International Standard 7846

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION•МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ•ORGANISATION INTERNATIONALE DE NORMALISATION

Industrial real-time FORTRAN — Application for the control of industrial processes

Langage FORTRAN en temps réel industriel — Application pour la commande des processus industriels

First edition — 1985-09-15

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

UDC 681.3.06 : 800.92 **Ref. No. ISO 7846-1985 (E)**

Descriptors : data processing, programming languages, fortran.

Price based on 32 pages

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work.

Draft International Standards adopted by the technical committees are circulated to the member bodies for approval before their acceptance as International Standards by the ISO Council. They are approved in accordance with ISO procedures requiring at least 75 % approval by the member bodies voting.

iTeh STANDARD PREVIEW

International Standard ISO 7846 was prepared by Technical Committee ISO/TC 97, *Information processing systems*.

(standards.iteh.ai)

Users should note that all International Standards undergo revision from time to time and that any reference made herein to any other International Standard implies its latest edition, unless otherwise stated.

ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea15e7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

© International Organization for Standardization, 1985 •

Printed in Switzerland

Contents

Page

0 Introduction ... 1

1 Scope and field of application ... 1

2 Definitions ... 1

Section one: Multiprogramming and real-time features ... 3

3 Introduction ... 3

4 Date and time information ... 3

4.1 Obtain date and time ... DATIM 3

4.2 Obtain clock counts ... CLOCK 3

5 General aspects of tasking ... 3

5.1 States and transitions ... 3

5.2 Multiple activation calls ... 5

5.3 Synchronization concepts ... 5

5.3.1 Eventmarks ... 5

5.3.2 Resourcemarks ... 5

5.3.3 Semaphores ... 6

6 Procedure references ... 6

6.1 Terms and summary of procedure references ... 6

6.2 Creation of a new task ... CREATE 8

6.3 Eliminating a task from the real-time system ... KILL 8

6.4 Scheduling a task ... SKED 8

6.5 Starting a task by simplified calls ... 9

6.5.1 Starting a task immediately ... STRT 9

6.5.2 Starting a task after a specified time delay ... STRTAF 9

6.5.3 Starting a task at a specified absolute time ... STRTAT 10

6.5.4 Starting a task in repeated execution ... 10

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-8625be543084/iso-7846-1985

6.5.5 Initial start immediately ..... CYCL 10

6.5.6 Initial start after a specified time delay ..... CYCLAF 10

6.5.7 Initial start at specified absolute time ..... CYCLAT 10

6.5.8 Connection of a task to an event ..... CON 10

6.6 Elimination of previous scheduling ..... DSKED 11

6.6.1 Elimination of event connections ..... DCON 11

6.6.2 Elimination of time connections ..... CANCEL 11

6.7 Delaying continuation of a task ..... SUSPND 11

6.8 Suspending until event or time 12

6.8.1 Delay until event has occurred ..... HOLD 12

6.8.2 Delay for a specified relative time ..... DELAY 12

6.9 Eventmark operations 12

6.9.1 Setting an eventmark to the ON condition ..... POST 12

6.9.2 Clearing an eventmark ..... CLEAR 12

6.9.3 Testing an eventmark condition ..... TESTEM 12

6.9.4 Masking an eventmark ..... MKEM 13

6.9.5 Unmasking an eventmark ..... UNMKEM 13

6.10 Resourcemark operations 13

6.10.1 Setting a resourcemark to the locked condition ..... LOCK 13

6.10.2 Setting a resourcemark to the unlocked condition ..... UNLOCK 13

6.10.3 Testing and setting a resourcemark to the locked condition ..... TLOCK 13

6.11 Semaphore operations 13

6.11.1 Initialization of semaphore ..... PRESEM 14

6.11.2 Wait on semaphore ..... WAITS 14

6.11.3 Release of semaphore ..... SIGNAL 14

6.11.4 Reading a semaphore value ..... IRDSEM 14

6.12 Normal termination of execution ..... EXIT 15

Section two: Binary pattern and bit processing 16

7 Introduction 16

8 Binary pattern processing 16

8.1 Boolean operations 16

8.1.1 Inclusive OR ..... IOR 16

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

8.1.2 Boolean AND ..... IAND 16
8.1.3 Boolean complement ..... NOT 16
8.1.4 Exclusive OR ..... IEOR 16
8.2 Shift operations 16
8.2.1 Logical shift ..... ISHL 16
8.2.2 Arithmetic shift ..... ISHA 17
8.2.3 Circular shift ..... ISHC 17
9 Bit processing 17
9.1 Bit testing ..... BTEST 17
9.2 Set bit ..... IBSET 17
9.3 Clear bit ..... IBCLR 17
9.4 Change bit ..... IBCHNG 17

Section three: Process input/output 18
10 Introduction 18
11 Scope of the process I/O and general structure of I/O routines 18
12 Input/output of analog values 18
12.1 Sequential analog data input ..... AISQW 18
12.2 Analog data input in random sequence ..... AIRDW 19
12.3 Analog data output ..... AOW 19
13 Input/output of digital values 19
13.1 Digital input ..... DIW 19
13.2 Digital output 19
13.2.1 Digital pulse output ..... DOMW 19
13.2.2 Latched digital output ..... DOLW 19

Section four: File handling 20
14 Introduction 20
15 Background information 20
16 File system environment 20
17 Procedures to control file access 21
17.1 Introduction 21
17.2 Creation of files ..... CFILW 21
17.3 Deletion of files ..... DFILW 21

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13e7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

17.4 Opening files ..... OPENW 21

17.5 Closing files ..... CLOSEW 22

17.6 Modify access mode ..... MODAPW 22

Annexes

A Historical background 23

B Further descriptions pertaining to individual clauses 26

C Abortion of tasks 30

D File handling 31

Bibliography 32

iTeh STANDARD PREVIEW
(standards.iteh.ai)

ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

Industrial real-time FORTRAN — Application for the control of industrial processes

0 Introduction

This International Standard describes a task model and a set of related routines to allow control of multi-task systems using FORTRAN as the programming language. Information concerning the development of this International Standard is given in annex A, and suggestions and justifications for some features are given in annex B. Annex C deals with the problem of aborting tasks and annex D with file handling. These annexes do not form part of this International Standard.

1 Scope and field of application

This International Standard establishes external procedure references for use in industrial computer control systems. These external procedure references provide access to time and date information, permit interface of programs with executive systems, process input and output functions, allow manipulation of bit strings and provide a method for file handling.

These procedures are intended for use with programs written in FORTRAN in accordance with ISO 1539.[1] These programs are expected to be executable both in a solitary and in a multiprogramming environment under the control of a real-time executive system.

It is the responsibility of the executive system to deal with exceptions or errors, such as division by zero or referencing SQRT(−1.0), so that such errors do not cause serious effects, for example unwanted or uncontrolled abortion, or affect other tasks.

This International Standard is applicable to all FORTRAN systems which require multi-tasking features.

NOTE — No extensions or variations from this International Standard should be implemented except for features explicitly declared in this International Standard to be processor dependent.

2 Definitions

For the purpose of this International Standard, the following definitions apply.

NOTE — Terms printed in italics in the definitions are themselves defined in this clause. However, in each definition, such items are only printed in italic at their first occurrence.

2.1 access mode: The right or permission to access (read or write) a *file* granted by the *processor* following a request for such permission.

2.2 basic clock counts: Counts of the basic unit of the system real-time clock as available for the user's program.

2.3 computation: A set of *operations* carried out on a set of data, as available for the user's program.

2.4 critical region: A part of a *sequential order of operations* operating on shared data such that this part of the program must have exclusive access to the shared data during the *execution*.

2.5 DORMANT: One of the definite states of a *task*.

A dormant task is known to the *executive system* and is not in any of the states *PENDING, RUNNING*, or *SUSPENDED*.

2.6 event: A significant discrete occurrence or incident which is intended to affect the *execution* of some *task* in a planned manner.

An event itself occurs instantaneously and sets an *eventmark*[1].

2.7 eventmark: Internal variable of the *executive system*, used to indicate that an *event* has occurred.

If the user's system contains *parallel tasks*, the eventmarks are shared data elements for the *tasks*.[1]

2.8 execution: The collection of actions performed by a computer *processor* carrying out instructions in a sequential manner.

2.9 executable program: A program including all of its functions and subroutines in a form suitable for *execution*.

1) See also 5.3.1.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

2.10 executive routine; executive system: The part of the *processor* which supports the procedures described in this International Standard.

2.11 file: A collection of related records treated as a unit.

For the purpose of this International Standard, the records are considered to be of fixed length. Record storage and access are independent of the internal format of records.

2.12 initiation: The action taken by the *executive system* to start the *execution* of a *task* at its first executable statement.

2.13 multiprocessing[1]**:** A mode of *operation* that provides for parallel processing by two or more *processors* of a multiprocessor.

2.14 multiprogramming[1]**:** A mode of *operation* that provides for the interleaved *execution* of two or more computer programs by a single processor.

2.15 multitasking[1]**:** A multiple *operation* that provides for the concurrent performance or interleaved *execution* of two or more *tasks*.

2.16 NON-EXISTENT: One of the definite (formal) states of a *task*.

A non-existent task is unknown to the *executive system*.

2.17 object task; designated task; referenced task: The *task* that is wanted or expected to be started, halted, stopped, or otherwise affected as a consequence of a system subroutine call.

2.18 operation: A deterministic rule for the generation of a finite set of data from another finite set of data.

2.19 overrun: Occurs when the condition for *initiation* of a *task* becomes true while the task is still running because of a previous *initiation*.

2.20 parallel tasks; concurrent tasks: A set of *tasks* whose *operations* may overlap in *time*.

2.21 PENDING: One of the definite states of a *task*.

A pending task has been associated with an *event* or *time* condition such that when the condition occurs, the task will be transferred to state *RUNNING* and initiated.

2.22 processor: The combination of a data processing system and the mechanism by which programs are transformed for use on that data processing system.

2.23 processor dependent: Describes an action of the *processor* that is not specified in this International Standard.

2.24 repetitive execution: Occurs when a *task* is repeatedly initiated, whether at fixed intervals or by repetitive *events*.

2.25 resourcemark: Internal variable of the *executive system*, used to indicate that a resource is exclusively reserved for a *task*.[2]

2.26 RUNNING: One of the definite states of a *task*.

A running task is executing in its *virtual processor*.

2.27 semaphore: A variable of the *executive system*, used for the exchange of synchronizing information between interacting parallel *tasks*.

All semaphore *operations* in this International Standard imply *critical region* protection, provided by the executive system.

2.28 sequential order of operations: An *order of operations* the results of which are as if the operations are performed strictly one after another.

2.29 SUSPENDED: One of the definite states of a *task*.

A suspended task has temporarily halted the *execution* of its *virtual processor*, and is waiting for a specified condition to continue the execution of its virtual processor.

2.30 task: A *computation* which can be scheduled.

The *operations* of this computation are performed in a strict *sequential order of operations*.

(See also 2.17 and 2.28.)

2.31 time:

a) **absolute time:** Complete time and date specification.

b) **relative time:** A time increment or difference.

2.32 virtual processor: An environment in which a *task* can run from the time it is initiated until it terminates without consideration of availability of resources, managed by the *processor* and not by the user's program.

A particular implementation serves to map a set of virtual processors onto a set of real processor(s). This mapping is *processor dependent*.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/6ca15c7e-6780-42d5-96a7-18623ba34308/iso-7846-1985

1) Definition taken from ISO 2382/10.[2]

2) See also 5.3.2.

Section one: Multiprogramming and real-time features

3 Introduction

This section describes several procedure references available for the user's program, and relating to multiprogramming and, in particular, real-time operation. For all calls given in this section, the operations are generally considered indivisible, i.e. their operation will behave as if they are not interrupted.

4 Date and time information

For programming in a real-time environment, the user must have access to the time variables of the executive system. These time variables are obtained by system calls as described below.

Unambiguous time specification requires unique designation of time including complete date and an acknowledged calendar, defining time zero. Execution of reference to subroutine DATIM provides this complete information. The date refers to the Gregorian calendar.

The calls are

CALL DATIM(t1) for obtaining current date and time,

CALL CLOCK(j,k1,k2) for obtaining the basic clock counts.

iTeh STANDARD PREVIEW
(standards.iteh.ai)

4.1 Obtain date and time

The form of the call is

CALL DATIM(t1)

ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

where t1 designates an integer array, into whose first 8 elements will be placed the absolute time, as expressed by the executive system's real-time clock at the time when the call is executed. These elements are as follows :

first element: Counts of the basic clock

second element: Milliseconds (0 to 999)

third element: Seconds (0 to 59)

fourth element: Minutes (0 to 59)

fifth element: Hours (0 to 23)

sixth element: Day (1 to 31)

seventh element: Month (1 to 12)

eighth element: Year

4.2 Obtain clock counts[1)]

Execution of a reference to this subroutine allows the user program to obtain the current value of the system real-time clock, expressed in basic clock counts.

The form of the call is

CALL CLOCK(j,k1,k2)

where

j designates an integer variable or integer array element into which the current value of the clock will be placed as a positive integer. j is counted up to a maximum value as given by k2, then set to zero and counted again;

k1 is the number of basic clock counts per second. It is an integer value returned by the system. This argument shall be an integer variable or an integer array element;

k2 is the maximum value j can attain. It is an integer value returned by the system. This argument shall be an integer variable or an integer array element.

The modulus size of j is given by argument k2 (modulus = k2 + 1). In practice, this will usually be equal to the modulus of the hardware counter realizing the clock.

5 General aspects of tasking

5.1 States and transitions

At any time, a task is in one and only one state. Actions executed by the executive system, other tasks, or the designated task itself, may cause transition from one state to another. These transitions are performed instantly, i.e. they are considered ideally to take no time.

This mathematical model of a task may be visualized by a "state diagram", such as that shown in the figure, in which the states are nodes, illustrated as circles, while transitions are drawn as pointed arrows from one node to another. [3]

1) See also annex B.

A multitasking system consists of several parallel tasks and can be considered as modelled by a number of disjointed but similar diagrams. It is feasible to apply a three-dimensional picture, with the similar diagrams sandwiched on top of each other and oriented so that the identical states of the individual diagrams cover each other.

State transitions are generally caused by subroutine calls, occurrence of events, or expiration of time limits. The name, form, and interpretation of the subroutine calls are standardized and described below.

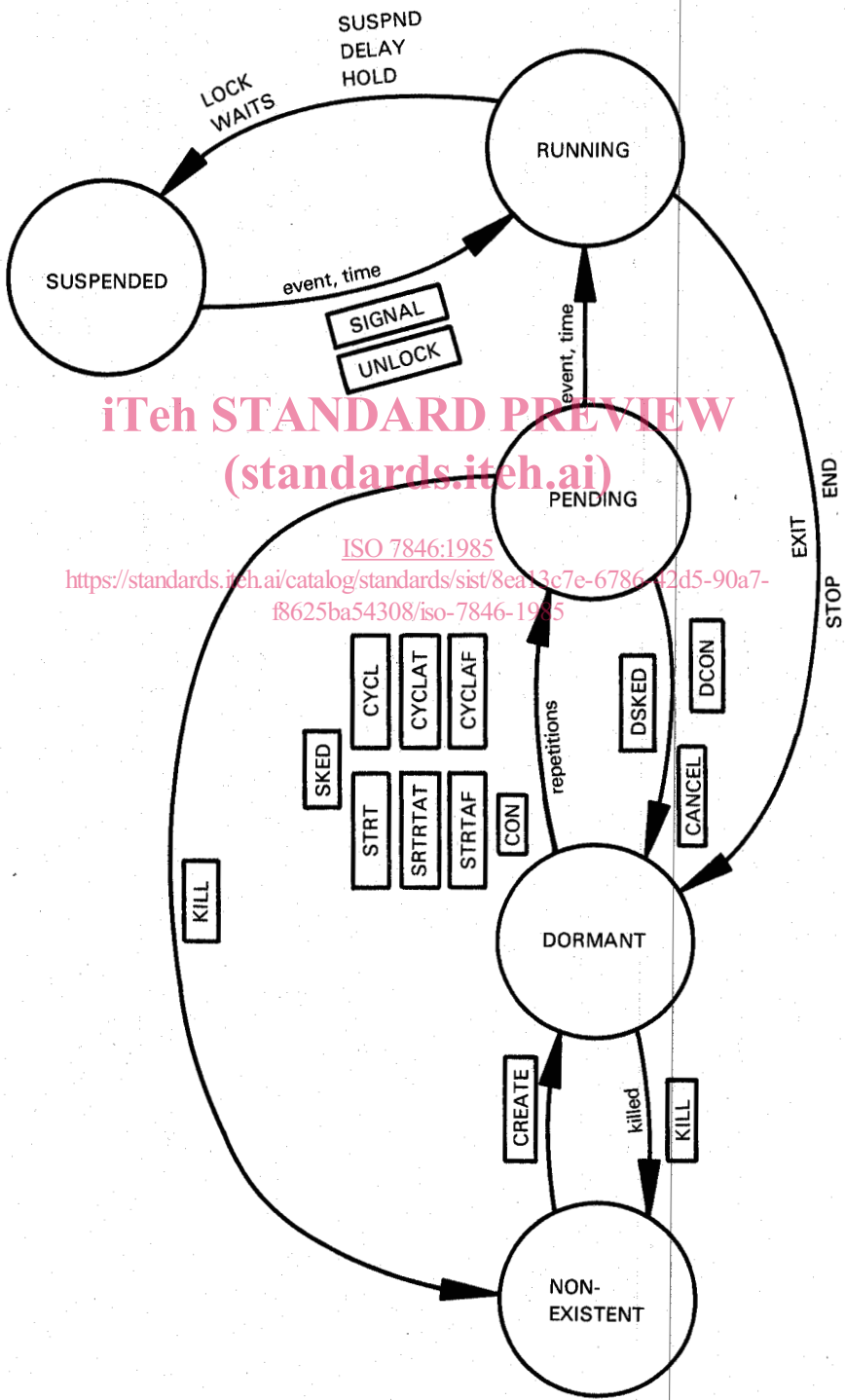
In this International Standard, a task is described by a mathematical model illustrated by the state diagram shown in the figure. This model adheres to the following basic principles:

a) transitions are non-ambiguous, i.e. for a given stimulus in a given state, the task can transit to only one possible new state;

b) transitions are performed instantly, i.e. in zero time;

c) a task exists in one state only at a time;

d) the state model describes the behaviour of tasks as seen by the application programmer.



iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985

Figure — State model and transition diagram

The following symbolism is used for transitions in the state diagram:

— Capital letters in box : Effect on a task imposed by another task, i.e. a subroutine call in one task has the indicated effect on the designated task.

— Capital letters without box : Effect imposed by the task itself while in the state RUNNING.

— Small letters : Conditions under which the executive system performs the indicated state transitions.

With reference to principle d) above, no attempt is made to describe executive system actions transparent to the application programmer. Consequently, the state RUNNING is related to the task's virtual processor (see 2.32). It is immaterial for the state of a task whether a physical processor happens to be assigned to it or the execution is temporarily hampered by the executive system due to limited availability of physical processors and the task's low priority. Thus, the model is as well adapted to multiprocessors as it is to single processor computers.

5.2 Multiple activation calls [1)]

Different tasks may issue apparently conflicting transition calls for the same object task. This will be the case during normal operation, as well as under error conditions, since the state of an object task is unknown at the time a transition call is made by another task.

In accordance with the principle of a task being in one state only at a time (see 5.1), a distinction is made between state transitions and calls for such transitions. Transition calls are received by the executive system, which will apply its own scheduling strategy in handling such calls.

Depending on available resources, such as internal table space, etc., of the executive system, a transition call will be accepted or rejected. An argument will be returned after the reference, with an appropriate value indicating whether the reference has been accepted normally or rejected.

5.3 Synchronization concepts

This International Standard provides three concepts for synchronization between tasks and the resolution of resource contention, as follows:

— eventmarks;
— resourcemarks;
— semaphores.

Eventmarks and semaphores are mainly used for synchronization purposes, whereas the resourcemarks are mainly used for the resolution of resource contention.

Eventmarks, resourcemarks, and semaphores are local variables of the executive system and they may not be accessed other than by the mechanisms described in this International Standard.

5.3.1 Eventmarks

In the management of concurrent tasks, it is necessary to associate certain tasks with certain events. These events may be either external or internal events. External events are of some physical nature, such as a contact closure, but the connection between an external event and its eventmark is beyond the scope of this International Standard. An internal event arises from specific program action (see 6.9.1 for a description of CALL POST).

Eventmarks are selected by reference to a numeric selector in the range 1 (one) to n, where n is processor dependent. Eventmarks have two states

ON
OFF

The association of events to tasks is done by the subroutine calls CALL SKED (see 6.4), CALL CON (see 6.5.8), CALL SUSPND (see 6.7), and CALL HOLD (see 6.8.1).

An eventmark is turned to ON by the occurrence of an internal or external event. If one or more tasks are associated with this event, the executive system will cause each associated task to begin or continue execution. An eventmark is turned to OFF by direct program control or by the executive system as it services the tasks associated with the eventmark. Eventmarks can be changed only by the procedure references defined in this International Standard and by the executive system that services the events.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea15e7e-6786-42d3-90a7-f8625ba54308/iso-7846-1985

Eventmarks may also be set to the OFF state by a specific program action, the CALL CLEAR (see 6.9.2), and they may be masked and unmasked (see 6.9.4 and 6.9.5). Additionally, the value of an eventmark may be tested by the logical function TESTEM (see 6.9.3).

5.3.2 Resourcemarks

A simple means to resolve contention for various resources is provided by the resourcemark concept, which permits one and only one task to use a resource at a time.

Resourcemarks are selected by reference to a numeric selector in the range 1 (one) to n, where n is processor dependent. Resourcemarks have two states

LOCKED
UNLOCKED

This International Standard does not define what can be considered to be a resource. It is the responsibility of the user to associate a resource with a resourcemark. If the user wants to reserve a resource exclusively for the running task, he chooses a resourcemark for the resource and performs the reference CALL LOCK (see 6.10.1). Should the resourcemark be in the state UNLOCKED at this moment, the resourcemark will change to state LOCKED, reserving the corresponding resource exclusively for this task. This reservation is later released by the task, by execution of a reference to EXIT (see 6.12) or UNLOCK (see 6.10.2).

1) See also annex B.

Should a task attempt to LOCK a resourcemark which is already locked, the executive system will transfer this task into the state SUSPENDED, where it remains until the resourcemark becomes unlocked by some other task.

If several tasks are waiting for a resourcemark to be UNLOCKED, only one will be selected for execution by the executive system. (For details, see 6.10.2.) This is a main difference from the eventmark concept, where all tasks waiting for an eventmark are transferred to state RUNNING if the eventmark is set to ON.

A resourcemark can also be set to LOCKED by reference to the logical function TLOCK (see 6.10.3).

5.3.3 Semaphores

Semaphores represent a synchronization concept useful for cases where a more advanced mechanism than those available by resourcemarks and eventmarks is desired. Semaphores are selected by reference to a numeric selector in the range 1 (one) to n, where n is processor dependent. Unlike eventmarks and resourcemarks, a semaphore has an integer value, s . The value s of a semaphore must be initialized and may later be set by the reference CALL PRESEM (see 6.11.1).

If the task needs to wait until the value of a semaphore is greater than or equal to a specific value, the task uses the reference CALL WAITS (see 6.11.2). By execution of CALL WAITS, the executive system tests if the specified decrement j is greater than s. In this case, the calling task is transferred to state SUSPENDED, where it remains until the semaphore value s is incremented by another task to a value greater than or equal to j. If j is not greater than s , the value s is decremented by j and the calling task continues its execution.

By execution of CALL SIGNAL (see 6.11.3), the executive system increments by j the value s of the specified semaphore. This incrementation may bring a task, waiting for this semaphore, from state SUSPENDED to state RUNNING, if s becomes greater than or equal to j of the suspended task. If multiple tasks are waiting for this semaphore and are candidates for running, after the incrementation of s by j, the order in which these tasks transit to state RUNNING is processor dependent.

By setting s to certain values and choosing different values for j, a variety of advanced synchronization concepts and solutions for resource contention are possible.

In addition to the calls described above, the value of a semaphore may be read using the integer function IRDSEM (see 6.11.4).

6 Procedure references

6.1 Terms and summary of procedure references [1]

This clause contains a summary of the subroutine calls and function references described in subsequent clauses.

The following designations for parameters apply to several of the calls. If the exact meaning of these parameter designations deviates from that described below, it will be marked specifically in the detailed description of the call. If the meaning is exactly as defined here, the description of the parameter will be omitted in the description of the call and a reference will be made to this subclause.

i specifies the task to be affected (object task). The argument shall be an integer array.
The contents of i may be partly generated by CREATE. i is used as input parameter in all other calls.

t,t1,t2 designate integer arrays, whose first 8 elements contain a specification of absolute or relative time. Negative values of elements are not permitted. These elements are as follows :

first element: Basic clock counts

second element: Milliseconds

third element: Seconds

fourth element: Minutes

fifth element: Hours

sixth element: Day(s)

seventh element: Month(s)

eighth element: Year(s)

If, for absolute times, value 0 is used for one of the three date elements, this shall be interpreted as "current date", "current month", or "current year" by the executive system.

The interpretation of a relative time specification containing months or years different from zero is processor dependent.

m is set on return to the calling program, to indicate the disposition of the request as follows :

0 or less: undefined

1: request accepted

2 or greater: request rejected (error condition)

This argument shall be an integer variable or integer array element. The processor may define specific values greater than or equal to 2 to distinguish between certain reasons for rejection.

The list of function and subroutine calls, described in detail in subsequent subclauses, is given in table 1.

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8e…
f8625ba54308/iso-7846-1985

1) See also annex B.

Table 1

Full description in subclause	Call	Parameters
6.2	CALL CREATE(i,m)	i : identification of created task and associated program
6.3	CALL KILL(i,m) (opposite of CREATE)	
6.4	CALL SKED(i,s,e1,t1,t2,e2,m) (general scheduling)	s : mode selector e1 : eventmark reference t1 : absolute or relative time for first initiation t2 : time period for cyclic initiations e2 : reference to eventmark for overrun
6.5.1	CALL STRT(i,m) (start immediately)	
6.5.2	CALL STRTAF(i,t1,m) (start after time delay)	t1 : time delay before initiation
6.5.3	CALL STRTAT(i,t1,m) (start at absolute time)	t1 : absolute time for initiation
6.5.5	CALL CYCL(i,t2,m) (cyclic, with immediate first initiation)	t2 : length of time interval
6.5.6	CALL CYCLAF(i,t1,t2,m) (cyclic, with delayed first initiation)	t1 : time delay before first initiation t2 : length of time interval
6.5.7	CALL CYCLAT(i,t1,t2,m) (cyclic, with absolute time specification of first initiation)	t1 : absolute time for first initiation t2 : length of time interval
6.5.8	CALL CON(i,e,m) (establish event connection)	e : eventmark reference
6.6	CALL DSKED(i,s,e,m) (eliminate scheduling)	s : mode selector e : eventmark reference
6.6.1	CALL DCON(i,e,m) (eliminate event connection)	e : eventmark reference
6.6.2	CALL CANCEL(i,m) (eliminate time scheduling)	
6.7	CALL SUSPND(s,e,t,n,m) (suspend continuation of calling task for time period or until event)	s : mode selector e : reference to eventmark for end of delay t : time delay n : indicator for cause of end of delay
6.8.1	CALL HOLD(e,m) (suspend until event occurs)	e : reference to eventmark for end of delay
6.8.2	CALL DELAY(t,m) (suspend for a relative time)	t : time delay
6.9.1	CALL POST(e,m) (setting of an eventmark)	e : eventmark reference
6.9.2	CALL CLEAR(e,m) (resetting of an eventmark)	e : eventmark reference
6.9.3	TESTEM(e,m) (testing the state of an eventmark)	e : eventmark reference function value : condition of the eventmark
6.9.4	CALL MKEM(e,m) (setting the mask of an eventmark)	e : eventmark reference
6.9.5	CALL UNMKEM(e,m) (clearing the mask of an eventmark)	e : eventmark reference
6.10.1	CALL LOCK(r,m) (locking of a resourcemark)	r : resourcemark reference
6.10.2	CALL UNLOCK(r,m) (unlocking of a resourcemark)	r : resourcemark reference

iTeh STANDARD PREVIEW
(standards.iteh.ai)
ISO 7846:1985
https://standards.iteh.ai/catalog/standards/sist/8ea13c7e-6786-42d5-90a7-f8625ba54308/iso-7846-1985