Performance Evaluation of Control Networks for Manufacturing Systems

F.-L. Lian¹, J. R. Moyne², and D. M. Tilbury¹

1. Department of Mechanical Engineering and Applied Mechanics

2. Department of Electrical Engineering and Computer Science

The University of Michigan, Ann Arbor, Michigan 48109

{fengli, moyne, tilbury}@umich.edu

ABSTRACT There are many different networks which have been promoted for use in manufacturing control systems. In this paper, we discuss the features of three possible candidates: Ethernet (CSMA/CD), ControlNet (Token Bus) and DeviceNet (CAN Bus). We consider how each of these control networks could be used as a communication backbone for a networked control system. A detailed discussion of the medium access control protocol for each network is provided. The medium access control protocol is responsible for providing both the satisfaction of the time-critical/real-time response requirement over the network and the quality and reliability of the communication between devices on the network. For each protocol, we derive the key parameters of the corresponding network when used in a control situation, including bandwidth, magnitude of the expected time delay, and variation in time delays. Simulation results are presented for several different scenarios, and the advantages and disadvantages of each network are summarized.

INTRODUCTION

A current trend in manufacturing control is towards distributed implementations of control systems. When the control algorithms and hardware are physically distributed across the manufacturing system, individual machines (or even parts of a machine) can be run autonomously, which can help in any problem diagnostics or reprogramming that is necessary. These distributed control systems use networks to communicate the data, in contrast to the traditional point-to-point communications with a centralized controller. Thus, the use of networks in a manufacturing control system can reduce the amount of electrical wiring that is required. Not only does this reduce the cost and complexity of a control system, it can also reduce the floor space needed for electrical cabinets to hold all the wiring. This network architecture also makes any changes to the manufacturing system much simpler to implement, since re-wiring is virtually eliminated. Although the network simplifies the wiring of the system, it also introduces a small time delay into the control loop. Before a networked architecture is chosen, the effect of this time delay and any bandwidth limitations on the performance of the control system must be well-understood.

The common features of modern distributed systems are a large number of devices interconnected together to perform the

desired operations, and a large area of coverage. Examples of these systems include industrial automation, building automation, office and home automation, intelligent vehicle systems, and advanced aircraft and spacecraft (Biegacki and VanGompel, 1996; Eccles, 1998; Gibson, 1995; Ray, 1989; Schickhuber and McCarthy, 1997; Song et al., 1996). These networks may be data networks or control networks depending on the information exchanged. Generally speaking, data networks use large data packets and relatively infrequent bursty transmission over a wide area, with high data rates to support the transmission of large data files. Data networks generally do not have hard real-time (i.e. time-critical) constraints. Control networks, in contrast, must shuttle countless small but frequent packets among a relatively large set of nodes to meet the time-critical requirements. The key element that distinguishes control networks from data networks is the capability to support time-critical applications (Raji, 1994).

The performance metrics of network systems that impact the requirements of control systems are access delay, transmission time, response time, message delay, message collisions (percentage of collision), message throughput (percentage of packets discarded), packet size, network utilization, deadline meeting, etc. For control systems, candidate control networks generally must meet two main criteria: bounded time delay and guaranteed transmission, that is, a message should be transmitted successfully within bounded time delay. Unsuccessfully transmitted or large time-delay messages from a sensor to an actuator, for example, may deteriorate system performance or make systems unstable. Several protocols have been proposed to meet these requirements for control systems, including, for example, Ethernet (IEEE 802.3:CSMA/CD), Token Bus (IEEE 802.4), Token Ring (IEEE 802.5), and Controller Area Network (CAN:CSMA/AMP). Control networks are typically based on one of two protocols, CAN (including Smart Distributed System (SDS), DeviceNet (DN97, 1997), and CAN Kingdom), and the Token Ring or Bus (including PROFIBUS (Cena et al., 1997), MAP (Wheelis, 1993), ControlNet (CN97, 1997), and FDDI (Saad-Bouzeffrane and Cottet, 1997)).

In this paper, we present the results of a study of the performance of three candidate control networks: Ethernet (CSMA/CD), ControlNet (Token Bus), and DeviceNet (CAN

Bus). For each network we briefly discuss its medium access mechanism and analyze the timing behavior. First, a discussion of Medium Access Control (MAC) mechanisms of three networks is provided and several comparisons of three control networks are provided based on the requirements of control systems. Then, the timing analysis of three control networks is derived and several performance indexes are provided to evaluate these control networks. A simulation study of message periods and releasing policies is applied to the three control networks. The paper ends with conclusions and future work.

CONTROL NETWORKS BASICS

In this section, we will briefly discuss the Medium Access Control (MAC) protocol of three candidate control networks: Ethernet (CSMA/CD), ControlNet (Token Bus) and DeviceNet (CAN Bus). The MAC protocol is responsible for providing satisfaction of the time critical/real time response requirement over the network and for the quality and the reliability of the communication between network nodes (Koubias and Papadopoulos, 1995).

Ethernet(CSMA/CD):

Ethernet adopts the Carrier Sense Multiple Access with Collision Detection (CSMA/CD) mechanism for resolving contention on the communication medium. The CSMA/CD protocol is specified in the IEEE 802.3 network standard and is described briefly as follows (Bertsekas and Gallager, 1992; Casey, 1990; Tanenbaum, 1996). When a node wants to transmit, it listens to the network. If the network is busy, it waits until the network is idle; otherwise it transmits immediately. If two or more nodes listen to the idle network and decide to transmit simultaneously, the messages of these transmitting nodes collide. While each node is transmitting, it must also listen to detect a collision of messages. On detecting a collision between two or more messages, a transmitting node stops transmitting and waits a random time to re-try its transmission. This random time is determined by the standard Binary Exponential Backoff (BEB) algorithm. The time before trying again is randomly chosen between 0 and $(2^i - 1)$ slot times where i denotes the i th collision event detected by the node. However, after ten collisions have been reached, the interval is frozen at a maximum of 1023 slots. After 16 collisions, the controller stops attempting to transmit and reports failure back to the node microprocessor. Further recovery may be attempted in higher layers (Tanenbaum, 1996).

The Ethernet frame format is shown in Figure 1 (Bartlett and Harvey, 1995; Tanenbaum, 1996). The total overhead is 26 bytes. The size of Data Packet Frame is between 46 and 1500 bytes. There is a non-zero minimum data size requirement because the standard states that valid frames must be at least 64 bytes long, from Destination Address to Checksum. If the data portion of a frame is less than 46 bytes, the Pad field is used to fill out the frame to the minimum size. There are two reasons for this minimum size limitation. Firstly, valid frames are easier to distinguish from "garbage". When a transceiver detects a collision, it truncates the current frame, which means that stray bits and pieces of frames appear on the cable all the time.

Figure 1: Ethernet (CSMA/CD) frame format.

Secondly, it prevents a node from completing the transmission of a short frame before the first bit even reached the far end of cable, where it may collide with another frame (Tanenbaum, 1996).

ControlNet(Token Bus):

MAP, PROFIBUS, and ControlNet are typical examples of the token passing bus control networks. These are deterministic networks because the maximum waiting time before sending a message frame can be characterized by the token rotation time. The token bus protocol (IEEE 802.4) allows a linear, multi-drop, tree-shaped, or segmented topology (Wheelis, 1993). The nodes in the token bus network are arranged logically into a ring, and, in the case of ControlNet, each node knows the address of its predecessor and its successor. During operation of the network, the node with the token transmits data frames until either it runs out of data frames to transmit, or the time it has held the token reaches the token holding time. Then the node re-generates the token and transmits it to its logical successor on the network. If a node has no message to send, it just passes the token to the successor node. Collision of data frames does not occur, as only one node at a time can transmit. The protocol also guarantees a maximum time between network accesses for each node.

The message frame format of ControlNet is shown in Figure 2 (CN97, 1997). The total overhead is 7 bytes, including Preamble, Start Delimiter, Source MAC ID, Cyclic Redundancy Check (or CRC), and End Delimiter. The Data Packet Frame, namely Lpacket or Link Packet Frame, may include several Lpackets which contain the size, control, tag, data and its own destination address with total frame size between 0 and 510 bytes. The size field specifies the number of byte pairs (from 4 to 255) contained in an individual Lpacket. Each byte pair must include the size, control, tag, and link data fields. There are two kinds of tags: fixed tag and generic tag. Each fixed tag must be at least two bytes long. The first byte indicates the service and the second byte is the destination address. The destination address is either the MAC ID of the destination or 0xFF (for broadcast). The generic tag contains a 3-byte connection ID which is used to send data on connections.

The ControlNet protocol adopts an implicit token passing mechanism and assigns a unique MAC ID (from 0 to 99) to each node. Like the general token passing buses, the node with a token has the permission to send data. However, there is no real token passing around the network. Instead, each node monitors the source MAC ID of each message frame received. At the end of a message frame, each node sets an "implicit token register" to the received source MAC ID + 1. If the "implicit token register" is equal to the node's own MAC ID, the that node may now transmit messages. All nodes have the same value in their "implicit token registers" preventing collisions on the medium. If a node has no data to send, it just sends a message with an

Figure 2: The message frame of ControlNet.

Figure 3: Medium access during scheduled, unscheduled and guardband time.

empty Lpacket field, called a null frame.

Each cycle, called a Network Update Time (NUT) in ControlNet or Token Rotation Time (TRT) in general, is divided into three major parts: scheduled, unscheduled, and guardband as shown in Figure 3. During the scheduled part of a NUT, each node can get a chance to transmit time-critical/scheduled data by obtaining the implicit token from 0 to N_{smax} . During the unscheduled part of a NUT, each node from 0 to N_{umax} shares the opportunity to transmit non-time critical data in a round robin fashion, until the allocated unscheduled duration is expired. When the guardband time is reached, all nodes stop transmitting and only the node with lowest MAC ID, called the “moderator”, can transmit a maintenance message, called the “moderator frame”. The moderator frame accomplishes the synchronization of all timers inside each node and publishing of critical link parameters such as NUT, node time, N_{smax} , N_{umax} , etc.

DeviceNet(CAN Bus):

CAN is a serial communication protocol, developed mainly for applications in the automotive industry but also capable of offering good performance in other time-critical industrial applications. The CAN protocol is optimized for short messages and utilizes a Carrier Sense Multiple Access/Arbitration on Message Priority (CSMA/AMP) media access method. Thus the protocol is message oriented and each message has a specific priority which is utilized to gain access to the bus in case of simultaneous transmission. The bit-stream of a transmission is synchronized on the start bit and the arbitration is performed on the message identifier that follows, where a logic zero is dominant over

Figure 4: The message frame format of DeviceNet (Standard CAN format).

a logic one. Any node that wants to transmit a message waits until the bus is free and then starts to send the identifier of its message bit by bit. Conflicts for access to the bus are solved by an arbitration process at the bit level during transmission of the Arbitration field, which is the initial part of each frame. Hence, if two devices want to send messages at the same time, they first continue to send the message frames and then listen to the network. If one of them receives a bit different from the one it sends out, it loses the right to send its message bits and the other wins the arbitration. With this method, an ongoing transmission is never corrupted.

DeviceNet, which is based on the standard CAN specification (11-bit identifier only¹), is a relatively low-cost communication link connecting devices to a network; it has received considerable acceptance in device-level manufacturing applications. In a DeviceNet network, data is transmitted and received using Message Frames which carry data from a transmitting node to one or more receiving nodes. Transmitted data does not necessarily contain addresses of either the source or the destination of the message. Instead, each message is labeled by an identifier that is unique throughout the network. All other nodes on the network receive the message and accept or reject it depending upon the configuration of mask filters for the identifier. This mode of operation is known as multi-cast.

The frame format of DeviceNet is shown in Figure 4 (DN97, 1997). The total overhead is 47 bits which include Start of Frame (SOF), Arbitration (11-bit Identifier), Control, Data, Cyclic Redundancy Check (CRC), Acknowledgment (ACK), End of Frame (EOF), and Intermission (INT) Fields. The size of a Data Field is between 0 and 8 bytes. The DeviceNet protocol uses the arbitration field to provide source and destination addressing as well as message prioritization.

Comparisons of Three Control Networks

In this section, we provide a comparative analysis of three network types. Comparisons of aspects of three typical control networks, Ethernet, ControlNet and DeviceNet are shown in Figure 5. The parameters used in these figures are listed in Table 1.

Figure 5a illustrates that Ethernet requires more overhead to transmit small pieces of data (less than 10 bytes) than either DeviceNet or ControlNet. This is the requirement of a minimum frame size in Ethernet format. However, for large data size message, DeviceNet requires more bits to transmit the same amount of data. This is because DeviceNet can only transmit packets containing a maximum of 8 bytes of data and must therefore

¹The CAN protocol supports two message frame formats: standard CAN (version 2.0A, 11-bit identifier), and extended CAN (version 2.0B, 29-bit identifier).

TABLE 1: System parameters of control networks.

	Ethernet	ControlNet	DeviceNet
Data Rate (Mbps)	10	5	0.5
Bit Time (μs)	0.1	0.2	2
Max Length (m)	2500	1000	100
Max Data Size (byte)	1500	504	8
Min Message Size (byte) ^a	72	7	47/8 ^b
Typical Tx Speed (m/s)	coaxial cable:10 ⁸		

a: zero data size. b: DeviceNet overhead is 47 bits

fragment large data transmissions.

As shown in Figure 5b, the transmission time for DeviceNet, a CAN-based bus, is longer than the others for small data sizes because the data rate (500 Kbps) is slower. For large data sizes, the longer transmission time results from both slow data rate and the large overhead. Ethernet requires less transmission time on larger data sizes (> 20 bytes) compared with the others. Although ControlNet uses less time to transmit the same size of small data, it needs some amount of time (NUT) to gain access to the network.

Figure 5c shows the data coding efficiency of three control networks vs the data size. The data coding efficiency is defined by the ratio of the data size and the message size, i.e. the total number of bytes used to transmit the data. For small data sizes, DeviceNet is the best among these three cases and Ethernet is the worst. For large data size, ControlNet and Ethernet are better than DeviceNet, (DeviceNet is only 58%, but ControlNet and Ethernet are near 98% for large data transmission). For control systems, the data size is generally small. Therefore, the above analysis seems to suggest that DeviceNet may be preferable in spite of the slow data rate. Before making that decision, we must analyze the average and total time delay and the throughput of the network. This is discussed in following section.

The discontinuities seen in Figure 5 are due to the effect of data fragmentation, i.e. the limitation of maximum size per message. The maximum data sizes are 1500, 504, and 8 bytes for Ethernet, ControlNet and DeviceNet, respectively. The flat portions of the Ethernet plots for small data sizes are due to the minimum data size requirement (46 bytes).

TIMING ANALYSIS OF CONTROL NETWORKS

In this section, we will characterize the time delay of these three control networks by studying the timing parameters. Figure 6 shows a general timing diagram of the initialization and ending of a task which requests a message sent by the network.

The time delay of a message, T_{delay} , is defined as the time difference between the time when the source node wants to send a message, T_{src} , and the time when the destination node receives this message, T_{dest} , i.e. $T_{delay} = T_{dest} - T_{src}$.

The total time delay can be broken into three parts: the time delays at the source node, on the network channel, and at the destination node, as shown in Figure 6. The time delay at the source node includes the pre-processing time, T_{pre} , which is the sum of the computation time, T_{scomp} , and the encoding time, T_{scode} , and the waiting time, T_{wait} , which is itself the sum of the queueing time, T_{queue} , and the blocking time, T_{block} . De-

Figure 5: A comparison of system indices vs the data size for each network.

Figure 6: A timing diagram showing time spent sending a message from a source node to a destination node.

pending on the amount of data that the source node must send and the traffic on the network, the waiting time may be significant. The network time delay includes the total transmission time of a message and the propagation delay of the network. This will depend on message size, data rate, and the length of the network cable. The time delay at the destination node only includes the post-processing time, T_{post} , which is the sum of the decoding time, T_{dcode} , and the computation time, T_{dcomp} , at the destination node. Because the pre-processing time and the post-processing time are typically small compared with the waiting time and the transmission time, and are a limitation of computer processing parameters rather than network physical and protocol parameters, we will ignore them in the following discussion.

The time delay can be explicitly expressed by the following equations:

$$T_{delay} = T_{dest} - T_{src} \quad (1)$$

$$= T_{pre} + T_{wait} + T_{tx} + T_{post}, \quad (2)$$

$$= \underbrace{T_{scomp} + T_{scode}}_{T_{pre}} + \underbrace{T_{queue} + T_{block} + T_{frame}}_{T_{wait}} + \underbrace{T_{prop}}_{T_{tx}} + \underbrace{T_{dcode} + T_{dcomp}}_{T_{post}}, \quad (3)$$

$$\approx T_{wait} + T_{tx}. \quad (4)$$

In the following subsections, we analyze the queueing time, blocking time, frame time, and propagation time for each network.

Queueing Time:

The waiting time, $T_{wait} = T_{queue} + T_{block}$, depends on the network protocol and is a major factor in the determinism of a control network. The queueing time, T_{queue} , is the time a message waits in the buffer at the source node before it is ready for sending, i.e. the time required to finish sending any previous messages in queue. The blocking time, T_{block} , is the time that a message must wait once the node is ready to send it. It includes waiting time while other nodes are sending messages and the time needed to re-send the message if messages collide. The queueing time depends on both the blocking time of previous messages in the queue and the periodicity of the messages.

Blocking Time:

Ethernet Blocking Time: We first consider the blocking time for Ethernet. Recall that the blocking time, T_{block} , is the time taken by collisions with other messages and the subsequent time waiting to be re-transmitted. The BEB algorithm described indicates a probabilistic waiting time. An exact analysis of expected blocking time delay for Ethernet is very difficult, but it is instructive to analyze an approximate model. From the discuss in (Lian et al., 1999), the expected blocking time, $E\{T_{block}\}$, can be expressed by the following equation:

$$E\{T_{block}\} = [(g_1 + g_2 + \dots + g_{15})Q_r + g_{16}]T_{slot} + T_{resid}, \quad (5)$$

where g_k is the time taken from the releasing time of a message to the k th-backoff time, Q_r is the probability that the current

node is the only node attempting to send a backlogged message in any given slot, T_{slot} is the time needed for a round trip transmission, and T_{resid} denotes the residual time which is seen by node i until the network is idle. For the 16th collision, the node discards this message and reports an error message to the higher level processing units (Tanenbaum, 1996). It can be seen that T_{block} is not deterministic and may be unbounded because of the discarding of messages.

ControlNet Blocking Time: Next, we consider the ControlNet protocol. For a token bus protocol, if a node wants to send a message, it must wait to receive the token from the logically previous node. Therefore, the blocking time, T_{block} can be expressed by the transmission time and token rotation time of previous nodes. The general formula for T_{block} can be described by the following equation:

$$T_{block} = T_{resid} + \sum_{j \in \mathcal{N}_{noqueue}} T_{token}^{(j)} + \sum_{j \in \mathcal{N}_{queue}} T_{tx}^{(j, n_j)}, \quad (6)$$

where T_{resid} is residual time needed by the current node to finish transmitting, and $\mathcal{N}_{noqueue}$ and $\mathcal{N}_{queue}$ denote the sets of nodes with messages and without messages in the queues, respectively. Let n_j denote the number of messages queued in the j th node and let T_{node} be the maximum possible time, i.e. token holding time, assigned to each node to fully utilize the network channel, for example, $T_{node} = 255 \mu s$ in ControlNet. T_{token} is the token passing time, the time needed to transmit a token. The token passing time depends on the time needed to transmit a token and the propagation time from node $i - 1$ to node i . ControlNet uses an implicit token and T_{token} is nothing but the sum of T_{frame} with zero data size and T_{prop} . If a message is requested while the node is holding the token, then $T_{block} = T_{tx}^{(j, n_j)}$ where j is the node number. In the worst case, if there are N master nodes on the bus and each one has multiple messages to send, which means each node utilizes the maximum token holding time, then $T_{block} = (N - 1) \min(T_{tx}^{(j, n_j)}, T_{node})$ where the min function is used because even if it has more messages to send, a node cannot hold the token longer than T_{node} , i.e. $T_{tx}^{(j, n_j)} \leq T_{node}$. ControlNet is a deterministic network because the maximum time delay is bounded and can be characterized by Equation (6). If the periods of each node and message are known, then we can explicitly describe the sets $\mathcal{N}_{noqueue}$ and $\mathcal{N}_{queue}$ and n_j . Hence, T_{block} in Equation (6) can be determined explicitly.

DeviceNet Blocking Time: The blocking time, T_{block} in DeviceNet can be described by the following equation:

$$T_{block}^{(k)} = T_{resid} + \sum_{\forall j \in \mathcal{N}_{hp}} \lceil \frac{T_{block}^{(k-1)} + T_{bit}}{T_{peri}^{(j)}} \rceil T_{tx}^{(j)}, \quad (7)$$

where T_{resid} is residual time needed by the current node to finish transmitting, $\mathcal{N}_{hp}$ is the set of nodes with higher priority than the waiting node, and $T_{peri}^{(j)}$ is the period of the j th message. The summation denotes the time needed to send all the higher-priority messages. A low-priority node has to wait until all high-priority nodes have finished transmitting all messages in their

queues. This situation increases the total blocking time. The worst-case T_{resid} under a low traffic load is:

$$T_{resid} = \max_{\forall j \in \mathcal{N}_{node}} T_{tx}^{(j)}, \quad (8)$$

where $\mathcal{N}_{node}$ is the set of nodes on the network. However, because of the priority-arbitration mechanism, low priority nodes/messages may not be deterministic under high loading of high priority nodes/messages. Hence, the residual time for low priority nodes/messages under high traffic load may be unbounded because these nodes/messages have no chance of gaining access to the network and will be queued at the buffer.

Frame Time:

The frame time, T_{frame} , depends on the size of the data, overhead, any padding, and the bit time. Let N_{data} be the size of data in terms of bytes, N_{ovhd} be the number of bytes used as overhead, N_{pad} be the number of bytes used to pad the remaining part of frame and meet the requirement of minimum frame size, and N_{stuff} be the number of bytes used in a stuffing mechanism (on some protocols), then the frame time can be expressed by the following equation:

$$T_{frame} = [N_{data} + N_{ovhd} + N_{pad} + N_{stuff}] \times 8 \times T_{bit}, \quad (9)$$

The values, N_{data} , N_{ovhd} , N_{pad} and N_{stuff} will be explicitly described based on the Ethernet, ControlNet and DeviceNet protocols shown in Figures 1, 2 and 4. The detailed formulae of determining the number of bytes of N 's are described in (Lian et al., 1999).

Propagation Time:

The propagation time, T_{prop} , depends on the signal transmission speed and the distance between the source and destination nodes. For the worst case, the propagation delay from one end to the other end of the network cable for these three control networks are: $T_{prop} = 25.6\mu s$ for Ethernet (2500 m), $T_{prop} = 10\mu s$ for ControlNet (1000 m), and $T_{prop} = 1\mu s$ for DeviceNet (100 m). The length in parenthesis represents the typical maximum cable length used. The propagation delay is not easily characterized because the distance between the source and destination nodes is not constant for different transmissions. For comparison, we will assume that the propagation time of these three network types are the same, say, $T_{prop} = 1\mu s$ (100 m). Note that T_{prop} in DeviceNet is generally less than one bit time because DeviceNet is a synchronized network. Hence, its shortest maximum length is used to guarantee the bit synchronization among nodes.

PERFORMANCE EVALUATION

In this section, we will derive several indices to evaluate the performance of these three control networks.

Average Time Delays: For a given running time, say $T_{max} = 10sec$, we can calculate the average time delays from each node

for each network.

$$T_{delay}^{avg} = \frac{1}{N} \sum_{i \in \mathcal{N}_{node}} \left[\frac{\sum_{j=1}^{M^{(i)}} T_{delay}^{(i,j)}}{M^{(i)}} \right], \quad (10)$$

where N is the total number of nodes, $\mathcal{N}_{node}$ is the set of nodes, and $M^{(i)}$ the number of messages requested at node i . We assume all messages are periodic, thus the total number of messages is equal to the total running time divided by the period of messages, i.e. $M^{(i)} = \lfloor T_{max}/T_{peri}^{(i)} \rfloor$ where $\lfloor x \rfloor$ denotes the largest integer number less than x . The average delay can be computed for the entire network, as shown, or for any node.

Efficiency of Networks: We will define the efficiency of a network, P_{eff} , as the ratio of the total data size to the time used to send messages containing data, that is,

$$P_{eff} = \frac{\sum_{i \in \mathcal{N}_{node}} \sum_{j=1}^{M^{(i)}} T_{tx}^{(i,j)}}{T_{delay}^{sum}}. \quad (11)$$

Therefore, $P_{eff} \rightarrow 1$ denotes that all the time delay is from the transmission delay and the network performance is good. On the other hand, $P_{eff} \rightarrow 0$ means that most of the time is due to message contention or collision.

Utilization of Networks: The utilization of a network, P_{util} , is defined by the ratio of the total time used to transmit data and the total running time, that is,

$$P_{util} = \frac{\sum_{i \in \mathcal{N}_{node}} \sum_{j=1}^{M^{(i)}} T_{tx}^{(i,j)}}{T_{max}}. \quad (12)$$

P_{util} describes the percentage of network bandwidth used by the nodes or, on the other hand, the useful utilization of the network. If $P_{util} \rightarrow 1$, then the network is saturated, and we have to re-design the network layout or re-assign the traffic load. If $P_{util} \rightarrow 0$, then there is plenty of bandwidth left on the network for other purposes.

Number of Un-sent Messages: Control systems need information to be transmitted successfully and immediately. If the information transmission on a network induces lost messages, it may deteriorate system performance and make the system unstable. Hence, it is very important to evaluate a network protocol according to the number or the possibility of un-sent messages.

Stability of Control Networks:

The stability of a control network is defined by the number of messages in the queue of each node. If the number tends to infinity as time increases, then this network is unstable (even though we assume infinite buffer size in our analysis).

CASE STUDY: 10 NODES, EQUAL PERIODS

In this section, we present a system with 10 nodes and assume each node sends 8 bytes of data every time period. We

consider three releasing policies for sending this data over the network, and compare the three network performances. The simulation time is 0.5 *sec* for each case.

The three releasing policies are as follows. The first one, which we call the “zero releasing policy,” assumes every node tries to send its first message at $t = 0$ and sends a new message every period. This type of situation happens when a system powers up and there has been no pre-scheduling of messages. The second policy, namely the “random releasing policy,” assumes a random start time for each node, each node still sends a new message every period. The third policy, namely the “scheduled releasing policy,” considers a releasing policy which is scheduled to be the best for individual networks, that is, the start-sending time is scheduled to occur (as much as possible) when the network is available to the node. In addition to varying the release policy, we also change the period of each node. This will demonstrate the effect of traffic load on the network. For each releasing policy and period, we calculate the average time delay and the total time delay of these ten nodes, and the efficiency and the utilization of the three different control networks.

Table 2 presents the performance comparison of the zero releasing policy for four different periods; The result for the other periods and releasing policies can be found in (Lian et al., 1999). The periods were chosen based on the total time to send all messages over different networks. For example, the total time to send all 10 messages is 576, 320, and 2200 (μs), for Ethernet, ControlNet, and DeviceNet, respectively. As shown in Table 2, when the period of messages is larger than the total message transmission time, the performance indices, such as average message time delay, total time delay, and utilization, are similar. However, when the period is smaller than the total transmission time, i.e. traffic load is larger than the availability of network, then the utilization approaches 100% and the time delay will become unbounded as time goes infinity. Therefore, the network system becomes unstable. For the two deterministic control networks, i.e. DeviceNet and ControlNet, if the traffic load is not saturated, i.e. $P_{util} \leq 100\%$, there are no messages queued at the buffer. However, for Ethernet, there are almost always some messages queued at the buffer; moreover, some messages have been discarded due to the BEB algorithm. The number of discarded messages is listed in the Failure row. We also notice that the average time delay of each message is not constant, even though the network is under saturated. The discarded messages and non-constant time delay make Ethernet unsuitable in this loading situation for control applications.

A graphical comparison of Table 2 for the three control networks is shown in Figure 7. Figure 7a illustrates that when the network is saturated, the average time delay of the three networks is increasing. However, before reaching saturation, the average time delay is almost the same. For the zero releasing policy, the network efficiency, shown in Figure 7b, is low due to a significant amount of message contention occurring at the time zero. However, using the scheduled releasing policy (Lian et al., 1999), the efficiency of all three control networks can reach 100% if the traffic load is not saturated. The result of the network utilization and the number of un-sent messages is shown in Figures 7c and 7d. Figure 7c shows that when the network is

TABLE 2: Simulation result of the zero releasing policy.

Period (μs)	5000	2000	500	300
Frequency (Hz)	200	500	2000	3333
Average time delay (μs)				
Ethernet	1081	4573	$1.8e+5$	$3.6e+5$
ControlNet	241	241	242	433
DeviceNet	1221	28749	$8.6e+5$	$1.6e+6$
Efficiency (%)				
Ethernet	5.3	1.3	$3.2e-2$	$1.6e-2$
ControlNet	13.3	13.3	13.2	5.8
DeviceNet	18.2	0.8	$2.6e-2$	$1.4e-2$
Utilization (%)				
Ethernet	11.6	29.1	69.4	80.4
ControlNet	6.4	16.0	64.0	84.1
DeviceNet	44.4	99.9	99.9	99.9
Un-sent messages				
Ethernet	0	11	4043	9720
ControlNet	0	0	0	2
DeviceNet	0	247	7747	14407
Failure messages				
Ethernet	0	0	3	6

saturated, DeviceNet gives priority to the messages with lower MAC ID, i.e. higher priority nodes, so, the network utilization is 100%. However, in this situation, the messages with higher MAC ID, i.e. lower priority nodes, are queued at the buffer and cannot get access to the network medium. ControlNet has a similar problem when the network is saturated. However, for Ethernet, the network utilization approaches 100% slowly even when the network is saturated. This is because the BEB algorithm lets the nodes which have collided wait for some amount of time. This phenomenon is also shown in Figure 7d where Ethernet and DeviceNet have a lot of unsent messages. For DeviceNet, it is because low-priority messages cannot gain access to the medium. For Ethernet, it is because nodes attempt to re-send collided messages again and again. However, for ControlNet, due to the available bandwidth of each node, a node with multiple messages can send the messages when it has the implicit token. Hence, the number of unsent message is lower than the others.

CONCLUSION AND FUTURE WORK

This paper provides a detailed discussion of the features of three candidate control networks: Ethernet (CSMA/CD), ControlNet (Token Bus) and DeviceNet (CAN) for manufacturing systems. We first described the medium access control mechanism of these protocols which is responsible for providing both the satisfaction of the time-critical/real-time response requirement over the network, and the quality and reliability of communication between devices on the network. These timing parameters, which will influence the ultimate control applications, are affected by the network bandwidth/data rate, the period of messages, the data or message size of the information, and the communication protocol. For each protocol, we derived the key elements of the corresponding network when used in a control situation, including bandwidth, magnitude of the expected time delay, and variation in time delays. Simulation results were presented for several different scenarios. The timing analyses and comparisons of message time delay which are given in this paper will be useful for anyone designing a networked control system.

The evaluation study of this paper is based on the char-

Figure 7: A comparison of system performance vs the frequency (Hz) of messages.

acteristics and the requirements of control systems. The characteristics are small data sizes and periodic messages, and the requirements are the guaranteed transmission and the bounded time delay. The suitability of network systems for control systems is greatly influenced by these two criteria. Although Ethernet is widely used for many data transmission and can support high data rates up to 1 Gbps, it is generally not suitable for the communication in control systems when compared with other deterministic network systems. However, because of its high data rate, Ethernet can be used for non-time-critical and large data size messages such as communication between workstations or machine cells. For intra-workstation cell communication with controller, sensors, and actuators, deterministic network systems are more suitable to meet the characteristics and requirements of control systems. For those control systems with short and/or prioritized messages, DeviceNet demonstrates better performance. The scheduled and unscheduled messaging capabilities in ControlNet make it suitable for time-critical and non-time-critical messages. ControlNet is also suitable for the large data size message transmission.

Controller design for networked control systems can differ significantly from the design of traditional control systems. Our future efforts will focus on controller design for networked con-

trol systems. This will include conducting experimental studies of control networks for control applications. We then plan to use this analysis along with performance evaluation of candidate control networks as the basis for future message scheduling and control algorithm designs for networked control systems.

ACKNOWLEDGMENTS

The authors would like to acknowledge the contributions of John Korsakas and the support from the Open DeviceNet Vendor Association, ControlNet International, and Allen-Bradley.

*

REFERENCES

- Bartlett, H. and Harvey, J., 1995, "The Modeling and Simulation of a Pick and Place Computer-Integrated Manufacturing (CIM) Cell," *Computers in Industry*, Vol. 26, No. 3, pp. 253-260.
- Bertsekas, D. and Gallager, R., 1992, *Data Networks*, 2nd ed., Prentice-Hall.
- Biegacki, S. and VanGompel, D., 1996, "The Application of DeviceNet in Process Control," *ISA Transactions*, Vol. 35, No. 2, pp. 169-176.
- Casey, B. J., 1990, "Implementing Ethernet in the Industrial Environment," in: *IEEE Industry Applications Society Annual Meeting*, Vol. 2, pp. 1469-1477.
- Cena, G., Demartini, C., and Valenzano, A., 1997, "On the Performances of Two Popular Fieldbuses," in: *IEEE Int'l Workshop on Factory Comm. Syst.*, pp. 177-186.
- CN97, 1997, *ControlNet Specifications*, ControlNet Int'l, 1st ed.
- DN97, 1997, *DeviceNet Specifications*, Open DeviceNet Vendors Association, 2nd ed.
- Eccles, L. H., 1998, "A Smart Sensor Bus for Data Acquisition," *Sensors*, Vol. 15, No. 3, pp. 28-36.
- Gibson, L. D., 1995, "Autonomous Control with Peer-to-Peer I/O Networks," *Sensors*, Vol. 12, No. 9, pp. 83-90.
- Koren, Y., Pasek, Z. J., Ulsoy, A. G., and Benchetrit, U., 1996, "Real-time Open Control Architectures and System Performance," *CIRP Annals*, Vol. 45, No. 1, pp. 377-380.
- Koubias, S. A. and Papadopoulos, G. D., 1995, "Modern Fieldbus Communication Architectures for Real-Time Industrial Applications," *Computers in Industry*, Vol. 26, No. 3, pp. 243-252.
- Lian, F.-L., Moyne, J. R., and Tilbury, D. M., 1999, "Performance Evaluation of Control Networks: Ethernet, ControlNet, and DeviceNet," Tech. Rep. MEAM-99-02, Univ. of Michigan.
- Paula, G., 1997, "Building a Better Fieldbus," *Mechanical Engineering*, pp. 90-92.
- Raji, R. S., 1994, "Smart Networks for Control," *IEEE Spectrum*, Vol. 31, No. 6, pp. 49-55.
- Ray, A., 1989, "Introduction to Networking for Integrated Control Systems," *IEEE Control Sys. Mag.*, Vol. 9, No. 1, pp. 76-79.
- Saad-Bouzeffrane, S. and Cottet, F., 1997, "A Performance Analysis of Distributed Hard-Real Time Applications," in: *IEEE Int'l Workshop on Factory Comm. Sys.*, pp. 167-176.
- Schickhuber, G. and McCarthy, O., 1997, "Distributed Fieldbus and Control Network Systems," *Comp. & Control Eng.*, Vol. 8, No. 1, pp. 21-32.
- Song, D., Divoux, T., and Lepage, F., 1996, "Design of the Distributed Architecture of a Machine-Tool Using FIP Fieldbus," in: *IEEE Int'l Conf. on Application-Specific Systems, Architectures, Processors*, pp. 250-260.
- Tanenbaum, A. S., 1996, *Computer Networks*, 3rd ed., Prentice-Hall Inc.
- Wheelis, J. D., 1993, "Process Control Communications: Token Bus, CSMA/CD, or Token Ring?" *ISA Transactions*, Vol. 32, No. 2, pp. 193-198.