Intellux

System Design

Mechtron 4TB6 • Prof. Alan Wassyng

Group 34

Authors:

Ahmed Afifi

Abdulrahman Elgendy

Mina Ghaly

Omar Mouftah

Table Of Contents

1. Table Of Revisions

Version	Date	Authors	Description Of Revision
0	2022-01-03	Abdulrahman Elgendy Ahmed Afifi Mina Ghaly Omar Mouftah	Initial revision of the system design document
1	2022-02-28	Abdulrahman Elgendy Ahmed Afifi Mina Ghaly Omar Mouftah	Updated requirements tracing, scope and description to include new requirements and system calibration.

2. Purpose

The purpose of this project is to create a system that enables users to control their blinds remotely and automate them based on their personal preferences. Users will have the option of choosing a brightness level that they prefer through a GUI on a mobile app, and the system will continue to adjust the blinds in order to maintain the desired brightness level. Users will also have the option of manually adjusting the blinds through their smart device by increasing/decreasing the blind angle until they reach their desired brightness. All automated blinds currently on the market must be purchased as a full set, which can be quite expensive. Moreover, any accessory that tries to automate traditional blinds lacks many features when it comes to customization and truly smart features. This project is meant to allow owners of non-smart blinds the option of automating their blinds at a lower cost with similar functionality to smart blinds.

3. Scope

The scope of this document is to provide insight into the current state of the design process for Intellux. The system and its modules will be discussed in detail in the component overview allowing the reader to understand the direction the team is taking for the design of Intellux. Assumptions the team has will also be outlined in this document. The monitored variables and constants that will be discussed throughout the document are outlined and explained in the corresponding section. The overall behaviour of the system and each component will be discussed as well as possible interactions between the user and the system. Exception handling and timing constraints will also be discussed in this document.

4. Context Diagram

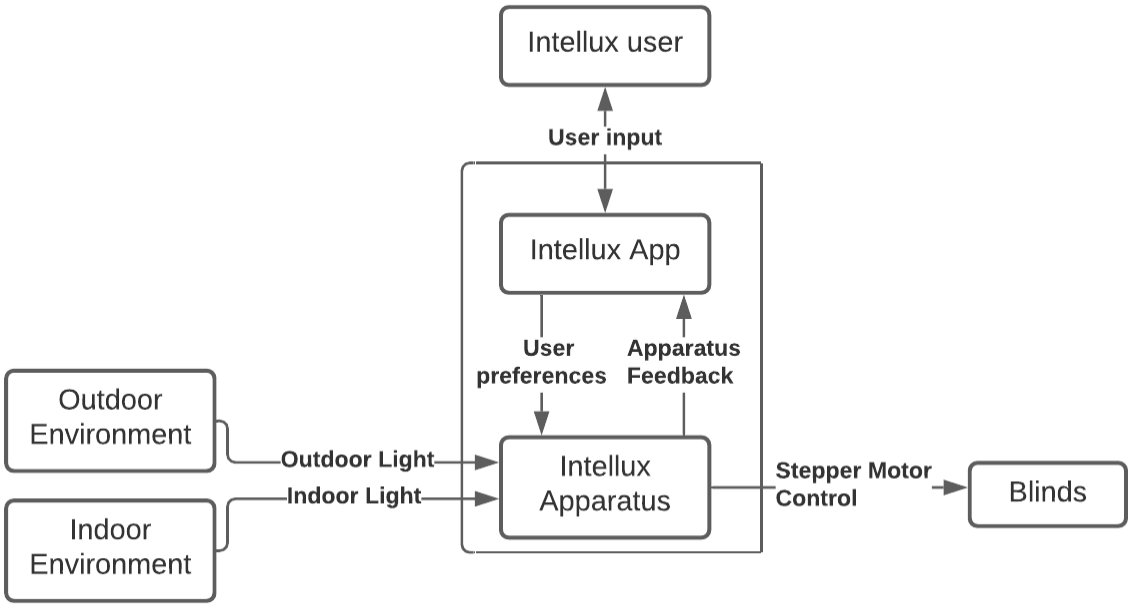

Figure 1: Intellux Context Diagram

5. Diagram Of Components

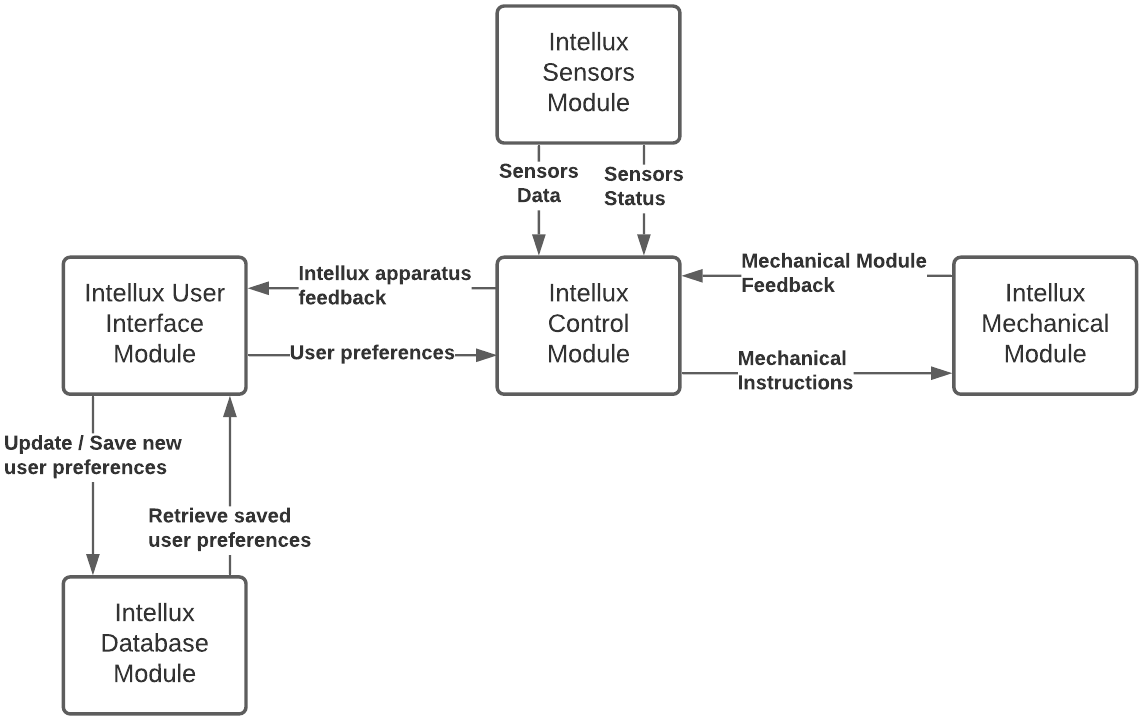

Figure 2: Intellux Diagram of Components

6. Assumptions

A1	Once Intellux is installed on a blind, the user will calibrate the system and only control the blinds through the Intellux app either in AutoMode or ManualMode
Rationale	This is to maintain the known position of the blinds angle within the control software

A2	The user must install and control the Intellux apparatus when the blinds are fully closed
Rationale	To ensure that the starting position is as expected within the control software

A3	Intellux will be installed onto vertical beaded blinds

Rationale	Cannot ensure that Intellux will perform as required on other types of blinds

A4	The user will install the apparatus correctly and with the right orientation
Rationale	This is to ensure the internal and external sensors are oriented in the right direction

A5	The user will install the apparatus on blinds with standard beads (5 mm in diameter and that the distance between them is 1.5 mm)
Rationale	To ensure that the gear teeth will be able to fit the bead size

A6	User will not obstruct sensors
Rationale	To ensure proper performance

A7	The Intellux system will only be installed in a room that has one set of blinds
Rationale	To ensure no other blinds or natural light sources interfere with the system's operation

7. Variables and Constants

7.1. Monitored Variables

Table 1: Intellux monitored variables

Variable	Type	Range	Units	Description
m_brightnessIn	Lux	0-100,000	Lux	Lux value of internal brightness
m_brightnessExt	Lux	0-100,000	Lux	Lux value of external brightness
I_Preferences	Int	0-100%	N/A	Desired brightness level as a percentage

				between 0 - 100% **(AutoMode Only)**
b_isAutoMode	Boolean	[0,1]	N/A	Boolean input whether to run auto or manual mode
I_ManualAdjustment	Int	0-100%	N/A	The desired position the blinds should be placed , where 0 - 100% covers the full range of motion of the blinds **(ManualMode Only)**

7.2. Controlled Variables

Table 2: Intellux controlled variables

Variable	Type	Range	Units	Description
targetStepperPos	Unsigned int	0 - # of steps needed to cover the full range of motion of the blinds which is 180 degrees	N/A	- The target stepper motor position based on the controller output. -The position of the stepper motor will be determined by keeping count of the number of steps taken. -The position 0 will be assigned when the blinds are fully closed at 0 degrees, and the position will be incremented with every CW step and decremented with every CCW step. The max stepper position will be assigned when the blinds are fully closed at 180 degrees

7.3. Constants

Table 3: Intellux constants

Variable	Unit	Description
k_motorSpeed	degrees/sec	The speed at which motor will rotate the blinds
k_maxAngle	degrees	Maximum Angle the blinds will reach
k_minAngle	degrees	Minimum Angle the blinds will reach
k_fullRangeSteps	N/A	The maximum number of steps taken by the stepper motor to rotate the blinds from 0º - 180º

8. Behaviour Overview

8.1. System Overview:

Typically, the user would open their Intellux application and connect to the Intellux apparatus they wish to operate. Once the devices are paired, the user will now calibrate the device to let the control module know how many stepper motor steps are needed to complete a full range of motion. The user is now free to select the mode of operation, the two available options are Auto mode and Manual mode. In Auto Mode, the user would simply select the level of brightness they wish to have in their room and Intellux would handle the rest. By measuring the brightness of the room and the external brightness, Intellux would be able to determine if the desired level is possible. Once it determines that it is possible, the blinds will automatically be rotated to match the actual measured indoor brightness with the user's desired selection. The blinds should also

automatically maintain this level of brightness throughout the day until it is no longer possible to do so. At night, Intellux would switch to Manual mode. If the user selects to operate the device in Manual mode, the application will only require an angle input from the user and the blinds will be rotated to match the user's selection.

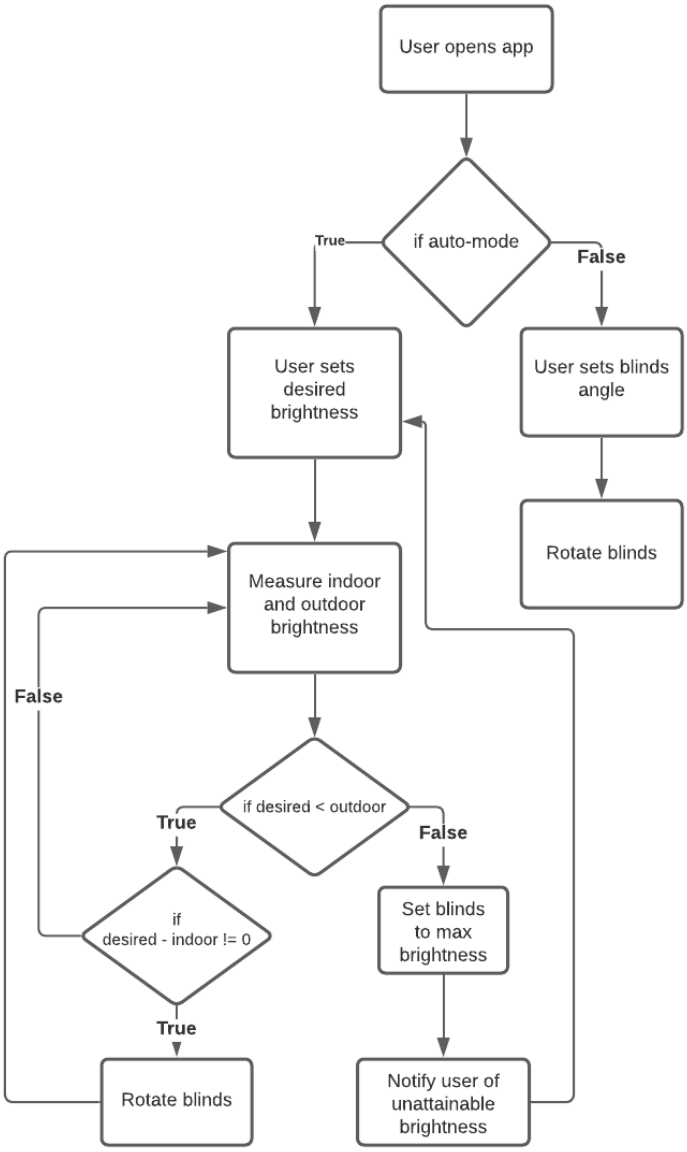

Figure 3: State Machine Diagram

Table 4: Behavior block diagram description

Step	Behavior	Mode	Description

1	User opens app	N/A	The user will open the app to see the blinds paired with the device to begin interacting with Intellux. Proceed to Step 2.
2	If auto-mode	N/A	The system will check if the user has selected to operate in Auto or Manual mode. If Auto mode, proceed to Step 3. Otherwise, go to Step 10.
3	User sets desired brightness	Auto	In Auto Mode, the user will set the brightness they wish to have in the room. Proceed to Step 4.
4	Measure indoor and outdoor brightness	Auto	Intellux must measure the current brightness of the room as well as the external brightness to know the maximum currently attainable brightness level. Proceed to Step 5.
5	If desired < outdoor	Auto	If the user has set a brightness level higher than is currently attainable, go to Step 8. Otherwise, proceed to Step 6.
6	If desired - indoor != 0	Auto	If the indoor brightness is still not matching the user's desired brightness, proceed to Step 7.
7	Rotate blinds	Auto	Rotate the blinds to match the desired brightness. Proceed to Step 4 to automatically repeat the process.
8	Set blinds to max brightness	Auto	Set the blinds to the currently maximum attainable brightness. Proceed to Step 9.
9	Notify user	Auto	Notify the user that the desired brightness is currently unattainable and that the blinds were set to the current maximum attainable brightness. Proceed to Step 4.
10	User sets blinds angle	Manual	In Manual mode, the user will set the blinds to the angle they desire. Proceed to Step 11.
11	Rotate blinds	Manual	Rotate the blinds to match the given angle in Step 10.

8.2. UI Module:

The purpose of this module is to allow the user to interact with the Intellux apparatus and save their preferences. The user interface will be displayed on mobile devices such as phones and tablets and will allow the user to insert and retrieve data through different buttons on the interface.

8.3. Database Module:

The database will store the user information and their preferences. This data will be used for other modules to ensure that the automatic adjustment is working and so that the user can view the information if desired.

8.4. Sensors Module:

The internal and external sensors will measure brightness values, which will be provided to the ADC as analog voltage readings. These values will then be sent to the control algorithm running on the Raspberry Pi to be used in calculations for the blinds angle.

8.5. Control Module:

This module acts as the main controller that commands the stepper motor to turn the beads in the direction that rotates the blinds in a direction that will bring in the desired amount of light. The two main control modes of operation are Auto Mode and Manual Mode. In Auto Mode, the user would select the desired level of brightness and the apparatus measures the brightness of the room and the external brightness. This is sent to a hysteresis controller which would decide the course of action to attain and maintain the desired brightness level. The other control mode is Manual Mode where the application will only require an angle input from the user and the blinds will be rotated to match the user's selection.

8.6. Mechanical Module:

The mechanical module is responsible for controlling the blind opening angle through a spur gear placed in between the two beaded ropes of the blinds. The gear is to be controlled by a stepper motor that will be supplied with the required voltage from an external power adaptor via a wall socket which is passed through a power circuit to allow a motor rotation speed of 30 degrees/sec.

9. Component Traceability

Table 5: User interface module traceability

Component Module	Functional and Non-Functional Requirements:
Intellux User Interface Module	**AFR1**
	AFR2
	AFR3
	AFR4
	IFR8
	IFR13
	NFR3

Table 6: Database module traceability

Component Module	Functional and Non-Functional Requirements:
Intellux Database Module	**AFR1**
	AFR2
	AFR3
	AFR4
	AFR5
	AFR6
	AFR7
	AFR8
	IFR8

Table 7: Control module traceability

Component Module	Functional and Non-Functional Requirements:
Intellux Control Module	**IFR2**
	IFR3
	IFR4

	IFR5
	IFR8
	IFR9
	IFR10
	IFR12
	IFR13

Table 8: Sensor module traceability

Component Module	Functional and Non-Functional Requirements:
Intellux Sensors Module	**IFR1**
	IFR2
	IFR4
	IFR5
	IFR8
	IFR9
	IFR11

Table 9: Mechanical module traceability

Component Module	Functional and Non-Functional Requirements:
Intellux Mechanical Module	**IFR3**
	IFR4
	IFR6
	IFR7
	IFR12
	IFR13
	NFR1

	NFR2
	NFR3
	NFR4
	NFR5
	NFR6

10. Component Overview

10.1. Intellux User Interface Module

10.1.1. Inputs And Outputs

Table 10: User interface module inputs

Input Name	Type	Range	Units	Comments
I_Username	String	1-30 Characters	N/A	Name of User
I_Preferences	Int	0-100%	N/A	- Percentage of desired maintained brightness - This variable will be outputted to the control module as **b_SetPoint** in volts
I_ManualAdjustment	Int	0-100%	N/A	Incremental brightness adjustment (**Manual mode only)** - This variable will be outputted to the control module as **I_blindsAngle** in degrees
b_isAutoMode	Boolean	[0,1]	N/A	Boolean input whether to run auto or manual mode
b_isAchievable	Boolean	[0,1]	N/A	Feedback to the user interface to inform the user whether their target brightness level could be achievable or not

b_sensorError	Boolean	[0,1]	N/A	Feedback to the user interface to notify the user of sensor error
b_rotationDirection	Boolean	[0,1]	N/A	Direction of desired rotation for calibration mode (0 corresponds to clockwise, 1 to counter-clockwise)

Table 11: User interface module outputs

Output Name	Type	Range	Units	Comments
b_SetPoint	Unsigned float	0-3.3	V	Target internal lux sensor reading
b_isAutoMode	Boolean	[0,1]	N/A	Boolean input whether to run auto or manual mode
I_blindsAngle	Unsigned int	0-180	degrees	The target blinds angle based on the user's preference when operating in manual mode. This will be fed to the mechanical module which will handle turning the blinds to the desired angle. (**Manual mode only)**

10.1.2. Description Of Behaviour

This module allows users to interact with the Intellux system from any remote device. It provides an interface for the user to input information for the system to then use to make any necessary adjustments to fulfill the users' desires. The user will be able to calibrate the system to match their specific blinds in order for it to function properly. The user will also be able to enable auto mode to allow for automatic adjustments of the blinds based on preferences or use manual mode to manually adjust the blinds based on the user's direct input.

10.1.3. User interface Development tools

The user interface will be developed using Node Red to allow easy integration between the front end and the backend for Intellux.

10.1.4. Overview Of Possible User Interactions

Table 12: Possible user interactions

Username Input	Allows users to input their username
Switch User	Allows users to switch from one saved user to another
Brightness Preferences Input	Allows users to input their preferred brightness preferences
Manually Adjust Blinds	Allow users to manually adjust blinds by increasing/decreasing blind angles incrementally **(Only when ManualMode is selected)**
View Saved Preferences	Allows users to view what their previously saved preferences are and adjust them
Mode Toggle	Allows users to switch between manual and auto mode.

10.1.5. Timing Constraints

All inputs and outputs should occur almost instantly depending on internet speed.

10.1.6. Initialization

Displays user preferences and will allow setting and adjusting preferences through different buttons.

10.1.7. Exception Handling

Table 13: User interface exception handling

Input Name	Type	Exception	Exception Handling
I_Username	String	Username already exists	Alert user to pick new username

10.2. Intellux Database Module

10.2.1. Inputs And Outputs

Table 14: Database module inputs

Input Name	Type	Range	Units	Comments
I_Username	String	1-30 Characters	N/A	Stores name of User
I_Preferences	Int	0-100%	N/A	Stores percentage of desired maintained brightness per user
I_ManualAdjustment	Int	0-100%	N/A	Stores new blind state after incremental brightness adjustment
b_isAutoMode	Boolean	[0,1]	N/A	Stores Boolean value whether to run auto or manual mode

Table 15: Database module outputs

Output Name	Type	Range	Units	Comments
I_Preferences	Int	0-100%	Lux	Outputs percentage of desired maintained brightness per user
I_CurrentState	Int	0-100%	N/A	Output current state of blinds
b_isAutoMode	Boolean	[0,1]	N/A	Boolean output whether to run auto or manual mode

10.2.2. Description Of Behaviour

The user will have no direct interaction with the database. The database will retrieve all its data from the user input to the UI. This data is stored to maintain user preferences so that auto mode can function based on these saved preferences. System state is also saved in order for manual mode to function properly.

10.2.3. Database Development tools

Node Red has a built in SQLite module that we will be using for database development for Intellux.

10.2.4. Relational Database Schematic

The database will have one table that stores the users' usernames and preferences, blind state, and current mode with the username being the key ID.

10.2.5. Initialization

The database will have an empty table with the aforementioned schematic until data is input by users.

10.2.6. Exception Handling

Table 16: Database module exception handling

Input Name	Type	Exception	Exception Handling
I_Username	String	Username already exists	Alert user to pick new username

10.3. Intellux Sensors Module

10.3.1. Inputs And Outputs

Table 17: Sensor module inputs

Input Name	Type	Range	Units	Comments
m_brightnessIn	Lux	0-100,000	Lux	Analog Lux value of internal brightness
m_brightnessExt	Lux	0-100,000	Lux	Analog Lux value of external brightness

Table 18: Sensor module outputs

Output Name	Type	Range	Units	Comments
f_brightnessIn	Unsigned float	0-3.3	V	Float voltage of internal sensor reading
f_brightnessExt	Unsigned float	0-3.3	V	Float voltage of external sensor

				reading
b_sensorError	Boolean	[0,1]	N/A	Boolean value sent to Control Module to notify the user of sensor error.

10.3.2. Description Of Behaviour

The internal and external sensors will read Lux values based on the brightness levels. They will then provide analog voltage readings to the ADC between 0 and 3.3V, which will convert it into a digital value. This value will be read by the Raspberry Pi and then converted into an analog voltage value in code to be interpreted by the control algorithm. This is because the Raspberry Pi cannot interpret analog voltage values.

10.3.3. Sensors Setup

The sensors will be mounted on an adjustable part of the apparatus for users to ensure correct orientation. The setup configuration will be finalized in the final system design.

10.3.4. Sensors Circuit Diagram

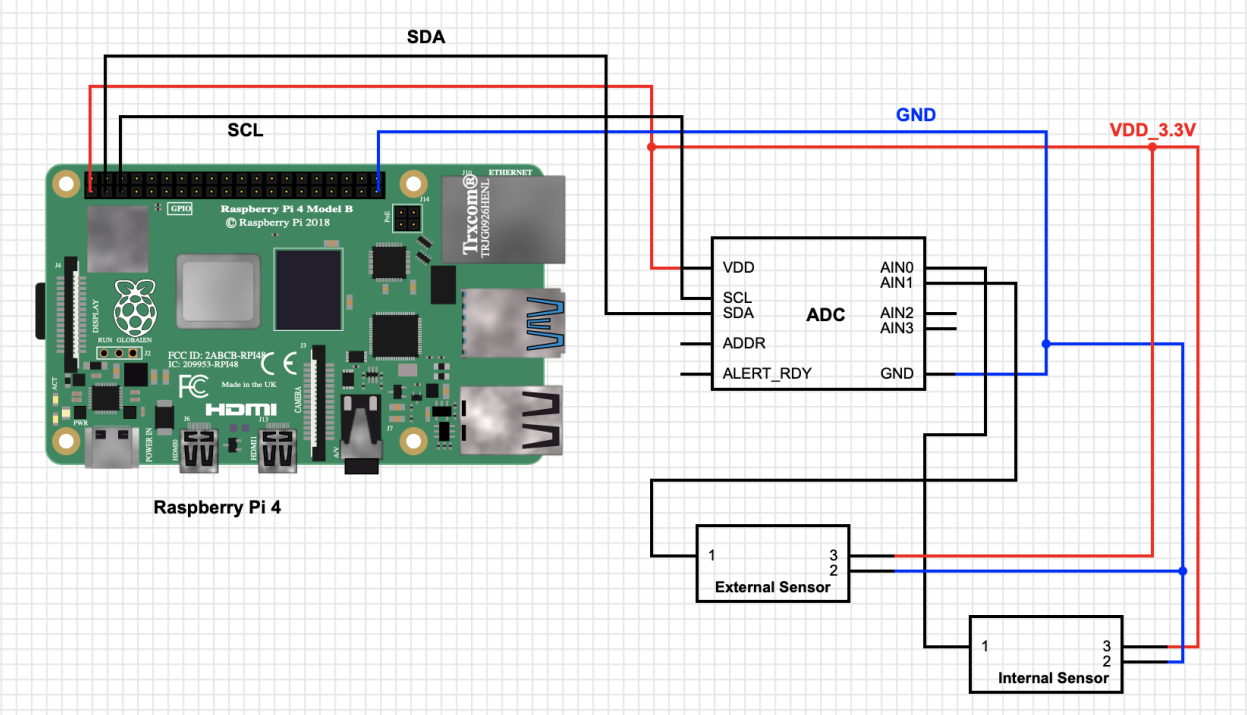

Figure 4: Sensor Module Circuit Diagram

10.3.5. Sensors Specifications

Table 19: Sensor specifications

Sensor Part Number	TEMT6000
Wavelength	570 nm
Viewing Angle	120º
Operating Temperature	-40ºC - 100ºC

10.3.6. ADC Specifications

Table 20: ADC specifications

ADC Part Number	ADS1115
Voltage Supply (Analog)	2V - 5.5V
Voltage Supply (Digital)	2V - 5.5V
Sampling Rate	860Hz
Data Interface	I2C
Operating Temperature	-40ºC - 125ºC

10.3.7. Timing Constraints

The timing requirement follows the SCL clock frequency f_SCL from the ADC which varies based on the operating mode:

Table 21: Sensor module ADC timing constraints

	Fast Mode		High-Speed Mode		Unit
	MIN	MAX	MIN	MAX	
f_SCL	0.01	0.4	0.01	3.4	MHz

10.3.8. Initialization

Upon initialization, this module will start reading brightness levels from the indoor and outdoor environments. These measurements will be interpreted by the ADC and sent to the controller to be used by the control software.

10.3.9. Exception Handling

Table 22: Sensor module exception handling

Input Name	Type	Exception	Exception Handling
m_brightnessIn	Lux	Sensor obstruction	Notify user that sensor is obstructed
m_brightnessExt	Lux	Sensor obstruction	Notify user that sensor is obstructed

The sensor obstruction will be detected through software and it will notify the user to check the obstruction for proper operation.

10.4. Intellux Control Module

10.4.1. Inputs And Outputs

Table 23: Control module inputs

Input Name	Type	Range	Units	Comments
f_brightnessIn	Unsigned float	0-3.3	V	Float voltage of internal lux sensor reading
f_brightnessExt	Unsigned float	0-3.3	V	Float voltage of external lux sensor reading
b_isAutoMode	Boolean	[0,1]	N/A	Boolean input that determines whether the control module should run in

				auto-mode or manual-mode
b_SetPoint	Unsigned float	0-3.3	V	Target internal lux sensor reading
I_blindsAngle	Unsigned int	0-180	degrees	- The target blinds angle based on the user's preference when operating in manual mode. - This will be fed to the mechanical module which will handle turning the blinds to the desired angle. **(Manual mode only)**
b_sensorError	Boolean	[0,1]	N/A	Boolean value sent to Control Module from Sensors module to notify the controller of any sensor errors.

Table 24: Control module outputs

Output Name	Type	Range	Units	Comments
b_isAchievable	bool	[0,1]	N/A	Feedback to the user interface to inform the user whether their target brightness level could be achievable or not
b_sensorError	Boolean	[0,1]	N/A	Boolean value sent to User interface Module from Control module to notify the user of any sensor errors.
targetStepperPos	Unsigned int	0 - # of steps needed to cover the full range of motion of the blinds which is 180 degrees	N/A	- The target stepper motor position based on the controller output. -The position of the stepper motor will be determined by keeping count of the number of steps taken. -The position 0 will be assigned when the blinds are fully closed at 0 degrees, and the position will be

				incremented with every CW step and decremented with every CCW step. The max stepper position will be assigned when the blinds are fully closed at 180 degrees -This will be fed to the mechanical module which will handle turing the stepper motor and in turn rotate the blinds to the desired angle

10.4.2. Description Of Behaviour

This module acts as the main controller that commands the stepper motor to turn the beads in the direction that rotates the blinds in a direction that will bring in the desired amount of light. This module communicates directly with three of the four other modules that make up the system. Therefore, it is the most integral module in the intellux system. The control module will run on a Raspberry Pi.

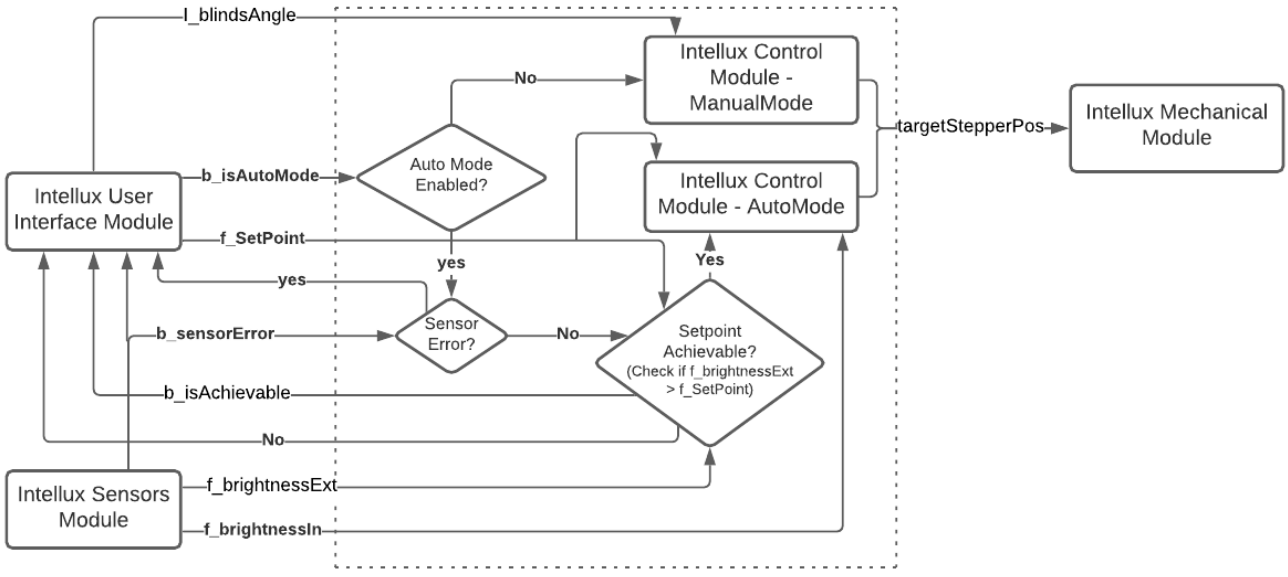

Figure 5: Intellux Control Module

The high level operation of the module can be seen in the figure above. The module begins by going through the following checks to determine the course of action that the controller should follow.

1. Is AutoMode enabled? This information is delivered through a boolean signal from the user interface module
    a. If the signal is TRUE, then the AutoMode controller is selected. The checks below will need to be processed before moving to the AutoMode controller
    b. If the signal is FALSE, then the ManualMode controller is selected. The target blinds angle is then taken from the user interface and passed onto the ManualMode controller. No further checks are required.
2. Are there any sensor errors? This information is delivered through a boolean signal from the sensors module
    a. If the signal is FALSE then the system can move on to the next check
    b. If the signal is TRUE, then the current process will be halted and the user will be notified through the user interface to avoid using AutoMode until the system is inspected.
3. Is the Setpoint achievable? This can be determined by comparing the external brightness levels with the target brightness levels. If the user demands an internal brightness level that is greater than the external brightness, then the target setpoint is not achievable
    a. If the setpoint is achievable, then the system can finally initiate the AutoMode controller and will receive the setpoint from the user interface module and the Internal brightness from the sensors module
    b. If the setpoint is not achievable, then the current process will be halted and the user will be notified through the user interface to change their preferences to a setting that is achievable.

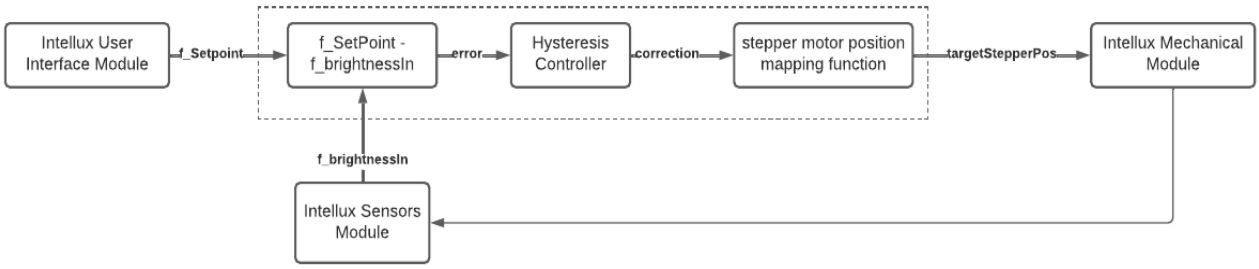

Figure 6: Intellux Control Module - AutoMode

The high level operation of the AutoMode controller can be seen in the figure above. This controller will mainly be composed of a Hysteresis controller to try and meet the timing and performance requirements set in the requirements specifications. The controller output will then be mapped to a specific stepper motor position that the mechanical module would have to adjust to. This new stepper motor position would in turn reflect a new blinds angle. After the adjustment, the sensors module would then gather new brightness readings, thus forming a feedback loop.

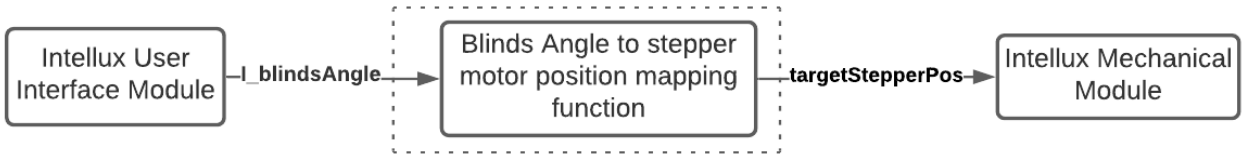

Figure 7: Intellux Control Module - ManualMode

The high level operation of the ManualMode controller can be seen in the figure above. This controller will be an open loop system which is composed of a mapping function that will map the input blinds angle to a specific stepper motor position. This mapping function will be determined in the future by inspecting the physical system through trial and error until we can approximate a linear relationship between the stepper motor position and the blinds angle.

10.4.3. Raspberry Pi Specification

Table 25: Raspberry Pi Specifications

Processor	Broadcom BCM2711, Quad core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
Memory	4GB
Power	Micro USB socket 5V1, 2.5A
GPIO	40 pin GPIO header also containing +3.3 V, +5 V and GND supply lines
Bluetooth	2.4 GHz and 5.0 GHz IEEE 802.11ac wireless, Bluetooth 5.0, BLE Gigabit Ethernet
Memory Card Slot	Push/pull Micro SDIO

10.4.4. Timing Constraints

The controller will have to be tuned in a way that allows the system to satisfy the following timing constraint: When a new desired brightness level is received from the Intellux App, the Intellux apparatus must be able to reach the desired brightness level within 6 seconds.

10.4.5. Initialization

At startup, the control module should load the current stepper motor position. The module would then wait to receive user preferences from the user interface to begin operating.

10.4.6. Exception Handling

If the intellux user decides to use artificial lights in their rooms while intellux is running in AutoMode, then we could get an event where the BrightnessIn > BrightnessExt. In that case, intellux mode should warn the user that other light sources might be interfering with the AutoMode operations and should revert back to Manual Mode until this exception is handled by the user.

10.5. Intellux Mechanical Module

10.5.1. Inputs And Outputs

Table 26: Mechanical module inputs

Input Name	Type	Range	Units	Comments
targetStepperPos	Unsigned int	0 - # of steps needed to cover the full range of motion of the blinds	N/A	- The target stepper motor position based on the controller output.

10.5.2. Description Of Behaviour

The mechanical module will be enclosed in a custom 3D-printed cover that secures all mechanical components in place for safe use. The user will only have access to the gear to install the beads and cover the gear back up once the beads are secured in place. The module will consist of a rectangular box that houses the sensor circuit, power circuit, Raspberry Pi and motor sink. Above the box will be the motor in its own small enclosure which is connected to the gear via the motor shaft. The module will have 2 openings on the side walls of the enclosure to house the gooseneck for the sensors. It will also have 2 openings on the front of the enclosure for the power sources of the Raspberry Pi and the power circuit.

10.5.3. Mechanical Setup Diagram

The gear will be placed at the bottom of the bead chain such that the bead loop goes around the gear as can be seen in Figure 8 (below):

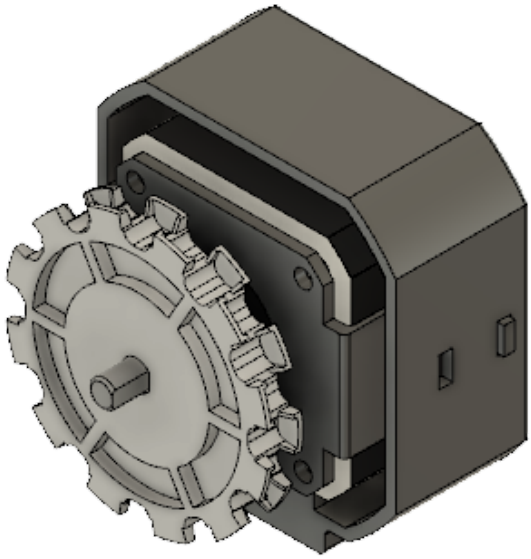

Figure 8: Mechanical Module Setup Diagram

10.5.4. Stepper Motor Circuit Diagram

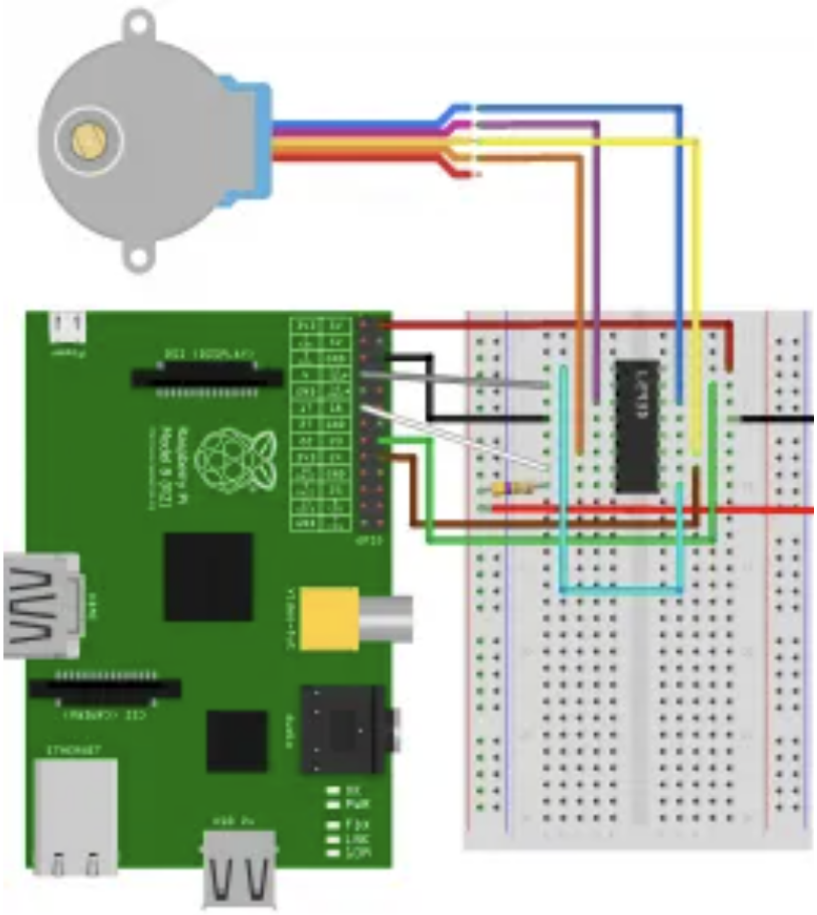

Figure 9: Mechanical Module Circuit Diagram

10.5.5. Stepper Motor Specifications

Table 27: Stepper motor specifications

Part Number	SM-17HS4023
Rated Voltage	DC12V 2-phase
Insulation Resistance	>10MΩ (500V)
Dielectric Strength	600V AC / 1mA / 1s
Step angle	5.625 x 1/64

DC Resistance	200Ω±7% (25C)
Reduction ratio	1/64
Insulation Grade	A
No-load Pull in Freq	>600Hz
No-load Pull out Freq	>1000Hz
Pull in Torque	>34.3mN.m(120Hz)
Detent Torque	>34.3mN.m
Temp Rise	40K(120Hz)

10.5.6. H-Bridge Specifications

Table 28: H-bridge specifications

Part Number	L298N
Motor Voltage	4.5V to 36V
Maximum Peak motor current	1.2A
Maximum Continuous Motor Current	600mA
Supply Voltage to Vcc1	4.5V to 7V
Transition time	300ns (at 5V & 24V)

10.5.7. Mechanical Gear Specification

Table 29: Mechanical gear specifications

Number of Teeth	9

Gear Diameter	12 mm
Width	8 mm
Tooth hole diameter	5 mm
Shaft Diameter	5 mm

10.5.8. Timing Constraints

The mechanical module must be able to reach any desired blinds angle within 6 seconds.

10.5.9. Initialization

Once the user installs the device on the wall they will be asked to calibrate the setup by firstly setting the blinds to a fully closed position (0 degrees) and pressing the *Calibrate* button on the user interface. The device will then start turning the blinds until it reaches 180 degrees where the user will have to click again on the *Calibrate* button to stop the process. This will allow the device to correlate the number of steps with the blind angle in order to function correctly.

11. Normal Operation

11.1. Description

When operating under normal conditions, Intellux constantly measures the brightness inside the living space and compares it to the desired brightness inputted by the user. The goal is to always keep the m_brightnessIn variable equal to the m_desiredBrightness variable with a 5% tolerance. Intellux also measures the brightness outside the living space to inform itself on how much it should turn the blinds in order to match the desired brightness with the indoor brightness.

11.2. Normal Use Cases

11.2.1. Auto Mode

During the daytime, Intellux constantly measures the brightness inside the room and compares it to the desired brightness fetched from the application. If the two values do not match then

Intellux will adjust the blind angles and repeat the process until the two values (m_brightnessIn & m_desiredBrightness) match within a certain tolerance.

- Cloudy Day:
    - Intellux should compare the desired brightness level inputted by the user to the level of brightness outside the living space. If the level of brightness outside the living space is lower than the one desired by the user, then Intellux opens the blinds at full brightness and notifies the user that the inputted level of brightness cannot be achieved.

11.2.2. Manual Mode

Intellux could be turned into manual mode through the Intellux phone application. In manual mode, Intellux stops measuring brightnesses and simply opens the blinds to the angle inputted by the user on the app.

- Night Time:
    - Intellux should be able to detect when the sun is down based on the external brightness. It will then turn into manual mode where the user can select if the blinds should be open or closed. As a default case, once night time is detected, Intellux will keep the blinds fully open unless instructed otherwise by the user.

12. Undesired Event Handling

12.1. Application failure to send preferences

If the Intellux app fails to send the user preferences to the Intellux device, it would prompt the user to re-pair the device and allow the user to control the blinds by hand.

12.2. Intellux apparatus not responding

If the Intellux device is not responding to the signals sent by the application, then the app will inform the user to make sure the device is paired and that the user is within the local network range. Intellux will let the user control the blinds by hand until the issue is resolved.

12.3. Unable to find/pair application to the device

If the Intellux app fails to detect the device's local IP network, the user can instead scan the QR code on the side of the device to automatically pair the apparatus to the application.

12.4. Beads slip out of the actuator

If Intellux is not able to detect the beads passing through the stepper motor then the application automatically notifies the user to reinstall the beads by hand.

12.5. Sensor Error

Should the sensor be faulty, the system will detect incorrect readings and notify the user to check the apparatus condition and whether they are being obstructed.

12.6. Power Surge

Hardware solutions are being explored to protect against power surges and outages such as fuses.

13. References

"ADS111x Ultra-Small, Low-Power, I2C-Compatible, 860-SPS, 16-Bit ADCs With Internal Reference, Oscillator, and Programmable Comparator." https://www.ti.com/lit/ds/symlink/ads1115.pdf?ts=1641317713134&ref_url=https%253A%252F%252Fwww.google.com%252F. Accessed 2 January 2022.

"Ambient Light Sensor." https://www.vishay.com/docs/81579/temt6000.pdf. Accessed 2 January 2022.

"L293x Quadruple Half-H Drivers datasheet (Rev. D)." *Texas Instruments*, https://www.ti.com/lit/ds/symlink/l293.pdf. Accessed 3 January 2022.

"Raspberry Pi 4 Model B specifications – Raspberry Pi." *RaspberryPi.com*, https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/. Accessed 2 January 2022.

"28BYJ-48 – 5V Stepper Motor." *Mouser Electronics*, https://www.mouser.com/datasheet/2/758/stepd-01-data-sheet-1143075.pdf. Accessed 2 January 2022.