Networked Embedded Systems Lab
Prof. Marco Zimmerling

Introduction to Embedded Systems – WS 2022/23

Test Exam

1 Real-Time Scheduling (39 Points)

1.1 Short Questions (8 Points)

Answer the following questions for the MSP-432 processor:

(a) (3 points) A system uses UART to transmit data, with the following configuration: the baud rate is 115 200 bits/s, 1 start bit, 2 stop bits, 7 data bits and 1 parity bit. How much time is needed to transmit 240 KB of data (1 KB = 1024 bytes)?

(b) (2 points) How many bytes can be written in a block of memory accessed using byte-addresses `0x3000_0000` through `0x308F_FFFF`?

(c) (3 points) The following function returns the value that has been read from the appropriate GPIO port (pin 7 being the MSB).

```
uint8_t GPIO_getInputPortValue(uint_fast8_t selectedPort);
```

Pins 0 through 7 of GPIO port PORT1 are equipped with buttons with pull-up resistors (when the button is not pressed, the GPIO is connected to the supply voltage; when the button is pressed, the GPIO is connected to the ground). If only buttons connected to pins 2 and 7 of PORT1 are pressed, what value will variable `uint8_t kk` have after the following line of code is executed?

```
uint8_t kk = GPIO_getInputPortValue(PORT1) & 0x3C;
```

Hint: `&` is the logical AND operator.

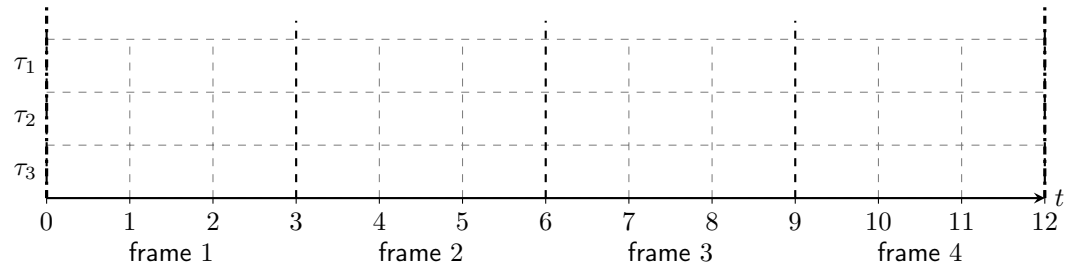
1.2 Cyclic-Executive Scheduling (8 Points)

Cyclic-executive scheduling with a period $P = 12$ and a frame length $f = 3$ is used to schedule the task-set given in Table 1. Note that "frame 1" is the first frame of each period.

Table 1: A task set

Task	Period	Deadline	Phase	Execution Time	Frames
τ_1		4		1	2, 4
τ_2	12	10	2	2	
τ_3	6	5	1	1.5	

(a) (4 points) Determine one feasible assignment of tasks τ_2 and τ_3 to frames, construct the schedule for one period P and illustrate it graphically.



(b) (4 points) Determine the period of task τ_1 and its minimal possible phase, if the task is executed in frames 2 and 4.

1.3 Rate Monotonic Scheduling (11 Points)

A periodic task set is given in Table 5. Assume all phases are zero, and deadlines equal periods.

Table 2: A task set

	τ_{P1}	τ_{P2}	τ_{P3}	τ_{P4}
Period	5	7	11	13
Execution Time	1	2	3	3

(a) (2 points) Test if the given task set is schedulable under rate monotonic (RM) scheduling, using the *sufficient* test (the utilization bound test).

Table 3: The task set, repeated

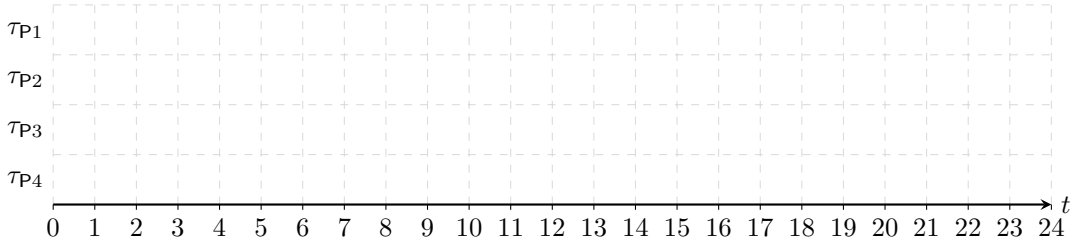
	τ_{P1}	τ_{P2}	τ_{P3}	τ_{P4}
Period	5	7	11	13
Execution Time	1	2	3	3

(b) (5 points) Test whether the given task set is schedulable under rate monotonic (RM) scheduling, using the *necessary and sufficient* test.

Table 4: The task set, repeated

	τ_{P1}	τ_{P2}	τ_{P3}	τ_{P4}
Period	5	7	11	13
Execution Time	1	2	3	3

(c) (4 points) Using *earliest deadline first* (EDF) scheduling, construct a schedule from time 0 to time 24, and illustrate it graphically. Note if any task misses its deadline.



1.4 Scheduling Mixed Tasks (12 Points)

We have a task set with periodic and aperiodic tasks. The periodic tasks are τ_{P1}, τ_{P2} and τ_{P3}; the aperiodic tasks are τ_{A1}, τ_{A2}, and τ_{A3}. Additionally, task τ_{PS} is a polling server meant to service the aperiodic tasks. All tasks are specified in Table 5.

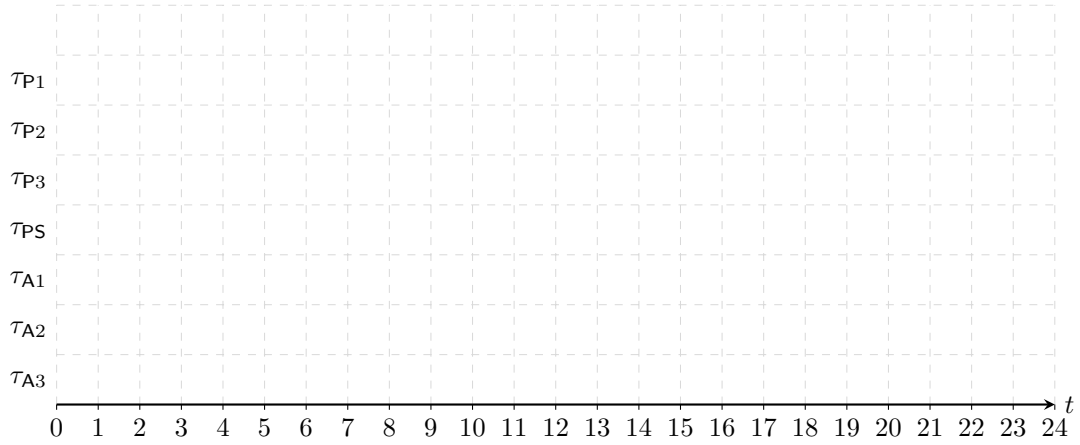
Deadline Monotonic scheduling is used to schedule the periodic tasks and the polling server, while the polling server services aperiodic tasks on a first-come-first-serve basis.

Assume task τ_{A1} arrives at time 1, task τ_{A2} at time 0 and task τ_{A3} at time 14.

Table 5: Mixed task set

Task	Period	Phase	Deadline	Execution Time
τ_{P1}	5	0	5	2
τ_{P2}	9	7	2	1
τ_{P3}	10	1	7	1
τ_{PS}	8	2	6	3
τ_{A1}			5	1
τ_{A2}			1	1
τ_{A3}			9	3

Illustrate the schedule of all of the tasks graphically, including the polling server, from time 0 to time 24. Note if any task misses its deadline.



2 Low Power Design (40 Points)

2.1 Dynamic Voltage Scaling (10 Points)

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

The processor must execute the following task set:

Task	τ_1	τ_2	τ_3
Arrival Time (ms)	0	3	5
Absolute Deadline (ms)	8	6	10
Cycles (x10^3)	8	12	2

Table 6: Task set.

Apply the *offline YDS* algorithm and plot the schedule of tasks and frequencies in Figure 1. Provide the steps of your solution in detail.

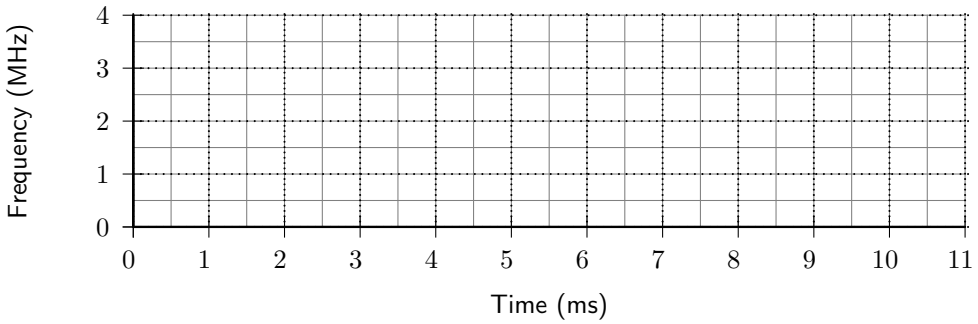


Figure 1: Offline YDS Schedule.

2.2 Dynamic Power Management (14 Points)

Consider an embedded system with a processor that can execute in one of three modes, namely *HIGH*, *LOW*, and *SLEEP* mode. The valid transitions between the three execution modes, the power dissipation, and the processor frequency within each mode are summarized in Figure 2. The transition cost in terms of both time and energy are denoted by t_1, t_2 and E_1, E_2, respectively.

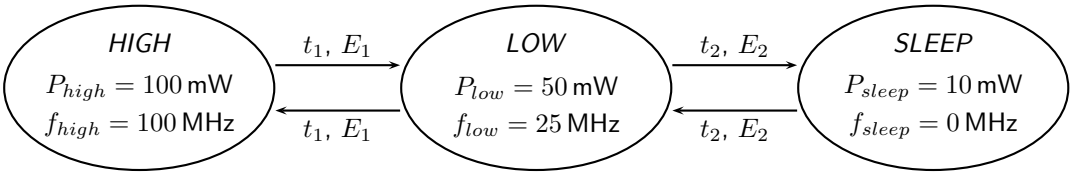


Figure 2: Processor execution modes.

The processor must schedule the following periodic real-time tasks:

Task	τ_1	τ_2	τ_3
Period (ms)	10	10	10
Arrival of First Task (ms)	0	4	7
Relative Deadline (ms)	3	8	10
Cycles ($\times 10^5$)	1	2	2

Table 7: Periodic task set.

(a) (4 points) Assume zero transition energy (i.e. $E_1 = E_2 = 0$) and instantaneous mode transitions (i.e. $t_1 = t_2 = 0$). Plot in Figure 3 a schedule that includes the power consumption over time, as well as tasks that are executing, using a workload-conserving scheduler that minimizes the average power. A workload-conserving scheduler always executes when a ready task is available.

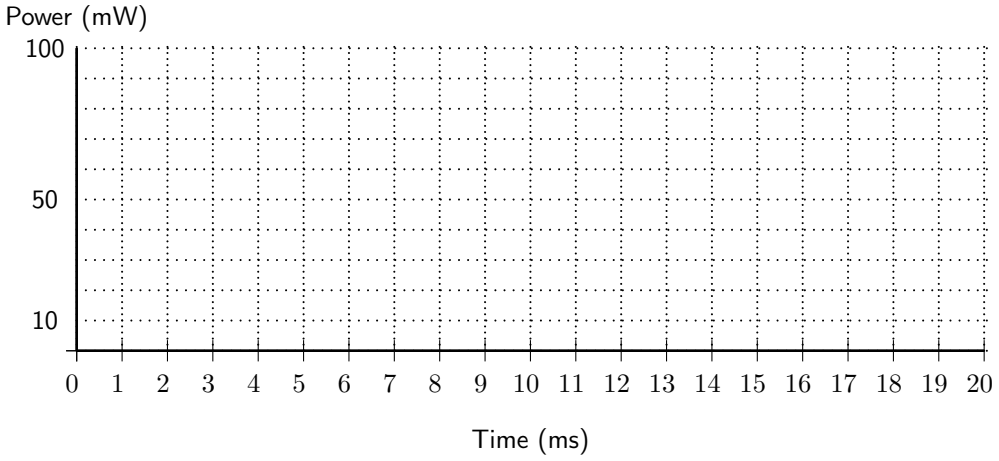


Figure 3: Energy efficient schedule using a workload-conserving scheduler.

For the following tasks assume the time and energy overhead between the execution modes as defined in Figure 2 to be $t_1 = 0.1\,\text{ms}$, $t_2 = 0.9\,\text{ms}$, and $E_1 = 10\,\mu\text{J}$, $E_2 = 40\,\mu\text{J}$, respectively.

(b) (5 points) Calculate the break-even time for the *HIGH* to *SLEEP* mode transitions.

Definition of break-even time: The minimum idle time required to compensate the cost of entering an inactive (sleep) state.

(c) (5 points) Find a schedule (not necessarily workload conserving) that satisfies all task deadlines and minimizes the energy consumption. Plot the schedule in Figure 4 that includes the power consumption over time, as well as tasks that are executing. Assume constant power consumption during the transition between different execution modes.

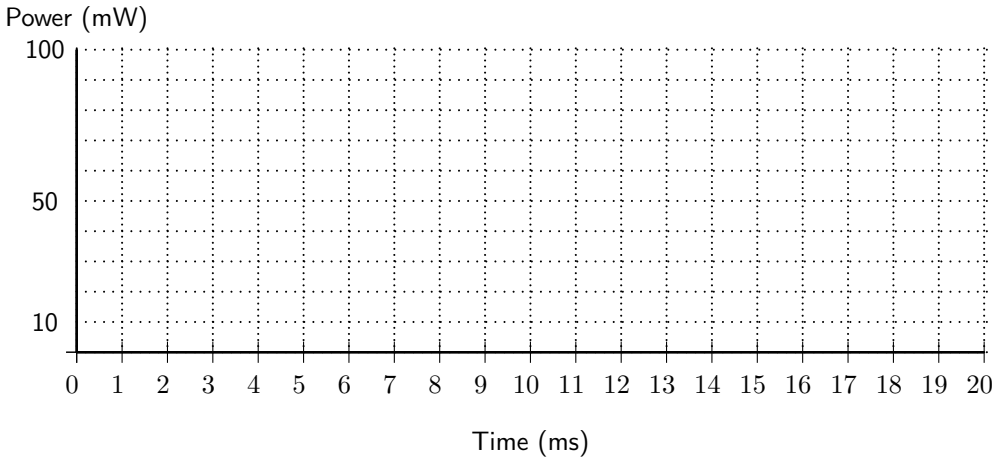


Figure 4: Energy minimizing schedule satisfying all task deadlines.

2.3 Solar Cells and Power Point Tracking (5 Points)

We are given a solar cell with characteristics

$$I = G - 10^{-4} \times (\frac{U}{0.05})^3, \tag{1}$$

where I is the normalized current, U is the normalized voltage, and G is the relative solar irradiance.

Execute by hand the power point tracking algorithm as presented in the lecture for $G = 0.8$. Determine the voltage $U(k)$ and power $P(k)$ for $k = 0, \ldots, 5$ with $U(0) = 0.5$ and $U(1) = 0.55$, using the stepsize 0.05.

2.4 Application Control (11 Points)

We consider an application control scenario with the harvested energy in time interval $[t, t+1]$ of $p(t)$, used energy of $u(t)$ and battery capacity B. As the utility function we use $\mu(u) = \sqrt{u}$. The units of p, u, and B are Wh, the unit of t is h.

(a) (3 points) Suppose the energy is harvested by a solar panel, which generates $a(t) = 0.05\,\mathrm{Wh/cm^2}$ per hour in the daylight, which lasts 8 hours every day. During the rest of the day, the solar panel generates $a(t) = 0\,\mathrm{Wh/cm^2}$. For a solar panel with a size of S, the harvested energy function is $p(t) = a(t) \cdot S$. The system is equipped with a solar panel of size $S = 200\,\mathrm{cm^2}$. Is it possible for the system to continuously dissipate a constant power of $3\,\mathrm{W}$ for infinite days? If yes, what is the minimal battery size B in Wh to sustain this operation?

(b) (8 points) Suppose now that $B = 33\,\mathrm{Wh}$. The harvested energy function $p(t)$ is given in Figure 5. Determine an optimal energy usage function $u^*(t)$ that maximizes the utility and never leads to failure state. Draw $u^*(t)$ into Fig. 5.

Hint: All values of $u^*(t)$ are integers.

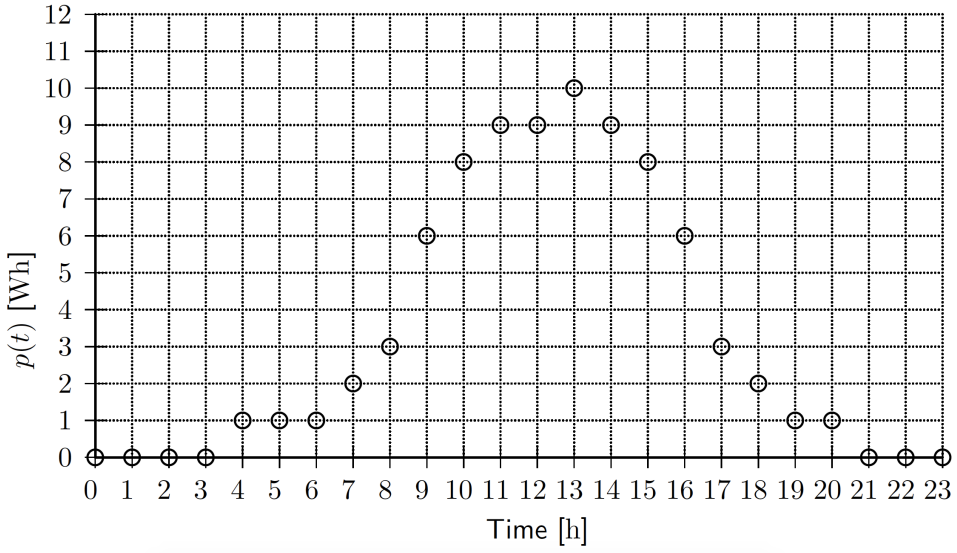


Figure 5: Application Control, Task (b)

3 Models and Architecture Synthesis (41 Points)

3.1 Architecture Synthesis Fundamentals (9 Points)

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

- (1 point) The "Stack Policy" as introduced in the lecture could be used in combination with EDF task scheduling

  ☐ True ☐ False

  Explanation: _____

  _____

- (1 point) A dependence graph as defined in the lecture can represent parallelism in a program and can represent branches in control flow.

  ☐ True ☐ False

  Explanation: _____

  _____

- (1 point) The throughput of an implementation of an iterative algorithm can be increased by decreasing the iteration interval.

  ☐ True ☐ False

  Explanation: _____

  _____

- (1 point) A marked graph is designed to be implemented in software only.

  ☐ True ☐ False

  Explanation: _____

  _____

- (1 point) In a marked graph, a node with 2 input edges requires at least 2 tokens on any of the input edges to be activated.

  ☐ True ☐ False

  Explanation: _____

  _____

- (2 points) Given two operations, v_1 and v_2, with execution time $w(v_1) = 2$ and $w(v_2) = 2$, respectively. "$\tau(v_2) - \tau(v_1) \leq 4$" models the following constraint : "v_2 must start not later than 2 time units after the end of v_1".

  ☐ True ☐ False

  Explanation: _____

  _____

- (1 point) Energy consumption can be improved by using pipelining instead of sequential processing of a given task set.

☐ True ☐ False

Explanation: _____

_____

- (1 point) List scheduling is an optimal algorithm for task scheduling with resource constraints.

  ☐ True ☐ False

  Explanation: _____

  _____

3.2 LIST Scheduling (13 Points)

Given the sequence graph in Figure 6.

(a) (8 points) Suppose that **one** multiplier and **one** adder are available as resources. Addition take one time unit for execution with the adder (r_1). Multiplication take two time units for execution with the multiplier (r_2). The first operation starts at $t = 0$ and the top node ('nop') is executed within zero time unit. The priority is assigned for each operation as maximal distance to the bottom node ('nop'). In case of equal priorities choose the node with the lowest index number. Fill out Table 8 using LIST scheduling algorithm. For a timestep t, the value $U_{t,k}$ denotes the set of operations that are ready to be scheduled on resource r_k (to be more specific, the set of operations that can be mapped on resource r_k and whose predecessors are all completed). $S_{t,k}$ denotes the set of operations that start at time t on resource r_k, while $T_{t,k}$ is the set of operations in execution at time t on resource r_k.

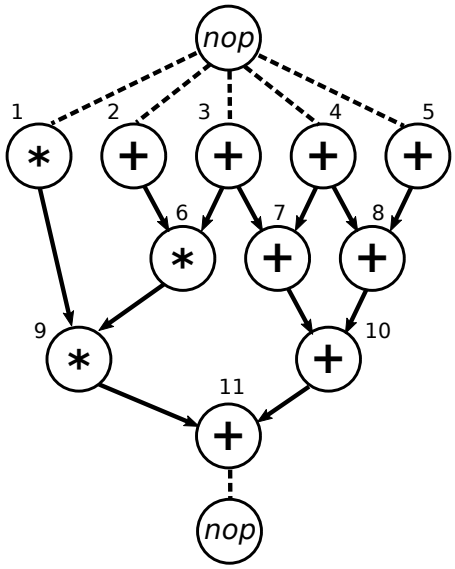


Figure 6: A sequence graph

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

Table 8: Table for subtask (a)

(b) (1 point) What is the resulting latency?

(c) (2 points) Suppose it is allowed to add either one adder or one multiplier, which resource should be added to shorten the latency? What is the corresponding latency?

(d) (2 points) In case of unlimited hardware resource, what is the minimized latency? Explain why.

3.3 Iterative Algorithms (19 Points)

Consider the marked graph G_M in Figure 7. The nodes labeled with +, **x2**, **+4** represent addition, multiplication by 2, and addition with 4, respectively. f_1, f_3 and f_4 need **1** time unit each, f_2 needs **2** time units. The input u is a sequence of numbers, with $u(k)$ representing the k-th number.

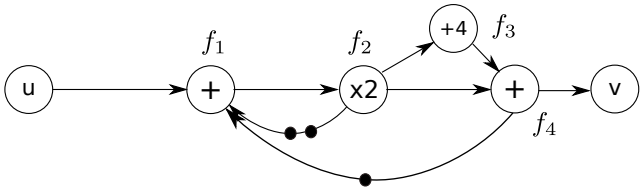


Figure 7: Marked graph G_M

(a) (7 points) Determine the output value $v(k)$ corresponding to the graph in Figure 7 as a function of the input values $u(\cdot)$ and previous output values $v(\cdot)$. Assume $k > 2$.

(b) (4 points) Suppose that we implement the algorithm using functional pipelining. Express all constraints which stem from the data dependencies in G_S, considering also the data dependencies among successive iterations.

Hint: Determine constraints of the form $\tau(f_j) - \tau(f_i) \geq w(f_i) - d_{ij} \cdot P$, $\forall (f_i, f_j) \in E_S$, where P is the iteration interval of the pipelined implementation, $w(f_i)$ denotes the execution time of f_i, and d_{ij} represents the index displacement associated with edge (f_i, f_j).

(c) (3 points) Assuming unlimited resources, determine the smallest feasible iteration interval P_{min}. Justify your solution.

(d) (3 points) Now assume that, there are only **one** multiplier to compute f2 and **one** adder to compute f1,f3 or f4 available. First, depict a pipelined scheduling in Figure 8 under this resource constraint with a predefined minimal iteration interval $P = 5$. Then, indicate the latency L of the schedule. Draw **three** consecutive iterations, and mark different iterations clearly, e.g. with different colors, different textures, or different numbers.

$L =$

(e) (2 points) Can the latency of a schedule given the marked graph be decreased by using an unlimited number of adders and multipliers? Explain why?

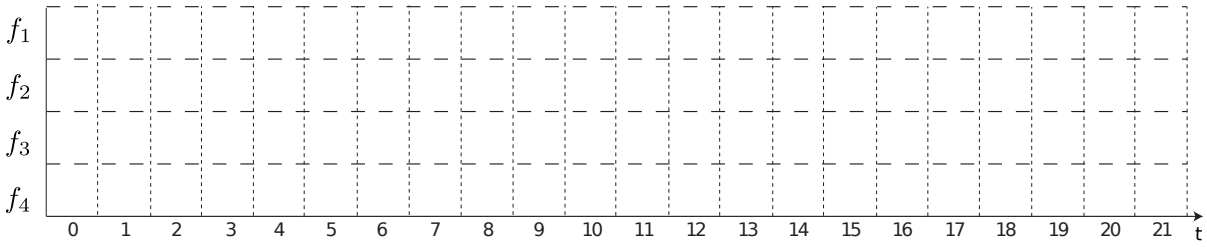


Figure 8: Pipelined Scheduling