```
machine Mach_IPC_Conds
//* ************************************************
//    The Event-B model of ARINC 653 Part 1
//    Created by Yongwang Zhao ( zhaoyongwang@gmail.com)
//    National Key Laboratory of Software Development Environment (NLSDE)
//    School of Computer and Engineering, Beihang University, Beijing, China
//    ************************************************/
 refines Mach_PartProc_Manage   sees Ctx_IPC

variables processes processes_of_partition  partition_mode process_state periodtype_of_process
          process_wait_type // mainproc_of_partition // the only one main proc of each partition
          locklevel_of_partition
          /* denotes the current lock level of the partition
              preemption_of_partitions */
          startcondition_of_partition
          /* denotes the reason the partition is started
              schedulable_of_partition //the scheduling of a partition is activated or disactivated? */
          basepriority_of_process // Denotes the capability of the process to manipulate other processes.
          period_of_process // Identifies the period of activation for a periodic process. A distinct and unique
value should be specified to designate the process as aperiodic
          timecapacity_of_process // Defines the elapsed time within which the process should complete its
```

execution.

deadline_of_process *// Specifies the type of deadline relating to the process, and may be "hard" or "soft".*

currentpriority_of_process *// Defines the priority with which the process may access and receive resources. It is set to base priority at initialization time and is dynamic at runtime.*

deadlinetime_of_process *// The deadline time is periodically evaluated by the operating system to determine whether the process is satisfactorily completing its processing within the allotted time.*

releasepoint_of_process

/ the release point of processes*

*nextreleasepoint_of_process // the next release point of processes */*

delaytime_of_process *// if the proc is delayed started, the delaytime should be saved(used when parttion START --> NORMAL)*

current_partition *// the partition in which a thread is now running. at each time, only one thread is running*

current_process

current_partition_flag *// true:indicate that the current_partition is valid, false: indicate NULL (unavailable)*

current_process_flag *// same as current partition flag*

clock_tick *// system clock ticks*

need_reschedule *// indicate the flag to reschedule after some events, for example suspend a thread*

need_procresch

```
        preempter_of_partition // the process who execute the lock_preemption (increase the locklevel and
disable scheduling), at most one preempter proc in a partition
        timeout_trigger // all processes waiting for resources with a timeout, will be triggered after the timeout
ellapsed.
        errorhandler_of_partition // each partition has one error handler at most. other error handler can be
created only after the previous handler is finished
        process_call_errorhandler
        /* error handler is created by a process, then the process is preempted by the error handler
            for inter-partition communication */
        ports
        /* the set of created ports
            RefreshPeriod_of_SamplingPorts */
        msgspace_of_samplingports
        /* the only one msg space of sampling ports
            needtrans_of_sourcesamplingport //indicate whether the msg in the source port has been
transfered to dest ports? */
        queue_of_queueingports // quediscipline_of_queueingports
        processes_waitingfor_queuingports // for intra-partition communication
        buffers blackboards semaphores events_ buffers_of_partition blackboards_of_partition
semaphores_of_partition events_of_partition MaxMsgNum_of_Buffers queue_of_buffers
        processes_waitingfor_buffers // quediscipline_of_buffers
```

msgspace_of_blackboards emptyindicator_of_blackboards processes_waitingfor_blackboards MaxValue_of_Semaphores

value_of_semaphores // quediscipline_of_semaphores

processes_waitingfor_semaphores state_of_events processes_waitingfor_events used_messages

invariants

@inv_used_msgs used_messages ∈ ℙ(**MESSAGES**)

@inv_ports ports∈ℙ(**PORTS**) // @inv_refreshprd_of_sampports RefreshPeriod_of_SamplingPorts ∈ SamplingPorts ⇸ ℕ1//partial function, the value will be assigned when created

@inv_msgsp_sampport msgspace_of_samplingports∈**SamplingPorts** ⇸ (**MESSAGES**×ℕ1)

/* partial function, each samp port has only one size space, and the space is null before writing, N1 is the written time*/

@inv_que_of_queports queue_of_queueingports∈**QueuingPorts**→ℙ(**MESSAGES**×ℕ1) // total function, each queport has a queue, although it is empty(so P, not P1). N1 is the written time

@inv_que_of_queports_finite ∀*p*·(*p*∈**QueuingPorts** ⇒ finite(queue_of_queueingports(*p*)))

@inv_processes_wf_qports processes_waitingfor_queuingports∈(processes×ℕ1×**MESSAGES**)⇸**QueuingPorts** // partial func: proc*(time of starting wait)*(msg of the wait proc to send)-->port

@inv_maxnummsg_queports ∀*p*·(*p*∈**QueuingPorts** ⇒ (finite(queue_of_queueingports(*p*)) ∧ card(queue_of_queueingports(*p*)) ≤ **MaxMsgNum_of_QueuingPorts**(*p*)))

@inv_buffers buffers∈ℙ(**BUFFERS**)

@inv_blackboards blackboards∈ℙ(**BLACKBOARDS**)

@inv_semaphores semaphores∈ℙ(**SEMAPHORES**)

@inv_events events_∈ℙ(**EVENTS**)

@inv_buf_part buffers_of_partition ∈ buffers → **PARTITIONS**

@inv_blkb_part blackboards_of_partition ∈ blackboards → **PARTITIONS**

@inv_evt_part events_of_partition ∈ events_ → **PARTITIONS**

@inv_semp_part semaphores_of_partition ∈ semaphores → **PARTITIONS**

@inv_maxnummsg_of_buf MaxMsgNum_of_Buffers ∈ buffers → ℕ1 *// @inv_quediscipline_of_buffers quediscipline_of_buffers∈buffers→QUEUING_DISCIPLINE*

@inv_queofbuffers queue_of_buffers∈buffers→ℙ(**MESSAGES**×ℕ1) *// total function*

@inv_queofbuffers_finite ∀*b*·(*b*∈buffers⇒finite(queue_of_buffers(*b*)))

@inv_procswfbuf processes_waitingfor_buffers∈(processes × (**BufferWaitingTypes**×ℕ1)× **MESSAGES**)⇸buffers *// partial func: proc*(waittype(send/rec)*(time of starting wait))*(msg of the wait proc to send)-->buffer*

@inv_maxnummsg_buffers ∀*p*·(*p*∈buffers ⇒ finite(queue_of_buffers(*p*)) ∧ card(queue_of_buffers(*p*)) ≤ MaxMsgNum_of_Buffers(*p*))

@inv_msgspace_blkb msgspace_of_blackboards∈blackboards ⇸ **MESSAGES** *// partial func: the blackboard may be empty*

@inv_emptyind_blkb emptyindicator_of_blackboards ∈ blackboards → **BLACKBOARD_INDICATORTYPE**

@inv_blkb_space_ind ∀*b*·(*b*∈blackboards⇒(emptyindicator_of_blackboards(*b*)=**BB_OCCUPIED** ⇔ *b*∈ dom(msgspace_of_blackboards)))

@inv_procswfblkb processes_waitingfor_blackboards∈processes⇸blackboards

/* partial func

@inv_quediscipline_of_semaphores quediscipline_of_semaphores∈semaphores→QUEUING_DISCIPLINE */

@inv_maxval_semp MaxValue_of_Semaphores ∈ semaphores → ℕ1

@inv_val_semp value_of_semaphores ∈ semaphores → ℕ

@inv_procswfsemp processes_waitingfor_semaphores ∈ (processes×ℕ1)⇸semaphores // partial func: proc*(time of starting wait)-->sem

@inv_procswfsemp_finite ∀s·(s∈semaphores⇒finite(processes_waitingfor_semaphores~[{s}]))

@inv_maxvalue_semophare ∀p·(p∈semaphores ⇒ value_of_semaphores(p) ≤ MaxValue_of_Semaphores(p))

@inv_stateofevt state_of_events ∈ events_ → **EVENT_STATE**

@inv_procswfevts processes_waitingfor_events ∈ processes⇸events_ // partial func

@inv_processes_wf_qports_part ∀port·(port∈**QueuingPorts** ⇒ (∀p,t,m·(p↦t↦m∈processes_waitingfor_queuingports~[{port}] ⇒ processes_of_partition(p)=**Ports_of_Partition**(port))))

events

event INITIALISATION **extends** INITIALISATION

then

@act301 ports ≔ ∅ // @act302 RefreshPeriod_of_SamplingPorts ≔ ∅

@act303 msgspace_of_samplingports ≔ ∅ // @act304 needtrans_of_sourcesamplingport ≔ ∅

@act305 queue_of_queueingports ≔ **QueuingPorts** ×{∅} // @act306 quediscipline_of_queueingports ≔ ∅

@act307 processes_waitingfor_queuingports ≔ ∅
@act308 buffers ≔ ∅
@act309 blackboards ≔ ∅
@act310 semaphores ≔ ∅
@act311 events_ ≔ ∅
@act312 buffers_of_partition ≔ ∅
@act313 blackboards_of_partition ≔ ∅
@act314 semaphores_of_partition ≔ ∅
@act3150 events_of_partition ≔ ∅
@act315 MaxMsgNum_of_Buffers ≔ ∅
@act316 queue_of_buffers ≔ ∅
@act317 processes_waitingfor_buffers ≔ ∅ *// @act318 quediscipline_of_buffers ≔ ∅*
@act319 msgspace_of_blackboards ≔∅
@act320 emptyindicator_of_blackboards ≔ ∅
@act321 processes_waitingfor_blackboards ≔ ∅
@act322 MaxValue_of_Semaphores ≔ ∅
@act323 value_of_semaphores ≔ ∅ *// @act324 quediscipline_of_semaphores ≔ ∅*
@act325 processes_waitingfor_semaphores ≔ ∅
@act326 state_of_events ≔ ∅
@act327 processes_waitingfor_events ≔ ∅
@act328 used_messages ≔ ∅

```
end

event create_sampling_port
  any port
  where
    @grd01 port∈SamplingPorts ∧ port∉ports
  then
    @act01 ports ≔ ports∪{port} // ports :| port∈ ports'
end

event write_sampling_message
  any port msg
  where
    @grd01 port∈SamplingPorts
    @grd03 Direction_of_Ports(port)=PORT_SOURCE
    @grd02 msg∈MESSAGES ∧ msg∉used_messages // @grd03 t∈ℕ
  then
    @act01 msgspace_of_samplingports:|∃t·(t∈ℕ ∧ port ↦ (msg↦t) ∈ msgspace_of_samplingports') //
@act01 msgspace_of_samplingports(port) ≔ msg ↦ t //msgspace_of_samplingports:|∃t·(t∈ℕ ∧ port ↦ (msg↦t)
∈ msgspace_of_samplingports')
    @act02 used_messages ≔ used_messages ∪ {msg} // used_messages :| msg∈used_messages'
```

```
end

event read_sampling_message
  any port m
  where
    @grd01 port∈SamplingPorts
    @grd03 Direction_of_Ports(port)=PORT_DESTINATION
    @grd02 port∈dom(msgspace_of_samplingports) ∧ (∃t·(t∈ℕ ∧ (m↦t)=msgspace_of_samplingports(port)))
end

event create_queuing_port
  any port
  where
    @grd01 port∈QueuingPorts ∧ port ∉ports
  then
    @act01 ports ≔ ports∪{port} // ports :| port∈ ports'
end

event send_queuing_message
  any port msg
  where
```

@grd01 *port*∈ports
@grd02 *port*∈**QueuingPorts**
@grd03 **Direction_of_Ports**(*port*)=**PORT_SOURCE**
@grd04 *msg*∈**MESSAGES** ∧ *msg*∉used_messages
@grd05 card(queue_of_queueingports(*port*))<**MaxMsgNum_of_QueuingPorts**(*port*) // *there is sufficient space in the port's message queue to accept the message*
@grd06 processes_waitingfor_queuingports~[{*port*}] = ∅
/* *no other process is waiting to send a message to that port*
*@grd07 t∈ℕ */

then
@act01 queue_of_queueingports:|∃t·(t∈ℕ ∧ (*msg*↦t) ∈ queue_of_queueingports'(*port*)) // *queue_of_queueingports(port) ≔ queue_of_queueingports(port) ∪ {msg↦t}//*
@act02 used_messages ≔ used_messages ∪ {*msg*} // *used_messages :| msg∈used_messages'*

end

event send_queuing_message_needwait **extends** req_busy_resource
any *port*
msg // *t*

where
@grd51 *port*∈ports

@grd52 *port*∈**QueuingPorts**

@grd53 **Direction_of_Ports**(*port*)=**PORT_SOURCE**

@grd54 *msg*∈**MESSAGES** ∧ *msg*∉used_messages

@grd55 card(queue_of_queueingports(*port*))=**MaxMsgNum_of_QueuingPorts**(*port*) ∨ processes_waitingfor_queuingports~[{*port*}] ≠ ∅ // @grd56 t∈ℕ

then

@act52 processes_waitingfor_queuingports :|(∃t·(t∈ℕ ∧ (current_process↦t↦*msg*)↦*port*∈processes_waitingfor_queuingports')) // processes_waitingfor_queuingports ≔ processes_waitingfor_queuingports ∪ {(current_process↦t↦msg)↦port}//

@act55 used_messages ≔ used_messages ∪ {*msg*} // used_messages :| msg∈used_messages'

end

event wakeup_waitproc_on_srcqueports **extends** resource_become_available

any *port*

msg // t

where

@grd502 *port* ∈ **Source_QueuingPorts** ∧ *port*∈ports

@grd504 card(queue_of_queueingports(*port*)) < **MaxMsgNum_of_QueuingPorts**(*port*)

@grd506 ∃*t*·(*t*∈ℕ ∧ (proc↦*t*↦ *msg*) ∈processes_waitingfor_queuingports~[{*port*}]) // @grd507 t∈ℕ

then

@act501 processes_waitingfor_queuingports :| (¬∃t·(t∈ℕ ∧ ((proc↦t↦ *msg*)↦*port*)∈ processes_waitingfor_queuingports')) // *processes_waitingfor_queuingports ≔ processes_waitingfor_queuingports ∖ {(proc↦t↦ msg)↦port}*//

@act506 queue_of_queueingports:|(∃t·(t∈ℕ ∧ (*msg*↦t)∈queue_of_queueingports'(*port*))) // *queue_of_queueingports(port) ≔ queue_of_queueingports(port)∪{msg↦t}* //

end

event wakeup_waitproc_on_destqueports **extends** resource_become_available

any *port*

msg // t t1

where

@grd502 *port* ∈ **Dest_QueuingPorts** ∧ *port*∈ports

@grd504 card(queue_of_queueingports(*port*)) > 0

@grd506 ∃*t*·(*t*∈ℕ ∧ (proc↦*t*↦ *msg*) ∈processes_waitingfor_queuingports~[{*port*}]) // *@grd507 t∈ℕ ∧ t1 ∈ℕ*

then

@act501 processes_waitingfor_queuingports :| (¬∃t·(t∈ℕ ∧ ((proc↦t↦ *msg*)↦*port*)∈ processes_waitingfor_queuingports')) // *processes_waitingfor_queuingports ≔ processes_waitingfor_queuingports ∖ {(proc↦t↦msg)↦port}*//

@act506 queue_of_queueingports:|(¬∃t·(t∈ℕ ∧(*msg*↦t)∈queue_of_queueingports'(*port*))) //

```
queue_of_queueingports(port) ≔ queue_of_queueingports(port)∖{msg ↦ t1}//
  end

  event receive_queuing_message
    any port
        msg // t

    where
      @grd01 port∈ports
      @grd02 port∈QueuingPorts
      @grd03 Direction_of_Ports(port)=PORT_DESTINATION
      @grd04 msg∈MESSAGES
      @grd06 card(queue_of_queueingports(port)) > 0
      @grd05 ∃t·(t∈ℕ ∧ (msg ↦t) ∈ queue_of_queueingports(port)) // @grd07 t∈ℕ
    then
      @act01 queue_of_queueingports:|(¬∃t·(t∈ℕ ∧ (msg↦t) ∈queue_of_queueingports'(port))) //
queue_of_queueingports(port) ≔ queue_of_queueingports(port) ∖ {msg↦t}//
  end

  event receive_queuing_message_needwait extends req_busy_resource
    any port
```

```
        msg // t

    where
      @grd502 port∈ports
      @grd503 port∈QueuingPorts
      @grd504 Direction_of_Ports(port)=PORT_DESTINATION
      @grd505 card(queue_of_queueingports(port)) = 0
      @grd506 msg∈MESSAGES // ∧ t∈ℕ
    then
      @act52 processes_waitingfor_queuingports :|(∃t,m·(t∈ℕ ∧ m∈MESSAGES ∧ (current_process↦t↦m)↦port
∈processes_waitingfor_queuingports')) // processes_waitingfor_queuingports ≔
processes_waitingfor_queuingports ∪ {(current_process↦t↦msg)↦port}//
  end

  event create_buffer
    any buf max_msg_size
    where
      @grd00 buf∈BUFFERS ∧ buf∉buffers
      @grd03 max_msg_size∈ℕ1
    then
      @act01 buffers ≔ buffers∪{buf} // buffers :| buf∈buffers'
```

```
    @act02 MaxMsgNum_of_Buffers(buf) ≔ max_msg_size
    @act05 queue_of_buffers(buf) ≔ ∅
    @act04 buffers_of_partition(buf) ≔ current_partition
  end

  event send_buffer
    any buf
        msg // t

    where
      @grd01 buf ∈ buffers
      @grd02 msg∈MESSAGES ∧ msg ∉ used_messages
      @grd05 card(queue_of_buffers(buf))<MaxMsgNum_of_Buffers(buf)
      /* buffer is not full
         @grd06 t∈ℕ */
    then
      @act01 queue_of_buffers:| ∃t·(t∈ℕ ∧ (msg↦t) ∈queue_of_buffers'(buf)) // queue_of_buffers(buf) ≔
queue_of_buffers(buf) ∪ {msg↦t}//
      @act05 used_messages ≔ used_messages ∪ {msg} // used_messages :| msg∈used_messages'
  end
```

```
event send_buffer_needwakeuprecvproc extends resource_become_available
  any buf msg
      m // t

  where
    @grd502 buf ∈ buffers
    @grd503 msg∈MESSAGES ∧ msg ∉ used_messages
    @grd504 card(queue_of_buffers(buf))<MaxMsgNum_of_Buffers(buf) // buffer is not full
    @grd505 card(processes_waitingfor_buffers~[{buf}])>0
    @grd506 m∈MESSAGES // ∧ t∈ℕ
  then
    @act501 used_messages ≔ used_messages ∪ {msg} // used_messages :| msg∈used_messages'
    @act502 processes_waitingfor_buffers:|(¬∃t,m·(t∈ℕ ∧ m∈MESSAGES ∧ (proc↦(WAITING_R↦t)↦m)↦buf∈
processes_waitingfor_buffers')) // processes_waitingfor_buffers ≔ processes_waitingfor_buffers ∖ {(proc↦
(WAITING_R↦t)↦m) ↦ buf}//
end

event send_buffer_withfull extends req_busy_resource
  any buf
      msg // t
```

```
    where
      @grd503 buf ∈ buffers
      @grd502 msg∈MESSAGES ∧ msg ∉ used_messages
      @grd505 card(queue_of_buffers(buf))=MaxMsgNum_of_Buffers(buf)
      /* buffer is  full
         @grd506 t∈ℕ */
    then
      @act501 used_messages ≔ used_messages ∪ {msg} // used_messages :| msg∈used_messages'
      @act502 processes_waitingfor_buffers:|(∃t·(t∈ℕ ∧ (current_process ↦ (WAITING_W↦t)↦msg)↦buf∈
processes_waitingfor_buffers')) // processes_waitingfor_buffers ≔ processes_waitingfor_buffers ∪
{(current_process ↦ (WAITING_W↦t)↦msg)↦buf}//
  end

  event receive_buffer
    any buf
        msg // t

    where
      @grd01 buf ∈ buffers
      @grd02 msg∈MESSAGES
      @grd03 card(queue_of_buffers(buf))>0
```

```
    /* buffer is not empty
       @grd04 t∈ℕ */
  then
    @act01 queue_of_buffers:|(¬∃t·(t∈ℕ ∧ (msg↦t)∈queue_of_buffers'(buf))) // queue_of_buffers(buf) ≔
queue_of_buffers(buf) ∖ {msg↦t}//
end

event receive_buffer_needwakeupsendproc extends resource_become_available
  any buf msg
      m // t_

  where
    @grd506 buf ∈ buffers
    @grd502 msg∈MESSAGES
    @grd503 card(queue_of_buffers(buf))>0 // buffer is not empty
    @grd505 card(processes_waitingfor_buffers~[{buf}])>0
    @grd507 m∈MESSAGES // ∧ t_∈ℕ
  then
    @act501 queue_of_buffers:|(¬∃t·(t∈ℕ ∧ msg↦t∈queue_of_buffers'(buf))) // queue_of_buffers(buf) ≔
queue_of_buffers(buf)∪{m↦t_}//
    @act502 processes_waitingfor_buffers:|(¬∃t,m·(t∈ℕ ∧ m∈MESSAGES∧(proc↦(WAITING_W↦t)↦m)↦buf∈
```

```
processes_waitingfor_buffers)) // processes_waitingfor_buffers ≔ processes_waitingfor_buffers ∖ {(proc↦
(WAITING_W↦t_)↦m) ↦ buf}//
  end

  event receive_buffer_whenempty extends req_busy_resource
    any buf
        msg // t

    where
      @grd504 buf ∈ buffers
      @grd502 card(queue_of_buffers(buf))=0 // buffer is empty
      @grd503 msg∈MESSAGES // @grd505 t∈ℕ
    then
      @act501 processes_waitingfor_buffers:|(∃t,m·(t∈ℕ ∧m∈MESSAGES∧ (current_process ↦ (WAITING_R↦t)
↦m)↦buf∈processes_waitingfor_buffers')) // processes_waitingfor_buffers ≔ processes_waitingfor_buffers ∪
{(current_process ↦(WAITING_R↦t)↦msg) ↦ buf}//
  end

  event create_blackboard
    any bb
    where
```

```
    @grd00 bb∈BLACKBOARDS ∧ bb∉blackboards
  then
    @act01 blackboards ≔ blackboards∪{bb} // blackboards:| bb∈blackboards
    @act04 emptyindicator_of_blackboards(bb)≔BB_EMPTY
    @act03 blackboards_of_partition(bb)≔ current_partition
end

event display_blackboard
  any bb msg
  where
    @grd00 bb∈blackboards
    @grd02 msg∈MESSAGES ∧ msg∉used_messages
    @grd03 processes_waitingfor_blackboards~[{bb}] = ∅
  then
    @act01 msgspace_of_blackboards(bb) ≔ msg // msgspace_of_blackboards :|
msgspace_of_blackboards'(bb) = msg
    @act02 used_messages ≔ used_messages ∪ {msg} // used_messages :| msg∈used_messages'
    @act03 emptyindicator_of_blackboards(bb) ≔ BB_OCCUPIED
end

event display_blackboard_needwakeuprdprocs extends resource_become_available2
```

```
    any bb msg
    where
      @grd500 bb∈blackboards
      @grd504 msg∈MESSAGES ∧ msg∉used_messages
      @grd505 processes_waitingfor_blackboards~[{bb}] ≠ ∅
    then
      @act501 msgspace_of_blackboards(bb) ≔ msg // msgspace_of_blackboards :|
msgspace_of_blackboards'(bb) = msg
      @act502 processes_waitingfor_blackboards ≔ procs⩤processes_waitingfor_blackboards //
processes_waitingfor_blackboards :| ∀p·(p∈procs⇒p↦bb∉processes_waitingfor_blackboards')
      @act504 used_messages ≔ used_messages ∪ {msg} // used_messages :| msg∈used_messages'
      @act503 emptyindicator_of_blackboards(bb) ≔ BB_OCCUPIED
  end

  event read_blackboard
    any bb msg
    where
      @grd00 bb∈blackboards
      @grd02 msg∈MESSAGES
      @grd03 emptyindicator_of_blackboards(bb) = BB_OCCUPIED
  end
```

```
  event read_blackboard_whenempty extends req_busy_resource
    any bb
    where
      @grd500 bb∈blackboards
      @grd502 emptyindicator_of_blackboards(bb) = BB_EMPTY
    then
      @act501 processes_waitingfor_blackboards ≔ processes_waitingfor_blackboards ∪ {current_process ↦
bb} // processes_waitingfor_blackboards:| current_process ↦ bb∈processes_waitingfor_blackboards'
  end

  event clear_blackboard
    any bb
    where
      @grd00 bb∈blackboards
    then
      @act01 emptyindicator_of_blackboards(bb) ≔ BB_EMPTY
  end

  event create_semaphore
    any sem maxval currentval
```

```
  where
    @grd01 sem∈SEMAPHORES ∧ sem∉semaphores
    @grd07 maxval∈ℕ1
    @grd08 currentval∈ℕ ∧ currentval≤maxval
  then
    @act01 semaphores ≔ semaphores∪{sem} // semaphores :| sem∈semaphores'
    @act03 value_of_semaphores(sem)≔currentval
    @act04 MaxValue_of_Semaphores(sem)≔maxval
    @act05 semaphores_of_partition(sem) ≔current_partition
end

event wait_semaphore
  any sem
  where
    @grd00 sem∈semaphores
    @grd02 value_of_semaphores(sem) > 0
  then
    @act01 value_of_semaphores(sem) ≔ value_of_semaphores(sem) −1
end

event wait_semahpore_whenzero extends req_busy_resource
```

```
    any sem // t

    where
      @grd500 sem∈semaphores
      @grd504 value_of_semaphores(sem) = 0 // @grd501 t∈ℕ
    then
      @act501 processes_waitingfor_semaphores :| (∃t·(t∈ℕ ∧ (current_process ↦ t)↦sem∈
processes_waitingfor_semaphores')) // processes_waitingfor_semaphores ≔ processes_waitingfor_semaphores
∪ {(current_process ↦ t)↦ sem}//
  end

  event signal_semaphore
    any sem
    where
      @grd00 sem∈semaphores
      @grd02 value_of_semaphores(sem) ≠ MaxValue_of_Semaphores(sem)
      @grd03 processes_waitingfor_semaphores~[{sem}] = ∅
    then
      @act01 value_of_semaphores(sem) ≔ value_of_semaphores(sem) + 1
  end
```

```
event signal_semaphore_needwakeupproc extends resource_become_available
  any sem // t

  where
    @grd500 sem∈semaphores
    @grd503 value_of_semaphores(sem) ≠ MaxValue_of_Semaphores(sem)
    @grd506 card(processes_waitingfor_semaphores~[{sem}])>0 // @grd504 t∈ℕ
  then
    @act501 processes_waitingfor_semaphores :| (¬∃t·(t∈ℕ ∧ proc ↦ t↦sem∈
processes_waitingfor_semaphores')) // processes_waitingfor_semaphores ≔ processes_waitingfor_semaphores ∖
{proc ↦ t↦sem}//
end

event create_event
  any ev
  where
    @grd01 ev∈EVENTS ∧ ev ∉ events_
  then
    @act01 events_ ≔ events_∪{ev}
    @act02 state_of_events(ev) ≔ EVENT_DOWN
    @act03 events_of_partition(ev) ≔ current_partition
```

```
end

event set_event
  any ev
  where
    @grd00 ev∈events_
    @grd03 processes_waitingfor_events~[{ev}] = ∅
  then
    @act01 state_of_events(ev)≔EVENT_UP
end

event set_event_needwakeupprocs extends resource_become_available2
  any ev
  where
    @grd500 ev∈events_
    @grd503 processes_waitingfor_events~[{ev}] ≠ ∅
  then
    @act501 state_of_events(ev)≔EVENT_UP
    @act503 processes_waitingfor_events ≔ procs⩤processes_waitingfor_events //
processes_waitingfor_events :| ∀p·(p∈procs⇒p↦ev∉processes_waitingfor_events')
end
```

```
event reset_event
  any ev
  where
    @grd00 ev∈events_
  then
    @act01 state_of_events(ev):=EVENT_DOWN
end

event wait_event
  any ev
  where
    @grd00 ev∈events_
    @grd02 state_of_events(ev)=EVENT_UP
end

event wait_event_whendown extends req_busy_resource
  any ev
  where
    @grd500 ev∈events_
    @grd504 state_of_events(ev)=EVENT_DOWN
```

```
    then
      @act501 processes_waitingfor_events ≔ processes_waitingfor_events ∪ {current_process ↦ ev} //
processes_waitingfor_events:| current_process ↦ ev ∈ processes_waitingfor_events'
  end

  event ticktock // timer interrupt event, triggered by the timer in hardware. one tick in each ONE_TICK_TIME
  extends ticktock
  end

  event partition_schedule extends partition_schedule
  end

  event process_schedule // if there is not error handler and preempter in this partition
  extends process_schedule
  end

  event run_errorhandler_preempter // if there is the error handler, it is executed, otherwise the preempter is
executed
  extends run_errorhandler_preempter
  end
```

```
event get_partition_status extends get_partition_status
end

event set_partition_mode_to_idle // shutdown the partition
extends set_partition_mode_to_idle
  then
    @act501 ports ≔ ports\Ports_of_Partition~[{part}] // @act602 RefreshPeriod_of_SamplingPorts ≔
Ports_of_Partition~[{part}] ⩤ RefreshPeriod_of_SamplingPorts
    @act503 msgspace_of_samplingports ≔ Ports_of_Partition~[{part}] ⩤ msgspace_of_samplingports //
@act604 needtrans_of_sourcesamplingport ≔ Ports_of_Partition~[{part}] ⩤ needtrans_of_sourcesamplingport
    @act505 queue_of_queueingports ≔ Ports_of_Partition~[{part}] ⩤ queue_of_queueingports // @act606
quediscipline_of_queueingports ≔ Ports_of_Partition~[{part}] ⩤ quediscipline_of_queueingports
    @act507 processes_waitingfor_queuingports ≔
processes_waitingfor_queuingports⩥Ports_of_Partition~[{part}]
    @act508 buffers ≔ buffers\buffers_of_partition~[{part}]
    @act509 blackboards ≔ blackboards\blackboards_of_partition~[{part}]
    @act510 semaphores ≔ semaphores\semaphores_of_partition~[{part}]
    @act511 events_ ≔ events_\events_of_partition~[{part}]
    @act512 buffers_of_partition ≔ buffers_of_partition⩥{part}
    @act513 blackboards_of_partition ≔ blackboards_of_partition⩥{part}
    @act514 semaphores_of_partition ≔ semaphores_of_partition⩥{part}
```

```
        @act515 events_of_partition ≔ events_of_partition⩥{part}
        @act516 MaxMsgNum_of_Buffers ≔ buffers_of_partition~[{part}] ⩤ MaxMsgNum_of_Buffers
        @act517 queue_of_buffers ≔ buffers_of_partition~[{part}]⩤  queue_of_buffers
        @act518 processes_waitingfor_buffers ≔ processes_waitingfor_buffers ⩥ buffers_of_partition~[{part}] //
@act619 quediscipline_of_buffers ≔ buffers_of_partition~[{part}]⩤quediscipline_of_buffers
        @act520 msgspace_of_blackboards ≔ blackboards_of_partition~[{part}] ⩤ msgspace_of_blackboards
        @act521 emptyindicator_of_blackboards ≔ blackboards_of_partition~[{part}] ⩤
emptyindicator_of_blackboards
        @act522 processes_waitingfor_blackboards ≔ processes_waitingfor_blackboards ⩥
blackboards_of_partition~[{part}]
        @act523 MaxValue_of_Semaphores ≔ semaphores_of_partition~[{part}] ⩤ MaxValue_of_Semaphores
        @act524 value_of_semaphores ≔ semaphores_of_partition~[{part}] ⩤ value_of_semaphores // @act625
quediscipline_of_semaphores ≔ semaphores_of_partition~[{part}] ⩤ quediscipline_of_semaphores
        @act526 processes_waitingfor_semaphores ≔ processes_waitingfor_semaphores ⩥
semaphores_of_partition~[{part}]
        @act527 state_of_events ≔ events_of_partition~[{part}] ⩤ state_of_events
        @act528 processes_waitingfor_events ≔ processes_waitingfor_events ⩥ events_of_partition~[{part}]
    end

    event set_partition_mode_to_normal extends set_partition_mode_to_normal
    end
```

event set_partition_mode_to_coldstart **extends** set_partition_mode_to_coldstart

then

@act501 ports ≔ ports∖**Ports_of_Partition**~[{part}] *// @act602 RefreshPeriod_of_SamplingPorts ≔ Ports_of_Partition~[{part}] ⩤ RefreshPeriod_of_SamplingPorts*

@act503 msgspace_of_samplingports ≔ **Ports_of_Partition**~[{part}] ⩤ msgspace_of_samplingports *// @act604 needtrans_of_sourcesamplingport ≔ Ports_of_Partition~[{part}] ⩤ needtrans_of_sourcesamplingport*

@act505 queue_of_queueingports ≔ **Ports_of_Partition**~[{part}] ⩤ queue_of_queueingports *// @act606 quediscipline_of_queueingports ≔ Ports_of_Partition~[{part}] ⩤ quediscipline_of_queueingports*

@act507 processes_waitingfor_queuingports ≔ processes_waitingfor_queuingports⩥**Ports_of_Partition**~[{part}]

@act508 buffers ≔ buffers∖buffers_of_partition~[{part}]

@act509 blackboards ≔ blackboards∖blackboards_of_partition~[{part}]

@act510 semaphores ≔ semaphores∖semaphores_of_partition~[{part}]

@act511 events_ ≔ events_∖events_of_partition~[{part}]

@act512 buffers_of_partition ≔ buffers_of_partition⩥{part}

@act513 blackboards_of_partition ≔ blackboards_of_partition⩥{part}

@act514 semaphores_of_partition ≔ semaphores_of_partition⩥{part}

@act515 events_of_partition ≔ events_of_partition⩥{part}

@act516 MaxMsgNum_of_Buffers ≔ buffers_of_partition~[{part}] ⩤ MaxMsgNum_of_Buffers

@act517 queue_of_buffers ≔ buffers_of_partition~[{part}]⩤ queue_of_buffers

@act518 processes_waitingfor_buffers ≔ processes_waitingfor_buffers ⩤ buffers_of_partition~[{part}] // *@act619 quediscipline_of_buffers ≔ buffers_of_partition~[{part}]⩤quediscipline_of_buffers*

@act520 msgspace_of_blackboards ≔ blackboards_of_partition~[{part}] ⩤ msgspace_of_blackboards

@act521 emptyindicator_of_blackboards ≔ blackboards_of_partition~[{part}] ⩤ emptyindicator_of_blackboards

@act522 processes_waitingfor_blackboards ≔ processes_waitingfor_blackboards ⩤ blackboards_of_partition~[{part}]

@act523 MaxValue_of_Semaphores ≔ semaphores_of_partition~[{part}] ⩤ MaxValue_of_Semaphores

@act524 value_of_semaphores ≔ semaphores_of_partition~[{part}] ⩤ value_of_semaphores // *@act625 quediscipline_of_semaphores ≔ semaphores_of_partition~[{part}] ⩤ quediscipline_of_semaphores*

@act526 processes_waitingfor_semaphores ≔ processes_waitingfor_semaphores ⩤ semaphores_of_partition~[{part}]

@act527 state_of_events ≔ events_of_partition~[{part}] ⩤ state_of_events

@act528 processes_waitingfor_events ≔ processes_waitingfor_events ⩤ events_of_partition~[{part}]

end

event set_partition_mode_to_warmstart **extends** set_partition_mode_to_warmstart

then

@act501 ports ≔ ports⩥**Ports_of_Partition**~[{part}] // *@act602 RefreshPeriod_of_SamplingPorts ≔ Ports_of_Partition~[{part}] ⩤ RefreshPeriod_of_SamplingPorts*

@act503 msgspace_of_samplingports ≔ **Ports_of_Partition**~[{part}] ⩤ msgspace_of_samplingports //

@act604 needtrans_of_sourcesamplingport ≔ Ports_of_Partition~[{part}] ⩤ needtrans_of_sourcesamplingport

@act505 queue_of_queueingports ≔ **Ports_of_Partition**~[{part}] ⩤ queue_of_queueingports *// @act606 quediscipline_of_queueingports ≔ Ports_of_Partition~[{part}] ⩤ quediscipline_of_queueingports*

@act507 processes_waitingfor_queuingports ≔ processes_waitingfor_queuingports⩥**Ports_of_Partition**~[{part}]

@act508 buffers ≔ buffers\buffers_of_partition~[{part}]

@act509 blackboards ≔ blackboards\blackboards_of_partition~[{part}]

@act510 semaphores ≔ semaphores\semaphores_of_partition~[{part}]

@act511 events_ ≔ events_\events_of_partition~[{part}]

@act512 buffers_of_partition ≔ buffers_of_partition⩥{part}

@act513 blackboards_of_partition ≔ blackboards_of_partition⩥{part}

@act514 semaphores_of_partition ≔ semaphores_of_partition⩥{part}

@act515 events_of_partition ≔ events_of_partition⩥{part}

@act516 MaxMsgNum_of_Buffers ≔ buffers_of_partition~[{part}] ⩤ MaxMsgNum_of_Buffers

@act517 queue_of_buffers ≔ buffers_of_partition~[{part}]⩤ queue_of_buffers

@act518 processes_waitingfor_buffers ≔ processes_waitingfor_buffers ⩥ buffers_of_partition~[{part}] *// @act619 quediscipline_of_buffers ≔ buffers_of_partition~[{part}]⩤quediscipline_of_buffers*

@act520 msgspace_of_blackboards ≔ blackboards_of_partition~[{part}] ⩤ msgspace_of_blackboards

@act521 emptyindicator_of_blackboards ≔ blackboards_of_partition~[{part}] ⩤ emptyindicator_of_blackboards

@act522 processes_waitingfor_blackboards ≔ processes_waitingfor_blackboards ⩥

```
blackboards_of_partition~[{part}]
        @act523 MaxValue_of_Semaphores ≔ semaphores_of_partition~[{part}] ⩤ MaxValue_of_Semaphores
        @act524 value_of_semaphores ≔ semaphores_of_partition~[{part}] ⩤ value_of_semaphores // @act625
quediscipline_of_semaphores ≔ semaphores_of_partition~[{part}] ⩤ quediscipline_of_semaphores
        @act526 processes_waitingfor_semaphores ≔ processes_waitingfor_semaphores ⩥
semaphores_of_partition~[{part}]
        @act527 state_of_events ≔ events_of_partition~[{part}] ⩤ state_of_events
        @act528 processes_waitingfor_events ≔ processes_waitingfor_events ⩥ events_of_partition~[{part}]
    end

    event get_process_id extends get_process_id
    end

    event get_process_status extends get_process_status
    end

    event create_process extends create_process
    end

    event set_priority extends set_priority
    end
```

```
event suspend_self
/* extends suspend_self
    any timeout timeouttrig waittype */
extends suspend_self
end

event suspend // extends suspend
extends suspend
end

event resume // extends resume
extends resume
end

event stop_self extends stop_self
end

event stop extends stop
  then
    @act501 processes_waitingfor_queuingports :| ∀p,t,m,pt·((p↦t↦m)↦pt∈processes_waitingfor_queuingports
```

⇒ (p=proc⇒(p↦t↦m)↦pt∉processes_waitingfor_queuingports')∧(p≠proc⇒(p↦t↦m)↦pt∈
processes_waitingfor_queuingports'))

@act502 processes_waitingfor_buffers :| ∀p,w,t,m,pt·((p↦(w↦t)↦m)↦pt∈processes_waitingfor_buffers ⇒
(p=proc⇒(p↦(w↦t)↦m)↦pt∉processes_waitingfor_buffers')∧(p≠proc⇒(p↦(w↦t)↦m)↦pt∈
processes_waitingfor_buffers'))

@act503 processes_waitingfor_blackboards ≔ {proc}⩤processes_waitingfor_blackboards

@act504 processes_waitingfor_semaphores :| ∀p,t,pt·((p↦t)↦pt∈processes_waitingfor_semaphores ⇒
(p=proc⇒(p↦t)↦pt∉processes_waitingfor_semaphores')∧(p≠proc⇒(p↦t)↦pt∈processes_waitingfor_semaphores'))

@act505 processes_waitingfor_events ≔ {proc}⩤processes_waitingfor_events

end

event start_aperiodprocess_instart

/ start an aperiodic process in COLD_START or WARM_START mode
extends start */*

extends start_aperiodprocess_instart

end

event start_aperiodprocess_innormal

/ start an aperiodic process in NORMAL mode
extends start */*

extends start_aperiodprocess_innormal

```
end

event start_periodprocess_instart
/* start a periodic process in COLD_START or WARM_START mode
    extends start */
extends start_periodprocess_instart
end

event start_periodprocess_innormal
/* start a periodic process in NORMAL mode
    extends start */
extends start_periodprocess_innormal
end

event delaystart_aperiodprocess_instart  // extends delayed_start
extends delaystart_aperiodprocess_instart
end

event delaystart_aperiodprocess_innormal
/* if delaytime=0, then immediately transit to READY, this is modelled in start_aperiod_process_whennormal
    extends delayed_start
```

```
    any delaytime */
extends delaystart_aperiodprocess_innormal
end

event delaystart_periodprocess_instart  // extends delayed_start
extends delaystart_periodprocess_instart
end

event delaystart_periodprocess_innormal  // extends delayed_start
extends delaystart_periodprocess_innormal
end

event lock_preemption extends lock_preemption
end

event unlock_preemption extends unlock_preemption
end

event get_my_id extends get_my_id
end
```

```
event timed_wait extends timed_wait
end

event period_wait extends period_wait
end

event get_time extends get_time
end

event replenish extends replenish
end

event aperiodicprocess_finished  extends aperiodicprocess_finished
end

event periodicprocess_finished  extends periodicprocess_finished
end

event time_out // should refined to support remove process on waiting queue of comm resources
extends time_out
  then
```

```
        @act501 processes_waitingfor_queuingports :| ∀p,t,m,pt·((p↦t↦m)↦pt∈processes_waitingfor_queuingports
⇒ (p=proc⇒(p↦t↦m)↦pt∉processes_waitingfor_queuingports')∧(p≠proc⇒(p↦t↦m)↦pt∈
processes_waitingfor_queuingports'))
        @act502 processes_waitingfor_buffers :| ∀p,w,t,m,pt·((p↦(w↦t)↦m)↦pt∈processes_waitingfor_buffers ⇒
(p=proc⇒(p↦(w↦t)↦m)↦pt∉processes_waitingfor_buffers')∧(p≠proc⇒(p↦(w↦t)↦m)↦pt∈
processes_waitingfor_buffers'))
        @act503 processes_waitingfor_blackboards ≔ {proc}⩤processes_waitingfor_blackboards
        @act504 processes_waitingfor_semaphores :| ∀p,t,pt·((p↦t)↦pt∈processes_waitingfor_semaphores ⇒
(p=proc⇒(p↦t)↦pt∉processes_waitingfor_semaphores')∧(p≠proc⇒(p↦t)↦pt∈processes_waitingfor_semaphores'))
        @act505 processes_waitingfor_events ≔ {proc}⩤processes_waitingfor_events
    end

    event periodicproc_reach_releasepoint extends periodicproc_reach_releasepoint
    end

    event coldstart_partition_fromidle extends coldstart_partition_fromidle
    end

    event warmstart_partition_fromidle extends warmstart_partition_fromidle
    end
end
```