МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Кафедра авіоніки

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
_____ Павлова С.В.
«____» _____ 2021 р.

ДИПЛОМНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
“МАГІСТР”

Тема: Структурна організація та архітектура комплексів бортового обладнання літака

Виконавець:_____ Іщук Дмитро Леонідович
Керівник:_____ Слободян Олександр Петрович
Нормоконтролер:_____ Левківський Василь Васильович
Охорона праці: _____ Козлітін Олексій Олександрович
Охорона навколишнього середовища: _____ Дмитруха Тетяна Іллівна

Київ 2021

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет аеронавігації, електроніки та телекомунікацій
Кафедра авіоніки
Напрям підготовки 173 «Авіоніка»

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ (Павлова С.В.)
«____» _____ 2021 р.

ЗАВДАННЯ
на виконання дипломної роботи

Іщука Дмитра Леонідовича
(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. Тема дипломної роботи «Структурна організація та архітектура комплексів бортового обладнання літака» затверджена наказом ректора від «_____»_____20__р. №_____
2. Термін виконання роботи : з _____ по _____
3. Вихідні дані до роботи (проекту): Архітектура бортового обладнання на базі інтегральної модульної авіоніки. Методи аналізу на основі моделі з кількома обмеженнями для інтегрованого процесу динамічної реконфігурації модульної авіоніки.
4. Зміст пояснювальної записки (перелік питань, що їх належить розробити):
Розділ 1. Інтегрована модульна авіоніка — минуле, сьогодення та майбутнє.
Розділ 2. Еволюція мереж авіоніки від ARINC 429 до AFDX.
Розділ 3. Роль ARINC 653 в інтегрованій модульній авіоніці.
Розділ 4. Методи аналізу на основі моделі з кількома обмеженнями для інтегрованого процесу динамічної реконфігурації модульної авіоніки.
Розділ 5. Охорона праці
Розділ 6. Охорона навколишнього середовища
5. Перелік обов’язкового графічного матеріалу (з точним визначенням обов’язкових рисунків, діаграм, таблиць тощо):

6. Календарний план-графік

№ п/п	Завдання	Термін виконання	Відмітка про виконання
1.	Аналіз літератури	1.10.2020 - 15.9.2021	
2.	Написання Розділу 1	15.9.2021 - 1.10.2021	
3.	Написання Розділу 2	1.10.2021 - 24.10.2021	
4.	Написання Розділу 3	24.10.2021 - 6.11.2021	
5.	Написання Розділу 4	6.11.2021 - 14.11.2021	
6.	Написання Розділу 5 та Розділу 6	14.11.2021- 22.11.2021	
7.	Оформити пояснювальну записку та графічні матеріали	22.11.2021 - 1.12.2021	

7. Дата видачі завдання: «__» _____ 2021 р.

Керівник дипломної роботи _____ Слободян О.П.

Завдання прийняв до виконання _____ Іщук Д.Л.

РЕФЕРАТ

Пояснювальна записка до випускової роботи «Структурна організація та архітектура комплексів бортового обладнання літака»:

94 сторінjr, __ рисунок, 21 використаних джерел.

ЛІТАК, КОМПЛЕКС БОРТОВОГО ОБЛАДНАННЯ, ІНТЕГРОВАНА МОДУЛЬНА АВІОНІКА, БОРТОВА МЕРЕЖА, СИСТЕМА ВІДОБРАЖЕННЯ ІНФОРМАЦІЇ, ПРОЕКТУВАННЯ КБО, АЛГОРИТМИ ПЕРЕВІРКИ

Об’єкт дослідження – процес формування структури комплексів бортового обладнання, принципи побудови та їх архітектура, предмет дослідження – синтез структури комплексу бортового обладнання.

Мета дипломного проекту – проаналізувати основні етапи розвитку та введення інтегральної модульної авіоніки а також структуру комплексів бортового обладнання літака.

Метод дослідження – теоретичний, аналітичний, синтез.

Установлено, що розроблена архітектура комплексу та його структурна організація відповідає вимогам прийнятої концепції та сучасним технологіям і може застосовуватися на перспективних літаках нового покоління.

Матеріали дипломного проекту рекомендуються використовувати при проведенні наукових досліджень, навчальному процесі та в практичній діяльності фахівців авіаційних конструкторських бюро.

Прогнозовані припущення щодо розвитку об’єкта дослідження – розроблення та удосконалення структурної складової систем з урахуванням вимог нормативних документів та стандартів, наприклад, стандарту AFDX.

ЗМІСТ

ВСТУП

Структурні пристрої, що знаходяться на борту ЛА, об'єднуються в системи для вирішення окремих завдань. Окремі системи можуть бути об'єднані у великі структури - комплекси. Комплекс бортового обладнання - це набір функціонально - пов'язаних систем, датчиків, лічильників.

Побудова бортового комплексу, що базується на відкритій мережевій архітектурі і єдиної обчислювальної платформі називають інтегрованою модульною авіонікою. Поняття "інтегрована" використовується як об'єднання загальних ресурсів - джерел живлення, процесора, пам'яті, комунікаційних шин, джерел введення-виведення для вирішення єдиного завдання - управління. Функції систем комплексу в цьому випадку виконують програмні додатки, що розділяють загальні обчислювальні та інформаційні ресурси. Поняття функції є ключовим поняттям IMA. Під функцією повітряного судна розуміються функціональні можливості, які можуть бути забезпечені апаратними і програмними засобами, наявними на ПС, наприклад: літаководіння, зв'язок, індикація і т. д.

Перехід на IMA дозволив перейти від ідеї "система-одна функція"до мультифункціональної структури - "багато функцій в одному обчислювальному ядрі". Якщо поділ апаратних і програмних платформ, тобто незалежно від ядра їх розрахунку, вирішення такої проблеми технічно легше. Практично інтеграція функцій, раніше прийнятих в якості інтеграції систем, полягає в створенні функцій і сигналів в новій ланці КБО, а також в комунікаторі функцій на рівні ПЗ.

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

ІМА - інтегральна модульна авіоніка (*з англ.* Integrated Modular Avionic)

КБО - комплекс бортового обладнання

БОК - бортовий обчислювальний комплекс

БЦОС - бортова центральна обчислювальна система

ЦАП - цифро-аналговий перетворювач

ТМС - твердотільні мікросхеми

ТТЛ - транзисторно-транзистерна логіка

ЦМП - циліндричні магнітні плівки

КМОН - комплементарна структура метал-оксид-напівпровідник

КНС - «кремній на сапфірі»

ВІС - великі інтегральні схеми

НВІС - надвеликі інтегральні схеми

ПЗМ - пристої зв’язку по магістралі

ПКА - перетворювальна комутаційна апаратура

СНК - «система на кристалі»

ARINC - Aeronautical Radio Inc

КФМ - конструктивно-функціональні модулі

СОІ - системи відображення інформації

DDA - Dumb Display Architecture («Нерозумний» тип побудови системи електронної індикації)

EIS - Electronic Indication System

LRU - Line Replaceable Unit

SSDA - Semi-Smart Display Architecture

IDA - Integrated Display Architecture

EFIS - Electronic Flight Instrument System

ECAM - Electronic Centralized Aircraft Monitor

FMS - Flight Management System

AIMS - Aircraft Information Management System

AFDX - Avionics Full-Duplex Switched Ethernet

Розділ 1. Інтегрована модульна авіоніка — минуле, сьогодення та майбутнє

Вступ

Як у військових, так і в комерційних авіаційних системах, як правило, існує часткова інтегрована модульна авіоніка (IMA) зі змішаною конфігурації федеративних і IMA підсистем для подовження життєвого циклу цих систем. В результаті виникає деяка плутанина, тому що фактично вставка IMA може виконуватися на рівні підсистеми, сегмента, платформи та до рівня систем. IMA також може бути поєднанням систем, що критично важливі для польоту, і систем, які важливі для виконання завдань. Крім того, є багато збігів з архітектурою відкритих систем (OSA) та IMA. У подальшому, нові проблеми IMA повинні бути вирішені для введення IMA в аерокосмічні системи. Ці проблеми включають: 1) використання мікросхем з багатьма ядрами (деякі з яких неоднорідні) для обмежень доступності, розміру, ваги та потужності (SWAP); 2) впровадження нових та/або вдосконалених стандартів програмного забезпечення для покращення повторного використання та використання новітніх програмних технологій; 3) уніфікація змішаних мережевих технологій для підтримки пропускної здатності та гнучкості підключення; 4) введення більшої автономності для покращення безпеки та доступності; і 5) покращення безпеки в середовищі систем-систем.

Існують дві основні загальноприйняті рекомендації щодо архітектури авіоніки: директива OSA від Міністерства оборони (DoD) і рекомендація стандартів IMA від DO-297. Системи військової авіоніки наступного покоління часто описують як передову архітектуру відкритих систем (AOSA), а цивільний транспорт наступного покоління — як передову IMA (A-IMA). Підхід Міністерства оборони, який надає перевагу для впровадження відкритих систем, раніше називався модульним підходом відкритих систем (MOSA), тепер називається OSA. Зі зміною підходів до архітектур наголос робиться на розробці архітектури, яка підтримує доступні зміни,

дозволяє еволюційне придбання та спіральний розвиток, а також активується завдяки інтегрованій дорожній карті, яка підтримує повторне використання заздалегідь запланованого дизайну. Використання підходу до проектування, заснованого на широко підтримуваних відкритих стандартах, підвищує ймовірність того, що майбутні зміни в системі будуть інтегровані економічно ефективним способом. OSA — це як бізнес, так і технічна стратегія для розробки нової системи або модернізації існуючої. Відкриті системи використовують модульну конструкцію, використовують широко підтримувані та засновані на консенсусі стандарти для ключових інтерфейсів і отримують перевірка функціональної сумісності та верифікаційні тести для забезпечення відкритості їхніх ключових інтерфейсів.

IMA є прикладом OSA, де акцент робиться на забезпеченні інтегрованої архітектури з програмним забезпеченням, яке переноситься через топологію мережі загальних апаратних модулів, які можуть підтримувати численні програми різного рівня критичності. Загалом, архітектури OSA і IMA мають такі характеристики:

1. Загальні апаратні та програмні елементи, засновані на комерційному готовому ринку (COTS) або на певному наборі стандартів домену,
2. Фізична інтеграція мереж, модулів і пристроїв введення-виведення (I/O) з розділенням на основі пропускної здатності, продуктивності та безпеки,
3. Багатошарова архітектура програмного забезпечення, що використовує стандартні рівні інтерфейсу програмування, щоб приховати обладнання та програми один від одного та дозволити повторне використання та переносимість коду,
4. Переналаштування додатків на модулях. Це може бути статична реконфігурація (літак не використовується) або динамічна реконфігурація (у польоті),
5. Операційна система та середовище проміжного програмного забезпечення для керування додатками,

6. Забезпечене зростання обробки на основі умовної дорожньої карти майбутньої функціональності за підсистем,
7. Оновлення ключових інтерфейсів за допомогою мостів до застарілих інтерфейсів,
8. Механізми захисту, які дозволяють використовувати такі ресурси, як пам'ять, кількома додатками різного рівня критичності на одному процесорі, а також дозволяти вставляти/змінювати програми без впливу на решту системи на одній процесорній карті,
9. Детермінований розклад, щоб дотримуватись кінцевих термінів усіх додатків, для кожної життєздатної конфігурації та коли система оновлюється через наскрізний розподілений ланцюг обробки.

IMA дає змогу кільком непов’язаним додаткам з різними критичними значеннями використовувати одну й ту саму обчислювальну платформу без перешкод. Завданням проектування IMA є відображення системних і підсистемних обмежень на рівні платформи та підсистеми в режимі реального часу та безпеки на доступну цільову архітектуру процесорів, мереж і програмних компонентів. Структура системи тепер включає понад 100 процесорів, з’єднаних між собою за допомогою уніфікованих і гібридних каналів зв’язку. Поточні зміни в рішеннях IMA обумовлюються сегментами ринку та міркуваннями бізнес-кейсів. На малюнку 1.1 показано умовне представлення високого рівня OSA/IMA для комерційної та військової авіоніки.

OSA в першу чергу керується необхідністю підтримки нових, майбутніх можливостей місії, які можуть або не можуть бути повністю охарактеризовані під час розробки транспорту. OSA забезпечує гнучкість додавати або змінювати можливості, коли це необхідно. На відміну від цього, архітектура IMA обумовлена збереженням ваги літака, простотою встановлення системи за рахунок скорочення проводки та фізичних одиниць, а також меншою вартістю змін під час початкової розробки та для майбутніх оновлень системи. Цікаво відзначити, що і OSA, і IMA забезпечують подібні переваги, хоча й обумовлені окремими пріоритетами. Однак

IMA приділяє більше уваги безпеці та сертифікації, ніж OSA, орієнтовану на ціль. Безпрецедентний рівень інтеграції в середовищі IMA створив потребу в нових стандартах цивільної безпеки та сертифікації, таких як DO-297, DO-178 і ARP-4754.

Точніше, варіанти IMA за сегментами ринку включають:

- Цивільний комерційний транспорт FW - загальноприйнята IMA,
- Civil Business RW/FW – гібридні OSA/IMA та загальноприйнята IMA,
- Цивільний БПЛА - малий форм-фактор OSA,
- Військово-транспортний багатоцільовий FW - гібридниі OSA/IMA,
- Військовий багатоцільовий RW - гібридні OSA/IMA,
- Військовий багатоцільовий винищувач - гібридні OSA/IMA,
- Військовий багатоцільовий БПЛА - OSA,
- Військовий одномісний БПЛА – OSA малого форм-фактор.

Обсяг комбінації OSA/IMA для показаних категорій авіоніки також залежить від того, коли платформу було запущено, ключову продуктивність обробки та мережі, необхідну для архітектури, як часто платформу оновлювали технології та які можливості повторного використання лінійки продуктів були використані. Крім того, для авіоніки, що швидко розвивається, як-от менші безпілотні літаки (БПЛА), розробляються індивідуальні високоінтегровані рішення OSA, щоб відповідати складному SWAP, призначеному для їх можливостей місії. Також для кожної з показаних категорій є флагманські програми для найкращих інтегрованих рішень OSA/IMA, які часто обумовлені пріоритетами їхнього ринкового сегмента.

1.1. Історія IMA

Стрімке зростання ринку комерційних вбудованих комп’ютерів у 1990-х роках і проблеми застаріння в розгорнутих системах авіоніки сприяли ініціативі MOSA. Ця ініціатива спочатку була зосереджена на тому, як розробити відкриті комп’ютерні системи для критичних додатків у військових літаках з використанням надійних комп’ютерних компонентів COTS у стандартних форм-факторах 6U та 3U. Основою MOSA є гнучка система, заснована на добре налагоджених апаратних і програмних інтерфейсах, що дозволяє використовувати найсучасніші технології. Додаткові переваги MOSA включають зниження витрат на розробку, закупівлю та підтримку. Це призвело до стандартизованих апаратних і програмних інтерфейсів, включаючи: шину Versa Module Europa (VMEbus) і інтерфейс компактного периферійного компонента (cPCI), єврокартки для плат з'єднання, стійки для повітряного транспорту (ATR) для шасі стандартного форм-фактора та інтерфейс портативної операційної системи (POSIX) як стандарт програмного забезпечення для додатків та інтерфейсу користувача. Ключовими стандартами з'єднання MOSA були ARINC 429, MIL-STD-1553B, Fibre Channel і Ethernet. Центральним аспектом проектування моделі спільної оперативної групи відкритої системи (OSJTF) була багатошарова архітектура апаратного та програмного забезпечення. Рівень програмного забезпечення включав додатки з системним керуванням і контролем, рівень операційної системи та рівень драйвера апаратного забезпечення. Щоб забезпечити стабільність, рішення для авіоніки на основі MOSA використовують основні інвестиції в продукти сторонніх розробників, загальні лінії компонентів, вибране оновлення технологій, багатошарову архітектуру програмного забезпечення та вибіркове придбання компонентів протягом усього життєвого циклу.

Системна інтеграція системи управління інформацією літака 777 (AIMS) від Honeywell для Boeing була однією з перших основних ініціатив IMA в цивільній авіації на початку 1990-х років. Система AIMS складається з подвійних шаф у відсіку для електроніки 777, кожна з яких містила 4 модулі ядра процесора (CPM) і 4 модулі вводу/виводу. Програмні додатки були розроблені для роботи в розподіленій

архітектурі з використанням операційної системи в режимі реального часу разом із плануванням розділів, керованим таблицею, і зв'язком із платою автозбереження (SAFEbus).

Boeing 777 був першим цивільним транспортним літаком, який використовував шину багаторазового доступу (ARINC 629) на цивільному транспортному літаку для підтримки мережі IMA. ARINC 629 дуже детермінований в роботі, і кожен термінал має високу цілісність щодо доступу асоційованого пристрою до шини, правильне використання смуги пропускання та вибір переданих і прийнятих даних. Хоча це було критичне покращення зі швидкістю 2 Мбіт/с порівняно з мережами ARINC 429 «точка-точка», які раніше працювали зі швидкістю 12,5 та 1000 кбіт/с, ARINC 629 вимагали користувацьких галузевих ASIC та роз'єднувачів.

В 2001 році були запропоновані наступні проблеми для розвитку IMA.

► Хоча в IMA був досягнутий прогрес із такими зусиллями, як AIMS, впровадження було оптимізовано для конкретних доменів і відповідало лише частині запланованих галузевих цілей IMA,

► Дотримання встановленого набору стандартних апаратних модулів може створити перешкоди для отримання переваг від швидкого розвитку електронних технологій,

► Існує можливість для нової програми для літаків, щоб забезпечити спільність між платформами різних доменів додатків шляхом співпраці між постачальниками платформ,

► Ключовими інтерфейсами, які потребують керування, є міжмодульні шини даних (об'єднані панелі), мережа та інтерфейс прикладного програмного забезпечення (APEX),

► Більше використання інтелектуальних периферійних пристроїв і віддалених концентраторів даних значно зменшить потребу в аналоговому та дискретному вводі/виводі в шафах IMA та змінних блоках комп'ютерної лінії (LRU).

1.2. IMA сьогодні

Кутовим каменем нинішньої IMA було сприяння документу «Integrated Modular Avionics (IMA) Development and Guidance and Certification Considerations» (RTCA DO-297) у листопаді 2005 р., і він зосередив увагу на нових галузевих стандартах. На той час концепція CPM була добре сформована. Стандарт «Aircraft Data Network, Part 7, Avionics Full-Duplex Switched Ethernet Network» (ARINC 664 Part 7) приніс адаптацію мережевої технології на основі COTS з підтримкою QoS. APplication EXecutive (APEX) як ARINC 653, забезпечив основу для розробки додатків, розділених у часі та просторі, у середовищі гіпервізора з досить пропрацьованою операційною системою реального часу. ARINC 653 APEX вперше був опублікований у 1996 році та оновлений у 2003 році. У військовій авіоніці були розробки стандартів, пов’язані з високоефективним середовищем авіоніки Fibre Channel за розкладом – анонімний обмін повідомленнями абонента (FC-AE-ASM) для архітектури IMA з високою пропускною здатністю, опублікований у 2007 році.

Уніфікований набір CPM зазвичай використовується в комерційних транспортних підсистемах у повній конфігурації IMA з використанням стандарту «Design Guidance For Integrated Modular Avionics» (ARINC 651) для визначення загальної архітектури апаратного забезпечення, змішаної з обладнанням на основі спеціальної стійки або шафи. Військові IMA зазвичай базуються на ряді COTS і надійних стандартних (ROTS) OpenVPX або VMEbus CPM на основі ATR або обладнання на базі шафи. У зв’язку з зростаючими потребами військових і необхідністю розширеної обробки нових функцій у вибраних підсистемах, кілька поколінь CPM розгортаються в даній системній архітектурі платформи. Також у військовій авіоніці деякі підсистеми базуються на IMA, деякі використовують спільні кластери обробки IMA, а деякі зберігають об’єднані рішення. Однією з областей небажання використовувати IMA як цивільних, так і військових систем є системи управління польотом. Занепокоєння викликає додаткова потреба в більш жорсткій синхронізації

каналів і складність управління резервуванням для більш розподілених конфігурацій.

Програми авіоніки складаються з набору функцій або завдань, які виконуються на одному або кількох розподілених CPM. ARINC 653 і його стандарти програмного забезпечення інтерфейсу APEX використовуються як на комерційних, так і на військових платформах, щоб забезпечити просторове та тимчасове розділення функцій авіоніки архітектури платформи. Стандарт ARINC 653 був представлений у 2005 році. APEX використовується для встановлення тимчасового розділення шляхом введення фіксованих періодично запланованих розділів з обмеженими часовими вікнами, в яких програмам дозволяється виконувати. Основний кадр представляє основну періодичну послідовність виконання розділу. Основний кадр розбивається на цілу кількість другорядних кадрів однакового періоду та тривалості. Просторове розбиття реалізується шляхом визначення віртуального адресного простору для кожного розділу статично під час запуску. Таке логічне розділення дозволяє додаткам змішаного рівня критичності (як визначено в ARP-4754) працювати на одній CPM. У військових системах для виконання важливих функцій, не пов’язаних з безпекою, також використовуються інші операційні системи реального часу, які базуються на міркуваннях переносимості, а також драйвер OSA для використання Linux як ключового інтерфейсу.

Уніфікація мережі залежить від архітектури IMA, але стандарт ARINC 664 Частина 7 використовується в комерційному просторі IMA, щоб забезпечити якість обслуговування між розподіленими CPM. Важливою перевагою ARINC 664 Part 7 є те, що він може утворювати безперервну мережу, що охоплює як рівень літака (зв'язок між LRU, центрами IMA та іншими інтелектуальними периферійними пристроями), так і рівень системного домену (зв'язок між модулями, пов'язаними з певною групою систем).). Стандарт ARINC 664 Part 7 був представлений у 2005 році. Віддалені завдання обмінюються інформацією через порти APEX на каналі APEX. Два розподілених завдання зв'язку належать до вихідного розділу та розділу призначення відповідно. З боку передачі швидкість передачі даних пов’язана з

одним віртуальним каналом багатоадресного односпрямованого зв’язку, який називається «Віртуальне посилання». Налаштований комутатор ARINC 664 Part 7 виконує пересилання пакетів, а також забезпечує політику трафіку на своїх вхідних портах і працює зі швидкістю 10 або 100 Мбіт/с. У військових архітектурах IMA існує значне використання стандартного з’єднання Ethernet в системах, де адекватне керування пропускною здатністю достатньо для задоволення вимог датчиків і каналів передачі даних без потреби в спеціалізованих комутаторах ARINC 664 Part 7. Крім того, у військових архітектурах IMA можливості безпеки польотів ізольовані від можливостей місії, щоб забезпечити найкраще як OSA, так і IMA. Мережі IMA є надзвичайно гнучкими, інтелектуальними магістральними мережами, які об’єднують зв’язок між несхожими форматами та структурами даних. Типова структура мережі IMA показана на малюнку 1.2.

Ключовим елементом IMA є відокремлення прямого введення-виводу від кожної функції і перетворення її в мережевий елемент даних. У сучасних реалізаціях деякий тип моста вводу-виводу або віддаленого терміналу є другим найважливішим апаратним елементом для CPM, а третій — мережевим комутачем. Крім того, діаграма показує похідну від чистого IMA, де такі функції, як керування польотом, інформаційно-розважальна програма та комунікації, можуть використовувати лінійку продуктів OSA, а не звичайні стандартизовані блоки IMA. Контроль польотів можливий у чистому режимі IMA, але його зазвичай уникають через проблеми загального режиму (якщо архітектура IMA не включає різний дизайн, що не є стандартом на цивільних ринках). Інформаційно-розважальні засоби та комунікації – це той випадок, коли IP-мережі можуть вимагати більш широкої підтримки IP і більше стандартів щодо інфраструктури комутаторів. Однак навіть у цих випадках потрібен деякий міст до мережі ARINC 664 Part 7.

Товариство інженерії автоматизації (SAE) AS6802 (Ethernet із запуском часу) — це мережева технологія, що досі розвивається, але вже використовується в архітектурах IMA. Цей стандарт забезпечує покращення рівня 2 QoS для загального стандарту Ethernet, що робить Ethernet придатним для критично важливих для

безпеки авіаційних мереж. Трафік ARINC 664 Part 7 обмежений затримкою і повинен витримувати значні похибки, що вимагає асинхронного архітектурного інтерфейсу між системами. На відміну від цього, Ethernet із запуском часу забезпечує критичний за часом потік даних для синхронних архітектур. Це забезпечує детерміновану мережу Ethernet, яка може бути більш придатною для виконання критичних у часі функцій літака, таких як керування польотом. Синхронна природа Ethernet із запуском часу також підходить для архітектур розподіленої обробки, оскільки забезпечує жорсткий контроль похибок даних і суворий детермінізм між елементами обробки. Завдання обробки синхронізовані з передачею даних мережі з періодичним множинним доступом з розділенням часу (TDMA). Подібно до того, як ARINC 653 забезпечує розділення за допомогою розподілу часу на блоці обробки, протокол Ethernet з моменту запуском забезпечує подібне розподілення між критичними та некритичними зв'язками через заздалегідь визначені часові інтервали в мережі. Це тимчасове розділення дозволяє кільком класам трафіку співіснувати в одній мережі без перешкод. Багатоцільовий екіпажний транспортний засіб NASA, який розробляється, Orion, використовує технологію Ethernet із запуском часу як основу даних, що дозволяє трьом класам трафіку співіснувати в одній мережі. У військових системах авіоніки FC-AE-ASM використовувався для реалізації планованих мереж як з періодичними, так і з асинхронними операціями як попередник повної можливості SAE AS6802, але з використанням Fibre Channel, а не Ethernet. Там, де плановані мережі не потрібні, існує значне використання стандартних комутаційних мереж GigE у військових системах місії OSA разом із додаванням шлюзів безпеки та маршрутизаторів.

Найбільш популярними комерційними платформами IMA є Airbus A380 і Boeing 787. На Airbus A380 стандартні блоки авіоніки ARINC 600 використовувалися для розміщення основного процесора та модулів вводу-виводу (CPIOM), підключених через Ethernet-комутатор на основі ARINC 664 Part 7 мережі. Airbus використовував відкрите середовище IMA, де вони виступали системним інтегратором для всіх додатків у кабіні та допоміжних додатків. Фактична відповідальність за розробку

систем і функцій була на постачальниках систем. Відкрита мережа IMA A380 була поділена на такі домени: керування польотом, кабіна, енергія, паливо/шасі та кабіна з відповідною відмовостійкістю в топології. Конфігурація обробки CIOPM включає виділення частин пам'яті, портів сигналів і часу циклу процесора для різних розділів. Конфігурація мережі для комутаторів ARINC 664 Part 7 включає статичні адресні таблиці, розмір кадру даних і пропускну здатність для віртуальних каналів. Розділи програмного забезпечення та файли таблиці конфігурації складають остаточне програмне завантаження.

Boeing 787 також використовує відкриту архітектуру IMA, де для модулів обробки ядра та модулів вводу-виводу використовується кабінетний підхід. У корпусі ядра процесора розміщено понад 80 різних програм. Версія ARINC 664 Частина 7 була використана для комутованої мережі зв'язку Ethernet, яка включала резервну топологію для систем літака. На основі відкритої архітектури IMA Boeing 787 Gulfstream G500/G600 стане «першим додатком для бізнес-джетів з інтегрованою мережею передачі даних, що включає кілька систем літаків», за словами Алана Чаславки, президента авіоніки та цифрових систем GE Aviation. Gulfstream підкреслює, що ця архітектура значно зменшує кількість кабелів і деталей, забезпечуючи при цьому більшу резервність і легше обслуговування. Хоча складність збільшується з поширенням системної інтеграції в інфраструктурі IMA, цивільні ринки були готові продати цю складність на кращий, легший і зручніший продукт.

Відмінності у військовій реалізації OSA/IMA показані в таблиці 1.1.

Інтеграція в рамках розробки IMA вимагає додаткової суворості, особливо коли інтеграція відбувається поза межами організації та компанії. Детальність документа керування інтерфейсом (ICD) стає набагато критичнішою. Традиційно ICD використовувався для документування проводки та швидкості передачі даних між федеративними юнітами. IMA має містити більше інформації, яка може бути використана як гарантія проектування іншими окремими постачальниками функцій. Наприклад, нетрадиційні параметри ICD включають, але не обмежуються ними :

- Часовий фрагмент обробки
- Розмір пам'яті
- Розмір повідомлення
- Швидкість передачі даних
- Затримка даних
- Завади даних
- Визначення програмного API та терміни
- Цілісність процесора
- Цілісність мережі

Щоб зрозуміти потребу в цих нетрадиційних параметрах ICD, розглянемо сценарій додавання нової програми до IMA, який включає наступні кроки:

1. Виділення необхідних ресурсів обробки CPM і пам'яті програми,
2. Призначення набору розділів у межах необхідних CPM на основі цілісності обробки,
3. Визначення каналів зв'язку APEX на основі обміну повідомленнями та цілісності мережі,
4. Призначення тимчасових періодів обробки в межах розділів CPM,
5. Перевірка наскрізного функціонального виконання, включаючи затримку та відмовостійкість.

Перевірка затримки наскрізної функціональної інтеграції повинна враховувати затримку буферизації джерела, тривалість виконання джерела, затримку в мережі, затримку призначення та тривалість виконання за допомогою N вихідних розділів і M розділів призначення, які беруть участь у виконанні нової програми. У військових системах для таких можливостей, як радар, акустика, датчики та канали передачі даних, зазвичай використовуються виділені ресурси обробки.

1.3. IMA в майбутньому

Розглядаючи поточне поєднання військових рішень OSA/IMA та комерційних рішень IMA, підсумок ключових атрибутів цих двох конфігурацій узагальнено в таблиці 1.2. У цьому розділі обговорюються деякі з ключових міркувань IMA наступного покоління, які потребують дослідження та зосередженості галузі.

Багатоядерні процесори на базі OSA пропонують трансформаційне покращення SWaP, але несуть із собою спільні ресурси, які можуть порушити незалежність розділів від ядра до ядра чистої IMA. Одним із підходів до пом’якшення перерв є виконання жорстких розділів у реальному часі за «ізольований період часу» основного кадру. На данний момент низка дослідницьких робіт триває, щоб вирішити цю важливу проблему за допомогою більш потужних рішень із використанням передових середовищ віртуалізації.

Високий рівень складності системи авіоніки як для OSA/IMA, так і для чистої IMA робить більш важливим, ніж будь-коли, створення деяких основних моделей даних і програмних платформ, щоб уможливити повторне використання програмного забезпечення на платформах авіоніки. У військовій авіоніці поява Future Avionics Capability Environment (FACE TM) пропонує крок у правильному напрямку. Цю сферу уваги у військовій авіоніці також просуває об'єднана спільна архітектура (JCA), яка спрямована на створення моделі даних та архітектури для наступного покоління завдань, польоту та автономності щодо одиниць прикладного програмного забезпечення портативності. У червні 2013 року армія США опублікувала проект огляду моделі JCA, який пропонує спосіб декомпозиції функціональних можливостей систем місії на домени, підсистеми та компоненти, які перебувають у взаємодії галузі.

Поява мереж OSA 10/40 Gig Ethernet, високошвидкісного бездротового підключення, управління OpenVPX і взаємозв’язку на площині даних, віддалених терміналів малого форм-фактора та багатоядерних систем на чіпі надає низку нових варіантів для змішаних мережевих технологій для підтримки пропускної здатності і

гнучкості підключення. Нові стандарти, такі як SAE AS6802 і звичайний промисловий Ethernet QoS, пропонують альтернативи для управління QoS для систем з високою пропускною здатністю.

Додавання більшої автономності для покращення безпеки та доступності тягне за собою більш агресивні обчислювальні потреби, крім багатоядерних, включаючи середовища програмування OSA GPGPU. Крім того, з належним розділенням систем завдань є нові багатоядерні операційні системи на основі гіпервізора для систем місії, які можуть бути доповненням до повного розділеного ARINC 653.

Завдяки взаємозв'язку як до інфраструктурної хмари, так і майбутнього спеціального підключення платформи до платформи, існує потреба в покращенні безпеки авіоніки в середовищі системних систем. Необхідно забезпечити належні шлюзи OSA через декілька каналів зв'язку. Інтерфейси технічного обслуговування мають бути захищені, а також шлюзи між завданнями, польотами, інформаційно-розважальними та інформаційними системами. Як результат, IMA повинні продемонструвати як сертифікацію з безпеки, так і акредитацію щодо безпеки. Потрібні додаткові вдосконалення багаторівневої безпеки.

Набори інструментів інтеграції повинні бути цілком розвинутими, щоб забезпечити ефективний аналіз і керування високоінтегрованим середовищем IMA. З'являється підвищений рівень досліджень проблеми інтеграції IMA щодо дотримання наскрізних термінів у системах, розділених у часі та просторі. Деякі ключові області досліджень для наборів інструментів IMA включають:

► Інструменти моделювання для наскрізного тимчасового розподілу,

► Інструменти моделювання для оптимізації просторового розподілу ресурсів,

► Формальні методи для систем управління польотом на основі IMA,

► Розширення програмного середовища сервіс-орієнтованої архітектури (SOA).

Відображення цих засобів подвійного використання на майбутню архітектуру OSA/IMA показано на малюнку 1.3.

Додаткові ключові елементи домену для цієї еволюції архітектури IMA, орієнтованої на завдання, будуть включати те, як передні панелі сенсорів будуть спільно використовуватися в майбутньому, наскільки добре нові датчики підключаються і будуть працюють, наскільки безпілотники продвинуть навігацію до більш інтегрованих рішень, як розвиваються передові рішення для охолодження та упаковки, а також майбутнє як для пілотованого інтерфейсу оператора, так і для пілотованих технічних засобів.

Щоб підтвердити деякі з щойно описаних тенденцій IMA, нижче наведено підсумок деяких репрезентативних досліджень IMA, що проводяться в усьому світі, крім тих, що згадувалися раніше:

► Російський державний науково-дослідний інститут авіаційних систем - Дослідження використання багатоядерних процесорів, уніфікованих відмовостійких мережевих архітектур, нових стандартів упаковки наступного покоління, багатофункціональних моносенсорів і передових програмних компонентів авіоніки.

► Європейський проект EURO-MILS - Дослідження безпечної віртуалізації для надійних додатків у критичних доменах.

► Європейський проект SCARLETT – Дослідження електронних платформ та інструментів SCALable & Reconfigurable Electronics.

► США: Industry Open RFM TM – Дослідження спільних стандартів інтерфейсу RF.

► США: DARPA High Assurance Cyber Military Systems (HACMS) – дослідження математично перевіреного програмного забезпечення для надійної безпеки та захисту.

Майбутні програми на основі IMA повинні розробити більш загальноприйнятий підхід до кордонів, які існують між постачальниками в середовищі IMA. Структура артефактів сертифікації, обсяг тестування та аналізу безпеки між організаціями, а також підвищена роль формальних методів аналізу – це нові виклики, які мають стати стандартною практикою в організаціях із сертифікації. Життєздатність систем

IMA в майбутньому залежить від постійного розвитку стандартів сертифікації, які забезпечують безпеку в високоінтегрованому середовищі, але не вимагають стандартів, які неможливо або непрактично вивести на ринок. Ці конкуруючі фактори є серйозною проблемою.

Поява більших рівнів автономності в системах авіоніки вимагатиме нового зосередження на динамічній реконфігурації IMA.

Поява вразливостей мережі в літаках, підключених до хмари, також потребуватиме нового уточнення критеріїв кібербезпеки IMA. Необхідно розглянути потребу в сертифікаціях мостів IMA для гібридних архітектур. Не всі обчислення повинні бути на ARINC 653.

Існують нові проблеми у виконанні ефективного стримування змін у середовищі IMA через архітектурні особливості та структуру артефактів сертифікації. Також виникають проблеми, пов’язані з прийняттям нових ролей та обов’язків у системній інтеграції, оскільки функції літаків частіше розподіляються між багатьма елементами системи та багатьма постачальниками систем. Зокрема, оскільки загальний збір даних і процесорний пул стає все більш популярним, нова модель продажу програмного забезпечення без апаратного забезпечення повинна бути прийнята постачальниками та з прийнятною бізнес-моделью, щоб повернути інвестиції та зробити її прибутковою. Аналогічно, інтегратори літаків повинні взяти на себе більший рівень відповідальності за системну інтеграцію на нижчих рівнях, ніж вони звикли в минулому. Інтеграція на рівні літака більше не обмежується з’єднанням юнітів разом і тестуванням їх як одиниці літака. Інтеграція на рівні літака тепер включає функціональний розподіл між системними постачальниками та керування ситуаціями безпеки та багатоступеневу інтеграцію, коли вони переходять до традиційного рівня літака. Підхід до ланцюга поставок спонукав OEM-виробників літаків посилити свої можливості для розробки програмного забезпечення та внутрішніх груп системної інтеграції на рівні постачальників.

Оскільки все більше і більше систем об’єднуються разом, масштаби загрози атаки стають все більшими і, отже, більш привабливими. Безпека вже давно стоїть на першому місці на військових ринках. Промисловість цивільної авіації традиційно зосереджується лише на безпеці, тобто на ненавмисній втраті функції або ненавмисному помилковому функціонуванні. Тема безпеки стосується того, як керувати навмисними спробами спричинити втрату або, що ще гірше, помилкову функціональність. Інтеграція в середовище IMA посилила занепокоєння щодо безпеки, якою традиційно керували за допомогою засобів контролю фізичного доступу. Наприклад, для Boeing 787 у 2008 році FAA випустила спеціальну умову, в якій говорилося: «Пропонована архітектура 787 відрізняється від архітектури існуючих (і модернізованих) літаків. Завдяки цьому новому з’єднанню пасажирів запропонований дизайн та інтеграція мережі передачі даних можуть призвести до вразливості безпеки внаслідок навмисного чи ненавмисного пошкодження даних та системи, критичні для безпеки та обслуговування літака». Boeing було притягнуто до відповідальності за докази, що мережа IMA була захищена таким чином, що пасажиру не дозволили зламати літак із розважального інтерфейсу спинки сидіння. Їм вдалося довести, що 787 забезпечив безпечне рішення IMA, що також допомогло створити прецедент безпеки в системах IMA для цивільної сфери. Переходячи до 2015 року, нещодавній звіт GAO підкреслював важливість постійного фокусування на подоланні потенційних нових кіберзагроз для військових і цивільних платформ, включаючи 787.

Інтернет речей (IoT) привів до того, що розподілені вбудовані системи стали нормою. Безпека для складних систем тепер включає критичну інфраструктуру, медичні пристрої, а в майбутньому навіть автомобілі без водія. Більш широке визнання IMA на ринку в цих кібер-фізичних системах запропонує додатковий потенціал для подвійного використання в загальному апаратному забезпеченні та мережах для паралельних доменів.

В автомобільній промисловості такі програми, як AUTomotive Open System Architecture (AutoSAR), розробляють архітектуру програмного компонента IMA,

щоб забезпечити спільне використання програмних компонентів, які будуть виконуватися на апаратно-незалежних структурах ECU. Така потенційна синергія є набагато більш імовірною спочатку в малогабаритних БПЛА, де автомобільні мікроконтролери є критичними факторами. Автомобільні ринки використовують Ethernet із запуском часу, що демонструє рішення для об'єднання IMA в різних галузях у зв'язку з системою даних. Крім того, виробники автомобілів кажуть, що злиття даних є їх проблемою номер один. Проблема того, як розробити архітектуру IMA, яка може поступово покращувати як злиття даних, так і автономію, надасть багато можливостей для дослідження подвійного використання.

З огляду на майбутнє, як безпілотні автомобілі, так і БПЛА будуть ринковими драйверами для тактичної та стратегічної взаємодії з хмарою, як показано на малюнку 1.4. У таких архітектурах безпека, доступність і ціна повинні будуть перетинати межі платформи IMA.

Враховуючи ці нові ринки IMA з подвійним використанням [наприклад, транспортні засоби, безпілотні авіаційні системи (UAS)], виникне потреба в розгляді бізнес-модулів і сегментів ринку; не кожну платформу можна примусити притримуватись загальної бізнес-моделі або заборонити її використовувати. Одна з конкретних можливостей буде стосуватися ступеня зв’язку критичної безпеки та обробки загального призначення. Це буде особливо актуально, коли з’являться нові гіпервізори та віртуальні мережі.

Висновки

IMA нового покоління обговорюється в багатьох кібер-фізичних системах за межами традиційних автомобільних, космічних та аерокосмічних ринків. Кожен з цих IMA матиме власні моделі даних та вибрані набори загальної інфраструктури

обладнання, спільної інфраструктури програмного забезпечення та уніфікованої мережевої інфраструктури для розміщення систем змішаної важливості. Високоінтегровані рішення в галузі цивільних БПЛА та автомобільної промисловості можуть запропонувати нові стандарти з відповідних доменів IMA, які слід розглянути для подвійного використання. У той же час, архітектури IMA високого класу повинні будуть приймати важливі рішення щодо обробки, програмного забезпечення та мережевих технологій, які зараз керуються високоякісними Інтернет-технологіями, розподіленими вбудованими системами. На малюнку 1.6 наведено приклад деяких ключових компонентів, які можуть мати значення для розробки в межах і всередині доменів IMA щодо безпеки для авіоніки літаків та невеликих БПЛА. У майбутньому IMA продовжить розвиватися у своїй реалізації, щоб забезпечити безпечніі, масштабовані та доступні рішення для розширення ринку.

Розділ 2. Еволюція мереж авіоніки від ARINC 429 до AFDX

Вступ

Сигналізація та міжсистемний зв’язок у авіоніці були головною темою з тих пір, як електронні пристрої вперше почали використовуватися в літаках. Спочатку простий сенсорний зворотний зв’язок і компоненти, такі як радар і двигуни, потрібно було з’єднати з елементами керування в кабіні. З плином часу все більше і більше систем, які передають і приймають дані, впроваджувалися в авіоніку, які в якийсь момент стали вирішальними навіть для найважливіших завдань, таких як рульове керування, а потім і передача даних. Щоб впоратися з цими проблемами в комерційній авіоніці, стандарти, такі як ARINC 419 (а пізніше 429), були розроблені та прийняті не лише окремими корпораціями, а разом майже всією галуззю.

На сьогоднішній день ARINC 429 можна знайти в більшості активних і звільнених серії літаків. Незважаючи на те, що він добре зарекомендував себе в галузі, він був мало адаптований і розширений з тих пір, як початкові специфікації були сформульовані в кінці 1970-х років. На відміну від стандартів авіоніки, численні технологічні революції відбулися в комп'ютерній індустрії швидкими темпами. Мережа комп'ютерів на борту літака, можливо, була немислимою в 1970 році, в той час як сучасні літаки без будь-яких мережевих комп'ютерів зустрічаються дуже рідко. Застарілі стандарти зв'язку авіоніки все ще відображають минулі погляди на обчислювальну техніку.

Зрештою, сучасна мережева архітектура, для використання авіоніки, повинна забезпечувати максимальну безпеку, резервування та безпеку, а також застосовувати безвідмовні параметри за замовчуванням. Отримана інфраструктура повинна бути ефективна в обслуговуванні, гнучкою та забезпечувати міцну основу для розробки програмного забезпечення. Останні стандарти відображають ці вимоги, хоча мало хто з них знайшов широке застосування в галузі.

На відміну від Інтернету, безпека та економічна ефективність не є ключовими цілями авіоніки; радше безпека. Однак більшість сучасних мережевих стандартів спрямовані на досягнення традиційних цілей безпеки світу ПК і лише опосередковано вирішують вимоги безпеки (шляхом виконання традиційних цілей безпеки).

У ARINC 664 Частина 7, також відомий як AFDX, стандартна технологія Ethernet розширена, а цілі проектування базуються на безпеці.

У наступному розділі детально пояснюється найпоширеніший стандарт ARINC 429. У розділі 3 зображено перехід від федеративних мережевих архітектур, таких як 429, до сучасної інтегрованої модульної авіоніки .

Проведено дослідження еталонної операційної системи, запропонованої в ARINC 653 для використання з інтегрованими архітектурами. У розділі 4, ARINC 629 і Mil-Std-1553, коротко представлені два новітні мережеві стандарти. Розділ 5

зосереджено на мережевому стандарті AFDX. Акцент робиться на покращеннях Ethernet, необхідних для відповідності вимогам авіоніки. Останній розділ присвячений узагальненню переваг і недоліків двох основних іменованих архітектур.

2.1. Структура ARINC 429

У наступному розділі стандарт ARINC 429 буде детально описано. Особливо зупинимося на його архітектурних принципах, історії та можливостях. Далі будуть окреслені обмеження, що накладаються на проектування мереж.

ARINC 429 (повністю, Mark 33 Digital Information Transfer System), реалізує послідовний лінійний зв'язок і був одним із перших стандартів, спеціально орієнтованих на застосування авіоніки. Попередник ARINC 429, комерційний стандарт 419, вперше був опублікований у 1966 році, а 429 був заснований (з точки зору 1978 року) на новітній технології.

429 визначає пряме підключення LRU за допомогою послідовного інтерфейсу на основі витої екранованої пари, який може з’єднувати однорангові пристрої на відстані приблизно 90 метрів один від одного. На відміну від сучасних мережевих протоколів, це стандарт сигналізації; таким чином відправник завжди відправляє на лінію, а одержувачі завжди читають з неї. Якщо дані недоступні для відправки, лінія встановлюється на нульову напругу. Незважаючи на те, що використовуються кабелі з витою екранованою парою , усі лінії є симплексними з’єднаннями, заповненими однією станцією-відправником і кількома одержувачами (до 19), як показано на малюнку 1.

Шина, яку також називають багатоцільовою шиною , може працювати на низькій або високій швидкості . Низька швидкість використовує змінну тактову частоту і номінальну пропускну здатність 12-14 кбіт/с, тоді як високошвидкісний режим вимагає фіксованої тактової частоти і дозволяє 100 кбіт/с.

У термінології ARINC 429, кожен фрагмент даних, що передається за посиланням, називається словом ; формат слова визначено в наступному розділі. Існують два типи слів: слова даних і контрольні слова повідомлення. Повідомлення складаються з кількох слів даних, які потім називають записами.

Повідомлення керування посиланнями використовуються так само, як і в сучасних мережевих стеках. Якщо LRU, що прослуховує, готовий отримати дані, повідомлення «Запит на відправку» буде передане через його виділене посилання для відправлення. «Очистити для відправки» використовується для протилежного. «дані слідують», «дані отримані нормально», «дані отримані не в порядку» та «синхронізація втрачена» можуть використовуватися відправником/одержувачем для подальшої взаємодії. Кожне повідомлення починається з контрольного слова «дані слідують» , за яким слідує до 126 слів даних.

На відміну від сучасних мережевих стандартів, оскільки 429 визначає односпрямовану та симплексну шину, LRU-одержувачі можуть не надсилати повідомлення на шину, яку вони прослуховують. Сюди входять повідомлення, пов’язані з керуванням посиланнями.

Станція може бути приєднана до кількох шин і працює як відправник, так і як одержувач, тому можливі ієрархічні схеми. Однак двонаправлений обмін інформацією між системами потрібен, принаймні, для контролю та підтвердження повідомлення. У цьому випадку LRU-одержувачі відповідають через вторинний інтерфейс, на якому позиції відправника та одержувача змінюються місцями. Оскільки роль відправника може виконувати лише одна станція, яка бере участь у каналі ARINC 429, потрібен один зворотний канал для кожного LRU.

Визначено кілька форматів кодування даних, які орієнтовані на різні сценарії використання в авіоніці: двійковий , двійковий десятковий (BCD, див. малюнок 3), дискретні дані, передача файлів із використанням набору символів ISO 646, а також дані обслуговування та підтвердження.

Структура ARINC 429 слів не відповідає сучасним стандартам комунікації; він не вирівняний по байтам, але оптимізований для зниження затримки.

Загальні поля в усіх форматах слова:

- етикетка (8 біт),
- ідентифікатори джерела та призначення (необов'язково, 2 біт),
- знак / матриця стану (2 біта), і
- поле даних (19 біт).

Слова закінчуються одним бітом парності. Загальний розмір усіх слів становить 32 біти, як показано на малюнку 2.

Невеликий розмір повідомлення призводить до дуже низької затримки, мінімізує затримки під час обробки та гарантує час, оскільки не може статися черга на транспортній стороні або перепланування трафіку. Розмір повідомлення є одним із наріжних каменів для досягнення безпеки, стійкості та надійності. Жоден інший стандарт зв’язку, який зараз використовується в авіоніці, не пропонує такого ж рівня реагування, що робить його цікавим вибором для додатків, де затримка в кілька мілісекунд може бути занадто великою.

Залежно від вибраного формату даних, прапорці Sign/Status Matrix мають попередньо визначене значення; наприклад, якщо використовується формат BCD, встановлення обох бітів SSM на нуль означає північ, схід, правий, до (направлений), вище або плюс. Інші бітові шаблони та варіанти кодування даних мають різні попередньо визначені значення, які LRU мають підтримувати для збереження сумісності.

Мітка слова використовується як заголовок кадру, що містить інформацію про формат кодування та три вісімкові числа, які можуть використовуватися LRU для вибору відповідних повідомлень після отримання. Мітка може бути розширена на три біти, які потім служать четвертою цифрою мітки. Цифрові коди етикеток попередньо визначені в стандарті і також мають фіксоване значення. Мітка спочатку

надсилається старшим розрядом, тоді як решта повідомлення спочатку надсилається найменшим бітом.

Через спрощену компоновку з 429 посилань кожне окреме з’єднання є фізичним кабелем, що дозволяє легко тестувати, оскільки може бути несправним або LRU, або сама лінія. Це також створює серйозні проблеми при проектуванні систем із щільним взаємозв’язком.

Як показано на малюнку 4, навіть середовище з кількома присутніми станціями може стати дуже складним, як тільки потрібен певний ступінь взаємодії. У сучасних комерційних літаках може виникнути взаємозв’язок між безліччю систем, а також надзвичайні витрати кабелів, що накладає серйозні обмеження на проект мережі, а також впливає на загальну вагу такого судна.

Виправлення бітових помилок за допомогою симетричних або криптографічних алгоритмів контрольної суми не розроблено для ARINC 429, а натомість має бути реалізовано на рівні програми. Фактично, вся обробка даних повинна здійснюватися безпосередньо програмним забезпеченням кожного LRU. Уніфікованого, незалежного від пристрою стека програмного забезпечення 429 не існує.

Загалом, для реалізації установки ARINC 429, яка є поширеною в аерокосмічній промисловості, потрібне спеціальне (і, отже, дороге) обладнання. Майже немає готового обладнання для споживачів, за винятком кабелів. Однак слід зазначити, що на кабельну прокладку застосовуються окремі авіаційні стандарти. Розробка програмного забезпечення також є проблематичною, оскільки не можна використовувати сучасні мережеві протоколи, а розробка здійснюється для вузькоспеціалізованого обладнання. Таким чином, модернізація старих літаків новою технологією може бути дорогою.

2.2. Принципи побудови IMA

Як було зазначено в попередньому розділі, об'єднані архітектури серйозно обмежують масштабованість і гнучкість комп'ютеризованого середовища з щільним взаємозв'язком. Таким чином, у цьому розділі ми розглянемо сучасну альтернативу об’єднаній авіоніці. Буде представлена перша інтегрована модульна авіоніка, а потім короткий аналіз ARINC 653.

ARINC 429 дозволяє реалізувати об'єднані архітектури мережі. Він не розрізняє апаратне та програмне забезпечення, а скоріше між пристроями, спеціально створеними для однієї мети (наприклад, опитування датчика, передача радіолокаційних сигналів, подача команд рульового керування тощо). Таким чином, LRU до певної міри унікальні, а також повинні бути сертифіковані в цілому. Оскільки немає різниці між апаратним і програмним забезпеченням, повторна сертифікація має відбуватися, навіть якщо оновлено або замінено лише програмне забезпечення.

Оскільки програмне забезпечення в майбутньому може вимагати додаткових системних ресурсів (більше оперативної пам’яті, дискового простору, потужності ЦП, ...), пристрої необхідно створювати з відповідними резервами. Компоненти в об’єднаних мережах (зазвичай) не можуть поділитися ресурсами, і резерви повинні бути визначені на системному рівні, в усій мережі, що призводить до високого рівня неактивності, додаткової ваги та додаткових витрат. Загальним результатом є запатентована мережа з потенційно екстремальними накладними кабелями, що містить безліч різних змінних ліній або модулів, які виконують унікальні ролі.

Через описані обмеження аерокосмічна промисловість почала відходити від федеративних архітектур на користь інтегрованої модульної авіоніки (IMA). Найважливішим є те, що IMA розрізняє програмне забезпечення та апаратне забезпечення та визначає рівень абстракції між фізичним обладнанням та програмно-реалізованими функціями. Однак IMA все ще дозволяє використовувати монолітні, унікальні LRU, але заохочує гомогенізацію бортового обладнання.

Декілька функціональних пакетів, реалізованих різним програмним забезпеченням, можуть виконуватися на одному хості IMA, як показано на малюнку 5. Таким чином, програми повинні бути ізольовані один від одного, порти вводу/виводу та системні ресурси повинні бути надані відповідним програмам. Оскільки ресурси на хості IMA є спільними і доступними для всіх програм, резерви можна розрахувати спільно для всіх запущених програм.

Як описано в наступних розділах, тимчасові та часові обмеження мають вирішальне значення при проектуванні авіоніки та мереж. Ці обмеження не можуть бути задоволені звичайними операційними системами; отже, використання операційної системи реального часу (RTOS) є обов'язковим в IMA. Еталонна реалізація такої операційної системи зазначена в стандарті ARINC 653, включаючи відповідний API та концепції зв'язку для використання на хості IMA. Стандарт поділяється на різні частини, охоплюючи основні та додаткові послуги, які можуть або повинні надаватися ОС, а також рекомендації щодо тестування.

На відміну від операційних систем, які використовуються в більшості комп’ютерної індустрії, API APEX 653 дозволяє програмам залишатися повністю незалежними від архітектури базової системи, як показано на малюнку 7. Він надає програмам доступ до апаратного забезпечення, дозволяє обробляти файли за допомогою спеціальної основної служби. і дозволяє отримати доступ до об'єднавчої панелі віртуальних мереж. Додаткам призначаються порти черги, які є буферами FIFO для пакетів, і порти вибірки, які є буферами одного пакета, які перезаписуються на кожній ітерації. У поєднанні з APEX порти зв’язку дозволяють створювати мережу з детермінованим часом.

У 653 визначено основні компоненти ОСРВ; багато компонентів ідентичні або дуже схожі на ті, що використовуються в звичайних неавіотехнічних операційних системах реального часу, і тому не будуть описуватися далі. Інші, такі як керування розділами та монітор працездатності, зустрічаються рідко. Монітор працездатності забезпечує апаратне, програмне забезпечення та стану мережі інформацію ОС, а

також розділи. Таким чином, як основні служби ОСРВ, так і програми повинні підтримувати функції моніторингу працездатності.

Зворотній зв'язок щодо несправностей, пов'язаних із системними службами, дозволяє монітору працездатності ініціювати контрзаходи у разі збою (наприклад, переміщення пам'яті, запуск процедур обслуговування тощо). Якщо розділи програми отримують відповідну інформацію про проблеми, вони також можуть самостійно вживати контрзаходів, вибираючи інший шлях до датчика в мережі або перемикаючись на функцію перемикання збоїв.

Моніторинг здоров’я вимагає класифікації та категоризації на основі інформації, наданої автором програми або виробником окремого пристрою. Таким чином, помилки, що стосуються моніторингу працездатності, обробляються і також виявляються на різних рівнях на хості IMA. Інформація про всі ці стратегії обробки помилок збирається в таблицях стратегій відновлення на рівні окремих розділів і на рівні системи. Самі таблиці повинні підтримуватися авіаконструктором.

Ізоляція була однією з ключових цілей проектування в 653 і IMA, оскільки вони повинні дозволяти виконання

програм без побічних ефектів. Програми повинні лише належним чином використовувати APEX і можуть бути повністю ізольовані один від одного або можуть бути дозволені IPC. APEX і базові системні бібліотеки зазвичай розглядаються як один функціональний блок. Кожна програма запускається всередині ізольованого розділу. Служби керування розділами ОСРВ відповідають за призначення пріоритетів і обмежень часу для окремих розділів програми.

Тому, якщо впроваджено ARINC 653, апаратне та програмне забезпечення можна сертифікувати поступово. Якщо програмне забезпечення замінено або оновлено, лише окрему програму або ОС потрібно повторно сертифікувати. Базове обладнання залишається незмінним і не потребує додаткових кроків, крім планування на випадок надзвичайних ситуацій та зменшення потреби в можливих додаткових системних ресурсах.

Суворі вказівки також накладаються на дизайн додатків. У 653, серед інших рекомендацій, визначено спосіб, яким програмне забезпечення та основні служби ОС повинні зберігати та обробляти файли (тобто файли конфігурації повинні зберігатися як файли XML). Додатки, сумісні з APEX, можуть працювати постійно або тимчасово, і, таким чином, можуть бути запущені, зупинені або переміщені на інші хости IMA, якщо це дозволено конфігурацією систем і мереж. Таким чином, IMA вимагає абстракції мережевого обладнання, щоб задовольнити вимоги до програмного забезпечення.

2.3. Стандарти мережевих протоколів

Строго об'єднані мережі ARINC 429 не пропонують логічної абстракції, необхідної для розгортання інтегрованої модульної авіоніки; необхідний спільний носій і логічна абстракція інтерфейсів. У наступному розділі будуть коротко описані застарілі архітектури мережі, що пропонують такі абстракції. Спочатку ми розглянемо ARINC 629; хоча він рідко використовується в авіоніці, він представив дуже важливу концепцію для досягнення детермінізму в спільній середній мережі. Потім ми розглянемо один з найбільш поширених військових мережевих стандартів, Mil-Std-1553b.

ARINC 629, також відомий як цифровий автономний термінальний доступ, був розроблений спільно Boeing і NASA для подолання обмежень, накладених 429, і пізніше був переданий ARINC. Хоча він більш сучасний, ніж попередні стандарти, він ніколи не знайшов широкого застосування в промисловості. Він майже виключно використовувався в Boeing 777, і навіть цей літак містив додаткову 429 резервну інфраструктуру. Згідно з 629, передбачалося розгортання триплекс-шини, подібної до тієї, що визначена 10BASE2/10BASE5. Він також реалізує CSMA/CD, що використовується в Ethernet; таким чином, на автобусі можуть виникнути зіткнення. Оригінальний стандарт визначав тактову частоту 2 МГц, яку можна підвищити в міру розвитку технологій, а шина була побудована на витій парі з

самого початку. Як і ARINC 429, він використовує 32-розрядні слова даних, але елемент даних вирівняний за 2 байтами.

У той час як архітектури на основі 429 підтримують лише прямий одноранговий зв’язок, 629 в основному використовує спрямовану одноадресну передачу та дозволяє широкомовну передачу на спільному носії. Це була рання спроба підвищити гнучкість і зменшити зусилля на кабелі. Гнучкість і додаткова складність призвели до збільшення затримки сигналу та зниження детермінізму, тоді як концепція в цілому все ще не підтримує апаратне забезпечення COTS. Таким чином, доступ до шини кількома станціями в мережі 629 може відбуватися у випадковому порядку, залежно від того, яка система надсилає першою.

Частини основного протоколу в 629 знайшли свій шлях у сучасні стандарти, особливо використання заздалегідь визначених проміжків між передачами для запобігання зупинки або втрати пакетів, щоб додати детермінізму. Як показано на малюнку 6, розрив синхронізації є випадковим інтервалом для кожного кадру, тоді як термінальний проміжок є відмінним для кожного терміналу.

MIL-STD-1553b, мультиплексна шина даних командування/відповіді внутрішнього розподілу часу літака, широко використовується у військових літаках (наприклад, A400M Airbus) і навіть на Міжнародній космічній станції. Це цікавий підхід, який також допоміг подолати розрив між базовою сигналізацією та сучасними мережевими стандартами, такими як AFDX. Стандарт був розвинений від початкової застарілої швидкості передачі даних 1 Мбіт/с до розширеного та гіпер-варіантів, які використовують новітнє обладнання, щоб пропонувати 120 і 200 Мбіт/с відповідно. На відміну від інших стандартів на основі шини, MIL-STD-1553b визначає логічну зірку поверх топології фізичної шини. Ця топологія називається надійним фізичним рівнем і використовує триаксіальний кабель.

У стандарті визначено роль контролера шини . Цей пристрій відповідає за ініціювання всього зв’язку між підсистемами (рівноправними) на шині через протокол командної відповіді. У разі відмови контролера шини цю роль на даний

момент може взяти інший віддалений термінал . Щоб забезпечити відмову, декілька резервних екземплярів шини працюють паралельно (див. малюнок 8), в той час як кожна шина дозволяє лише напівдуплексний зв’язок.

Підсистеми, що надсилають дані через шину, не підключаються безпосередньо, а замість цього використовують окремі віддалені термінали для доступу до мережі. Усі комунікації на шині контролюються монітором шини , який також може виконувати реєстрацію частин або всього зв’язку. Зазвичай передачі є одноадресними, і, таким чином, обмінюються лише між двома віддаленими терміналами. Трансляція підтримується задумом, але не рекомендується.

У 1553 слова даних мають довжину всього 20 біт, але накладні витрати на протокол менше, ніж в інших стандартах, оскільки всі комунікації спрямовуються контролером шини , а однорангові користувачі виконують лише команди, які їм були дані (наприклад, читання з шини, відправлення на термінал) .

Щоб заощадити бітовий простір, існують три формати слів:

- командне слово (відправляється контролером шини),
- слово стану (відповідь від LRU на контролер шини),
- і формат слова даних.

Хоча стандарт все ще несумісний із сучасними мережевими протоколами та не дотримується моделі рівня OSI, він дозволяє LRU емулювати логічні зв’язки, що необхідно для архітектур інтегрованих систем.

2.4. Повнодуплексний обмін пакетами даних

У цьому останньому розділі ми детально розглянемо комутацію Avionics Full-DupleX Ethernet (AFDX). Ми проаналізуємо, як він був розроблений для розширення стандарту Ethernet, щоб відповідати сучасним вимогам в аерокосмічному

середовищі. Далі будуть описані ключові елементи мережі AFDX та зміни, необхідні в протоколах верхнього рівня.

Як розвинений стандарт, 429 мав багато обмежень, але це перевірений і широко використовуваний протокол. З плином часу і розвитком технологій з'явилася більша пропускна здатність, більш гнучкі топології та нові проблеми, такі як інтегрована модульна авіоніка, які вийшли за межі можливостей ARINC 429.

ARINC 664 (Частина VII) спочатку був розроблений підрозділом EADS Airbus як комутація Avionics Full-DupleX Ethernet (AFDX). Незважаючи на те, що попередні літаки вже використовували повністю електронні комунікаційні системи, проводка з використанням попередніх стандартів більше не могла відповідати вимогам сучасних сучасних літаків. У випадку з AFDX Airbus A380 запропонував впровадити нову технологічну базу; таким чином було створено AFDX. Пізніше Airbus AFDX був перетворений у дійсний стандарт ARINC. На малюнку 9 показано просту мережу на основі AFDX.

Ethernet використовується протягом десятиліть за межами аерокосмічної промисловості і виявився надійною, недорогою, розширюваною та гнучкою технологією. Однак він не може запропонувати основні функції, необхідні для високої доступності та надійності. Таким чином, він безпосередньо не підходить для авіоніки. 664 пропонує сучасні швидкості передачі даних, будучи на основі раніше дуже ненависного стандарту Ethernet 802.3. AFDX успадковує частини термінології MIL-STD-1553 і загальні налаштування. Пристрої, що передають дані через мережу, називаються підсистемами , які підключаються до мережі через кінцеві системи . Сама повнодуплексна мережа називається AFDX Interconnect ; в термінах Ethernet це включає всі пасивні фізичні частини мережі, але не комутатори та інші активні пристрої.

Найбільш помітною перешкодою для використання мережі Ethernet в авіоніці є недетермінізм Ethernet. Для звичайних комп'ютерних мереж втрата пакетів і тайм-аути є звичайним явищем проблема. Верхні рівні, такі як операційна система або

програми станції, повинні вирішувати ці проблеми задумом. Якщо повідомлення буде втрачено або пошкоджене під час передачі, воно буде просто надіслано або його втрата повністю пом’якшена. Під час надсилання даних у не мікросегментовану мережу можуть виникнути зіткнення в кожному сегменті, що змушує всі станції, які беруть участь у зіткненні, надсилати повторно. Передача пакетів повторюється через випадковий інтервал часу, незалежно від того, яка станція почнеться першою. Знову ж таки, може статися зіткнення, яке може призвести до майже невизначеного повторення, що згодом може призвести до заклинювання шини.

Іншим змінним фактором мережі Ethernet, а потім і ARINC 664, є комутатори/мости. Хоча вони додають гнучкості в мережу, вводиться додатковий недетермінізм, оскільки кадри можна змінювати або маніпулювати під час передачі. Перемикачі пропонують мікросегментацію сегментів мережі, але, у свою чергу, також збільшують кількість переходів, які кадр виконує від джерела до призначення. Таким чином, затримка збільшується, а поведінка в часі може змінюватися, якщо кадри рухаються за кількома шляхами. У дуже перевантажених установках комутатори можуть навіть навмисне скинути пакети, якщо були досягнуті межі буфера.

У Ethernet колізії обробляються через CSMA/CD, але верхні рівні можуть зіткнутися з втратою пакетів. Там протоколи (наприклад, TCP, SCTP тощо) в мережевому стеку операційної системи повинні мати справу з втратою пакетів. Однак це не життєздатне рішення в умовах критичного безпеки. Деякі програми вимагають гарантій пропускної здатності, тоді як інші можуть вимагати, щоб поведінка часу залишалася в строгих межах. Ні те, ні інше не може бути запропоновано Ethernet. Чи не важко якості обслуговування гарантій не є в ванілі Ethernet, і м'яке планування пропонується тільки через розширення протоколу , такі як Ethernet-QOS IEEE 802.1p. Те ж саме стосується розподілу пропускної здатності, яке не може бути гарантовано в Ethernet на рівні кожного потоку, але реалізується за допомогою різних алгоритмів. Хоча існує кілька власних підходів для того, щоб Ethernet можна

було використовувати в середовищах реального часу, жоден із цих стандартів не можна використовувати безпосередньо в авіоніці. Таким чином, новий стандарт вимагав детермінізму, щоб його можна було використовувати в авіоніці.

Ethernet не залежить від фізичних з'єднань і дозволяє визначити логічні кінцеві точки. Таким чином, кілька фізичних або віртуальних пристроїв можуть спільно використовувати одне посилання, підтримуючи віртуальні підсистеми або віртуальні машини в IMA. Для кількох додатків або пристроїв можуть знадобитися різні часові характеристики або фіксована мінімальна пропускна здатність.

Віртуальні з'єднання «точка-точка» реалізують ту ж концепцію, що використовується в ARINC 429. На відміну від 429, вони існують не фізично, а як логічні зв'язки. Вони реалізовані як віртуальні посилання (VL) поверх рівня Ethernet AFDX. Приклад віртуальних каналів наведено на малюнку 10. До певної міри VL дуже схожі на тегування VLAN, як визначено в IEEE 802.1Q, але пропонують додаткову інформацію на додаток до ізоляції мережі. Кожен віртуальний канал, крім ідентифікатора каналу, має три властивості: проміжок розподілу пропускної здатності, максимальний розмір кадру L2, який називається LMAX або Smax, і обмеження пропускної здатності.

LMIN і LMAX використовуються для встановлення попередньо визначеного найменшого та найбільшого загального розміру кадру Ethernet вздовж шляху передачі пакету, та може зайняти, усуваючи необхідність фрагментації IP-пакетів та подібних механізмів, таким чином усуваючи ще одне джерело недетермінізму з Ethernet.

Як тільки буфери, наявні в комутаторах або кінцевих станціях, заповнені, фізичні канали можуть стати перевантаженими, і пакети будуть втрачені або затримані під час переходу. У критичних для безпеки середовищах це неприпустимо; таким чином, було визначено так званий розрив розподілу пропускної здатності (BAG). BAG визначається програмним забезпеченням для кожного VL, встановлюючи затримку (1-128 мілісекунд) між відправкою кадру VL і наступним. У сценаріях

перевантаженої мережі кадри надсилатимуться протягом часового вікна, заданого BAG.

Встановивши LMAX і розумний BAG, пропускна здатність сегментується і, таким чином, може бути гарантована для кожного VL окремо. Недоліком запровадження розподілу часу є значна втрата пропускної здатності, якщо визначено кілька посилань із різними властивостями. Таблиця 1 відображає цю втрату пропускної здатності залежно від параметрів мережі. Дані взяті з бенчмаркінгу, проведеного NASA у на основі конфігурації допоміжної сенсорної мережі 100 Мбіт/с. Навіть якщо надсилається мало даних, один VL з високим BAG (довгим часом очікування між кадрами) і низьким LMAX (невеликий підтримуваний розмір кадру) може різко знизити загальну пропускну здатність мережі.

Віртуальні посилання позначаються за допомогою так званих ідентифікаторів віртуальних посилань (VLID). VLID замінює доставку на основі MAC-адреси, займаючи біти, які зазвичай використовуються для MAC-адреси призначення. Щоб зберегти сумісність з Ethernet, AFDX розбиває поле MAC-адреси призначення на кілька частин: початкові біти встановлюються так, щоб відображати локально керовану MAC-адресу (локальний сайт), останні 16 біт зберігають VLID.

Тільки одна підсистема може надсилати дані за допомогою даного VLID, тому віртуальні посилання знову односпрямовані. Як і в ARINC 429, підсистема може виконувати різні ролі в кількох VL, використовуючи різні порти (див. нижче), і кілька одержувачів можуть брати участь у віртуальному зв’язку . Підсистеми не адресовані явно, як у звичайному Ethernet, де використовуються MAC-адреси, але значення ідентифікатора віртуальних посилань визначається та забезпечується конфігурацією мережі. Однак частини вихідної області даних MAC-адреси визначаються користувачем.

Для використання можливостей AFDX традиційного програмування сокетів недостатньо. Таким чином, був доданий шар абстракції: комунікаційні порти. Підсистема отримує доступ до портів зв’язку, тобто портів вибірки та черги , через

спеціальний мережевий API (див. ARINC 653). Порти призначаються віртуальним посиланням і використовуються для обміну даними між підсистемами; кінцеві системи доставляють повідомлення до одного з портів підсистеми, кілька портів у підсистемі можуть бути членами VL, але кожен порт може бути приєднаний лише до однієї VL.

Порти вибірки мають виділені буферні простори, в яких можна прочитати та зберегти одне повідомлення. Якщо надходить нове повідомлення, попередні дані будуть перезаписані. А масового обслуговування порту буфер може містити до фіксованого числа повідомлень, які зберігаються в черзі FIFO; після прочитання найстарішого повідомлення воно видаляється з черги. Послуги обробника для портів зв’язку необхідно надавати відповідно до специфікацій ARINC 653. BAG і LMAX VL повинні бути встановлені відповідно до колективних вимог усіх портів, які беруть участь у каналі.

Планування виконується на рівні порту та каналу в кожній кінцевій системі незалежно на основі раніше названих властивостей. Кінцеві системи гарантують, що віртуальні канали не перевищують їх межі пропускної здатності при мультиплексуванні передачі з різних портів і віртуальних каналів . Крім того, тремтіння має утримуватися у фіксованих межах (визначених стандартом), оскільки передачі кадрів компенсуються тремтінням у BAG. Віртуальні посилання плануються до того, як резервування буде застосовано до передач.

Згодом дані, що проходять через віртуальний канал, завжди будуть проходити одним і тим же шляхом в активній мережі AFDX, оскільки вони попередньо визначені на AFDX-перемикачах уздовж маршруту. Оскільки реконфігурація пристроїв AFDX не відбувається під час виконання, цей шлях зберігатиметься, доки відповідні пристрої в мережі не будуть повторно ініціалізовані. Це означає, що поведінка синхронізації під час виконання залишатиметься незмінною для кожного кадру (як визначено BAG), навіть якщо окремі пристрої під час польоту виходять з ладу. Поодинокі збої не впливатимуть на загальну установку через надану резервування. Детальніше про AFDX-перемикання буде розглянуто в Розділі 5.2.4

Середовища високої доступності також вимагають резервування на шині, а також на станціях. Знову ж таки, Ethernet за замовчуванням не пропонує жодного роду перемикання збоїв, однак додаткова агрегація каналів, визначена в IEEE 802.1AX, може запропонувати таку функціональність. 664 за проектом визначає складні концепції резервування для кінцевих станцій, а також для кабелів, забезпечуючи дві виділені мережі (мережа A і B). Після планування кадрів Ethernet вводиться резервування. Кожна підсистема AFDX має два інтерфейси, які називаються кінцевими системами . Надлишковість додається прозоро шляхом надсилання кожного кадру через обидві кінцеві системи із застосуванням порядкового номера кадру (див. малюнок 10). Якщо припустити, що помилок передачі не сталося, один дублікат прибуде до місця призначення для кожного переданого кадру.

Повторювані кадри необхідно виявити та відкинути. Для цього до кожного пакету рівня 3 OSI був доданий один байтовий лічильник послідовності, перетворюючи результуючий кадр Ethernet у кадр AFDX, як показано на малюнку 11. Послідовний номер AFDX додається як кінцева інформація до корисного навантаження рівня 2. Для кожного пакета, надісланого через віртуальне посилання, лічильник збільшується. Перший кадр з правильною контрольною сумою для надходження в підсистему використовується для обробки, наступні кадри з повторюваним порядковим номером можуть бути ідентифіковані та відкинуті.

У разі збою на з’єднанні AFDX, програмне забезпечення та комутатори можуть бути поінформовані про цю проблему і можуть перенаправляти трафік за іншими, заздалегідь визначеними шляхами перемикання в межах тієї ж мережі, не покладаючись виключно на незалежне резервування мережі.

AFDX наразі підтримує швидкість лінії до 1 Гбіт/с, але може підтримувати швидший транспорт у міру розвитку технологій. Кінцеві системи підключаються до комутаторів безпосередньо і використовують повнодуплексні канали. Як визначено в стандарті та завдяки використанню повністю комутованого Ethernet, у кожному сегменті рівня 2 AFDX існують лише дві станції: кінцева система і комутатор, два комутатори або дві кінцеві системи . Таким чином, домени зіткнення зведені до

мінімуму. Згодом зіткнення кадрів, які необхідно розв’язати через CSMA/CD на одному сегменті, по суті, не повинно відбуватися.

Більшість функцій, з яких складається AFDX, також можна реалізувати за допомогою звичайного обладнання Ethernet, якщо запустити спеціальні реалізації стеку AFDX. Хоча існують суто програмні реалізації, ці рішення не можуть гарантувати детермінізм. Вони не можуть утримувати тремтіння в межах, встановлених AFDX, і корисні лише для базового тестування сумісності.

Щоб досягти детермінізму, спеціалізоване обладнання для застосування правил Virtual Link, які засновані на параметрах VL. потрібен ARINC 664. Перемикачі AFDX виконують цю роль і забезпечують затримку, обмеження пропускної здатності для VL і забезпечують надійну фіксовану конфігурацію. Ця конфігурація зчитується під час завантаження і залишається постійною під час виконання, щоб уникнути коливань у топології мережі та забезпечити однакову поведінку часу.

З міркувань цілісності, комутація каналів «зберігати і пересилати» використовується при передачі пакетів, на відміну від більшості сучасних високошвидкісних комутаторів Ethernet, які виконують комутацію наскрізне. Конфігурація для всіх віртуальних зв'язків (LMIN, LMAX, BAG, пріоритет) і параметри комутатора повинні бути встановлені відповідно до однієї з математичних моделей перевірки, які використовуються сьогодні.

Виправлення параметрів мережі під час завантаження запобігає змінам під час виконання, а мережа зберігає постійні властивості часу та статичний макет під час роботи. Відхилення від налаштувань за замовчуванням можуть не відбутися і враховуються під час математичного розрахунку глобальних параметрів. Перемикачі ізолюють віртуальні посилання один від одного і виконують планування для прохідних кадрів на основі їх VLID. Інші параметри, зазначені в конфігурації комутатора та системи, включають пріоритет, LMIN (еквівалент LMAX) і тремтіння для Virtual Link . Порти мають фіксовану максимальну затримку та розмір буфера.

Сертифікація компонентів для використання в середовищі авіоніки вимагає доказових властивостей і зазвичай призводить до гіршого випадку, але без перевантажень. Мережне обчислення широко використовується, але альтернативні підходи, такі як моделі на основі траєкторії або перевірка моделі, також були б життєздатними альтернативами; загальним для всіх є формальний доказ правильності функціональності мережі.

AFDX дотримується моделі рівня OSI і базується на загальних протоколах зі світу Інтернету. Згодом використовуються знайомі протоколи, такі як IP, UDP і IP-multicast. Чужорідні мережеві середовища, такі як посилання ARINC 429, можна прозоро транспортувати в рамках віртуального каналу до окремих додатків, тим самим зменшуючи зусилля на розробку. Фактично, практично будь-який попередній мережевий стандарт, який за можливостями не перевищує ARINC 664, може бути реалізований поверх нього.

На рівні 3 розгортається протокол IPv4, хоча поля, які зазвичай використовуються для IP-адрес джерела та призначення, були перепризначені, як показано на малюнку 12. Версія верхнього пакета показує IP-пакет, який спрямовується до окремої системи за допомогою VLID, тоді як нижній пакет використовує багатоадресну адресацію. 32 біти поля вихідної IP-адреси розділені на:

- однобітовий ідентифікатор класу,
- 7-розрядна приватна адреса,
- визначений користувачем 16-бітовий ідентифікатор,
- а також 8-розрядний ідентифікатор розділу .

Ідентифікатор розділу використовується для адресації віртуальних підсистем у віртуалізованому середовищі IMA.

IP-адреса призначення або використовується для позначення багатоадресної передачі

IP-адреса або містить поле з 16 біт із префіксом VLID. Перші 16 біт містять фіксоване число (зазначене стандартом), а друга частина містить VLID, якщо використовується пряма IP-адресація та IMA.

Завдяки гарантіям, наданим AFDX, деякі функції, які зазвичай вводяться на вищих рівнях OSI (наприклад, обробка втрат пакетів і зміна порядку пакетів), уже реалізовані базовою мережевою структурою L2/3. У комерційних мережах для забезпечення цієї функціональності використовуються такі протоколи, як TCP або SCTP. У AFDX контроль і цілісність передачі вже забезпечені на нижніх рівнях, тому UDP був обраний протоколом за замовчуванням в AFDX .

Порти AFDX відображаються безпосередньо в полях порту джерела та порту призначення UDP. Потоки AFDX ідентифікуються за допомогою комбінації наступних параметрів:

- MAC-адреса призначення (містить VLID),
- IP-адреса джерела та призначення,
- вихідний і цільовий порт UDP,

Через архітектурні обмеження мінімальний розмір корисного навантаження для пакетів, що передаються всередині пакета AFDX-L3, становить 144 біти. Якщо довжина пакета UDP падає нижче цієї межі, додавання додається в кінці пакета L4 .

Стандарт також визначає моніторинг, який має виконуватися через SNMP, і внутрішньокомпонентну передачу даних через TFTP. Корисне навантаження, що передається всередині L4-структури, зазвичай не має фіксованого заздалегідь визначеного значення, на відміну від попередніх стандартів. Однак ARINC 664 визначає ряд поширених структур даних, таких як формати чисел з плаваючою комою та логічні значення. Вони не мають прямого впливу на корисне навантаження мережі, але пропонують спільну основу для розробки програмного забезпечення

.

Висновки

ARINC 429 був розроблений в той час, коли використання взаємопов'язаних програмованих підсистем на борту літака було просто неможливим через такі аспекти, як розмір, споживання енергії, крихкість та вартість обладнання. 429 розглядає виключно передачу даних між системами на рівні кожного пристрою, з'єднуючи системи на рівні контактів. Хоча він має переваги перед більш сучасними стандартами, він явно досяг своїх меж, коли багатоцільові комп’ютери з’єдналися між собою. Однак 429, швидше за все, не просто зникне; він все одно використовуватиметься в сценаріях, де достатньо простої сигналізації, а також у сценаріях із критичною затримкою. Це перевірена і надзвичайно надійна технологія, тому вона також використовується як резервна мережа для мережі AFDX, наприклад, в Airbus A380.

Більшість обмежень 429 були визначені десятиліття тому, але жодного єдиного стандарту не було прийнято в аерокосмічній промисловості. Інші стандарти представили більш сучасні, швидші або гнучкіші моделі мережі, на відміну від базової сигналізації, як у 429. Однак відомі спроби або використовуються виключно в окремих моделях літаків, або застосовуються лише внутрішньо виробниками (629, ASCB, CSCB, EFAbus) пропонуючи невелику сумісність з іншими реалізаціями. Тим не менш, ці стандарти запровадили підходи, які можна знайти у зміненому вигляді в AFDX.

AFDX поєднує перевірені функції безпеки та доступності з сучасними технологіями, щоб відповідати сучасним вимогам. Він дотримується моделі рівня OSI і окреслює сумісну архітектуру стека, дозволяючи імітувати попередні стандарти зв’язку зверху. Крім того, використовується набір протоколів Інтернету (IP/UDP) і Ethernet і вносяться лише незначні зміни до окремих структур даних, що значно знижує планку для проектування апаратного забезпечення та розробки програмного забезпечення в авіоніці.

Для певних частин мережі AFDX апаратне забезпечення COTS можна використовувати разом із відповідним програмним забезпеченням, хоча для збереження детермінізму необхідно використовувати апаратні реалізації AFDX. Проте, додавши стандартне обладнання Ethernet у поєднанні з реалізацією стека AFDX в операційній системі, апаратне забезпечення, яке не є AFDX, можна було б використовувати без додаткових змін.

Зміни в загальному макеті мережі не впливають негативно на окремі віртуальні посилання або порти окремих кінцевих і підсистем через додану абстракцію [12,18]. Діагностика проблем у мережі AFDX вимагає висококваліфікованого персоналу, на відміну від 429. Тим не менш, заміну обладнання в мережі 664 можна виконати з невеликими зусиллями, тоді як виправлення лінії, що проходить через весь літак через несправність, може вимагати значних зусиль . Пристрої, що підтримують ARINC 664, також можуть підтримувати віртуалізацію та апаратне розбиття, оскільки можна використовувати віртуальні/логічні пристрої, як зазначено в IMA/ARINC 653 . 429 ніколи не зможе підтримувати архітектури IMA, не кажучи вже про нефізичне обладнання.

Реалізації AFDX все ще залишаються відносно консервативними, використовуючи перевірену та зрілу технологічну базу замість найсучаснішого обладнання . Наприклад, мережі Airbus A380 і A350 засновані на мідних кабелях, тоді як оптичні кабелі стали де-факто стандартом високошвидкісного з’єднання на магістралях в корпоративних і операторських магістралях. Однак архітектури ARINC 664 можуть легко інтегрувати технології майбутнього. Майбутні реалізації AFDX, подібні до того, що використовується в Boeing 787, будуть використовувати волоконну оптику.

Гнучкість 664 у складних налаштуваннях робить його очевидним вирішенням проблем із накладними кабелями та складністю 429. Пропонуючи порівнянний рівень надійності, як Mil-Std-664 у цивільному застосуванні, його архітектура є значно більш адаптивною та може безперешкодно отримати вигоду від паралельна еволюція сімейства стандартів Ethernet. Крім того, швидкості лінії, які раніше були недосяжні в комерційній авіоніці, тепер доступні і вимагають набагато менше

зусиль для розробки як апаратного, так і програмного забезпечення, оскільки технологічною базою все ще залишається Ethernet.

У довгостроковій перспективі технологічне багатошаровість зведе до мінімуму необхідність перегляду AFDX для включення нових функцій, оскільки вони можуть бути введені через базові стандарти. AFDX є набагато більш досконалим, сучасним і потужнішим, ніж попередні стандарти, але зберігає свою гнучкість. У майбутньому ми, швидше за все, спостерігатимемо прискорення впровадження цього нового мережевого стандарту в комерційну авіоніку.

Розділ 4. Роль ARINC 653 в інтегрованій модульній авіоніці

Вступ

Інтегрована модульна авіоніка (IMA) вже досягла повноліття, встановлюючись практично в кожній новій моделі літака, що надходить на озброєння сьогодні. Ця концепція упаковки авіоніки високої щільності, яка вперше була представлена авіакомпаніям на Boeing 777, швидко отримала визнання як у військових, так і в комерційних літаках, від Lockheed C130 AMP до Airbus A380 і Boeing 787, наприклад.

Зараз це найочевидніше навіть для найбільш випадкових спостерігачів; IMA має сенс з точки зору бізнесу. Зменшення ваги та об’єму авіоніки, пов’язане з IMA, може вплинути безпосередньо на здатність операторів повітряного транспорту перевозити корисне навантаження, що приносить прибуток. Цей дохід може призвести до прибутковості авіакомпаній, уможливити придбання нових літаків і забезпечити стабільну торгівлю, необхідну для зростання галузі. IMA продемонструвала велику корисність в установках повітряного транспорту, і у неї попереду довге та продуктивне майбутнє.

3.1. Зародження ARINC 653

У середині 1980-х років промисловість визнала, що всі цифрові системи авіоніки мають спільні компоненти; серед них центральний процесор, пам'ять і введення-виводу. Операційна система реального часу (RTOS) також була визначена як компонент, який заслуговує на особливу увагу. Виявилося, що ОСРВ розробляється спеціально для кожної цифрової системи авіоніки. Цей підхід був і дорогим, і непотрібним. Завдяки цьому визнанню фокус стандартизації перемістився з апаратного забезпечення на програмне забезпечення. Хоча галузь неохоче розробляла стандарти обладнання для IMA, вона всією душею прийняла стандарти програмного забезпечення для IMA, зокрема: ARINC 653: Стандартний інтерфейс програмного забезпечення авіоніки.

ARINC 653 вперше був прийнятий у 1997 році Комітетом електронної техніки авіакомпаній (AEEC). Прийняття ARINC 653 як Airbus, так і Boeing для їхніх останніх літаків зміцнило роль ARINC 653 як промислового стандарту. Рівень сприйняття галуззю в результаті розширився і тепер включає весь ланцюжок поставок; від постачальників систем IMA, до постачальників різноманітних модулів і компонентів, програмних функціональних додатків та багатьох інших, включаючи постачальника ОСРВ.

Специфікація ARINC 653 складається з чотирьох частин:

• Специфікація ARINC 653: Частина 1, Необхідні функції

• Специфікація ARINC 653: Частина 2, Розширені функції

• Специфікація ARINC 653: Частина 3, Специфікація перевірки відповідності

• Специфікація ARINC 653: Частина 4, Піднабір функцій

Оскільки в цій статті висвітлюється роль стандарту ARINC 653 в системах IMA, у читача може виникнути спокуса запитати: навіщо розробляти стандартний інтерфейс операційної системи реального часу? Відповідь полягає в баченні, технічній концепції та перевагах, які очікується отримати спільнотою користувачів.

Стандартизований інтерфейс ОСРВ забезпечує дві ключові переваги. По-перше, він встановлює відомі межі інтерфейсу для розробки програмного забезпечення авіоніки, таким чином дозволяючи незалежність програмного забезпечення авіоніки та основного програмного забезпечення. Це дозволяє паралельно розробляти компоненти прикладного програмного забезпечення та ОСРВ. Ті самі програми можна зробити портативними серед сумісних платформ ARINC 653.

По-друге, стандартне визначення інтерфейсу ОСРВ дозволяє базовій апаратній платформі та основному програмному забезпеченню розвиватися незалежно від програмних додатків. Разом цей підхід до розробки систем IMA забезпечує рентабельний шлях модернізації протягом усього терміну служби літака.

Індустрія повітряного транспорту розробила ARINC 653 як стандартизоване визначення інтерфейсу RTOS. Він вказується як межа інтерфейсу між програмним забезпеченням авіоніки та основним програмним забезпеченням ядра, включаючи власне ОСРВ. Зусилля зі стандартизації були спонсоровані спільнотою користувачів авіакомпаній через ARINC і залучили багато зацікавлених сторін. Фахівці з програмного забезпечення зібралися та визначили конкретні потреби в обладнанні авіоніки:

• Критично важливий для безпеки – відповідно до частини FAR

25.1309

• У режимі реального часу – відповіді мають бути в межах встановленого проміжку часу

• Детермінований – результати передбачувані та повторювані

ARINC 653 — єдине визначення інтерфейсу ОСРВ, яке підтримує ці потреби. Крім того, програмне забезпечення авіоніки кваліфіковане на відповідний рівень RTCA DO-178 / EUROCAE ED-12. Цей процес вимагає суворості та уваги до деталей. Необхідно продемонструвати, що програмне забезпечення відповідає відповідним державним стандартам, встановленим для забезпечення безпечної роботи. Таким

чином, ОСРВ розроблено так, щоб бути простим і детермінованим у своїх робочих станах.

Архітектура IMA сприяє інтеграції багатьох програмних функцій. Механізми надаються ОСРВ для управління потоком зв’язку між програмними додатками та даними, з якими вони працюють. Підпрограми керування виконуються в результаті подій системного рівня. Це, в свою чергу, впливає на роботу літака.

Функціональність програмного забезпечення в рамках IMA забезпечується програмними додатками, які покладаються на обчислювальні ресурси, надані платформою авіоніки. Використання прикладного програмного забезпечення забезпечує більшу гнучкість для задоволення потреб користувачів і клієнтів, включаючи функціональне покращення. Оскільки розмір і складність вбудованих програмних систем збільшуються,

сучасні методи розробки програмного забезпечення (такі як об’єктно-орієнтований аналіз і проектування, функціональна декомпозиція за допомогою структурованого дизайну, проектування зверху вниз, проектування знизу вгору тощо) можуть полегшити процес розробки програмного забезпечення та підвищити продуктивність. Серед багатьох переваг програмне забезпечення можна визначити, розробляти, керувати та підтримувати як модульні компоненти.

Модель планування ARINC 653 можна описати як дворівневу модель, при цьому верхній рівень використовує фіксоване тимчасове планування програмних розділів. Програми, що працюють у цих розділах, мають повний доступ до обчислювальних ресурсів IMA. На нижньому рівні кожен розділ може використовувати переривання та інші асинхронні події, щоб забезпечити виконання вимог реального часу в межах його розділу. RTOS розподіляє ресурси та застосовує всі правила розподілу.

3.2. Загальна архітектура системи

Специфікація програмного інтерфейсу ARINC 653 була розроблена в першу чергу для використання з системами IMA. Однак ці концепції також підходять для традиційного об’єднаного обладнання, яке може містити більше однієї розділених функцій.

Приклад архітектури програмного забезпечення показано на малюнку 1. Основними компонентами системи є: прикладне програмне забезпечення, яке виконує функції авіоніки, і основне програмне забезпечення, яке забезпечує стандартне та загальне середовище для програмних додатків. Основне програмне забезпечення включає:

• Операційна система реального часу (RTOS) -

Керує логічними відповідями на запити програм. Він розподіляє час обробки, канали зв'язку та ресурс пам'яті.

• Heath Monitor (HM) — ініціює стратегії відновлення помилок або реконфігурації, які виконують певну дію відновлення.

Система апаратного інтерфейсу (HIS) — керує фізичними апаратними ресурсами від імені ОСРВ.

Розділ подібний до розділу багатозадачної програми в комп’ютері загального призначення. Кожен розділ складається з одного або кількох одночасно виконуваних процесів, які спільно використовують доступ до ресурсів процесора відповідно до вимог програми. Усі процеси однозначно ідентифікуються, мають атрибути, які впливають на планування, синхронізацію та загальне виконання.

ОСРВ відповідає за розподіл часу процесора та областей пам'яті для кожної програми. Це конфігуровані елементи в ОСРВ, якими керує системний інтегратор.

Щоб забезпечити мобільність прикладного програмного забезпечення, зв’язок між розділами не залежить від розташування вихідного та цільового розділу. Програма, яка надсилає повідомлення або отримує повідомлення від іншої програми, не міститиме чіткої інформації щодо розташування свого власного хост-розділу або його партнера по комунікації. Інформація, необхідна для правильного

маршрутизації повідомлень від джерела до призначення, міститься в таблицях конфігурації, які розробляються та обслуговуються системним інтегратором, а не окремим розробником програми. Системний інтегратор налаштовує середовище для забезпечення правильної маршрутизації повідомлень між розділами, розміщеними на платформі IMA.

3.3. Інтерфейс додатка/виконавця (APEX).

Основною метою інтерфейсу APEX є забезпечення інтерфейсу загального призначення між прикладним програмним забезпеченням і власне ОСРВ. Стандартизація цієї межі інтерфейсу (також званого інтерфейсом RTOS) дозволяє переносити додатки між платформами, що сумісні з ARINC 653, і закуповувати апаратне та програмне забезпечення від широкого кола постачальників. Це сприяє конкуренції, зменшує витрати на розробку та зменшує загальну вартість володіння. Основними характеристиками інтерфейсу ARINC 653 RTOS є:

1. Інтерфейс RTOS надає мінімальну кількість послуг, що відповідає потребі виконання вимог IMA. Очікується, що інтерфейс полегшить розробку програми та допомога розробникам програмного забезпечення у виробництві надійних продуктів.

2. Інтерфейс RTOS можна розширити для врахування майбутніх удосконалень системи. Інтерфейс RTOS зберігає сумісність з попередніми версіями програмного забезпечення.

3. Інтерфейс RTOS задовольняє загальні вимоги реального часу для мов програмування Ada і C. Ці різні моделі повинні використовувати базові послуги, що надаються інтерфейсом ОСРВ, відповідно до їх рівня критичності сертифікації та перевірки.

4. Інтерфейс RTOS відокремлює прикладне програмне забезпечення від фактичної архітектури процесора. Таким чином, вплив апаратних змін на програмне

забезпечення можна мінімізувати. Інтерфейс RTOS дозволяє прикладному програмному забезпеченню отримати доступ до виконавчих служб, але ізолює прикладне програмне забезпечення від архітектурної залежності.

5. Специфікація інтерфейсу RTOS не залежить від мови. Це є необхідною умовою для підтримки прикладного програмного забезпечення, написаного різними мовами високого рівня, або прикладного програмного забезпечення, написаного однією мовою, але з використанням різних компіляторів. Виконання цієї вимоги забезпечує гнучкість у виборі компіляторів, засобів розробки та платформ розробки. Інтерфейс ОСРВ ставить вимоги до розшарування та розподілу на власне ОСРВ, щоб забезпечити зростання та функціональне покращення.

Запити на зв'язок можна зробити в одному з кількох режимів. Додаток може попросити, щоб функція була виконана та призупинена, очікуючи завершення цього запиту, або продовжити роботу та або опитувати статус запиту, або просто отримати сповіщення про завершення транзакції.

3.4. Внутрішня структура ARINC 653 RTOS

Основна роль ОСРВ полягає в забезпеченні функціональної цілісності при плануванні та диспетчеризації прикладних програм. Функцією ОСРВ є виділення часу обробки кожному розділу, відправлення кожного процесу на виконання, надання необхідної пам’яті та ресурсів вводу-виводу та абсолютного забезпечення ізоляції розділу в часі та просторі (пам’ять). Повинно бути можливим довести, що рівень тимчасового детермінізму (тобто специфічна поведінка в певний час) не залежить від додавання інших додатків до платформи IMA. Слід зазначити, що розділення не може бути забезпечено лише ОСРВ. Швидше, це комбінована функція ОСРВ і апаратного забезпечення управління пам'яттю.

Одним із методів досягнення цього є застосування методу розподілу часу та розділення додатків на суворий темп або заплановані групи. Кожній програмі строгого темпу забезпечується певна кількість часу обробки в кожному часовому

зрізі, щоб вона могла виконувати певну кількість визначених алгоритмів у кожному виділеному часовому інтервалі. Якщо програма намагається перевищити свій часовий проміжок, її тайм-аут зупиняється ОСРВ. Планування забезпечує достатню кількість часу для кожного проміжку часу для ефективної роботи.

RTOS здатний розпізнавати як періодичні, так і аперіодичні процеси, а також планувати та диспетчеризувати всі процеси. Він надає інформацію про стан здоров’я та дані про помилки для кожного розділу. Оскільки ОСРВ має працювати з високим ступенем цілісності для критичних додатків, вона матиме критерії сертифікації, які відповідають набору функцій. Отже, дизайн ОСРВ має бути максимально простим.

RTOS також керує розподілом логічних і фізичних ресурсів від імені програм і гарантує, що ці виділені ресурси ізольовані один від одного. Він відповідає за управління пам'яттю та комунікаціями. Він отримує переривання, пов’язані зі збоєм живлення та апаратною помилкою, передаючи ці інциденти функції Health Monitor, яка, в свою чергу, спрямовує необхідні дії для відновлення або іншим чином. Він також повинен направляти програмні переривання або події, що стосуються програми, у відповідну програму у визначеному часовому масштабі. Як менеджер усіх фізичних ресурсів у цьому середовищі, ОСРВ відстежує запити на ресурси та контролює доступ до тих ресурсів, які не можуть використовуватися одночасно більш ніж однією програмою. Він має доступ до всієї пам’яті, переривань і апаратних ресурсів.

RTOS здатний повідомляти про помилки та виконувати наступні дії. У своєму розподілі та управлінні запитами ресурсів і комунікаційними інтерфейсами від програм і до них він має обмежений доступ до пам’яті програм. Оскільки моніторинг здоров’я відіграє особливу роль в IMA, він розглядається як окрема тема в цій статті. Зокрема, функції Health Monitor мають бути призначені як для прикладного програмного забезпечення, так і для ОСРВ.

Обробка запиту зв'язку в ОСРВ складається з налаштування відповідного формату введення-виводу та передачі повідомлення апаратному інтерфейсу. Щоб

забезпечити тимчасовий детермінізм, кожне визначення повідомлення включає максимальний і мінімальний час відповіді. Деякі типи повідомлень також містять специфікацію тайм-ауту, яка може бути встановлена програмою. Отже, всі комунікації, включаючи запити, команди, відповіді, введення-виведення даних тощо, між додатками та ОСРВ строго визначаються ARINC 653.

Частина 1 ARINC 653 розглядає ці аспекти ОСРВ:

•Управління розділами

•Управління процесами

•Управління часом

•Управління пам'яттю

•Міжпартійний зв'язок

•Внутрішньорозділове спілкування

•Монітор здоров'я

З огляду на точне визначення ОСРВ, можна інтегрувати та тестувати програми на комп’ютері загального призначення, який моделює інтерфейс ОСРВ. Також можна розміщувати різноманітні програми без необхідності змінювати ОСРВ. Стандартизовані визначення повідомлень RTOS дозволяють розвивати шлях для майбутнього вдосконалення інтерфейсу. Поки будь-яке майбутнє визначення ОСРВ є надмножиною попередніх визначень, тоді ніяких конфліктів не виникне в результаті будь-якого вдосконалення.

3.5. Функція Health Monitor

Функція Health Monitor знаходиться разом із RTOS і має інтерфейс до таблиці стратегії відновлення, визначеної конструктором літака або системним інтегратором. Монітор працездатності відповідає за моніторинг апаратних збоїв у платформі IMA та збоїв у ОСРВ. Помилки обладнання, виявлені за допомогою

вбудованого тесту (BIT), включають несправності пам’яті та процесора. Будь-яка локальна реконфігурація апаратного забезпечення є специфічною для апаратної реалізації і відбувається в системі апаратного інтерфейсу. Це прозоро для решти системи IMA.

RTOS відстежує апаратне забезпечення, відповідальне за цілісність програмних розділів, і взаємодіє з функцією Health Monitor у разі збоїв у програмному та апаратному забезпеченні цілісності. Функція моніторингу працездатності є специфічною для платформи IMA та для конкретної установки на літаку. Таким чином, це програмне забезпечення поділено на RTOS Health Monitor і таблицю стратегій відновлення. Останнє вимагає визначення конфігурації та вбудованих в неї стратегій відновлення. RTOS містить можливість звітування про помилки, до якої можуть отримати доступ програми. Якщо програма виявляє помилку в роботі, вона може повідомити про це ОСРВ, яка, у свою чергу, викликає функцію моніторингу працездатності. Роль програми — запитувати стан здоров’я та будь-яку переконфігурацію, яка могла бути виконана. RTOS пов’язана зі специфікацією інтерфейсу RTOS для забезпечення стандартного інтерфейсу прикладних програм (API) для всіх типів прикладного програмного забезпечення.

Несправності виявляються на різних рівнях. Мета полягає в тому, щоб стримувати помилки до того, як вони поширюються через кордон інтерфейсу. На додаток до методів самоконтролю, RTOS повідомляє про порушення додатків, збої зв’язку та несправності, виявлені програмами. Таблиця відновлення несправностей використовується для визначення дії, яку необхідно виконати у відповідь на конкретну несправність. Ця дія ініційована Health Monitor і може включати припинення дії програми та запуск альтернативної програми разом із відповідним рівнем звітності. Дія відновлення значною мірою залежить від конструкції системи IMA.

Можливі два типи таблиць відновлення Health Monitor. На рівні розділу надається одна таблиця. Інша таблиця надається на системному рівні для інтеграції кількох

розділів модуля. Для помилок рівня розділу таблиця моніторингу стану локального розділу:

•Визначте рівень помилки (розділ або процес)

•Визначте дію на рівні розділу

•Визначте код помилки у випадку рівня помилки процесу

Прийнятні методи моніторингу здоров’я розвиваються в промисловості, і це буде відображено в майбутніх оновленнях ARINC 653. Очікується, що вони з’являться у 2009–2010 роках.

3.6. Відповідне програмне забезпечення

Прикладне програмне забезпечення розробляється у вигляді модулів, які виконують певну функцію на літаку. Сьогодні такі функції, як керування польотами та керування зв'язком даних, можна реалізувати в програмному забезпеченні, використовуючи загальні обчислювальні ресурси IMA. Програмні додатки специфікуються, розробляються та перевіряються до рівня критичності, відповідного його функції. У платформі IMA кожен модуль прикладного програмного забезпечення працює незалежно, використовуючи ресурси, надані платформою IMA.

Прикладне програмне забезпечення відповідає за керування резервуванням для певної функції, а також за вибір сигналу та моніторинг несправностей вхідних даних із зовнішніх датчиків або інших систем. Модульний дизайн програмного забезпечення дозволяє реалізувати програмні розділи для ізолювання функцій авіоніки в загальному апаратному середовищі. Прикладне програмне забезпечення не залежить від апаратного забезпечення, хоча для деяких програм знадобляться спеціальні датчики введення-виводу, наприклад, статичний датчик Піто.

Програмні модулі можуть бути розроблені незалежними джерелами та інтегровані в платформу IMA. Таким чином, необхідно, щоб програмне забезпечення всередині

розділу було повністю ізольовано від інших розділів, щоб один розділ не міг спричинити несприятливий вплив на інший розділ. Для забезпечення цілісності розділу зображення завантаження розділу статично і окремо будуються. Ці програми є самостійними як незалежні програмні модулі без взаємозалежності з іншими програмними модулями. Для цих програм можуть знадобитися спеціальні елементи керування та дисплей, обробка вводу/виводу та пов’язане обладнання.

Атрибути планування процесу використовуються кодом розділу для зміни виконання або стану процесу в цьому розділі. Таким чином, цілісність однієї програми не порушується поведінкою іншої програми, незалежно від того, чи вважається інша програма більш чи менш критичною. Усі комунікації між програмами здійснюються через ОСРВ, механізми якого забезпечують відсутність порушення інтерфейсу, і жодна програма не монополізує ресурс або не залишає інший ресурс назавжди призупиненим.

Обробка вводу-виводу є однією з важливих областей, яка в традиційних додатках дуже специфічна для конфігурації датчиків літака. В інтересах переносимості та повторного використання, розділення архітектури програмного забезпечення програмного забезпечення має ідентифікувати спеціальне програмне забезпечення введення/виводу для літака та розділити його з функціональних та алгоритмічних елементів програми. Таке кондиціонування введення-виведення датчика логічно визначається як окрема функція, пов'язана з датчиком.

Більшість програм вимагають функціональних даних із певною швидкістю. Велика частина дизайну додатків, яка пов’язує їх конкретно з конкретною системою літака, — це керування датчиками. Видалення цієї функції з програми збільшує портативність і, крім того, зосереджує роботу з датчиком в одній області, дозволяючи даним датчиків із певними характеристиками бути загальнодоступними для більш ніж однієї програми. Зміни в характеристиках датчиків обмежуються менеджерами даних датчиків, тим самим збільшуючи портативність і повторне використання програмного забезпечення. Прикладне програмне забезпечення викликається компонентом планування ОСРВ детермінованим способом.

3.7. Поточний статус ARINC 653

ARINC 653 розвивається, щоб задовольнити потреби промисловості. Частина 1 оновлюється, щоб узгодити термінологію з термінологією, що використовується в RTCA DO-297/EUROCAE ED-124: Інструкції з розробки інтегрованої модульної авіоніки (IMA) і міркування щодо сертифікації)!] . Очікується, що ці зміни ще більше додадуть загального розуміння

Принципи ARINC 653, у тому числі більш широке визнання серед регуляторного співтовариства. Визначені терміни:

Основний модуль: апаратний компонент, який містить апаратні ресурси (основний процесор і пристрої).

Основне програмне забезпечення: операційна система та допоміжне програмне забезпечення, яке керує ресурсами платформи для забезпечення середовища, в якому програма може виконуватися.

У ARINC 653 ці терміни розробляються, щоб пояснити, що кожен модуль ядра може містити один або кілька окремих процесорів. Архітектура основного модуля впливає на реалізацію RTOS, але не на інтерфейс APEX, який використовується програмним забезпеченням кожного розділу. Процеси розділу не можуть бути розподілені між кількома процесорами ні в одному базовому модулі, ні в різних модулях ядра. Прикладне програмне забезпечення має бути переносимим між основними модулями та між окремими процесорами основного модуля без зміни його інтерфейсу з ОСРВ.

3.8. ARINC 653 Додаткові можливості

ARINC 653 Частина 2 визначає розширені послуги, які вважаються бажаними, але їх не потрібно використовувати в реалізації ARINC 653. Частина 2 описує програмні

послуги, які вважаються цінними для прикладних програм після багаторічного досвіду роботи з системами IMA.

Розширені послуги ARINC 653 включають:

•Файлова система

•Вибірка структур даних порту

•Кілька розкладів модулів

•Бортові журнали

•Розширення порту вибірки

•Точки доступу до послуг

Файлова система ARINC 653 визначена в Частині 2. Файлова система — це загальний спосіб керування зберіганням даних. Подібно до файлової системи в настільному середовищі, файлова система приховує й керує деталями різних форм зберігання даних на модулі. Зокрема, файлова система — це набір служб, які надають можливість відкривати, закривати, читати, записувати та видаляти файли та каталоги. Кожна реалізація надає унікальний набір одного або кількох типів носіїв даних, таких як RAM, Flash, EPROM або мережевий носій. Файлова система дозволяє кільком розділам використовувати служби для доступу до носія даних. Файлова система керує низькорівневими деталями носія для розділів, які її використовують.

Багато програм авіоніки використовують файлову систему для організації постійних даних для зберігання та отримання з локальних або мережевих комп’ютерів або для тимчасового зберігання даних у енергонезалежній пам’яті. Наприклад, системі керування польотами (FMS) потрібна файлова система для пошуку інформації про маршрут під час польоту. Системі обізнаності та попередження про місцевість (TAWS) потрібна файлова система для доступу до цифрових даних про висоту місцевості та перешкод зі своїх баз даних.

Структури даних порту вибірки визначені в частині 2. Це надає стандартизований набір структур даних для обміну параметричними даними. У це визначення входить стандартизований метод передачі статусу даних. Мета полягає в тому, щоб зменшити непотрібну мінливість у форматах параметрів, таким чином зменшуючи потребу в користувацькій обробці вводу-виводу. При правильному використанні це підвищить переносимість програм і покращить ефективність основного програмного забезпечення. Це аналогічно типовим шинам авіаційних даних (наприклад, ARINC 429, ARINC 629, ARINC 664 P7 тощо), де формат корисного навантаження повідомлення відповідає набору стандартних правил індикації типу та статусу. Оскільки комунікація між розділами має відповідати багатьом з тих же цілей, що й передача по шині даних, має сенс мати для неї подібний набір правил. Ця можливість полегшує передачу повідомлень між розділами ARINC 653 через шини літака та мережі літаків.

ARINC 653 Частина 2 також підтримує декілька розкладів IMA. ARINC 653 представив кілька розкладів IMA, щоб дозволити системам IMA планувати різні функції в різний час. Нижче наведено приклади того, де є корисними розклади кількох модулів:

1. Ініціалізація/розробка програми – кожному розділу потрібен певний час для ініціалізації змінних, даних, структур та пов’язаного обладнання.

2. Збій компонентів. Програми прикладних програм, критичні для безперервного польоту та льотної придатності, можуть бути призначені для виконання на іншому процесорі, замінюючи менш критичну програму, коли основний процесор виходить з ладу.

Як приклад кількох розкладів, функція завантаження даних знаходиться в одному розділі, тому вона отримує певний розподіл часу та простору. У режимі завантаження даних функція завантаження даних отримує всю пропускну здатність центрального процесора для максимальної швидкості та ефективності. Але після завершення завантаження даних вбудовані механізми дозволяють програмному

забезпеченню перейти в нормальний робочий режим польоту, при цьому завантаження даних деактивовано. Літак, який використовує кілька розкладів, повинен буде сертифікувати кожен режим і переходи між режимами.

Журнали визначені в ARINC 653 Частина 2 як засіб, що використовується для зберігання повідомлень. Журнал зберігає збережені дані у разі відключення електроенергії. Дані можна відновити після відновлення живлення модуля. Кожен журнал доступний лише з одного розділу. Журнал складається з буфера в енергонезалежній пам'яті (NVM). Повідомлення, зареєстроване в журналі, спочатку зберігається в буфері перед записом у пам'ять. Буфер дозволяє програмі швидко записувати кілька повідомлень у журнал без необхідності чекати часу, необхідного для запису NVM між кожним повідомленням.

Основною метою цих послуг є забезпечення більшої гнучкості додатку під час читання повідомлень порту вибірки.

Точки доступу до послуг (SAP) визначені в ARINC 653, частина 2. SAP — це особливий вид порту черги, який надає доступ до адресної інформації під час надсилання та отримання повідомлень. Служби SAP подібні до служб порту черги ARINC 653, але матимуть додаткові параметри для підтримки адресної інформації. Порти SAP зазвичай використовуються в програмах клієнт/сервер, де є кілька можливих адресатів, але для конкретного повідомлення необхідно вказати одного адресата. Оскільки це може змінюватися в реальному часі, бажано мати механізм, щоб програма мала певний контроль над адресацією.

Висновки

ARINC 653 є ключовим фактором у розробці інтегрованої модульної авіоніки (IMA). Прихильність, яку галузь продемонструвала перед IMA, не може бути більш очевидною, ніж та, яку демонструють Airbus A380 і комплекти авіоніки Boeing 787. Основне програмне забезпечення IMA може підтримувати декілька програм авіоніки; його можна розробляти одночасно і розміщувати на одній обчислювальній

платформі. Поява ARINC 653 створила конкурентне ділове середовище, де постачальники авіоніки можуть легко придбати RTOS, який підлягає «сертифікації» EUROCAE ED-12 і RTCA DO-178B до рівня A – найсуворішого рівня сертифікації програмного забезпечення. Таке бізнес-середовище гарантує, що ринок заповнюють тільки найкращі продукти.

На додаток до мінімізації прямих витрат на RTOS, ARINC 653 допомагає керувати вартістю неповторної інженерії (NRE) на розробку програмного забезпечення для авіоніки. Оскільки нові продукти RTOS, ймовірно, відповідатимуть як ARINC 653, так і RTCA DO-178B, середовище прийняття та розуміння розвивається. Розробники додатків, інженери програмного забезпечення, програмісти, регуляторні органи та громадськість в цілому отримають користь від IMA.

Разом принципи IMA дозволяють розробляти і затверджувати мільйони ліній програмного забезпечення для авіоніки в міру необхідності для задоволення експлуатаційних потреб авіакомпаній протягом усього терміну служби літака.

Розділ 4. Новий метод аналізу на основі моделі з кількома обмеженнями для інтегрованого процесу динамічної реконфігурації модульної авіоніки

Вступ

Система авіоніки розвивається від дискретної до федеральної та до інтегрованої модульної авіоніки (IMA). Система має більш відкриту та складнішу архітектуру. Система IMA виконує функції на основі загальних функціональних модулів (CFM). CFM допомагають зменшити вагу та розмір літака. У системі IMA різні програмні

функції виконуються на CFM. Програмна система дуже інтегрована через її складну структуру.

На основі опису IMA в ARINC 653 і Радою архітектури авіоніки Allied Standards (ASAAC) архітектура програмного забезпечення має тришарову структуру. Прикладне програмне забезпечення спочатку зберігається в пристрої зберігання даних, а не в CFM. Програмне та апаратне забезпечення не є обов’язковими. Програмне забезпечення можна використовувати на різному обладнанні з різними конфігураціями. З міркувань безпеки літаки зазвичай перезапускають програми за допомогою надлишкових резервних копій, коли відбувається якийсь збій. У цій статті динамічна реконфігурація відноситься до змін конфігурації, що проводяться, коли під час польоту відбувається збій. Динамічна реконфігурація може допомогти у створенні нових областей резервного копіювання для перезапуску програми, що робить літак більш гнучким і ефективніше використовує апаратні ресурси.

Є багато досліджень по динамічної реконфігурації з статичної точки зору. Дін, М. запропонував підхід до побудови автоматної моделі та верифікації діаграми діяльності мови моделювання системи (SysML), щоб забезпечити відповідність логічної архітектури переконфігурованої системи функціональним вимогам. Шукла, Дж. запропонував методологію реконфігурації системи розподілу з обмеженою стабільністю сигналу (DSR) в умовах невизначеності, пов'язаної з попитом на навантаження та вихідною потужністю розподіленої генерації на основі відновлюваних джерел енергії. Елліс, С.М. встановив доцільність апаратної відмовостійкості за допомогою динамічної реконфігурації програмного забезпечення та продемонстрував її життєздатність у контексті типового застосування авіоніки в реальному часі. Однак досліджень, присвячених процесу динамічної реконфігурації, мало. Слід враховувати правильність процесу динамічної реконфігурації. Розуміння обмежень, що стосуються динамічної реконфігурації, може допомогти у правильному та плавному завершенні процесу. Однак, як змоделювати та додати обмеження для аналізу динамічної реконфігурації є проблемою. Для аналізу процесу динамічної реконфігурації ми запропонували в

цьому дослідженні метод на основі моделі з кількома обмеженнями. Моделювання динамічної реконфігурації IMA може бути корисним для аналізу.

Система IMA є вбудованою системою реального часу. Мова аналізу та проектування архітектури (AADL) — це міжнародний стандарт SAE (раніше відомий як Товариство автомобільних інженерів) (SAE AS5506), заснований на інженерії, керованої моделлю (MDE). AADL використовує концепції моделювання для опису програмної/апаратної архітектури та середовища виконання з точки зору окремих компонентів та їх взаємодії, і це особливо ефективно для моделювання проектування складних вбудованих систем реального часу. Тому AADL широко застосовується у вбудованих системах, особливо в аерокосмічній, поданої. Zhang, F. застосував AADL до моделі F-16 «Auto Pilot Controller» і проаналізував властивості поведінки живості та уточнення сліду з різними припущеннями справедливості, враховуючи тимчасові можливості та терміни. Zhao, Z. побудував модель AADL для складної апаратної структури та надійного програмного забезпечення системи авіоніки. Liu, Z. представив метод моделювання розділення IMA на основі AADL з двох аспектів: моделювання архітектури та моделювання політики планування, а також проаналізував планування розподілу IMA.

AADL не тільки описує компоненти системи, але також описує поведінку системи та інші елементи, такі як режим і всі типи додатків. Режими можуть представляти різні стани конфігурації системи або компонента, коли подія ініціює зміну режиму. Усі ці особливості роблять AADL хорошим методом для опису процесу переходу системи, наприклад, динамічної реконфігурації IMA.

Однак AADL є лише напівформальною моделлю, і вона не зріла для аналізу надійності. Використання AADL для аналізу надійності вбудованих систем не є точним. Хоча AADL забезпечує ефективну підтримку для моделювання вбудованих систем, його необхідно формалізувати, щоб зробити модель зручною для формальної перевірки. Перетворення моделей є центральними в інженерії, керованої моделлю (MDE), де вони використовуються для перетворення моделей між різними мовами; для реорганізації та моделювання моделей або для генерування коду з

моделей. Тому науковці та представники промисловості, як правило, використовують методологію трансформації моделі для перевірки та аналізу моделі AADL за допомогою існуючих інструментів перевірки та аналізу. Було запропоновано багато досліджень щодо трансформації AADL: перетворення AADL у пріоритет взаємодії поведінки (BIP), у Fiacre, у мережі Петрі, в EDA (Event-Data Automata) тощо. Метою такого перекладу є повторне використання існуючих інструментів перевірки та аналізу та їх формальної моделі обчислень і зв'язку з метою перевірки моделей AADL.

Мережі Петрі — це формальний графічний і математичний інструмент, здатний моделювати та аналізувати динамічну поведінку систем. Вони також все частіше використовуються для системної безпеки, надійності та оцінки ризиків. Мережі Петрі були доведені як потужний інструмент моделювання та аналізу та використовуються в моделюванні та аналізі кооперативних систем і систем дискретних подій завдяки їх обґрунтованій формалізації. Потім мережа Петрі триває до кольорової мережі Петрі (ВПП), узагальнена стохастична мережа Петрі (GSPN), нечіткої мережі Петрі для підвищення його опису здатності, так що він може ефективніше моделювати динамічний процес. Тому мережі Петрі широко застосовуються в системі безпеки, надійності та оцінки ризиків у багатьох галузях. Лі та ін. запропонував метод моделювання надійності на основі PN, коли оцінювалася надійність системи, враховуючи залежність механізму відмови. Wieland та ін. запропонував модель на основі PN для розрахунку даних надійності пакетів полімер-електроліт-мембрана паливних елементів. Дані про надійність включають середній термін служби окремої стопки або надійність стопок цілого парку транспортних засобів на паливних елементах протягом певного часу. Сунанда та ін. запропонував підхід до моделювання розломів на основі мережі Петрі, і цей підхід був підтверджений шляхом застосування його до прототипу системи залізничного та автомобільного перетину. Лі, В. запропонував новий метод моделювання та обґрунтування багатошарових нечітких мереж Петрі для оцінки ризику відмови технологічного обладнання, щоб чітко описати зв'язок і зробити

обчислювальний процес гнучким. Gongalves, P. представив моделювання процесу оцінки безпеки БПЛА Петрі Нетсом для аналізу умов несправності, які призводять до найбільш страшних подій. P. Лю переклав початкову модель AADL GSPN для аналізу та оцінки надійності системної платформи IMA. Лі, З. запропонував аналіз небезпеки за допомогою покращеного синхронізованого CPN з обмеженням безпеки зв'язку час-простір для IMA.

Проте в аспекті моделювання вбудованих систем мережі Петрі страждають від комбінаторних і складних проблем і, отже, їх важко використовувати при моделюванні складних систем зі значною кількістю станів . Тому в цьому дослідженні ми застосували AADL для моделювання процесу динамічної реконфігурації IMA, а також для моделювання та аналізу моделі шляхом перетворення в кольорові мережі Петрі.

4.1. Основна інформація

Система IMA – це складна система, яка має більш відкриту архітектуру, більш широку інтеграцію, більш інтегровані функції та високу зв’язок між модулями. Багато проблем також виникають, коли відбувається реконфігурація. Динамічна реконфігурація в цьому дослідженні стосується програмного забезпечення. Потім у цій статті представлено архітектуру програмного забезпечення IMA. Система IMA включає основну систему IMA та неосновне обладнання відповідно до стандарту ASAAC. Основна система IMA містить кілька стійок авіоніки. Ці стійки містять CFM та комунікаційні мережі між ними. Крім того, стійки мають функціональні програми, засновані на апаратному забезпеченні, операційній системі та програмному забезпеченні керування системою.

CFM забезпечує обчислювальну потужність, підтримку мережі та перетворення енергії для основної системи IMA. Програмна система поділена на три рівні — рівень підтримки модулів (MSL), рівень операційної системи (OSL) і рівень прикладних програм (AL). MSL забезпечує інтерфейс для вищевказаного рівня для доступу до ресурсів і розділяє операційну систему та апаратну платформу. OSL

включає в себе операційну систему в режимі реального часу та управління системою. Програмне забезпечення та керування додатками здійснюються на AL. Загальне управління системою (GSM) в OSL, яке виконує керування системою, налаштовує та керує системою шляхом доступу до плану системи. GSM включає в себе управління працездатністю, керування несправністю, керування конфігурацією та керування безпекою. Керування додатками є частиною системного керування і керується на основі програми. Архітектура програмного забезпечення показана на малюнку 1.

Реконфігурації IMA включають як статичні, так і динамічні реконфігурації. Деякі механізми є загальними як для статичної, так і для динамічної реконфігурації. У системі ядра IMA всі програмні додатки зберігаються в масовій пам'яті. Коли система ініціалізована, програми завантажуються на цільовий модуль. Ця операція зменшує потребу в технічному обслуговуванні та гарантує можливість заміни модуля. Коли виникають деякі збої, менеджер працездатності виявляє збій і повідомляє менеджеру збоїв, щоб вирішити її. Менеджер несправностей може обробляти серію збоїв у порядку під усіма типами механізмів. З одного боку, менеджер збоїв може виявити дефекти, локалізувати та зв’язати збої, а потім повідомити про результати аналізу на верхній рівень. З іншого боку, файл systfm вимагає від конфігураційного менеджера почати реконфігурацію, щоб уникнути збою. У підсумку, менеджер помилок синтезує багато технологій управління несправностями. Потім диспетчер конфігурації починає свою роботу: після отримання повідомлення від диспетчера помилок.

У ASAAC є багато принципів реконфігурації. Систему IMA слід переналаштувати між стабільними станами. Реконфігурація повинна зупинити поширення несправності: якомога швидше. Проект системи повинен враховувати всю ситуацію, коли існує певна реконфігурація. Реконфігурація передбачає переміщення програми через певні вимоги або коли відбувається збій. Дії реконфігурації відповідають порядку з креслення. Рейс та ін. зазначив, що реконфігурація корисна, коли є

динамічні нефункціональні вимоги, дефекти обладнання або вимоги до програми для системи.

Однак між статичною та динамічною реконфігураціями є відмінності. Зазвичай система статичної реконфігурації включає резервні модулі. Наприклад, CRACK2 є надлишковим краком для CRACK1. Тоді CFM3/REP2 є резервною копією для CFM3/REPC. Коли CFM3/REP1 виходить з ладу, програми працюють на CFM3 і можуть бути передані в REP2 з REPt, щоб система не виходила з ладу. Статична реконфігурація системи представлена на малюнку 2.

На основі ARINC 653 помилки (помилки) в IMA та механізмах реагування класифікуються, як показано в таблиці 1.

Тому, коли в системах IMA виникають помилки, першими почнуть реагувати механізми реагування, наприклад, методи відмовостійкості тощо. Коли механізми реагування не в змозі вирішити проблемупомилка (помилка), помилка призведе до динамічної реконфігурації. У цьому дослідженні ми прагнемо обговорити та проаналізувати ситуації після початку процесу реконфігурації, тому ми припускаємо, що механізми реагування не можуть вирішити несправність, і що це викликало реконфігурацію.

Під час роботи системи відбувається динамічна реконфігурація IMA. У цьому дослідженні реконфігурація стосується лише програмного забезпечення, оскільки збій устаткування є необоротним. Динамічна реконфігурація IMA може змінити свої завдання відповідно до вимог і швидко відновитися після збою. Це робить систему більш гнучкою та зменшує резервування апаратного забезпечення та витрати на позапланове обслуговування. Більше того, коли задіяна людина, складність динамічної реконфігурації зростає. Тоді визначити, як стримати процес, щоб забезпечити його безпеку, стає проблемою. Багато дослідників досліджували вдосконалення динамічної реконфігурації за всіма типами аспектів.

Топінг, С. представив динамічно реконфігурований модуль обробки (DRPM) для динамічної реконфігурації. DRPM складається з перепрограмованих польових

програмованих вентильних матриць (FPGA), які є основним обладнанням, необхідним для зручної реконфігурації. Suo запропонував, що традиційні методи аналізу здебільшого зосереджуються на відмовах компонентів. STPA використовується для виконання аналізу небезпек, який фокусується на людських факторах, що лежать в основі динамічного процесу. Часові планувальники запропоновані для планування процесу динамічної реконфігурації за жорстких обмежень часу та ресурсів. Підготовка плану реконфігурації була складною для більшості дослідників у минулому через відсутність автоматизованих та інтелектуальних інструментів. Вони автоматизували динамічну реконфігурацію, використовуючи тимчасові планувальники штучного інтелекту (AI), щоб зменшити її складність. Планувальники штучного інтелекту проводять оптимізоване планування завдань, що ускладнює для людей визначення моменту переналаштування системи. Монтано визначив елементи динамічної реконфігурації у своїй дисертації. Динамічна реконфігурація критично важливих для безпеки пілотованих систем (SCMS) обумовлена подією для зміни функцій або ресурсів відповідно до вимог оператора. У дисертації обговорюється автоматизація та участь людини під час динамічної реконфігурації критично важливої для безпеки системи.

Для моделювання динамічної реконфігурації IMA Чжан запропонував метод надійності, заснований на AADL для реконфігурації IMA. Потім систему було переведено в мережу Петрі для аналізу надійності. Розрахунок надійності для компонентів архітектури системи. В іншому дослідженні Суо представив метод вирішення проблем у реальному часі при реконфігурації . Система IMA також моделюється за допомогою AADL. Потім система перекладається на TPN для перевірки.

Перш за все, аналіз процесу динамічної реконфігурації не привернув особливої уваги. У цьому дослідженні запропоновано метод аналізу процесу, заснований на моделях підвищення коректності, безпеки та надійності динамічної реконфігурації.

AADL є ефективним інструментом моделювання для аналізу вбудованих систем і складних систем реального часу. У цьому дослідженні AADL використовувався для моделювання процесу динамічної реконфігурації IMA.

Компоненти є основними елементами системи. Компоненти поділяються на три набори — програмне забезпечення, апаратні засоби та композитні. Стан конфігурації системи можна описати за допомогою AADL. Крім того, можна описати структуру системи та пристрої. Архітектура програмного забезпечення системи IMA розділена. Тоді для структури логічної конфігурації необхідний додаток ARINC 653 в AADL. Елементи ARINC 653 формували архітектуру системи та відповідають компонентам AADL.

Стабільний конфігураційний стан системи під час динамічної реконфігурації представлено режимом. Режими можуть представляти різні стани системи або компонента, зв’язки та асоціації значень властивостей. Переходи режимів визначають, коли система динамічно переналаштовується на нову конфігурацію. Текстове та графічне представлення специфікацій переходу режиму для простого; приклад показано на малюнку 3 .

Кожен стан системи та детальні переходи є; виражається додатком поведінки в AADL. Додаток поведінки визначає специфікації поведінки компонентів AADL у більш витонченому, ніж соае мови. Поведінки, описані в цьому додатку, засновані на змінних стану, еволюція яких визначається переходами, які можна охарактеризувати умовами та діями.

AADL широко використовується для моделювання вбудованої системи. AADL не може виконувати імітаційний аналіз динамічного процесу візуально. Мережа Петрі в сумісному інструменті для проведення імітаційного аналізу.

Мережа Петрі — це інструмент графічного та математичного моделювання для опису систем, які є постійно діючими та асинхронними. Petri Nett може не тільки моделювати динамічну діяльність або передачу інформації в мережі, але також використовувати математичні моделі для керування поведінкою систем. Мережа

Петрі описується як трикортежний PN = (P, T , F), де P — скінченна множина місць; T — скінченна множина переходів; F — множина спрямованих дуг. Класична мережа Петрі містить місце, переходи та дуги між місцями. Порушення системи зазвичай описується місцем. Переходи представляють процес зміни систем. Дуги від переходу до місця або від місця до переходу мають свої ваги. У кожній місцевості є маркери, які показують стан місця. Токени також використовуються для представлення даних або ресурсів. Проте є деякі недоліки класичної мережі Петрі. Класична мережа Петрі не має уявлення про час. Більше того, метод опису шин у класичних мережах Петрі є занадто єдиним. Таким чином, існує багато розширень і доповнень для мереж Петрі. Було запропоновано деякі високорівневі мережі Петрі, такі як CPN і синхронізовані мережі Phtri.

CPN — це високорівневі мережі Петрі, які використовуються для проектування, аналізу специфікацій, перевірки та перевірки. CPN — це кортеж CPN = (Σ, P, T, A, N, C, G, E, I), де: Σ — кінцева множина непорожніх типів, яку також називають наборами кольорів; P — скінченна множина місць; T — скінченна множина переходів; A — скінченна множина дуг; N — функція вузла; C — функція кольору; G — захисна функція; E — функція дугового виразу; I є функцією ініціалізації. CPN можуть описувати стани складних систем і зміни стану через події ініціювання. Особливістю CPN є те, що він надає визначення наборів кольорів. Набір кольорів, прикріплений до місця, містить маркери. Кожен жетон повинен мати колір. Охорона переходу повинна бути задоволена до того, як буде здійснено перехід. CPN поєднує мережі Петрі та стандарт мови програмування ML.

Нещодавно було проведено деякі дослідження щодо підходів до аналізу на основі моделі для динамічної реконфігурації. Метод надійності на основі AADL для реконфігурації IMA був запропонований Чжаном. Розрахунок надійності компонентів системи проводиться після переходу моделі від AADL до мережі Петрі. Suo представив метод, змодельований AADL, і переведений на TPN для вирішення проблем у реальному часі під час реконфігурації. Ван дер Алст зазначив, що

Мережі Петрі не тільки використовуються як мова проектування для специфікації складних робочих процесів, але й забезпечують потужні методи аналізу для перевірки правильності процедур робочого процесу. Наскільки нам відомо, жоден підхід до аналізу не зосереджується на процесі динамічної реконфігурації. У цьому дослідженні був запропонований метод аналізу на основі моделі процесу динамічної реконфігурації для проведення аналізу небезпек.

4.2. Обмеження для процесу динамічної реконфігурації

Тут був запропонований набір обмежень для процесу динамічної реконфігурації IMA. Обмеження охоплюють багато аспектів, таких як стан системи, можливості реального часу та ресурсні можливості. Усі зазначені обмеження були інтегровані в метод аналізу для перевірки правильності проектування динамічної реконфігурації IMA.

Перед запуском динамічної реконфігурації необхідно провести деякі попередні перевірки модулів у системі, крім модуля відмови. Якщо відбулося поширення несправності, слід відмовитися від початкової стратегії налаштування динамічної реконфігурації. Слід розглянути нову динамічну реконфігурацію. Булеве значення S встановлюється для представлення початкового стану системи. Якщо інші модулі системи також виходять з ладу після розповсюдження, то S = 0. Таким чином, реконфігурація припиняється. Якщо не відбувається поширення збою, система може почати переконфігурацію, S = 1.

Короткий вступ до процесу динамічної реконфігурації наведено у Розділі 2 . Під час процесу система переходить з одного стану в інший через ініційні події та дії. Це важлива проблема, де повинні бути забезпечені обмеження часу. Якщо час між двома станами дуже довгий, це впливає на наступний стан і спричиняє певні небезпеки переналаштування. Властивості аналізу слід додати в модель.

Властивість часу пов’язана з переходом з одного стану в наступний. Щоб гарантувати тимчасові обмеження, було запропоновано алгебраїчне рівняння, яке

порівнює суму часу, витраченого всіма підстанами під час процесу динамічної реконфігурації, із значенням обмеження.

На малюнку 4 показано, що в процесі є п'ять станів. Існує тригер для переходу між станом 0 і станом 1. Час переходу дорівнює T 1 . Існує дію часу витрат T 2 між станом 1 і станом 2. Це аналогічно для інших держав. Кожна дія або перехід між двома станами в процесі позначається витраченим часом. Якщо Граничний час дорівнює t. Тоді загальний час споживання T s є сумою fi. Весь час задовольняє рівняння (1).

Використовуючи це рівняння, можна порівняти різні значення часу e s; може дати нам результат, чи зможе система досягти обмеження в реальному часі go a l s .

Подібно до обмеження часу, всі операції nssd мають свій простір пам'яті, як показано на малюнку 5. Очевидно, що розмір пам'яті thn, виділений кожним станом під час динамічної реконфігурації, може бути змінений. Об’єм пам’яті шин має бути гарантований, незалежно від того, як змінюється вимоглива пам’ять підстанів.

Кожен стан, включаючи дії, займає пам'ять Mj. Максимальне обмеження розміру пам'яті становить m. Тоді слід дотримуватися рівняння (3).

$$M_j \leq m(i = 1, 2 \ldots, n)$$

У різних режимах системи компоненти системи взаємодіють з компонентами даних читання і запис. Спільне використання ресурсів, таких як дані, має бути позначено серійним номером, пов’язаним із часом після операції в різних станах. Якщо немає мітки на Componenta даних aftea зміни даних в стані 1, то система не може вирішити , чи є дані результатами система хоче в Stase 2 перевірки mfter на початковому O3 стані 2. atsence від операції в стані 1 може зупинити перехід системи в наступний стан. Це можна представити як малюнок 6.

Якщо позначка правильна, то можливість спільного використання ресурсу (даних) перевіряється належним чином. Припустимо, що компонент спільного доступу до даних позначений D_j у стані і після кожного стану. Тоді M (D_j) представляє

порядковий номер D_j. На початку наступного стану перевіряється D і компонента даних і значення дорівнює C (D_j + f).

Якщо рівняння (4) задовольняється, то виявляється, що ресурс даних працює правильно. Інакше щось не так або відсутнє в колишній державі.

4.3. Метод аналізу на основі моделі

Складний процес динамічної реконфігурації важко проаналізувати без моделювання. AADL є ефективним інструментом моделювання для вбудованої системи реального часу. Динамічна реконфігурація — це процес, який бере участь у роботі людини-оператора та автоматизації. Діапазон аналізу досить широкий. Однак у цьому дослідженні події та умови, які змінюють процес, спрощені як деякі тригери. Об’єкт – це лише сам процес. Таким чином, тут обговорюється детальна декомпозиція та формалізоване вираження спрощеного процесу.

Коли в модулі IMA виникає одна або кілька збоїв, менеджер працездатності виявляє збій і повідомляє диспетчеру збоїв, щоб він впорався з ним. Менеджер несправностей може обробляти серію збоїв у всіх типах механізмів. Потім диспетчер помилок визначає тип відмови, щоб вжити заходів для її вирішення, наприклад, закриття динамічної реконфігурації або звітування менеджеру верхнього рівня.

Якщо диспетчер несправностей не зможе вирішити проблему, він ініціює динамічну реконфігурацію. Коли обмеження не задовольняються в процесі динамічної реконфігурації IMA, процес зупиниться, і система вийде з ладу. Коли починається процес динамічної реконфігурації, система зупиняє програму з невдачею та створює резервну копію даних. Зв'язки зруйновані. Потім вибирається цільовий модуль реконфігурації, виходячи з функціональних і нефункціональних вимог, наприклад мінімізації вартості зв'язку. Наступним кроком є створення нового розділу на іншому модулі для програми. Згодом виконується перезавантаження програми та відновлення з’єднання. У цьому дослідженні процес динамічної реконфігурації IMA описується як послідовні блок-схеми, виходячи з припущення, що ймовірність

появи кожного кроку процесу реконфігурації становить 100%. Ми прагнемо змоделювати весь процес динамічної реконфігурації та проаналізувати, які кроки та незадоволені обмеження зупиняють процес реконфігурації.

Типовий процес динамічної реконфігурації представлений на малюнку 7. Стрілки вказують, що повідомлення надсилаються під час процесу. Прямокутники представляють важливі дії, які відбулися. У порівнянні з процесом реконфігурації, згаданим в іншому дослідженні, яке завжди має резервні модулі, динамічна реконфігурація, що обговорюється в цьому дослідженні, відноситься до системи без запасних модулів, особливо коли реконфігурація у разі резервування не розроблена або використовується в системі при динамічній реконфігурації. починається.

Наступна частина описує метод моделювання цього процесу, за яким слід інтерпретація методу аналізу на основі моделі з цими логічними обмеженнями.

AADL введено вище для опису системи IMA. Режим системи може бути пов'язаний з логічними конфігураціями. Переходи режимів означають, що стан конфігурації змінюється з одного на інший. Система або компонент мають різні статичні структури та властивості в різних режимах. Властивість може описувати планування завдань, характеристики реального часу, зв’язок, пам’ять тощо. Тоді режими на системному рівні представляють вміст конфігурації системи. Система має власні модулі, розділи, процесори та комунікаційну шину в кожному режимі. Таким чином, статична структура системи в одному режимі будується додатком ARINC 653 в AADL.

На малюнку 7 показано серію підстанів між двома режимами. Підстани та переходи між режимами можна описати в додатку поведінки. Початковий стан у додатку відповідає попередньому режиму, тоді як останній повний стан є останнім режимом. Інші підстани можуть описувати певний стан системи, коли перехід закінчується під час динамічної реконфігурації. Перехід і дії в додатку можуть описати перехід режимів. Поведінковий додаток може описувати припинення та перезапуск додатків, встановлення та знищення процесів та їх потоків, створення та видалення

інтерфейсів зв’язку, створення та розрив з’єднань передачі та віртуальних каналів, а також надсилання та отримання повідомлень з іншими компонентами GSM.

Додаток до моделі помилок представляє умови запуску в реконфігурації шин, викликані несправністю. Тип моделі помилки може оголошувати стани помилок, події помилок і поширення помилок. Реалізації помилок mydel оголошують переходи стану помилок. Переходи призначені для представлення помилок, які поширюються з компонента на основі поточного стану помилки цього компонента. Властивість помилки захисної події може вказувати, що певні шаблони станів помилок і їх поширення виявляються і викликають основну подію AADL, наприклад, ініціюючи перехід режиму ttansition.

Метод моделювання, запропонований у цьому дослідженні, представлений на малюнку 8. Зміна режиму означає, що відбулася динамічна реконфігурація. Більш детальна інформація та підстани між режимами описані за допомогою додатку поведінки. Умова тригера переходу режиму оголошується в додатку до моделі etror.

Властивості додаються до моделі, особливо до додатка поведінки для наступного аналізу. В якості основи аналізу з багатьма обмеженнями значення sgch як властивості часу, розмір пам'яті та стани даних є важливими елементами системи.

Модель AADL динамічної реконфігурації IMA ефективна для опису структури системи та складного процесу реконфігурації. Деякі аналізи можна провести в інструментах для AADL, таких як Open Source AADL Tool Environment (OSATE). Однак автоматичне моделювання та аналіз не є сильними сторонами AADL, але підходять для мережі Петрі. Між тим, у моделюванні вбудованих систем для мереж Петрі є й недоліки. У цьому дослідженні режими в моделі coufd AADL будуть представлені місцями в CPN. Активні режими в AADL можуть бути представлені за місцем із певним маркером кольору в CPN. Перехід режимів у моделі AADL може бути перетворений на перехід токенів у CPN. Часові властивості переходів станів у моделі AADL відповідають часовим штампам маркерів у дугах у CPN. Такі ресурси, як пам'ять і дані в моделі AADL, які спільно використовуються в системі, можуть

бути представлені маркерами в місці в CPN. Фінахір, обмеження пам’яті та часу в моделі AADL можна перетворити на захисні функції в мережі Петрі. Таким чином, модель AADL може бути переведена в CPN інтегрально, як показано на малюнку 9.

Щоб наочно провести демонстрацію нашої моделі CPN на основі методу аналізу, був використаний приклад CPN. Інтуїтивно, ось приклад простого CPN, показаного на малюнку 10. Інструменти CPN ara використані CO створюють мережі Peeri. Приклад неф Петрі має шість місць. Три O1 місця представляють стани системи — початок, A і B. Місце, відоме як збій, показує ініційну подію. Місце, позначене D, представляє компонент даних. Місце з назвою M представляє ресурс пам'яті. Дуга з'єднує місце і перехід.

Для представлення запису даних, розміру пам’яті та активності стану використовуються різні набори кольорів. Мітка часу використовується для представлення часу, витраченого на перехід.

Після моделювання за допомогою цієї мережі Петрі можна було б отримати кілька типів результатів на основі умов обмеження.

1. Якщо всі обмеження виконуються, сітка моделюється до останнього місця і зупиняється.
2. Необхідно перевірити стан системи для динамічної реконфігурації. Перехід T0 може бути запущений, тільки якщо виконується охоронна функція [f = f1 і f' = nof2]. Це означає, що подія збою сталася для запуску динамічної реконфігурації і не поширилася на інші модулі системи.
3. Слід визначити, чи задовольняються обмеження в режимі реального часу переходу стану системи чи ні. Кожен крок у процесі моделювання записується міткою часу в переході. Коли один крок завершено, витрачений час порівнюється з обмеженнями в реальному часі. Результат може сказати нам, чи дотримані обмеження на час повторення. У цьому обмеженні є недолік, оскільки моделювання має виконуватися вручну крок за кроком.

4. Обмеження пам'яті стану системи не відповідають вимогам. Охоронна функція T1 [y = Y fnd m < mem_size] встановлюється, щоб визначити, чи є розмір пам’яті, зайнятий у стані (набір кольорів M), менший за обмеження розміру пам’яті. У цій мережі обмеження розміру пам’яті становить 10 Мб, тоді як
5. Обмеження можливостей для спільного використання ресурсів даних виконано. Якщо маркер з позиції A до переходу W1 не відповідає захисній тункції, моделювання припиняється. У цій мережі Y у наборі кольорів W надсилається до W1. Функція [y = Y] виконана, і моделювання шин продовжується. Це означає, що докомпонента даних (місце D) додається позначка затребуваного. З цією міткою можна ініціювати наступний стан.

Висновки

Існуючі дослідження динамічної реконфігурації рідко зосереджувалися на аналізі небезпек процесу. У цьому дослідженні, нова модель на основі методу аналізу з допомогою кількох Constrain т з для динамічного reconf IMO я був запропонований процес конфігурації, який є основним внеском даного дослідження. Т він ана лізису метод проводиться в три етапи-моделювання, моделі переходу і моделювання. Для моделювання процесу динамічної реконфігурації було застосовано новий метод моделі, заснований на AADL, а перехід перетворює модель на CPN для моделювання. Деякі обмеження зосереджені на кількох різних аспектах моделювання. Цей підхід було продемонстровано за допомогою чотиримодульної системи IMA. Результати прикладного дослідження показали ефективність цього методу. У цьому дослідженні був запропонований метод аналізу на основі моделі з кількома обмеженнями для процесу динамічної реконфігурації IMA, який допомагає вирішити проблеми безпеки при динамічній реконфігурації IMA, спричинені високою інтеграцією та складністю.

У майбутній роботі необхідно додати більше обмежень для більш всебічного аналізу. Якщо система стане складнішою, а кількість станів динамічної

реконфігурації вибухне, це буде складніше для аналізу. Велике навантаження виникає через аналіз. Таким чином, необхідно розробити інструмент для автоматичного додавання обмежувальних умов. Потрібна перевірка цього підходу за допомогою більш складних випадків та інженерних практик.

Розділ 5. Охорона праці

Вступ

Дипломний проект присвячений темі «Комплекс бортового радіоелектронного обладнання на базі технологій інтегрованої модульної авіоніки (IMA)» та безпосередньо пов’язаний з розробкою, аналізом та вдосконаленням роботи бортового радіоелектронного обладнання на базі технологій інтегрованої модульної авіоніки.

Суб’єктом охорони праці є інженер-дослідник бортового радіоелектронного обладнання, що аналізує, моделює та досліджує параметри інтегрованої модульної авіоніки. Місцем його роботи є дослідно-конструкторська лабораторія, яка знаходиться в авіаційному науково-дослідному центрі.

Розробка заходів з охорони праці потрібна для того, щоб вказати інженеру-досліднику, у якому порядку забезпечити особисту безпеку, зменшити вплив шкідливих виробничих факторів на робочому місці та забезпечувати безаварійну експлуатацію обладнання.

5.1. Організація робочого місця інженера-конструктора

Робоче місце інженера-дослідника знаходиться у технічному приміщенні на першому поверсі одноповерхової будівлі. Використано змішане освітлення. Стіни пофарбованні бежевий колір, на підлозі темний паркет та побілена стеля. Має такі параметри:

1.Довжина - 6м;

2.Ширина - 3м;

3.Висота приміщення 3 метра;

4. Площа приміщення $6\times3=18м^2$;

5. Об'єм приміщення $6\times3\times3=54м^3$.

У цьому приміщенні знаходиться два робочих місця. Згідно до НПАОП 0.00.-1.28-10 «Правила охорони праці під час експлуатації електронно-обчислювальних машин»: площу приміщень слід приймати з розрахунку не менше $6м^2$ на робоче місце працівника, а об'єм – не менше $20м^3$. У приміщенні 2 робочих місця $6\times3=18м^2$, що менше загальної площі приміщення $18м^2$,отже, площа відповідає вимогам.

Розташування робочих місць приведено на (рис. 5.1).

На робочому місці інженера-конструктора є прилади, які негативно впливають на нього, а саме:

- Бортове радіоелектронне обладнення;
- Система електроживлення;
- ПК Комп’ютер;
- Багатофункціональний пристрій (принтер, ксерокс, факс).

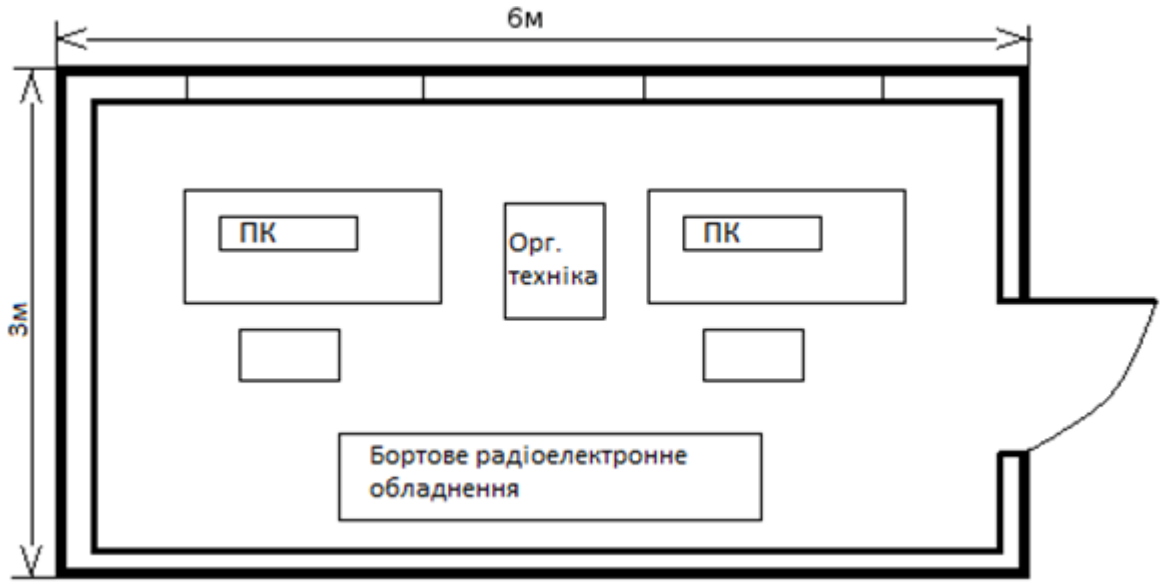

Рис. 5.1 Схема робочого технічного приміщення

5.2. Перелік шкідливих та небезпечних виробничих чинників

Основні небезпечні та шкідливі виробничі фактори діють на інженера - конструктора згідно з Державно-санітарними нормами «Гігієнічна класифікація праці за показниками шкідливості та небезпечності факторів виробничого середовища, важкості та напруженості трудового процесу». Затверджено наказом Міністерства охорони здоров'я України від 08.04.2014 № 248:

1. Мікроклімат: температура, вологість, швидкість руху повітря, теплове випромінювання.
2. Іонізація повітря.
3. Підвищений рівень шуму на робочому місці.
4. Електромагнітне випромінювання.
5. Електростатичне поле між екраном та інженером.

5.3. Мікроклімат: температура, вологість, швидкість руху повітря, теплове випромінювання.

Згідно з вимогами норм і стандартів ДСН 3.3.6.042-99 «Санітарні норми мікроклімату виробничих приміщень», в залежності від складності виконуваних робіт, інженер-дослідник відноситься до категорії Іа (легкі роботи, не потребують

фізичної напруги).

Мікроклімат на роб. місці характеризується: температура, t, ° С; відносна вологість, j,%; швидкість руху повітря на робочому місці, V, м / с; інтенсивність теплового випромінювання W, Вт / м2; температура поверхні.

При виконанні робіт пов'язаних з нервово-емоційним напруженням в кабінетах, пультах і постах керування технологічними процесами, в залах обчислювальної техніки та інших приміщеннях повинні дотримуватися оптимальні умови мікроклімату (температура повітря 22 - 24 ° С, відносна вологість 60 - 40%, швидкість руху повітря не більше 0,1 м / сек, допустиме значення повітрообміну 10 м3/год).

В даному приміщені рівень вологості занижений і складає ~ 35%.
Повітря, що надходить у приміщення, також, варто очищати від забруднення, у тому числі від пилу й мікроорганізмів.

5.4. Іонізація повітря

У повітрі зовнішнього природного середовища, як і в повітрі приміщень, наявна певна кількість заряджених частинок, що називаються іонами. У приміщеннях, де працюють з комп’ютерами, концентрація легких негативних іонів зменшується (за 5 хвилин у вісім разів, а за 3 години — до 0). Така зміна складу повітря призводить до несприятливого впливу на здоров’я користувачів ПЕОМ, їх розумову та фізичну діяльність. Гранично допустимі норми рівнів іонізації повітря приміщень при роботі з ЕОМ повинні відповідати ДНАОП 003-3.06-80.

Засоби забезпечення необхідної концентрації іонізації повітря такі:

• кондиціювання повітря;

• генератори негативних іонів (іонізатори);

• збільшення вологості повітря у приміщеннях.

5.5. Підвищений рівень шуму на робочому місці

На комп’ютеризованих робочих місцях основними джерелами шуму є кулери системного блоку, принтери. Сильний шум викликає труднощі з розпізнаванням колірних сигналів, знижує швидкість сприйняття кольорів, гостроту зору, зорову

адаптацію, порушує сприйняття візуальної інформації, зменшує на 5–12% продуктивність праці. Згідно ДСН 3.3.6.037-99 «Санітарні норми виробничого шуму, ультразвуку та інфразвуку», припустимий рівень шуму для лабораторії з тестування програмних розробок – 65 дБ. Фактичне значення рівня шуму становить 71,3 дБ. Порівнявши допустиме і фактичне значення рівня шуму можна зробити висновок, що для забезпечення нормальних умов праці необхідно зменшити рівень шуму на 6,3 дБ. Для забезпечення нормованого рівня шуму в лабораторії і на робочому місці інженера дослідника використовуються шумопоглинаючі засоби, вибір яких обґрунтовується спеціальними інженерно-акустичними розрахунками.

5.6. Пожежна безпека

При виникненні пожежі на робітників можуть впливати небезпечні чинники: відкритий вогонь та іскри; підвищена температура повітря, предметів, обладнання; токсичні продукти горіння, дим; знижена концентрація кисню; обвалення і пошкодження будівель, споруд, установок, вибух.

Основними причинами пожежі та вибуху на підприємствах є наступні:

- несправність виробничого обладнання;
- несправність та перенавантаження електричного обладнання;
- необережне ставлення до вогню (паління, використання відкритого вогню в недозволених місцях, залишання без нагляду електрообладнання);
- порушення правил пожежної безпеки.

Згідно з НАПБ А.01.001-2004 «Правила пожежної безпеки в Україні», для усунення цих причин необхідно підвищувати виробничу дисципліну, встановити суворий протипожежний режим. У приміщеннях встановлюються надійні засоби попереднього сповіщення небезпеки виникнення пожежі, та розміщуються схеми евакуації (рис. 5.2).

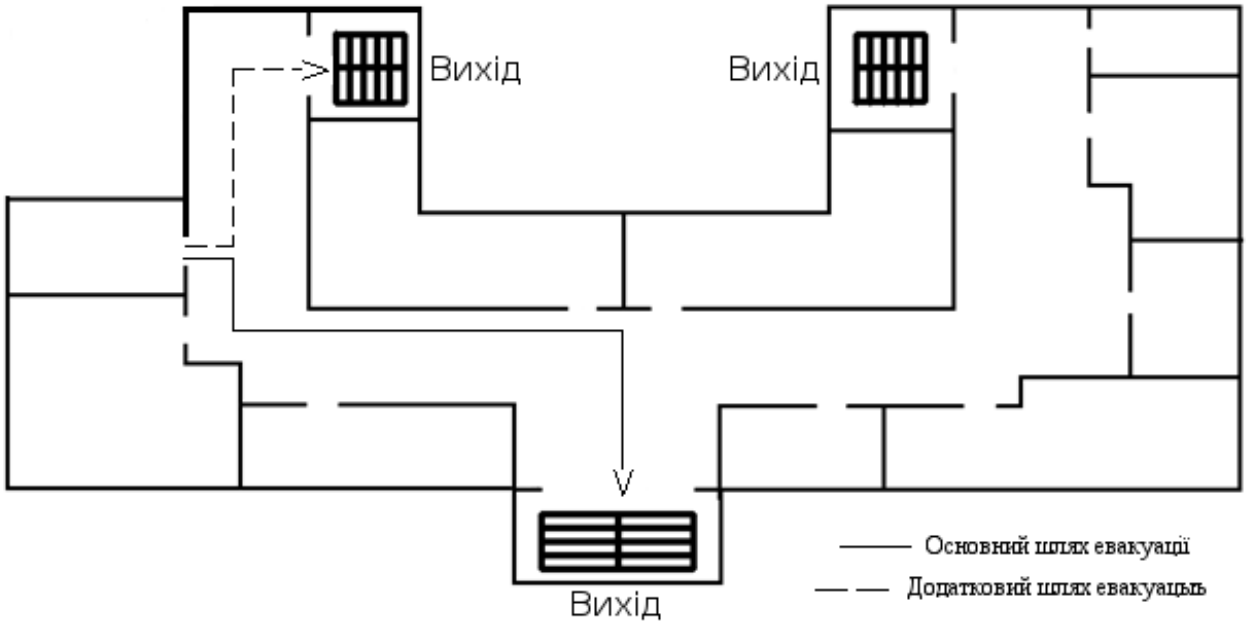

Рис. 5.2. План евакуації з приміщення у випадку пожежі

В приміщені класу «В» (у відповідності із НАПБ Б.03.002-2007 «Визначення категорій приміщень і будівель по вибухопожежній і пожежній небезпеці»), що розглядається, оснащене вогнегасником типу ВП-5.

У приміщенні повинні бути встановлені пожежні сповіщувачі, що реагують на дим. Вони повинні бути сертифіковані в Державному центрі сертифікації МНС України.

Забезпечити пожежну безпеку можливо наступними шляхами:

- Застосовувати запобіжники ;
- Контролювати справність обладнання;
- Намагатися якмога менше застосовувати у роботі легкозаймисті та вибухові речовини;
- Підтримувати чистоту та провітрювати приміщення;
- Вимикати електричне обладнання після закінчення робочого дня.

5.7. Розрахункова частина з охорони праці

Розрахунок повітрообміну за надлишками тепла у приміщенні.

1. Розраховуємо виділення тепла при роботі обладнання:

$$Q_{облад}=n \cdot P \cdot k1 \cdot k2 ,$$

де n – кількість комп'ютерів (обладнання) , n = 3 (2 комп'ютерів, 1 стенд);

P – встановлена потужність комп'ютерів, Рпк = 400 Вт., Рстенда = 800 Вт;

k1 – коефіцієнт використання встановленої потужності, k1 =0,8;

k2 – коефіцієнт одночасної роботи обладнання, k2 =0,5.

$$Q_{облад}= (2 \text{ x } 400 + 1 \text{ x } 800) \text{ x } 0,8 \text{ x } 0,5 = 640 \text{ Вт}$$

2. Кількість тепла, що виділяється одним чоловіком при 23°С, який виконує легку фізичну роботу, дорівнює 114 Вт.

$$q_ч = 114 \text{ Вт.}$$

3. Визначаємо кількість тепла, що виділяється однією жінкою, за формулою:

$$q_ж = q_ч \cdot 0,85 = 97 \text{ Вт.}$$

4. Розраховуємо виділення тепла від людей:

$$Q_л= n_ч \cdot q_ч + n_ж \cdot q_ж,$$

де $n_ч$ –кількість чоловіків, які працюють у приміщенні, $n_ч = 2$;

$n_ж$ – кількість жінок, які працюють у приміщенні, $n_ж = 0$;

$q_ч$ – кількість тепла, що виділяється одним чоловіком, $q_ч = 114$ Вт;

$q_ж$ – кількість тепла, що виділяється однією жінкою, $q_ж = 97$ Вт.

$$Q_л= 2 \text{ x } 114 + 0 \text{ x } 92 = 228 \text{ Вт.}$$

5. Розрахувати надходження тепла в приміщення офісу:

$$Q_{над} = Q_{облад}+ Q_л+ Q_{осв}+ Q_{рад}$$

де Qоблад – виділення тепла від обладнання, Qоблад = 640 Вт;

Qл – виділення тепла від людей, Qл = 228 Вт;

Qосв – виділення тепла від приладів освітлення, Qосв = 300 Вт;

Qрад – надходження тепла через зовнішні огороджуючі конструкції від сонячної радіації, Qрад = 180 Вт.

$$Q_{над} = 640 + 228 + 300 + 180 = 1348 \text{ Вт}$$

6. Проводимо розрахунок повітрообміну за надлишками тепла у приміщенні офісу за формулою:

$$L = \frac{3600 x Q_{над}}{C_p x \rho x (t_{вил} - t_{пр})}$$

де 3600 – коефіцієнт для переведення $м^3/с$ в $м^3/год.$;

L – кількість необхідного припливу повітря;

Qнад – кількість надходження тепла в офіс;

$c_р$ – питома теплоємкість повітря, р с =1000 Дж/(кг· о С);

ρ – щільність повітря, ρ = 1,2 кг/м 3 ;

tвил – температура повітря, що вилучається з приміщення, tвил = 23°С;

tпр– температура припливного повітря, tпр = 17°С.

$$L = \frac{3600x1348}{1000x1,2x(23-17)} = 674\ ^{м^3}/_{год}$$

Для даного повітрообміну можна запропонувати встановити вентиляційну установку “ВЕНТС ВУТ В/ ВБ” з повітрообміном 700 $м^3$ / год.

Висновок

Дослідивши робоче місце інженера-коструктора, були визначені загальні шкідливі чинники та методи їх усунення.

Основним шкідливим чинником підвищення температури повітря у приміщені через роботу комп’ютерів, стенду бортового радіообладнання та тепла, яке виділяють працівники. Для даного типу приміщення можна запропонувати встановити вентиляційну установку “ВЕНТС ВУТ В/ ВБ” з повітрообміном 700$м^3$/год.

Розділ 6. Охорона навколишнього середовища

Вступ

Повітряний транспорт має великий вплив на атмосферу Землі. Газотурбінні двигуни літаків працюють на авіакеросині, хімічний склад якого дещо відрізняється від автомобільного бензину та дизельного палива кращою якістю з меншим вмістом сірки та механічних домішок. Проте головна маса відпрацьованих газів викидається повітряними суднами безпосередньо у повітряному просторі на відносно великій висоті, при високій швидкості та турбулентному потоці, і лише невелика частка – у

безпосередній близькості від аеропортів та населених пунктів.

Основними компонентами, які забруднюють довкілля, є: окис вуглецю, неспалені вуглеводні, окиси азоту та сажа. На режимах холостого ходу та при русі по доріжках, при заході на посадку у відпрацьованих газах суттєво збільшується вміст окису вуглецю і вуглеводів, але при цьому зменшується кількість окису азоту.

Найбільш небезпечним є надходження цих речовин у стратосферу, що може бути однією з причин руйнування озонового шару.

6.1. Викиди шкідливих речовин

Джерелами викидів шкідливих речовин є відпрацьовані гази авіаційних двигунів, випаровування з системи живлення, підтікання пального і мастил у процесі роботи та обслуговування, а також продукти зносу фрикційних накладок зчеплення, накладок гальмівних колодок, шин. Потрапляючи в атмосферу, грунт шкідливі речовини, що викидаються ПС, негативно впливають на біосферу.

До числа шкідливих компонентів відносяться і тверді викиди, що містять свинець і сажу, на поверхні якої адсорбуються циклічні вуглеводні.

Закономірності розповсюдження в навколишньому середовищі твердих викидів відрізняються від закономірностей, характерних для газоутворюючих продуктів. Окремі фракції, осідаючи поблизу від центра емісії на поверхі грунту і рослин, в результаті накопичуються у верхньому шарі грунту. Дрібні фракції утворюють аерозолі і розповсюджуються з повітряними масами на великі відстані**.**

Хоча діоксид вуглецю не токсичний компонент, нагромадження його в атмосфері небезпечне, оскільки призводить до виникнення так званого парникового ефекту**.**

Оксид вуглецю**.** Він утворюється переважно в двигунах при роботі на багатих паливоповітряних сумішах. Виходить при неповному згорянні вуглецевих речовин. Причиною виникнення оксиду вуглецю в цьому випадку є нестача кисню для повного окислення вуглецю, який входить до складу палива. Незначна кількість оксиду вуглецю, що утворюється під час роботи на бідних сумішах, у тому числі і в дизелях, є продуктом проміжного окислення вуглецю, який через

нестачу часу на процес згоряння не встигає доокислитись до діоксиду вуглецю. У повітря він попадає в результаті спалювання твердих відходів, з вихлопними газами й викидами промислових підприємств. Щорічно цього газу надходить в атмосферу не менш ніж 1250 млн.т. Оксид вуглецю є сполукою, що активно реагує зі складовими частинами атмосфери й сприяє підвищенню температури на планеті, і створенню парникового ефекту. Він інертний і зберігається в повітрі 0,1-5 років.

Оксид азоту, що утворюється головним чином природним шляхом, нешкідливий для людини. Він представляє собою безбарвний газ зі слабким запахом і солодкуватим смаком. Вдихання невеликих кількостей N_2O призводить до притуплення больової чутливості, внаслідок чого цей газ іноді в суміші з киснем застосовують для наркозу. Вдихання чистого N_2O швидко викликає наркотичний стан і задуха.

Оксид азоту NO і діоксид азоту N_2O в атмосфері зустрічаються разом, тому найчастіше оцінюють їх сумісна дія на організм людини. Тільки поблизу від джерела викидів спостерігається висока концентрація NO.

Токсичний вплив оксиду азоту при його викидах проявляється в двох шарах атмосфери – страто- і тропосфері. В стратосфері він пов’язаний з існуванням захисного шару Землі. Каталітичне руйнування озонового шару NO_x спричиняє недопустиме зростання біологічно активної радіації і ставить під загрозу існування біосфери. Частина оксиду азоту потрапляє в стратосферу з тропосфери. Оксид азоту зберігається в оточуючій його атмосфері протягом 3-4 днів.

Вуглеводні (пари бензину, метану тощо) мають наркотичну дію, у малих концентраціях викликає головний біль, запаморочення і т. п. Так при вдиханні протягом 8 годин парів бензину в концентрації 600 мг/м3 виникають головні болі, кашель, неприємні відчуття в горлі. У відпрацьованих газах міститься кілька десятків різних вуглеводнів, які різняться за токсичністю. Джерелом вуглеводневих сполук є шари паливної суміші, прилеглі до стінок камери згоряння, де відбувається гасіння полум’я, частини камери згоряння, в яких через нерівномірний розподіл суміші виникає нестача кисню, а також циліндри, що працюють з пропусками запалювання та згоряння.

Вуглеводневі сполуки, які потрапляють в атмосферу, є також однією з складових, що утворює смоги у великих містах. Особливу небезпеку становить наявність у складі вуглецю канцерогенних речовин, які викликають захворювання на рак (наприклад, бензапірен).

Сірчаний газ. Розповсюдження сірчаного газу в повітряному середовищі відрізняється великою нерівномірністю. Сірчаний газ не отруйний, але в сполученні з іншими забрудненнями і вологою подразнює очі, ніс та горло, шкідливо впливає на легені, вбиває рослини, викликає корозію металів і зменшує прозорість атмосфери. При середньодобовій концентрації в повітряному середовищі більше 0,05 мг/м^3 сірчаного газу, що справляє токсичний вплив на флору, фауну, людину. Менші концентрації сірчаного газу в результаті зіткнення з водою призводять до закислення води і грунту.

Вуглекислий газ. Вміст в повітрі вуглекислого газу не нормований. Зростання концентрації вуглекислого газу небезпечне в тому відношенні, що при поглинанні довгохвильового теплового проміння створює “парниковий ефект”, що обумовлює перегрів поверхні землі.

Тривалість знаходження вуглекислого газу в повітрі – 4 роки. Вміст в повітрі вуглекислого газу знижує вміст в ньому кисню і тим самим зменшує значення порогових, небезпечних для людини концентрацій токсичних речовин.

Сажа. Утворення сажі в процесі згоряння вуглеводневих палив в циліндрах двигуна пов'язане з термічним розкладанням (піролізом) вуглеводнів палива під впливом високих температур і умовах нестачі кисню.

Підраховано викиди шкідливих речовин в зоні аеропорту за такий злітно-посадочний цикл для літаків різних типів (табл. 6.1.1).

Таблиця 6.1.1

Емісія з авіаційних двигунів для літаків різних типів

Тип літака	Викиди шкідливих речовин, кг/год				
	CO	CxHy	NOx	SOx	Попіл
Ту-154	48,8	45,5	68.3	0,6	2,0

Як-42	7,8	1,5	12,7	0,2	0,7
Ту-154М	53,2	9,3	15,6	0,5	1,8
Як-40	22,5	4,5	4,7	0,1	0,5

При неповному згорянні палива з відпрацьованими газами викидається сажа. Вона утворюється в камерах згоряння двигунів внаслідок піролізу палива при високих температурах і тиску в середовищі з нестачею кисню. Особливо багато сажі утворюється в дизелях. Головна небезпека, яку несе сажа, в тому, що вона може бути носієм канцерогенних речовин, які адсорбуються на поверхні її частинок**.**

6.2. Вплив шуму

Шум - безладне сполучення різних по рівню і частоті звуків. Численні експерименти і практика показують, що антропогенний шумовий вплив несприятливо впливає на організм людини і скорочує тривалість його життя, тому що звикнути до шуму фізично неможливо. Людина може суб'єктивно не помічати звуки, але від цього руйнівна дія на його органи слуху не тільки не зменшується, але і збільшується. При пристосуванні до сильного шуму організм людини втрачає велику кількість енергії, розвивається гіпертонія, підвищується агресивність. Жінки більш чутливі до сильного шуму й у них за умови шумового дискомфорту виникають ознаки неврастенії.

Шумовий антропогенний вплив несприятливий і для тварин. Маються дані, що поблизу аеропортів відбувається передчасне линяння птахів, вони починають погано орієнтуватися, тріскаються яйця в гніздах, у бджіл гинуть личинки. У США становили, що безладний шум потужністю 100 дБ призводить до спізнілого проростання насіння.

Визначений внесок у захист середовища від шумового впливу вносять заборона звукового сигналу автотранспорту, авіаційних польотів над містом, обмеження злетів і посадок літаків у нічний час.

Однак зазначені міри навряд чи дадуть належні екологічні ефекти, якщо не буде зрозуміло головне: захист від шумового впливу - проблема не тільки технічна,

але і соціальна, вона потребує виховання звукової культури у населення.

6.3. Заходи рішення проблем

Шкідливий вплив авіації на довкілля має глобальний і локальний характер. Глобальним є вплив авіації на озоновий шар атмосфери та пов'язані з цим наслідки, основні локальні - проблеми авіаційного шуму, забруднення викидами та скидами шкідливих речовин в атмосферне повітря, підземних вод та грунту у районі розташування аеропортів. Для розв'язання екологічних проблем цивільної авіації насамперед слід розробити:

- принципи та методи захисту повітря від забруднення двигунами повітряних суден;

- принципи та методи захисту від електромагнітних полів радіочастот аеропортів;

- технології захисту грунтів та води від забруднення стоками аеропортів;

- оптимізаційні схеми керування повітряним рухом на трасі, в зоні аеропортів з урахуванням екологічного стану довкілля;

- методи кількісної інтегральної оцінки екологічного стану підприємств авіаційного транспорту.

А також зменшення кількості шкідливих викидів може бути досягнуто при підвищенні економічності двигунів, а отже – зменшенні кількості відпрацьованих газів. Скорочення витрат палива, а від цього – і викидів токсичних речовин досягається також удосконаленням методів експлуатації літаків, а саме: підвищенням ступеня заповнення літаків корисним вантажем, зменшенням пробігу літаків на аеродромах під тягою власних двигунів за рахунок буксирування їх тягачами на злітну смугу, а також за рахунок розташування аеропортів на значній відстані від міст.

ВИСНОВКИ

Розвиток авіоніки в перспективних ЛА полягає в інтеграції інформаційних датчиків усіх бортових підсистем з метою формування об'єктивної реальної картини зовнішньої обстановки, а також для виведення результуючої інформації на індикатори. Впровадження перспективних технологій передачі даних в КБО ЛА нового покоління доцільно тільки при значному вдосконаленні інформаційних характеристик функціонального обладнання (датчиків, засобів відображення і т.п.) та інформаційно-алгоритмічного забезпечення, що безпосередньо обґрунтовує необхідність застосування інформаційних технологій з підвищеними функціональними характеристиками. Попередній аналіз потоків функціональних параметрів, які передбачається передавати в перспективних ЛА, показує різке збільшення величини інформаційного трафіку відповідно до нових тактичних задач, що вирішують перспективні КБО.

Нове покоління інтегрованої модульної авіоніки допоможе об'єднати різні обчислювальні пристрої та включти їх в централізовану систему, у результаті цього вийде, що з різних окремих систем авіоніки літака буде виключено більш ніж 100 різних лінійних замінних вузлів, які будуть включені в інтегровану систему із загальним ядром . Сам комплекс модульної авіоніки організований у вигляді єдиного апаратного середовища, системи перетворилися на функції, реалізовані програмно у цьому середовищі. Окремі БЦОМ і обчислювачі, замінені загальними процесорними ресурсами, які розподіляють між собою і виконують всі прикладні програми. Така організація дозволяє оптимально використовувати обчислювальні ресурси. Прикладне ПЗ не залежить від типів застосовуваних процесорів, їх взаємодія будується через проміжні стандартні інтерфейси. Це дозволяє удосконалювати апаратне середовище без необхідності переробляти програмне забезпечення.

За останні 50 років змінилося три покоління комплексів бортового обладнання. Комплекси першого покоління складалися з незалежних систем, кожна з яких містила свої власні датчики, обчислювачі, індикатори і пульти управління. Зв'язки систем один з одним були мінімальні і представляли собою

радіальні з'єднання джерело-приймач. Друге покоління мало федеративну архітектуру. Для неї характерне використання різними системами загальних ресурсів. Сучасне третє покоління бортових комплексів представляє інтегровану модульну авіоніку. Нове покоління відрізняється набагато більш високим ступенем інтеграції та узагальнення ресурсів. Ідея полягає в тому, щоб не розбивати комплекс на ряд автономних систем, а побудувати його на основі єдиної обчислювальної платформи.

Збільшення кількості та складності функцій бортового обладнання стало причиною переходу від найпростіших аналогових обчислювачів до бортових цифрових обчислювальних систем відкривши шлях до процесів інтеграції бортового обладнання і функцій управління, що, в свою чергу, забезпечило зростання надійності комплексу, незважаючи на зростання складності. Інтегрована модульна авіоніка (IMA) дозволила перенести всі функції управління на рівень програмного забезпечення. Це забезпечило апаратну побудову обчислювальної системи у вигляді набору обмеженого числа стандартних модулів. Використання операційних систем реального часу, в свою чергу, дозволило побудувати програмне забезпечення у вигляді окремих функціонально-програмних модулів.

5 покоління характеризується: інтегрованою структурою КБО, використанням високошвидкісних каналів інформаційного обміну; скороченням кількості окремих обчислювачів; комплексною обробкою інформації; перенесенням третинної і вторинної (частково) обробки інформації в «ядро»; застосуванням багатопроцесорних обчислювальних систем

5+ покоління характеризується: інтегрованою модульною авіонікою з використанням високошвидкісних каналів інформаційного обміну в якості базового інструменту міжмодульного і міжблочного зв'язку; використанням однорідного обчислювального середовища і уніфікованих модулів у різних системах КБО; комплексною обробкою інформації і гнучкою реконфігурацією при відмовах; використанням багатоядерних процесорів тощо.

Всі ці модулі допомагають здійснювати індикацію а також аналіз отриманих даних і потребують постійної перевірки оскільки від їх роботи залежать життєво

важливі системи літака. Одним із шляхів підвищення експлуатаційної надійності комплексів є використання ефективних методів контролю. На сучасному етапі розвитку авіаційного обладнання ускладнені завдання, які виконуються БЦОС, введення нових структурних і технічних рішень призвело до появи різних підходів до організації тестування БЦОС.

На практиці процедури тестового контролю БЦОС четверного покоління не задовольняють принципам побудови перспективних БЦОС, тому в дипломній роботі було досліджено таке:

- реалізація в обчислювачі послідовної схеми перевірки працездатності функціональних модулів виявляється ресурсномісткою і недопустимою для пристроїв, що працюють в реальному часі;

- реалізація в обчислювачі паралельної схеми перевірки здійснюється одночасно для всіх КФМ, однак внутрішні вузли модуля перевіряються як і раніше по послідовній схемі.

ВИКОРИСТАНІ ДЖЕРЕЛА

1. P. Frodyma and B. Waldmann. ARINC 429 Specification Tutorial. AIM GmbH, 2.1 edition, 2010.
2. Working Group. ARINC 615, P2: ARINC Airborne Computer Data Loader. Technical Report 1, Aronautical Radio, INC, June 1991.
3. P. Frodyma, J. Furgerson, and B. Waldmann. MIL-STD-1553 Specification Tutorial. AIM GmbH, 2.3 edition, 2010.
4. L. Buckwalter. Avionics Databuses, Third Edition. Avionics Communications Incorporated, 2008.
5. Miguel A. S´anchez-Puebla and Jes´us Carretero. A new approach for distributed computing in avionics systems. In Proceedings of the 1st international symposium on Information and communication technologies, ISICT 2003, pages 579 – 584. Trinity College Dublin, 2003.
6. Zhang, F.; Zhao, Y.; Ma, D.; Niu, W. Formal verification of behavioral AADL models by stateful timed CSP. IEEE Access 2017, 5, 27421–27438.
7. Airlines Electronic Engineering Committee. Avionics Application Software Standard Interface Part 1-Required
8. Services; ARINC Document ARINC Specification 653 P1-3; Aeronautical Radio, Inc.: Annapolis, MD, USA, 2010.
9. Standardization Agreement (STANAG), North Atlantic Treaty Organization (NATO). 4626-2005 Modular and Open Avionics Architecture (Part I: Architecture); North Atlantic Treaty Organization: Brussels, Belgium, 2005; pp. 24–34.
10. Jolliffe, G. Producing a safety case for IMA blueprints. In Proceedings of the 24th Digital Avionics Systems Conference, Washington, DC, USA, 30 October–3 November 2005; IEEE: Piscataway, NJ, USA, 2005; Volume 2.
11. López-Jaquero, V.; Montero, F.; Navarro, E.; Esparcia, A.; Catal'n, J.A. Supporting ARINC 653-based dynamic reconfiguration. In Proceedings of the 2012 Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture, Helsinki, Finland, 20–24 August 2012; IEEE: Piscataway, NJ, USA, 2012.
12. Bieber, P.; Noulard, E.; Pagetti, C.; Planche, T.; Vialard, F. Preliminary design of future reconfigurable IMA platforms. ACM Sigbed Rev. 2009, 6, 1–5. [CrossRef]
13. ARINC Specification 653: Part 1, Avionics Application Software Standard Interface, Required Services (March 7, 2006) available from ARINC, 2551 Riva Road, Annapolis, MD 21401 https://www.arinc.com/cf/store/index.cfm
14. ARINC Specification 653: Part 2, Avionics Application Software Standard Interface, Extended Services (January 22, 2007) available from ARINC, 2551 Riva Road, Annapolis, MD 21401 https://www.arinc.com/cf/store/index.cfm
15. ARINC Specification 653: Part 3, Avionics Application Software Standard Interface, Conformity Test Specification (October 16, 2006) available from ARINC, 2551 Riva Road, Annapolis, MD 21401

https://www.arinc.com/cf/store/index.cfm
16. RTCA DO-297/ED-124: Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations available from RTCA, 18th Ave NW, Washington D.C., www.rtca.org and EUROCAE, Paris, France, www.eurocae.org
17. Watkins, C. B., and Walter, R. Transitioning from federated avionics architectures to integrated modular avionics. In 2007 IEEE/AIAA 26th Digital Avionics Systems Conference (DASC), Dallas, TX, Oct. 21–25, 2007.
18. Gaska, T. Optimizing an incremental modular open system approach (MOSA) in avionics systems for balanced architecture decisions. In 2012 IEEE/AIAA 31st Digital Avionics Systems Conference (DASC), Williamsburg, VA, Oct. 14–18, 2012.
19. RTCA, DO-297 Integrated Modular Avionics (IMA) Development Guidance and Certification Considerations, 2005.
20. RTCA, DO-178B, Software Considerations in Airborne Systems and Equipment Certification, 1992.
21. SAE, ARP4754, The Aerospace Recommended Practice (ARP) (Guidelines For Development Of Civil Aircraft and Systems), 2010.