Assignment SMV

CS4211 - Formal Methods for Software Engineering

October, 2021

Instructions

Note:

- **This assignment is due before 9:59 PM, Sunday, 14th November, 2021**
  No late submissions!
- This is an individual assignment. Acts of plagiarism are subjected to disciplinary action by the university
- Create a folder named your matriculation number YourMatricNumber, e.g. U123456M.
- Create the following files in this folder (name these files exactly as instructed.)
  - Assignment
    - Problem 1: Include your translated NuSMV source file needed
    - Problem 2: Include your implemented NuSMV source file needed
    - Problem 3: Include your implemented NuSMV source file needed
  - Report.PDF
  - ReadME.txt
- Zip (using WinZip) the entire YourMatricNumber folder (including the folder itself and all files in it) into a file YourMatricNumber.zip.
- Submit YourMatricNumber.zip to the Luminus Workbin Folder

Problem 1

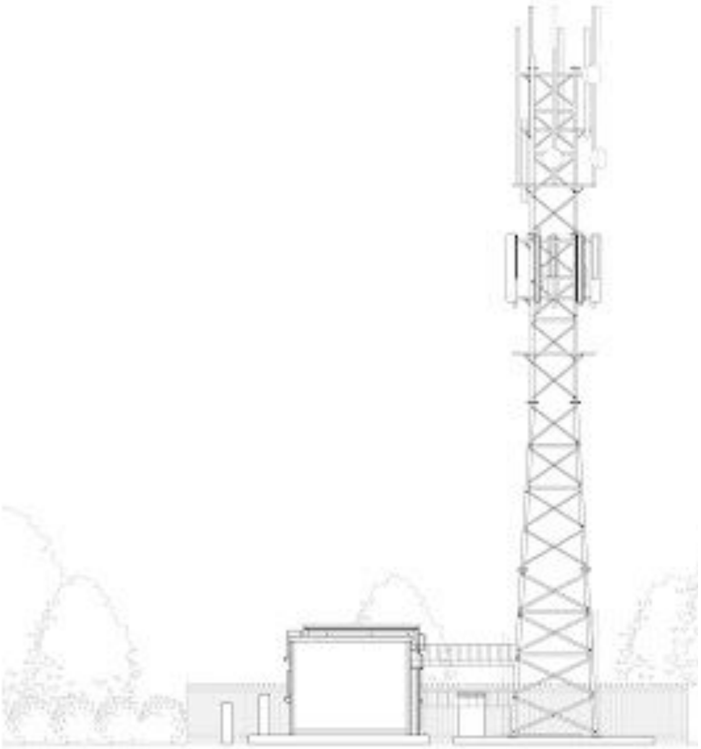
Communication Protocol

Entities

- 2 Communication Stations
- 4 Operators

Abstracted Information

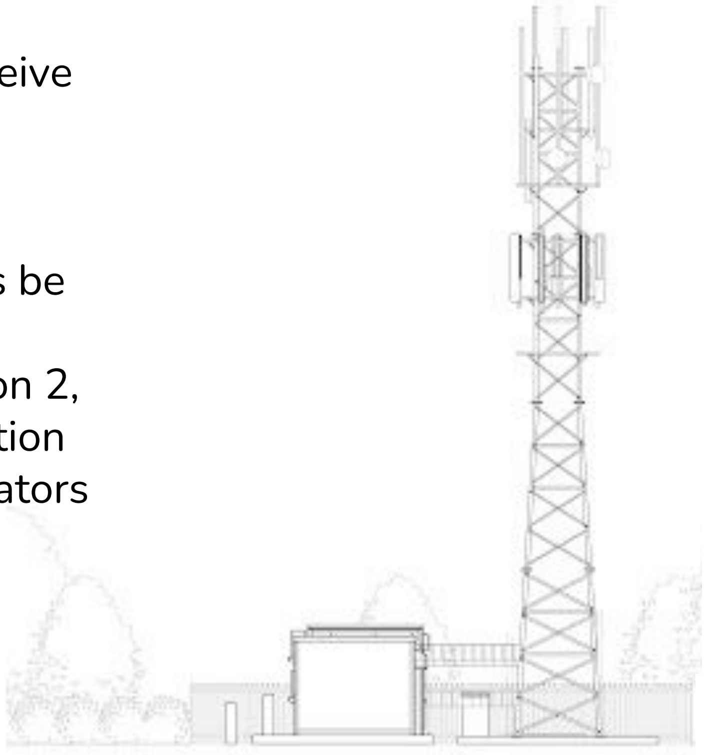
- Data/Information Communicated



Communication Protocol

- Consider two stations connected in a ring where each station can both send and receive data. Consequently, we need to assign two operators to each station.

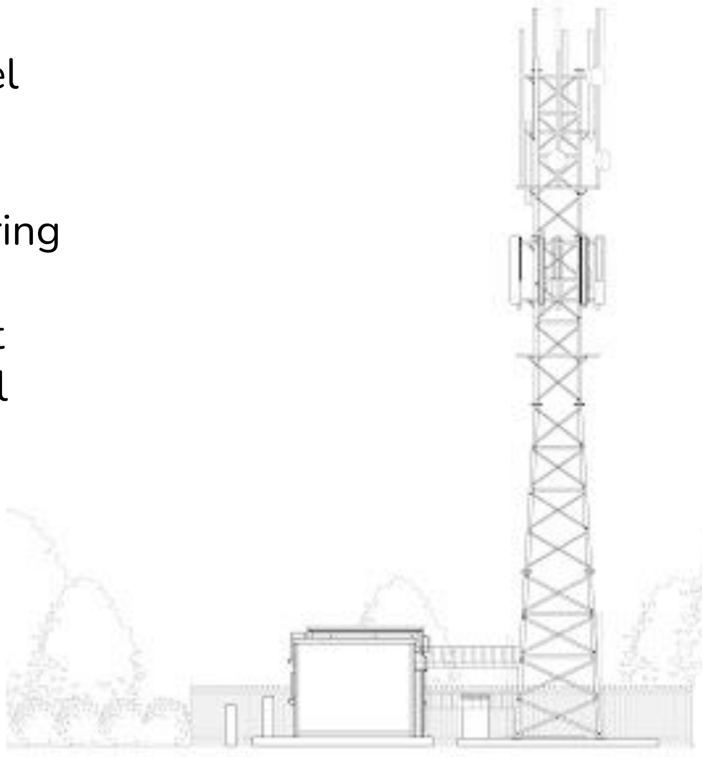
- Let the stations be 1 and 2, and the operators be 1a, 1b, 2a, 2b. Operator 1a handles data-communication from station 1 to station 2, while operator 1b handles data-communication from station 2 to station 1. Similarly for operators 2a and 2b.



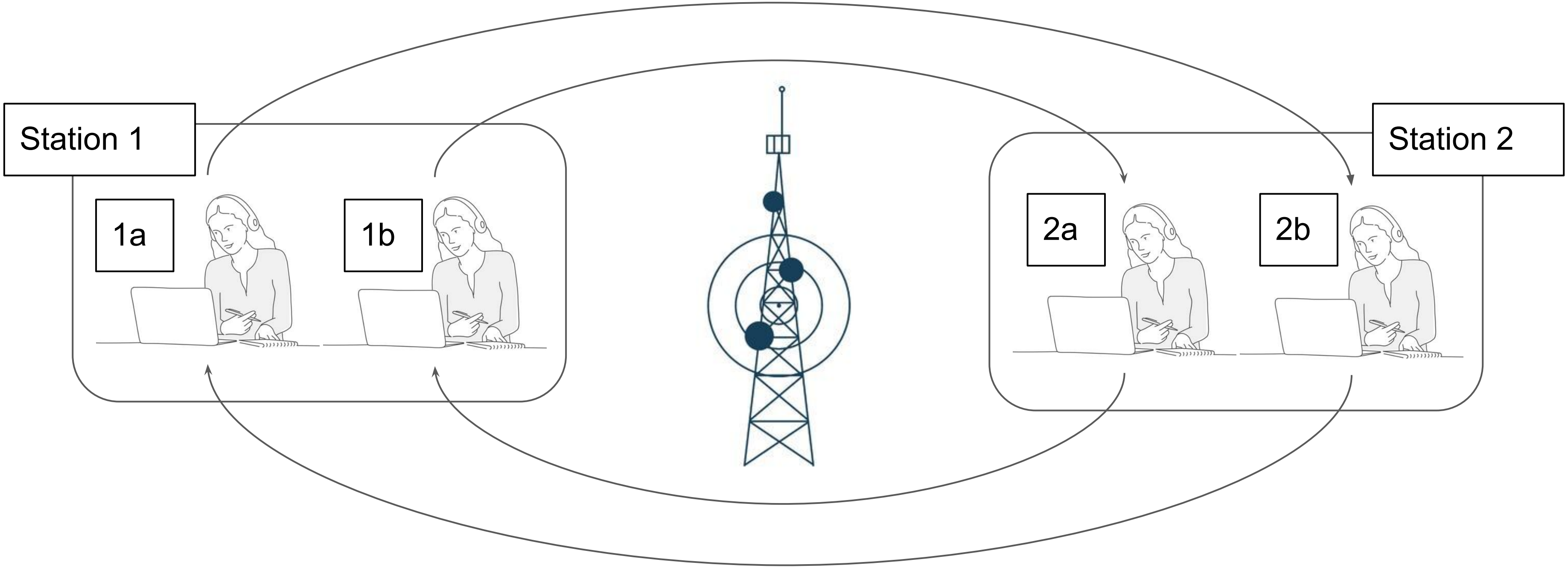
Communication Protocol

- We may not always perform communication from station 1 to station 2 via a single channel (from 1 to 2).

- For example, if the protocol involves transferring data (from 1 to 2) followed by acknowledgment(from 2 to 1), then we might want to have separate channels for the actual data items and the acknowledgment signal.



Communication Protocol



Communication Protocol

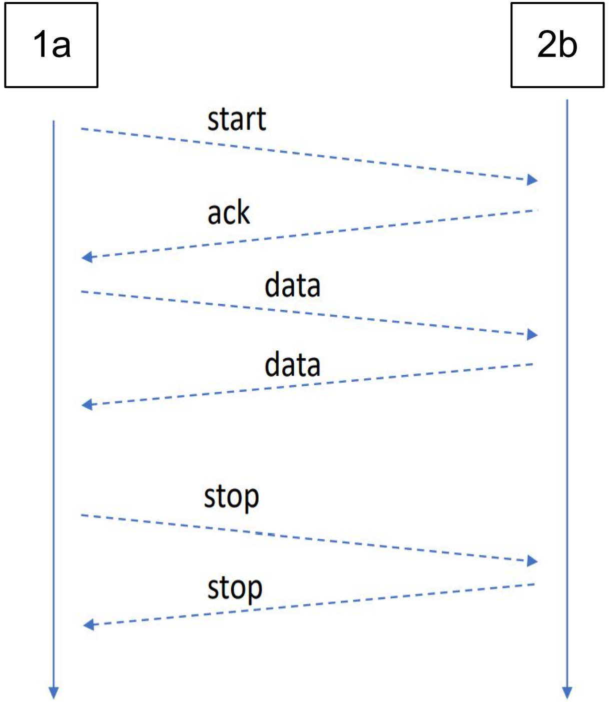
```
#define true 1
#define false 0
#define limit 3

bool busy[2];
int count = 0;

mtype = {start, stop, data, ack};
chan up[2] = [1] of { mtype };
chan down[2] = [1] of { mtype };
```

Communication Protocol

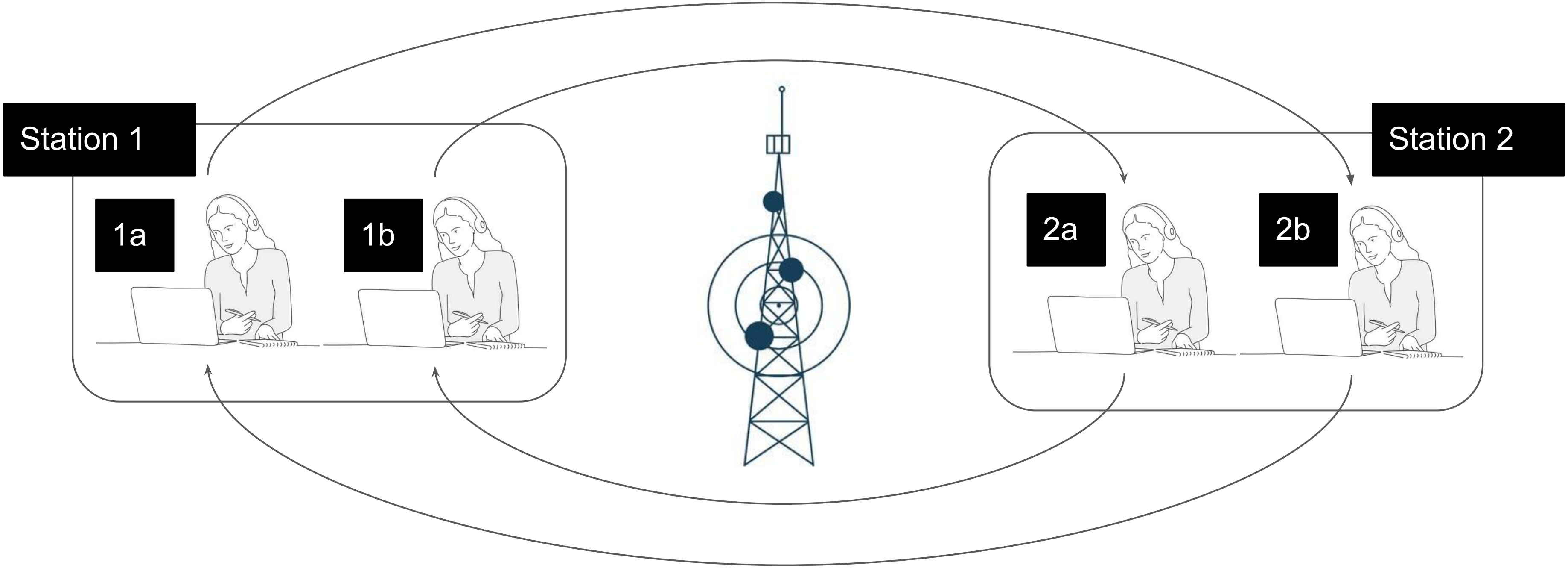
```
proctype operator(byte id; chan in, out) {
        do
        ::      in?start -> atomic { !busy[id] -> busy[id] = true };
                out!ack;
                do
                        :: in?data -> out!data
                        :: in?stop -> break
                od;
                out!stop;
                busy[id] = false
        ::      atomic { !busy[id] && !busy[1-id]  -> busy[id] = true };
                out!start;
                in?ack;
                do
                        :: (count < limit) -> out!data; count++ ; in?data
                        :: (count <= limit) -> out!stop; count = 0; break
                od;
                in?stop;
                busy[id] = false
        od
}
```



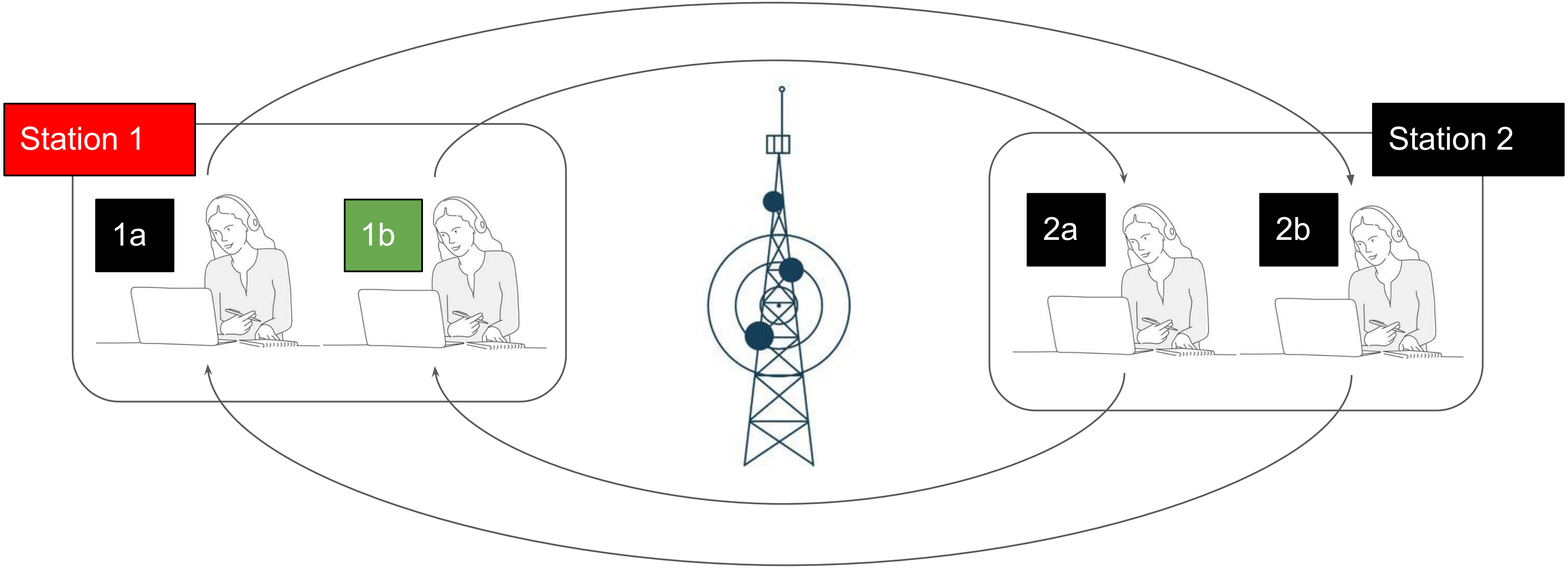
Communication Protocol

```
init {
    atomic {
        run operator(0, up[1], down[1]);
        run operator(1, up[0], down[0]);
        run operator(0, down[0], up[0]);
        run operator(1, down[1], up[1]);
    }
}
```

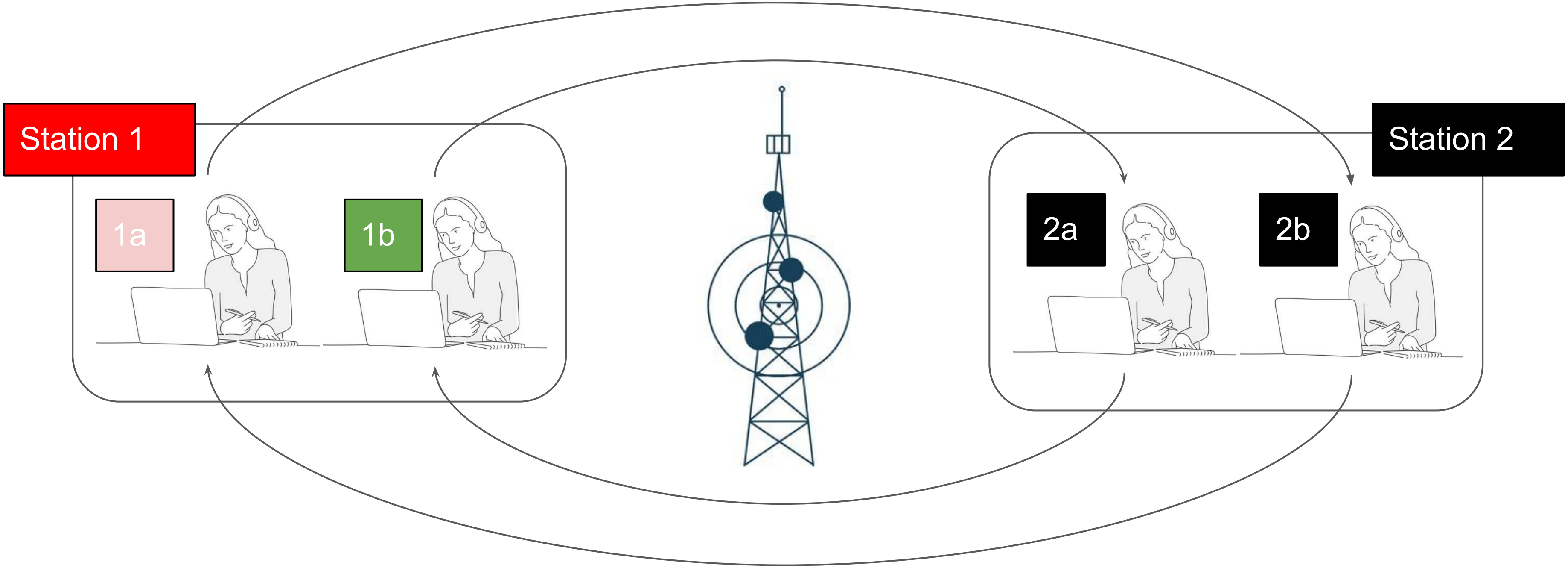
Communication Protocol



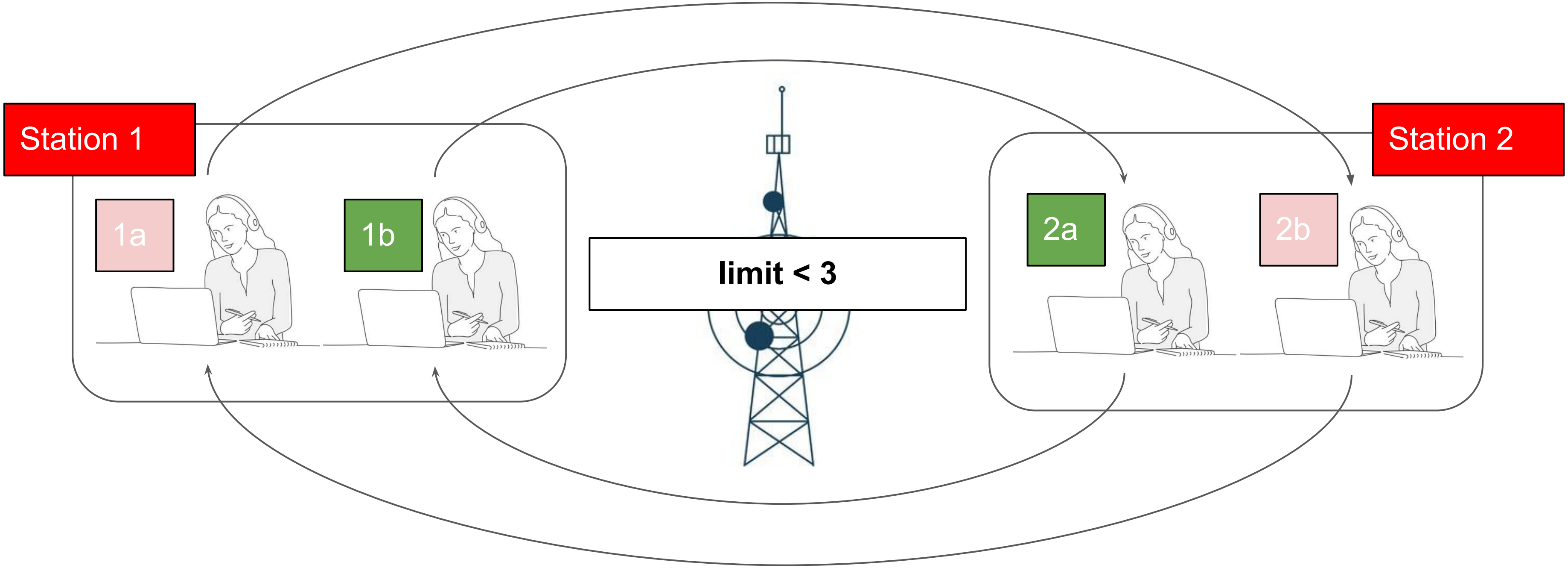
Communication Protocol



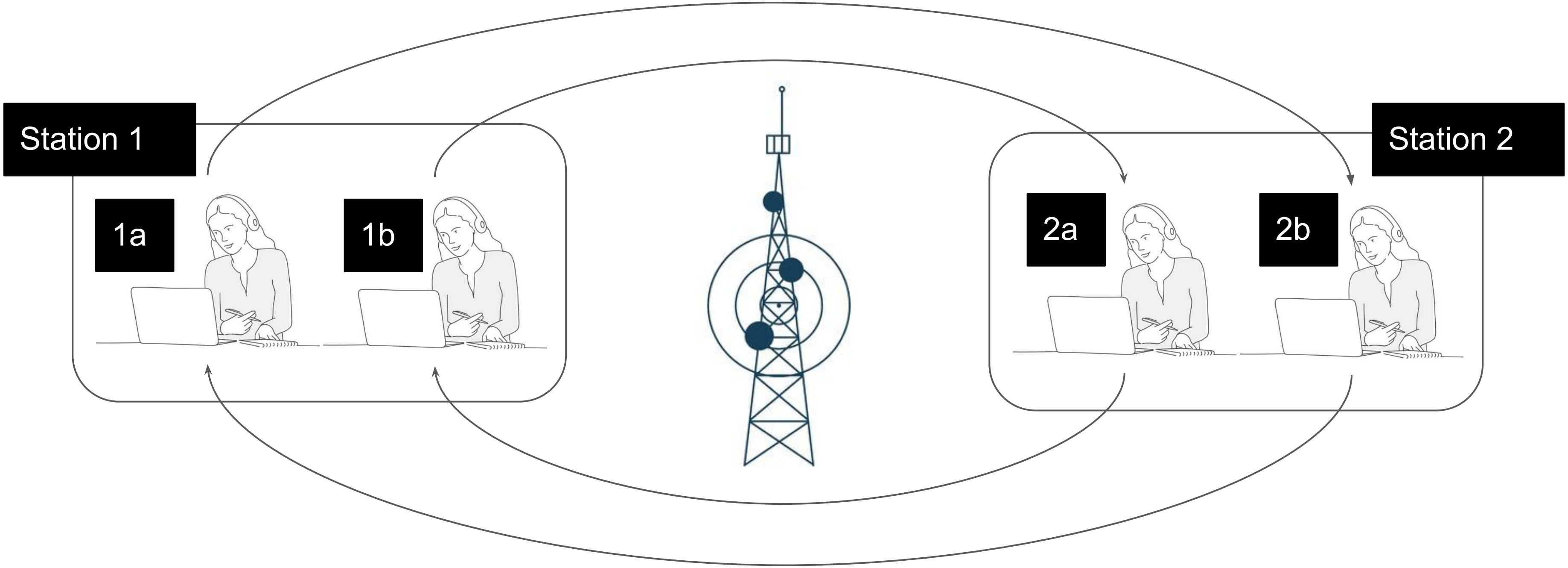
Communication Protocol



Communication Protocol



Communication Protocol



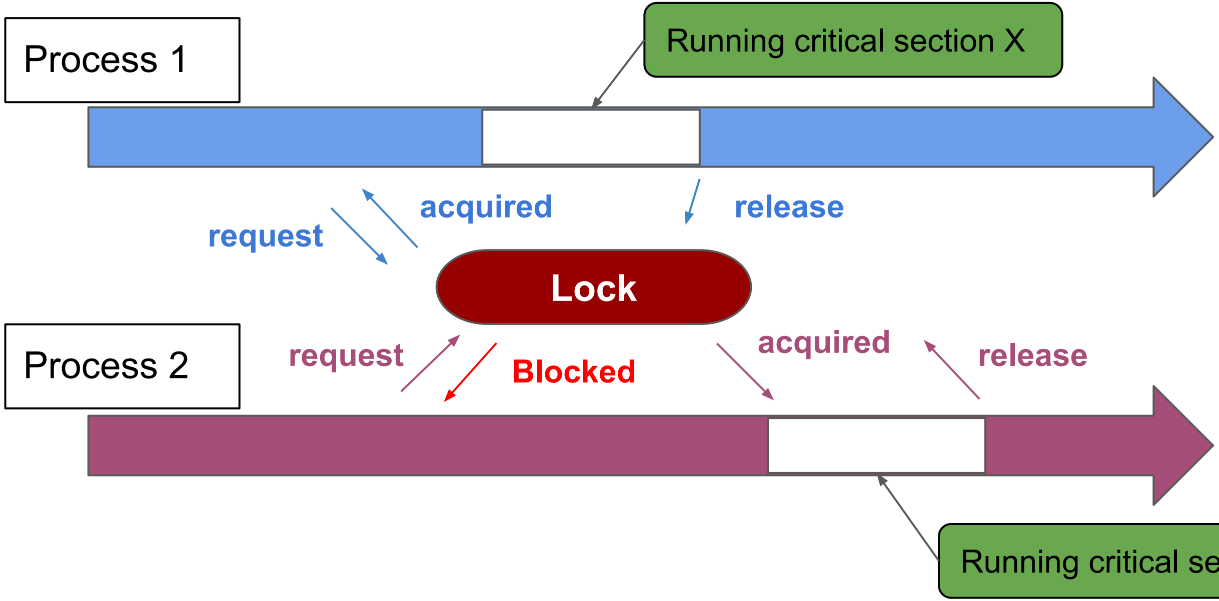
Questions [7 Marks]

A. Translate above Promela code into SMV code while adhering the protocol defined. **(5 marks)**

B. Formalize the property that no communication between stations is aborted in the above protocol, i.e. any communication that is started (with a start signal) is finished (with a stop signal). Use Linear time temporal logic (LTL) to express your property description and show whether it holds true using NuSMV. **(2 marks)**

Problem 2

Mutual Exclusion



```
do {

entry section

     critical section

exit section

     remainder section


} while (TRUE)
```

Algorithm for Mutual Exclusion

```
loop forever {

    /* non-critical section */
    key1 := false;

    while (key1 = false) do
        swap(key1, lock); // atomic operation

    /* critical section */

    lock := true

}
```

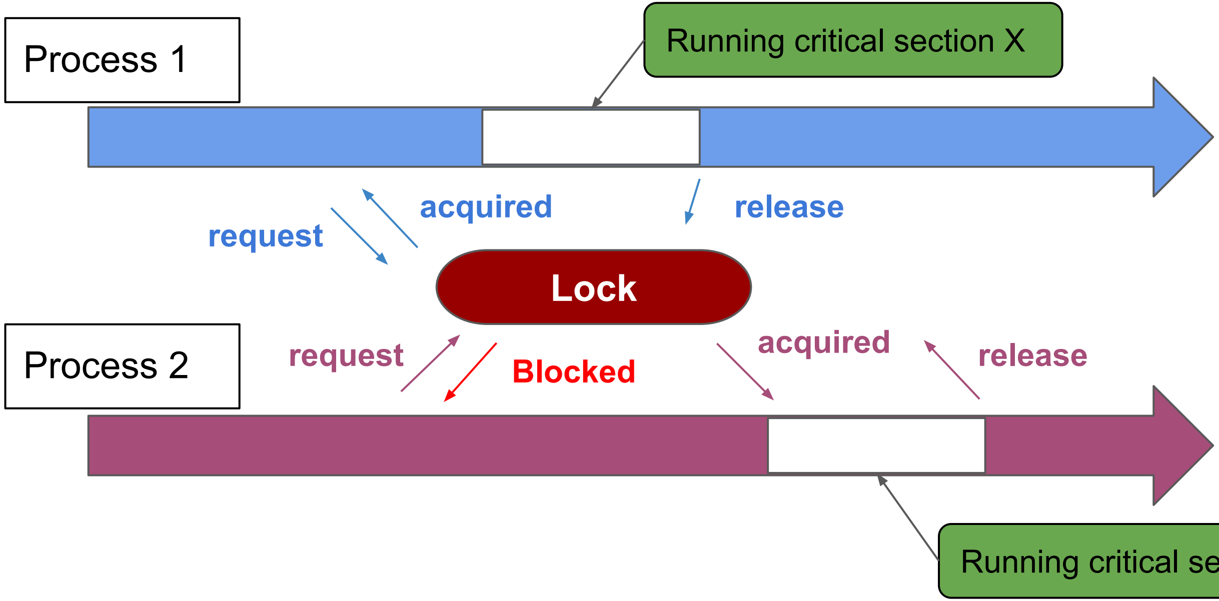
Questions [4 Marks]

A. Using NuSMV model checker show that the above solution preserves mutual exclusion.
   a. Write a SMV model to implement above solution
      (1 mark)
   b. Using a LTL formula verify whether above solution preserves mutual exclusion
      (1 marks)

B. Using a Linear time temporal logic (LTL) property show that this is not a good solution to the mutual exclusion problem. Show your proof using NuSMV and explain your answer
   (2 marks)

Problem 3

Mutual Exclusion



```
do {

entry section

    critical section

exit section

    remainder section


} while (TRUE)
```

Algorithm for Mutual Exclusion

```
C0: b[i] := true;

    k = 1 - i;

C1: if k != i and b[1-i]

                then go to C1;

    else /*critical section*/;
    b[i] := false;

    /* non-critical section */;
    go to C0;
```

Questions [4 Marks]

A. Using NuSMV model checker show that the above solution preserves mutual exclusion.
   a. Write a SMV model to implement above pseudo-code
      (1 mark)
   b. Using a LTL formula verify whether above solution preserves mutual exclusion
      (1 marks)

B. The above candidate solution is starvation-free.
   a. Write a LTL formula to show the solution is starvation free
      (1 mark)
   b. Show proof using NuSMV and explain your answer
      (1 marks)

Q&A