Performance

of Real-time Bus Scheduling Algorithms

John P. Lehoczky* and Lui Sha**

*Department of Statistics
**Department of Computer Science
Carnegie-Mellon University
Pittsburgh, PA 15213

Abstract

When periodic tasks with hard deadlines communicate over a bus, the problem of hard real-time bus scheduling arises. This paper addresses several problems of hard real-time bus scheduling, including the evaluation of scheduling algorithms and the issues of message packet pacing, preemption, priority granularity and buffering.

1. Introduction

The scheduling of periodic tasks with hard deadlines is an important problem for real-time control systems. For a distributed system, the scheduling problem has at least two distinct aspects: the scheduling of tasks in an individual processor and the scheduling of message communications among processors across a bus. The problem of processor scheduling has been extensively addressed in the literature. However, the problem of bus scheduling with hard real-time deadlines has received little if any attention.

1.1. Background

In early seventies, Liu and Layland [3] considered the case of task scheduling on a single processor. Under the assumption that the deadline for a task is the same as the periodicity of the task, the optimal fixed priority and dynamic priority scheduling algorithms were determined. They were shown to be the *rate monotonic* and *dynamic deadline* scheduling algorithms respectively. In this case, the word "optimal" means that if a task set can be scheduled by any fixed (dynamic) priority algorithm, then it can also be scheduled by the rate monotonic (dynamic deadline) scheduling algorithm. Liu and Layland derived the worst case scheduling bounds for these two algorithms. They proved that the dynamic deadline algorithm can schedule any set of periodic tasks with processor utilization no larger than 100%. They proved that the rate monotonic algorithm can schedule any set of n periodic tasks with processor utilization no larger than $n(2^{1/n}-1)$ or any task set with processor utilization below $\ln 2 \approx 0.693$.

This work has been extended in a variety of ways. For example, Leung and Merrill [2] and Lawler [1] considered deadline scheduling on multiple processors, while Martel [4] extended Lawler's work to introduce release times and due times. Mok [5] proved that the dynamic deadline scheduling algorithm remains optimal with respect to a mixed set of periodic and sporadic tasks with hard deadlines. A sporadic task has a random arrival time, a computation time and a hard deadline. Mok assumed a minimum separation time between two consecutive arrivals of sporadic tasks. Mok also showed that the least slack time algorithm is optimal as well. The slack time is the time remaining until the task deadline reduced by the remaining task computation time.

1.2. Real-time Bus Scheduling

Bus scheduling problems arise when periodic tasks with hard deadlines must communicate over a bus. This creates a periodic communication with a hard deadline. If there are many nodes connected to the bus and many periodic messages which have hard deadlines, one needs a bus access protocol which will allow all the deadlines to be met. The problem has a strong similarity to the processor scheduling problem. Here the bus is the resource which must be scheduled. The tasks are the messages which

must be sent, and each task has a hard deadline. Standard protocols such as Ethernet or a token ring do not support real-time hard deadline traffic. The natural bus protocol for this situation is a slotted contention protocol which will be collision free. In this type of protocol, each node desiring to transmit in a particular slot will contend at a certain priority level. The node contending at the highest priority will win. Ties are resolved arbitrarily, for example by using the node addresses. The problem is to assign the priorities in such a way that all of the individual message deadlines are met.

There are three important issues which distinguish the bus scheduling problem from the processor scheduling problem: task preemption, priority level granularity, and buffering. Each must be considered. The goal of this paper is to investigate the behavior of the rate monotonic scheduling algorithm in the context of bus scheduling.

1.2.1. Task Preemption

In processor scheduling, it may be reasonable to assume that a task can be preempted at any time and resumed later at the point of preemption. Such an assumption is not appropriate for bus scheduling. Any bus transmission that is preempted must be entirely retransmitted. This observation coupled with the hard deadline environment limits attention to slotted collision free protocols. Given such an environment, preemption is no longer an issue, provided message lengths and message deadlines are an integer number of slots or packet lengths, and packet transmission times are synchronized to slot boundaries.

1.2.2. The Granularity of Priority Levels

In the processor scheduling case, it is reasonable to assume that there is a sufficient number of priority levels to assign distinct priorities to each task. In the bus scheduling context, this is no longer true. To resolve each contention, it takes at least a round trip propagation delay of the bus. Consequently, a bus protocol will allow packets to contend for slots at only a limited number of distinct priority levels. Packets may contend at one fixed level or may be allowed to change their priority over time as a function of the message deadline; however, the choice of priority levels is limited to a relatively smaller set, compared with the essentially infinite set for a processor protocol. This could result in two packets contending at the same level even though one should have higher priority than the other if more priority levels were available. Since ties are resolved arbitrarily, this can result in a *priority inversion*--the situation in which a packet which an optimal algorithm would assign highest priority does not win the contention for the slot, because it is forced to contend at the same level as a lower priority packet and loses in the tie-breaking stage. The insufficient priority granularity problem will cause some loss in scheduling potential. We will precisely quantify this loss of scheduling potential for the rate monotonic scheme in Section 4.

1.3. Buffering Problems

There are two buffering issues which are unique to bus scheduling. First, hard deadlines for message packet transfers can arise from task deadlines. They can also arise from buffer limitations. If only one buffer is available, then the message packet in the buffer must be sent before the second packet arrives. When additional buffers are available, the deadline of the transmission of a message packet can be extended. If two buffers are available (double buffering), the deadline of a packet can be delayed until the arrival of the third rather than the second packet. The effect of multiple buffering will be discussed in Section 5.

Second, common scheduling algorithms associate a priority with a message. Once a message becomes the highest priority one contending for the bus, it is assumed that all the packets of this messages will be sent in consecutive time slots until all the packets are sent or until a higher priority message appears. Unfortunately, this may be difficult to implement. The message dispatching and receiving function is typically performed by a bus interface unit (BIU). Such units often consist of a relatively small number of fast buffers which can match the speed of the bus. Consequently, many BIUs can neither continuously send nor receive many packets in consecutive slots.

There is a simple solution to this problem, namely *packet pacing*. Consider a periodic message consisting of n packets, which must be be sent within a deadline of p slots. If the message is of the highest priority, then these packets will be sent in the first n consecutive slots. This could cause buffering problems. The idea of packet pacing is to divide the message into p/n single packet messages. Thus, one packet is sent every p/n slots. For example, a 10 packet message with a period of 100 slots would be treated as 10 single packet messages with a period of 10 slots each. This smooths out the transmission and minimizes the buffering problem.

There is, however, a potential problem of *pacing loss*. This occurs when p/n is not an integer. One

must round p/n down to the next integer in order to ensure that the deadline is met. This rounding shortens the deadlines of the resulting single packet messages. If the scheduling algorithm is a function of the packet deadlines, then the priorities of these single packet messages are higher than would be necessary to ensure their deadlines. As a result, other packets in the message set may miss their deadlines unnecessarily. For example, if a message has 3 packets every 5 slots, pacing would force the packets to be sent in the first 3 slots. If a second message requires one slot out of every 3, they could not be scheduled together using pacing, while they could without it. We assume that pacing will be used in a manner that does not incur a scheduling problem. For example, if a message has 3 packets with a period of 10, it will be transformed into 1 single packet message with a period of 5 and into another single packet message with a period of 10. If we have a 3 packet message with a period of 13, then we will replace it with 3 single packet messages with a period of 13. That is, we transform a given message set into a single packet message set with equivalent bus utilization. We perform pacing whenever there is no pacing loss.

This paper is organized as follows. Section 2 formulates the bus scheduling problem and Section 3 presents scheduling bounds for the single buffered case with sufficient priority granularity. Section 4 develops bounds for the case with insufficient priority granularity. Section 5 discusses the effect of multiple buffering. Section 6 summarizes the results of this paper.

2. Problem Formulation

We begin by reviewing the assumptions underlying the bus scheduling problem.

1. We consider only periodic messages transmitted over a common bus. The transmission of packets on the common bus are divided into time slots. Each slot transmits a single packet. Message packets are assumed to be initiated at the slot boundaries. They contend for the slot, and the one with the highest priority wins the slot. Ties, if any, are broken arbitrarily. Once a packet starts transmission, it cannot be preempted.
2. Messages are periodic. If a message is initiated at time I(0) with period T, then it will be ready for transmission at times $I(k) = I(0)+kT$, where $k \geq 0$. The deadline of the message packet is a function of the number of available buffers. With B buffers, the associated deadlines are $D(k) = I(k)+BT$. The period T is assumed to be an integer number of time slots. In addition, each message contains an integer number, C, of packets.
3. The scheduling algorithm used to assign priorities to the packets is the *rate monotonic* scheduling algorithm. This algorithm assigns priorities to messages in an inverse relation to their periods. Messages with shorter periods get higher priorities.

The utilization factor of a periodic message j is $U_j = C_j/T_j$. The bus utilization factor of a set of messages is the sum of individual message bus utilization factors. For a given set of periodic messages, $\{(I_i(0), T_i, C_i), 1 \leq i \leq n\}$, we assume that the messages are labeled so that $T_1 \leq T_2 \leq \ldots \leq T_n$. Consequently, the messages are labeled in the rate monotonic priority order. We say that a particular set of messages can be scheduled by a particular algorithm if, using this algorithm, all the deadlines for all messages are met. A useful lemma regarding the scheduling of a set of periodic messages is the *critical zone* lemma due to Liu and Layland [3]. Given a set of messages with periods $\{T_1, \ldots, T_m\}$ where T_m is the longest period, the time zone from time 0 to T_m is called the critical zone when all m messages (tasks) are initiated at time zero. The critical zone lemma states that if a message set can meet all the deadlines during the critical zone, then it can meet all its deadlines all the time. This lemma is used throughout this paper.

3. Scheduling Bounds

In this section, we assume that there is a sufficient number of priority levels and that message packets are single buffered. These two assumptions will be relaxed later. A scheduling bound with respect to some scheduling algorithm refers to the bus utilization of the worst case message set. Consequently, if the bus utilization of a message set is below this bound, then this set can be scheduled by the algorithm in question.

A given set of messages can be categorized in three important ways: the length of the longest period, the number of distinct periods and the number of single packet messages. To derive the bounds on the scheduling potential of a given algorithm, we can fix one of these three variables and consider all the possible message sets. This leads to three distinct formulations of the bus scheduling problem, and each allows us a distinct bound for utilization. To deter-

mine if a given message set is schedulable, we can calculate all three bounds. If the bus utilization factor of a message set is below the largest of the three bounds, then it is guaranteed to be schedulable.

3.1. Problem Formulation 1: The Fixed Longest Period

In this formulation, we assume that the longest period of a set of single packet messages is n. Consequently, we consider message sets consisting of m messages with the longest period n. For each n, we seek the message set which fully utilizes the available slots during the critical zone but has the minimal bus utilization, $U = C_1/T_1 + ... + C_m/T_m$.

We introduce m single packet messages with $T_1 \leq ... \leq T_m = n$. We assume that this set of messages fully utilizes the available slots during the critical zone. To identify the worst case, we proceed by a series of simple lemmas. These lemmas parallel the development in the Liu and Layland paper [3].

Lemma 1.1: We need to only consider the case $T_1 > n/2$.

Proof: If $T_i \leq n/2$ for any message packet i, then it can be replaced by two single packet messages each having period $2T_i$. The modified message packet set also fully utilizes the bus during the critical zone, has longest period n and has the same bus utilization. □

This lemma allows us to assume $T_1 > n/2$. In addition, we can combine all messages with the same T_i into one message with C_i as the number of packets during a period. If there is no packet for some i we set $C_i = 0$. This reduces consideration to message sets with the form: $\{(i, C_i), H((n+1)/2) \leq i \leq n\}$, where H(.) is the ceiling function. The reduction to the case $(T_n/T_1) < 2$ mandated by Lemma 1.1 produces the following property. Given any message packet (i, C_i), the message packets (j, C_j) with $j < i$ are initiated exactly twice during the first i time slots.

Lemma 1.2: The minimal bus utilization case satisfies the property $C_i \leq 1$ for $H((n+1)/2) \leq i \leq n-1$, where H(.) is the ceiling function.

Proof: Suppose that $C_i > 1$ for some i. Consider the modified message set with $C_i^* = C_i - 1$ and $C_{i+1}^* = C_{i+1} + 1$. The modified message set still fully utilizes the available slots during the critical zone, but the bus utilization is reduced by $1/i - 1/(i+1)$. Consequently, the minimal utilization case must satisfy $C_i \leq 1$, $H((n+1)/2) \leq i \leq n-1$. □

Since we are concerned only with cases in which slots in the critical zone are fully utilized, it follows that the first n slots must be filled. Furthermore, since $C_i \leq 1$, for $i \leq n-1$ and each is initiated exactly twice in the first n slots, for full critical zone utilization we must have

$2\sum_{i=H((n+1)/2)}^{n-1} C_i + C_n \geq n$, thus

$C_n \geq n - 2(n - H((n+1)/2)) = 2H((n+1)/2) - n$.

In addition, if there are k levels with $C_i = 0$, then $C_n \geq 2H((n+1)/2) - n + 2k$.

Lemma 1.3: The minimal bus utilization case satisfies $C_i = 1$, $H((n+1)/2) \leq i \leq n-1$, and $C_n = 2H((n+1)/2) - n$.

Proof: Suppose that slots in the critical zone are fully utilized and $C_i = 0$. It follows that C_n is at least 2. Consider a modified message set with $C_i^* = 1$ and $C_n^* = C_n - 2$. The modified message set also fully utilizes the slots in the critical zone, but the bus utilization is decreased by $2/n - 1/i > 0$. Consequently, no message packet set with $C_i = 0$ can have minimal bus utilization. It follows that in order to minimize the bus utilization and fully utilize the slots in the critical zone, we must select $C_i = 1$, $H((n+1)/2) \leq i \leq n-1$ and $C_n = 2H((n+1)/2) - n$. □

These lemmas yield a proof of the following theorem.

Theorem 1: For the bus scheduling problem with single buffering, sufficient priority levels and a maximal period n, the worst case message set is given by single packet messages with periods p, p+1, ..., n−1 and (2p−n) single packet messages with period n, where $p = H((n+1)/2)$.

For example, if n = 7, the worst case is given by four single packet messages with periods 4, 5, 6 and 7. If n = 8, the worst case is given by 5 single packet messages with periods 5, 6, 7, 8 and 8. The lower bound on utilization is given as a function of n by,

$$U_n^* = \sum_{i=H((n+1)/2)}^{n} 1/i + (1/n)(2H((n+1)/2) - (n+1))$$

One can easily verify that $U_n^* > U_{n+1}^*$, and as $n \to \infty$, $U_n^* \to \ln 2 \approx 0.693$. A graph of U_n^* is presented in Figure 1. The bound $U_n^* > n(2^{1/n} - 1)$, which is the bound for the rate monotonic scheduling algorithm in processor scheduling [3]. However, both converge to the same limit. The bus scheduling bounds are higher because T_i and C_i are restricted to integer values.

3.2. Problem Formulation 2: A Fixed Number of Distinct Periods

In the previous formulation, the longest period was fixed but the task set was otherwise arbitrary. In this formulation, we consider message sets having n dis-

tinct periods, $T_1 < ... < T_n$, but the task set is otherwise arbitrary. We again focus on the full utilization of the available slots in the critical zone and seek to find the message set which has the minimal bus utilization. This analysis is again similar to that of Liu and Layland. First, suppose that we remove the restriction that C_i and T_i must be integers. It follows from Liu and Layland that the worst case message packet set is given by:

- $C_i^* = T_{i+1}^* - T_i^*, 1 \le i \le n-1$
- $C_n^* = 2T_1^* - T_n^*$
- $T_i^*/T_1^* = 2^{(i-1)/n}, 1 \le i \le n$

The resulting bound is given by $U_n^* = n(2^{1/n} - 1)$. This bound converges to ln 2 as n approaches infinity. In this solution, the minimal bus utilization case is one in which not all T_i and C_i are integer. Consequently, this construction does not directly apply to the bus scheduling problem where integer values are required. Nevertheless, we can approximate this solution arbitrarily closely with integers, since the T_i and C_i are unbounded. This is done in the following way. Let T_1 be a very large integer, and then pick integers $T_2 < ... < T_n$ so that T_i/T_1 is approximately $2^{(i-1)/n}$. Choose $C_i = T_{i+1} - T_i$ for $1 \le i \le n-1$ and $C_n = 2T_1 - T_n$. The result is a full utilization case with Ts and Cs integer with bus utilization approximately $n(2^{1/n} - 1)$. By letting T_1 increase, the approximation will become exact. We thus have proved the following theorem.

Theorem 2: For the bus scheduling problem with single buffering, sufficient priority levels and n distinct periods, the worst case bound is

$U_n^* = n(2^{1/n} - 1)$.

A graph of U_n^* is presented in Figure 2.

3.3. Problem Formulation 3: A Fixed Number of Single Packet Messages

In this formulation, we consider n *single packet* messages with $T_1 \le T_2 \le ... \le T_n$. We wish to identify a set of values for Ts which lead to full utilization of the slots in the critical zone and which minimize the utilization of the bus. This is similar to formulation 1 except that n now indexes the number of messages rather than the longest message period. Nevertheless, the structure of the worst case and the proofs are similar. Indeed, we have the following theorem.

Theorem 3: The set of n single packet messages which fully utilizes the slots in the critical zone and which minimizes the bus utilization for the single buffer and sufficient priority level case is given by

$(T_1, ..., T_n) = (n, n+1, ..., 2n-1)$.

For example, the worst case for n = 5 consists of message periods given by 5, 6, 7, 8 and 9. The utilization of this case is given by $U_n^* = \sum_n^{2n-1} 1/i$. Again we have $U_i^* > U_{i+1}^*$, and U_n^* approaches ln 2 as n approaches infinity. A graph of U_n^* is presented in Figure 3.

We have focused on full utilization cases, that is message sets for which if the number of packets in any message is increased, the set can no longer be scheduled. Given the discrete nature of bus scheduling, it may be more appropriate to characterize the worst non-schedulable case. This refers to the message set which cannot be scheduled and has minimal bus utilization. For formulation 3, we seek a minimum utilization message set which contains n messages and misses a deadline. To minimize utilization, only the message with the longest period should miss its deadline. This worst case is given by messages with periods n−1, n, n+1, ..., 2n−3, 2n−1. The message with period 2n−1 will miss its deadline. The utilization is larger than the full utilization bound by 1/2(n−1), which converges to 0 as $n \to \infty$. For example, if n= 6, then the worst case non-schedulable message set is given by single packet messages with periods 5, 6, 7, 8, 9, and 11.

4. The Effect of Insufficient Priority Levels

In this section, we investigate the implementation of a finite priority granularity for the rate monotonic scheme. Assume that there is a total of K distinct priority levels and a corresponding priority grid $\{0, L_1, ..., L_k\}$. If a period P falls between two grid levels, i.e. $L_{i-1} < P \le L_i$ for $1 \le i \le K$, then the message will be assigned priority i. The first observation is that the maximal message period, P_{max}, must be bounded, or else the worst case utilization can be made to approach 0.

Theorem 4: For a finite priority granularity, the worst case utilization bound approaches 0 as the longest period, P_{max}, approaches infinity.

Proof: Fix the value of L_{k-1}. Since P_{max} is unbounded, we must let L_k be infinite. All messages with period longer than L_{k-1} will be given priority K. Since the ties are broken arbitrarily within a single priority level, it is possible that a message with period N will be given priority higher than a message with period M, where $N > M > L_{k-1}$. That is, the message with period M could win a bus contention between the two. Now if there were M such messages with period N or the period N message required M slots, then the period M message would miss its deadline. The bus utilization of this set of

messages is given by $1/M + M/N$. When M and N are unbounded, the utilization can be made arbitrarily small. □

We now assume P_{max} is finite and set it equal to L_k. It is useful to define a measure of the granularity of the grid. For a given priority i, let $G_i = (L_{i-1}+1)/L_i$. If there is only one period per priority level i, then $G_i = 1$, the maximum possible value. We define $G = \text{Min}_{1 \le i \le K} G_i$ to be the granularity measure of the grid.

In the previous section where $G = 1$, we observed that no matter which formulation was chosen, the corresponding bound converged to ln 2. In this section, we seek to determine the corresponding limiting bound for values of $G < 1$ under the rate monotonic scheduling algorithm. We recall that the worst full utilization case consisted of n messages having periods n, $n+1$, ..., $2n-1$. The worst non-schedulable case had n messages with periods given by $n-1$, n, $n+1$, ..., $2n-3$, $2n-1$. In the latter case, the message with the longest period was the one which missed its deadline. Each of the messages with shorter periods is initiated exactly twice during the critical zone. The only way the utilization can be reduced and the same message remain non-schedulable is for one of the shorter messages to be replaced by *two* messages having the same priority as $2n-1$ but a longer period. If $L_{i-1} + 1 \le 2n-1 \le L_i$, then we should consider replacing the message having period $n-1$ with two having period L_i. This will reduce utilization if $1/(n-1) > 2/L_i$. We continue to replace shorter messages with period $n+j$ by two messages having period L_i until the total utilization is no longer reduced. This means that we continue through j satisfying the inequality $1/(n+j) > 2/L_i$ and $1/(n+j+1) < 2/L_i$ or equivalently $(L_i/2)-n-1 < j \le (L_i/2)-n$ and $j \le n-2$. If $L_i/2 > L_i+1$, then $G < 0.5$, and one should replace all shorter periods with two messages of period L_i. This results in a set consisting of one message having period $2n-1$ and $2n-2$ messages of period L_i. The utilization is minimized by choosing $2n-1$ as small as possible subject to being in the i th priority class. This entails $2n-1 = L_{i-1}+1$. The worst case utilization for priority level i and $G_i < 0.5$ is given by $1/(L_{i-1}+1) + L_{i-1}/L_i$ which is approximately equal to G_i.

If $G_i > 0.5$, then one selects j to be approximately $L_i/2 - n$. This results in a message set having periods $n+j+1$, ..., $2n-1$ and 2j messages of length L_i. This results in a fully utilized case. More precisely, define $j = F(L_i/2 - n)$ where F(.) is the floor function. The resulting utilization is given by

$$\sum_{k=n+j+1}^{2n-1} 1/k + 2j/L_i.$$

For large $n+j$ this utilization is approximately $\ln((2n-1)/(n+F(L_i/2 - n)) + 2F(L_i/2 - n)/L_i$. This quantity must be minimized over n satisfying $L_i+1 \le 2n-1 < L_i$. We pick $L_{i-1}+1 = 2n-1$. This results in the utilization being given approximately by

$$R(G_i) = \ln(2G_i) + 1 - G_i$$
$$= \ln 2 + 1 - G_i + \ln G_i.$$

The utilization bound for priority level i is given approximately by

$$U_i(G_i) = \begin{cases} R(G_i) & 1/2 \le G_i \le 1 \\ G_i & 0 \le G_i \le 1/2 \end{cases} \quad (1)$$

We combine the results for all K priority levels to find the final utilization bound:

$$U(G) = \begin{cases} R(G) & 1/2 \le G \le 1 \\ G & 0 \le G \le 1/2 \end{cases} \quad (2)$$

where $G = \text{Min}_{1 \le i \le k} G_i$.

Equation (2) reduces to the Liu and Layland result for $G = 1$, $U(1) = \ln 2$. If $G < 1$, then there is some loss due to having an insufficient number of priority levels as given by the above formula. Figure 4 presents a graph of equation (2) as a function of G. As we can see from Figure 4, once $G > 0.7$, the utilization is not far below ln 2. This issue is discussed further in Section 6.3.

5. The Effect of Multiple Buffering

We have investigated the scheduling bounds when messages packets are single buffered. We now investigate the effect of multiple buffering. In the case of single buffering, denoted by $B = 1$, the deadline of a message is the initiation time of the next message. In the case of double buffering, i.e. $B = 2$, the message deadline becomes the initiation time of the third message. As deadlines are able to be postponed, we can expect improvements in scheduling potential. Indeed, the scheduling potential becomes 1 when the number of buffers is infinite. In this case, there are no more deadlines to be met, and one only needs to keep the bus utilization to be at or

below its capacity of one.

We now investigate the scheduling bounds when $B > 1$. As in the $B = 1$ case, there is more than one possible formulation of the problem. When considering the schedulability of a particular message set, we can compare its utilization to the largest of the available bounds.

5.1. Problem Formulation 1: A Fixed Number of Distinct Periods

This formulation is similar to that of Section 3.2, except $B > 1$. Let $T_1 < ... < T_n$ be n distinct periods and $C_1, ..., C_n$ be the number of single packet messages at each of the respective packet periods. In this formulation, the number of single packet messages or the length of the longest period can be any positive integer. The utilization is given by $U_n = \sum_{i=1}^{n} C_i/T_i$. If we multiply both C_i and T_i by B, then the utilization remains unchanged. Consequently, we assume that both C_i and T_i are divisible by B. We proceed by a series of lemmas which parallel the development given in Section 3.2. The first indicates that we can restrict consideration to the case $(T_n/T_1) < (1 + 1/B)$. If not, then we can find a modified message packet set which also fully utilizes the slots in the critical zone but which has no larger bus utilization.

Lemma 5.1: If a set of message packets has n distinct periods and fully utilizes the critical zone, then there exists a modified message set whose bus utilization is no larger and which also fully utilizes the critical zone.

Proof: We assume $BT_n = qT_1 + r$ with $q \geq B + 1$ and $0 \leq r < T_1$. Consider a modified message set with T_1 replaced by qT_1/B, C_1 unchanged, and C_n increased by $(q-B)C_1/B$. Note that C_1 and T_1 are assumed to be divisible by B. The resulting message set fully utilizes the critical zone; however, the bus utilization is decreased by $(q-B)C_1/qT_1 - (q-B)C_1/(qT_1+r) \geq 0$. □

We now restrict our attention to cases in which $(T_n/T_1) < (1 + 1/B)$, $1 \leq i \leq n$. The next step is to identify choices of C_i and T_i which minimizes the bus utilization. Following the analysis of Section 3.2, we can show that the worst case full utilization message set is given by

$C_i = BT_{i+1} - BT_i, 1 \leq i \leq n-1$

$C_n = (B+1)T_1 - BT_n$.

The bus utilization is given by

$U_n = (\sum_{i=1}^{n-1} T_{i+1}/T_i + (1 + 1/B)T_1/T_n - n)B.$

We must find T_i which minimizes this quantity subject to two conditions: T_i is an integer and $(T_n/T_1) < (1 + 1/B)$. Let $S_i = T_i/T_1$ such that $1 \leq S_i \leq (1 + 1/B)$. We seek to minimize

$U_n = (\sum_{i=1}^{n-1} S_i + (1 + 1/B)/S_n - n)B.$

We temporarily ignore the integer restriction on the S_i. Later one can approximate the optimum by letting T_i grow large as explained in Section 3.2. Using standard calculus, we find U_n/B is minimized if $S_i = (1 + 1/B)^{(i-1)/n}$. This leads to the scheduling bound,

$U_n = nB((1 + 1/B)^{1/n} - 1).$

This bound becomes $B\ln(1 + 1/B)$ as $n \rightarrow \infty$. As we can see the result of Section 3.2 is a special case when $B = 1$. When $B = 2$, the scheduling bound improves 17%, from 0.693 to 0.811. A graph of the limiting utilization bound as a function of B is shown in Figure 5.

5.2. Problem Formulation 2: The Fixed Longest Period

This formulation is similar to that of section 3.1, except $B > 1$. In this formulation, we put a bound on the longest period n and so assume $T_1 \leq ... \leq T_m = n$. We adopt Lemma 5.1 that $n/T_1 \leq (1 + 1/B)$. One can follow the analysis of Section 3.1 to show that the values of C_i which minimizes bus utilization are given by,

$C_i = BT_{i+1} - BT_i, 1 \leq i \leq m - 1$

$C_m = (B + 1)T_1 - BT_m$.

This gives a bus utilization of

$U = (\sum_{i=1}^{m-1} T_{i+1}/T_i + (1 + 1/B)T_1/T_m - m)B.$

We seek to minimize this quantity by picking integers $nB/(1 + B) \leq T_1 < T_2 < ... < T_m = n$. This analysis can no longer be carried out by calculus, since T_i's now are assumed to be integers and T_1 is bounded.

We let $T_1, T_2, ..., T_m$ be given and assume $T_i + 1 < T_{i+1}$ for some i. For the message set to have worst case utilization, C_i must equal $B(T_{i+1} - T_i)$. We now insert a new message having $C_i - B$ packets and period $T_i + 1$. In addition, we change C_i to B. The critical zone is still fully utilized, but the bus utilization is reduced. Consequently, a worst full utilization case must have $T_{i+1} = T_i + 1$. This leads to the following message set as the worst full utilization message set with fixed longest period:

$T_1 = F(nB/(1+B)+1)$, where F(.) is the floor function.

$T_{i+1} = T_i + 1, C_i = B;$

$T_m = n, C_m = (1+B)T_1 - nB.$

Several examples will help clarify the nature of this solution. Suppose that we let $n = 10$ be the longest

period. The following table gives the smallest full bus utilization message set for various values of B.

B	Message Set	Utilization
1	6,7,8,9, 10, 10	0.746
2	7,7,8,8, 9, 9, 10	0.858
3	8,8,8,9, 9,9,10,10	0.908
4	9,9,9,9,10, 10,10,10,10	0.944
5	9,9,9,9,9, 10,10,10,10	0.956

One can see that adding one or two extra buffers (i.e. B = 2 or 3) can significantly increase the worst case bus utilization; however, the benefit diminishes as B increases.

The general formula for the worst case bus scheduling bound is given by,

$$U_n = B\sum_{i=F(nB/(1+B)+1)}^{n-1} 1/i + (1/n)[(1+B)(F(nB/(1+B))+1)-nB]$$

This expression converges to $B\ln(1+1/B)$ as $n \rightarrow \infty$. Consequently, the two formulas agree for large message sets with values for the longest period.

6. Summary

In this section, we summarize the engineering implications of our assumptions and main results.

6.1. The Pacing and Preemption Assumptions

Packet pacing is a technique in which the packet transmission associated with a single message are spread evenly over the message period. This helps to minimize the buffering problem at the bus interface units. Pacing with no loss in bus schedulability is easy to accomplish if the message period is not a prime number. For example, a single message consisting of 3 packets can be replaced by one single packet message with period 5 and a second single packet message with period 10. The bus scheduling potential could suffer, however, if we treat it as a single packet message with period 3. When a message period is a prime number such as a 3 packet message with period 13, we can either treat it as 3 single packet message with period of 13 or one single packet message with period 4. In the former case, the purpose of pacing is seriously compromised. In the latter case, the scheduling potential could suffer, because we replace the original message with single packet messages that appear to have higher bus utilization, 3/12 rather than the original 3/13. Hence, a designer of multiple packet periodic messages should avoid the use of prime numbers as message periods whenever possible. Once messages are packetized as required, the original set of multiple packet messages is replaced with a set of equivalent single packet messages that have equivalent bus utilization.

The second assumption concerns preemption. In bus scheduling, a packet cannot be preempted in the middle of transmission and resumed later. However, the bus schedulability is not adversely affected by this seeming inability to preempt, since the worst case bound is still ln 2. Actually, preemption is never needed because we have assumed that the contention for the bus is synchronized to the beginning of each slot. Furthermore, each packet will be dispatched to contend for the bus with its assigned priority slot by slot starting from its arrival. Under these two assumptions, a higher priority packet cannot be blocked by lower priority ones, and the preemption problem vanishes. Both assumptions can be supported by existing hardware, but it does require moderately sophisticated bus and bus interface unit engineering.

6.2. Scheduling Bounds with Single Buffering and Sufficient Priority Levels

With the additional assumption of sufficient priority levels and single buffering, we have derived the scheduling bounds for bus scheduling potential. A set of single packet message can be characterized in one of three ways: with a fixed longest period, a fixed number of distinct periods or a fixed number of single packet messages. These give us three bounds. All three bounds converge to ln 2 in the worst case. However, for a given set of particular single packet messages, we have three distinct bounds, and the largest bound can be used to determine if this set is schedulable.

These three bounds are:

1. For a fixed longest period n:

$$U_n = \sum_{i=H((n+1)/2)}^{n} 1/i + (1/n)(2H((n+1)/2)-(n+1))$$

2. For a fixed number of distinct periods n:

$U_n = n(2^{1/n} - 1)$.

3. For a fixed number of single packet messages n:

$U_n = \sum_{i=n}^{2n-1} 1/i$.

These three bounds are plotted as a function of the maximal period, number of distinct periods and number of single packet messages in Figure 1, 2, and 3 respectively.

6.3. The Effect of Insufficient Priority Levels

When there are more distinct message packet periods than available priority levels, some message packets with different periods must be assigned the same priority level. This insufficiency of priority levels could lead to a reduction of bus scheduling potential. Suppose that we now have a total number of priority levels given by K and a priority grid (0, L_1, ..., L_K). The highest and lowest priority levels are 1 and K respectively. The shortest and longest periods are 1 and L_K respectively. When a period P falls between two grid boundaries $L_{j-1} < P \leq L_j$, priority j will be assigned. The worst case utilization bound in this situation is given approximately by the formula

$$U = \begin{cases} MIN_{1 \leq i \leq K}(1 - G_i + ln(2G_i)) & 1/2 \leq G_i \leq 1 \\ G_i & 0 \leq G_i \leq 1/2 \end{cases}$$

where $G_i = (L_{i-1} + 1)/L_i$

A simple priority grid that provides good results is the logarithmic design. In this design, the ratio of two adjacent grid lines is constant. For example, if we have a shortest period of 10 slots, a longest period of 24428 slots and 48 priority levels, then G = 0.85 and the grid lines are (10, 11.76, 13.84, ..., 20763.8, 24428), which are rounded to (10, 12, 14, ..., 20764, 24428). In this case, the utilization bound is 0.681, which is close to ln 2, the bound for full priority granularity.

6.4. The Effect of Multiple Buffering

When a message is single buffered (B = 1), the deadline for the message is the initiation time of the next message. When a message is double buffered (B = 2), the deadline becomes the initiation time of the third message. As additional buffers are added, we can expect an increase in the bus scheduling potential.

We carried out two formulations of the multiple buffering case. In the case of fixed distinct periods, the bound is given by,

$U_n = nB((1+1/B)^{1/n} - 1)$, where n denotes the number of distinct periods.

In the case of a fixed longest period, the bound is

$$U_n = B\sum_{i=F(nB/(1+B)+1)}^{n-1} 1/i + (1/n)[(1+B)(F(nB/(1+B))+1) - nB]$$

where n denotes the longest period.

For a given message set, one can use the largest of the two bounds to determine the scheduling potential. The worst case for both bounds converges to the same expression, Bln(1+1/B). The worst case limiting bound as a function of buffers is plotted in Figure 5.

6.5. Future Work

There are additional interesting problems which can be worked out and will be reported elsewhere. These include:

- The optimization of the choice of priority levels when there is a total of K levels.
- The calculation of scheduling potential for the dynamic deadline algorithm when there is an insufficient number of priority levels.
- The optimization of the finite priority grid for the dynamic deadline scheduling algorithm.
- The combination of insufficient priority levels and multiple buffers.

Acknowledgement

This work was sponsored in part by the Office of Naval Researach under contract N00014-84-K-0734 and in part by the Federal Systems Division of IBM Corporation under University Agreement YA-278067, which support the Archons decentralized computer system project at Carnegie-Mellon University, E. Douglas Jensen, Director. The authors also wish to thank George Dailey and Jay Strosnider of IBM Corporation for their contributions to the work.

References

1. Lawler, E. L. "Scheduling Periodically Occurring Tasks on Multiprocessors". *Information Processing Letters 12 (1)* (February, 1981), 9 - 12.

2. Leung, J. Y. and Merrill M. L. "A Note on Preemptive Scheduling of Periodic, Real Time Tasks". *Information Processing Letters 11 (3)* (Nov. 1980), 115 - 118.

3. Liu, C. L. and Layland J. W. "Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment". *JACM 20 (1)* (1973), 46 - 61.

4. Martel, C. "Preemptive Scheduling with Release Times, Deadlines and Due Times". *JACM 29 (3)* (July, 1982), 812 - 829.

5. Mok, A. K. *Fundamental Design Problems of Distributed Systems For The Hard Real Time Environment.* Ph.D. Th., M.I.T., 1983.

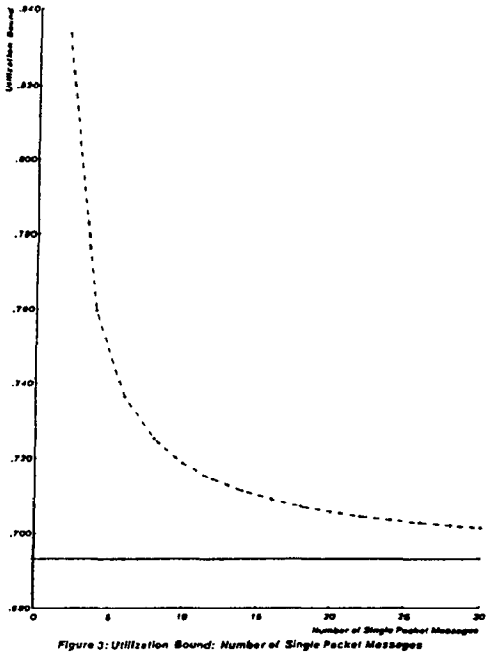

Figure 3: Utilization Bound: Number of Single Packet Messages

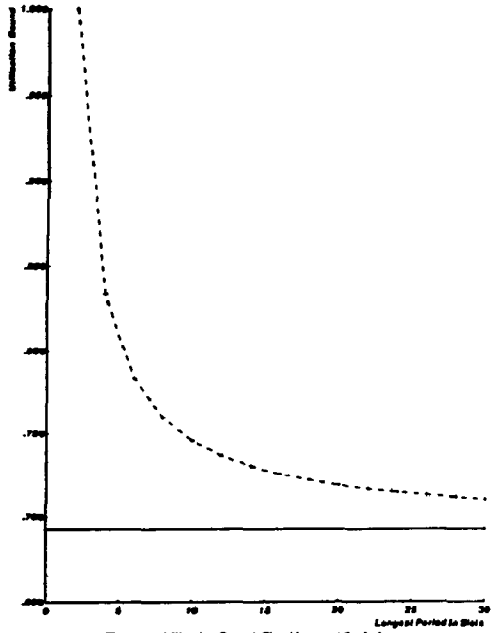

Figure 1: Utilization Bound: Fixed Longest Periods

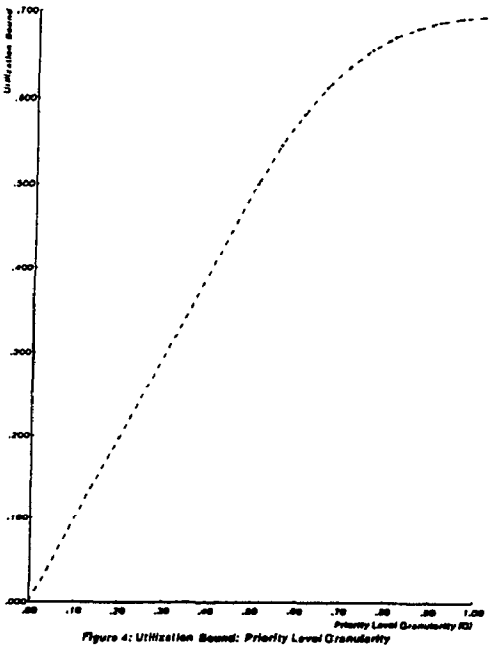

Figure 4: Utilization Bound: Priority Level Granularity

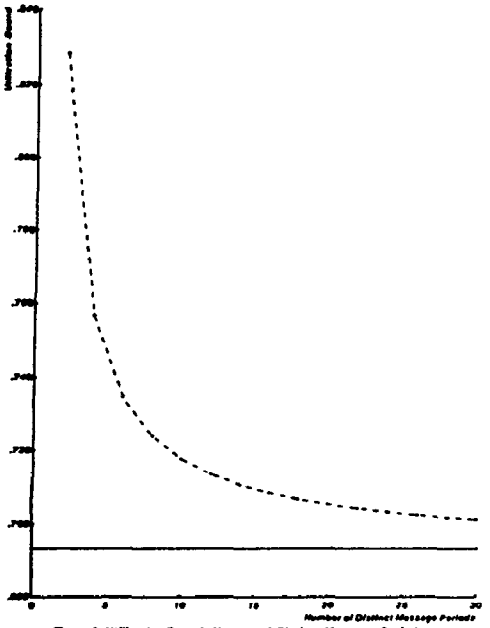

Figure 2: Utilization Bound: Number of Distinct Message Periods

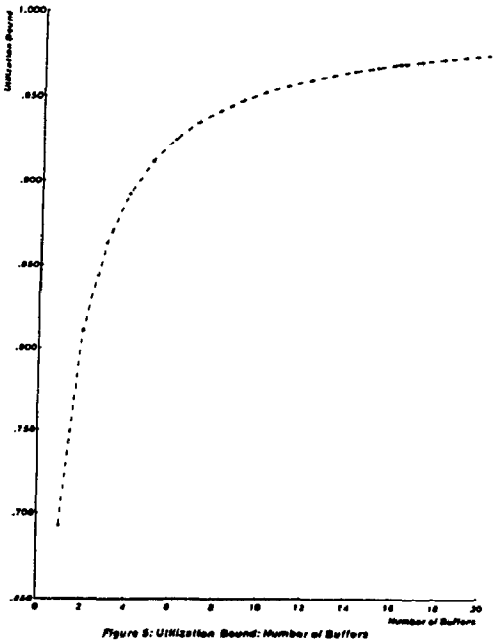

Figure 5: Utilization Bound: Number of Buffers