МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Центральноукраїнський національний технічний університет

Кафедра кібербезпеки та програмного забезпечення

На правах рукопису

Малуха Сергій Михайлович

Програмне забезпечення системи інформаційної кабельної мережі автомобіля

Спеціальність: 123 «Комп'ютерна інженерія»

Освітній ступінь: бакалавр

Науковий керівник:

Петренюк Володимир Ілліч _____

(підпис) (дата)

кандидат фізико-математичних наук, доцент

ДОПУЩЕНО ДО ЗАХИСТУ

Завідувач кафедри

_____ О.А. Смірнов

(підпис) ПІБ

«____»_____2021 р.

Кропивницький – 2021

Міністерство освіти і науки України
Центральноукраїнський національний технічний університет

Факультет *Механіко-технологічний*
Кафедра *Кібербезпеки та програмного забезпечення*
Освітній ступінь *бакалавр*
Спеціальність *123 Комп'ютерна інженерія*

ЗАТВЕРДЖУЮ
Завідувач кафедри
д.т.н., проф.
_____О.А.Смірнов
« 21 » грудня 2020 року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Малусі Сергію Михайловичу
(прізвище, ім'я, по батькові)

1. Тема роботи *Програмне забезпечення системи інформаційної кабельної мережі автомобіля*

керівник роботи *Петренюк Володимир Ілліч, канд. фіз.-мат. наук, доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом вищого навчального закладу № 189-02 від 28.12.2020 року

2. Строк подання студентом роботи до захисту *22.05.2021 р.*

3. Мета та завдання кваліфікаційної бакалаврської роботи*: Метою розробки є програмне забезпечення системи інформаційної кабельної мережі автомобіля*

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

1. Призначення та область використання.
2.Перегляд аналогічних існуючих систем.
3. Опис і обґрунтування проектних рішень.
4. Етапи програмування системи.
5. Впровадження системи в промислову експлуатацію.
6. Висновки

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

Структурна схема системи	*1 аркуш*
Функціональна схема системи	*1 аркуш*
Діаграма процесів	*1 аркуш*
Блок-схема алгоритму роботи додатку	*2 аркуша*

6. Дата видачі завдання « 21 » грудня 2020 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної бакалаврської роботи	Строк виконання етапів кваліфікаційної бакалаврської роботи	Примітка
1.	Аналіз існуючих систем	10.03.2021 р.	
2.	Постановка задачі, оформлення ТЗ	15.03.2021 р.	
3.	Розробка моделі компонента	20.03.2021 р.	
4.	Розробка структур даних	25.03.2021 р.	
5.	Розробка алгоритмів зв’язку та відображення	30.03.2021 р.	
6.	Програмування алгоритмів	10.04.2021 р.	
7.	Оформлення ПЗ	17.04.2021 р.	
8.	Попередній захист роботи	22.05.2021 р.	

Студент _____ _____
(підпис) (прізвище та ініціали)

Керівник роботи _____ _____
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Малуха С.М. Програмне забезпечення системи інформаційної кабельної мережі автомобіля. 123 Комп'ютерна інженерія. Центральноукраїнський національний технічний університет. Кропивницький. 2021.

У даній кваліфікаційній бакалаврській роботі розроблено програмне забезпечення, яке призначено для системи інформаційної кабельної мережі автомобіля.

Метою роботи є створення системи інформаційної кабельної мережі автомобіля.

Результат роботи – програмна реалізація системи інформаційної кабельної мережі автомобіля.

В процесі роботи над реалізацією системи виконано дослідження існуючих методів, алгоритмів та програмних засобів. Розроблено та реалізовано власне програмне забезпечення, здійснено опис всіх його компонентів.

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

Програма може використовуватися на ПЕОМ архітектури IBM PC з ОС Windows 10.

Програму розроблено в середовищі Delphi.

Ключові слова: комп'ютерна інженерія, комп'ютерна діагностика автомобіля, комп'ютерна система

ABSTRACT

Malukha S.M. Software of the system of information cable network of a car. 123 Computer Engineering. Central Ukrainian National Technical University. Kropyvnytskyi 2021

In this bachelor qualification work, software that is designed for system of information cable network of a car has been developed.

The purpose of the development is the software of the system of information cable network of a car.

The result of the work is the software implementation of the system of information cable network of a car.

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

User-friendly interface is developed. These are instructions for working with software.

The program can be used on an IBM PC running Windows 10.

The program was developed in the Delphi environment.

Keywords: computer engineering, computer diagnostics of the car, computer system

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ І ТЕРМІНІВ

ВМ	–	виконавчі механізмами двигуна
ЕБУ	–	електронний блок управління
OBDII	–	протокол читання даних з автомобіля
USB	–	універсальна серійна шина

ВСТУП

Актуальність теми. Будь-який сучасний автомобіль, оснащений інжекторним двигуном внутрішнього згорання, має «бортовий комп'ютер» для управління й контролю параметрів роботи двигуна. Більш правильно замість «бортовий комп'ютер» застосувати іншу назву – контролер управління двигуном, тому що даний пристрій звичайно виконаний на 8-мі бітному мікроконтролері, або найбільше що часто зустрічається в технічній літературі термін – електронний блок управління двигуном (ЕБУ) або інформаційна кабельна система автомобіля. Основне завдання блоку управління двигуном автомобіля – на основі показань датчиків формування стехіометричної паливної суміші й своєчасний підпал останньої. Стехіометрична паливна суміш (при якій паливо згоряє повністю) – співвідношення кількості палива до кількості повітря від 12 до 16 залежно від навантаження двигуна. Програма управління двигуном міститься в програмному запам'ятовуючому пристрої. Крім того, можливо в ЕБУ наявність Flash пам'яті, що разом з високошвидкісним мікроконтролером C509 і внутрисхемною програмою завантажником дозволяє оперативно змінити прошивку й програму управління двигуном. Власне програма управління побудована на принципі табличної конвертації. Набагато простіше й швидше виконати кілька програмних переходів, ніж робити якісь обчислення. Тому для зміни режимів роботи двигуна не потрібно міняти програму цілком, досить змінити частину пошивки – дані калібрувань.

Робота з бортовим комп'ютером забезпечується за рахунок використання інтерфейсу обміну даними за стандартом ISO 14230 (ISO9141, OBD-II, KWP2000). Таким чином, виходячи з вищеперерахованого, розробка програмного забезпечення діагностування електронної системи автомобіля за стандартом ISO 14230, є актуальною задачею, яка потребує вирішення у даній кваліфікаційній бакалаврській роботі.

Мета й завдання дослідження. Метою роботи є програмне забезпечення системи інформаційної кабельної мережі автомобіля.

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

– Огляд існуючих систем інформаційної кабельної мережі автомобіля.

– Дослідження системи інформаційної кабельної мережі автомобіля.

– Програмна реалізація системи інформаційної кабельної мережі автомобіля.

Практична цінність отриманих результатів полягає в тому, що розроблені алгоритми дозволяють успішно вирішувати задачі інформаційної кабельної мережі автомобіля.

Таким чином, виходячи з вищеперерахованого, програмне забезпечення системи інформаційної кабельної мережі автомобіля, є актуальною задачею, яка потребує вирішення у даній кваліфікаційній бакалаврській роботі.

1 ПРИЗНАЧЕННЯ ТА ОБЛАСТЬ ВИКОРИСТАННЯ

1.1 Призначення системи

Система, яка розроблена, у результаті виконання даної кваліфікаційної бакалаврської роботи, призначена для діагностування електронної системи автомобіля за стандартом ISO 14230. При всьому різноманітті більшість автомобільних мікропроцесорних систем, управління побудовано по єдиному принципі. Архітектурно цей принцип такий: датчики стану – командний комп'ютер ECU – виконавчі механізми зміни стану.

Основними функціями ЕБУ є:

– управління і контроль за уприскуванням палива в інжекторних двигунах;

– контроль за запалюванням;

– управління фазами газорозподілу;

– регулювання і підтримання температури в системі охолодження двигуна;

– контроль за становищем дросельної заслінки;

– аналіз складу вихлопних газів;

– контроль за роботою системи рециркуляції відпрацьованих газів.

Крім того на контролер надходить інформація про стан і частоті обертання колінчастого валу, поточної швидкості руху транспортного засобу, про напругу в бортовій мережі автомобіля. Також ЕБУ оснащений системою діагностики і в разі виявлення будь-яких неполадок або збоїв інформує про них власника за допомогою кнопки Check-Engine.

Кожна помилка має свій код і ці коди зберігаються на пристрої зберігання даних.

При проведенні діагностики фахівці підключають до контролера через роз'єм скануючий пристрій, на екран якого виводяться всі коди помилок, а також інформація про стан двигуна.

Командний комп’ютер автомобіля збирає інформацію з наступних датчиків:

– температура двигуна.

– температура навколишнього середовища.

– подача кисню.

– подача палива.

– холостого ходу.

– датчики антиблокувальної системи, стабілізації, антизаносу, і інших систем безпеки коліс.

– швидкість.

– положення дросельної заслінки.

– положення педалі газу.

– колінчастого валу (іноді їх два).

– контроль охолоджуючої рідини.

– датчик кондиціонера і його системи.

– контроль, за гальмівною системою і її рідиною.

– дані з бачка гідропідсилювача або з ланцюга ЕУРа.

– аналіз напруги бортової мережі живлення.

Це стандартний набір, який зустрічається практично на будь-якому авто, якщо ваш автомобіль більш складний, то кількість датчиків буде більшою, можуть додатися датчики підвіски, наприклад, на деяких позашляховиках вона пневматична.

Командний комп’ютер автомобіля дає накази, тобто відсилає команди, наступним блокам автомобіля:

– положення дросельної заслінки та подачі повітря.

– подача палива (впорскування з інжекторів).

– система запалювання.

– управління фазами газорозподілу.

– управління температурою і її підтримкою.

– аналіз вихлопних газів, до і після каталізатора.

– система кондиціонування.

– система освітлення, і іншим електрообладнанням, іноді навіть магнітоли.

– склопідйомники.

– підігрів.

– управління автоматичною коробкою передач.

Знову таки це мінімальний стандартний набір, таких команд може бути куди більше в залежності від класу автомобіля.

Пам'ять ЕБУ поділяється на декілька частин:

– ППЗУ – програмований постійний запам'ятовуючий пристрій – сюди закладають необхідні програми і функції роботи мотора;

– ОЗУ – оперативний запам'ятовуючий пристрій. Служить для роботи з проміжними даними, які обробляються «тут і зараз».

– ЕРПЗУ – електронний репрограмуємий пристрій – потрібний для зберігання тимчасової інформації – наприклад, коди доступу, коди блокування, час роботи різних агрегатів, пробіг, витрата палива, температура двигуна і т.д.

Програмне забезпечення ЕБУ:

– Функціональне – найважливіше, отримує інформацію з датчиків, аналізує її і відправляє виконавцям.

– Контрольні мікросхеми (модулі) – контролюють отримані дані на помилки, якщо раптом вони виявлені, намагаються їх виправити, якщо не виходить – видають помилку, або може повністю блокувати запуск двигуна.

Програмне забезпечення ЕБУ піддається корекції. Блок управління двигуном можна перепрошити, тим самим замінивши штатну програму і внісши

зміни в базові настройки і параметри роботи силового агрегату. Даний спосіб отримав назву чіп-тюнінг бензинового або дизельного двигуна.

1.2 Область застосування

Областю застосування розроблювальної системи є різноманітні станції технічного обслуговування (СТО) або діагностичні центри автомобілів, на яких можливо проводити діагностику автомобіля використовуючи інтерфейс ISO14230.

Технічна діагностика автомобіля це сукупність цілей і завдань, пов'язаних з пошуком несправностей механізмів і систем автомобіля, для їх подальшого усунення. Для проведення робіт з діагностування автомобіля створюються спеціальні ділянки діагностики автомобіля.

Технічне діагностування забезпечує значну економію коштів на утримання автомобілів за рахунок скорочення їх простою під час обслуговування і ремонту, виконання дійсно необхідних регулювальних і ремонтних операцій, скорочення витрат запасних частин і палива. Це досягається шляхом своєчасного виявлення і усунення незначних несправностей в системах запалювання, живлення, а також в агрегатах трансмісії.

Ефективне використання засобів діагностування на станціях обслуговування автомобілів і автотранспортних підприємствах можливо лише в результаті правильного їх застосування і експлуатації. Тому вивчення та використання діагностичного обладнання, а також методів прогнозування залишкового ресурсу автомобіля у наш час приділяється особлива увага.

Комп'ютерна діагностика автомобіля дозволяє швидко і точно визначити які несправності в ньому виникли, і дати точну оцінку його технічного стану.

Процес діагностування включає тестування всіх основних параметрів і характеристик систем автомобіля, що впливають на його роботу, зокрема, блоку управління двигуна, автоматичної трансмісії, пневмопідвіски, система АБС,

системи безпеки, круїз контролю тощо.

Сучасні автомобілі отримали настільки великий перелік функцій і інструментів, що їх вже з легкістю можна прирівняти до роботів. За виконання поставлених перед ним завдань відповідають безліч систем і датчиків, за якими повинен здійснюватися контроль.

Якби на сучасних автомобілях не використовувалася комп'ютерна діагностика, довелося б шукати несправності вручну по ланцюжку, а це і довго, і дорого, враховуючи, що в сучасному автомобілі тисячі деталей.

Таким чином, виходячи з вищеперерахованого, розробка програмного забезпечення системи інформаційної кабельної мережі автомобіля для діагностування електронної системи автомобіля за стандартом ISO 14230, є актуальною задачею, яка потребує вирішення у дан ій кваліфікаційній бакалаврській роботі.

2 ПЕРЕГЛЯД АНАЛОГІЧНИХ ІСНУЮЧИХ СИСТЕМ

2.1 Огляд існуючих систем, технологій, архітектур, програмних рішень за профілем теми кваліфікаційної бакалаврської роботи

Було проведено дослідження існуючих програм діагностики автомобіля з використанням його інформаційної кабельної мережі, результати якого наведені нижче.

ELM327

ELM327 – новітня розробка OBDII сканера, що використовується для діагностики автомобілів за допомогою персонального комп'ютера. Підтримує всі відомі протоколи ODB2 і сполучимий з безліччю діагностичних програм. Інтерфейс зв'язку з персональним комп'ютером – USB.

Сканери OBD-2, побудовані на базі мікроконтролера ELM327 добре відомі закордонним користувачам і завоювали собі прекрасну репутацію завдяки своїй універсальності й надійності. Завдяки цим якостям багато розроблювачів діагностичного програмного забезпечення випускають свої програми для сканерів цього типу. В інтернеті розміщена безліч програм, що володіють різними функціональними можливостями, інтерфейсами і язиковою підтримкою. Серед них є як комерційні продукти, так і безкоштовні версії.

Програмне забезпечення OBD-II для ELM327 здебільшо безкоштовне, воно дозволяє за допомогою цього адаптера діагностувати автомобілі, що підтримують один з діагностичний протоколів ODBII.

Не буде перебільшенням сказати, що серед всіх OBD-II сканерів ELM327 є чемпіоном у галузі установки з'єднання з ЕБУ автомобіля. Там, де інші сканери відмовляються працювати, ELM327 прекрасно справляється із завданням.

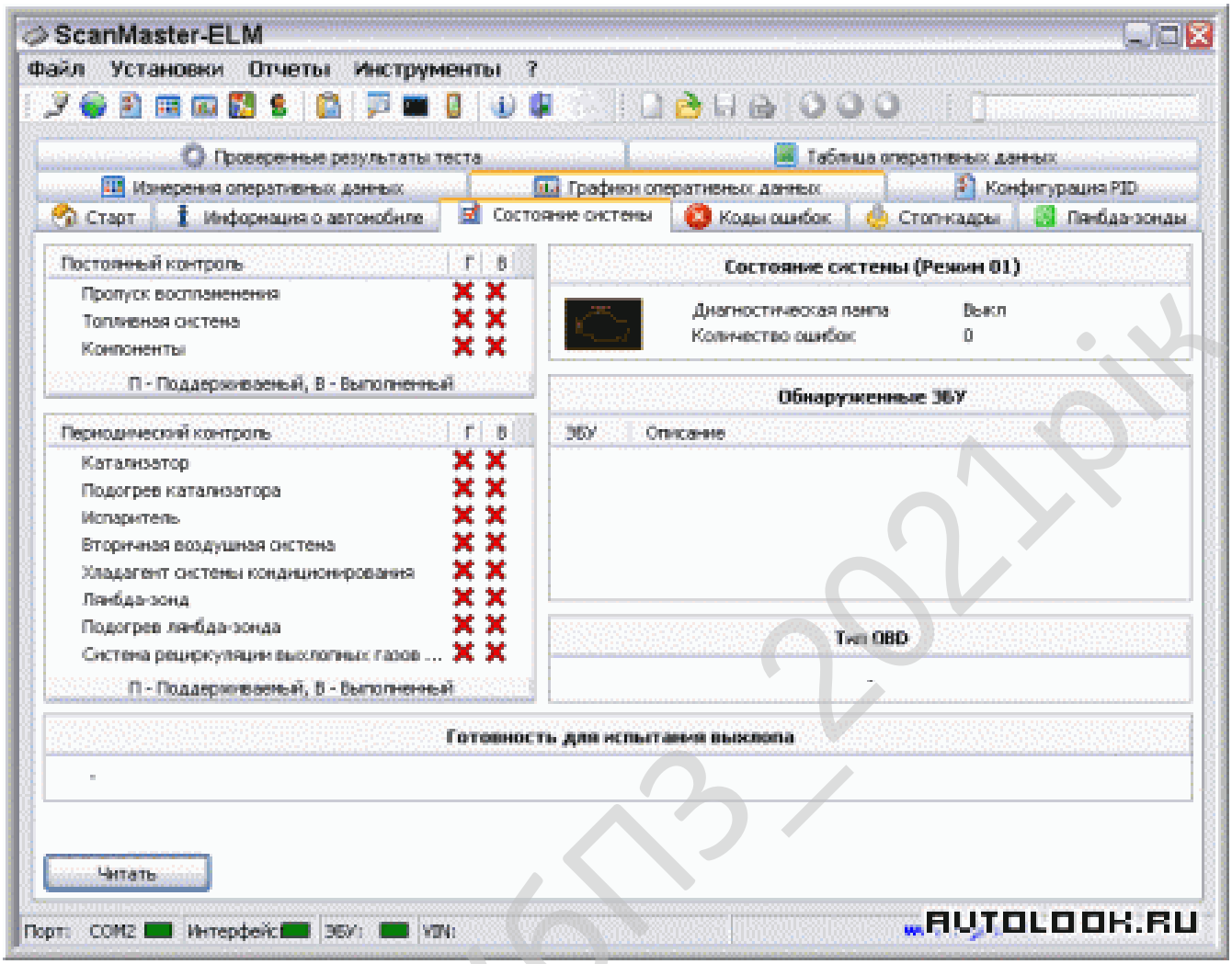

Рисунок 2.1 – Інтерфейс користувача ELM327

Мотор тестер 1.2.0.9

Програма ‘‘Мотор-тестер’’ призначен для діагностики двигунів внутрішнього згоряння автомобілів, оснащених системами електронного управління упорскуванням палива. Програма використовується для проведення технічного обслуговування й ремонту автомобілів на станціях технічного обслуговування, автосервісу, власником автомобіля при наявності комп’ютера типу IBM PC.

Програма ‘‘Мотор-тестер’’ зчитує й обробляє дані з електронного блоку управління (ЕБУ) автомобіля через поставляється адаптер, що, забезпечує

можливість зберігати, переглядати й роздруковувати отриману інформацію, а також управляти виконавчими механізмами двигуна (ВМ).

Рисунок 2.2 – Інтерфейс користувача ‘‘Мотор-тестер’’

Програма дозволяє:

1. Відображати в динаміку всі контрольовані параметри ЕБУ, переглядати як цифровому, так і в графічному виді до 7 параметрів одночасно.

2. Управляти виконавчими механізмами двигуна в процесі відображення параметрів, що цікавлять.

3. Система запису й перегляду вступник інформації, постачена набором візирів, дозволяє визначати значення параметрів у необхідний момент часу.

4. Одержувати відомості про помилки ЕБУ, паспортах ЕБУ, двигуна, калібрувань, таблицях коефіцієнтів паливоподач.

5. Проводити випробування для визначення частоти обертання коленвала, механічних втрат, швидкості прогріву двигуна й інші, залежно від типу ЕБУ.

6. Вести базу даних про клієнтів – власників автомобілів і персональні бази даних для кожного автомобіля по проведених діагностиках, зберігати в базі дані графіки параметрів.

7. Завдяки зручному інтерфейсу легко управляти процесом діагностики автомобіля.

Carman Scan Wi

Carman Scan Wi – програмно-апаратний мультимарочный сканер на базі ПК для діагностики азіатських (включаючи праворульні японські автомобілі) і європейських (німецькі й французькі моделі) до поточного року випуску включно. Ця модель є програмною версією сканера Carman Scan Lite.

Carman Scan Wi є професійним діагностичним апаратом, виконаним у вигляді незалежного модуля з USB – портом і картою LAN для бездротового з'єднання.

Основні мови програмного забезпечення: Російський, Англійський, Китайський, Німецький, Французький, Іспанський, Італійський і ін. Відновлення виходять періодично й накачуються через інтернет у прилад через комп'ютерну програму.

Carman Scan Wi на сьогоднішній день є першим і єдиним сканером, у якому реалізовано ПЗ для діагностики іранських атомобилей Саманд (Samand).

Програмне забезпечення для ПК:

– Аналіз записаних даних за результатами вимірів.

– Завантаження відновлень через спеціальну програму.

Вимоги до ПК:

– Процесор: мінімум Pentium IV.

– USB: мінімум 2.0.

– ОС: Windows 2000/XP/Vista, Linux.

Характеристики основного блоку сканера:

– Пам'ять: 32Мб Внутрішня Флэш Карта.

– Робоча напруга: 7-36 Вольт.

– Споживання електрики: 4,5 Ват.

– Зв’язок із PC: USB кабель або бездротова LAN.

– Робоча температура: 0-60°C.

– Колір корпуса: чорний.

– Колір захисного гумового чохла: чорний.

– Either NET: IEEE 802.11b 11M bps.

– Ширина*Довжина*Товщина: 100*180*25 мм.

– Вага основного блоку: 300 р.

Рисунок 2.3 – Інтерфейс користувача Carman Scan Wi

Диагностуємі автомобілі:

– Asia: Toyota, Lexus, Honda, Nissan, Mitsubishi, Proton, Mazda, Subaru, Suzuki, Isuzu, Infiniti, Holden, Hyundai, Kia, Daewoo, Ssangyong, Китайські.

– Europe: Benz, BMW, Audi, Volkswagen, Saab, Seat, Scoda, Opel, Renault, Citroen, Peugeot, Ford, Fiat, Iran Khodro (Samand), LADA, GAZ, UAZ.

– USA: GGeneral Motors, Chrysler, Ford.

Диагностуємі системи: всі електронні системи й електричні кола автомобіля: ENG, ENG-2, BM/GM, EA, CCS, TCS, ISC, ESCM, IFI/ERE, ELR, EDS, ABS/ETS/ASR, AIRBAG/ETR (SRS), A/T, BAS, ADS, ASD, SPS, 4WD, RB, RST, A/C, IMMO, EPS, ECS, AHLS, AAC, FWDS, FFH, KCS і ін.

Функції сканера:

– Читання й розшифровка кодів помилок.

– Стирання помилок.

– Вивід поточних даних (у цифровому й графічному виді).

– Формування груп параметрів вручну.

– Перевірка (активація) виконавчих механізмів.

– Можливість графічного порівняння обраного параметра з усіма іншими.

– Запис поточних параметрів.

– Ідентифікація систем (блоків управління).

– Проведення адаптації.

– Скидання сервісних інтервалів.

– Читання й програмування іммобілайзера.

Додатково:

– Можливість підключення до стандартного комп’ютера (обробка результатів вимірів).

– Можливість підключення додаткового встаткування через роз’єм USB.

– Можливість виходу на принтер (стаціонарний, переносний).

– Універсальне живлення.

opelscan

Комплекс призначений для діагностики електронних блоків управління автомобілів Opel/Holden/Vauxhall, зроблених з 1987 по 2005 рік. Комплекс має можливість діагностувати наступні модулі:

– Engine – Двигун.

– ABS – Антиблокувальна система гальм.

– Airbag – Подушки безпеки.

– CDLS – Центральний замок.

– Automatic Transmission – автоматична трансмісія.

– Immobilizer – Іммобілізатор.

– Instrument – Комбінація приладів.

– MID – Мультиінформаційний дисплей.

– TID – Трьох-інформаційний дисплей.

– CID – Кольоровий інформаційний дисплей.

– Radio – Радіо.

– Climat Control – Управління клімат-контролем.

– Traction Control – Протибуксовочна система.

– ESP – Протизаносна система.

– EPS – Електричне рульове управління.

– EHPS – Електрогідравлічне рульове управління.

– BC – бортовий комп'ютер.

– Add-On-Heater – Додатковий підігрівник.

– Cooling System – Система охолодження.

При запуску програми відкривається меню вибору модуля для діагностики. Спочатку вибирається Рік Випуску (Select Year) автомобіля, потім модель (Select Model), далі система (Select System) і конкретний ECU (Select Engine/Module). У програму закладена інформація блоків у які роки вони встановлювалися на автомобіль.

Кожний диагностируемый блок управління має свій паспорт із ідентифікаційними даними (ECU Identification), що містить наприклад таку інформацію, як номер блоку по маркуванню Опеля, номер блоку по маркуванню виробника блоку, дата програмування, дата випуску, альфа-код, і ін. Дане вікно показує паспорт блоку.

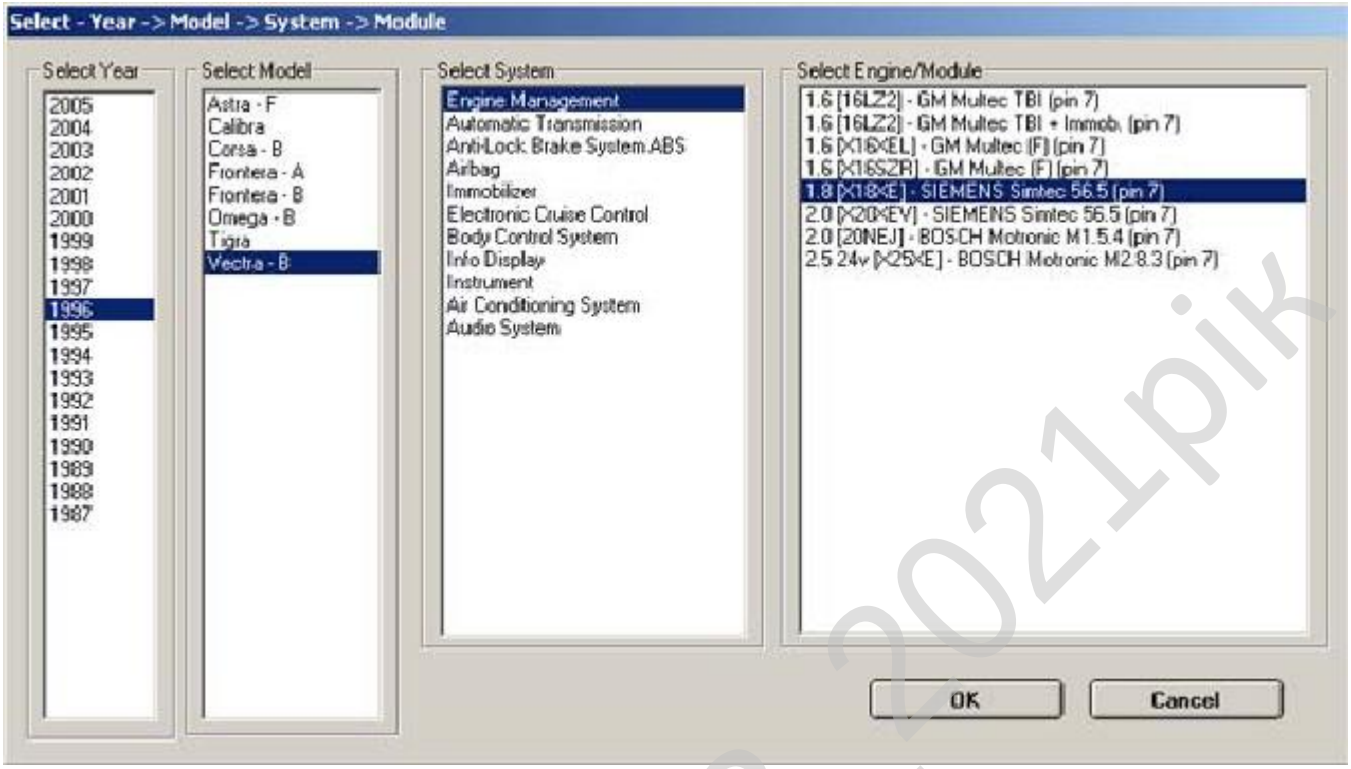

Рисунок 2.4 – Інтерфейс користувача opelscan

Кожний ECU має можливість видачі своїх параметрів у реальному часі. Дане вікно призначене для показу таких даних. У лівій частині вікна розташована таблиця з усіма доступними для даного блоку параметрами. Включити відображення й відновлення всіх параметрів можна кнопкою "Check/Uncheck All". Також є можливість включити/виключити кожної з параметрів, поставивши (або відповідно забравши) напроти назви параметра галочку. Необхідно також ураховувати, що для деяких протоколів включення великої кількості параметрів може привести до істотної з відновлення. Крім того, поточне значення будь-якого одного параметра можливо подивитися великим шрифтом (може бути зручно, якщо комп'ютер перебуває вдалечині), а також у вигляді графіка. Відображення у вигляді графіка може бути зручним при вивченні сигналів, що змінюються в часі, наприклад сигналу датчика кисню й ін. Також для багатьох блоків для більшості параметрів у спеціальному вікні

виводиться інформація про номінальне значення обраного параметра, що дозволяє судити про коректність роботи системи.

У частини блоків є можливість завантажити дамп ЕЕПРОМ. При зчитуванні спочатку задається назва файлу, куди буде записаний завантажений ЕЕПРОМ (за замовчуванням – C:\EEPROM_.bin). Потім необхідно нажати на кнопку – "Download".

У частини блоків є можливість завантажити FLASH пам’ять. Спочатку задається назва файлу, куди буде записаний завантажений FLASH (за замовчуванням – C:\FLASH_.bin). Потім необхідно нажати на кнопку – "Download".

Існує вікно, яке служить для одночасного відображення 4 параметрів у вигляді графіків. Потрібний параметр вибирається зі спадаючого списку (Parameter). Також фіксується для кожного параметра його поточне, мінімальне, максимальне й середнє значення.

Деякі блоки мають можливість програмування певних параметрів системи, наприклад, зміни заданих обертів холостого ходу. Дане вікно призначене для виконання цієї процедури. Спочатку необхідно вибрати з лівого списку, який параметр необхідно змінити. Праворуч у верхнім вікні з’явиться поточне значення даного параметра. Далі, пересуваючи бігунок, можна змінити його бажаним образом, причому нове значення буде знизу. Після вибору потрібного параметра необхідно нажати на кнопку (Program New Value), тим самим змінений параметр збережеться в ECU і буде постійно зберігатися при роботі блоку управління.

Існує ще безліч програм діагностики, які приведемо в наступній таблиці.

Таблиця 2.1 – Існуючі програми діагностики

Програми для автодіагностики	
Назва програми	Короткий опис
1	2
Для російських машин	
Мотор тестер 1.2.0.6 + crack	Сама популярна програма для діагностики автомобілів сімейства ВАЗ, ГАЗ (не стирає помилки, працює під Windows 95, 98).
Мотор тестер 1.2.0.7 + crack	Оновлена версія популярної програми для діагностики автомобілів сімейства ВАЗ, ГАЗ (не стирає помилки, працює під Windows 95, 98).
Tester Plus v1.2.0.9	Діагностична програма автомобілів ГАЗ (підтримує блоки 301.3763, 302.3763, працює під Windows).
Abbat My Tester GAZ v 0.1 beta	Діагностика систем Мікас 5.X, 7.X.
Abbat My Tester VAZ v 1.0	Діагностика систем Січень 5.1, Bosch M1.5.4 (N).
ICD 1.2.0.1	Версія з розширеними можливостями для діагностики систем Bosch M1.5.4, Січень 5, Bosch MP7.0.
BOSCH diag	Діагностика ВАЗ
MP7check	Діагностика ВАЗ
Автоваз new	Нова версія програми вміє працювати із блоками Bosch N154, Січень 5, Січень 4, Bosch MP70, Bosch MP70 EURO 3, GM, M 7.9.7.
kwp2000	Діагностика ВАЗ
Auto	Діагностика ВАЗ
Auto2	Діагностика ВАЗ нова версія

Продовження таблиці 2.1

1	2
Diagnostic Tool v1.0.1b	Діагностика Січень5х і Bosch 154. Діагностує, читає параметри, пише лог-файли, у т.ч. по масовій витраті, детонації, таблицям самонавчання по лямбді. Експорт TXT або Excel, графіки.
Для іномарок	
CarSoft BMW 5.8 + кряк + схема	Сама популярна діагностична програма для сімейства BMW, для роботи із програмою потрібний спец адаптер, прошивання до адаптера в комплект не входить (англійська версія під Windows 98).
Tacho CarSoft BMW 3.4	Програма для діагностики BMW, працює із двопортовим адаптером Uniscan (російська DOS).
CarSoft Mercedes 30	Програма для діагностики Mercedes, працює зі звичайним KL-Line адаптером (російська DOS).
Daewoo GM monitor	Програма діагностики GM-monitor для діагностики автомобілів DAEWOO Nexia і Espero.
Daewoo scanner AKM	Програма для діагностики, тюнингу й калібрування автомобілів Daewoo Lanos, Sens.Програма дозволяє робити діагностику автомобілів DAEWOO, а також змінювати параметри ЕБУ (УОЗ, оберти ХХ, і.т.ін.).
Multec Monitor 1.0 (ламаний)	Призначена для діагностики автомобілів марки OPEL із двигунами C13NZ,C16LZ,C16NZ, E16NZ. На цих автомобілях установлений контролер упорскування системи Multec с одне-одне-спрямованим протоколом передачі даних. Працює з ALDL адаптером (Windows версія).

Продовження таблиці 2.1

1	2
Motornic Diagnost v1.01	Призначена для діагностики автомобілів марки Opel із системами управління Motronic ML4.1 і M1.5 (двигуни C20NE, 20SE, C20SE, C24NE і ін.). Працює зі звичайним K-Line адаптером. Дозволяє вести моніторинг ряду параметрів електронної системи упорскування вищевказаних систем у числовому й графічному видах, переглядати й стирати помилки, управляти виконавчими пристроями (Windows версія).
Uniscan 1.83 + схема адаптера	Сама популярна програма-сканер майже всіх європейських виробників, працює зі своїм адаптером, прошивання в комплект не входять (DOS версія).
Vehicle Explorer Scan Tool Browser v1.03	Програма для діагностики автомобілів за протоколом OBD-2
VAG Scan 3.2	VAG-Scan є повним аналогом спеціалізованих дилерських приладів VAG-1551/52 і призначена для проведення первинної діагностики автомобілів: Audi, VW, Skoda, Seat. Первинна діагностика містить у собі повний контроль комп'ютерних систем автомобіля (двигуна, трансмісії, ABS, підвіски, пристрою проти викрадення, ...). Вторинну діагностику двигуна (функції мотор-тестера) по системах запалювання, вихлопу й оцінки стану двигуна, генератора, стартера, компресії – система не підтримує

Продовження таблиці 2.1

1	2
VAG-COM v. 311n	Програма для діагностики й адаптації автомобілів сімейства VAG. Мова – англійська (Windows версія).
VAG-COM v. 303	Програма для діагностики й адаптації автомобілів сімейства VAG. Мова – англійська (Windows версія).
VAG-beta 78	Програма для діагностики й адаптації автомобілів сімейства VAG. Мова – англійська (Windows версія).
VAG – TOOL 2.0.9	Дозволяє практично повністю здійснювати діагностику автомобілів концерну VAG – VW, Audi, Seat, Skoda. Можливості: читання кодів несправностей, читання потоків даних, активні тести виконавчих вузлів, адаптація, запис (при наявності кодів доступу), робить усе, що робить VAG 1552 (Windows версія).
Volvo Fault Code Reader 1.3.06	Діагностичний сканер. Дозволяє практично повністю здійснювати діагностику автомобілів Volvo.
FreeScan v2.0.6	Програма діагностики американців- Lotus Esprit, Opel/Lotus Omega/Carlton, GM 89-93,Chevrolet Corvette
FreeScan v2.0.7	Програма діагностики Lotus Esprit, Opel/Lotus Omega/Carlton, GM 89-93,Chevrolet Corvette
FreeScan v2.1.0	Програма діагностики Lotus Esprit, Opel/Lotus Omega/Carlton, GM 89-93,Chevrolet Corvette
Honda Logger 2.5	Показує в реальному часі показання всіх датчиків

Продовження таблиці 2.1

1	2
OBD Tool	Діагностика авто за протоколом OBD-II
AutoTap	Діагностика американських авто за протоколом OBD-II
VDS PRO	Діагностика автомобілів групи VAG
CarChat	Діагностика авто концерну GM. Протокол ALDL Win
ALDL Monitor	Діагностика авто концерну GM. Протокол ALDL DOS
Car Scanner 1.0b	Діагностика BMW, Opel і ін.
Fault Code Scanner v 0.1.4c	Програма для старих машин, працює із блінк-кодами, свій адаптер, схема в комплекті.
Програми для чип-тюнинга	
Редактори прошивань	
Для російських машин	
Chip Tuning Pro 2.10 + keygen	Програма для редагування прошивань блоків управління.
Chip Tuning Pro 2.12 + keygen	Програма для редагування прошивань блоків управління.
Програма модифікатор прошивань для Січень-5.1. v1.3	Призначена для підготовки (модифікації софта) прошивань до одночасного завантаження в ЕБУ й перемиканням під час руху, при цьому сам ЕБУ не вимагає ніякої апаратної доробки.
Модифікатор прошивань "B 154-2in1" v1.2 від ALMI	Призначена для підготовки дворежимного прошивання, що містить 2 набори калібрувань. Прошивання міститься в одну звичайну ПЗУ типу 27C512.

Продовження таблиці 2.1

1	2
Dual Bosch M1.5.4 v1.0	Модифікатор дозволяє створювати дворежимне прошивання, що складається з одного програмного модуля й двох, що перемикаються між собою наборів калібрувань.
CTv2.5	Програма чип тюнинга v 2.5 для блоків Bosh M1.5.4, Січень-4, Січень-5
CTuning 1.1.4.1	Програма призначена для зміни калібрувань прошивань блоків Січень-4, Січень-5, Бош-М1.5.4,
Тюнинг іномарок	
BMW Tuning	Програма-завантажник Car Tune для BMW
WinTec2	Програма редагування калібрувань фірми Electromotive
WinTec 3.1.1	Програма редагування калібрувань фірми Electromotive. Остання версія
Програми -завантажники	
Для російських машин	
Схема, прошивання й програма для завантажника блоків Bosch MP 7.0.	Схема й прошивання для мікросхеми AT89C2051, а також Freeware програма MP7 Loader 1.0.
WinFlashECU 1.8	Програма для програмування блоків Січень 5.X, Мікас 7.X і VS 5.X, також у програмі доступні функції очищення EEPROM, на даний момент сама навернена програма для даного софта.
EEPROM Writer 1.1	Програмка для перезапису EEPROM блоків Bosch N 1.5.4, також вона необхідна для деактивації іммобілайзера.

Продовження таблиці 2.1

1	2
WLoader 2.2	Завантажник Січень 5, Мікас 7 з функцією захисту від копіювання
WLoader 2.3	Завантажник Січень 5, Мікас 7 з функцією захисту від копіювання
CombiLoader 2.1.2 (програма)	Універсальний завантажник Bosh MP7, Січень 5, VS5, Мікас 7, з функціями діагностики. (без спеціального адаптера працює тільки функція створення дворежимних прошивань для Мікас 7 і Січень 5)
Ecu Prog v1.8.2+keygen	Самий "просунутий" на сьогоднішній день завантажник ЕБУ, дозволяє робити читання й запис FLASH – пам'яті (ПЗ й калібрування) і EEPROM, а так само робити порівняння буфера із умістом відповідної пам'яті ЕБУ. Працює із блоками Мікас 7 і Січень 5.
Програми корекції одометрів	
Для російських машин	
Combiset 1.3	Програма для коректування одометрів VDO, Счетмаш, нової комбінації Счетмаш, АП і очищення ЕЕПРОМ контролерів BOSCH M1.5.4(N).
Combiset 1.4	Програма для коректування одометрів VDO, Счетмаш, нової комбінації Счетмаш, АП, нова комбінація АП і очищення ЕЕПРОМ контролерів BOSCH M1.5.4(N).
Combiset 1.5	Програма для коректування одометрів VDO, Счетмаш, нової комбінації Счетмаш, Счетмаш 528 T7A, АП, нова комбінація АП і очищення ЕЕПРОМ контролерів BOSCH M1.5.4(N).

Продовження таблиці 2.1

1	2
VDO Reset	Програма для коректування одометрів VDO, Счетмаш, АП.
EEPROM Programmator v0.98 beta 93C46,56,66	Программатор EEPROM серії 93C46/56/66, уміє читати у файл уміст мікросхеми, записувати в мікросхему вміст файлу (розмір не перевіряється). Робота виробляється через порт принтера LPT1.
Kurskset	Програма "kurskset"(c)UncleSam для зміни показань одометра курських комбінацій. Процедура виробляється без розбирання панелі.
Для іномарок	
"Spidometr"	Великий архів готових дампов для приладових панелей, дампи з відключеним іммо для багатьох марок автомобілів. А також багато алгоритмів скрутки приладових панелей.
Tacho Pro Box ID 13011	Програма +схема, +2 прошивання. Робоча версія TachoPro Box id 13011. Програма працює через діагностичний роз'єм , може змінювати значення одометра в багатьох машин (Audi, BMW, Ford, Mercedes, Seat, Shkoda, VW, Opel) – DOS версія.
Tacho Calc	Найбільш великий з відомих на сьогодні калькуляторів пробігу для величезної кількості машин (audi, bmw, fiat, honda, isuzu, citroen, kawasaki, mazda, mercedes, mitsubishi, nissan, opel, peugeot, porshe, renault, rover, saab, skoda, subaru, suzuki, toyota, volvo, vw, yamaha) – Windows версія.

Продовження таблиці 2.1

1	2
Tacho1	Програма для корекції показань одометра через EPROM, працює як программатор мікросхем 93C46, 93C56, 93C66 і 24C02, використовуваних в автомобілях VW, RENAULT, PORSCHE, OPEL, NISSAN, MB, BMW, AUDI, SKODA – DOS версія. Підключається прямо до виводів мікросхеми.
Dashmaster Pro Eprommer v2.2	Програма для корекції показань одометра. Підключається прямо до виводів мікросхеми. Схожа чимсь на Tacho1, тільки працює з більшою кількістю машин. Докладніше про програму читайте на http://www.dashmaster.com/.
Digital DashboardCalculator(DDC)	Корекція одометрів для машин Audi A4 VDO, Audi A3 (1998), Golf 3 VDO, Golf 4 VDO, Golf 3 Motometer, Golf 3 TDI Motometer, VW Vanta Motometer, Mercedes C200,Opel Vectra B VDO, по дампу.
VW Mtm 16	Сервісна утиліта для панелей MotoMeter автомобілів VW Passat B4 1995г, VW Transporter 1996-1997рр, VW Caravella 1998р. EEPROM 93S56. Працює через сервісний роз'єм на задній кришці приладів.
Ford	Зміна значень одометра для практично всіх фордів до 2000р, працює через кабель LPT.
Astra 2	Корекція значень одометра для машин Opel Astra і Opel Zafira (1998-2000) року з панелями VDO (93c46) через діагностичний роз'єм .

Продовження таблиці 2.1

1	2
Vectra	Програма для читання/запису EEPROM і прямого редагування значення одометра для машин Opel Vectra.
Sharan new	Корекція одометрів для машин VW Sharan, Ford Galaxy і SEAT Alhambra (VDO) через діагностичне роз'єм V1+V2 до 07.2000.
VW New Beetle Instruments Scanner V1.2s	Працює тільки з панелями приладів VW New Beetle з 1998 року, розроблених на основі контролера MC68HC912D60, служить для коректування пробігу й читання логіна (Magneti Marelli v.23)
Подушки безпеки	
Airbag diagnose tool v35	Універсальна діагностична програма для подушок безпеки працює з airbag автомобілів VW, Seat, Skoda, Audi серії VW3.
All serie airbag modul	Читання/запис EEPROM, видалення crash data через діагностичний роз'єм для подушок безпеки автомобілів VW, Seat, Skoda (включаючи HC908 і HC12).
Air OPEL	Програма для скидання помилок з пам'яті подушок безпеки автомобілів Opel, працює тільки через COM1.
Air SAAB	Програма для скидання помилок з пам'яті подушок безпеки автомобілів Saab, працює тільки через COM1.
Rover DCU Diagnostic Toll v06.041	Програма для роботи з подушками безпеки автомобілів Rover.

2.2 Обґрунтування вибору засобів для побудови системи та мови програмування

Розробка програмного забезпечення проводилася на мові високого рівня Delphi. Delphi – це комбінація декількох найважливіших технологій:

– Високопродуктивний компілятор у машинний код.

– Об’єктно-орієнтована модель компонентів.

– Візуальна побудова додатків із програмних прототипів.

– Масштабовані засоби для побудови.

Компілятор, вбудований в Delphi, забезпечує високу продуктивність, необхідну для побудови додатків в архітектурі "клієнт-сервер". Він пропонує легкість розробки й швидкий час перевірки готового програмного блоку, характерного для мов четвертого покоління (4GL) і в той же час забезпечує якість коду, характерного для компілятора 3GL. Крім того, Delphi забезпечує швидку розробку без необхідності писати вставки на С або ручного написання коду.

Проектування в Delphi мало чим відрізняється від проектування в інтерпретуючому середовищі, однак після виконання компіляції ми одержуємо код, що виконується в 10-20 разів швидше, ніж те ж саме, зроблене за допомогою інтерпретатора. Основна увага об’єктно-орієнтованої моделі програмних компонентів в Delphi приділяється на максимальному повторному використанні коду. Це дозволяє будувати додатки досить швидко із заздалегідь підготовлених об’єктів, а також дає їм можливість створювати свої власні об’єкти для середовища Delphi. Ніяких обмежень по типах об’єктів, які можуть створювати розроблювачі, не існує. Розробка інтерфейсу для Windows в Delphi є найпростішим завданням для програміста. Середа Delphi містить у собі повний набір візуальних інструментів для швидкісної розробки додатків (RAD – rapid application development), що підтримує розробку користувальницького інтерфейсу й підключення до корпоративних баз даних. VCL – бібліотека візуальних компонентів, містить у собі стандартні об’єкти побудови користувальницького

інтерфейсу, об’єкти управління даними, графічні об’єкти, об’єкти мультимедіа, діалоги й об’єкти управління файлами, управління DDE і OLE. В Delphi візуальні компоненти пишуться на об’єктному Паскалі, на тому же Паскалі, на якому пишеться алгоритмічна частина додатка. Візуальні компоненти Delphi виходять відкритими для надбудови й переписування. Об’єкти БД в Delphi засновані на SQL. До складу Delphi також включений Borland SQL Link, тому доступ до СУБД Oracle, Sybase, Informix і InterBase відбувається з високою ефективністю. Крім того, Delphi містить у собі локальний сервер Interbase для того, щоб можна було розробити розширювані на будь-які зовнішні SQL-сервери додатка в офлайновому режимі програмування. Завдяки відкритій компонентній архітектурі додатка, виготовлені за допомогою Delphi, працюють надійно й стійко. Delphi підтримує використання вже існуючих об’єктів, включаючи DLL, написані на C и C++, OLE серверу, VBX, об ’єкти, створені за допомогою Delphi. З готових компонентів працюючі додатки збираються дуже швидко. Крім того, оскільки Delphi має повністю об’єктну орієнтацію, розроблювачі можуть створювати свої повторно використовувані об’єкти для того, щоб зменшити витрати на розробку. Розроблювачі можуть додавати CASE-інструменти, кодові генератори, а також авторські help’и, доступні через меню Delphi. Two-way tools – однозначна відповідність між візуальним проектуванням і класичним написанням тексту програми

У додатку даються описи всіх процедур і функцій, що містяться в програмі.

2.3 Розгорнута постановка завдання

Згідно з технічним завданням на кваліфікаційну бакалаврську роботу, реалізації підлягає програмне забезпечення, яке призначено для системи інформаційної кабельної мережі автомобіля.

В процесі розробки кваліфікаційної бакалаврської роботи 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

Система функціонує наступним чином. Спершу ЕОМ через кабель з стандартом OBDII (ISO 14230) підключається до бортового комп'ютера автомобіля. Після цього завантажуються стандартні тестові послідовності, які або повертають дані, що системи автомобіля є справними, або повертають коди помилок у роботі того, або іншого блоку автомобіля. Цих кодів помилок дуже багато, порядку 2000, тому у даній кваліфікаційні бакалаврській роботі, приведемо, для прикладу тільки деякі помилки у кодах несправностей стандарту OBDI:

P0 – 1XX – Вимірники палива й повітря.

PO – 100 – Несправність ланцюга датчика витрати повітря.

PO – 101 – Вихід сигналу із припустимого діапазону.

PO – 102 – Низький рівень вихідного сигналу.

PO – 103 – Високий рівень вихідного сигналу.

PO – 105 – Несправність датчика тиску повітря.

PO – 106 – Вихід сигналу із припустимого діапазону.

PO – 107 – Низький рівень вихідного сигналу.

PO – 108 – Високий рівень вихідного сигналу.

PO – 110 – Несправність датчика температури усмоктуваного повітря.

….

P1 – 101 – Сигнал датчика витрати повітря виходить із допустимого діапазону.

P1 – 112 – Сигнал датчика температури повітря на впуску перемежований.

P1 – 116 – Сигнал датчика температури охолоджуючої рідини виходить із допустимого діапазону.

3.2 Розробка структурної схеми

Структурна схема системи зображена на рисунку 3.1.

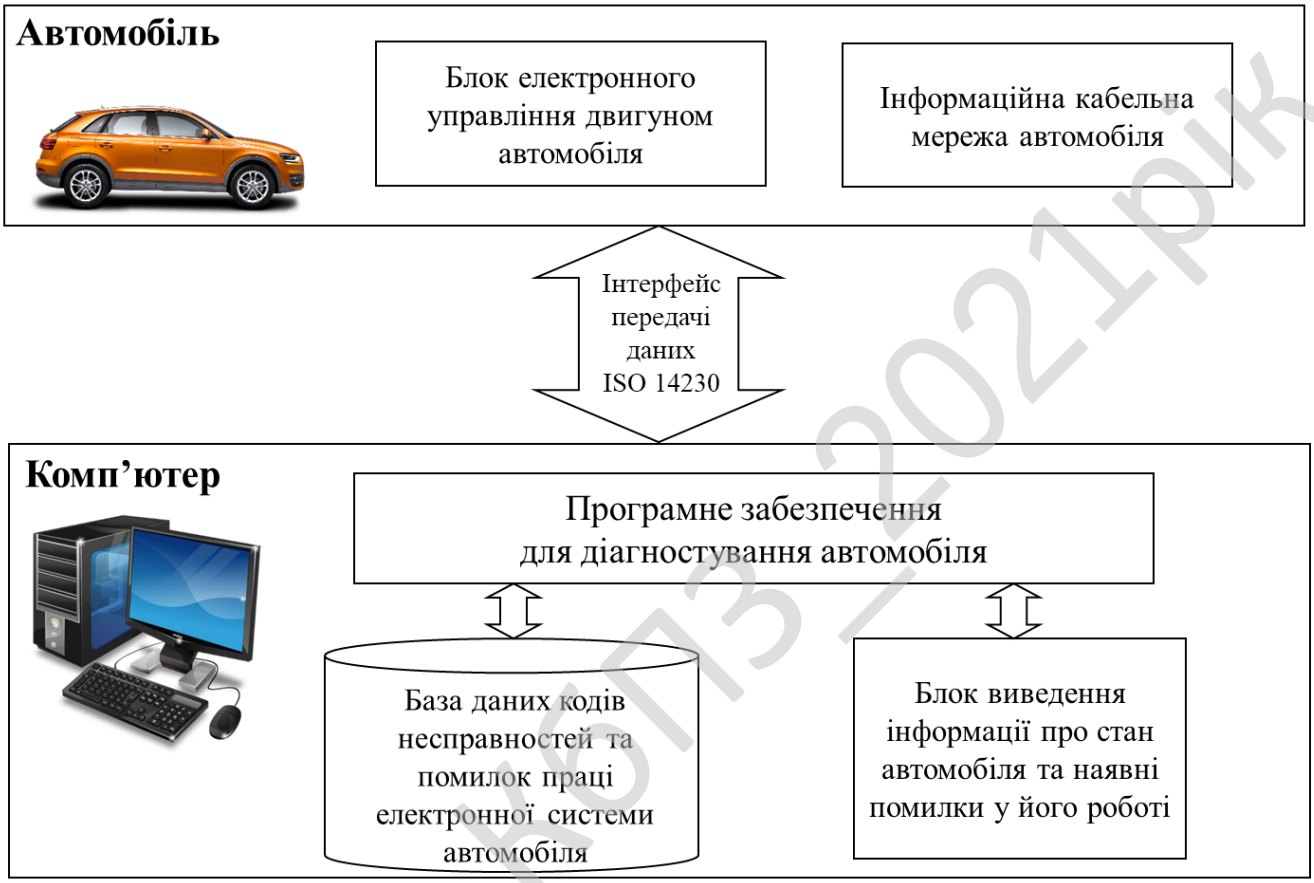

Рисунок 3.1 – Структурна схема системи

З нього ми бачимо, що основними структурними блоками, які взаємодіють між собою, у ній є наступні:

– Персональній комп'ютер.

– Автомобіль (блок електронного управління двигуном автомобіля).

– Інтерфейс передачі даних ISO 14230.

– Програма діагностування.

– База даних кодів несправностей та помилок праці електронної системи автомобіля.

– Блок виведення помилок, або виведення повідомлення про те, що помилок немає.

Розглянемо як працює розроблена система. Абревіатура Е-К-В в автомобільній тематиці має на увазі електронний блок управління або ЕБУ. Тобто ECU це Electronic Control Unit. У сучасному автомобілі є безліч різноманітних ECU. Вони відносяться до гальм, трансмісії, підвісці, системі охорони, кліматичній установці, навігації й іншому. Найважливіший це блок управління двигуном. У різних джерелах йому можуть відповідати назви як ECU, так і DME (Digital Motor Electronics), ECM (Engine Control Module), PCM (Powertrain Control Module) і деякі ін. Будемо дотримуватися загальноприйнятого терміна ECU як «електронний блок управління», доповнюючи його вказівкою приналежності в міру появи необхідності.

Які можуть бути блоки управління:

– Блок управління автономного нагрівника.

– Блок управління АБС гальм з EDS.

– Блок управління системи підтримки безпечної дистанції.

– Передавач системи контролю тиску в шинах, передній лівий.

– Блок управління бортовою мережею.

– Блок управління у двері водія.

– Блок управління доступом і старту.

– Блок управління в комбінації приладів.

– Блок управління електронними приладами на кермовому стовпчику.

– Блок управління телефоном, системою телематик; приймально-передавач для телефону.

– Блок управління двигуном.

– Блок управління Climatronic.

– Блок управління регулюванням сидіння із запам’ятовувальним пристроєм і регулюванням кермового стовпчика.

– Блок управління регулюванням дорожнього просвіту.

– Блок управління коректором фар.

– Блок управління системою контролю тиску в шинах.

– Блок управління 2 бортовою мережею.

– Блок управління MMI передньої інформаційно-командної панелі.

– Діагностичний інтерфейс для шин даних.

– Модуль, приймаче/зчитування, системи антен для доступу без ключа.

– CD-чейнджер.

– CD-ROM-дисковід.

– Блок управління в задніх лівих дверях.

– Блок управління системою Air-Bag.

– Датчик швидкості обертання автомобіля навколо вертикальної осі.

– Блок управління у двері переднього пасажира.

– Блок управління регулюванням сидіння переднього пасажира із запам’ятовувальним пристроєм.

– Блок управління в задніх правих дверях.

– Передавач системи контролю тиску в шинах, задній лівий.

– Радіоприймач стояночного нагрівника.

– Блок управління системою навігації з CD-дисководом.

– Блок управління голосовим уведенням.

– Блок управління цифровою звуковою системою; радіомодуль; TV-тюнер; цифрове радіо.

– Передавач системи контролю тиску в шинах, задній правий.

– Блок управління системою полегшення паркування.

– Центральний блок управління системою комфорту.

– Блок управління електричним стояночним "ручним" гальмом.

– Блок управління енергопостачанням (менеджер батареї).

Універсальний алгоритм

Спосіб діагностики, що викладається, використовує принцип: якщо немає прямих доказів виходу ECU з ладу, то варто почати пошук причини неполадки в

системі в припущенні справності ECU. Прямих доказів дефектності блоку управління існує всього два. Або ECU має видимі ушкодження, або проблема йде при заміні ECU на свідомо справний (ну, або переноситься на свідомо справний а/м разом з підозрілим блоком; іноді це робити небезпечно, до того ж тут зустрічається виключення, коли блок управління ушкоджений так, що не здатний працювати у всьому діапазоні експлуатаційного розкиду параметрів різних екземплярів однієї й тої ж системи управління, але на одному із двох а/м все-таки працює).

Діагностика повинна розвиватися в напрямку від простого до складного й згідно з логікою роботи системи управління. Саме тому припущення про дефект ECU варто залишити «на потім». Спочатку розглядаються загальні міркування здорового глузду, потім послідовній перевірці підлягають функції системи управління. Ці функції чітко розділяються на забезпечуючі роботу ECU і на функції ECU, що виконуються. Спочатку повинні перевірятися функції забезпечення, потім – функції виконання. У цьому головну відмінність послідовної перевірки від довільної: вона виконується по пріоритеті функцій. Відповідно, кожний із цих двох видів функцій може бути представлений своїм списком у порядку убування значимості для роботи системи управління в цілому.

Діагностика успішна тільки тоді, коли вказує на найважливішу із втрачених або порушених функцій, а не на довільний набір таких. Це істотний момент, тому що втрата однієї функції забезпечення може приводити до неможливості роботи декількох функцій виконання. Останні не будуть працювати, але аж ніяк не будуть втрачені, їхня відмова відбудеться просто в результаті причинно-наслідкових зв’язків. Саме тому такі несправності прийнята називати наведеними.

При непослідовному пошуку наведені несправності маскують справжню причину проблеми (досить характерно для діагностики сканером). Зрозуміло, що спроби боротися з наведеними несправностями «у чоло» ні до чого не приводять, повторне сканування ECU дає колишній результат. Ну а ECU «є предмет темний і

науковому дослідженню не підлягає», та й замінити його для проби, як правило, нема чим – от схематичні начерки процесу помилкового вибракування ECU.

Отже, універсальний алгоритм пошуку несправності в системі управління такий:

– візуальний огляд, перевірка найпростіших міркувань здорового глузду;

– сканування ECU, читання кодів несправностей (по можливості);

– огляд ECU або перевірка шляхом заміни (по можливості);

– перевірка функцій забезпечення роботи ECU;

– перевірка функцій виконання ECU.

Важлива роль належить докладному опитуванню власника про те, які зовнішні прояви несправності він спостерігав, як виникла або розвивалася проблема, які дії в цьому зв'язку вже були початі. Якщо проблема в системі управління двигуном, варто приділити увагу питанням про сигналізацію (протиугінну систему), так як електрика додаткових устроїв свідомо менш надійна через спрощені прийоми їхньої установки (наприклад, пайка або стандартні з'єднувачі в призначуваних точках розгалуження й розсічення штатного проведення при підключенні додаткового джгута, як правило, не застосовуються; причому пайка найчастіше не застосовується свідомо через нібито її нестійкість перед вібрацією, що для якісної пайки, звичайно, не так).

Крім того, необхідно точно встановити, який саме а/м перед вами. Усунення скільки-небудь серйозної несправності в системі управління припускає використання електричної схеми останньої. Електросхеми зведені в спеціальні автомобільні комп'ютерні бази по діагностиці й нині досить доступні, треба лише правильно вибрати потрібну. Звичайно, якщо задати саму загальну інформацію з а/м (відзначимо, що бази по електросхемам не оперують VIN-номерами), розвідувач бази знайде кілька різновидів моделі а/м, і буде потрібно додаткова інформація, що може повідомити власник. Наприклад, назву двигуна завжди записано в техпаспорті – букви перед номером двигуна.

Візуальний огляд відіграє роль найпростіших діагностичних затрат. Нерідко досить прості несправності приводять до складних зовнішніх проявів і змушують вважати складною просту проблему. Тому в процесі попереднього огляду повинно перевірятися:

– наявність палива в бензобаку (якщо підозра на систему управління двигуном);

– відсутність затички у вихлопній трубі (якщо підозра на систему управління двигуном);

– чи затягнуті клеми акумуляторної батареї (АКБ) і їхній стан;

– відсутність видимого ушкодження електропроводки;

– чи добре вставлені (повинні бути защелкнуті й не переплутані) роз'єми проведення системи управління;

– попередні чужі дії по подоланню проблеми;

– дійсність ключа запалювання – для автомобіля зі штатним іммобілайзером (якщо підозра на систему управління двигуном);

Іноді буває корисно оглянути місце установки ECU. Не так уже рідко воно виявляється залито водою, наприклад, після мийки двигуна установкою високого тиску. Вода згубна для ECU негерметичного виконання. Помітимо, що роз'єми ECU також бувають як герметичного, так і простого виконання. Роз'єм повинен бути сухим (припустимо застосовувати як водовідштовхувальні кошти, наприклад, WD-40).

Читання кодів несправностей

Якщо для читання кодів несправностей застосовується сканер або комп'ютер з адаптером, важливо, щоб їхнє з'єднання із цифровою шиною ECU було виконано правильно. Ранні ECU не встановлюють зв'язок з діагностикою, поки не приєднані обидві лінії K і L.

Сканування ECU, або активація самодіагностики автомобіля дозволять швидко визначити нескладні проблеми, наприклад, із числа виявлення

несправних датчиків. Причому для ECU однаково, несправні сам датчик або його дроти.

Завжди варто пам’ятати, що формування ECU діагностичних повідомлень, зчитувальних сканером, значно формалізоване й підлегле закладеній програмі самодіагностики. Остання видає найбільш імовірну на думку розроблювача, що написав програму, причину несправності. Але очевидно, що найбільш імовірна причина буде реалізовуватися не щораз.

Ще гірше, коли найбільш імовірну причину алгоритм самодіагностики ігнорує (очевидно, у чинність великої віри розроблювача в надійність того або іншого вузла). Так, наприклад, самодіагностика деяких французьких автомобілів при відсутності пуску двигуна через обрив ланцюга датчика положення коленвала не передбачає тестування цього датчика. І тому навіть дилерський прилад DIAG-2000 у цілому ряді випадків при перевірці системи управління двигуном не знаходить придбаний обрив зазначеного ланцюга.

Виконавчі механізми, наприклад, реле, керовані ECU, перевіряються сканером у режимі примусового включення навантажень. Цей режим називається тестом виконавчих механізмів. Тут знов-таки важливо відрізняти дефект у навантаженні від дефекту в її проведенні.

По-справжньому повинна насторожувати ситуація, коли спостерігається сканування множинних кодів несправностей. При цьому досить велика ймовірність того, що частина з них відноситься до наведених несправностей. Така вказівка на несправність ECU, як «немає зв’язку», – означає, швидше за все, що ECU знеструмлено або відсутній яке-небудь одне його живлення або заземлення.

Якщо немає у своєму розпорядженні сканер або його еквівалент у вигляді комп’ютера з адаптером ліній K і L, більшу частину перевірок можна зробити вручну (див. розділи «Перевірка функцій...»). Звичайно, це буде повільніше, але при послідовному пошуку й обсяг роботи може бути невеликий.

Огляд і перевірка ECU

У тих випадках, коли доступ до ECU простий, а сам блок може бути легко розкритий, варто оглянути його. От що може спостерігатися в несправному ECU:

– обриви, відшарування струмоведучих доріжок, часто з характерними підпалинами;

– спучені або тріснуті електронні компоненти;

– прогари друкованої плати аж до наскрізних;

– вода;

– окисли білий, синьо-зелений або коричневий кольори;

Як уже було сказано, вірогідно перевірити ECU можна шляхом заміни на свідомо справний. Дуже добре, якщо діагност розташовує перевірочним ECU. Однак варто зважати на ризик вивести цей блок з ладу, адже часто першопричина проблеми – несправність зовнішніх ланцюгів. Тому необхідність мати перевірочні ECU не очевидна, а сам прийом варто застосовувати з великою обачністю. На практиці набагато продуктивніше в початковій фазі пошуку вважати ECU справним уже тільки тому, що його огляд не переконує у зворотному. Буває нешкідливо просто переконатися, що ECU на місці.

Перевірка функцій забезпечення

До функцій забезпечення роботи ECU системи управління двигуном відносяться:

– живлення ECU як електронного устрою;

– обмін з керуючим блоком іммобілайзера – якщо є штатний іммобілайзер;

– запуск і синхронізація ECU від датчиків положення коленвала й/або розподільчого вала;

– інформація з інших датчиків.

3.3 Розробка функціональної схеми

Функціональна схема – це схема взаємодії компонентів програмного забезпечення та інформаційних потоків, а також даних у потоках. Функціональні схеми, більш інформативні, ніж структурні.

На рисунку 3.2 зображена функціональна схема системи. Нижче розглянемо її більш докладно.

Функціональна схема складається із взаємодії наступних функціональних блоків:

– Блок доступу до функцій читання й стирання кодів несправностей (DTC).

– Блок виведення опису кодів несправностей (DTC definitions).

– Блок підтримки автомобілів обладнаних шиною CAN.

– Блок читання й видалення всіх загальних кодів несправностей (DTC).

– Блок читання й видалення кодів несправностей заданих виробником (manufacture specific DTC).

– Блок відображення поточних параметрів OBDII (ISO 14230) (Live Data у момент виникнення несправності).

– Блок виведення стоп-кадрів помилок OBDII.

– Блок виведення результатів внутрішніх тестів системи самодіагностики.

– Блок визначення заводського номера кузова VIN (для автомобілів з 2002 р.в.).

– Блок визначення статусу значка "Check Engine".

Розглянувши усі блоки функціональної схеми перейдемо до розгляду діаграми взаємодії процесів, які відбуваються у системі.

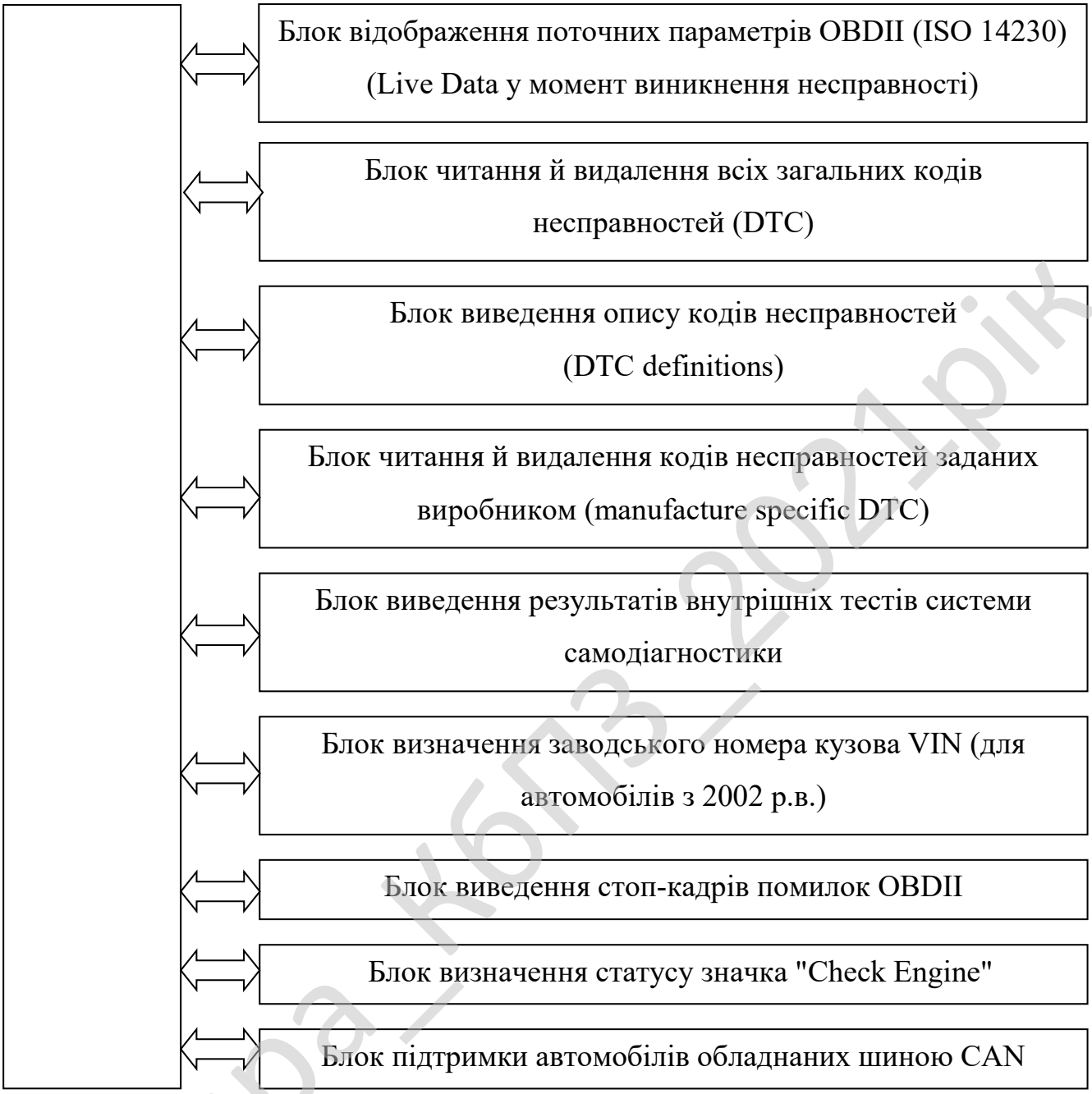

Рисунок 3.2 – Функціональна схема системи

3.4 Розробка діаграми процесів

Діаграма взаємодії процесів системи, розробленої у результаті виконання кваліфікаційної бакалаврської роботи, наведена на рисунку 3.3.

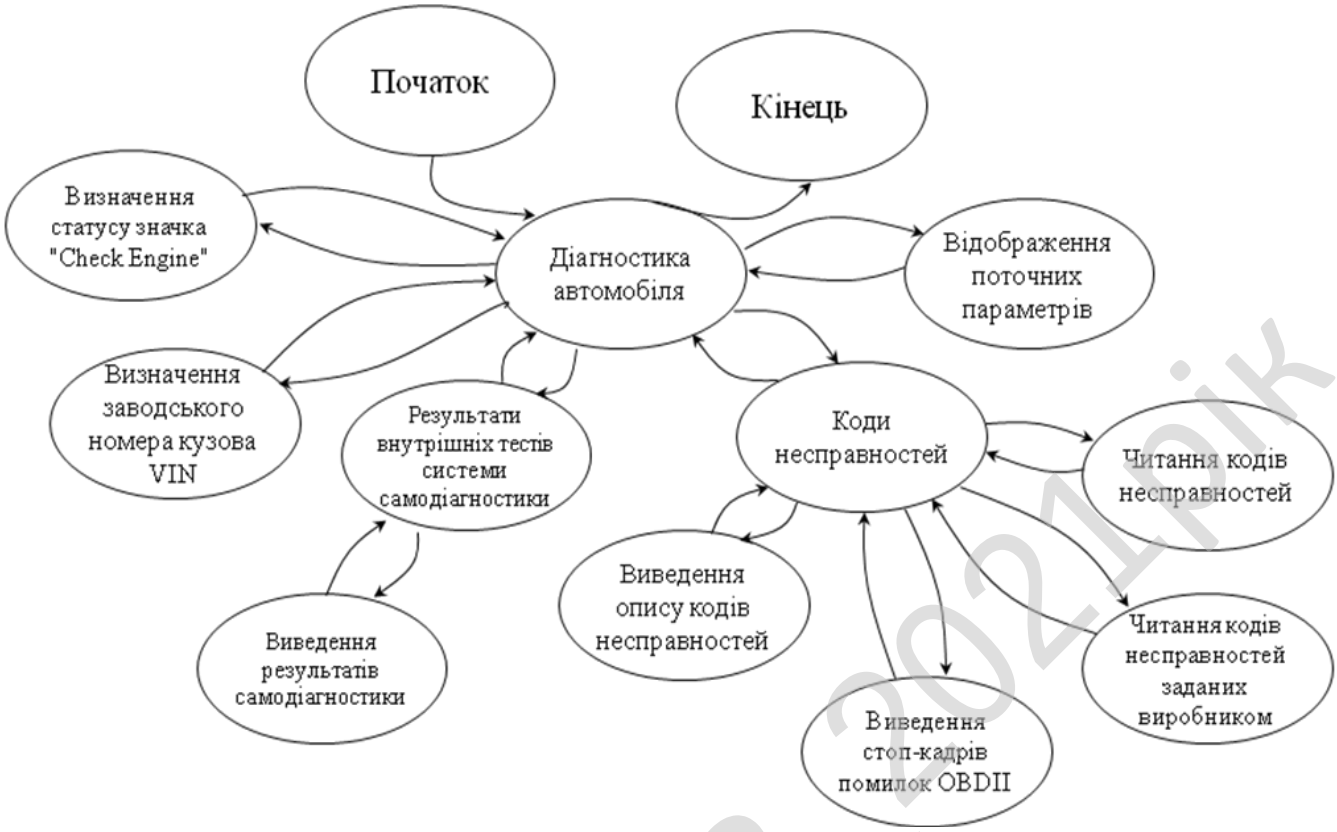

Рисунок 3.3 – Діаграма взаємодії процесів

Першим процесом який запускається у системі, є процес завантаження вікна діагностики автомобіля.

Він взаємодіє з наступними процесами:

– Процес визначення статусу значка «Check Engine».

– Процес визначення заводського номера кузова.

– Процес запуску виведення внутрішніх тестів системи самодіагностики, який взаємодіє з процесом виведення результатів самодіагностики.

– Процес роботи з кодами несправностей.

– Процес відображення поточних параметрів.

Процес роботи з кодами несправностей, у свою чергу взаємодіє з наступними процесами: процес читання кодів несправностей, процес читання кодів несправностей заданих виробником, процес виведення опису кодів несправностей, процес виведення стоп-кадрів помилок OBDII.

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

4.1 Блок-схеми та опис алгоритмів функціонування системи

На рисунку 4.1 наведено блок-схему основної програми. Її робота складається з виконання наступних кроків.

Спершу відбувається виведення основного вікна програми. Після цього користувач обирає, проводити йому діагностику, або ні.

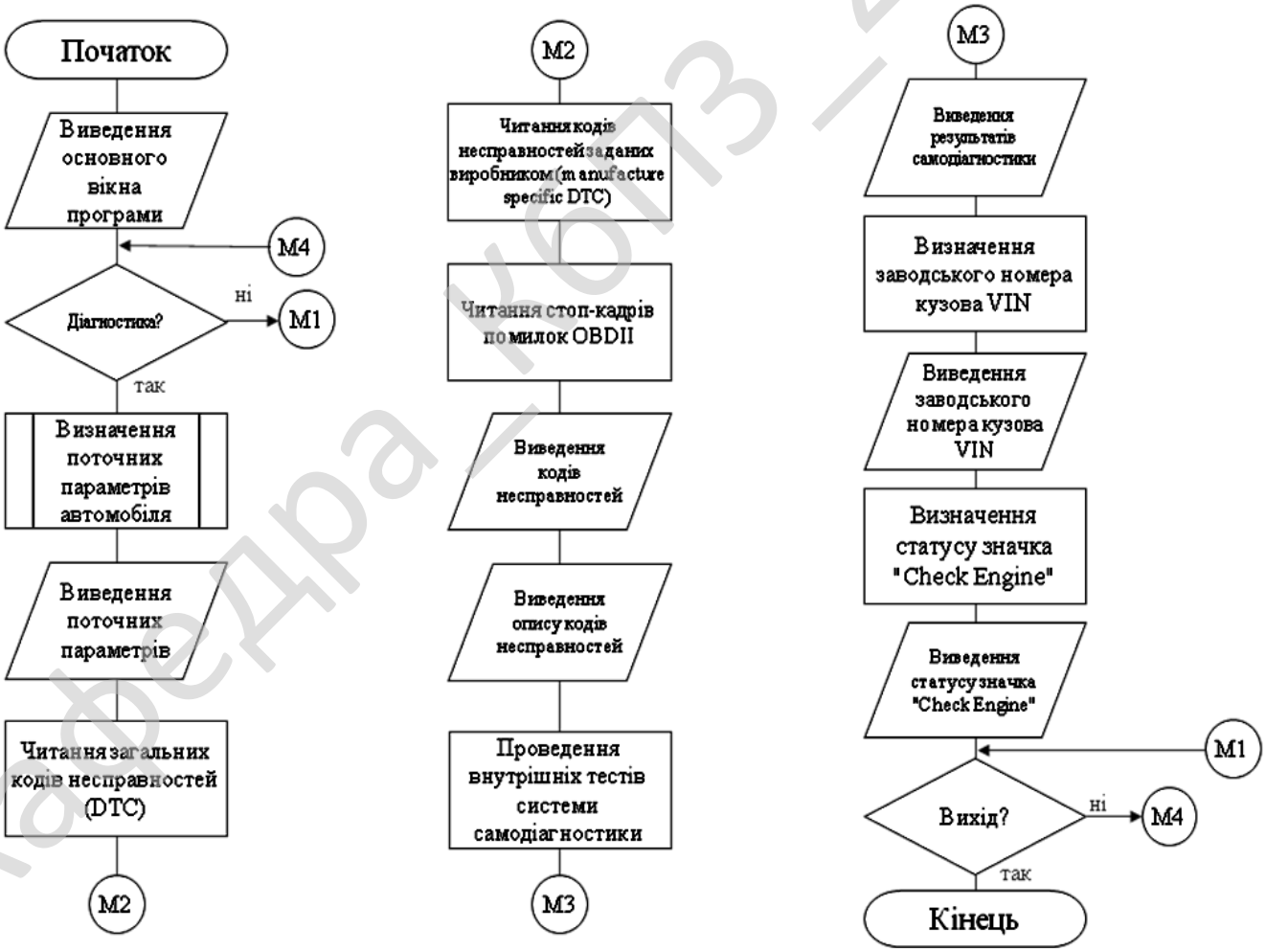

Рисунок 4.1 – Блок-схема роботи основної програми

Якщо він обирає діагностику, то виконується послідовність наступних дій:

– Визначення поточних параметрів автомобіля.

– Виведення поточних параметрів.

– Читання загальних кодів несправностей.

– Читання кодів несправностей заданих виробником.

– Читання стоп-кадрів помилок OBDII

– Виведення кодів несправностей.

– Виведення опису кодів несправностей.

– Проведення внутрішніх тестів системи самодіагностики.

– Виведення результатів самодіагностики.

– Визначення заводського номера кузова.

– Виведення заводського номера кузова.

– Визначення статусу значка «Check Engine».

На рисунку 4.2 зображено блок-схему роботи підпрограми визначення поточних параметрів автомобіля. Вона працює наступним чином.

Спершу відбувається спроба з’єднання з автомобілем. Після цього, якщо сигналу немає, тоді виводиться повідомлення про помилку, й підпрограма завершає свою роботу.

У іншому випадку, тобто, якщо сигнал є, відбувається покрокове виконання наступних ітерацій:

– Отримуються значення температури мотору.

– Отримуються значення оборотів мотора.

– Отримуються значення тиску наддуву.

– Отримуються значення подачі палива.

– Отримуються значення температури палива.

– Отримуються значення температури повітря.

– Отримуються значення атмосферного тиску.

– Отримуються значення пробігу.

– Отримуються значення стану педалі зчеплення.

– Отримуються значення педалі гальма.

– Отримуються значення педалі газу.

– Отримуються значення температури насосу.

– Отримуються значення затримки ротора.

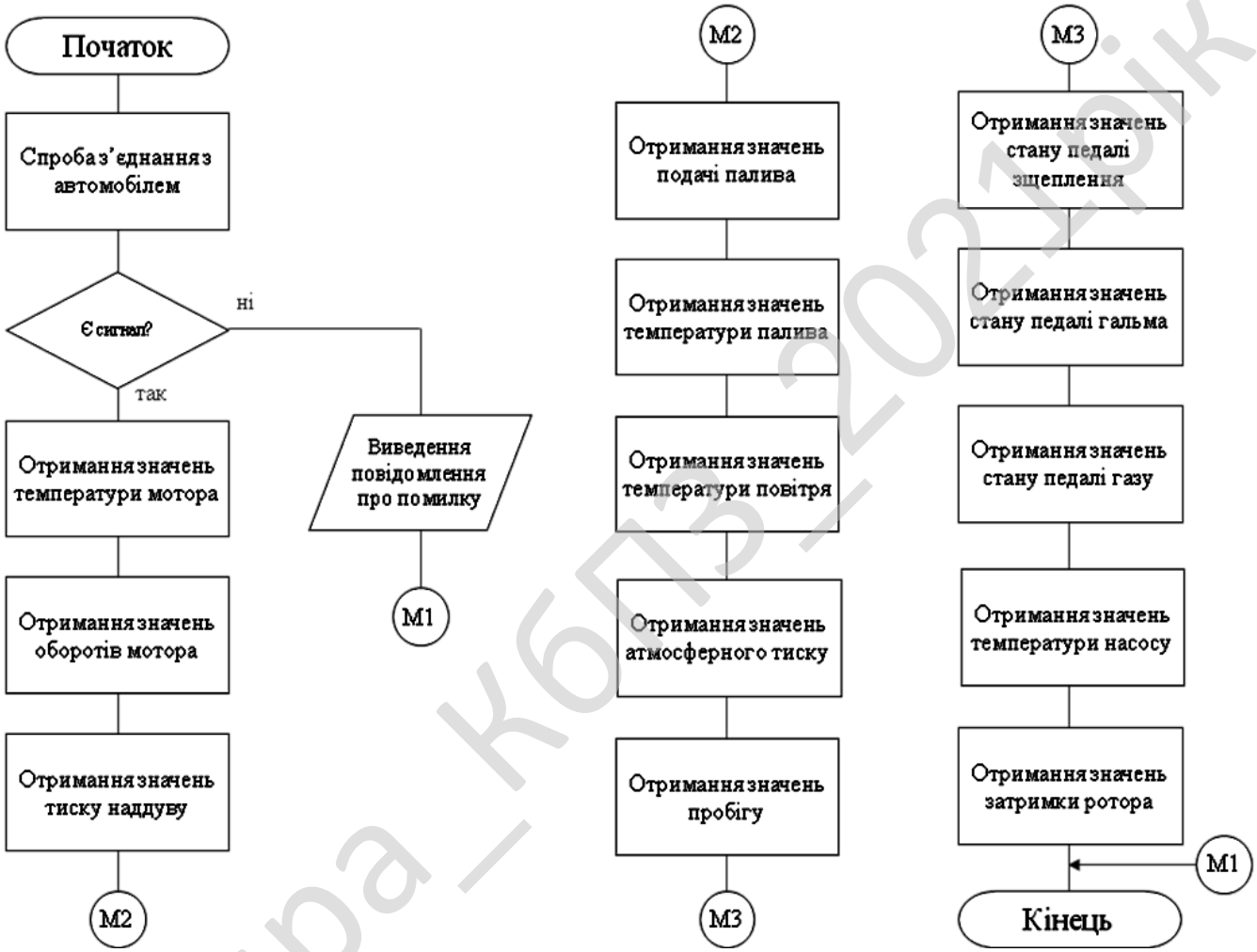

Рисунок 4.2 – Блок-схема роботи підпрограми визначення поточних параметрів автомобіля

Приведемо приклад самодіагностики систем керування на автомобілі OPEL. Іноді, у системі керування упорскуванням автомобілів виникають несправності. Довідатися причину, у більшості випадків, можна самостійно, зчитавши код помилки в накопичувачі ECU через діагностичний роз'єм.

На автомобілі OPEL випуску 86-94 років установлювалися системи діагностики з 10-ти контактним роз'ємом і протоколами діагностики ALDL (Multec) або K-Line (Motronic).

На OPEL випуску 94-96 років установлювалася системи діагностики з 16-ти контактним роз'ємом і протоколом діагностики K-Line. Більше пізні моделі мали систему діагностики, сумісну зі стандартом OBD-II і 16 -ти контактним роз'ємом. З 2002 року вводиться CAN протокол на вже стандартному 16-ти контактному OBD-II рознімані.

Варто пам'ятати, що помилка в накопичувачі може мати статус "активної", тобто несправність у системі присутня в цей момент. При цьому буде горіти лампа "check engine" на панелі приладів. І "збереженої" – лампа "check engine" горіти не буде. Якщо ця помилка не виникне повторно протягом 20 запусків двигуна, то вона буде стерта з накопичувача. Так-же накопичувач скидається при знятті клеми акумулятора на 30 секунд. У сучасних системах, помилки можна видалити тільки за допомогою діагностичного встаткування, після усунення їхньої причини.

Розглянемо варіант самодіагностики систем Motronic і Multec з 10 контактним діагностичним роз'ємом.

1. Знаходимо сам роз'єм.

В OPEL Kadett, Vectra і Omega він перебуває біля опори стійки, з лівої або правої сторони моторного відсіку. В OPEL Tigra, Corsa і Astra – у щитку запобіжників салону. В OPEL Frontera його ставлять над правою (пасажирської) фарою.

Розташування й призначення його контактів:

A – Маса автомобіля.

B – L-Line Самодіагностика ECU engine.

C – Самодіагностика АКПП.

D – Самодіагностика Trip computer.

E – Канал даних двигуна.

F – "+" Акумуляторної батареї.

G – K-Line канал даних комп'ютерів.

H – Самодіагностика круїзу-контролю.

J – Самодіагностика повного привода.

K – Самодіагностика ABS.

2. Ставимо дротову перемичку залежно від мети діагностики:

AB – комп'ютер двигуна (Motronic, Multec).

AC – електронна автоматична коробка передач.

AD – бортове табло LCD і бортовий комп'ютер.

AG – стандартна, перешкодозахисна заглушка.

AH – комп'ютер круїз-контролю, пристрою проти викрадення (ATWS).

AJ – електронне керування повним приводом.

AK – комп'ютер ABS.

У сервісних мануалах, ця перемичка називається "діагностичне пристосування KM 602-2"

3. Включаємо запалювання й зчитуємо flash коди, які нам починає видавати лампа індикації відповідного пристрою. Деякі версії систем можуть не підтримувати самодіагностикові.

Структура показу кодів проста:

Перша серія спалахів з короткими паузами – десятки.

Пауза побільше – роздільник.

Друга серія спалахів з короткими паузами – одиниці.

Довга пауза – роздільник кодів.

x-xx (12).

Кожний цикл починається з коду входу в режим самодіагностики – 12.

Далі видаються коди по зростанню номера помилки. Усе коди повторюються по 3 рази. Після закінчення показу кодів помилок, цикл повторюється заново.

Розшифровка flash кодів помилок наведена в таблицях Flash codes Opel.

Варіант самодіагностики з роз'ємом OBD-II (якщо система його підтримує), мало відрізняється від описаного вище.

1. Знаходимо сам роз'єм.

В OPEL Vectra, Astra і Zafira він перебуває під захисною кришкою, у салоні під важелем ручного гальма.

В OPEL Omega – у щитку запобіжників салону, під кермовою колонкою.

В OPEL Corsa – у ніші центральної консолі.

2. Ставимо перемичку між контактами 5 і 6.

3. Включаємо запалювання й зчитуємо flash коди, які нам починає видавати лампа індикації. Деякі моделі систем можуть не підтримувати самодіагностики.

Для ранніх систем, принцип зчитування кодів такой-жі. Для більше пізніх, є невеликі відмінності:

10 спалахів лампи відповідає початку коду P0. Далі треба сам 3х значний код.

x-x-xxx (P0113)

Коди повторюються 3 рази і йдуть по зростанню номера помилки. Після закінчення, цикл повторюється заново.

Розшифровка flash кодів помилок стандарту OBD-II наведена в таблицях Flash codes Opel.

Розшифровка flash кодів помилок не OBD-II стандарту наведена в таблицях Flash codes Opel.

Розшифровка flash кодів помилок не OBD-II стандарту для АКПП наведена в таблицях Flash codes Opel, тільки замикати необхідно не 5-6 контакти, а 3-5.

У більш пізніх систем, категорії несправностей розділені, відповідно розділені й лампи індикатори.

Варто пам'ятати, що помилки можуть бути як прямі (точно вказувати на несправність), так і непрямі (указувати на несправність систем, що перебувають

за діапазоном перевірки). Наприклад, помилка лямбда зонда (oxygen sensor), регулятора холостого ходу (IAC) або витратоміра повітря (AFS), може означати так-же й просос неврахованого повітря у впускну систему.

Самі по собі помилки систем не можуть дати повну й точну інформацію про проблеми, більше того – проблеми можуть бути й зовсім без виникнення помилок.

З портами з-під Win32 працюють так само, як і зі звичайними файлами, використовуючи при цьому всього кілька специфічних функцій WinAPI. Однак комунікаційний порт – це не зовсім звичайний файл. Для нього, наприклад, не можна виконати позиціонування файлового покажчика, або ж створити порт, якщо такий відсутній. Будь-яка робота з портом починається з його відкриття. Для цього використовується файлова функція WinAPI (опис WinAPI-функцій узяті з MSDN (Microsoft Developer Network), отже, приводяться в синтаксисі C):

```
HANDLE CreateFile(
 LPCTSTR lpFileName,
 DWORD dwDesiredAccess,
 DWORD dwShareMode,
 LPSECURITY_ATTRIBUTES lpSecurityAttributes,
 DWORD dwCreationDistribution,
 DWORD dwFlagsAndAttributes,
 HANDLE hTemplateFile
);
```

lpFileName – покажчик на рядок з нульовим завершальним символом. Звичайно це ім’я файлу, що відкривається, але в нашім випадку це повинне бути назва порту (COM1, COM2, ...).

dwDesiredAccess – тип доступу. У нашім випадку повинен бути дорівнює GENERIC_READ|GENERIC_WRITE.

dwShareMode – параметр спільного доступу. Для комунікаційних портів завжди дорівнює 0.

lpSecurityAttributes – атрибут захисту. Для комунікаційних портів завжди дорівнює NULL.

dwCreationDistribution – режим автостворення. Для комунікаційних портів завжди дорівнює OPEN_EXESTING.

dwFlagsAndAttributes – атрибут режиму обробки. Для комунікаційних портів повинен бути дорівнює 0 або FILE_FLAG_OVERLAPPED.

hTemplateFile – описувач файлу-шаблона. Для комунікаційних портів повинен бути дорівнює NULL.

При успішному відкритті порту функція повертає його описувач, а у випадку помилки повертає INVALID_HANDLE_VALUE.

Із всіх параметрів функції CreateFile() особливого пояснення вимагає dwFlagsAndAttributes. Робота з портом може бути організована в синхронному (nonoverlapped) або асинхронному (overlapped) режимах обробки, що й задається цим прапором. При синхронному режимі (коли параметр dwFlagsAndAttributes = 0) тільки один потік додатка може або читати, або писати в порт.

Синхронний режим обробки простий у реалізації. Якщо треба записати дані в порт, то викликаємо функцію запису й очікуємо, поки вона не завершиться. Якщо ж треба читати дані, то викликаємо функцію читання й чекаємо, поки вона не відробить. Для простих завдань синхронний режим обробки цілком підходить, однак у світі Windows він майже завжди приречений на невдачу. Очікування операції читання або записи сприймається користувачем програми як «зависання».

Асинхронний режим (коли параметр dwFlagsAndAttributes = FILE_FLAG_OVERLAPPED) дозволяє робити операції читання й запису в порт паралельно з різних потоків. У той час, поки один потік додатка приймає дані, інший потік може паралельно з першим передавати дані – як при розмові по телефоні, коли ви можете слухати й говорити одночасно. Даний режим обробки більше імпонує ідеї багатозадачності Windows. Але за все треба платити: для реалізації цього режиму обробки потрібно у два рази більше написаного коду, до

того ж, уміння писати багатопоточні програми. Який режим вибрати – вирішуйте самі. Розглянемо більше складний варіант – асинхронну обробку.

На практиці відкриття порту для асинхронного режиму обробки із програми на Delphi виглядає так:

```
hPort := CreateFile('COM1', GENERIC_READ or GENERIC_WRITE, 0, nil,
 OPEN_EXISTING, FILE_FLAG_OVERLAPPED, 0);
if hPort = INVALID_HANDLE_VALUE then
 raise Exception.Create('Помилка відкриття порту');
```

Функція повертає описувач порту (hPort), що нам потім придасться для виклику інших функцій роботи з портом. Якщо в результаті відкриття порту описувач не отриманий, то збуджується виключення з відповідним текстом помилки. Відкривши порт, ми одержуємо його у своє розпорядження. Тепер із цим портом може працювати тільки наша програма (точніше, тільки наш процес). По закінченні роботи з портом його варто закрити, викликавши функцію:

```
BOOL CloseHandle( HANDLE hObject );
```

Як єдиний параметр треба передати отриманий раніше описувач порту (hPort).

Хоч система при завершенні виконання програми й звільняє всі виділені їй ресурси (у тому числі й порти), гарним тоном програмування вважається власноручне закриття портів. Відкривати/закривати порт начебто нескладно. Крім того, нам буде потрібно програмне настроювання порту. Думаю, усі бачили діалог настроювання послідовного порту в диспетчері пристроїв системи. Всі ці настроювання ми можемо зробити програмно. Для цих цілей використовується функція WinAPI:

```
BOOL SetCommState( HANDLE hFile,  LPDCB lpDCB );
```

hFile – описувач відкритого порту.

lpDCB – покажчик на структуру DCB.

Основні параметри послідовного порту описуються структурою DCB. Вона містить масу полів, кожне з яких відповідає певному параметру настроювання порту. Ми розглянемо кілька полів, які нам потрібні:

BaudRate – швидкість передачі даних. Можлива вказівка констант - CBR_100, CBR_300, CBR_600, CBR_1200, …, CBR_256000...

Parity – схема контролю парності. Може містити одне з наступних значень: EVENPARITY, MARKPARITY, NOPARITY, ODDPARITY, SPACEPARITY.

ByteSize – число інформаційних біт у переданих і прийнятих байтах.

StopBits – кількість стопових біт. Може бути ONESTOPBIT, ONE5STOPBIT, TWOSTOPBIT.

Щоб не заповнювати структуру DCB вручну, її можна заповнити інформацією про поточний стан порту викликом функції GetCommState(), потім змінити необхідні поля й установити настроювання викликом функції SetCommState(). Настроювання порту бажано робити відразу після його відкриття. На Delphi це виглядає так:

```
var
 Dcb: TDcb;
…
if not GetCommState(hPort, Dcb) then
 raise Exception.Create('Помилка установки стану порту');
Dcb.BaudRate := CBR_9600;
Dcb.Parity := NOPARITY;
Dcb.ByteSize := 8;
Dcb.StopBits := ONESTOPBIT;
if not SetCommState(hPort, Dcb) then
 raise Exception.Create('Помилка установки стану порту');
```

Ще одна операція, що нам знадобиться відразу після відкриття порту – його скидання.

```
BOOL PurgeComm(
 HANDLE hFile,
 DWORD dwFlags
);
```

Виклик цієї функції очищає черга прийому/передачі й завершує всі запити, що перебувають чекаючи, уведення/виводу.

hFile – описувач відкритого порту.

dwFlags – вироблені дії у вигляді набору прапорів PURGE_TXABORT, PURGE_RXABORT, PURGE_TXCLEAR, PURGE_RXCLEAR.

```
if not PurgeComm(hPort, PURGE_TXCLEAR or PURGE_RXCLEAR) then
 raise Exception.Create('Помилка, що очищає порт');
```

На цьому підготовча фаза закінчується, і можна приступати безпосередньо до прийому/передачі даних. Прийом даних у нас буде відбуватися по подійній схемі; програма буде очікувати прийом одного або декількох символів (байт). Для перекладу порту в цей режим необхідно викликати функцію SetCommMask() із прапором EV_RXCHAR:

```
if not SetCommMask(hPort, EV_RXCHAR) then
 raise Exception.Create('Помилка установки маски порту');
```

Прийом і передача даних виконується функціями ReadFile() і WriteFile(), тобто тими ж самими функціями, які використовуються для роботи з дисковими файлами. От їхній опис:

```
BOOL ReadFile(
 HANDLE hFile,
 LPVOID lpBuffer,
 DWORD nNumberOfBytesToRead,
 LPDWORD lpNumberOfBytesRead,
 LPOVERLAPPED lpOverlapped
);
BOOL WriteFile(
 HANDLE hFile,
 LPCVOID lpBuffer,
 DWORD nNumberOfBytesToWrite,
 LPDWORD lpNumberOfBytesWritten,
 LPOVERLAPPED lpOverlapped
);
```

hFile – описувач відкритого порту.

lpBuffer – адреса буфера.

nNumberOfBytesToRead/nNumberOfBytesToWrite – число очікуваних до прийому або призначених для передачі байт.

lpNumberOfBytesRead/lpNumberOfBytesWritten – число фактично прийнятих або переданих байт.

lpOverlapped – адреса структури OVERLAPPED, використовуваної для асинхронних операцій.

Передача даних є досить швидкою операцією, тому як правило її виконують із головного потоку додатка. На Delphi це виглядає так:

```
var
 dwWrite: DWORD;
 OverWrite: TOverlapped;
 WriteBytes: array of Byte;
…
begin
 OverWrite.hEvent := CreateEvent(nil, True, False, nil);
 if OverWrite.hEvent = Null then
  raise Exception.Create('Помилка створення події запису');
 …
 if (not WriteFile(hPort, WriteBytes, SizeOf(WriteBytes),
  dwWrite, @OverWrite))
 and (GetLastError <> ERROR_IO_PENDING) then
  raise Exception...Create('Помилка запису в порт');
end;
```

Функція WriteFile() виконує асинхронний запис масиву байтів WriteBytes у порт. Вона відразу повертає керування, і запис у порт відбувається паралельно з виконанням основного коду потоку. Якщо результат WriteFile() дорівнює False, то це значить, що на момент повернення керування передача масиву байтів ще не закінчилася. Тому код помилки виконання WriteFile() у цьому випадку повинен бути дорівнює ERROR_IO_PENDING. Змінна OverWrite – overlapped-структура, необхідна для асинхронних операцій.

У принципі, не повинне хвилювати, коли закінчиться передача масиву байтів. Зате момент прийому одного або декількох символів дійсно важливий. Тому його можна розбити на дві частини: ініціювання прийому й визначення моменту прийому з наступним читанням символів. Оскільки при цьому доводиться зважати на фактора очікування прийому символу, рекомендується функції прийому даних винести в окремий потік. Передавати дані можна й з основного потоку додатка, оскільки це відбувається досить швидко. А от подію прийому символу будемо очікувати в окремому потоці.

В будь-якого потоку є головна функція, що починає виконуватися після його створення. В Delphi для потоків існує клас TThread, а його головна процедура називається TThread.Execute().

От так виглядає головна процедура окремого потоку, що очікує появу одного або декількох символів і зчитує їх:

```
procedure TReadThread.Execute;
var
 ComStat: TComStat;
 dwMask, dwError: DWORD;
 OverRead: TOverlapped;
 Buf: array[0..$FF] of Byte;
 dwRead: DWORD;
begin
 OverRead.hEvent := CreateEvent(nil, True, False, nil);
 if OverRead.hEvent = Null then
  raise Exception.Create(' Помилка створення події читання ');
 FreeOnTerminate := True;
 while not Terminated do
 begin
  if not WaitCommEvent(hPort, dwMask, @OverRead) then
  begin
   if GetLastError = ERROR_IO_PENDING then
    WaitForSingleObject(OverRead.hEvent, INFINITE)
   else
    raise Exception.Create('Помилка очікування події порту');
  end;
  if not ClearCommError(hPort, dwError, @ComStat) then
   raise Exception.Create('Помилка очищення порту');
  dwRead := ComStat.cbInQue;
  if dwRead > 0 then
  begin
   if not ReadFile(hPort, Buf, dwRead, dwRead, @OverRead) then
    raise Exception.Create('Помилка читання порту');
   // В Buf перебувають прочитані байти
   // Далі йде обробка прийнятих байтів
  end;
 end; {while}
end;
```

У наведеному коді в потоці крутиться цикл, тим самим ініціюється очікування події порту викликом функції WaitCommEvent(), очікування ж самої цієї події задається функцією WaitForSingleObject(). Для визначення кількості прийнятих символів використовується функція ClearCommError(). Коли кількість прийнятих символів (dwRead) відомо, безпосереднє читання символів виконується функцією ReadFile().

Використовуючи вищеописані викладення, я написав на Borland Delphi 7 клас TComPort для роботи з COM-портами. Для передачі й прийому даних передбачені окремі вікна. Формат переданих даних – рядок. Прийняті дані представляються у вигляді масиву байт.

4.2 Захист розробленого програмного забезпечення

Розроблене програмне забезпечення пропонується захищати від несанкціонованого копіювання та використання прив'язкою до інформації про комп'ютер користувача. Це можна реалізувати через прив'язку до серійного номеру компонентів його комп'ютера і подальшу активацію програмного забезпечення.

В процесі установки програма підраховує код активації – контрольне значення, що однозначно відповідає встановленим комплектуючим комп'ютера і параметрам встановленої ОС. Це значення передається розробнику програми. На його основі розробник генерує ключ активації, що підходить для активації програми лише на зазначеному комп'ютері, копіювання встановлених виконуваних файлів на інший комп'ютер призведе до непрацездатності програми.

Але якщо користувач проводить модернізацію комп'ютера, то втрачає доступ до програмного забезпечення. Цю проблему легко вирішити. Автори багатьох програм в подібних випадках готові дати новий реєстраційний код. Наприклад, Microsoft в Windows XP дозволяв раз в 120 днів генерувати новий

реєстраційний код, але у виняткових випадках, зателефонувавши в службу активації, можна отримати новий код і після закінчення цього терміну.

Як прив'язку до комп'ютера користувача пропонується використовувати серійний номер BIOS материнської плати та серійний номер вінчестера. З метою приховування від користувача даних про захист пропонується розташовувати їх в нерозмічену область жорсткого диску та шифрувати алгоритмом DES.

DES є класичною мережею Фейштеля із двома гілками. Дані шифруються 64-бітними блоками, використовуючи 56-бітний ключ. Алгоритм перетворить за кілька раундів 64-бітний вхід в 64-бітний вихід. Довжина ключа дорівнює 56 бітам. Процес шифрування складається із чотирьох етапів. На першому з них виконується початкова перестановка (IP) 64-бітного вихідного тексту (забілювання), під час якої біти переставляються у відповідності зі стандартною таблицею. Наступний етап складається з 16 раундів однієї й тої ж функції, що використовує операції зрушення й підстановки. На третьому етапі ліва й права половини виходу останньої (16-й) ітерації міняються місцями. Нарешті, на четвертому етапі виконується перестановка IP^{-1} результату, отриманого на третьому етапі. Перестановка IP^{-1} інверсна початковій перестановці.

5 МЕТОДИКА ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

Для установки утиліти запустіть установочний файл відповідно на КПК, CarPC, персональному комп'ютері або ноутбуці й додержуйтеся інструкцій програми установки. Після установки утиліти натисніть «Настроювання». Уведіть серійний номер адаптера й номер COM порту. У Рядку «Автомобіль» виберіть модель вашого автомобіля. У рядку «Система» виберіть систему (АКПП, ABS і ін.) яку ви збираєтеся діагностувати. Якщо для обраного автомобіля доступна робота тільки з однією системою її назва відобразиться в рядку «Система» сірим кольором. Після натискання «ОК» утиліта настроїться на діагностику обраної системи.

Робота з утилітою

1. Підключити адаптер до автомобіля.
2. Запустити утиліту й виберіть модель автомобіля й діагностуєму систему.
3. Включити запалювання автомобіля не заводячи двигун.
4. Натисніть клавішу «Підключитися». Утиліта встановити зв'язок з адаптером і з електронним блоком управління (ЕБУ) діагностуємої системи автомобіля. Перевірити з'єднання можна нажавши «Інформація», на екрані відобразяться відомості про використовуваний адаптер, бортова напруга автомобіля й ідентифікаційні (паспортні) дані ЕБУ діагностуємої системи*
5. Натисніть «Коди несправностей» на екрані відобразяться коди несправностей діагностуємої системи з розшифровкою російською мовою або повідомлення «Коди несправностей не виявлені».
6. Усунете несправність і натисніть «Скидання кодів». По цій команді коди несправностей зітруться з пам'яті ЕБУ діагностуємої системи.
7. Натисніть клавішу «Відключитися».

Якщо несправність не буде усунута, то ЕБУ знову зафіксує відповідний код у своїй пам’яті.

Основне вікно програми наведено на рис. 5.1.

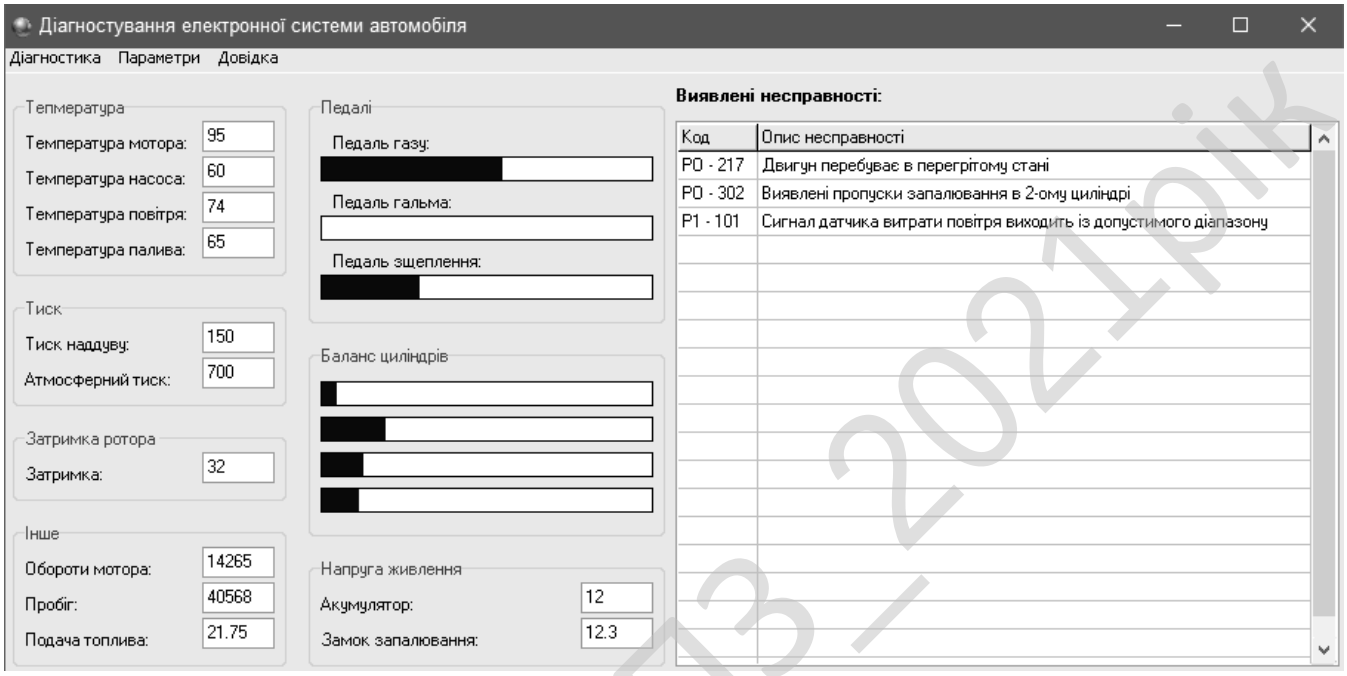

Рисунок 5.1 – Основне вікно програми

6 ОСНОВНІ ВИСНОВКИ

Програмне забезпечення, створене в результаті виконання кваліфікаційної бакалаврської роботи, призначено для системи інформаційної кабельної мережі автомобіля.

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

Рішення завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем інформаційної кабельної мережі автомобіля.

– Досліджена система інформаційної кабельної мережі автомобіля.

– На основі отриманих результатів досліджень створена програмна реалізація системи інформаційної кабельної мережі автомобіля.

Розроблені під час виконання кваліфікаційної бакалаврської роботи алгоритми дозволяють успішно вирішувати завдання інформаційної кабельної мережі автомобіля.

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. Дана мова програмування дозволяє найбільш ефективно обробляти дані призначені для системи інформаційної кабельної мережі автомобіля. 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 10.

Даються необхідні рекомендації з установки розробленого програмного забезпечення.

Для підвищення рівня безпеки запропоновано застосовувати алгоритм DES.

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.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Регулировочные данные для автомобилей с дизельными двигателями 1995-2007 гг. (легковые и небольшие коммерческие автомобили) (125 параметров). Издательство Автодата (лиц.). Ключ на русском языке. Более 900 с.

2. Яковлев В. Ф. Диагностика электронных систем автомобиля. Солон-пресс, 2003 г. – 272 стр.

3. Тюнин А. А. Диагностика электронных систем управления двигателями легковых автомобилей. Практическое пособие. СОЛОН-пРЕСС. 2007 – 352с.

4. Волгин В.В. Справочник по диагностике неисправностей автомобилей. Атласы автомобилей, 2000. – 96с.

5. Диагностические коды неисправностей 2009. Легион-Автодата. – 1312 с.

6. Архангельский А.Я. Программирование в Delphi: Учебник по классическим версиям Delphi / А.Я. Архангельский. - М.: Бином-Пресс, 2013. - 816 с.

7. Белов В.В. Программирование в Delphi: процедурное, объектно-ориентированное, визуальное: Учебное пособие для вузов / В.В. Белов, В.И. Чистякова. - М.: РиС, 2014. - 240 с.

8. Осипов Д. Delphi. Профессиональное программирование / Д. Осипов. - СПб.: Символ-плюс, 2015. - 1056 с.

9. Санников Е. Курс практического программирования в Delphi. Объектно - ориентированное программирование / Е. Санников. - М.: Солон-пресс, 2013. - 188 с.

10. Фаронов В. Delphi. Программирование на языке высокого уровня / В. Фаронов. - СПб.: Питер, 2012. - 640 с.

11. Шенг Лиэнг Интерфейс JNI. Руководство по программированию и

спецификация / Шенг Лиэнг. - М.: ДМК Пресс, 2016. - 849 с.

12. Басов К. Деятельностный подход при проектировании человеко-компьютерного взаимодействия. На примере медицинских интерфейсов / В.Л. Авербух и др. - М.: Ленанд, 2014. - 527 с.

13. Рассел Дж. Интерфейс программирования приложений / Джесси Рассел. - М.: VSD, 2012. - 811 с.

14. Купер А. об интерфейсе. Основы проектирования взаимодействия / Алан Купер. - М.: Символ-плюс, 2006. - 765 с.

15. Борц А.Д., Закин Я.Х., Иванов Ю.В. Диагностика технического состояния автомобиля. М.: Транспорт, 2008. 159 с.

16. Эксплуатация электронных средств технического диагностирования сельскохозяйственной техники. Костенко С.И. и др. – Высшая школа, 2007. 54 с.

17. Delphi C/S 2. Русскоязычная документация; Borland Press - М., 2015. - 321 с.

18. Архангельский А.Я. Программирование в Delphi 6; Бином - М., 2018. - 258 с.

19. Бобровский С. Delphi 5 Учебный курс; СПб: Питер - М., 2017. - 640 с.

20. Григорьев А.Б. О чем не пишут в книгах по Delphi; БХВ -Петербург - М., 2016. - 576 с.

21. Дарахвелидзе П.Г.; Марков, Е.П. Delphi 2005 для Win32 наиболее полное руководство; БХВ-Петербург - М., 2018. - 234 с.

22. Калверт Ч. Базы данных в Delphi 4; Киев: ДиаСофт - М., 2013. - 464 с.

23. Культин Никита Основы программирования в Delphi 7; СПб: БХВ - М., 2014. - 608 с.

24. Марков Е.П.; Никифоров, В.В. Delphi 2005 для .NET; БХВ-Петербург - М., 2017. - 896 с.

25. Понамарев В. Базы данных в Delphi 7. Самоучитель; СПб: Питер - М., 2015. - 224 с.

26. Сван Т. Секреты 32-разрядного программирования в Delphi (+

дискета); Диалектика - М., 2015. - 480 с.

27. Сухарев М.В. Основы Delphi. Профессиональный подход; Наука и техника - М., 2018. - 600 с.

28. Федоров А. Delphi 2.0 для всех; Компьютер-пресс - М., 2013. - 464 с.

29. Шумаков П.В. Delphi 3 и разработка приложений баз данных; Нолидж - М., 2018. - 704 с.

30. Власов В.М. Техническое обслуживание и ремонт автомобилей: Учебник / В.М. Власов. - М.: Академия, 2018. - 352 с.

31. Шестопалов С.К. Устройство, техническое обслуживание и ремонт легковых автомобилей: Учебник / С.К. Шестопалов. - М.: Академия, 2018. - 288 с.

32. Шиловский В.Н. Сервисное обслуживание и ремонт машин и оборудования: Учебное пособие / В.Н. Шиловский, А.В. Питухин, В.М. Костюкевич. - СПб.: Лань, 2019. - 240 с.

33. Осипов Дмитрий Delphi. Программирование для Windows, OS X, iOS и Android / Дмитрий Осипов. - М.: "БХВ-Петербург", 2014. - 464 с.

34. Культин Никита Основы программирования в Delphi 7 / Никита Культин. - М.: БХВ-Петербург, 2013. - 640 с.

35. Основы программирования. Учебник с практикумом / Под ред. Макаровой Н.В. - М.: КноРус, 2017. - 352 с.

36. Дорогов В.Г. Основы программирования на языке С: Учебное пособие / В.Г. Дорогов, Е.Г. Дорогова. - М.: Форум, 2015. - 320 с.

37. Колдаев В.Д. Основы алгоритмизации и программирования: Учебное пособие / В.Д. Колдаев. - М.: Форум, 2015. - 352 с.

38. Макарова, Н.В. Основы программирования. учебник с практикумом - М.: КноРус, 2016. - 112 с.

39. Окулов С.М. Основы программирования / С.М. Окулов. - М.: Бином, 2015. - 336 с.

40. Парфилова Н.И. Программирование: Основы алгоритмизации и программирования: Учебник / Н.И. Парфилова; Под ред. Трусова Б.Г. - М.:

Academia, 2018. - 32 с.

41. Грушецький С.М. Конспект лекцій з основ технічної діагностики автомобілів : [навч. посіб. для студентів інжен. спец. за напрямом підготовки 6.070106 – Автомобільний транспорт, кваліфікація 3710 – Фахівець з автомобільного транспорту] / Грушецький С.М. – Кам'янець-Подільський: ФОП Сисин О.В., 2013. – 340 с.

42. Гради Буч. Объектно-ориентированное программирование с примерами применения на C++. 2-е издание. 558 с.

43. Кьоу Дж. Объектно-ориентированное программирование / Дж. Кьоу, М. Джеанини. - М.: Питер, 2015. - 240 с.

44. Санников Е.В. Курс практического программирования в Delphi. Объектно-ориентированное программирование / Е.В. Санников. - М.: Солон-Пресс, 2017. - 188 с.

45. Хорев, П.Б. Объектно-ориентированное программирование: Учебное пособие / П.Б. Хорев. - М.: Академия, 2018. - 384 с.

46. Лесневский, А. С. Объектно-ориентированное программирование для начинающих / А.С. Лесневский. - М.: Бином. Лаборатория знаний, 2005. - 232 с.

47. Белов В.В. Программирование в Delphi: процедурное, объектно-ориентированное, визуальное: Учебное пособие / В.В. Белов. - М.: ГЛТ, 2009. - 240 с.

48. Вайсфельд М. Объектно-ориентированное мышление / М. Вайсфельд. - М.: Питер, 2014. - 998 с.

49. Амблер С. Гибкие технологии: экстремальное программирование и унифицированный процесс разработки / С. Амблер. - М.: СПб: Питер, 2005. – 416 с.

50. Могилев А. Методы программирования. Компьютерные вычисления / А. Могилев, Л. Листрова. - М.: БХВ-Петербург, 2008. - 320 с.

Додаток А
(обов'язковий)

Технічне завдання

Зміст

1 Найменування та область застосування

Це технічне завдання розповсюджується на розробку системи інформаційної кабельної мережі автомобіля.

2 Підстава для розробки

Підставою для розробки служить завдання на кваліфікаційну бакалаврську роботу, видане на кафедрі кібербезпеки та програмного забезпечення (нак. № 189-02 від 28.12.2020 року).

3 Мета та призначення розробки

Метою кваліфікаційної бакалаврської роботи є розробка програмного забезпечення системи інформаційної кабельної мережі автомобіля.

4 Джерела розробки

Джерелом цієї кваліфікаційної бакалаврської роботи 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;

– розробка програми, що реалізує спроектовані алгоритми роботи системи.

5.2 Показники призначення

Система повинна забезпечувати:

– системи інформаційної кабельної мережі автомобіля;

– 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 10 і з сумісними з цією платформою пристроями і прикладним програмним забезпеченням.

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 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.

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 Перелік документів, що розробляються

– Структурна схема системи – 1 аркуш.
– Функціональна схема системи – 1 аркуш.
– Діаграма процесів – 1 аркуш.
– Блок-схема алгоритму роботи програми – 2 аркуша.
– Пояснювальна записка – 66 аркушів.

8 Етапи розробки

8.1 Збір і обробка інформації по темі кваліфікаційної бакалаврської роботи. Постановка задачі на виконання кваліфікаційної бакалаврської роботи (складання ТЗ).

8.2 Проведення досліджень або експериментальних робіт для уточнення основних положень кваліфікаційної бакалаврської роботи.

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

8.4 Побудова схем взаємодії даних.

8.5 Створення прототипу ПЗ.

8.6 Віднаходження ПЗ, аналіз отриманих результатів.

8.7 Оформлення пояснювальної записки і виконання робіт по графічній частині.

11 Порядок контролю та приймання

11.1 Подання кваліфікаційної бакалаврської роботи на попередній захист 22.05.2021 р.

11.2 Подання кваліфікаційної бакалаврської роботи на захист 11.06.2021 р.

Додаток Б

(обов’язковий)

Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

ЗАТВЕРДЖУЮ

Керівник кваліфікаційної бакалаврської роботи

_____ Петренюк В.І.

Програмне забезпечення системи інформаційної кабельної мережі автомобіля

Лістинг програми

Код документу 12

Носій: CD/DVD-диск

Загальна кількість аркушів: 77

Літера: РП

Кропивницький – 2021 року

Diag_ISO14230.dpr – головний файл проекту

```
program Diag_ISO14230_;

uses
  Forms,
  umain in 'umain.pas' {FMain},
  common in 'common.pas',
  CPort in 'CPort.pas',
  CPortCtl in 'CPortCtl.pas',
  CPortEsc in 'CPortEsc.pas',
  _ISO14230_ in '_ISO14230.pas',
  ubipwm in 'ubipwm.pas' {FBiPWM},
  uwiper in 'uwiper.pas' {FWiper},
  ubridge in 'ubridge.pas' {FBridge},
  ulist in 'ulist.pas' {FList},
  unit_ajoutboitier in 'unit_ajoutboitier.pas' {F_AjoutBoitier},
  unit_configuration in 'unit_configuration.pas' {F_Config},
  unit_connection in 'unit_connection.pas' {F_Connect},
  uonoff in 'uonoff.pas' {FOnOff},
  upwm in 'upwm.pas' {FPwm},
  about in 'about.pas' {Fabout},
  uvalue in 'uvalue.pas' {FValue};

{$R *.res}

begin
  Application.Initialize;
  Application.CreateForm(TFMain, FMain);
  Application.CreateForm(TFBiPWM, FBiPWM);
  Application.CreateForm(TFWiper, FWiper);
  Application.CreateForm(TFBridge, FBridge);
  Application.CreateForm(TFList, FList);
  Application.CreateForm(TF_AjoutBoitier, F_AjoutBoitier);
  Application.CreateForm(TF_Config, F_Config);
  Application.CreateForm(TF_Connect, F_Connect);
  Application.CreateForm(TFOnOff, FOnOff);
  Application.CreateForm(TFPwm, FPwm);
  Application.CreateForm(Tabout, Fabout);
  Application.CreateForm(TFValue, FValue);
  Application.Run;
end.
```

umain.pas – основна програма

```
unit umain;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus,Grids, ComCtrls, unit_connection, CPort,
  common, Buttons, _ISO14230_, unit_Configuration,
  DateUtils, ImgList, ExtCtrls, StdCtrls, XPMan, about;
{
uses
  Classes, SysUtils, LResources, Forms, Controls, Graphics, Dialogs, Menus,
  StdCtrls, ExtCtrls, Grids, ComCtrls, unit_connection, CPort,
  common, Buttons, _ISO14230_, unit_Configuration, EditBtn,u_open,u_save,
  DateUtils,LCLType;
}

const slash = '\';

      Etat_Debut_Connection = 1;
      Etat_Connecte=2;
      Etat_Attente_Reponse =3;

      CT_SensHL = False;
      CT_SensLH = True;



type

  { TFMain }

  TFMain = class(TForm)
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    Group_Debug: TGroupBox;
    Image1: TImage;
    ImageList1: TImageList;
    ImageList2: TImageList;
    I_Attente: TImage;
    MainMenu1: TMainMenu;
    Menu_Langues: TMenuItem;
    Menu_Quitter: TMenuItem;
    Menu_Setting: TMenuItem;
    M_Debug: TMemo;
    OpenDialog1: TOpenDialog;
    Panel1: TPanel;
    Panel2: TPanel;
    SaveDialog1: TSaveDialog;
    Splitter1: TSplitter;
    Menu_Connect: TMenuItem;
    T_Attente: TTimer;
    T_StatusIO: TTimer;
    TreeView1: TTreeView;
    IList_Rose: TImageList;
    IListConnect: TImageList;
    IList_Led: TImageList;
    Panel_Status: TPanel;
    L_Titre: TLabel;
    M_Status: TMemo;
    Panel_Command: TPanel;
    GroupBox4: TGroupBox;
    Label3: TLabel;
    Label4: TLabel;
```

```
E_Sid: TEdit;
E_Data: TEdit;
B_Send: TBitBtn;
M_Hist: TMemo;
Panel_Grille: TPanel;
StringGrid1: TStringGrid;
Panel_Grid_Cde: TPanel;
B_Mode_Maitre: TBitBtn;
B_SaveGrid: TBitBtn;
Panel_Liste_Para: TPanel;
Label1: TLabel;
Label2: TLabel;
E_Fichier_Para: TEdit;
B_Rech_Fichier_Para: TButton;
B_Lecture_Liste: TButton;
B_Sauve_Liste: TButton;
GroupBox3: TGroupBox;
StringGrid2: TStringGrid;
B_Export: TButton;
Panel_Parametre: TPanel;
L_Parametre: TLabel;
L_Numero: TLabel;
L_Para_Valeur: TLabel;
B_Para_Lecture: TButton;
B_Para_Ecriture: TButton;
E_Para_Numero: TEdit;
M_Parametre: TMemo;
E_Para: TEdit;
RadioAffVal: TRadioGroup;
Panel_Value: TPanel;
L_Valeur: TLabel;
RadioAffVal1: TRadioGroup;
E_Valeur: TEdit;
M_Valeur: TMemo;
B_Valeur_Ecriture: TButton;
B_Valeur_Lecture: TButton;
XPManifest1: TXPManifest;
T_Command: TTimer;
procedure B_ExportClick(Sender: TObject);
procedure B_Lecture_ListeClick(Sender: TObject);
procedure B_Mode_MaitreClick(Sender: TObject);
procedure B_Para_EcritureClick(Sender: TObject);
procedure B_Para_LectureClick(Sender: TObject);
procedure B_Rech_Fichier_ParaClick(Sender: TObject);
procedure B_Sauve_ListeClick(Sender: TObject);
procedure B_SaveGridClick(Sender: TObject);
procedure B_SendClick(Sender: TObject);
procedure B_Valeur_EcritureClick(Sender: TObject);
procedure B_Valeur_LectureClick(Sender: TObject);
procedure E_ParaChange(Sender: TObject);
procedure FormClose(Sender: TObject; var CloseAction: TCloseAction);
procedure FormCreate(Sender: TObject);
procedure FormDestroy(Sender: TObject);

procedure Menu_ConnectClick(Sender: TObject);
procedure Menu_QuitterClick(Sender: TObject);
procedure Menu_SettingClick(Sender: TObject);
procedure Panel1Click(Sender: TObject);
procedure RadioAffVal1ChangeBounds(Sender: TObject);
procedure RadioAffValChangeBounds(Sender: TObject);
procedure StringGrid1Click(Sender: TObject);
procedure StringGrid1DrawCell(Sender: TObject; aCol, aRow: Integer;
  aRect: TRect; aState: TGridDrawState);
procedure StringGrid1Resize(Sender: TObject);
procedure T_AttenteTimer(Sender: TObject);
procedure T_StatusIOTimer(Sender: TObject);
procedure TreeView1Click(Sender: TObject);
procedure Menu_langueAddClick(Sender: TObject);
procedure T_CommandTimer(Sender: TObject);
```

```
  private
    { private declarations }
    num_image : integer;
    diag_actif : boolean;
    ledverteOn,ledVerteOFF,ledRougeOn,ledamberon,ledamberOff,ledRougeOff :
TBitmap;
    cpt_erreur : integer;
    Grille_Up,nb_Row_visible : integer; // Дозволяє дізнатися діапазон видимих
входів та виходів

    procedure _ISO14230_Connect(Sender: TObject);
    procedure _ISO14230_Disconnect(Sender: TObject);
    procedure _ISO14230_Error(Sender: TObject; Erreur: integer; MsgErreur:
string
      );
    procedure _ISO14230_ReceiveTrame(Sender: TObject);
    procedure _ISO14230_Status(Sender: TObject; Messages: string);
    procedure _ISO14230_Step(Sender: TObject; Messages: string);
  public
    { public declarations }
    MainExit : boolean;
    BeginExit : Boolean;
    Variable : array[0..9] of integer;
    Table : array[0..9] of TStringList;
    Type_Connection: integer;
    Debug : integer;
    State : integer;
    TypeAff : integer; // Цей тип визначає видимість на екрані: 0->StringGrid
    Vehicule : TStringList;
    MenuList : TStringList;
    ActualMenu : TStringList;
    ExtendedMenu : TStringList;
    ClickList : TStringList;
    ligneMenu : integer;
    Actual_ISO14230_ : TStringList;
    ColorList : TStringList;
    Execution : boolean;
    TimeRepeat : integer;
    MenuName : string;
    TitleMenu : string;
    Fichier : TStringlist;
    nom_fichier : string;
    CdeReadParam,CdeWriteParam : string;
    CdeValueLength : string;
    CdeValueAction : integer;
    ClickTree : boolean; // при натисканні на дереві, ви не робите якісь речі
    TypeCdeActif : integer;
    LigneGrid2 : integer;
    ClickActif : Boolean;
    ClickGlobal : string; // відправлення команди, чи слід активізувати активну
діагностику
    ClickExit : string; // відправлення команди, чи не слід активізувати активну
діагностику
    NoReceive : boolean;
    InIOTimer : boolean;
    ComPort1: TComPort;
    _ISO14230_: T_ISO14230_;
    procedure GlobalReset;
    procedure Search_Item(ItemSearch : string;ListSearch : TStringList;var
searchList: TStringList);
    Procedure Attente_Start;
    Procedure Attente_Stop;
    Function Cde2Send(CDE: string): integer;
    Procedure Create_Menu_Langue;
    Procedure Init_Message;
    Procedure Ajust_StringList(Count : integer; VAR Liste : TSTringList);
  end;

var
```

```
  FMain: TFMain;
  M : Array[0..255] of TMenuItem;
  Nb_Langue : integer;


implementation

{$R *.dfm}
{ TFMain }

uses upwm,ubridge,uwiper,uonoff,ubipwm,uvalue,ulist;

//*****************************************************************************
//Кнопка у вікні для з'єднання з бортовим комп'ютером автомобіля
//*****************************************************************************
procedure TFMain.Menu_ConnectClick(Sender: TObject);
var i : integer;
    textel : string;
    noeud : TTreeNode;
    ListConnect : TStringList;
    type_diagnosticsess : byte;
begin
  FMain.cpt_erreur:=0;
  F_Connect.Valid:=False;

  try
    ListConnect :=TStringList.Create;
    ListConnect.Clear;
    If FMain.Menu_Connect.Caption='&'+MenuProg[CMP_Connecter] then
     begin
       if F_Connect.execute then
          begin
            type_diagnosticsess:=$81;

FMain.Vehicule.LoadFromFile(Repertoire_courant+'Vehicule\'+F_Connect.Vehicule_Na
me);
          FMain.Search_Item('CONNECTION',FMain.Vehicule,ListConnect);
          Type_Connection:=0;
            FMain.TreeView1.Enabled:=False;
          for i:=0 to ListConnect.Count-1 do
            begin
              textel:=ListConnect.Strings[i];
            if uppercase(copy(textel,1,5))='INIT=' then
              begin
                delete(textel,1,5);
                if uppercase(textel)='FAST' then
                      begin
                        Type_Connection:=1;
                        FMain.ISO14230.ModeConnection:=1;
                      end;
              end else
            if uppercase(copy(textel,1,7))='ECU_ID=' then
              begin
                delete(textel,1,7);
                FMain.ISO14230.Target:=hextodec(textel);
                end else
            if uppercase(copy(textel,1,8))='TOOL_ID=' then
              begin
                delete(textel,1,8);
                FMain.ISO14230.Source:=hextodec(textel);
              end else
            if uppercase(copy(textel,1,6))='SPEED=' then
              begin
                delete(textel,1,6);
                try
                  FMain.ISO14230.DiagnosticSpeed:=Strtoint(textel);
                except
                  FMain.ISO14230.DiagnosticSpeed:=10400;
                end;
```

```
              end else
                if uppercase(copy(textel,1,8))='SESSION=' then
              begin
                delete(textel,1,8);
                type_diagnosticsess:=hextodec(textel);
                end else
                if uppercase(copy(textel,1,14))='TESTERPRESENT=' then
              begin
                delete(textel,1,14);
                FMain.ISO14230.Msg_TesterPresent:=textel;
                end;
            end;
          // Меню діагностики ITEM
          MenuList.Clear;
            FMain.ExtendedMenu.Clear;
          FMain.Treeview1.Items.Clear;
          Noeud:=FMain.TreeView1.Items.Add(nil,(MenuProg[CMP_Status]));
            Noeud.ImageIndex:=5;
            Noeud:=FMain.TreeView1.Items.Add(nil,(MenuProg[CMP_Command]));
            Noeud.ImageIndex:=1;

            FMain.Panel_grille.Visible:=False;
            FMain.Panel_Parametre.Visible:=False;
            FMain.Panel_Status.Visible:=True;
            MenuI:=1;
            Sous_MenuI:=0;
            FMain.ComPort1.Port:=F_Connect.Combo_Port.Text;
            FMain.M_Status.Clear;
            DateTimeToString(textel,'h-nn-ss-zzz',Now);
            FMain.M_Status.Lines.Add(textel+(MenuProg[CMP_TestConnect]));
            state:=1; // Спроба підключення до COM-порту
            FMain.Attente_Start;
            FMain.ISO14230.Connect;
            FMain.Attente_Stop;
            if State<>0 then
               begin
                 // З'єднаня з  COM-портом встановлено
                 State:=2;
                 FMain.Attente_Start;
                 FMain.M_Debug.Lines.Add('Lancement StartDiagnosis');
                 FMain.ISO14230.StartDiagnosis(type_diagnosticsess);
               end;
          end;
      end else
      begin
        FMain.GlobalReset;
      end;
  finally
      ListConnect.Free;
  end;
end;

//***************************************************************************
//Закриття форми
//***************************************************************************
procedure TFMain.FormClose(Sender: TObject; var CloseAction: TCloseAction);
begin
  MainExit:=True;
  FMain.Execution:=True;
  FMain.T_StatusIO.Enabled:=False;
  //if FMain.ComPort1.Connected then FMain.ComPort1.Close;
end;

//***************************************************************************
procedure TFMain.B_Para_LectureClick(Sender: TObject);
var n_para : integer;
    textel : string;
begin
  try
```

```
    n_para:=hextodec(uppercase(FMain.E_Para_Numero.Text));
  except

Application.MessageBox(PCHAR(MenuProg[CMP_NumberNotHex]),PCHAR(MenuProg[CMP_Erro
r]),0);
    exit;
  end;
  if n_para<=255 then
     begin
       textel:=FMain.CdeReadParam;
       delete(textel,1,1);
       if length(textel)=0 then exit;
       FMain.Cde2Send(textel);
       FMain.ISO14230.FData[FMain.ISO14230.SendLength]:=n_para;
       FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+1;
       // Конфігурація системи
       FMain.CdeValueAction:=1;
       FMain.ISO14230.SendResponse;
       FMain.Attente_Start;
     end else
Application.MessageBox(PCHAR(MenuProg[CMP_ParamOutRange]),PCHAR(MenuProg[CMP_Err
or]),0);
end;

//*****************************************************************************
procedure TFMain.B_Rech_Fichier_ParaClick(Sender: TObject);
var textel : string;
    i : integer;
    erreur : integer;
begin
  erreur:=1;
  //if FOpen.Execute then
  FMain.OpenDialog1.Filter:='Fichier XML|*.xml';
  if FMain.OpenDialog1.Execute then
     begin
       //FMain.E_Fichier_Para.Text:=FOpen.FileName;
       FMain.E_Fichier_Para.Text:=FMain.OpenDialog1.FileName;
       AssignFile(fichier_txt,FMain.E_Fichier_Para.Text);
       reset(fichier_txt);
       FMain.StringGrid2.RowCount:=1;
       FMain.StringGrid2.Cells[0,0]:=MenuProg[CMP_HexaAdress];
       FMain.StringGrid2.Cells[1,0]:=MenuProg[CMP_Name];
       FMAin.StringGrid2.Cells[2,0]:=MenuProg[CMP_PhysicalAdress];
       FMain.StringGrid2.Cells[3,0]:=MenuProg[CMP_Length];
       FMain.StringGrid2.Cells[4,0]:=MenuProg[CMP_Position];
       FMain.StringGrid2.Cells[5,0]:=MenuProg[CMP_Fault];
       FMain.StringGrid2.Cells[6,0]:=MenuProg[CMP_Min];
       FMain.StringGrid2.Cells[7,0]:=menuProg[CMP_Max];
       FMain.StringGrid2.Cells[8,0]:=MenuProg[CMP_DecimalValue];
       For i:=0 to 8 do

FMain.StringGrid2.ColWidths[i]:=FMain.StringGrid2.Canvas.TextWidth(FMain.StringG
rid2.Cells[i,0])+10;
       while eof(fichier_txt)=false do
         begin
           readln(fichier_txt,textel);
           i:=pos('PARM ID',textel);
           if i>0 then
              begin
                erreur:=0;
                delete(textel,1,i+9);
                i:=pos(']',textel);
                FMain.StringGrid2.RowCount:=FMain.StringGrid2.RowCount+1;
                FMain.StringGrid2.Cells[0,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[1,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
```

```
                delete(textel,1,i);
                i:=pos('Адреса=',textel);
                delete(textel,1,i+8);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[2,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
                i:=pos('Розмір=',textel);
                delete(textel,1,i+5);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[3,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
                i:=pos('Позиція=',textel);
                delete(textel,1,i+9);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[4,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
                i:=pos('Інше=',textel);
                delete(textel,1,i+8);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[5,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
                i:=pos('Мінімум =',textel);
                delete(textel,1,i+4);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[6,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
                i:=pos('Максимум =',textel);
                delete(textel,1,i+4);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[7,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
                i:=pos('Значення =',textel);
                delete(textel,1,i+6);
                i:=pos('"',textel);
                FMain.StringGrid2.Cells[8,FMain.StringGrid2.RowCount-
1]:=copy(textel,1,i-1);
                delete(textel,1,i);
              end;
           For i:=0 to 8 do
             begin
               if
(FMain.StringGrid2.ColWidths[i]<FMain.StringGrid2.Canvas.TextWidth(FMain.StringG
rid2.Cells[i,FMain.StringGrid2.RowCount-1])+10)
                  then
FMain.StringGrid2.ColWidths[i]:=FMain.StringGrid2.Canvas.TextWidth(FMain.StringG
rid2.Cells[i,FMain.StringGrid2.RowCount-1])+10;
             end;
         end;
       if FMain.StringGrid2.RowCount>1 then FMain.StringGrid2.FixedRows:=1;
       CloseFile(fichier_txt);
       if erreur=1 then
          begin

Application.MessageBox(PCHAR(MenuProg[CMP_WrongFile]),PCHAR(MenuProg[CMP_Error])
,0);
            exit;
          end;
     end;
end;

//***************************************************************************
procedure TFMain.B_Sauve_ListeClick(Sender: TObject);
var n_para,val_para : integer;
```

```
    textel : string;
    k : integer;
    l,lg : integer; // використовується для значення вмісту октету
begin
  if FMain.StringGrid2.RowCount<2 then exit;
  FMain.TypeCdeActif:=2;
  try
    n_para:=hextodec(uppercase(FMain.StringGrid2.Cells[0,1]));
  except

Application.MessageBox(PCHAR(MenuProg[CMP_NumberNotHex]),PCHAR(MenuProg[CMP_Erro
r]),0);
    exit;
  end;
  try
    val_para:=strtoint(FMain.StringGrid2.Cells[8,1]);
  except

Application.MessageBox(PCHAR(MenuProg[CMP_ValueNotHex]),PCHAR(MenuProg[CMP_Error
]),0);
    exit;
  end;
  if n_para<=255 then
     begin
       textel:=FMain.CdeWriteParam;
       if textel[1]='L' then l:=0 else l:=1;
       delete(textel,1,1);
       FMain.Cde2Send(textel);
       FMain.ISO14230.FData[FMain.ISO14230.SendLength]:=n_para;
       FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+1;
       // записати значення в буфер
       try
         textel:=dectohex(val_para);
       except
         textel:='00';
       end;
       if (FMain.StringGrid2.Cells[3,1]='16')
          and (length(textel)=2)
          then textel:='00'+textel;
       if (l=0) and (length(textel)>2) then
         begin
           textel:=copy(textel,3,2)+copy(textel,1,2);
         end;
       lg:=(length(textel) div 2);
       for k:=0 to lg-1 do
         begin

FMain.ISO14230.FData[k+FMain.ISO14230.SendLength]:=hextodec(copy(textel,1,2));
         delete(textel,1,2);
         end;

       FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+lg;
       // Конфігурація системи
       //FMain.Timer_Maintient.Enabled:=False;
       FMain.LigneGrid2:=1;
       FMain.ISO14230.SendResponse;
       FMain.Attente_Start;
     end else
Application.MessageBox(PCHAR(MenuProg[CMP_ParamOutRange]),PCHAR(MenuProg[CMP_Err
or]),0);
end;

//*****************************************************************************
procedure TFMain.B_SaveGridClick(Sender: TObject);
var i,j : integer;
    textel : string;
begin
  //FSave.RadioGroup1.Visible:=false;
  //FSave.RadioGroup1.ItemIndex:=1;
```

```
  FMain.OpenDialog1.Filter:='Fichier Excel (CSV)|*.csv';
  if FMain.OpenDialog1.execute then
    begin
      FMain.nom_fichier:='';
      Fichier.Clear;
      for i:=0 to FMain.StringGrid1.RowCount-1 do
        begin
          textel:='';
          for j:=0 to FMain.StringGrid1.ColCount-1 do
            textel:=textel+FMain.StringGrid1.Cells[j,i]+';';
          Fichier.Add(textel);
        end;
      textel:=lowercase(FMain.OpenDialog1.FileName);
      i:=pos('.csv',textel);
      if i=0 then textel:=textel+'.csv';
      Fichier.SaveToFile(textel);
    end;
  //FSave.RadioGroup1.Visible:=True;
end;

//***************************************************************************
procedure TFMain.B_SendClick(Sender: TObject);
var datetxt : string;
    textel : string;
    i : integer;

begin
  DateTimeToString(datetxt,'h : nn m ss sec zzz ms - ',Now);
  FMain.E_Sid.Text:=uppercase(FMain.E_Sid.Text);
  FMain.E_Data.Text:=uppercase(FMain.E_Data.Text);
  try
    FMain.ISO14230.Sid:=hextodec(trim(FMain.E_Sid.Text));
  except
    FMain.M_Hist.Lines.Add(datetxt+MenuProg[CMP_SidError]);
    exit;
  end;
  textel:=FMain.E_Data.Text;
  textel:=trim(textel);
  i:=0;
  while length(textel)>0 do
    begin
      FMain.ISO14230.FData[i]:=hextodec(copy(textel,1,2));
      i:=i+1;
      delete(textel,1,3);
    end;
  FMain.ISO14230.SendLength:=i;
  FMain.Execution:=False;
  textel:=dectohex(FMain.ISO14230.Sid)+' ';
  for i:=0 to FMain.ISO14230.SendLength-1 do
textel:=textel+dectohex(FMain.ISO14230.FData[i])+' ';
  FMain.M_Hist.Lines.Add(datetxt+MenuProg[CMP_Send]+' - '+textel);
  FMain.Attente_Start;
  FMain.ISO14230.SendResponse;
  While (Execution=False) and (MainExit=False) do Application.ProcessMessages;
  if MainExit then Exit;
  textel:=dectohex(FMain.ISO14230.SidReceive)+' ';
  for i:=0 to FMain.ISO14230.ReceiveLength-1 do
textel:=textel+dectohex(FMain.ISO14230.TrameRx[i])+' ';
  FMain.M_Hist.Lines.Add(datetxt+MenuProg[CMP_Receive]+' - '+textel);
end;

//***************************************************************************
{procedure TFMain.B_UpMouseDown(Sender: TOBject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if FMain.Grille_Up>0 then
     begin
       FMain.Grille_Up:=FMain.Grille_Up-1;
       FMain.rafraichir_diag;
```

```
     end;
end;
}
//***************************************************************************
procedure TFMain.B_Valeur_EcritureClick(Sender: TObject);
var val_para : integer;
    textel,textel1 : string;
    i,j : integer;
    d : array[0..20] of byte;
    nd : integer;
    sens_octet : integer;
    Nb_BitValue : integer;
begin
  if FMain.RadioAffVal.ItemIndex=0 then
     begin
       try
         val_para:=strtoint(FMain.E_Valeur.Text);
       except

Application.MessageBox(PCHAR(MenuProg[CMP_ValueNotHex]),PCHAR(MenuProg[CMP_Error
]),0);
         exit;
       end;
     end else
     begin
       try
         val_para:=hextodec(uppercase(FMain.E_Valeur.Text));
       except

Application.MessageBox(PCHAR(MenuProg[CMP_ValueNotHex]),PCHAR(MenuProg[CMP_Error
]),0);
         exit;
       end;
     end;
  textel:=FMain.CdeWriteParam;
  if textel[1]='L' then sens_octet:=0 else sens_octet:=1;
  delete(textel,1,1);
  if length(textel)=0 then exit;
  Fmain.Cde2Send(textel);
  j:=FMain.ISO14230.SendLength;
  textel:=dectohex(val_para);
  if Length(CdeValueLength)>0 then
    begin
      try
        Nb_BitValue:=strtoint(CdeValueLength);
        while length(textel)<(Nb_BitValue div 4) do textel:='00'+textel;
      except
      end;
    end;
  nd:=0;
  while length(textel)>0 do
    begin
      textel1:=copy(textel,1,2);
      d[nd]:=hextodec(textel1);
      nd:=nd+1;
      delete(textel,1,2);
    end;
  FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+nd;
  //j:=2;
  if sens_octet=0 then
    begin
      for i:=nd-1 downto 0 do
        begin
          FMain.ISO14230.FData[j]:=d[i];
          j:=j+1;
        end;
    end else
    begin
      for i:=0 to nd-1 do
```

```
        begin
          FMain.ISO14230.FData[j]:=d[i];
          j:=j+1;
        end;
    end;
  // Конфігурація системи
  //FMain.Timer_Maintient.Enabled:=False;
  FMain.CdeValueAction:=2;
  FMain.ISO14230.SendResponse;
  FMain.Attente_Start;
end;

//***************************************************************************
procedure TFMain.B_Valeur_LectureClick(Sender: TObject);
var textel : string;
begin
  textel:=FMain.CdeReadParam;
  delete(textel,1,1);
  if length(textel)=0 then exit;
  FMain.CdeValueAction:=1;
  FMain.Cde2Send(textel);
  // Конфігурація системи
  FMain.ISO14230.SendResponse;
  FMain.Attente_Start;
end;

//***************************************************************************
procedure TFMain.E_ParaChange(Sender: TObject);
begin

end;

//***************************************************************************
procedure TFMain.B_Para_EcritureClick(Sender: TObject);
var n_para,val_para : integer;
    textel,textel1 : string;
    i,j : integer;
    d : array[0..20] of byte;
    nd : integer;
    sens_octet : integer;
begin
  try
    n_para:=hextodec(uppercase(FMain.E_Para_Numero.Text));
  except

Application.MessageBox(PCHAR(MenuProg[CMP_NumberNotHex]),PCHAR(MenuProg[CMP_Erro
r]),0);
    exit;
  end;
  if FMain.RadioAffVal.ItemIndex=0 then
    begin
      try
        val_para:=strtoint(FMain.E_Para.Text);
      except

Application.MessageBox(PCHAR(MenuProg[CMP_ValueNotHex]),PCHAR(MenuProg[CMP_Error
]),0);
        exit;
      end;
    end else
    begin
      try
        val_para:=hextodec(uppercase(FMain.E_Para.Text));
      except

Application.MessageBox(PCHAR(MenuProg[CMP_ValueNotHex]),PCHAR(MenuProg[CMP_Error
]),0);
        exit;
      end;
```

```
      end;

    if n_para<=255 then
      begin
        textel:=FMain.CdeWriteParam;
        if textel[1]='L' then sens_octet:=0 else sens_octet:=1;
        delete(textel,1,1);
        if length(textel)=0 then exit;
        Fmain.Cde2Send(textel);
        FMain.ISO14230.FData[FMain.ISO14230.SendLength]:=n_para;
        FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+1;
        j:=FMain.ISO14230.SendLength;
        textel:=dectohex(val_para);
        nd:=0;
        while length(textel)>0 do
          begin
            textel1:=copy(textel,1,2);
            d[nd]:=hextodec(textel1);
            nd:=nd+1;
            delete(textel,1,2);
          end;
        FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+nd;
        //j:=2;
        if sens_octet=0 then
          begin
            for i:=nd-1 downto 0 do
              begin
                FMain.ISO14230.FData[j]:=d[i];
                j:=j+1;
              end;
          end else
          begin
            for i:=0 to nd-1 do
              begin
                FMain.ISO14230.FData[j]:=d[i];
                j:=j+1;
              end;
          end;
        // Конфігурація системи
        //FMain.Timer_Maintient.Enabled:=False;
        FMain.CdeValueAction:=2;
        FMain.ISO14230.SendResponse;
        FMain.Attente_Start;
      end else
Application.MessageBox(PCHAR(MenuProg[CMP_ParamOutRange]),PCHAR(MenuProg[CMP_Err
or]),0);
end;

//***************************************************************************
procedure TFMain.B_ExportClick(Sender: TObject);
var i,j:integer;
    textel : string;
begin
  //If FSave.Execute then
  FMain.SaveDialog1.Filter:='Fichier XML|*.xml|Fichier Excel (CSV)|*.csv';
  FMain.SaveDialog1.FilterIndex:=0;
  if FMain.SaveDialog1.Execute then
     begin
       if FMain.SaveDialog1.FilterIndex<1 then FMain.SaveDialog1.FilterIndex:=0;
       textel:=FMain.SaveDialog1.FileName;
       case FMain.SaveDialog1.FilterIndex of
         1 : begin
           i:=pos('.xml',lowercase(textel));
           if i<1 then
FMain.SaveDialog1.FileName:=FMain.SaveDialog1.FileName+'.xml';
         end;
         2 : begin
           i:=pos('.csv',lowercase(textel));
```

```
           if i<1 then
FMain.SaveDialog1.FileName:=FMain.SaveDialog1.FileName+'.csv';
         end;
       end;
       if pos('.xml',lowercase(FMain.SaveDialog1.FileName))<1 then
          begin
            AssignFile(fichier_txt,FMain.SaveDialog1.FileName);
            Rewrite(fichier_txt);
            Writeln(fichier_txt,'Таблиця конфігурації'+dateTimeToStr(Now));
            Writeln(fichier_txt,'');
            for i:=0 to FMain.StringGrid2.RowCount-1 do
              begin
                textel:='';
                for j:=0 to FMain.StringGrid2.ColCount-1 do
                    begin
                      textel:=textel+FMain.StringGrid2.Cells[j,i]+';';
                    end;
                writeln(fichier_txt,textel);
              end;
            Closefile(fichier_txt);
          end else
          begin // sauve xml
            AssignFile(fichier_txt,FMain.SaveDialog1.FileName);
            Rewrite(fichier_txt);
            Writeln(fichier_txt,'<?xml:lang="ru" version="1.0"
standalone="no"?>');
            Writeln(fichier_txt,'<!DOCTYPE ParamEEPROMDescr SYSTEM
"ParamDesc.dtd">');
            Writeln(fichier_txt,'');
            Writeln(fichier_txt,'<ParamEEPROMDescr>');
            Writeln(fichier_txt,'');
            Writeln(fichier_txt,'   <ParamEEPROM-List>');
            Writeln(fichier_txt,'');
            for i:=1 to FMain.StringGrid2.RowCount-1 do
              begin
                textel:='     <ParamEEPROM-PARM ID="['
                      +FMain.StringGrid2.Cells[0,i]
                      +'] '
                      +FMain.StringGrid2.Cells[1,i]
                      +'" ADDRESS="'
                      +FMain.StringGrid2.Cells[2,i]
                      +'" SIZE="'
                      +FMain.StringGrid2.Cells[3,i]
                      +'" POSITION="'
                      +FMain.StringGrid2.Cells[4,i]
                      +'" DEFAULT="'
                      +FMain.StringGrid2.Cells[5,i]
                      +'" MIN="'
                      +FMain.StringGrid2.Cells[6,i]
                      +'" MAX="'
                      +FMain.StringGrid2.Cells[7,i]
                      +'" VALUE="'
                      +FMain.StringGrid2.Cells[8,i]
                      +'" />'
                      ;

                writeln(fichier_txt,textel);
              end;
            Writeln(fichier_txt,'');
            Writeln(fichier_txt,'   </ParamEEPROM-List>');
            Writeln(fichier_txt,'');
            Writeln(fichier_txt,'</ParamEEPROMDescr>');
            Writeln(fichier_txt,'');
            Closefile(fichier_txt);
          end;
     end;
end;
```

```
{
procedure TFMain.B_DownMouseDown(Sender: TOBject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
  if FMain.Grille_Up+FMain.nb_Row_visible<FMain.StringGrid3.RowCount then
     begin
       FMain.Grille_Up:=FMain.Grille_Up+1;
       FMain.rafraichir_diag;
     end;
end;
}
//***************************************************************************
procedure TFMain.B_Lecture_ListeClick(Sender: TObject);
var n_para : integer;
    textel : string;
begin
  if FMain.StringGrid2.RowCount<2 then exit;
      try
        n_para:=hextodec(uppercase(FMain.StringGrid2.Cells[0,1]));
      except

Application.MessageBox(PCHAR(MenuProg[CMP_NumberNotHex]),PCHAR(MenuProg[CMP_Erro
r]),0);
        exit;
      end;
      if n_para<=255 then
         begin
           textel:=FMain.CdeReadParam;
           delete(textel,1,1);
           FMain.Cde2Send(textel);
           FMain.ISO14230.FData[FMain.ISO14230.SendLength]:=n_para;
           FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+1;

           // Конфігурація системи
           //FMain.Timer_Maintient.Enabled:=False;
           FMain.LigneGrid2:=1;
           FMain.TypeCdeActif:=1;
           FMain.ISO14230.SendResponse;
           FMain.Attente_Start;
         end;
end;

//***************************************************************************
procedure TFMain.B_Mode_MaitreClick(Sender: TObject);
begin
  while (FMain.ISO14230.EndReceive=false)
    and (MainExit=False) do application.ProcessMessages;
  if MainExit=True then Exit;

  FMain.Cde2Send(FMain.ClickExit);
  FMain.Execution:=False;
  FMain.NoReceive:=True;
  FMain.ISO14230.SendResponse;
  FMain.Attente_Start;
  FMain.B_Mode_Maitre.Visible:=False;
  FMain.ClickActif:=False;
  if FBiPWM.Actif then FBiPwm.Close;
  if FPwm.Actif then FPWM.Close;
  if FBridge.Actif then FBridge.Close;
  if FOnOff.Actif then FOnOff.Close;
  if FValue.Actif then FValue.Close;
  if FWiper.Actif then FWiper.Close;
end;

//***************************************************************************
procedure TFMain.FormCreate(Sender: TObject);
var i,j : integer;
    textel : string;
begin
```

```
  MainExit:=False;
  ledverteOn:=Tbitmap.Create;
  ledVerteOFF:=Tbitmap.Create;
  ledRougeOn:=Tbitmap.Create;
  ledRougeOff:=Tbitmap.Create;
  FMain.IList_Led.GetBitmap(0,FMain.ledVerteOFF);
  FMain.IList_Led.GetBitmap(3,FMain.ledVerteON);
  FMain.IList_Led.GetBitmap(1,FMain.ledRougeON);
  FMain.IList_Led.GetBitmap(2,FMain.ledRougeOFF);
  ledamberOn:=Tbitmap.Create;
  ledamberOff:=Tbitmap.Create;
  //ledamberOn.LoadFromLazarusResource('LEDpurpleon');
  //ledamberOff.LoadFromLazarusResource('LEDpurpleoff');
  FMain.ComPort1:=TComPort.Create(Self);
  FMain.ISO14230_:=T_ISO14230.Create(Self);
  FMain.ISO14230.Comport:=FMain.ComPort1;
  FMain.ISO14230.OnReceiveTrame:=FMain.ISO14230_ReceiveTrame;
  FMain.ISO14230.OnError:=FMain.ISO14230_Error;
  FMain.ISO14230.OnStep:=FMain.ISO14230_Step;
  FMain.ISO14230.OnConnect:=FMain.ISO14230_Connect;
  FMain.ISO14230.OnDisconnect:=FMain.ISO14230_Disconnect;
  FMain.ISO14230.OnStatus:=FMain.ISO14230_Status;
  // відновлення поточного каталогу
  textel:=Application.ExeName;
  i:=length(textel);
  while (i>1)and (textel[i]<>'\') and (textel[i]<>'/') do i:=i-1;
  Repertoire_courant:=copy(textel,1,i); // copie du repertoire courant avec le
slash de fin
  if FileExists(Repertoire_courant+'_ISO14230.ini')
    then FMain.ISO14230.LoadConfiguration(Repertoire_courant+'_ISO14230.ini')
    else FMain.ISO14230.SaveConfiguration(Repertoire_courant+'_ISO14230.ini');
  if not FileExists(Repertoire_courant+'languages\ukrainian.ISO14230_')
    then
FMain.ISO14230.SaveMessageTXT(Repertoire_courant+'languages\ukrainian.ISO14230_'
);
  Fichier := TStringlist.Create;
  Vehicule:=TStringList.Create;
  MenuList:=TStringList.Create;
  ActualMenu := TStringList.Create;
  Actual_ISO14230_ := TStringList.Create;
  FMain.ExtendedMenu:=TStringList.Create;
  FMain.ClickList:=TStringList.Create;
  FMain.ColorList := TStringList.Create;
  MenuTextInit;
  if not FileExists(Repertoire_courant+'languages\ukrainian.dia')
    then Writelanguage(Repertoire_courant+'languages\ukrainian.dia');
  FMain.Create_Menu_Langue;
  //
  if FileExists(Repertoire_courant+'Diag_ISO14230.ini') then
    begin
     Fichier.LoadFromFile(Repertoire_Courant+'Diag_ISO14230.ini');
      for i:=0 to Fichier.Count-1 do
        begin
          textel:=fichier.Strings[i];
          if (length(textel)>0) and (copy(textel,1,2)<>'//')
            then begin
              if copy(textel,1,5)='LANG=' then
                begin
                  delete(textel,1,5);
                  textel:=trim(textel);
                end else
              if copy(textel,1,6)='DEBUG=' then
                begin
                  delete(textel,1,6);
                  textel:=trim(textel);
                  try
                    FMain.Debug:=StrtoInt(textel);
                  except
                    FMain.Debug:=0;
```

```
                end;
              end;
            end;
        end;
      j:=-1;
      i:=0;
      while (j=-1) and (i<Nb_Langue) do
        if uppercase(M[i].Caption)=uppercase(textel) then j:=i else i:=i+1;
      if j>-1 then
        begin
          ReadLanguage(Repertoire_courant+'Languages\'+textel+'.dia');

FMain.ISO14230.LoadMessageTXT(Repertoire_courant+'Languages\'+textel+'.ISO14230_
');
          M[j].Checked:=True;
        end;
    end else
    begin
      Fichier.Clear;
      Fichier.Add('// Diag_ISO14230_ конфігураційний файл');
      Fichier.Add('');
      Fichier.Add('LANG=ukrainian');
      Fichier.Add('DEBUG=0');
      Fichier.SaveToFile(Repertoire_Courant+'Diag_ISO14230.ini');
      j:=-1;
      i:=0;
      while (j=-1) and (i<Nb_Langue) do
        if uppercase(M[i].Caption)='UKRAINIAN' then j:=i else i:=i+1;
      if j>-1 then
        begin
          M[j].Checked:=True;
        end;
    end;
  //
  Fmain.Init_Message;
  //FMain.StringGrid3.RowCount:=0;
  FMain.StringGrid1.RowCount:=0;
  if FMain.Debug=0
    then FMain.Group_Debug.Visible:=False
    else FMain.Group_Debug.Visible:=True;
  FMain.GlobalReset;
end;

//***************************************************************************
procedure TFMain.FormDestroy(Sender: TObject);
var i : integer;
begin
  FMain.Vehicule.Free;
  FMain.MenuList.Free;
  Fichier.Free;
  ActualMenu.Free;
  Actual_ISO14230.Free;
  FMain.ExtendedMenu.Free;
  FMain.ClickList.Free;
  FMain.ColorList.Free;
  ledverteOn.Free;
  ledVerteOFF.Free;
  ledRougeOn.free;
  ledRougeOff.free;
  ledAmberOn.free;
  ledAmberOff.free;
  FMain.ComPort1.Free;
  FMain.ISO14230.Free;
  //master.Free;
  for i:=0 to 9 do if Table[i]<>nil then Table[i].Free;
end;

//***************************************************************************
procedure TFMain.ISO14230_Connect(Sender: TObject);
```

```
var textel,textel1 : string;
    liste_connection : TStringList;
    ListCommand : TStringList;
    cond : string; // вимога повторити команду
    cde : string; // команда відправки
    i,j,k,l,m,n : integer;
    outdoor : boolean;
    byteRule: string;
    DebutByte,FinByte,DebutBit,FinBit : integer;
    valeur,val_cond : integer;
    f_Byte,f_Bit : string;
    mask : byte;
    noeud : TTreeNode;
    n_ligne,max_val,min_val : integer;
begin
  DateTimeToString(textel,'h:nn.ss.zzz',Now);
  FMain.M_Status.Lines.Add(textel+MenuProg[CMP_ConnectOK]);
  FMain.M_Debug.Lines.Add(textel+MenuProg[CMP_ConnectOK]);
  FMain.Image1.Picture.Bitmap:=nil;
  FMain.IListConnect.GetBitmap(1,FMain.Image1.Picture.Bitmap);
  //FMain.Timer_Maintient.Enabled:=True;
  State:=Etat_Connecte;
  FMain.Menu_Connect.Caption:=MenuProg[CMP_Deconnecter];
  FMain.Panel_Parametre.Visible:=False;
  FMain.Panel_grille.Visible:=False;
  FMain.Panel_Status.Visible:=True;
  FMain.Panel_Liste_Para.Visible:=False;
  F_Config.GroupBox2.Enabled:=False;
  // Виконати всі команди пошуку  відразу після з'єднання
 { try
    liste_Connection:=TStringlist.Create;
    ListCommand:=TStringList.Create;
    FMain.Search_Item('CONNECTION',FMain.Vehicule,Liste_Connection);
    //Liste_Connection.SaveToFile('c:\Connection.txt');
    i:=0;
    FMain.TypeAff:=-1;
    while i<Liste_Connection.Count do
      begin
        Application.ProcessMessages;
        if MainExit then exit;
        textel:=Liste_Connection.Strings[i];
            if uppercase(copy(textel,1,9))='<COMMAND>' then
              begin
            ListCommand.Clear;
            while (textel<>'</COMMAND>') and (i<Liste_Connection.Count)do
              begin
                textel:=Liste_Connection.Strings[i];
                textel:=trim(textel);
                ListCommand.Add(textel);
                if textel<>'</COMMAND>' then i:=i+1;
              end;
            //ListCommand.SaveToFile('c:\Command'+inttostr(i)+'.txt');
            // ми отримуємо все поле COMMAND
            // Таким чином, ми виконаємо команду
            j:=0;
            while j<ListCommand.Count do
              begin
                Application.ProcessMessages;
                if MainExit then exit;
                textel:=trim(ListCommand.Strings[j]);
                if copy(textel,1,5)='COND=' then
                  begin
                    delete(textel,1,5);
                    cond:=textel;
                  end else
                if copy(textel,1,5)='<_ISO14230_>' then
                  begin
                    FMain.Actual_ISO14230.Clear;
                    cde:='';
```

```
                    while j<ListCommand.Count do
                      begin
                        textel:=trim(ListCommand.Strings[j]);
                        if copy(textel,1,2)<>'</' then
FMain.Actual_ISO14230.Add(textel);
                        if copy(textel,1,4)='CDE=' then cde:=textel;
                        if copy(textel,1,2)='</' then j:=ListCommand.Count;
                        j:=j+1;
                      end;
                  end;
                j:=j+1;
              end; // end of while j<ListCommand.Count do
          end;  // end of if uppercase(copy(textel,1,9))='<COMMAND>' then
        if length(cde)>0 then
          begin
            delete(cde,1,4);
            FMain.ISO14230.Sid:=hextodec(copy(cde,1,2));
            delete(cde,1,3);
            j:=0;
            while length(cde)>0 do
              begin
                Application.ProcessMessages;
                if MainExit then Exit;
                FMain.ISO14230.FData[j]:=hextodec(copy(cde,1,2));
                delete(cde,1,3);
                j:=j+1;
              end;
            FMain.ISO14230.SendLength:=j;
            if length(cond)>0 then
              begin
                outdoor:=false;
                // Розрахунок фільтра на дані
                j:=pos('/',cond);
                    ByteRule:=copy(cond,1,j-1);
                    delete(cond,1,j);
                    // видаляє невідфільтровані дані
                    j:=pos('[',ByteRule);
                    if j>0 then delete(ByteRule,1,j);
                    j:=pos(']',ByteRule);
                    if j>0 then delete(ByteRule,j,1);
                    j:=pos('|',ByteRule);
                    // Отримує дані з бітів
                    if j>0 then
                     begin
                      f_Byte:=copy(ByteRule,1,j-1);
                      delete(ByteRule,1,j);
                      f_bit:=ByteRule;
                     end else
                     begin
                       f_Byte:=ByteRule;
                       f_bit:='';
                     end;
                    j:=pos('-',f_Byte);
                    if j>0 then
                      begin
                        textel1:=copy(f_Byte,1,j-1);
                        try
                          DebutByte:=Strtoint(textel1);
                        except
                          DebutByte:=0;
                        end;
                        delete(f_byte,1,j);
                        try
                          FinByte:=Strtoint(f_Byte);
                        except
                          FinByte:=0;
                        end;
                      end else
                      begin
```

```
      try
        DebutByte:=Strtoint(f_Byte);
      except
        DebutByte:=0;
      end;
      FinByte:=DebutByte;
    end;
  // отримує площину біт
  if length(f_bit)>0 then  // відфільтровує біти
   begin
     j:=pos('-',f_Bit);
     if j>0 then  // є площина
       begin
         textel1:=copy(f_Bit,1,j-1);
         try
           DebutBit:=Strtoint(textel1);
         except
           DebutBit:=0;
         end;
         delete(f_bit,1,j);
         try
           FinBit:=Strtoint(f_Bit);
         except
           FinBit:=0;
         end;
       end else
       begin   //Немає площини
         try
           DebutBit:=Strtoint(f_Bit);
         except
           DebutBit:=0;
         end;
         FinBit:=DebutBit;
       end;
     // відновлення створення маски
     mask:=0;
     for j:=DebutBit to FinBit do
       mask:=mask or (1 shl j);
   end else mask:=$FF;
  textel:=cond;
  delete(textel,1,1);
  try
    val_cond:=strtoint(textel);
  except
    outdoor:=true;
  end;
  // повторення виконання
  sleep(500);
  beginExit:=false;
  While (outdoor=False) and (BeginExit=False) do
begin
  FMain.M_Debug.Lines.Add('_ISO14230_Connect');
  Application.ProcessMessages;
  if MainExit then Exit;
  FMain.ISO14230.SendResponse;
  FMain.Attente_Start;
  Execution:=False;
  While (Execution=False)
    and (MainExit=False) do Application.ProcessMessages;
  if MainExit then Exit;
  // визначення фільтра для байта
  valeur:=0;
  for j:=DebutByte to FinByte do
    begin
      valeur:=valeur*256+(FMain.ISO14230.TrameRx[j] and mask);
    end;
  if copy(cond,1,1)='=' then
    begin
      if valeur=val_cond then outdoor:=true;
```

```
                    end else
                  if copy(cond,1,1)='>' then
                    begin
                      if valeur>val_cond then outdoor:=True;
                    end else
                  if copy(cond,1,1)='<' then
                    begin
                      if valeur<val_cond then outdoor:=True;
                    end else outdoor:=True; // є помилка

                end;
              end else
              begin
                sleep(500);
                FMain.ISO14230.SendResponse;
                FMain.Attente_Start;
                Execution:=False;
                While (Execution=False)
                  and (MainExit=False)do Application.ProcessMessages;
                if MainExit then Exit;
              end;
          end;
        i:=i+1;
      end; // end of  while i<ListConnect.Count do
finally
  liste_Connection.Free;
  ListCommand.Free;
end;

// Виведення меню
i:=0;
while i<Vehicule.Count do
  begin
    Application.ProcessMessages;
    if MainExit then Exit;
    textel:=Vehicule.Strings[i];
    trim(textel);
    if uppercase(copy(textel,1,4))='ICO=' then
      begin
        //noeud:=FMain.TreeView1.Items.GetFirstNode;
        if noeud<>nil then
          begin
            delete(textel,1,4);
            j:=pos(';',textel);
            try
              j:=strtoint(copy(textel,1,j-1));
            except
              j:=-1;
            end;
            noeud.ImageIndex:=j;
          end;
      end else
    if uppercase(copy(textel,1,8))='<EXMENU=' then
      begin
        DebutByte:=i;
        FMain.ExtendedMenu.Clear;
        delete(textel,2,2);
        FMain.ExtendedMenu.Add(textel);
        // відновлення розширеного меню, операція повинна бути повторена
        outdoor:=False;
        i:=i+1;
        while outdoor=False do
          begin
            Application.ProcessMessages;
            if MainExit then Exit;
            if i<Vehicule.Count then textel:=Vehicule.Strings[i];
            trim(textel);
            if (i>=Vehicule.Count)
              or (uppercase(copy(textel,1,8))='<EXMENU=')
```

```
                or (uppercase(copy(textel,1,8))='</EXMENU')
                or (uppercase(copy(textel,1,6))='<MENU=') then outdoor:=True;
              if not outdoor then
                begin
                  if (length(textel)>0)
                    and (copy(textel,1,2)<>'//') then
                    begin
                      k:=pos(';',textel);
                      if k=0 then FMain.ExtendedMenu.Add(textel) else
FMain.ExtendedMenu.Add(copy(textel,1,k));
                    end;
                  i:=i+1;
                end else FinByte:=i;
            end;
          //FMain.ExtendedMenu.SaveToFile('c:\ExtendedMenu.txt');
          // Виводячи розширене меню, видаляємо файл даних параметрів автомобіля
          for j:=DebutByte to FinByte do
            begin
              FMain.Vehicule.Delete(DebutByte);
            end;
          // ми повинні відновити умови повторення val_cond
          j:=0;
          while j<FMain.ExtendedMenu.Count do
            begin
              Application.ProcessMessages;
              if MainExit then Exit;
              textel:=FMain.ExtendedMenu.Strings[j];
              if copy(textel,1,9)='EXREPEAT=' then
                begin
                  delete(textel,1,9);
                  k:=pos(';',textel);
                  if k>0 then textel:=copy(textel,1,k-1);
                  textel:=trim(textel);
                  try
                    val_cond:=strtoint(textel);   // повторення змінної
                  except
                    val_cond:=-1;
                  end;
                  FMain.ExtendedMenu.Delete(j); // Стираємо це попередження
                  j:=FMain.ExtendedMenu.Count;
                end;
              j:=j+1;
            end;
          if copy(FMain.ExtendedMenu.Strings[FMain.ExtendedMenu.Count-
1],1,7)<>'</MENU>'
              then FMain.ExtendedMenu.Add('</MENU>');
          // Ми починаємо меню інтеграції  файлу  автомобіля до пам'яті
          if (val_cond>-1) and (FMain.Variable[Val_cond]>0) then
            begin
              l:=0;
              n_ligne:=0;
              max_val:=0;
              min_val:=10000;
              m:=-1;
              for j:=1 to FMain.Variable[Val_cond] do
                begin
                  for k:=0 to FMain.ExtendedMenu.Count-1 do
                    begin
                      Application.ProcessMessages;
                      if MainExit then Exit;
                      textel:=FMain.ExtendedMenu.Strings[k];
                      if copy(textel,1,4)='<MEN' then
                        begin
                          debutBit:=pos('>',textel);
                          if debutBit>0 then textel:=copy(textel,1,debutbit-1);
                          textel:=textel+' '+inttostr(j)+'>';
                        end else
                      if copy(textel,1,4)='CLK=' then
                        begin
```

```
                            m:=pos('EXV',textel);
                            textel1:=dectohex(j);
                            while m>0 do
                              begin
                                textel:=copy(textel,1,m-
1)+textel1+copy(textel,m+3,length(textel)-3);
                                m:=pos('EXV',textel);
                              end;
                          end else
                        if copy(textel,1,4)='CDE=' then
                          begin
                            debutBit:=pos(';',textel);
                            if debutBit>0 then textel:=copy(textel,1,debutbit-1);
                            textel:=textel+' '+dectohex(j)+';';
                          end else
                        if copy(textel,1,3)='LG=' then
                          begin
                            debutbit:=pos('TAB[',textel);
                            if debutbit>0 then
                              begin
                                cond:=copy(textel,1,debutbit+3);
                                delete(textel,1,debutbit+3);
                                debutbit:=pos(',',textel);
                                cond:=cond+copy(textel,1,debutbit);
                                delete(textel,1,debutbit);
                                debutbit:=pos(']',textel);
                                valeur:=strtoint(copy(textel,1,debutbit-1));
                                delete(textel,1,debutbit-1);
                                if max_val<valeur then max_val:=valeur;
                                if min_val>valeur then min_val:=valeur;
                                valeur:=valeur+n_ligne;
                                textel:=cond+inttostr(valeur)+textel;
                              end;
                          end else
                        if copy(textel,1,3)='TR=' then
                          begin
                            debutbit:=pos('TAB[',textel);
                            if debutbit>0 then
                              begin
                                cond:=copy(textel,1,debutbit+3);
                                delete(textel,1,debutbit+3);
                                debutbit:=pos(',',textel);
                                cond:=cond+copy(textel,1,debutbit);
                                delete(textel,1,debutbit);
                                debutbit:=pos(']',textel);
                                valeur:=strtoint(copy(textel,1,debutbit-1));
                                delete(textel,1,debutbit-1);
                                if max_val<valeur then max_val:=valeur;
                                if min_val>valeur then min_val:=valeur;
                                valeur:=valeur+n_ligne;
                                textel:=cond+inttostr(valeur)+textel;
                              end;
                          end;
                        FMain.Vehicule.Insert(DebutByte+k+((j-
1)*FMain.ExtendedMenu.Count),textel);
                      end;
                    // розраховуємо l
                    // n_ligne => текст у таблицю
                    // valeur => читати останнє значення
                    n_ligne:=n_ligne+(max_val-min_val)+1;
                  end;
              end;
            // Ми закінчили підключенняї, це переводить курсор на початок цієї
області
            i:=DebutByte-1;
        end else
        if uppercase(copy(textel,1,6))='<MENU=' then
          begin
          delete(textel,1,6);
```

```
        j:=pos('>',textel);
        if j>0 then
          begin
            MenuList.Add(copy(textel,1,j-1)+'/'+inttostr(i)+'/');
            noeud:=FMain.TreeView1.Items.Add(nil,copy(textel,1,j-1));
          end;
       end else
       if uppercase(copy(textel,1,6))='</MENU' then
         begin
         delete(textel,1,6);
         if MenuList.Count>0 then MenuList.Strings[MenuList.Count-
1]:=MenuList.Strings[MenuList.Count-1]+inttostr(i)+'/';
       end ;
     i:=i+1;
  end;
  //FMain.TreeView1.Enabled:=True;
  FMain.Vehicule.SaveToFile('C:\vehicule.txt');
  }
  FMain.T_Command.Enabled:=True;
end;

//*****************************************************************************
// Роз'єднання
//*****************************************************************************
procedure TFMain.ISO14230_Disconnect(Sender: TObject);
begin
  //FMain.Timer_Maintient.Enabled:=False;
  F_Config.GroupBox2.Enabled:=True;
  FMain.TreeView1.Enabled:=False;
  FMain.TreeView1.Items.Clear;
  FMain.Menu_Connect.Caption:=MenuProg[CMP_Connecter];
  FMain.Image1.Picture.Bitmap:=nil;
  FMain.IListConnect.GetBitmap(0,FMain.Image1.Picture.Bitmap);
  F_Config.GroupBox2.Enabled:=True;
  State:=0;
  FMain.ComPort1.Close;
  FMain.Attente_Stop;
end;

//*****************************************************************************
//Помилки
//*****************************************************************************
procedure TFMain.ISO14230_Error(Sender: TObject; Erreur: integer;
  MsgErreur: string);
var textel : string;
begin
  DateTimeToString(textel,'h:nn.ss.zzz',Now);
  execution:=True;
  FMain.M_Status.Lines.Add(textel+' - '+MsgErreur);
  FMain.M_Debug.Lines.Add(textel+' - MainError: '+MsgErreur);
  FMain.Attente_Stop;
  if erreur=Err_5Baud  then
     begin
       {FMain.Attente_Stop;

       FMain.cpt_erreur:=FMain.cpt_erreur+1;
       if FMain.cpt_erreur<=3
          then  FMain.ISO14230.StartDiagnosis
          else FMain.GlobalReset;
          }
     end;
  if Erreur=Err_TO then
     begin
       if FMain.State=Etat_Connecte then
          begin
            //FMain.cpt_erreur:=FMain.cpt_erreur+1;
            //if FMain.cpt_erreur<=3
            //   then  FMain.ISO14230.SendResponse
            //   else FMain.Menu_ConnectClick(nil);
```

```
          end;
        F_Config.GroupBox2.Enabled:=True;
        FMain.GlobalReset;
      end;

  //FMain.diag_actif:=False;
  //FMain.Attente_Start;
  //FMain.Timer_Maintient.Enabled:=True;
  //FMain.Execution:=True;
end;


//*****************************************************************************
procedure TFMain.ISO14230_ReceiveTrame(Sender: TObject);



var i,j,k,l,m,n,k1 : integer;
    textel,datetxt : string;
    octet,octet1 : byte;
    noeud : TTreeNode;
    ByteRule : string;
    RepeatRule : string;
    f_Byte,f_bit : string;
    DebutByte,FinByte : integer;
    DebutBit,FinBit : integer;
    Mask : Byte;
    textel1,textel2,textel3 : string;
    value : real;
    Val_Int : integer;
    nb_byteValue : array[0..100] of integer;
    nb_GroupValue : integer;
    ByteValue : array[0..100,0..255] of byte;
    ax,bx : real; // Це значення розраховується  Value=ax*X+bx
    couleur : string;
    IRow,AffRow : integer;
    sens_byte : Boolean;
    etat_timerIO : boolean;

begin
  FMain.cpt_erreur:=0;
  IRow:=-1;
  //FMain.Timer_Maintient.Enabled:=True;
  FMain.Attente_Stop;
  textel:='';
  For i:=0 to FMain.ISO14230.ReceiveLength-1 do
     textel:=textel+' '+dectohex(FMain.ISO14230.TrameRx[i]);
  DateTimeToString(datetxt,'h:nn.ss.zzz',Now);
  FMain.M_Debug.Lines.Add(Datetxt+' -Fonction Receive- '+textel);
  FMain.M_Debug.Lines.Add('');
  if (FMain.ClickTree) or (FMain.ISO14230.SidReceive=$7E) then
    begin
      Execution:=TRue;
      exit;
    end;
  if FMain.NoReceive then
    begin
      if FOnOff.Actif=True then FOnOff.Enable_Screen;
      if FBiPWM.Actif then FBiPwm.Enable_Screen;
      if FPwm.Actif then FPWM.Enable_Screen;
      if FBridge.Actif then FBridge.Enable_Screen;
      if FValue.Actif then FValue.Enable_Screen;
      if FWiper.Actif then FWiper.Enable_Screen;
      FMain.NoReceive:=False;
      Execution:=TRue;
      Exit;
    end;
  if State>0 then
     begin
```

```
        //if (FMain.ISO14230.Sid=$12) and (FMain.ISO14230.TrameRx[0]=$01) then
        //  begin
        //    Couleur:='zz';
        //  end;
        octet:=FMain.ISO14230.Sid+$40;
        if octet<>FMain.ISO14230.SidReceive then
           begin
             FMain.M_Debug.Lines.Add(Datetxt+' - octet='+dectohex(octet)+'-
Sid='+dectohex(FMain.ISO14230.Sid));
             //FMain.Timer_Maintient.Enabled:=False;
             FMain.M_Debug.Lines.Add(Datetxt+MenuProg[CMP_SidError]);
             FMain.M_Status.Lines.Add(Datetxt+MenuProg[CMP_SidError]);
             //FMain.ISO14230.SendResponse;
             //FMain.Attente_Start;
             Execution:=True;
             if FMain.TypeAff=CT_AFFONEVALUE then
               begin
                 textel:='';
                 for k:=0 to FMain.ISO14230.ReceiveLength-1 do
                   begin
                     textel:=textel+dectohex(FMain.ISO14230.TrameRx[k])+' ';
                   end;
                 textel:=dectohex(FMain.ISO14230.SidReceive)+' '+textel+' /
'+MenuProg[CMP_SidError];
                 FMain.M_Valeur.Lines.Add(Datetxt+' - '+textel);
                 FMain.E_Valeur.Text:='';
               end else
             if FMain.TypeAff=CT_AFFONEPARAMETER then
               begin
                 textel:='';
                 for k:=0 to FMain.ISO14230.ReceiveLength-1 do
                   begin
                     textel:=textel+dectohex(FMain.ISO14230.TrameRx[k])+' ';
                   end;
                 textel:=dectohex(FMain.ISO14230.SidReceive)+' '+textel+' /
'+MenuProg[CMP_SidError];
                 FMain.M_Parametre.Lines.Add(Datetxt+' - '+textel);
                 FMain.E_Para.Text:='';
               end;
             exit;
           end;
        //FMain.Timer_Maintient.Enabled:=True;
        etat_TimerIO:=FMain.T_StatusIO.Enabled;
        FMain.T_StatusIO.Enabled:=False;
        Couleur:='';
        // Ми беремо дані з байта для перегляду
        for i:=0 to Actual_ISO14230.Count-1 do
          begin
            textel:=Actual_ISO14230.Strings[i];
            if uppercase(copy(textel,1,3))='TR=' then
              begin
                sens_byte:=CT_SensHL;
                delete(textel,1,3);
                // ми форматуємо вид значення даних на екрані
                j:=pos('/',textel);
                ByteRule:=copy(textel,1,j-1);
                delete(textel,1,j);
                //delete(ByteRule,1,1);
                //ByteRule:=copy(ByteRule,1,length(ByteRule)-1);
                nb_GroupValue:=1;
                if length(ByteRule)>0 then
                  begin
                    // створюємо правило для пошуку даних
                    if uppercase(ByteRule[1])='R' then
                      begin
                        j:=pos('[',ByteRule);
                        RepeatRule:=copy(ByteRule,2,j-2); // je rйcurйre le debut
de la loi de rйpйtition
                        delete(ByteRule,1,j-1);
```

```
                    end else
                    begin
                      // ми не повторюємо правило
                      RepeatRule:='';
                    end;
                  if uppercase(copy(ByteRule,length(ByteRule),1))='L' then
                    begin
                      delete(byteRule,length(byteRule),1);
                      sens_Byte:=CT_SensLH;
                    end;
                  // видаляє невідфільтровані дані
                  j:=pos('[',ByteRule);
                  if j>0 then delete(byteRule,1,j);
                  j:=pos(']',ByteRule);
                  if j>0 then ByteRule:=copy(ByteRule,1,j-1);
                  j:=pos('|',ByteRule);
                  // Отримує дані з бітів
                  if j>0 then
                    begin
                      f_Byte:=copy(ByteRule,1,j-1);
                      delete(ByteRule,1,j);
                      f_bit:=ByteRule;
                    end else
                    begin
                      f_Byte:=ByteRule;
                      f_bit:='';
                    end;
                  j:=pos('-',f_Byte);
                  if j>0 then
                    begin
                      textel1:=copy(f_Byte,1,j-1);
                      try
                        DebutByte:=Strtoint(textel1);
                      except
                        DebutByte:=0;
                      end;
                      delete(f_byte,1,j);
                      try
                        FinByte:=Strtoint(f_Byte);
                      except
                        FinByte:=0;
                      end;
                    end else
                    begin
                      try
                        DebutByte:=Strtoint(f_Byte);
                      except
                        DebutByte:=0;
                      end;
                      FinByte:=DebutByte;
                    end;
                  if DebutByte>FinByte then
                    begin
                      j:=DebutByte;
                      DebutByte:=FinByte;
                      FinByte:=j;
                    end;
                  if FinByte>FMain.ISO14230.ReceiveLength then
FinByte:=FMain.ISO14230.ReceiveLength-1;
                  if DebutByte>FMain.ISO14230.ReceiveLength then
DebutByte:=FMain.ISO14230.ReceiveLength;
                  // отримує площину біт
                  if length(f_bit)>0 then  // відфільтровує біти
                    begin
                      j:=pos('-',f_Bit);
                      if j>0 then  // є площина
                        begin
                          textel1:=copy(f_Bit,1,j-1);
                          try
```

```
                    DebutBit:=Strtoint(textel1);
                  except
                    DebutBit:=0;
                  end;
                  delete(f_bit,1,j);
                  try
                    FinBit:=Strtoint(f_Bit);
                  except
                    FinBit:=0;
                  end;
                end else
                begin   //Немає площини
                  try
                    DebutBit:=Strtoint(f_Bit);
                  except
                    DebutBit:=0;
                  end;
                  FinBit:=DebutBit;
                end;
              // відновлення створення маски
              mask:=0;
              for j:=DebutBit to FinBit do
                mask:=mask or (1 shl j);
            end else
            begin
              mask:=$FF;
              DebutBit:=0;
              FinBit:=0;
            end;
          if DebutBit>FinBit then
            begin
              j:=DebutBit;
              DebutBit:=FinBit;
              FinBit:=j;
            end;
          // значення заповнюються в буфер значення
          if length(RepeatRule)>0 then
            begin
              try
                j:=strtoint(RepeatRule);
              except
                j:=0;
              end;
              nb_GroupValue:=0;
              repeat
                if (j+DebutByte<FMain.ISO14230.ReceiveLength)then
                  begin
                    l:=0;
                    for k:=j+DebutByte to j+FinByte do
                      begin
                        if Sens_Byte=CT_SensHL
                          then
ByteValue[nb_GroupValue,l]:=(FMain.ISO14230.TrameRx[k] and mask) shr DebutBit
                          else
ByteValue[nb_GroupValue,l]:=(FMain.ISO14230.TrameRx[k] and mask) shr debutBit;
                        l:=l+1;
                      end;
                    nb_ByteValue[nb_GroupValue]:=FinByte-DebutByte+1;
                    nb_GroupValue:=nb_GroupValue+1;
                  end;
                j:=j+FinByte+1;
              until j>FMain.ISO14230.ReceiveLength-1;
            end else  // end of if length(RepeatRule)>0 then
            begin
              if Sens_Byte=CT_SensHL then l:=0 else l:=FinByte-
DebutByte;
              for j:=DebutByte to FinByte do
                begin
                  if Sens_Byte=CT_SensHL then
```

```
                              begin
                                ByteValue[0,l]:=(FMain.ISO14230.TrameRx[j] and
mask) shr DebutBit;
                                l:=l+1;
                              end else
                              begin
                                ByteValue[0,l]:=(FMain.ISO14230.TrameRx[j] and
mask) shr DebutBit;
                                l:=l-1;
                              end;
                            end;
                          nb_ByteValue[0]:=FinByte-DebutByte+1;
                          nb_GroupValue:=1;
                        end;     // end of ELSE of if length(RepeatRule)>0 then
                    end else  // end of if length(ByteRule)>0 then
                    begin
                      nb_ByteValue[0]:=FMain.ISO14230.ReceiveLength;
                      nb_GroupValue:=1;
                      if Sens_Byte=CT_SensHL then l:=0 else l:=FinByte-DebutByte;
                      for j:=0 to nb_byteValue[0]-1 do
                        begin
                          if Sens_Byte=CT_SensHL then
                            begin
                              ByteValue[0,l]:=(FMain.ISO14230.TrameRx[j]) shr
DebutBit;
                              l:=l+1;
                            end else
                            begin
                              ByteValue[0,l]:=(FMain.ISO14230.TrameRx[j]) shr
DebutBit;
                              l:=l-1;
                            end;
                        end;
                    end;     // end of ELSE  if length(ByteRule)>0 then
                  // ми будемо шукати, якщо є набір значень
                  ax:=1;
                  bx:=0;
                  if length(textel)>0 then
                    if uppercase(copy(textel,1,2))='V=' then
                      begin
                        j:=pos('/',textel);
                        textel1:=copy(textel,1,j-1);
                        delete(textel,1,j);
                        delete(textel1,1,2);
                        j:=pos('|',textel1);
                        if j>0 then
                          begin
                            ax:=strtoReel(copy(textel1,1,j-1));
                            delete(textel1,1,j);
                            bx:=strtoReel(textel1);
                          end else
                          begin
                            ax:=strtoReel(textel1);
                            bx:=0;
                          end;
                      end;
                    if uppercase(copy(textel,1,4))='Значення=' then
                      begin
                        delete(textel,1,4);
                        j:=pos('/',textel);
                        if j=0 then j:=pos(';',textel);
                        if j>0 then textel2:=copy(textel,1,j-1) else
textel2:=textel;
                        delete(textel,1,length(textel2)+1);
                        try
                          j:=strtoint(textel2);
                        except
                          j:=0;
                        end;
```

```
                    FMain.Variable[j]:=0;
                    for k:=0 to nb_ByteValue[0]-1 do
                      begin

FMain.Variable[j]:=FMain.Variable[j]*256+ByteValue[0,k];
                      end;
                  end;
                if uppercase(copy(textel,1,6))='Таблиця=' then
                  begin
                    delete(textel,1,6);
                    j:=pos('/',textel);
                    if j=0 then j:=pos(';',textel);
                    if j>0 then textel2:=copy(textel,1,j-1) else
textel2:=textel;
                    delete(textel,1,length(textel2)+1);
                    try
                      j:=strtoint(textel2);
                    except
                      j:=0;
                    end;
                    textel2:='';
                    for k:=0 to nb_ByteValue[0]-1 do
                      begin
                        //textel2:=textel2+dectohex(ByteValue[0,k])+' ';
                        if ByteValue[0,k]>=32
                          then textel2:=textel2+chr(ByteValue[0,k]);
                      end;
                    if FMain.Table[j]= nil then
FMain.Table[j]:=TStringList.create;
                    FMain.Table[j].Add(textel2);

//FMain.Table[j].SaveToFile('c:\table_'+inttostr(j)+'.txt');
                  end;
              if uppercase(copy(textel,1,4))='Колонка=' then
                begin
                  delete(textel,1,4);
                  j:=pos('/',textel);
                  textel2:=copy(textel,1,j-1);
                  delete(textel,1,j);
                  IRow:=strtoint(textel2);
                end;
              // ми записуємо значення на екран

              for j:=0 to nb_GroupValue-1 do
                begin
                  textel1:=textel;
                  if FMain.TypeAff=CT_AFFMULTIVALUE then
                    begin
                      textel2:='';
                      val_int:=0;
                      for k:=0 to nb_ByteValue[j]-1 do
                        begin
                          val_int:=val_int*256+ByteValue[j,k];
                        end;
                      if FMain.TypeCdeActif=1 then
FMain.StringGrid2.Cells[FMain.StringGrid2.ColCount-
1,FMain.ligneGrid2]:=inttostr(val_int);
                      if FMain.ligneGrid2<FMain.StringGrid2.RowCount-1 then
                        begin
                          FMain.ligneGrid2:=FMain.ligneGrid2+1;
                          try

k:=hextodec(uppercase(FMain.StringGrid2.Cells[0,FMain.ligneGrid2]));
                          except

Application.MessageBox(PCHAR(MenuProg[CMP_NumberNotHex]),PCHAR(MenuProg[CMP_Erro
r]),0);
                            exit;
                          end;
```

```
                            if k<=255 then
                              begin
                                if FMain.TypeCdeActif=1
                                  then textel:=FMain.CdeReadParam
                                  else textel:=FMain.CdeWriteParam;
                                if textel[1]='L' then l:=0 else l:=1;
                                delete(textel,1,1);
                                FMain.ISO14230.Sid:=hextodec(copy(textel,1,2));
                                delete(textel,1,3);
                                FMain.ISO14230.SendLength:=0;
                                while length(textel)>0 do
                                  begin

FMain.ISO14230.FData[FMain.ISO14230.SendLength]:=hextodec(copy(textel,1,2));

FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+1;
                                    delete(textel,1,3);
                                  end;

FMain.ISO14230.FData[FMain.ISO14230.SendLength]:=k;

FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+1;
                                if FMain.TypeCdeActif=2 then
                                  begin

textel:=FMain.StringGrid2.Cells[FMain.StringGrid2.ColCount-1,FMain.ligneGrid2];
                                    try
                                      textel:=dectohex(strtoint(textel));
                                    except
                                      textel:='00';
                                    end;
                                    if
(FMain.StringGrid2.Cells[3,FMain.ligneGrid2]='16')
                                      and (length(textel)=2)
                                        then textel:='00'+textel;
                                    if (l=0) and (length(textel)>2) then
                                      begin

textel:=copy(textel,3,2)+copy(textel,1,2);
                                      end;
                                    l:=(length(textel) div 2);
                                    for k:=0 to l-1 do
                                      begin

FMain.ISO14230.FData[k+FMain.ISO14230.SendLength]:=hextodec(copy(textel,1,2));
                                        delete(textel,1,2);
                                      end;

FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+l;
                                  end; // end of if FMain.TypeCdeActif=2 then
                                // Конфігурація системи
                                FMain.ISO14230.SendResponse;
                                FMain.Attente_Start;
                                exit;
                              end;
                            end;   // end of   if
FMain.ligneMenu<FMain.StringGrid2.RowCount then
                          exit;
                        end else
                      if FMain.TypeAff=CT_AFFONEVALUE then
                        begin
                          textel2:='';
                          val_int:=0;
                          for k:=0 to nb_ByteValue[j]-1 do
                            begin
                              textel2:=textel2+dectohex(ByteValue[j,k])+' ';
                              val_int:=val_int*256+ByteValue[j,k];
                            end;
                          textel2:=dectohex(FMain.ISO14230.SidReceive)+' '+textel2;
```

```
                        FMain.M_Valeur.Lines.Add(Datetxt+' - '+textel2);
                        if FMain.CdeValueAction=1 then
                          begin
                            if FMain.RadioAffVal1.ItemIndex=0
                              then FMain.E_VAleur.Text:=inttostr(val_int)
                              else FMain.E_Valeur.Text:=dectohex(val_int);
                          end;
                      end else
                    if FMain.TypeAff=CT_AFFONEPARAMETER then
                      begin
                        textel2:='';
                        val_int:=0;
                        for k:=0 to nb_ByteValue[j]-1 do
                          begin
                            textel2:=textel2+dectohex(ByteValue[j,k])+' ';
                            val_int:=val_int*256+ByteValue[j,k];
                          end;
                        textel2:=dectohex(FMain.ISO14230.SidReceive)+' '+textel2;
                        FMain.M_Parametre.Lines.Add(Datetxt+' - '+textel2);
                        if FMain.CdeValueAction=1 then
                          begin
                            if FMain.RadioAffVal.ItemIndex=0
                              then FMain.E_Para.Text:=inttostr(val_int)
                              else FMain.E_Para.Text:=dectohex(val_int);
                          end;
                      end else  //end of if Fmain.TypeAff=CT_AFFONEPARAMETER then
                    if Fmain.TypeAff=CT_AFFCONFIRM then
                      begin

Application.MessageBox(PCHAR(MenuProg[CMP_MsgConfirm]),PCHAR(MenuProg[CMP_MsgOK]
),0);
                        exit;
                      end else // end of if Fmain.TypeAff=CT_AFFCONFIRM then
                    if FMain.TypeAff=CT_AFFGRID then
                      begin
                        if IRow=-1 then
                          begin

//FMain.StringGrid3.RowCount:=FMain.StringGrid3.RowCount+1;
                            //for k:=0 to FMain.StringGrid3.ColCount-1 do
                            //
FMain.StringGrid3.Cells[k,FMain.StringGrid3.RowCount-1];
                            //AffRow:=FMain.StringGrid3.RowCount-1;

FMain.StringGrid1.RowCount:=FMain.StringGrid1.RowCount+1;

FMain.Ajust_StringList(FMain.ColorList.Count+1,FMain.ColorList);
                            for k:=0 to FMain.StringGrid1.ColCount-1 do

FMain.StringGrid1.Cells[k,FMain.StringGrid1.RowCount-1]:='';
                            AffRow:=FMain.StringGrid1.RowCount-1;
                          end else
                          begin
                            //if IRow>=FMain.StringGrid3.RowCount then
                            //  begin
                            //    FMain.StringGrid3.RowCount:=IRow+1;
                            //    for k:=0 to FMain.StringGrid3.ColCount-1 do
                            //      FMain.StringGrid3.Cells[k,IRow]:='';
                            //  end;
                            if IRow>=FMain.StringGrid1.RowCount then
                              begin
                                FMain.StringGrid1.RowCount:=IRow+1;
                                FMain.Ajust_StringList(IRow+1,FMain.ColorList);
                                for k:=0 to FMain.StringGrid1.ColCount-1 do
                                  FMain.StringGrid1.Cells[k,IRow]:='';
                              end;
                            AffRow:=IRow;
                          end;
                        repeat
```

```
                          Application.ProcessMessages;
                          if MainExit then Exit;
                          k1:=pos('/',textel1);
                          if k1>0 then textel2:=copy(textel1,1,k1-1) else
textel2:=textel1;
                          if uppercase(copy(textel2,1,3))='CL=' then
                            begin
                              delete(textel2,1,3);
                              FMain.ColorList.Strings[AffRow]:=textel2;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-2,AffRow]:=textel2;

                              textel2:='';
                            end else // end of if
uppercase(copy(textel2,1,3))='CL=' then
                          if uppercase(copy(textel2,1,2))='GR' then
                            begin
                              delete(textel2,1,2);
                              k:=pos('=',textel2);
                              try
                                l:=strtoint(copy(textel2,1,k-1));
                              except
                                l:=0;
                              end;
                              delete(textel2,1,k);
                              if l<FMain.StringGrid1.ColCount then
                                begin
                                  if uppercase(copy(textel2,1,1))='[' then
                                    begin   // ми отримали список значень
                                      Val_int:=0;
                                      for k:=0 to nb_ByteValue[j]-1 do
                                        begin
                                          Val_int:=Val_int*256+ByteValue[j,k];
                                        end;
                                      // ми шукаємо мітки
                                      delete(textel2,1,1);
                                      delete(textel2,length(textel2),1);
                                      k:=0;
                                      textel3:='';
                                      while (length(textel2)>0) do
                                        begin
                                          m:=pos(',',textel2);
                                          if m>0 then
                                            begin
                                              if k=Val_int then
                                                begin
                                                  textel3:=copy(textel2,1,m-1);
                                                  textel2:='';
                                                end else k:=k+1;
                                              delete(textel2,1,m);
                                            end else
                                            begin
                                              if length(textel2)>0 then
                                                if k=Val_int then
textel3:=textel2;
                                                textel2:='';
                                            end;
                                        end;
                                      if length(textel3)=0 then
textel3:=MenuProg[CMP_Unknow];

//FMain.StringGrid3.Cells[l,AffRow]:=textel3;
                                      FMain.StringGrid1.Cells[l,AffRow]:=textel3;
                                      if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel3)+32
                                        then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel3)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
```

```
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'T'+Textel3+'|';
                                    end else
                                  if uppercase(copy(textel2,1,7))='VAL_TXT' then
                                    begin
                                      textel2:='';
                                      for k:=0 to nb_ByteValue[j]-1 do
                                        begin
                                          if ByteValue[j,k]>31
                                            then
textel2:=textel2+chr(ByteValue[j,k]);
                                        end;

//FMain.StringGrid3.Cells[l,AffRow]:=textel2;
                                      FMain.StringGrid1.Cells[l,AffRow]:=textel2;
                                      if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel2)+32
                                        then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel2)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'T'+Textel2+'|';
                                    end else
                                  if uppercase(copy(textel2,1,5))='VALUE' then
                                    begin
                                      Value:=0;
                                      for k:=0 to nb_ByteValue[j]-1 do
                                        begin
                                          Value:=Value*256+ByteValue[j,k];
                                        end;
                                      Value:=Value*ax+bx;
                                      textel2:=reeltostr(Value);
                                      FMain.StringGrid1.Cells[l,AffRow]:=textel2;
                                      if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel2)+32
                                        then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel2)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'V'+Textel2+'|';
                                    end else
                                  if uppercase(copy(textel2,1,7))='VAL_HEX' then
                                    begin
                                      textel2:='';
                                      for k:=0 to nb_ByteValue[j]-1 do
                                        begin

textel2:=textel2+dectohex(ByteValue[j,k])+' ';
                                        end;
                                      FMain.StringGrid1.Cells[l,AffRow]:=textel2;
                                      if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel2)+32
                                        then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel2)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'H'+Textel2+'|';
                                    end else
                                  if uppercase(copy(textel2,1,4))='IMG_' then
                                    begin

//FMain.StringGrid3.Cells[l,AffRow]:=textel2;
                                      Val_Int:=0;
                                      for k:=0 to nb_ByteValue[j]-1 do
                                        begin
                                          Val_Int:=Val_Int*256+ByteValue[j,k];
```

```
                                        end;
                                      if Val_int=0 then textel2:=textel2+'0' else
textel2:=textel2+'1';


//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=inttostr(Val_Int);
                                      FMain.StringGrid1.Cells[l,AffRow]:=textel2;
                                      if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel2)+32
                                        then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel2)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'V'+textel2[length(textel2)]+'|';
                                    end else
                                  if uppercase(copy(textel2,1,5))='Файл(' then
                                    begin
                                      delete(textel2,1,5);
                                      k:=pos(')',textel2);
                                      if k>0 then
                                        begin
                                          textel2:=copy(textel2,1,k-1);
                                          textel2:=uppercase(trim(textel2));
                                          if
uppercase(FMain.nom_fichier)<>textel2 then
                                            if
FileExists(Repertoire_courant+'Vehicule\'+textel2) then
                                              begin
                                                FMain.nom_fichier:=textel2;

FMain.Fichier.LoadFromFile(Repertoire_courant+'Vehicule\'+textel2);
                                              end;
                                          Val_Int:=0;
                                          for k:=0 to nb_ByteValue[j]-1 do
                                            begin

Val_Int:=Val_Int*256+ByteValue[j,k];
                                            end;
                                          textel2:=dectohex(Val_Int);
                                          k:=0;
                                          while k<FMain.Fichier.Count do
                                            begin
                                              m:=pos(';',Fichier.Strings[k]);
                                              if copy(Fichier.Strings[k],1,m-
1)=textel2 then

                                                begin
                                                  textel2:=Fichier.Strings[k];
                                                  delete(textel2,1,m);
                                                  k:=FMain.Fichier.Count;
                                                end;
                                              k:=k+1;
                                            end;

FMain.StringGrid1.Cells[l,AffRow]:=textel2;
                                          if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel2)+32
                                            then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel2)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'T'+textel2+'|';
                                        end;
                                    end else
                                  if uppercase(copy(textel2,1,4))='TAB[' then
                                    begin
                                      delete(textel2,1,4);
```

```
                                        k:=pos(',',textel2);
                                        if k>0 then
                                          begin
                                            try
                                              m:=strtoint(copy(textel2,1,k-1));
                                            except
                                              m:=-1;
                                            end;
                                            delete(textel2,1,k);
                                            k:=pos(']',textel2);
                                            try
                                              n:=strtoint(copy(textel2,1,k-1));
                                            except
                                              n:=-1;
                                            end;
                                            if (m>-1) and (m<10) then
                                              begin
                                                if FMain.Table[m]<>nil then
                                                  begin
                                                    if (n>-1) and
(n<FMain.Table[m].Count)then
                                                      begin

textel2:=FMain.Table[m].Strings[n];
                                                      end else
textel2:=MenuProg[CMP_ERRORLIGNETABLE];
                                                  end else
textel2:=MenuProg[CMP_TableUnknow];
                                              end else
textel2:=MenuProg[CMP_ERRORNUMBERTABLE];
                                          end;
                                        FMain.StringGrid1.Cells[l,AffRow]:=textel2;
                                        if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel2)+32
                                          then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel2)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'T'+textel2+'|';
                                      end else
                                      begin
                                        FMain.StringGrid1.Cells[l,AffRow]:=textel2;
                                        if
FMain.StringGrid1.ColWidths[l]<FMain.StringGrid1.Canvas.TextWidth(textel2)+32
                                          then
FMain.StringGrid1.ColWidths[l]:=FMain.StringGrid1.Canvas.TextWidth(textel2)+32;

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]:=FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-
1,AffRow]+'V'+textel2+'|';
                                      end;
                                  end;
                              end;
                            if k1>0 then delete(textel1,1,k1) else textel1:='';
                          until length(textel1)<1;
                          if FMain.StringGrid1.RowCount>1 then
FMain.StringGrid1.FixedRows:=1;
                        end;
                    end;  // end of for j:=0 to nb_GroupValue-1 do
                  //FMain.rafraichir_diag;
                end;
            end; // end of for i:=0 to Actual_ISO14230.Count-1 do
          FMain.Execution:=True;
          if InIOTimer then Exit;
          FMain.M_Debug.Lines.Add('Час повторення: '+inttostr(FMain.TimeRepeat)+'
ms');
          if (FMain.TypeAff<>CT_AFFCONFIRM)
             and (FMain.TypeAff<>CT_AFFONEPARAMETER)
```

```
          and (FMain.ligneMenu<FMain.ActualMenu.Count) then
          begin
            FMain.T_StatusIO.Interval:=100;
            FMain.T_StatusIO.Enabled:=True;
          end else
          begin
            If (FMain.TimeRepeat>0) and (FMain.TypeAff<>CT_AFFCONFIRM) then
              begin
                FMain.ligneMenu:=0;
                FMain.T_StatusIO.Interval:=FMain.TimeRepeat;
                FMain.T_StatusIO.Enabled:=True;
              end;
          end;
     end;
end;

//*****************************************************************************
procedure TFMain.ISO14230_Status(Sender: TObject; Messages: string);
var textel : string;
begin
  DateTimeToString(textel,'h:nn.ss.zzz',Now);
  FMain.M_Status.Lines.Add(textel+' - '+Messages);
end;

//*****************************************************************************
procedure TFMain.ISO14230_Step(Sender: TObject; Messages: string);
var textel : string;
begin
  DateTimeToString(textel,'h:nn.ss.zzz',Now);
  FMain.M_Debug.Lines.Add(textel+' - '+Messages);
  FMain.num_image:=FMain.num_image+1;
  if FMain.num_image>14 then FMain.num_image:=1;
  FMain.I_Attente.Picture.Bitmap:=nil;
  FMain.IList_Rose.GetBitmap(FMain.num_image,FMain.I_Attente.Picture.Bitmap);
end;

//*****************************************************************************
procedure TFMain.Menu_QuitterClick(Sender: TObject);
begin
  FMain.Execution:=True;
  FMain.T_StatusIO.Enabled:=False;
  FMain.Close;
end;

//*****************************************************************************
procedure TFMain.Menu_SettingClick(Sender: TObject);
begin
  F_Config.execute;
end;

//*****************************************************************************
procedure TFMain.Panel1Click(Sender: TObject);
begin
  FMain.SaveDialog1.FileName:=repertoire_courant+'Debug.txt';
  FMain.SaveDialog1.Filter:='*.txt';
  if FMain.SaveDialog1.Execute then
     begin
       FMain.M_Debug.Lines.SaveToFile(FMain.SaveDialog1.FileName);
     end;
end;

//*****************************************************************************
procedure TFMain.RadioAffVal1ChangeBounds(Sender: TObject);
var textel : string;
    value : integer;
begin
  textel:=FMain.E_Valeur.Text;
  textel:=trim(textel);
  try
```

```
    if FMain.RadioAffVal.ItemIndex=0 then
      begin
        value:=hextodec(textel);
        textel:=inttostr(value);
      end else
      begin
        value:=strtoint(textel);
        textel:=dectohex(value);
      end;
    FMain.E_Valeur.Text:=textel;
  except
  end;
end;

//*****************************************************************************
procedure TFMain.RadioAffValChangeBounds(Sender: TObject);
var textel : string;
    value : integer;
begin
  textel:=FMain.E_Para.Text;
  textel:=trim(textel);
  try
    if FMain.RadioAffVal.ItemIndex=0 then
      begin
        value:=hextodec(textel);
        textel:=inttostr(value);
      end else
      begin
        value:=strtoint(textel);
        textel:=dectohex(value);
      end;
    FMain.E_Para.Text:=textel;
  except
  end;
end;

//*****************************************************************************
procedure TFMain.StringGrid1Click(Sender: TObject);
var aRow,aCol,ligne : integer;
    textel,textel1 : string;
    i,j,k : integer;
    x,y : integer;
    tx,ty : string;
begin
  // Натиснути кнопку у головному вікні

  aRow:=FMain.StringGrid1.Row;
  aCol:=FMain.StringGrid1.Col;
  if aRow=0 then exit;
  //ligne:=aRow+FMain.Grille_Up; // на наступний рядок
  ligne:=aRow;
  if (FMain.ClickActif=False) and (length(FMain.ClickGlobal)>0) then
    begin
      While (FMain.ISO14230.EndReceive=False)
        and (MainExit=False) do Application.ProcessMessages;
      if MainExit then Exit;
      FMain.Cde2Send(FMain.ClickGlobal);
      FMain.Attente_Start;
      FMain.Execution:=False;
      FMain.ISO14230.SendResponse;
      While (not FMain.Execution)
        and (MainExit=False) do application.ProcessMessages;
      if MainExit then Exit;
      FMain.ClickActif:=True;
      FMain.B_Mode_Maitre.Visible:=True;
    end;
  i:=0;
  while i<ClickList.Count do
    begin
```

```
textel:=ClickList.Strings[i];
if uppercase(copy(textel,1,4))='CLK=' then
  begin
   delete(textel,1,4);
   j:=pos('/',textel);
   if j>0 then
     begin
       textel1:=copy(textel,1,j-1);
       delete(textel,1,j);
       j:=pos(',',textel1);
       if j>0 then
         begin
           tx:=copy(textel1,1,j-1);
           delete(textel1,1,j);
           ty:=textel1;
           try
             if length(tx)>0 then x:=strtoint(tx) else x:=-1;
           except
             x:=-1;
           end;
           try
             y:=strtoint(ty);
           except
             y:=-1;
           end;
           if ((y>=0) and (x>=0) and (x=aCol) and (y=ligne))
             or ((x<0) and (y>=0) and (y=ligne))
             or ((x>=0) and (y<0) and (x=aCol))
             or ((x<0) and (y<0)) then
             begin
               // переходимо до умов описаних низче
               while length(textel)>0 do
                 begin
                   j:=pos('/',textel);
                   if j>0 then
                     begin
                       textel1:=copy(textel,1,j-1);
                       delete(textel,1,j);
                     end else
                     begin
                       textel1:=textel;
                       textel:='';
                     end;
                   if uppercase(textel1)='ONOFF' then
                     begin
                       while length(textel)>0 do
                         begin
                           j:=pos('/',textel);
                           if j>0 then
                             begin
                               textel1:=copy(textel,1,j-1);
                               delete(textel,1,j);
                             end else
                             begin
                               textel1:=textel;
                               textel:='';
                             end;
                           if copy(textel1,1,5)='Вихід=' then
                             begin
                               delete(textel1,1,5);
                               if uppercase(textel1)='GLOBAL'
                                 then FOnOff.Cde_Exit:=''
                                 else FOnOff.Cde_Exit:=textel1;
                             end else
                           if copy(textel1,1,3)='ON=' then
                             begin
                               delete(textel1,1,3);
                               FOnOff.Cde_ON:=textel1;
                             end else
```

```
                    if copy(textel1,1,4)='OFF=' then
                      begin
                        delete(textel1,1,4);
                        FOnOff.Cde_OFF:=textel1;
                      end else
                    if copy(textel1,1,5)='I’мя=' then
                      begin
                        delete(textel1,1,5);
                        FOnOff.Caption:=textel1;
                      end else
                    if copy(textel1,1,6)='Опис=' then
                      begin
                        delete(textel1,1,6);
                        FOnOff.L_Pin.Caption:=textel1;
                      end;
                  end; // end of while length(textel)>0 do
                // ми конфігуруємо вікно
                FOnOff.Enable_Screen;
                FOnOff.Actif:=True;
                FOnOff.Show;
              end else // end of if uppercase(textel1)='ONOFF' then
            if uppercase(textel1)='PWM' then
              begin
                //CLK=,41/PWM/EXIT=GLOBAL/NAME=Динамічне управління
частотою/TITLE=OUT 4 - CN 5.24/MODE=FREQ/TYPE=16L/CDE=30 04 00 VALUE 00;
                FPWM.diviseur:=1;
                FPWM.Pas:=1;
                FPWM.sens_octet:=CT_SensHL;
                while length(textel)>0 do
                  begin
                    j:=pos('/',textel);
                    if j>0 then
                      begin
                        textel1:=copy(textel,1,j-1);
                        delete(textel,1,j);
                      end else
                      begin
                        textel1:=textel;
                        textel:='';
                      end;
                    if copy(textel1,1,5)='Вихід =' then
                      begin
                        delete(textel1,1,5);
                        if uppercase(textel1)='GLOBAL'
                          then FPWM.Cde_PWM_Exit:=''
                          else FPWM.Cde_PWM_Exit:=textel1;
                      end else
                    if copy(textel1,1,5)='Тип =' then
                      begin
                        delete(textel1,1,5);
                        if pos('L',textel1)>0 then
                          begin
                            FPWM.sens_octet:=CT_SensLH;
                            delete(textel1,pos('L',textel1),1);
                          end else
                        if pos('H',textel1)>0 then
                          begin
                            delete(textel1,pos('H',textel1),1);
                          end;
                        try
                          FPWM.lg_data:=strtoint(textel1);
                        except
                          FPWM.lg_data:=8;
                        end;
                      end else
                    if copy(textel1,1,4)='CDE=' then
                      begin
                        delete(textel1,1,4);
                        FPWM.Cde_PWM:=textel1;
```

```
                                    end else
                                  if copy(textel1,1,4)='DIV=' then
                                    begin
                                      delete(textel1,1,4);
                                      FPWM.diviseur:=strtoreel(textel1);
                                    end else
                                  if copy(textel1,1,4)='Прохід  =' then
                                    begin
                                      delete(textel1,1,4);
                                      FPWM.Pas:=strtoreel(textel1);
                                    end else
                                  if copy(textel1,1,5)='Модуль  =' then
                                    begin
                                      delete(textel1,1,5);
                                      FPWM.Unite:=textel1;
                                    end else
                                  if copy(textel1,1,5)='Ім'я =' then
                                    begin
                                      delete(textel1,1,5);
                                      FPWM.Caption:=textel1;
                                    end else
                                  if copy(textel1,1,6)='Назва =' then
                                    begin
                                      delete(textel1,1,6);
                                      FPWM.L_Pin.Caption:=textel1;
                                    end;
                                end;
                              FPWM.Calcul_Min_Max;
                              FPWM.Actif:=True;
                              FPWM.Enable_Screen;
                              FPWM.B_Fermer.Caption:=MenuProg[CMP_Close];
                              FPWM.Show;
                            end else  // end of if uppercase(textel1)='PWM' then
                          if uppercase(textel1)='BRIDGE' then
                            begin
                              // CLK=,76/BRIDGE/EXIT=GLOBAL/NAME=Управління
мостом n°3/TITLE=OUT 13 &
14/STOP=00/SENS1=01/SENS2=02/DIV=50/TYPE=8/UNIT=Hz/PAS=50/CDE=30 11 EXV SENS
VALUE;
                              FBridge.diviseur:=1;
                              FBridge.Pas:=1;
                              FBridge.sens_octet:=CT_SensHL;
                              while length(textel)>0 do
                                begin
                                  j:=pos('/',textel);
                                  if j>0 then
                                    begin
                                      textel1:=copy(textel,1,j-1);
                                      delete(textel,1,j);
                                    end else
                                    begin
                                      textel1:=textel;
                                      textel:='';
                                    end;
                                  if copy(textel1,1,5)='Вихід =' then
                                    begin
                                      delete(textel1,1,5);
                                      if uppercase(textel1)='GLOBAL'
                                        then FBridge.Cde_Exit:=''
                                        else FBridge.Cde_Exit:=textel1;
                                    end else
                                  if copy(textel1,1,5)='Тип =' then
                                    begin
                                      delete(textel1,1,5);
                                      if pos('L',textel1)>0 then
                                        begin
                                          FBridge.sens_octet:=CT_SensLH;
                                          delete(textel1,pos('L',textel1),1);
                                        end else
```

```
                    if pos('H',textel1)>0 then
                      begin
                        delete(textel1,pos('H',textel1),1);
                      end;
                    try
                      FBridge.lg_data:=strtoint(textel1);
                    except
                      FBridge.lg_data:=8;
                    end;
                  end else
                if copy(textel1,1,4)='CDE=' then
                  begin
                    delete(textel1,1,4);
                    FBridge.Cde_PWM:=textel1;
                  end else
                if copy(textel1,1,5)=Умова остановки =' then
                  begin
                    delete(textel1,1,5);
                    FBridge.Cde_Stop:=textel1;
                  end else
                if copy(textel1,1,6)='SENS1=' then
                  begin
                    delete(textel1,1,6);
                    FBridge.Cde_Sens1:=textel1;
                  end else
                if copy(textel1,1,6)='SENS2=' then
                  begin
                    delete(textel1,1,6);
                    FBridge.Cde_Sens2:=textel1;
                  end else
                if copy(textel1,1,4)='DIV=' then
                  begin
                    delete(textel1,1,4);
                    FBridge.diviseur:=strtoreel(textel1);
                  end else
                if copy(textel1,1,4)='Прохід  =' then
                  begin
                    delete(textel1,1,4);
                    FBridge.Pas:=strtoreel(textel1);
                  end else
                if copy(textel1,1,5)='Модуль  =' then
                  begin
                    delete(textel1,1,5);
                    FBridge.Label_Unite:=textel1;
                  end else
                if copy(textel1,1,5)='Ім'я =' then
                  begin
                    delete(textel1,1,5);
                    FBridge.Caption:=textel1;
                  end else
                if copy(textel1,1,6)='Назва =' then
                  begin
                    delete(textel1,1,6);
                    FBridge.L_Pin.Caption:=textel1;
                  end;
              end;
            FBridge.Calcul_Mini_Maxi;
            FBridge.Actif:=True;
            FBridge.Enable_Screen;
            FBridge.B_Fermer.Caption:=MenuProg[CMP_Close];
            FBridge.Show;
          end else  // end of if uppercase(textel1)='BRIDGE'
then
        if uppercase(textel1)='WIPER' then
          begin
             while length(textel)>0 do
              begin
                j:=pos('/',textel);
                if j>0 then
```

```
                    begin
                      textel1:=copy(textel,1,j-1);
                      delete(textel,1,j);
                    end else
                    begin
                      textel1:=textel;
                      textel:='';
                    end;
                  if copy(textel1,1,5)='Вихід =' then
                    begin
                      delete(textel1,1,5);
                      if uppercase(textel1)='GLOBAL'
                        then FWiper.Cde_Wiper_Exit:=''
                        else FWiper.Cde_Wiper_Exit:=textel1;
                    end else
                  if copy(textel1,1,3)=Умова остановки =' then
                    begin
                      delete(textel1,1,3);
                      FWiper.Cde_Wiper_Stop:=textel1;
                    end else
                  if copy(textel1,1,4)='SLOW=' then
                    begin
                      delete(textel1,1,4);
                      FWiper.Cde_Wiper_Slow:=textel1;
                    end else
                  if copy(textel1,1,4)='FAST=' then
                    begin
                      delete(textel1,1,4);
                      FWiper.Cde_Wiper_Fast:=textel1;
                    end else
                  if copy(textel1,1,5)='Ім'я =' then
                    begin
                      delete(textel1,1,5);
                      FWiper.Caption:=textel1;
                    end else
                  if copy(textel1,1,6)='Назва =' then
                    begin
                      delete(textel1,1,6);
                      FWiper.L_Pin.Caption:=textel1;
                    end;
                end;
              FWiper.Actif:=True;
              FWiper.Enable_Screen;
              FWiper.B_Fermer.Caption:=MenuProg[CMP_Close];
              FWiper.Show;
            end else  // end of if uppercase(textel1)='WIPER'
then
          if uppercase(textel1)='BIPWM' then
            begin
              FBiPWM.diviseur_1:=1;
              FBiPWM.Pas_1:=1;
              FBiPWM.diviseur_2:=1;
              FBiPWM.Pas_2:=1;
              FBiPWM.sens_octet1:=CT_SensHL;
              FBiPWM.sens_octet2:=CT_SensHL;
              while length(textel)>0 do
                begin
                  j:=pos('/',textel);
                  if j>0 then
                    begin
                      textel1:=copy(textel,1,j-1);
                      delete(textel,1,j);
                    end else
                    begin
                      textel1:=textel;
                      textel:='';
                    end;
                  if copy(textel1,1,5)='Вихід =' then
                    begin
```

```
    delete(textel1,1,5);
    if uppercase(textel1)='GLOBAL'
      then FBiPWM.Cde_Exit:=''
      else FBiPWM.Cde_Exit:=textel1;
  end else
if copy(textel1,1,6)='TYPE1=' then
  begin
    delete(textel1,1,6);
    if pos('L',textel1)>0 then
      begin
        FBiPWM.sens_octet1:=CT_SensLH;
        delete(textel1,pos('L',textel1),1);
      end else
    if pos('H',textel1)>0 then
      begin
        delete(textel1,pos('H',textel1),1);
      end;
    try
      FBiPWM.lg_data_1:=strtoint(textel1);
    except
      FBiPWM.lg_data_1:=8;
    end;
  end else
if copy(textel1,1,6)='TYPE2=' then
  begin
    delete(textel1,1,6);
    if pos('L',textel1)>0 then
      begin
        FBiPWM.sens_octet2:=CT_SensLH;
        delete(textel1,pos('L',textel1),1);
      end else
    if pos('H',textel1)>0 then
      begin
        delete(textel1,pos('H',textel1),1);
      end;
    try
      FBiPWM.lg_data_2:=strtoint(textel1);
    except
      FBiPWM.lg_data_2:=8;
    end;
  end else
if copy(textel1,1,4)='CDE=' then
  begin
    delete(textel1,1,4);
    FBiPWM.Cde_BiPWM:=textel1;
  end else
if copy(textel1,1,5)='DIV1=' then
  begin
    delete(textel1,1,5);
    FBiPWM.diviseur_1:=strtoreel(textel1);
  end else
if copy(textel1,1,5)='DIV2=' then
  begin
    delete(textel1,1,5);
    FBiPWM.diviseur_2:=strtoreel(textel1);
  end else
if copy(textel1,1,5)='PAS1=' then
  begin
    delete(textel1,1,5);
    FBiPWM.Pas_1:=strtoreel(textel1);
  end else
if copy(textel1,1,6)='UNIT1=' then
  begin
    delete(textel1,1,6);
    FBiPWM.Label_Unit1:=textel1;
  end else
if copy(textel1,1,5)='PAS2=' then
  begin
    delete(textel1,1,5);
```

```
                    FBiPWM.Pas_2:=strtoreel(textel1);
                  end else
                if copy(textel1,1,6)='UNIT2=' then
                  begin
                    delete(textel1,1,6);
                    FBiPWM.Label_Unit2:=textel1;
                  end else
                if copy(textel1,1,5)='Ім'я =' then
                  begin
                    delete(textel1,1,5);
                    FBiPWM.Caption:=textel1;
                  end else
                if copy(textel1,1,6)='Назва =' then
                  begin
                    delete(textel1,1,6);
                    FBiPWM.L_Pin.Caption:=textel1;
                  end;
              end;
            FBiPWM.Calcul_Min_Max;
            FBiPWM.Actif:=True;
            FBiPWM.Enable_Screen;
            FBiPWM.B_Fermer.Caption:=MenuProg[CMP_Close];
            FBiPWM.Show;
          end else // end of if uppercase(textel1)='BIPWM' then
        if uppercase(textel1)='VALUE' then
          begin
            FValue.Cde_Exit:='';
            FValue.Cde_Send:='';
            FValue.E_Value.Text:='';
            FValue.diviseur:=1;
            FValue.sens_octet:=CT_SensHL;
            while length(textel)>0 do
              begin
                j:=pos('/',textel);
                if j>0 then
                  begin
                    textel1:=copy(textel,1,j-1);
                    delete(textel,1,j);
                  end else
                  begin
                    textel1:=textel;
                    textel:='';
                  end;
                //
VALUE/EXIT=GLOBAL/TYPE=8/NAME=Співвідношення/TITLE=xxx/DIV=1/CDE=21 40 VAL 00
                if copy(textel1,1,5)='Вихід =' then
                  begin
                    delete(textel1,1,5);
                    if uppercase(textel1)='GLOBAL'
                      then FValue.Cde_Exit:=''
                      else FValue.Cde_Exit:=textel1;
                  end else
                if copy(textel1,1,5)='Тип =' then
                  begin
                    delete(textel1,1,5);
                    if pos('L',textel1)>0 then
                      begin
                        FValue.sens_octet:=CT_SensLH;
                        delete(textel1,pos('L',textel1),1);
                      end else
                    if pos('H',textel1)>0 then
                      begin
                        delete(textel1,pos('H',textel1),1);
                      end;
                    try
                      FValue.lg_data:=strtoint(textel1);
                    except
                      FValue.lg_data:=8;
                    end;
```

```
                                    end else
                                  if copy(textel1,1,4)='CDE=' then
                                    begin
                                      delete(textel1,1,4);
                                      FValue.Cde_Send:=textel1;
                                    end else
                                  if copy(textel1,1,4)='DIV=' then
                                    begin
                                      delete(textel1,1,4);
                                      FValue.diviseur:=strtoreel(textel1);
                                    end else
                                  if copy(textel1,1,5)='Ім'я =' then
                                    begin
                                      delete(textel1,1,5);
                                      FValue.Caption:=textel1;
                                    end else
                                  if copy(textel1,1,6)='Назва =' then
                                    begin
                                      delete(textel1,1,6);
                                      FValue.L_Pin.Caption:=textel1;
                                    end else
                                  if copy(textel1,1,5)='Вибір =' then
                                    begin
                                      delete(textel1,1,5);
                                      j:=pos(',',textel1);
                                      try
FValue.ColGrid:=strtoint(copy(textel1,1,j-1));
                                      except
                                        FValue.ColGrid:=-1;
                                      end;
                                      delete(textel1,1,j);
                                      try
                                        FValue.RowGrid:=strtoint(textel1);
                                      except
                                        FValue.RowGrid:=-1;
                                      end;
                                      if (FValue.ColGrid>-1)and (FValue.RowGrid>-
1) then
                                        begin
FValue.E_Value.Text:=FMain.StringGrid1.Cells[FValue.ColGrid,FValue.RowGrid];
                                        end;
                                      FValue.L_Pin.Caption:=textel1;
                                    end;
                                end;

                                FValue.Actif:=True;
                                FValue.Enable_Screen;
                                FValue.B_Fermer.Caption:=MenuProg[CMP_Close];
                                FValue.Show;
                              end else; // end of if uppercase(textel1)='VALUE'
then
                            if uppercase(textel1)='LIST' then
                              begin
                                FLIST.Cde_Exit:='';
                                FList.Cde_Send:='';
                                FList.E_Value.Text:='';
                                while length(textel)>0 do
                                  begin
                                    j:=pos('/',textel);
                                    if j>0 then
                                      begin
                                        textel1:=copy(textel,1,j-1);
                                        delete(textel,1,j);
                                      end else
                                      begin
                                        textel1:=textel;
                                        textel:='';
```

```
                                    end;
                                  // CLK=,1/LIST/EXIT=/CASE=1,2-9/NAME=Введіть
дату й час/
                                  // TITLE=Дата й
час/COR=0.25|0:1|0:1|0:1|0:0.25|0:1|0:1|0:1|0/
                                  // TYPE=8:8:8:8:8:8:8/CDE=2E F9 0B VALUE;
                                  if copy(textel1,1,5)='Вихід =' then
                                    begin
                                      delete(textel1,1,5);
                                      if uppercase(textel1)='GLOBAL'
                                        then FList.Cde_Exit:=''
                                        else FList.Cde_Exit:=textel1;
                                    end else
                                  if copy(textel1,1,5)='Тип =' then
                                    begin
                                      delete(textel1,1,5);
                                      FList.type_value:=textel1;
                                    end else
                                  if copy(textel1,1,4)='CDE=' then
                                    begin
                                      delete(textel1,1,4);
                                      FList.Cde_Send:=textel1;
                                    end else
                                  if copy(textel1,1,4)='COR=' then
                                    begin
                                      delete(textel1,1,4);
                                      FList.correction:=textel1;
                                    end else
                                  if copy(textel1,1,5)='Ім'я =' then
                                    begin
                                      delete(textel1,1,5);
                                      FList.Caption:=textel1;
                                    end else
                                  if copy(textel1,1,6)='Назва =' then
                                    begin
                                      delete(textel1,1,6);
                                      FList.L_Pin.Caption:=textel1;
                                    end else
                                  if copy(textel1,1,5)='Вибір =' then
                                    begin
                                      delete(textel1,1,5);

FList.CaseToInteger(textel1,FList.ListColMin,FList.ListColMax,Flist.ListRowMin,F
List.ListRowMax);

FList.StringGrid1.ColCount:=FMain.StringGrid1.ColCount;
                                      j:=FList.ListRowMax-FList.ListRowMin+2;
                                      FList.StringGrid1.RowCount:=j;
                                      for j:=0 to FList.StringGrid1.ColCount-1 do

FList.StringGrid1.Cells[j,0]:=FMain.StringGrid1.Cells[j,0];
                                      for j:=0 to FList.StringGrid1.ColCount-1 do
                                        for k:=FList.ListRowMin to
FList.ListRowMax do
                                          begin
                                            FList.StringGrid1.Cells[j,k-
FList.ListRowMin+1]:=FMain.StringGrid1.Cells[j,k];
                                          end;
                                    end;
                                end;

                              FList.Actif:=True;
                              FList.Enable_Screen;
                              FList.B_Fermer.Caption:=MenuProg[CMP_Close];
                              FList.Show;
                            end; // end of if uppercase(textel1)='LIST' then
                          end; // end of while length(textel)>0 do
```

```
                    end; // end of if ((y>=0) and (x>=0) and (x=aCol) and
(y=ligne))
                end; // end of  if j>0 then
             end;  // end of if j>0 then
          end; // end of if uppercase(copy(textel,1,4))='CLK=' then
        i:=i+1;
      end; // end of while i<ClickList.Count do
end;

//*****************************************************************************
procedure TFMain.StringGrid1DrawCell(Sender: TObject; aCol, aRow: Integer;
  aRect: TRect; aState: TGridDrawState);
var d : integer;
    couleur : TColor;
    bRect : TRect;
    textel : string;
begin
  if FMain.ClickTree then exit;
  if aRow>0 then
    begin
      try
        if length(FMain.ColorList.Strings[aRow])>0
           then couleur:=hextodec(FMain.ColorList.Strings[aRow])
           else couleur:=clWhite;
      except
        couleur:=clWhite;
      end;
    end else couleur:=clSkyBlue;
  FMain.StringGrid1.Canvas.Brush.Color:=couleur;
  FMain.StringGrid1.Canvas.FillRect(aRect);
  if aRow=0 then
    begin
      FMain.StringGrid1.Canvas.Font.Color:=clBlack;

FMain.StringGrid1.Canvas.TextOut(aRect.Left,aRect.Top,FMain.StringGrid1.Cells[aC
ol,aRow]);
      exit; // Якщо залишемо на дисплеї назви прапорців
    end;
  d:=(FMain.StringGrid1.RowHeights[aRow]-13) div 2;
  bRect.Top:=aRect.Top+d;
  bRect.Bottom:=bRect.Top+13;
  d:=(FMain.StringGrid1.ColWidths[aCol]-13)div 2;
  bRect.Left:=aRect.Left+d;
  bRect.Right:=bRect.Left+13;
  textel:=FMain.StringGrid1.Cells[aCol,aRow];
  if copy(textel,1,4)='IMG_' then
    begin
      delete(textel,1,4);
      if copy(textel,1,3)='RED' then
       begin
        if copy(textel,4,1)='1' then
          begin
            FMain.StringGrid1.Canvas.Draw(bRect.Left,bRect.Top,ledrougeon);
          end else
          begin
            FMain.StringGrid1.Canvas.Draw(bRect.Left,bRect.Top,ledrougeoff);
          end;
       end else
      if copy(textel,1,3)='AMB' then
        begin
          if copy(textel,6,1)='1' then
            begin
              FMain.StringGrid1.Canvas.Draw(bRect.Left,bRect.Top,ledamberon);
            end else
            begin
              FMain.StringGrid1.Canvas.Draw(bRect.Left,bRect.Top,ledamberoff);
            end;
        end else
      if copy(textel,1,3)='GRE' then
```

```
        begin
          if copy(textel,6,1)='1' then
            begin
              FMain.StringGrid1.Canvas.Draw(bRect.Left,bRect.Top,ledverteon);
              //FMain.StringGrid1.Canvas.Brush.Color:=couleur;
              //FMain.StringGrid1.Canvas.FillRect(aRect);
            end else
            begin
              FMain.StringGrid1.Canvas.Draw(bRect.Left,bRect.Top,ledverteoff);
              //FMain.StringGrid1.Canvas.Brush.Color:=clGreen;
              //FMain.StringGrid1.Canvas.FillRect(aRect);
            end;
        end;
    end else
    begin
      FMain.StringGrid1.Canvas.Font.Color:=clBlack;

FMain.StringGrid1.Canvas.TextOut(aRect.Left,aRect.Top,FMain.StringGrid1.Cells[aC
ol,aRow]);
    end;

end;

//*****************************************************************************
procedure TFMain.StringGrid1Resize(Sender: TObject);
begin
  //FMain.rafraichir_diag;
end;

//*****************************************************************************
procedure TFMain.T_AttenteTimer(Sender: TObject);
begin
  FMain.num_image:=FMain.num_image+1;
  if FMain.num_image>14 then FMain.num_image:=1;
  FMain.I_Attente.Picture.Bitmap:=nil;
  FMain.IList_Rose.GetBitmap(FMain.num_image,FMain.I_Attente.Picture.Bitmap);
end;

procedure TFMain.T_CommandTimer(Sender: TObject);
var textel,textel1 : string;
    liste_connection : TStringList;
    ListCommand : TStringList;
    cond : string; // вимога повторити команду
    cde : string; // команда відправки
    i,j,k,l,m,n : integer;
    outdoor : boolean;
    byteRule: string;
    DebutByte,FinByte,DebutBit,FinBit : integer;
    valeur,val_cond : integer;
    f_Byte,f_Bit : string;
    mask : byte;
    n_ligne,max_val,min_val : integer;
    Noeud : TTreeNode;
begin
  // Виконати всі команди пошуку  відразу після з'єднання
  FMain.T_Command.Enabled:=False;
  try
    liste_Connection:=TStringlist.Create;
    ListCommand:=TStringList.Create;
    FMain.Search_Item('CONNECTION',FMain.Vehicule,Liste_Connection);
    //Liste_Connection.SaveToFile('c:\Connection.txt');
    i:=0;
    FMain.TypeAff:=-1;
    while i<Liste_Connection.Count do
      begin
        Application.ProcessMessages;
        if MainExit then exit;
        textel:=Liste_Connection.Strings[i];
            if uppercase(copy(textel,1,9))='<COMMAND>' then
```

```
            begin
          ListCommand.Clear;
          while (textel<>'</COMMAND>') and (i<Liste_Connection.Count)do
            begin
              textel:=Liste_Connection.Strings[i];
              textel:=trim(textel);
              ListCommand.Add(textel);
              if textel<>'</COMMAND>' then i:=i+1;
            end;
          //ListCommand.SaveToFile('c:\Command'+inttostr(i)+'.txt');
          // ми отримуємо все поле COMMAND
          // Таким чином, ми виконаємо команду
          j:=0;
          while j<ListCommand.Count do
            begin
              Application.ProcessMessages;
              if MainExit then exit;
              textel:=trim(ListCommand.Strings[j]);
              if copy(textel,1,5)='COND=' then
                begin
                  delete(textel,1,5);
                  cond:=textel;
                end else
              if copy(textel,1,5)='<_ISO14230_>' then
                begin
                  FMain.Actual_ISO14230.Clear;
                  cde:='';
                  while j<ListCommand.Count do
                    begin
                      textel:=trim(ListCommand.Strings[j]);
                      if copy(textel,1,2)<>'</' then
FMain.Actual_ISO14230.Add(textel);
                      if copy(textel,1,4)='CDE=' then cde:=textel;
                      if copy(textel,1,2)='</' then j:=ListCommand.Count;
                      j:=j+1;
                    end;
                end;
              j:=j+1;
            end; // end of while j<ListCommand.Count do
        end;  // end of if uppercase(copy(textel,1,9))='<COMMAND>' then
      if length(cde)>0 then
        begin
          delete(cde,1,4);
          FMain.ISO14230.Sid:=hextodec(copy(cde,1,2));
          delete(cde,1,3);
          j:=0;
          while length(cde)>0 do
            begin
              Application.ProcessMessages;
              if MainExit then Exit;
              FMain.ISO14230.FData[j]:=hextodec(copy(cde,1,2));
              delete(cde,1,3);
              j:=j+1;
            end;
          FMain.ISO14230.SendLength:=j;
          if length(cond)>0 then
            begin
              outdoor:=false;
              // Розрахунок фільтра на дані
              j:=pos('/',cond);
                  ByteRule:=copy(cond,1,j-1);
                  delete(cond,1,j);
                  // видаляє невідфільтровані дані
                  j:=pos('[',ByteRule);
                  if j>0 then delete(ByteRule,1,j);
                  j:=pos(']',ByteRule);
                  if j>0 then delete(ByteRule,j,1);
                  j:=pos('|',ByteRule);
                  // Отримує дані з бітів
```

```
if j>0 then
 begin
   f_Byte:=copy(ByteRule,1,j-1);
   delete(ByteRule,1,j);
   f_bit:=ByteRule;
 end else
 begin
   f_Byte:=ByteRule;
   f_bit:='';
 end;
j:=pos('-',f_Byte);
if j>0 then
  begin
    textel1:=copy(f_Byte,1,j-1);
    try
      DebutByte:=Strtoint(textel1);
    except
      DebutByte:=0;
    end;
    delete(f_byte,1,j);
    try
      FinByte:=Strtoint(f_Byte);
    except
      FinByte:=0;
    end;
  end else
  begin
    try
      DebutByte:=Strtoint(f_Byte);
    except
      DebutByte:=0;
    end;
    FinByte:=DebutByte;
  end;
// отримує площину біт
if length(f_bit)>0 then  // відфільтровує біти
 begin
   j:=pos('-',f_Bit);
   if j>0 then  // є площина
     begin
       textel1:=copy(f_Bit,1,j-1);
       try
         DebutBit:=Strtoint(textel1);
       except
         DebutBit:=0;
       end;
       delete(f_bit,1,j);
       try
         FinBit:=Strtoint(f_Bit);
       except
         FinBit:=0;
       end;
     end else
     begin   //Немає площини
       try
         DebutBit:=Strtoint(f_Bit);
       except
         DebutBit:=0;
       end;
       FinBit:=DebutBit;
     end;
   // відновлення створення маски
   mask:=0;
   for j:=DebutBit to FinBit do
     mask:=mask or (1 shl j);
 end else mask:=$FF;
textel:=cond;
delete(textel,1,1);
try
```

```
                    val_cond:=strtoint(textel);
                  except
                    outdoor:=true;
                  end;
                  // повторення виконання
                  beginExit:=false;
                  While (outdoor=False) and (BeginExit=False) do
                 begin
                  Application.ProcessMessages;
                  if MainExit then Exit;
                  FMain.ISO14230.SendResponse;
                  FMain.Attente_Start;
                  Execution:=False;
                  While (Execution=False)
                    and (MainExit=False) do Application.ProcessMessages;
                  if MainExit then Exit;
                  // визначення фільтра для байта
                  valeur:=0;
                  for j:=DebutByte to FinByte do
                    begin
                      valeur:=valeur*256+(FMain.ISO14230.TrameRx[j] and mask);
                    end;
                  if copy(cond,1,1)='=' then
                    begin
                      if valeur=val_cond then outdoor:=true;
                    end else
                  if copy(cond,1,1)='>' then
                    begin
                      if valeur>val_cond then outdoor:=True;
                    end else
                  if copy(cond,1,1)='<' then
                    begin
                      if valeur<val_cond then outdoor:=True;
                    end else outdoor:=True; // є помилка

                 end;
              end else
              begin
                FMain.ISO14230.SendResponse;
                FMain.Attente_Start;
                Execution:=False;
                While (Execution=False)
                  and (MainExit=False)do Application.ProcessMessages;
                if MainExit then Exit;
              end;
          end;
        i:=i+1;
      end; // end of  while i<ListConnect.Count do
finally
  liste_Connection.Free;
  ListCommand.Free;
end;
// Виведення меню
i:=0;
while i<Vehicule.Count do
  begin
    Application.ProcessMessages;
    if MainExit then Exit;
    textel:=Vehicule.Strings[i];
    trim(textel);
    if uppercase(copy(textel,1,4))='ICO=' then
      begin
        //noeud:=FMain.TreeView1.Items.GetFirstNode;
        if noeud<>nil then
          begin
            delete(textel,1,4);
            j:=pos(';',textel);
            try
              j:=strtoint(copy(textel,1,j-1));
```

```
              except
                j:=-1;
              end;
              noeud.ImageIndex:=j;
            end;
        end else
      if uppercase(copy(textel,1,8))='<EXMENU=' then
        begin
          DebutByte:=i;
          FMain.ExtendedMenu.Clear;
          delete(textel,2,2);
          FMain.ExtendedMenu.Add(textel);
          // відновлення розширеного меню, операція повинна бути повторена
          outdoor:=False;
          i:=i+1;
          while outdoor=False do
            begin
              Application.ProcessMessages;
              if MainExit then Exit;
              if i<Vehicule.Count then textel:=Vehicule.Strings[i];
              trim(textel);
              if (i>=Vehicule.Count)
                or (uppercase(copy(textel,1,8))='<EXMENU=')
                or (uppercase(copy(textel,1,8))='</EXMENU')
                or (uppercase(copy(textel,1,6))='<MENU=') then outdoor:=True;
              if not outdoor then
                begin
                  if (length(textel)>0)
                    and (copy(textel,1,2)<>'//') then
                    begin
                      k:=pos(';',textel);
                      if k=0 then FMain.ExtendedMenu.Add(textel) else
FMain.ExtendedMenu.Add(copy(textel,1,k));
                    end;
                  i:=i+1;
                end else FinByte:=i;
            end;
          //FMain.ExtendedMenu.SaveToFile('c:\ExtendedMenu.txt');
          // Видалення з розширеного меню файлу з даними автомобіля
          for j:=DebutByte to FinByte do
            begin
              FMain.Vehicule.Delete(DebutByte);
            end;
          // ми повинні відновити умови повторення val_cond
          j:=0;
          while j<FMain.ExtendedMenu.Count do
            begin
              Application.ProcessMessages;
              if MainExit then Exit;
              textel:=FMain.ExtendedMenu.Strings[j];
              if copy(textel,1,9)='EXREPEAT=' then
                begin
                  delete(textel,1,9);
                  k:=pos(';',textel);
                  if k>0 then textel:=copy(textel,1,k-1);
                  textel:=trim(textel);
                  try
                    val_cond:=strtoint(textel);   // повторення змінної
                  except
                    val_cond:=-1;
                  end;
                  FMain.ExtendedMenu.Delete(j); // Стираємо це попередження
                  j:=FMain.ExtendedMenu.Count;
                end;
              j:=j+1;
            end;
          if copy(FMain.ExtendedMenu.Strings[FMain.ExtendedMenu.Count-
1],1,7)<>'</MENU>'
             then FMain.ExtendedMenu.Add('</MENU>');
```

```
            // Ми починаємо меню інтеграції  файлу  автомобіля до пам'яті
            if (val_cond>-1) and (FMain.Variable[Val_cond]>0) then
              begin
                l:=0;
                n_ligne:=0;
                max_val:=0;
                min_val:=10000;
                m:=-1;
                for j:=1 to FMain.Variable[Val_cond] do
                  begin
                    for k:=0 to FMain.ExtendedMenu.Count-1 do
                      begin
                        Application.ProcessMessages;
                        if MainExit then Exit;
                        textel:=FMain.ExtendedMenu.Strings[k];
                        if copy(textel,1,4)='<MEN' then
                          begin
                            debutBit:=pos('>',textel);
                            if debutBit>0 then textel:=copy(textel,1,debutbit-1);
                            textel:=textel+' '+inttostr(j)+'>';
                          end else
                        if copy(textel,1,4)='CLK=' then
                          begin
                            m:=pos('EXV',textel);
                            textel1:=dectohex(j);
                            while m>0 do
                              begin
                                textel:=copy(textel,1,m-
1)+textel1+copy(textel,m+3,length(textel)-3);
                                m:=pos('EXV',textel);
                              end;
                          end else
                        if copy(textel,1,4)='CDE=' then
                          begin
                            debutBit:=pos(';',textel);
                            if debutBit>0 then textel:=copy(textel,1,debutbit-1);
                            textel:=textel+' '+dectohex(j)+';';
                          end else
                        if copy(textel,1,3)='LG=' then
                          begin
                            debutbit:=pos('TAB[',textel);
                            if debutbit>0 then
                              begin
                                cond:=copy(textel,1,debutbit+3);
                                delete(textel,1,debutbit+3);
                                debutbit:=pos(',',textel);
                                cond:=cond+copy(textel,1,debutbit);
                                delete(textel,1,debutbit);
                                debutbit:=pos(']',textel);
                                valeur:=strtoint(copy(textel,1,debutbit-1));
                                delete(textel,1,debutbit-1);
                                if max_val<valeur then max_val:=valeur;
                                if min_val>valeur then min_val:=valeur;
                                valeur:=valeur+n_ligne;
                                textel:=cond+inttostr(valeur)+textel;
                              end;
                          end else
                        if copy(textel,1,3)='TR=' then
                          begin
                            debutbit:=pos('TAB[',textel);
                            if debutbit>0 then
                              begin
                                cond:=copy(textel,1,debutbit+3);
                                delete(textel,1,debutbit+3);
                                debutbit:=pos(',',textel);
                                cond:=cond+copy(textel,1,debutbit);
                                delete(textel,1,debutbit);
                                debutbit:=pos(']',textel);
                                valeur:=strtoint(copy(textel,1,debutbit-1));
```

```
                                delete(textel,1,debutbit-1);
                                if max_val<valeur then max_val:=valeur;
                                if min_val>valeur then min_val:=valeur;
                                valeur:=valeur+n_ligne;
                                textel:=cond+inttostr(valeur)+textel;
                              end;
                          end;
                        FMain.Vehicule.Insert(DebutByte+k+((j-
1)*FMain.ExtendedMenu.Count),textel);
                      end;
                    // розраховуємо l
                    // n_ligne => текст у таблицю
                    // valeur => читати останнє значення
                    n_ligne:=n_ligne+(max_val-min_val)+1;
                  end;
              end;
            // Ми закінчили підключення, це переводить курсор на початок цієї
області
            i:=DebutByte-1;
       end else
       if uppercase(copy(textel,1,6))='<MENU=' then
         begin
         delete(textel,1,6);
         j:=pos('>',textel);
         if j>0 then
           begin
             MenuList.Add(copy(textel,1,j-1)+'/'+inttostr(i)+'/');
             noeud:=FMain.TreeView1.Items.Add(nil,copy(textel,1,j-1));
           end;
       end else
       if uppercase(copy(textel,1,6))='</MENU' then
         begin
         delete(textel,1,6);
         if MenuList.Count>0 then MenuList.Strings[MenuList.Count-
1]:=MenuList.Strings[MenuList.Count-1]+inttostr(i)+'/';
       end ;
     i:=i+1;
   end;
   //FMain.TreeView1.Enabled:=True;
   FMain.Vehicule.SaveToFile('C:\vehicule.txt');
   FMain.TreeView1.Enabled:=True;
end;

//*****************************************************************************
// Цей таймер використовується для виконання всіх _ISO14230_ Cde   меню, якщо
треба повторити умови, цей таймер є виконуваний іншим разом
//*****************************************************************************
procedure TFMain.T_StatusIOTimer(Sender: TObject);
var i,j : integer;
    textel : string;
    outdoor : boolean;
begin
   InIOTimer:=True;
   FMain.T_StatusIO.Enabled:=False;
   i:=FMain.ligneMenu;
   textel:='';
   if length(FOnOff.Buffer)>0 then
     begin
       textel:=FOnOff.Buffer;
       FOnOff.Buffer:='';
     end else
   if length(FbiPWM.Buffer)>0 then
     begin
       textel:=FBiPWM.Buffer;
       FBiPWM.Buffer:='';
     end else
   if length(FPwm.Buffer)>0 then
     begin
       textel:=FPWM.Buffer;
```

```
      FPWM.Buffer:='';
    end else
  if length(FBridge.Buffer)>0 then
    begin
      textel:=FBridge.Buffer;
      FBridge.Buffer:='';
    end else
  if length(Fvalue.Buffer)>0 then
    begin
      textel:=FValue.Buffer;
      FValue.Buffer:='';
      if textel='#' then
        begin
          // Створюємо запит на читання
          FMain.TreeView1Click(nil);
          exit;
        end;
    end else
  if length(FList.Buffer)>0 then
    begin
      textel:=FList.Buffer;
      FList.Buffer:='';
      if textel='#' then
        begin
          // Створюємо запит на читання
          FMain.TreeView1Click(nil);
          exit;
        end;
    end else
  if length(Fwiper.Buffer)>0 then
    begin
      textel:=FWiper.Buffer;
      FWiper.Buffer:='';
    end;
  // якщо користувач нажимає на кнопку
  if length(textel)>0 then
    begin
      While (FMain.ISO14230.EndReceive=False) and (MainExit=False) do
Application.ProcessMessages;
      if MainExit then Exit;
      FMain.Cde2Send(textel);
      FMain.Attente_Start;
      FMain.Execution:=False;
      //while FMain.Execution= false do Application.ProcessMessages;
      FMain.NoReceive:=True;
      FMain.ISO14230.SendResponse;
      FOnOff.Buffer:='';
      if (i<FMain.ActualMenu.Count) or
        ((FMain.TimeRepeat>0) and (FMain.TypeAff<>CT_AFFCONFIRM)) then
        begin
          FMain.T_StatusIO.Interval:=100;
          FMain.T_StatusIO.Enabled:=True;
        end;
      InIoTimer:=False;
      exit;
    end;
  // ми отримуємо поточну команду по протоколу _ISO14230_
  outdoor:=False;
  FMain.Actual_ISO14230.Clear;
  if (i<FMain.ActualMenu.Count) then
    begin

      // ми отримуємо поточну команду по протоколу _ISO14230_
      While (i<FMain.ActualMenu.Count) and (OutDoor=False) do
        begin
          Application.ProcessMessages;
          if MainExit then Exit;
          textel:=trim(FMain.ActualMenu.Strings[i]);
          if uppercase(copy(textel,1,5))='<_ISO14230_>' then
```

```
            begin
              i:=i+1;
              while (i<FMain.ActualMenu.Count) and (outdoor=False) do
                begin
                  textel:=trim(FMain.ActualMenu.Strings[i]);
                  if uppercase(textel)<>'</_ISO14230_>' then
                    begin
                      Actual_ISO14230.Add(textel);
                      i:=i+1;
                    end else outdoor:=TRUE;
                end;
            end;
          i:=i+1;
        end;
      if outdoor=False then
        begin
          If (FMain.TimeRepeat>0) and (FMain.TypeAff<>CT_AFFCONFIRM) then
            begin
              FMain.ligneMenu:=0;
              //i:=0;
              FMain.T_StatusIO.Interval:=FMain.TimeRepeat;
              FMain.T_StatusIO.Enabled:=True;
              {
              While (i<FMain.ActualMenu.Count) and (OutDoor=False) do
                begin
                  Application.ProcessMessages;
                  textel:=trim(FMain.ActualMenu.Strings[i]);
                  if uppercase(copy(textel,1,5))='<_ISO14230_>' then
                    begin
                      i:=i+1;
                      while (i<FMain.ActualMenu.Count) and (outdoor=False) do
                        begin
                          textel:=trim(FMain.ActualMenu.Strings[i]);
                          if uppercase(textel)<>'</_ISO14230_>' then
                            begin
                              Actual_ISO14230.Add(textel);
                              i:=i+1;
                            end else outdoor:=TRUE;
                        end;
                    end;
                  i:=i+1;
                end;
                }
            end;
          InIOTimer:=False;
          exit;
        end;

//FMain.Actual_ISO14230.SaveToFile('c:\Actual_ISO14230_'+inttostr(i)+'.txt');
      // Записуємо дані з _ISO14230_
      // Посилаємо команду в
      outdoor:=False;
      j:=0;
      while (j<FMain.Actual_ISO14230.Count) and (outdoor=False) do
        begin
          textel:=FMain.Actual_ISO14230.Strings[j];
          if uppercase(copy(textel,1,4))='CDE=' then
            begin
              outdoor:=True;
            end;
          j:=j+1;
        end;
      // Виконуємо команду
      delete(textel,1,4);
      textel:=trim(textel);
      while (FMain.Execution= false) and (MainExit=False) do
Application.ProcessMessages;
      if MainExit then Exit;
      FMain.ISO14230.SendLength:=0;
```

```
      if FMain.TypeAff=CT_AFFCONFIRM then
        begin
          if MessageDlg(FMain.TitleMenu,mtConfirmation, [mbYes, mbNo],0)= mrYes
then
            begin
              if length(textel)>0 then
                begin
                  FMain.Cde2Send(textel);
                end;
              FMain.Attente_Start;
              FMain.Execution:=False;
              //while FMain.Execution= false do Application.ProcessMessages;
              FMain.ISO14230.SendResponse;
            end;
          InIOTimer:=False;
          exit;
        end else
      if FMain.TypeAff=CT_AFFONEPARAMETER then
        begin
          InIOTimer:=False;
          exit;
        end;
      if length(textel)>0 then
        begin
          FMain.Cde2Send(textel);
          FMain.Attente_Start;
          FMain.Execution:=False;
          //while FMain.Execution= false do Application.ProcessMessages;
          FMain.ISO14230.SendResponse;
        end;
      FMain.ligneMenu:=i;

      //FMain.T_StatusIO.Enabled:=True;
    end else
    begin
      If (FMain.TimeRepeat>0) and (FMain.TypeAff<>CT_AFFCONFIRM) then
        begin
          FMain.ligneMenu:=0;
          FMain.T_StatusIO.Interval:=FMain.TimeRepeat;
          FMain.T_StatusIO.Enabled:=True;
        end;
    end;
  InIOTimer:=False;
end;

//*****************************************************************************
procedure TFMain.TreeView1Click(Sender: TObject);
var textel,textel1 : string;
    i,j,k,l,m,n : integer;
    ligne : integer;
begin
  if FMain.TreeView1.Selected=nil then exit;
  //if FMain.Menu_Connect.Caption<>menuprog[CMP_Deconnecter] then exit;
  FMain.T_StatusIO.Enabled:=False; // stop the engine of _ISO14230_ command
  textel:=FMain.TreeView1.Selected.Text;
  MenuName:=textel;
  MenuI:=FMain.TreeView1.Selected.AbsoluteIndex;
  FMain.ClickTree:=True;
  FMain.Panel_Parametre.Visible:=False;
  FMain.Panel_grille.Visible:=False;
  FMain.Panel_Status.Visible:=False;
  FMain.Panel_Liste_Para.Visible:=False;
  FMain.Panel_Command.Visible:=False;
  FMain.Panel_Value.Visible:=False;
  FMain.Grille_Up:=0;
  if MenuI=0 then
    begin
      // Ми знаходимося в меню «Статус»
      FMain.Panel_Parametre.Visible:=False;
```

```
      FMain.Panel_grille.Visible:=False;
      FMain.Panel_Status.Visible:=True;
      FMain.Panel_Liste_Para.Visible:=False;
      FMain.Panel_Command.Visible:=False;
      FMain.ClickTree:=False;
    end else // end if MenuI=0
  if MENUI=1 then
    begin
      // команда управління _ISO14230_
      FMain.Panel_Parametre.Visible:=False;
      FMain.Panel_grille.Visible:=False;
      FMain.Panel_Status.Visible:=False;
      FMain.Panel_Liste_Para.Visible:=False;
      FMain.Panel_Command.Visible:=True;
      FMain.TypeAff:=CT_AFFNONE;
      FMain.E_Sid.Text:='';
      FMain.E_Data.Text:='';
      FMain.M_Hist.Clear;
      FMain.ClickTree:=False;
    end else
    begin
      // меню діагностики
      ActualMenu.Clear;
      FMain.Search_Item('MENU='+MenuName,Vehicule,ActualMenu);
      ClickList.Clear;
      FMain.Search_Item('CLICK',ActualMenu,ClickList); // On recupйre les
йvйnements click sur la grille
      // Діагностика помилок
      i:=0;
      FMain.ClickGlobal:='';
      FMain.ClickExit:='';
      FMain.B_Mode_Maitre.Visible:=False;
      While i<ClickList.Count do
        begin
          textel:=trim(ClickList.Strings[i]);
          if uppercase(copy(textel,1,5))='Вихід =' then
            begin
              delete(textel,1,5);
              FMain.ClickExit:=textel;
              FMain.B_Mode_Maitre.Visible:=True;
            end else

          if uppercase(copy(textel,1,6))='ACTIF=' then
            begin
              delete(textel,1,6);
              FMain.ClickGlobal:=textel;
            end;
          i:=i+1;
        end;
      // Шукаємо тип Panel
      i:=0;
      //ActualMenu.SaveToFile('C:\ActualMenu_'+MenuName+'.txt');
      while (i<ActualMenu.Count) do
        begin
          Application.ProcessMessages;
          if MainExit then Exit;
          textel:=trim(ActualMenu.Strings[i]);
          if uppercase(copy(textel,1,5))='Тип =' then
            begin
              delete(textel,1,5);
              if uppercase(copy(textel,1,4))='GRID' then
                begin

                  FMain.TypeAff:=CT_AFFGRID;
                  delete(textel,1,5);
                  textel:=trim(textel);
                  //FMain.StringGrid1.Clean;
                  //FMain.StringGrid3.Clean;
                  try
```

```
      j:=strtoint(textel);
    except
      j:=1;
    end;
    FMain.StringGrid1.ColCount:=j;
    //j:=FMain.StringGrid1.ColCount;
    //FMain.StringGrid3.ColCount:=j+2;
    //j:=FMain.StringGrid3.ColCount;
    FMain.Panel_grille.Visible:=True;
    FMain.StringGrid1.RowCount:=1;
    //FMain.StringGrid3.RowCount:=1;
    FMain.Ajust_StringList(1,FMain.ColorList);
    // Ми шукаємо рядки, що починаються з LG=
    while (i<ActualMenu.Count) do
      begin
        //Application.ProcessMessages;
        textel:=trim(ActualMenu.Strings[i]);
        if uppercase(copy(textel,1,3))='LG=' then
          begin
            // Ми записуємо отримані дані у буфер
            delete(textel,1,3); // видаляємо LG=
            delete(textel,1,4); // видаляємо ROW=
            j:=pos('/',textel);
            try
              ligne:=strtoint(copy(textel,1,j-1));
            except
              ligne:=1;
            end;
            if ligne>=FMain.StringGrid1.RowCount then
              begin
                k:=FMain.StringGrid1.RowCount;
                FMain.StringGrid1.RowCount:=ligne+1;
                for l := k to FMain.StringGrid1.RowCount - 1 do
                  for m := 0 to FMain.StringGrid1.ColCount-1 do
                    FMain.StringGrid1.Cells[m,l]:='';
                FMain.Ajust_StringList(Ligne+1,FMain.ColorList);
              end;
            delete(textel,1,j);
            while length(textel)>0 do
              begin
                j:=pos('/',textel);
                if j=0 then j:=pos(';',textel);
                if j>0 then
                  begin
                    textel1:=copy(textel,1,j-1);
                    delete(textel,1,j);
                  end else
                  begin
                    textel1:=textel;
                    textel:='';
                  end;
                if copy(textel1,1,3)='CL=' then
                  begin
                    delete(textel1,1,3);

//FMain.StringGrid3.Cells[FMain.StringGrid3.ColCount-2,ligne]:=textel1;
                    FMain.ColorList.Strings[ligne]:=textel1;
                  end else // fin if copy(textel1,1,3)='CL=' then
                if copy(textel1,1,2)='GR' then
                  begin
                    delete(textel1,1,2);
                    k:=pos('=',textel1);
                    try
                      l:=strtoint(copy(textel1,1,k-1));
                    except
                      l:=0;
                    end;
                    delete(textel1,1,k);
                    if l<FMain.StringGrid1.ColCount then
```

```
                                    begin
                                      if uppercase(copy(textel1,1,4))='TAB['
then
                                        begin
                                          delete(textel1,1,4);
                                          k:=pos(',',textel1);
                                          if k>0 then
                                            begin
                                              try
                                                m:=strtoint(copy(textel1,1,k-
1));
                                              except
                                                m:=-1;
                                              end;
                                              delete(textel1,1,k);
                                              k:=pos(']',textel1);
                                              try
                                                n:=strtoint(copy(textel1,1,k-
1));
                                              except
                                                n:=-1;
                                              end;
                                              delete(textel1,1,k);
                                              if (m>-1) and (m<10) then
                                                begin
                                                  if FMain.Table[m]<>nil then
                                                    begin
                                                      if (n>-1) and
(n<FMain.Table[m].Count)then
                                                        begin

textel1:=FMain.Table[m].Strings[n];
                                                        end else

textel1:=MenuProg[CMP_ERRORLIGNETABLE];
                                                    end else

textel1:=MenuProg[CMP_TableUnknow];
                                                end else

textel1:=MenuProg[CMP_ERRORNUMBERTABLE];
                                            end;

FMain.StringGrid1.Cells[l,ligne]:=textel1;
                                        end else
                                        begin

FMain.StringGrid1.Cells[l,ligne]:=textel1;
                                        end;
                                    end;
                                end;
                            end;
                        end; // end of if uppercase(copy(textel,1,3))='LG=' then

                      i:=i+1;
                    end; // end of  while (i<ActualMenu.Count) do
                  if FMain.StringGrid1.RowCount>1 then
FMain.StringGrid1.FixedRows:=1;
                end else
              if uppercase(copy(textel,1,7))='CONFIRM' then
                begin
                  FMain.Panel_Parametre.Visible:=False;
                  FMain.Panel_grille.Visible:=False;
                  FMain.Panel_Status.Visible:=True;
                  FMain.Panel_Liste_Para.Visible:=False;
                  FMain.Panel_Command.Visible:=False;
                  FMain.Panel_Value.Visible:=False;
                  FMain.TypeAff:=CT_AFFCONFIRM;
                end else
              if uppercase(copy(textel,1,12))='ONEPARAMETER' then
                begin
                  FMain.Panel_Parametre.Visible:=False;
```

```
      FMain.Panel_grille.Visible:=False;
      FMain.Panel_Status.Visible:=False;
      FMain.Panel_Liste_Para.Visible:=False;
      FMain.Panel_Command.Visible:=False;
      FMain.Panel_Value.Visible:=True;
      FMain.TypeAff:=CT_AFFONEPARAMETER;
      FMain.E_Para.Text:='';
      FMain.E_Para_Numero.Text:='';
      Fmain.M_Parametre.Clear;
      FMain.B_Para_Lecture.Caption:='';
      FMain.B_Para_Ecriture.Caption:='';
      FMain.CdeReadParam:='';
      FMain.CdeWriteParam:='';
      FMain.CdeValueLength:='';
      while (i<ActualMenu.Count) do
        begin
          Application.ProcessMessages;
          if MainExit then Exit;
          // шукаємо CDE_READ,CDE_WRITE,LABEL_READ,LABEL_WRITE
          textel:=trim(ActualMenu.Strings[i]);
          if uppercase(copy(textel,1,11))='LABEL_READ=' then
            begin
              delete(textel,1,11);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.B_Para_Lecture.Caption:=textel;
            end else
          if uppercase(copy(textel,1,12))='LABEL_WRITE=' then
            begin
              delete(textel,1,12);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.B_Para_Ecriture.Caption:=textel;
            end else
          if uppercase(copy(textel,1,9))='CDE_READ=' then
            begin
              delete(textel,1,9);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.CdeReadParam:=textel;
            end else
          if uppercase(copy(textel,1,10))='CDE_WRITE=' then
            begin
              delete(textel,1,10);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.CdeWriteParam:=textel;
            end else
          if uppercase(copy(textel,1,7))='LENGTH=' then
            begin
              delete(textel,1,7);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              CdeValueLength:=textel;
            end;
          i:=i+1;
        end;
      if length(FMain.CdeReadParam)=0
        then FMain.B_Para_Lecture.Visible:=False
        else FMain.B_Para_Lecture.Visible:=true;
      if length(FMain.CdeWriteParam)=0
        then FMain.B_Para_ecriture.Visible:=False
        else FMain.B_Para_ecriture.Visible:=true;
    end else
  if uppercase(copy(textel,1,8))='ONEVALUE' then
    begin
      FMain.Panel_Parametre.Visible:=True;
      FMain.Panel_grille.Visible:=False;
      FMain.Panel_Status.Visible:=False;
```

```
      FMain.Panel_Liste_Para.Visible:=False;
      FMain.Panel_Command.Visible:=False;
      FMain.Panel_Value.Visible:=False;
      FMain.TypeAff:=CT_AFFONEPARAMETER;
      FMain.E_Para.Text:='';
      FMain.E_Para_Numero.Text:='';
      Fmain.M_Parametre.Clear;
      FMain.B_Para_Lecture.Caption:='';
      FMain.B_Para_Ecriture.Caption:='';
      FMain.CdeReadParam:='';
      FMain.CdeWriteParam:='';
      FMain.CdeValueLength:='';
      while (i<ActualMenu.Count) do
        begin
          Application.ProcessMessages;
          if MainExit then Exit;
          // шукаємо CDE_READ,CDE_WRITE,LABEL_READ,LABEL_WRITE
          textel:=trim(ActualMenu.Strings[i]);
          if uppercase(copy(textel,1,11))='LABEL_READ=' then
            begin
              delete(textel,1,11);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.B_Para_Lecture.Caption:=textel;
            end else
          if uppercase(copy(textel,1,12))='LABEL_WRITE=' then
            begin
              delete(textel,1,12);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.B_Para_Ecriture.Caption:=textel;
            end else
          if uppercase(copy(textel,1,9))='CDE_READ=' then
            begin
              delete(textel,1,9);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.CdeReadParam:=textel;
            end else
          if uppercase(copy(textel,1,10))='CDE_WRITE=' then
            begin
              delete(textel,1,10);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              FMain.CdeWriteParam:=textel;
            end else
          if uppercase(copy(textel,1,7))='LENGTH=' then
            begin
              delete(textel,1,7);
              j:=pos(';',textel);
              if j>0 then textel:=copy(textel,1,j-1);
              CdeValueLength:=textel;
            end;
          i:=i+1;
        end;
      if length(FMain.CdeReadParam)=0
        then FMain.B_Valeur_Lecture.Visible:=False
        else FMain.B_Valeur_Lecture.Visible:=true;
      if length(FMain.CdeWriteParam)=0
        then FMain.B_Valeur_ecriture.Visible:=False
        else FMain.B_Valeur_ecriture.Visible:=true;
    end else
  if uppercase(copy(textel,1,10))='MULTIVALUE' then
    begin
      FMain.Panel_Parametre.Visible:=False;
      FMain.Panel_grille.Visible:=False;
      FMain.Panel_Status.Visible:=False;
      FMain.Panel_Liste_Para.Visible:=True;
      FMain.Panel_Command.Visible:=False;
```

```
                    FMain.TypeAff:=CT_AFFMultiVALUE;
                    while (i<ActualMenu.Count) do
                      begin
                        Application.ProcessMessages;
                        if MainExit then Exit;
                        // шукаємо CDE_READ,CDE_WRITE,LABEL_READ,LABEL_WRITE
                        textel:=trim(ActualMenu.Strings[i]);
                        if uppercase(copy(textel,1,11))='LABEL_READ=' then
                          begin
                            delete(textel,1,11);
                            j:=pos(';',textel);
                            if j>0 then textel:=copy(textel,1,j-1);
                            FMain.B_Lecture_Liste.Caption:=textel;
                          end else
                        if uppercase(copy(textel,1,12))='LABEL_WRITE=' then
                          begin
                            delete(textel,1,12);
                            j:=pos(';',textel);
                            if j>0 then textel:=copy(textel,1,j-1);
                            FMain.B_Sauve_Liste.Caption:=textel;
                          end else
                        if uppercase(copy(textel,1,9))='CDE_READ=' then
                          begin
                            delete(textel,1,9);
                            j:=pos(';',textel);
                            if j>0 then textel:=copy(textel,1,j-1);
                            FMain.CdeReadParam:=textel;
                          end else
                        if uppercase(copy(textel,1,10))='CDE_WRITE=' then
                          begin
                            delete(textel,1,10);
                            j:=pos(';',textel);
                            if j>0 then textel:=copy(textel,1,j-1);
                            FMain.CdeWriteParam:=textel;
                          end;
                        i:=i+1;
                      end;
                  end;
                i:=ActualMenu.Count;
              end;
            i:=i+1;
          end;

        // Шукаємо the title
        i:=0;
        while (i<ActualMenu.Count) do
          begin
            Application.ProcessMessages;
            if MainExit then Exit;
            textel:=trim(ActualMenu.Strings[i]);
            if uppercase(copy(textel,1,6))='Назва =' then
              begin
                delete(textel,1,6);
                textel:=trim(textel);
                FMain.TitleMenu:=textel;
                if FMain.TypeAff=CT_AFFGRID then
                  begin
                    For k:=0 to FMain.StringGrid1.ColCount-1 do
FMain.StringGrid1.ColWidths[k]:=16;
                    if length(textel)>0 then
                      if textel[length(textel)]<>'/' then textel:=textel+'/';
                    j:=0;
                    while length(textel)>0 do
                      begin
                        k:=pos('/',textel);
                        if k>0 then
                          begin
                            if j<FMain.StringGrid1.ColCount then
                              begin
```

```
                                //FMain.StringGrid3.Cells[j,0]:=copy(textel,1,k-
1);
                                FMain.StringGrid1.Cells[j,0]:=copy(textel,1,k-1);
                                if
FMain.StringGrid1.ColWidths[j]<FMain.StringGrid1.Canvas.TextWidth(copy(textel,1,
k-1))+32
                                  then
FMain.StringGrid1.ColWidths[j]:=FMain.StringGrid1.Canvas.TextWidth(copy(textel,1
,k-1))+32;
                              end;
                            j:=j+1;
                            delete(textel,1,k);
                          end else textel:='';
                        end;
                    end else
                  if FMain.TypeAff=CT_AFFCONFIRM then
                    begin
                    end else
                  if FMain.TypeAff=CT_AFFONEPARAMETER then
                    begin
                    end else
                  if FMain.TypeAff=CT_AFFMULTIVALUE then
                    begin
                    end else
                  if FMain.TypeAff=CT_AFFONEVALUE then
                    begin
                    end;
                  i:=ActualMenu.Count;
                end;
              i:=i+1;
            end;
          FMain.GroupBox2.Caption:=FMain.MenuName;
          // Шукаємо the Repeat time
          FMain.TimeRepeat:=0;
          i:=0;
          while (i<ActualMenu.Count) do
            begin
              textel:=trim(ActualMenu.Strings[i]);
              if uppercase(copy(textel,1,7))='REPEAT=' then
                begin
                  delete(textel,1,7);
                  textel:=trim(textel);
                  try
                    FMain.TimeRepeat:=StrToInt(textel);
                  except
                    FMain.TimeRepeat:=0;
                  end;
                  i:=ActualMenu.Count;
                end;
              i:=i+1;
              Application.ProcessMessages;
              if MainExit then Exit;
            end;

          // Використовуємо команди_ISO14230_
          FMain.ClickTree:=False;
          FMain.T_StatusIO.Interval:=200;
          FMain.ligneMenu:=0;
          FMain.T_StatusIO.Enabled:=True;
        end; // end if MenuI<>0
end;

//***************************************************************************
{
procedure TFMain.rafraichir_diag;
var i,j : integer;
    textel : string;
begin
  if FMain.TypeAff=CT_AFFGRID then
```

```
    begin
      if FMain.ClickTree then exit;
      FMain.nb_Row_visible:=FMain.StringGrid1.Height div
FMain.StringGrid1.DefaultRowHeight;
      if FMain.nb_Row_visible>FMain.StringGrid3.RowCount
         then FMain.nb_Row_visible:=FMain.StringGrid3.RowCount;
      if FMain.Grille_Up+FMain.nb_Row_visible>FMain.StringGrid3.RowCount
         then FMain.Grille_Up:=FMain.StringGrid3.RowCount-FMain.nb_Row_visible-
1;
      if FMain.StringGrid3.RowCount>FMain.nb_Row_visible
         then FMain.StringGrid1.RowCount:=FMain.nb_Row_visible
         else
      //FMain.StringGrid1.RowCount:=FMain.StringGrid3.RowCount;
      // Передача даних StringGrid3 до StringGrid1
      for i:=0 to FMain.StringGrid1.ColCount-1 do
        begin
          for j:=Grille_up+1 to Grille_up+FMain.StringGrid1.RowCount-1 do
            begin
              textel:=FMain.StringGrid3.Cells[i,j];
              textel:=copy(textel,1,length(textel)-1);
              if (textel<>'IMG_AMBER')
                 and (textel<>'IMG_RED')
                 and (textel<>'IMG_BLUE')
                 and (textel<>'IMG_GREEN')
                 then textel:=FMain.StringGrid3.Cells[i,j]
                 else textel:='';
              FMain.StringGrid1.Cells[i,j-Grille_Up]:=textel;
              if
FMain.StringGrid1.ColWidths[i]<FMain.StringGrid1.Canvas.TextWidth(textel)+8
                 then
FMain.StringGrid1.ColWidths[i]:=FMain.StringGrid1.Canvas.TextWidth(textel)+8;
            end;
        end;
      FMain.StringGrid1.Refresh;
    end;

end;}

//***************************************************************************
procedure TFMain.GlobalReset;
var i : integer;
    //noeud : TTreeNode;
begin
  FMain.nom_fichier:='';
  BeginExit:=True;
  //FMain.ISO14230.Tps_Maintient:=2000;
  FMain.Grille_Up:=0;
  FMain.nb_Row_visible:=FMain.StringGrid1.Height div
FMain.StringGrid1.DefaultRowHeight;
  //FMain.StringGrid3.RowCount:=0;
  FMain.ColorList.Clear;
  Fichier.Clear;
  Vehicule.Clear;
  MenuList.Clear;
  ActualMenu.Clear;
  Actual_ISO14230.Clear;
  FMain.ExtendedMenu.Clear;
  FMain.Panel_Liste_Para.Visible:=False;
  FMain.Panel_Parametre.Visible:=False;
  FMain.Panel_grille.Visible:=False;
  FMain.Panel_Command.Visible:=False;
  FMain.Panel_Status.Visible:=True;
  FMain.Panel_Value.Visible:=False;
  // nettoyage des panels
  FMain.M_Hist.Clear;
  FMain.M_Parametre.Clear;
  FMain.M_Valeur.Clear;
  FMain.E_Data.Clear;
  FMain.E_Fichier_Para.Clear;
```

```
  FMain.E_Para.Clear;
  FMain.E_Para_Numero.Clear;
  FMain.E_Sid.Clear;
  FMain.E_Valeur.Clear;
  //
  State:=0;
  FMain.Attente_Stop;
  FMain.ISO14230.StopDiagnosis;
  FMain.Menu_Connect.Caption:=MenuProg[CMP_Connecter];
  FMain.Image1.Picture.Bitmap:=nil;
  FMain.IListConnect.GetBitmap(0,FMain.Image1.Picture.Bitmap);
  for i:=0 to 9 do if FMain.Table[i]<>nil then FMain.Table[i].Clear;
  FMain.Treeview1.Items.Clear;
  //Noeud:=FMain.TreeView1.Items.Add(nil,MenuProg[CMP_Status]);
  //Noeud.ImageIndex:=5;
  //Noeud:=FMain.TreeView1.Items.Add(nil,MenuProg[CMP_Command]);
  //Noeud.ImageIndex:=1;
  FMain.TreeView1.Enabled:=False;
  FMain.diag_actif:=False;
  FMain.ClickActif:=False;
  FMain.ClickGlobal:='';
  FMain.ClickExit:='';
  FMain.B_Mode_Maitre.Visible:=false;
  FMain.NoReceive:=False;
  Application.CleanupInstance;

  try
  if (FBiPWM<>nil) and (FBiPWM.Actif) then FBiPwm.Close;
  if (FPWM<>nil) and (FPwm.Actif) then FPWM.Close;
  if (FBridge<>nil) and (FBridge.Actif) then FBridge.Close;
  if (FOnOff<>nil) and (FOnOff.Actif) then FOnOff.Close;
  if (FValue<>nil) and (FValue.Actif) then FValue.Close;
  if (FWiper<>nil) and (FWiper.Actif) then FWiper.Close;
  //F_Config.GroupBox2.Enabled:=True;
  except
  end;
end;

//*****************************************************************************
// Цей пошук має спеціальний пункт процедури у StringList ListSearch і повертає
всі параметри включно між двома тегами.
//*****************************************************************************
procedure TFMain.Search_Item(ItemSearch : string;ListSearch : TStringList;var
searchList: TStringList);
var textel : string;
    i,j : integer;
begin
  searchList.Clear;
  i:=0;
  while i<ListSearch.Count do
    begin
      textel:=ListSearch.Strings[i];
      textel:=trim(textel);
      if length(textel)>0 then
        if copy(textel,1,2)<>'//' then
          if
uppercase(copy(textel,1,length(ItemSearch)+1))=uppercase('<'+ItemSearch) then
            begin
              //i:=i+1;
              if i<ListSearch.Count then textel:=ListSearch.Strings[i] else
textel:='';
              while (i<ListSearch.Count) and
(copy(textel,1,length(ItemSearch)+2)<>uppercase('</'+ItemSearch))
                   and (uppercase(copy(textel,1,7))<>'</MENU>')do
                begin
                  textel:=ListSearch.Strings[i];
                  textel:=trim(textel);
                  j:=pos(';',textel);
                  if (copy(textel,1,2)<>'//') and (length(textel)>0) then
```

```
                    begin
                      if j>0
                        then SearchList.Add(copy(textel,1,j-1))
                        else SearchList.Add(textel);
                    end;
                  i:=i+1;
                end;
              i:=ListSearch.Count;
            end;
      i:=i+1;
    end;
end;

//***************************************************************************
procedure TFMain.Ajust_StringList(Count: integer; var Liste: TSTringList);
var i : integer;
begin
  if Count>Liste.Count then For i:=1 to Count-Liste.Count do Liste.Add('')
    else
  if Count<Liste.Count then For i:=1 to Liste.Count-Count do
Liste.Delete(Liste.Count-1);
end;

procedure TFMain.Attente_Start;
begin
  FMain.T_Attente.Enabled:=True;
end;

//***************************************************************************
procedure TFMain.Attente_Stop;
begin
  FMain.T_Attente.Enabled:=False;
end;

//***************************************************************************
Function TFMain.Cde2Send(CDE: string): integer;
var textel : string;
    erreur: integer;
begin
  while (FMain.Execution=false) and (MainExit=False) do
application.ProcessMessages;
  if MainExit then Exit;
  erreur:=0;
  FMain.ISO14230.SendLength:=0;
  textel:=CDE;
  try
    FMain.ISO14230.Sid:=hextodec(copy(textel,1,2));
  except
    FMain.ISO14230.Sid:=0;
    erreur:=1;
    Result:=Erreur;
    exit;
  end;
  delete(textel,1,3);
  while length(textel)>0 do
    begin

FMain.ISO14230.FData[FMain.ISO14230.SendLength]:=hextodec(copy(textel,1,2));
      FMain.ISO14230.SendLength:=FMain.ISO14230.SendLength+1;
      delete(textel,1,3);
    end;
  Result:=erreur;
end;

//***************************************************************************
procedure TFMain.Create_Menu_Langue;
var s : TSearchRec;
    textel : string;
begin
```

```
  Nb_Langue:=0;
  if FindFirst(Repertoire_courant+'Languages\*.dia',faAnyFile,s)=0 then
    begin
      repeat
        M[Nb_Langue]:=TMenuItem.Create(Self);
        m[Nb_Langue].OnClick:=Menu_langueAddClick;
        m[Nb_Langue].Visible:=True;
        m[Nb_Langue].Enabled:=True;
        m[Nb_Langue].Tag:=Nb_Langue;
        textel:=s.Name;
        textel:=copy(textel,1,length(textel)-4);
        m[Nb_Langue].Caption:=textel;
        FMain.Menu_Langues.Add(m[Nb_Langue]);
        Nb_Langue:=Nb_Langue+1;
      until (FindNext(s)<>0) or (Nb_Langue>255);
    end;
  FindClose(s);
end;

//***************************************************************************
procedure TFMain.Init_Message;
begin
  FMain.Menu_Connect.Caption:=MenuProg[CMP_Connecter];
  FMain.Menu_Setting.Caption:=MenuProg[CMP_Configuration];
  FMain.Menu_Quitter.Caption:=MenuProg[CMP_Quitter];
  FMain.Menu_Langues.Caption:=MenuProg[CMP_Language];
end;

//***************************************************************************
procedure TFMain.Menu_langueAddClick(Sender: TObject);
var textel : string;
    i:integer;
begin
  textel:=TMenuItem(Sender).Caption;
  ReadLanguage(Repertoire_courant+'Languages\'+textel+'.dia');

FMain.ISO14230.LoadMessageTXT(Repertoire_courant+'Languages\'+textel+'.ISO14230_
');
  FMain.Init_Message;
  for i:=0 to Nb_Langue-1 do M[i].Checked:=False;
  i:=TMenuItem(Sender).Tag;
  M[i].Checked:=True;
  F_Config.CB_Langue.Text:=textel;
end;

end.
```

unit_configuration.pas – файл конфігурації

```
unit Unit_Configuration;

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
  ExtCtrls, Buttons, Grids,Common;

type

  { TF_Config }

  TF_Config = class(TForm)
    B_Save: TBitBtn;
    B_Exit: TBitBtn;
    CB_Langue: TComboBox;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    Label2: TLabel;
    Panel1: TPanel;
    Grid_Boitier: TStringGrid;
    procedure B_QuitterClick(Sender: TObject);
    procedure B_SaveClick(Sender: TObject);
    procedure CB_LangueChange(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Grid_BoitierClick(Sender: TObject);
    procedure Grid_BoitierEditingDone(Sender: TObject);
    procedure Grid_BoitierSelectCell(Sender: TObject; aCol, aRow: Integer;
      var CanSelect: Boolean);
  private
    { private declarations }
    oldValue : string;
  public
    { public declarations }
    Langue : string;
    sortie : boolean;
    Procedure Init_Langue;

    function execute: boolean;
    Procedure Lecture_ini;
    Procedure Sauve_Ini;
  end;

var
  F_Config: TF_Config;

implementation

{$R *.dfm}
uses uMain;

{ TF_Config }
procedure TF_Config.B_QuitterClick(Sender: TObject);
begin
  F_Config.Close;
end;

procedure TF_Config.B_SaveClick(Sender: TObject);
begin
  sortie:=True;
end;

procedure TF_Config.CB_LangueChange(Sender: TObject);
var textel : string;
```

```
begin
  textel:=F_Config.CB_Langue.Text;
  if FileExists(Repertoire_courant+'Languages\'+textel+'.dia') then
    begin
      ReadLanguage(Repertoire_courant+'Languages\'+textel+'.dia');
      F_Config.Init_Langue;
    end;
end;

procedure TF_Config.FormCreate(Sender: TObject);
var s : TSearchRec;
    textel : string;
begin
  F_Config.CB_Langue.Clear;
  if FindFirst(Repertoire_courant+'Languages\*.dia',faAnyFile,s)=0 then
    begin
      repeat
        textel:=s.Name;
        textel:=copy(textel,1,length(textel)-4);
        F_Config.CB_Langue.Items.Add(textel);
      until (FindNext(s)<>0) or (Nb_Langue>255);
    end;
  FindClose(s);
  F_Config.CB_Langue.Text:='ukrainian';
  //if FileExists(Repertoire_Courant+'Diag_ISO14230.ini') then Lecture_Ini else
Sauve_Ini;
end;

procedure TF_Config.Grid_BoitierClick(Sender: TObject);
begin
  OldValue:=Grid_Boitier.Cells[Grid_Boitier.Col,Grid_Boitier.Row];
end;

procedure TF_Config.Grid_BoitierEditingDone(Sender: TObject);
var ACol,ARow : integer;
    i : integer;
    textel : string;
begin
  aCol:=Grid_Boitier.Col;
  aRow:=Grid_Boitier.Row;
  try
    case aRow of
      1 : begin // com-порт
        textel:=Grid_Boitier.Cells[aCol,aRow];
        delete(textel,1,3); // видаляємо the "COM"
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Comport.Port:=Grid_Boitier.Cells[aCol,aRow];
      end;
      2 : begin // Версія
        F_Config.Grid_Boitier.Cells[aCol,aRow]:=OldValue;
      end;
      3 : begin // Швидкість діагностики
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.DiagnosticSpeed:=i;
      end;
      4 : begin // Інформаційний режим
        F_Config.Grid_Boitier.Cells[aCol,aRow]:=OldValue;
      end;
      5 : begin // Номер спроби
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Tentative:=i;
      end;
      6 : begin // Довжина байту адреси
        textel:=Grid_Boitier.Cells[aCol,aRow];
```

```
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Time_Ad:=i;
      end;
      7 : begin // Час між байтами adress та Sync
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Time_Ad_Sync:=i;
      end;
      8 : begin // Час між байтами Sync та Key1
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Time_Sync_Key1:=i;
      end;
      9 : begin // Час між байтами Key1 та Key2
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Time_Key1_Key2:=i;
      end;
      10 : begin // Час між байтами Key2b та Adb
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Time_Key2b_Adb:=i;
      end;
      11 : begin // Час закінчення
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Time_TO:=i;
      end;
      12 : begin // Time_tester представлення
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Tps_Maintient:=i;
      end;
      13 : begin // Чекаємо 1 REQ
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Wait_1Req:=i;
      end;
      14 : begin // Чекаємо відповідь на запит
        textel:=Grid_Boitier.Cells[aCol,aRow];
        textel:=trim(textel);
        i:=strtoint(textel);
        FMain.ISO14230.Wait_Response_Request:=i;
      end;
    end;
  except
    F_Config.Grid_Boitier.Cells[aCol,aRow]:=OldValue;
  end;
end;

procedure TF_Config.Grid_BoitierSelectCell(Sender: TObject; aCol,
  aRow: Integer; var CanSelect: Boolean);
begin
  OldValue:=Grid_Boitier.Cells[aCol,aRow];
end;

procedure TF_Config.Init_Langue;
var i : integer;
begin
  F_Config.Caption:=MenuProg[CMP_Configuration];
  F_Config.GroupBox1.Caption:=MenuProg[CMP_DefautLang];
```

```
  F_Config.Label2.Caption:=MenuProg[CMP_DefautLang];
  F_Config.B_Save.Caption:=MenuProg[CMP_Save];
  F_Config.B_Exit.Caption:=MenuProg[CMP_Quitter];
  F_Config.GroupBox2.Caption:=MenuProg[CMP_SetCom];
  F_Config.Grid_Boitier.RowCount:=15;
  F_Config.Grid_Boitier.Cells[0,0]:=MenuProg[CMP_Parametre];
  F_Config.Grid_Boitier.Cells[1,0]:=MenuProg[CMP_Value];
  F_Config.Grid_Boitier.Cells[0,1]:='COM';
  F_Config.Grid_Boitier.Cells[0,2]:=menuProg[CMP_Versus];
  F_Config.Grid_Boitier.Cells[0,3]:=MenuProg[CMP_DiagSpeed];
  F_Config.Grid_Boitier.Cells[0,4]:=MenuProg[CMP_ModeInformation];
  F_Config.Grid_Boitier.Cells[0,5]:=MenuProg[CMP_Tentative];
  F_Config.Grid_Boitier.Cells[0,6]:=MenuProg[CMP_Time_Ad];
  F_Config.Grid_Boitier.Cells[0,7]:=MenuProg[CMP_Time_Ad_Sync];
  F_Config.Grid_Boitier.Cells[0,8]:=MenuProg[CMP_Time_Sync_Key1];
  F_Config.Grid_Boitier.Cells[0,9]:=MenuProg[CMP_Time_Key1_Key2];
  F_Config.Grid_Boitier.Cells[0,10]:=MenuProg[CMP_Time_Key2b_Adb];
  F_Config.Grid_Boitier.Cells[0,11]:=MenuProg[CMP_Time_TO];
  F_Config.Grid_Boitier.Cells[0,12]:=MenuProg[CMP_testerPresent];
  F_Config.Grid_Boitier.Cells[0,13]:=MenuProg[CMP_Wait_1REq];
  F_Config.Grid_Boitier.Cells[0,14]:=MenuProg[CMP_Wait_Res_Req];
  for I := 0 to 14 do
    if
F_Config.Grid_Boitier.ColWidths[0]<F_Config.Grid_Boitier.Canvas.TextWidth(F_Conf
ig.Grid_Boitier.Cells[0,i])+10
      then
F_Config.Grid_Boitier.ColWidths[0]:=F_Config.Grid_Boitier.Canvas.TextWidth(F_Con
fig.Grid_Boitier.Cells[0,i])+10;
end;

Function TF_Config.execute : boolean;
var i : integer;

begin
  sortie:=False;
  langue:=F_Config.CB_Langue.Text;
  if FileExists(Repertoire_Courant+'Diag_ISO14230.ini') then Lecture_Ini else
Sauve_Ini;
  F_Config.Init_Langue;
  F_Config.Grid_Boitier.Cells[1,1]:=FMain.ISO14230.Comport.Port;
  F_Config.Grid_Boitier.Cells[1,2]:=FMain.ISO14230.Version;
  F_Config.Grid_Boitier.Cells[1,3]:=inttostr(FMain.ISO14230.DiagnosticSpeed);

F_Config.Grid_Boitier.Cells[1,4]:=FMain.ISO14230.ModeInfoToStr(FMain.ISO14230.Mo
de_Information);
  F_Config.Grid_Boitier.Cells[1,5]:=inttostr(FMain.ISO14230.Tentative);
  F_Config.Grid_Boitier.Cells[1,6]:=inttostr(FMain.ISO14230.Time_Ad);
  F_Config.Grid_Boitier.Cells[1,7]:=inttostr(FMain.ISO14230.Time_Ad_Sync);
  F_Config.Grid_Boitier.Cells[1,8]:=inttostr(FMain.ISO14230.Time_Sync_Key1);
  F_Config.Grid_Boitier.Cells[1,9]:=inttostr(FMain.ISO14230.Time_Key1_Key2);
  F_Config.Grid_Boitier.Cells[1,10]:=inttostr(FMain.ISO14230.Time_Key2b_Adb);
  F_Config.Grid_Boitier.Cells[1,11]:=inttostr(FMain.ISO14230.Time_TO);
  F_Config.Grid_Boitier.Cells[1,12]:=inttostr(FMain.ISO14230.Tps_Maintient);
  F_Config.Grid_Boitier.Cells[1,13]:=inttostr(FMain.ISO14230.Wait_1Req);

F_Config.Grid_Boitier.Cells[1,14]:=inttostr(FMain.ISO14230.Wait_Response_Request
);
  F_Config.ShowModal;
  if sortie then
    begin

FMain.ISO14230.LoadMessageTXT(Repertoire_courant+'Languages\'+F_Config.CB_Langue
.Text+'.ISO14230_');
      FMain.Init_Message;
      FMain.ISO14230.SaveConfiguration(Repertoire_courant+'_ISO14230.ini');
      F_Config.Sauve_Ini;
    end else
    begin
      F_Config.CB_Langue.Text:=langue;
```

```
      ReadLanguage(Repertoire_courant+'Languages\'+langue+'.dia');
      FMain.ISO14230.LoadConfiguration(Repertoire_courant+'_ISO14230.ini');
    end;
  Result:=Sortie;
end;

procedure TF_Config.Lecture_ini;
var Fichier : TStringList;
    i : integer;
    textel : string;
begin
  try
    Fichier:=TStringList.Create;
    Fichier.LoadFromFile(Repertoire_Courant+'Diag_ISO14230.ini');
    for i:=0 to Fichier.Count-1 do
      begin
        textel:=fichier.Strings[i];
        if (length(textel)>0) and (copy(textel,1,2)<>'//')
          then begin
            if copy(textel,1,5)='LANG=' then
              begin
                delete(textel,1,5);
                textel:=trim(textel);
                F_Config.CB_Langue.Text:=textel;
              end else
            if copy(textel,1,6)='DEBUG=' then
              begin
                delete(textel,1,6);
                textel:=trim(textel);
                try
                  FMain.Debug:=StrtoInt(textel);
                except
                  FMain.Debug:=0;
                end;
              end;
          end;
      end;
  finally
    fichier.Free;
  end;
end;

procedure TF_Config.Sauve_Ini;
var Fichier : TStringList;
begin
  try
    Fichier:=TStringList.Create;
    Fichier.Add('// Diag_ISO14230_ конфігураційний файл');
    Fichier.Add('');
    Fichier.Add('LANG='+F_Config.CB_Langue.Text);
    Fichier.Add('DEBUG='+inttostr(FMain.Debug));
    Fichier.SaveToFile(Repertoire_Courant+'Diag_ISO14230.ini');
  finally
    fichier.Free;
  end;
end;
end.
```

unit_connection.pas – файл підключення

```
unit Unit_Connection;

interface

uses
  Classes, SysUtils, Forms, Controls, Graphics, Dialogs, StdCtrls,
  Grids, ExtCtrls, Buttons,CPort,Common;

type

  { TF_Connect }

  TF_Connect = class(TForm)
    B_Quitter: TBitBtn;
    B_Connecter: TBitBtn;
    Combo_Port: TComboBox;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    L_PortCOM: TLabel;
    Panel1: TPanel;
    StringGrid1: TStringGrid;
    procedure B_ConnecterClick(Sender: TObject);
    procedure B_QuitterClick(Sender: TObject);
  private
    { private declarations }
  public
    { public declarations }
    Valid : boolean;
    Vehicule_Name : string;
    Procedure Init;
    Function execute : boolean;
  end;

var
  F_Connect: TF_Connect;

implementation

{$R *.dfm}

{ TF_Connect }

procedure TF_Connect.B_ConnecterClick(Sender: TObject);
begin
  If F_Connect.StringGrid1.Row<1 then
     begin

Application.MessageBox(PCHAR(MenuProg[CMP_NoECUSelect]),PCHAR(MenuProg[CMP_Error
]),0);
       exit;
     end else
Vehicule_Name:=F_Connect.StringGrid1.Cells[0,F_Connect.StringGrid1.Row];
  F_Connect.Valid:=True;
  F_Connect.Close;
end;

procedure TF_Connect.B_QuitterClick(Sender: TObject);
begin
  F_Connect.Valid:=False;
  F_Connect.Close;
end;

procedure TF_Connect.Init;
var sFileData : TSearchREC;
```

```
begin
  F_Connect.StringGrid1.RowCount:=1;
  if ( FindFirst( Repertoire_courant + 'Vehicule\'+'*.veh', faAnyFile,
sFileData) = 0 ) then
    begin
      repeat
        if ( ( sFileData.Name <> '.' ) and ( sFileData.Name <> '..' ) ) then
           begin
             F_Connect.StringGrid1.RowCount:=F_Connect.StringGrid1.RowCount+1;
             F_Connect.StringGrid1.Cells[0,F_Connect.StringGrid1.RowCount-
1]:=sFileData.Name;
           end;
      until ( FindNext( sFileData ) <> 0 );
    end;
  FindClose( sFileData );
  if F_Connect.StringGrid1.RowCount>1 then
    begin
      F_Connect.StringGrid1.FixedRows:=1;
    end;
  EnumComports(F_Connect.Combo_Port.Items);
  F_Connect.Combo_Port.ItemIndex:=F_Connect.Combo_Port.Items.IndexOf(Port_Com);
  if (F_Connect.Combo_Port.ItemIndex<0)
     and (F_Connect.Combo_Port.Items.Count>0)
     then F_Connect.Combo_Port.ItemIndex:=0
     else F_Connect.Combo_Port.Text:=MenuProg[CMP_NoComPort];
  F_Connect.Caption:=MenuProg[CMP_Connecter];
  F_Connect.GroupBox1.Caption:=MenuProg[CMP_ECU];
  F_Connect.StringGrid1.Cells[0,0]:=MenuProg[CMP_ECU];
  F_Connect.B_Connecter.Caption:=MenuProg[CMP_Connecter];
  F_Connect.B_Quitter.Caption:=MenuProg[CMP_Quitter];
  F_Connect.GroupBox2.Caption:=MenuProg[CMP_Configuration];
  F_Connect.L_PortCOM.Caption:=MenuProg[CMP_Comport];
end;

function TF_Connect.execute: boolean;
begin
  F_Connect.Init;
  F_Connect.Valid:=False;
  F_Connect.ShowModal;
  Result:=F_Connect.Valid;
end;

end.
```