



A10.9 Project: a tiny robot operating system

Operating systems, software and translators · A level · OCR H446 1.2.1,
AQA 7517 4.6.1.4, Eduqas A500QS 2.6 · about 50 min

Name _____

Class _____

Date _____

What this lesson is about

Share the robot between three tasks with round robin, drive it through device drivers, and stop the patrol with an interrupt.

- The brief
- Decompose it
- Step 1: the drivers
- Step 2: plan the trace before you code
- Step 3: the scheduler without the interrupt
- Step 4: add the interrupt
- Test plan

Questions

5 marks in all

- In the tiny robot OS, what does the check after every step model? [1 mark]
 - ☐ A The check for interrupts at the end of each fetch-decode-execute cycle
 - ☐ B Paging
 - ☐ C Lexical analysis
 - ☐ D The BIOS power-on self-test
- Why does the project run every step through a driver table instead of calling forward, led and tone directly? [1 mark]
 - ☐ A The scheduler does not need to know how any device works, and a new device only needs a new driver
 - ☐ B Driver tables make the robot drive faster
 - ☐ C Python cannot call robot commands from inside a loop
 - ☐ D It stops the interrupt from firing
- What does this program print? [1 mark]

```
queue = ['patrol', 'lights', 'music']
left = {'patrol': 3, 'lights': 1, 'music': 2}
order = []
while queue:
    name = queue.pop(0)
    left[name] = left[name] - 1
    order.append(name[0])
    if left[name] > 0:
        queue.append(name)
print(''.join(order))
```

4. BugBot's motor loop must correct the robot's heading within a few milliseconds every time. Which two [1 mark]
types of operating system describe FreeRTOS on the robot?
- ☐ A Embedded and real-time
 - ☐ B Distributed and multi-user
 - ☐ C Multi-user and real-time
 - ☐ D Distributed and embedded
5. In the project, the patrol task is stopped by the interrupt with one step still to run. Which scheduling [1 mark]
idea does this show?
- ☐ A Pre-emption: a running task loses the processor before it has finished its slice
 - ☐ B First come first served
 - ☐ C Shortest job first
 - ☐ D Disk thrashing

The task: a tiny robot OS

Build the operating system from the brief. The inputs are: - `tasks`, a list of `(name, steps)` tuples in their starting queue order, where `steps` is a list of `(command, value)` pairs; `command` is "forward" (value: cm, a positive integer), "led" (value: a colour name) or "tone" (value: hertz, 100 to 10000); - `TIME_SLICE`, the most steps a task runs before the next task starts (2); - `SAFE_CM`, the interrupt threshold: the obstacle interrupt fires when `distance()` is less than this (25). Write a `drivers` dictionary with a driver for each command: "forward" drives forward at speed 50, "led" sets the LED colour, and "tone" plays the note for 0.2 seconds. Run every step through `drivers`. Print exactly these lines, built from your variables: `<name>: <command> <value>` after each step, `interrupt: obstacle` and `patrol stopped` from the handler (with the 220 Hz, 0.2 second tone between them), and `<name> finished` when a task runs its last step. The wall is 51 cm ahead of the robot.

```
# the two lines every program starts with: the commands, then the robot
from bugbot import *
connect()

tasks = [
    ("patrol", [("forward", 10), ("forward", 10), ("forward", 10), ("forward", 10)]),
    ("lights", [("led", "blue"), ("led", "yellow"), ("led", "green")]),
    ("music", [("tone", 523), ("tone", 659), ("tone", 784)]),
]
TIME_SLICE = 2
SAFE_CM = 25
```

Plan your program here, then type it in and press Run.



Do it on the robot

www.bugbotlab.com/learn/a10-9-project-a-tiny-robot-os/

The simulator checks it and tells you when it passes. Nothing to install, no account.

Challenges

1. Change `TIME_SLICE` to 1 and then to 4. Predict the order of the lines each time, then check. Does the slice change how close the patrol gets to the wall? Explain why.
2. Give each task a priority and replace round robin with a priority scheduler. Can a low-priority task starve?
3. Add a second interrupt, `low battery`, with a higher priority than the obstacle. What should happen if both are raised after the same step?