

***Modulul 2***

***Proiectarea sistemului (System Design)***

# Obiective

* Prezentarea generală a conceptului de proiectare şi a arhitecturii de sistem.
* Descompunerea unui sistem în blocuri funcţionale şi arii disciplinare.
* Identificarea blocurilor funcţionale şi a ariilor disciplinare în cazul proiectului propus.

Contents

* [Obiective 1](#_Toc495914378)
* [De ce este necesară proiectarea unui sistem ? 2](#_Toc495914379)
* [Ce reprezintă un sistem embedded? 3](#_Toc495914380)
* [Descompunerea unui sistem în blocuri funcţionale 4](#_Toc495914381)
* [Descompunerea unui sistem in arii disciplinare 8](#_Toc495914382)
* [Lucrare practică 9](#_Toc495914383)

# De ce este necesară proiectarea unui sistem ?

Dezvoltarea unui produs, în industria automotive şi nu numai, începe cu definirea unui set de **cerințe (requirements) de sistem**, rezultate în urma analizei cerințelor clientilor şi a tuturor celor implicați dar şi a standardelor şi restricțiilor legale.

Setul de cerinţe exprimă doar **ce** trebuie produsul să facă şi atributele acestuia, nu şi **cum** trebuie el implementat.

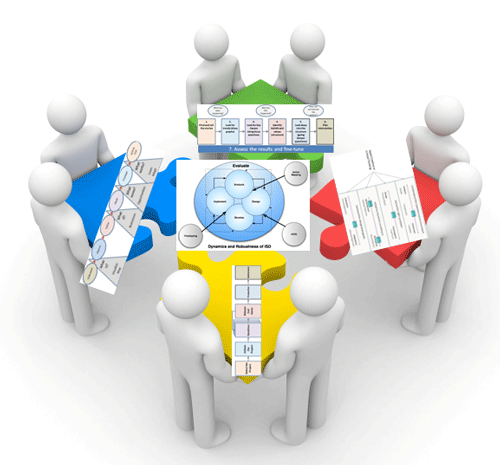
Soluția tehnică care corespunde cerinţelor definite rezultă din **proiectarea sistemului**. În stabilirea soluţiei tehnice se ţine cont de următorii factori:

* financiari (bugetul alocat, marja de profit, etc);
* tehnologiile tehnico-ştiințifice disponibile la acel moment (cunoştinţe, tehnologii noi state-of-the-art, inovaţii, etc);
* resursele umane şi materiale disponibile (angajaţi, echipamente de lucru, etc).

Astfel, **proiectarea sistemului** este procesul prin care se definesc arhitectura sistemului, elementele componente şi interacțiunile dintre ele, precum şi fluxul de date din sistem.

Proiectarea sistemelor ca şi concept a apărut înainte de al Doilea Război mondial, când inginerii care încercau să rezolve probleme complexe au simțit nevoia de a standardiza metodele aplicate.

Fără o astfel de proiectare prealabila şi fără a crea o trasabilitate între cerințe şi componentele sistemului, exista riscul ca sistemul dezvoltat să nu îndeplinească total cerințele. Acest fapt este cu atât mai important în cadrul sistemelor embedded de o complexitate ridicată cum sunt cele din industria Automotive.

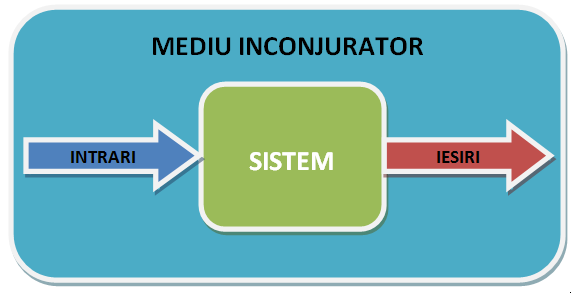


# Ce reprezintă un sistem embedded?

În general, prin noțiunea de sistem se ințelege un set de obiecte/entități care împreună cu interacțiunile dintre ele formează un întreg.

Sistemul este delimitat temporar şi spaţial. El poate fi descris prin structura lui, prin scopurile pe care le îndeplineşte şi prin funcţionalităţile sale.

Un sistem este influenţat de mediul înconjurător, prin intrările sale şi influenţează la rândul său mediul înconjurător, prin ieşirile sale.



Există practic, un număr nelimitat de posibilităţi de clasificare a sistemelor. De exemplu, putem vorbi despre sisteme sociale, sisteme financiare, sisteme organizaţionale, sisteme politice, sisteme tehnice, etc.

Un ***sistem* *embedded*** (embedded system) este un sistem tehnic ce conţine un procesor ca element central de decizie şi control. De regulă, el este folosit pentru a controla un proces fizic (electric, mecanic, hidraulic, etc).

În continuare, prin noţiunea de sistem ne vom referi la sistemele embedded.

# Descompunerea unui sistem în blocuri funcţionale

Pentru a uşura descrierea soluţiei tehnice, sistemul este descompus în **componente** sau **blocuri funcţionale**. Acestea, la rândul lor pot fi considerate sisteme şi pot fi descompuse în alte componente. În acest fel, pentru a se reduce complexitatea, sistemul complex se „sparge” într-o mulţime de sub-sisteme. Vom descompune până cand vom obţine componente atomice (indivizibile), ce vor realiza implementarea fizică a sistemului.

Procesul invers, de re-compunere a componentelor pentru a forma sistemul iniţial, pe baza arhitecturii sistemului, se numeşte **integrare**.

La sfârşitul procesului de integrare, sistemul trebuie să funcţioneze ca un tot unitar, respectând cerinţele specificate.

***Sistem embedded***

Actuatoare şi dispozitive de interfatare cu actuatoarele

Unitatea electronică de control

Senzori şi dispozitive de interfaţare cu aceştia

Sursa de energie

Elemente de interfaţare cu utilizatorul

**Proces controlat**

COMANDĂ

FEEDBACK

*Componentele principale ale unui sistem embedded*

## Sursa de energie

Din legea de conservare a energiei aflăm că nici un sistem tehnic nu produce lucru mecanic sau energie (chimică, electrică, mecanică, termică, etc), dacă nu o consumă dintr-o ***sursă de energie***. Mai mult, energia consumată de sistem este întotdeauna mai mare decat energia utilă sau lucrul mecanic pe care sistemul o/îl generează.

În sistemele embedded, sursa de energie este o sursă de curent electric, ce poate fi: o baterie, un acumulator, o sursă de alimentare de la reţeaua de curent alternativ sau o baterie solară.

În alegerea sursei de energie se ține cont atât de: prețul, masă și dimensiunile geometrice, cât și de parametrii sistemului și ai sursei: tensiunea nominală de funcționare, consumul de curent, puterea electrică.

## Unitate electronica de control

***Unitatea electronică de control***, ***ECU*** (Electronic Control Unit) este numit „creierul” sistemului și are abilitatea de a executa un ***program software*** (firmware), care este scris de regulă în ***limbajul C***.

Aceasta este formată dintr-un ***microcontroler*** (mai rar un microprocesor), împreună cu electronica necesară funcționării lui (circuit de clock, condensatori de decuplare pe pinii de alimentare, circuit de reset, etc). Totodată se urmărește simplificarea schemei electronice a sistemului prin implementarea unor strategii și algoritmi complecși ( de interfațare cu utilizatorul sau de control ) în software.

Microcontrolerul, din punct de vedere electronic, este conectat la restul compomentelor sistemului prin intermediul:

* **porturilor** de intrare-ieşire (***D****igital* ***I****nput*, ***D****igital* ***O****utput*) sau analogice (***A****nalog* ***I****nput*),
* **perifericelor** incorporate (***P****ulse* ***W****idth* ***M****odulation*, ***U****niversal* ***S****ynchronous* ***A****syncronous* ***R****eceiver* ***T****ransmitter*, ***C****ontrol* ***A****rea* ***N****etwork*, ***L****ocal* ***I****nterconnection* ***N****etwork*, ***S****erial* ***P****eripheral* ***I****nterface*, ***I****nter-****I****ntegrated* ***C****ircuit or* ***I2C***, *Timer*, etc).

În alegerea unui microcontroler se ţine cont de:

* resursele microcontrolerului: puterea de procesare, capacitatea memoriei, existenţa perifericelor necesare;
* costul şi disponibilitatea uneltelor de dezvoltare: programator, *debugger*, compilator pentru limbajul C, **IDE** (***I****ntegrated* ***D****evelopment* ***E****nvironment*);
* tensiunea de alimentare, puterea consumată, precum şi existenta unor moduri de funcţionare de tip *sleep;*
* data până la care microcontrolerul este produs;
* preţul de achiziţie al microcontrolerului.

## Senzori şi dispozitive de interfaţare cu aceştia

Orice sistem trebuie să obțină și să interpreteze informații din mediul său înconjurător. Iar, un ***senzor*** este un obiect ce poate detecta valoarea unor mărimi fizice din mediul înconjurător (de intrare) și apoi să reacționeze prin modificarea corespunzătoare a stării (ieșirii) sale.

În cazul sistemelor embedded, un senzor este un dispozitiv ce poate detecta mărimi fizice de intrare precum: tensiunea, curentul electric, temperatura, presiunea, radiația sau intensitatea luminoasă, forță, masă, etc; și apoi să reacționeze la ieșire prin modificarea unei ***tensiuni* *electrice*** (mai rar a unui ***curent* *electric***). Apoi, prin intermediul porturilor de intrare digitale sau analogice, ieșirea senzorului va fi citită de programul aflat în microcontroler.

Așadar, în proiectarea sistemului embedded trebuie avut grijă ca între senzor și microcontroler să existe ***dispozitivele de interfațare*** cu senzorii. Acestea au rolul de a converti, după o relație cunoscută, tensiunea sau curentul de ieșire al senzorului, într-o tensiune sau curent de intrare compatibil cu porturile microcontrolerului (de regulă [0..5V] sau [0..3.3V] și 10-30mA). Simultan au și rolul de a limita curentul care circulă prin portul microcontrolerului, protejându-l astfel de supra-tensiuni sau supra-curenți ce l-ar putea distruge.

În alegerea unui senzor, se ține cont în general, de următoarele caracteristici:

* tipul de mărime fizică masurată,
* plaja de valori ale mărimii de ieşire a senzorului,
* caracteristica (funcţia de tansfer) intrare-ieşire: poate fi liniară, neliniară, tabelară, etc
* rezoluţia senzorului (cea mai mica schimbare a mărimii fizice de intrare ce mai poate determina schimbarea mărimii de ieşire),
* costul, disponibilitatea şi complexitatea dispozitivelor de interfaţare,
* necesitatea calibrării senzorului,
* preţul senzorului.

## Elemente de interfaţare cu utilizatorul

Am văzut că senzorii au rolul de a „citi” și a „traduce” parametrii mediului înconjurător astfel încât să fie „înțeleşi” de către ECU. În foarte multe cazuri însă, este necesar ca un sistem embedded să poată fi controlat sau să pună la dispoziție informații pentru un om. Cum oamenii nu pot sesiza variații de tensiuni în plaja [0..5V] sau curenți de ordinul zecilor de mA, este nevoie de utilizarea unor elemente de ***interfațare cu utilizatorul*** (*user interface*). Pentru a controla un sistem embedded oamenii se folosesc, de regulă, butoane (cu reținere sau fără), comutatoare cu două sau mai multe poziții, potențiometre, *touch-screen-uri* și altele. Iar pentru a transmite informații către oameni, sistemele embedded folosesc dispozitive de afișare (LCD, LED cu 7 segmente), dispozitive luminoase (LED-uri), dispozitive sonore (difuzor, buzzer) sau cu vibrații.

Pentru alegerea interfeţei cu utilizatorul se vor lua în considerare urmatoarele:

* distanţa şi vizibilitatea dintre om şi sistem (îl pot vedea? îl pot auzi?);
* usurinţa de manipulare sau de citire a informaţiilor;
* usurinţa de interfaţare cu microcontrolerul;
* preţul elementelor de interfaţare.

## Actuatoare şi dispozitive de interfaţare cu actuatoarele

Majoritatea sistemelor embedded transferă energie în mediul înconjurător prin intermediul lucrului mecanic. Cu alte cuvinte, se „miscă” ele însele sau „miscă” ceva din mediul înconjurator. Un ***actuator*** sau ***element* *de* *execuţie*** este o componentă ce transformă o formă de energie în lucru mecanic (sau, cu alte cuvine, în miscare). Exemple uzuale de actuatoare folosite în sistemele embedded sunt motoarele electrice, releele, valvele, pompele. Ca şi în cazul senzorilor, pentru a putea controla aceste dispozitive este nevoie de putere electrică mult mai mare decât se poate transfera direct de la microcontroler. De aceea este nevoie de ***dispozitive de interfaţare*** cu actuatoarele ce au rolul de a converti semnalul de putere scazută de la ECU în semnal de putere mare ce poate controla actuatorul.

Printre cele mai importante criterii în alegerea actuatoarelor se pot menţiona:

* tipul de lucru mecanic (mişcare) necesar: liniară, rotaţie, ritmică, oprit-pornit;
* tipul de mărime fizică controlată: viteză, debit, curent electric;
* plaja de valori ale mărimii fizice controlate;
* caracteristica intrare-ieşire: liniară, neliniară;
* costul, disponibilitatea şi complexitatea dispozitivelor de interfaţare.

## Procesul controlat

Sistemele embedded sunt construite cu un anumit scop. Produc sau transferă energie, efectuează o mişcare, se mişcă ele însele.

***Procesul controlat*** reprezintă tocmai tipul de acțiune pe care sistemul embedded trebuie să o efectueze pentru a-și îndeplini scopul pentru care a fost construit. Ca și exemple simple de procese controlate putem menționa: deplasarea unui robot dintr-un punct în altul, controlul intensității curentului electric printr-un circuit electric, ridicarea unui ascensor cu viteză constantă, etc. Sistemul embedded acționează asupra procesului controlat prin ***comenzi*** și primește informații despre procesul controlat prin ***feedback***.

# Descompunerea unui sistem în arii disciplinare

Complexitatea crescândă a sistemelor embedded a dus la o implementare a acestora, în care sunt implicate arii disciplinare (discipline) precum electronică, mecanică sau software (aceste sunt cele mai folosite în proiectarea sistemelor embedded). Întrebarea cheie este: cum poate fi descris și cum funcționează sistemul din punct de vedere Software (electronic, mecanic) ?

Orice componentă a sistemului poate fi împărțită la rândul ei, în arii disciplinare, iar un exemplu poate fi cel prezentat mai jos, în care se poate vedea împărțirea unui senzor de măsurare a temperaturii.

**SOFTWARE**

**SISTEM**

**MECANICA**

**ELECTRONICA**

În **disciplina mecanică**, se urmăresc în principal următoarele aspecte: dimensiunile geometrice, masa, comportamentul mecanic la temperatură, vibrații, coroziune, etc.

În **disciplină electronică**, se urmăresc în principal următoarele aspecte: tensiunea nominală de funcționare, consumul de curent și puterea disipată, caracteristicile electrice maxime, comportamentul în funcție de temperatură, caracteristica de funcționare intrare-ieșire, influența zgomotului electro-magnetic, etc.

În **disciplină software**, se urmăresc în principal următoarele aspecte: modul de citire a mărimilor de ieșire, intervalul de valori a mărimii de ieșire, plauzabilizarea valorilor mărimii de ieșire, frecvența de citire a mărimii de ieșire pentru senzori; modul de comandă, plaja de valori de comandă, frecvența de trimitere a comenzii pentru actuatoare; diagnoza defectelor pentru toate componentelor sistemului.

La nivelul fiecarei discipline se va respecta ciclul de dezvoltare (*V-cycle*) pentru implementarea soluţiei aferente disciplinei respective.

Ca și în cazul descompunerii în componente și în această situație există procesul invers de **integrare** a disciplinelor pentru a forma inapoi soluția sistemului. De această integrare este responsabilă disciplina numită **sistem**.

În cadrul proiectului nostru, datorită faptului că utilizăm kit-uri deja fabricate, vom aprofunda aspectele de implementare din punctul de vedere al disciplinei **software** și vom accentua aspectele de interfațare cu celelalte două discipline.

# Lucrare practică

Identificarea blocurilor funcționale şi a ariilor disciplinare pentru proiectul curent. Detaliați rolul fiecărui bloc funcțional şi a fiecărei discipline identificate. Verificați trasabilitatea între componentele alese şi cerințele proiectului.

