Modelling and analysis of a data communication protocol for integrated control of advanced aircraft

Arun Ayyagari and Asok Ray

Integrated Control System (ICS) networks in future generation aircraft must coordinate mission, avionic, subsystem utility, engine control, and flight control and management functions to improve aircraft performance and utilization of available resources. This paper presents modelling and performance analysis of the *ARINC 629* protocol which is potentially a standard for data communications in advanced commercial aircraft. Finite-state-machine models and timed Petri net models of *ARINC 629* under the Basic Protocol (BP) and the Combined Protocol (CP) modes have been developed. The timed Petri net model also serves to formulate the structure of a discrete-event simulation model for performance analysis of the communication network and the integrated control system.

Keywords: distributed digital avionics, data communication networks, integrated control

Future-generation advanced aircraft are envisioned to require communications between various avionic processing elements, sensors, data processors, signal processors, servo-actuators and display devices. The interconnection of multi-vendor equipment across a single communication network enhances the concept of common avionics at the system level. This distributed processing imposes performance and functionality requirements on the network to guarantee that the delivery of flight-critical and mission-critical data is timely and reliable.

Integrated Control Systems (ICS) for advanced aircraft coordinate and execute diverse but interrelated functions[1, 2], which include vehicle control, monitoring and warning, flight management, navigation, communications and surveillance, aircraft systems management, and aircraft and systems support.

Functions related to the Vehicle Management System (VMS), which include control and monitoring of aerodynamic, structural and propulsion systems, are often identified as *flight critical*, and the rest as *avionics*. In the current state-of-the-art, the flight critical and avionic functions are designed to operate largely independent of each other. However, in future generation aircraft, the VMS will encompass integrated control functions (e.g. integrated flight-propulsion control) and flight management systems (e.g. trajectory management) that will have direct flight-criticality implications. These functions, combined with new strategies (e.g. self-repairing and reconfigurable flight control systems, management of actuator failures and surface damage, control surface reconfiguration and applications of AI techniques to distributed decision support systems), would generate significantly large and distributed computational requirements.

Both flight-critical and avionic functions are crucial for establishing computer communication system requirements in future-generation aircraft, of which data rate and induced delays are the most important. Dynamics of control surfaces that determine inherent aerodynamic characteristics of the airframe on the basis of flight sensor inputs are highly sensitive to delays. (Nominal data rates of 100 samples per second and a maximum delay of 20 ms can be considered as representative design criteria for stability and command augmentation functions for a conventional aircraft.) On the other hand, flutter control may be relatively less sensitive to delays but needs a high data rate, for example, 160 samples per second in conventional aircraft. The communication system in an advanced

---

Mechanical Engineering Department, The Pennsylvania State University, University Park, PA 16802, USA (email: A2R@ECL.PSU.edu)

Paper received: 22 September 1991; revised paper received: 7 February 1992
This work was supported in part by National Science Foundation under Research grant No. DDM-9015173.

0140-3664/93/060350-16

aircraft is responsible for supplying the engine electronics with air data and information for engine instrumentation. Therefore, the data rate and transport delay requirements from engine sensor inputs to fuel flow control outputs are also crucial.

A communication network is needed for information processing on and between a number of onboard and spatially dispersed computers, intelligent terminals, sensors and actuators to implement the above functions. Basic methods for networking that are available for interconnecting the system components[1,2] are broadly classified as point-to-point dedicated connections, and multiplexed data communication networks.

The major advantages of multiplexed networks over the conventional point-to-point dedicated connections include:

- reduced wiring mass and power requirements
- flexibility of operations
- evolutionary design process
- improved maintenance, diagnostics, and monitoring.

However, the network-induced delays have the potential of significantly degrading the dynamic performance of some of the fast control loops (e.g. structural and aerodynamic) even if the average offered traffic is maintained well below the saturation limit of the network. Contrary to the concept of a dedicated bus that serves a limited number of flight critical functions, ICSs in future generation aircraft would be subjected to randomly varying network-induced delays[3-5], in addition to the sampling time and data processing delays that are inherent in digital control systems. Furthermore, the ICS could be subjected to recurrent loss of data due to noise corruption in the communication medium and malfunction of the network protocol. Therefore, timely arrival of transmitted data from the source (e.g. sensor terminal) to the destination (e.g. controller) is not assured at all times.

Potentially viable protocols for ICS networks in advanced transport aircraft include MIL-STD-1553B[6] (or its optical equivalent MIL-STD-1773), AS4074 Linear Token Passing Bus (LTPB)[7], and *ARINC 629*[8] that belong to the respective categories of bus access control: (i) centralized; (ii) distributed; and (iii) contention/reservation. Bus-induced delays in the contention/reservation scheme are generally predictable because of a static transmission schedule and cycle times. However, the scheme suffers from large delays as the bus bandwidth is likely to be wasted or underutilized. The centralized scheme that operates on simple poll/response algorithms induces relatively smaller delays at the expense of being less reliable. The delays induced by distributed token passing are difficult to compute, although they are deterministic under non-random traffic. The linear token passing protocol bus (LTPB) has the advantage of smaller induced delays than both centralized and contention/reservation protocols because of low overhead at the individual stations. Brief descriptions of the MIL-STD-1553B, LPTB and *ARINC 629* protocols are presented below.

The MIL-STD-1553B protocol has a centralized bus controller (BC) that manages all communications in a command/response mode and ensures data integrity. The BC initiates the network functions and establishes the communication paths for data transfer between the remote terminals (RT). MIL-STD-1553B has been extensively used for distributed digital avionic systems of military aircraft.

The LTPB protocol has no central bus controller. It operates in a distributed mode in the sense that bus access control is passed from one station to another in a round-robin fashion around a logical ring. A station, upon receiving a special frame, called a *token*, may transmit a waiting message and then pass on the token to its successor on the logical ring in a predefined sequence. A station with no waiting messages may have the ability to remove itself from the logical ring to avoid the delay during each token passing cycle. Timers are used to ensure a fair management of the data bus and to detect network failures. The LTPB protocol has been considered for digital flight control systems in military aircraft[9]. The LTPB and MIL-STD-1553B protocols were systematically evaluated with respect to a set of performance criteria that include data latency and throughput[10].

The *ARINC 629* protocol uses a collision avoidance technique, i.e. it attempts to ensure that no two terminals simultaneously transmit their messages. Each terminal is synchronized relative to the other terminals on the bus such that it relinquishes control of the medium upon transmission of messages. Each terminal has a unique timer interval based on the network characteristics and the scheduled transmission sequence for the terminals. *ARINC 629* is capable of efficient handling of synchronous transmissions provided that asynchronous message transmissions are inhibited. Message retries and asynchronous message transmissions must be handled by increasing the transmission cycle beyond that required for the synchronous transmissions.

The *ARINC 629* protocol is a revised version of DATAC[11], and is under consideration for advanced transport aircraft in the commercial sector. The Boeing 777 subsonic jet transport and the PAVE PACE concept on a Boeing 737 aircraft, envisioned by NASA, are expected to implement *ARINC 629* for avionic systems integration. Although *ARINC 629* has been designed to provide a high level of reliability, its performance analysis in view of delay and throughput has not apparently been reported. Therefore, it is timely that *ARINC 629* be systematically analysed and evaluated relative to other avionic network protocols.

The objective of this paper is to report modelling and analysis of the *ARINC 629* protocol in view of its use in the communication network for integrated control of advanced aircraft. Emphasis is laid on: (i) providing the background necessary for understanding the protocol; (ii) development of finite-state-machine and timed Petri net models to describe the protocol operations in detail; and (iii) presentation of the results of discrete-event simulation in terms of network-induced delay statistics and throughput.

ARINC 629 PROTOCOL DESCRIPTION

ARINC 629 is a carrier-sense multiple access protocol with bus topology, and functions in the broadcast mode with a finite number of terminals. A functional description of *ARINC 629* is given in Appendix A, and further details of message formatting and scheduling are given in the protocol's specification[8]. The scheduling mechanism of *ARINC 629* is formulated on the principle of *collision avoidance*. This is essentially accomplished with the aid of a number of timers that are located at each terminal. Each timer has three states – *running*, *reset* and *elapse*. The *running* state implies that a timer has been started and is currently in the running state. The *reset* state implies that the clock for a timer is reset to be started. The *elapse* state indicates expiry of a timer after the preset interval or by any external interrupt signal, whichever occurs first. A terminal is permitted to transmit a message based on the states of these timers. *ARINC 629* is designed to operate in two modes, Basic Protocol (BP) and Combined Protocol (CP).

BP mode

The BP mode gives equal access to both periodic and aperiodic data at a terminal. The access to the transmission medium is established by a distributed medium access control protocol among all the terminals on the bus. A terminal is given an opportunity to access the medium once in every cycle. A collision-free cyclic access mechanism is implemented by use of three timers in each terminal to schedule the bus operation.

- *Transmit Interval (TI) Timer: TI* is reset and immediately started at the instant the terminal is permitted to access the medium. The nominal settings of *TI* range from 1000 to 128,000 units of bit time (BT), where one unit of BT is 0.5 μs at a transmission rate of 2 Mbit/s. *TI* is identically set at all terminals.
- *Synchronization Gap (SG) Timer: SG* is started at the instant no channel activity is observed. If channel activity is observed by a terminal before its *SG* has elapsed, then *SG* is reset. Once *SG* has elapsed, it is reset only when the terminal is permitted to access the medium. Typical settings of *SG* are 16, 32, 64 and 127 units of BT. Since the setting of *SG* may result in a transport delay, it should be maintained as small as possible. *SG* is identically set at all terminals.
- *Terminal Gap (TG) Timer: TG* at a terminal is started only when its *SG* has elapsed and no channel activity is observed. If any activity is observed by a terminal while its *TG* is in the running or in the elapsed state, then *TG* is reset. *TG* is also reset when the terminal is permitted to access the medium. Typical settings for *TG* range from 1 through 126 units of BT in steps of one. Since the setting of *TG* may result in a transportation delay, *TG* at the individual terminal is recommended to be set starting from the smallest value. The setting of *TG* at the individual terminal is unique, and the largest *TG* setting must be less than the *SG* setting of the network.

Remark 1: Since *TG* is allowed to be started only after *SG* has elapsed, *TG* and *SG* at a terminal are not expected to run concurrently. □

Remark 2: Since *TI* is started immediately after it is reset, *TI* has essentially two states: *running* and *elapse*. In contrast, both *SG* and *TG* have three states: *running*, *reset* and *elapse*. □

A terminal is permitted to access the medium only when its *TI*, *SG* and *TG* are in the elapsed state. At this stage, the terminal resets all three timers and starts *TI*, and transmission of the backlogged message(s) follows. Upon completion of the message transmission, *SG* is started. On the contrary, if this terminal does not have any backlogged message, then the terminal with the next higher *TG* setting would possibly take its place. This situation generates a potential problem of large variations in network-induced delays. Therefore, as an implementation issue, the buffer of an *ARINC 629* terminal with periodic data may not be allowed to be empty at the instant of all three timers being in the elapsed state. However, as this criterion is not explicitly stated in the *ARINC 629* specifications[8], the protocol has been modelled to include scenarios where the data buffer is empty at the instant of all three timers being in the elapsed state.

Remark 3: A terminal with only aperiodic data may find its buffer empty when all three timers are in elapsed state. In that case, the terminal with the next higher *TG* setting would be given access to the medium. Therefore, to overcome the problem of large variations in network induced delays, the terminals with periodic data may be allocated smaller *TG* values while higher *TG* values are terminals with only aperiodic data. □

The protocol in the BP mode operates in two submodes, namely, periodic and aperiodic.

- *Periodic submode:* All terminals on the bus complete transmissions of their respective message(s) in a particular cycle before their *TI*s have elapsed. Therefore, in this mode every terminal is assured a periodic access to the transmission medium in this mode. The sequence in which the terminals in a particular cycle gain access to the medium is based on the power-up instants of the individual terminals, at which the respective *TI*s are started.

- *Aperiodic submode:* Since *TI* of a terminal is in the elapsed state at the completion of the cycle, all terminals on the network may not complete their respective message(s) transmissions in a particular cycle. The sequence in which the terminals in a particular cycle gain access to the medium is based on the ascending order of their *TG* settings. However, the terminals are not assured a periodic access to the transmission medium in this mode.

CP mode

The CP mode is an enhancement of the BP mode in the previous section. Unlike BP, CP provides for priority access for periodic data transmissions (level 1) and two lower priority access levels of aperiodic data transmissions. Level 2 is intended for short (e.g. possibly less than 10 words), infrequent, high priority message transmissions. Level 3 data are usually non-time-critical (e.g. data for maintenance and service functions) and rather long (limited to 257 words).

The periodic data transmissions from each terminal are concatenated and transmitted forming a periodic transmission train. The remaining bandwidth of the bus following periodic data transmissions is available for aperiodic data transmissions. Transmissions from terminals with backlogged aperiodic data at level 2 are given preference over the backlogged aperiodic data at level 3 during the left-over time.

The frequency of periodic data transmissions at priority level 1 is ensured by a timer-based preemption mechanism at every terminal. For example, a terminal with backlogged periodic data is given an opportunity to access the medium once in every cycle, but no such guarantee exists for a terminal with backlogged aperiodic data. Aperiodic messages are structured into blocks that are relatively shorter than the permissible blocks in the BP mode to accommodate for their transmission within the time available between periodic data transmissions. A collision-free, prioritized, cyclic access mechanism is implemented by use of five timers in each terminal to schedule the bus operations, as explained below.

- *Transmit Interval (TI) Timer: TI* is used only for level 1 periodic data transmissions. *TI* is started at each terminal at the onset of level 1 data transmission. However, *TI* is active only if the terminal is the first to commence a level 1 transmission in a particular cycle: in this case, *TI* elapses naturally. As the first terminal starts its transmission beginning with a Concatenation Event (CE) signal, natural elapse of *TI*s in the remaining terminals is preempted. Nominal settings of *TI* range from 1000 to 128,000 units of bit time (BT), where one BT is 0.5 μs at a transmission rate of 2 Mbit/s. The setting of *TI* at all terminals must be identical.

Remark 4: The Concatenation Event (CE) signal transmitted at the beginning of the first periodic data transmission in a particular cycle ensures formation of a periodic data transmission train. The remaining bus time is available for aperiodic data transmissions in a continuous block. □

- *Aperiodic Synchronization Gap (ASG) Timer: ASG*, used only for levels 2 and 3, is started at the instant no channel activity is observed. If any channel activity is observed by a terminal before its *ASG* has elapsed, then *ASG* is reset. Once *ASG* has elapsed, it is reset only if one of the following two conditions hold:

  a transition in access level, e.g. level 1 to level 2
  a transition to permit new or repeat access at level 3.

  In addition to determining the transitions to each level, *ASG* ensures equal access to all terminals with level 3 messages, irrespective of time constraints and number of periodic cycles encompassed. Typical settings of *ASG* are 16, 32, 64 and 127 units of BT. Since the setting of *ASG* may result in a transport delay, it should be maintained as small as possible. The *ASG* setting for all terminals must be identical.
- *Periodic Synchronization Gap (PSG) Timer: PSG* initiates a transmission cycle beginning with periodic data at level 1 if the bus is idle, and is reset whenever any channel activity is observed. *PSG* is identically set to a length of five times *ASG* at all terminals to ensure a sufficient gap between transmissions within a particular cycle.
- *Terminal Gap (TG) Timer: TG* at a terminal is started only when its *ASG* has elapsed and no channel activity is observed. If any activity is observed by a terminal while its *TG* is in the running or elapsed state, then *TG* is reset. *TG* is also reset when the terminal is permitted to access the medium. Typical settings of *TG* range from 1 through 126 units of BT in steps of one. Since the setting of *TG* results in a transportation delay, *TG* at any individual terminal is recommended to be set as small as possible. The setting of *TG* at each individual terminal must be unique, and the largest *TG* setting in the network should be less than the *ASG* setting.

Remark 5: TG can be started only after *ASG* has elapsed. However, unlike the BP mode, *TG* and *ASG* at a terminal are expected to run concurrently at levels 2 and 3, and during the period between termination of level 3 and commencement of level 1 because *ASG* is used to initiate and terminate an operation at various levels and sublevels. □

- *Aperiodic Access Time-out (AT) Timer: AT* ensures timely return of control at level 1 access for periodic data transmission by preventing the start of aperiodic data transmissions that would jeopardize the start of the subsequent cycle. The timer is reset and started with transmission of periodic data at each cycle. The start of the first periodic data transmission at each cycle is identified by either the detection of a CE signal or when *TI* elapses, whichever is the sooner.

Remark 6: TI and *AT* are started immediately after they are reset. Therefore, *TI* and *AT* have essentially two states: *running* and *elapse*. In contrast, *ASG*, *PSG* and *TG* have three states: *running*, *reset* and *elapse*. □

The protocol in the CP mode operates in three levels of bus access priority: level 1 (periodic transmission) and level 2 and 3 (aperiodic transmission).

- *Level 1 (Periodic Transmission):* If *TI* is in the elapsed state and the bus is detected to be idle for one unit of *PSG*, then a terminal issues a level 1 'go-ahead' signal (interpreted as the CE signal by the rest of the terminals), resets and starts its *AT* and *TI*, and initiates transmission of periodic data. However, if the CE signal is detected before *TI* has elapsed, *TI* is preempted and set to the elapsed state, *AT* is reset and started, and the timers associated with aperiodic transmission are reset. The level 1 operation is terminated and the level 2 operation is initiated when a terminal detects no bus activity for a period of one *ASG*.

Remark 7: Reset and start of *TI* at each terminal ensure that no further periodic data is transmitted by the particular terminal until *TI* has elapsed or the CE signal is detected. □

- *Level 2 (Aperiodic Transmission):* If no bus activity is detected for a period of *TG* after initiation of the level 2 operation, a terminal with backlogged (level 2) aperiodic data may issue a go-ahead signal and starts transmitting. Any terminal is given an opportunity to transmit data only once in a given cycle. The level 2 operation is terminated, and the level 3 operation is initiated when the terminal detects no bus activity for a period of one *ASG*. However, if *AT* elapses, the level 2 operation is terminated and the terminal must wait for the next cycle. Terminals that do not gain access to the medium to transmit their backlogged aperiodic data may compete again for access in the subsequent cycle.
- *Level 3 (Aperiodic Transmission):* If no bus activity is detected for a period of *TG* after initiation of the level 3 operation, a terminal with backlogged aperiodic data issues a level 3 go-ahead signal and starts transmitting. The terminal ignores new requests until all the level 3 backlogged messages have been serviced. The indication that all backlogged messages have been serviced is derived by detecting no bus activity for a period of one *ASG*. At this time, new level 3 requests are entered as backlogged messages, and each of these terminals may once again be given an opportunity to transmit provided that no bus activity is detected for a period equal to the respective *TG*. The level 3 operation is terminated following the service of all backlogged level 3 messages if no bus activity is reported for a period of one *ASG*, and the terminal awaits the commencement of a new level 1 operation. If *AT* elapses, level 3 operation is terminated and the terminal awaits the commencement of level 1 operation. However, if *AT* elapses while the terminal is transmitting aperiodic data at level 2 (or level 3), then the operation cannot be terminated before the end of message transmission, i.e. an ongoing message transmission is not preempted. (Note: If the bus time is available, level 3 allows repeated access by the terminals having waiting message(s).)

The terminals that are unable to transmit their backlogged aperiodic data may re-compete for access based on its *TG* in the subsequent cycle(s); and new arrivals are held in the buffers for transmission in any subsequent cycle. To this effect, level 3 operates in two priority sublevels. The sublevel 1 contains those messages that were denied transmission in previous cycle(s); and the new messages belong to the sublevel 2 which has a lower priority relative to sublevel 1.

ARINC 629 PROTOCOL MODELLING

Operations of the BP mode are described in this section along with a finite state machine (FSM) model[12, 13] and a Petri net model[14]. While the FSM model is more convenient than the Petri net for understanding of the basic operations of the protocol, it is only the Petri net model that does represent concurrence of temporal events. The operations of the CP mode, albeit more complex than those of the BP mode, have similar characteristics. An FSM model of the CP mode operations is described in Appendix B, along with an FSM model. A Petri net model of the CP mode, which is structurally similar to that of the BP mode, is not presented in this paper due to space limitations.

FSM model of BP mode operations

The access mechanism is represented by a finite state machine (FSM) model as shown in *Figure 1*, where a state is represented by S_j, where j denotes the state number. Every terminal is assumed to have one buffer in this model, and the logic variables for protocol operations at a terminal are listed in *Table 1*. The states of the finite-state-machine model are then defined in terms of these logic variables, as listed in *Table 2*, and the corresponding state transitions are given in *Table 3*. The operations of the protocol in the BP mode are briefly explained below with reference to *Figure 1* and *Tables 1–3*:

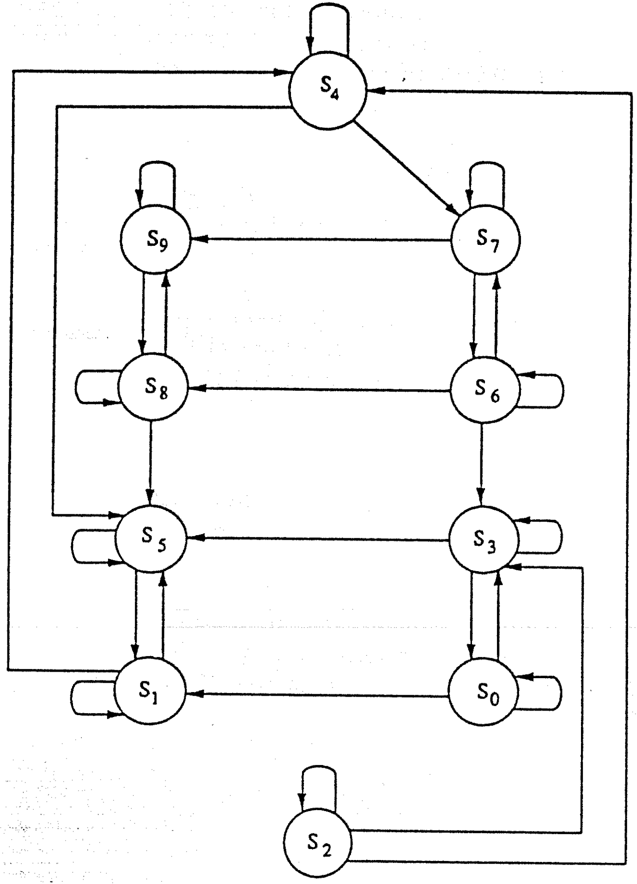


Figure 1 Finite state machine model of *ARINC 629* in BP mode

Table 1 Definition of logic variables in BP mode

TI_RU: Running state of *TI* timer;
TI_E: Elapsed state of *TI* timer;

SG_RU: Running state of *SG* timer;
SG_E: Elapsed state of *SG* timer;
SG_RE: Reset state of *SG* timer;

TG_RU: Running state of *TG* timer;
TG_E: Elapsed state of *TG* timer;
TG_RE: Reset state of *TG* timer;

Table 2 Description of states in BP mode

S_0: $TI\_RU \cap SG\_E \cap TG\_RU$
S_1: $TI\_E \cap SG\_E \cap TG\_RU$
S_2: $TI\_RU \cap SG\_E \cap TG\_E$
S_3: $TI\_RU \cap SG\_E \cap TG\_RE$
S_4: $TI\_E \cap SG\_E \cap TG\_E$
S_5: $TI\_E \cap SG\_E \cap TG\_RE$
S_6: $TI\_RU \cap SG\_RU \cap TG\_RE$
S_7: $TI\_RU \cap SG\_RE \cap TG\_RE$
S_8: $TI\_E \cap SG\_RU \cap TG\_RE$
S_9: $TI\_E \cap SG\_RE \cap TG\_RE$

Table 3 Description of state transitions in BP mode

S_0/S_0: $TI\_RU \cap TG\_RU$
S_0/S_1: TI_E
S_0/S_2: TG_E
S_0/S_3: TG_RE
S_1/S_1: TG_RU
S_1/S_4: TG_E
S_1/S_5: TG_RE
S_2/S_2: TI_RU
S_2/S_3: TG_RE
S_2/S_4: TI_E
S_3/S_3: $TI\_RU \cap TG\_RE$
S_3/S_0: TG_RU
S_3/S_5: TI_E
S_4/S_4: $TI\_E \cap SG\_E \cap TG\_E$
S_4/S_5: TG_RE
S_4/S_7: $TI\_RU \cap SG\_RE \cap TG\_RE$
S_5/S_5: TG_RE
S_5/S_1: TG_RU
S_6/S_6: $TI\_RU \cap SG\_RU \cap TG\_RE$
S_6/S_3: SG_E
S_6/S_7: SG_RE
S_6/S_8: TI_E
S_7/S_7: $TI\_RU \cap SG\_RE \cap TG\_RE$
S_7/S_6: SG_RU
S_7/S_9: TI_E
S_8/S_8: $SG\_RU \cap TG\_RE$
S_8/S_5: SG_E
S_8/S_9: SG_RE
S_9/S_9: $SG\_RE \cap TG\_RE$
S_9/S_8: SG_RU

- A terminal in S_0 has its *TI* and *TG* timers running. The running state of the *TG* timer indicates that the transmission medium is idle.
- A terminal in S_1 has its *TG* timer running. The running state of the *TG* timer indicates that the transmission medium is idle.
- A terminal in S_2 has its *TI* timer running. The *SG* and *TG* timers are in elapsed state. The elapsed state of the *SG* timer while the *TI* timer is running indicates that the protocol operation is in the periodic mode.
- A terminal in S_3 has its *TI* timer running and the *TG* timer is in reset state. The reset state of the *TG* timer indicates that the terminal is currently detecting bus activity.
- All timers are elapsed in S_4. This enables the terminal to access the medium to transmit any of its backlogged messages.
- *TG* timer is reset, and *TI* and *SG* are elapsed in S_5. The reset state of the *TG* timer indicates that the terminal is currently detecting bus activity.
- *TI* and *SG* timers running and *TG* is reset in S_6. The terminal has already gained access to the medium in the current cycle and is not currently detecting bus activity.
- *TI* is running, and the *SG* and *TG* are reset in S_7. The terminal is either transmitting its backlogged message(s) or has already gained access to the medium in the current cycle, and is currently detecting bus activity.
- *SG* is running and *TI* is elapsed in S_8. This indicates that the transmission medium is idle, and the protocol is operating in the aperiodic mode.
- *TG* is running, and *SG* and *TG* are reset in S_9. This indicates that the terminal is currently detecting bus activity, and the protocol is operating in the aperiodic mode.

Timed Petri net model of protocol operations in BP mode

Petri net modelling provides a systematic approach to evaluation and verification of communication protocols where multiple events occur concurrently. The basic concepts of Petri net modelling are outlined in

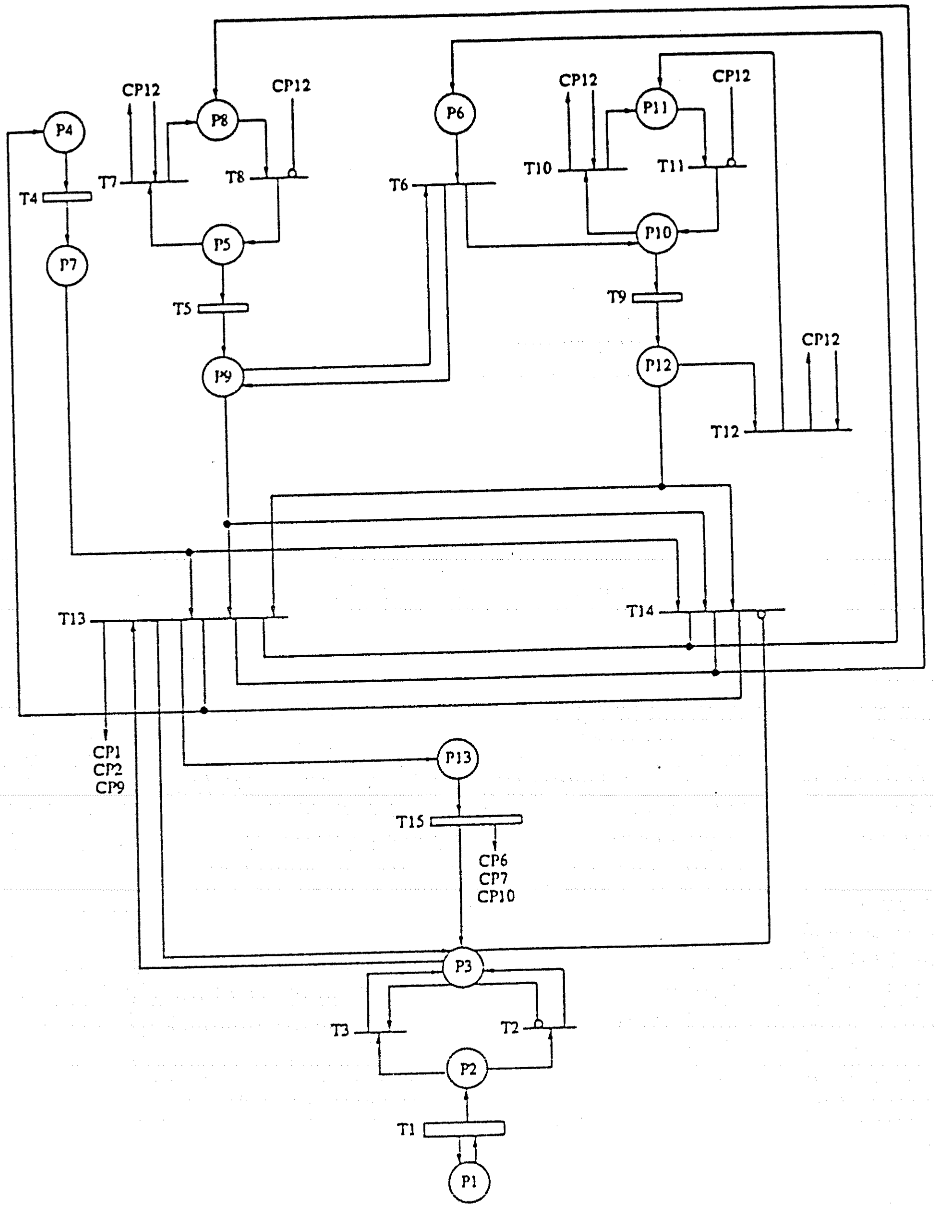


Figure 2 Timed Petri net model of *ARINC 629* terminal in BP mode

Appendix C. The timed Petri net approach has been used to model the *ARINC 629* protocol operation in both BP and CP modes, and their correctness has been verified following the methodology described by Merlin[15].

Since the medium access control (MAC) mechanism is identical for each terminal, only one terminal suffices to be considered for modelling the MAC layer protocol. The protocol operation is represented by two interacting models: (1) operations within a terminal where the place @ is denoted as *P@* and the transition \$ as *T\$*, and (2) communication channel where the place @ is denoted as *P@* and the transition as *T\$*.

The firing time t_f, which represents the processing delay of the protocol, is usually very small compared to the enabling time t_e, which includes the propagation delay and message transmission time. Therefore, t_f for all transitions is assumed to be zero. Furthermore, for convenience of explaining the protocol operations, the buffer capacity at a terminal is limited to one in the timed Petri net model. The access mechanism of the protocol, shown in *Figure 2*, consists of two parts: (1) message generation and (2) arbitration for medium access.

The message generation mechanism is initiated by inserting a token in place *P1*. This enables the timed transition *T1* representing the inter-arrival periods of the messages. The firing of *T1* deposits a token at *P1* and *P2*. The deposition of a token back into *P1* respawns the process of message generation. The message buffer capacity of one is modelled by places *P2* and *P3*, and transitions *T2* and *T3*. If *P3* contains a token indicating that the buffer is full and *P2* has a token indicating that a new message has arrived, then

T3 is enabled and fires, removing a token from *P2* and *P3*, and depositing a token back into *P3*. This conserves the token at *P3* but destroys the token at *P2*, representing a message rejection as the buffer is full. If the buffer is empty, i.e. *P3* does not have a token and *P2* has a token, *T2* would immediately fire resulting in removal of a token from *P2* and deposit of a token into *P3*.

Arbitration for medium access is established by the timers *TI*, *SG* and *TG*. Places *P4*, *P7* and transition *T4* model the *TI* operation. Similarly places *P5*, *P8*, *P9* and transitions *T5*, *T7*, *T8* model the *SG* operation. The *TG* operation is modelled by places *P6*, *P9*, *P10*, *P11*, *P12* and transitions *T6*, *T9*, *T10*, *T11*, *T12*.

A token at *P4* indicates that *TI* is running. A token at *P4* enables the timed transition *T4* which represents *TG*. As *T4* is enabled and fires, a token is removed from *P4* and deposited into *P7* after a delay indicating that the *TI* is elapsed.

A token at *P5*, indicating that *SG* is running, enables the timed transition *T5*. As *T5* is enabled and fires, a token is removed from *P5* and deposited into *P9* after a delay indicating that *SG* is elapsed. On the contrary, while *SG* is running, if any channel activity is detected, transition *T7* is enabled and fires removing a token from *P5* and *CP12* depositing a token into *P8* and *CP12*. A token in *P8* indicates that *SG* is reset. While *SG* is reset if no channel activity is detected, transition *T8* is enabled and fires removing a token from *P8* and depositing it into *P5*.

A token at *P6* indicates that *TG* is reset, and *SG* is not elapsed. As *T6* is enabled and fires, i.e. *SG* is elapsed, a token is removed from *P6* and *P9* and deposited into *P9* and *P10*. A token at *P10* indicates that *TG* is running. A token at *P10* enables the timed transition *T9*, which represents *TG*. As *T9* is enabled and fires, a token is removed from *P10* and deposited into *P12*. A token at *P12* indicates that *TG* is elapsed. On the contrary, while *TG* is running, if channel activity is detected, transition *T10* is enabled and fires, removing a token from *P10* and *CP12* and depositing a token into *P11* and *CP12*. A token in *P11* indicates that *TG* is reset and *SG* is elapsed. While *TG* is reset if no channel activity is detected, transition *T11* is enabled and fires, removing a token from *P11* and depositing a token into *P10*. Upon detection of channel activity in the elapsed state of *TG* (i.e. *P12* having a token), transition *T12* is enabled and fires removing a token from *P12* and *CP12* and depositing a token into *P11* and *CP12*.

A terminal is allowed to access the transmission medium when *TI*, *SG* and *TG* are elapsed, i.e. places *P7*, *P9* and *P12* contain a token. At this instant either of the conflicting transitions *T13* or *T14* is enabled depending on the status of the buffer. *T13* is enabled and fires when the terminal has a backlogged message i.e. *P3* has a token removing a token from *P3*, *P7*, *P9* and *P12* and depositing a token into *P3*, *P4*, *P8*, *P6*, *P13*, *CP1*, *CP2* and *CP9*. A token at *P13* indicates that the terminal is currently transmitting its backlogged message. A token at *P13* enables the timed transition *T15*, which represents the total transmission time for the message. As *T15* is enabled and fires, a token is removed from *P3* and *P13* and deposited into *CP6*, *CP7* and *CP10*, indicating the end of the transmission. On the contrary if the buffer is empty, i.e. *P3* does not have a token, transition *T14* is enabled and fires, removing a token from *P7*, *P9* and *P12*, and depositing a token into *P4*, *P8* and *P6*.

Timed Petri net model of channel operations in BP mode

The channel submodel, shown in *Figure 3*, describes the terminal's connection with and propagation delays between its immediate neighbours on the left and right. The timed transitions, *CTL1* through *CTL4* and *CTR1* through *CTR4*, represent the propagation delays between their respective left and right neighbours. The start of a transmission onto the left and right is modelled by places *CP1* and *CP2*. Similarly, the end of a transmission onto the left and right is modelled by places *CP5* through *CP8*. *CP3* and *CP4* model the behaviour of the channel as sensed by a terminal on the left and right, respectively. The existence of a token at *CP3* and *CP4* indicates that the terminal has received a start of transmission signal. When a start of transmission signal is transmitted by a terminal, a token is deposited into *CP1*, *CP2* and *CP9*. Similarly, at the end of a transmission, a token is deposited into *CP6*, *CP7* and *CP10*. The set of places *CP9* through *CP11* and transitions *CT7* and *CT8* represent the channel state due to the activity of the terminal. A token at *CP11* indicates that the terminal is currently accessing the medium. *CP12* and the set of transitions *CT9* through *CT12* represent the overall state of the channel as observed by the terminal. The presence of a token at *CP12* indicates that the terminal is detecting channel activity either due to its own transmission or due to the transmission from any other terminal on the bus.

SIMULATION MODEL OF THE PROTOCOL

Performance of *ARINC 629* was evaluated via simulation of different scenarios of aircraft operations. To this effect, combined discrete-event and continuous-time simulation models for the BP and CP modes of *ARINC 629* were coded in Fortran in the environment of SIMAN[16]. A simulation model of *ARINC 629* directly follows the timed Petri net model, and is made up of two submodels: terminal and channel (C) activities. The terminal submodel, in turn, is divided into message generation and protocol operation subsystems. The message generation subsystem schedules the arrival of messages at each terminal, while the protocol operation subsystem defines the medium arbitration mechanism of the protocol without imposing any modelling approximations.

The discrete-event simulation model consists of

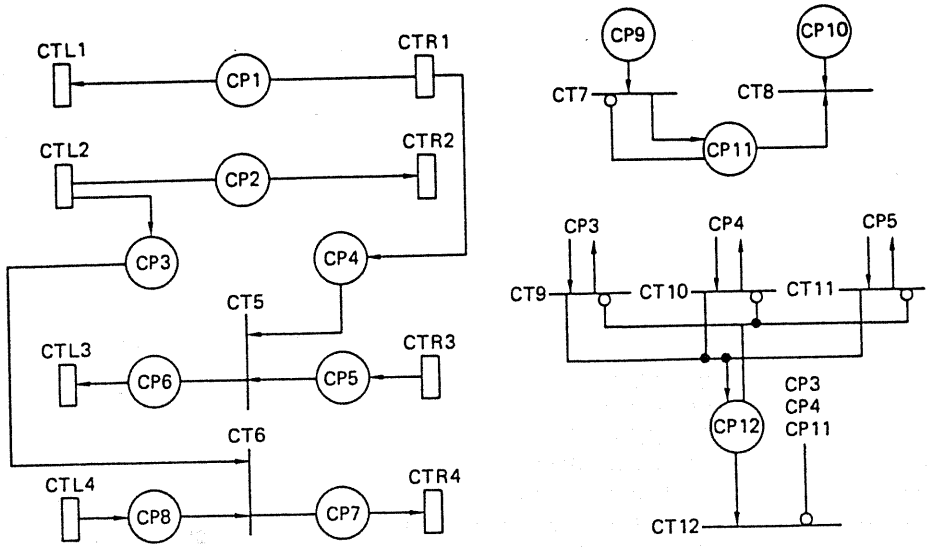
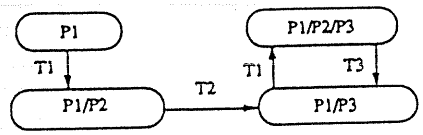


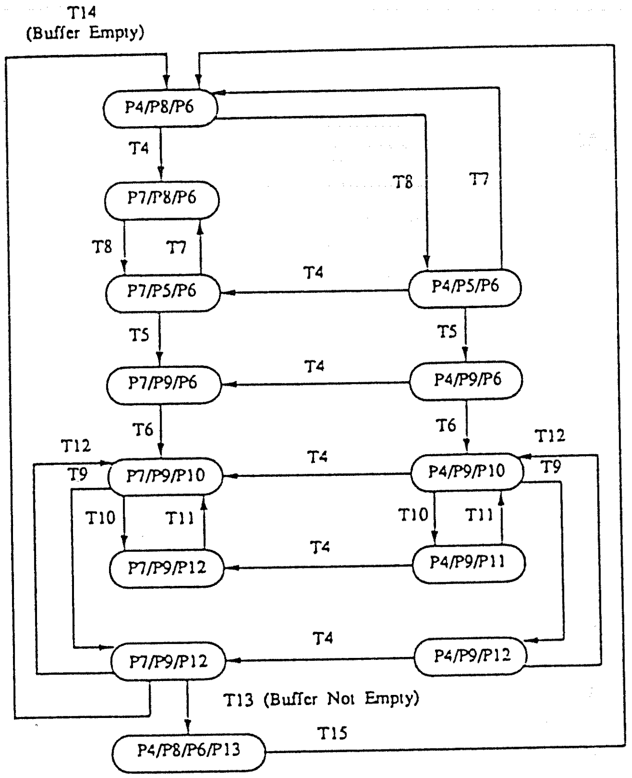
Figure 3 Timed Petri net model of *ARINC 629* channel in BP mode

several events. The token machine of the BP mode, shown in *Figure 4*, is formulated on the basis of the timed Petri net model in the previous section. Following *Figure 4*, the sequence of discrete events in the simulation model of the BP mode is described below.

1 *Initialization:* the simulation and network parameters for a given traffic condition are defined by this event.
2 *Message generation:* this event reschedules next message generation instant according to the given message inter-arrival time. The timed transition *T1* continuously generates new messages with a given distribution and puts a token into *P2*. Unlike the timed Petri net model, the simulation model message buffer capacity has not been restricted to units. Attributes are assigned to each message entity.
3 *TI running:* this event resets and starts the *TI* timer by firing *T13* or *T14* and depositing a token at *P4*.
4 *TI elapse:* the firing of *T4* puts a token into *P7* indicating that *TI* timer has elapsed.
5 *SG running:* this event indicates that the *SG* timer is running and is indicated by the presence of a token at *P5* and enabling *T5*.
6 *SG elapse:* the firing of *T5* depositing a token into *P9* indicates that *SG* timer has elapsed.
7 *SG reset: SG* timer is reset upon the firing of *T7*, *T13* or *T14* and depositing a token into *P8*.
8 *TG running:* this event indicates that the *TG* timer is running, and is indicated by the presence of a token at *P10* and enabling *T9*.
6 *TG elapse:* the firing of *T9* depositing a token into *P12* indicates that the *TG* timer has elapsed.
7 *TG reset: TG* timer is reset upon the firing of *T10* or *T13* or *T14* and depositing a token into *P6* or *P11*.
8 *Message transmit:* this event schedules the beginning and the end of message transmission by a terminal.



Message Generation Submodel



Protocol Operation Submodel

Figure 4 Token machine of *ARINC 629* terminal operation in BP mode

Transitions *T13*, *T15* and places *P3*, *P13* model this event. In addition, it also schedules the delivery of the message at the destination terminal using the channel submodel of the timed Petri net model.

9 *Message receive:* this event resets the *SG* and *TG* timers if they are in a running state by firing transitions *T7*, *T10* or *T12*. The indication of the bus activity required for this event is derived from the channel model discussed earlier.

The control system simulation model can now be developed under a combined continuous-time discrete-event environment[3, 10] using a standard simulation language[16]. Whereas discrete-event models are suitable for representing the time-ordered sequences of network operations, continuous-time models are essential for solving the initial value problems associated with aircraft control systems. Coupling of these two types of simulation permits evaluation of the network as a dynamic element of the closed loop control system, and yields a numerical solution to the set of coupled differential-difference equations. This procedure would allow the design engineer to monitor the effects of network-induced delays on the dynamic performance of the aircraft where the individual models of different protocols and aircraft control systems can be replaced without any structural modifications of the simulation program.

Simulation results and discussion

Performance of *ARINC 629* was evaluated in terms of offered traffic, throughput, and statistics of queueing delays. (Definitions of these terms are provided in Appendix D.) Results were generated by varying the normalized offered traffic in the network. The parameters used in the simulation models and their values are presented below:

BP mode

- Number of stations in the network: 10
- Data transmission rate = 2 Mbit/s
- The normalized offered traffic is increased from 0.01 to 1.00
  - Message queue capacity = 1
  - Traffic loading at all nodes is balanced
  - Message arrival is fixed and periodic = 10 ms
  - Average message length is updated according to offered traffic.

CP mode

- Number of stations in the network: 10
- Data transmission rate = 2 Mbit/s
- The normalized offered traffic is increased from 0.21 to 1.00
  - Message queue capacity = 1
  - Traffic loading at all nodes is balanced
  - Message arrival is fixed and periodic = 10 ms
  - Offered traffic at level 1 is fixed = 0.10
  - Offered traffic at level 2 is fixed = 0.10
  - Offered traffic at level 3 is increased from 0.01 to 0.80
  - Average message length is updated according to offered traffic.

Simulation results for throughput and statistics of queueing delays *versus* normalized offered traffic are presented in *Figures 5–8* for both BP and CP modes.

Figure 5 shows a comparison of statistical averages of the queueing delay under different operational modes of *ARINC 629*. As the offered traffic is increased, the average queueing delay for BP mode gradually rises. In contrast, the average queueing delay for levels 1 and 2 of CP mode remain constant because the offered traffic at levels 1 and 2 are fixed, and their transmission schedules are not affected by increasing the total offered traffic in the network. However, the average queueing delay for level 3 of CP mode shows an exponential increase with increasing offered traffic. This increase results from periodic preemption caused by *AT* timers (described earlier) at level 3 transmissions to maintain the periodicity of level 1 transmissions. *Figures 6* and *7* present comparisons of the standard deviation and range (i.e. the difference between maximum and minimum values) of queueing delay, respectively. The delay characteristics of levels 1 and 2 of CP mode remain invariant because the respective offered traffic and transmission schedules are fixed and periodic. In contrast, for BP mode and level 3 of CP mode, queueing delay statistics monotonically increase with offered traffic. Standard deviation of queueing delays could cause significant fluctuations in network-induced delays, and may cause jitters in dynamic response[5]. *Figure 8* presents a comparison of the normalized throughput under BP and CP modes. A maximum normalized throughput of 0.77 is achieved

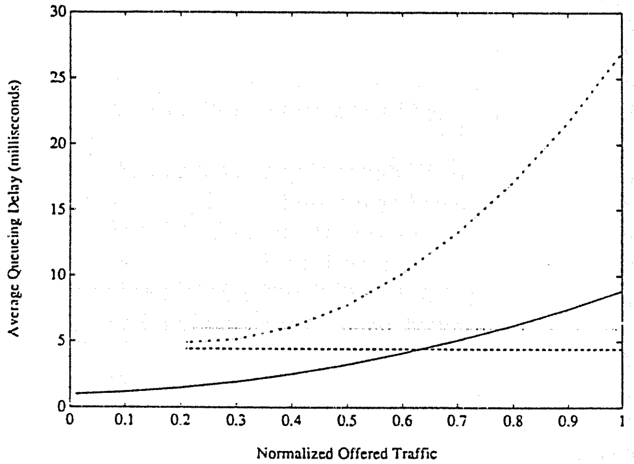


Figure 5 Average queueing delay *versus* normalized offered traffic. ——: BP mode; - - -: CP mode level 1; · · · · · ·: CP mode level 2; —·—·—: CP mode level 3

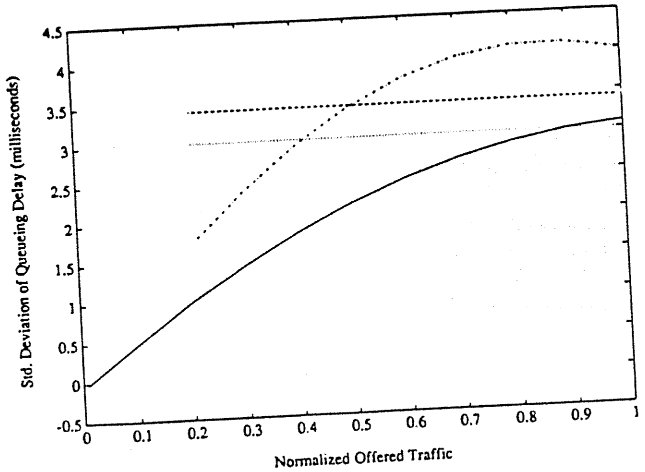


Figure 6 Standard deviation of queueing delay *versus* normalized offered traffic (for key see *Figure 5* caption)

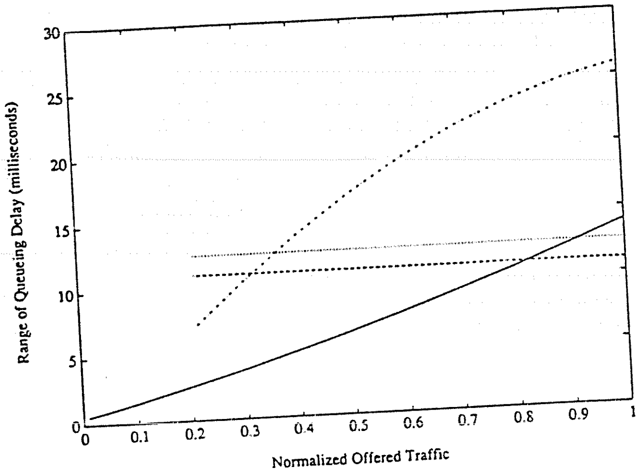


Figure 7 Range of queueing delay *versus* normalized offered traffic (for key see *Figure 5* caption). Range: Maximum–minimum

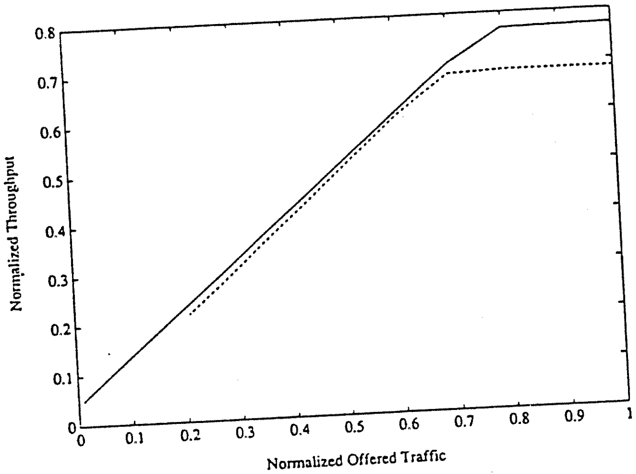


Figure 8 Normalized throughput *versus* normalized offered traffic. —: BP mode: · · · ·: CP mode

in BP mode in contrast to 0.68 in CP mode. The lower throughput of CP mode results from the higher overheads associated with additional timeout periods for protocol priority.

The above results indicate that analysis of network delay statistics is critical for integrated vehicle control and management of advanced aircraft. Furthermore, since *ARINC 629* terminals operate in an asynchronous autonomous mode, the impact of induced delays on the system performance would significantly increase in redundant data communication networks. (Note: Redundant rings are not synchronized and hence could have different minor cycle transmission schedules[8].)

SUMMARY AND CONCLUSIONS

Future generation advanced aircraft would require high-performance, fault-accommodating, distributed vehicle control and management systems. To this effect, *ARINC 629* has been proposed as a MAC protocol intended primarily for commercial transport aircraft, but it is also applicable to military aircraft. Although *ARINC 629* has been standardized and is being actively considered for design of integrated control systems of advanced aircraft, the delay and throughput characteristics of *ARINC 629* have not been thoroughly examined. Therefore, it is timely that this protocol be systematically analysed and evaluated relative to other avionic network protocols (e.g. AS4074.1 linear token bus and MIL-STD-1553B) for assessing the impact of the network-induced delays on dynamic performance and stability of the ICS. To address the above issues for design of aircraft communication control systems, this paper explains the essential operations of *ARINC 629* and presents a finite-state-machine model and a timed Petri net model of the protocol for its performance evaluation. The Petri net model provides the structure of a discrete-event simulation model which, in turn, can be combined with the continuous-time simulation model of an aircraft control system. Coupling of these two types of simulation permits evaluation of the network as a dynamical element of the closed loop control system of an aircraft.

REFERENCES

1 **Hopkins, A L** *et al.* *System Data Communication Structures for Active-Control Transport Aircraft*. Volume I. NASA Report CR 165774 (June 1981)

2 **Hopkins, A L** *et al.* *System Data Communication Structures for Active-Control Transport Aircraft*. Volume II. NASA Report CR 165774 (June 1981)

3 **Ray, A** 'Performance evaluation of medium access control protocols for distributed digital avionics'. *ASME J. Dynamic Syst., Measure. & Control*. Vol 109 No 4 (December 1987) pp 370–377

4 **Halevi, Y and Ray, A** 'Integrated communication and control

systems: Part I - analysis', *ASME J. Dynamic Syst., Measure. & Control*, (December 1988) pp 367-373

5 **Ray, A and Halevi, Y** 'Integrated communication and control systems: Part II - design considerations', *ASME J. Dynamic Syst., Measure. & Control* (December 1988) pp 374-381

6 *MIL-STD-1553B Protocol Manual*, United States Government Printing Office (February 1980)

7 *SAE Linear Token Passing Multiplex Data Bus*, AS4074, Revision 1.2 (December 1991)

8 *ARINC Specification 629 Multitransmitter Data Bus: Part I - Protocol Description*, Aeronautical Radio Inc., Annapolis, MD

9 **McGough, J** 'Evaluation of data buses for flight critical control applications', *IEEE Digital Avionics Syst. Conf.*, Fort Worth, TX (October-November 1986)

10 **Ray, A** 'Network access protocols for real-time distributed control systems', *IEEE Trans. Ind. Applic.* (September-October 1988) pp 897-904

11 **Shaw, J L, Herzog, H K and Okubo, K** 'Digital autonomous terminal access communication (DATAC)', *IEEE Digital Avionics Syst. Conf.*, Fort Worth, TX (October-November 1986)

12 **Gill, A** *Introduction to the Theory of Finite-State Machines*, McGraw Hill, New York (1962)

13 **Hennie, F C** *Finite-State Models for Logical Machines*, Wiley, New York (1968)

14 **Peterson, J L** *Petri Net Theory and The Modeling of Systems*, Prentice-Hall, NJ (1981)

15 **Merlin, P M** 'Specification and validation of protocols', *IEEE Trans. Commun.*, Vol 27 No 11 (November 1979) pp 1671-1680

16 **Banks, J and Carson, J S** *Discrete-Event System Simulation*, Prentice-Hall, NJ (1984)

APPENDIX A: *ARINC 629* PROTOCOL FUNCTIONAL DESCRIPTION

The *ARINC 629* protocol is designed to meet the communications requirements of commercial aircraft, and is apparently suitable for applications to digital avionics systems in military aircraft. *ARINC 629* is a carrier-sense multiple access protocol with bus topology, and functions in the broadcast mode with a finite number of terminals. Since *ARINC 629* is an autonomous terminal access bus, it is necessary for each terminal on the bus to contain its own control. The scheduling mechanism of *ARINC 629* is formulated on the principle of *collision avoidance* in which the terminals listen to the bus and wait for a quiet period prior to transmitting. Presence of a transmitted signal is used to inhibit all other terminals. The absence of this carrier signal serves as a signal that the bus is free and that another terminal may begin to transmit. Each terminal has a unique window of time during which it may transmit. This control is achieved by two Erasable Programmable Read Only Memories (EPROM) known as transmit and receive 'personalities'. The transmit personality EPROM contains the logic to ensure that the conditions for collision free transmissions are satisfied before enabling the transmitter. The receive personality EPROM serves the purposes of: (i) selection of only those messages intended for the terminal; and (ii) monitoring of the transmitted signal to guard against babbling and other transmitter malfunctions.

In its present form, *ARINC 629* is intended to interconnect up to 128 terminals across physical bus lengths of up to 100 metres. Data is transmitted over the bus using Manchester II Biphase Coding at a serial bit rate of 2 Mbit/s. Data identification is accomplished through the use of labels which precede the data. Receiving terminals decode the label and determine if the information following it is required by the attached subsystem(s). An *ARINC 629* word is 20 bits long with 16 bits of data and a parity bit. A label word has a 3-bit time high-low synchronization pattern, and a data word has the inverse 3-bit time low-high pattern. A message is composed of 1-31 word strings. Each word string has a label word followed by up to 256 data words. A terminal is limited to transmitting a maximum of 31 word strings upon each access to the transmission medium. The 16 bit data field for a label word consists of a 12 bit label field and four extension bits. There are 4096 labels and 65,536 addresses available. The specification of physical medium hardware allows for three types of data buses: (i) twisted pair with voltage taps; (ii) twisted pair with current mode coupling; and (iii) optical fibres. The remote stations connect to the bus via inductive couplers in an electrical physical medium implementation.

The applicability of *ARINC 629* is intended over a broad number of disciplines with requirements to transmit both periodic and aperiodic data. Periodic data is data that is transmitted on a regular basis with fixed transmission lengths and data latency guarantees. The transmission rate of the data is established to be once every bus cycle or once every few bus cycles. An example of periodic data is inertial sensor attitude data. Aperiodic data is that data which is transmitted only occasionally and may have fixed or variable transmission lengths. This data has no specific transmission rate, or may have rates that exceed the bus cycle time. Examples of aperiodic data are: (i) in-flight maintenance data and flight management data transmitted every few seconds; (ii) initialization data transmitted only once per flight; and (iii) emergency messages transmitted as and when they arrive. The strict data latency bounds are, however, applicable to only some of the aperiodic data such as emergency and flight management data that are critical to flight safety. As such, continuous information transfer (e.g. data transfer for video communications or asynchronous communication of large volumes of data for fileservers and printers) may not be practical.

APPENDIX B: FSM MODEL OF *ARINC 629* IN CP MODE

The access mechanism is represented by a finite state machine (FSM) model as shown in *Figures B1* and *B2*, where a state is represented by $S_{j,i}$, where i is the state number j is the access level defined as:

$$j := \begin{cases} 0 \Rightarrow \text{Awaiting commencement of level 1} \\ 1 \Rightarrow \text{Level 1} \\ 2 \Rightarrow \text{Level 2} \\ 31 \Rightarrow \text{Level 3 (sublevel 1)} \\ 32 \Rightarrow \text{Level 3 (sublevel 2)} \end{cases}$$

Every terminal is assumed to have three buffers, one for each of the levels 1, 2 and 3. The logic variables for protocol operations at a terminal are listed in *Table B1*. The states of the finite-state-machine model are then defined in terms of these logic variables as listed in *Table B2*, and the corresponding state transitions are given in *Table B3*. The operations of the protocol in the CP mode are briefly explained below with reference to *Figures B1a* and *B1b*, and *Tables B1-B3*:

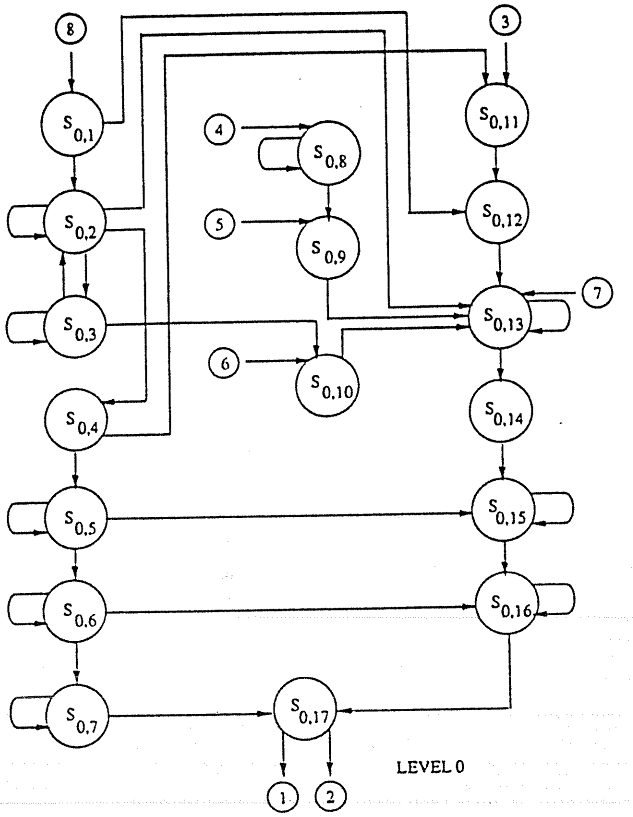


Figure B1 Finite state machine model of *ARINC 629* in CP mode. 3-8 = from *Figure 2b*; 1-2: to *Figure 2b*

Table B1 Definition of logic variables in CP mode

TI_RU: Running state of *TI*;
TI_E: Elapsed state of *TI*;

ASG_RU: Running state of *ASG*;
ASG_E: Elapsed state of *ASG*;
ASG_RE: Reset state of *ASG*;

PSG_RU: Running state of *PSG*;
PSG_E: Elapsed state of *PSG*;
PSG_RE: Reset state of *PSG*;

TG_RU: Running state of *TG*;
TG_E: Elapsed state of *TG*;
TG_RE: Reset state of *TG*;

AT_RU: Running state of *AT*;
AT_E: Elapsed state of *AT*;

L_0: Level awaiting commencement of level 1 operation;
L_1: Level 1;
L_2: Level 2;
L_31: Level 3, sublevel 1; and
L_32: Level 3, sublevel 2.

- All timers are elapsed in $S_{1,4}$. Any terminal in $S_{1,4}$ indicates that it is the first to commence level 1 transmission in a given cycle.
- *TI* and *AT* are running in $S_{1,5}$. The terminal is either transmitting its backlogged message(s), or has already gained access to the medium and is currently detecting bus activity.
- *TI*, *ASG*, *PSG* and *AT* are running in $S_{1,6}$. The terminal has already gained access to the medium, and is not currently detecting bus activity.
- *TI* and *AT* are running in $S_{2,5}$. In this state, the terminal is either transmitting its backlogged message(s) or is currently detecting bus activity.

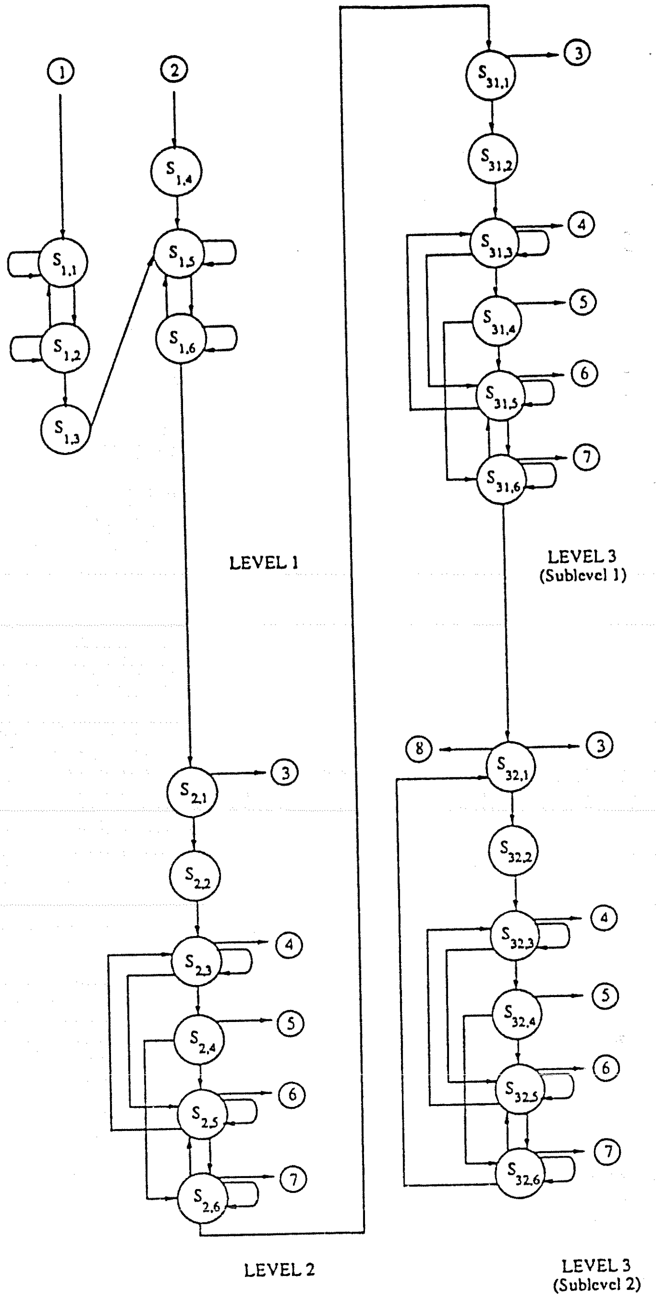


Figure B2 FSM model of *ARINC 629* in CP mode (continued). 3-8: to *Figure 2a*; 1-2: from *Figure 2a*

- A terminal *TI*, *ASG*, *PSG* and *AT* timers are running in $S_{2,6}$. In this state the terminal has already gained access to the medium in the current cycle and is not currently detecting bus activity.
- *TI* and *AT* are running in $S_{31,5}$. The terminal is either transmitting its backlogged message(s) or is currently detecting bus activity.
- *TI*, *ASG*, *PSG* and *AT* are running in $S_{31,6}$. The terminal has already gained access to the medium in the current cycle and is not currently detecting bus activity.
- *TI* and *AT* timers are running in $S_{32,5}$. The terminal is either transmitting its backlogged message(s) or is currently detecting bus activity.

Table B2 Description of states in CP mode

$s_{0,1}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_0
$s_{0,2}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_0
$s_{0,3}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU ∩ L_0
$s_{0,4}$:	TI_RU ∩ ASG_E ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_0
$s_{0,5}$:	TI_RU ∩ ASG_E ∩ TG_RU ∩ PSG_RU ∩ AT_RU ∩ L_0
$s_{0,6}$:	TI_RU ∩ ASG_E ∩ TG_E ∩ PSG_RU ∩ AT_RU ∩ L_0
$s_{0,7}$:	TI_RU ∩ ASG_E ∩ TG_E ∩ PSG_E ∩ AT_RU ∩ L_0
$s_{0,8}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,9}$:	TI_RU ∩ ASG_RU ∩ TG_E ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,10}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_E ∩ L_0
$s_{0,11}$:	TI_RU ∩ ASG_E ∩ TG_RE ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,12}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,13}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,14}$:	TI_RU ∩ ASG_E ∩ TG_RE ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,15}$:	TI_RU ∩ ASG_E ∩ TG_RU ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,16}$:	TI_RU ∩ ASG_E ∩ TG_E ∩ PSG_RU ∩ AT_E ∩ L_0
$s_{0,17}$:	TI_RU ∩ ASG_E ∩ TG_E ∩ PSG_E ∩ AT_E ∩ L_0
$s_{1,1}$:	TI_E ∩ ASG_E ∩ TG_RE ∩ PSG_E ∩ AT_RU ∩ L_1
$s_{1,2}$:	TI_E ∩ ASG_E ∩ TG_RU ∩ PSG_E ∩ AT_RU ∩ L_1
$s_{1,3}$:	TI_E ∩ ASG_E ∩ TG_E ∩ PSG_E ∩ AT_RU ∩ L_1
$s_{1,4}$:	TI_E ∩ ASG_E ∩ TG_E ∩ PSG_E ∩ AT_E ∩ L_1
$s_{1,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU ∩ L_1
$s_{1,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_1
$s_{2,1}$:	TI_RU ∩ ASG_E ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_2
$s_{2,2}$:	TI_RU ∩ ASG_RE ∩ TG_RU ∩ PSG_RU ∩ AT_RU ∩ L_2
$s_{2,3}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU ∩ AT_RU ∩ L_2
$s_{2,4}$:	TI_RU ∩ ASG_RU ∩ TG_E ∩ PSG_RU ∩ AT_RU ∩ L_2
$s_{2,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU ∩ L_2
$s_{2,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_2
$s_{31,1}$:	TI_RU ∩ ASG_E ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_31
$s_{31,2}$:	TI_RU ∩ ASG_RE ∩ TG_RU ∩ PSG_RU ∩ AT_RU ∩ L_31
$s_{31,3}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU ∩ AT_RU ∩ L_31
$s_{31,4}$:	TI_RU ∩ ASG_RU ∩ TG_E ∩ PSG_RU ∩ AT_RU ∩ L_31
$s_{31,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU ∩ L_31
$s_{31,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_31
$s_{32,1}$:	TI_RU ∩ ASG_E ∩ TG_RE ∩ PSG_RU ∩ AT_RU ∩ L_32
$s_{32,2}$:	TI_RU ∩ ASG_RE ∩ TG_RU ∩ PSG_RU ∩ AT_RU ∩ L_32
$s_{32,3}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU ∩ AT_RU ∩ L_32
$s_{32,4}$:	TI_RU ∩ ASG_RU ∩ TG_E ∩ PSG_RU ∩ AT_RU ∩ L_32
$s_{32,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU ∩ L_32
$s_{32,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RE ∩ AT_RU ∩ L_32

Table B3 Description of state transitions in CP mode

$s_{0,1}/s_{0,2}$:	ASG_RU
$s_{0,2}/s_{0,2}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU
$s_{0,2}/s_{0,3}$:	ASG_RE ∩ PSG_RE
$s_{0,2}/s_{0,4}$:	ASG_E
$s_{0,2}/s_{0,13}$:	AT_E
$s_{0,3}/s_{0,3}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU
$s_{0,3}/s_{0,2}$:	ASG_RU ∩ PSG_RU
$s_{0,3}/s_{0,13}$:	AT_E
$s_{0,4}/s_{0,5}$:	TG_RU
$s_{0,4}/s_{0,11}$:	AT_E
$s_{0,5}/s_{0,5}$:	TI_RU ∩ TG_RU ∩ PSG_RU ∩ AT_RU
$s_{0,5}/s_{0,6}$:	TG_E
$s_{0,5}/s_{0,15}$:	AT_E
$s_{0,6}/s_{0,6}$:	TI_RU ∩ TG_E ∩ PSG_RU ∩ AT_RU
$s_{0,6}/s_{0,7}$:	PSG_E
$s_{0,6}/s_{0,16}$:	AT_E
$s_{0,7}/s_{0,7}$:	TI_RU ∩ AT_RU
$s_{0,7}/s_{0,17}$:	AT_E
$s_{0,8}/s_{0,8}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU
$s_{0,8}/s_{0,9}$:	TG_E
$s_{0,9}/s_{0,13}$:	TG_RE
$s_{0,10}/s_{0,13}$:	ASG_RU ∩ PSG_RU
$s_{0,11}/s_{0,12}$:	ASG_RE
$s_{0,12}/s_{0,13}$:	ASG_RU
$s_{0,13}/s_{0,13}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU
$s_{0,13}/s_{0,14}$:	ASG_E
$s_{0,14}/s_{0,15}$:	TG_RU
$s_{0,15}/s_{0,15}$:	TI_RU ∩ TG_RU ∩ PSG_RU
$s_{0,15}/s_{0,16}$:	TG_E
$s_{0,16}/s_{0,16}$:	TI_RU ∩ PSG_RU
$s_{0,16}/s_{0,17}$:	PSG_E
$s_{0,17}/s_{1,1}$:	TI_E ∩ TG_RE ∩ AT_RU ∩ L_1
$s_{0,17}/s_{1,4}$:	TI_E ∩ L_1
$s_{1,1}/s_{1,1}$:	TG_RE ∩ AT_RU
$s_{1,1}/s_{1,2}$:	TG_RU
$s_{1,2}/s_{1,2}$:	TG_RU ∩ AT_RU
$s_{1,2}/s_{1,1}$:	TG_RE
$s_{1,2}/s_{1,3}$:	TG_E
$s_{1,3}/s_{1,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE
$s_{1,4}/s_{1,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU
$s_{1,5}/s_{1,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU
$s_{1,5}/s_{1,6}$:	ASG_RU ∩ PSG_RU
$s_{1,6}/s_{1,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU
$s_{1,6}/s_{1,5}$:	ASG_RE ∩ PSG_RE
$s_{1,6}/s_{2,1}$:	ASG_E ∩ PSG_RU ∩ AT_RU ∩ L_2
$s_{2,1}/s_{2,2}$:	ASG_RE ∩ TG_RU
$s_{2,1}/s_{0,11}$:	AT_E ∩ L_0
$s_{2,2}/s_{2,3}$:	ASG_RU
$s_{2,3}/s_{2,3}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU ∩ AT_RU
$s_{2,3}/s_{2,4}$:	TG_E
$s_{2,3}/s_{2,5}$:	ASG_RE ∩ TG_RE ∩ PSG_RE
$s_{2,3}/s_{0,8}$:	AT_E ∩ L_0
$s_{2,4}/s_{2,5}$:	ASG_RE ∩ TG_RE ∩ PSG_RE
$s_{2,4}/s_{2,6}$:	TG_RE
$s_{2,4}/s_{0,9}$:	AT_E ∩ L_0
$s_{2,5}/s_{2,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU
$s_{2,5}/s_{2,3}$:	ASG_RU ∩ TG_RU ∩ PSG_RU
$s_{2,5}/s_{2,6}$:	ASG_RU ∩ PSG_RU
$s_{2,5}/s_{0,10}$:	AT_E ∩ L_0
$s_{2,6}/s_{2,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU
$s_{2,6}/s_{2,5}$:	ASG_RE ∩ PSG_RE
$s_{2,6}/s_{31,1}$:	ASG_E ∩ L_31
$s_{2,6}/s_{0,13}$:	AT_E ∩ L_0
$s_{31,1}/s_{31,2}$:	ASG_RE ∩ TG_RU
$s_{31,1}/s_{0,11}$:	AT_E ∩ L_0
$s_{31,2}/s_{31,3}$:	ASG_RU
$s_{31,3}/s_{31,3}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU ∩ AT_RU
$s_{31,3}/s_{31,4}$:	TG_E
$s_{31,3}/s_{31,5}$:	ASG_RE ∩ TG_RE ∩ PSG_RE
$s_{31,3}/s_{0,8}$:	AT_E ∩ L_0
$s_{31,4}/s_{31,5}$:	ASG_RE ∩ TG_RE ∩ PSG_RE
$s_{31,4}/s_{31,6}$:	TG_RE
$s_{31,4}/s_{0,9}$:	AT_E ∩ L_0
$s_{31,5}/s_{31,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU
$s_{31,5}/s_{31,3}$:	ASG_RU ∩ TG_RU ∩ PSG_RU
$s_{31,5}/s_{31,6}$:	ASG_RU ∩ PSG_RU
$s_{31,5}/s_{0,10}$:	AT_E ∩ L_0
$s_{31,6}/s_{31,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU
$s_{31,6}/s_{31,5}$:	ASG_RE ∩ PSG_RE
$s_{31,6}/s_{32,1}$:	ASG_E ∩ L_32
$s_{31,6}/s_{0,13}$:	AT_E ∩ L_0
$s_{32,1}/s_{32,2}$:	ASG_RE ∩ TG_RU
$s_{32,1}/s_{0,1}$:	ASG_RE ∩ L_0
$s_{32,1}/s_{0,11}$:	AT_E ∩ L_0
$s_{32,2}/s_{32,3}$:	ASG_RU
$s_{32,3}/s_{32,3}$:	TI_RU ∩ ASG_RU ∩ TG_RU ∩ PSG_RU ∩ AT_RU
$s_{32,3}/s_{32,4}$:	TG_E
$s_{32,3}/s_{32,5}$:	ASG_RE ∩ TG_RE ∩ PSG_RE
$s_{32,3}/s_{0,8}$:	AT_E ∩ L_0
$s_{32,4}/s_{32,5}$:	ASG_RE ∩ TG_RE ∩ PSG_RE
$s_{32,4}/s_{32,6}$:	TG_RE
$s_{32,4}/s_{0,9}$:	AT_E ∩ L_0
$s_{32,5}/s_{32,5}$:	TI_RU ∩ ASG_RE ∩ TG_RE ∩ PSG_RE ∩ AT_RU
$s_{32,5}/s_{32,3}$:	ASG_RU ∩ TG_RU ∩ PSG_RU
$s_{32,5}/s_{32,6}$:	ASG_RU ∩ PSG_RU
$s_{32,5}/s_{0,10}$:	AT_E ∩ L_0
$s_{32,6}/s_{32,6}$:	TI_RU ∩ ASG_RU ∩ TG_RE ∩ PSG_RU ∩ AT_RU
$s_{32,6}/s_{32,5}$:	ASG_RE ∩ PSG_RE
$s_{32,6}/s_{32,1}$:	ASG_E ∩ L_32
$s_{32,6}/s_{0,13}$:	AT_E ∩ L_0

- *TI*, *ASG*, *PSG* and *AT* are running in $S_{32,6}$. The terminal has already gained access to the medium in the current cycle and is not currently detecting bus activity.

APPENDIX C: INTRODUCTION TO PETRI NET MODELLING

Petri net is known as an abstract, formal model of information flow. Petri nets are used extensively in modelling systems events in which concurrence of events is possible. However, there are constraints in the concurrence, precedence, or frequency of these occurrences. Petri net models are built on the basis of *events* and *conditions* and the relationships between them. Petri nets are ideally suited for use in modelling discrete-event systems with concurrent or parallel events.

Petri net graphs

A Petri net graph is a representation of a Petri net structure as a bipartite directed multigraph. *Figure C1* shows a simple marked Petri net. The graph contains two types of nodes:

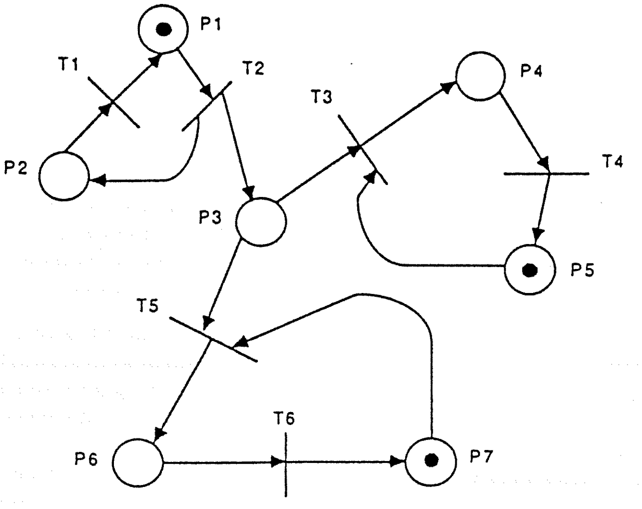


Figure C1 Marked Petri net

- The circles are called *places* and they represent conditions.
- The bars are called *transitions* and they represent events.

The places and transitions are interconnected by directed arcs from places to transitions and transitions to places. An arc directed from the place *Pi* to a transition *Tj* defines *Pi* to be the *input* to *Tj* and *Tj* is defined as the *output* of *Pi*. Similar definition follows for a directed arc from a transition *Tj* to a place *Pk*. A Petri net is controlled by the position and movement of markers called *tokens*. A token at a place indicates the holding of the condition of a place. A pattern of tokens in a Petri net called a *marking* represents the state of the system. A Petri net with tokens is called a *marked Petri net*.

The dynamic behaviour of a system execution is represented by the firing of the corresponding transition. The changes in the state are represented by the movement of tokens in the Petri net. The firing rules of Petri nets are as follows:

- a transition is enabled if and only if each of its input places has at least one token
- a transition can fire only if it is enabled
- when a transition fires:
  - — a token is removed from each of its input places
  - — a token is deposited into each of its output places.

The execution of a simple Petri net is shown in *Figure C2*. The Petri net with the initial marking is shown in *Figure C2a*. The transition *T1* is initially enabled because each of the places *P1* and *P2* has a token. Firing the transition *T1* removes a token from places *P1* and *P2*, and deposits a token at each of the places *P3* and *P4*, shown in *Figure C2b*. At this instant transition *T2* is enabled because its input place has a token. Since transition *T2* is enabled it fires; consequently, a token is removed from place *P3* and deposited at place *P5*, shown in *Figure C2c*. Now the transition *T3* is enabled because both its input places *P4* and *P5* have a token. The firing of transition *T3* removes the token from places *P4* and *P5* and deposits a token at places *P1* and *P2*, returning the system into its initial configuration.

Extensions to Petri nets

Petri nets can be used to model a variety of systems, but there are limitations on the modelling power of Petri nets in its basic form. Subsequently, a number of extensions have been developed to address the limitations that occur when applying Petri nets to real complex systems. Some of the major extensions are place-transition nets, place-coloured nets, predicate-transition nets, predicate action nets, and timed nets. The *timed Petri net* which has been used in modelling the protocol is briefly discussed below.

Timed Petri nets

To study and model some real complex systems, the Petri net model is extended to include the notion of time. Petri nets

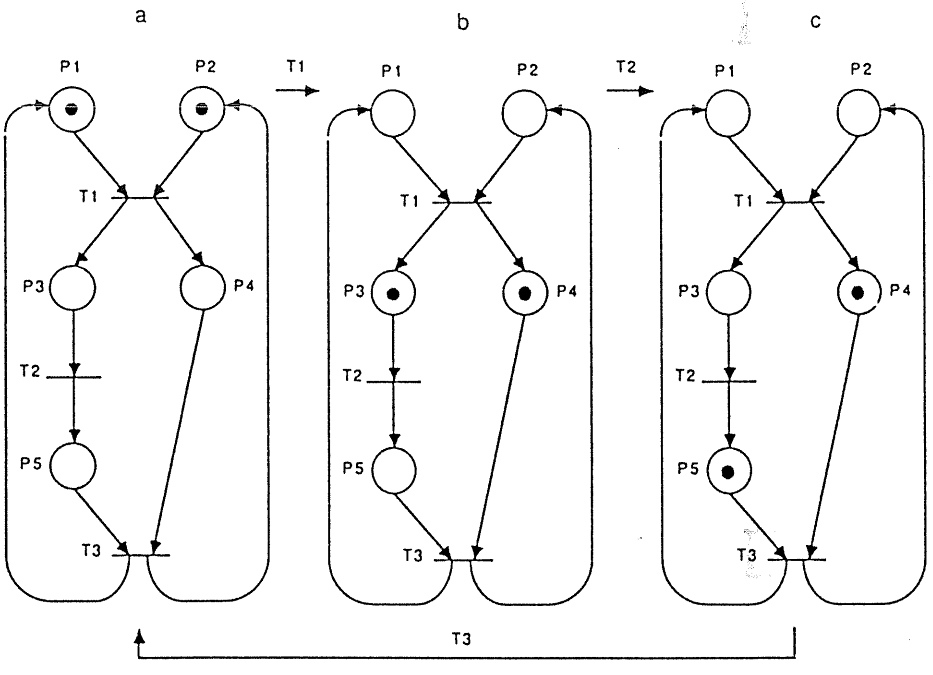


Figure C2 Execution of a simple Petri net have been used to describe systems performing actions. In real situations these actions or operations occur after some finite time, or require some finite time to be executed.

Two intervals, namely *enabling time* and *firing time*, are associated with each transition in a timed Petri net model. Each transition in the net must remain enabled for a time period *te* before it can fire. A transition is then said to be *firable*, and it immediately begins firing by absorbing the token(s) from its input place(s). If a conflict occurs at the firing instant (i.e. if more than one transitions are firable), then the selection of the transition that will fire is assumed to be a random event. The transition continues to fire for a period *tf*. At the end of the firing time *tf*, the transition finishes firing and deposits the token(s) into its output place(s). The firing *tf* models the processing time and the enabling time *te* models the timeout in real-time processes. An assumption is that no transition can be enabled while it is firing. The enabling time *te* and the firing time *tf* can be expressed as functions of the particular transition which can be deterministic or stochastic.

APPENDIX D: PERFORMANCE PARAMETER DEFINITIONS

Definition 1 Normalized offered traffic is defined as the expected value of the sum of the lengths (in units of time) of messages that arrive at all terminals of the network per unit time.

Definition 2 Normalized throughput is defined as the expected value of the total length (in units of time) of messages that flow through the network per unit time.

Definition 3 Queueing delay is defined as the time interval between the instant of arrival of a message at the transmitter queue of a source terminal and the instant of transmission of its first bit.

Definition 4 Data latency is defined as the time interval between the instant of arrival of a message at the transmitter queue of a source station and the instant of reception of its last bit, at the receiver station.

Remark D-1 Offered traffic does not include the protocol overhead whereas normalized throughput does.

Remark D-2 The CP mode of *ARINC 629* has three levels of operation unlike the BP mode that has only a single level. Therefore, normalized offered traffic in the CP mode can be defined for the individual levels 1, 2 and 3; total offered traffic on the network is then the sum of offered traffics at the levels, 1, 2 and 3. For example, if the offered traffic at each of the levels 1 and 2 is maintained at fixed values, the total offered traffic in the network can be regulated by varying the offered traffic at the level 3.