MASTER SETI / MASTER CPS / Option SE
Critical Real-Time Systems

Points for grading may vary, **no documents allowed (closed book)**.

The clarity and style of your explanations will be considered during grading.
Only readable, precise and sufficiently explained answers will be considered.

Part 6 is exclusive for SETI and SE301 (except apprentices) students,
Part 3 is exclusive for CPS and SE301 (apprentices only) students.

1. Code Coverage (3 points)

1.1. Code coverage [course] (1 point)

Indicate what should be done by a test suite to guarantee the coverage of code according to the approaches 1) Statement Coverage, 2) Decision Coverage and 3) Modified Condition / Decision Coverage..

1.2. Applying these approaches SC, DC, MC/DC (2 points)

Consider the code below. Illustrate **each** of these approaches by providing possible values of a, b, and c for the conditions. Particular attention will be paid to readability and clarity.

if ((a == 0) || ((b != 1) && (c == 2))) {F_1(a, b, c)}; F_2(a, b, c) ;

2. Integrated Modular Avionic (3 points)

2.1. Scheduling configuration (2 points)

The table below provides a set of tasks with period, criticality level, and budget (CPU utilization). Assuming each task is assigned to a unique ARINC653 partition of its criticality level, determine the **Major Frame**, the **Minor Frame** and the different **Partition Windows**. By assuming the most critical partitions are privileged in the scheduling, calculate a possible timing diagram.

Task	Period	Criticality	Budget
T1	100 ms	A	5.00%
T2	200 ms	A	20.00%
T3	100 ms	B	20.00%
T4	200 ms	B	30.00%
T5	100 ms	C	25.00%

2.2. Communication time (1 point)

It is now assumed that at each of its executions, task T1 sends data d to task T5. It is assumed that the inter-partition communication ports are **flushed** at each minor frame (MIF). We consider that the date of sending d corresponds to the end date of the partition

window in which T1 is executed, and the date of receiving d corresponds to the start date of the partition window in which T5 is executed. By reusing the schedule configuration you established in the previous question, after how long does task T5 receive the last value produced by task T1? Justify your answer and represent this delay on the timetable obtained in the previous question.

3. Mono-processor static priority scheduling (3 points) [CPS, SE apprentices]

In this exercise, we seek to obtain a static priority assignment to ensure the scheduling of the following system of periodic synchronous tasks (with non-implicit deadlines) a mono-core processor:

Task	Budget	Period	Deadline
T1	2	10	10
T2	3	14	7
T3	3	15	10
T4	4	21	21

3.1. Feasibility of a priority assignment [course] (0.5 point)

Given the task model, describe the only method that (**reasonably**) validates a given fixed priority assignment applied to this set of tasks. A detailed answer is expected.

3.2. Approach to find all priority assignments [course] (1 point)

Describe the approach for finding all fixed priority assignments for scheduling a set of periodic synchronous tasks (with non-implicit deadlines).

3.3. Finding all valid priority assignments (1.5 points)

Provide all priority assignments that can schedule this system. A detailed answer is expected.

4. Multi-processors scheduling (4 points)

In this exercise, we seek to schedule the following system of periodic synchronous tasks with implicit deadlines on a two-core processor:

Task	*Budget*	*Period*
T1	2	8
T2	2	10
T3	36	40

4.1. Global-EDF Scheduling [course] (1 point)

Recall how Global-EDF works.

4.2. Applying Global-EDF to a given task set (1 point)

Is this task set schedulable with Global-EDF ?

4.3. Global-EDF applied to a less loaded task set (1 point)

The period of T1 is increased from 8 to 10. Is this system schedulable using Global-EDF?

4.4. Multi-processors scheduling issue [course] (1 point)

What classic issue of multi-processor schedulers do these examples illustrate?

5. WCET Analysis (3 points)

5.1. Static Analysis (1.5 points)

a) Explain the three components that constitutes the "Constant Propagation" analysis covered in the course: the abstract domain, the transfer function, and the meet/join operator. Which mathematical structure is used to represent the domain? Explain the link between this mathematical structure and the meet/join operator.

b) What is a Basic Block? Which role do basic blocks play in control flow graphs (CFG)?

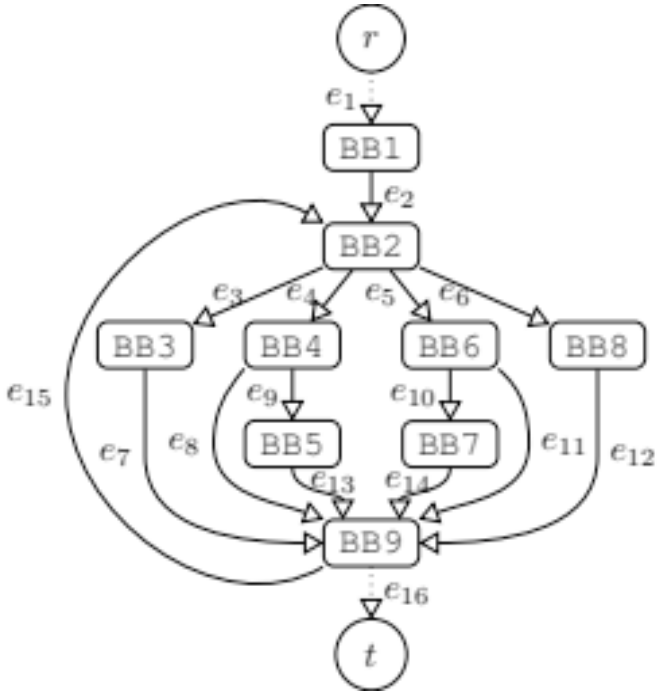
5.2. Implicit Path Enumeration (IPET) (1.5 points)

For the following questions consider the C code and control-flow graph (CFG) shown on the next page. The code implements a state machine that handles the reception of data in bursts (state RECEIVE) and the processing of that data (state PROCESS).

a) The number of times that this state machine executes the basic block **BB6** (state RECEIVE) is executed can be bounded by NUM_BLOCKS and BURST_SIZE. Give a linear equation for the Implicit Path Enumeration Technique (IPET) that allows to bound the number of executions of this block. You may use symbolic names from the code, names of basic blocks, as well as names of edges from the control-flow graph in your formula. Justify your solution in 2-3 sentences.

b) We now wish to add additional equations to bound the number of executions of **BB2**. If this is possible, provide a solution and explain it. Otherwise, explain why it is not possible to such IPET equations.

```
// BB1
b = 0;
state = INIT;
do
{
  // BB2
  switch (state)
  {
    case PROCESS:
      // BB3
      b++;
   state = IDLE;
      break;
    case IDLE:
      // BB4
      if (request)
      {
        // BB5
        state = RECEIVE;
        transferred = 0
      }
      break;
    case RECEIVE:
      // BB6
      transferred++;
      if (transferred == BURST_SIZE)
      {
        // BB7
        state = PROCESS;
      }
      break;
    case INIT:
      // BB8
      state = IDLE;
      break;
  }
  // BB9
} while (b != NUM_BLOCKS);
```



6. WCET and Cache Analysis (3 points) [SETI and SE301 not-apprentices]

6.1. Cache/Memory

The following questions concern the analysis of **instruction caches**, i.e., the classification of memory accesses based on the memory blocks that are accessed during the execution of the basic blocks. Consider a **fully-associative cache** with a size of **4 blocks** and the **least-recently used** replacement policy (LRU). The following figure shows the association between memory blocks and the basic blocks of the program from the previous page:

6.2. Hit/Miss Classification (1 point)

a) Explain the **MUST** analysis and which kind of classification it may provide for accesses to memory blocks. What is the link between the age af a memory block during the actual execution of the program and the information computed by the analysis?

b) Indicate a basic block of the program for which the **MUST** analysis is able to provide a classification – if such a block exists. Justify your answer.

6.3. Persistence (2 points)

a) Hit/Miss classification may sometimes give pessimistic results for certain memory accesses, notably in loops. Explain how **persistence** is able to overcome this pessimism?

b) Verify whether the memory block m_1 accessed during the execution of **BB2** is persistent. Based on your analysis answer to one of the following two questions:

- m_1 is persistent:
  Does this imply that all memory blocks accessed inside the loop are also persistent?
- m_1 is not persistent:
  Does this imply that all the memory blocks accessed in the loop are non-persistent?

7. Energy-aware scheduling (4 points)

In this exercise, we consider the following system of periodic synchronous tasks with implicit deadlines running on a mono-core processor. The priority of tasks is pre-established (see below) and the higher the priority value, the higher the task priority.

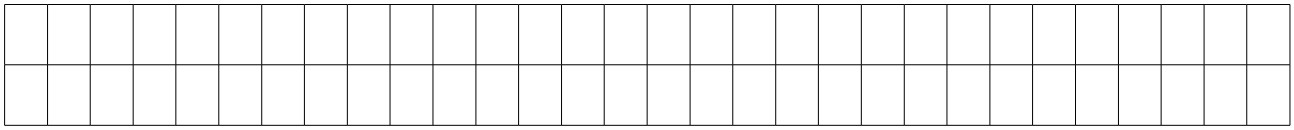
Tasks	Budget (HIE)	Budget (LOE)	Period	Priority
T1	4	5	10	3
T2	3	0	15	2
T3	4	5	15	1
T4	4	5	30	0

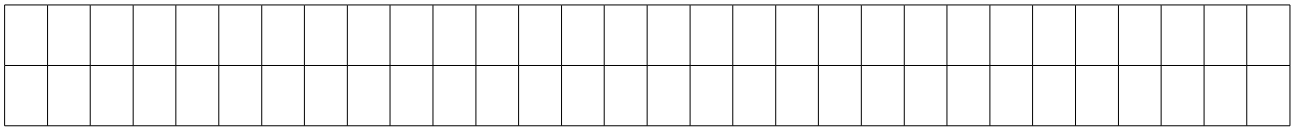
Our system is powered by a photovoltaic panel so that the system has to change behaviour when its energy level is below a given threshold. When the energy level is above the threshold (HIgh-Energy mode), the system runs at normal processor frequency and the scheduler executes 4 tasks with the allocated budgets given in the HIE column. When the energy level reaches the threshold (LOw-Energy mode), the system will reduce the processor frequency by 25%. As a consequence the WCET and the task budget will increase and task T2 will no longer execute to compensate for the loss of computing power. In this case, the scheduler executes only 3 tasks (T2 is no longer running) with the allocated budget given in the LOE column.

To refine the system model, the scheduler will detect the frequency change when a task has exhausted its budget without completing. For instance, when the scheduler detects that task T2 has not completed after 3 units of execution, it will conclude that a frequency change happened, and will shift from scheduling 4 tasks with HIE budgets to scheduling 3 tasks with LOE budgets without restarting the tasks. Only task T2 will stop, while tasks T1, T3 and T4 are allowed to execute for one extra unit (+25%).

7.1. Schedulabity in HIE and LOE modes (case 1) (1 point)

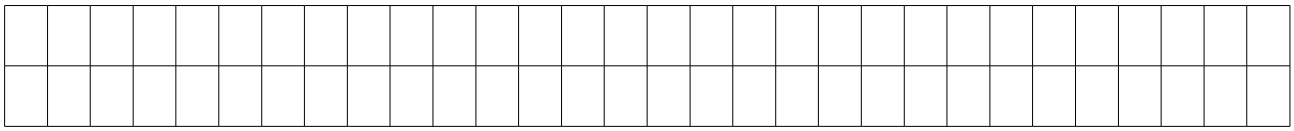
In this question, no frequency change happens. Provide the timeline (chronogram) in HIE mode (which includes T1, T2, T3 and T4 tasks) without changing frequency. Then provide the timeline in LOE mode (which includes only T1, T3 and T4 tasks). In both cases, we assume that all tasks execute for their maximum budget. Is the system schedulable in both modes ? Use the timelines given below if needed.





7.2. Frequency change (case 1) (1 point)

In this question, a frequency change happens while executing task T2. After executing task 2 for 3 units, the scheduler detects that task T2 has not completed. We assume that all tasks execute for their maximum budget. Provide the timeline when the frequency change is detected at the end of task T2 because of the exceeded budget. Is the system schedulable in this case?



7.3. Schedulabity in HIE and LOE modes (case 2) (1 point)

We now assume task T2 is has priority 1 and task T3 has priority 2. In this question, no frequency change happens. In both cases, we assume all tasks execute for their maximum

budget. Provide the timeline in HIE mode (which therefore includes T1, T2, T3 and T4 tasks) without changing frequency. Then provide the timeline in LOE mode (which includes only T1, T3 and T4 tasks). Is the system schedulable in both modes ?

7.4.Frequency change (case 2) (1 point)

In this question, the frequency change happens while executing task T2. After executing task 2 for 3 units, the scheduler detects that task T2 has not completed. We assume that all tasks execute for their maximum budget. Provide the timeline when the frequency change is detected at the end of task T2 because of the exceeded budget. Is the system schedulable in this case?