Heriot-Watt University
School of Mathematical and Computer Sciences

Distributed Systems Programming F29NM1

Possible Solutions to SPIN Exercise Sheet 2

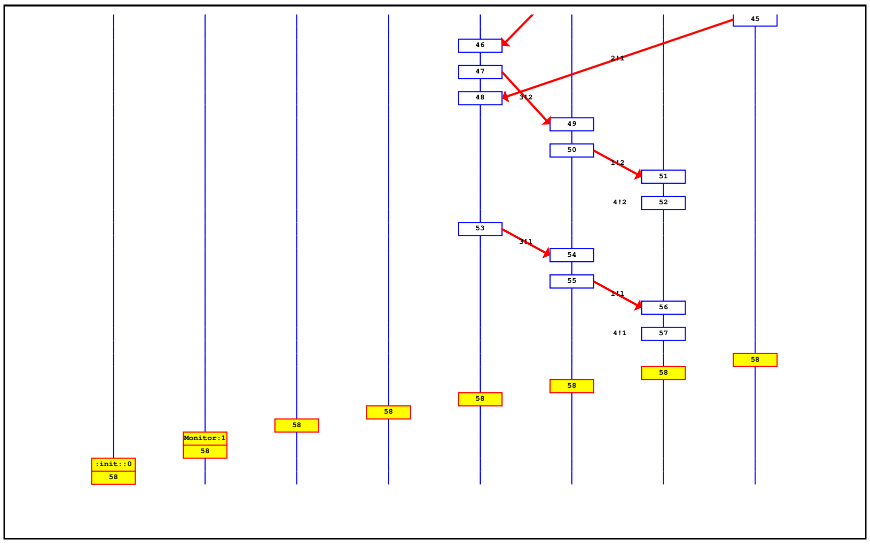
Andrew Ireland

Exercise 1

```
byte value1 = 1, value2 = 2, value3 = 3;

proctype A() { value3 = value3 + value2;}

proctype B() { value2 = value2 + value1;}

proctype Monitor(){ do :: assert(value3 == 3 ||
                                 value3 == 5 ||
                                 value3 == 6) od;}
init { atomic{ run Monitor(); run A(); run B()}}
```

Exercise 2

```
proctype Monitor()
{
        assert(nfull(TunnelAB) &&
               nfull(TunnelBC) &&
               nfull(TunnelCD) &&
               nfull(TunnelDA))
}
```



Exercise 3

```
#define p (full(TunnelAB) || full(TunnelBC) || full(TunnelCD) || full(TunnelDA))

[] !p
```

Exercise 4

> *If a request for chocolate is sent to the vending machine then chocolate will eventually be delivered.*

Translates into `[](p -> <> q)` where `p` is defined to be `len(coin_channel == 1)` and `q` is defined to be `len(choc_channel == 1)`.

Exercise 5

The property relating the `coin_box` to the number of `milk_bars` and `plain_bars` only holds after a transaction is complete therefore the best we can specify is the following:

```
[](p -> <> q)
```

where `p` is defined to be `len(coin_channel == 1)` and `q` is defined to be:

```
coin_box == (450-(milk_bars*20 + plain_bars*50))
```