Центральноукраїнський національний технічний університет
Центр заочної та дистанційної освіти
Кафедра кібербезпеки та програмного забезпечення

”Допущено до захисту”
Завідувач кафедри кібербезпеки
та програмного забезпечення
д.т.н., професор
_____ Олексій СМІРНОВ
“____” _____ 2021 р.

ВИПУСКНА КВАЛІФІКАЦІЙНА РОБОТА
за другим (магістерським) рівнем вищої освіти
на тему
“Дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S”

Виконав здобувач вищої освіти
ІІ курсу, групи КІ-20МЗ
ОПП «Комп’ютерна інженерія»
спеціальності 123 «Комп’ютерна інженерія»
_____ Сільченко С.М.
«____» _____ 2021 р.

Керівник проекту
доктор технічних наук, доцент
_____ Олександр КОВАЛЕНКО
«____» _____ 2021 р.
Рецензент _____
_____

м. Кропивницький

Центральноукраїнський національний технічний університет

Центр *Заочної та дистанційної освіти*

Кафедра *Кібербезпеки та програмного забезпечення*

Рівень вищої освіти *магістр*

Галузь знань ._12 _____*"*Інформаційні технології*"*_____

Спеціальність *123 "*Комп'ютерна інженерія*"*

Освітньо-професійна (освітньо-наукова) програма _*"*Комп'ютерна інженерія*"*_

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____ Олексій СМІРНОВ
« 6 » вересня 2021 року

ЗАВДАННЯ НА ВИПУСКНУ КВАЛІФІКАЦІЙНУ РОБОТУ ЗА ДРУГИМ (МАГІСТЕРСЬКИМ) РІВНЕМ ВИЩОЇ ОСВІТИ ЗДОБУВАЧА ВИЩОЇ ОСВІТИ

Сільченку Сергію Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S*

2. Керівник роботи *Коваленко Олександр Володимирович, докт. техн. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 41-13 від 02.08.2021 року

3. Строк подання студентом роботи до захисту *10.12.2021 р.*

4. Мета та завдання випускної кваліфікаційної роботи*: Метою розробки є дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S*

5. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

1. Призначення та область використання.	*7. Економічна ефективність розробленої*
2.Перегляд аналогічних існуючих систем.	*програми.*
3. Опис і обґрунтування проектних рішень.	*8. Заходи з охорони праці та техніки безпеки*
4. Етапи програмування системи.	*9. Висновки.*
5. Впровадження системи в промислову експлуатацію	
6. Наукова новизна	

6. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)

Наукова новизна	*1 аркуш*
Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*
Показники економічної ефективності	*1 аркуш*

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Економічний	Савеленко Г.В.	05.10.2021	14.11.2021
Охорона праці	Оришака О.В.	06.10.2021	16.11.2021

7. Дата видачі завдання « 6 » вересня 2021 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Строк виконання етапів випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти	Примітка
1.	Аналіз існуючих систем	10.10.2021 р.	
2.	Постановка задачі, оформлення ТЗ	15.10.2021 р.	
3.	Розробка моделі компонента	20.10.2021 р.	
4.	Розробка структур даних	25.10.2021 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.10.2021 р.	
6.	Програмування алгоритмів	10.11.2021 р.	
7.	Розрахунок економічної ефективності	13.11.2021 р.	
8.	Розрахунки з охорони праці та техніки безпеки	15.11.2021 р.	
9.	Оформлення ПЗ	17.11.2021 р.	
10.	Попередній захист роботи	10.12.2021 р.	

Дата видачі завдання
« 6 » вересня 2021 р.

Підпис керівника

_____ _____
(прізвище та ініціали)

Завдання прийнято до виконання
« 6 » вересня 2021 р.

Підпис здобувача

_____ _____
(прізвище та ініціали)

АНОТАЦІЯ

Сільченко С.М. Дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2021.

В даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи керування супутниковим HD ресивером на базі процесору GX6605S.

Метою розробки є дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S.

Об'єктом дослідження є процес керування супутниковим HD ресивером на базі процесору GX6605S.

Предметом дослідження є методи керування супутниковим HD ресивером на базі процесору GX6605S.

Методи дослідження базуються на методах теорії інформації та кодування, методах математичної статистики, методах розробки програмного забезпечення.

Результат роботи – програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S.

В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Розроблено зручний інтерфейс користувача. Наведені інструкції по роботі з програмними засобами.

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows XP/Vista/7/8/10.

Програму розроблено в середовищі Delphi 10.4 Sydney.

Ключові слова: комп'ютерна інженерія, HD ресивер, GX6605S

ABSTRACT

Silchenko S.M. Research and software implementation of the GX6605S processor-based satellite HD receiver control system. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi. 2021

In this final qualifying work for the second (master's) level of higher education, software has been developed that is designed for the control system of the satellite HD receiver based on the GX6605S processor.

The purpose of development is research and software implementation of the control system of the satellite HD receiver based on the GX6605S processor.

The object of research is the process of controlling a satellite HD receiver based on the GX6605S processor.

The subject of research is the methods of controlling a satellite HD receiver based on the GX6605S processor.

Research methods are based on methods of information theory and coding, methods of mathematical statistics, methods of software development.

The result is a software implementation of the control system of the satellite HD receiver based on the GX6605S processor.

In the process of working on the software model, an analysis of existing hardware and software was performed. All components of the developed software are fully described.

Developed user-friendly interface. Instructions for working with software are given.

The program can be used on an IBM PC with Windows XP / Vista / 7/8/10.

The program is developed in the environment of Delphi 10.4 Sydney.

Keywords: computer engineering, HD receiver, GX6605S

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

АЛП – арифметично-логічний перетворювач

АЦП – аналогово-цифровий перетворювач

БІ – блок індикації

В/В – ввод/вивід

ВІС – велика інтегральна схема

ДК – дистанційне керування

ЕОМ – електронно–обчислювальна машина

ІТ – інформаційні технології

ІЧ – інфрачервоне

МП – мікропроцесор

ОЗП – оперативний запам'ятовувальний пристрій

ПЗП – постійний запам'ятовувальний пристрій

СТБ – супутникове телебачення

ТБ – телебачення

ЦАП – цифрово-аналоговий перетворювач

ЦП – центральний процесор

ША – шина адреси

ШД – шина даних

ШК – шина керування

ШСЗ – штучний супутник землі

USB – universal serial bus

ВСТУП

Актуальність теми. Передача даних через супутники – область техніки зв'язку, що займається питаннями передачі Інтернету, телефонного зв’язку та телевізійних програм від передавальних земних станцій до приймачів із використанням штучних супутників землі (ШСЗ) як активних ретрансляторів [1-5]. Супутникове віщання є сьогодні самим економічним, швидким і надійним способом передачі ТБ сигналу високої якості в будь-яку точку великої території. До переваг СТБ відносяться також можливість використання сигналу необмеженим числом прийомних установок, висока надійність ШСЗ, невеликі витрати і їхня незалежність від відстані між джерелом і споживачем [4].

Важливою проблемою в прийомних установках СТБ є можливість автоматичного керування ними. Вирішити цю проблему можна за допомогою мікропроцесорних пристроїв.

Використання мікроелектронних засобів у виробах виробничого й культурно-побутового призначення не тільки приводить до підвищення техніко-економічних показників виробів (вартості, надійності, споживаній потужності, габаритних розмірів) і дозволяє багаторазово скоротити строки розробки, відсунути строки “морального старіння” виробів, але й надає їм принципово нові споживчі якості (розширені функціональні можливості) [3-8].

Використання мікропроцесорів у системах керування забезпечує досягнення високих показників ефективності при низькій вартості, таким чином, немає розумної альтернативної елементарної бази для побудови керуючих і/або регулюючих систем, як використання мікропроцесорів.

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем керування супутниковим HD ресивером на базі процесору GX6605S.

– Дослідження системи керування супутниковим HD ресивером на базі процесору GX6605S.

– Програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S.

Об'єктом дослідження є процес керування супутниковим HD ресивером на базі процесору GX6605S.

Предметом дослідження є методи керування супутниковим HD ресивером на базі процесору GX6605S.

Методи дослідження базуються на методах теорії інформації та кодування, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод керування супутниковим HD ресивером на базі процесору GX6605S.

– Розроблено вітчизняний продукт керування супутниковим HD ресивером на базі процесору GX6605S, який має більш широкі можливості, на відміну від існуючих аналогів.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі керування супутниковим HD ресивером на базі процесору GX6605S.

Достовірність наукових результатів підтверджена теоретичними викладеннями, даними комп'ютерного моделювання, коректними дослідженнями параметрів на функціонуючій обчислювальній мережі, а також відповідністю отриманих результатів окремим результатам, наведеним у науковій літературі.

Робота апробована на LV Науково-технічна конференція здобувачів вищої освіти «Наука – виробництву», 2021, основні положення випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти надруковані у статті збірника праць молодих науковців ЦНТУ, випуск №12.

Таким чином, виходячи з в ищеперерахованого, дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Віщання із супутників відбувається в такий спосіб – на земній станції в напрямку супутника передається сигнал (передача нагору) на певній частоті. На супутнику цей сигнал приймається, певним чином обробляється, підсилюється й спеціальними антенами передається на землю для прийому на наші супутникові антени (передача униз). Супутники ці перебувають на геостаціонарній орбіті. Геостаціонарної вона називається тому, що супутник перебуваючи над екватором обертається навколо землі зі швидкістю 1 оберт у добу в тому же напрямку що й Земля. І тому для нас він перебуває нерухомо в одній точці. Тому виходить що всі геостаціонарні супутники перебувають для нашої півкулі в південному напрямку. Кожний супутник перебуває на своєму, строго певному місці орбіти й залежно від положення відзначається як Х градусів східної або західної довготи. Якщо з'єднати однією лінією місце розташування всіх супутників, то для спостерігача із землі ця лінія буде виглядати дугою. Починаючи зі східного напрямку ця дуга буде підніматися до півдня й опускатися до заходу, краями гублячись за лінією обрію. Тому чим більше величина в градусах позиції супутника, тим нижче треба опускати супутникову антену.

Супутникова антена влаштована таким чином, щоб енергія високочастотного випромінювання, що попадає на дзеркало антени (поверхня тарілки) відбивалася у фокус антени, де розташований конвертер. Антени бувають різних видів і типів залежно від форми й матеріалу з якого неї виготовляють. Два основних види антен це прямофокусні й офсетні. Прямофокусні як правило круглої форми й кріплення конвертера перебуває в центрі. Офсетні більше нагадують яйце, вони витягнуті по горизонталі й звужені по вертикалі. Конвертер кріпиться внизу, зорово набагато нижче центра. Як

правило, в основному застосовують прямофокусні антени коли потрібний великий діаметр, а офсетні при невеликому діаметрі. Хоча зустрічалися моделі офсетних антен 1.85 метра а прямофокусних 0.60 метра. Від діаметра антени напряму залежить коефіцієнт підсилення. За рівнем поля в точці прийому розраховується діаметр антени. Наприклад, при рівні 50 ДБ/ВТ діаметр антени досить 0.60 метра, а при 40 ДБ/ВТ діаметр повинен бути мінімум 1.8 метра.

1.2 Область застосування

Тепер про конвертери. Сигнал, сфальцьований антеною попадає в випромінювач конвертера виконаний у вигляді хвилеводу й попадає на прийомний штир усередині хвилеводу. У всіх сучасних конвертерах таких штирів – 2, для прийому горизонтальної й вертикальної поляризації. Потім цей сигнал у штирі перетвориться в електричний струм високої частоти, підсилюється й конвертується в супутникову проміжну частоту (ПЧ) від 950 Мгц. до 2150 Мгц. І вже ця ПЧ по кабелі передається на вхід ресивера. Конвертери для супутникового прийому діляться на два типи. Для прийому діапазону 3.7-4.2 Ггц це C-BAND, для прийому діапазону 10.7-12.75 Ггц це KU-BAND. Відрізнити один від іншого дуже легко. У конвертерів C-BAND діаметр хвилеводу набагато більший чим в KU-BAND. Причому в C -BAND конвертерів форма хвилеводу розрахована на установку в прямофокусних антенах(за рідкісними винятками), а в KU-BAND конвертерів хвилевід розрахований як на установку в прямофокусних антенах, так і в офсетні (якщо зовнішній зріз хвилеводу сходами йде усередину те це конвертер для офсетної антени). Хоча при наявності запасу по діаметрі антени конвертер для прямофокусної антени можна встановити в офсетну й навпаки. Але краще підбирати конвертер по типі антени.

На більшості європейських супутників сигнал передається в горизонтальній і вертикальній поляризації в діапазоні KU-BAND. Поділ на 2 поляризації дозволяє повніше використовувати частотний ресурс, тобто ближче зрушити по частоті передані сигнали. На супутнику з якого транслюються

програми XTRA.TV і на більшості супутників працюючих у діапазоні C-BAND використовується кругова поляризація – Ліва й Права. Для прийому кругової поляризації конвертер необхідно доустатковувати спеціальною пластиною – деполяризатором. Вона виготовлена із фторопласта або гетинаксу й має товщину 2 мм. при довжині 3 див. для KU-BAND конвертера приблизно 5-6 мм. товщину й 5-6 см. довжину для C -BAND конвертера. Розташовується пластина усередині хвилеводу приблизно під однаковим кутом між штирями (45 градусів). Причому при налаштуванні ресивера необхідно встановити поляризацію H для прийому лівої й V для прийому правої кругової поляризації. У деяких моделях конвертерів C-BAND усередині є напрямні пази для правильної установки деполяризатора. Хоча для точної юстировки це незручно.

Налаштування антени й ресивера

Отже в нас є антена, конвертер і ресивер (цифровий) підходящі для прийому потрібних нам програм з певного супутника. Тепер треба все це настроїти. Розглянемо найпростіший випадок з нерухливою антеною на один супутник. У першу чергу треба настроїти ресивер. У більшості ресиверів у меню налаштування є індикатор рівня і якості сигналу. Рівень сигналу другорядний показник, тому що він вимірює просто всі що йде з конвертера незалежно від природи – шуми конвертера, теплові шуми атмосфери й тільки при влученні на супутник може збільшиться на деяку величину. Хоча за допомогою його можна переконатися в тому що конвертер справний і кабель до нього приєднаний. Самий головний показник це індикатор якості сигналу. Поки він на нулі, ніякого прийому не буде. Уводите в меню налаштування в потрібних місцях частоту конвертера (вказується на конвертері, якщо універсальний те це 9750/10600. Якщо для XTRA.TV (штатний конвертер, хоча може попастися перероблений для прийому кругової поляризації універсальний) то це 10750. Якщо це для C-BAND то це 5150), частоту сигналу в GHz, швидкість потоку – SR, поляризацію H, V або R, L і якщо треба FEC – від 1/2 до 7/8 (на багатьох сучасних ресиверах ці дані визначаються автоматично й в установках не запитуються). Дані по всіх цих пунктах можна знайти в посиланнях на цьому сайті. Перевірте як установлений конвертер – якщо прийом буде в H, V те перевірте щоб стрілка на корпусі

конвертера визначаюча положення нагору дивилася саме туди (нагору). Якщо стрілки на корпусі немає, то вихід конвертера повинен приблизно дивитися униз. Для прийому R, L це байдуже. До речі стійка на якій установлюється антена повинна бути встановлена дуже жорстко без люфту й строго виставлена по горизонталі. Якщо приблизно обчислили напрямок на супутник то направте антену в цьому напрямку. Якщо не маєте подання то знайдіть напрямок на ПІВДЕНЬ. Саме собою в напрямку на супутник не повинне бути висотних будинків, дерев і інших перешкод які зроблять прийом неможливим. Плавно рухаючи антену вліво-вправо й після кожного повороту потроху нахиляючи нижче по 2,3 градуси (заздалегідь взяти найвище положення нахилу антени) стежите за індикатором якості. Якщо все робите акуратно й ніяких перешкод на шляху проходження сигналу немає, то рано або пізно сигнал з'явиться. Потім треба точніше від'юстувати положення антени, небагато лівіше-правіше й нижче по максимуму якості сигналу. Після цього можна точніше настроїти положення конвертера, ближче-далі від антени і якщо приймаєте H, V, повертаєте по годинниковій або проти годинникової стрілки конвертер. Якщо приймаєте кругову поляризацію то спробуйте покрутити деполяризатор і поміняти його положення, глибше усередину або навпаки назовні. Не забудьте що в процесі налаштування конвертера сигнал буде зменшуватися в той момент коли ви затінюєте рукою (або іншими частинами тіла) дзеркало антени. Зручніше за все робити налаштування антени встановивши біля її телевізор з ресивером і з'єднати на час налаштування конвертер з ресивером окремим шматком кабелю. Є й спеціальні прилади для налаштування антен, але їх використовують в основному досвідчені зі стажем установники.

Таким чином, виходячи з вищеперерахованого, дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S, є актуальною задачею, яка потребує вирішення у даній випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Дамо визначення ряду термінів, що використовуються при установці керованого тюнера супутникової антени.

«ПОЯС КЛАРКА» Штучний супутник землі виведений на кругову екваторіальну орбіту на висоту 35800 км, буде обертатися навколо землі за 24 години. Для спостерігача, що перебуває на землі, цей супутник завжди буде перебувати на одному місці і його орбіті називається геостаціонарної. Якщо представити всі супутники, що перебувають на геостаціонарній орбіті, то в південній частині неба (для північної півкулі) ланцюжок супутників вишикується в дугу. Ця дуга названа поясом Кларка. Самий верхній супутник буде тим вище, ніж південніше перебуває спостерігач. На екваторі верхній супутник буде прямо над головою, пояс Кларка буде виглядати рівною лінією й ділити небо на дві рівні частини. Кожний супутник має свою орбітальну позицію яка визначається меридіаном над яким він розташовується. Наприклад відомий супутник HotBird перебуває над меридіаном 13 градусів східної довготи, тому і його позиція пишеться 13Е.

«КУТ МІСЦЯ» (кут піднесення) Кут місця, або кут піднесення – це кут між лінією обрію й напрямком на супутник у вертикальній площині. Чим ближче орбітальна позиція супутника до географічної довготи місця прийому, тим більше кут місця, тобто тим вище супутник над обрієм. У міру видалення орбітальної позиції від географічної довготи кут місця зменшується й зрештою стає негативним, тобто супутник з такою орбітальною позицією ховається за обрієм.

«АЗИМУТ» Азимут – це напрямок на супутник у горизонтальній площині. Азимут супутника, орбітальна позиція якого збігається з довготою місця прийому, буде дорівнює 180, тобто антена буде дивитися строго на південь.

«ЕЛЕВАЦІЯ» Елевація – це один з найважливіших кутів для налаштування полярної підвіски супутникової антени. Для того, щоб приймати всі доступні супутники, потрібно щоб антена поверталася навколо певної осі. Ця вісь називається полярної й вона повинна бути паралельна земної осі. На екваторі полярна вісь буде паралельна земної поверхні, чим північніше крапка прийому, тим на менший кут від вертикалі буде відрізнятися нахил полярної осі. Отож елевація і є кут, на який нахилена полярна вісь щодо вертикалі Кут елевації визначається так: 90 градусів – широта місця прийому

«ДЕКЛІНАЦІЯ» Якщо ми виставимо тільки кут елевації для осі обертання антени, то не прийняти нам жодного супутника. «Промінь» від антени буде малювати в небі пряму лінію високо над всіма супутниками. Щоб виправити цю ситуацію антену варто нахилити щодо полярної осі на певний кут. Цей кут і називається кутом деклінації.

Дуга, яка описується полярною антенною, кут деклінації =0

Дуга, по якій налаштовуємо азимутальну підвіску

Супутник на який налаштовуємося

Південний супутник

$A_2B_2 > A_1B_1$
$A_2C_2 > A_1C_1$
$B_2C_2 = B_1C_1$
звідки
$A_1 > A_2$

Рисунок 2.1 – Ілюстрація основних понять супутникового зв'язку

«КОРИГУВАЛЬНИЙ КУТ» Тепер ми знаємо, що для того, щоб наша антена поверталася направляючи свій «промінь» точно на супутники необхідно

виставити кути елевації й деклінації. Але не всі так, як хотілося б. По розрахункових кутах елевації й деклінації антена буде своїм «променем» описувати в небі дугу небагато відрізняється від дуги пояса Кларка, що звичайно ж дуже небажано. І чим більше антена, тим сильніше ця погрішність буде позначатися на прийомі. Для цього вводиться невеликий коригувальний кут, що звичайно не перевищує одного градуса. На величину цього кута зменшується кут елевації й збільшується кут деклінації

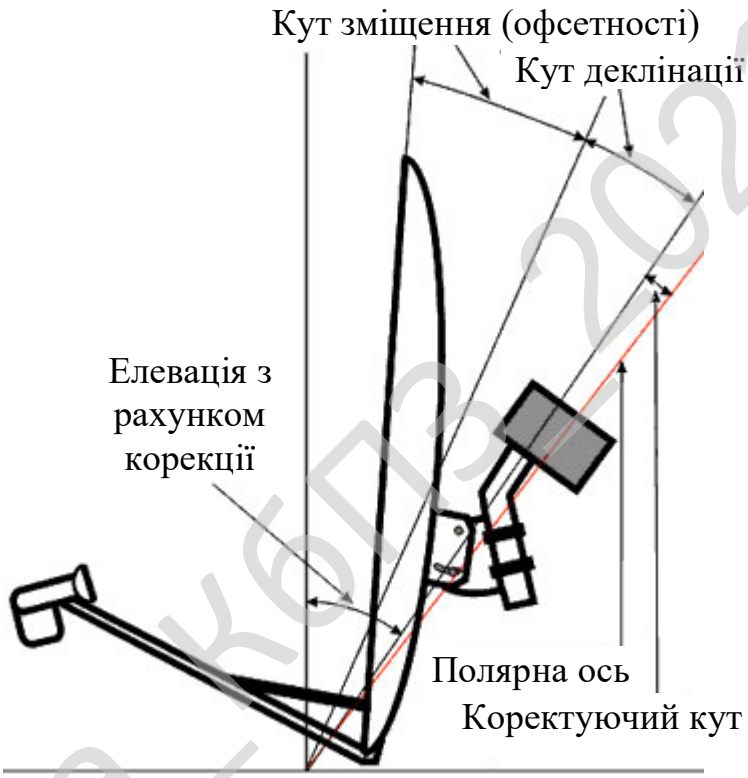

Рисунок 2.2 – Ілюстрація основних понять супутникової антени

«КУТ ПОВОРОТУ ПОЛЯРНОЇ ПІДВІСКИ» Кут повороту полярної підвіски – це кут, на який потрібно повернути антену навколо полярної осі щодо напрямку на південь для налаштування на певний супутник. На перший погляд здається, що цей кут повинен бути дорівнює різниці між азимутами супутника й азимутом напрямку на південь. Але це тільки так здається. Розрахункові кути повороту полярної підвіски наведені в існуючих таблицях.

«КУТ ОФСЕТНОСТІ АНТЕНИ» Сама назва антени говорить про те, що фокус у неї зміщений. Конвертер винесений із зони затінення дзеркала антени, саме тому «промінь» антени не перпендикулярний площини раскрива антени, а спрямований вище саме на цей кут офсетності. Завдяки цьому офсетна антена не дивиться в небо як прямофокусна, а розташовується майже вертикально.

Розглянемо основні етапи встановлення керованої супутникової антени.

1. Придбаємо офсетну антену. Є маса критеріїв вибору супутникової антени. Одним із кращих варіантів у цьому випадку може стати антена фірми «Triax» 1.1м (по технічним характеристикам перевершує своїх конкурентів, а по вазі вона легше аналогічних супутникових антен даного діаметра). Мінімальна вага обумовлена конструктивними особливостями мотоподвісу. Чим менше на нього буде навантаження, тим довше він проробить. Природно до антени буде потрібно ще й конвертер.

2. Придбаємо мотоподвіс. Є безліч різних моделей мотоподвісів, але конструктивно вони дуже мало відрізняються друг від друга. Питанню вибору мотоподвісу варто приділити увагу, пошукати інформацію в Інтернеті, порадитися у фірмах де ви купуєте встаткування й т.д. Найбільш надійний мотоподвіс Strong SRT DM-2100.

3. Запастися рівним відрізком труби діаметром 1/2 дюйма або біля того, довжиною ледве більше діаметра антени.

4. Зробіть розрахунок кутів для налаштування деклінації й елевації.

Поля, позначені сірим треба заповнити. У нього вам потрібно буде перенести необхідні дані з таблиці розрахунку елевації, деклінації, азимутів і т.д.

Таблиця 2.1 – Розрахунок кутів для налаштування деклінації й елевації

Великий діаметр антени	980 мм
Малий діаметр антени	900 мм
Кут офсетності антени	23,3º
Деклінація	7,2º
Елевація	33,2º
Кут вигину хобота мотора	30,0º
Кут між площиною антени й трубою для установки деклінації	0,5º
Різниця у вимірах для установки деклінації	8,7 мм
Кут між площиною антени й трубою для установки елевації	2,7º
Різниця у вимірах для установки елевації	46,0 мм

Таблиця 2.2 – Розрахунок кутів для налаштування супутникової антени

Параметр	**Значення**
Місце установки антени	Кропивницький
Широта місцевості	
Довгота місцевості	
Нахил полярної осі щодо вертикалі (елевація)	33,9º
Радіус землі	6378 км
Радіус геостанціонарной орбіти	42233
Відношення радіуса геостаціонарної орбіти до радіуса землі	6,62
Малий діаметр антени	900 мм
Великий діаметр антени	980 мм
Кут офсетності антени (для прямофокусних =0)	23,3º
Коригувальний кут полярної підвіски	0,62º
Деклінація (відкоректована)	7,2º
Елевація (відкоректована)	33,2º

Таблиця 2.3 – Внесені дані для супутників

Супутник	Довгота супутника	Поворот полярки	Азимут	Кут піднесення	Відстань супутника
NSS	-177,0º	–	–	–	–
EchoStar 4	-157,0º	–	–	–	–
EchoStar 1/2	-148,0º	–	–	–	–
AMC 11	-146,0	–	–	–	–
AMC 8	-139,0	–	–	–	–
AMC 7	-137,0	–	–	–	–
AMC 10	-135,0º	–	–	–	–

У розрахунку є незнайома цифра – кут вигину хобота мотора. Цю цифру варто пошукати в паспорті на мотор, для SRT DM-2100 цей кут дорівнює 30 градусів. Найчастіше цього кута дотримуються й всі інші виробники мотоподвісів.

Процедура установки супутникової антени.

Знадобиться тільки дзеркало, піддзеркальник, елементи підвіски й всі хомути кріплення до опори. Збираємо все це й на місце кріплення до опори прикручуємо відрізок труби. Тепер нам потрібно виміряти відстань від труби до краю антени зверху й знизу, стежите щоб лінійка була як можна більш точно перпендикулярна площини раскрива антени.

Заглядаємо в табличку розрахунку й шукаємо цифру «Різниця у вимірах для установки деклінації». От на таку величину повинні розрізнятися відстані від краю антени до труби зверху й знизу. Якщо в розрахунку цифра вийшла позитивна, то зверху відстань повинне бути більше чим знизу на розрахункову величину. Якщо негативна, то навпаки. Тепер акуратно затягуємо болти кріплення підвіски. Не забудьте після затягування болтів ще раз проконтролювати розміри, вони можуть зміститься при затягуванні гайок. Відкручуємо відрізок труби й на її місце встановлюємо мотор.

При установці мотора варто звернути увагу на те, що в нього є два рознімання в нижній частині до яких буде підключатися кабель. Підвіска антени повинна кріпитися якнайближче до підшипника хобота, так ми максимально розвантажимо мотор. У теж час не забувайте про кабель, якщо ви підсунете підвіску занадто близько, те кабель може бути ушкоджений під час обертання мотора. Для підключення кабелю до мотора непогано б застосувати Г-образні переходники. Мотор варто виставити строго перпендикулярно площини раскрива антени. І не забувайте що хобот мотора повинен перебувати в нульовій позиції. Для визначення перпендикулярності мотора можна замірити відстані із двох сторін від мотора до країв антени, вони повинні бути однакові.

Тепер прикручуємо відрізок труби на місце кріплення мотора до опори.

І повторюємо цю процедуру як при налаштуванні деклінації, тільки використовуємо іншу цифру з таблиці «Різниця у вимірах для установки елевації». Залишається перевірити на яку оцінку встав покажчик по шкалі «ELEVATION» на підвіску мотора. Ця оцінка повинна майже збігтися з розрахункової элевацией, все залежить звичайно від точності виготовлення цієї шкали й точності пророблених вами налаштувань. Наступним кроком буде визначення верхового для вашої місцевості супутника. Для цього заглядаємо в таблицю кутів, азимутів і т.д.

Шукаємо той супутник, для якого кут повороту полярної підвіски самий маленький. Ще одним необхідним критерієм вибору верхового супутника є рівень сигналу, з яким він приймається у вашій місцевості, чим сильніше, тим краще. От на цей кут ми зараз і повернемо хобот нашого мотора. Негативне значення означає, що потрібно крутити на захід. Виставляти точне значення цього кута не настільки важливо, досить повернути мотор, орієнтуючись по шкалі біля хобота мотора. Однаково не вдасться виставити точний напрямок на південь, орієнтуючись по цьому куту й напрямку на супутник. Напрямок на південь будемо відбудовувати іншим способом уже на місці.

Таблиця 2.4 – Кути й азимути для супутників

Hellas Sat	39,9º	-5,7º	186,3º	25,9º
Express A1R	40,0º	-4,6º	185,1º	25,9º
Turksat 1C Eurostatsar 1	42,0º	-2,4º	182,6º	26,0º
NewSat 1	42,5º	-1,9º	182,0º	26,0º
Europe Star 1	45,0º	0,9º	179,0º	26,0º
Eutelsat W3 & EutelSat II 2	48,0º	4,1º	175,4º	26,0º
Yamal 202	49,0º	5,2º	174,2º	25,9º
IntelSat 602	50,5º	6,9º	172,4º	25,0º
Express AM 22	53,0º	9,6º	185,4º	25,6º

Тепер, коли основні маніпуляції з підвіскою зроблені можна водрузити на місце штангу конвертеродержача й сам конвертер і відкласти антену. Конструюємо опору. Конструкцію й розміри опори вам доведеться придумувати самим, орієнтуючись по місцю установки. Вона повинна відповідати наступним вимогам:

1. Забезпечувати максимальний розворот антени, не забувайте що в нульовій позиції вона буде дивитися строго на південь.

2. Забезпечувати максимальну твердість.

3. Забезпечити вертикальність труби, на яку буде вішатися антена.

4. Забезпечити надійність кріплення опори до стіни або іншої підстави.

При установки мотоподвіса на опорі, що не має вузла для регулювання вертикальності, цілком достатньо при розмітці отворів під анкери проконтролювати вертикальність і при необхідності підігнути нижні откосини або підкласти під них шайби.

Вертикальну трубу, на яку будемо кріпити антену, варто ретельно виставити за рівнем, вона повинна бути строго вертикальна в усіх напрямках. Вішаємо антену в зборі на опору. Залишилася справа зовсім за малим, виставити напрямок на південь. Для налаштування вам знадобиться який-небудь індикатор, по якому ви зможете оцінити точність орієнтації антени. Як цей індикатор може послужити САТФАІНДЕР, шкала рівня по приймачу на екрані телевізора, динамік або навушники підключені до звукової карти комп'ютера на якому запущена програма Fast SatFinder від Mixe або спектроаналізатор. Все ваше встаткування потрібно включити, виставити на приймачі параметри транспондера з найдужчим сигналом з верхового супутника й можна приступати до юстировки антени. Плавно повертаючи всю конструкцію разом з мотором навколо опори стежимо за показанням індикатора. З появою показань, що свідчать про налаштування на супутник, доможіться їхнього максимуму й зафіксуйте антену в такому положенні. Якщо для налаштування ви використовуєте САТФАІНДЕР, то необхідно впевниться, на чи той супутник ви потрапили. Якщо все в порядку й ви потрапили на потрібний супутник, спробуйте злегка подати антену спочатку нагору, а потім долілиць, природно не відвертаючи при цьому ніякі гайки. Сигнал при цих маніпуляціях повинен зменшуватися однаково в обох напрямках. Якщо сигнал збільшується при подачі антени нагору, це швидше за все результат провису підвіски й мотора під вагою всієї конструкції. У такому випадку варто послабити гвинти кріплення елевації й подрегулювати антену до одержання максимального сигналу й знову затягти гвинти. Якщо ж сигнал збільшується при подачі антени долілиць, то це швидше за все результат неточної установки кутів або невертикальність опори. Якщо опора свідомо невертикальна в напрямку північ – південь, то це можна виправити знов-таки подкоректувавши кут елевації. Наступним кроком буде контроль сигналу на крайніх супутниках на заході й сході. Для цього вам потрібно вибрати два супутники із сильним сигналом із близькими за значенням кутами повороту полярної підвіски на захід і схід. Повертаємо антену, уже за допомогою мотора, на один із цих супутників

контролюючи кут повороту по шкалі мотора. Настроївшись на супутник, перевіряємо й запам'ятовуємо зміни сигналу при подачі антени нагору й долілиць. Повертаємо антену на другий супутник і також запам'ятовуємо зміна сигналу при подачі антени нагору й долілиць. Тепер порівнюємо зміни сигналу при подачі антени нагору й долілиць на східному й західному супутниках.

1. Якщо сигнал зменшувався в обох напрямках і на західному й на східному супутнику, вас можна поздоровити, напрямок на південь у вас настроєно дуже точно й на цьому налаштування можна вважати завершеної.

2. Якщо на західному супутнику сигнал збільшувався при подачі наверх, а на східному збільшувався при подачі долілиць, то у вас підвіска спрямована на схід напрямку на південь. Поверніть всю конструкцію разом з мотором небагато на захід і повторіть усе заново.

3. Якщо на західному супутнику сигнал збільшувався при подачі долілиць, а на східному збільшувався при подачі нагору, то у вас підвіска спрямована на захід напрямку на південь. Поверніть всю конструкцію разом з мотором небагато на схід і повторіть усе.

4. Якщо ж на сході й на заході сигнал збільшується при подачі антени наверх, то кут елевації великий і його потрібно зменшувати, а кут деклінації збільшувати

5. Якщо ж на сході й на заході сигнал збільшується при подачі антени долілиць, то кут елевації малий і його потрібно збільшувати, а кут деклінації зменшувати.

Якщо у вас вийшов 4 або 5 варіант, то ви допустили неточність при налаштуванні підвіски й вам довед еться проробити операцію з поворотом антени на захід і схід кілька разів потроху змінюючи кути елевації й деклінації, не забувайте контролювати сигнал на верховому супутнику. Якщо 2 або 3 варіант, то повернувши антену разом з мотором на невеликий кут проробіть все заново. Не виключено що в підсумку у вас може вийти 4 або 5 варіанти. Проробляючи всі маніпуляції ви повинні чітко собі представляти що ви робите.

Якщо вам доводиться змінювати кут елевації не забувайте в протилежному напрямку міняти кут деклінації. Найкраще це робити так: на крайньому супутнику підбудовуємо деклінацію до максимального рівня сигналу, вертаємося на верховий супутник і також по максимуму сигналу відбудовуємо елевацію. Цю операцію потрібно буде проробити кілька разів.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Embarcadero Delphi, раніше Borland Delphi і Codegear Delphi, – інтегроване середовище розробки ПЗ для Microsoft Windows, Mac OS, iOS і Android мовою Delphi (що раніше носила назву Object Pascal), створена спочатку фірмою Borland і на даний момент приналежна й розроблювальна Embarcadero Technologies. Embarcadero Delphi є частиною пакета Embarcadero RAD Studio і поставляється в чотирьох редакціях: Community (поширюється безкоштовно й має обмежену ліцензію на використання в комерційних цілях), Professional, Enterprise і Architect.

Delphi 10.4 Sydney

Випущено 26 травня 2020 року. RAD Studio Delphi 10.4 забезпечує значно поліпшену високопродуктивну нативну підтримку Windows, кращу продуктивність розробки, миттєві підказки code completion, прискорення виконання коду із синтаксисом керованих записів, поліпшення виконання паралельних завдань на сучасних багатоядерних CPU, а також містить більш 1000 виправлень багів, поліпшення продуктивності середовища й бібліотек і багато чого крім того.

Основні можливості Delphi 10.4.1:

− Істотні розширення для Windows: поліпшення для застосунків на моніторах 4K High DPI, інтеграція з новим WebView2 на базі Chromium, використання розширених title bars, таких же, як в Offi ce, Explorer, Google Chrome.

– Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

– Істотне поліпшення Delphi Code Insight (без можливого блокування IDE – в окремому процесі), що допоможе при роботі з великими проектами.

– Тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання.

– Розширена підтримка бібліотек C++: ZeroMQ, SDL2, SOCI, libSI MDpp і Nematode.

– Відладник Win 64 (на LLDB) і збирач для C++.

– Поліпшення для C++: Включена велика кількість поліпшень STL з Dinkumware.

– Підтримка Metal Driver GPU для macOS і iOS.

– Вбудований Fmxlinux.

– Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API. Реалізація компонента Media Player для macOS тепер використовує Avfoundation. Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.

– Численні поліпшення швидкості й стабільності роботи нашої бібліотеки The Parallel Programming Library (PPL).

– Додані оновлені драйвери для FireBird, PostgreSQL і SQLite.

– Клієнтські бібліотеки HTTP і REST Client розширені застосунковими можливостями роботи з HTTPS. Також були розширені можливості підтримки Amazon AWS services

– У технологію Visual LiveBindings внесена безліч поліпшень, у тому числі швидкодії, що стосуться, застосунків на VCL і FireMonkey

RAD Studio 10.4 Короткий огляд:

– Істотні розширення для Windows. Створення застосунків, що чудово виглядають, із чіткими елементами інтерфейсу на 4k моніторах High DPI за

допомогою нової гнучкої підтримки стилів елементів керування на екрані. Інтеграція із сучасними, безпечними web-технологіями від Microsoft – новим WebView2 на базі Chromium. Використання сучасних розширених title bars, таких же, як в Office, Explorer, Google Chrome, у своїх проектах. Істотні поліпшення надійності налагодження в новому відладнику для C++ Windows 64-bit.

– Зросла продуктивність розробки. Ріст продуктивності за рахунок миттєвої реакції підказок code completion у середовищі IDE. Краща сумісність із уже наявною кодовою базою, і спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю. Швидке зв'язування даних і візуальних елементів за допомогою розширеної технології Visual LiveBindings з підвищеною швидкодією. Просте використання розповсюджених бібліотек C++, наприклад, ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode. Оновлена підтримка Amazon AWS cloud.

– Поліпшення швидкодії і якості. Більш 1000 поліпшень швидкодії і якості. Краща ефективність коду за допомогою нового синтаксису custom managed records. Більш швидке виконання паралельних завдань на сучасних багатоядерних CPU. Переконаєтеся в прискоренні відображення на екрані з підтримкою Metal API на macOS і iOS. Краща сумісність із уже наявною кодовою базою й спрощення програмування за рахунок уніфікованої архітектури керування пам'яттю.

Істотне поліпшення Delphi Code Insight

Як найбільше й головне поліпшення інструментів програмування Delphi за багато років, в 10.4 Delphi Code Insight реалізований через Language Server Protocol (LSP). LSP – це технологія генерації результатів для code completion, навігації й інших сервісів в окремому процесі. Це значить, що code completion і Code Insight одержать більш точні результати без блокування IDE. 10.4 забезпечує набагато більш високу продуктивність розроблювачів, які працюють із більшими проектами, що містять мільйони рядків коду.

Delphi Custom Managed Records

Ключове розширення мови Delphi: тип даних Delphi «record» тепер підтримуть довільні ініціалізацію, фіналізацію й операції копіювання. Управляйте тем, як ці структури створюються, копіюються й звільняються з допомогу вашого коду, який буде виконуватися у відповідний момент.

Це розширює потужність конструкцій records в Delphi, які використовуться щоб одержати більшу ефективність у порівнянні із класами.

Єдине керування пам'яттю

Керування пам'яттю в Delphi тепер стандартизоване на всіх підтримуваних платформах – мобільних, настільних і серверних – використовучи класичну реалізацію керування пам'яттю об'єктів.

У порівнянні з Automatic Reference Counting (ARC), це дає кращу сумісність із існучим кодом і спрощує написання компонентів, бібліотек і застосунків.

ARC модель керування пам'яттю model залишилася для керування рядками й посиланнями на тип інтерфейсу на всіх платформах. Для C++ це означає, що при створенні й звільненні Delphi-style класів в C++ використовується звичайне керування пам'яттю, як у будь-якого heap-allocated класу C++, що значно знижує складність коду.

Розширена підтримка бібліотек C++

В 10.4 ми портували багато популярних бібліотек C++ у C++Builder.

Забезпечивши оптимізовану підтримку бібліотек ZeroMQ, SDL2, SOCI, libSIMDpp і Nematode, поряд із уже підтримуваними Boost і Eigen, які можуть бути додані за допомогою менеджера пакетів Getit.

Win 64-відладник і збирач для C++

В 10.4 з'явився новий відладник C++ для Windows 64 -bit. Відладник заснований на LLDB і показує значне збільшення стабільності при налагодженні 64-bit застосунків поряд з новими відладочними можливостями, такими як перегляд і інспекція типів начебто рядків C++ і Delphi, а також колекцій STL,

включаючи std::vector, std::map і інших. Крім того, згенерована для застосунку відладочна інформація має інший внутрішній формат, сприяючи більш стабільному й багатому на можливості процесу налагодження, більш докладним перегляду й інспекції в debug-time.

Підвищення якості й швидкодії інструментів

– Велика кількість поліпшень STL від Dinkumware.

– Поліпшені деякі найважливіші методи й області RTL, на базі поліпшень сумісності з популярними бібліотеками C++.

– Поліпшена підтримка Cmake.

– Велика кількість виправлень для підвищення стабільності і якості.

– Відновлення Windows API – Обновлено й додали безліч декларацій API щоб добитися ще більшої інтеграції із платформою Windows.

– Загальні вдосконалення в бібліотеці доступу до БД FireDAC, включаючи оновлені драйвера для FireBird, PostgreSQL і SQLite. Вибір статичного або динамічного підключення SQLite до застосунку.

Змінені стилі VCL для High DPI

В 10.4, архітектура стилізації VCL була суттєво розширена для підтримки High DPI і 4K моніторів. Тепер усі елементи UI на формі VCL автоматично масштабуться під відповідне до монітора дозвіл для показу форми. Був оновлений API стилізації для підтримки стилів high DPI.

Кожний графічний елемент UI може бути обраний з наборів різних масштабів і масштабований до потрібного DPI, що дає чітке зображення елементів UI на всіх моніторах.

Нові High DPI стилі й стилізація окремих VCL компонент

Обновлено велике число вбудованих і преміальних VCL стилів для підтримки нового режиму стилізації High-dpi. Це дозволяє вам створювати застосунку з відмінним дизайном для всіх моніторів.

Розроблювачі VCL застосунків тепер можуть використовувати трохи VCL стилів на різних формах в одному застосунку або в різних компонентів на одній

формі. Це також включає стилізацію компонентів загальною темою для платформи. Крім застосункової гнучкості використання стилів, це дозволяє використовувати нестилізуємі компоненти із зовнішніх бібліотек в VCL застосунках, що використовуть стиль.

Поліпшена кроссплатформеність

– Додана підтримка Metal Driver GPU для macOS і iOS.

– Крім підтримки останнього iOS SDK, в RAD Studio 10.4 розроблювачі можуть задовольнити нові вимоги Apple до набору стартових екранів.

– Реалізований заново стилізуємий FMX компонент TMemo на платформі Windows значно поліпшений і тепер має відмінну підтримку IME.

– Користувачам редакцій Enterprise або Architect доступна повна інтеграція Fmxlinux з IDE для створення клієнтських застосунків Linux з GUI.

– Компонент Twebbrowser для iOS тепер реалізований на Wkwebview API.

– Реалізація компонента Media Player для macOS тепер використовує Avfoundation.

Оновлений менеджер пакетів Getit

Менеджер пакетів Getit в IDE був значно вдосконалений.

Дати випуску релізів пакетів тепер видні, і можливе сортування списку по цих датах; відбір тільки встановлених пакетів, контенту, доступного тільки при наявності підписки, багато чого іншого.

Універсальний інсталятор для установки Online і Offline

В 10.4 включений новий універсальний інсталятор, який використовує технологію на базі Getit. Цей інсталятор підтримує як online, так і offline (з ISO) варіанти установки.

Тепер обоє варіанта установки дозволяють вам указати початковий набір можливостей RAD Studio для установки, наприклад, свою комбінацію мов програмування й цільових платформ, мов інтерфейсу, і додавати до нього або видаляти непотрібне в будь-який момент.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, реалізації підлягає програмне забезпечення, яке призначено для системи керування супутниковим HD ресивером на базі процесору GX6605S.

В процесі розробки випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти необхідно виконати наступний обсяг роботи:

а) провести аналіз існуючих систем-аналогів для виявлення їх позитивних і негативних якостей. Результати аналізу врахувати в подальших розробках;

б) вибрати та обґрунтувати методику побудови системи контролю роботи технологічного обладнання на виробництві в автоматизованому режимі. Розробити функціональну та структурну схеми системи;

в) розробити програмне забезпечення системи, що дозволить реалізувати поставлену технічним завданням задачу. Побудувати блок-схеми алгоритмів програми та підпрограми;

г) організувати інтерфейс користувача з метою формування та виводу на екран ЕОМ повідомлень про некоректні дії користувача та нестандартні ситуації в роботі технологічного обладнання;

д) розробити рекомендації по організаційних та методичних заходах, які забезпечать впровадження системи в промислову експлуатацію та її подальшу успішну експлуатацію;

е) провести розрахунки по визначенню економічної ефективності розробленої системи;

ж) розробити заходи по охороні праці при впровадженні та експлуатації системи, а також розробити заходи з цивільного захисту;

з) сформувати висновки про виконаний обсяг робіт та одержані результати.

3 ОПИС І ОБҐРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ

3.1 Опис функціонування системи

Проектування пристрою вводу/виводу

МП GX6605S є поліпшеною модифікацією попередніх МП та спеціально розроблена в ньому система введення/виведення інформації забезпечує необхідну функціональність. От чому свій вибір і зупинив саме на цій мікросхемі.

GX6605S забезпечує введення/виведення інформації паралельної інформації, застосовується як елемент загального призначення, що сполучає різні типи периферійних пристроїв з магістраллю даних систем обробки інформації.

Обмін інформацією між магістраллю даних систем і мікросхемою здійснюється через 8 розрядний двонаправлений трьохстабільний канал даних. Для зв'язку з периферійними пристроями використовується 24 лінії В/В, згруповані в три 8 розрядних канали ВА, ВВ, ВС, напрямок передачі інформації й режими роботи яких визначаються програмним способом.

Мікросхема може функціонувати в 3-х основних режимах.

У режимі 0 забезпечується можливість синхронної програмно керованої передачі даних через 2 незалежних 8 розрядних канали ВА, ВВ і два 4 розрядних канали ВС.

У режимі 1 забезпечується можливість вводу або виводу інформації в/або з периферійного пристрою через 2 незалежних 8 розрядних канали ВА, ВВ по сигналах квітирування. При цьому лінії каналу С використовуються для прийому й видачі сигналів керування обміном.

У режимі 2 забезпечується можливість обміну інформацією з периферійними пристроями через двохнаправлену 8 розрядну шину ВА по сигналах квітирування. Для передачі й прийому сигналів керування обміном використовуються 5 ліній каналу ВС.

Вибір відповідного каналу й напрямок передачі інформації через канал визначається сигналами A0, A1 і сигналами $\overline{RD}$, $\overline{WR}$, $\overline{CS}$. Режим роботи кожного з каналів ВА, ВВ, ВС визначається вмістом регістра керуючого слова (РКС). Роблячи запис керуючого слова в РКС можна перевести мікросхему в один з 3 -х режимів роботи: режим 0-простий В/В; режим 1-стробуємий В/В; режим 2-двонаправлений канал. При подачі сигналу SR РКС установлюється в стан, при якому всі канали настроюються на роботу в режимі 0 для вводу інформації. Режим роботи каналів можна змінити як на початку, так і в процесі виконання працюючої програми, що дозволяє обслуговувати різні периферійні пристрої в певному порядку однією мікросхемою. При зміні режиму роботи будь-якого каналу всі вхідні й вихідні регістри каналів і тригери стану скидаються.

У нашому випадку необхідно запрограмувати мікросхему на вивід інформації у режимі 0. Саме тому далі буде розглянутий тільки цей режим.

При роботі мікросхеми у режимі 0 забезпечується простий В/В інформації через кожний з 3-х каналів і сигналів керування обміном інформацією з периферійними пристроями не потрібно.

У цьому режимі мікросхема являє собою сукупність 2 -х 8 розрядних і 2-х 4 розрядних каналів введення або виведення. У режимі 0 можливі 16 різних комбінацій схем введення/виведення каналів ВА, ВВ, ВС. Це визначається комбінаціями в розрядах D4; D3; D1; D0 регістра керуючого слова.

Для нашого випадку код повинен мати наступну вказівку:

D4	D3	D1	D0	BA;BB;BC
0	0	0	0	Виведення

У режимі 0 вхідна інформація не запам'ятовується, а вихідна зберігається у вихідних регістрах до запису нової інформації в канал або до запису нового режиму.

Для електричного з'єднання мікросхеми і схеми керування необхідно:

1) шину даних D0÷ D7 схеми керування з'єднати з виводами D0÷ D7 мікросхеми 580ВВ55.

2) Два молодших розряди адресної шини з'єднати з виводами A0÷ A1 мікросхеми.

3) Виводи $\overline{RD}$, $\overline{WR}$ мікропроцесора GX6605S з'єднати з виводами $\overline{RD}$, $\overline{WR}$ мікросхеми відповідно.

4) На вхід SR “Установка у вихідний стан” мікросхеми подати низький рівень (підключити до корпуса).МП GX6605S має вбудовану статичну оперативну пам'ять (SRAM) розміром 16...64 Кбайт, що може використатися для зберігання коду й/або даних. SRAM може зберігати 8, 16 і 32-бітні дані.

Вміст SRAM при скиданні МП зберігається. Контролер SRAM має у своєму складі буфер відкладеного запису, необхідний для забезпечення роботи центрального процесора без його зупинки на час запису в SRAM. У буфері відкладеного запису завжди втримуються останні дані, передані програмою для запису в SRAM.

Уміст буфера записується у SRAM, коли програма передає у буфер наступну порцію даних, призначених для приміщення у SRAM. Якщо в деякий момент часу відбувається скидання мікроконтролера, то остання поміщена в буфер відкладеного запису порція даних буде загублена. Щоб уникнути цього, а також перед перекладом мікроконтролера у режим зниженого споживання розроблене програмне забезпечення періодично робить фіктивну операцію запису. Це дає гарантію того, що останні записані дані будуть перебувати в SRAM і після скидання.

Проектування фіксуючої схеми

В блок індикації необхідно подавати сигнали № каналу (2 індикатори) у строго певні моменти часу.Для цього необхідно передбачити пристрій, що по сигналах від процесора, буде пропускати інформацію на один з індикаторів блоку індикації. Як елементи фіксуючої схеми будемо використовувати 2 регістри.

	ЕО	С	D_n	Вихід
Завантаження й зчитування	Н	↗	“Н”, “В”	“Н”, “В”
Завантаження регістра й розрив виходів	В	↗	“Н”, “В”	відповідно

Рисунок 3.1 – Регістр приймає й відображає інформацію синхронно з позитивним перепадом на тактовому вході

Таким чином, подаючи тактуючі сигнали на вхід С (№11) регістра 1533UP23, ми дозволяємо проходження сигналів на відповідний індикатор у строго певні моменти часу.

Проектування схеми, що погоджує

Для організації виводу інформації в інші блоки тюнера будемо використовувати регістр тактуємий сигналами від мікропроцесора.

Для прийому інформації до пристрою керування будемо використовувати шинний формувач. Як відомо шинний формувач забезпечує передачу інформації в обох напрямках.

Для забезпечення тільки вводу даних вивід №1 з'єднаємо з корпусом. Якщо з'явиться необхідність у виводі більшої кільк ості інформації із пристрою керування, то за допомогою мікросхеми SN74ALS245 можна буде вирішити дану проблему.

Проектування схеми дешифрації

Раніше були розглянуті основні блоки схеми керування й було відзначено, що МП у строго певні моменти часу повинен взаємодіяти з певними мікросхемами. Тому в даній схемі необхідно передбачити пристрій, що по сигналах від процесора, буде підключати до його шин адреси або даних ту або іншу мікросхему або групу мікросхем. Із цього можна укласти, що в схемі системи повинен протікати деякий процес однозначного вибору й він організується подачею на лінії адреси A11 ÷ A15 певного коду вибору або

сигналу дозволу доступу до окремого блоку або блоків. На щастя, ця проблема є класичною й вона має просте рішення. Зокрема можна використовувати дешифратор, виконаний у вигляді ТТЛ пристрою середнього рівня інтеграції, призначеного для перетворення двійкового коду в напругу логічного рівня, що з'являється в тім вихідному проведенні, десятковий номер якого відповідає двійковому коду. У наслідку вихідне проведення дешифратора підключають до входу “Вибір мікросхеми” потрібної мікросхеми (наприклад вивід №18 (CS) мікросхеми HM6516-9).

Як дешифратор будемо використовувати мікросхему SN74ALS138. Вибір даного дешифратора обумовлений кількістю вихідних ліній і навантажувальною здатністю.

Мікросхема SN74ALS138 – високошвидкісний дешифратор, що перетворить трьохразрядний код A0 ÷ A2 (№1 ÷ 3) у напругу низького логічного рівня, що з'являється на одному з восьми виходів 0 ÷ 7. Дешифратор має трьохвходовий логічний елемент дозволу.

Як інформаційні сигнали будемо використовувати сигнали, що надходять по адресних лініях A11 ÷ A13; сигнали дозволу, сигнали, що надходять по адресних лініях A14 ÷ A15 (вхід №4 приєднаємо до корпуса).

Проектування цифро-аналогового перетворювача

Для перетворення цифрової інформації в аналогову необхідно використовувати ЦАП. Основною характеристикою ЦАП є розв'язна здатність, обумовлена числом розрядів N. Теоретично ЦАП, що перетворить N-розрядні двійкові коди, повинен забезпечувати 2^N різних значень вихідного сигналу з розв'язною здатністю $(2\ N\text{-}1)^{-1}$.

З динамічних параметрів основними є: час установки вихідного сигналу; f_{max} перетворення.

У нашому випадку необхідно організувати формування 3-х аналогових сигналів ANL1, ANL2 і ANL3, які будуть пропорційні цифровим сигналам на виходах каналу A, B, C мікросхеми 82C55 відповідно. Значить необхідно

передбачити 3 цифро-аналогових перетворювачі. Свій вибір я зупинив на 10 розрядному ЦАП прецизійного типу AD7520. Для побудови повної схеми перетворювача до мікросхеми AD7520 необхідно підключити операційний підсилювач. Як операційний підсилювач будемо використовувати MC1556G, що має схему внутрішньої корекції.

Час реакції на переривання в мікропроцесорній системі критично, як правило, тільки для швидких переривань FI. Значення часу реакції лежить у деякому можливому діапазоні, тобто може бути максимальним і мінімальним. Коли переривання FI дозволені, максимальний час реакції на FI (для самого "поганого" випадку) складається з:

– $T_{syncmax}$ найбільш тривалий час запиту, що може знадобитися на реакцію синхронізатора. Цей час становить два процесорних цикли;

– T_{ldm} час, що вимагається для завершення самої довгої команди. (Сама довга команда – LDM, що завантажує всі регістри, включаючи PC.);

– T_{ldm} дорівнює 20 процесорним циклам у системі з нульовим часом очікування;

– T_{exc} час входу в оброблювач Data Abort. Воно становить три процесорних цикли;

– T_{fiq} час входу в оброблювач FI. Воно становить два процесорних цикли.

Таким чином, повний час очікування для самого "поганого" випадку становить 27 циклів процесора, що небагато менше 0.7 мкс у системі з тактовою частотою процесора 40 Мгц. При обчисленні максимального часу очікування IRQ необхідно враховувати ту обставину, що обробка переривань FI, що мають пріоритет вище, ніж в IRQ, може затримати вхід в оброблювач IRQ.

Мінімальний час очікування для FI або IRQ – це найкоротший час запиту, що може знадобитися на реакцію синхронізатора $T_{syncmin}$, складене згодом T_{fiq}, що в сумі становить чотири процесорних цикли.

Додаткові пояснення до схеми керування

1) Щоб уникнути запису або зчитування “помилкової” інформації під час включення або вимикання напруги живлення в схемі пристрою керування передбачена мікросхема DD8 – чотирьох канальний комутатор цифрових і аналогових сигналів. Кожний ключ має свій вхід і вихід сигналу, а також вхід дозволу проходження сигналу EI. Канал провідності двох направлений. Комутатор CD4066B має опір каналу 80 Ом, опір входу керування 10^{12}Ом. Відкриваюча напруга на вході EI – 3У. Канал пропустить цифрові рівні з амплітудою до $U_{ип}$. Час затримки поширення сигналу 10...25 мс.

2) У схемі керування використовується мікросхема DD6: логічний елемент АБО із двома виходами. Ці функції реалізуються за допомогою мікросхеми SN74ALS32. Також використовується мікросхема DD9: логічний елемент АБО-НІ з одним входом (інвертор). Ці функції реалізуються за допомогою мікросхеми SN54ALS04.

3) При вхідному імпульсному сигналі з пологими фронтом і зрізом імпульс на вході формуючого логічного елемента також не буде прямокутним, оскільки якийсь час ключова схема буде перебувати в підсилювальному режимі. Крім того, на фронті й зрізі вихідного імпульсу будуть присутні посилені перешкоди, що надійшли в “підсилювач” із проведення живлення. Імпульс із зашумленими й несформованими фронтом і зрізом непридатний для перемикання тактових входів тригерів, регістрів і лічильників.

Підвищення K_U формувача до 10^3 разів і більше за рахунок послідовного включення декількох буферних елементів не дає точної прив'язки моменту перемикання до певного граничного вхідного імпульсу.

У таких випадках використовують так звану схему тригера Шмідта, що складається із двокаскадного підсилювача, охопленого слабким позитивним зворотним зв'язком. Тригери Шмідта виявилися незамінними й в інтегральної схемотехніці, як в аналоговій, так і цифровій. Передатна характеристика тригера Шмідта має значний гістерезис.

Вихідний сигнал логічного елемента Шмідта має круті імпульсні перепади, тривалість яких не залежить від швидкості наростання або спаду вхідного сигналу. Імпульсні перепади за часом відповідають моментам, коли вхідний сигнал перевищує напруга спрацьовування $U_{спр}$ і стає менше, ніж напруга відпускання $U_{відп}$. У даній схемі пристрою керування тригер Шмідта – у вигляді мікросхеми SN74ALS14 (DD2).

4) Перш ніж послідовність коротких імпульсів подавати на вхід SID мікропроцесора, необхідно забезпечити гарну стабільність тривалості даних імпульсів, тому що на вході елемента Шмідта всі вони будуть мати різну тривалість.

У складі серій ТТЛ є кілька аналого-імпульсних схем – мультивібраторів, що чекають.

Вони дозволяють розширити тривалість коротких імпульсів, сформувати імпульси потрібної тривалості з гарною стабільністю по тривалості. Свій вибір я зупинив на мікросхемі SN74ALS123 – два чекаючи мультивібратори з можливістю перезавантаження. Кожний мультивібратор має виходи Q і $\overline{Q}$, вхід скидання, 2 входи дозволу запуску: В-Прямій, $\overline{A}$-інверсний.

Тривалість вихідного імпульсу визначається часозадаючими елементами C_τ и R_τ ; $\tau_{вих}$=0,45 R_τ $C.\tau$.

Таким чином розглянувши апаратну частину системи керування супутниковим HD ресивером на базі процесору GX6605S перейдемо до її програмного забезпечення.

3.2 Розробка структурної схеми

Відповідно до поставленого завдання магістерської роботи – розробка програмного забезпечення системи керування супутниковим HD ресивером на базі процесору GX6605S, була розроблена програма, яка дозволяє виконувати

поставлені завдання. Перед розглядом структурної схеми системи, розглянемо роботу супутникового ТБ і розпишемо виниклу проблематику.

Як показано на рисунку 3.2, супутники щодо поверхні землі розташовані під різними кутами геостаціонарної орбіти. Для роботи з ними використовується чотири дорогих конверторних головки зі спеціальним перехідником під комп'ютерну плату (рисунок 3.3).

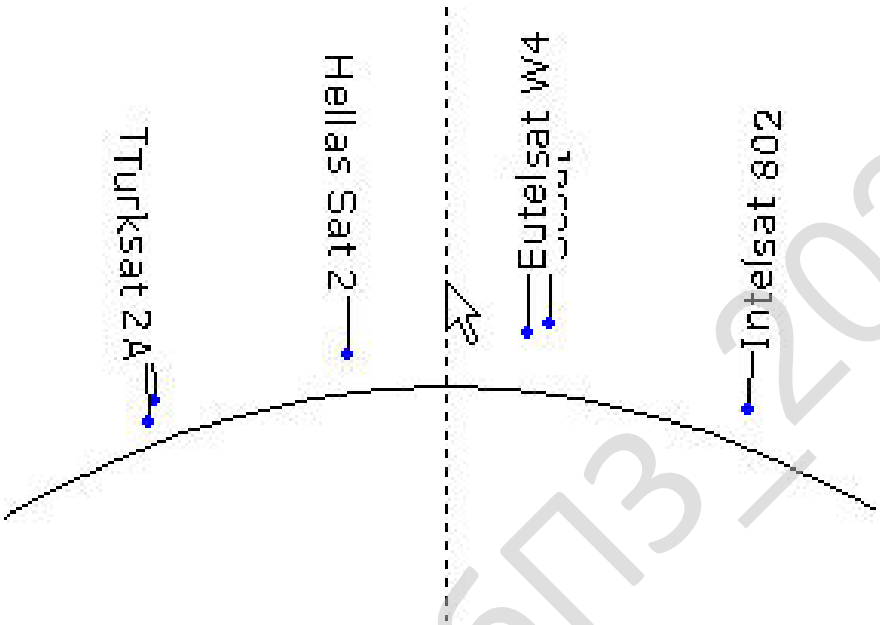

Рисунок 3.2 – Розташування супутників відносно поверхні Землі

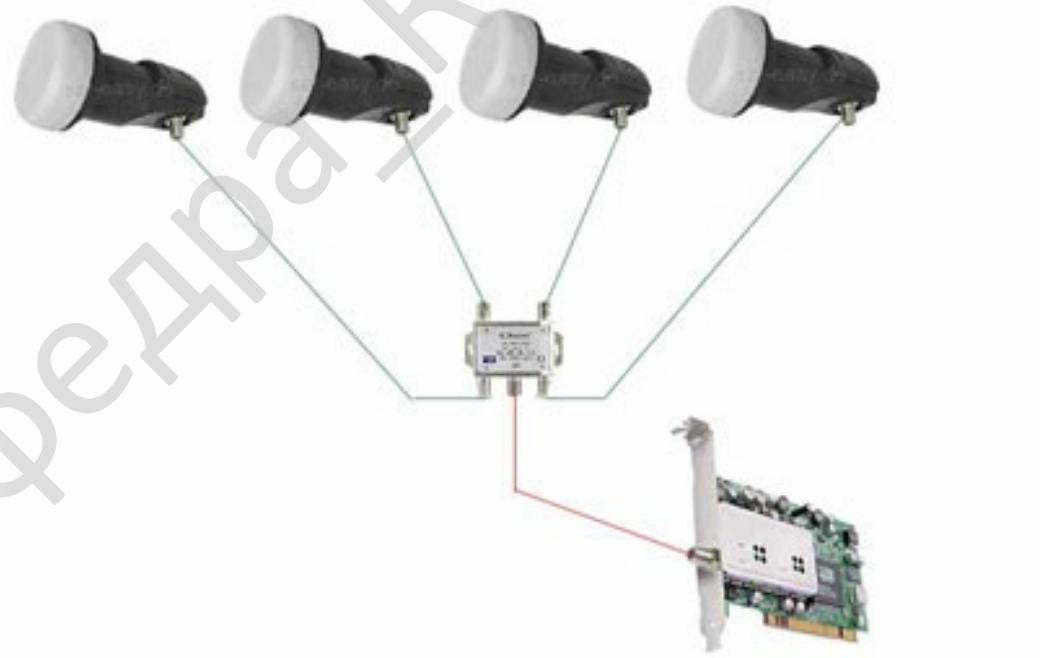

Рисунок 3.3 – Підключення чотирьох дорогих конверторні головки із дзеркалом на різні супутники

Головки направляються під різними кутами до супутників подаючи сигнал, але дана система дорога та має ряд обмежень.

Для подолання проблеми використання великої кількості головок використовують мотоприводи які направляють одну головку на необхідний супутник як показано на рисунку 3.4.

Управляють положенням супутникової тарілки (далі дзеркалом) за допомогою спеціального перемикача (рисунок 3.5), що взаємодіє із ПК через протокол DiSEq 1.2.

DiSEq (Digital Satellite Equipment Control) – спеціальний протокол зв'язку для обміну даними між супутниковим ресивером і іншими пристроями – такими, як: перемикачі, поляризатори, позиціонери й т.п. Для передачі сигналу використовується коаксіальний кабель. Режим обміну даними через кабель – двосторонній, з можливістю подачі живлення. Стандартом передбачена сумісність із традиційним перемиканням напруги 13/17 вольт і тоном 22 кГц.

Рисунок 3.4 – Підключення однієї конверторних головки із дзеркалом і мотоприводом на різні супутники

Протокол DiSEq використовується для керування різною периферією в прийомних системах супутникового ТБ. Команди DiSEq передаються по лінії постійного живлячої напруги 12-20В за допомогою тонових посилок частотою 22кГц (±20%) і номінальною амплітудою 650мВ (±250мВ) при напрузі живлення 13/18В.

DiSEq використовує для передачі широтно-імпульсну маніпуляцію, при якій від ширини огібаючих імпульсів залежить переданий біт. Час передачі одного біта становить 1.5мс і умовно розділено на 3 рівні частини по 500мкс (±100мкс).

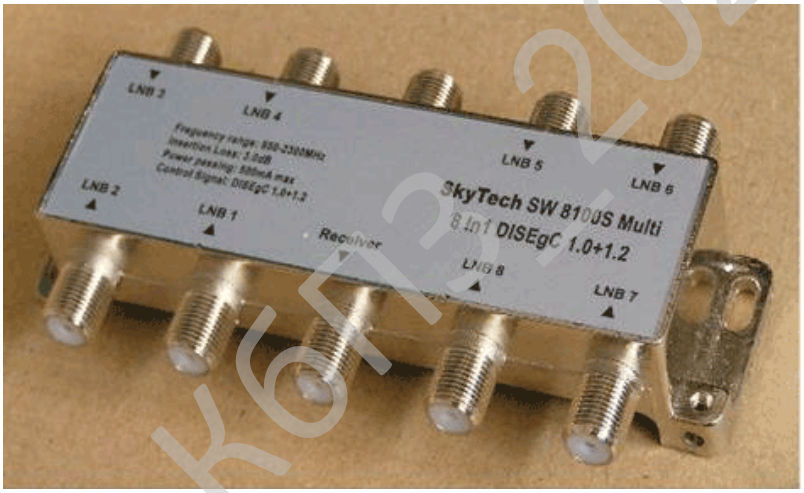

Рисунок 3.5 – Зовнішній вигляд перемикача SkyTech SW8100S, 8LNB

Для біта 0 ширина що обгинає становить 1.0 мс, що відповідає 22 імпульсам, а для біта 1 ширина що обгинає становить 0.5мс, а це 11 імпульсів.

Але в системи з однією конверторною головкою, дзеркалом і мотоприводом є істотний недолік – відсутність універсального програмного забезпечення, яке має можливість автоматично шукати сигнал і набудовувати позиції.

У зв'язку із цим розроблене магістерське програмне забезпечення розроблено з розумінням даного факту, й взаємодіє із протоколом DiSEq 1.2.

При розробці програми використовувався наступний набір устаткування.

1. Супутникова тарілка (дзеркало) – без назви харківського виробництва.

2. Конверторна головка – LNB.

3. Мотопривод – Strong 21100.

4. Тюнер (ресивер) – StarTrack 750CU.

5. Комп'ютерна плата керування мотоприводом – TT-PCline Budget 1401 (SkyStar 3) рисунок 3.6.

6. Перемикач DiSEq – SkyTech SW8100S, 8LNB.

7. Кабелі й кронштейни – без назви харківського виробництва.

Рисунок 3.6 – Використовувана плата TT-PCline Budget 1401 (SkyStar 3)

Розглянемо структурну схему системи зображену на рисунку 3.7, на якій розглянута робота й взаємодія всієї системи в цілому.

Розроблена програма на персональному комп'ютері робить моніторинг роботи системи за допомогою комп'ютерної плати керування. Дзеркало може бути орієнтоване на 16 різних позицій – геостаціонарної орбіти супутників, згідно протоколу DiSEq 1.2.

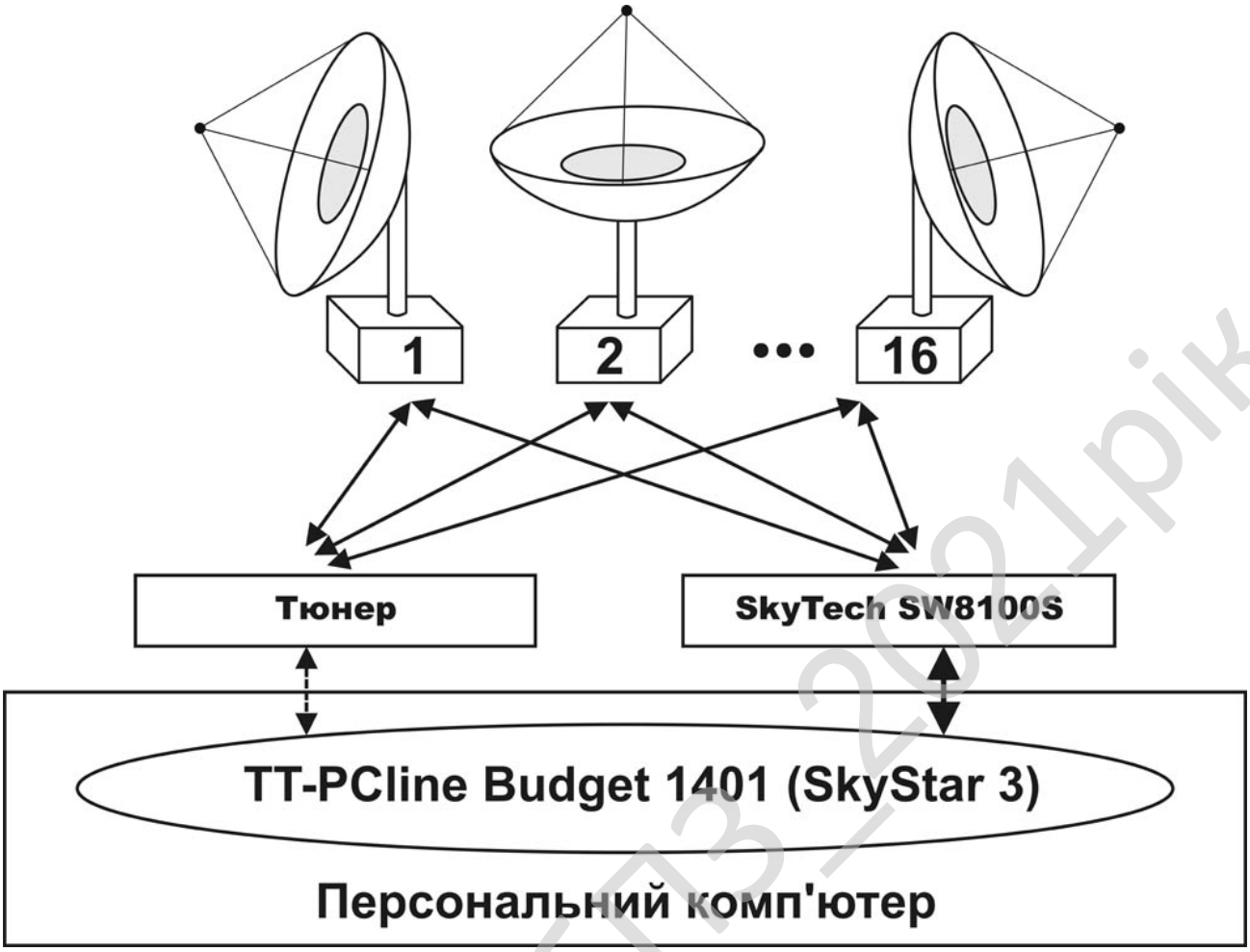

Рисунок 3.7 – Структурна схема системи

У свою чергу супутниковий тюнер управляє й відображає отриману картинку залежно від бажання користувача на ПК або на звичайний телевізор. Комп'ютерна плата керування теж може відігравати роль тюнера але через погане відображення картинки був обраний варіант із зовнішнім тюнером, що дає додаткову перевагу переклад сигналу або його дублювання з монітора комп'ютера на телевізор.

3.3 Розробка функціональної схеми

На рисунку 3.8 розглянута функціональна схема апаратної частини плати контролю, схема дистанційного керування (ДК) генерує послідовність коротких імпульсів ІЧ випромінювання, відповідно до натиснутої кнопки на панелі ДК.

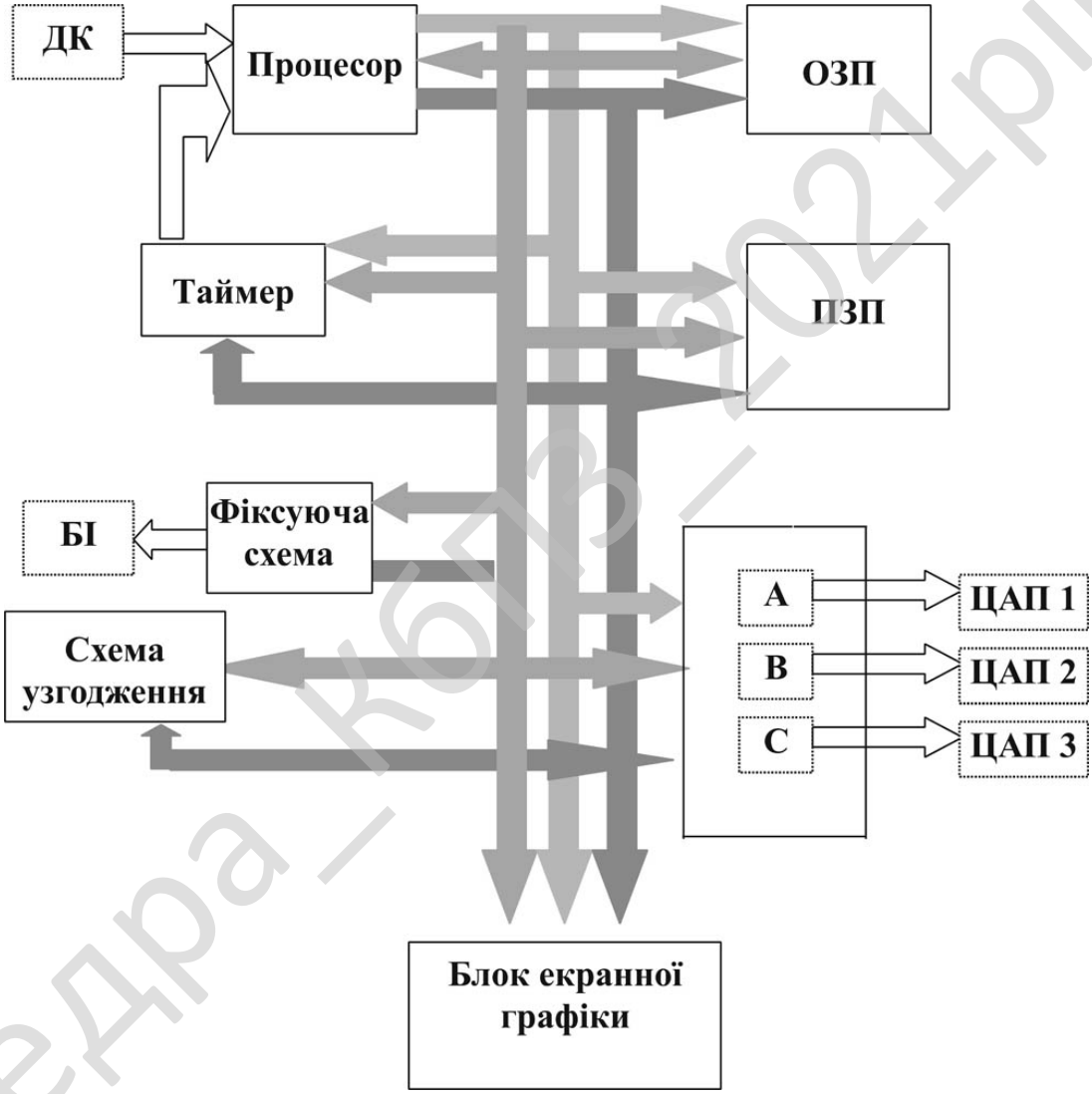

Рисунок 3.8 – Функціональна схема системи

БІ – блок індикації

ОЗП – оперативний запам'ятовувальний пристрій

ПЗП – постійний запам'ятовувальний пристрій

ДК – дистанційне керування

Кожна послідовність складається з 14 імпульсів, з яких 11 і мпульсів інформаційних, а також імпульси: попередній, що запускає й зупиняє.

За допомогою 11 інформаційних імпульсів, ми передаємо сигнал ДК, що являє собою десятибітове слово. Його чотири перших біти відведені для передачі адреси, а інші для передачі команди. У такий спосіб можна сформувати 16 груп адрес по 64 команди в кожній (у нашім випадку будемо використовувати 16 команд із однією строго заданою адресою).

Двійкова інформація кожного біта визначається тривалістю інтервалів між імпульсами. Логічному “0” відповідає основний інтервал часу Т, логічній “1” – 2Т. Часовий інтервал між попереднім і імпульсами, що запускає – 3Т, між тим, що запускає й першим інформаційним – Т, між останнім інформаційним і тим, що зупиняє – 3Т.

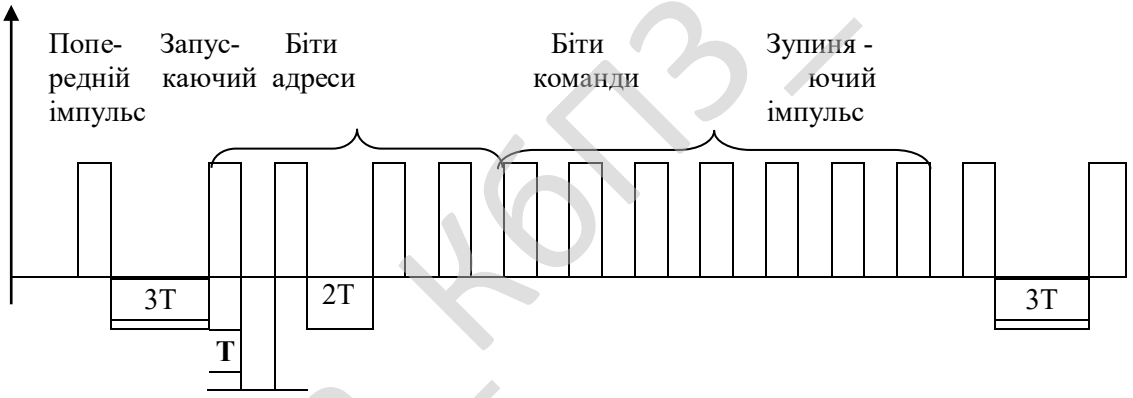

Рисунок 3.9 – Часова діаграма керуючого імпульсу

Дана інформація надходить у процесор, функції якого:

1) Прийняти сигнали ДК;

2) Виділити біти команди;

3) Визначити якій кнопці ДК відповідає дана команда;

4) Забезпечити виконання даної команди, управляючи й синхронізуючи діяльність всього пристрою керування.

Як відомо процесор виконує всі дії відповідно до програми, що зберігається в ПЗП. Питання запису програми в ПЗП в цьому випадку

розглядатися не будуть. Значить для функціонування процесору необхідно зчитувати інформацію (програму), що зберігається в ПЗП. Для цього процесор з'єднаний із ПЗП трьома шинами:

1) шиною адреси;

2) шиною даних;

3) шиною керування.

Для зчитування інформації із ПЗП необхідно виконати наступні дії:

1) забезпечити стабільність рівнів сигналів на адресній шині;

2) підготувати шину даних для прийому даних у мікропроцесор;

3) після кроків 1 і 2 активувати шину керування читанням з пам'яті.

Значить мікропроцесор обробляє сигнали ДК, відповідно до програми, що зберігається в ПЗП.

Так як в процесі виконання програми будуть формуватися дані, що знадобляться для подальшого функціонування схеми пристрою керування, то потрібно передбачити додаткову область пам'яті, де ці дані будуть зберігатися й звідки при необхідності будуть зчитуватися. Для цього в даній схемі використовується ОЗП.

Відмінною рисою ОЗП від ПЗП є те, що дані з ОЗП можуть не тільки зчитуватися, але й записуватися в ОЗП.

Для сполучення мікропроцесора й ОЗП використовуються ті ж 3 шини:

1) шина адреси;

2) шина даних;

3) шина керування.

Зчитування даних з ОЗП аналогічно зчитуванню даних із ПЗП, а для запису необхідно виконати наступні дії:

1) на адресній шині повина бути активований адреса пам'яті (тобто адреса осередку, куди записуються дані);

2) на шину даних повинні надійти дані з мікропроцесора;

3) після здійснення дій 1 і 2 на лінію запису до пам’яті шини керування повинен надійти імпульс дозволу запису.

Вивід: Мікропроцесор обробляє сигнали ДК й “приймає” рішення відповідно до програми, що зберігається в ПЗП. Дані, які з'являються в процесі виконання програми, зберігаються в ОЗП.

Таким чином, на рівні блок-схеми розглянуті 4 блоки пристрою керування, їхньої функції й сполучення між собою.

Для кращого розуміння функціонального призначення інших блоків пристрою керування спочатку познайомимося із класифікацією сигналів, що надходять із ДК:

1) сигнали ДК, відповідно до яких відбувається включення необхідного каналу з наступним налаштуванням на потрібну частоту відео, звуку й налаштуванням на відповідну поляризацію. Якщо на потрібному каналі вже зроблене налаштування на потрібну частоту відео й звуку й налаштування на відповідну поляризацію, ці дані зберігаються в ОЗП й зчитуються при включенні відповідний канал.

2) сигнали ДК, якими можна управляти годинниками реального часу з будильником і календарем.

3) сигнал ДК, яким можна виключити систему в цілому.

Значить необхідно, щоб пристрій керування, аналізуючи сигнали з ДК відповідно до програми, що зберігається із ПЗП, виконував наступні функції:

1) видавав аналогові сигнали в блоці налаштування відео, звуку й поляризації. Для цього необхідно забезпечити сполучення периферійних пристроїв із шиною дані пристрої керування й перетворити цифрові сигнали в аналогові. Як пристрій, що виконує дані функції, будемо використовувати програмний пристрій В/К паралельній інформації (містить 3 вихідних канали) і 3 цифро-аналогових перетворювачі. Таким чином, на виході ЦАП будемо мати аналоговий сигнал пропорційний коду на вході відповідного каналу. У

подальшому цей сигнал можна використовувати в блоках налаштування відео, звуку, поляризації.

2) видавав сигнали в блок індикації для візуального контролю. Для цього в даному пристрої керування необхідно передбачити блок, що буде фіксувати сигнали, що надходять по шині даних у відповідні моменти часу.

3) забезпечував організацію годинника реального часу з будильником і календарем з наступною подачею сигналів у блок екранної графіки й процесор. Для цього необхідно в пристрої керування використовувати таймер, що виконує дані функції.

4) забезпечував видачу й прийом сигналів до інших блоків тюнера. Для цього необхідно передбачити блок, що погодить внутрішню шину даних пристрою керування із зовнішніми блоками тюнера у відповідні моменти часу.

3.4 Розробка діаграми процесів

Важливим критерієм при розробці програмного забезпечення це є грамотна розробка структури роботи системи. На діаграмі процесів зображеній на рисунку 3.10 можна точно зрозуміти як працює і взаємодіє розроблене програмне забезпечення. в цілому.

Починається і закінчується програма в першому блоці це є основною крапкою відрахунку діаграми. При переміщенні по стрілках можна побачити загальну схему взаємодії блоків і їх входження один в одного.

Як можна побачити з основного блоку програми управління переходить спочатку до інтерфейсу ПЗ потім з цим блоком взаємодіють модулі Налагодження ПЗ, Дані координат супутників та встановлення координат супутників які можуть бути встановленні у ручному режимі введення координат та у автоматичному режимі.

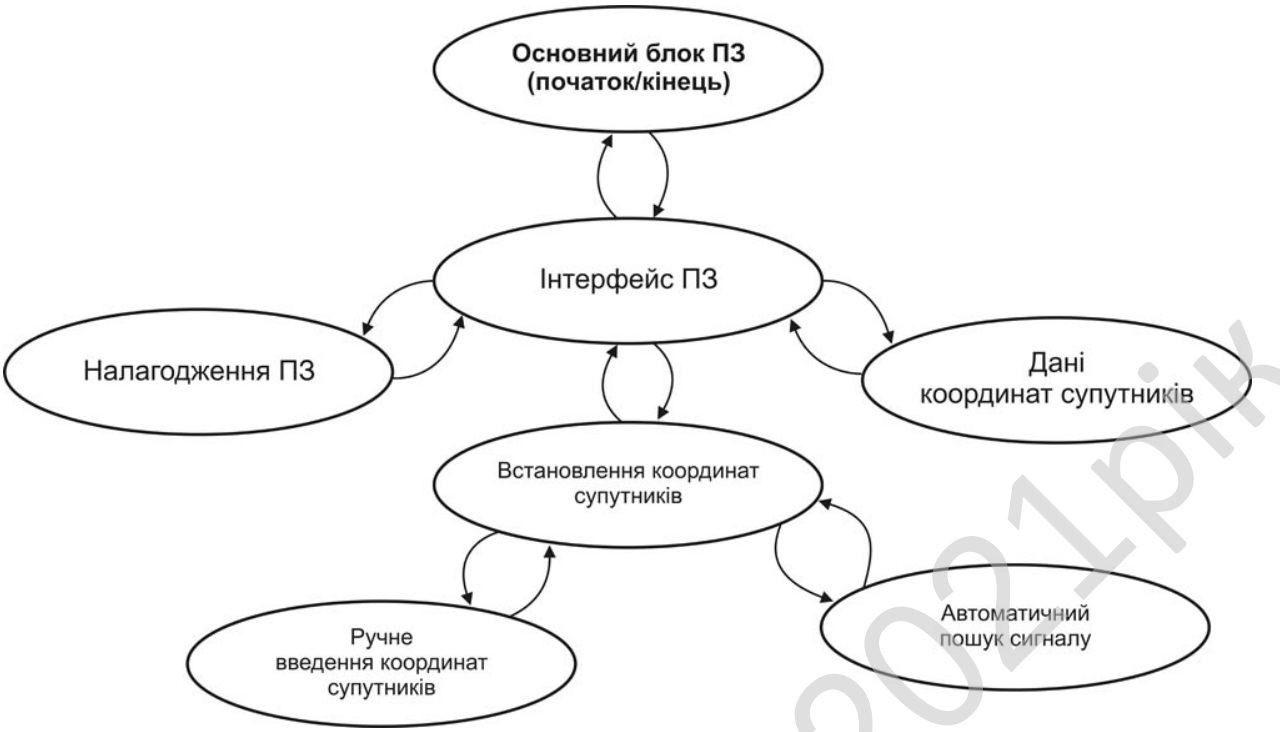

Рисунок 3.10 – Діаграма процесів

Таким чином, розглянувши опис системи, структурну, функціональну схеми системи, та діаграму взаємодії процесів перейдемо до опису блок-схем основної програми, та підпрограм, які використовуються, для реалізації системи.

4 РЕАЛІЗАЦІЯ РОБОТИ. РОЗРАХУНКИ І ЕКСПЕРИМЕНТАЛЬНІ ДАНІ, ЩО ПІДТВЕРДЖУЮТЬ ВІРНІСТЬ ПРОЕКТНИХ ТА ПРОГРАМНИХ РІШЕНЬ

4.1 Блок-схеми та опис алгоритмів функціонування системи

На рисунках 4.1, 4.2, 4.3 зображена розроблена блок-схема програми. Первинною стадією без якої не відбувається розробка програмного забезпечення це звичайно є розробка блок-схем.

Розглянемо основні блоки на блок-схемі:

Початок роботи та перевірки

– Початкова ініціалізація змінних.

– Підключення бібліотек SkyStar 3.

– Підключення бібліотеки позиціонування по протоколу DiSEqC 1.2.

– Ініціалізація доступу до RS-232.

– Перевірка наявності драйверів та плати.

– Перевірка підтримки протоколу DiSEqC 1.2.

– Встановлення початкових значень пакету DiSEqC 1.2.

– Очікування дій користувача.

Робота програми

– Запит отримання та обробки даних.

– Підпрограма пошуку та збереження позиції супутника.

– Запит отримання та обробки даних.

– Підпрограма встановлення позиції дзеркала.

Завершення роботи

– Виведення вікна повідомлення про завершення роботи ПЗ.

– Відключення додаткових модулів.

– Звільнення ресурсів програми.

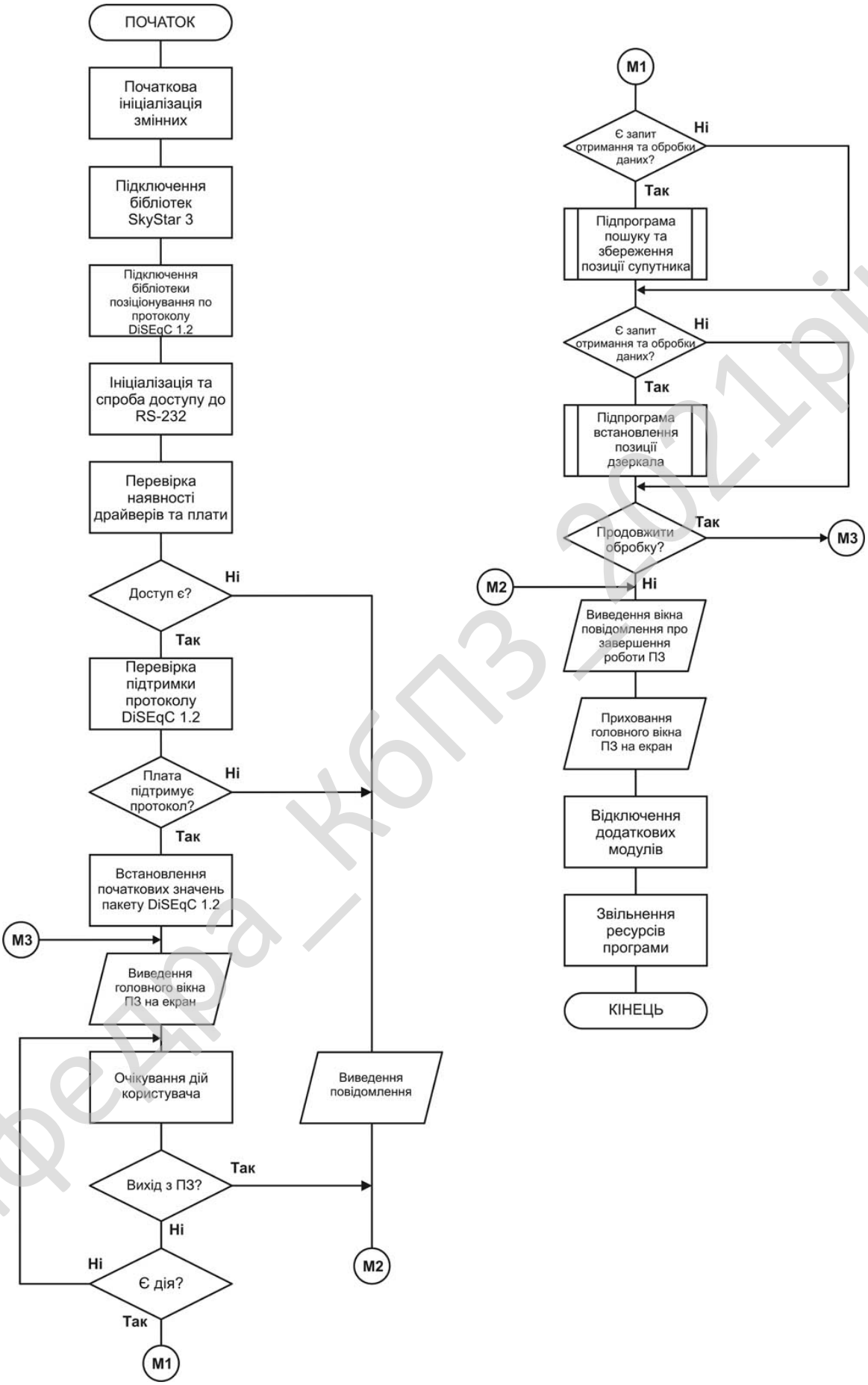

Рисунок 4.1 – Блок схема основної частини ПЗ

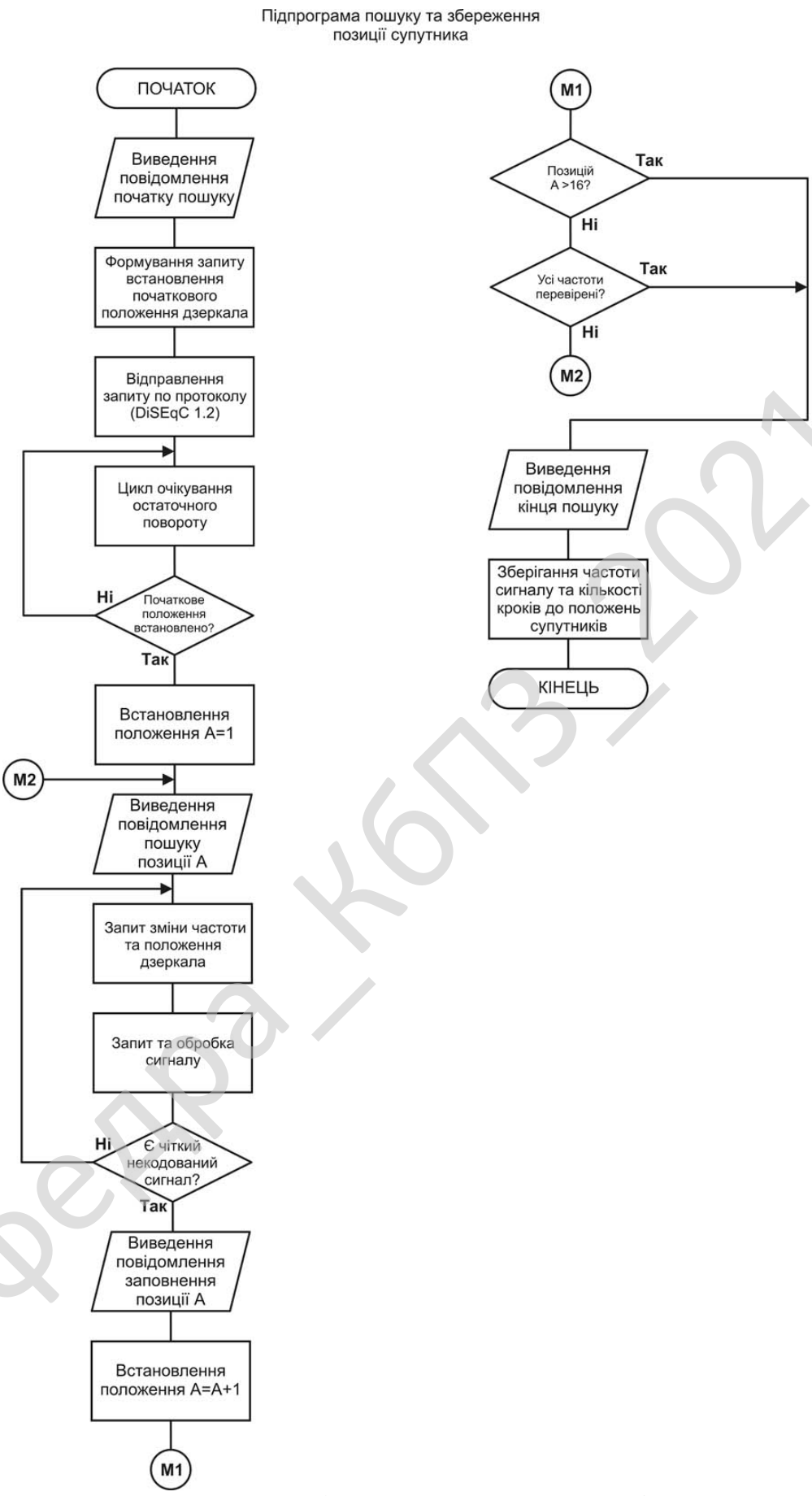

Рисунок 4.2 – Блок-схема підпрограми пошуку та збереження позиції супутника

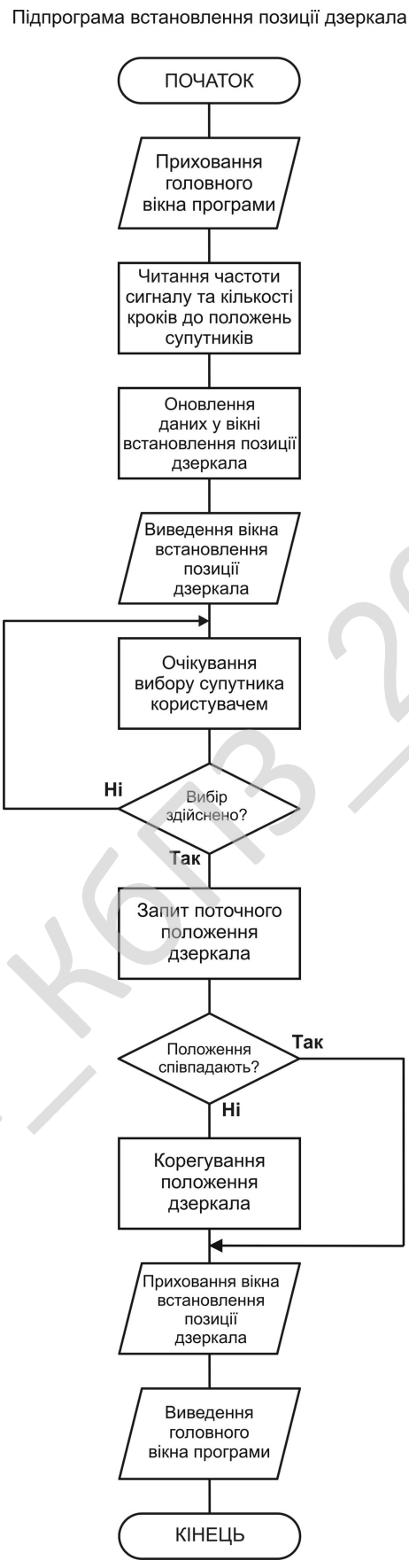

Рисунок 4.3 – Блок-схема підпрограми позиції дзеркала

4.2 Захист розробленого програмного забезпечення

Для захисту розробленого програмного забезпечення запропоновано використовувати алгоритм SHACAL-1, який заснований на перетвореннях алгоритму SHA-1. SHACAL-1 шифрує 160-бітний блок даних з використанням 512-бітного ключа шифрування. Допускається використання більше коротких ключів шифрування (не коротше 128 біт), які перед виконанням розширення ключа (процедура розширення ключа також успадкована від SHA-1 і буде описана нижче) повинні бути доповнені нульовими бітами для досягнення 512-бітного розміру.

В алгоритмі SHACAL-1 передбачено 80 раундів шифрування. Шифруєме повідомлення представляється у вигляді п'яти 32-бітних субблоків A, B, C, D і E, над якими в кожному раунді виконуються наступні дії:

$$A_{i+1} = K_i + (A_i <<< 5) + f_i(B_i, C_i, D_i) + E_i + M_i,$$

$$B_{i+1} = A_i,$$

$$C_{i+1} = B_i <<< 30,$$

$$D_{i+1} = C_i,$$

$$E_{i+1} = D_i,$$

де i – номер раунду (i = 0...79),

K_i – фрагмент розширеного ключа для і-го раунду,

f_i – функція для і-го раунду (див. нижче),

<<< – операція побітового циклічного зрушення вліво,

M_i – константи, що модифікують, певні в такий спосіб:

Раунди	Значення константи
0...19	5A827999
20...39	6ED9EBA1
40...59	8F1BBCDC
60...79	CA62C1D6

Використовувані в раундах функції f_i визначені так:

Раунди	Функція
0...19	$f(x,y,z)=(x\&y)\|(x'\&z)$
20...39,60…79	$f(x,y,z)=x \oplus y \oplus z$
40...59	$f(x,y,z)=(x \oplus y)\|(x \oplus z)\|(y \oplus z)$

У таблиці символами &, | і ⊕ позначені, відповідно, побітові логічні операції «і», «або» й «або, що виключає» (XOR); x' позначає побітовий комплемент до x.

Шифртекстом є конкатенація вмісту змінних A_{80}, B_{80}, C_{80}, D_{80} і E_{80}.

Процедура розширення ключа в алгоритмі SHACAL-1 також досить проста, вона виконується у два етапи:

Етап 1. 512-бітний вихідний ключ шифрування ділиться на 16 фрагментів по 32 біта $K_0 \ldots K_{15}$...

Етап 2. Інші фрагменти розширеного ключа $K_{16} \ldots K_{79}$ обчислюються з перших 16 фрагментів у такий спосіб:

$$K_i = (K_{i-3} \oplus K_{i-8} \oplus K_{i-14} \oplus K_{i-16}) <<< 1.$$

Раунди розшифрування виконуються у зворотній послідовності; кожний з них має на увазі виконання наступних операцій:

$$A_i = B_{i+1},$$

$$B_i = C_{i+1} <<< 2,$$

$$C_i = D_{i+1},$$

$$D_i = E_{i+1},$$

$$E_i = K'_i + (B_{i+1} <<< 5)' + f_i'(C_{i+1} <<< 2, D_{i+1}, E_{i+1}) + A_{i+1} + M'_i + 4.$$

Тут запис $f'(x)$ позначає побітовий комплемент результату виконання операції $f(x)$.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Перш ніж перейти до роботи з супутниковою антеною, необхідно встановити потрібні драйвера. Це відбувається наступним чином. При першому включенні комп'ютера із установленої DVB-Картою **TT-PCline Budget 1401** система визначає її як **Мультимедіа контролер** (рисунок 5.1).

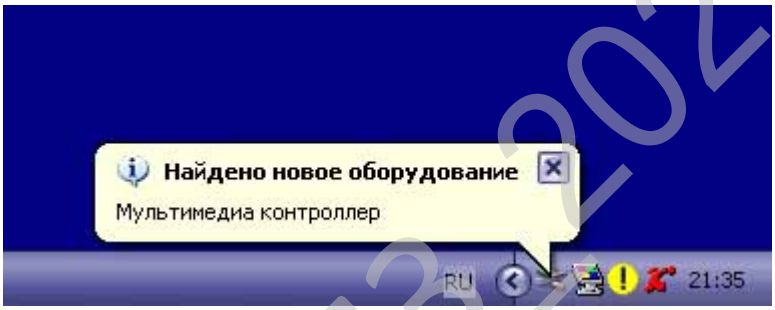

Рисунок 5.1 – Вікно пошуку нового обладнання

Після цього пропонує встановити драйвер (рисунок 5.2).

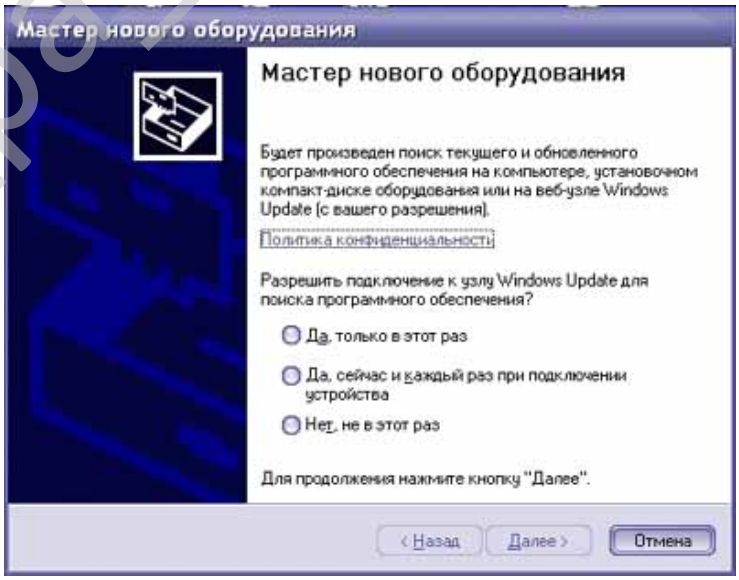

Рисунок 5.2 – Вікно майстра нового обладнання

Для цього необхідно вибрати режим "автоматичної установки" (рисунок 5.3).

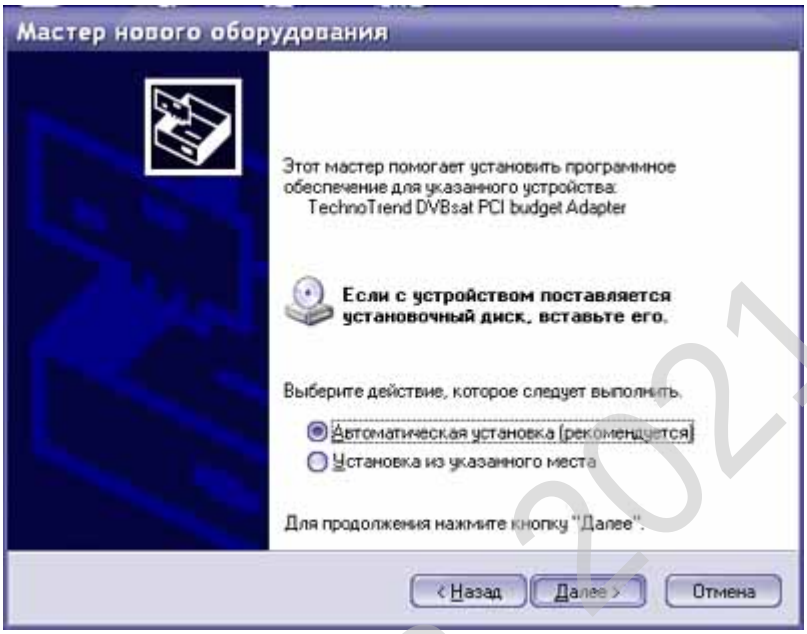

Рисунок 5.3 – Вікно вибору режиму "автоматичної установки"

Система сама знайде й перепише "куди треба" – потрібні файли (точніше файл) (рисунок 5.4)

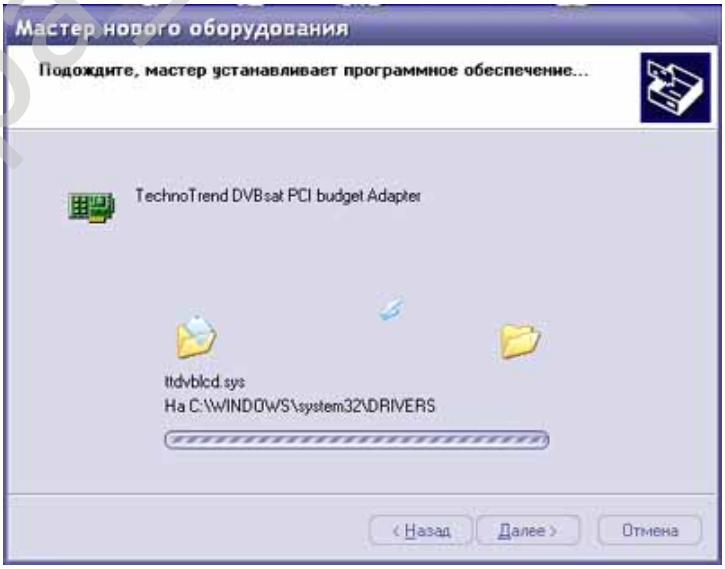

Рисунок 5.4 – Вікно переносу потрібних файлів

Після нормальної установки карта прописується як **мережна карта** за назвою **TechnoTrend DVBsat PCI budget Adapter** це можна перевірити нажавши **пуск >>> панель керування >>> система >>> устаткування >>> диспечер пристроїв** (рисунок 5.5).

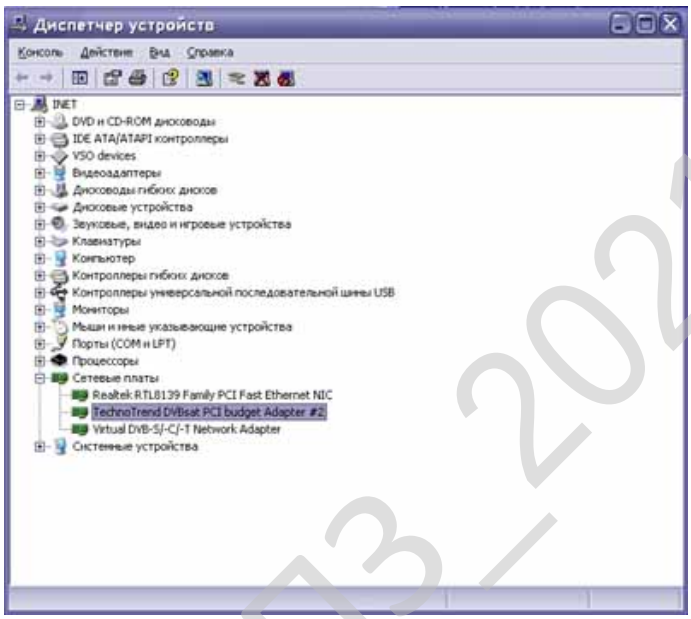

Рисунок 5.5 – Вікно диспетчеру пристроїв

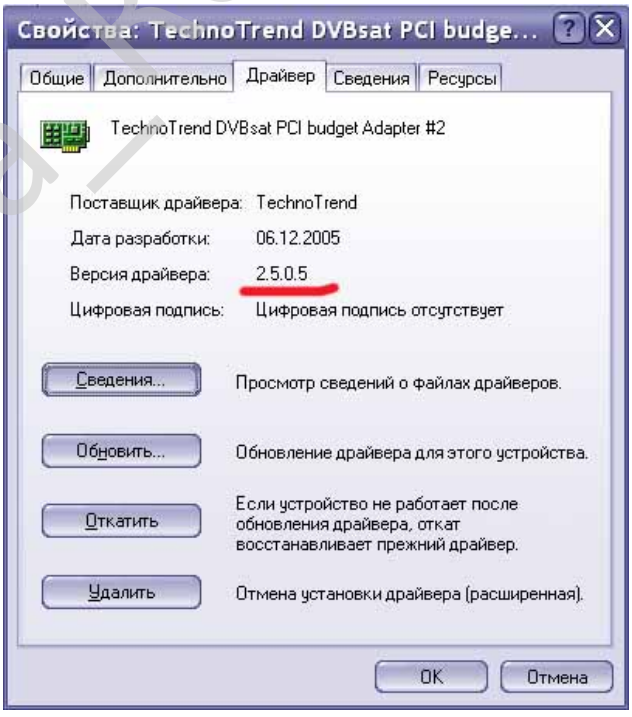

Рисунок 5.6 – Властивості обладнання

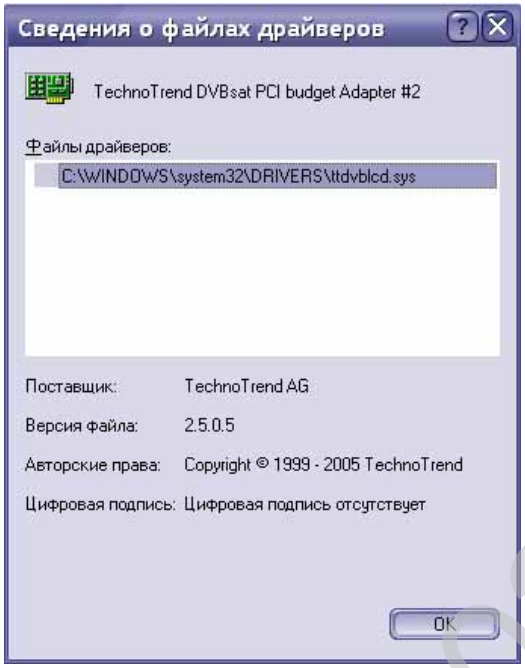

Рисунок 5.7 – Вікно даних про файли драйверу

На рисунку 5.8 зображене основне вікно розробленого програмного забезпечення.

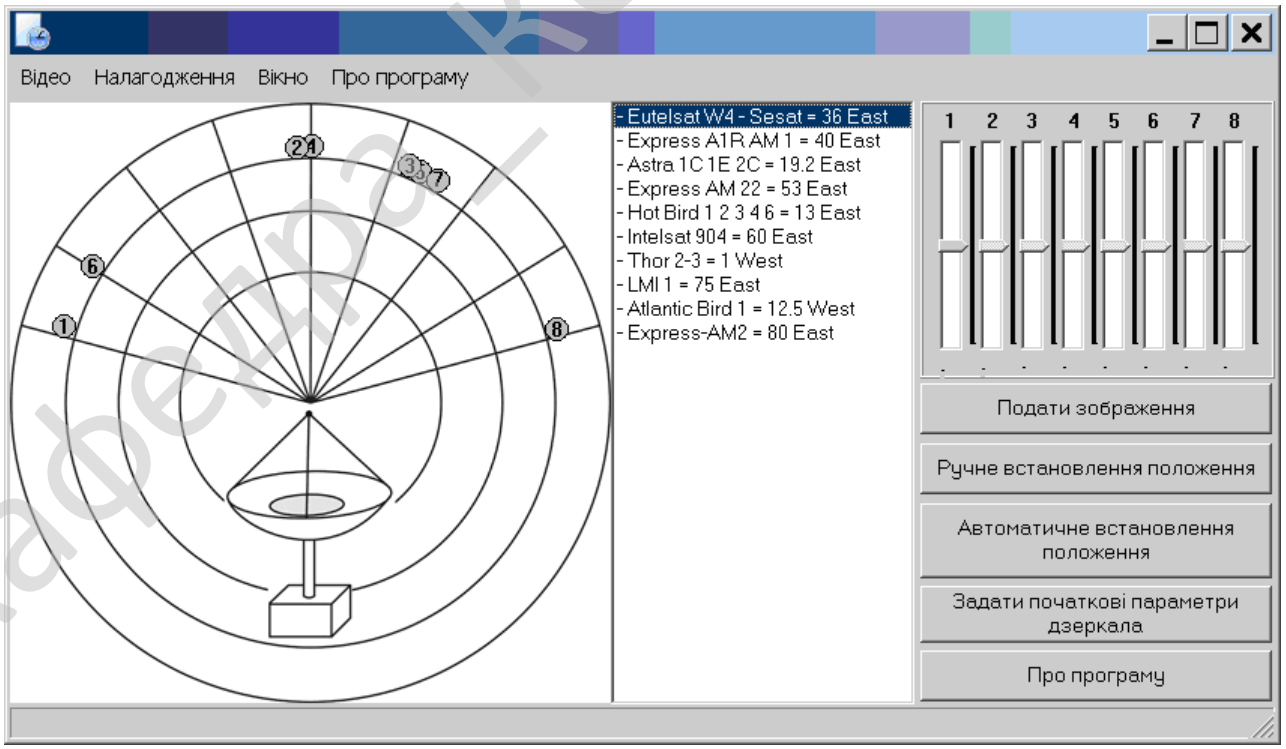

Рисунок 5.8 – Основне вікно розробленого програмного забезпечення

А сам драйвер іде НЕ під номером **2.1.9.E** а під номером **2.5.0.5** (рисунок 5.6). Якщо нажати кнопку **Відомості**, то можна побачити який саме файл (**ttdvblcd.sys**) і куди записала розумна система (рисунок 5.7). На рисунку 5.9 зображене вікно встановлення початкових значень.

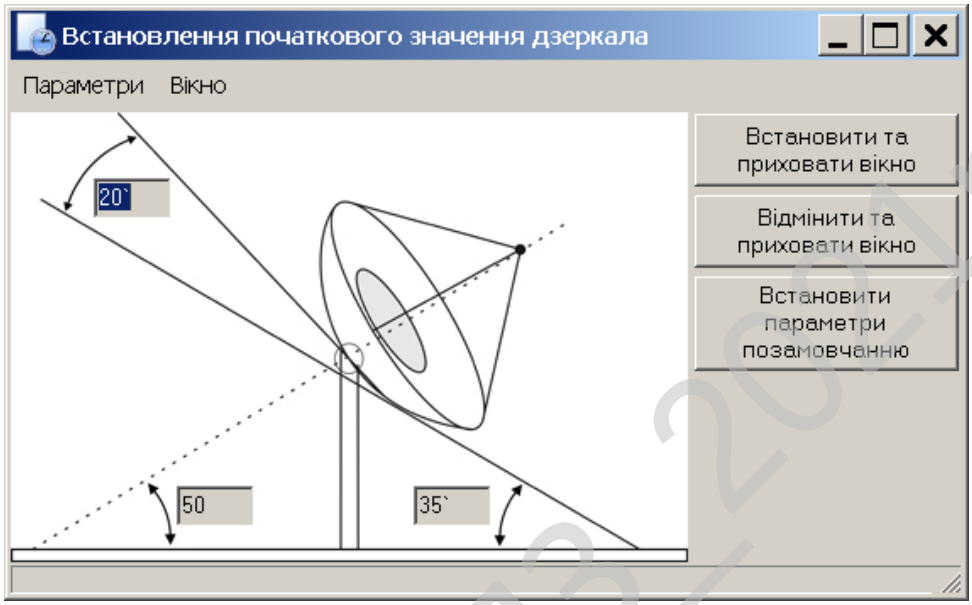

Рисунок 5.9 – Вікно встановлення початкових значень

На рисунку 5.10 наведено вікно авторського права, у якому повідомляється де, коли та під ч им керівництвом був розроблений даний програмний продукт.

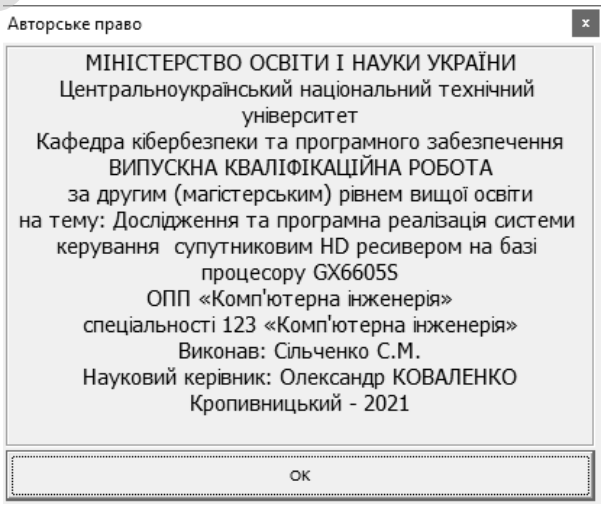

Рисунок 5.10 – Вікно авторського права

6 НАУКОВА НОВИЗНА

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти розроблено програмне забезпечення, яке призначено для системи керування супутниковим HD ресивером на базі процесору GX6605S.

Метою розробки є дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S.

Об'єктом дослідження є процес керування супутниковим HD ресивером на базі процесору GX6605S.

Предметом дослідження є методи керування супутниковим HD ресивером на базі процесору GX6605S.

Методи дослідження базуються на методах теорії інформації та кодування, методах математичної статистики, методах розробки програмного забезпечення.

Наукова новизна отриманих результатів. У процесі рішення завдань, обумовлених цілями дослідження, отримані наступні результати:

– Удосконалено метод керування супутниковим HD ресивером на базі процесору GX6605S.

– Розроблено вітчизняний продукт керування супутниковим HD ресивером на базі процесору GX6605S, який має більш широкі можливості, на відміну від існуючих аналогів.

7 ДАНІ ПРО ЕКОНОМІЧНУ ЕФЕКТИВНІСТЬ РОЗРОБЛЕНОЇ ПРОГРАМИ

7.1 Техніко-економічне обґрунтування теми випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти

Після ознайомлення з підприємством та засобами розробки програмної продукції був розроблений план розробки програми. Був підрахований необхідний час для розробки та впровадження програми. Цей час склав 60 днів (три місяці).

В магістерській роботі було проведене дослідження та виконана програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S. Розроблене програмне забезпечення має достатню надійність і задовольняє усім поставленим умовам, а саме:

а) невеликий розмір;

б) невеликі системні потреби;

в) незалежність від встановлених на комп'ютері баз даних;

г) зручність у користуванні та надійність

Таблиця 7.1 – Початкові дані

Показники	Позначення	Характеристика або величина
1	2	3
1. Кількість розроблених програм період, шт	N	1
2. Кількість екземплярів програм, шт	Ne	40 (2 ост. цифри № зал*10^1)
3. Запланований термін розробки, днів	Fpq	60 (3 місяці)
4. Група задачі підсистеми управління (1-6)	–	1
5. Ступінь новизни задачі (А, Б, В, Г)	–	Б
6. Складність алгоритму (1, 2, 3)	–	2
7. Кількість макетів вхідної інформації	–	3

Продовження табл. 7.1

1	2	3
8. Кількість форм вихідної інформації.	–	4
9. Мова програмування (1-6)	–	2
10. Попередній досвід (1-6)	–	3
11. Гнучкість проекту ПП (1-6)	–	3
12. Детальність проекту ПП (1-6)	_	2
13. Рівень спрацьованості колективу (1-6)	_	2
14. Ступінь вимірності процесів (1-6)	_	3
15. Необхідна надійність програмного забезпечення (1-6)	_	2
16. Розмір бази даних (порівняно з розміром програми) (1-6)	_	2
17. Складність кінцевого програмного продукту (1-6)	_	2
18. Необхідний рівень забезпечення повторного використання (1-6)	_	2
19. Документованість відповідно до планованого життєвого циклу (1-6)	_	2
20. Вимоги до швидкодії ПП (1-6)	_	2
21. Обмеження на розміри основного сховища даних (1-6)	_	2
22. Різноманітність використовуваних обчислювальних платформ (1-6)	_	2
23. Професійний рівень аналітиків (1-6)	_	2
24. Професійний рівень програмістів (1-6)	_	2
25. Постійність складу команди розробників (1-6)	_	2
26. Досвід розробки додатків (1-6)	_	2
27. Досвід роботи з обчислювальною платформою (1-6)	_	2

Продовження табл. 7.1

1	2	3
28. Досвід роботи з мовою і інструментами середовища розробки (1-6)	–	2
29. Досвід роботи з програмними інструментами розробки (1-6)	–	3
30. Розробка ПЗ для декількох серверів одночасно (1-6)	–	2
31. Вимоги до дотримання встановленого графіка робіт (1-6)	–	2
32. Вартість ПЗ у розробника (НМА), грн	–	40000 (2 ост. цифри № зал*10^4)
33. Норматив додаткової зарплати, % :	Нд	10
34. Норматив відрахувань у соціальні фонди, %	Нс	22
35. Норматив загальногосподарських витрат, %	Нг	15
36. Норматив витрат на освоєння нових мов програмування, %	Нп	15
37. Рівень рентабельності програмної продукції, %	Ре	55
38. Ставка податку на додану вартість, %	Ндв	20

7.2 Розрахунок трудомісткості розробки програмної продукції

Значення трудомісткості розробки програмного забезпечення для стадій ТЗ, ЕК, ТП та ВП визначаємо по типовим нормам часу приведеним в додатках МВ. Стадія РП є найбільш тривалою і трудомісткою, що робить значний вплив на інші стадії проекту.

Визначимо трудомісткість розробки ПЗ для стадії РП.

Обчислюємо номінальні трудовитрати, люд-міс.:

$$T_{ном} = A \, \text{Size}^B , \qquad (7.1)$$

де A – коефіцієнт Боема, A=2,45; Size – загальний об'єм відлагодженого програмного коду, тис. рядків; B – показник ступеня, що визначається співвідношенням

$$B = 1{,}01 + 0{,}001\sum W_i \qquad (7.2)$$

де Wi – сумарне значення п'яти показників (МВ, додаток 2), що відображають особливості розробки проекту програмного продукту (ПП) і колективу розробників.

$$B=1{,}01+0{,}001(2{,}43+3{,}64+3{,}38+3{,}95+2{,}73) = 1{,}026$$

$$T_{ном} = 2{,}45 \cdot 2{,}7^{1{,}026} = 6{,}78 \text{ люд-міс.}$$

Визначаємо уточнені (з урахуванням приведених в МВ додатку 3 сімнадцяти додаткових коефіцієнтів) трудовитрати, люд-міс.:

$$T_{уточн}=T_{ном}\, ПВ_j, \qquad (7.3)$$

де $ПВj$ – добуток сімнадцяти додаткових коефіцієнтів, приведених в МВ додатку 3.

$$T_{уточн}= 6{,}78\cdot(0{,}88\cdot0{,}93\cdot0{,}88\cdot0{,}91\cdot0{,}95\cdot1\cdot1\cdot0{,}87\cdot1{,}22\cdot1{,}16\cdot1{,}1\cdot1{,}1\cdot$$

$$\cdot1{,}12\cdot1{,}1\cdot1\cdot1{,}1\cdot1{,}1)=9{,}37 \text{ люд-міс.}$$

Ці коефіцієнти дозволяють диференційовано оцінювати результати роботи програмістів, беручи до уваги швидкодію програми, використання різноманітних обчислювальних платформ і інструментів розробки, взаємодію декількох серверів, вимоги до об'ємів баз даних і ін.

Визначаємо підсумкові трудовитрати по стадії робочий проект, люд-дні:

$$T_{РП} = 0{,}3CT_{уточн}^{0{,}33+0{,}2(B-1{,}01)}S, \qquad (7.4)$$

де C – визначений емпірично коефіцієнт, запропонований авторами методики, (МВ, додаток 4); S – коефіцієнт стиснення (або подовження) графіка робіт %, що дозволяє коректувати терміни розробки ПЗ згідно встановленим вимогам. Вибираємо в межах (25…350)%

$$T_{РП}=0{,}3\cdot2{,}66\cdot9{,}37^{0{,}33+0{,}2(1{,}026-1{,}01)}\cdot100=168 \text{ люд/день}$$

Для зручності визначення загальної трудомісткості на розробку програмного забезпечення результати розрахунків по стадіям зводимо до таблиці 7.2.

Таблиця 7.2 – Визначення трудомісткості розробки програмного забезпечення

Стадії розробки	Трудомісткість за типовими нормами та розрахунками	
	Величина, люд/дні	Підстава
Технічне завдання	9	Д5
Ескізний проект	10	Д6
Технічний проект	9	Д7
Робочий проект	168	Ф 7.1-7.4
Впровадження	13	Д13
Всього	209	–

7.3 Визначення чисельності виконавців і планового фонду зарплати

Чисельність ставок інженерів-програмістів для розробки програмного забезпечення визначається за формулою

$$Ч = \frac{T_{пз} N}{F_{рq} - H_{ев}}, \quad (7.5)$$

де Fpq – плановий фонд робочого часу одного спеціаліста, днів, Тпз – трудомісткість розробки програмного забезпечення люд-дні,

$$Ч = \frac{209 \cdot 1}{60 - 5} = 3{,}8 \text{ ставки}$$

Чисельність інженерів-електронщиків для проведення технічного обслуговування та ремонту комп'ютерних мереж визначається в залежності від наявності технічних засобів і норм витрат часу на виконання профілактичних робіт на протязі року.

Визначаємо затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за період розробки. Результати розрахунку зводимо до таблиці 7.3

Таблиця 7.3 – Затрати часу на виконання профілактичних робіт по обслуговуванню обладнання за розрахунковий період

Найменування обладнання	Профілактичне обслуговування			
	Кількість хв. на один. обл.	Кількість обладнання	Затрати часу в хв.	Затрати часу в год.
Системний блок ПК	385	12	4620	77
Монітор	160	12	1920	32
Клавіатура	140	12	1680	28
Маніпулятор «мишка»	30	12	360	6
Принтер матричний	185	1	185	3
Принтер лазерний	355	2	710	12
Принтер струминний	300	1	300	5
Сканер	155	2	310	5
Концентратор-маршрутизатор	155	2	310	5
Кабельні господарства ЛОМ на 1 м. п.	2,5	100	250	4
Кабельне господарство електромережі	48	50	2400	40
Копіювальний апарат	285	2	570	10
Усього за рік:			$З_ч$	227

Час на профілактику обладнання в загальному балансі робочого часу інженерів-електронщиків неповинен складати більше 10%

Виходячи з цього фонд робочого часу інженерів-електронщиків складає:

$$Ф_{др}^{с} = \frac{З_ч \cdot n_{міс}}{1,2} \quad (7.6)$$

$$Ф_{др}^{с} = \frac{227 \cdot 3}{1,2} = 567,5 \text{ год}$$

Визначаємо необхідну кількість ставок штатного персоналу сектора ТО:

$$Ч_{ел} = \frac{Ф_{др}^{с}}{F_{др} \cdot T_{зм}} \quad (7.7)$$

$Ч_{ел}$=567,5/(60·8)=1,2 ставки

Для забезпечення нормального технічного обслуговування засобів ТО та мереж, необхідно прийняти найбільше ціле значення розрахункової чисельності інженерів – електронщиків.

Чисельність інженерів-системотехніків, адміністраторів мережі, дизайнерів WEB вузлів, системних програмістів (аналітиків), бухгалтерів-економістів визначається за потребою в залежності від функціональних обов'язків. Після визначення чисельності персоналу складається штатний розклад.

Таблиця 7.4 – Розрахунок чисельності штатного персоналу сектору системного та адміністративного обслуговування засобів ОТ та комп'ютерних мереж

Посада	Вид роботи	Час	Кількість штатних одиниць
Адміністратор загальної мережі, аналітик	Адміністрування локальної мережі, поштового та серверу DNS (ОС FreeBSD), маршрутизатора Cisco, серверу доступу АДСЛ (ОС Linux), Wi-Fi налаштування ADSL, VPN, PPPoE, Frame Relay	2	0,5
	Налаштування і конфігурування базової станції безпровідного зв'язку (CMTS)	0,5	
	Розробка та впровадження проектів з організації зв'язку між віддаленими об'єктами, ЛОМ	0,5	
	Забезпечення цілодобової роботи зв'язку клієнтів д мережі Інтернет	1	
Всього		4	

Продовження таблиці 7.4

Посада	Вид роботи	Час	Кількість штатних одиниць
Продакт-менеджер	Презентації нової продукції, пошук каналів збуту	1	0,25
	Підтримка постійних клієнтів	0,5	
	Оформлення договорів, ведення тендерів	0,25	
	Контроль взаєморозрахунків з постачальниками	0,25	
Всього		2	
Дизайнер WEB	Розробка концепції оформлення та інтерфейсу сайту, оптимізація дизайну існуючих, проектує їх структуру та навігацію	1	0,25
	Створення графічних і стилістичних елементів сайту	0,5	
	Оформлення банерів і промо-сторінок	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	
Інженер верстальник	Розробка та верстка макетів рекламної продукції та технічної документації	1	0,25
	Верстка друкованих видань	0,5	
	Додрукова підготовка макетів	0,25	
	Розміщення графіки і контенту на Інтернет сторінках	0,25	
Всього		2	

Складемо штатний розклад виконавців:

Таблиця 7.5 – Штатний розклад виконавців

Посада	Кількість ставок	Середньо-місячний оклад, грн.	Всього за період роз-робки, грн.
Керівник (ІТ-менеджер)	1	12000	36000
Продакт-менеджер	0,25	8000	6000
Інженер-програміст	3,8	8000	91200
Інженер-електронщик	1,2	6000	21600
Інженер-системотехнік	0,25	6000	4500
Адміністратор мережі	0,5	6000	9000
Системний програміст	0,25	6000	4500
Дизайнер WEB	0,25	8000	6000
Інженер-верстальник	0,25	6000	4500
Бухгалтер-економіст	0,5	10000	15000
Всього за період розробки	$R_{сп}$=8,25	-	$Ф_{роб}$=198300

Розрахуємо середньоденну зарплату одного виконавця:

$$З_{сд} = \frac{Ф_{роб}}{R_{сп} F_{рд}}, \quad (7.8)$$

де Фроб – загальна сума зарплати за плановий період, грн.

$$З_{сд} = \frac{198300}{8,25 \cdot 60} = 400 \; грн.$$

7.4 Розрахунок капітальних вкладень та амортизаційних відрахувань у розробника

Балансова вартість будівель визначається з урахуванням кількості робочих місць виконавців, питомої площі на одне робоче місце, та вартості одного квадратного метра виробничої площі

$$Б_{уд} = R^1_{сп} S_у Ц_{пл}, \quad (7.9)$$

де $R^1_{сп}$ – кількість робочих місць виконавців, шт. Приймаємо 8 робочих місць. Sу – питома площа на одне робоче місце, $м^2$; Цпл – вартість одного квадратного метра площі, грн.

Згідно даних ТОВ науково-дослідницького консалтингового підприємства «Пекторaль» ціна одного квадратного метра площі новобудови, вік якої не перевищує 25 років, по місту складає 800…1600 $у.о./м^2$. Враховуючи, що курс складає 1 у.о. = 25 грн. приймаємо для розрахунку вартість одного метра квадратного рівною 20000 грн./м^2. На кожне робоче місце у середньому потрібно 8 $м^2$. З урахуванням цього:

$$Б_{уд} = 8 \cdot 8 \cdot 20000 = 1280000 \text{ грн.}$$

Вартість передавальних пристроїв складає 10% від вартості будівель, і у даному випадку вона складе: 128000 грн.

Балансова вартість інвентарю розраховується за нормою 3500 грн на одне робоче місце. Тобто

$$І_{нв} = R^1_{сп} \cdot Ц_м, \quad (7.10)$$

де Цм – ціна меблів для одного робочого місця, грн.

$$І_{нв} = 8 \cdot 3500 = 28000 \text{ грн}$$

Балансова вартість обчислювальної техніки визначається по оптовим цінам постачальника з врахуванням витрат на транспортування.

Специфікація на обчислювальну техніку наведена в таблиці 7.7. Дані по оптовій ціні на обладнання та комплектуючі вибирались по прайсу Інтернет магазину Компбест за 01.11.21 – джерело https://compbest.com.ua.

Таблиця 7.6 – Специфікація

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Персональний комп’ютер		10947
Системний блок	CASECOM Technology Tower NEW	7347
Процесор	Intel Xeon E5-2640 (6 (12) ядер по 2.5 – 3.0 GHz); Cache Memory 15 MB. (аналог Intel Core i7-3770)	
Системна плата	Atermiter X79, Socket 2011, Intel B75, MicroATX, DDR3	
Відеокарта	nVidia GeForce GT 710, 2 GB DDR3, 64 -b D-Sub+HDMI+DVI	
Жорсткий диск	240 GB SSD GOODRAM CX400	
Оперативна пам’ять	AVANT4GB 2Rx4 PC3-8500R DDR3-1066MHz ECC Registered HS (AVF7251R62F7066G4NYABP-IS) – 2 модулі	
DVD-привод	DVD -RW/+RW , LG SATA SuperMulti Bulk 22x, SecurDisc, black	
Корпус	CASECOM 500W (120mm big fan), 2xIDE, full-ATX, БЖ 2xSATA, 1xFDD, Air Duct, 2xUSB 2.0, Mic+Audio, silver/black	
Кардрідер внутрішній	USB 2.0 Card reader STORM CR -35U1A4-B, int. 3.5", 1*USB2.0+AUDIO+1394, multi: All Type Cards, black	
інше	Клавіатура, мишка	-

Продовження таблиці 7.6

Найменування комплектуючої або обладнання	Тип	Оптова ціна
Монітор	LG W2363V-WF Wide LCD 2ms, 70 000:1, 300кд/м2, 170/160, D-Sub / Glossy White	3600
Принтер лазерний	Canon i-SENSYS LBP6030W	2700
Принтер струменевий	Epson Stylus Photo P50 (C11CA45341) + USI cable	5500
Копіювальний апарат	Canon i-SENSYS MF217W with Wi-Fi	5965

Витрати на транспорт, монтаж та випробування можуть бути прийняті в межах до 10% від оптової ціни.

Для визначення необхідної кількості капітальних вкладень складемо таблицю 7.8.

Таблиця 7.7 – Балансова вартість обчислювальної техніки

Найменування обчислювальної техніки	Кількість, шт.	Ціна за одиницю, грн.	Витрати на транспортування, монтаж та випробовування.	Загальна вартість, грн.
Персональні комп’ютери	8	10947	8757,6	96333,6
Принтер лаз.	2	2700	540	5940
Принтер струм.	1	5500	550	6050
Копіюв. апарат	1	5965	596,5	6561,5
Всього	–	–	–	114885,1

Таблиця 7.8 – Вартість основних фондів та амортизаційні відрахування розробника

Групи та види основних фондів	Балансова вартість, грн.	Амортизація	
		Норма, %	Відрахування, грн.
1	2	3	4
Група 3			
1. Будівлі	1280000	-	-
2. Передавальні пристрої	128000	-	-
Всього по групі	1408000	5	70400
Група 4			
3. Обчислювальна техніка	114885	-	-
Всього по групі	114885	50	57442,5
Група 5			
4. Вимірювальні пристрої	5190	-	-
5. Господарський інвентар	28000	-	-
Всього по групі	33190	25	8297,5
Нематеріальні активи			
6. Нематеріальні активи	40000	10	4000
Разом	$К_р$= 1596075		$А_р$= 140140

7.5 Визначення собівартості розробки та ціни програмної продукції

Визначимо основну зарплату виконавців

$$З_о = \frac{З_{сд} \cdot T_{пз}}{N_e}, \qquad (7.11)$$

де Ne – Кількість екземплярів програм, шт.

$$З_о = 400 \cdot 209/40 = 2090 \text{ грн}$$

Визначимо додаткову зарплату (оплата відпусток, виконання державних та суспільних обов'язків) на рівні 10%

$$З_д = З_о \cdot Н_q \cdot 0{,}01, \quad (7.12)$$

де $Н_q$ – норматив додаткової зарплати, %

$$З_д = 2090 \cdot 10 \cdot 0{,}01 = 209 \text{ грн}$$

Відрахування на соціальні потреби за нормативом Нс=37% від суми основної та додаткової зарплати

$$С_{оц} = 0{,}01 \cdot Н_с (З_о + З_д), \quad (7.13)$$

де $Н_с$ – відрахування на соціальні потреби, %

$$С_{оц} = 0{,}01 \cdot 22(2090+209) = 506 \text{ грн}$$

Визначимо загальногосподарські витрати (електроенергію, ремонт і утримання приміщень і т.д) за нормативом Нг=15% від основної зарплати

$$Г_{осп} = З_о \cdot Н_г \cdot 0{,}01, \quad (7.14)$$

де $Н_г$ – загальногосподарські витрати, %

$$Г_{осп} = 2090 \cdot 15 \cdot 0{,}01 = 314 \text{ грн}$$

Визначимо витрати на матеріали для розробки програмної продукції за нормами споживання та діючими цінами за одиницю виміру:

$$З_М = (З_{М1} + З_{М2} + З_{М3})/N_е, \quad (7.15)$$

де $З_{М1}$ – вартість паперу, грн., $З_{М2}$ – вартість запам'ятовуючих пристроїв, грн., $З_{М3}$ – вартість фарби, картріджей, тонеру, грн., $N_е$ – кількість екземплярів програм, шт.

Згідно виданих викладачем норм $n_{міс}$ приймаємо 0,33 пачки паперу на місяць розробки. Тоді, враховуючи, що вартість пачки паперу складає $Ц_п$=105 грн., визначаємо вартість паперу за період розробки $N_м$ =3 міс:

$$З_{М1} = Ц_п \cdot N_м \, n_{міс}. \quad (7.16)$$

$$З_{М1} = 105 \cdot 3 \cdot 0{,}33 = 105 \text{ грн}.$$

Згідно виданих викладачем норм до вартості запам'ятовуючих пристроїв входить вартість CD дисків в кількості, що дорівнює кількості екземплярів програм та одного DVD диска для збереження резервної копії програми:

$$З_{М2} = \sum Ц_{д}., \quad (7.17)$$

де Цд – вартість дисків CD/DVD: CDR TDK 700Mb, 80Min, 52x Cake box – 2 грн/шт., DVD-R LG 4,7Gb, 16x speed Cake box – 2 грн/шт.

$$З_{М2} = 41 \cdot 12 = 492 \text{ грн.}$$

Згідно виданих викладачем норм одноразовій заправці підлягають усі друкуючі пристрої і становить:

$$З_{М3} = \sum Ц_{з}., \quad (7.18)$$

де: $Ц_з$ – вартість розхідних матеріалів друкуючих пристроїв: відновлення та заправка картриджу для Canon i-SENSYS LBP6030W – 574 грн.; картридж для Epson Stylus Photo P50 – 558 грн.; відновлення картриджу для MF217W – 570 грн.

$$З_{М3} = 574+558+570=1702 \text{ грн.}$$

$$З_{М} = (105 + 492 + 1702)/40=57 \text{ грн.}$$

Визначимо витрати на освоєння нових мов програмування або операційних систем за нормативом (Нп = 15%) від основної зарплати виконавців

$$О_{п} = З_{о} \cdot Н_{п} \cdot 0{,}01, \quad (7.19)$$

де Нп – норматив витрат на освоєння нових мов програмування, %

$$О_{п} = 2090 \cdot 15 \cdot 0{,}01=314 \text{ грн}$$

Визначимо витрати на амортизацію основних фондів з урахуванням загальної річної суми амортизаційних відрахувань та кількості екземплярів програм ($N_е$ = 40 прим.)

$$А_{м} = \frac{А_{р} \cdot N_{міс}}{N_{е} \cdot 12}, \quad (7.20)$$

де Ар – загальна річна сума амортизаційних відрахувань, грн.

$$А_{м} = 140140 \cdot 3/(40 \cdot 12)=876 \text{ грн}$$

Повна собівартість ПЗ визначається як сума витрат за попередніми статтями калькуляції

$$C_п = З_о + З_д + С_{оц} + Г_{осп} + З_м + О_п + А_м. \quad (7.21)$$

$$C_п = 2090+209+506+314+57+314+876 = 4366 \text{ грн.}$$

Визначимо плановий прибуток за рівнем рентабельності (Рп) програмної продукції, яка залежить від складності програми та ступеня новизни задачі.

Для даного програмного забезпечення рівень рентабельності складає 55%

$$П_р = 0{,}01 \cdot Р_п \cdot С_п, \quad (7.22)$$

де Рс – рівень рентабельності, %

$$П_р = 0{,}01 \cdot 55 \cdot 4366 = 2401 \text{ грн.}$$

Величини ціна підприємства, податок на додану вартість, відпускна ціна програмної продукції визначаються за формулами, приведеними в таблиці 7.9

Таблиця 7.9 – Нормативна калькуляція собівартості розробки програмного забезпечення задачі

Найменування статей витрат	Позначення	Величина, грн.
1. Основна зарплата виконавців	$З_о$	2090
2. Додаткова зарплата виконавців	$З_д$	209
3. Відрахування на соціальні потреби	$С_{оц}$	506
4. Загальногосподарські витрати	$Г_{осп}$	314
5. Витрати на матеріали	$З_М$	57
6. Освоєння нових операційних систем, мов програмування	$О_п$	314
7. Амортизація основних фондів	$А_м$	876
8. Повна собівартість програмного забезпечення	$С_п$	4366
9. Плановий прибуток	$П_р$	2401
10.Ціна підприємства $Ц_п=С_п+П_р$	$Ц_п$	6767
11.Податок на додану вартість $ПДВ = 0.01 \cdot Н_{дв} \cdot Ц_п$	$ПДВ$	1353,4
12.Відпускна ціна програмної продукції Ц = Цп + ПДВ	$Ц$	9168

7.6 Визначення об’єму капітальних вкладень у споживача програмної продукції

Об’єм капітальних вкладень у споживача програмної продукції визначаємо на основі балансової вартості основних фондів, яка враховує ціну, транспортно-заготівельні витрати, вартість будівель, монтажних та пусконалагоджувальних робіт, а також витрати на випробування у виробничих умовах. Результати розрахунків зводимо у таблицю 7.109.

Таблиця 7.10 – Розрахунок об’єму капітальних вкладень у споживача програмної продукції

Найменування капітальних вкладень	Сума за варіантами, грн	
	Базовий	Новий
Вартість програмної продукції	–	9168
Всього капітальних витрат	–	9168

7.7 Визначення експлуатаційних витрат

Експлуатаційні витрати у споживача програмної продукції визначаємо при умові роботи підсистеми на протязі року. Результати зводимо до таблиці 7.11.

Таблиця 7.11 – Розрахунок експлуатаційних витрат у споживача програмної продукції

Найменування статей витрат	Позначення	Сума витрат за варіантами, грн.	
		Базовий	Новий
1. Витрати на технічне обслуговування	$З_р$	107360	20130
2. Витрати на електроенергію	$З_{ел}$	3732	1865
3. Витрати на амортизацію	$З_{ам}$	0	2292
Всього витрат за рік	$І$	111092	24287

Витрати на оплату праці:

$$З_р = Т_р \cdot З_г \cdot (1 + 0{,}01 \cdot Н_q) \cdot (1 + 0{,}01 \cdot Н_с), \qquad (7.23)$$

де Тр – кількість годин обслуговування системи за рік, *год.*; Зг – заробітна плата обслуговуючого персоналу, *грн / год*.

Після купівлі нового програмного забезпечення кількість профілактичних годин робіт зменшилася з 800 годин на рік до 150 годин на рік, тому витрати на технічне обслуговування зменшилися з

$$З_{р\,баз} = 800 \cdot 100 \cdot 1{,}1 \cdot 1{,}22 = 107360\ грн.$$

до

$$З_{р\,нов} = 150 \cdot 100 \cdot 1{,}1 \cdot 1{,}22 = 20130\ грн.$$

Витрати на електроенергію визначаються з урахуванням споживаємої потужності (Пел) в кіловатах, часу експлуатації технічних засобів (Тр) в годинах та ціни однієї кіловат-години (Цел).

$$З_{ел} = П_{ел} \cdot Т_р \cdot Ц_{ел}. \qquad (7.24)$$

Зел баз=·0,475·2455·3,2 = 3732 грн

Зел нов=·0,475·1227·3,2 = 1865 грн

Витрати по амортизації визначаються на основі норм амортизаційних відрахувань, вартості програмної продукції і основних фондів. Для розрахунку складаємо таблицю 7.12.

Таблиця 7.12 – Розрахунок амортизаційних відрахувань

Групи основних фондів	Норма амортизації %	Балансова вартість, грн., за варіантами		Сума відрахувань, грн за варіантами	
		Базовий	Новий	Базовий	Новий
Програмна продукція	25	–	9168	–	2292
Всього відрахувань	-	–	9168	–	2292

7.8 Визначення економічної ефективності програмної продукції

Економічна ефективність програмного забезпечення визначається для виготовлювача і споживача за такими показниками.

Величина економічного ефекту при виготовленні програмної продукції, розраховуємо за формулою

$$E_в = (Ц_п - C_п) \cdot N_е - \sum_{i=1}^{m} E_{p_m} \cdot K_{p_m}, \quad (7.25)$$

де: $K_р$ – балансова вартість основних фондів розробника, грн.; $E_р$ – розрахунковий коефіцієнт капіталовкладень.

$$Ев = (6767 - 4366) \cdot 40 - (0,05 \cdot 1408000 + 0,5 \cdot 114885 +$$
$$+0,25 \cdot 33190 + 0,1 \cdot 40000) \cdot 3/12 = 61005 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у виробника програмної продукції:

$$T_в = \frac{K_р^*}{(Ц_п - C_п) \cdot N_е}, \quad (7.26)$$

де: $K_р^*$ – балансова вартість основних фондів розробника без врахування вартості ОФ третьої групи, так як їх строк служби на порядок більший ніж період розробки ПЗ.

$$T_в = \frac{188075}{(6767 - 4366) \cdot 40 \cdot 12 / 3} = 0,5 \text{ років}.$$

Визначимо величину економічного ефекту у користувача програмної продукції за формулою:

$$E_{сп} = (I_б - I_н) - E_н(K_н - K_б), \quad (7.27)$$

де $I_б$, $I_н$ – величина експлуатаційних витрат за базовим и новим варіантом відповідно, $K_б$, $K_н$ – об'єм капітальних вкладень за варіантами, що порівнюються

$$E_{сп} = (111092 - 24287) - 0,25 \cdot 9168 = 84513 \text{ грн.}$$

Визначимо період окупності додаткових капітальних вкладень у споживача програмної продукції за рахунок зниження експлуатаційних витрат

$$T_{сп} = \frac{K_н - K_б}{I_б - I_н} \quad (7.28)$$

$$T_{сп} = \frac{9168}{111092 - 24287} = 0{,}1 \; року$$

Показники економічної ефективності програмної продукції зводимо до таблиці 7.13.

Таблиця 7.13 – Показники економічної ефективності програмної продукції

Найменування показників	Одиниця виміру	Величина
1. Кількість екземплярів програми	Прим.	40
2. Повна собівартість розробленої програми	Грн	4366
3. Ціна розробленої програми	Грн.	6767
4. Плановий прибуток від реалізації розробленої програми	Грн.	2401
5. Рентабельність програмної продукції	%	55
6. Об'єм додаткових капітальних вкладень у виробника програмної продукції	Грн.	1596075
7. Загальний прибуток від реалізації програмної продукції	Грн.	96040
8. Величина економічного ефекту при виготовлені програмної продукції	Грн.	61005
9. Період окупності додаткових капітальних вкладень у виробника програмної продукції	Років.	0,5
10.Об'єм додаткових капітальних вкладень у споживача програмної продукції	Грн.	9168
11.Величина економічного ефекту у користувача програмної продукції	Грн.	84513
12.Період окупності додаткових капітальних вкладень у користувача програмної продукції	Років	0,1

7.9 Висновки

Розроблена програма економічно вигідна. За рахунок впровадження програмного забезпечення досягається скорочення часу обробки інформації, підвищується культура праці, підвищення якості приймаючих управлінських рішень.

8 ЗАХОДИ З ОХОРОНИ ПРАЦІ ТА ТЕХНІКИ БЕЗПЕКИ

8.1 Вступ

Людство протягом усієї історії приділяє особливу увагу безпеці життя. Охорона праці є складовою частиною безпеки життя [1].

Закон України “Про охорону праці” [2] регламентує загальні положення державної політики в галузі охорони праці, а конкретизуються ці положення нормативно-правовими актами про охорону праці, зокрема Наказом Міністерства соціальної політики України 14.02.2018 № 207, який зареєстровано в Міністерстві юстиції України 25 квітня 2018 р. за №508/31960 «Про затвердження Вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями», НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин», та ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин».

Програмісти у процесі роботи мають негативний вплив на органи зору, а також мають значну розумову напругою і нервово-емоційне навантаження. Руки (суглоби пальців та м'язи рук) при роботі з клавіатурою мають теж ісе тотне навантаженням. До шкідливих факторів, які впливають на робітників галузі інформаційних технологій (ІТ) спеціалісти відносять високочастотні електромагнітні коливання (випромінювання) роботи апаратної частини ЕОМ та виділення шкідливих газів.

Ці шкідливі фактори можуть привести до професійних захворювань.

При розгляді шкідливих чинників роботи програмістів та інших спеціалістів ІТ будемо керуватись наступними нормативно-правовими актами: «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» ДСанПіН 3.3.2-007-98, та

«Правила охорони праці під час експлуатації електронно-обчислювальних машин» НПАОП 0.00-1.28-10,

Умови праці програміста включають наступні фактори:

– параметри повітряного середовища в приміщенні;

– вентиляція приміщення;

– освітлення приміщення;

– параметри повітряного середовища в приміщенні, тощо.

Щоб запропонувати заходи щодо зменшення впливу комп'ютера на організм програміста визначимо фактори, які можуть викликати професійне захворювання і впливають на працездатність програміста,

8.2 Шкідливі і небезпечні фактори при роботі з комп'ютером

Програміст працює з електронно-обчислювальною машиною (ЕОМ) та іншим обладнанням, яке є джерелом небезпеки ураження електричним струмом. Так як робота програміста характеризується істотним зоровим навантаженням, то вимагає належного освітлення. Так як програміст постійно перебуває в приміщенні, тому для комфортних умов праці в цьому приміщенні необхідно створити належний мікроклімат.

При роботі з використанням ЕОМ відзначають наступні небезпечні та шкідливі фактори:

– ризик виникнення надзвичайних ситуацій природного або штучного характеру на об'єкті або території.

– ризик виникнення пожежі;

– ризики ураження електричним струмом;

– негативний вплив на органи зору людини;

– недостатня, або надмірна освітленість робочого місця;

– монотонність праці;

– електромагнітні (у т.ч. високочастотні) випромінювання (коливання);

– несприятливі мікрокліматичні умови;

– нервово-емоційна напруженість праці;

– інтелектуальні навантаження;

– невідповідність ергономічних показників робочого місця діючим вимогам;

– шуми;

– статичні навантаження на кістково-м’язовий апарат;

8.3 Аналіз санітарно-гігієнічних умов праці на робочому місці програміста

Розглянемо умови праці у приміщенні, в якому працюють програмісти. Геометричні розміри приміщення наведено у таблиці 8.1.

Таблиця 8.1 – Розміри приміщення

Найменування	Значення, м
Ширина	8*
Довжина	12,2*
Висота	2,9

* вказано загальні розміри поєднаного приміщення, де загалом працюють 14 людей, а фактично у наявності є дві кімнати, розділених перестінком.

Таблиця 8.2 – Площа та обсяг приміщення, на одного працюючого*

Геометрична характеристика	Одиниця виміру	Нормативне значення*	Фактичне значення
Площа, S	$м^2$	не менше 6.0	6,97
Обсяг, V	$м^3$	не менше 20.0	20,2

* Згідно ДСанПіН 3.3.2.007-98 (Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин)

У зазначеному приміщенні працює 14 осіб. За даними, які наведено у табл. 8.1 та табл. 8.2, можна зробити висновок, що площа та об'єм приміщення у розрахунку на одно робоче місце програміста відповідають нормативним вимогам (Наказу Міністерства соціальної політики України № 207, від 14.02.2018 «Про затвердження Вимог щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями», ДСанПіН 3.3.2-007-98 «Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин» та НПАОП 0.00-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин»).

Температура повітря в приміщенні визначається впливом температури зовнішнього повітря і тепловою енергією, яка виділяється всередині приміщення. Джерелами виділення теплоти в даному приміщенні є електроустаткування, освітлювальні прилади, а також люди. У світлий час доби джерелом надлишкового тепла є сонячна радіація. Згідно Постанови № 42 від 01.12.1999 Головного державного санітарного лікаря України, робота, яка виконується в даному приміщенні, відноситься до категорії Іа. В цьому випадку людина витрачає енергії до 120 ккал у годину. Вологість повітря у приміщенні визначається впливом багатьох факторів, серед яких: вологість атмосферного повітря, виділення вологи людьми (при диханні та випарами з поверхні шкіри).

Мікроклімат повітряного середовища в приміщенні характеризується запиленістю та загазованістю повітря. Мікроклімат приміщення визначається діючим на організм людини поєднанням, вологості, температури, швидкості руху повітря та інтенсивності теплового випромінювання. Аналіз мікроклімату складається з визначення зазначених вище факторів і порівняння результатів із встановленими нормами.

У таблиці 8.3 наведено оптимальні та фактичні значення параметрів мікроклімату як для категорії ваги робіт Іа, та к і розглянутого приміщення. У

приміщеннях, де встановлено ЕОМ, рекомендується застосування тільки оптимальних значень показників мікроклімату.

Таблиця 8.3 – Оптимальні і фактичні значення параметрів мікроклімату

Пора року	Оптимальні для Іа			Фактичні		
	Температура, °С	Воло-гість,%	Швидкість повітря, м/с	Температура, °С	Воло-гість%	Швидкість повітря, м/с
Холодна	22-24	40-60	0,1	21-24	44-55	0,14
Тепла	23-25	50-70	0,1	23-24	40-78	0,1

Проведений аналіз показує, що показники мікроклімату в приміщенні відповідають установленим нормам. Штучне опалення застосовується у холодний період року.

В літню пору застосовується кондиціонер.

Для боротьби з пилом робляться регулярні провітрювання та вологі прибирання приміщенні.

У приміщенні знаходяться наступні джерела шуму: принтер *Canon PIXMA G2411*, електродвигуни вентиляторів ЕОМ.

Одним з найважливіших факторів, які впливають на ефективність трудової діяльності людини, та попереджають травматизм і професійні захворювання програмістів є освітлення на робочому місці.

Працю працівника, який постійно працює за комп'ютером, згідно ДБН В.2.5 – 28 – 2006 р. можна віднести до роботи з малою точністю (найменший розмір об'єкта розрізнення від 1 до 5 мм) V-го розряду зорової роботи, з великою контрастністю об'єкта розрізнення (символів на екрані дисплея), з темним тлом (під розряд зорової роботи В). Приміщення можна віднести до 1-ої групи приміщень, у яких проводиться розрізнення об'єктів зорової роботи при фіксованому напрямку лінії зору того, що працює на робочу поверхню. Для такого типу приміщень і розряду зорової роботи нормоване значення коефіцієнта

природної освітленості (КПО) робочої поверхні (при поєднаному, спільному освітленні), повинен становити не більше 1,5%, освітленість при штучному висвітленні повинна становити 300 лк. Крім того все поле зору повинне бути освітлено достатньо рівномірно – ця основна гігієнічна вимога. Так як яскраве світло на ділянці периферійного зору значно збільшує напруженість очей і, як наслідок, призводить до їх швидкої стомлюваності, ступінь освітлення приміщення і яскравість екрану комп'ютера повинні бути приблизно однаковими.

8.4 Розробка заходів з умов поліпшення охорони праці

Згідно аналізу умов праці в розглянутому приміщенні, ми одержали наступні результати:

– розмірі приміщення, у розрахунку на одному працюючого, відповідають нормативам;

– мікроклімат відповідає нормативному значенню;

– акустичні умови роботи не перевищують нормативних значень;

Таким чином можна припустити, що основною причиною можливого зниження працездатності програміста є психофізіологічний фактор, тому основна пропозиція буде така: дотримання позитивної психологічної атмосфери в колективі та регламентованого режиму праці та відпочинку, організація робочого місця з урахуванням ергономічних вимог.

Рекомендовані заходи: регулярні періодичні наочні огляди персоналом шляхів для евакуації людей із приміщення, відповідно до плану евакуації (який повинен розташовуватись на видному місці у приміщенні), включення до колективного договору мінімально можливого вмісту аптечок з обов’язковою наявністю масок-клапанів, або іншого спорядження для штучного дихання. Регулярна періодична перевірка параметрів заземлення та занулення (вимірювання опору ланцюга).

8.5 Розрахункова частина

Проведемо розрахунок штучного освітлення за методом коефіцієнта використання світлового потоку для приміщення ширина якого складає 8 м, довжина 12,2 м, висота 2,9 м.

Для того, щоб визначити потрібну кількість світильників, які повинні забезпечити нормований рівень освітленості, визначимо світловий потік, що падає на робочу поверхню за формулою:

$$F=ESKZ/n,$$

де:

F – світловий потік, що розраховується, *Лм*;

E – нормована мінімальна освітленість, *Лк*; *E* = 300 *Лк*;

S – площа освітлюваного приміщення (у нашому випадку *S*= 8 х 12,2 = 97,6 м^2);

K – коефіцієнт запасу, що враховує зменшення світлового потоку лампи в результаті забруднення світильників в процесі експлуатації (його значення залежить від типу приміщення і характеру робіт, що проводяться в ньому, в нашому випадку *K* = 1,5);

Z – відношення середньої освітленості до мінімальної (зазвичай приймається рівним 1.1... 1.2, в нашому випадку *Z* = 1,1);

n – коефіцієнт використання світлового потоку, (відношення світлового потоку, що падає на розрахункову поверхню, до сумарного потоку всіх ламп і обчислюється в долях одиниці; залежить від характеристик світильника, розмірів приміщення, забарвлення стін і стелі, що характеризуються коефіцієнтами відбиття від стін ($\rho_{стін.}$) і стелі ($\rho_{стелі}$), значення коефіцієнтів дорівнюють $\rho_{стін}$ = 50% і $\rho_{стелі}$ = 50%.

Обчислимо індекс приміщення за формулою:

$$i=S/(h(A+B)),$$

де:

S – площа приміщення, *S* = 97,6 м^2;

h – розрахункова висота підвісу, *h* = 2,9 м (співпадає з висотою стелі, т.я.

лампи освітлення закріплюються на стелі);

A – ширина приміщення, A = 8 м;

B – довжина приміщення, B = 12,2 м.

Підставимо всі значення у формулу та визначимо індекс приміщення:

$$i=1,67.$$

Знаючи індекс приміщення (i), за знаходимо n = 0,56 (з табличних даних коефіцієнтів використання світлового потоку (n) світильників [3]). Підставимо всі значення у формулу, визначимо світловий потік: F=86281 *Лм*.

Для штучного освітлення приміщення використовуються *LED* світильники *Videx 54W-5000K*, світловий потік яких $F_л$ = 5940 *Лм*.

Число світильників визначається по формулі:

$$N=F/F_л$$

де:

F – світловий потік,

$F_л$ – світловий потік однієї лампи.

Підставимо всі значення у формулу та визначемо індекса приміщення: N= 86281/ *5940*=14,5 шт.

Для забезпечення нормованої мінімальної освітленості приймаємо необхідну кількість світильників 15 шт.

8.6 Висновки до розділу

Дотримання всіх необхідних умов праці не лише сприяє збереженню здоров'я працівників, а також підвищує ефективність виробництва вцілому.

З цих міркувань було здійснено аналіз приміщення, призначеного для праці програмістів, проведено розгляд небезпечних та шкідливих факторів, що негативно впливають на програмістів під час роботи. Виконано розрахунок штучного освітлення, як одного з ключових факторів впливу на працездатність та здоров'я програміста. Розроблено заходи з охорони праці.

9 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти, призначено для системи керування супутниковим HD ресивером на базі процесору GX6605S.

В межах України в недостатній мірі представлені вітчизняні розробки в цій області.

У випускній кваліфікаційній роботі за другим (магістерським) рівнем вищої освіти наведені теоретичне узагальнення й рішення наукового завдання дослідження методів керування супутниковим HD ресивером на базі процесору GX6605S.

Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем керування супутниковим HD ресивером на базі процесору GX6605S.

– Досліджена система керування супутниковим HD ресивером на базі процесору GX6605S.

– На основі отриманих результатів досліджень створена програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S.

Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання керування супутниковим HD ресивером на базі процесору GX6605S.

Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Розроблене програмне забезпечення має простий, дружній та зручний інтерфейс користувача, що забезпечує легкість у освоєнні роботи програмного продукту, зручність у використанні, і не потребує особливих спеціальних знань.

При створені програмного забезпечення було використано об'єктно-орієнтований підхід, що відповідає сучасним тенденціям у галузі розробки комерційних програмних систем.

Програма реалізована на мові високого рівня Delphi 10.4 Sydney. Дана мова програмування дозволяє найбільш ефективно обробляти дані. Це дозволило мінімізувати строк розробки програмного забезпечення, і, як слід, зменшити витрати на його розробку. Запропоноване програмне забезпечення ділиться на загальне програмне забезпечення, що поставляється із засобами обчислювальної техніки й спеціальне програмне забезпечення, що спеціально розроблене для даної конкретної системи й включає програми, що реалізують її функції.

Програма призначена для виконання під управлінням багатозадачної операційної системи Windows XP/Vista/7/8/10.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм SHACAL-1.

В цілому створене програмне забезпечення підтверджує правильність використаних проектних рішень та повністю відповідає вимогам технічного завдання. Створене програмне забезпечення має потенційну можливість для подальшого вдосконалення і застосування у різних галузях.

Розроблена програма має реальний економічний ефект від її впровадження у виробництво у сумі 84513 грн. З урахуванням вартості розробки програми та обладнання, строк окуплення становить 0,1 роки.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Сільченко С.М. Дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S // Збірник праць молодих науковців ЦНТУ. – Вип. 12. – Кропивницький: ЦНТУ, 2022.
2. Коваленко А.С. Разработка структуры базы данных интегрированной информационной системы / А.С. Коваленко, А.В. Коваленко // Информационные технологии и защита информации в информационно-коммуникационных системах: монографія / Под редакцией профессора В.С. Пономаренко. – Х.: Вид-во ТОВ «Щедра садиба плюс», 2015. – С. 54-64.
3. Кожанова А.С. Обґрунтування необхідності створення систем технічної діагностики інтегрованих інформаційних систем / О.А. Смірнов, А.С. Кожанова, О.В. Коваленко // Системи обробки інформації. – Х.: ХУПС, 2013. – Вип. 6(113). – С. 255-257.
4. Коваленко А.С. Задачи распознавания ситуаций в ERP системах / А.В. Коваленко ., А.А. Смірнов, А.С. Коваленко // Системи обробки інформації. – Х.: ХУПС, 2014. – Вип. 4(120). – С. 161-164.
5. Коваленко А.С. Підсистема технічної діагностики для автоматизації процесів керування в інтегрованих інформаційних системах / А.С. Коваленко , О.А.Смірнов, О.В. Коваленко // Системи озброєння і військова техніка.– Х.: ХУПС, 2014. – № 1(37). – С. 126-129.
6. Коваленко А.С. Анализ эффективности использования экспертной системы технической диагностики с традиционной структурой / А.С. Коваленко, А.А. Смирнов, А.В. Коваленко // Системи озброєння і військова техніка. – Х.: ХУПС, 2014. – № 2(38). – С. 106-108.
7. Коваленко А.С. Разработка структуры экспертной системы технической диагностики интегрированной информационной системы /

А.С. Коваленко, А.А. Смирнов, А.В. Коваленко // Наука і техніка Повітряних Сил Збройних Сил України. – Харків: ХУПС, 2014. – № 2(15). – С.154-157.

8. Коваленко А.С. Разработка структуры экспертной системы технической диагностики интегрированной информационной системы / А.С. Коваленко, А.А. Смирнов, А.В. Коваленко // Наука і техніка Повітряних Сил Збройних Сил України. – Харків: ХУПС, 2014. – № 2(15). – С.154-157.

9. Коваленко А.С. Структура системи технічної діагностики інтегрованих інформаційних систем / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Збірник наукових праць Кіровоградського національного технічного університету / техніка в сільськогосподарському виробництві, галузеве машинобудування, автоматизація. – Кіровоград: Вид-во КНТУ, 2014. – Вип. 27. – С. 245-251.

10. Коваленко А.С. Дослідження будови інтегрованої інформаційної системи та її елементів / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Системи озброєння і військова техніка. – Х.: ХУПС, 2014. – № 4(40). – С. 85-88.

11. Коваленко А.С. Розробка структури бази даних для обліку технічного стану елементів інтегрованої інформаційної системи з урахуванням вимог споживачів інформації / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Системи обробки інформації. – Х.: ХУПС, 2015. – Вип. 1(126). – С. 75-79.

12. Коваленко А.С. Обгрунтування набору даних для оцінки технічного стану інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Збірник наукових праць Харківського університету Повітряних Сил. – Харків: ХУПС, 2015. – Вип. 1(42). – С.39-41.

13. Коваленко А.С. Експертна система технічного діагностування інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Системи озброєння і військова техніка. – Х.: ХУПС, 2015. – № 1(41). – С. 106-111.

14. Коваленко А.С. Удосконалення методу технічного обслуговування об’єктів інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов,

О.В. Коваленко, О.П. Доренський // Системи озброєння і військова техніка. – Х.: ХУПС, 2016. – № 2(46). – С. 109-114.

15. Коваленко А.С. Метод визначення оптимального комплексу робіт з відновлення працездатності інтегрованої системи технічної діагностики в умовах ресурсних обмежень / А.С. Коваленко // Системи обробки інформації. – Х.: ХУПС, 2016. – Вип. 3(140). – С. 85-72.

16. Kovalenko A.S. Information model and its element for displaying information on technical condition of objects of integrated information system / A.S. Kovalenko, A.A. Smirnov, A.V. Kovalenko, A.P. Dorensky // International Journal of Computational Engineering Research (IJCER). – India: Delhi, 2016. – Volume 6, Issue 1. – P. 21-27.

17. Кожанова А.С. Система технічної діагностики інтегрованих інформаційних систем – обґрунтування необхідності створення, визначення понятійного апарату та напрямів досліджень / А.С. Кожанова, О.А. Смірнов, М.П. Савченко, Д.М. Ізосімов, В.В. Мороз // Створення та модернізація озброєння і військової техніки в сучасних умовах: Тринадцята наук.-техн. конф., 5-6 вер. 2013 р., м. Феодосія: тези доп. – Феодосія: ДНВЦ, 2013. – С. 187-188.

18. Кожанова А.С. Визначення основних напрямків досліджень щодо створення системи технічної діагностики інтегрованих інформаційних систем / А.С. Кожанова, О.А. Смірнов, А.В. Челпанов // Проблемні питання розвитку озброєння та військової техніки Збройних Сил України: IV наук.-техн. конф., 16-20 груд. 2013 р., м. Київ: зб. тез. – Київ: ЦНДІ ОВТ ЗСУ, 2013. – С. 293.

19. Коваленко А.С. Обґрунтування необхідності створення систем технічної діагностики інтегрованих інформаційних систем / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Інформатика та системні науки : V Всеукр. наук.-практ. конф., 13–15 бер. 2014 р., м. Полтава : зб. тез. – Полтава: ПУЕТ, 2014. – С. 292-294.

20. Коваленко А.С. Задачи распознавания ситуаций в системах организационной стратегии интеграции производства и операций

/ А.С. Коваленко, А.В. Коваленко // Комбінаторні конфігурації та їх застосування: XVI міжнар. наук.-практ. сем., 11-12 квіт. 2014 р., м. Кіровоград: зб. тез. – Кіровоград: КНТУ, 2014. – С. 53-55.

21. Коваленко А.С. Створення систем технічної діагностики для автоматизації процесів керування в інтегрованих інформаційних системах / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Проблеми і перспективи розвитку ІТ-індустрії: VI між нар. наук.-практ. конф., 17-18 квіт. 2014 р., м. Харків: зб. тез. – Харків: ХНЕУ, 2014. – С. 241.

22. Коваленко А.С. Визначення понятійного апарату та напрямів досліджень для синтезу систем технічної діагностики інтегрованих інформаційних систем / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Комп'ютерне моделювання у наукоємних технологіях (КМНТ-2014): наук.-техн. конф. з міжнар. участю, 28 -31 трав. 2014 р., м. Харків: зб. наук. праць. – Харків: ХНУ, 2014. – С. 190-193.

23. Коваленко А.С. Основні складові та функції системи технічної діагностики інтегрованих інформаційних систем / Коваленко А.С. // Інформаційні технології та комп'ютерна інженерія: наук.-практ. конф., 4 груд. 2014 р., м. Кіровоград: зб. тез доп. – Кіровоград: КНТУ, 2014. – С. 236.

24. Коваленко А.С. Розробка структури бази даних інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Проблеми і перспективи розвитку ІТ-індустрії: VII міжнар. наук.-практ. конф., 17-18 квіт. 2015 р., м. Харків: зб. тез. – Харків: ХНЕУ, 2015. – С. 15.

25. Коваленко А.С. Дослідження елементів інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Комбінаторні конфігурації та їх застосування: XVII між нар. наук.-практ. сем., 17-18 квіт. 2015 р., м. Кіровоград: зб. тез – Кіровоград: КНТУ, 2015. – С. 5.

26. Коваленко А.С. Метод автоматизованої перевірки результатів вимірювання параметрів об'єкті в інтегрованої інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Стратегія якості у

промисловості і освіті: XI міжнар. конф., 1 – 5 черв. 2015 р., м. Варна, Болгарія.: зб. матер. – Варна: ТУВ, 2015. – С. 423-426.

27. Коваленко А.С. Обґрунтування необхідності створення розподіленої бази даних для забезпечення захисту рухомих повітряних об`єктів / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Перспективні напрями захисту інформації: I всеукр. наук.-практ. конф., 07 вер. 2015 р., м. Одеса: зб. тез доп. – Одеса: ОНАЗ, 2015. – С. 35-39.

28. Коваленко А.С. Розробка інформаційної моделі автоматизованої оцінки технічного стану інтегральної інформаційної системи / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Інформаційні технології та взаємодії (IT & I): II між нар. наук.-практ. конф., 3-5 лист. 2015 р., м. Київ: тези доп. – Київ: КНУ ім. Т. Шевченка, 2015. – С. 41-42.

29. Коваленко А.С. Разработка метода усовершенствования технического обслуживания интегрированной информационной системы / А.С. Коваленко, О.А. Смірнов, О.В. Коваленко // Информационные и телекоммуникационные технологии: образование, наука, практика: II междунар. научн.-практ. конф., 3-4 дек. 2015 г., г. Алматы, Казахстан: сб. труд. – Алматы: КазНИТУ им. К.И. Сатпаева, 2015. – Т.2. – С. 423-427.

30. Королюк Н.А. Оценка временных интервалов работы лица, принимающего решение, на автоматизированном командном пункте / Н.А. Королюк, А.И. Тимочко // Системи обробки інформації. – Х.: ХУПС, 2005. – Вип. 8 (48). – С. 51-54.

31. Костерев В.В. Надёжность технических систем и управление риском: учебн. пособ. / В.В. Костерев. – М.: МИФИ, 2008. – 280 с.

32. Коффрон Дж. Технические средства микропроцессорних систем. – М.: Мир, 1983

33. Хвощ С.Т., Варлинский Н.Н., Попов Е.А. Микропроцессори и микроЭВМ в системах автоматического управления. – Л.: Машиностроение, 1987.

34. Хоровиц П., Хеши У. Искусство схемотехники. –М.: Мир, 1986.

35. Микропроцессори и микропроцессорние комплекти интегральних микросхем/справочник – М.: Радио и связь, 1986.

36. Шило В.Л. Популярние цифровие микросхеми: справочник. – Челябинск: Металлургия, 1986.

37. Якубовский С.В. Цифровие и аналоговие интегральние микросхеми: Справочник. – М.: Радио и связь, 1989.

38. Александров К.К., Кузьмина Е.Г. Электротехнические чертежи и схеми. – М.: Энергоатомиздат, 1990.

39. Павловский В.В., Васильев В.И., Гутман Т.Н. Проектирование технологических процессов изготовления РЭА / Пособие по курсовому проектированию для ВУЗов. – М.: Радио и связь, 1982.

40. Парфенов К.М. Проектирование конструкций РЭА. – М.: Радио и связь, 1989.

41. Егоров В.А., Лебедев К.М. и др. Конструкторско-технологическое проектирование печатних узлов / Учебное пособие. – СПб, 1995.

42. Литвак Б.Г. Экспертные технологии в управлении. Учебное пособие / Б.Г. Литвак. – М.: Дело, 2014. – 318 с.

43. Локазюк В.М. Надійність, контроль, діагностика і модернізація ПК: Посібн. / В.М. Локазюк, Ю.Г. Савченко. – К.: Видавничий центр «Академія», 2004. – 376 с.

44. Лопатников Л. И. Экономико-математический словарь: Словарь современной экономической науки / Л.И. Лопатников. – М.: Дело, 2003. – 520 с.

45. Манухина С.Ю. Инженерная психология и эргономика: хрестоматия / С.Ю Манухина. – М.: Изд. центр ЕАОИ, 2009. –224 с.

46. Мартыненко М.В. Человекомашинные процедуры поддержки организационно–управленческих решений: учеб. пособие СПбГЭТУ / М.В. Мартыненко, О.И. Шеховцов. – СПб, 2012. – 250 с.

47. Мунипов О.В. Эргономика: человекоориентированное проектирование техники, программных средств и среды: Учебник / О.В. Мунипов, В.П. Зинченко. – М.: Логос, 2001. – 356 с.

48. Надеев А.И. Математическая модель эксплуатационной надежности интеллектуальных датчиков / А.И. Надеев, Р.А. Юсупов, Ю.К. Свечников, Д.Р. Юсупов // Измерительная техника. – М: Стандартинформ, 2004. – № 1. – С. 8-11.

49. Надійність техніки. Аналіз надійності. Основні положення: ДСТУ 2861-94 – [Чинний від 1997–01–01]. – Київ: Держстандарт України, 1995. – 33 с. – (Національний стандарт України).

50. Надійність техніки. Терміни та визначення: ДСТУ 2860-94 – [Чинний від 1996–01–01]. – Київ: Держстандарт України, 1994. – 36 с. – (Національний стандарт України).

51. Нейлор К. Как построить свою экспертную систему / К. Нейлор. – М.: Энегроатомиздат, 2007. – 242 с.

52. Зеркалов Д. В. Охорона праці в Галузі: Загальні вимоги: навч. посіб. Київ: Основа. 2011. 551 с.

53. Про охорону праці: Закон України від 14.10.1992 р. № 2694-XII. Режим доступу до ресурсу: https://zakon.rada.gov.ua/laws/show/2694-12#Text

54. Сакулин В.П., Шептовицкий В.М. Безопасность труда при монтаже и эксплуатации электроустановок / В.П.Сакулин, В.М.Шептовицкий. – Л. : “Колос”, 1973. – 238 с.

55. Охорона праці. Ч. 1. Захисне заземлення: метод. вказ. до викон. розрахунків з викор. персон. ЕОМ ІВМ сумісного типу / Кіровоград. ін-т с.-г. машинобуд.; [укл. О. В. Оришака, Є. К. Солових, В. О. Оришака]. – Кіровоград: КІСМ, 1997. – 20 с. Режим доступу до ресурсу: http://dspace.kntu.kr.ua/jspui/handle/123456789/4358

56.

Додаток А
(обов'язковий)

Технічне завдання

Зміст

					ВКРМ-123.21.0085.00.00.ТЗ		
Вим.	Арк.	№ документа	Підпис	Дата			
Розробив		Сільченко С.М.			Дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S	Літ.	Аркуш 1 / Аркушів 6
Перевірив		Коваленко О.В.				М	
Н. Контр.		Гермак В.С.				ЦНТУ КІ-20М3	
Затв.		Смірнов О.А.					

1 Найменування та область застосування

Це технічне завдання розповсюджується на дослідження та програмну реалізацію системи керування супутниковим HD ресивером на базі процесору GX6605S.

2 Підстава для розробки

Підставою для розробки служить завдання на випускну кваліфікаційну роботу за другим (магістерським) рівнем вищої освіти, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 41-13 від 02.08.2021 року).

3 Мета та призначення розробки

Метою випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є дослідження та програмна реалізація системи керування супутниковим HD ресивером на базі процесору GX6605S.

4 Джерела розробки

Джерелом цієї випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти є стосовна до теми література і існуючі аналоги.

5 Технічні вимоги

5.1 Склад продукції

Складниками розробки є:

– вибір і обґрунтування методів реалізації проекту;

– розробка програмної частин системи, а також розробка взаємодії системи з ОС та з користувачем;

– техніко-економічне обґрунтування доцільності прийнятого до розробки програмного забезпечення;

– аналіз умов праці;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– програмну реалізацію системи керування супутниковим HD ресивером на базі процесору GX6605S;

– цілісність даних у процесі роботи та при зберіганні;

– простий, інтуїтивно зрозумілий інтерфейс.

5.3 Вимоги до функціональних характеристик

Розроблене програмне забезпечення не повинно мати обмежень на версію драйверів та операційної системи.

5.4 Вимоги до архітектури

Компонент, що розробляється повинен використовувати системні засоби та апаратні засоби, що на даному етапі розвитку обчислювальної техніки найбільше поширені.

5.5 Вимоги до надійності

Програмні модулі написані по всім правилам, які стосуються стандартних викликів процедур, функцій, методів і форм, визначених технічною документацією на середовище розробки.

5.6 Умови експлуатації

Робочі місця користувачів ПЗ повинні задовольняти наступним умовам експлуатації:

– температура повітря: 19-20 град. по Цельсію;

– відносна вологість повітря до 80%;

– атмосферний тиск 107 кПа.

5.7 Вимоги до складу та параметрів технічних засобів

Програмне забезпечення повинно бути реалізоване на ПЕОМ архітектури IBM PC, працювати в ОС Windows XP/Vista/7/8/10 і з сумісними з цією платформою пристроями і прикладним програмним забезпечення.

5.8 Вимоги до інформаційної і програмної сумісності

Переносність програмного забезпечення повинна бути забезпечена за рахунок його реалізації стандартного інтерфейсу взаємодії з ОС, що працюють під управлінням ОС Windows XP/Vista/7/8/10.

5.8.1 Обладнання

Комп'ютер Intel® Celeron/8 Mb/1.2 Gb/SVGA 14" 1Mb або сумісні з ним.

5.8.2 Мова програмування

Середовище Delphi 10.4 Sydney.

5.8.3 Вхідні дані

Опис алгоритму роботи запропонованої системи.

5.8.4 Вихідні дані

Робоча програма.

6 Вимоги до програмної документації

Програмна продукція повинна бути представлена у виді опису структури даних, схем та опису алгоритму, а також текстів вихідних модулів програмного забезпечення згідно ЄСПД .

7 Економічні вимоги

7.1 Для ПЗ необхідно виробити функціонально-вартісний аналіз варіантів розробки.

7.2 Виконати розрахунок витрат показників економічного ефекту з урахуванням цін на 3 вересня 2021 року.

8 Вимоги щодо охорони праці

В частині охорони праці випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти повинні бути розглянуті шкідливі і небезпечні фактори при роботі з комп’ютером.

9 Перелік документів, що розробляються

– Наукова новизна – 1 аркуш.
– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Показники економічної ефективності – 1 аркуш.
– Пояснювальна записка – 96 аркушів.

10 Етапи розробки

10.1 Збір і обробка інформації по темі випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти. Постановка задачі на виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти (складання ТЗ).

10.2 Проведення досліджень або експериментальних робіт для уточнення основних положень випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти.

10.3 Розробка функціональних схем, блок схем алгоритмів роботи програмного забезпечення.

10.4 Побудова схем взаємодії даних.

10.5 Створення прототипу ПЗ.

10.6 Віднаходження ПЗ, аналіз отриманих результатів.

10.7 Робота над питанням охорони праці і техніки безпеки.

10.8 Розрахунок з техніко-економічного обґрунтування.

10.9 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на попередній захист 10.12.2021 р.

11.2 Подання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти на захист 22.12.2021 р.

Додаток Б

(обов'язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник випускної кваліфікаційної роботи за

другим (магістерським) рівнем вищої освіти

_____ Коваленко О.В.

Дослідження та програмна реалізація

системи керування супутниковим HD ресивером на базі процесору GX6605S

Лістинг програми

Код документу 12

Носій: CD/DVD-диск

Загальна кількість аркушів: 53

Літера: РП

Кропивницький – 2021 року

Програма організації зв`язку

```
program MRTuner;
//Дослідження та програмна реалізація системи керування  супутниковим HD
ресивером на базі процесору GX6605S
// Розробив - Сільченко Сергій Миколайович, КІ-20МЗ, 2021 рік
uses
  PlataSaa7146aIOControl,
  Forms, tDiSEqC in 'DiSEqC.pas' {frmDiSEqC},
tMain in 'Main.pas' {frmMain};

{$R *.RES}
// Параметри супутників по замовчанню
//Satellite 1 = Astra   19.2E
//Satellite 2 = Hotbird 13E

begin
  Application.Title := 'TUNER';
  Application.Initialize;
  Application.CreateForm(TfrmMain, frmMain);
  Application.CreateForm(TfrmDiSEqC, frmDiSEqC);
  Application.Run;
end.
```

Файл програми Main.pas

```
unit tMain;

interface
// Розробив - Сільченко Сергій Миколайович, КІ-20МЗ, 2021 рік
uses
  IdGlobal, IdUDPClient,
  Forms, ExtCtrls, StdCtrls, Mask, Controls, Classes, Gauges, ComCtrls,
  UnitVolume,
  SetupCard,
  SetupDiSEqC,
  SetupDirectShow,
  SetupFiles,
  SetupMiscellaneous,
  SetupRemote,
  Recording,
  Messages, Buttons, Graphics, Menus, Dialogs, Grids, Outline, DirOutln,
  FileCtrl, Spin,
  RichView, RVStyle,
  Windows;

const
  WM_REMOTE = WM_USER + 1;
  WM_SIGNALLING = WM_USER + 10;
  MessageRight: array[0..3] of Char = 'RIGH';
  MessageRecord: array[0..3] of Char = 'RECO';
  MessageRecordOn: array[0..3] of Char = 'REON';
  MessageRecordOff: array[0..3] of Char = 'REOF';
  MessageEPGShort: array[0..3] of Char = 'EPGS';
  MessageEPGCurrent: array[0..3] of Char = 'EPGC';
  MessageVideo: array[0..3] of Char = 'VIDE';

type
  TfrmMain = class(TForm)
    tmrUpdate: TTimer;
    pgInfo: TPageControl;
    tsTuning: TTabSheet;
    tsProgram: TTabSheet;
    tsMessages: TTabSheet;
    mmoDriver: TMemo;
    cmbTeletextPids: TComboBox;
    Label16: TLabel;
    mskPmt: TMaskEdit;
    chkRecordTransponder: TCheckBox;
    Label28: TLabel;
    txtVideoPackets: TLabel;
    stbStatus1: TStatusBar;
    stbStatus2: TStatusBar;
    chkEpg: TCheckBox;
    tbFile: TTrackBar;
```

```
mnuMain: TMainMenu;
mnuExit: TMenuItem;
Processrawdatastream1: TMenuItem;
ProcessRecordedTSFile: TMenuItem;
mnuPlugins: TMenuItem;
mnuDummyPlugins: TMenuItem;
chkEPGShortNameOnly: TCheckBox;
rgOperation: TRadioGroup;
Label19: TLabel;
txtPacketSyncErrors: TLabel;
chkAutoECM: TCheckBox;
tmrStateMachineAutoECM: TTimer;
chkRecordPat: TCheckBox;
pnlVideo: TPanel;
pnlBorder: TPanel;
Settings1: TMenuItem;
mnuTV: TMenuItem;
mnuRadio: TMenuItem;
mnuData: TMenuItem;
cmbServices: TComboBox;
tmrStateMachineKeys: TTimer;
tmrInitStart: TTimer;
chkRecordEmm: TCheckBox;
chkRecordMandatory: TCheckBox;
chkRecordCat: TCheckBox;
pnlControl: TPanel;
imgBackgroundButtons: TImage;
imgRecord2: TImage;
imgRecord: TImage;
txtFiltersDefinedPlugin3: TLabel;
txtFiltersDefinedPlugin4: TLabel;
Label2: TLabel;
txtFiltersPlugin1: TLabel;
txtFiltersPlugin2: TLabel;
txtFiltersPlugin3: TLabel;
txtFiltersPlugin4: TLabel;
mnuDisabled: TMenuItem;
btnDisableEnable: TButton;
mnuDirectShow: TMenuItem;
SetProcessDelay: TMenuItem;
SetProcessDelayTo5ms: TMenuItem;
SetProcessDelayTo10ms: TMenuItem;
SetProcessDelayTo15ms: TMenuItem;
SetProcessDelayTo20ms: TMenuItem;
SetProcessDelayTo50ms: TMenuItem;
SetProcessDelayTo0ms: TMenuItem;
txtNativeVideoSize: TLabel;
Label3: TLabel;
procedure imgRightClick(Sender: TObject);
procedure imgRecordClick(Sender: TObject);
procedure mskPmtKeyPress(Sender: TObject; var Key: Char);
procedure WriteSettings;
procedure FormKeyPress(Sender: TObject; var Key: Char);
```

```
    procedure tmrInitStartTimer(Sender: TObject);
    procedure FormResize(Sender: TObject);
    procedure pnlVideoDblClick(Sender: TObject);
    procedure btnScrambleClick(Sender: TObject);
    procedure mnuDirectShowClick(Sender: TObject);
    procedure mnuDirectShowGraphClick(Sender: TObject);
    procedure SetProcessDelayToMsClick(Sender: TObject);
    procedure mnuDirectShowOptionsClick(Sender: TObject);
    procedure SetupDirectShowExit(Sender: TObject);
    procedure mnuSetupCardClick(Sender: TObject);
    procedure SetupCardExit(Sender: TObject);
    procedure mnuSetupFilesClick(Sender: TObject);
    procedure SetupFilesExit(Sender: TObject);
    procedure mnuSetupDiSEqCClick(Sender: TObject);
    procedure SetupDiSEqCExit(Sender: TObject);
    procedure mnuSetupMiscellaneousClick(Sender: TObject);
    procedure SetupMiscellaneousExit(Sender: TObject);
    procedure mnuSetupRemoteClick(Sender: TObject);
  private
    FFullScreenMode: Boolean;
    FFullScreenForm: TForm;
    FStateMachineAutoEcm: Integer;
    FRichViewEventStart: array of TDateTime;
    FRichViewEventEnd: array of TDateTime;
    FSetupDirectShow: TfrmSetupDirectShow;
    procedure WMUser(var Msg: TMessage); message WM_USER;
    procedure WMRemote(var Msg: TMessage); message WM_REMOTE;
    procedure WMSignalling(var Msg: TMessage); message WM_SIGNALLING;
    procedure FullScreenMode(FullScreen: Boolean);
    procedure RichViewJump(Sender: TObject; id: Integer);
    function StartRecordingProgram: Byte;
    function StartRecording(Pids: array of TPidList): Byte;
    function GetRecordingPids(var PidsRecord: TPidLists): Boolean;
    function TransponderDisplay: Boolean;
    procedure ReadFavourites;
    procedure ReadLists;
    procedure ReadTransponders(TransponderFile: string);
    procedure EventUpdate;
    procedure TransponderChanged;
    procedure TransponderChangedTuner;
    procedure ResetDiSEqC;
    procedure FavouriteChanged;
    procedure SyncService(ServiceToSyncWith: Word);
    procedure SyncFavouriteToService(ServiceToSyncWith: Word);
    procedure ProgramChanged(UseCurrentSettings: Boolean);
    procedure PidChanged;
    procedure Scan;
    procedure VolumeUpdate(Sender: TObject);
    procedure DirectShowFiltersAddToMenu;
    procedure DirectShowGraphsAddToMenu;
  public
    { Public declarations }
    PrevSliderPosition: Integer;
```

```
  end;

var
  frmMain: TfrmMain;

implementation

{$R *.DFM}
uses
adCpuUsage,  DvbDirectShow2,  DvbDirectShow,  DvbStreamBuffering,
FlexCopGeneral, FlexCopDvbStreamBuffering,  FlexCopInterface,  FlexCopI2c,
FlexCopIoControl, FlexCopRegisters,  Saa7146AInterface,  Saa7146AI2c,
Saa7146aIoControl, Saa7146aRegisters, Stv0299bRegisters,  Tsa5059Registers,
Registry,  SyncObjs,  SysUtils;

const
  Version = $0007;
  SubVersion = '3';
  Build = $20060427;

type
  TOperation = (otNoOp, otNormal, otError); // Type of 'hardware' using

  TDataThread = class(TThread)
  private
    FHasStopped: Boolean; // Flag indicating thread not running
    FProgrammedStop: Boolean; // If true indicates programmed termination
    FPacketBuffers: Word; // Number of buffers used
    FFileStream: TFileStream; // Assigned if file stream instead of satellite
    FFileStreamSize: Int64; // Size of file currently handling
    FFileStreamStop: Boolean; // Set to True when file stream should be
stopped
    FFileStreamDelay: Word; // Delay to use during file stream mode
    FOperation: TOperation; // Operational types
    FStreamData: PDvbTransportPackets; // Current pointer to stream data
    procedure DecodeTransportStreamData;
    procedure DescrambleFFDeCsa;
  protected
    procedure Execute; override;
  end;

  TRcThread = class(TThread)
  private
    HasStopped: Boolean; // Flag indicating thread not running
    ProgrammedStop: Boolean; // If true indicates programmed termination
  protected
    procedure Execute; override;
  end;

  TMDAPIFilterProc = procedure(hFilter: DWORD; Len: Cardinal; Buf:
PByteArray);
  cdecl;
```

```
  TFilter = record
    Name: string; // Name of filter
    Active: Boolean; // Filter is (to be) set active
    Pid: Word; // Only used for cross reference
    FilterId: Word; // Filter ID as used by plugin
    CallBackFunction: TMDAPIFilterProc; // Function to call
  end;
  PTFilters = ^TFilters;
  TFilters = record
    Filters: array[1..255] of TFilter; // The filter definitions
    CrossReference: array[0..$1FFF] of Byte; // Pid index references to
<Filters>
    Active: Word; // Number of active filters
    ActiveText: string; // Active filters as text
  end;

procedure GetAndWriteDirectShowSetup(SetupForm: TfrmSetupDirectShow;
ForceWrite:
  Boolean); forward;
procedure SetDirectShowSetup(SetupForm: TfrmSetupDirectShow); forward
  procedure GetAndWriteCardSetup(SetupForm: TfrmSetupCard; ForceWrite:
Boolean);
  forward;
procedure SetCardSetup(SetupForm: TfrmSetupCard); forward
  procedure GetAndWriteFilesSetup(SetupForm: TfrmSetupFiles; ForceWrite:
  Boolean); forward;
procedure SetFilesSetup(SetupForm: TfrmSetupFiles); forward
  procedure GetAndWriteDiSEqCSetup(SetupForm: TfrmSetupDiSEqC; ForceWrite:
  Boolean); forward;
procedure SetDiSEqCSetup(SetupForm: TfrmSetupDiSEqC); forward
  procedure GetAndWriteMiscellaneousSetup(SetupForm: TfrmSetupMiscellaneous;
  ForceWrite: Boolean); forward;
procedure SetMiscellaneousSetup(SetupForm: TfrmSetupMiscellaneous); forward
  procedure GetAndWriteRemoteSetup(SetupForm: TfrmSetupRemote; ForceWrite:
  Boolean); forward;
procedure SetRemoteSetup(SetupForm: TfrmSetupRemote); forward
  procedure GetAndWriteRecordings(SetupForm: TfrmRecording; ForceWrite:
  Boolean); forward;
function SetRecordings(SetupForm: TfrmRecording): Boolean; forward
  procedure ReadRecordings; forward;
procedure WriteRecordings; forward;

const
  MDAPI_GET_TRANSPONDER = $01020000;
  MDAPI_SET_TRANSPONDER = $01020001;
  MDAPI_GET_PROGRAM = $01020010;
  MDAPI_SET_PROGRAM = $01020011;
  MDAPI_RESCAN_PROGRAM = $01020012;
  MDAPI_SCAN_CURRENT_CAT = $01020031;
  MDAPI_START_OSD = $01020040;
  MDAPI_OSD_DRAWBLOCK = $01020041;
  MDAPI_OSD_SETFONT = $01020042;
  MDAPI_OSD_TEXT = $01020043;
```

```
  MDAPI_SEND_OSD_KEY = $01020044;
  MDAPI_STOP_OSD = $01020049;
  MDAPI_DVB_COMMAND = $01020060;
  MDAPI_GET_VERSION = $01020100;

type
  PDVB_COMMAND = ^TDVB_COMMAND;
  TDVB_COMMAND = record
    CmdLength: Word;
    CmdBuffer: array[0..31] of Byte;
  end;

  PProgramNumberParam = ^TProgramNumberParam;
  TProgramNumberParam = record
    RealNumber: Integer;
    VirtNumber: Integer;
  end;

  PStartFilterParam = ^TStartFilterParam;
  TStartFilterParam = record
    DLLIdentifier: Word;
    Filter_ID: Word;
    Pid: Word;
    Name: array[0..31] of Byte;
    CallAddress: LongInt;
    Running_ID: Integer;
  end;

  TPIDFilters = record
    FilterName: array[0..4] of Char;
    FilterId: Byte;
    PID: Word;
  end;

  TCA_System82 = record
    CA_Type: Word;
    ECM: Word;
    EMM: Word;
    Provider_Id: Dword;
  end;

  PProgram82 = ^TProgram82;
  TProgram82 = record
    Name: array[00..29] of Char;
    Provider: array[00..29] of Char;
    Country: array[00..29] of Char;
    Freq: Dword;
    PType: Byte;
    Voltage: Byte;
    Afc: Byte;
    DiSEqC: Byte;      Symbolrate: Word;
    Qam: Word;         Fec: Word;
    Norm: Byte;        Tp_id: Word;
```

```
  Video_pid: Word;   Audio_pid: Word;
  TeleText_pid: Word;PMT_pid: Word;
  PCR_pid: Word;     ECM_pid: Word;
  SID_pid: Word;     AC3_pid: Word;
  TVType: Byte;      ServiceTyp: Byte;
  CA_ID: Byte;       Temp_Audio: Word;
  FilterNr: Word;    Filters: array[00..31] of TPIDFilters;
  CA_Nr: Word;       CA_System82: array[0..31] of TCA_System82;
  CA_Country: array[0..5] of Char;
  Marker: Byte;      Link_TP: Word;
  Link_SID: Word;    PDynamic: Byte;
  Extern_Buffer: array[00..15] of Char;
end;

TMDAPIInitPluginProc =
  procedure(MDInstance: LongWord; MDWnd: HWND;
  Log_Set: Bool; DLL_ID: Integer; HotKey: PChar; Vers: PChar;
  var ReturnValue: Integer); cdecl;
TMDAPIGetPluginName =
  procedure(Buf: PByteArray); cdecl;
TMDAPIProcessPluginMenuCommandProc =
  procedure(MenuID: integer); cdecl;
TMDAPIProcessChannelChange =
  procedure(prg: TProgram82); cdecl;
TMDAPIExitPluginProc =
  procedure(MDInstance: LongWord; MDWnd: HWND; Log_Set: Bool); cdecl;
TMDAPIProcessHotKeyProc =
  procedure; cdecl;
TMDAPIFilterCloseProc =
  procedure(MDInstance: LongWord); cdecl;
TMDAPIProcessRecPlayProc =
  procedure(Mode: Integer); cdecl;

TMdPlugin = record
  DllIdentifier: THandle; // Handle to DLL
  Name: string; // Name of plugin
  HotKey: Char; // Hot key
  PacketSize188: Boolean; // Packet size 188 bytes
  Filters: TFilters; // Pid filters
  PluginMenu: HMENU; // Menu of plugin
  GetPluginName: TMDAPIGetPluginName;
  InitPlugin: TMDAPIInitPluginProc;
  ProcessPluginMenuCommand: TMDAPIProcessPluginMenuCommandProc;
  ProcessChannelChange: TMDAPIProcessChannelChange;
  ExitMDPlugin: TMDAPIExitPluginProc;
  ProcessHotKey: TMDAPIProcessHotKeyProc;
  ProcessFilterClose: TMDAPIFilterCloseProc;
  ProcessRecPlay: TMDAPIProcessRecPlayProc;
end;

TSetKeyProc = procedure(Key: PByteArray; KeyStruct: PByteArray); cdecl;
TCSADecryptProc = procedure(KeyStruct: PByteArray; Encrypted: PByteArray);
cdecl;
```

```
  // FFDeCSA
  TFFDeCsaCluster = packed record
    StartBuffer: Pointer;
    EndBuffer: Pointer;
  end;
  PFFDeCsaClusters = ^TFFDeCsaClusters;
  TFFDeCsaClusters = array[0..CDvbPacketVSync] of TFFDeCsaCluster;
  TFFSetKeyProc = function(Even, Odd, KeySet: PByteArray): Integer; stdcall;
  TFFDecryptProc = function(Cluster: PFFDeCsaClusters; KeySet: PByteArray):
    Integer; stdcall;
  TFFKeySizeProc = function: Integer; stdcall;
  TFFParallelismProc = function: Integer; stdcall;
  TRecording = (rtIdle, rtStartRecording, rtRecording, rtStopRecording);

const
  COsdSizeBitmapHeader = sizeof(BITMAPINFO) + 256 * sizeOf(TRGBQuad);
  COsdTransparentColor = $00112211;
  COsdTransparentColorRGB: TRGBQuad = (
    rgbBlue: ((COsdTransparentColor shr 0) and $FF);
    rgbGreen: ((COsdTransparentColor shr 8) and $FF);
    rgbRed: ((COsdTransparentColor shr 16) and $FF);
    rgbReserved: 0);
  COsdWidth = 500;
  COsdHeight = 500;

type
  PTMainWindowOsdData = ^TMainWindowOsdData;
  TMainWindowOsdData = array[0..(COsdWidth * COsdHeight) - 1] of TRGBQuad;

  BitmapDef = record
    Header: array[0..COsdSizeBitmapHeader] of Byte;
    Info: PBitmapInfo;
    Handle: HBITMAP;
    DC: HDC;
    Data: PTMainWindowOsdData;
  end;

var
  OsdBitmaps: array[0..4] of BitmapDef;
  OsdTimeout: Integer;

 var
  AppName: string; // Application name (header)
  ExeDirectory: string; // Directory of executable

  GUDPClient   : TIdUDPClient;
  GUDPBuffer   : TIdBytes;
  GUDPPointer  : PByte;
  GUDPLastIndex: Integer;
  GUDPIndex    : Integer;

  IsSlave: Boolean; // True if act as slave
  ThreadHandle: THandle; // Handle to driver in thread
```

```
PacketThread: TDataThread; // Thread handling packets
RemoteThread: TRcThread; // Thread handling remote control
CardHandle: THandle;
CardNumber: Integer; // Selected card (INI)
CardInUse: Integer; // Selected card actually in use
VideoPacketCount: Word; // Counts received video packets
VideoPacketPreviousCount: Word; // Counts received buffers previous check
NewChannelSet: Boolean; // Set to True when a new channel has been set
FiltersOff: Boolean; // TRUE when filters are not to be called
Rps0Program: Integer; // RPS0 program running
RecordingAllData: Boolean; // True indicates all data recording
RecordingPids: array[0..127] of TRecordPid; // What to record
RecordingTimerIndex: Byte; // Recording identifier timed/manual recording
RecordAllAudio: Boolean; // True will record all audio channels
RecordingDirectory: string; // Path of destination directory for recording
RecordDefault: string; // Recording defaults ('default' button')
RecordingAllFile: TFileStream; // Filestream for recording all data
RecordingAllFilePart: Integer; // Part of split file
RecordingAllFileFileName: ShortString; // Name of recording
RecordItems: TRecordItems; // Recordings
Satellite: Integer; // Current satellite
List: Integer; // Current list of favourites (index in list)
Favourite: Integer; // Current favourite (index in list)
Transponder: Integer; // Current transponder (index in list)
Frequency: Dword; // Current frequency
ProgramName: string; // Current program name
ServiceNumber: Word; // Current service identifier
PidVideo: Word; // Current PID for video
PidAudio: Word; // Current PID for audio
PidAudioIsAc3: Boolean; // Indicates that audio PID is AC3
PidPCR: Word; // Current PID for PCR
PidPMT: Word; // Current PID for PMT
PidEcm: Word; // Current PID for ECM
PidTeletext: Word; // Current PID for Teletext
PidSubtitle: Word; // Current PID for subtitle
UpdatePmt: Boolean; // True if new PMT information available
UpdatePmtCount: Word; // Increases for each PMT information update
UpdatePat: Boolean; // True if new PAT information available
UpdateCat: Boolean; // True if new CAT information available
UpdateCatCount: Word; // Increases for each CAT information update
UpdateEvent: Boolean; // True if new event information available
Scanning: Boolean; // True when scanning
ProgramOptions: Word; // Current options of program
TimeCorrection: TDateTime; // Correction to apply to time of EPG
TimeCorrectionGmt: Boolean; // If True the TimeCorrectionBias is used
HighPerformanceFrequency: TLargeInteger; // High performance frequency
LNBLofLowBand: Integer; // Low band Local Oscillator Frequency
LNBLofHighBand: Integer; // High band Local Oscillator Frequency
DiSEqCRepeats: Byte; // Repeats for DiSEqC
DiSEqCUsePositioner: Boolean;
// True will use positioner command instead of committed switch
DiSEqCMini: Boolean; // True will use mini-DiSEqc (burst) only
FontName: string; // Name of font we are using
```

```
  FontSize: Integer; // Font size
  FontNameEpg: string; // Name of font we are using
  FontSizeEpg: Integer; // Font size
  //  StreamScrambled: Boolean;
  // Flag indicating scrambled state of stream (audio or video)
  VideoScrambled: Boolean; // Flag indicating scrambled state of video stream
  FreezeTimeout: Integer; // Exception setting
  // True if to continue on exception (otherwise exception box)
  KeySwapUpDown: Boolean; // Flag which swaps up/down button actions

type
  TCsaMethod = (cmNone, cmInternal, cmCsa, cmFfCsa);
var
  CsaMethod: TCsaMethod; // CSA method
  CsaName: string; // Current CSA (empty for internal)
  CsaDLLIdentifier: Integer;
  SetKeyProc: TSetKeyProc;
  CSADecryptProc: TCSADecryptProc;
  KeyStruct: array[0..800] of Byte;
  Key: array[0..15] of Byte;
  KeysOdd: TBlock;
  KeysEven: TBlock;
  KeysPreviousOdd: TBlock;
  KeysPreviousEven: TBlock;
  FFSetKeyProc: TFFSetKeyProc;
  FFDecryptProc: TFFDecryptProc;
  FFKeySizeProc: TFFKeySizeProc;
  FFParallelismProc: TFFParallelismProc;

  FFCsaKeys: PByteArray;
  FFCsaKeySize: Integer;
  FFCsaClusters: TFFDeCsaClusters;

procedure OsdFillColor(Index: Byte; Color: TRGBQuad);
var
  Width: Integer;
  height: Integer;
begin
  if (Index > High(OsdBitmaps)) then
    Exit;
  for Width := 0 to COsdWidth - 1 do
    for Height := 0 to COsdHeight - 1 do
      OsdBitmaps[Index].Data[Width * COsdWidth + Height] := Color;
end;

procedure OsdClear;
begin
  if (DsOptions and CDirectShowMethodAlternative) <> 0 then
    DvbDirectShow2.DirectShowBlendImage(OsdBitmaps[0].DC, Rect(0, 0,
COsdWidth,
      COsdHeight), 0.0, COsdTransparentColor)
  else
```

```
    DvbDirectShow.DirectShowGraph.DirectShowBlendImage(OsdBitmaps[0].DC,
Rect(0,
      0, COsdWidth,
      COsdHeight), 0.0, COsdTransparentColor);
end;

procedure OsdShowInfo;
var
  DisplayStr: string;
begin
  SetBkColor(OsdBitmaps[1].DC, clWhite); // Pure white background
  SetTextColor(OsdBitmaps[1].DC, clBlack); // Write text with requested color
  OsdFillColor(1, COsdTransparentColorRGB);
  DisplayStr := format(' %4.4d - %s ', [Favourite + 1, ProgramName]);
  TextOut(OsdBitmaps[1].DC, 200, 250, @DisplayStr[1], Length(DisplayStr));
  if (DsOptions and CDirectShowMethodAlternative) <> 0 then
    DvbDirectShow2.DirectShowBlendImage(OsdBitmaps[1].DC, Rect(0, 0,
COsdWidth,
      COsdHeight), 1.0, COsdTransparentColor)
  else
    DvbDirectShow.DirectShowGraph.DirectShowBlendImage(OsdBitmaps[1].DC,
Rect(0,
      0, COsdWidth,
      COsdHeight), 1.0, COsdTransparentColor);
end;

function EnableShutdownPrivileges: Boolean;
var
  ProcessHandle: THandle;
  TokenHandle: THandle;
  Token: TTokenPrivileges;
  Luid: TLargeInteger;
  PreviousState: TTokenPrivileges;
  ReturnLength: Dword;
  VersionInfo: TOSVersionInfo;
begin
  Result := False;
  VersionInfo.dwOSVersionInfoSize := SizeOf(TOSVersionInfo);
  if not GetVersionEx(VersionInfo) then
    Exit;
  if not ((VersionInfo.dwPlatformId = VER_PLATFORM_WIN32_NT) and
    (VersionInfo.dwMajorVersion >= 4)) then
  begin
    Result := True;
    Exit;
  end;
  ProcessHandle := GetCurrentProcess;
  if ProcessHandle <> 0 then
  begin
    if OpenProcessToken(ProcessHandle, TOKEN_ADJUST_PRIVILEGES or TOKEN_QUERY,
      TokenHandle) then
    begin
      if LookupPrivilegeValue('', 'SeShutdownPrivilege', Luid) then
```

```
      begin
        Token.PrivilegeCount := 1;
        Token.Privileges[0].Luid := Luid;
        Token.Privileges[0].Attributes := SE_PRIVILEGE_ENABLED;
        if AdjustTokenPrivileges(TokenHandle, False, Token,
          SizeOf(PreviousState), PreviousState, ReturnLength) then
          Result := True
      end;
    end;
  end;
end;

function ChangeTimeSeparator(ToChange: string): string;
var
  ThePos: Integer;
begin
  if TimeSeparator <> ':' then
  begin
    ThePos := Pos(':', ToChange);
    while ThePos <> 0 do
    begin
      ToChange[ThePos] := TimeSeparator;
      ThePos := Pos(':', ToChange);
    end;
  end;
  Result := ToChange;
end;

procedure Shutdown;
var
  Flags: Integer;
begin
  Flags := EWX_POWEROFF or EWX_FORCE or EWX_FORCEIFHUNG;
  if EnableShutdownPrivileges then
    ExitWindowsEx(Flags, 0);
end;

procedure ToLog(LogString: string; Level: Byte);
var
  NewLog: string;
begin
  if ((Level and $80) <> 0) and
    Assigned(frmMain) and
    Assigned(frmMain.mmoDriver) then
    frmMain.mmoDriver.Lines.Add(LogString);
  if (Level and $0F) > LogLevel then
    Exit;
  if not Assigned(LogStream) then
    Exit;
  NewLog := FormatDateTime('YYYYMMDD"T"HHMMSS"  "', Now) + LogString + #13#10;
  LogStream.Write(NewLog[1], Length(NewLog));
end;
procedure FilterUpdateCrossReference(Filter: PTFilters);
```

```
var
  FilterIndex: Integer;
begin
  // Clear cross references
  Filter.Active := 0;
  Filter.ActiveText := '';
  for FilterIndex := Low(Filter.CrossReference) to High(Filter.CrossReference)
    do
    Filter.CrossReference[FilterIndex] := 0;
  // Renew cross reference
  for FilterIndex := Low(Filter.Filters) to High(Filter.Filters) do
    if Filter.Filters[FilterIndex].Active then
    begin
      Filter.CrossReference[Filter.Filters[FilterIndex].Pid] := FilterIndex;
      Inc(Filter.Active);
      if Filter.ActiveText = '' then
        Filter.ActiveText := format('%d', [Filter.Filters[FilterIndex].Pid])
      else
        Filter.ActiveText := Filter.ActiveText + ' ' + format('%d',
          [Filter.Filters[FilterIndex].Pid]);
    end;
end;

procedure FilterRemoveByName(Filter: PTFilters; Name: string);
var
  FilterIndex: Integer;
begin
  ToLog('FilterRemoveByName: [' + Name + ']', $5);
  FilterLock.Acquire;
  try
    for FilterIndex := Low(Filter.Filters) to High(Filter.Filters) do
    begin
      Name := LowerCase(Name);
      if Name = 'video' then
        PidVideo := 0;
      if Name = 'audio' then
        PidAudio := 0;
      if Name = 'pmt' then
        PidPmt := 0;
      if Name = 'pcr' then
        PidPcr := 0;
      if Name = 'ecm' then
        PidEcm := 0;
      if Name = 'teletext' then
        PidTeletext := 0;
      if Name = 'subtitle' then
        PidSubtitle := 0;
      if LowerCase(Filter.Filters[FilterIndex].Name) = Name then
      begin
        Filter.Filters[FilterIndex].Active := False;
        FilterUpdateCrossReference(Filter);
      end;
    end;
```

```
  finally
    FilterLock.Release;
  end;
end;

function DvbSetFilter(Filter: PTFilters; Pid: Word; FilterId: Word;
FilterProc:
  Pointer; Name: PChar): Dword; stdcall;
var
  FilterIndex: Integer;
  Valid: Boolean;
  PString: string;
begin
  ToLog(format('DvbSetFilter %d (%s).', [Pid, Name]), $04);
  PString := Name;
  if PString <> '' then
    FilterRemoveByName(Filter, PString);

  Result := 0;
  if Pid > High(Filter.CrossReference) then
    Exit;
  FilterLock.Acquire;
  try
    Valid := False;
    // First try to find existing active filter
    if Filter.CrossReference[Pid] <> 0 then
    begin
      FilterIndex := Filter.CrossReference[Pid];
      Valid := True;
    end
    else
    begin
      FilterIndex := Low(Filter.Filters);
      repeat
        if Filter.Filters[FilterIndex].Active then
          Inc(FilterIndex)
        else
          Valid := True;
      until Valid or (FilterIndex > High(Filter.Filters));
    end;
    if Valid then
    begin
      // Now set filter data and
      Filter.Filters[FilterIndex].Pid := Pid;
      Filter.Filters[FilterIndex].FilterId := FilterId;
      Filter.Filters[FilterIndex].CallBackFunction := FilterProc;
      Filter.Filters[FilterIndex].Name := Name;
      if LowerCase(Name) = 'video' then
        PidVideo := Pid;
      if LowerCase(Name) = 'audio' then
        PidAudio := Pid;
      if LowerCase(Name) = 'pmt' then
        PidPmt := Pid;
```

```
      if LowerCase(Name) = 'pcr' then
        PidPcr := Pid;
      if LowerCase(Name) = 'ecm' then
        PidEcm := Pid;
      if LowerCase(Name) = 'teletext' then
        PidTeletext := Pid;
      if LowerCase(Name) = 'subtitle' then
        PidSubtitle := Pid;
      Filter.Filters[FilterIndex].Active := True;
      FilterUpdateCrossReference(Filter);
      Result := FilterIndex;
    end;
  finally
    FilterLock.Release;
  end;
end;

procedure MdPluginExit(Plugin: Word);
var
  WriteLog: Boolean;
  Index: Integer;
begin
  ToLog(format('MdPlugExit: [%d]', [Plugin]), $5);
  try
    WriteLog := True;
    if MdPlugin[Plugin].DllIdentifier <> 0 then
      if Assigned(@MdPlugin[Plugin].ExitMdPlugin) then
      begin
        FilterLock.Acquire;
        try
          // Remove all filters
          for Index := Low(AppFilters.Filters) to High(AppFilters.Filters) do
            MdPlugin[Plugin].Filters.Filters[Index].Active := False;
          FilterUpdateCrossReference(@MdPlugin[Plugin].Filters);
        finally
          FilterLock.Release;
        end;
        MdPlugin[Plugin].ExitMdPlugin(HInstance, frmMain.Handle, WriteLog);
      end;
  except
  end;
end;

procedure CleanupMdPlugins;
var
  Index: Word;
begin
  ToLog('CleanupMdPlugins', $5);
  try
    try
      if MdPlugins > 0 then
        for Index := 0 to MdPlugins - 1 do
          MdPluginExit(Index);
```

```
      if MdPlugins > 0 then
        for Index := 0 to MdPlugins - 1 do
          DestroyMenu(MdPlugin[Index].PluginMenu);
    finally
      MdPlugins := 0;
    end;
  except
  end;
end;

procedure MdApiOnCloseFilters;
var
  Index: Word;
begin
  ToLog('MdApiOnCloseFilters', $5);
  try
    if MdPlugins > 0 then
      for Index := 0 to MdPlugins - 1 do
        if MdPlugin[Index].DllIdentifier <> 0 then
          if Assigned(@MdPlugin[Index].ProcessFilterClose) then
            MdPlugin[Index].ProcessFilterClose(HInstance);
  except
  end;
end;

procedure MdApiOnChannelChange(Prg: TProgram82);
var
  Index: Word;
begin
  ToLog('MdApiOnChannelChange', $5);
  try
    if MdPlugins > 0 then
      for Index := 0 to MdPlugins - 1 do
        if MdPlugin[Index].DllIdentifier <> 0 then
          if Assigned(@MdPlugin[Index].ProcessChannelChange) then
            MdPlugin[Index].ProcessChannelChange(Prg);
  except
  end;
end;

procedure SetCSAKeys(Command: array of Byte);
var
  i: Integer;
  j: Integer;
  Change: Boolean;
begin
  if CsaMethod = cmNone then
    Exit;
  if Command[4] = 1 then
    OddKeyFound := True
  else
    EvenKeyFound := True;
  for i := 0 to 3 do
```

```
  begin
    Key[(Command[4]) * 8 + i * 2 + 0] := Command[6 + i * 2 + 1];
    Key[(Command[4]) * 8 + i * 2 + 1] := Command[6 + i * 2 + 0];
    // If we are using the internal decrytor then prepare keys
    if CsaMethod in [cmInternal, cmFfCsa] then
    begin
      if Command[4] = 0 then
        for j := 0 to 3 do
        begin
          KeysEven[j * 2 + 0] := Command[6 + j * 2 + 1];
          KeysEven[j * 2 + 1] := Command[6 + j * 2 + 0];
        end
      else
        for j := 0 to 3 do
        begin
          KeysOdd[j * 2 + 0] := Command[6 + j * 2 + 1];
          KeysOdd[j * 2 + 1] := Command[6 + j * 2 + 0];
        end;
    end;
  end;
  if OddKeyFound and EvenKeyFound then
  begin
    KeyFound := True;
    // Internal handling checks for a change here
    if CsaMethod = cmInternal then
    begin
      Change := False;
      for j := 0 to 7 do
      begin
        if KeysOdd[j] <> KeysPreviousOdd[j] then
          Change := True;
        if KeysEven[j] <> KeysPreviousEven[j] then
          Change := True;
      end;
      if Change then
      begin
        CSASetKeys(KeysOdd, KeysEven);
        for j := 0 to 7 do
        begin
          KeysPreviousOdd[j] := KeysOdd[j];
          KeysPreviousEven[j] := KeysEven[j];
        end;
      end;
    end
    else
    begin
      if CsaMethod = cmCsa then
        SetKeyProc(@Key[0], @KeyStruct[0]);
      if CsaMethod = cmFfCsa then
        FFSetKeyProc(@KeysEven[0], @KeysOdd[0], FFCsaKeys);
    end;
  end;
end;
```

```
procedure TfrmMain.WMCommand(var Msg: TWMCommand);
var
  Index: Word;
begin
  ToLog(format('WMCommand: %d', [Msg.ItemId]), $5);
  try
    if MdPlugins > 0 then
      for Index := 0 to MdPlugins - 1 do
        if MdPlugin[Index].DllIdentifier <> 0 then
          if Assigned(@MdPlugin[Index].ProcessPluginMenuCommand) then
            MdPlugin[Index].ProcessPluginMenuCommand(Msg.ItemId);
  except;
  end;
  inherited;
end;

procedure UpdateProgram82(Program82: PProgram82);
var
  Index: Integer;
  EmmCaSystemId: Word;
  EmmCaPid: Word;
  EmmIndex: Word;
  DummyP: TDvbPids;
  DummyW: Word;
  Language: TDvbLanguages;
  PidsEcm: TDvbPids;
  CasId: TDvbPids;
  ValidCount: Integer;
  Nothing: Boolean;
  LogString: string;
  PrgName: string;
  Error: Integer;
  ProcessString: string;
begin
  StrPCopy(Program82.Name, Copy(ProgramName, 1, SizeOf(Program82.Name)));
  Program82.Freq := Frequency;
  Program82.Video_pid := PidVideo;
  Program82.Audio_pid := PidAudio;
  Program82.Teletext_pid := PidTeletext;
  Program82.PCR_pid := PidPcr;
  Program82.PMT_pid := PidPmt;
  Program82.ECM_pid := PidEcm;
  Program82.CA_ID := 0;
  Program82.SID_pid := ServiceNumber;

  Program82.CA_NR := 0;
  if TransponderHasChanged = $0000 then
  begin
    ToLog('UpdateProgram82 (not using realtime data yet)', $3);
    Exit;
  end;
```

```
LogString := 'UpdateProgram82 (' + ProgramName + '): ECM: ';
ValidCount := 5;
Nothing := True;
repeat
  if DvbGetProgramInfo($FF, ServiceNumber, DummyW, DummyW, DummyP, DummyP,
    DummyP, DummyP, Language, Language, PidsEcm, CasId, PrgName) then
  begin
    if PrgName <> '' then
      ProgramName := PrgName;
    StrPCopy(Program82.Name, Copy(ProgramName, 1, SizeOf(Program82.Name)));
    Index := 0;
    while ((PidsEcm[Index] <> 0) and (Index <= High(Program82.CA_System82)))
      do
    begin
      Program82.CA_System82[Index].CA_Type := CasId[Index];
      Program82.CA_System82[Index].ECM := PidsEcm[Index];
      Program82.CA_System82[Index].EMM := 0;
      Program82.CA_System82[Index].Provider_Id := ServiceNumber;
      LogString := Logstring + format('%d ', [PidsEcm[Index]]);
      Inc(Index);
    end;
    Program82.CA_NR := Index;
    ValidCount := 1;
    Nothing := False;
  end
  else
    Sleep(10);
  Dec(ValidCount);
until ValidCount = 0;
if Nothing then
  LogString := LogString + 'none  ';
LogString := LogString + '  EMM: ';
Nothing := True;
if frmMain.chkEmmManual.Checked then
begin
  if Program82.CA_NR > 0 then
  begin
    Nothing := False;
    ProcessString := frmMain.edtEmmPids.Text;
    ProcessString := Trim(ProcessString);
    while ProcessString <> '' do
    begin
      ProcessString := Trim(ProcessString);
      Val(ProcessString, EmmCaPid, Error);
      if Error <> 1 then
      begin
        if Error = 0 then
          ProcessString := ''
        else
          Delete(ProcessString, 1, Error - 1);
        Index := Program82.CA_NR;
        Program82.CA_System82[Index].ECM := 0;
        Program82.CA_System82[Index].Provider_Id := 0;
```

```
          Program82.CA_NR := Index + 1;

          Program82.CA_System82[Index].EMM := EmmCaPid;
          LogString := Logstring + format('%d ', [EmmCaPid]);
        end
        else
          Delete(ProcessString, 1, 1);
      end;
    end;
  end
  else
  begin
    frmMain.edtEmmPids.Text := '';
    if Program82.CA_NR > 0 then
    begin
      Nothing := False;
      // Get through EMM's one at the time
      EmmIndex := 0;
      while (DvbFilterGetEmm(EmmIndex, EmmCaSystemId, EmmCaPid) and
        (Program82.CA_NR <= High(Program82.CA_System82))) do
      begin
        Index := Program82.CA_NR;
        Program82.CA_System82[Index].ECM := 0;
        Program82.CA_System82[Index].Provider_Id := 0;
        Program82.CA_NR := Index + 1;

        Program82.CA_System82[Index].CA_Type := EmmCaSystemId;
        Program82.CA_System82[Index].EMM := EmmCaPid;
        LogString := Logstring + format('%d ', [EmmCaPid]);
        frmMain.edtEmmPids.Text := frmMain.edtEmmPids.Text + format('%d ',
          [EmmCaPid]);
        Inc(EmmIndex);
      end;
    end;
  end;
  if Nothing then
    LogString := LogString + 'none  ';
  ToLog(LogString, $3);
end;

procedure TfrmMain.WMUser(var Msg: TMessage);
var
  StartFilterParam: PStartFilterParam;
  Index: Byte;
  DebugString: string;
  Change: Boolean;
  Plugin: Integer;
begin
  if MdPlugins = 0 then
    Exit;
  case Msg.WParam of
    MDAPI_OSD_TEXT: DebugString := 'MDAPI_OSD_TEXT';
    MDAPI_SEND_OSD_KEY: DebugString := 'MDAPI_SEND_OSD_KEY';
```

```
  MDAPI_STOP_OSD: DebugString := 'MDAPI_STOP_OSD';
  MDAPI_DVB_COMMAND: DebugString := 'MDAPI_DVB_COMMAND';
  MDAPI_GET_VERSION: DebugString := 'MDAPI_GET_VERSION';
ToLog(format('MDAPI command %s %d %d.', [DebugString, Msg.LParam,
  Msg.WParam]), $05);

case Msg.WParam of
   MDAPI_GET_PROGRAM:
   UpdateProgram82(PProgram82(Msg.LParam));
   MDAPI_SET_PROGRAM:
    begin
      if MdPluginAllowSetPids then
      begin
        Change := False;
        if PidVideo <> PProgram82(Msg.LParam)^.Video_pid then
        begin
          PidVideo := PProgram82(Msg.LParam)^.Video_pid;
          frmMain.cmbVideoPids.Text := format('%4.4d', [PidVideo]);
          Change := True;
        end;
        if PidAudio <> PProgram82(Msg.LParam)^.Audio_pid then
        begin
          PidAudio := PProgram82(Msg.LParam)^.Audio_pid;
          frmMain.cmbAudioPids.Text := format('%4.4d', [PidAudio]);
          Change := True;
        end;
        if PidTeletext <> PProgram82(Msg.LParam)^.Teletext_pid then
        begin
          PidTeletext := PProgram82(Msg.LParam)^.Teletext_pid;
          frmMain.cmbTeletextPids.Text := format('%4.4d', [PidTeletext]);
          Change := True;
        end;
        if PidPCR <> PProgram82(Msg.LParam)^.PCR_pid then
        begin
          PidPCR := PProgram82(Msg.LParam)^.PCR_pid;
          frmMain.mskPcr.Text := format('%4.4d', [PidPCR]);
          Change := True;
        end;
        if PidPMT <> PProgram82(Msg.LParam)^.PMT_pid then
        begin
          PidPMT := PProgram82(Msg.LParam)^.PMT_pid;
          frmMain.mskPmt.Text := format('%4.4d', [PidPMT]);
          Change := True;
        end;
        if PidEcm <> PProgram82(Msg.LParam)^.ECM_pid then
        begin
          PidEcm := PProgram82(Msg.LParam)^.ECM_pid;
          frmMain.cmbEcmPids.Text := format('%4.4d', [PidEcm]);
          Change := True;
        end;
        if Change then
          SendMessage(frmMain.Handle, WM_REMOTE,
            Integer(MessageSettings), 0);
```

```
        end;
      end;
    MDAPI_GET_PROGRAM_NUMBER:
      begin
        PProgramNumberParam(Msg.LParam)^.RealNumber := ServiceNumber;
        PProgramNumberParam(Msg.LParam)^.VirtNumber := ServiceNumber;
      end;
    MDAPI_START_FILTER:
      begin
        StartFilterParam := PStartFilterParam(Msg.LParam);
        if (StartFilterParam^.DLLIdentifier >= MdPlugins) then
          Exit;
        StartFilterParam^.Running_ID := ((StartFilterParam^.DLLIdentifier and
          $FF) shl 8)
          or DvbSetFilter(@MdPlugin[StartFilterParam^.DLLIdentifier].Filters,
          StartFilterParam^.Pid, StartFilterParam^.Filter_ID,
          Pointer(StartFilterParam^.CallAddress),
          PChar(@(StartFilterParam^.Name[0])));
      end;
    MDAPI_STOP_FILTER:
      begin
        Plugin := (Msg.LParam and $FF00) shr 8;
        if Plugin >= MdPlugins then
          Exit;
        DvbStopFilter(@MdPlugin[Plugin].Filters, Msg.LParam and $FF);
      end;
    MDAPI_GET_VERSION:
      begin
        StrPCopy(PChar(Msg.LParam), 'MD-API Version 01.02 Root 254376');
      end;
    MDAPI_DVB_COMMAND:
      begin
        if PDVB_COMMAND(msg.LParam)^.CmdLength = 7 then
          for Index := 4 to 13 do
            SAA_COMMAND[Index] := PDVB_COMMAND(Msg.LParam)^.CmdBuffer[Index];
        SetCSAKeys(SAA_COMMAND);
      end;
  end;
end;

procedure TfrmMain.WMSignalling(var Msg: TMessage);
begin
  case Msg.WParam of
    CSignalEvent:
      begin
        if (Msg.LParam = ServiceNumber) or (ServiceNumber = 0) then
        begin
          ToLog(format('WMSignalling - Event (Id: %d)', [Msg.LParam]), $5);
          UpdateEvent := True;
        end
        else
          ToLog(format('WMSignalling - Event (Id: %d) - not handled',
            [Msg.LParam]), $5);
```

```
      end;
    CSignalPat:
      begin
        ToLog(format('WMSignalling - PAT (Version: %d)', [Msg.LParam]), $5);
        UpdatePAT := True;
      end;
    CSignalCat:
      begin
        ToLog(format('WMSignalling - CAT (Version: %d)', [Msg.LParam]), $5);
        UpdateCAT := True;
      end;
    CSignalPmt:
      begin
        if (Msg.LParam = ServiceNumber) or (ServiceNumber = 0) then
        begin
          ToLog(format('WMSignalling - PMT (Id: %d)', [Msg.LParam]), $5);
          UpdatePmt := True;
          UpdatePat := True;
        end
        else
          ToLog(format('WMSignalling - PMT (Id: %d) - not handled',
            [Msg.LParam]), $5);
      end;
    CSignalPid: case Msg.LParam of
        CPidNIT:
          begin
            ToLog('WMSignalling - NIT', $5);
            UpdateProgram := True;
          end;
      else
        ToLog(format('WMSignalling - PID: %d', [Msg.LParam]), $5);
      end;
  else
    ToLog(format('WMSignalling - [%d / %d]', [Msg.WParam, Msg.LParam]), $5);
  end;
end;

procedure TDataThread.DescrambleFfDeCsa;
var
  StreamPacket: Word;
  Pid: Word;
  Plugin: Word;
  Processed: Integer;
  ProcessIndex: Integer;
  ProcessIt: Boolean;
begin

  if (not FilteringOff) and
    (not FiltersOff) then
  begin
    if (CsaMethod = cmFfCsa) then
    begin
      ProcessIndex := 0;
```

```
    for StreamPacket := 0 to CDvbPacketVSync - 1 do
    begin
      if DvbGetPid(@FStreamData[StreamPacket], Pid) then
      begin
        ProcessIt := False;
        if MdPlugins > 0 then
          for Plugin := 0 to MdPlugins - 1 do
            if MdPlugin[Plugin].Filters.CrossReference[Pid] <> 0 then
            begin
              ProcessIt := True;
              Break;
            end;
        if not ProcessIt then
          if AppFilters.CrossReference[Pid] <> 0 then
            ProcessIt := True;
        if Processit and ((FStreamData[StreamPacket][3] and $80) <> 0) then
        begin
          if Pid <> 0 then
          begin
            if Pid = PidVideo then
              VideoCheckScrambled := True;
            if Pid = PidAudio then
              AudioCheckScrambled := True;
          end;
        end
        else
          ProcessIt := False;
        if ProcessIt then
        begin
          FFCsaClusters[ProcessIndex].StartBuffer :=
            @FStreamData[StreamPacket][0];
          FFCsaClusters[ProcessIndex].EndBuffer :=
            @FStreamData[StreamPacket][187];
          Inc(ProcessIndex);
        end;
      end;
    end;
    FFCsaClusters[ProcessIndex].StartBuffer := nil;
    FFCsaClusters[ProcessIndex].EndBuffer := nil;
    if KeyFound then
     repeat
       try
         try
           Processed := FFDecryptProc(@FFCsaClusters, FFCsaKeys);
         finally
           Set8087CW(Default8087CW);
         end;
       except
         Processed := 0;
       end;
     until Processed = 0;
  end;
end;
```

```
end;

procedure TfrmMain.ReadSettings;
var
  Default: string;
  Value: Integer;
  Error: Integer;
  Hour: Word;
  Minute: Word;
  Second: Word;
  Msec: Word;
  TheTime: TDateTime;
  DeltaTime: TDateTime;
  CheckTime: TDateTime;
begin
  ToLog('ReadSettings', $5);
  // Get variable settings
  Default := GetParameter('Settings', 'FormTop', '-4');
  Val(Default, Value, Error);
  if Error = 0 then
    Self.Top := Value;
  Default := GetParameter('Settings', 'FormLeft', '-4');
  Val(Default, Value, Error);
  if Error = 0 then
    Self.Left := Value;
  Default := GetParameter('Settings', 'FormWidth', '1032');
  Val(Default, Value, Error);
  if Error = 0 then
    Self.Width := Value;
  Default := GetParameter('Settings', 'FormHeight', '748');
  Val(Default, Value, Error);
  if Error = 0 then
    Self.Height := Value;

  Default := GetParameter('Settings', 'Transponder', '0');
  Val(Default, Transponder, Error);
  Error := -1;
  if ParameterFavourite <> '' then
  begin
    Val(ParameterFavourite, Value, Error);
    cmbLists.ItemIndex := Value;
    ParameterFavourite := '';
  end;
  if Error <> 0 then
  begin
    Default := GetParameter('Settings', 'FavouriteList', '0');
    Val(Default, Value, Error);
    cmbLists.ItemIndex := Value;
  end;
  Default := GetParameter('Settings', 'Service', '0');
  cmbServices.Text := Default;
  Default := LowerCase(GetParameter('Settings', 'TransponderDisplay', 'No'));
  if Default = 'yes' then
```

```
    chkTransponderDisplay.Checked := True
  else
    chkTransponderDisplay.Checked := False;
  chkTransponderDisplayClick(nil);

  Default := LowerCase(GetParameter('Settings', 'AutoECM', 'Yes'));
  if Default = 'yes' then
    chkAutoECM.Checked := True
  else
    chkAutoECM.Checked := False;
  if ParameterShutdown <> '' then
  begin
    try
      ParameterShutdown := ChangeTimeSeparator(ParameterShutdown);
      TheTime := StrToTime(ParameterShutdown) + TimeCorrectionParameters;
      dtShutdown.Time := TheTime;
      ParameterShutdown := '';
      chkShutdown.Checked := True;
      Error := 0;
    except
    end;
  end;
  if Error <> 0 then
  begin
    Default := GetParameter('Settings', 'ShutdownTime', '00' + TimeSeparator +
      '00' + TimeSeparator + '00');
    try
      Default := ChangeTimeSeparator(Default);
      TheTime := StrToTime(Default);
      dtShutdown.Time := TheTime;
    except
      dtShutdown.Time := dtStopRecording.Time + EncodeTime(0, 1, 0, 0);
    end;
  end;

  if ParameterExit <> '' then
  begin
    try
      ParameterExit := ChangeTimeSeparator(ParameterExit);
      TheTime := StrToTime(ParameterExit) + TimeCorrectionParameters;
      dtExit.Time := TheTime;
      dtExit.Enabled := True;
      ParameterExit := '';
    except
    end;
  end;
  ReadRecordings;
end;

procedure TfrmMain.VolumeUpdate(Sender: TObject);
begin
  if not Mixer.VolumeAvailable then
  begin
```

```
    imgVolumeUp.Visible := False;
    imgVolumeDown.Visible := False;
    imgVolumeOn.Visible := False;
    imgVolumeOff.Visible := False;
  end
  else
  begin
    if Mixer.VolumeMute then
    begin
      imgVolumeOn.Visible := True;
      imgVolumeOff.Visible := False;
    end
    else
    begin
      imgVolumeOn.Visible := False;
      imgVolumeOff.Visible := True;
    end;
  end;
end;

procedure TfrmMain.imgVolumeOffClick(Sender: TObject);
var
  Mute: Boolean;
begin
  if not Mixer.VolumeAvailable then
    Exit;
  Mute := False;
  if Sender = nil then
    Mute := Mixer.VolumeMute;
  if Sender = imgVolumeOff then
    Mute := True;
  if Sender = imgVolumeOn then
    Mute := False;
  Mixer.VolumeMute := Mute;
  VolumeUpdate(imgVolumeOff);
end;

procedure TfrmMain.imgVolumeUpClick(Sender: TObject);
begin
  imgVolumeOffClick(imgVolumeOn);
  Mixer.VolumeUp;
end;

procedure TfrmMain.imgVolumeDownClick(Sender: TObject);
begin
  imgVolumeOffClick(imgVolumeOn);
  Mixer.VolumeDown;
end;

procedure TfrmMain.btnPositionerStopClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
```

```
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $60; // Stop positioner movement
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerResetClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $00; // Reset
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerGotoClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $6B; // Goto position xx
  DiSEqCData[3] := spnSatellitePosition.Value; // Position xx
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerStoreClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $6A; // Store position xx
  DiSEqCData[3] := spnSatellitePosition.Value; // Position xx
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData)
```

```
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData);
  if spnSatellitePosition.Value < spnSatellitePosition.MaxValue then
    spnSatellitePosition.Value := spnSatellitePosition.Value + 1;
end;

procedure TfrmMain.btnPositionerGoEastClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $68; // Drive East
  DiSEqCData[3] := $00; // Timeout (no timeout)
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerStepEastClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $68; // Drive East
  DiSEqCData[3] := $FF; // $00 - steps, $FF = smallest
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerSetEastLimitClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $66; // Set East limit
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData);
end;
```

```
procedure TfrmMain.btnPositionerGoWestClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $69; // Drive West
  DiSEqCData[3] := $00; // Timeout (no timeout)
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerStepWestClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $69; // Drive West
  DiSEqCData[3] := $FF; // $00 - steps, $FF = smallest
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 4,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerSetWestLimitClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $67; // Set West limit
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData);
end;

procedure TfrmMain.btnPositionerDisableLimitsClick(Sender: TObject);
var
  DiSEqcData: TDiSEqCData;
begin
  DiSEqCData[0] := 0; // Will be written over
```

```
  DiSEqCData[1] := $30; // Any positioner
  DiSEqCData[2] := $63; // Disable limits
  if UseFlexCop then
    FlexCopDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData)
  else
    Saa7146aDiSEqCCommand(CardHandle, DiSEqCRepeats, CDiSEqCCommand, 3,
      DiSEqCData);
end;

procedure TfrmMain.mnuScrambledClick(Sender: TObject);
begin
  mnuScrambled.Checked := not mnuScrambled.Checked;
  ReadFavourites;
  UpdateSettings(nil);
end;

procedure TfrmMain.mnuDataClick(Sender: TObject);
begin
  mnuData.Checked := not mnuData.Checked;
  ReadFavourites;
  UpdateSettings(nil);
end;

procedure TfrmMain.mnuRadioClick(Sender: TObject);
begin
  mnuRadio.Checked := not mnuRadio.Checked;
  ReadFavourites;
  UpdateSettings(nil);
end;

procedure TfrmMain.mnuTVClick(Sender: TObject);
begin
  mnuTV.Checked := not mnuTV.Checked;
  ReadFavourites;
  UpdateSettings(nil);
end;
procedure TfrmMain.mnuDisabledClick(Sender: TObject);
begin
  mnuDisabled.Checked := not mnuDisabled.Checked;
  ReadFavourites;
  UpdateSettings(nil);
end;
procedure TfrmMain.mnuSetupFilesClick(Sender: TObject);
begin
  if not Assigned(FSetupFiles) then
  begin
    FSetupFiles := TfrmSetupFiles.Create(Application);
    // Set all settings
    SetFilesSetup(FSetupFiles);
  end;
  FSetupFiles.ExitNotify := SetupFilesExit;
  FSetupFiles.Show;
```

```
end;
procedure TfrmMain.SetupFilesExit(Sender: TObject);
begin
  GetAndWriteFilesSetup(FSetupFiles, False);
  if FSetupFiles.SettingChanged then
  begin
    ReadLists;
    ReadFavourites;
  end;
  // Since the form closes itself make sure we know it too
  FSetupFiles := nil;
end;
procedure TfrmMain.mnuSetupDiSEqCClick(Sender: TObject);
begin
  if not Assigned(FSetupDiSEqC) then
  begin
    FSetupDiSEqC := TfrmSetupDiSEqC.Create(Application);
    SetDiSEqCSetup(FSetupDiSEqC);
  end;
  FSetupDiSEqC.ExitNotify := SetupDiSEqCExit;
  FSetupDiSEqC.Show;
end;
procedure TfrmMain.SetupDiSEqCExit(Sender: TObject);
begin
  GetAndWriteDiSEqCSetup(FSetupDiSEqC, False);
  // Since the form closes itself make sure we know it too
  FSetupDiSEqC := nil;
end;
procedure TfrmMain.mnuSetupMiscellaneousClick(Sender: TObject);
begin
  if not Assigned(FSetupMiscellaneous) then
  begin
    FSetupMiscellaneous := TfrmSetupMiscellaneous.Create(Application);
    // Set all settings
    SetMiscellaneousSetup(FSetupMiscellaneous);
  end;
  FSetupMiscellaneous.ExitNotify := SetupMiscellaneousExit;
  FSetupMiscellaneous.Show;
end;
procedure TfrmMain.SetupMiscellaneousExit(Sender: TObject);
begin
  GetAndWriteMiscellaneousSetup(FSetupMiscellaneous, False);
  // Since the form closes itself make sure we know it too
  FSetupMiscellaneous := nil;
end;
procedure TfrmMain.mnuSetupRemoteClick(Sender: TObject);
begin
  if not Assigned(FSetupRemote) then
  begin
    FSetupRemote := TfrmSetupRemote.Create(Application);
    SetRemoteSetup(FSetupRemote);
  end;
  FSetupRemote.ExitNotify := SetupRemoteExit;
```

```
  FSetupRemote.Show;
end;
procedure TfrmMain.SetupRemoteExit(Sender: TObject);
begin
  GetAndWriteRemoteSetup(FSetupRemote, False);
  FSetupRemote := nil;
end;
procedure TfrmMain.imgRecordListClick(Sender: TObject);
var
  ItemsAvailable: Boolean;
begin
  if not Assigned(FRecordings) then
  begin
    FRecordings := TfrmRecording.Create(Application);
    ItemsAvailable := SetRecordings(FRecordings);
  end
  else
    ItemsAvailable := True;
  if not ItemsAvailable then
  begin
    if Assigned(FRecordings) then
      FreeAndNil(FRecordings);
    Exit;
  end;
  FRecordings.ExitNotify := RecordingsExit;
  FRecordings.Show;
end;
procedure TfrmMain.RecordingsExit(Sender: TObject);
begin
  GetAndWriteRecordings(FRecordings, False);
  FRecordings := nil;
end;
end.
```

Файл програми PlataSaa7146aIOControl.pas

```
unit PlataSaa7146aIOControl;

interface
uses
// Розробив - Сільченко Сергій Миколайович, КІ-20МЗ, 2021 рік
  Windows;

type
  Pvoid = Pointer;
  PPhysicalMemoryAddress = ^PhysicalMemoryAddress;
  PhysicalMemoryAddress = LARGE_INTEGER;

  TSaa7146AGetDriverVersion = record
    MajorVersion: Word; // Major version
    MinorVersion: Word; // Minor version
    Build: Dword; // Build (date): $YYYMMDD
  end;

  TSaa7146aGetDmaStatus = record
    Interrupts: Longint; // Number of interrupts counted sofar
    Isr: Dword; // Interrupt status register
    VirtualAddress: Pvoid; // Virtual address of common buffer
    PhysicalAddress: PhysicalMemoryAddress;
    Size: Dword;
    FifoOverflows: Dword;
  end;

  TSaa7146aDmaBuffer = record
    Identifier: Longint; // ID of buffer
    VirtualAddress: Pvoid; // Virtual address of buffer
    PhysicalAddress: PhysicalMemoryAddress; // Physical address of buffer
    Size: Dword; // Size of buffer
  end;

  TSaa7146aTransferBuffer = record
    Identifier: Longint; // ID of buffer
    TransferAddress: Pchar; // Pointer to source/target buffer
    SourceIndex: Dword; // Index into source buffer
    TargetIndex: Dword; // Index into target buffer
    TransferLength: Dword; // Number of bytes to transfer
  end;

  TSaa7146aFifoTransferBuffer = record
    Identifier: Longint; // ID of buffer
    NumberOfBuffers: Dword; // Number of buffers
    TransferAddress: Pchar; // Pointer to source/target buffer
    TransferLength: Dword; // Number of bytes to transfer
    IsValid: Boolean; // Copy of <writeTag> indicating buffer has been written
to
    Overflows: Dword;
```

```
    Irqs: Dword;
    AllOverflows: Dword;
  end;

  TSaa7146aIrqBuffer = record
    Irqs: Dword;
    IrqsWhenActive: Dword;
    IrqBufferingIsActive: Boolean; // Global activation flag of this interrupt
    UseNotificationEvent: Boolean; // Indicates notify event to be triggered
    UseSignaling: Boolean; // Indicates signal to be triggered
    UseFifo: Boolean; // Indicates FIFO to be used
    IrqAutoDisable: Boolean;
    SignalAndValue: Dword;
    SignalOrValue: Dword;
    SignalXorValue: Dword;
    FifoBufferPreviousIndex: Byte;
    FifoBufferFirstIndex: Byte; // First FIFO to use (index FIFO buffer array)
    FifoBufferLastIndex: Byte; // Last  FIFO to use (index FIFO buffer array)
    FifoBufferCirculations: Dword;
    FifoOverflows: Dword;
  end;
  TSaa7146aIrqTransferBuffer = record
    Identifier: Longint; // ID of SAA7146A IRQ bit number (0..31)
    Information: TSaa7146aIrqBuffer; // Information structure
  end;
const
  CSaa7146aDriverName = '\\.\SAA7146A1';
  CIoctlSaa7146aGetDriverVersion = $222000;
  CIoctlSaa7146aGetDmaStatus = $222004;
CIoctlSaa7146aReadFromSaa7146aRegister = $222008; // Read  a SAA7146A register
CIoctlSaa7146aWriteToSaa7146aRegister = $22200C; // Write a SAA7146A register
  CIoctlSaa7146aGenerateManualNotification = $222014;
  CIoctlSaa7146aAllocateDma = $222018;
  CIoctlSaa7146aReleaseDma = $22201C;
  CIoctlSaa7146aReadFromDma = $222020;
  CIoctlSaa7146aWriteToDma = $222024;
  CIoctlSaa7146aAllocateFifo = $222028;
  CIoctlSaa7146aReleaseFifo = $22202C;
  CIoctlSaa7146aReadFromFifo = $222030;
  CIoctlSaa7146aWriteToIrqHandling = $222034;
  CIoctlSaa7146aReadFromIrqHandling = $222038;
implementation
end.
```

Файл програми DiSEqC.pas

```
unit DiSEqC;

interface
uses
// Розробив - Сільченко Сергій Миколайович, КІ-20МЗ, 2021 рік
  Saa7146aI2c,
  PlataCopInterface,
  PlataCopRegisters,
  PlataCopGeneral,
  Sl1935Registers,
  Stv0299bRegisters,
  Windows;

const
  CDiSEqCBurstA  = $01;
  CDiSEqCBurstB  = $02;
  CDiSEqCCommand = $04;

type
  TDiSEqCData = array[0..5] of Byte;

function PlataCopUsDelay(UsDelay : Dword): Boolean;
function PlataCopReadEeprom (Handle:THandle; Address : Word; var Data: array
of Byte): Boolean;
function PlataCopWriteEeprom (Handle : THandle; Address:Word; Data: array of
Byte): Boolean;
function PlataCopEepromChecksum  (Data   : array of Byte): Byte;
function PlataCopCheckQpskResetValues (Handle: THandle): Boolean;
function PlataCopWriteToSynthesizer (Handle: THandle; AddressIndex: Byte;
Data: array of Byte): Boolean;
function PlataCopDetectSynthesizerAddress (Handle: THandle): Byte;
function PlataCopQpskSetAGCInversion(Handle: THandle; Inversion: Boolean):
Boolean;
function PlataCopQpskSetInversion(Handle: THandle; Inversion:
Boolean):Boolean;
function PlataCopQpskGetLockDetectorPercentage(Handle: THandle; var
Percentage: Byte): Boolean;
function PlataCopLowBandLNB(Handle: THandle): Boolean;
function PlataCopHighBandLNB (Handle: THandle): Boolean;
function PlataCopDiSEqCCommand (Handle: THandle; Repeats: Byte; CommandType:
Byte; CommandLength: Byte; CommandData: TDiSEqCData): Boolean;

implementation

uses
  Math,
  SyncObjs,
  SysUtils;
```

```
var
HighPerformanceAvailable: Boolean;
HighPerformanceFrequency: TLargeInteger;
I2cLock:TCriticalSection;


function PlataCopEepromChecksum(Data: array of Byte): Byte;
var
  Loop: Integer;
begin
  Result := 0;
  for Loop := 0 to High(Data) do
    Result := Result xor Data[Loop];
end;

function PlataCopReadFromQpsk(Handle: THandle; AccessRegister: Byte; var Data:
Byte): Boolean;
begin
  I2cLock.Acquire;
  try
    Result := PlataCopI2cReadBytes(Handle, CPlataCopI2cChannelFrontEnd,
              CStv0299bDemodulatorReadAddress, AccessRegister, Data);
  finally
    I2cLock.Release;
  end;
end;

function PlataCopWriteDefaultsToQpsk(Handle: THandle; TunerType: Byte):
Boolean;
var
  Loop: Byte;
begin
  Result := False;
  for Loop := Low(CStv0299bDefaultsTBMU24112) to
High(CStv0299bDefaultsTBMU24112) do
  begin
    if CStv0299bDefaultsTBMU24112[Loop, 0] <> $FF then
if not PlataCopWriteToQpsk(Handle, CStv0299bDefaultsTBMU24112[Loop, 0],
CStv0299bDefaultsTBMU24112[Loop, 1]) then
      Exit;
  end;
    Result := True;
end;

function PlataCopCheckQpskDefaultValues(Handle: THandle; TunerType: Byte):
Boolean;
var
  Loop: Byte;
  Data: Byte;
begin
  Result := False;
  for Loop := Low(CStv0299bDefaultsTBMU24112) to
High(CStv0299bDefaultsTBMU24112) do
```

```
  begin
    if CStv0299bDefaultsTBMU24112[Loop, 0] <> $FF then
    begin
      if not PlataCopReadFromQpsk(Handle, CStv0299bDefaultsTBMU24112[Loop, 0],
Data) then
        Exit;
      // Check it with expected value
      if Data <> CStv0299bDefaultsTBMU24112[Loop, 1] then
        Exit;
    end;
  end;
  Result := True;
end;

function PlataCopCheckQpskResetValues(Handle: THandle): Boolean;
var
  Loop: Byte;
  Data: Byte;
begin
  Result := False;

  for Loop := Low(CStv0299bResets) to High(CStv0299bResets) do
  begin
    if CStv0299bResets[Loop, 0] <> $FF then
    begin
      if not PlataCopReadFromQpsk(Handle, CStv0299bResets[Loop, 0], Data) then
        Exit;
      if Data <> CStv0299bResets[Loop, 1] then
        Exit;
    end;
  end;
  Result := True;
end;

function PlataCopDiSEqCWaitIdle(Handle: THandle): Boolean;
var
  Data : Byte;
  Empty: Boolean;
  Retry: Integer;
begin
  Result := False;
  Retry  := 100;
  repeat
    if not PlataCopReadFromQpsk(Handle, CStv0299bDiSEqCStatus, Data) then
      Exit;
    Empty := ((Data and CStv0299bFifoMask) = CStv0299bFifoEmpty);
    if not Empty then
    begin
      PlataCopUsDelay(1000);
      Dec(Retry);
    end;
  until Empty or (Retry = 0);
  if Retry = 0 then
```

```
    Exit;

  Result := True;
end;

function PlataCopDiSEqCWaitFree(Handle: THandle): Boolean;
var
  Data : Byte;
  Full : Boolean;
  Retry: Integer;
begin
  Result := False;
  Retry  := 20;
  repeat
    if not PlataCopReadFromQpsk(Handle, CStv0299bDiSEqCStatus, Data) then
      Exit;
    Full := ((Data and CStv0299bFifoFull) = CStv0299bFifoFull);
    if Full then
    begin
      PlataCopUsDelay(1000);
      Dec(Retry);
    end;
  until (not Full) or (Retry = 0);
  if Retry = 0 then
    Exit;

  Result := True;
end;

function PlataCopDiSEqCSendByte(Handle: THandle; Data: Byte): Boolean;
begin
  Result := False;

  // Wait for space in fifo
  if not PlataCopDiSEqCWaitFree(Handle) then
    Exit;
  // Write data to fifo
  if not PlataCopWriteToQpsk(Handle, CStv0299bDiSEqCFifo, Data) then
    Exit;

  Result := True;
end;

function PlataCopSendDiSEqCMsg(Handle: THandle; CommandType: Byte;
  CommandLength: Byte; CommandData: TDiSEqCData; Delay: Boolean): Boolean;
var
  Loop: Byte;
begin
  Result := False;

  case (CommandType and (CDiSEqCBurstA or CDiSEqCBurstB or CDiSEqCCommand)) of
    $00:  begin
            Result := True;
```

```
            Exit;
          end;
    CDiSEqCBurstA: ;
    CDiSEqCBurstB: ;
    CDiSEqCCommand: if CommandLength = 0 then
         begin
           Result := True;
           Exit;
         end;
    else Exit;
  end;
  if not PlataCopDiSEqCWaitIdle(Handle) then
    Exit;
  if not PlataCopWriteToQpsk(Handle, CStv0299bDiSEqC,
CStv0299bModulationDiSEqC) then
    Exit;
  if Delay then
    PlataCopUsDelay(16000);

  if ((CommandType and CDiSEqCCommand) = CDiSEqCCommand) then
  begin
    for Loop := 0 to CommandLength-1 do
    begin
      if not PlataCopDiSEqCSendByte(Handle, CommandData[Loop]) then
        Exit;
    end;
    if not PlataCopDiSEqCWaitIdle(Handle) then
      Exit;
  end;
  if ((CommandType and (CDiSEqCBurstA or CDiSEqcBurstB)) <> 0) then
  begin
    if not PlataCopWriteToQpsk(Handle, CStv0299bDiSEqC,
CStv0299bModulationUnmodulated) then
      Exit;
    if (CommandType and CDiSEqCBurstA) <> 0 then
      PlataCopDiseqcSendByte(Handle, $00)
    else
      PlataCopDiseqcSendByte(Handle, $FF);
    if not PlataCopDiSEqCWaitIdle(Handle) then
      Exit;
  end;
  Result := True;
end;
function PlataCopDiSEqCCommand(Handle: THandle; Repeats: Byte; CommandType:
Byte; CommandLength: Byte; CommandData: TDiSEqCData): Boolean;
var
  DiSEqCOriginal: Byte;
  RepeatCommands: Byte;
begin
  Result := False;
  if not PlataCopReadFromQpsk(Handle, CStv0299bDiSEqC, DiSEqCOriginal) then
    Exit;
```

```
  if ((CommandType and CDiSEqCCommand) = CDiSEqCCommand) and (CommandLength >
0) then
  begin
    CommandData[0] := $E0;
    if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCCommand, CommandLength,
CommandData, True) then
      Exit;
    RepeatCommands := Repeats;
    while RepeatCommands > 0 do
    begin
      PlataCopUsDelay(100000);
      CommandData[0] := $E1;
if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCCommand, CommandLength,
CommandData, True) then
        Exit;
      Dec(RepeatCommands);
    end;
  end;
  if (CommandType and (CDiSEqCBurstA or CDiSEqCBurstB)) <> 0 then
  begin
    if (CommandType and CDiSEqCBurstA) <> 0 then
    begin
      if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCBurstA, 0, CommandData,
True) then
        Exit;
    end
    else
      if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCBurstB, 0, CommandData,
True) then
        Exit;
  end;

  // Restore 22kHz tone, we need a minimum of 15 ms delay first
  PlataCopUsDelay(16000);
if (DiSEqCOriginal and not CStv0299bModulationModeMask) =
CStv0299bModulationContinuous then
    PlataCopHighBandLNB(Handle)
  else
    PlataCopLowBandLNB(Handle);

  Result := True;
end;

function PlataCopDiSEqCCommand2(Handle: THandle; CommandType: Byte;
  CommandLength: Byte; CommandData: TDiSEqCData): Boolean;
var
  DiSEqCOriginal: Byte;
begin
  Result := False;

  if not PlataCopReadFromQpsk(Handle, CStv0299bDiSEqC, DiSEqCOriginal) then
    Exit;
```

```
  if ((CommandType and CDiSEqCCommand) = CDiSEqCCommand) and (CommandLength >
0)
    then
    if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCCommand, CommandLength,
      CommandData, True) then
      Exit;
  if (CommandType and (CDiSEqCBurstA or CDiSEqCBurstB)) <> 0 then
  begin
if (CommandType and CDiSEqCBurstA) <> 0 then
    begin
if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCBurstA, 0, CommandData, True)
        then
        Exit;
    end
    else if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCBurstB, 0, CommandData,
      True) then
      Exit;
  end;

  PlataCopUsDelay(16000);
  if (DiSEqCOriginal and CStv0299bModulationModeMask) =
    CStv0299bModulationContinuous then
    PlataCopHighBandLNB(Handle)
  else
    PlataCopLowBandLNB(Handle);

  Result := True;
end;

function ExtractSpacedPart(var Extracted: ShortString): ShortString;
var
  SpacePos: Integer;
begin
  Result := '';
  Extracted := Trim(Extracted);
  SpacePos := Pos(' ', Extracted);
  if SpacePos <> 0 then
  begin
    Result := Copy(Extracted, 1, SpacePos - 1);
    Delete(Extracted, 1, SpacePos);
  end
  else
  begin
    Result := Extracted;
    Extracted := '';
  end;
end;

function ExtractFromDiSEqCSetting(
  Setting: ShortString;
  var Satellite: ShortString;
  var SLOF: Integer;
  var Polarity: Char;
```

```
  var LOF: Integer;
  var DiSEqC: ShortString): Boolean;
var
  ProcessString: ShortString;
  Error: Integer;
begin
  Result := True;
  Satellite := '';
  SLOF := 0;
  Polarity := ' ';
  LOF := 0;
  DiSEqC := '';

  Satellite := ExtractSpacedPart(Setting); // Satellite
  ProcessString := ExtractSpacedPart(Setting); // SLOF
  Val(ProcessString, SLOF, Error);
  if (ProcessString = '') or (Error <> 0) then
  begin
    SLOF := 0;
    Result := False;
  end;
  ProcessString := ExtractSpacedPart(Setting); // Polarity
  if Length(ProcessString) <> 1 then
  begin
    Polarity := ' ';
    Result := False;
  end
  else
  begin
    Polarity := ProcessString[1];
    if not ((Polarity) in ['V', 'v', 'H', 'h']) then
      Result := False;
  end;
  ProcessString := ExtractSpacedPart(Setting); // LOF
  Val(ProcessString, LOF, Error);
  if (ProcessString = '') or (Error <> 0) then
  begin
    LOF := 0;
    Result := False;
  end;
  DiSEqC := Setting; // DiSEqC
end;

function CreateDiSEqCSetting(
  Satellite: ShortString;
  SLOF: Integer;
  Polarity: Char;
  LOF: Integer;
  DiSEqC: ShortString): ShortString;
begin
  Result := format('%s %5.5d %s %5.5d %s', [Satellite, SLOF, Polarity, LOF,
    DiSEqC]);
end;
```

```
function PlataCopDiSEqCCommandVdr(Handle: THandle; CommandData: ShortString):
  Boolean;
var
  Satellite: ShortString;
  Slof: Integer;
  Polarity: Char;
  Lof: Integer;
  DiSEqCCommand: ShortString;
  ProcessString: ShortString;
  Error: Integer;
  Value: Integer;
  DiSEqCData: TDiSEqCData;
  DiSEqCIndex: Integer;
  EndDetected: Boolean;
begin
  Result := True;
if ExtractFromDiSEqCSetting(CommandData, Satellite, Slof, Polarity, Lof,
DiSEqCCommand) then
while (DiSEqCCommand <> '') and (Result = True) do
begin
ProcessString := ExtractSpacedPart(DiSEqCCommand);
if ProcessString <> '' then
begin
case Ord(ProcessString[1]) of
Ord('t'):
begin
if not PlataCopLowBandLNB(Handle) then
begin
Result := False;
Exit;
end;
end;
Ord('T'):
begin
if not PlataCopHighBandLNB(Handle) then
begin
Result := False;
Exit;
end;
end;
Ord('v'):
begin
if not PlataCopVerticalPolarityLNB(Handle) then
begin
Result := False;
Exit;
end;
end;
Ord('V'):
begin
if not PlataCopHorizontalPolarityLNB(Handle) then
begin
```

```
Result := False;
Exit;
end;
end;
Ord('A'), Ord('a'):
begin
// Burst A
if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCBurstA, 0, DiSEqCData,
False) then
begin
Result := False;
Exit;
end;
end;
Ord('B'), Ord('b'):
begin
// Burst B
if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCBurstB, 0, DiSEqCData,
False) then
begin
Result := False;
Exit;
end;
end;
Ord('W'), Ord('w'):
begin
// Wait
Delete(ProcessString, 1, 1);
Val(ProcessString, Value, Error);
if (Error = 0) and (Value > 0) then
PlataCopUsDelay(Value * 1000);
end;
Ord('['):
begin
// DiSEqC command sequence
Delete(ProcessString, 1, 1);
// In case a '[' without trailing data was used read next data
if ProcessString = '' then
ProcessString := ExtractSpacedPart(DiSEqCCommand);
DiSEqCIndex := Low(DiSEqCData);
EndDetected := False;
repeat
if DiSEqCIndex > High(DiSEqCData) then
begin
Result := False;
Exit;
end;
if Length(ProcessString) <> 2 then
begin
Result := False;
Exit;
end;
// Convert as hexadecimal
```

```
Val('$' + ProcessString, DiSEqCData[DiSEqCIndex], Error);
// Check on conversion error
if Error <> 0 then
begin
Result := False;
Exit;
end;
Inc(DiSEqCIndex);
if not EndDetected then
begin
ProcessString := ExtractSpacedPart(DiSEqCCommand);
if (Length(ProcessString) = 1) and (ProcessString[1] = ']')
then
begin
// If we are dealing with a separate ']' we are done too
ProcessString := '';
end;
if Length(ProcessString) > 2 then
begin
if ProcessString[3] <> ']' then
begin
Result := False;
Exit;
end;
Delete(ProcessString, 3, 1);
EndDetected := True;
end;
end
else
ProcessString := '';
until (ProcessString = '');
// Send DiSEqCData
if not PlataCopSendDiSEqCMsg(Handle, CDiSEqCCommand, DiSEqCIndex,
DiSEqCData, False) then
begin
Result := False;
Exit;
end;
end;
else
Result := False;
end;
end;
end
else
Result := False;
end;


function PlataCopReadFromSynthesizer(Handle: THandle; AddressIndex: Byte; var
Data: Byte): Boolean;
var
  AByte    : Byte;
```

```
begin
  Result := False;
  if AddressIndex > High(CSl1935SynthesizerWriteAddress) then
    Exit;

  I2cLock.Acquire;
  try
      Exit;
    Result := PlataCopI2cReadBytes(Handle, CPlataCopI2cChannelFrontEnd,
              CSl1935SynthesizerReadAddress[AddressIndex], 0, Data);
  finally
    I2cLock.Release;
  end;
end;

function PlataCopDetectSynthesizerAddress(Handle: THandle): Byte;
var
  Address     : Byte;
  Success     : Boolean;
ReturnedByte: Byte;
begin
  Address := 0;
  repeat
    Success := PlataCopReadFromSynthesizer(Handle, Address, ReturnedByte);
    if not Success then
      Inc(Address);
  until Success or (Address > 3);
  if Success then
    Result := Address
  else
    Result := 255;
end;

function PlataCopQpskSetSymbolRate(Handle: THandle; SymbolRateKS: Dword):
Boolean;
var
  Data: Dword;
  AClc: Byte;
  BClc: Byte;
begin
  Result := False;

  if (SymbolRateKS * 1000) > CStv0299bMasterClock then
    Exit;

  AClc := $04;
  BClc := $11;
  if SymbolRateKS < 30000 then
  begin
    AClc := $06;
    BClc := $13;
  end;
  if SymbolRateKS < 14000 then
```

```
  begin
    AClc := $07;
    BClc := $13;
  end;
  if SymbolRateKS < 7000 then
  begin
    AClc := $07;
    BClc := $0F;
  end;
  if SymbolRateKS < 3000 then
  begin
    AClc := $07;
    BClc := $0B;
  end;
  if SymbolRateKS < 1500 then
  begin
    AClc := $07;
    BClc := $07;
  end;
  AClc := AClc or $B0;
  BClc := BClc or $40;
  if not PlataCopWriteToQpsk(Handle, CStv0299bAclc, AClc) then
    Exit;
  if not PlataCopWriteToQpsk(Handle, CStv0299bBclc, BClc) then
    Exit;

  Data := Round((SymbolRateKS * 1000) / CStv0299bSetSymbolrateUnits);
  Data := Data shl 4;
  if not PlataCopWriteToQpsk(Handle, CStv0299bSfrH, (Data shr 16) and $FF)
then
    Exit;
  if not PlataCopWriteToQpsk(Handle, CStv0299bSfrM, (Data shr 8) and $FF) then
    Exit;
  if not PlataCopWriteToQpsk(Handle, CStv0299bSfrL, Data and $F0) then
    Exit;

  Result := True;
end;

function PlataCopQpskGetLockDetectorPercentage(Handle: THandle; var
Percentage: Byte): Boolean;
var
  Data: Byte;
  Temp: Integer;
begin
  Result := False;
  if not PlataCopReadFromQpsk(Handle, CStv0299bCldi, Data) then
    Exit;
  if (Data and $80) <> 0 then
    Temp := Data - 256
  else
    Temp := Data;
  Temp := Temp + 128;
```

```
  Temp := (Temp * 100) div 255;
  Percentage := Temp;
  Result := True;
end;

function PlataCopQpskGetSignalPower(Handle: THandle;
  var StrengthPercentage: Double;
  var StrengthDecibel   : Double): Boolean;
var
  DataMSB    : Byte;
  DataLSB    : Byte;
  Data       : Integer;
begin
  Result := False;

  if not PlataCopReadFromQpsk(Handle, CStv0299bAgc2I1, DataMSB) then
    Exit;
  if not PlataCopReadFromQpsk(Handle, CStv0299bAgc2I2, DataLSB) then
    Exit;
  Data := (DataMSB shl 8) or DataLSB;
  StrengthPercentage := 100 - ((Data * 100) / $FFFF);
  if StrengthPercentage <> 0 then
    StrengthDecibel := (Log10(4) * 20) + Log10(StrengthPercentage/100) * 20
  else
    StrengthDecibel := 0;
    y
  Result := True;
end;

function PlataCopQpskGetInputLevel(Handle: THandle; var InputLevelDbm:
Integer;
  var InputLevelPercentage: Byte): Boolean;
var
  Data: Byte;
  NewData: Integer;
  Calc: Integer;
begin
  Result := False;
  // Read values
  if not PlataCopReadFromQpsk(Handle, CStv0299bAgc1I, Data) then
    Exit;
  if Data > 128 then
    NewData := Data - 256
  else
    NewData := Data;
  NewData := NewData - 50;
  if NewData > 0 then
    NewData := 0;
  Calc := 100 - Abs((NewData * 2) div 3);
  if Calc < 0 then
    Calc := 0;
  InputLevelPercentage := Calc;
  NewData := NewData div 3;
```

```
  NewData := NewData - 20;
  InputLevelDbm := NewData;
  Result := True;
end;
function PlataCopQpskGetNoiseIndicator(Handle: THandle; var Noise: Double):
Boolean;
var
  DataMSB: Byte;
  DataLSB: Byte;
  Data   : Word;
begin
  Result := False;
  if not PlataCopReadFromQpsk(Handle, CStv0299bNirH, DataMSB) then
    Exit;
  if not PlataCopReadFromQpsk(Handle, CStv0299bNirL, DataLSB) then
    Exit;
  Data := (DataMSB shl 8) or DataLSB;
  Noise := (-0.0017 * Data) + 19.02;
  Result := True;
end;

function PlataCopQpskSetAGCInversion(Handle: THandle; Inversion: Boolean):
Boolean;
var
  OriginalData: Byte;
  Data        : Byte;
begin
  Result := False;
  if not PlataCopReadFromQpsk(Handle, CStv0299bAgc1R, OriginalData) then
    Exit;
  Data := OriginalData and not CStv0299bAGCInversion;
  if Inversion then
    Data := Data or CStv0299bAGCInversion;
  if Data <> OriginalData then
    if not PlataCopWriteToQpsk(Handle, CStv0299bAgc1R, Data) then
      Exit;
  Result := True;
end;

function PlataCopQpskSetInversion(Handle: THandle; Inversion: Boolean):
Boolean;
var
  OriginalData: Byte;
  Data        : Byte;
begin
  Result := False;

  if not PlataCopReadFromQpsk(Handle, CStv0299bIoCfg, OriginalData) then
    Exit;
  Data := OriginalData and not CStv0299bInversion;
  if Inversion then
    Data := Data or CStv0299bInversion;
  if Data <> OriginalData then
```

```
    if not PlataCopWriteToQpsk(Handle, CStv0299bIoCfg, Data) then
      Exit;

  Result := True;
end;

function PlataCopQpskGetCarrierDeviation(Handle: THandle; var DeviationKHz:
Integer): Boolean;
var
  DataM: Byte;
  DataL: Byte;
  Temp: Int64;
begin
  Result := False;
  if not PlataCopReadFromQpsk(Handle, CStv0299bCfrM, DataM) then
    Exit;
  if not PlataCopReadFromQpsk(Handle, CStv0299bCfrL, DataL) then
    Exit;
  Temp := (DataM shl 8) or DataL;
  if (DataM and $80) <> 0 then
    Temp := Temp - 65536;
  Temp := Temp * (CStv0299bMasterClock div 1000);
  DeviationKHz := (Temp div 65536);
  Result := True;
end;


function PlataCopLowBandLNB(Handle: THandle): Boolean;
var
  Data: Byte;
begin
  Result := False;
  if not PlataCopReadFromQpsk(Handle, CStv0299bDiSEqC, Data) then
    Exit;
  Data := Data and not CStv0299bModulationMask;
  Data := Data or CStv0299bModulation0;
  if not PlataCopWriteToQpsk(Handle, CStv0299bDiSEqC, Data) then
    Exit;
  Result := True;
end;
end.
```