```
machine Mach_PartProc_Trans_with_Events
//* *************************************************
//     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
//     ************************************************/
//this refinement defines the events to trigger the partition mode and process state transitions
//according to ARINC653 "Figure 2.3.1.4  –Partition Operating Modes and Transitions" and
//"Figure 2.3  –  Process States and State Transitions in Accordance with the Modes of the Partition "

refines Mach_PartProc_Trans   sees Ctx_PartProc_with_Events

variables processes
            processes_of_partition  // system_has_inited
            partition_mode
            process_state

            periodtype_of_process

invariants
```

```
    @inv_pertype_of_proc   periodtype_of_process  ∈  processes  → PROC_PERIOD_TYPE
    @inv_onlyone_runproc ∀p1,p2·(p1∈processes  ∧  p2∈processes  ∧  process_state(p1)=PS_Running  ∧
process_state(p2)=PS_Running  ⇒ p1=p2) //card(process_state∼[{PS_Running}])  ≤  1 // at most one RUNNING
proc in a single processor system


events
  event INITIALISATION extends INITIALISATION
  then
    @act11 periodtype_of_process  ≔ ∅
  end

  event partition_schedule
  any part
  when
    @grd01 part∈PARTITIONS
    @grd02 partition_mode(part)  = PM_NORMAL  ∨  partition_mode(part)  = PM_WARM_START  ∨
partition_mode(part)  = PM_COLD_START
  end

  event process_schedule
```

```
extends process_schedule
end

event create_process extends create_process
any ptype
where
  @grd11 ptype∈PROC_PERIOD_TYPE
then
  @act11 periodtype_of_process(proc) ≔ ptype
end

event set_partition_mode_to_idle  extends partition_modetransition_to_idle
then
  @act31 periodtype_of_process ≔ procs ⩤ periodtype_of_process
end

event set_partition_mode_to_normal  extends partition_modetransition_to_normal
end

event set_partition_mode_to_coldstart  extends partition_modetransition_to_coldstart
then
```

```
  @act31 periodtype_of_process  ≔ procs ⩤ periodtype_of_process
end

event set_partition_mode_to_warmstart  extends partition_modetransition_to_warmstart
then
  @act31 periodtype_of_process  ≔ procs ⩤ periodtype_of_process
end

event coldstart_partition_fromidle // idle transit to cold_start or warm_start
//The only mechanism available to transition from the IDLE mode is an action external to the
//partition, such as power interrupt, core module reset, or application reset, if an external
//means exist.
//So, we just reserve this external event
extends partition_modetransition_idle_to_coldstart
end

event warmstart_partition_fromidle// idle transit to cold_start or warm_start
//The only mechanism available to transition from the IDLE mode is an action external to the
//partition, such as power interrupt, core module reset, or application reset, if an external
//means exist.
//So, we just reserve this external event
```

```
extends partition_modetransition_idle_to_warmstart
end

event suspend_self
refines process_state_transition
any part proc newstate
where
  @grd01 part ∈ PARTITIONS
  @grd02 proc ∈ processes
  @grd03 newstate ∈ PROCESS_STATES
  @grd06 processes_of_partition(proc) = part
  @grd31 partition_mode(part) = PM_NORMAL
  @grd32 process_state(proc) = PS_Running
  @grd33 newstate = PS_Suspend
  @grd34 periodtype_of_process(proc) = APERIOD_PROC
then
  @act11 process_state(proc) ≔ newstate
end

event suspend
refines process_state_transition
```

```
  any part proc newstate
  where
    @grd01 part ∈ PARTITIONS
    @grd02 proc ∈ processes
    @grd03 newstate ∈ PROCESS_STATES
    @grd06 processes_of_partition(proc) = part
     @grd07 partition_mode(part) = PM_NORMAL ∨ partition_mode(part) = PM_WARM_START ∨
partition_mode(part) = PM_COLD_START
    @grd31 partition_mode(part) = PM_NORMAL ⇒ (process_state(proc) = PS_Ready ∧ newstate =
PS_Suspend)∨ (process_state(proc) = PS_Waiting ∧ newstate = PS_WaitandSuspend)
    @grd32 (partition_mode(part) = PM_COLD_START ∨ partition_mode(part) = PM_WARM_START)⇒
(process_state(proc) = PS_Waiting ∧ newstate = PS_WaitandSuspend)
    @grd34 periodtype_of_process(proc) = APERIOD_PROC
  then
    @act11 process_state(proc) ≔ newstate
  end

  event resume
  refines process_state_transition
  any part proc newstate
  where
```

```
    @grd01 part ∈ PARTITIONS
    @grd02 proc ∈ processes
    @grd03 newstate ∈ PROCESS_STATES
    @grd06 processes_of_partition(proc) = part
    //@grd31 partition_mode(part) = PM_NORMAL
    //@grd32  (process_state(proc) = PS_Suspend ∧ newstate = PS_Ready) ∨  (process_state(proc) =
PS_WaitandSuspend ∧ newstate = PS_Waiting)
    //these two lines are from ARINC 653, the state transition fig does not mention the RESUME in START mode.
    //the next two lines are correct
    @grd31 partition_mode(part) = PM_NORMAL ⇒ ((process_state(proc) = PS_Suspend ∧ newstate =
PS_Ready) ∨  (process_state(proc) = PS_WaitandSuspend ∧ newstate = PS_Waiting))
    @grd32 (partition_mode(part) = PM_COLD_START ∨  partition_mode(part) = PM_WARM_START)∧
partition_mode(part) ≠ PM_NORMAL⇒  (process_state(proc) = PS_WaitandSuspend ∧ newstate =
PS_Waiting)


    @grd34 periodtype_of_process(proc) = APERIOD_PROC
  then
    @act11 process_state(proc) ≔ newstate
  end
```

```
event stop_self
refines process_state_transition
any part proc newstate
where
  @grd01 part ∈ PARTITIONS
  @grd02 proc ∈ processes
  @grd03 newstate ∈ PROCESS_STATES
  @grd06 processes_of_partition(proc) = part
  @grd30 partition_mode(part) = PM_NORMAL
   @grd31 process_state(proc) = PS_Running ∧ newstate = PS_Dormant
then
  @act11 process_state(proc) ≔ newstate
end

event stop
refines process_state_transition
any part proc newstate
where
  @grd01 part ∈ PARTITIONS
  @grd02 proc ∈ processes
  @grd03 newstate ∈ PROCESS_STATES
```

```
    @grd06 processes_of_partition(proc) = part
    @grd07 partition_mode(part) = PM_NORMAL ∨ partition_mode(part) = PM_WARM_START ∨
partition_mode(part) = PM_COLD_START
    @grd31 partition_mode(part) = PM_NORMAL ⇒ ((process_state(proc) = PS_Ready ∨ process_state(proc) =
PS_Waiting ∨ process_state(proc) = PS_Suspend ∨ process_state(proc) = PS_WaitandSuspend) ∧ newstate =
PS_Dormant)
    @grd32 (partition_mode(part) = PM_COLD_START ∨ partition_mode(part) = PM_WARM_START)⇒
((process_state(proc) = PS_Waiting ∨ process_state(proc) = PS_WaitandSuspend) ∧ newstate = PS_Dormant)
  then
    @act11 process_state(proc) ≔ newstate
  end

  event start
  refines process_state_transition
  any part proc newstate
  where
    @grd01 part ∈ PARTITIONS
    @grd02 proc ∈ processes
    @grd03 newstate ∈ PROCESS_STATES
    @grd06 processes_of_partition(proc) = part
    @grd07 partition_mode(part) = PM_NORMAL ∨ partition_mode(part) = PM_WARM_START ∨
```

```
partition_mode(part) = PM_COLD_START
    @grd31 partition_mode(part) = PM_NORMAL ⇒  (process_state(proc) = PS_Dormant ∧
                     ((periodtype_of_process(proc) = APERIOD_PROC ⇒ newstate = PS_Ready) ∧
(periodtype_of_process(proc) = PERIOD_PROC ⇒ newstate = PS_Waiting)))
    @grd32 (partition_mode(part) = PM_COLD_START ∨  partition_mode(part) = PM_WARM_START)⇒
(process_state(proc) = PS_Dormant ∧ newstate = PS_Waiting)
  then
    @act11 process_state(proc) ≔ newstate
  end

  event delayed_start
  refines process_state_transition
  any part proc newstate
  where
    @grd01 part ∈ PARTITIONS
    @grd02 proc ∈ processes
    @grd03 newstate ∈ PROCESS_STATES
    @grd06 processes_of_partition(proc) = part
    @grd07 partition_mode(part) = PM_NORMAL ∨ partition_mode(part) = PM_WARM_START ∨
partition_mode(part) = PM_COLD_START
```

```
    //@grd31 partition_mode(part) = PM_NORMAL ⇒  (process_state(proc) = PS_Dormant ∧ newstate =
PS_Waiting)
    //this line is correct, the next line is from ARINC653
    @grd30 partition_mode(part) = PM_NORMAL ⇒  (periodtype_of_process(proc) = PERIOD_PROC ∧
process_state(proc) = PS_Dormant ∧ newstate = PS_Waiting)

    @grd32 (partition_mode(part) = PM_COLD_START ∨  partition_mode(part) = PM_WARM_START)⇒
(process_state(proc) = PS_Dormant ∧ newstate = PS_Waiting)
  then
    @act11 process_state(proc) ≔ newstate
  end

  event timed_wait
  refines process_state_transition
  any part proc newstate
  where
    @grd01 part ∈ PARTITIONS
    @grd02 proc ∈ processes
    @grd03 newstate ∈ PROCESS_STATES
    @grd06 processes_of_partition(proc) = part
    @grd31 partition_mode(part) = PM_NORMAL
    @grd32 process_state(proc) = PS_Running ∧ (newstate = PS_Ready ∨ newstate = PS_Waiting)
  then
    @act11 process_state(proc) ≔ newstate
  end

  event period_wait
  refines process_state_transition
  any part proc newstate
  where
    @grd01 part ∈ PARTITIONS
    @grd02 proc ∈ processes
    @grd03 newstate ∈ PROCESS_STATES
    @grd06 processes_of_partition(proc) = part
    @grd31 partition_mode(part) = PM_NORMAL
    @grd32 process_state(proc) = PS_Running ∧ newstate = PS_Waiting

  then
    @act11 process_state(proc) ≔ newstate
  end
```

```
event process_finished
refines process_state_transition
any part proc newstate
where
  @grd01 part ∈ PARTITIONS
  @grd02 proc ∈ processes
  @grd03 newstate ∈ PROCESS_STATES
  @grd06 processes_of_partition(proc) = part
  @grd31 partition_mode(part) = PM_NORMAL
  @grd32 process_state(proc) = PS_Running ∧ (newstate = PS_Dormant ∨ newstate = PS_Waiting)
then
  @act11 process_state(proc) ≔ newstate
end

event time_out
refines process_state_transition
any part proc newstate
where
  @grd01 part ∈ PARTITIONS
  @grd02 proc ∈ processes
  @grd03 newstate ∈ PROCESS_STATES
```

```
    @grd06 processes_of_partition(proc) = part
    @grd31 partition_mode(part) = PM_NORMAL
    @grd32 process_state(proc) = PS_Waiting ∨ process_state(proc) = PS_Suspend ∨ process_state(proc) =
PS_WaitandSuspend
    @grd33 process_state(proc) = PS_Waiting ∨ process_state(proc) = PS_Suspend ⇒ newstate = PS_Ready
    @grd34 process_state(proc) = PS_WaitandSuspend ⇒ newstate = PS_Suspend
  then
    @act11 process_state(proc) ≔ newstate
  end

  event req_busy_resource
  refines process_state_transition
  any part proc newstate
  where
    @grd01 part ∈ PARTITIONS
    @grd02 proc ∈ processes
    @grd03 newstate ∈ PROCESS_STATES
    @grd06 processes_of_partition(proc) = part
    @grd31 partition_mode(part) = PM_NORMAL
    @grd32 process_state(proc) = PS_Running
    @grd34 newstate = PS_Waiting
```

```
then
  @act11 process_state(proc) ≔ newstate
end

event resource_become_available
refines process_state_transition
any part proc newstate
where
  @grd01 part ∈ PARTITIONS
  @grd02 proc ∈ processes
  @grd03 newstate ∈ PROCESS_STATES
  @grd06 processes_of_partition(proc) = part
  @grd31 partition_mode(part) = PM_NORMAL
  @grd32 process_state(proc) = PS_Waiting ∨ process_state(proc) = PS_WaitandSuspend
  @grd33 process_state(proc) = PS_Waiting ⇒ newstate = PS_Ready
  @grd34 process_state(proc) = PS_WaitandSuspend ⇒ newstate = PS_Suspend
then
  @act11 process_state(proc) ≔ newstate
end

event resource_become_available2
```

```
  refines process_state_transition2
  any part procs newstates
  where
    @grd01 part ∈ PARTITIONS
    @grd02 procs ⊆ processes
    @grd03 newstates ∈ procs → PROCESS_STATES
    @grd06 procs ⊆ processes_of_partition∼[{part}]
    @grd31 partition_mode(part) = PM_NORMAL
    @grd32 ∀proc·(proc∈procs ⇒ process_state(proc) = PS_Waiting ∨ process_state(proc) = PS_WaitandSuspend)
    @grd33 ∀proc·(proc∈procs ∧ process_state(proc) = PS_Waiting ⇒ newstates(proc) = PS_Ready)
    @grd34 ∀proc·(proc∈procs ∧ process_state(proc) = PS_WaitandSuspend ⇒ newstates(proc) = PS_Suspend)
  then
    @act11 process_state ≔ process_state     newstates
  end

  event periodicproc_reach_releasepoint //monitoring the release point of periodic proc, if current time > release point, set from WAITING to READY
  refines process_state_transition
  any part proc newstate
```

```
 where
   @grd01 part ∈ PARTITIONS
   @grd02 proc ∈ processes
   @grd03 newstate ∈ PROCESS_STATES
   @grd04 processes_of_partition(proc) = part
   @grd05 partition_mode(part) = PM_NORMAL
   @grd06 periodtype_of_process(proc) = APERIOD_PROC
   @grd07 process_state(proc) = PS_Waiting
   @grd08 newstate = PS_Ready
 then
   @act01 process_state(proc) ≔ newstate
 end


end
```