|  |  |  |
| --- | --- | --- |
|  | UNIVERSITATEA DIN CRAIOVA  FACULTATEA DE AUTOMATICĂ, CALCULATOARE ȘI ELECTRONICĂ  DEPARTAMENTUL DE AUTOMATICĂ ȘI ELECTRONICĂ |  |

**Document de Design**

**Monitorizarea cantității de CO2 dintr-un avion**

SIC/SAI/2/PRJ/B

**Realizat de:** Andrei-Ioan-Mihnea Văduva

**Verificat de:**

**Aprobat de:** Andrei-Ioan-Mihnea Văduva

**Versiunea curentă**: 2.0

**Data**: 31/01/2022

**CUPRINS**

[1. Versiunile documentului 1](#_Toc94558210)

[2. Scopul documentului 1](#_Toc94558211)

[3. Documente asociate 1](#_Toc94558212)

[3.1 Documente Aplicabile 1](#_Toc94558213)

[3.2 Documente Referință 1](#_Toc94558214)

[4. Abrevieri 1](#_Toc94558215)

[5. Design General 2](#_Toc94558216)

[5.1 Arhitectura statică 2](#_Toc94558217)

[5.2 Arhitectura dinamică 2](#_Toc94558218)

[6. Design Detaliat 2](#_Toc94558219)

[7. Matrice de trasabilitate 2](#_Toc94558220)

# Versiunile documentului

|  |  |  |
| --- | --- | --- |
| **Versiunea** | **Data realizării** | **Obs.** |
| 1.0 | 10/01/2022 | Varianta inițială |
| 2.0 | 31/01/2022 | Popularea capitolelor 2, 3, 4, 5 |

# Scopul documentului

A fost realizat de echipa formată din Văduva Andrei-Ioan-Mihnea, Gavrilescu Dragoș-Mihai, Drăgan Alin-Ionuț, Băcanu Ana-Maria-Alexandra în cadrul proiectului la disciplina SIC. Acest document prezinta specificațiile pentru proiectul *Monitorizarea cantității de CO2 dintr-un avion*.

# Documente asociate

În acest capitol vor fi prezentate documentele care au asistat la realizarea acestui proiect.

## Documente Aplicabile

În acest capitol sunt prezentate documentele aplicabile utilizate în proiect:

* [AD1] teme proiect SIC.pdf
* [AD2] Document de Specificații, SIC/SAI/2/PRJ/B

## Documente Referință

În acest capitol sunt prezentate referințele utilizate în proiect:

* [RD1] Manual UML

# Abrevieri

În acest capitol sunt prezentate abrevierile utilizate în proiect:

|  |  |
| --- | --- |
| **Abreviere** | **Definiție originala** |
| CO2 | Dioxid de carbon |

# Design General

## Arhitectura statică

Arhitectura statică constă în scheletul proiectului ce poate fi refolosit pentru orice tip de aplicație de acest timp.

Componente statice principale:

* pachetul „Senzori”
* pachetul „Actuator”
* pachetul „Control”
* pachetul „Environment”

**Pachetul Environment:**

Este format din condițiile de sistem prezentate in cerințele proiectului pentru o funcționare cat mai optima.

Principalul scop al acestui pachet este asigurarea unui mediu unde „controlul”, „senzori” si „actuatorii” pot rula in condițiile optime cerute.

**Pachetul Senzori:**

Este format din metodele de citire si validare a semnalelor senzorilor, ca aceste informații sa ajungă într-un format preprocesat in sistem.

Principalul scop al acestui pachet este citirea semnalului senzorilor si validarea acestuia înainte de a fi livrat pentru folosința componentelor imediat următoare.

**Pachetul Actuatori:**

Este format din metodele de validare si scriere a semnalelor de control, principalul scop al acestui pachet este validare comenzii impuse si scrierea acesteia către actuatorul aferent.

**Pachetul Control**

Este format din metode de validare si comanda, aceste lucruri făcând din acest pachet un pachet de control. El va avea ca scop colectarea informațiile de la modulele conectate si oferirea de comenzi conform procesări datelor primite.

## Arhitectura dinamică

Citește senzor

Menționează alarma

Environment

Senzori

Control (CO2)

Cheamă funcționalitate

Actuatori

Acționează dispozitiv

Arhitectura dinamică în acest context se referă la fluxul de funcționare din software, astfel diagrama prezentată va încerca descrierea acestui flux între componentele principale.

Componente statice principale:

* pachetul „Senzori”
* pachetul „Actuator”
* pachetul „Control”
* Pachetul „Environment”

**Pachetul Environment:**

Va chema funcționalitatea sistemului nostru la momente de timp bine definite, îndeplinind astfel sarcinile programate.

In acest caz, pachetul environment va cere pachetului „Control” (CO2) începerea verificărilor/ controlului stării de CO2 din avion.

**Pachetul Control:**

Este pachetul ce va obține si prelucra informațiile sistemului, oferind o reacție pe măsura intrărilor.

Acesta o data chemat de către pachetul „Environment”, va citi datele de la pachetul „senzori” pe care le va prelucra generând semnale aferente (WANING/ALARMA) sau alte acționari ale pachetului „Actuatori”.

**Pachetul Senzori:**

Este un pachet simplu, menit utilizării externe, care oferă si validează date cu referire la senzorii sistemului.

Acest pachet are rolul dinamic de a fi citit in vederea controlului de CO2 din avion.

**Pachetul Actuatori:**

Este un pachet similar pachetului senzori, menit utilizării externe, care oferă posibilitatea acționarii anumitor ieșiri implementate.

Acest pachet are rolul de a acționa anumite dispozitive sub comanda primita de la centrul de control CO2 din avion.

# Design Detaliat

**Pachetul Environment:**

Descriere generală:

Este format din condițiile de sistem prezentate în cerințele proiectului pentru o funcționare cat mai optimă.

Principalul scop al acestui pachet este asigurarea unui mediu unde „controlul”, „senzori” si „actuatorii” pot rula în condițiile optime cerute.

Descriere conținut:

* asigurarea unei legături temporale strânsă si bine definita;
* asigurarea unei metode de inițializare sigure si protejata;
* asigurarea unei metode de prioritare a sarcinilor;

Enviroment

variabile:

- ENV\_TimeBase\_CNT

functii:

+ main()

+ isr\_time\_base()

Interfețe pachet:

Acest pachet va interacționa cu:

* Pachetul „Environment”, pentru asigurarea perioadei de timp a sistemului

Interfete externe:

-

Interfete interne:

1. NUME: main

INTRARI: -

IESIRI: -

SUMAR: Reprezintă intrarea in programul nostru.

Aceasta este formata din doua parți separate astfel: secțiunea inițializare și secțiune bucla. In secțiunea de inițializare se vor inițializa toate pachetele proiectului, urmând ca in secțiunea bucla funcționalitatea proiectului sa fie chemata sub o condiție temporala stricta.

1. NUME: isr\_time\_base

INTRARI: -

IESIRI: -

SUMAR: Reprezintă restricțiile temporale ale programului nostru.

Aceasta va contoriza timpul trecut de la momentul inițializării cu o sub perioada bine definita astfel încât programul sa o poată utiliza in crearea altor intervale necesare.

**Pachetul Senzori:**

Este format din metodele de citire si validare a semnalelor senzorilor, ca aceste informații sa ajungă într-un format preprocesat in sistem.

Principalul scop al acestui pachet este citirea semnalului senzorilor si validarea acestuia înainte de a fi livrat pentru folosința componentelor imediat următoare.

Pachetul presupune:

* asigurarea unei metode de citire a semnalului de la senzor;
* asigurarea unei metode validare a semnalului senzorilor;
* asigurarea unei metode de oferire a statusului aferent;

Senzori

variabile: -

functii:

+ SEN\_Init()

+ SEN\_GetValue()

Interfete pachet:

Acest pachet va interactiona cu:

* Pachetul „Control”, pentru oferirea datelor de intrare ale sistemului

Interfete externe:

1. NUME: SEN\_Init

INTRARI: pin\_ident (unsigned char)

IESIRI: func\_state (unsigned char)

SUMAR: Reprezintă funcția de inițializare a pinului aferent senzorului din cadrul microcontrolerului folosit. Aceasta va seta regiștrii necesari configurării pinului menționat si va returna statusul.

1. NUME: SEN\_GetValue

INTRARI: pin\_ident (unsigned char)

IESIRI: \*val\_ptr (unsigned char pointer), func\_state (unsigned char)

SUMAR: Reprezintă funcția ce va citi informația de la senzor si o va valida înainte de transmitere. Validarea se va face intern comparând valoarea senzorilor cu un interval de corectitudine.

Interfete interne:

-

**Pachetul Actuatori:**

Este format din metodele de validare si scriere a semnalelor de control, principalul scop al acestui pachet este validare comenzii impuse si scrierea acesteia către actuatorul aferent.

Pachetul presupune:

* asigurarea unei metode de validare a comenzii impuse;
* asigurarea unei metode scriere a comenzii impuse;
* asigurarea unei metode feedback in vederea obținerii unui status;

Actuatori

variabile: -

functii:

+ ACT\_Init()

+ ACT\_SetValue()

Interfețe pachet:

Acest pachet va interacționa cu:

* Pachetul „Control”, pentru primirea comenzilor de ieșire ale sistemului

Interfete externe:

1. NUME: ACT\_Init

INTRARI: pin\_ident (unsigned char)

IESIRI: func\_state (unsigned char)

SUMAR: Reprezintă funcția de inițializare a pinului aferent actuatorului din cadrul microcontrolerului folosit. Aceasta va seta regiștrii necesari configurării pinului menționat si va returna statusul.

1. NUME: ACT\_SetValue

INTRARI: pin\_ident (unsigned char), val (unsigned char)

IESIRI: func\_state (unsigned char)

SUMAR: Reprezintă funcția ce va valida comanda si o va scrie actuatorului aferent. Validarea se va face intern comparând valoarea comenzi primite cu un interval de corectitudine.

Interfete interne:

-

**Pachetul Control**

Este format din metode de validare si comanda, aceste lucruri făcând din acest pachet un pachet de control. El va avea ca scop colectarea informațiile de la modulele conectate si oferirea de comenzi conform procesări datelor primite.

Pachetul presupune:

* asigurarea unei metode de procesare a fiecărui semnal de la senzori in vederea generării comenzi aferente;

Control

variabile:

– no\_of\_errors

functii:

+ CTRL\_Init()

+ CTRL\_Main()

+ check\_alarns()

Interfete pachet:

Acest pachet va interactiona cu:

* Pachetul „Senzori”, pentru citirea intrarilor sistemului
* Pachetul „Actuatori”, pentru scrierea comenzilor sistemului
* Pachetul „Environment”, pentru legatura de timp

Interfete externe:

1. NUME: CTRL\_Init

INTRARI:

IESIRI: func\_state (unsigned char)

SUMAR: Reprezintă funcția de inițializare pachetului nostru pentru condiții inițiale, resetarea variabilelor pachetului la condițiile inițiale.

1. NUME: CTRL\_Main()

INTRARI:

IESIRI: func\_state (unsigned char)

SUMAR: Reprezintă funcția ce va colecta stările alarmelor si va impune comenzi pentru a le acționa. Comandarea fiecărui actuator in funcție de starea alarmei găsite va fi inițiata in aceasta funcție.

Interfete interne:

1. NUME: check\_alarms()

INTRARI: state (unsigned char)

IESIRI: func\_state (unsigned char)

SUMAR: Reprezintă funcția de validare a alarmelor, fiecare alarma necesara sistemului va fi verificata in cadrul acestei funcții, astfel generând un singur semnal de alarma la un moment dat ce va fi folosit in direcționarea comenzii aferente.

# Matrice de trasabilitate

|  |  |  |  |
| --- | --- | --- | --- |
| Identificator  Specificatie | Specificatie | Componenta software | Comentarii |
| S1000 | Senzorii se citesc la interval de o secundă. | SEN\_GetValue, isr\_time\_base |  |
| S1100 | Valorile senzorilor trebuie sa fie citite într-un interval predefinit. | isr\_time\_base |  |
| S1200 | Când trei citiri succesive ale unui senzor generează valori în afara intervalului predefinit un led, ce corespunde senzorului respectiv, își schimba culoarea din verde în roșu. | CheckAlarms, ACT\_SetValue |  |
| S1300 | În funcție de valorile citite se pot genera semnale WARNING sau ALARM. | checkAlarms |  |
| S1400 | Când un semnal WARNING este generat un led își schimba culoarea din verde în roșu. | ACT\_SetValue |  |
| S1500 | Un WARNING persista pana când este achitat de pilot sau dispar condițiile care l-au generat. | ACT\_SetValue |  |
| S1600 | După achitare un nou WARNING va fi generat doar dacă a existat cel puțin o citire pentru care nu s-a generat WARNING sau ALARM. | checkAlarms |  |
| S1700 | Când un semnal ALARM este generat un led își schimba culoarea din verde în roșu. | ACT\_SetValue |  |
| S1800 | Un ALARM persista pana când dispar condițiile care l-au generat. | checkAlarms |  |
| S1900 | Cantitatea de CO2 este citită cu ajutorul unui senzor. | SEN\_GetValue |  |
| S2000 | Cantitatea de CO2 este afișată pe un ecran de tip ceas. | CTRL\_Main |  |
| S2100 | Atunci când cantitatea de CO2 este mai mare decât 0.1%, un semnal WARNING este generat | checkAlarms |  |
| S2200 | Dacă semnalul WARNING este achitat atunci se pornește instalația de ventilație. | ACT\_SetValue |  |
| S2300 | Atunci când cantitatea de CO2 este mai mare decât 0.3%, un semnal ALARM este generat. | checkAlarms |  |
| S2400 | După apariția semnalului ALARM se pornește instalația de ventilație | ACT\_SetValue |  |
| S2500 | Instalația de ventilație se oprește dacă cantitatea de CO2 este mai mică decât 0.05%. | ACT\_SetValue |  |