МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет магістерської та
аспірантської підготовки
Кафедра інформаційних технологій

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему: РОЗРОБКА ТА ВСТАНОВЛЕННЯ СИСТЕМИ РОЗУМНИЙ БУДИНОК В РЕАЛЬНУ БУДІВЛЮ

Виконав студент 2 курсу групи МК-61
спеціальності 8.05010101 Інформаційні
управляючі системи та технології,
Гичак Станіслав Євгенійович

Керівник к.геогр.н., доцент,
Коваленко Людмила Борисівна

Рецензент
д.ф.-м.н., професор Ковальчук В.В.

Одеса 2017

АНОТАЦІЯ

Тема: "Розробка та встановлення системи розумний будинок в реальну будівлю"

Об'єм магістерської роботи 139 сторінок, на яких розміщено 27 рисунків, 2 таблиць, 5 додатків. При написанні магістерської роботи було використано 12 літературних джерел.

В даній магістерській роботі розглянуто принципи розробки системи "Розумний будинок", а також пристрої, які формують дану систему. В дану роботу входять вступ, 3 розділи та висновок. У вступі вказується задачі, актуальність даної роботи та наукова новизна. В першому розділі приводяться теоретичні відомості про систему "Розумний будинок", історія виникнення та її можливості. В другому розділі розповідається про вибрані автором даної роботи системи та програми, які застосовувались для її написання. Короткі теоретичні відомості про кожну з них. Третій розділ розповідає про написання самої системи. В кожному підрозділі вказано, як саме було написано програмне забезпечення під кожний елемент системи, а також приведені рисунки зі скріншотами середовищ розробки під час написання програм. Вказано деякі аспекти та особливості написання програм для кожного з процесорів та панель керування. У висновку зазначено, чи були виконані всі задачі, а також інформація про тестування самої системи.

Ключові слова: Розумний будинок, AMX, Lutron, Swift, NetLinx.

ANNOTATION

Theme: "Development and installation of smart home into a real building"

Master volume of 139 pages contain 27 illustrations, 2 tables, 5 additions. When writing a master's thesis has been used 12 literature.

In this master work the principles of development of "Smart house", as well as devices that form this system. In this work consists of introduction, three chapters and a conclusion. The introduction states the problem, the relevance of this work and scientific innovation. The first section provides information about the theoretical system of "Smart house", and the history of its capabilities. The second section describes the selected author of this work system and programs that were used to write it. Brief theoretical information about each of them. The third section talks about writing the system. Each division's how it was written software in each element of the system, and presented pictures with screenshots development environments when writing programs. Specified some aspects and features of writing programs for each processor and control panel. The conclusion states whether performed all tasks and information about testing the system.

Keywords: smart home, AMX, Lutron, Swift, NetLinx.

ЗМІСТ

ВСТУП

В теперішній час автоматизація систем побутових та домашніх електроприладів досягнула значного рівня розвитку. Автоматизація електроприладів пропонує нам керування різною технікою від найпростіших вмикачів до віддаленого керування всім приміщенням. Існують безліч платформ які служать для цих цілей, але лідируючими на ринку розумних домів є компанії Crestron, KNX та AMX. В даній роботі представлена система на технологіях компанії AMX. Для керування освітленням використана лідируюча компанія на ринку освітлення – Lutron. Інтерфейс користувача є невід'ємною, мало не найголовнішою, частиною системи автоматизації, найкраще з цим справляється компанія iRidium, вона представляє комплекс для візуалізації курування самими популярними системами автоматизації, зокрема системою AMX. За допомогою засобів автоматизації вирішується проблема надлишкового використання ресурсів. В основному використання електрики та водопостачання, тобто запрограмувавши сценарії, можна автоматично вимикати певні електроприлади, що значно економить ресурси. Автоматичне керування подачею води може застерегти від протікання та проривання труб водопостачання.

Метою магістерської роботи є розробка програмного забезпечення для автоматизованого керування магазином. Для цього нам потрібно було написати програму "сервер" для Розумного будинку, а також клієнт для керування ним.

Для досягнення основної мети роботи необхідно вирішити наступні завдання:

- провести аналіз існуючих систем автоматизації «Розумний будинок»;
- розробити систему керування освітленням за допомогою пристроїв Lutron;
- розробити систему керування приладами, які керуються ІЧ командами за допомогою пристроїв AMX;
- розробити систему керування поданням води та датчиків протікання за допомогою комплексу пристроїв Нептун та AMX;
- виконати вимоги керування з панелів, які працюють з ОС Windows та ОС IOS.

Дані задачі та проблеми є актуальними для нашого часу, адже все частіше людство турбують проблеми екології. А використання систем "Розумний дім", ми починаємо менше використовувати ресурси та навіть маємо можливість використовувати альтернативні джерела енергопостачання.

В ході роботи були досліджені системи "Розумного будинку". Також були використані системі, які є лідерами в економному енергопостачанні. Також з'явилися можливості економії електричної енергії, тобто використані меншої освітленості в денну пору, та під час відсутності клієнтів.

В Україні системи "Розумний будинок" з'явилися зовсім недавно, не багато компаній займаються встановленням та програмуванням таких систем. Також система є не дешевою, що не дозволяє її масового використання, але все частіше з'являються нові компанії, розробники програмного забезпечення та самих пристроїв для системі, які з часом коштують все менше і менше.

В даній роботі автором було розроблено програмне забезпечення, а також під його керуванням були встановлені всі проводові з'єднання та пристрої. Були протестовані всі пристрої на наявність правильних з'єднань та коректності працювання останніх.

Програмне забезпечення для керування системою було написане за допомогою мови програмування Swift. Воно дозволило значно розширити спектр можливостей, адже вона є розробкою компанії Apple та є створеною спеціально для пристроїв з встановленою операційною системою iOs.

В результаті проведеної роботи було помічено дійсне зменшення використання електричної енергії, а також дозволило ефективно виявляти затопленість з верхніх поверхів, що часто траплялося в минулому і можливий прорив водопостачання в ванній кімнаті.

Магістерська робота складається із 3 розділів, 27 рисунків, 2 таблиць, 5 додатків і 12 літературних джерел.

1 ОГЛЯД ІСНУЮЧИХ СИСТЕМ АВТОМАТИЗАЦІЇ "РОЗУМНИЙ БУДИНОК"

В даних, що наведені в [1] Розумний дім (розумний будинок/ smarthome, digitalhouse) – будинок, дача або приміщення комерційного призначення (бутік, офіс, будь-яка установа), які мають якісні системи забезпечення та операційний multiroom. За допомогою останнього, функціонально пов’язуються між собою усі електроприлади будівлі, якими можна керувати централізовано – з пульта-дисплею (рис. 1.1). Прилади можуть бути під'єднанні до комп’ютерної мережі, що дозволяє керувати ними за допомогою ПК та надає віддалений доступ до них через Інтернет. Завдяки інтеграції інформаційних технологій у домашні умови, усі системи та прилади узгоджують виконання функцій між собою, порівнюючи задані програми та зовнішні показники (обстановку).

Рисунок 1.1 – Мультимедійний дисплей
для керування розумним будинком

Для визначення високотехнологічних особливостей приміщення також вживають терміни: intelligentbuilding, smarthouse, digitalhome

Розумний дім створюється за допомогою професійного проектування та програмування компаніями, що займаються розробкою проектів smarthome. Програми, що вводяться до алгоритмів multiroom розумного дому, розраховані на певні потреби мешканців та ситуації, пов’язані із зміною середовища або безпекою. Особливістю smart-home є керування з пульта, на котрому людина може натиснути одну-єдину клавішу з метою створення певної обстановки. При цьому, сама система мультирум аналізує навколишню ситуацію та параметри усередині приміщення, та, керуючись власними висновками, виконує

задані користувачем команди із відповідними налаштуваннями. Окрім того, електронні побутові прилади, встановлені у розумному будинку, можуть бути об'єднані у домашню UniversalPlug'n'Play – мережу із виходом в Інтернет.

У розумному будинку все знаходить розум:

- Світло. Освітлення буде розуміти всі бажання господаря: по заданому господарем сценарієм включаються і вимикаються лампочки і світильники, у визначений час запалюється вечірнє освітлення, вночі будинок сам вимикає світло у всіх кімнатах ... З « розумним »світлом житло легко перетворити в надзвичайно затишний і казковий куточок, де так приємно проводити час. Одне натискання кнопки на панелі, і можна запустити будь-який світловий сценарій. Наприклад, «Романтична вечеря»: у кімнаті запалиться м'яке інтимне освітлення, спалахне камін, штори опустяться, заграє приємна музика ...
- Електроприлади. Будинок самостійно звертається до ваших електроприладів. Телевізор, праска, пральна і посудомийна машина, духова шафа, холодильник, електрообігрівач – всі електроприлади в будинку можуть підпорядковуватися одному натисненню клавіші на панелі управління.
- Клімат. Тепер вам зовсім байдуже, яка погода за вікном, бо погоду в будинку ви визначаєте самі! За допомогою системи контролю клімату ви зможете навіть прохолодною зимою відпочивати, як на Мальдівах, не встаючи зі свого дивана, а жарким літом – додати льодової прохолоди у ваш будинок. Система сама круглий рік підтримує в кімнатах комфортну температуру. Крім цього, повертаючись додому після роботи, ви можете надіслати з мобільного телефону завдання дому підігріти приміщення, наповнити ванну теплою водою, і все буде виконано прямо до вашого приїзду.
- Вентиляція. Розумний будинок завжди дихає свіжим і чистим повітрям, тому що в його системі передбачено управління вентиляцією . А це означає, що ви не знаєте, що таке духота і шкідливе повітря в кімнатах. А насолоджуєтеся свіжістю, чистотою і прохолодою свого житла.
- Домофон. Навіть домофон в «розумному» будинку стає набагато розумніше. Він фіксує всіх ваших відвідувачів, записує для вас повідомлення та фото гостей. Оповіщає про візіть вашої улюбленої мелодією, а відповісти на сигнал ви можете з будь-якого зручного місця в будинку.

- Мультирум. Перебуваючи в будь-якій кімнаті, ви за бажанням можете слухати улюблені пісні, проглядати відео, причому не важливо, де знаходиться джерело сигналу. А керувати всім цим можна зі зручною сенсорної панелі. І не треба захаращувати житло всілякими проводками, колонками, програвачами, вся апаратура компактно зібрана тільки в одному місці будинку.
- Полив. Навіть система поливу вашого саду та присадибної ділянки буде працювати по-розумному: сама визначати потрібну кількість вологи, час і необхідність поливу. Тому ваш сад завжди буде квітучим і запашним. Ще б пак, адже за ним доглядає такий «розумний садівник»!
- Басейн. «Розумний» басейн сам себе обслуговує, сам підігріває воду, самоочищається, наповнюється. Лише одного разу налаштувавши систему управління басейном на потрібний лад, ви можете більше не турбуватися про те, що потрібно вчасно набрати воду, підігріти її, очистити. Можна і значно заощадити на сервісі. Послуги спеціальних служб з очищення та ремонту басейну більше не знадобляться. Робота вашого басейну – під суворим контролем системи «Розумний Будинок».
- Відеоспостереження. Тепер ви можете контролювати ситуацію цілодобово, навіть перебуваючи поза домом! Система відео спостереження весь час напоготові. Ви можете мирно спати вночі, а кожна камера фіксуватиме будь-який рух в будинку і на прибудинковій території. Якщо на території хтось з'явиться, камери моментально зафіксують його присутність і повідомить про це вам.
- Безпека. Тепер ви будете спати тільки міцним сном, тому що у вас в будинку – найнадійніша система безпеки , датчики якої, як ваші особисті секьюріті, охороняють ваш будинок. Безпека абсолютно всіх приміщень будинку, а також прибудинкових територій – під найсуворішим контролем охоронної системи «Розумний Будинок». Тому ви завжди в курсі того, що відбувається у вашому будинку.
- Голосове управління. Ви хочете зробити своє життя в будинку ще простіше? Будь ласка! З системою голосового управління вам не потрібно, йдучи на роботу, оббігати всі вимикачі та електроприлади в будинку, щоб вимкнути їх. Досить буде скомандувати системі, і вона все виконати за вас і ще попередить вас про виконання завдання!

1.1 Історія створення

Розумні дома, як і більшість досягнень сучасної техніки, початково з’явилися на сторінках фантастичних оповідань. Але матеріалізовуватись ідея почала лише у XX-му сторіччі після широкого введення електрики у будівлях і розвитку інформаційних технологій. Перше повідомлення про віддалені прилади контролю можна віднести до розробки Ніколо Тесла дистанційного керування судами та транспортними засобами у 1898 році.

Електричні побутові прилади почали з’являтись між 1915 та 1920 рр. І одразу продемонстрували готовність суспільства замінити роботу домашнього персоналу дешевими механічними пристроями. Правда на той час, проблема енергозбереження при використанні нових технологій ще вирішена не була. Тому, певний час, новітні технологій були доступні лише дуже заможним людям.

Ідеї більш розвинені до понять сучасних систем автоматизації будинку були продемонстровані на ярмарках у Чикаго (1934) та Нью-Йорку. У “великому яблуці” трохи пізніше (1964-65), представили плани електрифікованих та автоматизованих приміщень. У решті-решт перший серйозний аналог розумного дому з’явився у 1966 році. Це була експериментальна система домашньої автоматизації – “домашній комп’ютер Exo IV”. Його винахідник – Джим Сазерленд, інженер компанії Westinghouse Electric. Його технологія була приватним, некомерційним проектом. Перші "дротові будинки" були зведені американськими винахідниками-любителями у 1960-х, але вони були суттєво обмежені можливостями тогочасних технологій.

Уперше термін "розумний будинок" був вигаданий Американською Асоціацією Housebuilders у 1984 році. Із винаходом мікроконтролерів, вартість на електроприлади швидко падала. Ця ж установа зазначила, що таке помешкання відмінне від звичайного своєю здатністю забезпечувати продуктивне та ефективне використання робочого та житлового середовища. За цим, віддалені інтелектуальні технології керування були прийняті будівельною промисловістю, яка поступово почала вводити їх не лише у бізнесових установах, але і у домашніх помешканнях. Під час активної домашньої автоматизації 90-х років інформатика та телевізійні системи були поєднані для підтримки інтелектуальних можливостей приміщень. У 1995 році винахідники технологій Java оголосили одним із основних призначень даної технології – “збільшення інтелекту побутових приладів”.

Сьогодні технології дозволяють збирати домашню автоматику покомпонентно: обирати лише ті функції розумного будинку, які дійсно потрібні користувачу. Тепер новітні технології керування приміщенням з'являються щодня. Навіть речі, котрі раніше розглядалися лише як красиві предмети інтер'єру тепер можуть виконувати ряд мультимедійних або побутових функцій.

Система "Розумний будинок" допомагає більш результативно використовувати комерційні приміщення, автоматизувати певні побутові процеси, урізноманітнити дозвілля. Попри те, що smart-home – дорога технологія, яка вимагає планування із самого початку зведення будинку та якісного устаткування, існують альтернативні рішення. Найпростіший за проектом дім можна доповнити певним прогресивним обладнанням, яке розширить функціональні можливості житлової площі та усучаснить приміщення .

Наприклад вже тепер, за допомогою технологій інтелектуального будинку, піч може повідомити хазяїв, коли вона потребує чистки. А коли холодильнику стане необхідний техогляд, він "скаже" про це. Сигналізація може одночасно подзвонити на номери служби безпеки та хазяїна будинку, якщо у домі з'явився незваний гість. За допомогою налаштувань мультирум, будинок може визначити, хто із членів родини переміщається по помешканню, і включити таке освітлення (температуру/ музику і т.д.), яке влаштовує саме цю людину.

Більш складні систему можуть вести облік продукції у комерційних закладах, облік її використання через зчитування штрих-кодів або RFID-тег. А у домашньому використанні: готувати список покупок.

Сучасні мобільні пристрої, забезпечені акселерометрами,мікрофонами, камерами, всілякими датчиками, здатними виміряти все і вся, можуть забезпечити потік даних, що однозначно і чітко описує все, що відбувається в навколишньому середовищі. І залишається тільки розробити досить складні і потужні універсальні програмні алгоритми, які здатні інтерпретувати цей потік даних, зробити висновки, прийняти відповідні рішення і виконати необхідні дії.

1.2 Особливості керування електроприладами системи

Що можна зробити, сидячи в кріслі?Здавалося б не так вже й багато. Ну, приміром, почитати книгу, подивитися фільм, випити чашечку кави, поговорити по телефону. Але є щасливчики, які можуть включати або вимикати будь побутовий прилад у будинку, не встаючи з улюбленого крісла. Хто вони, лю-

ди, що володіють екстрасенсорними здібностями? Ні, це щасливі власники системи «Розумний Будинок», яка сама керує електроприладами в будинку.

Телевізор, праска, пральна і посудомийна машина, духова шафа, холодильник, електрообігрівач – всі електроприлади в будинку можуть підпорядковуватися одному натисненню кнопки на панелі управління. Для цього всі розетки в будинку пов'язані в одну єдину мережу і підкоряються системі «Розумний Будинок», яка легко управляє цілою групою електроприладів і кожним приладом окремо. Одне натискання кнопки на зручній та зрозумілій панелі, і ви включили або вимкнули потрібний вам прилад або групу приладів, налаштували їх безперебійну та безпечну роботу. Це позбавляє Вас від щоденного метання від приладу до приладу для включення і виключення, або щоб дізнатися: чи все нормально працює? Крім того, електроприладами можна управляти і на відстані, наприклад, через мобільний телефон. Якщо ви забули вимкнули плиту, не варто хвилюватися. Просто вимкніть її через мобільний телефон, і все!

В системі "Розумний будинок" можна запрограмувати систему на включення або виключення будь-якого приладу в певний час. Вранці в точно заданий час будуть включатися чайник, кавоварка, пароварка. Дуже зручно ввечері закласти в пароварку необхідні продукти, вранці вона сама включиться, а через заданий час вимкнеться. Прокинувшись і пройшовши на кухню, ви отримаєте вже готовий гарячий сніданок і чашечку кави. Або система може сама включити електрообігрівач за годину до вашого приходу додому, щоб у будинку до вашого приходу був комфорт, тепло і затишок.

Якщо ви, наприклад, не хочете, щоб дитина дивилася телевізор після 21.00, ви можете налаштувати систему таким чином, що вона буде вимикати розетку, в яку включений телевізор, о 21.00. І ви буде спокійні за свою дитину.

Коли ви йдете з дому, система «Розумний Будинок» завбачливо знеструмлює розетки. Вам потрібно всього лише натиснути на панелі управління одну кнопку «Вимкнути всі прилади». І не потрібно бігати перед відходом по всьому будинку, гарячково згадуючи, а чи не забули ви щось вимкнути. Або ви можете заздалегідь налаштувати систему на автоматичне знеструмлення розеток в певний час. Наприклад, щоночі в 24.00 в усьому домі автоматично будуть знеструмлюватися розетки, вимикаючи електроприлади (крім деяких, наприклад, холодильника), а вранці система сама буде їх знову включати в точно заданий час. Система сама буде вчасно відключати невживані прилади або контролювати навантаження електромережі. Таким чином, ви будете не тільки відчувати себе в безпеці, але ще й економити електроенергію.

Будинок, який сам управляє своїм освітленням: по заданому господарем сценарієм включаються і вимикаються лампочки і світильники, у визначений час запалюється вечірнє освітлення, вночі будинок сам вимикає світло у всіх кімнатах ... Це вже не фантастика чи мрії, це реальність. Тому що є система «Розумний Будинок», яка розумно керує всіма інженерними системами будинку, і в першу чергу освітленням.

А тепер розглянемо на прикладі сценарії керування освітленням, тобто зануримося в атмосферу домашнього затишку з системою освітлення "розумного будинку".

У «розумному» будинку настає ранок (саме в той час, який ви самі визначите). Над узголів'ям ліжка, на якому солодко сплять господарі, поступово включається підсвічування, повідомляючи їм про настання нового дня. Штори або жалюзі, як за помахом чарівної палички, повільно відкриваються, впускаючи в кімнату денне світло. Господарі прокидаються і йдуть у ванну. І тут не треба намацувати на стіні вимикач світла, тому що є датчик руху. Він визначить присутність господарів, і світло запалиться сам. А коли господарі покинуть ванну, світло самостійно згасне.

Управління світлом – це одне з найголовніших переваг системи «Розумний Будинок» і одна з самих затребуваних її функцій. Бо саме світло ми використовуємо переважно у будинку і саме він є невід'ємною частиною інтер'єру. Налаштовувати освітлення та управляти ним за системою «Розумний Будинок» дуже просто. Тут немає складних, незрозумілих інструкцій. Одним натисканням кнопки можна включити або виключити, відрегулювати будь-яке джерело світла в будинку. При цьому господарі можуть перебувати в будь-якій кімнаті будинку або навіть в іншому місті або країні. Тому що управляти цією системою можна з мобільного телефону або за допомогою Інтернет! А, задавши певну настройку, світло може включатися, навіть реагуючи на голос або хлопком долонь.

Настав час бігти на роботу, їхати по справах або на важливу зустріч. Потрібно йти, а світло горить в декількох кімнатах, немає необхідності оббігати весь будинок, щоб вимкнути його. Одним натисканням кнопки на панелі, яка розташована в самому зручному для господарів місці або в декількох місцях, відключається весь світ одночасно.

Єдина панель управління світлом замінює всі вимикачі світла в будинку. Вона також забезпечує доступ до висвітлення у всіх приміщеннях будинку (гараж, коридор, комора тощо), перебуваючи в одній кімнаті можна керувати світлом у всіх інших приміщеннях будинку і в кожному з них окремо. Велику

роль відіграє і димер (світлорегулятор). З його допомогою можна управляти яскравістю кожної лампочки: поступове запалювання або потухання, поєднання декількох світлових фонів – можна підсвітити один кут кімнати, а інший заховати в напівтемряві, зробивши інтер'єр кімнати набагато цікавіше. Додатковий комфорт дає функція «Прив'язка вимикачів», коли кожен вимикач буде відповідати за той освітлювальний прилад (люстра, бра, система підсвічування і т.д.), який ви йому самі призначите.

Опівдні в будинку нікого немає, будинок живе своїм життям. Господарі відсутні, але вони не хвилюються, що з їхнім будинком може щось трапитися. Освітлення працює таким чином, що створюється враження, що мешканці знаходяться вдома.

Імітація присутності господарів – ви самі можете налаштувати систему так, щоб, коли вас немає вдома, вона управляла світлом, створюючи повну імітацію присутності господарів. Це особливо зручно, коли ви у відрядженні, відпустці або в тривалому від'їзду.

Як і кожен вечір в певний час на ганку будинку включився світло. До будинку під'їхала машина господарів. Залишивши машину в гаражі, вони йдуть за провідною до ганку доріжці. Реагуючи на їх присутність, автоматично включається підсвічування доріжки, прокладаючи їм шлях до будинку, і повільно згасає позаду них. У коридорі включився м'яке світло, гостинно зустрічаючи мешканців. Господарі приїхали додому в компанії друзів. Саме час влаштувати спільний перегляд фільму. Гості сідають зручно перед телевізором, запускається спеціальне освітлення для перегляду кіно: світло поступово гасне, а коли фільм закінчується, поступово запалюється. Після кіносеансу компанія переміщається у вітальню до бару. Одним натисканням кнопки на панелі системи господар включає спеціальне освітлення «Вечірка»: приглушене світло, різнокольорові вогні, як на танцполі. Це незабутній затишний вечір в «розумному» будинку.

Освітлення працює на вашу сценарієм. Управляти системою освітлення можна за допомогою заздалегідь заданих світлових сценаріїв, які зберігаються в пам'яті системи «Розумного будинку». Наприклад, сценарій «Вечір» буде керувати освітленням будинку всередині і зовні у вечірній час: включати світло на ганку, в гаражі, у дворі, на прибудинковій території. Налаштувавши ландшафтне освітлення, ви зможете створити по-справжньому казкову атмосферу вечора у себе у дворі або в саду. Ви можете самі задавати будь-які сценарії освітлення залежно від ваших бажань і потреб: «Читання книг», «Романтична вечеря» та інші сценарії за побажанням. Наприклад, «Аварійне освітлен-

ня»: у разі перебоїв з електроенергією або відсутності напруги освітлення в будинку буде підтримуватися завдяки акумуляторних батарей. Ви не будете захоплені зненацька і не залишитеся в темряві.

Коли в «розумному» будинку настає ніч, освітлення переходить в сплячий режим. Скрізь гасне світло. За бажанням господарів м'яка приглушена підсвічування в кімнатах буде включатися тільки тоді, коли хтось з родини прокинеться і захоче пройти на кухню чи в іншу кімнату.

Система освітлення «Розумний Будинок» істотно скорочує витрати на електроенергію, відключаючи непотрібне навантаження або переводячи в режим низького споживання енергії. Працюють тільки ті світлові прилади, які необхідні в даний момент. Решта відключені, їх включення і відключення розумно контролюється. Таким чином, встановивши систему освітлення «Розумний Будинок», ви будете постійно економити кошти і насолоджуватися комфортним життям у своєму «розумному» будинку.

Універсальний пульт. Тепер вам не потрібно в купі всіляких пультів від домашньої техніки шукати необхідний, забуваючи, який пульт від якого приладу, тому що всю цю гору замінює один – універсальний, «розумний» пульт. Такий пульт з легкістю впорається з керуванням будь-якого побутового приладу в будинку, словом, всього, що включається в розетку. Одним натисненням кнопки ви запустите або відкоригуєте роботу будь-якої техніки в будинку. «Розумний» пульт стане незамінним домашнім помічником для господарів.

Панель керування системою "Розумний будинок" можна повісити на стіну, а можна переносити з кімнати в кімнату. Панелі встановлюються в одному або декількох найзручніших місцях в будинку. Можна помістити панель, наприклад, поруч із вхідними дверима. Тоді, входячи в будинок, ви відразу можете натиснути на панелі одну кнопку, яка запускає сценарій «Господарі будинку»: налаштовує телевізор на необхідні канали, включає улюблену музику, готує гарячу каву. Загалом, можливі будь-які дії на ваш розсуд. Або ви можете переносити панель в те місце, де вона необхідна.

Легким дотиком до екрану панелі господар вибирає прилад, роботу якого потрібно відрегулювати, і задає необхідні настройки. При натисканні на кнопку сигнал по wi-fi передається на інфрачервоний передавач, який в свою чергу надсилає його потрібної техніки в будинку. Техніка отримує сигнал, і починає працювати саме так, як цього побажав господар. Величезний плюс, що не потрібно спеціально спрямовувати пульт на необхідний прилад, щоб включити або вимкнути його, це можна зробити, перебуваючи в будь-якій кімнаті будинку.

Тепер вам зрозуміло, що завдяки такій «чарівної палички», як панель керування, можна контролювати будь-які дії і функції кожного електроприладу в будинку одним дотиком пальця до потрібної кнопки на панелі. Заплутатися в її інструкції неможливо, тому що вона має російське меню і зручний зрозумілий інтерфейс. Без універсального пульта, дійсно, як без рук. Саме він робить життя в будинку по-справжньому зручною і комфортною. Господар оточений затишком і відчуває себе в повній безпеці.

1.3 Принципи дії охоронної системи «Розумний будинок»

З охоронною системою «Розумний Будинок» господар по-справжньому може бути спокійний за свою сім'ю, будинок і майно. Ви будете почувати себе в цілковитій безпеці, а ночами спати міцно і спокійно не переживаючи, що хтось може порушити вашу мирну, затишну атмосферу будинку.

Величезну роль у створенні спокійної і безпечної атмосфери в будинку грають всілякі датчики, які забезпечують безпеку як всередині будинку, так і зовні. Без них неможливо уявити охоронну систему будинку та території поруч з ним.

Датчики відкриття дверей і вікон. Завдання цих датчиків повідомляти на центральну панель про будь-яких спробах проникнення через двері та вікна. При фіксуванні таких вторгнень датчик передає звуковий, світловий або мовний сигнал, і оповіщає господаря. Світлові або мовні сигнали відлякують порушників, у них не буде ні найменшого бажання випробовувати долю.

Датчики руху. Всередині будинку, а також по периметру житла, обов'язково встановлюються датчики руху. З їх допомогою охоронна система «Розумний Будинок» запобігає проникнення на територію будинку. Якщо датчик спрацює, господар відразу ж про це дізнається. Наприклад, якщо непрошений гість зробить спробу підійти до вашого паркану або перелізти через нього, відразу загориться яскраве світло, щоб якомога краще висвітлити порушника, та щоб камери стеження чітко його зняли. Порушник відразу зрозуміє, що з такою охороною жарти погані. Це геть відіб'є в нього бажання порушувати ваш спокій і безпеку.

Вигідно, що крім своїх датчиків система охорони використовує і всі ті, які вже встановлені у вас в будинку і на території. Приміром, якщо у дворі у вас є датчики руху для підсвічування доріжки, то система охорони цілком успішно задіє для своїх цілей і їх.

Усі сигнали, які передають датчики, в тому числі і запис з камер відеоспостереження, в обов'язковому порядку вносяться в пам'ять системи, щоб господар завжди міг переглянути, коли і з якого датчика був переданий сигнал. Вести такий контроль можна навіть через мобільний телефон або комп'ютер, в будь-який зручний для господаря момент і в будь-якому зручному місці.

Система охорони «Розумний Будинок» дуже тісно взаємодіє з системою відео спостереження , виконуючи одну загальну функцію – захист і збереження безпеки вашого будинку. Завдяки відео спостереженню у охоронної системи завжди є повна інформація про всіх суб'єктів, що потрапляють на територію будинку, а охорона інформує систему відео спостереження про будь найменших рухах в будинку і поруч з ним, щоб камери могли зняти потрібні моменти.

Також датчики можуть самі контролювати своє включення і виключення. Наприклад, ви йдете з дому, система повідомляє вам, що в якийсь із кімнат відкрито вікно, ви вирішили не закривати його. У такому випадку датчик відкриття цього вікна відключиться, а датчики руху в цій кімнаті будуть переведені в режим зниженої чутливості, щоб не було помилкових спрацьовувань через коливання штор. Кожному датчику можна задати свій рівень чутливості, щоб він не реагував, наприклад, на найменші коливання або на домашніх тварин.

Вірний друг і захисник – система охорони «Розумний Будинок» – попередить свого господаря про наближення небажаних гостей до воріт або забору будинку. Для цього встановлюється лазерна або ємнісна система охорони периметра житла. І при будь-якому наближенні непрошених відвідувачів система буде реагувати так, як ви цього захочете: наприклад, буде різко включатися світло або лунати звуковий сигнал. Це відлякує зловмисників.

Також, за вашим бажанням на будь-який монітор в будинку буде виводитися відео фрагмент спрацьовування того чи іншого датчика. Дуже зручна функція перегляду активних зон. На панелі відображається детальна карта всіх приміщень будинку, і на ній різними кольорами підсвічуються так звані активні зони. Тобто, ті місця, в яких в останній відрізок часу (1, 5, 10 хвилин) були якісь рухи та ін. Колір зони залежить від часу, коли в ній відбувалося рух. Наприклад, вам здалося, що внизу був якийсь шум, ви можете швидко подивитися активні зони, і дізнатися, що ж там сталося.

Навіть якщо до вас в будинок проник підозрілий суб'єкт, ви завжди можете подати сигнал тривоги, всього лише натиснувши одну спеціальну кнопку, розташовану на кожній панелі в будинку. Кнопка передає тривожний сигнал на центральний пульт охорони і викликає допомогу. При цьому ваш «не-

званий гість» навіть не запідозрить, що ви вже повідомили про нього в відповідну інстанцію.

Читаючи про датчики руху та інші «навороти» охоронної системи «розумного» будинку , ви цілком можете задатися питанням, як же система розуміє, на кого реагувати, а на кого – ні? Адже ви теж можете пересуватися по будинку вночі, наприклад, за склянкою води або чашкою чаю. Хіба в такому випадку не спрацює сигналізація? Виявляється, немає. Адже охоронна система на то і «розумна» охоронна система, щоб «знати» своїх і «відлякувати» чужих. Наприклад, якщо мешканці будинку мирно сплять у себе в спальнях, а в цей момент в будинок з вулиці проникне зловмисник, система моментально спрацює і вживатиме всі необхідні заходи для його знешкодження. А якщо хтось із членів сім'ї спуститься зі своєї спальні вниз на кухню, то система «дізнається» господаря і ніякого реагування не буде.

2 ОБГРУНТУВАННЯ ВИБОРУ ВИКОРИСТАНИХ СИСТЕМ, КОМПЛЕКСІВ ТА ПРОГРАМ

2.1 Програма для створення цифрового плану будівлі КОМПАС-3D

«КО́МПАС»(КОМПлекс Автоматизованих Систем конструкторсько-технологічного проектування) – система автоматизованого проектування та підготовки до виробництва російської фірми «Аскон» [2].

СистемаКОМПАС-3D – інтерактивний графічний, з сучасним інтерфейсом, оснащений інструментальними засобами, які дозволяють створювати твердотілі об'єкти з використанням набору елементарних параметричних тіл (паралелепіпед,циліндр та ін.).

Просторові твердотілі та каркасні моделі об'єктів (деталей, вузлів, виробів, будівель і т. д.) при виконанні проектно-конструкторських, технологічних та дизайнерських робіт вмашинобудуванні, приладобудуванні, будівництві, архітектурі).

Під час вибору програм для створення цифрового плану будівлі були розглянуті різні варіанти, але саме КОМПАС-3D зі зручним та простим у використанні інтерфейсом та дійсно широким набором інструменті дозволив зробити це досить якісно, а також зручно редагувати створений план.

2.2 Середовище розробки Xcode

Xcode – інтегроване середовище розробки програмного забезпечення під macOS і iOS, розроблена корпорацією Apple. Перша версія випущена в 2001 році. Стабільні версії поширюються безкоштовно через MacAppStore. Зареєстровані розробники також мають доступ до бета-збірок через сайт AppleDeveloper

Xcode включає в себе більшу частину документації розробника від Apple і InterfaceBuilder – додаток, що використовується для створення графічних інтерфейсів.

Пакет Xcode включає в себе змінену версію вільного набору компіляторів GNUCompilerCollection і підтримує мови C, C ++, Objective-C, Objective-C ++, Swift, Java, AppleScript, Python і Ruby з різними моделями програмування, включаючи (але не обмежуючись) Cocoa, Carbon і Java. Сторонніми розробниками реалізована підтримка GNUPascal, FreePascal, Ada, C#, Perl, Haskell і D. Пакет Xcode використовує GDB як back-end'a для свого відладчика.

Xcode включає в себе все, що потрібно розробнику для створення програм для Mac, iPhone, IPAD, AppleTV, і AppleWatch. Xcode надає розробникам єдиний робочий процес для проектування користувальницького інтерфейсу, кодування, тестування і налагодження. XcodeIDE в поєднанні з мовою програмування Swift роблять розробку додатків простіше і веселіше, ніж колинебудь раніше.

Xcode включає в себе XcodeIDE, компілятор Swift та Objective-C, інструменти інструмент аналізу, імітатори, останні SDKs і сотні потужних функцій.

Інноваційні інструменти допоможуть Вам простіше створювати програми:

- Swift – революційна мова програмування, яка є безпечною, швидкою і сучасною;
- Playgrounds являє собою цікавий спосіб, щоб експериментувати і взаємодіяти з Swift кодом;
- InterfaceBuilder відображає з точністю до пікселя UI для кожного цільового пристрою і може допоможе адаптувати додаток для будь-якого масштабу;
- Viewdebugging показує 3D стек зору UI, всі верстви вашого додатку під час виконання.

InterfaceBuilder дозволяє легко створити свій інтерфейс без коду

- Розкадрування дозволяє організувати повний потік екранів в вашому додатку;
- Налаштування інтерфейсу для різних пристроїв, розмірів екрану і орієнтації;
- StackViews дозволяють легко відстежувати розташування кожного розділу інтерфейсу;
- Створення з'єднання вашого графічного дизайну безпосередньо до вихідного коду.

Інструменти аналізу продуктивності

- Порівняйте завантаженість центрального процесору, диску, пам'яті і продуктивністі OpenGL у вигляді графічних треків з плином часу;
- Визначити вузькі місця в продуктивності, а потім зануритися в код, щоб розкрити причину.

З 9 березня 2011 стала доступна нова версія Xcode 4, причому в нову версію входять набори SDK OS X 10.6 і SDK iOS 4.3. Вперше попередній реліз Xcode 4 був показаний на WWDC 2010 року.

У Xcode 4 був представлений новий призначений для користувача інтерфейс: єдине вікно, куди вбудований і Interface Builder, сам Xcode, Instruments, помічник, нова система аналізу коду Debug Console з більш сильним двигуном, що полегшує виправлення помилок і здійснює пошук логічних проблем в коді. Програма також отримала додатковий компілятор Apple LLVM.

З 12 вересня 2013 року стала доступна нова версія Xcode 5.0. У ній змінився інтерфейс і всі доопрацювання призначені для розробки ПО з урахуванням особливостей нових версій ОС: iOS 7 і OS X 10.9 Mavericks.

З 3 червня 2014 року стала доступна бета-версія нового Xcode 6. У неї внесені доопрацювання, призначені для розробки ПО з використанням нової мови програмування Swift, і з урахуванням особливостей 4000 нових API (наприклад, програмних інтерфейсів HealthKit, HomeKit і Metal) для версій ОС: iOS 8 і OS X 10.10.

З 8 червня 2015 року стала доступна бета-версія нового Xcode 7.0. У ній з'явилася підтримка нової версії мови Swift, був вдосконалений Interface Builder.

З 12 вересня 2016 року стала доступна версія нового Xcode 8. У ній з'явилася підтримка iOS 10, Swift 3.0, оновлений і розширений дебагер.

iOS (до 24 червня 2010 року – iPhone OS) – операційна система для смартфонів, електронних планшетів і кишенькових програвачів, що розробляється і випускається американською компанією Apple. Була випущена в 2007 році; спочатку – для iPhone і iPod touch, пізніше – для таких пристроїв, як iPad і Apple TV. На відміну від Windows Phone (Microsoft) і Android (Google), випускається тільки для пристроїв, вироблених фірмою Apple.

У iOS використовується ядро XNU, засноване на мікроядрі Mach і містить програмний код, розроблений компанією Apple, а також код з ОС NeXTSTEP і FreeBSD. Ядро iOS майже ідентично ядру настільної операційної системи Apple macOS (раніше називалася OS X). Починаючи з найпершої версії, iOS працює тільки на планшетних комп'ютерах і смартфонах з процесорами архітектури ARM.

Інтерфейс iOS заснований на концепції прямої взаємодії з використанням жестів «мультитач». Елементи управління інтерфейсом складаються з повзунків, перемикачів і кнопок.

iOS розроблена на основі операційної системи OS X (перейменованої в macOS) і використовує той же набір основних компонентів Darwin, сумісний зі стандартом POSIX.

Операційна система iPhone OS була представлена 9 січня 2007 року спільно з мобільним телефоном iPhone особисто Стівом Джобсом на виставці-конференції Macworld Conference & Expo і випущена в червні того ж року. Apple не припускала окремої назви для операційної системи, тому початковий слоган звучав так: «iPhone працює на OS X».

2.3 Середовище розробки Microsoft Visual Studio

Microsoft Visual Studio – лінійка продуктів компанії Microsoft, що включають інтегроване середовище розробки програмного забезпечення і ряд інших інструментальних засобів [6]. Дані продукти дозволяють розробляти як консольні додатки, так і додатки з графічним інтерфейсом, в тому числі з підтримкою технології Windows Forms, а також веб-сайти, веб-додатки, веб-служби як в рідному, так і в керованому кодах для всіх платформ, підтримуваних Windows, Windows Mobile, Windows CE, .NET Framework, Xbox, Windows Phone .NET Compact Framework і Silverlight.

Visual Studio включає в себе редактор вихідного коду з підтримкою технології IntelliSense і можливістю найпростішого рефакторінга коду. Вбудований відладчик може працювати як відладчик рівня вихідного коду, так і як відладчик машинного рівня. Решта вбудовуються інструменти включають в себе редактор форм для спрощення створення графічного інтерфейсу додатку, веб-редактор, дизайнер класів і дизайнер схеми бази даних. Visual Studio дозволяє створювати і підключати сторонні додатки (плагіни) для розширення функціональності практично на кожному рівні, включаючи додавання підтримки систем контролю версій вихідного коду (як, наприклад, Subversion і Visual SourceSafe), додавання нових наборів інструментів (наприклад, для редагування і візуального проектування коду на об'єктно-орієнтованих мовах програмування або інструментів для інших аспектів процесу розробки програмного забезпечення (наприклад, клієнт Team Explorer для роботи з Team Foundation Server).

Visual Studio включає один або декілька компонентів з наступних:

– Visual Basic .NET, а до його появи – Visual Basic;
– Visual C ++;
– Visual C #;
– Visual F # (включений починаючи з Visual Studio 2010).

Багато варіантів поставок також включають Microsoft SQL Server або Microsoft SQL Server Express.

У минулому до складу Visual Studio також входили продукти:

– Visual InterDev;
– Visual J ++;
– Visual J #;
– Visual FoxPro;
– Visual Source Safe – файл-серверна система керування версіями.

До виходу Visual Studio Version 4.0 середовища розробки Visual Basic 3, Visual C ++, Visual FoxPro і Source Safe поставлялися в якості самостійних пакетів.

Далі розглянемо версії Visual Studio (табл. 2.1), їх особливості та відмінності кожної з них.

Visual Studio 97 – перша випущена версія Visual Studio, в якій вперше були зібрані разом різні засоби розробки ПЗ. Вона була випущена в двох версіях – Professional і Enterprise, і включала в себе Visual Basic 5.0, Visual C ++ 5.0, Visual J ++ 1.1, Visual FoxPro 5.0 і вперше з'явилася середовище розробки ASP – Visual InterDev. Visual Studio 97 була першою спробою Microsoft створити єдине середовище для розробки на різних мовах програмування: Visual C ++, Visual J ++, Visual InterDev і MSDN використовували одну середу, звану Developer Studio. Visual Basic і Visual FoxPro використовували окремі середовища для розробки.

Visual Studio 6.0 – остання версія Visual Studio, що працює на платформі Windows 9x (випущена в червні 1998 року). Як і раніше популярна серед програмістів, що використовують Visual Basic. Дана версія була основною середовищем розробки додатків під Windows від Microsoft до появи платформи .NET.

Visual Studio .NET (кодове ім'я Rainier; внутрішня версія 7.0) – випущена в лютому 2002 року (включає .NET Framework 1.0). Service Pack 1 для Visual Studio .NET (2002) випущений в березні 2002.

Visual Studio .NET 2003 (кодове ім'я Everett; внутрішня версія 7.1) – випущена в квітні 2003 року (включає .NET Framework 1.1). У квітні 2005 року Microsoft оголосила про спеціальному випуску середовища, що отримала назву Microsoft Visual Studio .NET 2003 Professional Special Edition. Спецвипуск представляв собою звичайне видання Visual Studio .NET 2003 Professional Edition з додаванням в комплект серверного ПЗ та інших інструментів (зокрема, операційної системи Windows Server 2003 Standard Edition і SQL Server 2000 Developer Edition). Для заохочення переходу на нове середовище розробки корпорацією була оголошена спеціальна ціна оновлення, що діяла при пе-

реході на Visual Studio .NET 2003 Professional Special Edition з великого числа засобів розробки як Microsoft, так і основних його конкурентів.

Таблиця 2.1 – Версії Visual Studio

Офіційна назва	Кодова назва	Внутрішня версія	Версія .NET Framework	Дата виходу
Visual Studio	N/A	4.0	N/A	Весна 1995
Visual Studio 97	Boston	5.0	N/A	1997
Visual Studio 6.0	Aspen	6.0	N/A	1998-06
Visual Studio .NET (2002)	Rainier	7.0	1.0	2002-02-13
Visual Studio .NET 2003	Everett	7.1	1.1	2003-04-24
Visual Studio 2005	Whidbey	8.0	2.0	2005-11-07
Visual Studio 2008	Orcas	9.0	3.5	2007-11-19
Visual Studio 2010	Dev10/Rosario	10.0	4.0	2010-04-12
Visual Studio 11 Beta		11.0	4.5	2012-03-01
Visual Studio 2012	Dev11	11.0	4.5	2012-08-15
Visual Studio 2013	Dev12	12.0	4.5.1	2013-10-17
Visual Studio 2015 Preview		14.0	4.6	2014-11-21

Service Pack 1 для Visual Studio .NET 2003 випущений 13 вересня 2006.

Visual Studio 2005 (кодове ім'я Whidbey; внутрішня версія 8.0) – випущена наприкінці жовтня 2005 року (включає .NET Framework 2.0). Остання офіційно працює на Windows 2000. На початку листопада 2005 також вийшла серія продуктів в редакції Express: Visual C ++ 2005 Express, Visual Basic 2005 Express, Visual C # 2005 Express та ін. 19 квітня 2006 редакція Express стала безкоштовною. Service Pack 1 для VS2005 і всіх Express-редакцій випущений 14 грудня 2006. Додатковий патч для SP1, що вирішує проблему сумісності з Windows Vista, випущений 6 березня 2007.

Visual Studio 2008 (кодове ім'я Orcas; внутрішня версія 9.0) – випущена 19 листопада 2007, одночасно з .NET Framework 3.5. Націлена на створення додатків для ОС Windows Vista (але підтримує і XP), Microsoft Office 2007і веб-додатків. Включає в себе LINQ, нові версії мов C # і Visual Basic. У студію

не ввійшов Visual J #. З 28 жовтня 2008 року вперше доступна версія російською мовою.

Visual Studio 2010 (кодове ім'я Hawaii, для Ultimate – Rosario; внутрішня версія 10.0) – випущена 12 квітня 2010 разом з .NET Framework 4.0. Visual Studio включає підтримку мов C # 4.0 і Visual Basic .NET 10.0, а також мови F #, який був відсутній в попередніх версіях.

Visual Studio 2012 розповсюджується в тих же редакціях, що і 2010. Зміни торкнулися Visual Studio 2012 Express – встановлюються всі мови програмування, а не один, як раніше (Visual Basic 2010 Express, Visual C # 2010 Express), а також тепер існує п'ять версій Visual Studio Express: Visual Studio Express 2012 для Web, Visual Studio Express 2012 для Windows 8, Visual Studio Express 2012 для Windows Desktop, Visual Studio Express 2012 для Windows Phone і Visual Studio Team Foundation Server Express 2012. Всі версії поширюються як окремі додатки. Visual Studio Express 2012 для Windows 8 дозволяє розробляти програми для Windows Store з Modern-інтерфейсом, а Visual Studio Express 2012 для Windows Desktop дозволяє розробляти «класичні» додатка для Робочого столу. Що стосується Visual Studio Team Foundation Server Express 2012, то ця версія поставляється з оболонкою Visual Studio 2012.

Розробляти програми на C++ за допомогою Visual Studio 2012 можна тільки під Windows 7 SP1 і Windows 8. Вийшло виправлення, що дозволяє компілювати програми для запуску під Windows XP.

Фінальний реліз Visual Studio 2013 став доступний для завантаження 17 жовтня 2013 разом з .NET 4.5.1

Різні редакції Visual Studio дозволяють вибрати тільки ті інструменти які Вам потрібні.

Visual Studio Express. Набір легковагих середовищ розробки, що представляють собою урізану версію Visual Studio. Вона включає в себе невеликий набір інструментів, на відміну від повних версій: відсутня дизайнер класів і багато інших інструментів, а також підтримка плагінів і віддалених баз даних в дизайнера даних. Компілятори в 64-бітний код також недоступні в Express редакціях до версій 2012 року (хоча компілятор безкоштовно розповсюджується з Windows SDK і його можна використовувати, компілювати автоматично з IDE можна). Microsoft позиціонує цю лінійку IDE для студентів і аматорів. На даний момент існують наступні Express-редакції:

- Visual Basic Express;
- Visual C ++ Express;
- Visual C # Express;

– Visual Web Developer Express.

Разом з Visual Studio 2012 були випущені нові Express-версії продукту:

– Visual Studio Express 2012 for Web – для web-розробників;
– Visual Studio Express 2012 for Windows 8 – для розробки програм з modern-інтерфейсом (мови: C #, VB.Net, C ++, JavaScript);
– Visual Studio Express 2012 for Windows Desktop – для розробки звичайних десктопних додатків (мови: C #, Visual Basic.Net, C ++);
– Visual Studio Express 2012 for Windows Phone – для розробників під платформи Windows Phone 7.5 та 8.0;
– Visual Studio Team Foundation Server Express 2012.

Ключовими особливостями цих express-версій продуктів є:

– орієнтування на мету розробки, а не на мову;
– необхідність регулярно продовжувати безкоштовну реєстрацію для індивідуальних розробників, якщо розробка на Express-версії ведеться не з метою навчання;
– підтримка компіляції 64-бітного коду;
– підтримка unit-тестів.

Microsoft Visual Studio LightSwitch – це середовище розробки, націлена на створення Line of business додатків, побудованих на існуючих .NET-технологіях і платформах Microsoft. Створювані додатки складаються з трьох ярусів: інтерфейс користувача на Silverlight; логіка і доступ до даних на сервісах WCF RIA і Entity Framework; зберігання даних за допомогою Microsoft SQL Server Express, Microsoft SQL Server або SQL Azure. LightSwitch також підтримує інші джерела даних, включаючи SharePoint. LightSwitch включає в себе графічні дизайнери сутностей та їх відносин, запитів, а також інтерфейсу користувача. Бізнес-логіка може бути написана на Visual Basic або на Visual C #. LightSwitch може бути встановлений як самостійний додаток або як доповнення до Visual Studio 2010 Professional і більш високим редакціям.

Visual Studio Standard. Дана редакція надає IDE для всіх підтримуваних продуктів і підтримує повну версію бібліотеки MSDN. Підтримується редагування XML iXSLT, як і засоби для тестування об'єктів. Проте відсутня оглядач серверів і інтеграція з Microsoft SQL Server. Підтримка розробки під мобільні пристрої спочатку була включена в Visual Studio 2005 Standard, але у версії 2008 вона доступна тільки в Professional-редакції. Починаючи з версії 2010 більше не існує.

Visual Studio Professional. Редакція включає всі можливості Standard Edition, розширюючи їх додатковими, такими, як інтеграція з Microsoft SQL Server та підтримка віддаленої налагодження.

Visual Studio Tools for Office. Visual Studio Tools for Office включає SDK і розширення для Visual Studio, яке містить утиліти для розробки під платформу Microsoft Office. Починаючи з Visual Studio 2008 включено у версії Professional і вище.

Visual Studio Team System. Надає набір інструментів для спільної розробки, підрахунку метрик і створення звітів, на додаток до можливостей Professional редакції. Є різні редакції VSTS, що розділяються по ролям, для яких продукт буде використовуватися:

- Team Explorer (клієнт для TFS);
- Architecture Edition;
- Database Edition;
- Development Edition;
- Test Edition.

Поєднана функціональність всіх чотирьох редакцій представлена в окремому пакеті Visual Studio Team Suite Edition. Функціональність Database Edition буде поєднана з Development Edition в майбутній версії пакету – Visual Studio 2010.

Крім клієнтських додатків, Team System також включає в себе Team Foundation Server.

У вересні 2011 року було оголошено, що в жовтні Microsoft випустить спеціальну версію компілятора, що розробляється в рамках проекту Roslyn. Метою даного проекту була розробка «компілятора у вигляді сервісу» з можливістю видачі програмістам всієї генерується компілятором інформації. Даний компілятор буде мати підтримку мов C # і Visual Basic з повною сумісністю з Visual Studio. Крім того, заявлена можливість конвертації коду з однієї мови на іншу.

Visual Studio побудована на архітектурі, підтримуючої можливість використання вбудованих додатків (англ. Add-Ins) – плагінів від сторонніх розробників, що дозволяє розширювати можливості середовища розробки.

Деякі з найбільш популярних доповнень:

- ReSharper;
- Visual Assist X;
- AnkhSVN – вільна реалізація клієнта Subversion в Visual Studio (в даний час підтримуються версії з 2005 по 2013).

2.4 Обґрунтування вибору мови програмування C#

C# (вимовляється «сі шарп») – об'єктно-орієнтована мова програмування [7]. Розроблено в 1998-2001 роках групою інженерів під керівництвом Андерса Хейлсберг в компанії Microsoft як мова розробки додатків для платформи Microsoft.NET Framework і згодом був стандартизований як ECMA-334 і ISO / IEC 23270.

C# відноситься до сім'ї мов з C-подібним синтаксисом, з них його синтаксис найбільш близький до C ++ і Java. Мова має статичну типізацію, підтримує поліморфізм, перевантаження операторів (у тому числі операторів явного і неявного приведення типу), делегати, атрибути, події, властивості, узагальнені типи і методи, ітератори, анонімні функції з підтримкою замикань, LINQ, винятки, коментарі у форматі XML .

Перейнявши багато що від своїх попередників – мов C++, Pascal, Модула, Smalltalk і, особливо, Java – C#, спираючись на практику їх використання, виключає деякі моделі, що зарекомендували себе як проблематичні при розробці програмних систем, наприклад, C# на відміну від C++ не підтримує множинне спадкування класів (між тим допускається множинне спадкування інтерфейсів).

C# розроблявся як мова програмування прикладного рівня для CLR і, як такий, залежить, насамперед, від можливостей самої CLR. Це стосується, перш за все, системи типів C#, яка відображає BCL. Присутність або відсутність тих чи інших виразних особливостей мови диктується тим, чи може конкретна мовна особливість бути трансльований у відповідні конструкції CLR. Так, з розвитком CLR від версії 1.1 до 2.0 значно збагатився і сам C#; подібної взаємодії слід чекати і надалі (проте, ця закономірність була порушена з виходом C# 3.0, що представляє собою розширення мови, не спираються на розширення платформи .NET). CLR надає C#, як і всім іншим .NET-орієнтованим мовам, багато можливостей, яких позбавлені «класичні» мови програмування. Наприклад, збірка сміття не реалізована в самому C #, а проводиться CLR для програм, написаних на C# точно так само, як це робиться для програм на VB.NET, J# та ін.

Назва «Сі шарп» (від англ. Sharp – дієз) походить від музичної нотації, де знак дієз означає підвищення відповідного ноті звуку наполутон, що аналогічно назвою мови C++, де «++» позначає інкремент змінної. Назва також є грою з ланцюжком C → C++ → C++++ (C#), так як символ «#» можна скласти з 4х знаків «+».

Внаслідок технічних обмежень на відображення (стандартні шрифти, браузери і т. Д.) І тієї обставини, що знак дієз # не представлені на стандартній клавіатурі, знак номера # був обраний для представлення знака дієз при записі імені мови програмування. Ця угода відображено в специфікації мови C# ECMA-334. Проте, на практиці (наприклад, при розміщенні реклами та коробковому дизайні), Майкрософт використовує призначений музичний знак.

C# стандартизований в ECMA (ECMA-334) і ISO (ISO / IEC 23270).

Відомо як мінімум про три незалежних реалізаціях C#, що базуються на цій специфікації і знаходяться в даний час на різних стадіях розробки:

- Mono, розпочата компанією Ximian, продовжена її покупцем і наступником Novell, а потім Xamarin.
- dotGNU і Portable.NET, що розробляються Free Software Foundation.
- SharpDevelop.

Протягом розробки мови C# було випущено декілька його версій (табл. 2.2).

Таблиця 2.2 – Версії мови C#

Версія	Специфікація мови			Дата	.NET Framework	Visual Studio (VS)
	ECMA	ISO/IEC	Microsoft			
C# 1.0	Грудень 2002	Квітень 2003	Січень 2002	Січень 2002	.NET Framework 1.0	VS.NET 2002
C# 1.2			Жовтень 2003	Квітень 2003	.NET Framework 1.1	VS.NET 2003
C# 2.0	Червень 2006	Вересень 2006	Вересень 2005	Листопад 2005	.NET Framework 2.0	VS 2005
C# 3.0	Відсутня		Серпень 2007	Листопад 2007	.NET Framework 3.5	VS 2008
C# 4.0			Квітень 2010	Квітень 2010	.NET Framework 4	VS 2010
C# 5.0				Серпень 2012	.NET Framework 4.5	VS 2012

Версія 1.0. Проект C# був початий в грудні 1998 і отримав кодову назву COOL (C-style Object Oriented Language). Версія 1.0 була анонсована разом з платформою .NET в червні 2000 року, тоді ж з'явилася і перша загальнодосту-

пна бета-версія; C# 1.0 остаточно вийшов разом з Microsoft Visual Studio .NET в лютому 2002 року.

Перша версія C# нагадувала за своїми можливостями Java 1.4, декілька їх розширюючи: так, в C# були властивості (що виглядають в коді як поля об'єкта, але на ділі викликають при зверненні до них методи класу), індексатори (подібні властивостям, але приймають параметр як індекс масиву), події, делегати, цикли foreach, структури, що передаються за значенням, автоматичне перетворення вбудованих типів в об'єкти при необхідності (boxing), атрибути, вбудовані засоби взаємодії з некерованим кодом (DLL, COM) та інше.

Крім того, в C# вирішено було перенести деякі можливості C++, відсутні в Java: без знакові типи, перевантаження операторів (з деякими обмеженнями, на відміну від C++), передача параметрів в метод за посиланням, методи зі змінним числом параметрів, оператор goto (з обмеженнями). Також в C # залишили обмежену можливість роботи з покажчиками – у місцях коду, спеціально позначених словом unsafe і при вказівці спеціальної опції компілятору.

Версія 2.0. Проект специфікації C# 2.0 вперше був опублікований Microsoft у жовтні 2003 року; в 2004 році виходили бета-версії (проект з кодовою назвою Whidbey), C# 2.0 остаточно вийшов 7 листопада 2005 разом з Visual Studio 2005 і .NET 2.0.

Нові можливості у версії 2.0:

- Часткові типи (розділення реалізації класу більш ніж на один файл);
- Узагальнені, або параметризовані типи (generics). На відміну від шаблонів C ++, вони підтримують деякі додаткові можливості і працюють на рівні віртуальної машини. Разом з тим, параметрами узагальненого типу не можуть бути вирази, вони не можуть бути повністю або частково спеціалізовані, не підтримують шаблонних параметрів за замовчуванням, від шаблонного параметра не можна успадковуватися, і т.д.;
- Нова форма ітератора, що дозволяє створювати співпрограми за допомогою ключового слова yield, подібно Python і Ruby;
- Анонімні методи, що забезпечують функціональність замикання;
- Оператор '??': return obj1 ?? obj2; означає (в нотації C # 1.0) return obj1! = null? obj1: obj2 ;
- Обнуляти ('nullable') типи-значення (що позначаються знаком питання, наприклад, int? I = null;), що представляють собою ті ж самі типи-значення, здатні приймати також значення null. Такі типи дозволяють поліпшити взаємодію з базами даних через мову SQL;

- Можливість створювати збережені процедури, тригери і навіть типи даних на .Net мовах (у тому числі і на C #);
- Підтримка 64-розрядних обчислень, що крім усього іншого, дозволяє збільшити адресний простір і використовувати 64-розрядні примітивні типи даних.

Версія 3.0. У червні 2004 року Андерс Гейлсберг вперше розповів на сайті Microsoft про плановані розширеннях мови в C# 3.0. У вересні 2005 року вийшли проект специфікації C# 3.0 і бета-версія C# 3.0, що встановлюється у вигляді доповнення до існуючих Visual Studio 2005 і. NET 2.0. Остаточно ця версія мови увійшла в Visual Studio 2008 і .NET 3.5.

Нові можливості у версії 3.0

У C# 3.0 з'явилися наступні радикальні додавання до мови:

- Ключові слова select, from, where, що дозволяють робити запити з SQL, XML, колекцій і т. П. (Запит, інтегрований в мову, Language Integrated Query, або LINQ);
- Ініціалізація об'єкта разом з його властивостями;
- Лямбда-вирази;
- Дерева виразів:
  лямбда-вирази тепер можуть представлятися у вигляді структури даних, доступній для обходу під час виконання, тим самим дозволяючи транслювати строго типізовані C# -виражені в інші домени (наприклад, вирази SQL);
- Виведення типів локальної змінної: var x = "hello"; замість string x = "hello";
- Безіменні типи: var x = new {Name = "James"};
- Методи-розширення – додавання методу в існуючий клас за допомогою ключового слова this при першому параметрі статичної функції;
- Автоматичні властивості: компілятор згенерує закрите (private) поле і відповідні аксессор і мутатор для коду виду.

C # 3.0 сумісний з C # 2.0 по генеруємому MSIL-коду; поліпшення в мові – чисто синтаксичні і реалізуються на етапі компіляції. Наприклад, багато хто з інтегрованих запитів LINQ можна здійснити, використовуючи безіменні делегати в поєднанні з предикативними методами над контейнерами на подобі List.FindAll і List.RemoveAll.

Версія 4.0. Прев'ю C # 4.0 було представлено в кінці 2008 року, разом з CTP-версією Visual Studio 2010.

Visual Basic 10.0 і C # 4.0 були випущені в квітні 2010 року, одночасно з випуском Visual Studio 2010.

Нові можливості у версії 4.0:

1) Можливість використання пізнього зв'язування, для використання:
   а) з мовами з динамічною типізацією (Python, Ruby);
   б) з COM-об'єктами;
   в) відображення (reflection);
   г) об'єктів із змінною структурою (DOM). З'являється ключове слово dynamic;
2) Іменовані і опціональні параметри;
3) Нові можливості COM interop;
4) коваріантного і контраваріантним;
5) Контракти в коді (Code Contracts).

2.5 Обґрунтування вибору мови програмування Swift та Objective-C

Objective-C – об'єктно-орієнтована мова програмування, що використовується корпорацією Apple, побудована на основі мови C і парадигм Smalltalk. Зокрема, об'єктна модель побудована в стилі Smalltalk – тобто об'єктами надсилаються повідомлення. Мова Objective-C є потомком мови C, тому код C повністю зрозумілий компілятору Objective-C.

Компілятор Objective-C входить в GCC і доступний на більшості основних платформ. Мова використовується в першу чергу для Mac OS X (Cocoa) і GNUstep – реалізацій об'єктно-орієнтованого інтерфейсу OpenStep. Також мова використовується для iOS (Cocoa Touch).

На початку 1980-х років було популярне структурне програмування, що дозволяє розділити алгоритм на невеликі блоки. Однак, із зростанням складності завдань, структурне програмування призводило до зниження якості коду. Доводилося писати все більше функцій, які дуже рідко могли використовуватися в інших програмах.

Розробники побачили в об'єктно-орієнтованому програмуванні потенційне рішення виниклої проблеми. З одного боку, Smalltalk використовували майже всі більш-менш складні системи. З іншого – використання віртуальних машин підвищувало вимоги до ресурсів.

Objective-C був створений Бредом Коксом на початку 1980-х в його компанії Stepstone. Він намагався вирішити проблему повторного використання коду.

Метою Кокса було створення мови, що підтримує концепцію software IC, що має на увазі можливість збирати програми з готових компонентів (об'єктів), подібно до того як складні електронні пристрої можуть бути зібрані з набору готових інтегральних мікросхем. При цьому мова повинна бути простою і заснованою на мові C, щоб полегшити перехід розробників на неї.

Однією з цілей було також створення моделі, в якій самі класи є повноцінними об'єктами, підтримувалася б інтроспекція і динамічна обробка повідомлень. Objective-C є розширенням C, будь-яка програма на C є програмою на Objective-C. Однією з відмінних рис Objective-C є динамічність, рішення, зазвичай приймаються на етапі компіляції, тут відкладаються до етапу виконання.

Objective-C – message-oriented-мова, в той час як C ++ – function-oriented, в Objective-C виклики методу інтерпретуються не як виклик функції (хоча до цього зазвичай все зводиться), а як посилка повідомлення (з ім'ям і аргументами) об'єкту, подібно до того, як це відбувається в Smalltalk.

Будь-якому об'єкту можна послати будь-яке повідомлення. Об'єкт може замість обробки повідомлення переслати його іншому об'єкту для обробки (делегування), зокрема, так можна реалізувати розподілені (тобто знаходяться в різних адресних просторах і навіть на різних комп'ютерах) об'єкти. Прив'язка повідомлення до відповідної функції відбувається на етапі виконання.

Мова Objective-C підтримує роботу з метаінформацією – так, на етапі виконання можна дізнатися клас об'єкта, список його методів (з типами переданих аргументів) і instance-змінних, перевірити, чи є клас нащадком заданого і чи підтримує він заданий протокол і т.д.

У мові є підтримка протоколів (поняття інтерфейсу об'єкта і протоколу чітко розділені). Підтримується спадкування (не множинне), для протоколів підтримується множинне спадкування. Об'єкт може бути успадкований від іншого об'єкта і відразу декількох протоколів (хоча це скоріше не успадкування протоколу, а його підтримка).

На даний момент мова Objective-C підтримується компіляторами Clang і GCC (під управлінням Windows використовується в складі MinGW або cygwin).

Деякі функції мови перенесені в runtime-бібліотеку і сильно залежать від неї. Разом з компілятором gcc поставляється мінімальний варіант такої бібліотеки. Також можна вільно скачати runtime-бібліотеку компанії Apple: Apple's Objective-C runtime.

Swift – мова програмування загального призначення. Створена компанією Apple в першу чергу для розробників iOS і OSX. Swift використовує патерни безпечного програмування і містить сучасні функції, які допомагають зробити програмування легким, гнучким і захоплюючим. Створений з нуля, Swift, що спирається на зрілі та всіма улюблені фреймворки Cocoa і CocoaTouch – це можливість переосмислити розробку програмного забезпечення. Вона замислювалася як більш легка для читання і стійка до помилок програміста мова, ніж її попередник Objective-C. Програми на Swift компілюються за допомогою LLVM, що входить в інтегровану середу розробки Xcode 6 і вище. Swift може використовувати рантайм Objective-C, що робить можливим використання обох мов (а також C) в рамках однієї програми [11].

Старший віце-президент по розробці програмного забезпечення Apple Крейг Федеріго під час анонсу цього продукту заявив, що мова програмування Swift був закладений ще вплатформі NeXT (ОС NeXTSTEP випускаласяв 1989-1995 роках), яка стала основою для macOS, а потім і iOS.

Розробка поточного варіанту мови Swift почалася в 2010 році Крісом Латтнером, керівником відділу розробки інструментів для створення програмного забезпечення Apple і одним з основних розробників LLVM. Swift запозичив ідеї з Objective-C, Rust, Haskell, Ruby, Python, C #, CLU, і ще з багатьох мов програмування.

2 червня 2014 року на конференції WWDCSwift був офіційно представлений разом з безкоштовним керівництвом по використанню мови обсягом в 500 сторінок, доступним на сервісі «iBookStore».

Версія Swift 1.0 була випущена 9 вересня 2014 року, разом з «GoldMaster» версією Xcode 6.0 для iOS.

8 червня 2015 року компанія Apple оголосила про випуск нової версії Swift 2.0, яка отримала вищу продуктивність, нове API обробки помилок, поліпшення синтаксису мови, а також функцію перевірки доступності функцій Swift для цільових ОС.

3 грудня 2015 року було випущено бета версію Swift 3.0 з підтримкою операційних систем OSX, iOS і Linux і ліцензована під відкритою ліцензій Apache 2.0 та RuntimeLibraryException.

10 квітня 2016 року Google оголосила про наміри зробити Swift так званою «першою мовою» для Android. Мова програмування дуже швидка, тому Google планує нею скористатися. Швидше за все, це зменшить кількість додатків, які спочатку виходять на iOS, а пізніше на Android.

Swift здасться знайомим для розробників Objective-C. Він запозичує читабельність іменованих параметрів Objective-C і міць динамічної моделі об'єктів Objective-C. Він забезпечує плавний доступ до існуючих фреймворків Cocoa і можливість змішувати код з кодом Objective-C. Побудований на цій загальній основі, Swift надає багато нових можливостей і уніфікує процедурну і об'єктно-орієнтовану частини мови.

Swift доброзичливий для новачків в програмуванні. Це перша мова програмування промислового якості, який так само зрозумілий і цікавий, як скриптова мова. Він підтримує інноваційну функцію – playground, яка дозволяє експериментувати з кодом Swift і бачити результат миттєво, без необхідності компілювати і запускати додаток[12].

2.6 Прилади та програмні комплекси системи "Розумний будинок" AMX

Компанія AMX була заснована у Далласі, штат Техас, в 1982 році. За роки існування компанії було відкрито багато офісів по всьому світу, зокрема Європейські офіси у Великобританії, Німеччині, Франції, Росії, Нідерландах, Бельгії та Швеції [9].

Компанія пропонує рішення, обладнання і ПЗ для управління пристроями, комутації аудіо- і відеосигналів, IPTV, розподілу аудіо сигналів, цифрових інформаційних систем Digital Signage та автоматизованого керування ресурсами. AMX – провідний постачальник рішень, які спрощують впровадження, супровід, а також використання технологій для створення ефективного середовища роботи і проживання.

Завдання компанії AMX полягає в тому, щоб користувач системи управління максимально зручно і просто звертався з використовуваної технікою. Саме тому вона пропонуємо набір високотехнологічного обладнання, за допомогою якого можливо запропонувати унікальні рішення для задоволення конкретних потреб користувача.

AMX вже вийшов за рамки традиційного уявлення про систему управління та Аудіо-відео. Тепер всі системи включені в єдину систему управління об'єктом.

За допомогою обладнання AMX полегшується використання технологій і створюється зручна і ефективна технічна середовище для користувачів. Ефективність системи полягає в таких факторах як:

- Максимальна продуктивність, комфорт і безпека;
- Забезпечення надійності обладнання;

- Підтримання низьких операційних витрат;
- Енергозбереження;
- Максимальний «термін придатності» програмних продуктів;
- Відповідність вимогам державних стандартів;
- Спрощення експлуатації допомогою всеосяжної системи контролю та моніторингу роботи обладнання.

Модульне, виконане в єдиному стандарті цифрове обладнання AMX може бути налаштоване для задач візуалізації будь-якого рівня і масштабу, в тому числі і для комплексних рішень, які суміщають в собі цифрові інформаційно-рекламні системи та інтерактивні ефекти.

Розширена пропозиція AMX в області цифрових медіа продуктів дозволяє дилерам і інтеграторам пропонувати максимально гнучкі рішення, прекрасно адаптуються до будь-яких вимог замовника.

Модульність в продуктовій лінійці дає можливість використовувати ті можливості обладнання, які необхідні сьогодні з повною впевненістю в тому, що система може бути розширена в майбутньому і нові функції будуть додані максимально легко.

2.7 Прилади та програми системи керування освітленням Lutron

Lutron Electronics Co., Inc. – світовий лідер у виробництві систем керування освітленням. Компанія вийшла на ринок в 1961 році з першим у світі електронним димером, і з тих пір тримає пальму першості в розробці інноваційних рішень в управлінні освітленням. У кожному з представлених продуктів закладено низку унікальних технологій, що дозволяє добитися неперевершеної надійності та функціональності обладнання [10].

У 1959 році американський інженер Джоель Спіра (Joel Spira) сконструював і запатентував перший у світі напівпровідниковий світлорегулятор. Цей винахід ліг в основу компанії Lutron. Перша модель світлорегулятора отримала назву Capri. Пристрій мав компактний розмір і низьке тепловиділення, що дозволяло встановлювати його в стандартну монтажну коробку замість звичайного вимикача. Популярні в той час реостатні світлорегулятори використовувалися в основному на комерційних об'єктах (здебільшого в театрах), мали великі габарити і високу вартість. Capri став першим масовим світлорегулятором для побутового використання.

У 1971 році на ринку з'явився світлорегулятор Nova, який був виконаний у вигляді слайдера і підтримував управління навантаженням до 2кВт.

Діммер RanaX, який побачив світ у 1989 вдавав із себе модернізований Nova з вбудованим ІЧ-приймачем, працював з різними типами навантажень і продавався в комплекті з ІЧ пультом.

У 1993 році з'явилися два знакових для компанії продукту. Grafik Eye – багатоканальний димер, поєднаний зі сценарним контролером, який дозволяв налаштувати освітлення в окремо взятому приміщенні будівлі, конференц залі або домашньому кінотеатрі. Serena-моторизовані штори з цифровим керуванням, що дозволяють задавати не тільки крайні положення Відкрити і закрити, а й будь-які проміжні.

У 1997 на ринку США з'явилася бездротова система управління освітленням RadioRa. Рішення працювало зі зворотним зв'язком, що дозволяло знати рівні включення навантажень в кожен момент часу.

У портфоліо компанії – найкращі архітектурні проекти різного призначення – від офісів і адміністративних споруд до приватних резиденцій, і в тому числі, кращі готелі всесвітньо відомих брендів (Grand Hyatt, The Hilton, The Marriot, Grand Sheraton, Four Seasons, а також готелі Ritz Carlton (Німеччина), Imperial Hotel (Індонезія / Японія), The Landmark Hotel та ін.).

Варто лише сказати, що обладнання Lutron це не просто кращий у своєму роді продукт – це атрибут проекту найвищого статусу.

Lutron виробляє більше 10 тис. Найменувань продукції і володіє понад 2700 патентами в галузі управління освітленням. За результатами щорічних опитувань американського видання CePro компанія займає лідируючі позиції на ринку США в галузях Управління Освітленням і моторизованими Шторами.

Починаючи з 90-х років минулого століття основний фокус компанії спрямований на виробництво і впровадження енергозберігаючих технологій в освітленні. Змінилося декілька поколінь платформ для автоматизації будівель. На даний момент на ринку представлені дві: Homeworks QS – рішення для управління освітленням в приватних проектах і Quantum – рішення для управління освітленням в комерційних будівлях. Рішення включають в себе димери і реле, датчики присутності і освітленості, щити управління, софт і тд. Крім виробництва обладнання для управління штучним освітленням, компанія виробляє моторизовані штори Sivoia QS. Приводи Sivoia мають низькі показники за рівнем шуму (38 dB) і оснащені цифровим керуванням, завдяки чому стало можливо точно управляти положенням штор на їх основі. Для рулонних штор зміна положення протягом дня може бути прив'язане до руху сонця щодо фасаду будівлі, що використовується для підтримки постійного рівня освітленос-

ті на робочих місцях, перешкоджає проникненню прямих сонячних променів і зайвого нагрівання приміщень влітку.

Програмним забезпеченням для налаштування компонентів під керуванням процесору Lutron є Homeworks Illumination. Програма допомагає налаштувати та описати все керування процесору, тобто реакції на дії людини – натиснення кнопок на настінних панелях, спрацьовування датчиків і т.д. Вона допомагає наладити зв'язок між компонентами в системі, а також повністю відладити роботу процесору, за для стійкості його роботи.

Homeworks Illumination має графічний інтерфейс який зрозумілий користувачу. Також в ній можна в графічному режимі вистроїти всі компоненти, щоб більш зрозуміло було прописувати скрипти та налаштовувати роботу кожного з них окремо, наприклад, в рамках кожної кімнати.

3 РОЗРОБКА КОМПЛЕКСУ ПРОГРАМ СИСТЕМИ "РОЗУМНИЙ БУДИНОК"

В даному дипломному проекті необхідно розробити комплекс програм, які керувати будинком з встановленою системою "Розумний будинок". Комплекс складається з трьох частин (програм), а саме:

- Програма, яка буде встановлена на панель керування;
- Програма, яка буде встановлена на контролер AMX;
- Програма,яка буде встановлена на контролер Lutron.

Задачею є створення системи "Розумний будинок" для магазину, а саме керування освітленням, шторами, подачею води та аудіо технікою в залі магазину, в рекламних цілях.

Для виконання задачі перш за все потрібно скласти електронний план будівлі, для цього були використанні архітекторські креслення будівлі. План будівлі потрібний для затвердження розміщення усіх пристроїв, датчиків та інфрачервоних випромінювачів, а також для розмітки розміщень проводів живлення та передачі інформації.

Дана система була написана для магазину "Акустика Hi-Fi, Hi-End, Умный дом" який розташований в місті Одеса, на вулиці Рішельєвська, будинок 43 (рис. 3.1).

Рисунок 3.1 – Фотографія магазина з системою "Розумний будинок"

3.1Програмне забезпечення необхідне для написання програм

Для написання програми для панелі керування потрібно встановити Xcode для створення програми для панель керування(рис 3.3). При створені програми керування для персонального компютера будемо використовувати мову програмування C#, тому потрібно встановити Microsoft Visual Studio C# Express 2010, не нижчу, а також Microsoft .Net Framework 4.5 (рис 3.2). Щоб розробити програму для контролера AMX потрібно встановити NetLinxStudio. Для налаштування процесору Lutron потрібно встановити Homeworks Illumination (рис 3.4).

Після встановлення Microsoft Visual Studio C# Express 2010 і Microsoft .Net Framework 4.5 можна створювати програми на мові C# для ОС Windows. Також після встановлення Xcode ми можемо протестувати тестові додатки на вбудованому симуляторі (рис 3.5).

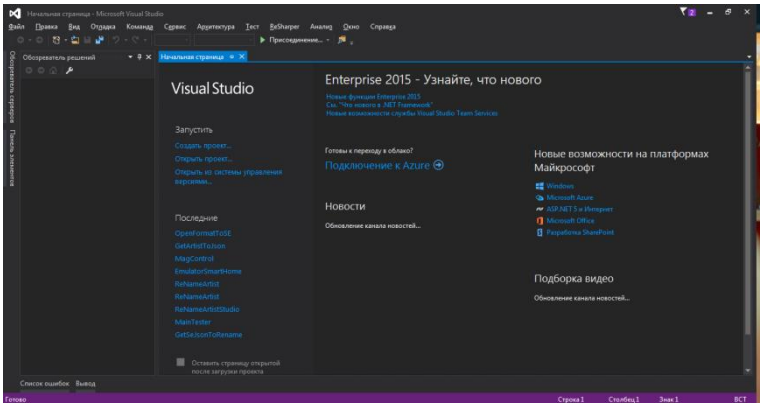

Рисунок 3.2 – Початкова сторінка Microsoft Visual Studio

Рисунок 3.3 – Початкова сторінка Xcode

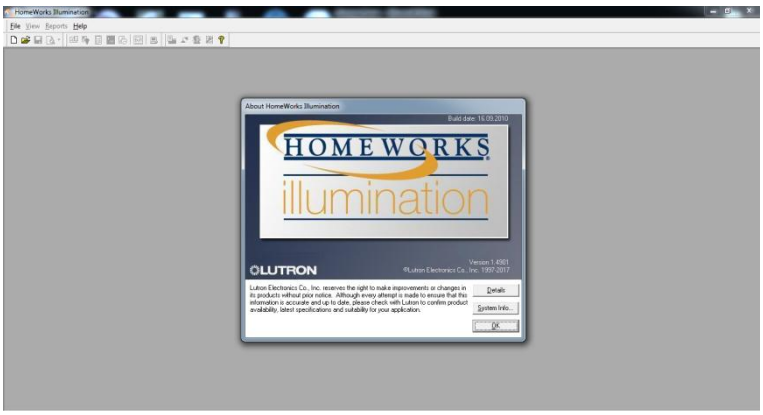

Рисунок 3.4 – Початкова сторінка Homeworks Illumination

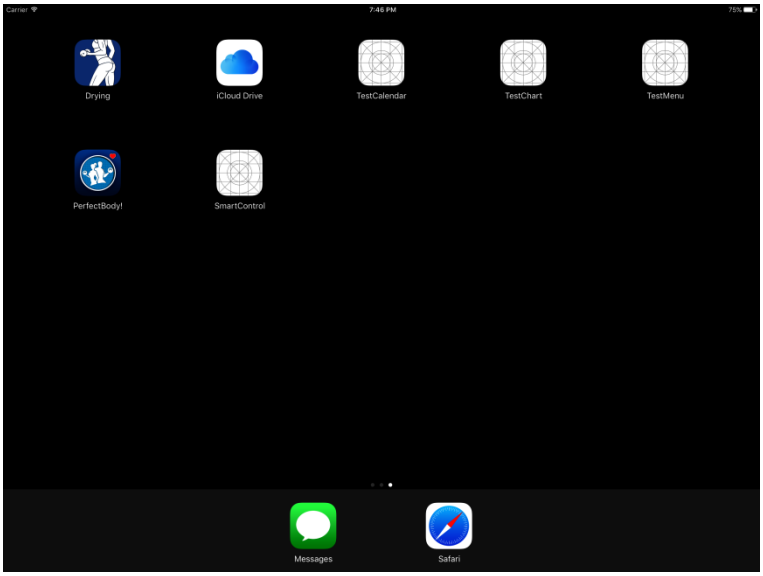

Рисунок 3.5 – Симулятор IPad 12.9 дюймів

3.2 Основні принципи створення програми в NetLinxStudio

Система "Розумний будинок" повинна мати мозок, тобто центральний контролер з прописаному у ньому діями, які і повинна виконувати система. Для створення заданого проекту буде використаний контролер AMX NI-2100 (рис 3.6), для керування димирами буде використано процесор Lutron P5(рис 3.7). В структурі цього контролеру є все те що нам потрібно, а саме 4 порти дискретного вводу/виводу, 3 інфрачервоних випромінювача, порт Ethernet (TCP/IP), обнаружувач панелів керування AMX. Програми для виконання певних дій для всіх моделей контролерів AMX можуть бути створені в NetLinxStudio. Сама мова програмування NetLinx створена на основі мови

програмування C++, але її синтаксис досить відрізняється від синтаксису будь якої мови програмування.

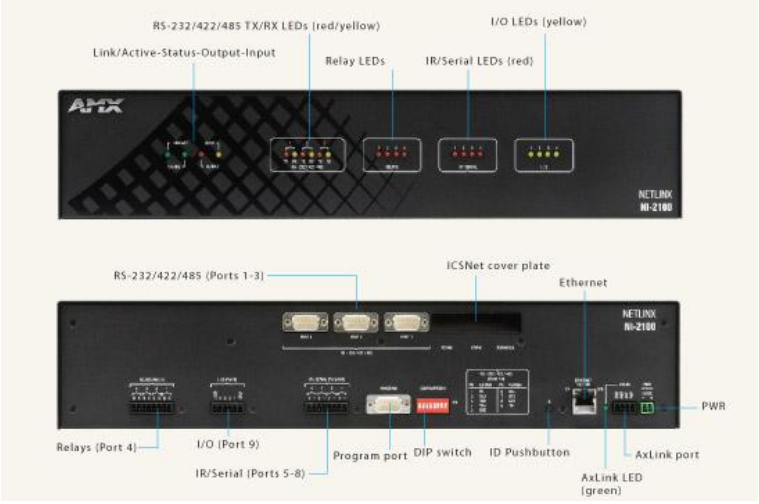

Рисунок 3.6 – Контролер AMX NI-2100

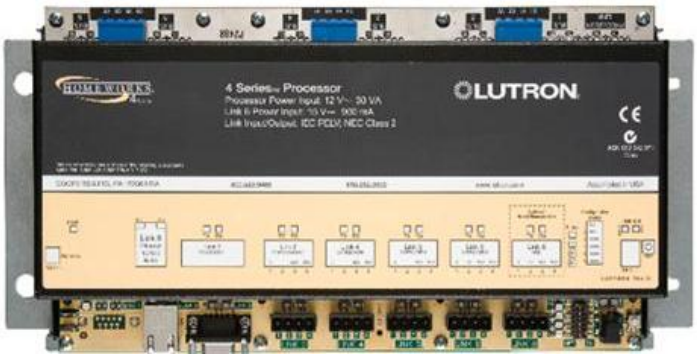

Рисунок 3.7 – Lutron P5 Processor

В новоствореному проекті створюється головний файл (програма/клас). Він в своїй структурі містить такі основні частини:

– PROGRAM_NAME – ім'я програми/класу;
– DEFINE_DEVICE – ініціалізація віртуальних та реальних пристроїв;
– DEFINE_CONSTANT – ініціалізація констант;
– DEFINE_VARIABLE – ініціалізація змінних примітивних типів;
– DEFINE_START – початок виконання програми;
– DEFINE_EVENT – ініціалізація слухачів подій.

Використовуючи NetLinxStudio (рис 3.8) ми маємо змогу підключитися до контролеру AMX, відправити на нього проект, а також відстежувати поведінку контролера в залежності від відправлених на нього даних та подій.

Також, за допомогою широкого набору інструментів, ми маємо змогу керувати портами та відправляти події без використання панелі керування. Це дає змогу перевірити правильність написання програми та реакцій на прописані події.

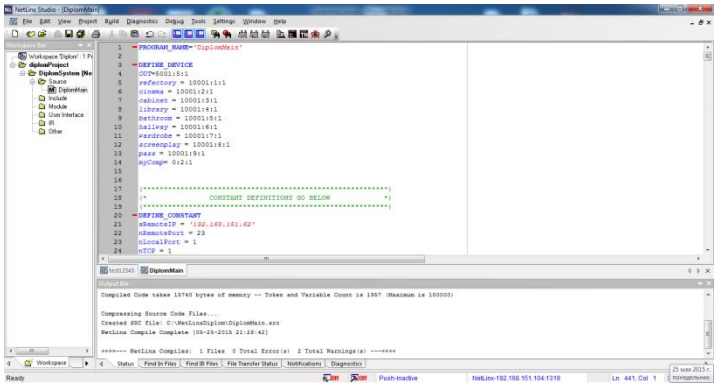

Рисунок 3.8 – Інтерфейс користувача NetLinxStudio

Спілкування контролера з іншими пристроями відбувається за допомогою різних типів зв'язку:

- по протоколу TCP або UDP – створюється клієнт або сервер в залежності від типу пристрою. Спілкування відбувається повідомленнями типу string. Відправлення повідомлення відбувається за допомогою функції SEND_STRING DEV, <string> або SEND_STRING DEV[ ], <string>. Отримання повідомлення відбувається по слухачу події DATA_EVENT[DEV] або DATA_EVENT[DEV[ ]]{ STRING:{// STRINGeventhandler}}. По слухачу подій DATA_EVENT також можна прописати різні дії на події появи чи зникнення в мережі пристрою або групи пристроїв DEV, або помилки у з'єднані або роботі заданого пристрою;
- по дискретним портам вводу/виводу. Вони можуть вмикатися по команді з контролеру передаючи напругу в 12В або можуть відслідковувати замкненість проводу, наприклад можуть бути підключені до датчика відчинення двері, що при включеній системі охорони може сповістити власника про те що двері були відчинені.
- по інфрачервоним випромінювачам. Односторонній зв'язок, який передає заданий контролером сигнал на приймачі, наприклад інфрачервоний діод може бути направленим на приймач телевізора, кондиціонера або, як в даному проекті, направленим на приймач AV ресивера.

– по RS-232 / RS-422 / RS-485 або AxLink порту. Створюється програмний зв'язок з потрібними атрибутами. Все інше працює так само як і по протоколу TCP або UDP.

Також у верхніх моделей є вбудовані реле, що дозволяють без сторонніх пристроїв подавати чи припиняти подачу електрики, води, газу та ін.

Для взаємодії програми з панеллю керування також є різні методи відправлення даних на панель, основні та часто застосовувані приставлені нижче:

– SEND_STRING DEV, <string> – відправлення повідомлення типу string;
– SEND_COMMAND DEV,"'@TXT',id,<string> – відправлення команди на панель (код)DEV (id). Основне використання є відправлення тексту який буде відображений на панелі;
– SEND_LEVEL DEV,ID,<level> – відправлення значення на рівень за адресою DEV (id) на панелі;
– ON[DEV,id] або OFF[DEV,id] – вмикає або вимикає кнопку тригер яка знаходиться за адресою DEV (id)на панелі;

Основні слухачі подій для взаємодії з панеллю:

– слухач натиску, утримання та відпуску кнопки на панелі BUTTON_EVENT[[<DEV>,<id>] або BUTTON_EVENT[(DEVCHAN[])]{PUSH:{} RELEASE:{} HOLD[TIME]: або HOLD[TIME, REPEAT]:{}}
– слухач зміни значення рівня на панелі LEVEL_EVENT[<DEV>,<id>]{};
– слухач зміни значення тригер кнопки CHANNEL_EVENT[<DEC>,<id>]{ON:{}OFF}}

В даному пункті ми розглянули методи зв'язку з різними пристроями та панелями керування. Також були розглянуті різні слухачу подій. Використовуючи дану інформацію була написана програма для контролера AMX код якої наведений в Додатку Г.

Для керування аудіо та відео технікою використовуються інфрачервоні випромінювання (ІЧ команди). Для їх запису необхідно записати ці команди через спеціальний пристрій AMXIris. Для цього необхідно відкрити програму IREdit (рис. 3.9), та створити новий файл команд. Після цього ми можемо записати до 256 команд в один файл. Далі тиснемо на запис, називаємо його, тобто даємо ім'я команди. Далі записуємо саму команду, для цього потрібно направити пульт керування, який ми хочемо записати, на лицьову сторону

пристрою Iris. Приблизно пів секунди, пристрій сам оброблює сигнал, ділить його на цикли, які повторюються, та записує один цикл для команди.

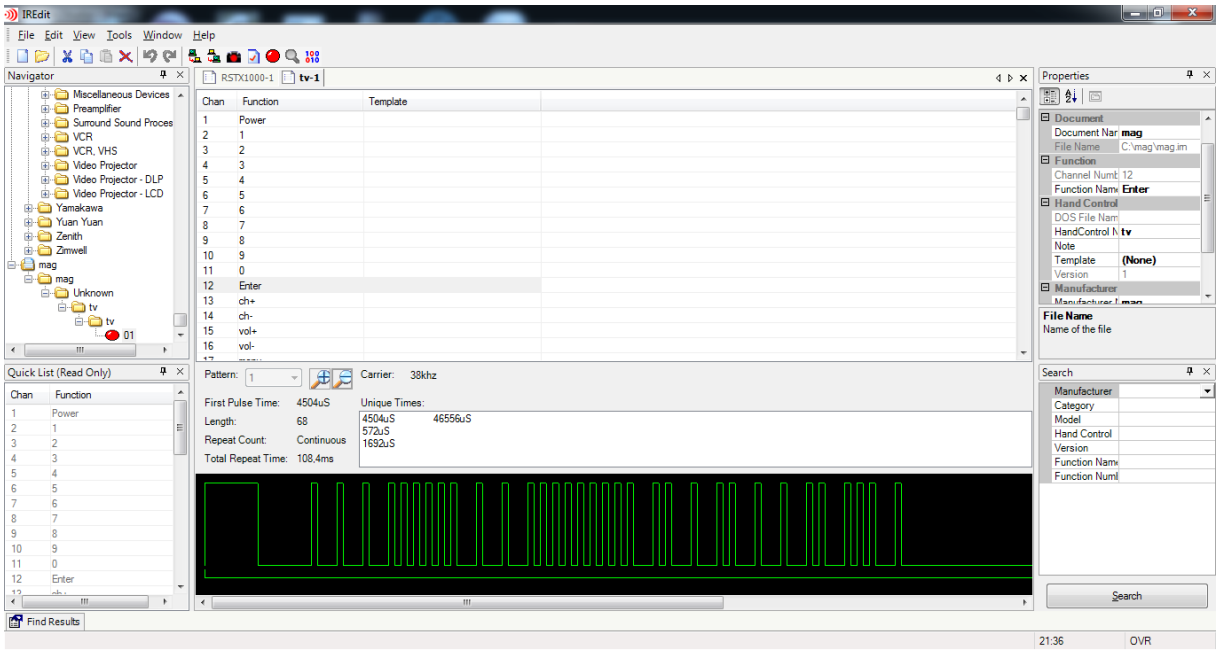

Рисунок 3.9 – Інтерфейс користувача IREdit

Також не обов'язково зчитувати пульти, команди можна скачувати з інтернету в вигляді HEX коду. Для цього необхідно знайти сайт на якому є такі команди у такому вигляді:

```
0000 006d 0022 0003 00a9 00a8 0015 003f 0015 003f
0015 003f 0015 0015 0015 0015 0015 0015 0015 0015
0015 0015 0015 003f 0015 003f 0015 003f 0015 0015
0015 0015 0015 0015 0015 0015 0015 0015 0015 0015
0015 003f 0015 0015 0015 0015 0015 0015 0015 0015
0015 0015 0015 0015 0015 0040 0015 0015 0015 003f
0015 003f 0015 003f 0015 003f 0015 003f 0015 003f
0015 0702 00a9 00a8 0015 0015 0015 0e6e
```

Після цього неохідно запустити процес створення ІЧ команди за допомогою IRHEXCodes (рис. 3.10), далі копіюємо hex в вікно створення команди(рис. 3.11).

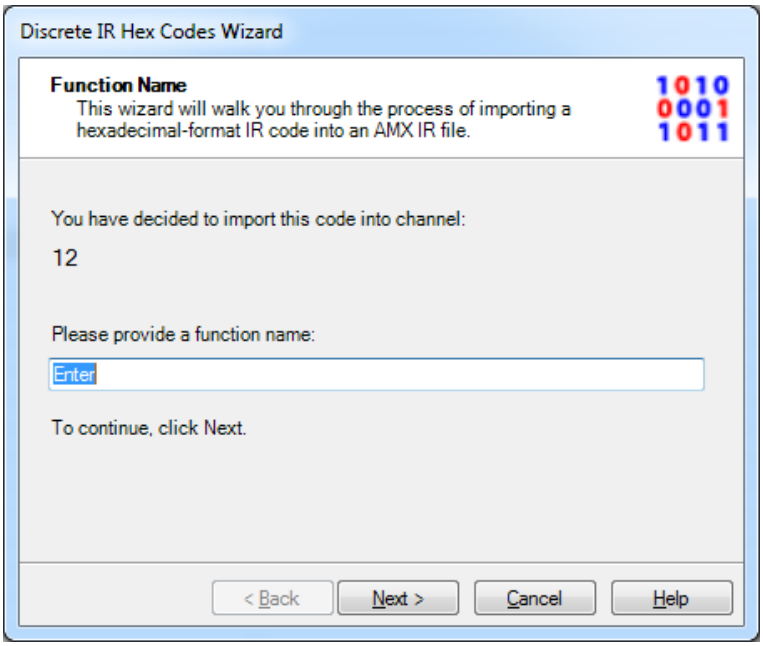

Рисунок 3.10 – Вікно створення нової ІЧ команди

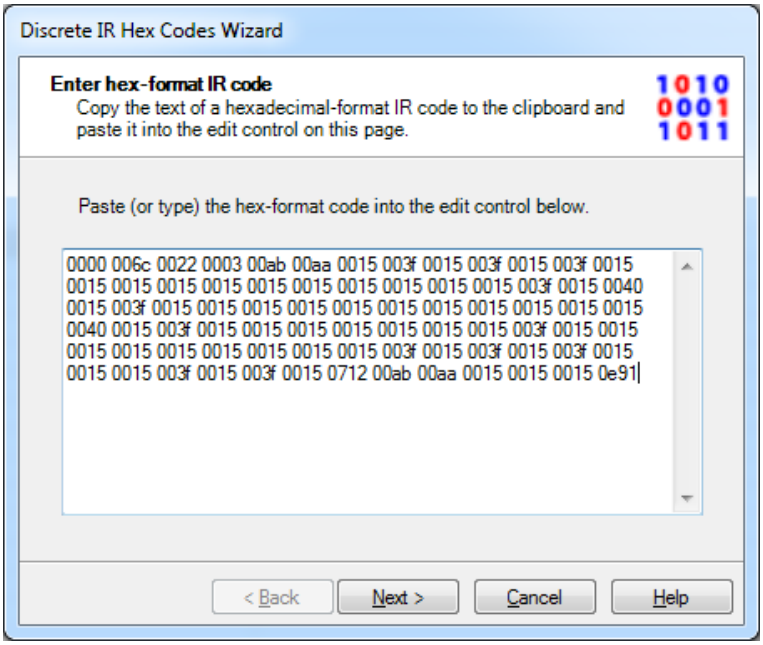

Рисунок 3.11 – Вікно створення ІЧ команди за допомогою hex

Також за допомогою NetLinxStudio ми можемо протестувати реакцію процесору на події. Для цього ми використовуємо модуль "ControlaDevice". Від дозволяє емулювати панель керування та посилати на процесор такі ж сигнали я вона. Тобто потрібно сказати на який порт потрібно відправити сигнал, або дію та відстежити реакцію процесора на неї.

3.3 Створення програми для панель керування на ОС iOs

Панелі керування, які дають змогу керувати пристроями через контролери AMX є різного типу. Це може бути примітивна кількох кнопкова настінна панель керування або може бути панель яка своїм виглядом нагадує звичайний пульт, але з маленьким екраном. В основному використовуються панелі які виглядають як сучасний планшет, який можна вбудувати як в стіну, так і робити переносною. Панель може буди навіть великих розмірів та займати досить велику частину офісного столу (рис 3.12).

Рисунок 3.12 – Панелі керування AMXModeroXSeries

Також в ролі панелі керування може виступати телефон, планшет або навіть комп'ютер. Можливе це за допомогою програмного забезпечення яке встановлюється на телефон, планшет та комп'ютер. Це програмне забезпечення емулює роботу таким чином, що обнаружувач панелів керування, який є в кожному контролеру AMX, відзначає їх як панелі керування AMX.

Компанія AMX розробила свій візуальний редактор для створення інтерфейсу користувача та відлагодження правильного керування панеллю. Називається цей продукт TPDesign4, а в минулому році була випущена нова версія даної програми TPDesign5. Програма досить зручна в користуванні, але нажаль має досить обмежений набір інструментів та дій які може виконувати панель. В даній ситуації найкращим виходом є використання нативних мов програмування, для iOs це Swift(рис. 3.13). Вона дає змогу виконувати керування з iPhone, iPodi і Pad,підключення до процесору відбувається за допомогою TCP протоколу, тобто в своєму роді повторює роботу програми TPDesign4, але дає змогу використовувати в процесі роботи панелі всі можли-

вості даних пристроїв, наприклад, таких, як Push-повідомлення та iCloud. Це значно розширює діапазон можливостей панелі, а також дозволяє створювати різні ефекти та анімації. Також Swift може використовувати різні властивості телефону, такі як акселерометр та вбудований компас, що також дозволяє робити унікальний дизайн та відстежувати місце знаходження кожної панелі в домі.

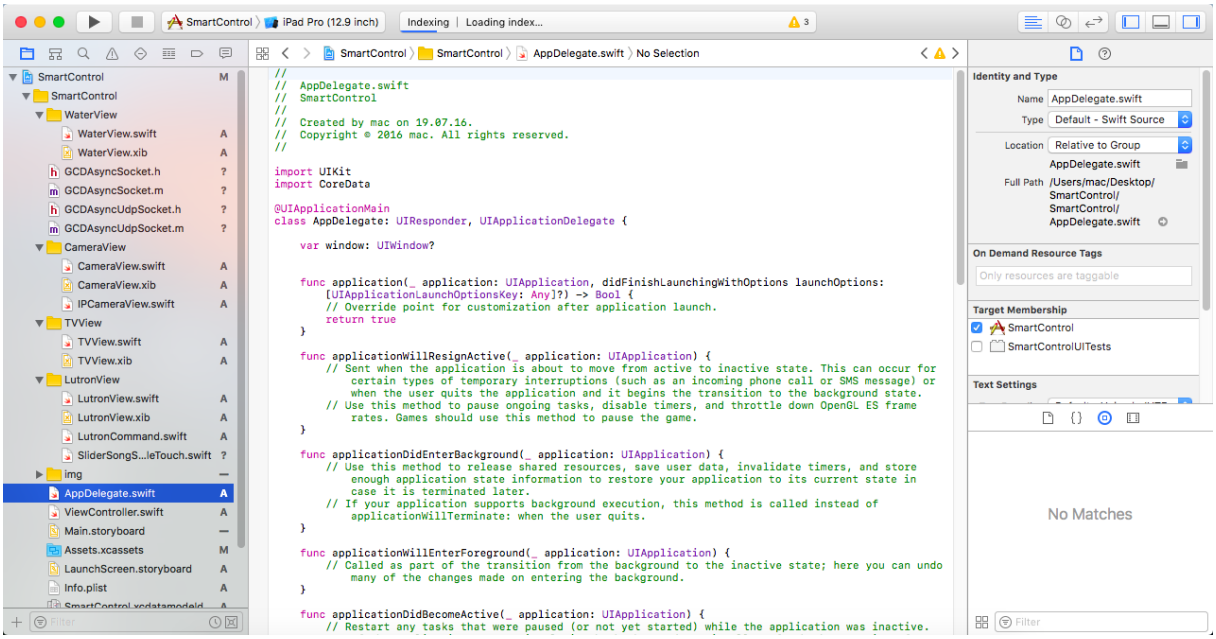

Рисунок 3.13 – Xcode: Редактор коду

Отже як створюється програма та дизайн панелі керування. Для цього потрібно створити новий проєкт Swift.

Основним графічним елементом на сторінках є UIView,це панель, яка може містити всередині любий інструмент, саме цей елемент є головним на вікні UIViewController (рис. 3.14). Далі розглянемо основні атрибути. Далі розглянемо елементи, які використовувалися в написанні даної магістерської роботи:

– UIButton – кнопка, яка при натисканні, або при винекнені жесту який прив'язаний до неї, виконує ті функції, які були прописані;
– UILabel – надпис, яка відтворює заданий текст в заданому форматі, в стандартному вигляді не реагує на події;
– UISlider – елемент, який має безліч станів, який застосовується для рівня освітлення в процентах;
– UIView – елемент, панель, яка може помістити в себе інші елементи, використовується для групування елементів;

– UISwitch – елемент, який має два стани, увімкнено та вимкнено, застосовується як перемикач;

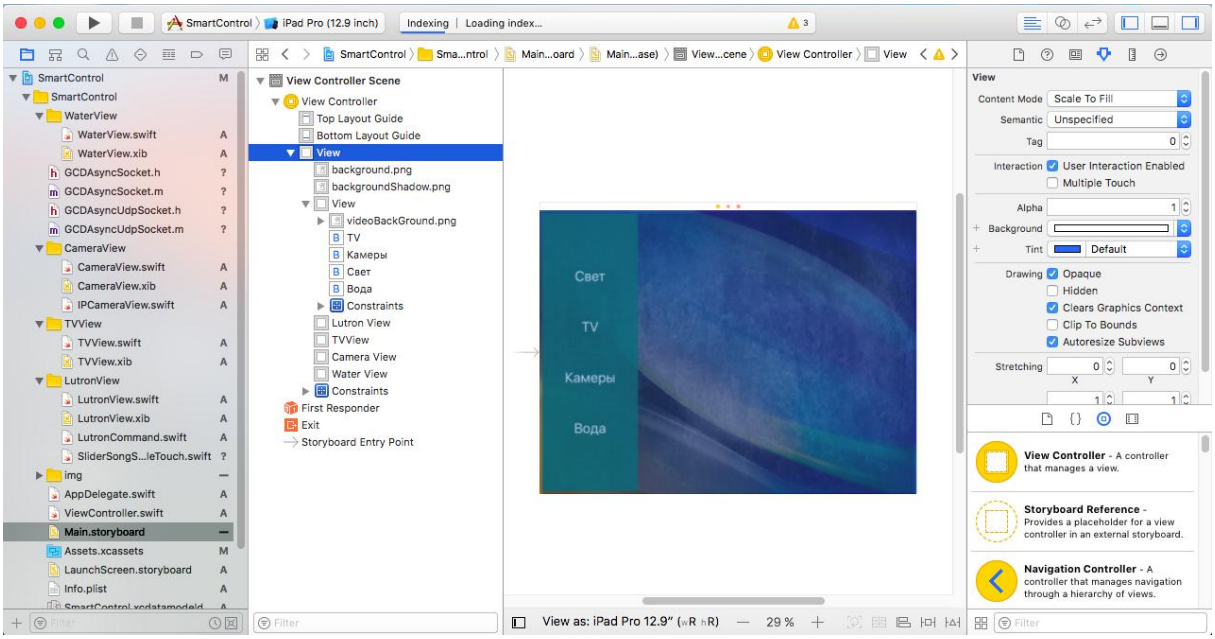

Рисунок 3.14 – Xcode: Редактор інтерфейсу користувача

Елементу UIButton та UISwitch можна стандартно присвоїти слухачів подій, іншим елементам потрібно прописувати їх вручну. Кожна подія прикріплена до певного методу, який виконує певні функції (рис. 3.15). Існує декілька подій, які застосовуються частіше інших:

– TouchUpInside – виникає при натисненні на елементі та віджатті в його межах;
– TouchUpOutside – виникає при натисненні на елементі та віджатті за його межами;
– ValueChanged – виникає при зміні стану елементу;
– EditingDidBegin – виникає при початку редактування елементу;
– EditingDidEnd – виникає при кінці редактування елементу;
– EditingChanged – виникає при редактування елементу.

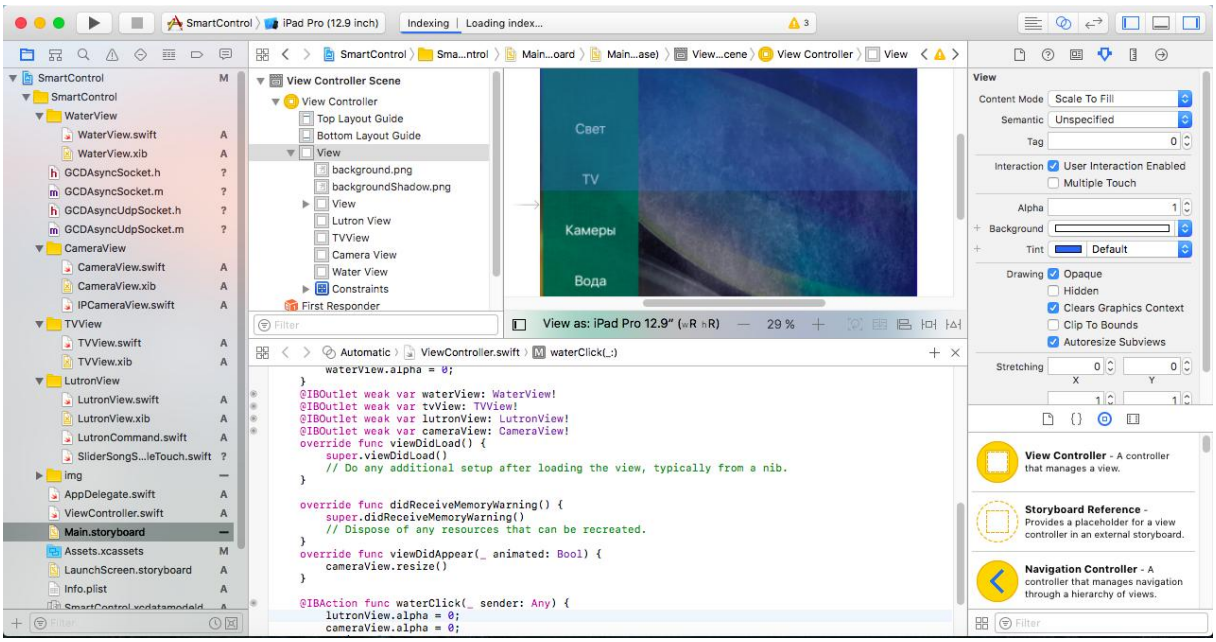

Рисунок 3.15 – Xcode: Вікно Асистента прив'язки подій

В проекті можуть бути два типи сторінок для відображення інформації – сторінка та спливаюче вікно, яке розміщене на цій сторінці. Сторінка (UIViewController) – базовий об'єкт проекту для розміщення графічних об'єктів має два варіанти орієнтації – горизонтальну та вертикальну. В проекті повинна бути хоча б одна сторінка. Сторінка підтримує авто-поворот (переключення орієнтації на панелі з акселерометром. Розмір сторінки завжди рівні розміру екрана панелі та визначають робочу зону проектую. Поява сторінок проводиться без або з анімацією, в залежності від налаштування сторінки, та сторінка може бути прозорою. Одночасно може бути відкрита безліч сторінок.

Спливаюче вікно (UIView) – допоміжний об'єкт проекту для розміщення графічних елементів (рис 3.16). Вона є не обов'язковим елементом в проекті. UIView не може бути горизонтальним або вертикальним, його координати є фіксованими. UIView може мати любий розмір вікна, любу прозорість та положення на сторінці. Одночасно може бути відкрито люба кількість попапів. При авто-повороті всі відкриті UIView перевертаються разом зі сторінкою.

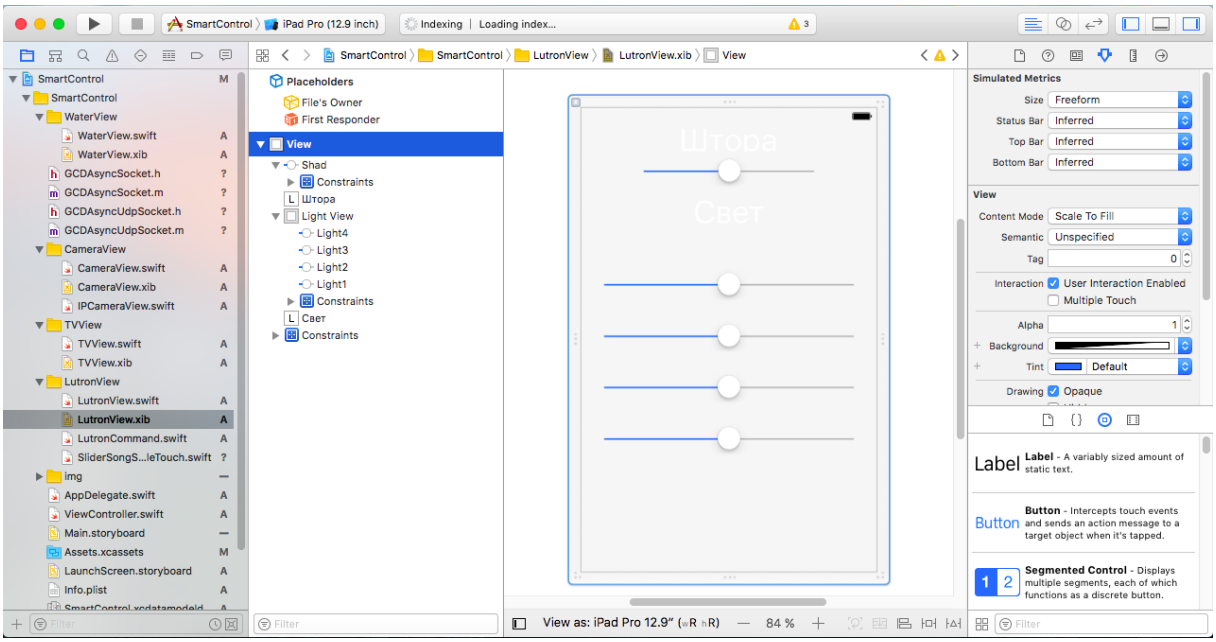

Рисунок 3.16 – Xcode: окреме вікно UIView

Для зв'язку с процесором були використані бібліотеки GCDAsyncSocket, тобто зв'язок за допомогою TCP-протоколу. Створено два окремих з'єднання для зв'язку з процесором Lutron та процесором AMX. Тобто при виході з ладу одного з процесорів, керування буде частково працювати. Далі представлений код який використовуючи GCDAsyncSocket підключается через TCP-протокол до процесора Lutron

```
.
static func setupConnection(_ cv:LutronView) -> GCDAsyncSocket{
        let client = GCDAsyncSocket(delegate: cv, delegateQueue:
DispatchQueue.main)
        do{
            try client?.connect(toHost: "192.168.1.50", onPort:
23, withTimeout: 5.0)
        }catch{}
        return client!;
    }
```

Проект також можна додатково налаштувати під потреби користувача, а також під версію і тип приладу на який буде встановлено додаток (рис 3.17).

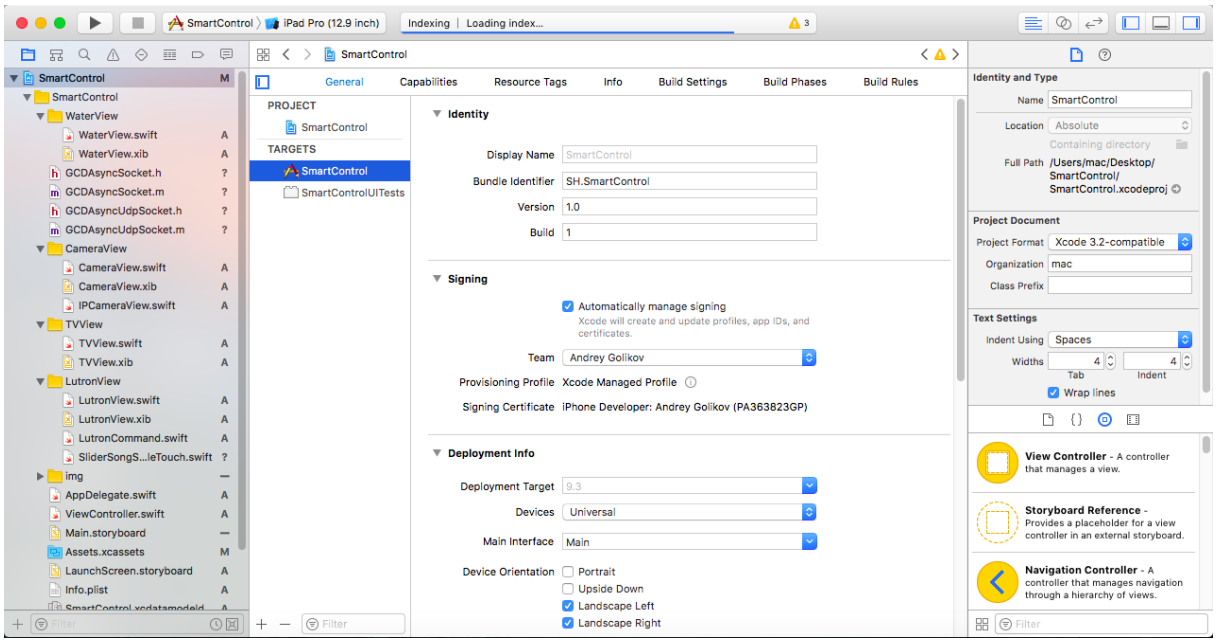

Рисунок 3.17 – Xcode: Головні налаштування проекту

3.4 Створення програми керування для процесору Lutron

Для керування освітленням та шторами з настінних кнопкових панель потрібно створити сценарій керування та обробки подій для процесору. Для цього нам знадобиться HW Illumination (рис. 3.18). Створюємо новий проект. Після цього необхідно добавити всі компоненти схеми освітлення, якими потрібно керувати (рис. 3.19). Далі потрібно встановити всі зв'язки між компонентами, та налаштувати їх (рис. 3.20). На налаштування кнопкової панелі потрібно встановити по окремому сценарію для кожної кнопки (рис. 3.21).

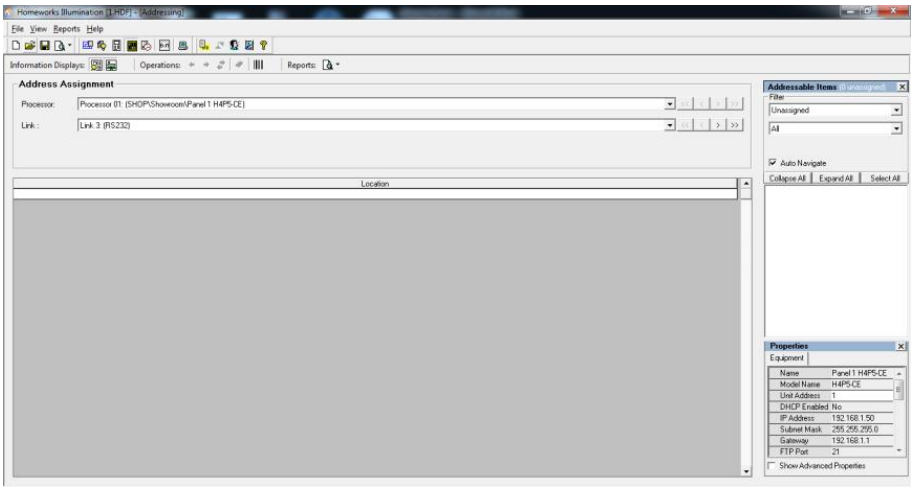

Рисунок 3.18 – Створення проекту в HW Illumination

Також для перевірки команд та налаштувань можна через HW Illumination відкрити термінал та зв'язатися з процесором напряму (рис. 3.22). Спеціальними командами можна перевірити всі компоненти.

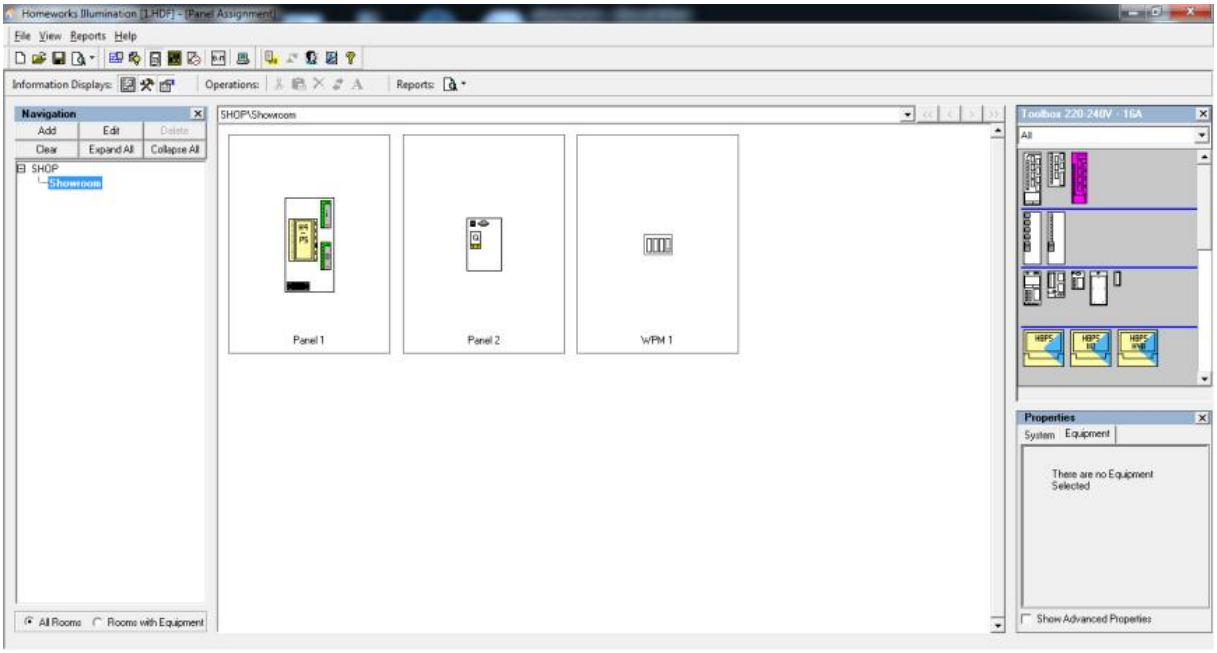

Рисунок 3.19 – Добавлення всіх компонентів в проект

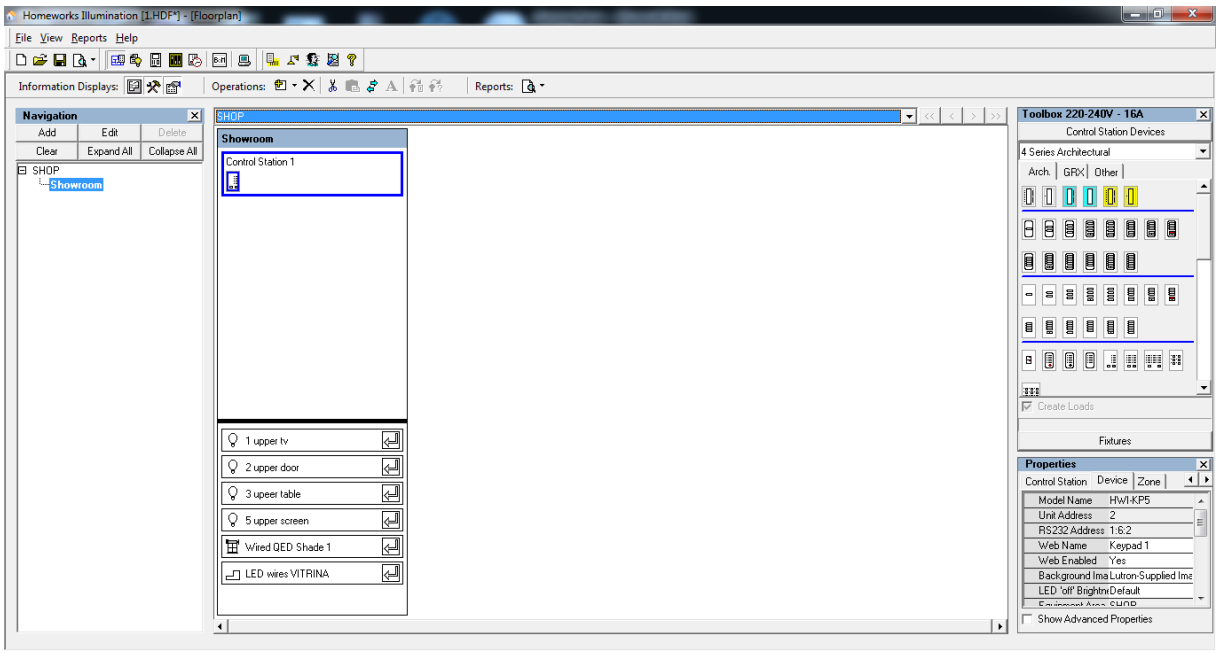

Рисунок 3.20 – Налаштування зв'язків між компонентами

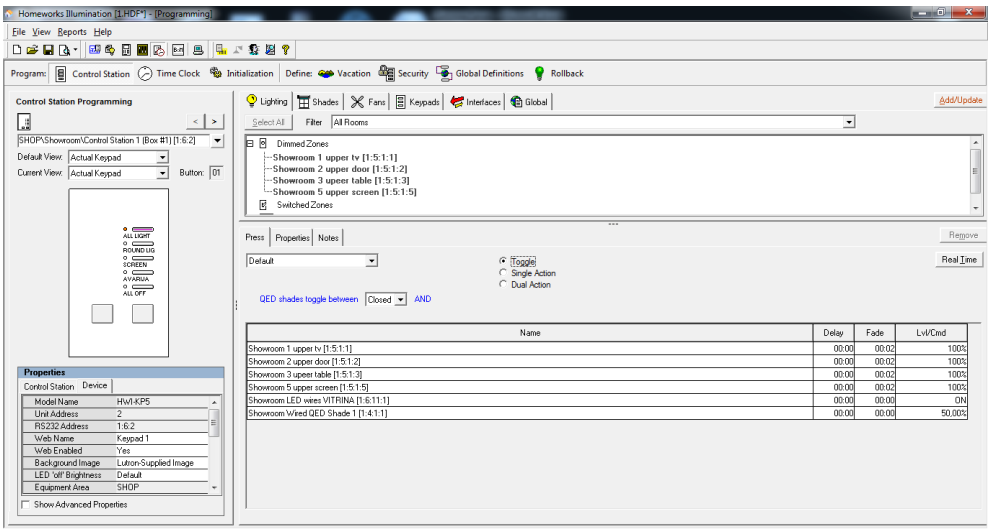

Рисунок 3.21 – Створення всіх сценаріїв під кожну кнопку

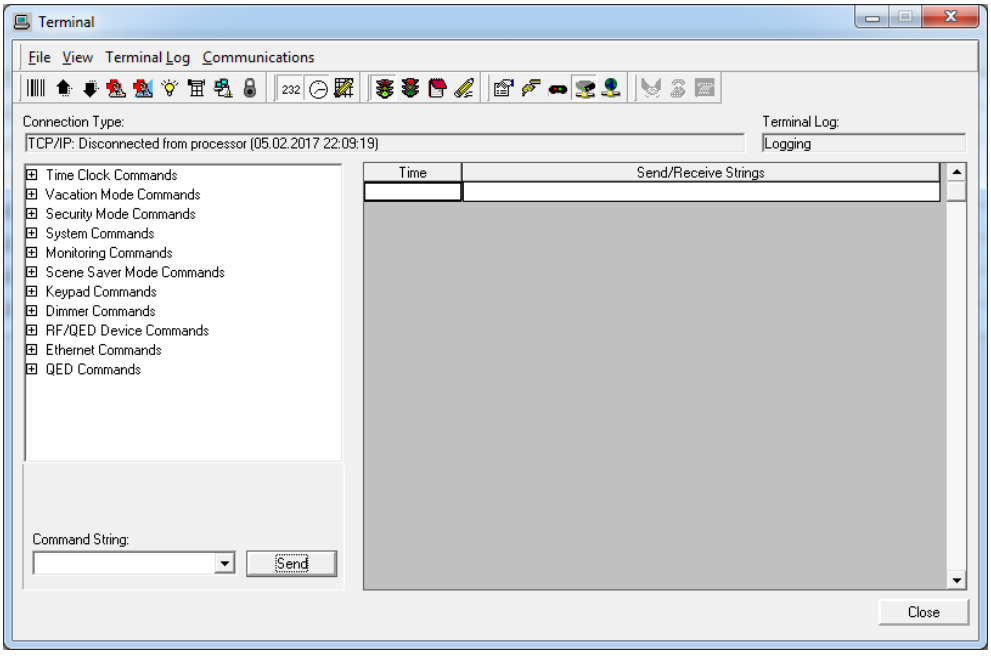

Рисунок 3.22 – Термінал HW Illumination

Для керування водою були використані компоненти системи Нептун. Система Нептун – це професійний захист від протікання води "Нептун" у Вашому домі, а також інноваційне рішення контролю і запобігання затоплення приміщень. Система Нептун використовується на всіх об'єктах житлової та комерційної нерухомості, таких як квартири, будинки, готелі, офіси, магазини та інші. Встановити систему Аквасторож Нептун може звичайний сантехнік. Система Нептун проста в розумінні, тому що має логічний принцип дії. При небажаної витік води, датчик подає сигнал на модуль управління, який в свою

чергу перекриває крани подачі води в приміщення. Автоматично перекриваючи крани Система Аквасторож Нептун 24/7 захищає Ваш будинок від проблем пов'язаних із затопленням.

3.5 Створення програми для панель керування на ОС Windows

Під панеллю керування на ОС Window розуміється персональний комп'ютер. Для цього використовуємо мову програмування C# та середовище MicrosoftVisualStudio. Панель по функціоналу буде повторювати панель керування iOs. Спочатку створюємо новий проект. Далі в редакторі інтерфейсу налаштовуємо та розміщуємо всі графічні елементи. Для цього потрібно перетягнути потрібні елементи на вікно форми. Далі в вікні налаштувань вводимо параметри, які нам потрібні. В даному проекті були використані не стандартні графічні елементи, які були розроблені окремо для цього проекту (рис 3.23).

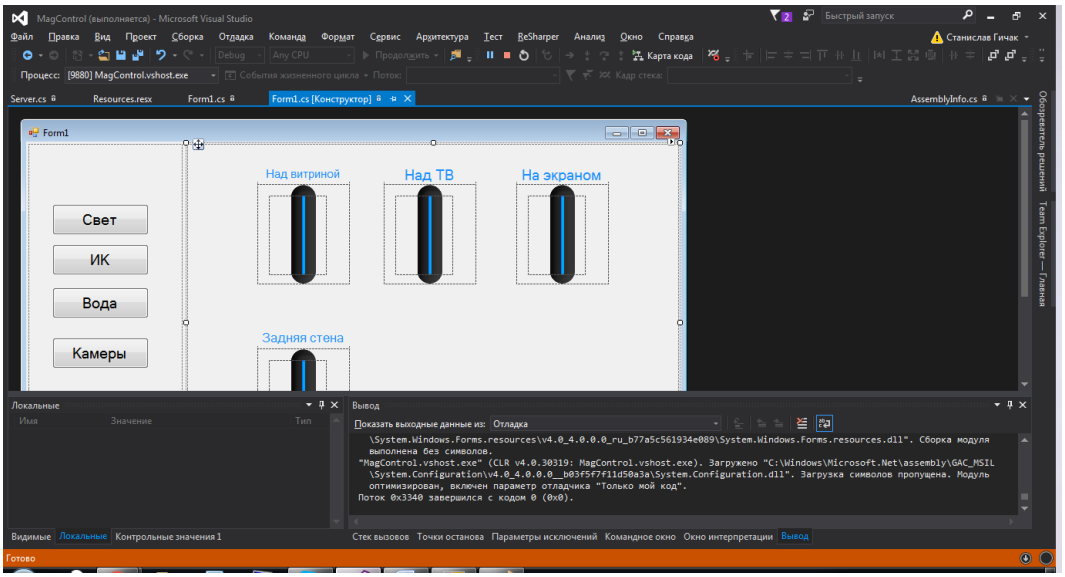

Рисунок 3.23 – Налаштування не стандартних елементів в редакторі

Для керування окремими функціями використовувалися різні панелі, які в певний момент часу могли з'являтися та зникати з екрану. Тобто для 4 виді керування використали 4 панелі:

- керування водою;
- ІЧ-керування;
- керування освітленням;
- перегляд камер спостереження.

Розглянемо окремо кожну панель керування. Першою є керування освітленням (рис. 3.24). На ній представлені 5 слайдерів, які дозволяють керувати рівнем освітлення від електричних ламп, а також рівнем відкритості штори. Ця панель напряму зв'язується з процесором Lutron, та взаємодіє з ним. Це дозволяє системі бути біль стабільною для збоїв, як програмних ,так і збоїв подачі електро енергії.

Наступною панеллю є панель керування ІЧ пристроями (рис. 3.25). Ця панель зв'язується з процесором AMX та вказує коли і яку команду потрібно передати на певний ІЧ випромінювач. Керування здійснюється трьома пристроями, а саме телевізором, проектором та ресивером.

Панель керування водою відповідає за три датчика протікання та один клапан для перекриття води (рис. 3.26). При спрацюванні датчика протікання передається команда на процесор, а він в свою чергу оповіщає панелі керування та, якщо це датчик протікання в ванній кімнаті, він автоматично перекриє воду. Всі стани датчиків відображені на панелях.

Останньою є панель, яка відтворю відео з камери спостереження. Ця панель напряму з'єднується з камерою по її ip-адресі (рис. 3.27).

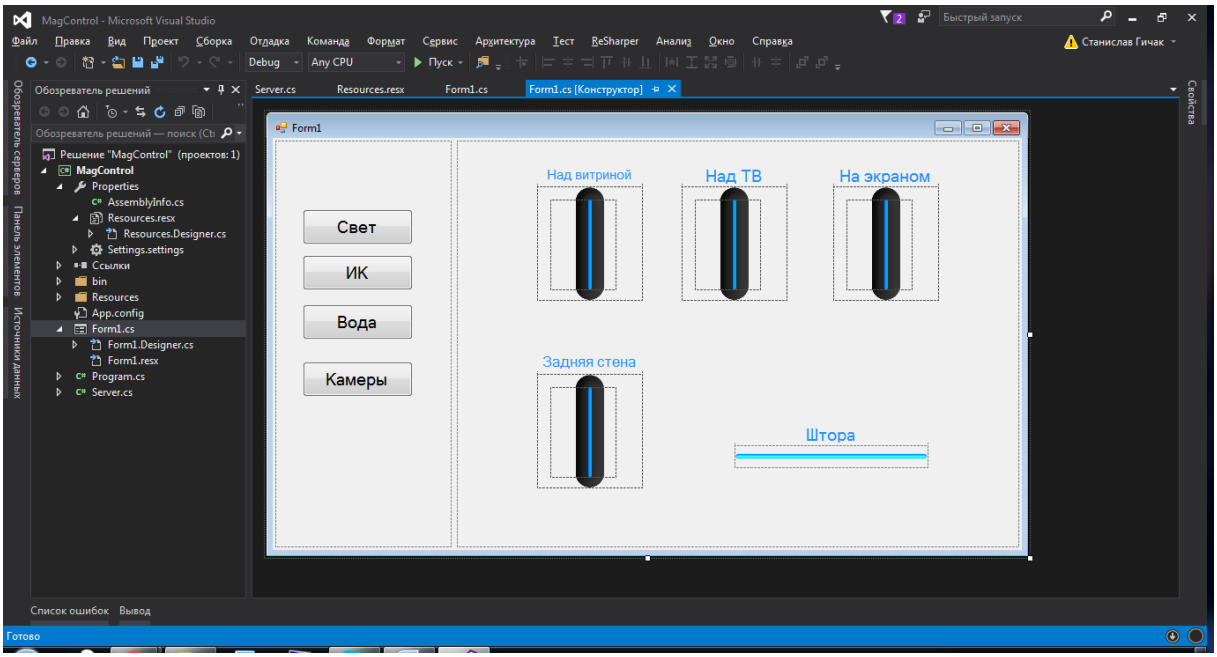

Рисунок 3.24 – Панель керування освітленням

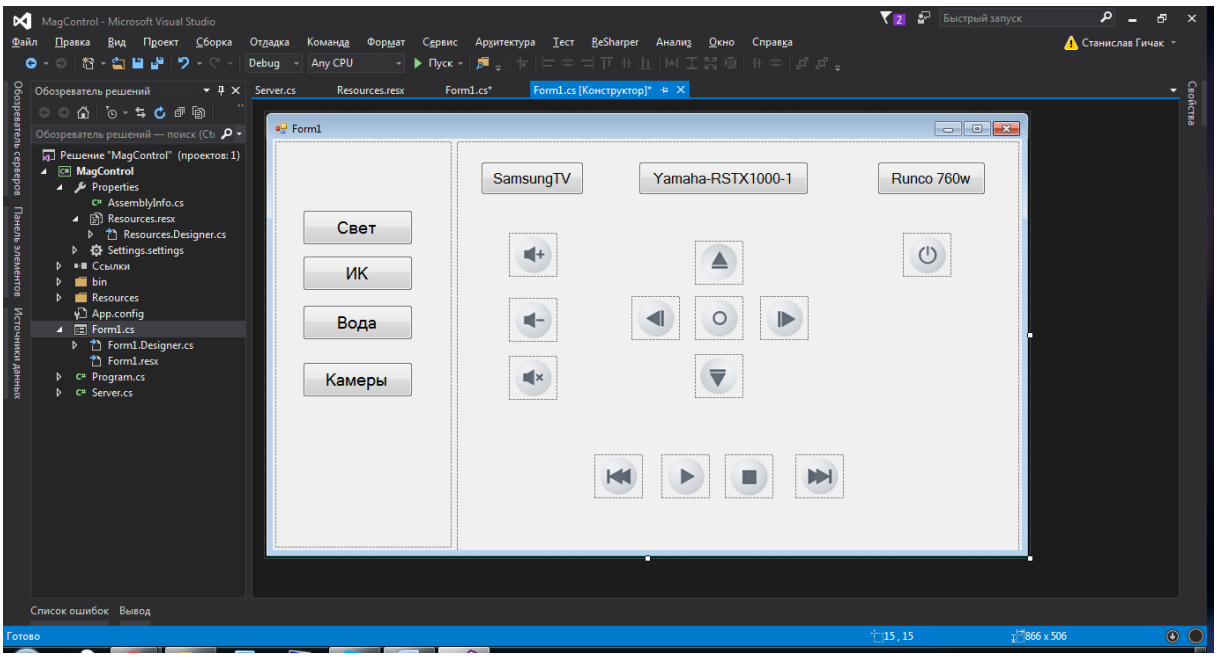

Рисунок 3.25 – Панель керування ІЧ пристроями

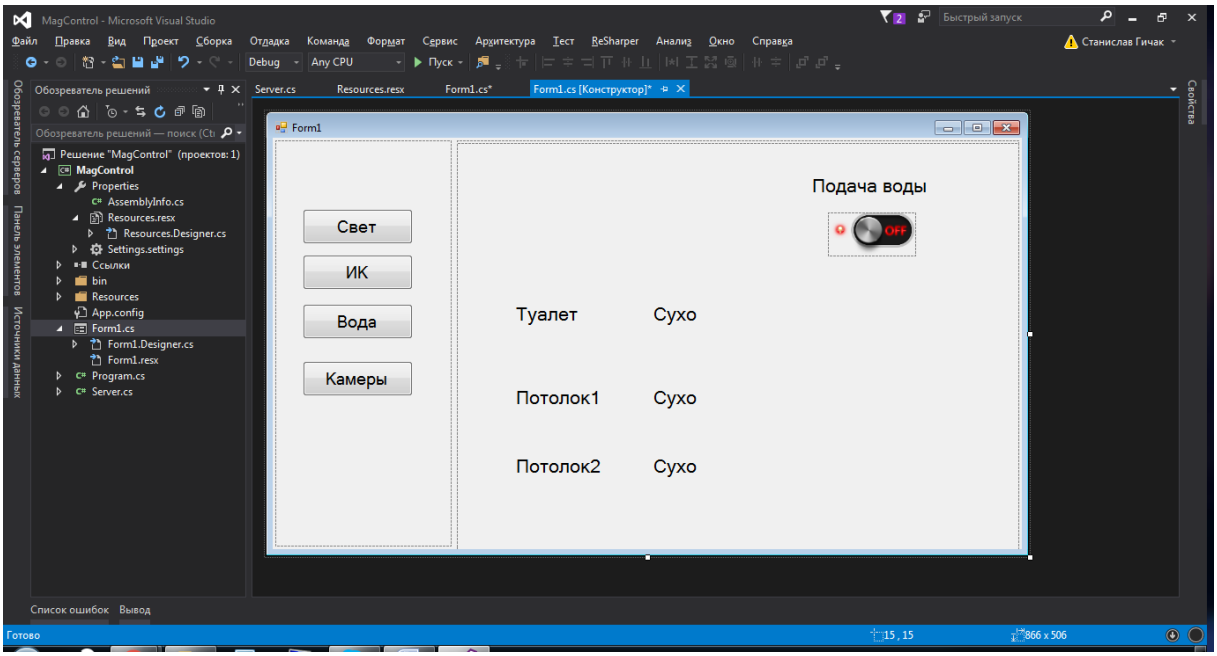

Рисунок 3.26 – Панель керування подачею води

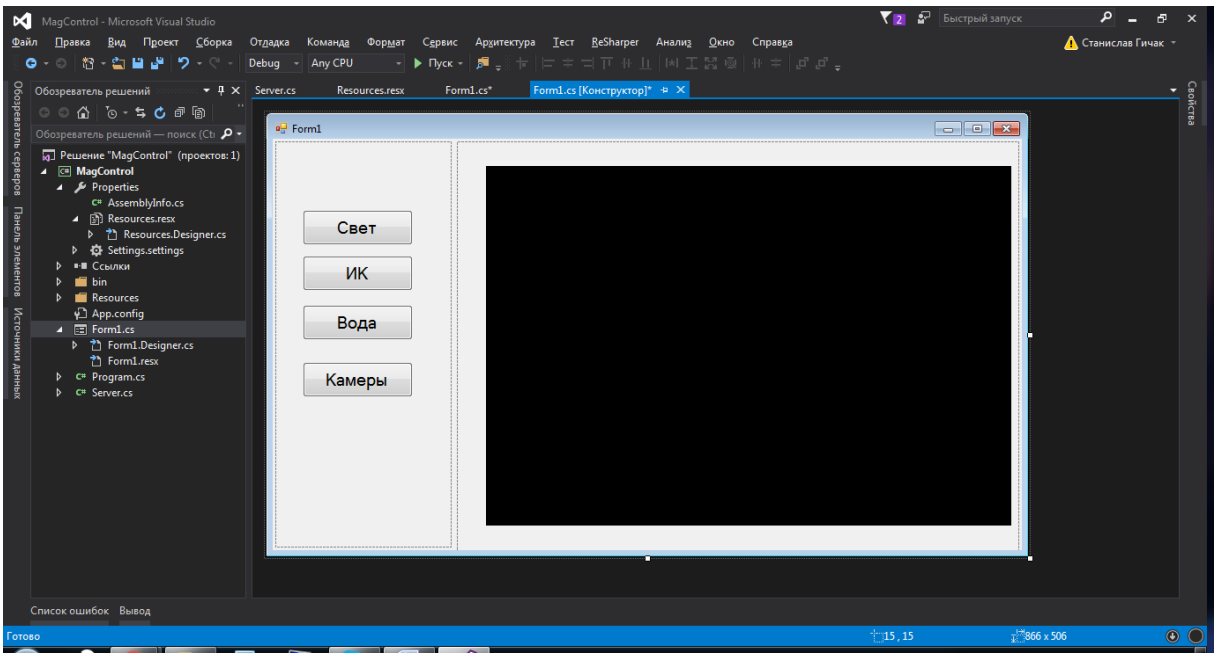

Рисунок 3.27 – Панель відображення відео потоку з камери спостереження

Після створення графічного інтерфейсу потрібно створити зв'язок з мозком "Розумного дому", тобто створити клієнта TCP-протоколу. Він створюється по аналогії тому, як створювався клієнт на панелі керування iOs. Після налагодження всіх процесів зв'язку та реакцій на події. Програма була протестована та відлагоджена.

ВИСНОВКИ

В результаті написання даної магістерської роботи було створено комплекс програм, які дають змогу керувати системою "Розумний будинок". Даний програмний комплекс був встановлений в реальну будівлю – магазин в місті Одеса. Комплекс складається з трьох програм, а саме програма яка встановлюється на панель керування на системі iOs, програма яка встановлюється на контролер "Розумного будинку" AMX та програма яка встановлюється на контролер Lutron, вся система була протестована, а також досі працює.

В даній магістерській роботі перед нами стояла задача автоматизування керування освітленям, аудио- та відео-техніки та керуванням подачою види. В результаті написаний комплекс програм повністю відповідає даним задачам, та максимально ефективно їх вирішує.

Результати роботи доповідались та були опубліковані у вигляді тезисів. Дана магістерська робота апробовувалась на студентській конференції молодих вчених та на Всеукраїнської науково-практичній конференції молодих вчених «Теоретичні та прикладні аспекти застосування інформаційних технологій в галузі природничих наук»,20-22 квітня 2016 р., ОДЕКУ.

При тестуванні програми було використані прилади:

- AMX NI-2100;
- Lutron P5 Processor;
- IPad Mini.

ПЕРЕЛІК ПОСИЛАНЬ

1. Розумний_дім[Електроний ресурс] – Режим доступу: http://uk.wikipedia.org/wiki/Розумний_дім
2. КОМПАС[Електроний ресурс] – Режим доступу: http://uk.wikipedia.org/wiki/ КОМПАС
3. Homestyler – программа для онлайн-проектирования[Електроний ресурс] – Режим доступу:http://kleontev.ru/2010/04/autodesk-homestyler-interior-design-on-line/
4. Бесплатное он-лайн приложение Autodesk Homestyler[Електроний ресурс] – Режим доступу: http://architech.com.ua/content/view/1158/45/lang,ru/
5. Autodesk Homestyler – сервис для проектирования планировки и интерьера помещений[Електроний ресурс] – Режим доступу: http://lifehacker.ru/2012/01/30/autodesk-homestyler/
6. Microsoft_Visual_Studio [Електроний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Microsoft_Visual_Studio
7. C_Sharp[Електроний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/C_Sharp
8. iRidium[Електроний ресурс] – Режим доступу: http://wiki2.iridiummobile.ru/
9. О компании AMX[Електроний ресурс] – Режим доступу: http://www.amx.ru/amx.htm
10. ЧТО ТАКОЕ УМНЫЙ ДОМ[Електроний ресурс] – Режим доступу: http://www.smarthouse.ua/
11. Язык программирования Swift[Електроний ресурс] –Режим доступу: https://habrahabr.ru/post/225841/
12. SWIFT (ЯЗЫК ПРОГРАММИРОВАНИЯ) [Електроний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Swift

Д О Д А Т К И

ДОДАТОК А
ІНТЕРФЕЙС ПРОГРАМИ КЕРУВАННЯ СИСТЕМОЮ "РОЗУМНИЙ БУДИНОК"

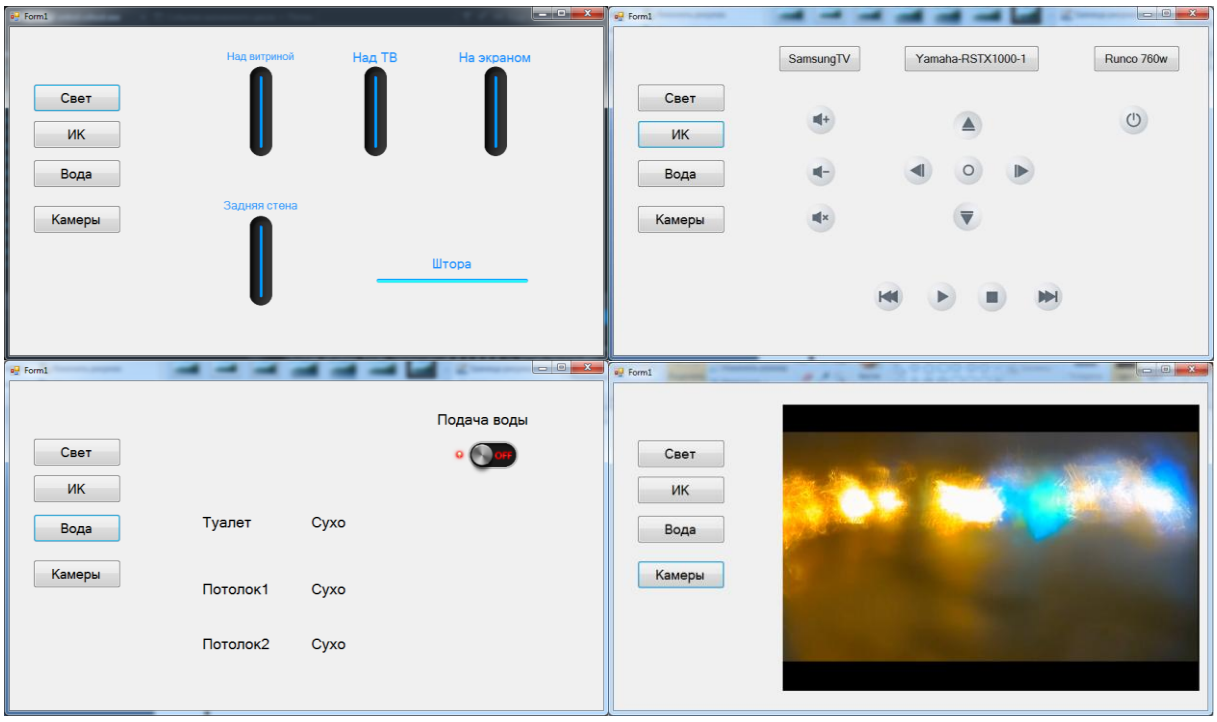

Рисунок А.1 – Скріншоти програми панелі керування на ОС Windows

ДОДАТОК Б
ІНТЕРФЕЙС ПРОГРАМИ КЕРУВАННЯ СИСТЕМОЮ "РОЗУМНИЙ БУДИНОК"

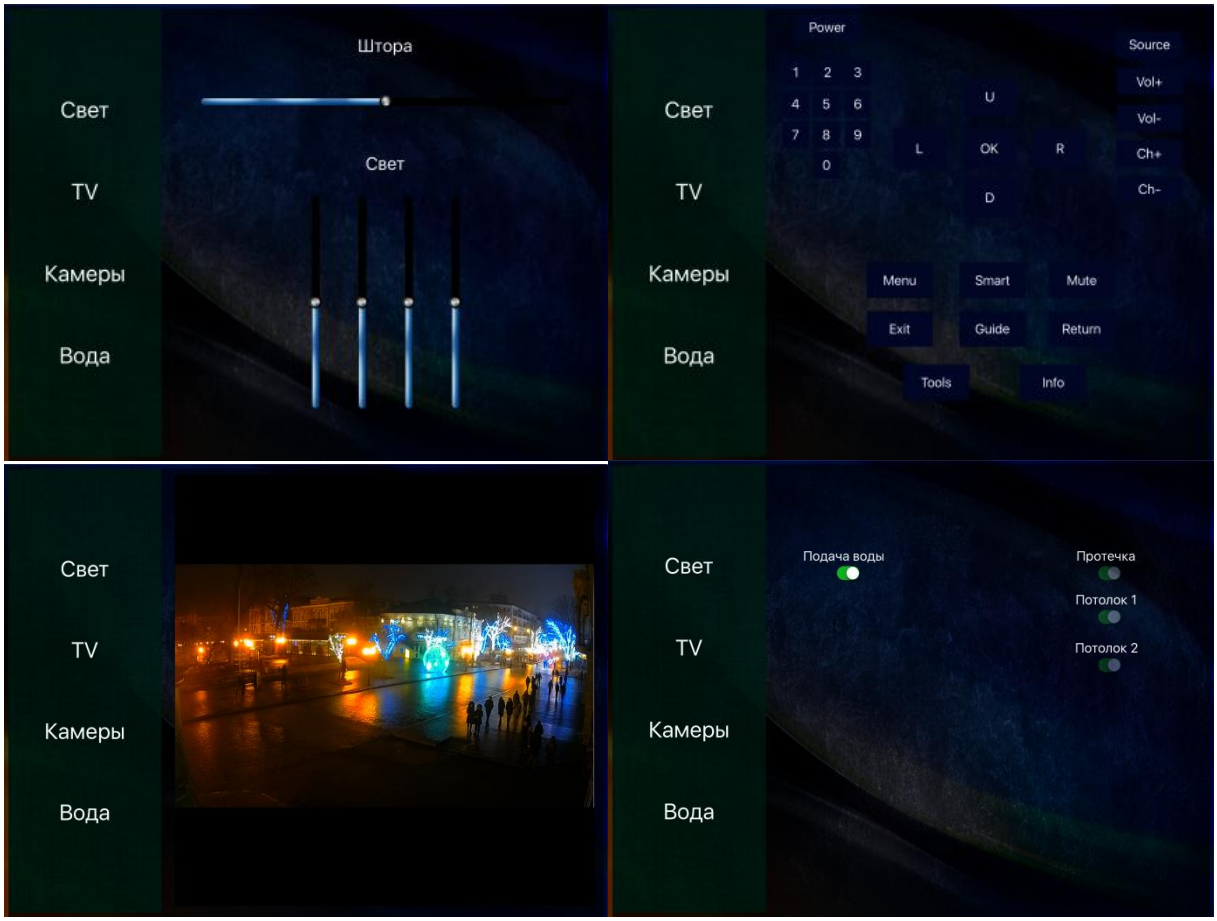

Рисунок Б.1 – Скріншоти програми панелі керування на ОС iOs

ДОДАТОК В
КОД РОЗРОБЛЕНОЇ ПРОГРАМИ ДЛЯ КОНТРОЛЕРА AMX

```
PROGRAM_NAME='Main'

DEFINE_DEVICE

IR1=5001:5:1
IR2=5001:6:1
IR3=5001:7:1
IR4=5001:8:1
Relay=5001:4:1


OUT=5001:9:1

myComp= 0:2:1


DEFINE_CONSTANT
portCl = 1515
ip='192.168.1.50'

DEFINE_TYPE

DEFINE_VARIABLE

DEFINE_LATCHING

DEFINE_MUTUALLY_EXCLUSIVE

DEFINE_START

IP_SERVER_OPEN (myComp.PORT,portCl,ip_tcp)
on[out,4];

DEFINE_EVENT
```

```
DATA_EVENT [myComp]{
    STRING:{
    if(mid_string(Data.TEXT, 0, 1) == 1 ){
        On[IR1,atoi(mid_string(Data.TEXT, 1,
length_string(Data.TEXT)-1))] WAIT 3
{Off[IR1,atoi(mid_string(Data.TEXT, 1,
length_string(Data.TEXT)-1))]}
    }
    if(mid_string(Data.TEXT, 0, 1) == 2 ){
        On[IR2,atoi(mid_string(Data.TEXT, 1,
length_string(Data.TEXT)-1))] WAIT 3
{Off[IR2,atoi(mid_string(Data.TEXT, 1,
length_string(Data.TEXT)-1))]}
    }
    if(mid_string(Data.TEXT, 0, 1) == 3 ){
        On[IR3,atoi(mid_string(Data.TEXT, 1,
length_string(Data.TEXT)-1))] WAIT 3
{Off[IR3,atoi(mid_string(Data.TEXT, 1,
length_string(Data.TEXT)-1))]}
    }

    if(mid_string(Data.TEXT, 0, 1) == 5 ){
        if(mid_string(Data.TEXT, 1, 1) == 1 ){
        ON[Relay,1]
        }else{
        OFF[Relay,1]
        }
    }

    if(mid_string(Data.TEXT, 0, 1) == 'w' ){
        if([OUT,1]){
        SEND_STRING myComp, 'Water 1 ON'
        }else{
        SEND_STRING myComp, 'Water 1 OFF'
        }

        if([OUT,2]){
        SEND_STRING myComp, 'Water 2 ON'
```

```
        }else{
        SEND_STRING myComp, 'Water 2 OFF'
        }

        if([OUT,3]){
        SEND_STRING myComp, 'Water 3 ON'
        }else{
        SEND_STRING myComp, 'Water 3 OFF'
        }

        if([Relay,1]){
        SEND_STRING myComp, 'Relay 1 ON'
        }else{
        SEND_STRING myComp, 'Relay 1 OFF'
        }
    }
    }
}



channel_event[OUT,1]{
    ON:{
    WAIT 5{ OFF[Relay,1]}
    SEND_STRING myComp, 'Water 1 ON'
    }
    OFF:{
    WAIT 5{ ON[Relay,1]}
    SEND_STRING myComp, 'Water 1 OFF'
    }
}

channel_event[OUT,2]{
    ON:{
    SEND_STRING myComp, 'Water 2 ON'
    }
    OFF:{
    SEND_STRING myComp, 'Water 2 OFF'
```

```
    }
}

channel_event[OUT,3]{
    ON:{
    SEND_STRING myComp, 'Water 3 ON'
    }
    OFF:{
    SEND_STRING myComp, 'Water 3 OFF'
    }
}

DATA_EVENT [myComp] {
online:{
    On[OUT,1];
}
offline:{
    off[OUT,1];
    IP_SERVER_CLOSE(myComp.PORT)
    IP_SERVER_OPEN (myComp.PORT,portCl,ip_tcp)
}
}

DEFINE_PROGRAM
```

ДОДАТОК Г
КОД ДОДАТКУНАМОВІ ПРОГРАМУВАННЯ SWIFT ДЛЯ ПАНЕЛІ КЕРУВАННЯ

```
import UIKit

class ViewController: UIViewController {

    @IBAction func tvClick(_ sender: AnyObject) {
        lutronView.alpha = 0;
        cameraView.alpha = 0;
        tvView.alpha = 1;
        waterView.alpha = 0;
    }
    @IBAction func cameraClick(_ sender: AnyObject) {
        lutronView.alpha = 0;
        cameraView.alpha = 1;
        tvView.alpha = 0;
        waterView.alpha = 0;
    }
    @IBAction func lutronClick(_ sender: AnyObject) {
        lutronView.alpha = 1;
        cameraView.alpha = 0;
        tvView.alpha = 0;
        waterView.alpha = 0;
    }
    @IBOutlet weak var waterView: WaterView!
    @IBOutlet weak var tvView: TVView!
    @IBOutlet weak var lutronView: LutronView!
    @IBOutlet weak var cameraView: CameraView!
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the
view, typically from a nib.
    }
    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
```

```
        // Dispose of any resources that can be
recreated.
    }
    override func viewDidAppear(_ animated: Bool) {
        cameraView.resize()
    }

    @IBAction func waterClick(_ sender: Any) {
        lutronView.alpha = 0;
        cameraView.alpha = 0;
        tvView.alpha = 0;
        waterView.alpha = 1;
    }
}


import Foundation
import UIKit
class WaterView:UIView{
    @IBOutlet var view: UIView!
    var root:ViewController?;
    required init(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)!;

        Bundle.main.loadNibNamed("WaterView", owner:
self, options: nil)
        self.addSubview(self.view);

self.view.translatesAutoresizingMaskIntoConstraints =
false;
        NSLayoutConstraint(item: self.view, attribute:
.leading, relatedBy: .equal, toItem: self, attribute:
.leading, multiplier: 1.0, constant: 0.0).isActive =
true
        NSLayoutConstraint(item: self.view, attribute:
.trailing, relatedBy: .equal, toItem: self, attribute:
.trailing, multiplier: 1.0, constant: 0.0).isActive =
true
```

```
        NSLayoutConstraint(item: self.view, attribute:
.top, relatedBy: .equal, toItem: self, attribute: .top,
multiplier: 1.0, constant: 0.0).isActive = true
        NSLayoutConstraint(item: self.view, attribute:
.bottom, relatedBy: .equal, toItem: self, attribute:
.bottom, multiplier: 1.0, constant: 0.0).isActive = true
    }
    @IBOutlet weak var switch1: UISwitch!
    @IBOutlet weak var switch2: UISwitch!
    @IBOutlet weak var switch3: UISwitch!
    @IBOutlet weak var switch4: UISwitch!

    @IBAction func switch1Click(_ sender: Any) {
        if(switch1.isOn){
            root?.tvView.send("51")
        }else{
            root?.tvView.send("50")
        }
    }
}

import Foundation
import UIKit
class TVView: UIView, GCDAsyncSocketDelegate  {

    @IBOutlet var view: UIView!
    required init(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)!;

        Bundle.main.loadNibNamed("TVView", owner: self,
options: nil)


        print("st")
         client = setupConnection();
        print("end")

        send("w");
```

```
        self.addSubview(self.view);

self.view.translatesAutoresizingMaskIntoConstraints =
false;
        NSLayoutConstraint(item: self.view, attribute:
.leading, relatedBy: .equal, toItem: self, attribute:
.leading, multiplier: 1.0, constant: 0.0).isActive =
true
        NSLayoutConstraint(item: self.view, attribute:
.trailing, relatedBy: .equal, toItem: self, attribute:
.trailing, multiplier: 1.0, constant: 0.0).isActive =
true
        NSLayoutConstraint(item: self.view, attribute:
.top, relatedBy: .equal, toItem: self, attribute: .top,
multiplier: 1.0, constant: 0.0).isActive = true
        NSLayoutConstraint(item: self.view, attribute:
.bottom, relatedBy: .equal, toItem: self, attribute:
.bottom, multiplier: 1.0, constant: 0.0).isActive = true
    }


    func setupConnection() -> GCDAsyncSocket{
        let client = GCDAsyncSocket(delegate: self,
delegateQueue: DispatchQueue.main)
        do{
            try client?.connect(toHost: "192.168.1.243",
onPort: 1515, withTimeout: 5.0)
        }catch{print("no connect")}
        return client!;
    }

    func socket(_ socket : GCDAsyncSocket,
didConnectToHost host:String, port p:UInt16) {

    print("Connected to \(host) on port \(p).")

    }
```

```
    var root:ViewController?
    var client:GCDAsyncSocket! = nil
    func socket(_ socket : GCDAsyncSocket!, didRead
data:Data!, withTag tag:Int){
        let readString = String(data: data, encoding:
String.Encoding.utf8)

        if(readString == "Water 1 ON"){
            root?.waterView.switch2.isOn = true;
        }

        if(readString == "Water 1 OFF"){
            root?.waterView.switch2.isOn = false;
        }

        if(readString == "Water 2 ON"){
            root?.waterView.switch3.isOn = true;
        }

        if(readString == "Water 3 ON"){
            root?.waterView.switch4.isOn = true;
        }

        if(readString == "Water 2 OFF"){
            root?.waterView.switch3.isOn = false;
        }

        if(readString == "Water 3 OFF"){
            root?.waterView.switch4.isOn = false;
        }

        if(readString == "Relay 1 ON"){
            root?.waterView.switch1.isOn = true;
        }

        if(readString == "Relay 1 OFF"){
            root?.waterView.switch1.isOn = false;
        }
```

```
        client.readData(withTimeout: -1.0, tag: 0)
    }


    func send(_ msgBytes: String) {
        if(!client.isConnected){
            //client = TCPClient.reconnect(client);
        }

        let s = msgBytes
        print(s);
        let msgData = s.data(using:
String.Encoding.utf8)
        client.write(msgData, withTimeout: -1.0, tag: 0)
        client.readData(withTimeout: -1.0, tag: 0)
    }
    @IBAction func powerClick(_ sender: AnyObject) {
        send("1");
    }
    @IBAction func click1(_ sender: AnyObject) {
        send("2");
    }
    @IBAction func click2(_ sender: AnyObject) {
        send("3");
    }
    @IBAction func click3(_ sender: AnyObject) {
        send("4");
    }
    @IBAction func click4(_ sender: AnyObject) {
        send("5");
    }
    @IBAction func click5(_ sender: AnyObject) {
        send("6");
    }
    @IBAction func click6(_ sender: AnyObject) {
        send("7");
    }
```

```
@IBAction func click7(_ sender: AnyObject) {
    send("8");
}
@IBAction func click8(_ sender: AnyObject) {
    send("9");
}
@IBAction func click9(_ sender: AnyObject) {
    send("10");
}
@IBAction func click0(_ sender: AnyObject) {
    send("11");
}
@IBAction func volclick(_ sender: AnyObject) {
    send("15");
}
@IBAction func volDclick(_ sender: AnyObject) {
    send("16");
}
@IBAction func chclick(_ sender: AnyObject) {
    send("13");
}
@IBAction func chDclick(_ sender: AnyObject) {
    send("14");
}
@IBAction func clickUp(_ sender: AnyObject) {
    send("18");
}
@IBAction func clickDown(_ sender: AnyObject) {
    send("19");
}
@IBAction func clickLeft(_ sender: AnyObject) {
    send("20");
}
@IBAction func clickRight(_ sender: AnyObject) {
    send("21");
}
@IBAction func clickEnter(_ sender: AnyObject) {
    send("12");
```

```
    }
    @IBAction func sourceClick(_ sender: AnyObject) {
        send("33");
    }
    @IBAction func menuClick(_ sender: AnyObject) {
        send("17");
    }
    @IBAction func SmartClick(_ sender: AnyObject) {
        send("35");
    }
    @IBAction func muteClick(_ sender: AnyObject) {
        send("22");
    }
    @IBAction func ExitClick(_ sender: AnyObject) {
        send("27");
    }
    @IBAction func guideClick(_ sender: AnyObject) {
        send("31");
    }
    @IBAction func ReturnClick(_ sender: AnyObject) {
        send("34");
    }
    @IBAction func toolsClick(_ sender: AnyObject) {
        send("36");
    }
    @IBAction func infoClick(_ sender: AnyObject) {
        send("32");
    }

}

import Foundation
import UIKit
import UIKit.UIGestureRecognizerSubclass

class SliderSongSingleTouch : UIGestureRecognizer{
```

```
    override func touchesBegan(_ touches: Set<UITouch>,
with event: UIEvent) {
        let sl:UISlider = self.view as! UISlider;
        sl.value = Float(self.location(in:
self.view).x/(self.view?.frame.width)!)
        sl.sendActions(for:
UIControlEvents.valueChanged)
    }

    override func touchesMoved(_ touches: Set<UITouch>,
with event: UIEvent) {
        let sl:UISlider = self.view as! UISlider;
        sl.value = Float(self.location(in:
self.view).x/(self.view?.frame.width)!)
        sl.sendActions(for:
UIControlEvents.valueChanged)
    }

}



import Foundation
import UIKit

class LutronView: UIView, GCDAsyncSocketDelegate {

    @IBOutlet weak var shad: UISlider!

    @IBAction func shadCh(_ sender: AnyObject) {
        let s = Int((shad.value*10));
        if(s != sd){

//send("L:[1:5:1:1],"+String(Int(light1.value*100))+",:0
");
            send("RTDIM,"+String(100-
s*10)+",5,[1:4:1:1]\n")
```

```
            sd = s;
        }
    }
    @IBOutlet weak var light4: UISlider!
    @IBOutlet weak var light3: UISlider!
    @IBOutlet weak var light2: UISlider!
    @IBOutlet weak var light1: UISlider!
    var sd:Int = 0;
    var sl1:Int = 0;
    var sl2:Int = 0;
    var sl3:Int = 0;
    var sl4:Int = 0;
    @IBAction func light1ch(_ sender: AnyObject) {
        let s = Int((light1.value*10));
        if(s != sl1){

//send("L:[1:5:1:1],"+String(Int(light1.value*100))+",:0
");
            send("RTDIM,"+String(s*10)+",5,[1:5:1:1]\n")
            sl1 = s;
        }
    }
    @IBAction func light4ch(_ sender: AnyObject) {
        let s = Int((light4.value*10));
        if(s != sl4){

//send("L:[1:5:1:1],"+String(Int(light1.value*100))+",:0
");
            send("RTDIM,"+String(s*10)+",5,[1:5:1:2]\n")
            sl4 = s;
        }
    }
    @IBAction func light3ch(_ sender: AnyObject) {
        let s = Int((light3.value*10));
        if(s != sl3){

//send("L:[1:5:1:1],"+String(Int(light1.value*100))+",:0
");
```

```
            send("RTDIM,"+String(s*10)+",5,[1:5:1:3]\n")
                sl3 = s;
            }
        }
        @IBAction func light2ch(_ sender: AnyObject) {
            let s = Int((light2.value*10));
            if(s != sl4){

//send("L:[1:5:1:1],"+String(Int(light1.value*100))+",:0
");
                send("RTDIM,"+String(s*10)+",5,[1:5:1:5]\n")
                sl2 = s;
            }
        }
        @IBOutlet weak var lightView: UIView!
        @IBOutlet var view: UIView!
        required init(coder aDecoder: NSCoder) {
            super.init(coder: aDecoder)!;

            Bundle.main.loadNibNamed("LutronView", owner:
self, options: nil)

            lightView.transform =
CGAffineTransform(rotationAngle: CGFloat(-M_PI_2));


            light1.setThumbImage(UIImage(named:
"sliderThumb.png"), for: UIControlState())
            light1.setMaximumTrackImage(UIImage(named:
"SliderBack.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
            light1.setMinimumTrackImage(UIImage(named:
"SliderTop.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
            light2.setThumbImage(UIImage(named:
"sliderThumb.png"), for: UIControlState())
```

```
        light2.setMaximumTrackImage(UIImage(named:
"SliderBack.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
        light2.setMinimumTrackImage(UIImage(named:
"SliderTop.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
        light3.setThumbImage(UIImage(named:
"sliderThumb.png"), for: UIControlState())
        light3.setMaximumTrackImage(UIImage(named:
"SliderBack.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
        light3.setMinimumTrackImage(UIImage(named:
"SliderTop.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
        light4.setThumbImage(UIImage(named:
"sliderThumb.png"), for: UIControlState())
        light4.setMaximumTrackImage(UIImage(named:
"SliderBack.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
        light4.setMinimumTrackImage(UIImage(named:
"SliderTop.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())

        shad.setThumbImage(UIImage(named:
"sliderThumb.png"), for: UIControlState())
        shad.setMaximumTrackImage(UIImage(named:
"SliderBack.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())
        shad.setMinimumTrackImage(UIImage(named:
"SliderTop.png")?.resizableImage(withCapInsets:
UIEdgeInsetsMake(0, 10, 0, 10)), for: UIControlState())

        let tabGestureRecognizer5:SliderSongSingleTouch
= SliderSongSingleTouch();
        shad.addGestureRecognizer(tabGestureRecognizer5)

        let tabGestureRecognizer1:SliderSongSingleTouch
= SliderSongSingleTouch();
```

```
light1.addGestureRecognizer(tabGestureRecognizer1)

        let tabGestureRecognizer2:SliderSongSingleTouch
= SliderSongSingleTouch();

light2.addGestureRecognizer(tabGestureRecognizer2)

        let tabGestureRecognizer3:SliderSongSingleTouch
= SliderSongSingleTouch();

light3.addGestureRecognizer(tabGestureRecognizer3)

        let tabGestureRecognizer4:SliderSongSingleTouch
= SliderSongSingleTouch();

light4.addGestureRecognizer(tabGestureRecognizer4)


        client = LutronCommand.setupConnection(self);


        send("Iphone,lutron\n");


        self.addSubview(self.view);

self.view.translatesAutoresizingMaskIntoConstraints =
false;
        NSLayoutConstraint(item: self.view, attribute:
.leading, relatedBy: .equal, toItem: self, attribute:
.leading, multiplier: 1.0, constant: 0.0).isActive =
true
        NSLayoutConstraint(item: self.view, attribute:
.trailing, relatedBy: .equal, toItem: self, attribute:
.trailing, multiplier: 1.0, constant: 0.0).isActive =
true
```

```
        NSLayoutConstraint(item: self.view, attribute:
.top, relatedBy: .equal, toItem: self, attribute: .top,
multiplier: 1.0, constant: 0.0).isActive = true
        NSLayoutConstraint(item: self.view, attribute:
.bottom, relatedBy: .equal, toItem: self, attribute:
.bottom, multiplier: 1.0, constant: 0.0).isActive = true
    }

    func socket(_ socket : GCDAsyncSocket,
didConnectToHost host:String, port p:UInt16) {

        print("Connected to \(host) on port \(p).")

    }
    var client:GCDAsyncSocket! = nil
    func socket(_ socket : GCDAsyncSocket!, didRead
data:Data!, withTag tag:Int){
        LutronCommand.recieve(self, data: data);
        client.readData(withTimeout: -1.0, tag: 0)
    }


    func send(_ msgBytes: String) {
        if(!client.isConnected){
            //client = TCPClient.reconnect(client);
        }

        let s = msgBytes
        print(s);
        let msgData = s.data(using:
String.Encoding.utf8)
        client.write(msgData, withTimeout: -1.0, tag: 0)
        client.readData(withTimeout: -1.0, tag: 0)
    }
}


import Foundation
```

```
class LutronCommand{



    static func setupConnection(_ cv:LutronView) ->
GCDAsyncSocket{
        let client = GCDAsyncSocket(delegate: cv,
delegateQueue: DispatchQueue.main)
        do{
            try client?.connect(toHost: "192.168.1.50",
onPort: 23, withTimeout: 5.0)
        }catch{}
        return client!;
    }

    static func recieve(_ cv:LutronView, data:Data){
        let readString = String(data: data, encoding:
String.Encoding.utf8)
        print(readString);
    }
}

import Foundation
import UIKit;
class IPCameraView: UIView, URLSessionDataDelegate {

    var imageView:UIImageView

    var url: URL?;
    var endMarkerData: Data
    var receivedData: NSMutableData
    var dataTask: URLSessionDataTask

    override init(frame: CGRect) {
        self.endMarkerData = Data(bytes:
UnsafePointer<UInt8>([0xFF, 0xD9] as [UInt8]), count: 2)
```

```
        //self.url = URL()
        self.receivedData = NSMutableData()

        self.imageView = UIImageView()

        self.dataTask = URLSessionDataTask()

        super.init(frame: frame)

        self.addSubview(self.imageView)
    }

    required init?(coder aDecoder: NSCoder) {

        fatalError("init(coder:) has not been
implemented")
    }

    deinit{
        self.dataTask.cancel()
    }

    func startWithURL(_ url:URL){
        let session =
Foundation.URLSession(configuration:
URLSessionConfiguration.default, delegate: self,
delegateQueue: nil)


        let request = URLRequest(url: url )

        self.dataTask = session.dataTask(with: request)


        // Initialization code

        self.dataTask.resume()
```

```
        let bounds = self.bounds
        self.imageView.frame = bounds
        self.imageView.contentMode =
UIViewContentMode.scaleAspectFit
    }

    override func layoutSubviews() {
        super.layoutSubviews()
    }

    func pause() {
        self.dataTask.cancel()
    }

    func stop(){
        self.pause()
    }

    func urlSession(_ session: URLSession,
                    dataTask: URLSessionDataTask,
                    didReceive didReceiveData: Data) {

        self.receivedData.append(didReceiveData)

        //NSLog( "Did receive data" )


        let endRange:NSRange =
self.receivedData.range(of: self.endMarkerData, options:
NSData.SearchOptions.anchored, in: NSMakeRange(0,
self.receivedData.length))

        //            NSRange endRange = [_receivedData
rangeOfData:_endMarkerData
        //                options:0
        //                range:NSMakeRange(0,
_receivedData.length)];
```

```
        let endLocation = endRange.location +
endRange.length
        //long long endLocation = endRange.location +
endRange.length;

        if self.receivedData.length >= endLocation {
            let imageData =
self.receivedData.subdata(with: NSMakeRange(0,
endLocation))
            let receivedImage = UIImage(data: imageData)

            DispatchQueue.main.async(execute: {
                self.imageView.image = receivedImage
            })

            //NSLog( "Length: %d", imageData.length )

            self.receivedData = NSData(data:
self.receivedData.subdata(with: NSMakeRange(endLocation,
self.receivedData.length - endLocation))) as!
NSMutableData;


        }

        //            if (_receivedData.length >=
endLocation) {
        //                NSData *imageData =
[_receivedData subdataWithRange:NSMakeRange(0,
endLocation)];
        //                UIImage *receivedImage =
[UIImage imageWithData:imageData];
        //                if (receivedImage) {
        //                    self.image =
receivedImage;
        //                }
        //            }
```

```
    }


    /*
     // Only override drawRect: if you perform custom
drawing.
     // An empty implementation adversely affects
performance during animation.
     override func drawRect(rect: CGRect)
     {
     // Drawing code
     }
     */

}

import Foundation
import UIKit
import AVFoundation
class CameraView: UIView {

    @IBOutlet weak var cam1: UIView!
    @IBOutlet var view: UIView!
    required init(coder aDecoder: NSCoder) {
        super.init(coder: aDecoder)!;

        Bundle.main.loadNibNamed("CameraView", owner:
self, options: nil)
         let url = NSURL(string:
"http://192.168.1.205/index.html")!
        videoPlayer1 = AVPlayer(url: url);
        videoLayer1  = AVPlayerLayer.init(player:
videoPlayer1)
        videoLayer1!.frame = self.cam1.bounds;
        videoLayer1!.videoGravity =
AVLayerVideoGravityResizeAspectFill;
        self.cam1.layer.addSublayer(videoLayer1!);
```

```
        self.videoPlayer1.play();



        self.addSubview(self.view);

self.view.translatesAutoresizingMaskIntoConstraints =
false;
        NSLayoutConstraint(item: self.view, attribute:
.leading, relatedBy: .equal, toItem: self, attribute:
.leading, multiplier: 1.0, constant: 0.0).isActive =
true
        NSLayoutConstraint(item: self.view, attribute:
.trailing, relatedBy: .equal, toItem: self, attribute:
.trailing, multiplier: 1.0, constant: 0.0).isActive =
true
        NSLayoutConstraint(item: self.view, attribute:
.top, relatedBy: .equal, toItem: self, attribute: .top,
multiplier: 1.0, constant: 0.0).isActive = true
        NSLayoutConstraint(item: self.view, attribute:
.bottom, relatedBy: .equal, toItem: self, attribute:
.bottom, multiplier: 1.0, constant: 0.0).isActive = true
    }



    func resize(){
        videoLayer1!.frame = self.cam1.bounds;
    }
    var videoLayer1:AVPlayerLayer?;
    var videoPlayer1 = AVPlayer()
    var videoLayer2:AVPlayerLayer?;
    var videoPlayer2 = AVPlayer()
    var videoLayer3:AVPlayerLayer?;
    var videoPlayer3 = AVPlayer()
}
```

```
import UIKit
import CoreData

@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {

    var window: UIWindow?


    func application(_ application: UIApplication,
didFinishLaunchingWithOptions launchOptions:
[UIApplicationLaunchOptionsKey: Any]?) -> Bool {
        // Override point for customization after
application launch.
        return true
    }

    func applicationWillResignActive(_ application:
UIApplication) {
        // Sent when the application is about to move
from active to inactive state. This can occur for
certain types of temporary interruptions (such as an
incoming phone call or SMS message) or when the user
quits the application and it begins the transition to
the background state.
        // Use this method to pause ongoing tasks,
disable timers, and throttle down OpenGL ES frame rates.
Games should use this method to pause the game.
    }

    func applicationDidEnterBackground(_ application:
UIApplication) {
        // Use this method to release shared resources,
save user data, invalidate timers, and store enough
application state information to restore your
application to its current state in case it is
terminated later.
```

```
        // If your application supports background
execution, this method is called instead of
applicationWillTerminate: when the user quits.
    }

    func applicationWillEnterForeground(_ application:
UIApplication) {
        // Called as part of the transition from the
background to the inactive state; here you can undo many
of the changes made on entering the background.
    }

    func applicationDidBecomeActive(_ application:
UIApplication) {
        // Restart any tasks that were paused (or not
yet started) while the application was inactive. If the
application was previously in the background, optionally
refresh the user interface.
    }

    func applicationWillTerminate(_ application:
UIApplication) {
        // Called when the application is about to
terminate. Save data if appropriate. See also
applicationDidEnterBackground:.
        // Saves changes in the application's managed
object context before the application terminates.
        self.saveContext()
    }

    // MARK: - Core Data stack

    lazy var applicationDocumentsDirectory: URL = {
        // The directory the application uses to store
the Core Data store file. This code uses a directory
named "SH.SmartControl" in the application's documents
Application Support directory.
```

```
        let urls = FileManager.default.urls(for:
.documentDirectory, in: .userDomainMask)
        return urls[urls.count-1]
    }()

    lazy var managedObjectModel: NSManagedObjectModel =
{
        // The managed object model for the application.
This property is not optional. It is a fatal error for
the application not to be able to find and load its
model.
        let modelURL = Bundle.main.url(forResource:
"SmartControl", withExtension: "momd")!
        return NSManagedObjectModel(contentsOf:
modelURL)!
    }()

    lazy var persistentStoreCoordinator:
NSPersistentStoreCoordinator = {
        // The persistent store coordinator for the
application. This implementation creates and returns a
coordinator, having added the store for the application
to it. This property is optional since there are
legitimate error conditions that could cause the
creation of the store to fail.
        // Create the coordinator and store
        let coordinator =
NSPersistentStoreCoordinator(managedObjectModel:
self.managedObjectModel)
        let url =
self.applicationDocumentsDirectory.appendingPathComponen
t("SingleViewCoreData.sqlite")
        var failureReason = "There was an error creating
or loading the application's saved data."
        do {
            try coordinator.addPersistentStore(ofType:
NSSQLiteStoreType, configurationName: nil, at: url,
options: nil)
```

```
        } catch {
            // Report any error we got.
            var dict = [String: AnyObject]()
            dict[NSLocalizedDescriptionKey] = "Failed to
initialize the application's saved data" as AnyObject?
            dict[NSLocalizedFailureReasonErrorKey] =
failureReason as AnyObject?

            dict[NSUnderlyingErrorKey] = error as
NSError
            let wrappedError = NSError(domain:
"YOUR_ERROR_DOMAIN", code: 9999, userInfo: dict)
            // Replace this with code to handle the
error appropriately.
            // abort() causes the application to
generate a crash log and terminate. You should not use
this function in a shipping application, although it may
be useful during development.
            NSLog("Unresolved error \(wrappedError),
\(wrappedError.userInfo)")
            abort()
        }

        return coordinator
    }()

    lazy var managedObjectContext:
NSManagedObjectContext = {
        // Returns the managed object context for the
application (which is already bound to the persistent
store coordinator for the application.) This property is
optional since there are legitimate error conditions
that could cause the creation of the context to fail.
        let coordinator =
self.persistentStoreCoordinator
        var managedObjectContext =
NSManagedObjectContext(concurrencyType:
.mainQueueConcurrencyType)
```

```
        managedObjectContext.persistentStoreCoordinator
= coordinator
        return managedObjectContext
    }()
    // MARK: - Core Data Saving support

    func saveContext () {
        if managedObjectContext.hasChanges {
            do {
                try managedObjectContext.save()
            } catch {
                // Replace this implementation with code
to handle the error appropriately.
                // abort() causes the application to
generate a crash log and terminate. You should not use
this function in a shipping application, although it may
be useful during development.
                let nserror = error as NSError
                NSLog("Unresolved error \(nserror),
\(nserror.userInfo)")
                abort()
            }
        }
    }

}
#import <Foundation/Foundation.h>
#import <Security/Security.h>
#import <Security/SecureTransport.h>
#import <dispatch/dispatch.h>
#import <Availability.h>
#include <sys/socket.h> // AF_INET, AF_INET6
@class GCDAsyncReadPacket;
@class GCDAsyncWritePacket;
@class GCDAsyncSocketPreBuffer;
extern NSString *const GCDAsyncSocketException;
extern NSString *const GCDAsyncSocketErrorDomain;
extern NSString *const GCDAsyncSocketQueueName;
```

```
extern NSString *const GCDAsyncSocketThreadName;
extern NSString *const
GCDAsyncSocketManuallyEvaluateTrust;
#if TARGET_OS_IPHONE
extern NSString *const GCDAsyncSocketUseCFStreamForTLS;
#endif
#define GCDAsyncSocketSSLPeerName     (NSString
*)kCFStreamSSLPeerName
#define GCDAsyncSocketSSLCertificates (NSString
*)kCFStreamSSLCertificates
#define GCDAsyncSocketSSLIsServer     (NSString
*)kCFStreamSSLIsServer
extern NSString *const GCDAsyncSocketSSLPeerID;
extern NSString *const
GCDAsyncSocketSSLProtocolVersionMin;
extern NSString *const
GCDAsyncSocketSSLProtocolVersionMax;
extern NSString *const
GCDAsyncSocketSSLSessionOptionFalseStart;
extern NSString *const
GCDAsyncSocketSSLSessionOptionSendOneByteRecord;
extern NSString *const GCDAsyncSocketSSLCipherSuites;
#if !TARGET_OS_IPHONE
extern NSString *const
GCDAsyncSocketSSLDiffieHellmanParameters;
#endif
#define GCDAsyncSocketLoggingContext 65535
typedef NS_ENUM(NSInteger, GCDAsyncSocketError) {
    GCDAsyncSocketNoError = 0,           // Never used
    GCDAsyncSocketBadConfigError,        // Invalid
configuration
    GCDAsyncSocketBadParamError,         // Invalid
parameter was passed
    GCDAsyncSocketConnectTimeoutError,   // A connect
operation timed out
    GCDAsyncSocketReadTimeoutError,      // A read
operation timed out
```

```
    GCDAsyncSocketWriteTimeoutError,     // A write
operation timed out
    GCDAsyncSocketReadMaxedOutError,     // Reached set
maxLength without completing
    GCDAsyncSocketClosedError,           // The remote
peer closed the connection
    GCDAsyncSocketOtherError,            // Description
provided in userInfo
};
#pragma mark -
@interface GCDAsyncSocket : NSObject
- (id)init;
- (id)initWithSocketQueue:(dispatch_queue_t)sq;
- (id)initWithDelegate:(id)aDelegate
delegateQueue:(dispatch_queue_t)dq;
- (id)initWithDelegate:(id)aDelegate
delegateQueue:(dispatch_queue_t)dq
socketQueue:(dispatch_queue_t)sq;
#pragma mark Configuration
@property (atomic, weak, readwrite) id delegate;
#if OS_OBJECT_USE_OBJC
@property (atomic, strong, readwrite) dispatch_queue_t
delegateQueue;
#else
@property (atomic, assign, readwrite) dispatch_queue_t
delegateQueue;
#endif
- (void)getDelegate:(id *)delegatePtr
delegateQueue:(dispatch_queue_t *)delegateQueuePtr;
- (void)setDelegate:(id)delegate
delegateQueue:(dispatch_queue_t)delegateQueue;
- (void)synchronouslySetDelegate:(id)delegate;
-
(void)synchronouslySetDelegateQueue:(dispatch_queue_t)de
legateQueue;
- (void)synchronouslySetDelegate:(id)delegate
delegateQueue:(dispatch_queue_t)delegateQueue;
```

```
@property (atomic, assign, readwrite,
getter=isIPv4Enabled) BOOL IPv4Enabled;
@property (atomic, assign, readwrite,
getter=isIPv6Enabled) BOOL IPv6Enabled;
@property (atomic, assign, readwrite,
getter=isIPv4PreferredOverIPv6) BOOL
IPv4PreferredOverIPv6;
@property (atomic, strong, readwrite) id userData;
#pragma mark Accepting
- (BOOL)acceptOnPort:(uint16_t)port error:(NSError
**)errPtr;
- (BOOL)acceptOnInterface:(NSString *)interface
port:(uint16_t)port error:(NSError **)errPtr;
- (BOOL)acceptOnUrl:(NSURL *)url error:(NSError
**)errPtr;
#pragma mark Connecting
- (BOOL)connectToHost:(NSString *)host
onPort:(uint16_t)port error:(NSError **)errPtr;
- (BOOL)connectToHost:(NSString *)host
              onPort:(uint16_t)port
         withTimeout:(NSTimeInterval)timeout
               error:(NSError **)errPtr;
- (BOOL)connectToHost:(NSString *)host
              onPort:(uint16_t)port
        viaInterface:(NSString *)interface
         withTimeout:(NSTimeInterval)timeout
               error:(NSError **)errPtr;
- (BOOL)connectToAddress:(NSData *)remoteAddr
error:(NSError **)errPtr;
- (BOOL)connectToAddress:(NSData *)remoteAddr
withTimeout:(NSTimeInterval)timeout error:(NSError
**)errPtr;
- (BOOL)connectToAddress:(NSData *)remoteAddr
           viaInterface:(NSString *)interface
            withTimeout:(NSTimeInterval)timeout
                  error:(NSError **)errPtr;
```

```
- (BOOL)connectToUrl:(NSURL *)url
withTimeout:(NSTimeInterval)timeout error:(NSError
**)errPtr;
#pragma mark Disconnecting
- (void)disconnect;
- (void)disconnectAfterReading;
- (void)disconnectAfterWriting;
- (void)disconnectAfterReadingAndWriting;
#pragma mark Diagnostics
@property (atomic, readonly) BOOL isDisconnected;
@property (atomic, readonly) BOOL isConnected;
@property (atomic, readonly) NSString *connectedHost;
@property (atomic, readonly) uint16_t  connectedPort;
@property (atomic, readonly) NSURL    *connectedUrl;
@property (atomic, readonly) NSString *localHost;
@property (atomic, readonly) uint16_t  localPort;
@property (atomic, readonly) NSData *connectedAddress;
@property (atomic, readonly) NSData *localAddress;
@property (atomic, readonly) BOOL isIPv4;
@property (atomic, readonly) BOOL isIPv6;
@property (atomic, readonly) BOOL isSecure;
#pragma mark Reading
- (void)readDataWithTimeout:(NSTimeInterval)timeout
tag:(long)tag;
- (void)readDataWithTimeout:(NSTimeInterval)timeout
                     buffer:(NSMutableData *)buffer
               bufferOffset:(NSUInteger)offset
                        tag:(long)tag;
- (void)readDataWithTimeout:(NSTimeInterval)timeout
                     buffer:(NSMutableData *)buffer
               bufferOffset:(NSUInteger)offset
                  maxLength:(NSUInteger)length
                        tag:(long)tag;
- (void)readDataToLength:(NSUInteger)length
withTimeout:(NSTimeInterval)timeout tag:(long)tag;
- (void)readDataToLength:(NSUInteger)length
             withTimeout:(NSTimeInterval)timeout
                  buffer:(NSMutableData *)buffer
```

```
            bufferOffset:(NSUInteger)offset
                     tag:(long)tag;
- (void)readDataToData:(NSData *)data
withTimeout:(NSTimeInterval)timeout tag:(long)tag;
- (void)readDataToData:(NSData *)data
           withTimeout:(NSTimeInterval)timeout
                buffer:(NSMutableData *)buffer
          bufferOffset:(NSUInteger)offset
                   tag:(long)tag;
- (void)readDataToData:(NSData *)data
withTimeout:(NSTimeInterval)timeout
maxLength:(NSUInteger)length tag:(long)tag;
- (void)readDataToData:(NSData *)data
           withTimeout:(NSTimeInterval)timeout
                buffer:(NSMutableData *)buffer
          bufferOffset:(NSUInteger)offset
             maxLength:(NSUInteger)length
                   tag:(long)tag;
- (float)progressOfReadReturningTag:(long *)tagPtr
bytesDone:(NSUInteger *)donePtr total:(NSUInteger
*)totalPtr;
#pragma mark Writing
- (void)writeData:(NSData *)data
withTimeout:(NSTimeInterval)timeout tag:(long)tag;
- (float)progressOfWriteReturningTag:(long *)tagPtr
bytesDone:(NSUInteger *)donePtr total:(NSUInteger
*)totalPtr;
#pragma mark Security
- (void)startTLS:(NSDictionary *)tlsSettings;
#pragma mark Advanced
@property (atomic, assign, readwrite) BOOL
autoDisconnectOnClosedReadStream;
-
(void)markSocketQueueTargetQueue:(dispatch_queue_t)socke
tQueuesPreConfiguredTargetQueue;
-
(void)unmarkSocketQueueTargetQueue:(dispatch_queue_t)soc
ketQueuesPreviouslyConfiguredTargetQueue;
```

```
- (void)performBlock:(dispatch_block_t)block;
- (int)socketFD;
- (int)socket4FD;
- (int)socket6FD;
#if TARGET_OS_IPHONE
- (CFReadStreamRef)readStream;
- (CFWriteStreamRef)writeStream;
- (BOOL)enableBackgroundingOnSocket;
#endif
- (SSLContextRef)sslContext;
#pragma mark Utilities
+ (NSMutableArray *)lookupHost:(NSString *)host
port:(uint16_t)port error:(NSError **)errPtr;
+ (NSString *)hostFromAddress:(NSData *)address;
+ (uint16_t)portFromAddress:(NSData *)address;
+ (BOOL)isIPv4Address:(NSData *)address;
+ (BOOL)isIPv6Address:(NSData *)address;
+ (BOOL)getHost:(NSString **)hostPtr port:(uint16_t
*)portPtr fromAddress:(NSData *)address;
+ (BOOL)getHost:(NSString **)hostPtr port:(uint16_t
*)portPtr family:(sa_family_t *)afPtr
fromAddress:(NSData *)address;
+ (NSData *)CRLFData;   // 0x0D0A
+ (NSData *)CRData;     // 0x0D
+ (NSData *)LFData;     // 0x0A
+ (NSData *)ZeroData;   // 0x00
@end
#pragma mark -
@protocol GCDAsyncSocketDelegate
@optional
-
(dispatch_queue_t)newSocketQueueForConnectionFromAddress
:(NSData *)address onSocket:(GCDAsyncSocket *)sock;
- (void)socket:(GCDAsyncSocket *)sock
didAcceptNewSocket:(GCDAsyncSocket *)newSocket;
- (void)socket:(GCDAsyncSocket *)sock
didConnectToHost:(NSString *)host port:(uint16_t)port;
```

```
- (void)socket:(GCDAsyncSocket *)sock
didConnectToUrl:(NSURL *)url;
- (void)socket:(GCDAsyncSocket *)sock
didReadData:(NSData *)data withTag:(long)tag;
- (void)socket:(GCDAsyncSocket *)sock
didReadPartialDataOfLength:(NSUInteger)partialLength
tag:(long)tag;
- (void)socket:(GCDAsyncSocket *)sock
didWriteDataWithTag:(long)tag;
- (void)socket:(GCDAsyncSocket *)sock
didWritePartialDataOfLength:(NSUInteger)partialLength
tag:(long)tag;
- (NSTimeInterval)socket:(GCDAsyncSocket *)sock
shouldTimeoutReadWithTag:(long)tag
elapsed:(NSTimeInterval)elapsed
bytesDone:(NSUInteger)length;
- (NSTimeInterval)socket:(GCDAsyncSocket *)sock
shouldTimeoutWriteWithTag:(long)tag
elapsed:(NSTimeInterval)elapsed
bytesDone:(NSUInteger)length;
- (void)socketDidCloseReadStream:(GCDAsyncSocket *)sock;
- (void)socketDidDisconnect:(GCDAsyncSocket *)sock
withError:(NSError *)err;
- (void)socketDidSecure:(GCDAsyncSocket *)sock;
- (void)socket:(GCDAsyncSocket *)sock
didReceiveTrust:(SecTrustRef)trust
completionHandler:(void (^)(BOOL
shouldTrustPeer))completionHandler;
@end
```

ДОДАТОК Д
КОД ДОДАТКУ НА МОВІ ПРОГРАМУВАННЯ C# ДЛЯ ПАНЕЛІ КЕРУВАННЯ

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
namespace MagControl
{ public partial class Form1 : Form{public Form1(){
InitializeComponent();
            server = new Server(this); server.start();}
        public Server server; bool[] b = new bool[6];
        private void slider1Click(object sender,
EventArgs e) { int pr = Cursor.Position.Y -
pan1D.Location.Y - pib1.Location.Y - pan1.Location.Y -
this.Location.Y - 30;
            pr = pr * 100 / pan1.Size.Height;
            setPercent(0, pr, pib1,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack); }
        private void slider2Click(object sender,
EventArgs e)        {int pr = Cursor.Position.Y -
pan2D.Location.Y - pib2.Location.Y - pan2.Location.Y -
this.Location.Y - 30; pr = pr * 100 / pan2.Size.Height;
            setPercent(0, pr, pib2,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack); }
private void slider3Click(object sender, EventArgs e) {
            int pr = Cursor.Position.Y -
pan3D.Location.Y - pib3.Location.Y - pan3.Location.Y -
this.Location.Y - 30;
            pr = pr * 100 / pan3.Size.Height;
```

```
            setPercent(0, pr, pib3,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack); }
        private void slider4Click(object sender,
EventArgs e) { int pr = Cursor.Position.Y -
pan4D.Location.Y - pib4.Location.Y - pan4.Location.Y -
this.Location.Y - 30;
            pr = pr * 100 / pan4.Size.Height;
            setPercent(0, pr, pib4,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack); }
        public void setPercent(int idRoom, int percent,
PictureBox p, Bitmap pic1, Bitmap pic2) {if (p == null)
            { switch (idRoom) {
                    case 0: p = pib1; break;
                    /*case 1: p = RefectoryPib; break;
                    case 2: p = Hallway1Pib; break;
                    case 3: p = Hallway2Pib; break;
                    case 4: p = LibraryPib; break;
                    case 5: p = CabinetPib; break;
                    case 6: p = WardrobePib; break;
                    case 7: p = Wardrobe2Pib; break;
                    case 8: p = BathroomPib; break;
                    case 9: p = PantryPib; break;*/
                }   } else {
               main.server.write("light," + idRoom + ","
+ (100 - percent) / 10 * 10); }  bool bbb = true;
            for (int h = (int)(0.10 * pic1.Height); h <
(int)(0.90 * pic1.Height) && bbb; h++)            {
                for (int w = (int)(0.40 * pic1.Width); w
< (int)(0.60 * pic1.Width); w++)                {
                    if ((h - (int)(0.10 * pic1.Height))
< percent * pic1.Height * 0.8 / 100)
                    {
                        pic1.SetPixel(w, h,
pic2.GetPixel(w, h));
                    }
```

```
                    else
                        bbb = false;
                }
            }
            p.Image = pic1;
            p.Update();
            if (percent >= 0 && percent <= 100)
            {


                //
main.setOpacityRoom(idRoom, 100 - percent);
            }
        }

        private void slider1Pan_MouseDown(object sender,
MouseEventArgs e)
        {
            b[0] = true;
        }

        private void slider2Pan_MouseDown(object sender,
MouseEventArgs e)
        {
            b[1] = true;
        }

        private void slider3Pan_MouseDown(object sender,
MouseEventArgs e)
        {
            b[2] = true;
        }

        private void slider4Pan_MouseDown(object sender,
MouseEventArgs e)
        {
            b[3] = true;
        }
```

```
        private void slider1Pan_MouseUp(object sender,
MouseEventArgs e)
        {
            b[0] = false;
        }
        private void slider2Pan_MouseUp(object sender,
MouseEventArgs e)
        {
            b[1] = false;
        }
        private void slider3Pan_MouseUp(object sender,
MouseEventArgs e)
        {
            b[2] = false;
        }
        private void slider4Pan_MouseUp(object sender,
MouseEventArgs e)
        {
            b[3] = false;
        }

        private void slider1Pan_MouseMove(object sender,
MouseEventArgs e)
        {
            if (b[0])
            {
                int pr = Cursor.Position.Y -
pan1D.Location.Y - pib1.Location.Y - pan1.Location.Y -
this.Location.Y - 30;
                pr = pr * 100 / pan1.Size.Height;
                setPercent(0, pr, pib1,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack);
            }
        }
```

```
        private void slider2Pan_MouseMove(object sender,
MouseEventArgs e)
        {
            if (b[1])
            {
                int pr = Cursor.Position.Y -
pan2D.Location.Y - pib2.Location.Y - pan2.Location.Y -
this.Location.Y - 30;
                pr = pr * 100 / pan2.Size.Height;
                setPercent(0, pr, pib2,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack);
            }
        }

        private void slider3Pan_MouseMove(object sender,
MouseEventArgs e)
        {
            if (b[2])
            {
                int pr = Cursor.Position.Y -
pan3D.Location.Y - pib3.Location.Y - pan3.Location.Y -
this.Location.Y - 30;
                pr = pr * 100 / pan3.Size.Height;
                setPercent(0, pr, pib3,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack);
            }
        }

        private void slider4Pan_MouseMove(object sender,
MouseEventArgs e)
        {
            if (b[3])
            {
                int pr = Cursor.Position.Y -
pan4D.Location.Y - pib4.Location.Y - pan4.Location.Y -
this.Location.Y - 30;
```

```
                pr = pr * 100 / pan4.Size.Height;
                setPercent(0, pr, pib4,
global::MagControl.Properties.Resources.sliderBack,
global::MagControl.Properties.Resources.sliderBack);
            }
        }



        public void allClose()
        {
            light.Height = 0;
            water.Height = 0;
            cams.Height = 0;
            ir.Height = 0;
        }

        private void button1_Click(object sender,
EventArgs e)
        {
            allClose();
            light.Height = 462;
        }

        private void label4_Click(object sender,
EventArgs e)
        {

        }

        private void pictureBox2_Click(object sender,
EventArgs e)
        {
            if (b[4])
            {
                water1.Image =
global::MagControl.Properties.Resources.sw1;
                //main.server.write("channel,4,0");
```

```
            }
            else
            {
                water1.Image =
global::MagControl.Properties.Resources.sw2;
                //main.server.write("channel,4,1");
            }
            b[4] = !b[4];
        }

        private void button3_Click(object sender,
EventArgs e)
        {
            allClose();
            water.Height = 462;
        }

        private void button4_Click(object sender,
EventArgs e)
        {
            allClose();
            cams.Height = 462;
        }

        private void button2_Click(object sender,
EventArgs e)
        {
            allClose();
            ir.Height = 462;
        }

        private void Refectory3Pib_MouseDown(object
sender, MouseEventArgs e)
        {
            b[5] = true;
        }
```

```
        private void Refectory3Pib_MouseUp(object
sender, MouseEventArgs e)
        {
            b[5] = false;
        }

        private void Refectory3Pib_MouseMove(object
sender, MouseEventArgs e)
        {
            if (b[5])
            {
                int pr = Cursor.Position.X -
this.Location.X - Refectory3.Location.X -
light.Location.X-7;
                pr = pr * 100 /
Refectory3Pib.Size.Width;
                setPercentH(8, pr, Refectory3Pib,
global::MagControl.Properties.Resources.sliderHor1,
global::MagControl.Properties.Resources.sliderHor2);
            }
        }

        private void Refectory3Pib_Click(object sender,
EventArgs e)
        {
            int pr = Cursor.Position.X - this.Location.X
- Refectory3.Location.X - light.Location.X-7;
            pr = pr * 100 / Refectory3Pib.Size.Width;
            setPercentH(8, pr, Refectory3Pib,
global::MagControl.Properties.Resources.sliderHor1,
global::MagControl.Properties.Resources.sliderHor2);
        }

        public void setPercentH(int idShade, int
percent, PictureBox p, Bitmap pic1, Bitmap pic2)
        {
            if (p == null)
            {
```

```
                switch (idShade)
                {
                    case 0: p = Refectory3Pib; break;
                }
            }
            else
            {
//                main.server.write("shade," + idShade +
"," + (percent) / 10 * 10);
            }
            //percent = 100 - percent;
            bool bbb = true;
            for (int w = 0; w < pic1.Width && bbb; w++)
            {
                for (int h = 0; h < pic1.Height; h++)
                {
                    if (w < percent * pic1.Width / 100)
                    {
                        pic1.SetPixel(w, h,
pic2.GetPixel(w, h));
                    }
                    else
                        bbb = false;
                }
            }
            p.Image = pic1;
            p.Update();




//            main.setOpacityShade(idRoom, percent);
        }
    }
}
namespace MagControl
{
    partial class Form1
```

```
    {
/// <summary>
        /// Обязательная переменная конструктора.
        /// </summary>
private System.ComponentModel.IContainer components =
null;

/// <summary>
        /// Освободить все используемые ресурсы.
        /// </summary>
        /// <param name="disposing">истинно, если управ-
ляемый ресурс должен быть удален; иначе ложно.</param>
protected override void Dispose(bool disposing)
        {
            if (disposing && (components != null))
            {
                components.Dispose();
}
            base.Dispose(disposing);
        }

        #region Код, автоматически созданный конструкто-
ром форм Windows

        /// <summary>
        /// Требуемый метод для поддержки конструктора —
не изменяйте
        /// содержимое этого метода с помощью редактора
кода.
/// </summary>
        private void InitializeComponent()
        {

System.ComponentModel.ComponentResourceManager resources
= new
System.ComponentModel.ComponentResourceManager(typeof(Fo
rm1));
```

```
            this.panel1 = new
System.Windows.Forms.Panel();
            this.button4 = new
System.Windows.Forms.Button();
            this.button3 = new
System.Windows.Forms.Button();
            this.button2 = new
System.Windows.Forms.Button();
            this.button1 = new
System.Windows.Forms.Button();
            this.light = new
System.Windows.Forms.Panel();
            this.Refectory3 = new
System.Windows.Forms.Panel();
            this.Refectory3Pib = new
System.Windows.Forms.PictureBox();
            this.Refectory3Lab = new
System.Windows.Forms.Label();
            this.pan4D = new
System.Windows.Forms.Panel();
            this.lab4 = new
System.Windows.Forms.Label();
            this.pib4 = new
System.Windows.Forms.PictureBox();
            this.pan4 = new
System.Windows.Forms.Panel();
            this.pan3D = new
System.Windows.Forms.Panel();
            this.lab3 = new
System.Windows.Forms.Label();
            this.pib3 = new
System.Windows.Forms.PictureBox();
            this.pan3 = new
System.Windows.Forms.Panel();
            this.pan2D = new
System.Windows.Forms.Panel();
            this.lab2 = new
System.Windows.Forms.Label();
```

```
            this.pib2 = new
System.Windows.Forms.PictureBox();
            this.pan2 = new
System.Windows.Forms.Panel();
            this.pan1D = new
System.Windows.Forms.Panel();
            this.lab1 = new
System.Windows.Forms.Label();
            this.pib1 = new
System.Windows.Forms.PictureBox();
            this.pan1 = new
System.Windows.Forms.Panel();
            this.water = new
System.Windows.Forms.Panel();
            this.label5 = new
System.Windows.Forms.Label();
            this.label6 = new
System.Windows.Forms.Label();
            this.label7 = new
System.Windows.Forms.Label();
            this.label4 = new
System.Windows.Forms.Label();
            this.label3 = new
System.Windows.Forms.Label();
            this.label2 = new
System.Windows.Forms.Label();
            this.label1 = new
System.Windows.Forms.Label();
            this.water1 = new
System.Windows.Forms.PictureBox();
            this.cams = new
System.Windows.Forms.Panel();
            this.axWindowsMediaPlayer1 = new
AxWMPLib.AxWindowsMediaPlayer();
            this.ir = new System.Windows.Forms.Panel();
            this.pictureBox13 = new
System.Windows.Forms.PictureBox();
```

```
            this.pictureBox12 = new
System.Windows.Forms.PictureBox();
            this.pictureBox11 = new
System.Windows.Forms.PictureBox();
            this.pictureBox10 = new
System.Windows.Forms.PictureBox();
            this.pictureBox9 = new
System.Windows.Forms.PictureBox();
            this.pictureBox8 = new
System.Windows.Forms.PictureBox();
            this.pictureBox7 = new
System.Windows.Forms.PictureBox();
            this.pictureBox6 = new
System.Windows.Forms.PictureBox();
            this.pictureBox5 = new
System.Windows.Forms.PictureBox();
            this.pictureBox4 = new
System.Windows.Forms.PictureBox();
            this.pictureBox3 = new
System.Windows.Forms.PictureBox();
            this.pictureBox2 = new
System.Windows.Forms.PictureBox();
            this.pictureBox1 = new
System.Windows.Forms.PictureBox();
            this.button7 = new
System.Windows.Forms.Button();
            this.button6 = new
System.Windows.Forms.Button();
            this.button5 = new
System.Windows.Forms.Button();
            this.panel1.SuspendLayout();
            this.light.SuspendLayout();
            this.Refectory3.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.Refecto
ry3Pib)).BeginInit();
            this.pan4D.SuspendLayout();
```

```
((System.ComponentModel.ISupportInitialize)(this.pib4)).
BeginInit();
            this.pib4.SuspendLayout();
            this.pan3D.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.pib3)).
BeginInit();
            this.pib3.SuspendLayout();
            this.pan2D.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.pib2)).
BeginInit();
            this.pib2.SuspendLayout();
            this.pan1D.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.pib1)).
BeginInit();
            this.pib1.SuspendLayout();
            this.water.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.water1)
).BeginInit();
            this.cams.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.axWindo
wsMediaPlayer1)).BeginInit();
            this.ir.SuspendLayout();

((System.ComponentModel.ISupportInitialize)(this.picture
Box13)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box12)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box11)).BeginInit();
```

```
((System.ComponentModel.ISupportInitialize)(this.picture
Box10)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box9)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box8)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box7)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box6)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box5)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box4)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box3)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box2)).BeginInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box1)).BeginInit();
            this.SuspendLayout();
            //
            // panel1
            //
            this.panel1.Controls.Add(this.button4);
            this.panel1.Controls.Add(this.button3);
            this.panel1.Controls.Add(this.button2);
            this.panel1.Controls.Add(this.button1);
```

```
            this.panel1.Location = new
System.Drawing.Point(3, 3);
            this.panel1.Name = "panel1";
            this.panel1.Size = new
System.Drawing.Size(200, 462);
            this.panel1.TabIndex = 0;
            //
            // button4
            //
            this.button4.Font = new
System.Drawing.Font("Microsoft Sans Serif", 14.25F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button4.Location = new
System.Drawing.Point(31, 251);
            this.button4.Name = "button4";
            this.button4.Size = new
System.Drawing.Size(125, 40);
            this.button4.TabIndex = 3;
            this.button4.Text = "Камеры";
            this.button4.UseVisualStyleBackColor = true;
            this.button4.Click += new
System.EventHandler(this.button4_Click);
            //
            // button3
            //
            this.button3.Font = new
System.Drawing.Font("Microsoft Sans Serif", 14.25F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button3.Location = new
System.Drawing.Point(31, 186);
            this.button3.Name = "button3";
            this.button3.Size = new
System.Drawing.Size(125, 40);
            this.button3.TabIndex = 2;
            this.button3.Text = "Вода";
            this.button3.UseVisualStyleBackColor = true;
```

```
            this.button3.Click += new
System.EventHandler(this.button3_Click);
            //
            // button2
            //
            this.button2.Font = new
System.Drawing.Font("Microsoft Sans Serif", 14.25F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button2.Location = new
System.Drawing.Point(31, 130);
            this.button2.Name = "button2";
            this.button2.Size = new
System.Drawing.Size(125, 40);
            this.button2.TabIndex = 1;
            this.button2.Text = "ИК";
            this.button2.UseVisualStyleBackColor = true;
            this.button2.Click += new
System.EventHandler(this.button2_Click);
            //
            // button1
            //
            this.button1.Font = new
System.Drawing.Font("Microsoft Sans Serif", 14.25F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button1.Location = new
System.Drawing.Point(31, 78);
            this.button1.Name = "button1";
            this.button1.Size = new
System.Drawing.Size(125, 40);
            this.button1.TabIndex = 0;
            this.button1.Text = "Свет";
            this.button1.UseVisualStyleBackColor = true;
            this.button1.Click += new
System.EventHandler(this.button1_Click);
            //
            // light
```

```
            //
            this.light.Controls.Add(this.Refectory3);
            this.light.Controls.Add(this.pan4D);
            this.light.Controls.Add(this.pan3D);
            this.light.Controls.Add(this.pan2D);
            this.light.Controls.Add(this.pan1D);
            this.light.Location = new
System.Drawing.Point(209, 3);
            this.light.Name = "light";
            this.light.Size = new
System.Drawing.Size(638, 462);
            this.light.TabIndex = 1;
            //
            // Refectory3
            //

this.Refectory3.Controls.Add(this.Refectory3Pib);

this.Refectory3.Controls.Add(this.Refectory3Lab);
            this.Refectory3.Location = new
System.Drawing.Point(315, 322);
            this.Refectory3.Name = "Refectory3";
            this.Refectory3.Size = new
System.Drawing.Size(220, 50);
            this.Refectory3.TabIndex = 11;
            //
            // Refectory3Pib
            //
            this.Refectory3Pib.Anchor =
((System.Windows.Forms.AnchorStyles)((((System.Windows.F
orms.AnchorStyles.Top |
System.Windows.Forms.AnchorStyles.Bottom)
            | System.Windows.Forms.AnchorStyles.Left)
            |
System.Windows.Forms.AnchorStyles.Right)));
            this.Refectory3Pib.Image =
global::MagControl.Properties.Resources.sliderHor2;
```

```
            this.Refectory3Pib.Location = new
System.Drawing.Point(0, 24);
            this.Refectory3Pib.Name = "Refectory3Pib";
            this.Refectory3Pib.Size = new
System.Drawing.Size(220, 26);
            this.Refectory3Pib.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.Refectory3Pib.TabIndex = 3;
            this.Refectory3Pib.TabStop = false;
            this.Refectory3Pib.Click += new
System.EventHandler(this.Refectory3Pib_Click);
            this.Refectory3Pib.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.Refectory3Pi
b_MouseDown);
            this.Refectory3Pib.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.Refectory3Pi
b_MouseMove);
            this.Refectory3Pib.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.Refectory3Pi
b_MouseUp);
            //
            // Refectory3Lab
            //
            this.Refectory3Lab.Dock =
System.Windows.Forms.DockStyle.Top;
            this.Refectory3Lab.Font = new
System.Drawing.Font("Microsoft Sans Serif", 14.25F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.Refectory3Lab.ForeColor =
System.Drawing.Color.DodgerBlue;
            this.Refectory3Lab.Location = new
System.Drawing.Point(0, 0);
            this.Refectory3Lab.Name = "Refectory3Lab";
            this.Refectory3Lab.RightToLeft =
System.Windows.Forms.RightToLeft.No;
            this.Refectory3Lab.Size = new
System.Drawing.Size(220, 21);
```

```
            this.Refectory3Lab.TabIndex = 2;
            this.Refectory3Lab.Text = "Штора";
            this.Refectory3Lab.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // pan4D
            //
            this.pan4D.BackColor =
System.Drawing.Color.Transparent;
            this.pan4D.Controls.Add(this.lab4);
            this.pan4D.Controls.Add(this.pib4);
            this.pan4D.Location = new
System.Drawing.Point(91, 241);
            this.pan4D.Name = "pan4D";
            this.pan4D.Size = new
System.Drawing.Size(120, 154);
            this.pan4D.TabIndex = 2;
            //
            // lab4
            //
            this.lab4.Dock =
System.Windows.Forms.DockStyle.Top;
            this.lab4.Font = new
System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.lab4.ForeColor =
System.Drawing.Color.DodgerBlue;
            this.lab4.Location = new
System.Drawing.Point(0, 0);
            this.lab4.Name = "lab4";
            this.lab4.RightToLeft =
System.Windows.Forms.RightToLeft.No;
            this.lab4.Size = new
System.Drawing.Size(120, 21);
            this.lab4.TabIndex = 1;
            this.lab4.Text = "Задняястена";
```

```
            this.lab4.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // pib4
            //
            this.pib4.Controls.Add(this.pan4);
            this.pib4.Dock =
System.Windows.Forms.DockStyle.Bottom;
            this.pib4.Image =
((System.Drawing.Image)(resources.GetObject("pib4.Image"
)));
            this.pib4.Location = new
System.Drawing.Point(0, 24);
            this.pib4.Name = "pib4";
            this.pib4.Size = new
System.Drawing.Size(120, 130);
            this.pib4.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pib4.TabIndex = 0;
            this.pib4.TabStop = false;
            this.pib4.WaitOnLoad = true;
            this.pib4.Click += new
System.EventHandler(this.slider4Click);
            this.pib4.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider4Pan_M
ouseDown);
            this.pib4.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider4Pan_M
ouseMove);
            this.pib4.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider4Pan_M
ouseUp);
            //
            // pan4
            //
            this.pan4.BackColor =
System.Drawing.Color.Transparent;
            this.pan4.Enabled = false;
```

```
            this.pan4.Location = new
System.Drawing.Point(15, 15);
            this.pan4.Name = "pan4";
            this.pan4.Size = new System.Drawing.Size(75,
103);
            this.pan4.TabIndex = 2;
            this.pan4.Visible = false;
            this.pan4.Click += new
System.EventHandler(this.slider4Click);
            this.pan4.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider4Pan_M
ouseDown);
            this.pan4.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider4Pan_M
ouseMove);
            this.pan4.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider4Pan_M
ouseUp);
            //
            // pan3D
            //
            this.pan3D.BackColor =
System.Drawing.Color.Transparent;
            this.pan3D.Controls.Add(this.lab3);
            this.pan3D.Controls.Add(this.pib3);
            this.pan3D.Location = new
System.Drawing.Point(427, 28);
            this.pan3D.Name = "pan3D";
            this.pan3D.Size = new
System.Drawing.Size(120, 154);
            this.pan3D.TabIndex = 2;
            //
            // lab3
            //
            this.lab3.Dock =
System.Windows.Forms.DockStyle.Top;
            this.lab3.Font = new
System.Drawing.Font("Microsoft Sans Serif", 14.25F,
```

```
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.lab3.ForeColor =
System.Drawing.Color.DodgerBlue;
            this.lab3.Location = new
System.Drawing.Point(0, 0);
            this.lab3.Name = "lab3";
            this.lab3.RightToLeft =
System.Windows.Forms.RightToLeft.No;
            this.lab3.Size = new
System.Drawing.Size(120, 21);
            this.lab3.TabIndex = 1;
            this.lab3.Text = "Наэкраном";
            this.lab3.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // pib3
            //
            this.pib3.Controls.Add(this.pan3);
            this.pib3.Dock =
System.Windows.Forms.DockStyle.Bottom;
            this.pib3.Image =
((System.Drawing.Image)(resources.GetObject("pib3.Image"
)));
            this.pib3.Location = new
System.Drawing.Point(0, 24);
            this.pib3.Name = "pib3";
            this.pib3.Size = new
System.Drawing.Size(120, 130);
            this.pib3.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pib3.TabIndex = 0;
            this.pib3.TabStop = false;
            this.pib3.WaitOnLoad = true;
            this.pib3.Click += new
System.EventHandler(this.slider3Click);
```

```
            this.pib3.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider3Pan_M
ouseDown);
            this.pib3.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider3Pan_M
ouseMove);
            this.pib3.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider3Pan_M
ouseUp);
            //
            // pan3
            //
            this.pan3.BackColor =
System.Drawing.Color.Transparent;
            this.pan3.Enabled = false;
            this.pan3.Location = new
System.Drawing.Point(15, 15);
            this.pan3.Name = "pan3";
            this.pan3.Size = new System.Drawing.Size(75,
103);
            this.pan3.TabIndex = 2;
            this.pan3.Visible = false;
            this.pan3.Click += new
System.EventHandler(this.slider3Click);
            this.pan3.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider3Pan_M
ouseDown);
            this.pan3.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider3Pan_M
ouseMove);
            this.pan3.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider3Pan_M
ouseUp);
            //
            // pan2D
            //
            this.pan2D.BackColor =
System.Drawing.Color.Transparent;
```

```
            this.pan2D.Controls.Add(this.lab2);
            this.pan2D.Controls.Add(this.pib2);
            this.pan2D.Location = new
System.Drawing.Point(255, 28);
            this.pan2D.Name = "pan2D";
            this.pan2D.Size = new
System.Drawing.Size(120, 154);
            this.pan2D.TabIndex = 2;
            //
            // lab2
            //
            this.lab2.Dock =
System.Windows.Forms.DockStyle.Top;
            this.lab2.Font = new
System.Drawing.Font("Microsoft Sans Serif", 14.25F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.lab2.ForeColor =
System.Drawing.Color.DodgerBlue;
            this.lab2.Location = new
System.Drawing.Point(0, 0);
            this.lab2.Name = "lab2";
            this.lab2.RightToLeft =
System.Windows.Forms.RightToLeft.No;
            this.lab2.Size = new
System.Drawing.Size(120, 21);
            this.lab2.TabIndex = 1;
            this.lab2.Text = "НадТВ";
            this.lab2.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // pib2
            //
            this.pib2.Controls.Add(this.pan2);
            this.pib2.Dock =
System.Windows.Forms.DockStyle.Bottom;
```

```
            this.pib2.Image =
((System.Drawing.Image)(resources.GetObject("pib2.Image"
)));
            this.pib2.Location = new
System.Drawing.Point(0, 24);
            this.pib2.Name = "pib2";
            this.pib2.Size = new
System.Drawing.Size(120, 130);
            this.pib2.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pib2.TabIndex = 0;
            this.pib2.TabStop = false;
            this.pib2.WaitOnLoad = true;
            this.pib2.Click += new
System.EventHandler(this.slider2Click);
            this.pib2.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider2Pan_M
ouseDown);
            this.pib2.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider2Pan_M
ouseMove);
            this.pib2.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider2Pan_M
ouseUp);
            //
            // pan2
            //
            this.pan2.BackColor =
System.Drawing.Color.Transparent;
            this.pan2.Enabled = false;
            this.pan2.Location = new
System.Drawing.Point(15, 15);
            this.pan2.Name = "pan2";
            this.pan2.Size = new System.Drawing.Size(75,
103);
            this.pan2.TabIndex = 2;
            this.pan2.Visible = false;
```

```
            this.pan2.Click += new
System.EventHandler(this.slider2Click);
            this.pan2.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider2Pan_M
ouseDown);
            this.pan2.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider2Pan_M
ouseMove);
            this.pan2.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider2Pan_M
ouseUp);
            //
            // pan1D
            //
            this.pan1D.BackColor =
System.Drawing.Color.Transparent;
            this.pan1D.Controls.Add(this.lab1);
            this.pan1D.Controls.Add(this.pib1);
            this.pan1D.Location = new
System.Drawing.Point(91, 28);
            this.pan1D.Name = "pan1D";
            this.pan1D.Size = new
System.Drawing.Size(120, 154);
            this.pan1D.TabIndex = 1;
            //
            // lab1
            //
            this.lab1.Dock =
System.Windows.Forms.DockStyle.Top;
            this.lab1.Font = new
System.Drawing.Font("Microsoft Sans Serif", 11.25F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.lab1.ForeColor =
System.Drawing.Color.DodgerBlue;
            this.lab1.Location = new
System.Drawing.Point(0, 0);
            this.lab1.Name = "lab1";
```

```
            this.lab1.RightToLeft =
System.Windows.Forms.RightToLeft.No;
            this.lab1.Size = new
System.Drawing.Size(120, 21);
            this.lab1.TabIndex = 1;
            this.lab1.Text = "Надвитриной";
            this.lab1.TextAlign =
System.Drawing.ContentAlignment.MiddleCenter;
            //
            // pib1
            //
            this.pib1.Controls.Add(this.pan1);
            this.pib1.Dock =
System.Windows.Forms.DockStyle.Bottom;
            this.pib1.Image =
((System.Drawing.Image)(resources.GetObject("pib1.Image"
)));
            this.pib1.Location = new
System.Drawing.Point(0, 24);
            this.pib1.Name = "pib1";
            this.pib1.Size = new
System.Drawing.Size(120, 130);
            this.pib1.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pib1.TabIndex = 0;
            this.pib1.TabStop = false;
            this.pib1.WaitOnLoad = true;
            this.pib1.Click += new
System.EventHandler(this.slider1Click);
            this.pib1.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider1Pan_M
ouseDown);
            this.pib1.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider1Pan_M
ouseMove);
            this.pib1.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider1Pan_M
ouseUp);
```

```
            //
            // pan1
            //
            this.pan1.BackColor =
System.Drawing.Color.Transparent;
            this.pan1.Enabled = false;
            this.pan1.Location = new
System.Drawing.Point(15, 15);
            this.pan1.Name = "pan1";
            this.pan1.Size = new System.Drawing.Size(75,
103);
            this.pan1.TabIndex = 2;
            this.pan1.Visible = false;
            this.pan1.Click += new
System.EventHandler(this.slider1Click);
            this.pan1.MouseDown += new
System.Windows.Forms.MouseEventHandler(this.slider1Pan_M
ouseDown);
            this.pan1.MouseMove += new
System.Windows.Forms.MouseEventHandler(this.slider1Pan_M
ouseMove);
            this.pan1.MouseUp += new
System.Windows.Forms.MouseEventHandler(this.slider1Pan_M
ouseUp);
            //
            // water
            //
            this.water.Controls.Add(this.label5);
            this.water.Controls.Add(this.label6);
            this.water.Controls.Add(this.label7);
            this.water.Controls.Add(this.label4);
            this.water.Controls.Add(this.label3);
            this.water.Controls.Add(this.label2);
            this.water.Controls.Add(this.label1);
            this.water.Controls.Add(this.water1);
            this.water.Location = new
System.Drawing.Point(209, 6);
            this.water.Name = "water";
```

```
            this.water.Size = new
System.Drawing.Size(638, 0);
            this.water.TabIndex = 3;
            //
            // label5
            //
            this.label5.AutoSize = true;
            this.label5.Font = new
System.Drawing.Font("Microsoft Sans Serif", 15.75F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.label5.Location = new
System.Drawing.Point(218, 182);
            this.label5.Name = "label5";
            this.label5.Size = new
System.Drawing.Size(61, 25);
            this.label5.TabIndex = 9;
            this.label5.Text = "Cyxo";
            //
            // label6
            //
            this.label6.AutoSize = true;
            this.label6.Font = new
System.Drawing.Font("Microsoft Sans Serif", 15.75F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.label6.Location = new
System.Drawing.Point(218, 355);
            this.label6.Name = "label6";
            this.label6.Size = new
System.Drawing.Size(61, 25);
            this.label6.TabIndex = 8;
            this.label6.Text = "Cyxo";
            //
            // label7
            //
            this.label7.AutoSize = true;
```

```
            this.label7.Font = new
System.Drawing.Font("Microsoft Sans Serif", 15.75F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.label7.Location = new
System.Drawing.Point(218, 277);
            this.label7.Name = "label7";
            this.label7.Size = new
System.Drawing.Size(61, 25);
            this.label7.TabIndex = 7;
            this.label7.Text = "Сухо";
            //
            // label4
            //
            this.label4.AutoSize = true;
            this.label4.Font = new
System.Drawing.Font("Microsoft Sans Serif", 15.75F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.label4.Location = new
System.Drawing.Point(398, 36);
            this.label4.Name = "label4";
            this.label4.Size = new
System.Drawing.Size(142, 25);
            this.label4.TabIndex = 6;
            this.label4.Text = "Подачаводы";
            this.label4.Click += new
System.EventHandler(this.label4_Click);
            //
            // label3
            //
            this.label3.AutoSize = true;
            this.label3.Font = new
System.Drawing.Font("Microsoft Sans Serif", 15.75F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.label3.Location = new
System.Drawing.Point(62, 182);
```

```
            this.label3.Name = "label3";
            this.label3.Size = new
System.Drawing.Size(82, 25);
            this.label3.TabIndex = 5;
            this.label3.Text = "Туалет";
            //
            // label2
            //
            this.label2.AutoSize = true;
            this.label2.Font = new
System.Drawing.Font("Microsoft Sans Serif", 15.75F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.label2.Location = new
System.Drawing.Point(62, 355);
            this.label2.Name = "label2";
            this.label2.Size = new
System.Drawing.Size(108, 25);
            this.label2.TabIndex = 3;
            this.label2.Text = "Потолок2";
            //
            // label1
            //
            this.label1.AutoSize = true;
            this.label1.Font = new
System.Drawing.Font("Microsoft Sans Serif", 15.75F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.label1.Location = new
System.Drawing.Point(62, 277);
            this.label1.Name = "label1";
            this.label1.Size = new
System.Drawing.Size(108, 25);
            this.label1.TabIndex = 2;
            this.label1.Text = "Потолок1";
            //
            // water1
            //
```

```
            this.water1.Image =
global::MagControl.Properties.Resources.sw1;
            this.water1.Location = new
System.Drawing.Point(421, 79);
            this.water1.Name = "water1";
            this.water1.Size = new
System.Drawing.Size(100, 50);
            this.water1.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.water1.TabIndex = 1;
            this.water1.TabStop = false;
            this.water1.Click += new
System.EventHandler(this.pictureBox2_Click);
            //
            // cams
            //

this.cams.Controls.Add(this.axWindowsMediaPlayer1);
            this.cams.Location = new
System.Drawing.Point(209, 3);
            this.cams.Name = "cams";
            this.cams.Size = new
System.Drawing.Size(638, 0);
            this.cams.TabIndex = 10;
            //
            // axWindowsMediaPlayer1
            //
            this.axWindowsMediaPlayer1.Enabled = true;
            this.axWindowsMediaPlayer1.Location = new
System.Drawing.Point(33, 28);
            this.axWindowsMediaPlayer1.Name =
"axWindowsMediaPlayer1";
            this.axWindowsMediaPlayer1.OcxState =
((System.Windows.Forms.AxHost.State)(resources.GetObject
("axWindowsMediaPlayer1.OcxState")));
            this.axWindowsMediaPlayer1.Size = new
System.Drawing.Size(596, 409);
            this.axWindowsMediaPlayer1.TabIndex = 0;
```

```
            //
            // ir
            //
            this.ir.Controls.Add(this.pictureBox13);
            this.ir.Controls.Add(this.pictureBox12);
            this.ir.Controls.Add(this.pictureBox11);
            this.ir.Controls.Add(this.pictureBox10);
            this.ir.Controls.Add(this.pictureBox9);
            this.ir.Controls.Add(this.pictureBox8);
            this.ir.Controls.Add(this.pictureBox7);
            this.ir.Controls.Add(this.pictureBox6);
            this.ir.Controls.Add(this.pictureBox5);
            this.ir.Controls.Add(this.pictureBox4);
            this.ir.Controls.Add(this.pictureBox3);
            this.ir.Controls.Add(this.pictureBox2);
            this.ir.Controls.Add(this.pictureBox1);
            this.ir.Controls.Add(this.button7);
            this.ir.Controls.Add(this.button6);
            this.ir.Controls.Add(this.button5);
            this.ir.Location = new
System.Drawing.Point(209, 3);
            this.ir.Name = "ir";
            this.ir.Size = new System.Drawing.Size(638,
0);
            this.ir.TabIndex = 4;
            //
            // pictureBox13
            //
            this.pictureBox13.Image =
global::MagControl.Properties.Resources._5;
            this.pictureBox13.Location = new
System.Drawing.Point(384, 356);
            this.pictureBox13.Name = "pictureBox13";
            this.pictureBox13.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox13.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox13.TabIndex = 15;
```

```
            this.pictureBox13.TabStop = false;
            //
            // pictureBox12
            //
            this.pictureBox12.Image =
global::MagControl.Properties.Resources._2;
            this.pictureBox12.Location = new
System.Drawing.Point(304, 356);
            this.pictureBox12.Name = "pictureBox12";
            this.pictureBox12.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox12.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox12.TabIndex = 14;
            this.pictureBox12.TabStop = false;
            //
            // pictureBox11
            //
            this.pictureBox11.Image =
global::MagControl.Properties.Resources._1;
            this.pictureBox11.Location = new
System.Drawing.Point(232, 356);
            this.pictureBox11.Name = "pictureBox11";
            this.pictureBox11.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox11.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox11.TabIndex = 13;
            this.pictureBox11.TabStop = false;
            //
            // pictureBox10
            //
            this.pictureBox10.Image =
global::MagControl.Properties.Resources._4;
            this.pictureBox10.Location = new
System.Drawing.Point(156, 356);
            this.pictureBox10.Name = "pictureBox10";
```

```
            this.pictureBox10.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox10.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox10.TabIndex = 12;
            this.pictureBox10.TabStop = false;
            //
            // pictureBox9
            //
            this.pictureBox9.Image =
global::MagControl.Properties.Resources._11;
            this.pictureBox9.Location = new
System.Drawing.Point(59, 244);
            this.pictureBox9.Name = "pictureBox9";
            this.pictureBox9.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox9.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox9.TabIndex = 11;
            this.pictureBox9.TabStop = false;
            //
            // pictureBox8
            //
            this.pictureBox8.Image =
global::MagControl.Properties.Resources._9;
            this.pictureBox8.Location = new
System.Drawing.Point(59, 178);
            this.pictureBox8.Name = "pictureBox8";
            this.pictureBox8.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox8.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox8.TabIndex = 10;
            this.pictureBox8.TabStop = false;
            //
            // pictureBox7
            //
```

```
            this.pictureBox7.Image =
global::MagControl.Properties.Resources._10;
            this.pictureBox7.Location = new
System.Drawing.Point(59, 104);
            this.pictureBox7.Name = "pictureBox7";
            this.pictureBox7.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox7.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox7.TabIndex = 9;
            this.pictureBox7.TabStop = false;
            //
            // pictureBox6
            //
            this.pictureBox6.Image =
global::MagControl.Properties.Resources._6;
            this.pictureBox6.Location = new
System.Drawing.Point(506, 104);
            this.pictureBox6.Name = "pictureBox6";
            this.pictureBox6.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox6.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox6.TabIndex = 8;
            this.pictureBox6.TabStop = false;
            //
            // pictureBox5
            //
            this.pictureBox5.Image =
global::MagControl.Properties.Resources._7;
            this.pictureBox5.Location = new
System.Drawing.Point(270, 242);
            this.pictureBox5.Name = "pictureBox5";
            this.pictureBox5.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox5.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox5.TabIndex = 7;
```

```
            this.pictureBox5.TabStop = false;
            //
            // pictureBox4
            //
            this.pictureBox4.Image =
global::MagControl.Properties.Resources._12;
            this.pictureBox4.Location = new
System.Drawing.Point(270, 176);
            this.pictureBox4.Name = "pictureBox4";
            this.pictureBox4.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox4.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox4.TabIndex = 6;
            this.pictureBox4.TabStop = false;
            //
            // pictureBox3
            //
            this.pictureBox3.Image =
global::MagControl.Properties.Resources._8;
            this.pictureBox3.Location = new
System.Drawing.Point(198, 176);
            this.pictureBox3.Name = "pictureBox3";
            this.pictureBox3.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox3.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox3.TabIndex = 5;
            this.pictureBox3.TabStop = false;
            //
            // pictureBox2
            //
            this.pictureBox2.Image =
global::MagControl.Properties.Resources._14;
            this.pictureBox2.Location = new
System.Drawing.Point(344, 176);
            this.pictureBox2.Name = "pictureBox2";
```

```
            this.pictureBox2.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox2.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox2.TabIndex = 4;
            this.pictureBox2.TabStop = false;
            //
            // pictureBox1
            //
            this.pictureBox1.Image =
global::MagControl.Properties.Resources._13;
            this.pictureBox1.Location = new
System.Drawing.Point(270, 113);
            this.pictureBox1.Name = "pictureBox1";
            this.pictureBox1.Size = new
System.Drawing.Size(55, 50);
            this.pictureBox1.SizeMode =
System.Windows.Forms.PictureBoxSizeMode.Zoom;
            this.pictureBox1.TabIndex = 3;
            this.pictureBox1.TabStop = false;
            //
            // button7
            //
            this.button7.Font = new
System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button7.Location = new
System.Drawing.Point(477, 23);
            this.button7.Name = "button7";
            this.button7.Size = new
System.Drawing.Size(123, 38);
            this.button7.TabIndex = 2;
            this.button7.Text = "Runco 760w";
            this.button7.UseVisualStyleBackColor = true;
            //
            // button6
            //
```

```
            this.button6.Font = new
System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button6.Location = new
System.Drawing.Point(206, 23);
            this.button6.Name = "button6";
            this.button6.Size = new
System.Drawing.Size(193, 38);
            this.button6.TabIndex = 1;
            this.button6.Text = "Yamaha-RSTX1000-1";
            this.button6.UseVisualStyleBackColor = true;
            //
            // button5
            //
            this.button5.Font = new
System.Drawing.Font("Microsoft Sans Serif", 12F,
System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)(204)));
            this.button5.Location = new
System.Drawing.Point(27, 23);
            this.button5.Name = "button5";
            this.button5.Size = new
System.Drawing.Size(117, 38);
            this.button5.TabIndex = 0;
            this.button5.Text = "SamsungTV";
            this.button5.UseVisualStyleBackColor = true;
            //
            // Form1
            //
            this.AutoScaleDimensions = new
System.Drawing.SizeF(6F, 13F);
            this.AutoScaleMode =
System.Windows.Forms.AutoScaleMode.Font;
            this.ClientSize = new
System.Drawing.Size(850, 468);
            this.Controls.Add(this.ir);
            this.Controls.Add(this.cams);
```

```
            this.Controls.Add(this.water);
            this.Controls.Add(this.light);
            this.Controls.Add(this.panel1);
            this.Name = "Form1";
            this.Text = "Form1";
            this.panel1.ResumeLayout(false);
            this.light.ResumeLayout(false);
            this.Refectory3.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.Refecto
ry3Pib)).EndInit();
            this.pan4D.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.pib4)).
EndInit();
            this.pib4.ResumeLayout(false);
            this.pan3D.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.pib3)).
EndInit();
            this.pib3.ResumeLayout(false);
            this.pan2D.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.pib2)).
EndInit();
            this.pib2.ResumeLayout(false);
            this.pan1D.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.pib1)).
EndInit();
            this.pib1.ResumeLayout(false);
            this.water.ResumeLayout(false);
            this.water.PerformLayout();

((System.ComponentModel.ISupportInitialize)(this.water1)
).EndInit();
            this.cams.ResumeLayout(false);
```

```
((System.ComponentModel.ISupportInitialize)(this.axWindo
wsMediaPlayer1)).EndInit();
            this.ir.ResumeLayout(false);

((System.ComponentModel.ISupportInitialize)(this.picture
Box13)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box12)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box11)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box10)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box9)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box8)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box7)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box6)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box5)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box4)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box3)).EndInit();
```

```
((System.ComponentModel.ISupportInitialize)(this.picture
Box2)).EndInit();

((System.ComponentModel.ISupportInitialize)(this.picture
Box1)).EndInit();
            this.ResumeLayout(false);

        }

        #endregion

        private System.Windows.Forms.Panel panel1;
        private System.Windows.Forms.Button button4;
        private System.Windows.Forms.Button button3;
        private System.Windows.Forms.Button button2;
        private System.Windows.Forms.Button button1;
        private System.Windows.Forms.Panel light;
        private System.Windows.Forms.Panel pan1D;
        private System.Windows.Forms.Label lab1;
        private System.Windows.Forms.PictureBox pib1;
        private System.Windows.Forms.Panel pan1;
        private System.Windows.Forms.Panel pan4D;
        private System.Windows.Forms.Label lab4;
        private System.Windows.Forms.PictureBox pib4;
        private System.Windows.Forms.Panel pan4;
        private System.Windows.Forms.Panel pan3D;
        private System.Windows.Forms.Label lab3;
        private System.Windows.Forms.PictureBox pib3;
        private System.Windows.Forms.Panel pan3;
        private System.Windows.Forms.Panel pan2D;
        private System.Windows.Forms.Label lab2;
        private System.Windows.Forms.PictureBox pib2;
        private System.Windows.Forms.Panel pan2;
        private System.Windows.Forms.Panel water;
        private System.Windows.Forms.Label label4;
        private System.Windows.Forms.Label label3;
        private System.Windows.Forms.Label label2;
```

```
        private System.Windows.Forms.Label label1;
        private System.Windows.Forms.PictureBox water1;
        private System.Windows.Forms.Label label5;
        private System.Windows.Forms.Label label6;
        private System.Windows.Forms.Label label7;
        private System.Windows.Forms.Panel cams;
        private AxWMPLib.AxWindowsMediaPlayer
axWindowsMediaPlayer1;
        private System.Windows.Forms.Panel ir;
        private System.Windows.Forms.Button button7;
        private System.Windows.Forms.Button button6;
        private System.Windows.Forms.Button button5;
        private System.Windows.Forms.PictureBox
pictureBox1;
        private System.Windows.Forms.PictureBox
pictureBox13;
        private System.Windows.Forms.PictureBox
pictureBox12;
        private System.Windows.Forms.PictureBox
pictureBox11;
        private System.Windows.Forms.PictureBox
pictureBox10;
        private System.Windows.Forms.PictureBox
pictureBox9;
        private System.Windows.Forms.PictureBox
pictureBox8;
        private System.Windows.Forms.PictureBox
pictureBox7;
        private System.Windows.Forms.PictureBox
pictureBox6;
        private System.Windows.Forms.PictureBox
pictureBox5;
        private System.Windows.Forms.PictureBox
pictureBox4;
        private System.Windows.Forms.PictureBox
pictureBox3;
        private System.Windows.Forms.PictureBox
pictureBox2;
```

```
        private System.Windows.Forms.Panel Refectory3;
        private System.Windows.Forms.PictureBox
Refectory3Pib;
        private System.Windows.Forms.Label
Refectory3Lab;
    }
}

using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace MagControl
{
    static class Program
    {
        /// <summary>
        /// Главная точка входа для приложения.
/// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);
Application.Run(new Form1());
        }
    }
}
```