

# Documentatie Mod Functionare Module `instr_dcd` si `regs`

## 1 Motivarea alegerii modului de functionare pentru `instr_dcd` si `regs`

### 1.1 `Instr_dcd`

Pentru modulul `instr_dcd` am ales o schema cu doua stari, `SETUP` si `DATA`, deoarece este cea mai clara si mai sigura metoda de a lucra cu instructiuni care vin pe 8 biti. Ideea principala a fost sa separ complet momentul in care instructiunea este citita de momentul in care se executa operatia, ca sa evit orice posibilitate ca modulul sa scrie sau sa citeasca din greseala.

In starea `SETUP` modulul primeste byte-ul de instructiune si il decodeaza. Aici se extrag informatiile importante: daca urmeaza o citire sau o scriere, pe ce adresa si ce parte (`high` sau `low`) trebuie folosita. Am ales sa pun aceasta logica in `SETUP` fiindca asa ma asigur ca, atunci cand intru in `DATA`, modulul stie deja ce are de facut si nu mai exista confuzii.

Starea `DATA` este cea in care se intampla efectiv operatia. Daca instructiunea indica o scriere, modulul ia byte-ul urmator si il trimite catre registrul corespunzator. Daca instructiunea este de citire, modulul scoate valoarea din registru. Totul este sincronizat cu `byte_sync`, ca sa evit operatii facute prea devreme sau prea tarziu. Aceasta abordare face modulul mai predictibil si mai usor de testat.

In plus, am scos semnalul `high_low` ca iesire separata pentru debug si testbench, astfel incat pot vedea direct ce parte de date a fost selectata. Per total, am ales aceasta structura simpla cu doua stari pentru ca reduce riscul de erori, organizeaza clar fluxul de date si permite extinderea usoara in viitor.

### 1.2 `Regs`

Pentru modulul `regs` am ales o organizare pe adrese fixe, unde fiecare registru este accesat printr-o adresa clara. Aceasta metoda este simpla, usor de folosit si evita ambiguitatile atunci cand se fac citiri sau scrieri din partea unui alt modul sau procesor.

Registrul de perioada, precum si cele doua registre de comparatie, sunt pe 16 biti, asa ca le-am impartit in doua adrese: una pentru partea joasa si una pentru partea inalta. Aceasta structura este standard, intuitiva si permite actualizarea valorilor fara logica complicata. Fiecare scriere actualizeaza direct registrul vizat, iar la citire se trimite exact byte-ul cerut de adresa.

Am introdus si un registru special pentru resetarea numaratorului: o simpla scriere genereaza un puls scurt de reset. Acest lucru permite resetarea rapida a numaratorului fara sa afectez alte functii ale modulului.

Pentru restul registrelor, cum ar fi prescalerul, enable-ul, directia de numarare sau enable-ul PWM, am folosit adrese simple si valori pe 1 sau 8 biti, astfel incat sa fie usor de controlat si inteles. La citire, modulul intoarce intotdeauna continutul real al registrului, ceea ce face sistemul usor de monitorizat.

Am ales acest mod de organizare pentru ca este clar, usor de extins si potrivit pentru orice modul care trebuie configurat prin adrese. Logica este directa, usor de verificat si reduce sansele ca o adresa sa fie interpretata gresit. Per ansamblu, aceasta structura face modulul robust, simplu de intretinut si foarte predictibil in utilizare.