6SENG002W Concurrent Programming

FSP Process Composition Analysis & Design Form

Name	K.M.Malmi Imasha
Student ID	20191224
Date	1/12/2023

1. FSP Composition Process Attributes

Attribute	Value
Name	PRINTER_SYSTEM
Description	The process of using a printer involves students and multiple steps. In this scenario, there are two students (student1, and student2) attempting to print a total of two and three documents respectively. Additionally, a technician (technician) is responsible for monitoring and maintaining the printer, including refilling it with papers when necessary.

Alphabet
(Use LTSA's compressed notation, if alphabet is large.)
	Alphabet (PRINTING_SYSTEM) = {
	{start, {student1, student2}.{{acquire, empty}, print[0..3], {refill_printer, release}}, technician.{{acquire, empty}, print[1..3], {refill_printer, release, wait}}}
	}

Sub-processes (List them.)	PRINTER, student1: STUDENT (2), student2:STUDENT(3) TECHNICIAN
Number of States	56
Deadlocks (yes/no)	NO
Deadlock Trace(s) **(If applicable)**	Not Applicable

2. FSP "main" Program Code

The code for the parallel composition of all of the sub-processes and the definitions of any constants, ranges & process labelling sets used. (Do not include the code for the other sub-processes.)

FSP Program:

```
// USERS SET
set All_USERS = {student1, student2, technician}

// PRINT ACTION SET
set PRINT_Actions = { acquire, print[DOCUMENT_COUNT], release, empty,
refill_printer}

|| PRINTING_SYSTEM = ( student1: STUDENT(2) || student2: STUDENT(3) ||
technician : TECHNICIAN || All_USERS::PRINTER )
/ {start/student1.start,start/student2.start}.
```

3. Combined Sub-processes

(Add rows as necessary.)

Process	Description
PRINTER	A printer with the capacity to hold only three sheets at a time.
STUDENT(2)	The action of a student who wants to print two documents
STUDENT(3)	The action of a student who wants to print three documents
TECHNICIAN	The task of technician responsible for refilling the printer with papers, especially refilling it with three sheets at a time.

4. Analysis of Combined Process Actions

- **Synchronous** actions are performed by at least two sub-process in the combination.
- **Blocked Synchronous** actions cannot be performed, since at least one of the sub-processes cannot preform them, because they were added to their alphabet using alphabet extension.
- **Asynchronous** actions are preformed independently by a single sub-process.

Group actions together if appropriate, for example if they include indexes,
e.g. in[0], in[1], …, in[5] as in[1..5].

(Add rows as necessary.)

Synchronous Actions	Synchronized by Sub-Processes (List)
start	STUDENT(2), STUDENT(3), PRINTER
s1.acquire, s1.print[1], s1.print[2], s1.release	STUDENT(2), PRINTER
s2.acquire, s2.print[1], s2.print[2], s2.print[3], s2.release	STUDENT(3), PRINTER
tcn.empty, tcn.refill_printer, tcn.release	TECHNICIAN, PRINTER

Sub-Process	Asynchronous Actions (List)
TECHNICIAN	technician.wait
PRINTER	None
STUDENT(2)	None

STUDENT(3)	None

5. Parallel Composition Structure Diagram

The structure diagram for the parallel composition.