Tema 1 – Prelucrarea imaginilor

Ioniță Cosmin – Ștefan

332 AC

Pentru implementarea acestei teme am folosit un circuit secvențial pentru parcurgerea prin anumite stări cu scopul de a sincroniza comportamentul variabilelor și soluționa problema.

În algoritmul folosit am modificat parametrii pentru a fi de tip reg astfel încât să permită scrierea acestora de-a lungul implementării. Pentru această problemă am adăugat variabile de tip reg ce ajută pentru memorarea pixelilor și pentru parcurgerea secvențială a soluției. În blocul combinațional am început cu o stare inițială pentru a inițializa unele variabile după care trecem în rezolvarea primului task: transformarea imaginii prin oglindire pe verticala.

Pentru acest task am avut nevoie de 4 stări, dintre care 1 este destinată pentru semnalul mirror_done. În primele 2 stări „jonglăm" cu rândurile imaginii în care memorăm valoarea pixelilor de pe poziția inițială și cea oglindită, primiți ca input, și scriem, în output pe poziția oglindită, primul pixel memorat. În cea de-a 3 a stare începem scrierea în output pe poziția inițială al celui de-al 2 lea pixel memorat (cel de pe poziția oglindită primit ca input) dupa care începe partea de incrementare a rândurilor și coloanelor și ne întoarcem înapoi în prima stare a acestui task. La final am adăugat o condiție în care trecem în starea destinată semnalului mirror_done, unde îi atribuim valoarea 1 dacă am parcurs toată matricea imaginii. După ajungerea în starea finală a acestui task, trecem la următorul: echivalarea imaginii în grayscale.

Pentru acest task am folosit de asemenea tot 4 stări, dintre care 1 este rezervată pentru semnalul gray_done. În prima stare am reinițializat semnalele row și col cu 0 pentru a porni de la primul pixel de pe imaginea prelucrată anterior. În a 2 stare folosim 3 condiții diferite pentru a găsi minimul și maximul dintre cele 3 canale ale pixelilor și a memora media dintre acestea. În cea de-a 3 a stare scriem valoarea memorată in canalul G al pixelului de output și le setăm pe cel de R și B la 0 după care începe partea de incrementare. La final, am folosit aceeași condiție din task-ul anterior pentru a trece în ultima stare destinată semnalului gray_done unde îi atribuim valoarea 1 și trecem apoi în starea unde vom începe rezolvarea ultimului task: tranformarea imaginii obținute prin aplicarea unui filtru de sharpness.

Pentru implementarea acestuia am considerat 3 vectori cache ce memorează coloanele vecine ale pixelului curent și am folosit 20 de stări, dintre care 9 sunt rezervate pentru cazurile in care pixelul curent se afla la marginile sau în mijlocul matricei, 7 stări pentru umplerea și manevrarea vectorilor și 5 stări pentru parcurgerea și rezolvarea task-ului (de asemenea, ultima stare este exclusiv pentru semnalul filter_done). Motivul pentru care am luat decizia de a folosi vectorii cache este de a reține starea inițială a pixelilor înainte de aplicarea filtrului pe aceștia și a evita folosirea pixelilor suprascriși. După ce am memorat coloanele, verificăm în ce zonă a matricei se afla pixelul curent (dacă se află la margine sau în mijloc) și aplicăm filtrul pe acesta. După care scriem pixelul modificat și trecem la partea de incrementare. La fiecare incrementare a coloanei, vectorul de pe coloana curentă devine vector anterior, vectorul de pe coloana următoare devine vectorul curent și după memorăm coloana următoare în vectorul respectiv. Când am ajuns la capătul matricei, trecem în starea finală unde îi dăm semnalului filter_done valoarea 1 și încheiem astfel implementarea temei.