Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Institut für Technische Informatik
und Kommunikationsnetze
Computer Engineering and Networks Laboratory

Exam Spring 2016

Embedded Systems

Prof. L. Thiele

Note:

The given solution is only a proposal. For correctness, completeness, or understandability, no responsability is taken.

Task 1 : Real-Time Scheduling (maximal 42 points)

1.1: Fixed-Priority Scheduling (maximal 17 points)

A periodic task-set as shown in Table 1 is to be scheduled on a processor. The first release of each task occurs at time zero.

Task	Computation Time	Period	Deadline
τ_1	2	5	4
τ_2	1	4	3
τ_3	2	8	8

Table 1: A periodic task-set

(a) (3 points) Construct the scheduling diagram under Deadline Monotonic (DM) preemptive scheduling.

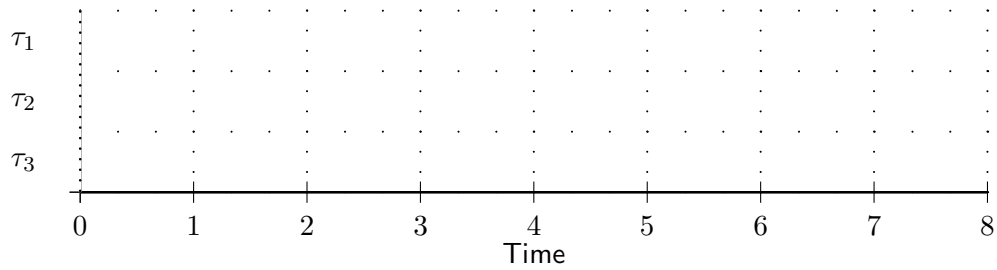


Figure 1: Preemptive DM schedule

(b) (2 points) Construct the scheduling diagram under DM non-preemptive scheduling (i.e. once started, any task has to run to completion before yielding the processor to the other tasks).

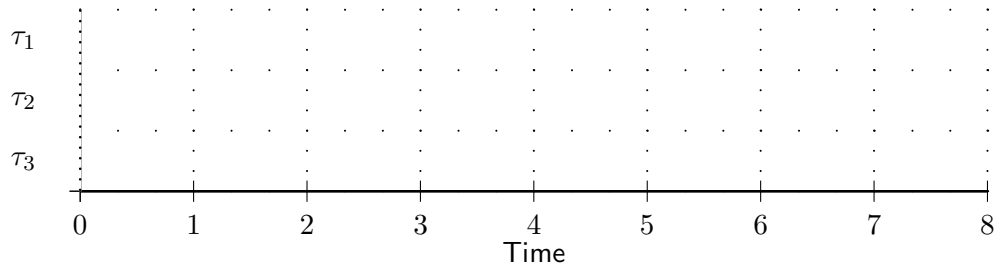


Figure 2: Non-preemptive DM schedule

(c) (9 points) Assume preemptive scheduling and DM priority assignment.

(1) (3 points) Does the task-set pass the sufficient schedulability test?

(2) (6 points) Does the task-set pass the necessary and sufficient schedulability test?

(d) (3 points) Assume non-preemptive fixed-priority scheduling. In the worst-case, how many lower priority jobs can delay the execution of one job? Justify your answer.

1.2: EDF Scheduling (maximal 16 points)

A periodic task-set as shown in Table 2 is to be scheduled on a preemptive processor under Earliest Deadline First (EDF). For a periodic task τ_i, we denote its computation time with C_i, period with T_i, and relative deadline with D_i. We assume that the first release of each task occurs at time zero.

Task	Computation Time	Period	Deadline
τ_1	1	4	3
τ_2	1	2	2
τ_3	2	8	8

Table 2: A periodic task-set

(a) (2 points) Construct the preemptive EDF scheduling diagram.

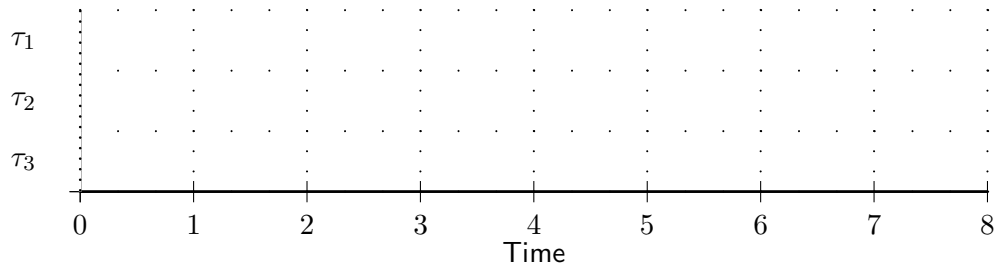


Figure 3: EDF schedule

(b) (2 points) A periodic task-set with constrained deadlines (i.e. $D_i \leq T_i, \forall \tau_i$) is schedulable if $\sum_i \frac{C_i}{D_i} \leq 1$. Does the task-set pass this sufficient schedulability test?

(c) (12 points) The demand bound (db) of a periodic task τ_i for any time interval of length Δ is defined as

$$\text{db}(\tau_i, \Delta) = \max\left\{\left\lfloor \frac{\Delta - D_i}{T_i} \right\rfloor + 1,\ 0\right\} \cdot C_i.$$

It is known that a set of such periodic tasks $\{\tau_i \mid 1 \leq i \leq n\}$ (with arbitrary first release times) is schedulable on a preemptive processor under EDF if and only if

$$\forall \Delta \geq 0, \sum_{i=1}^{n} \text{db}(\tau_i, \Delta) \leq \Delta.$$

(1) (9 points) Plot the total system db as a function of Δ for the task-set in Table 2.

Based on the above demand bound test, is the system schedulable under EDF?

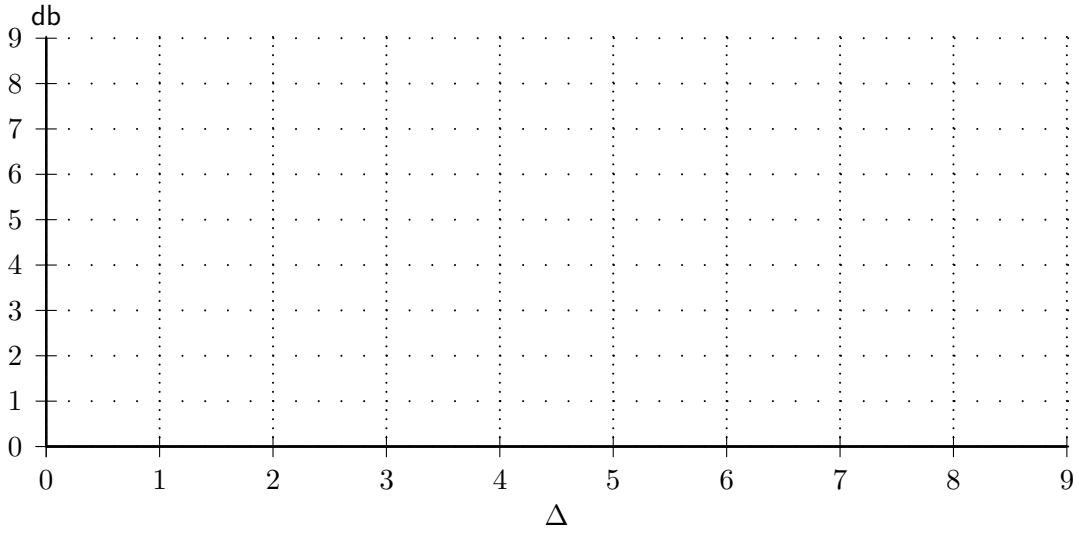


Figure 4: System db

(2) (3 points) Explain in a few sentences why passing the demand bound test is a necessary schedulability condition.
Hint: You can think about what db represents.

1.3: Server Scheduling (maximal 9 points)

A periodic task-set as shown in Table 3 is to be scheduled on a preemptive processor. In addition, a job J with unknown arrival time and computation time of 5 units needs to be scheduled on the processor.

Task	Computation Time	Period	Deadline
τ_1	1	4	4
τ_2	2	5	5

Table 3: A periodic task-set

(a) (3 points) Assume Rate Monotonic (RM) scheduling for the periodic tasks, while job J is served by a polling server with period $T_S = 3$ and computation time $C_S = 0.5$. What is the smallest relative deadline that J can always satisfy?

(b) (6 points) Assume EDF scheduling for the periodic tasks, while job J is served by a total bandwidth server. What is the smallest relative deadline that J can always satisfy such that the periodic tasks are also schedulable?

Task 2 : Communication & Low Power Design (maximal 37 points)

2.1: TDMA & Energy Harvesting (maximal 11 points)

Four sensor nodes (S_1, S_2, S_3, S_4) transmit messages to a host H over a wireless network with the topology depicted in Figure 5.

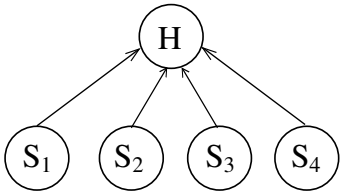


Figure 5: Network Topology

Communication occurs during frames that repeat periodically. Each frame is composed of four slots of equal duration T_{slot}, with one slot for each sensor. Sensors transmit messages during slots according to a Time Division Multiple Access (TDMA) protocol. The radio of the host and each sensor supports a data rate within the range of [1,10] Mbps (10^6 bits per second), and the propagation delay is negligible.

(a) (5 points) Assume each sensor S_i is assigned a slot duration $T_{\text{slot}} = 50$ ms. Each sensor makes periodically available 125 kb (1 kb $= 10^3$ bits) with a period of 250 ms. What is the minimum data rate given that the maximum allowed latency is 200 ms?

Definition: Latency is the time between making data available for transmission at the sensor and receiving the data at the host.

(b) (6 points) Consider the TDMA schedule shown in Figure 6, which includes additional time $T_{\text{processing}}$ for the host to process the received messages. During $T_{\text{processing}}$, the host must compute for 10^5 cycles. The host processor can execute at a clock frequency F from the discrete set $\{5, 10, 20\}$ MHz. Assume that frequency is proportional to voltage ($F \sim V_{\text{dd}}$) and the power P satisfies $P \sim V_{\text{dd}}^2 * F$.

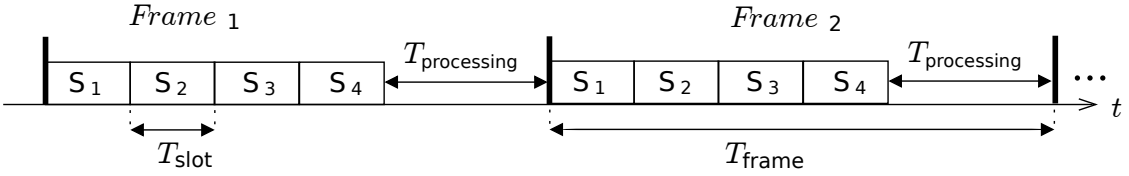


Figure 6: TDMA schedule including host processing time.

If $T_{\text{slot}} = 10$ ms and $T_{\text{frame}} \leq 50$ ms, find the host frequency which minimizes the **energy** required for processing. Justify your answer.

2.2: Dynamic Voltage Scaling (maximal 14 points)

Consider a processor whose dynamic power dissipation when running with a clock frequency f in Hz, is given by $P_{\text{dynamic}} = \left(\frac{f}{10^6 Hz}\right)^3$ mW. It is assumed that the processor has negligible static and leakage power dissipation.

The processor must execute the following task-set:

Task	τ_1	τ_2	τ_3
Arrival Time (ms)	0	2	4
Absolute Deadline (ms)	6	5	9
Cycles (x10^3)	3	9	3

Table 4: Task set.

Assume the clock frequency of the processor can be selected from a continuous range of frequencies. Apply the *online* YDS algorithm and plot the schedule in Figure 7. How much energy does the processor consume to complete all tasks?

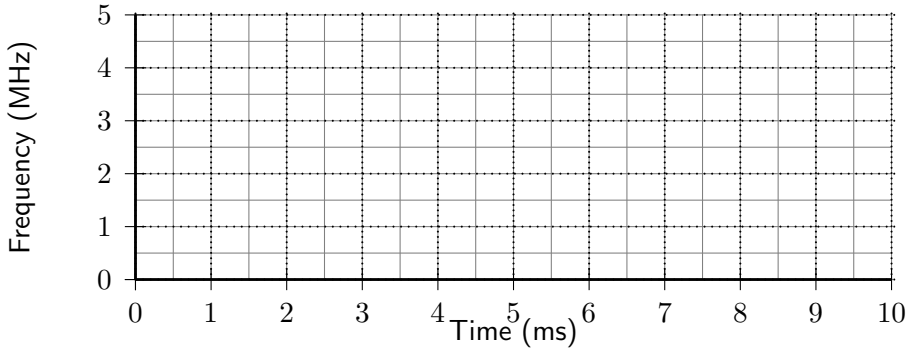


Figure 7: Online YDS schedule.

2.3: Dynamic Power Management (maximal 12 points)

Consider a processor supporting two modes: *IDLE* and *ACTIVE*. The power dissipation is $P_{\text{idle}} = 5$ mW and $P_{\text{active}} = 50$ mW, respectively. The processor must schedule the following periodic task-set:

Task	τ_1	τ_2	τ_3
Period (ms)	10	10	10
Arrival of First Task (ms)	0	3	5
Relative Deadline (ms)	3	7	10
Execution Time (ms)	1	1	2

Table 5: Periodic task-set.

Definition: A workload-conserving scheduler always executes a task when the ready queue is not empty.

Definition: The average power dissipation between time 0 and T is given by $P_{\text{avg}} = \frac{1}{T}\int_0^T P(\tau)d\tau$.

(a) (6 points) Assuming zero energy and time overhead between IDLE and ACTIVE mode, plot the power dissipation during **two periods** for a workload-conserving schedule using Figure 8. The processor is in IDLE mode at time 0. What is the average power P_{avg} between time 0 ms and 20 ms?

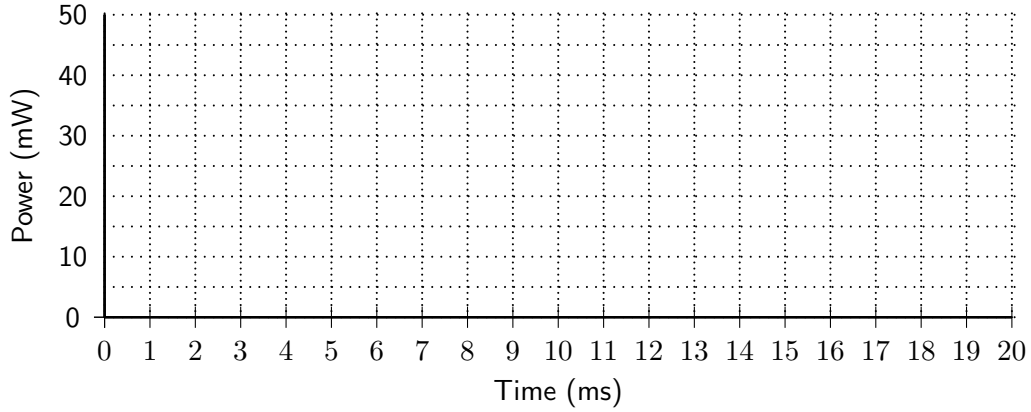


Figure 8: Power dissipation for a workload-conserving scheduler.

(b) (6 points) Now assume that there is a third mode called SLEEP, with $P_{\text{sleep}} = 0$ mW. The system modes and allowed transitions, along with their energy and time overheads, are shown in Figure 9. Power dissipation during a mode transition is **constant**. A transition to ACTIVE mode occurs when there is at least one task in the ready queue. Plot the power dissipation during two periods for an **energy-minimizing**, **workload-conserving** schedule using Figure 10. The processor is in SLEEP mode at time 0. What is the average power P_{avg} between time 0 ms and 20 ms?

Figure 9: System modes and allowed transitions.

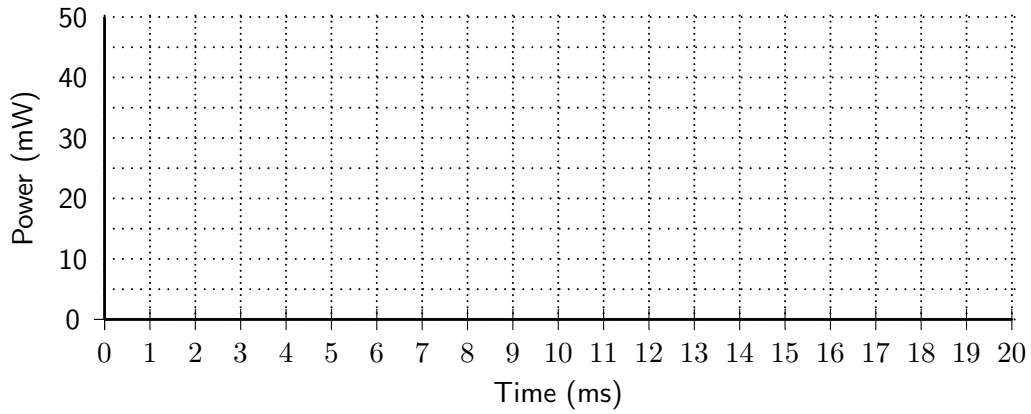


Figure 10: Power dissipation for an energy-minimizing workload-conserving scheduler.

Task 3 : Architecture Synthesis (maximal 41 points)

3.1: Architecture Synthesis Fundamentals (maximal 7 points)

Mark the following statements as *true* or *false* and provide a one sentence explanation.

- (1 point) In a sequence graph, a LOOP node is considered an *operation or task* node.
  ☐ True ☐ False

  Explanation: _____

- (1 point) *ALAP* (as late as possible) scheduling provides a schedule with maximal latency, when *limited* resources are available.
  ☐ True ☐ False

  Explanation: _____

- (1 point) A *dependence graph* does not specify a schedule.
  ☐ True ☐ False

  Explanation: _____

- (1 point) In a *marked graph*, tokens on edges can be interpreted as data stored in a LIFO (last in, first out) queue.
  ☐ True ☐ False

  Explanation: _____

- (1 point) A *marked graph* can be implemented in hardware.
  ☐ True ☐ False

  Explanation: _____

- (1 point) In *List* scheduling, node priorities change as the algorithm executes through different time steps.

  ☐ True ☐ False

  Explanation: _____

  _____

- (1 point) For a given problem, a *heuristic* algorithm guarantees optimal solutions.

  ☐ True ☐ False

  Explanation: _____

  _____

3.2: List Scheduling (maximal 13 points)

Consider the sequence graph in Figure 11, and answer the following questions.

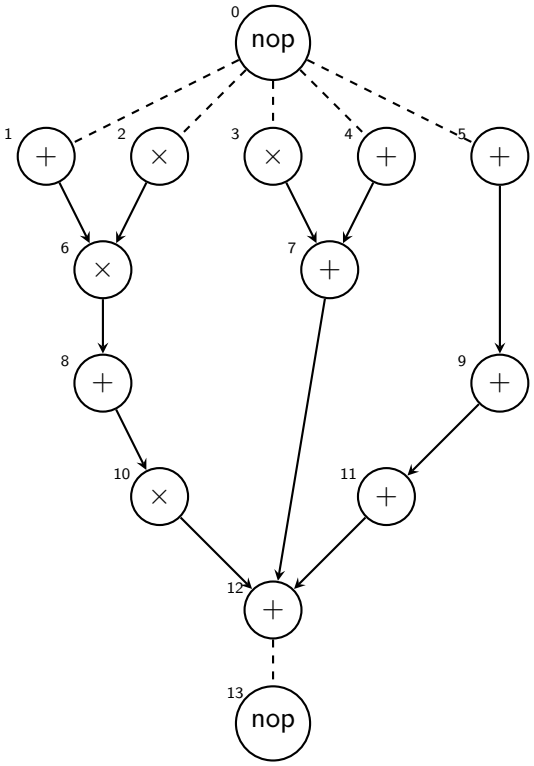


Figure 11: A sequence graph

(a) (10 Points) Suppose that one multiplier and two adders are available as resources. Addition and multiplication take **one** and **two** time units on the adders (r_1, r_2) and multiplier (r_3), respectively. The first operation starts at $t = 0$ and the top node ('nop') is executed in zero time. The priority is assigned for each operation as the *maximal distance* to the bottom node ('nop'). The maximal distance is defined as the number of edges on the longest path between two nodes. Fill out Table 6 using the **List scheduling algorithm**. For a timestep t: $U_{t,k}$ denotes the set of operations that are ready to be scheduled on resource r_k. $S_{t,k}$ denotes the set of operations that start at time t on resource r_k. $T_{t,k}$ denotes the set of operations in execution at time t on resource r_k.

t	k	$U_{t,k}$	$T_{t,k}$	$S_{t,k}$
0	r_1, r_2			
	r_3			
1	r_1, r_2			
	r_3			
2	r_1, r_2			
	r_3			
3	r_1, r_2			
	r_3			
4	r_1, r_2			
	r_3			
5	r_1, r_2			
	r_3			
6	r_1, r_2			
	r_3			
7	r_1, r_2			
	r_3			
8	r_1, r_2			
	r_3			
9	r_1, r_2			
	r_3			
10	r_1, r_2			
	r_3			
11	r_1, r_2			
	r_3			
12	r_1, r_2			
	r_3			
13	r_1, r_2			
	r_3			
14	r_1, r_2			
	r_3			

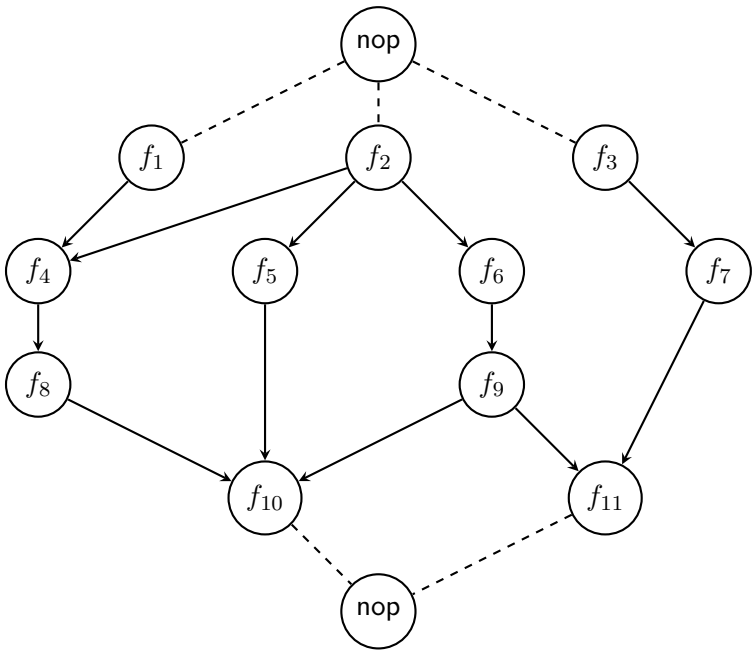
Table 6: Table for Task 3.2. (a)

(b) (1 Point) What is the corresponding latency?

(c) (2 Points) If it is allowed to add a single additional hardware unit, an adder or a multiplier, which resource should be added to reduce the latency? Explain why.

3.3: Timing Constraints (maximal 12 points)

Consider the sequence graph and execution times $w(f_i)$ of tasks f_i in Figure 12, answer the following questions.



Task	$w(f_i)$
f_1	1
f_2	5
f_3	1
f_4	2
f_5	2
f_6	2
f_7	4
f_8	3
f_9	2
f_{10}	3
f_{11}	2

Figure 12: A sequence graph and execution times

(a) (2 Points) Complete the Weighted Constraints Graph $G_C = (V_C, E_C, d)$ by annotating edges with numbers in Figure 12.

(b) (5 Points) Formulate the following constraints using $\tau(f_i)$ as the start time of task f_i. Add appropriate annotated edges to the constraint graph G_C in Figure 12.

i) f_2 should start no earlier than 5 time units after the start of f_3.

ii) f_4 should finish no earlier than 3 time units after the start of f_5.

iii) f_9 should start no later than 8 time units after the finish of f_5.

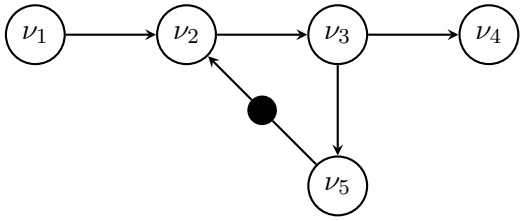
iv) f_{10} should start exactly 3 time units after the start of f_{11}.

v) f_{10} should finish no later than 14 time units after the finish of f_3.

(c) (5 Points) In case of unlimited resources, does a feasible schedule exist for the given sequence graph and all of the above timing constraints? Justify your answer.

3.4: Iterative Algorithm (maximal 9 points)

Consider the marked graph G_M and the task execution times in Figure 13, and answer the following questions.



Task	ν_1	ν_2	ν_3	ν_4	ν_5
$w(\nu_i)$	1	2	1	1	2

Figure 13: A marked graph with task execution times

(a) (5 points) Formulate all existing dependencies in Figure 13 from ν_i to ν_j in the form of

$$\tau(\nu_j) - \tau(\nu_i) \geq w(\nu_i) - d_{ij} \times P,$$

where P is the minimum iteration interval.

(b) (4 Points) Function ν_2 uses the result of ν_5 from the previous iteration. Suppose that any arbitrary number of tokens can be inserted on this edge to reduce P using functional pipelining. Assuming unlimited resources, and that the same operation of different iterations cannot be executed in parallel or overlap, how many tokens should be added on the edge $\nu_5 \rightarrow \nu_2$ to achieve the minimal iteration interval? To justify your answer, draw the pipelined schedule for **four** iterations in Figure 14. Determine the latency L of this schedule.

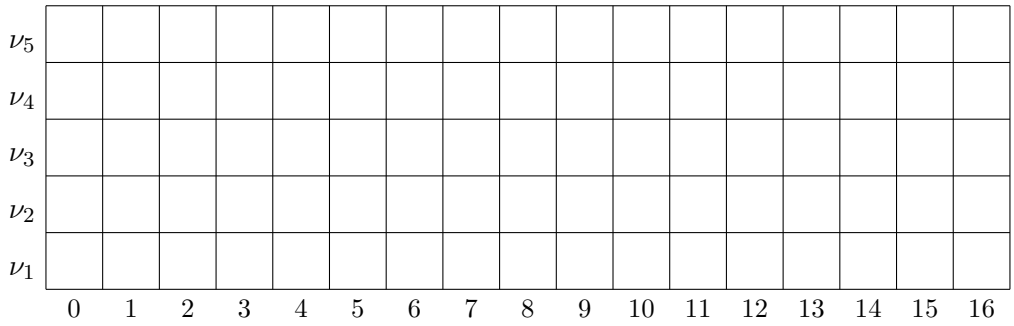


Figure 14: Scheduling grid for Task 3.4 (b)