of the page frequencies; a and b denote scan frequencies belonging to the future master's trains A and B, respectively.

During the first 1.28 s of a page, a *page scan* will occur at a random time. In the case where $T_{\text{wpagescan}} = (T_{\text{wpagescan}})_{\text{min}}$, where $(T_{\text{wpagescan}})_{\text{min}}$ is just long enough to scan through all 16 frequencies [1] (i.e. it is slightly larger than 10 ms), it is straightforward to calculate the average page duration in the error-free case. In such a situation, if the scan frequency of the slave belongs to train A, the page duration D can be modeled as a random variable uniformly distributed between $\{1, T_{\text{pagescan}}\}$. When the scan frequency belongs to train B, the first page scan will be unsuccessful; the slave will receive a page message in the next wake-up interval. In that case, the duration is a random variable uniformly distributed between $\{T_{\text{pagescan}} + 1, 2T_{\text{pagescan}}\}$. Overall,

$$P_D(d) = \begin{cases} 1/2T_{\text{pagescan}} & d = 1, 2, \ldots, 2T_{\text{pagescan}} \\ 0 & otherwise \end{cases} \tag{3.41}$$

Therefore the average page duration approximately equals T_{pagescan}, that is,

$$E[D] = T_{\text{pagescan}} + \frac{1}{2}\ slots \tag{3.42}$$

In the more general case $T_{\text{wpagescan}} = K(T_{\text{wpagescan}})_{\text{min}}$, where $1 \leq K \leq 128$. To find the PMF of the page duration we use the similar reasoning as in the previous case. We first consider the situation where the scan frequency of the slave belongs to train A. A *page scan* can start anywhere in the interval in which train A is used. As long as a *page scan* starts anywhere over $\{1, T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}}\}$, the situation is exactly the same as in the previous case and the duration is uniform over $\{1, T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}}\}$. The probability of this event is $1 - (K-1)(T_{\text{wpagescan}})_{\text{min}}/T_{\text{pagescan}}$. A *page scan* starting somewhere in the interval $\{T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}}, T_{\text{pagescan}}\}$ implies that a previous *page scan* overlapped with the master's use of train A giving a page duration between $\{1, 16\}$ slots. The probability of this event is $(K-1)(T_{\text{wpagescan}})_{\text{min}}/T_{\text{pagescan}}$. Situation is shown in Figure 3.10. In these reasoning we assume that the overlap occurs only after a $(T_{\text{wpagescan}})_{\text{min}}$ window of a previous *page scan* overlaps. Using these conditional PMFs we find the PMF of the page duration given a *page scan* is performed on the frequency from train A as,

$$P_{D|trainA}(d) = \begin{cases} 1/T_{\text{pagescan}} & d = 17, 18, \ldots, T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}} \\ K/T_{\text{pagescan}} & d = 1, \ldots, 16 \\ 0 & otherwise \end{cases} \tag{3.43}$$

Similar reasoning in the case where a *page scan* is performed on the frequency from

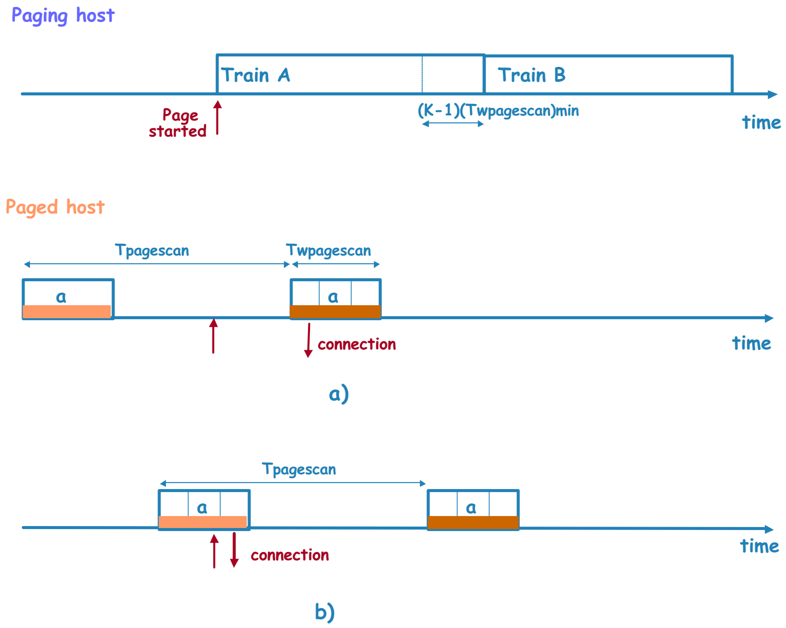

Figure 3.10: Paging in R1 Mode when $T_{\text{wpagescan}} \geq (T_{\text{wpagescan}})_{\text{min}}$. Scan frequencies of a paged host are shown in orange. Figure a) illustrates the situation where a *page scan* starts somewhere in the interval $\{1, T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}}\}$ and Figure b) illustrates the event where a *page scan* starts somewhere in the interval $\{T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}}, T_{\text{pagescan}}\}$ implying that a previous *page scan* overlapped with the master's use of train A giving a page duration between $\{1, 16\}$ slots.

train B, gives the PMF of a page duration for this case. In this situation, it takes an additional T_{pagescan} until a master starts using a train that contains the slave's scan frequency and therefore all the values from the previous case are offset for T_{pagescan}. Their probabilities are the same as in the previous case.

From the two conditional PMFs and using the fact that both situations are equally likely, we calculate the PMF of the page duration to be,

$$P_D(d) = \begin{cases} 1/2T_{\text{pagescan}} & d = 17, 18, \ldots, T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}} \\ 1/2T_{\text{pagescan}} & d = T_{\text{pagescan}} + 17, \ldots, 2T_{\text{pagescan}} - (K-1)(T_{\text{wpagescan}})_{\text{min}} \\ K/2T_{\text{pagescan}} & d = 1, \ldots, 16 \\ K/2T_{\text{pagescan}} & d = T_{\text{pagescan}} + 1, \ldots, T_{\text{pagescan}} + 16 \\ 0 & otherwise \end{cases} \tag{3.44}$$

We observe that a page duration can take two sets of values with higher probability for larger K. For $K = 1$, this equation becomes Equation (3.41). To derive the probabilities in Equation (3.44) we assumed that $(T_{\text{wpagescan}})_{\text{min}}$ was exactly 10 ms.

From the PMF given by Equation (3.44), the average page duration $E[D]$ is:

$$E[D] = T_{\text{pagescan}} + \frac{1}{2} - (K-1)(16 - 128\frac{K}{T_{\text{pagescan}}})\ slots \tag{3.45}$$

For $T_{\text{pagescan}} = 1.28s = 2048$ slots, $E[D]$ becomes,

$$E[D] = 2048.5 - (K-1)(16 - \frac{K}{16})\ slots \tag{3.46}$$

The minimum value for the average page duration $E[D] = 1032.5$ slots corresponds to $K = 128$. This is expected since in that case $T_{\text{pagescan}} = T_{\text{wpagescan}}$ and a future slave is in a *page scan* continuously.

Next, we look into the case where there are errors in the channel. Let D' denote the page duration in this case to distinguish from the error-free case. Assuming that $T_{\text{wpagescan}} = (T_{\text{wpagescan}})_{\text{min}} = 10$ ms, the page duration becomes $D' = D + 2T_{\text{pagescan}}L$

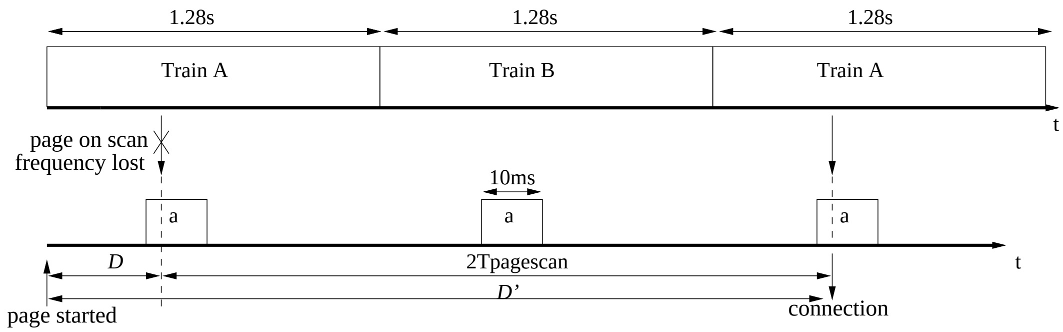

Figure 3.11: R1 mode when there are packet losses in the channel. This is the case $T_{\text{wpagescan}} = (T_{\text{wpagescan}})_{\text{min}}$ and therefore each of the 16 page frequencies of one train is used only once for paging during one *page scan*. Each time a page sent on the scan frequency is lost, the page duration is increased for $2T_{\text{pagescan}}$ time.

since when a page message sent on the scan frequency is lost, it will take $2T_{\text{pagescan}}$ until a future slave can receive a page message again. L represents the number of page packets sent on a slave's frequency or page responses from the slave that have been lost, as in the previous section. The situation is shown in Figure 3.11 for $L = 1$.

Therefore,

$$E[D'] = E[D] + 2T_{\text{pagescan}}E[L] \tag{3.47}$$

To see how the average page duration benefits from a longer *page scan* in this case, we look again into the situation where $T_{\text{wpagescan}} = K(T_{\text{wpagescan}})_{\text{min}}$ where $1 \leq K \leq 128$. In this case, a slave stays in a *page scan* while K page messages are sent on each of the 16 frequencies. The situation is shown in Figure 3.12 for the case $L = 1$ and $K = 2$. Figure shows how the loss of a page packet on the slave's scan frequency increased the page duration by only 16 slots instead of increasing it by $2T_{\text{pagescan}}$ which would happen if $T_{\text{wpagescan}} = (T_{\text{wpagescan}})_{\text{min}}$.

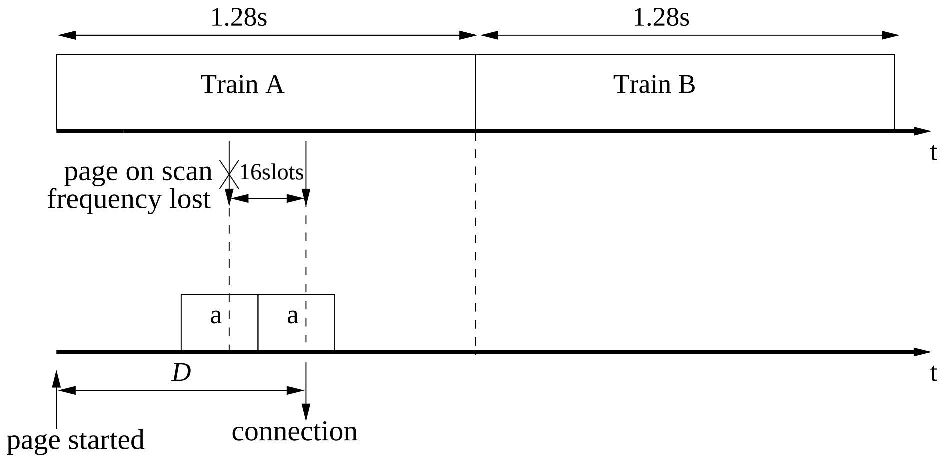

Figure 3.12: R1 mode when there are packet losses in the channel. The case $K = 2$ is shown and therefore one train is used twice during a single *page scan*. If a page on the scan frequency is lost a page may still be successful during the same *page scan* when the second page message sent on the same frequency is not lost.

Therefore, a *page scan* (given that a paging host is using the right train) is unsuccessful only if all K page messages sent on right frequency are lost and only then a page procedure takes additional $2T_{\text{pagescan}}$. Further, each loss of such a page message during a *page scan* that still ends successfully increases the page duration for 16 slots. Hence, it is easy to conclude that,

$$D' = D + \lfloor L/K \rfloor \, 2T_{\text{pagescan}} + 16(L \bmod K) \tag{3.48}$$

An example of such a situation when $L = 7$ and $K = 3$ is shown in Figure 3.13. From the figure, we see that in the example $D' = D + 4T_{\text{pagescan}} + 16$ slots which agrees with Equation (3.48).

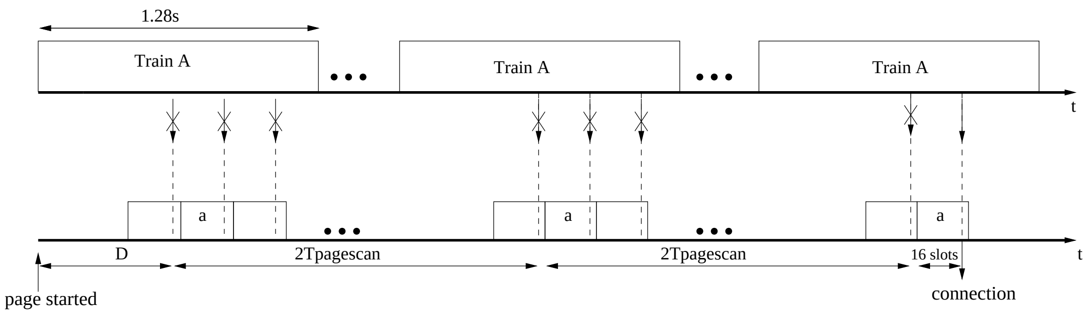

Figure 3.13: Page Duration when 7 page messages that were sent on the right frequency were lost.

From Equation (3.48) we calculate the mean page duration as,

$$E[D'] = E[D] + 2T_{\text{pagescan}}E[\lfloor L/K \rfloor] + 16E[(L \text{mod} K)] \tag{3.49}$$

Next we find the mean of the random variable $\lfloor L/K \rfloor$ appearing in (3.49). To calculate the PMF of this random variable we use the fact that,

$$P[\lfloor L/K \rfloor = z] = P[zK \leq L < (z+1)K] = F_L[(z+1)K - 1] - F_L[zK - 1] \tag{3.50}$$

with F_L being the cumulative distribution function of random variable L. From the previous equation and Equation (3.3) we calculate the PMF to be,

$$P_{\lfloor L/K \rfloor}(z) = \begin{cases} p^{zK}(1 - p^K) & z = 0, 1, ... \\ 0 & otherwise \end{cases} \tag{3.51}$$

Therefore, $\lfloor L/K \rfloor$ is a geometric random variable and its mean is given by,

$$E[\lfloor L/K \rfloor] = \frac{p^K}{1 - p^K} \tag{3.52}$$

We use the same approach to find the mean of random variable $L\text{mod}K$. We observe that,

$$P[L\text{mod}K = y] = \sum_{j=0}^{\infty} P[L = jK + y] \tag{3.53}$$

By substituting the PMF of L into the previous equation, we get the PMF of $L\text{mod}K$,

$$P_{L\text{mod}K}(y) = \begin{cases} \frac{(1-p)p^y}{1-p^K} & y = 0, 1, ...K-1 \\ 0 & otherwise \end{cases} \tag{3.54}$$

The mean of $L\text{mod}K$ is then,

$$E[L\text{mod}K] = \frac{p}{1-p} - \frac{Kp^K}{1-p^K} \tag{3.55}$$

Substituting equations (3.45), (3.52) and (3.55) into Equation (3.49) we obtain the average page duration for the error case to be,

$$E[D'] = T_{\text{pagescan}} + \frac{1}{2} - (K-1)(16 - 128\frac{K-1}{T_{\text{pagescan}}}) + 16\frac{p}{1-p} + \frac{p^K}{1-p^K}(2T_{\text{pagescan}} - 16K) \tag{3.56}$$

Figure 3.14 shows the mean page duration as a function of parameter K for different values of probability of a packet error p. For the error-free case, the average page duration decreases almost linearly with increase of K. However, in the error case we observe such a dependence only after K reaches a high enough value. Otherwise, for short *page scan* intervals, the dependence on K is stronger.

Note that this analysis is constrained to the case where $T_{\text{wpagescan}} = K(T_{\text{wpagescan}})_{\min}$, $K = 1, 2, \ldots$. The more detailed analysis would incorporate cases for which any value for $T_{\text{wpagescan}}$ is possible. The behavior of the mean page duration as the duration of

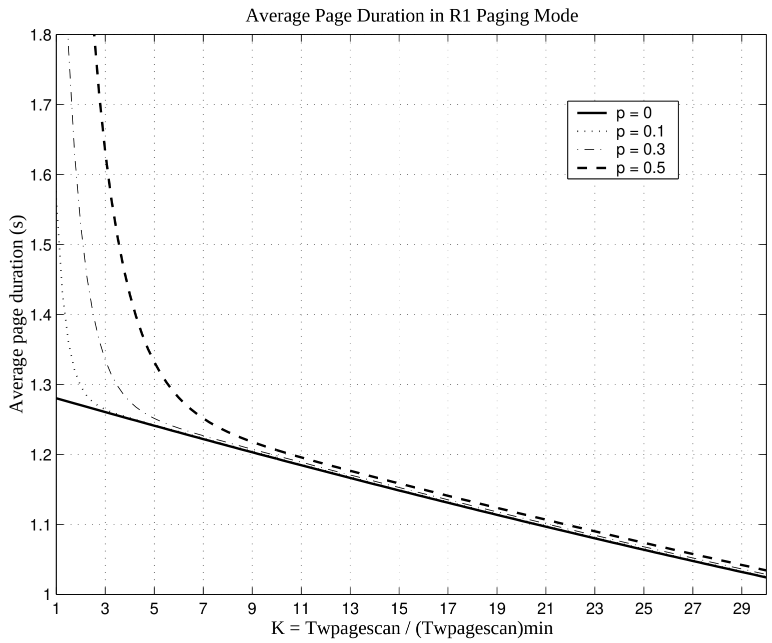

Figure 3.14: Average page duration as a function of K.

a *page scan* interval is varied is expected to be the same. Previous analysis holds for R2 paging mode as well. The difference is in values for T_{pagescan} and N_{page} as given in Section 2.1.11.

Chapter 4

Simulation Results

We now present simulation results on the page duration when paging modes R0 and R1 are used to establish a connection between two hosts. We also present results on the time needed for successful connection establishment when a future master has to perform both an inquiry and a page. The time a host dedicates to a *page scan* is varied to observe its impact on the connection establishment time. The objectives were to estimate:

- The Probability Mass Functions (PMF) of the page duration.
- The dependence of the average connection time on various parameters.

Averages were calculated from the PMFs previously obtained.

A simulation model was built using BONeS simulation software. Following the Specification, a Bluetooth unit was implemented as a finite state machine. The state machine was previously described in Section 2.1. All Bluetooth unit states and substates were modeled. The complete frequency hopping algorithm was modeled as described by the Bluetooth Specification. A separate block was built to handle frequency generation for each slot or half slot when needed. The block was built into each Bluetooth unit. Given the address, native clock and current state of the unit, it generates the frequency to be used in the next slot/half slot. The sample data was generated and compared with the results given by the Bluetooth Specification to verify frequency generation for different modes of operation.

In the experiments, data to be sent arrives at a random time to a host. Upon data arrival, the host initiates a connection establishment procedure with another host. It may perform one-time inquiry followed by a page or a page directly. A host without established connections and data to send performs a *page scan* periodically. We assume that two hosts are within range of each other and that the channel is error-free. After a successful page, both hosts enter a *connection* state and the experiment is halted.

We performed 5000 trials of the experiment to estimate the average page duration when the R0 paging mode was simulated. In the case where the R1 paging mode was used as well as for the case of a connection establishment that included both inquiry and page, 2000 experiment trials were conducted because of the longer simulation time needed per trial.

4.1 Simulation Results on Page Duration

We present results on the average page duration for the R0 paging mode first. The page duration was measured in two scenarios: the page was either preceded by an inquiry or only a page was performed. There was no significant difference in the results obtained.

In a Bluetooth system, a paging unit may have an estimate of its future slave's clock if the hosts have met before. Otherwise, the clock of a future slave is unknown to the paging unit. We distinguished between these two cases and therefore performed two experiments:

1. A future master has no knowledge about the future slave's native clock when it starts a *page*. This means that although the master knows the page hopping

sequence of the slave, it cannot predict which one of those frequencies the slave is currently using (if the slave is in a *page scan*) or the slave is going to use (when the slave enters a *page scan*) and therefore, the master has to choose randomly one of them to start the *page*. The same assumption is used in the analysis of average page duration presented in Chapter 3.

2. A future master has perfect knowledge about the future slave's native clock when it starts a *page*.

Figure 4.1 shows the mean page duration as a function of the parameter T_{SCAN} when $T_{\text{INTERRUPT}} = 0.1$ s for both cases. As expected, a slightly shorter set-up time is needed when a paging unit knows the clock of a paged unit. The maximum difference observed is 5 ms.

The range of values obtained as T_{SCAN} was changed from 0.02 s to 0.5 s is 21.4 ms - 58.3 ms in the case where the clock is known and 22 ms - 63.1 ms for the other case. We observe a less pronounced influence of the parameter T_{SCAN} on the page duration for larger values of T_{SCAN}. The reason is that time taken by the future master to find its future slave in a *page scan* contributes to most of the page duration. Therefore, once the parameters are chosen such that the probability of finding a future slave in a *page scan* is high, any further change of parameters will have no significant influence on the page duration. As given in (3.7), that probability increases as parameter T_{SCAN} increases.

The increase in the average page duration with longer $T_{\text{INTERRUPT}}$ is shown in Figure 4.2. As the value of $T_{\text{INTERRUPT}}$ increases from 0.02 s to 0.5 s the average page time increases from 13.4 ms to 142.4 ms. This is because the time interval between two

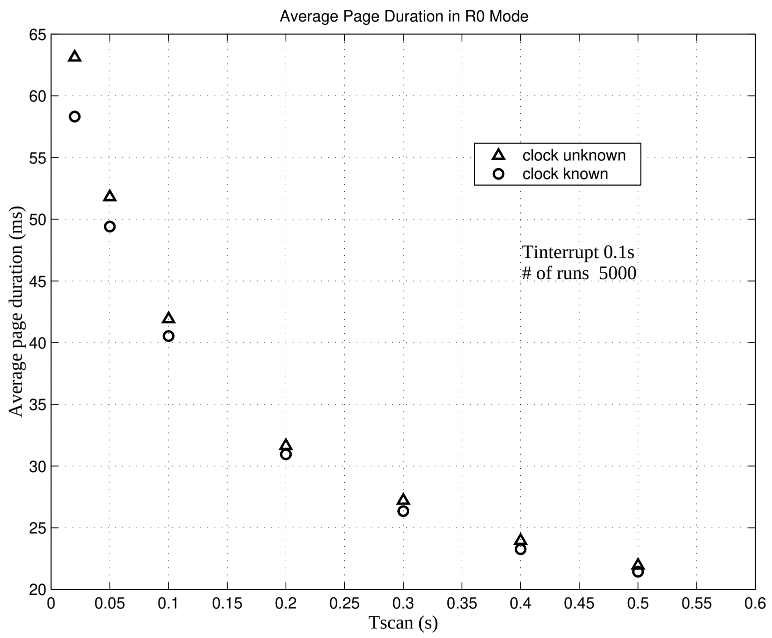

Figure 4.1: Impact of T_{SCAN} on the average page duration.

consecutive page scans increases. In these experiments, the parameter T_{SCAN} was fixed to 0.5 s and the slave's clock was not known to a paging host.

To compare the simulation results with the derived expression for the average page time, the combined plots are given in Figures 4.3 and Figure 4.4. Figure 4.3 shows simulation results for both simulated cases (master either knows or it doesn't know the slave's native clock). The case when slave's clock is not known to the master (labeled as simulation, clock unknown) follows the derived results more closely since the analytical results assume no knowledge of the slave's clock either.

Figure 4.5 presents the average page duration when R1 paging mode is used. It shows the average page time as a function of duration of a *page scan* interval. As K increases, the duration of a *page scan* interval increases and we observe the decrease in the average page duration. Here again, the simulation results are plotted together with the analytical results for comparison.

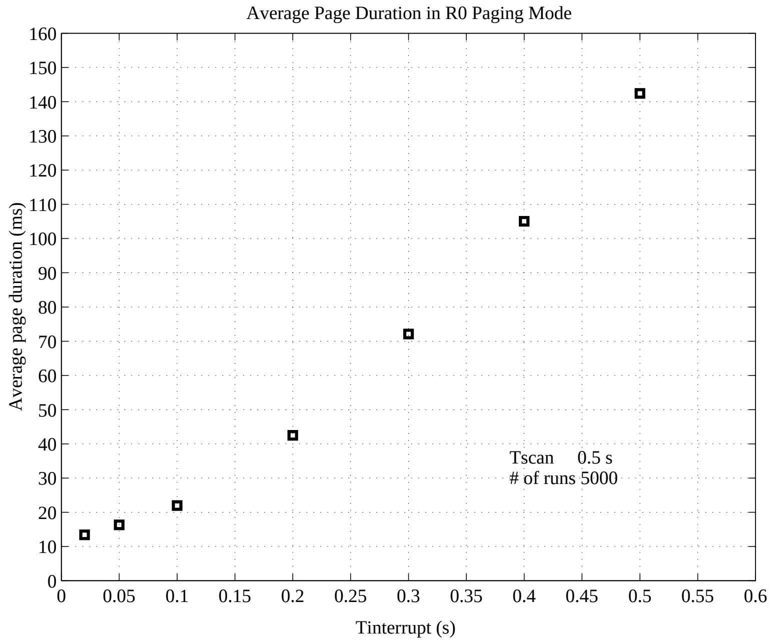

Figure 4.2: Impact of $T_{\text{INTERRUPT}}$ on the average page duration.

4.2 Simulation Results on Connection Establishment Duration

In this section we present results on the duration of the connection establishment that consists of the whole procedure, i.e., both inquiry and page. In this experiment, upon arrival of data, a paging unit performs one-time inquiry immediately followed by a page using R0 mode. The paging unit will leave an *inquiry* immediately upon receiving a response from a future slave (the required number of responses was set to 1). Therefore, the time spent in inquiry represents the time needed for a successful inquiry between two Bluetooth hosts rather than the actual duration of an *inquiry* since two criteria for leaving an *inquiry* state are timeout and collecting the required number of responses. We assume that an idle unit dedicates the time it doesn't spend in *page scan*, to *inquiry scan*. Therefore, in this experiment $T_{\text{INTERRUPT}} = T_{\text{winqscan}}$. This scenario will result in the most efficient establishment procedure (when inquiry is needed before paging) requiring the shortest establishment time for the given $T_{\text{INTERRUPT}}$. It comes at the

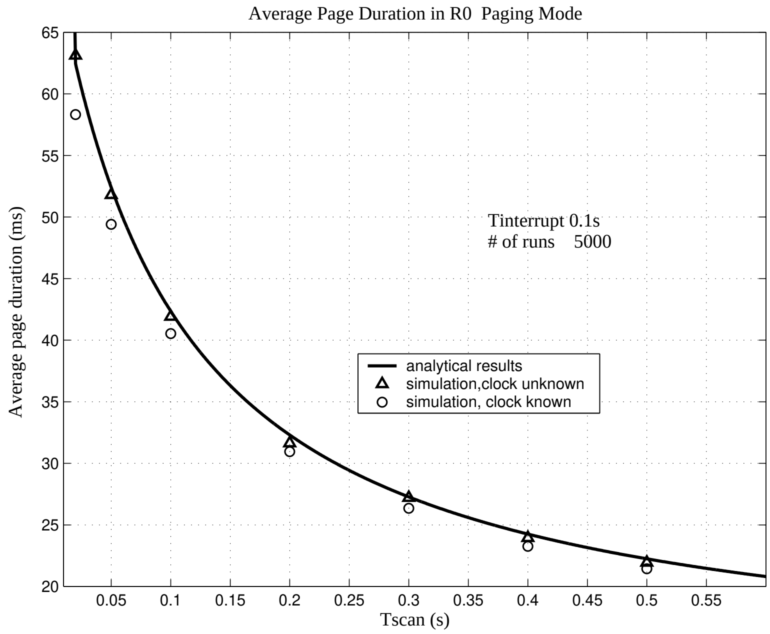

Figure 4.3: Comparison between simulation and analytical results for page duration.

expense of higher power consumption used for *inquiry scan*.

The range of values obtained for the average establishment time as the parameter T_{winqscan} was varied from 0.02s to 0.5s is shown in Figure 4.6. We observe the increase in the set-up duration as T_{winqscan} increases because the page procedure takes longer. The observed values are in the range: 1.37 s - 1.64 s.

During an inquiry, two different situations can arise with same probability giving significantly different values for the set-up time:

1. A frequency a slave is scanning on in *inquiry scan* is one of the frequencies that belongs to the master's train A which is used by the master for the first 2.56 s of *inquiry*, as explained in Section 2.1.

2. A frequency a slave is scanning on in *inquiry scan* belongs to the train B which master will use after 2.56 s. In this case, inquiry is prolonged for 2.56 s.

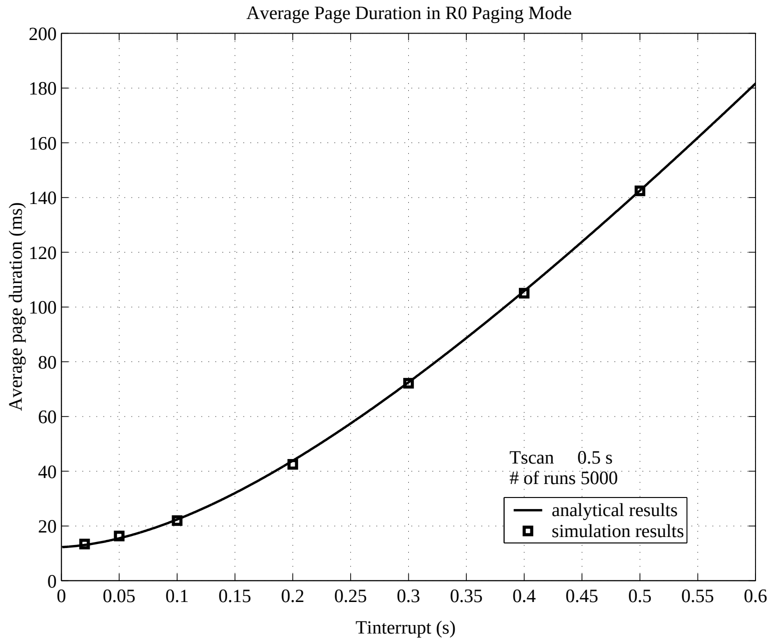

Figure 4.4: Comparison between simulation and analytical results for page duration.

To see the impact of the second case on the time needed for connection establishment, we measured the time that is needed for connection establishment in the experiment where a paging unit always starts an *inquiry* on the train containing slave's scan frequency (Case 1) and compared it with the establishment time when both cases occur with same probability shown in Figure 4.6. Figure 4.7 shows the influence of T_{winqscan} on the mean set-up time in the case where the master always starts an *inquiry* on the train containing slave's scan frequency (Case 1). We observe that the minimum average set-up time in this case was 544 ms for $T_{\text{winqscan}} = 0.1$ s. As T_{winqscan} decreases, the inquiry takes longer since the slave spends most of the time in the *page scan*, while as T_{winqscan} increases, establishment time takes longer because of the longer time intervals between consecutive *page scans* and hence, the longer page needed. In the latter case, the impact is less since in general the paging takes shorter time than inquiry.

The steep increase in the average set-up time for $T_{\text{winqscan}} = 5$ ms is expected

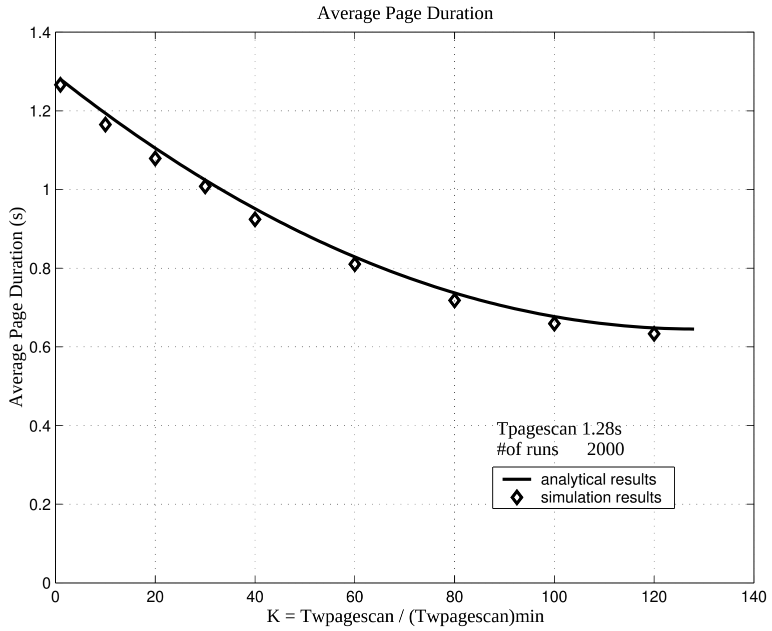

Figure 4.5: The Average Page Duration in R1 Paging Mode.

because the slave doesn't stay in the *inquiry scan* for the duration of one use of a train A (10 ms). That is why by Specification, T_{winqscan} is constrained to be greater than 10 ms. The values obtained for the establishment time are in the range: 544 ms - 616.7 ms revealing much more efficient connection establishment.

Figure 4.8 illustrates the dependence of the set-up duration on T_{SCAN} for Case 1. As the parameter increases from 0.02 s to 0.5 s, the duration varies from 296 ms to 544 ms because of the longer inquiry.

For this case, we show a histogram of the connection establishment time in Figure 4.9 for 2000 trials performed. The value of the parameter T_{SCAN} was set to 0.5 s. The average in this particular case was 0.544.

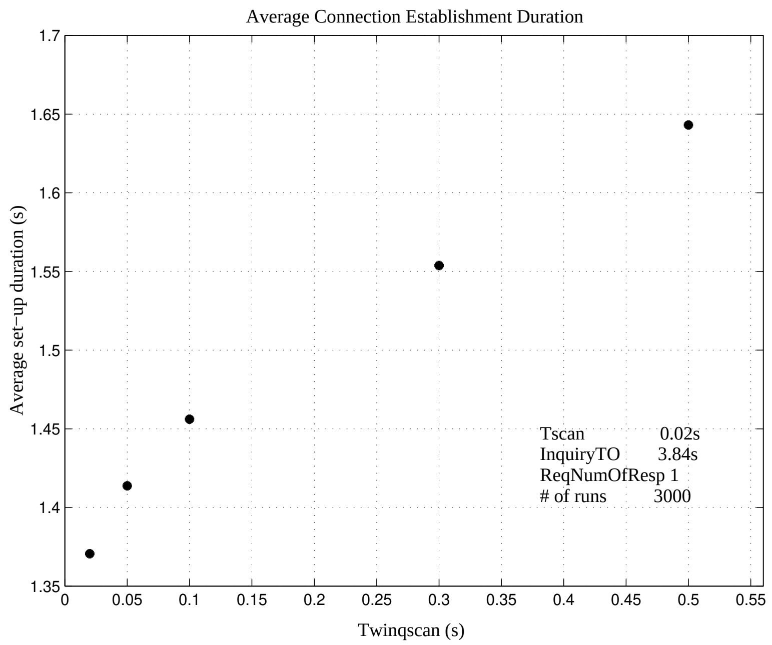

Figure 4.6: Impact of T_{winqscan} on connection establishment duration.

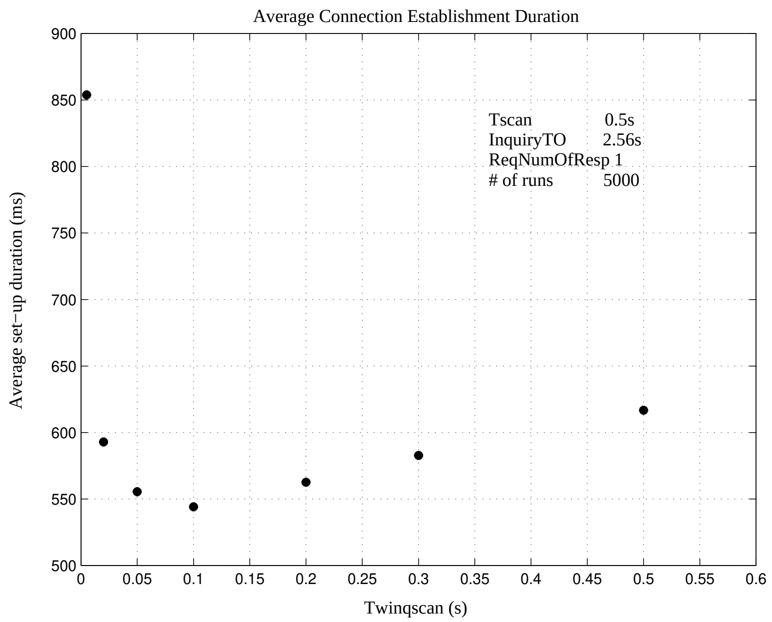

Figure 4.7: Impact of T_{winqscan} on connection establishment duration.

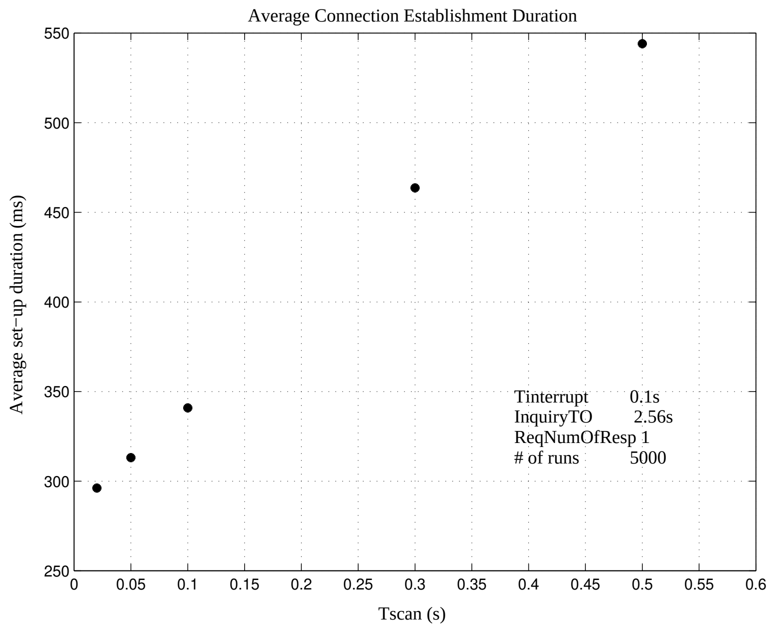

Figure 4.8: Impact of T_{SCAN} on connection establishment duration.

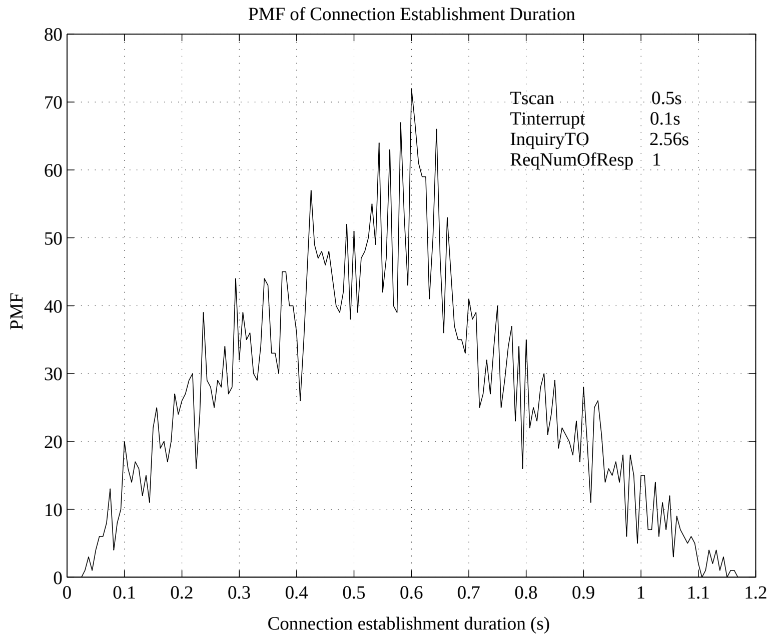

Figure 4.9: A histogram of connection establishment duration.

Chapter 5

Summary

The motivation behind this work was to understand the connection establishment procedure in a Bluetooth system. The objective was to determine the time it takes for two Bluetooth-equipped hosts using different hopping sequences for transmission and reception, to learn about each other, synchronize and establish a connection.

A brief overview of Bluetooth was presented followed by a description of the connection establishment procedure. In Chapter 3 we modeled and analyzed the page duration as a function of the channel packet error and certain system parameters for two different paging modes. Analytical results derived enable an approximate prediction of the expected page duration for chosen values of the time dedicated for a *page scan*.

The results for the $R0$ paging mode suggest that for efficient paging, it is more important to keep the time interval a host is not scanning for page messages $T_{\mathrm{INTERRUPT}}$ short, than to increase the time spent in *page scan* T_{SCAN}, since the small $T_{\mathrm{INTERRUPT}}$ will minimize the time a master wastes by paging a slave that is not performing a *page scan*. The results given in Figure 3.8 illustrate how the short interrupt and *page scan* intervals contribute to efficient paging. Once a master finds a slave in the *page scan*, the page duration is just the time required for a page to be sent on the slave's scan frequency. However, a lossy channel will require adapting the parameter T_{SCAN}, that is, increasing the value of T_{SCAN} to larger values. Simulation results on page duration were presented and compared to analytical results.

Paging time is minimized when a paged host is able to perform continuous *page scan* without any interruptions. In such a situation, a page takes up to a maximum 20 ms in an error-free environment. Such a host would never perform an *inquiry scan* although it is required by the Specification. However, some of the Bluetooth profiles that have been defined recently [1], allow a host to perform a continuous *page scan* without having to respond to an inquiry. Such a device is said to be in non-discoverable mode. This scenario excludes cases where a paged host has previously established connections with other hosts. In addition, this scenario leads to high power consumption that should be avoided.

Simulation results for the set-up procedure consisting of both inquiry and page show the significant increase in delays due to inquiry. The range of values obtained for the average connection set-up time was 1.37 s - 1.64 s. These values, however, may be misleading since there will be a significant number of cases (probability of such an event is 0.5) where a future master starts an inquiry on the right train (that is, on the one that contains the frequency a slave is currently scanning on) in which case the average set-up time is drastically reduced. In this case, the range of values observed was 312.5 ms - 625 ms. We observe significant delays to the set-up procedure when inquiry is performed. General profile recommends that a single inquiry last for 10.24 s [1].

As the results for the average page duration were compared for cases where a paging host has or doesn't have the clock of a host it is trying to page, we observe that, for the $R0$ paging mode, there is no significant reduction in the time needed for a page when the clock is known. The knowledge of the clock will enable a paging host to estimate the scan frequency of the slave and use it to send a page. However, since $N_{\text{page}} = 1$, it takes only 20 ms for the paging unit to sent a page on all 32 paging frequencies and page a slave even if the clock is not known.

The situation is just the opposite in the case of the $R1$ paging mode. Since one train is used for 1.28 s, it is important for a master to start a page on the right train. Results show that the range of values for the page duration when $R1$ paging mode is used and the clock is not known is between 0.64 - 1.28 s in an error-free environment, with a maximum possible value of $2T_{\text{pagescan}} = 2.56$ s. While the dependence of the page duration on the duration of the *page scan* interval in the error-free case is almost linear, we observe that the page duration increases significantly in the presence of errors if the *page scan* is made too short. We conclude that for smaller values of the scanning interval the page duration decreases rapidly as the scanning interval increases (K increases) and approaches values expected in an error-free environment and further on, the dependence is linear. The situation is shown in Figure 3.14.

The motivation behind this work was to analyze the Bluetooth system behavior to enable devising a routing algorithm that is well suited for such an environment. The results obtained suggest that the connection set-up time in Bluetooth could be expected to be large, that is, larger than what existing routing algorithms may have been designed for. The appropriate routing algorithm should take that into account. Since the Bluetooth technology was not developed with routing in its core, a careful design of a routing algorithm should attempt to overcome some of the constraints. An efficient routing algorithm should avoid solutions that involve setting up of new connections for route discovery since such solutions may suffer significant delays. Rather, it should exploit existing topology and perform routing discovery through already established connections. The situation may be improved if once established connections are preserved (in the case of slowly changing topology). For these purposes, a node could use the *park* mode.

Since the establishment procedure imposes the constraint on fast LAN access as well and delays presented will be present in such an application too, faster LAN access is made possible when it is for personal use only, by operating a LAN Access Point in the non-discoverable mode all the time.

5.1 Future Work

This work presents results on connection establishment for particular scenarios when two paging modes are used. Further analysis is possible for different values of some system parameters; however, with respect to the establishment time, this work considered the most favorable cases. Thus, the main conclusion of this work that Bluetooth requires significant amount of time for connection establishment is expected to stay the same. Hence, the next step is to devise routing algorithms that take this delay into account. Since one of the features of Bluetooth technology is low-power, the other objective of Bluetooth routing protocols should be power-awareness. Existing ad hoc routing algorithms typically employ metrics such as shortest path or link quality that often lead to energy-inefficient routes [18]. The metrics based on minimizing battery power consumption are presented in [18]. The main idea of that paper is to increase the life of the nodes and network by conserving power and sharing the cost of routing packets.

Appendix A

Expression for the average page duration derived in Section 3 contains conditional expectations $E[A'|A' \leq X_{\text{REM}}]$, $E[A'|A' \leq T_{\text{SCAN}}]$ and $E[X_{\text{REM}}|A' > X_{\text{REM}}]$. This Appendix presents the derivation of those expectations.

A.1 $E[A'|A' \leq X_{\textbf{REM}}]$

Using the notation introduced in Section 3.1.1, event $\{A' \leq X_{\text{REM}}\}$ is the event B_1 conditioned on occurrence of event C_1. Therefore, the conditional expected value $E[A'|A' \leq X_{\text{REM}}]$ can be found as,

$$E[A'|A' \leq X_{\text{REM}}] = \frac{1}{P[B_1|C_1]} \sum_{(a',x) \in B_1|C_1} a' P_{A',X_{\text{REM}}}(a',x) \tag{A.1}$$

$P[B_1|C_1] = 1 - P[B_2|C_1]$ where $P[B_2|C_1]$ is given by (3.38).

Since A' and X_{REM} are mutually independent random variables and X_{REM} is uniformly distributed it follows that,

$$P_{A',X_{\text{REM}}}(a',x) = P_{A'}(a') P_{X_{\text{REM}}}(x) = \frac{1}{T_{\text{SCAN}}} P_{A'}(a') \tag{A.2}$$

for $x = 1, \ldots, T_{\text{SCAN}}$ and zero otherwise.

x	1	...	$32(n-1)+32$	$32n+1$	$32n+2$	...	$32n+32$	...	$32n_f+m_f$
$a' = 32n+1$				p^n	p^n	...	p^n	...	p^n
$a' = 32n+2$					p^n		p^n		p^n
...					...		...		...
$a' = 32n+32$					p^n	...	p^n	...	p^n

Table A.1: The values for a' and x such that $(a', x) \in B_1|C_1$

.

From Equations (A.1) and (A.2),

$$E[A'|B_1] = \frac{1}{P[B_1|C_1]T_{\text{SCAN}}} \sum_{(a',x)\in B_1|C1} a' P_{A'}(a') \tag{A.3}$$

Next we find sum $\sum_{(a',x)\in B_1|C1} a' P_{A'}(a')$. Representing a' as $a' = 32n+m$ and using PMFs of A and L,

$$P[A' = 32n+m] = P[A = m]P[L = n] = \frac{1}{32}(1-p)p^n \tag{A.4}$$

for $m = 1, \ldots, 32$ when $n = 1, \ldots, n_f - 1$ and $m = 1, \ldots, m_f$ when $n = nf$.

Given that $A' = 32n+m$, the event $\{A' \leq X_{\text{REM}}\}$ means that X_{REM} can take values from $32n+m$ to T_{SCAN}. Using the notation $T_{\text{SCAN}} = 32n_f + m_f$ introduced in Section 3, we conclude that there are $(32(n_f - n) + m_f - m + 1)$ such events. They are all equally probable since X_{REM} is uniformly distributed. This is illustrated in Table A.1 where the values for a' and x such that $(a', x) \in B_1|C_1$ are given with the probabilities of occurrence from Equation (A.4). Only the non-zero entries are shown. The entries show only the term in which the probabilities differ (i.e. term $(1-p)/32$ is omitted). The number of non-zero elements in each row is $32(n_f - n) + m_f - m + 1$.

Therefore, we find $\sum a' P_{A'}(a')$ as,

$$
\begin{aligned}
\sum_{(a',x)\in B_1|C1} & a' P_{A'}(a') \\
&= \sum_{n=0}^{n_f-1} \sum_{m=1}^{32} (32n+m) P[a' = 32n+m] (32(n_f - n) + m_f - m + 1) \\
&\quad + \sum_{m=1}^{m_f} (32n_f + m) P[a' = 32n_f + m] (m_f - m + 1) \qquad \text{(A.5)}
\end{aligned}
$$

Summing over the values of m first and using Equation (A.4), $\sum a' P_{A'}(a')$ becomes,

$$
\begin{aligned}
\sum_{(a',x)\in B_1|C1} a' P_{A'}(a') &= (1-p) \left[32(T_{\mathrm{SCAN}} - 32) \sum_{n=0}^{n_f-1} n p^n - 32^2 \sum_{n=0}^{n_f-1} n^2 p^n \right] \\
&\quad + (1-p) \frac{33 T_{\mathrm{SCAN}} - 682}{2} \sum_{n=0}^{n_f-1} p^n \\
&\quad + \frac{1}{32} (1-p) p^{n_f} m_f (m_f + 1)(16 n_f + \frac{m_f}{6} + \frac{1}{3}) \qquad \text{(A.6)}
\end{aligned}
$$

Summing over the values of n gives the final expression for $\sum a' P_{A'}(a')$:

$$
\begin{aligned}
\sum_{(a',x)\in B_1|C1} a' P_{A'}(a') &= 32(T_{\mathrm{SCAN}} - 32) \frac{p}{1-p} - 32^2 \frac{p(1+p)}{(1-p)^2} \\
&\quad + \frac{33 T_{\mathrm{SCAN}} - 682}{2} (1 - p^{n_f}) \\
&\quad + \frac{1}{32} (1-p) p^{n_f} m_f (m_f + 1)(16 n_f + \frac{m_f}{6} + \frac{1}{3}) \qquad \text{(A.7)}
\end{aligned}
$$

Now, we obtain the conditional expected value $E[A'|B_1]$ from Equation (A.3).

A.2 $E[X_{\mathbf{REM}}|A' > X_{\mathbf{REM}}]$

When events C_1 and B_2 occur, conditional expectation $E[X_{\mathrm{REM}}|B_2]$ adds to the overall page duration as shown by Equation (3.25). Again, using the expression for conditional expectation,

$$E[X_{\text{REM}}|B_2] = \frac{1}{P[B_2|C_1]} \sum_{(a',x)\in B_2|C_1} xP_{A',X}(a',x) \tag{A.8}$$

Once again, we represent X_{REM} as $X_{\text{REM}} = 32N + M$ and use the fact that X_{REM} and A' are mutually independent random variables. We recall from section 3.1.1, Equation (3.34) that $P[A' > 32n+m] = P[L > n]+P[L = n]P[A > m]$. Equation (A.8) becomes:

$$\begin{aligned} &E[X_{\text{REM}}|B_2] \\ &\quad= \frac{1}{P[B_2|C_1]}(\sum_{n=0}^{n_f-1}\sum_{m=0}^{32}(32n+m)P[X_{\text{REM}} = 32n+m]P[A' > 32n+m] \\ &\quad+ \sum_{m=0}^{m_f}(32n_f+m)P[X_{\text{REM}} = 32n_f+m]P[A' > 32n_f+m]) \end{aligned} \tag{A.9}$$

Using (3.1), (3.3) and (3.22) we find the probabilities needed and Equation (A.9) becomes,

$$\begin{aligned} &E[X_{\text{REM}}|B_2] \\ &\quad= \frac{1}{P[B_2|C_1]T_{\text{SCAN}}}\left(\sum_{n=0}^{n_f-1}\sum_{m=0}^{32}(32n+m)(p^{n+1}+(1-p)p^n(1-\frac{m}{32}))\right. \\ &\quad\left.+ \sum_{m=0}^{m_f}(32n_f+m)(p^{n_f+1}+(1-p)p^{n_f}(1-\frac{m}{32}))\right) \end{aligned} \tag{A.10}$$

Calculating the sums, we obtain the conditional mean $E[X_{\text{REM}}|B_2]$ as,

$$\begin{aligned} &E[X_{\text{REM}}|B_2] \\ &\quad= \frac{1}{P[B_2|C_1]T_{\text{SCAN}}}\left(\frac{16p}{1-p}[(n_f-1)p^{n_f} - n_f p^{n_f-1} + 1](\frac{64p}{1-p}+31)\right. \\ &\quad+ (1-p^{n_f})(\frac{528p}{1-p}+170.5) + 32n_f m f p^{n_f} \\ &\quad\left.+ \frac{m_f(m_f+1)}{2}p^{n_f}\left[1-(1-p)(n_f+\frac{(2m_f+1)}{96})\right]\right) \end{aligned} \tag{A.11}$$

A.3 $E[A'|A' \leq T_{\mathbf{SCAN}}]$

The conditional expected value $E[A'|A' \leq T_{\text{SCAN}}]$ can be found as,

$$E[A'|A' \leq T_{\text{SCAN}}] = \frac{1}{P[A' \leq T_{\text{SCAN}}]} \sum_{a'=1}^{T_{\text{SCAN}}} a' P_{A'}(a') \tag{A.12}$$

As explained in Section 3.1, $P[A' \leq T_{\text{SCAN}}] = p_{suc}$ and is given by Equation (3.15) .

To find the sum in Equation (A.12), we represent a' as $a' = 32n + m$.

$$\begin{aligned} \sum_{a'=1}^{T_{\text{SCAN}}} a' P_{A'}(a') &= \sum_{n=0}^{n_f - 1} \sum_{m=1}^{32} (32n + m) P[A' = 32n + m] \\ &\quad + \sum_{m=1}^{m_f} (32n_f + m) P[A' = 32n_f + m] \end{aligned} \tag{A.13}$$

Summing over all the possible values of n and m we obtain,

$$\begin{aligned} \sum_{a'=1}^{T_{\text{SCAN}}} a' P_{A'}(a') &= 32p \frac{(n_f - 1)p^{n_f} - n_f p^{n_f - 1} + 1}{1 - p} + 16.5(1 - p^{n_f}) \\ &\quad + m_f (1 - p) p^{n_f} \left[n_f + \frac{(m_f + 1)}{64} \right] \end{aligned} \tag{A.14}$$

Combining equations (3.15), (A.12) and (A.14), we obtain final expression for conditional expected value $E[A'|A' \leq T_{\text{SCAN}}]$:

$$E[A'|A' \leq T_{\text{SCAN}}] = \frac{32p\frac{(n_f-1)p^{n_f} - n_f p^{n_f-1} + 1}{1-p} + 16.5(1 - p^{n_f}) + m_f(1-p)p^{n_f}(n_f + \frac{(m_f+1)}{64})}{1 - p^{n_f} + (1-p)p^{n_f}\frac{m_f}{32}} \tag{A.15}$$

References

[1] Bluetooth Specification Version 1.0A

[2] J.C. Haartsen, The Bluetooth Radio System, *IEEE Personal Communications*, pp. 28-36, Feb.2000.

[3] C.E. Perkins and E.M. Royer, Ad Hoc On Demand Distance Vector (AODV) Routing, Internet Draft, March 2000. http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-05.txt

[4] Z.J. Haas and M.R.Pearlman, The Performance of a New Routing Protocol for the Reconfigurable Wireless Networks, *Proc. ICC '98.*, pages 156-160.

[5] J. Broch, D.B. Johnson and D.A. Maltz, The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks, Internet Draft, October 1999. http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-03.txt

[6] P. Bhagwat and A. Segall, A Routing Vector Method (RVM) for Routing in Bluetooth Scatternets, *in The Sixth IEEE International Workshop on Mobile Multimedia Communications - MoMuC'99*, pp. 375-379, 1999.

[7] R. Morris, Fixing Timeout Intervals For Lost Packet Detection In Computer Communication Networks, *AFIPS Conf. Proc.*, pages 887-891, June 1979.

[8] J.C. Haartsen, Bluetooth - the Universal Radio Interface for Ad Hoc Wireless Connectivity, Erricson Review No 3, 1998.

[9] J. Zyren, Reliability of IEEE 802.11 Hi Rate DSSS WLANs in a High Density Bluetooth Environment, White paper, Harris Semiconductors, June 1999.

[10] J.C. Haartsen and S.Zurbes, Bluetooth Voice and Data Performance in 802.11 DS WLAN Environment, White paper, Ericsson, May 1999.

[11] D. Suvak, Comparing the Benefits of IrDA and Bluetooth, Wireless Systems Design, May 2000.

[12] R. Mettala, Bluetooth Protocol Architecture, White paper, Nokia, August 1999.

[13] M. Kalia, D. Bansal and R. Shorey, MAC Scheduling and SAR Policies for Bluetooth: A Master Driven TDD Pico-Cellular Wireless System, *in The Sixth IEEE International Workshop on Mobile Multimedia Communications - MoMuC'99*, pp. 375-379, San Diego, California, November 1999.

[14] D. Bertsekas and R. Gallager, *Data Networks,* Prentice Hall, 1994.

[15] M.B. Pursley, The Role of Spread Spectrum in Packet Radio Networks, *Proc. of the IEEE*, Vol. 75, No. 1, pp. 116-134, Jan.1987.

[16] R.G. Gallager, *Discrete Stochastic Processes,* Kluwer Academic Publishers, 1996.

[17] R.D. Yates and D.J. Goodman, *Probability and Stochastic Processes,* John Wiley & Sons, 1999.

[18] S. Singh, M. Woo and C.S. Raghavendra, Power-Aware Routing in Mobile Ad Hoc Networks, *in ACM International Conference on Mobile Computing and Networking - MobiCom '98*, pp. 181-190, Dallas, Texas, October 1998.