Certified LabVIEW Architect Examination

Examinee _____ Date: _____

Administrator _____ Date: _____

Instructions:

If you did not receive this exam in a sealed envelope stamped "NI Certification," **DO NOT ACCEPT** this exam. Return it to the proctor immediately. You will be provided with a replacement exam.

- **Please do not detach the binding staple of any section. If any part of the exam paper is missing or detached when returned to National Instruments, you will be deemed to have failed the exam**.
- This examination may not be taken from the examination area or reproduced in any way. You may not keep any portion of this exam after you have completed it.
- Please do not ask the proctor for help. If you believe the intent of any part of the exam is not clear, you may make appropriate assumptions. Please document your assumptions on the LabVIEW block diagram of the appropriate VI.
- The exam requires you to architect a LabVIEW application based on a set of requirements.
- A computer with a standard installation of LabVIEW is the only reference allowed for the examination. Externally developed code and third party tools are not allowed in the exam.
- You may use LabVIEW design patterns, templates, and examples available in the development environment as a guide/resource for the application development.
- The application architecture must be specifically developed for the exam submission.
- Submit your LabVIEW application on the USB memory stick provided.
- Total time allocated for the exam: 4 hours
- Exam passing grade: 70%

NON-DISCLOSURE AGREEMENT AND TERMS OF USE FOR NATIONAL INSTRUMENTS EXAMS

- This exam is confidential and is protected by trade secret law. It is made available to you, the examinee, solely for the purpose of becoming certified in the technical area referenced in the title of this exam.
- You are expressly prohibited from disclosing, publishing, reproducing, or transmitting this exam, in whole or in part, in any form or by any means, verbal or written, electronic or mechanical, for any purpose, without the prior express written permission of National Instruments - Training & Certification.
- By beginning work on the exam, you are accepting the NDA statement and agree not to disclose the content of this CLA Exam.

IMPORTANT:

- **When you have completed the exam, place the exam document, the USB memory stick with the saved application, and any deliverables in the envelope provided.**
- **Please SEAL the envelope.**
- **Give the sealed envelope to your proctor.**

Section I: Project Deliverables

For the purposes of this exam, you are required to design an architecture that covers all of the Project Specifications listed below. Your architecture should include implementations of the following key components:

- Project with hierarchy
- Main VI
- Shell (stub) modules and subVIs
- Interface for hardware
- Important data structures
- Inter-process communication mechanisms
- Application Programmers Interfaces (APIs) for all modules
- Error handling strategy
- Application shutdown strategy

Assume that specific algorithms, states, and functions will be implemented by a secondary developer. These do not need to be implemented as part of the test. However, for each requirement which is not implemented, you should include localized comments to a developer that describes a method for implementing the requirement.

Requirements Tracking

In order to demonstrate coverage of a requirement, you must include the ID of the requirement in the documentation of your architecture. Each requirement's ID precedes the requirement text in the specification below. You may cover requirements in any part of your architecture's documentation, including:

- VI Documentation Property
- Control Documentation Property
- Project or Library Documentation Property
- Comments on the front panel or block diagram

A single requirement may be covered by multiple sections of code if all of those sections are necessary to fulfill the requirement.

To cover a requirement, include the following text in the documentation of your code:

`[Covers: ID]`

A requirement tracking tool is used to verify your program, therefore, please be careful to use this exact syntax. You should also double-check that each requirement ID is correct.

The provided USB memory stick contains a text file that has all of the Tags. This file is provided as a convenience for use in placing the tags in the application code.

Grading:

The application architecture development exam consists of a total of 100 points which are allocated as follows:

- User interface and block diagram style : 10 points
- Documentation : 20 points
- Requirements coverage : 30 points
- Architecture development : 40 points

Section II: Elevator Control System Project Specifications

Objective

Design a scalable, modular architecture in LabVIEW for an Elevator Control System. The client has provided the following sketch of the user interface. The architecture will be distributed to a team of Certified LabVIEW Developers to work on one or multiple modules. Each module must have all the instructions for the developer to complete the development and integration into the main project.

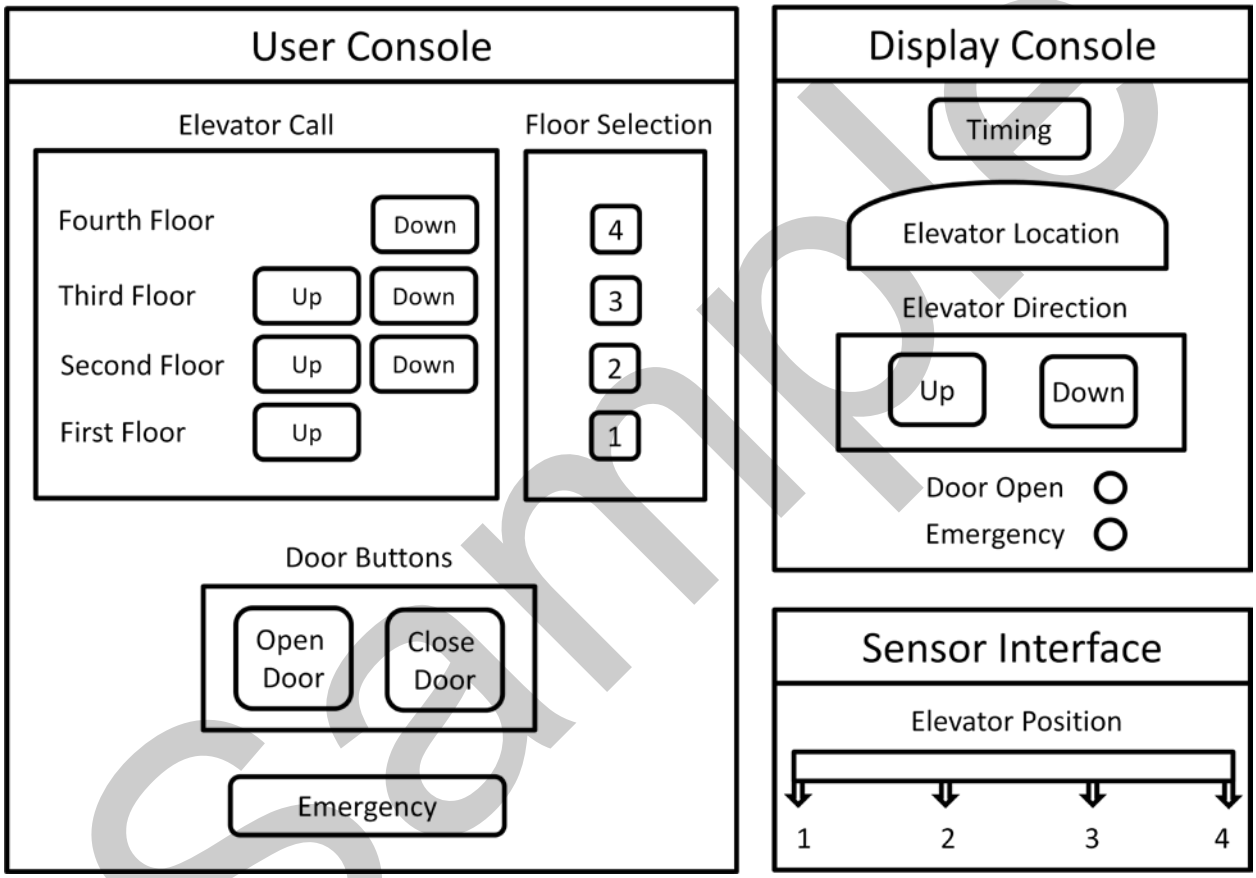

Figure 1: User Interface Sketch

Overview

The Elevator travels between floors and responds to requests to pick up and drop off passengers. The **User Console** is used to call the Elevator, select a floor from inside the elevator, operate the doors, and report an emergency situation. The **Display Console** displays the current Elevator location, elapsed time, indicates door status, and Elevator travel direction. The **Sensor Interface** is used to simulate the position of the Elevator. After completing all user requests, the Elevator times out and waits for another user request.

Definitions

Application Simulation Interface

A program that is run to test Elevator algorithms in a software-only environment.

Elevator Controller

A software component that monitors and controls the Elevator movement by interacting with the **User Console**, **Display Console**, and **Sensor Interface**. The Elevator Controller performs the following main functions:

- Uses the Configuration Database to read timing parameters.
- Monitors the **Sensor Interface** for the current **Elevator Position**.
- Updates the **Elevator Location**.
- Monitors the **User Console** to get new **Elevator Calls** and **Floor Selections**.
- When no Requests are pending, wait a specified time, then move Elevator to the first floor and wait.
- Responds to Emergency.

User Console

The interface where a user calls the Elevator, makes a floor selection, and reports an Emergency. All references to the **User Console** apply to a Simulated or Physical User Console.

Simulated User Console

A software interface that simulates a Physical User Console.

Physical User Console

A hardware interface that includes physical controls that allows a user to interact with the Elevator.

Display Console

The interface where a user receives status updates from the Controller. All references to the **Display Console** apply to a Simulated or Physical Display Console.

Simulated Display Console

A software interface that simulates a Physical Display Console.

Physical Display Console

A hardware interface that includes a physical display that allows a user to receive status updates from the Controller.

Sensor Interface

The interface where a user changes the floor location of the Elevator using the **Elevator Position** control. All references to the **Sensor Interface** apply to a Simulated or Hardware Sensor Interface.

Simulated Sensor Interface

A software interface that simulates physical sensors.

Hardware Sensor Interface

The hardware interface that includes physical sensors and a data acquisition device.

Configuration Database

An interface that stores delay time parameters for travel between floors, leaving doors open, and inactivity. All references to the Configuration Database apply to both the Simulated and Embedded Configuration Database.

Simulated Configuration Database Module

A software interface connected to a database that simulates an Embedded Configuration Database.

Embedded Configuration Database Module

A software interface that accesses the Embedded Configuration Database.

Terminology

The following is a description of the Elevator terminology and operation. Specific details for timing, control and indicator behavior, database access, data calculations, and error handling are described in the *Requirements* section of this specification.

Elevator Call

Elevator Call allows a user to call an Elevator from their floor and select the direction they want to travel.

Floor Selection

Floor Selection allows a user to select a floor destination from inside the Elevator.

Open Door/Close Door

Open Door and **Close Door** allow a user to open or close the doors. These controls are only enabled when the Elevator stops at a floor.

Emergency

Emergency allows a user report an emergency situation from inside the Elevator. When the **Emergency** button is pressed, the Elevator stops at the next floor and opens the door. The doors remain open and the Elevator does not travel.

Elevator Position

Elevator Position allows a user to confirm the Elevator has reached the destination floor. The **Elevator Position** is changed when the **Timing** has completed counting down from the Travel Time.

Travel Cycle

A Travel Cycle is initiated when a user calls an Elevator by pressing an **Elevator Call** control, and ends when all user Requests are complete and the Elevator returns to the first floor.

Request

A Request is placed when a user calls an Elevator from their floor or selects a floor destination from inside the Elevator.

Elevator Call

A Request from outside the Elevator to travel **Up** or **Down** from a floor. During Elevator travel, the Elevator only stops for **Elevator Calls** for floors with an **Elevator Call** Request in the same direction as the current **Elevator Direction**. Before each floor is reached, the Elevator Controller checks if there is a call initiated from that floor in the current direction of the travel.

Floor Selection

A Request from inside the Elevator to a destination floor. During Elevator travel, the Elevator only stops for floors with a **Floor Selection** Request in the same direction as the current **Elevator Direction**. Before each floor is reached, the Elevator checks if there is a selection for that floor.

Elevator Direction

Once the Elevator has started moving it continues to travel in the same direction until all Requests in that direction have been fulfilled.

The Elevator stops for all **Floor Selections** in the direction it travels. The Elevator stops for all **Elevator Calls** that are in the same direction as the current **Elevator Direction**. The Elevator continues in the same **Elevator Direction** until all calls and selections are complete. The Elevator then starts servicing Requests for the opposite direction. If there are no pending **Floor Selections** or **Elevator Calls**, the Elevator waits for the Inactivity Time to elapse and then returns to the first floor.

The Elevator Travel Cycle

The Elevator repeats the following cycle as it travels between floors. This cycle starts when the doors open on a floor, and ends when the Elevator stops at the destination floor.

Steps in the Travel Cycle:

1. Open doors
2. Elapse time for open doors
3. Close doors
4. Start travel
5. Elapse time for travel
6. Destination reached as indicated by the position of the elevator
7. Evaluate all calls and requests
8. Determine if the Elevator stops on a floor or continues to the next floor
9. Stop or continue travel

Simulation Interface

Due to the unavailability of a physical setup, the client has requested the application have simulations of each module. The simulated modules will be used to validate the Elevator Controller logic and algorithms.

The simulated modules will be replaced with physical modules in a subsequent project phase. For each module, the client must have the ability to choose either a simulated or hardware module.

Requirements

Simulation Interface Requirements

SI1. The components of the Simulation Interface are separated from the Elevator Controller by defined interfaces that simulate communication with the physical modules.

SI2. The Simulation Interface automatically loads and starts the **User Console**, **Display Console**, **Sensor Interface**, Configuration Database and other modules upon startup.

User Interface Requirements

Develop the front panels using appropriate control types in order to meet requirements.

UI1. The Simulation User Interface mimics the *Figure 1* sketch.

The **User Console** is used to call an Elevator, select floors, open and close doors, and make Emergency requests.

UI2. The Elevator includes a **User Console** component.

- UI2A. **Elevator Call**. These controls call the Elevator to the user's floor.
  - UI2A1. The **Elevator Call** controls are labeled **Up** and **Down** to indicate the direction the user wants to travel.
  - UI2A2. The **Elevator Call** controls remain ON until the **Elevator Call** Request is complete.
- UI2B. **Floor Selection**. These controls are pressed by the user from inside the Elevator to request a destination floor.
  - UI2B1. When an Elevator arrives on a floor with an active **Floor Selection** Request, the doors open.
  - UI2B2. Each **Floor Selection** control remains ON until the **Floor Selection** Request is complete.
- UI2C. **Open Door** and **Close Door**. These controls are used to request the doors to open or close.
  - UI2C1. **Open Door** opens the Elevator doors when the Elevator is stopped on a floor.
  - UI2C2. **Close Door** closes the Elevator doors when the Elevator is stopped on a floor.
  - UI2C3. **Open Door** and **Close Door** are disabled when the Elevator is travelling between floors.
- UI2D. **Emergency**. This button is used to report an Emergency.
  - UI2D1. The **Emergency** button is pressed by the user to signal that an emergency situation exists inside the Elevator.

The **Display Console** uses indicators to displays **Timing** information, current **Elevator Location**, current **Elevator Direction**, and indicates when the doors are open.

UI3. The Elevator includes a **Display Console** component.

- UI3A. **Elevator Location**. This indicator displays the location of the Elevator.
  - UI3A1. Continuously updates the location of the Elevator as it travels between floors.
  - UI3A2. Displays the current location of the Elevator when the Elevator stops.
- UI3B. **Door Open**. This indicator turns ON when the doors are open.
- UI3C. **Timing**. This indicator displays the elapsing time for each timed event (doors, travel, inactivity). The **Timing** counts down to 0 from the specified time in *Table 2*.
- UI3D. **Elevator Direction**. This indicator displays the direction of Elevator travel.
  - UI3D1. These indicators display the direction the Elevator is traveling.
  - UI3D2. When the Elevator has no Requests and is waiting for a Request, both direction indicators are OFF.
- UI3E. **Emergency**. This indicator is ON when the **Emergency** button is pressed on the **User Console**.

The **Sensor Interface** simulates the elevator travelling between floors.

UI4. The Elevator includes a **Sensor Interface** component.

- UI4A. **Elevator Position**. This indicator simulates the Elevator changing floors. It remains at the last floor position until the user moves it to a new floor when the Travel Time expires.

Initial State Requirements

This section defines how the Elevator application is initialized when the Run button is pressed, before any user controls are pressed.

IS1. **User Console**:

- All **Elevator Call** controls are OFF.
- All **Floor Selection** controls are OFF.
- **Open Door** and **Close Door** controls are OFF.
- The **Emergency** button is OFF.

IS2. **Display Console**:

- **Timing** is set to 0.0.
- **Elevator Location** is set to the first floor.
- **Elevator Direction**, **Door Open**, and **Emergency** indicators all OFF.

IS3. **Sensor Interface**

- The **Elevator Position** is set to the first floor.

Wait for First Request Requirements

WS1. Maintain *Initial State* defaults until an **Elevator Call** is made.

WS2. If an **Elevator Call** is requested from the first floor, start the *Open and Close Door Sequence*.

WS3. If an **Elevator Call** is made from another floor, start *Continue Travel State.*

Start and Move State Requirements

SS1. Complete the *Open and Close Door Sequence*.

SS2. Disable the **Open Door** and **Close Door** controls.

SS3. Turn the **Elevator Direction** indicator ON for the direction the Elevator is traveling.

SS4. Start the **Timing** indicator at the Travel Time from the Configuration Database specified in *Table 2*.

SS4A. Continuously update the **Elevator Location** indicator as the **Timing** elapses and the Elevator travels to the next floor.

SS4B. When **Timing** reaches 0, the **Elevator Location** indicator stops at the current floor.

SS5. Change the **Elevator Position** sensor to match the **Elevator Location** indicator on the **Display Console**.

SS6. Start *Analysis State.*

Analysis State Requirements

AS1. Read all **User Console** controls and the current **Elevator Direction** indicator on the **Display Console**.

AS2. The Controller determines if the Elevator should stop at the current floor, or if the Elevator should continue travelling in the current **Elevator Direction** and fulfill another Request, as specified in *Table 1*.

AS2A. If the Elevator stops on the current floor, start *Stop at Floor*.

AS2B. If the Elevator continues in the current **Elevator Direction**, start *Continue Travel State*.

AS2C. If the Elevator changes direction, start the *Start and Move State.*

		Total Request Status			
		Request in the current direction at current floor	No Request at current floor, but Call in current direction	No Request at current floor or have Call in opposite direction at current floor	Call in opposite direction
Current Elevator Requests for current direction	Request in current direction	*Stop at Floor*	*Continue Travel State*	*Continue Travel State*	*Continue Travel State*
	No Selection in current direction, but call in current direction	—	*Continue Travel State*	*Continue Travel State*	*Continue Travel State*
	No Selection, no call In current direction	—	—	Change direction, or travel in current direction to Call, then change direction	Change direction or travel in current direction to Call, then change direction

Table 1: Analysis Table

Continue Travel State Requirements

CT1. Start the **Timing** indicator using the Travel Time from the Configuration Database, specified in *Table 2*.

CT1A. Continuously update the **Elevator Location** indicator as the **Timing** elapses and the Elevator travels to the next floor.

CT1B. When the **Timing** reaches 0, the **Elevator Location** stops at the current floor.

CT2. The **Elevator Position** sensor is manually changed to match the **Elevator Location** to indicate the Elevator has reached its destination.

CT3. Start *Analysis State.*

Stop at Floor Requirements

RF1. Turn the **Floor Selection** control OFF for the current floor.

RF2. Turn the **Elevator Call** control OFF for the current floor and direction.

RF3. Start *Open and Close Door Sequence*.

Open and Close Door Sequence Requirements

OC1. When the **Elevator Position** is changed to the current floor, enable both **Door Buttons**.

OC2. The **Door Open** indicator turns ON.

OC3. Start **Timing** when the doors open and countdown to 0 using the time specified in *Table 2*.

OC3A. If the **Open Door** is pressed, restart the **Timing**.

OC3B. Once the **Timing** has reached 0, disable **Open Door**.

OC4. If **Close Door** is pressed, the **Timing** changes to 1 second if it is currently above 1 second.

OC5. The **Door Open** indicator turns OFF.

OC6. If there is a Request, disable the **Door Controls** and begin *Start and Move State*.

OC7. If there is no Request, start *No Request Remaining*.

Timed Action	Seconds Displayed on Timing
Open Door	5 seconds
Close Door	1 second
Travel Time (between two floors)	10 seconds
Inactivity Time	20 seconds

Table 2: Time Table

No Request Remaining Requirements

NR1. When all **Elevator Calls** and **Floor Selections** have been fulfilled and the doors are closed, the **Timing** displays the Inactivity Time and counts down to 0.

NR1A. When **Timing** reaches 0, the Elevator receives a Request for the first floor.

NR1B. If any Requests are received en route the Elevator removes the first floor Request, unless the new Request is for the first floor.

Emergency Call Requirements

EC1. If the **Emergency** button is pressed, the Elevator continues travelling to the next floor. If the Elevator is already stopped at a floor, it remains at the current **Elevator Location**.

EC2. Once the Elevator is stopped at a floor, the doors open and remain open.

EC3. If the **Emergency** button is turned OFF the Elevator clears all Requests and begins a new Elevator initialization.

Terminate Process Requirements

TS1. All components of the **User Console** are disabled.

TS2. Currently executing processes are aborted.

TS3. All Elevator indicators turn OFF.

TS4. **Timing** displays 0.0.

Configuration Database Requirements

CD1. The file format of the database must be appropriately chosen to support the following records.

- CD1A. **Door Open**. This value indicates the time the doors stay open.
- CD1B. **Door Close**. This value indicates the time the doors take to close.
- CD1C. **Travel Time**. This value indicates the travel time between floors.
- CD1D. **Inactivity Time**. This value indicates the time the Elevator waits without Requests before returning to the first floor.

CD2. The Controller reads and updates the Configuration Database.

CD3. If the Configuration Database is not found, the Controller reads and uses default parameters.

CD4. The Configuration Database has an interface used by the Controller to read, and update records.

Error Handling Requirements

The Elevator has centralized error handling with three categories of errors.

EH1. **Console Error**. Provides a warning to the user. The user is notified of the error and the Elevator continues operations without interruption.

- EH1A. If an error occurs in the **Display Console**, the centralized error handler displays a notification dialog to the user. When the user clicks OK, the error clears and the Elevator continues its last operation.
- EH1B. If an error occurs in the **User Console**, the centralized error handler displays a notification dialog to the user with the choice to clear or continue with the error.
  - EH1B1. If the error is cleared, the Elevator continues the last operation or process step.
  - EH1B2. If the user continues, without clearing the error, the error is handled as a Process Step Error.

EH2. **Sensor Interface IO Error**. The Controller executes a *Termination Process.*

EH3. **Process Step Error**. The Controller executes a *Termination Process.*

EH4. **Error Log**. The error handling system maintains a log file. All errors are logged with the following data.

- The error log includes a description of each error that occurs.
- The error log includes the level of the error.
- The error log includes the time and date at which the error occurred.