.

**Projektni zadatak iz predmeta LPRS2**

**Sudoku**

**Zadatak**

Projektni zadatak se sastoji od modifikacije prethodno realizovane igre Minesweeeper sa ciljem da se dobije Sudoku igra sa pratecim funkcionalni gameplay-em.

**Realizacija grafičke periferije procesora**

U projektu Super Mario realizovana je arhitektura koja se sastoji od *MicroBlaze* procesora, njegove *BRAM* memorije i grafičkog procesora. Zadatak periferija grafičkog procesora je da na osnovu ulaznih koordinata iscrtava linije koje sluze kao okvir kompletne Sudoku table, zatim okvir svake 3x3 table kao I okvir svakog pojedinacnog polja. Vrednost boje piksela zadaje se u RGB formatu.

Harvdver koji je korišćen u izradi ovg projekta u potpunosti je preuzet od projekta na koji se neposredno oslanja. Samim tim se zadatak svodi na implementaciju Sudoku igre na postojećem hardveru.

**Realizacija softvera**

Na samo početku je potrebno iscrtati Sudoku Tabelu što je ostvareno unutar funkcije drawSudokuGrid(). Ta funkcija se u svom radu oslanja na funkcije koje čija je funkcija iscrtavanje vertikalnih I horizontalnih linija različite širine (tanje su širine 1 piksela dok šire imaju 2 piksela). Šire linije imaju ulogu granice cele tabele ali I naglašavaju granice između 3x3 tabela kako bi tabela bila preglednija za igrača.

Logika igre ostvarena je upotrebnom tri matrice dimenzija 9x9 a to su indicationMap, solvedMap, userInputMap. Poćetno stanje igre se vuca unutra indicationMap matrice I njene vrednosti se ne menjaju. Nenulte vrednosti u odgovarajucim poljima diktiraju sadzaj odgovarajucih polja Sudoku slagalice. Kao što je napomenuto vrednosti ove matrice se ne menjaju bez obzira na unos korisnika što omogućava ponovno pravilno iscrtavanja početnih vrednsoti ukoliko igrač želi da igru započne ispočetka. Matrica solvedMap sadrži u potpunosti rešenu tabelu I koristi se za proveru kraja igre kada se ta provera pokrene na zahtev korisnika korišćenjem prekidača SW0. Svi korisnički potezi se čuvaju unutar matrice userInputMap. U početku se ona inicijalizuje vrednostima koje sadrži indicationMap a kasnije se pored tih na odgovarajuća mesta unosi korisnička vrednost uneta na istovetno mesto u igri. Inicijalizacija sve tri matrice se obavlja pre početka igre upotrebom funkcija navedenih u tabeli.

Poslednji korak pre pokretanja igre sastoji se u iscrtavanju vrednosti unutar indicationMap matrice na odgovarajuća mesta na ekranu. Ono što je potrebno uraditi tada jeste prolazak kroz inicijalne vrednsoti slagalice unutar indicationMap matrice I iscrtavanje odgovarajućeg broja u tabeli korišćenem drawMap() funkcije svaki put kada se u matrici početnih vrednosti naiđe na vrednost veću od nule.

Način na koji drawMap() funkcioniše je taj što joj se moraju zadati X I Y coordinate ekrana od kojih se počinje iscrtavanje željenog sadržaja, coordinate koje definišu položaj sprajta od koga se iscrtaavanje započinje kao I vrednosti dužine I širine iscrtanog elementa u pikselima.

Radi jednostavnijeg I bržeg iscrtavanja svih brojnih I slovnih vrednosti ne koristi se programsko iscrtavanje nego se upotrebom drawMap() funkcije isecaju željeni delovi sprite-ova. Sprite je predstavljen C-ovskim zaglavljen koji odgovara .doc dokumentu u koji su upisani brojevi od 0 do 9 u u različitim bojama a potom I kompletan alphabet čiji će elementi biti ispisivani na ekranu kako bi korisnika obavestili o rezultatu zahtevanih akcija ispravnoh unosa I kraja igre. Drugom bojom(crvenom) se iscrtavaju početne vrednosti kako bi se igraču naglasilo koje vrednosti su početne I koje vrednosti samim tim ne može menjati. Korisnički unos iscrtava se crnom bojom.

E:\blabla.png

**Slika 1. Grafički elementi Sudoku-a**

Kompletna igra se odvija unutar While petlje funckije move() koja se pokreće nakon što se iscrta Sudoku tabela I označava početak igre. Dozvoljeni potezi korisnika su pre svega pomeranje u levo, desno, na gore I na dole u cilju kretanja kroz tabelu. Kako bi korisnik uneo vrednost u tabelu neophodno je da se pozicionira na željeno polje a zatim klikom na centralni JOY se unosi vrednost. Brojna vrednost koja se unosi zavisi od broja klikova na taj taster, Nakon toga se ta vredenost iscratava na ekranu I unosi u userInputMap matricu na istovetnu poziciju. Sve vrednosti koje je korisnik uneo moguće je menjati.

Ukoliko igrač želi da proveri ispravnost svog dosadašnjeg unosa potrebno je da to učini korišćenjem prekidača SW1. Na ekranu se ispisuje poruka o ispravnom odnosno neispravnom unosu. Ovo se odnosi na sve vrednosti unite do tog trenutka. Provera o stizanju do kraja igre se vrši korišćenjem prekidača SW0 nakon čega se prvo korisnik obaveštava o tome da li je tabela u potpunosti I ispravno popunjena ili nije. Ukoliko je došlo do kraja igre tada se onemogućavaju dalji potezi.

Potpuna implementacija igre bi u sebe uključivana generator Sudoku slagalica koji bi definisao sadrćaj tri ranije pomenute matrice. Jednostavnosti radi ovde je korišćena jedna slagalica za koju su vrednosti ručno unete. Radi provere ispravne funkcionalnosti I kao moguća ispomoć pri rešavanju uz projekat su priloženi početna tabela I rešena tabela unutar primerSudokuaZaProjekatSaResenjem.pdf fajla.

|  |  |
| --- | --- |
| Definicija funkcija | opis funkcije |
| void drawSudokuGrid() | Iscrtavanje Sudoku tabele |
| void initializeIndicationMap() | Popunjavanje indicationMap matrice |
| void initializeUserInputMap() | Početno stanje userInputMap matrice |
| void initializeSolvedMap() | Inicijalizacija solvedMap matrice |
| void move() | Omogućava kretanje I pamćenje unosa |
| int proveriIspravnost() | Proverava ispravnost aktuelnog unosa |
| int proveriKraj() | Provera kraja igre |
| void drawMap(int in\_x, int in\_y, int out\_x, int out\_y, int width, int height) | Iscrtavanje brojnih I slovnih vrednosti |
| void drawingThinHorizontalLine(int startX, int startY, int endX, int endY) | Iscrtavanje tanke poprečne linije na ekranu |
| void drawingThickHorizontalLine(int startX, int startY, int endX, int endY) | Iscrtavanje široke poprečne linije na ekranu |
| void drawingThinVerticalLine(int startX, int startY, int endX, int endY) | Iscrtavanje tanke uspravne linije |
| void drawingThickVerticalLine(int startX, int startY, int endX, int endY) | Iscrtavanje široke uspravne linije |
| void drawingCursor(int startX, int startY, int endX, int endY) | Iscrtavanje kursora na željenoj poziciji |
| void deleteCursor(int startX, int startY, int endX, int endY) | Brisanje kursora sa prethodne pozicije |

**Tabela 1. Pregled bitnih funkcija**

|  |  |
| --- | --- |
| Definicija promenljive | opis promenljive |
| int endOfGame | Flag za označavanje kraja igre(vrednost 1 znači kraj) |
| int solvedMap[9][9] | Vrednosti rešene mape |
| int indicationMap[9][9] | Vrednosti početne slagalice |
| int userInputMap[9][9] | Korisnički unos zajedno sa početnim vrednostima |
| int startX | Početna vrednost x koordinate |
| int startY | Početna vrednost y koordinate |
| int endX | Krajnja vrednost x koordinate |
| int endY | Krajnja vrednost y koordinate |
| int oldStartX | Prethodna početna vrednosti x koordinate |
| int oldStartY | Prethodna početna vrednost y koordinate |
| int oldEndX | Prethodna krajnja vrednost x koordinate |
| int oldEndY | Prethodna krajnja vrednost y koordinate |
| int odabraniBroj | Vrednost korisničkog unosa |

**Tabela 2. Pregled bitnih promenljivih**



**Slika 2. Konačni izgled Sudoku igre sa pokrenutom proverom ispravnog unosa**