The University Of Sheffield.

Data Provided: None

DEPARTMENT OF COMPUTER SCIENCE **January 2021**

TESTING AND VERIFICATION IN SAFETY CRITICAL SYSTEMS 2 hours

Answer ALL THREE questions.

All questions carry equal weight. Figures in square brackets indicate the percentage of available marks allocated to each part of a question.

THIS PAGE IS BLANK

1. Consider the following requirement for software to control fuel and oxygen mixing in a rocket engine:

The software will control the fuel and oxygen flow rates using valves that accept digital control. It will accept a single integer input in the range 0-100 that is the percentage of maximum fuel flow to provide, and it is to output integer values in the range 0-100 indicating valve openness for each liquid. The valves have the same flow characteristics, but the controller should ensure that there is always a 60/40 mix of fuel/oxygen.

The engine is expected to run for 8.5 minutes per launch, with a target of 20 launches per year and a service life of 25 years. The software controller will be in continuous use while the engine is running, and there will be 8 engines on the vehicle. Supplying the liquids outside of the ratio requirement above can be assumed to catastrophically destroy the vehicle. Due to the high speeds and stresses of the flight profile of the vehicle it can also be assumed that delays in responding to inputs, or providing too high or low a fluid flow in response to the input will also be catastrophic for the vehicle.

a) Draw a simple block diagram of the system and its inputs and outputs. [20%]

b) For each of the inputs and outputs of the system, list the ranges of possible values, and any expected relationships between values. [20%]

c) Produce a simple HAZOP table, applying each of the keywords to the inputs and outputs of the system. Use the information presented in the requirement above, and make sensible assumptions about any other aspects of the system context. [40%]

d) What SIL level would you assign to this software system? Justify your answer. [20%]

2. Consider this Z specification for the controller's operation (recall that $\mathbb{Z}$ is the symbol for the Integer type in Z):

$$
\begin{array}{|l}
\textit{ValveControl} \\
\hline
power? : \mathbb{Z} \\
fuel! : \mathbb{Z} \\
oxygen! : \mathbb{Z} \\
\hline
0 \leq power? \leq 100 \\
fuel! = power? \\
oxygen! = \frac{power?}{60} \times 40 \\
\hline
\end{array}
$$

a) Describe how the variables $power?$, $fuel!$, and $oxygen!$ relate to the requirements detailed in Question 1. Does this operation schema meet the requirements? Specifically, describe how it ensures the correct fuel/oxygen ratio. [20%]

b) What is the possible range of values that can be produced on the $oxygen!$ output? Explain your answer [20%]

Consider this psudocode algorithm:

$$
\begin{array}{lll}
fuel & \leftarrow & power; \\
x & \leftarrow & power/60; \\
x & \leftarrow & x * 40; \\
oxygen & \leftarrow & x;
\end{array}
$$

c) Annotate these lines with pre- and post-conditions to form Hoare triples. Ensure that the post-condition of a statement before, and the pre-condition of a statement after a sequential composition are the same. [30%]

d) Do your Hoare triples convince you that the algorithm achieves the operation in the specification? Explain your answer. [30%]

3. Consider the following C code to implement the fuel flow controller:

```
1 void ValveControl(int power) {
2   int x;
3   if(power < 0 || power > 100) {
4     power = 0;
5   }
6   setFuel(power);
7   x = power / 60;
8   x = x * 40;
9   setOxygen(x);
10 }
```

Consider these tests of the code:

```
Power: 0, Fuel: 0, Oxygen: 0
Power: 100, Fuel: 100, Oxygen: 40
Power: 50, Fuel: 50, Oxygen: 0
Power: 60, Fuel: 60, Oxygen: 40
```

a) The test above do not evaluate the results - they do not say whether they have passed or failed. The code does not properly deliver the requirements. Explain why not with reference to these test results and the requirements and specifications in the previous questions, and also identify what is wrong with the code and how to fix it. [30%]

b) The tests do not achieve 100% branch coverage. Identify where, and suggest some additional tests to achieve 100% branch and statement coverage [20%]

c) Is it possible to add tests and increase the MC/DC coverage of this test set? Explain with examples. [20%]

d) Propose some simple mutations to the code that would not be killed by the test set above, that that would be killed by your additions. [30%]

END OF QUESTION PAPER