

DOKUMENTCIJA IZ PREDMETA

RAČUNARSKA ELEKTRONIKA

TEMA PROJEKTA:

Implementacija igre “Potapanje brodova” na Raspberry Pi pločici

TEKST ZADATKA:

Implementirati igru “Potapanje brodova” na Raspberry Pi pločici uz upotrebu dodatnog hardvera (tastera i LCD displeja) na DVK pločici. Isprojektovati grafički korisnički interfejs korišćenjem QtCreator razvojnog okruženja.

Mentor:

Prof. Ivan Mezei

Student:

Nina Antić EE47/2020

Novi Sad, 4.9.2024.

Sadržaj

Uvod.....	3
Korišćene komponente.....	4
Raspberry Pi mini računar.....	4
DVK512 pločica sa dodatnim hardverom.....	4
LCD displej.....	4
Kod.....	5
Klasa BattleShips.....	5
Klasa Dialog.....	6
Tok programa.....	6
Grafički korisnički interfejs.....	7
Zaključak.....	9
Literatura.....	10

Uvod

Potapanje brodova je klasik društvenih igara na papiru koji je obeležio detinjstvo mnogih generacija, a neki ga i sad rado igraju u dokolici. Pravila imaju više varijacija, ali suština svih je ista: igraju dva igrača, svaki poseduje mrežu 10x10 gde smešta svoje brodove. Ima više različitih tipova brodova gde svaki zauzima određeni broj kvadratića. Prilikom postavljanja brodova oni se ne smeju dodirivati (različiti brodovi ne mogu zauzimati susedne kvadratiće već treba da postoji jedan "prazan" između). Kada oba igrača postave svoje brodove (njihov broj i dužina svakog tipa su prethodno dogovoren) počinje igra pogadanja. Igrači naizmenično govore koordinate za koje misle da se na njima nalaze protivnički brodovi. Igrač koji "pogodi" brod ima pravo da ponovo pogada. Igra se završava onog trenutka kada se svi brodovi potope.

Tok igre unutar ovog zadatka je takav da korisnik prvo postavlja brodove čija kombinacija tipova nije strogo određena, jedino je važno daje ukupno zauzeto 20 kvadratića. Pre početka igre moguće je izbrisati neki brod sa table (Undo) u slučaju da se želi drugačija pozicija ili izbor broda.

Igra čovek protiv računara, gde računar po nasumičnom algoritmu popunjava svoju mrežu i "gađa" korisnikove brodove. Svaku pogodak kvadratića boji se crvenom bojom, a promašaj crnom.

Od dodatnog hardvera koristi se RPi Hat (DVK512 pločica o kojoj će biti više reči u nastavku), tačnije taster i LCD displej na njoj. Pritisom na prvi taster (obeležen kao KEY0 na pločici) moguće je bilo kad u toku igre videti pravila igre, dok LCD displej prikazuje trenutno stanje broja života - preostalih kvadratića brodova korisnika i računara.



Slika 1 - LCD displeji i tasteri na DVK512 ekspanzionoj pločici

Korišćene komponente

Raspberry Pi mini računar

Raspberry Pi je *embedded* računar opšte namene. Poseduje u sebi sistem na čipu (eng. system on chip) BCM283x (x=5, 6 ili 7 zavisno od verzije *RPi*) proizvođača *Broadcom*. Možda najznačajnija osobina mini računara bila bi mogućnost povezivanja sa raznovrsnim spoljašnjim uređajima. Poseduje USB portove, *Ethernet* konektor, konektore za displej i kameru, kompozitni audio i video konektor i HDMI konektor. Takođe, poseduje 40-pinski GPIO port za povezivanje sa nestandardnim periferijama. Nema analognih pinova, te je potreban dodatni hardver - konvertor kako bi se omogućio rad sa analognim signalima. (Za više detalja o ograničenjima pri radu pogledati dokumentaciju).

Pored velikog broja interfejsa, postoji mogućnost pričvršćivanja dodatnih kartica (HATs) na vrhu GPIO porta, koji "nadograđuje" mogućnosti u pogledu hardvera.

U zadatku je korišćen model *Raspberry Pi 2*.

DVK512 pločica sa dodatnim hardverom

Kao dodatni hardver zadatka koršćen je *HAT* - ekspanziona pločica DVK512 koja ima različite ugrađene komponente na sebi (tastere, LED idr.) kao i konektore za povezivanje sa drugim komponentama.

Dodaci koje ova pločica nudi su:

- UART konektor za povezivanje preko RS232, RS485, itd.
- konektor za povezivanje SPI modula kao npr. AT45DBXX fleš memorije
- konektor za povezivanje I2C modula
- konektor za povezivanje sa LCD (npr. LCD1602)
- USB konektor za konverziju USB na UART, podržano preko CP2102 konvertora
- četiri korisničke LED
- četiri korisnička tastera
- potenciometar za podešavanje kontrasta LCD-a
- PCF8563: sat realnog vremena
- 32.768KHz kristal
- CP2102 čip (USB → UART konvertor)

LCD displej

LCD displej smešten je na posebnom konektoru DVK512 pločice. U pitanju je LCD1602 dvoredni displej koji može da prikaže 16 karaktera po redu. Radom displeja upravlja SPLC780D kontroler (baziran na *Hitachi HD44780* kontroleru) koji koristi 3.3V napajanje i ima plavo pozadinsko osvetljenje. Takođe, postoji mogućnost podešavanja kontrasta ekrana pomoću trimer potenciometra na DVK pločici.

Za programiranje rada hardvera korišćena je *wiringPi* biblioteka. U okviru nje postoje funkcije koje olakšavaju programiranje LCD ekrana, bez razmišljanja o komandama na nivou bita.

Kod

Algoritam igrice je prvobitno implementiran u C++ preko klase *BattleShips* i testiran pokretanjem koda u terminalu. Potom je ugrađen u Qt aplikaciju gde se uz manje modifikacije koristi kao logika igrice, dok klasa *Dialog* povezuje sve elemente grafičkog korisničkog interfejsa i način na koji interakcija korisnika sa elementima utiče na dalji tok igre. Napisane su dve klase (izvorni fajlovi *BattleShips.cpp* i *Dialog.cpp*) koje će biti objašnjene u nastavku. Za inicijalizaciju i očitavanje tastera korišćena je biblioteka *wiringPi*, dok je za korišćenje ispisa na LCD-u korišćeno zaglavlje *lcd.h* u okviru iste biblioteke.

Klasa *BattleShips*

Klasa se sadrži za podatke o igraču. Poseduje polja:

grid_matrix - niz matricu celobrojnih vrednosti koja oslikava trenutne vrednosti na mreži ;
lives - koja nosi podatak o broju života (broju preostalih brodova).

Slede funkcije i njihov opis:

- *bool placeShip(int x, int y, int shipType = 1, char orientation = 'h')*

Funkcija kojoj se prosleđuju početne koordinate broda, tip broda koji ujedno označava i njegovo zauzeće susednih celija u mreži i orientaciju koja može biti vertikalna ili horizontalna. Funkcija vraća vrednost true ukoliko je brod postavljen u mrežu, u suprotnom je false

- *bool isPlacable(int _x, int _y, int shipType, char orientation)*

Funkcija koja se poziva unutar *placeShip* funkcije. Na osnovu prosleđenih koordinata, tipa broda i orientacije proverava da li je moguće postaviti takav brod unutar mreže, te vraća true ako jeste i false u suprotnom.

- *void undo(int _x, int _y, int shipType=1, char orientation='h')*

Funkcija koja briše brod prosleđenih parametara iz mreže. Nema povratnu vrednost.

- *bool isHit(int x, int y)*

Funkcija koja na osnovu proseđenih koordinata vraća da li je polje matrice zauzeto od strane nekog broda, odnosno da li je brod pogoden.

- *bool randomGenShips()*

Funkcija koja nasumično raspoređuje brodove u okviru protivnikove mreže. Vraća vrednost true kada su brodovi uspešno postavljeni.

- *int getLives(), int getCellValue(int x, int y)*

Getter metode koje redom služe za dobijanje broja života igrača i vrednosti celije određenih koordinata

- *void restartGame()*

Funkcija koja restartuje igru, odnosno vraća sve parametre u početno stanje

Klasa Dialog

U okviru nje implementirano je sve ono što korisnik treba da vidi, promeni, pokrene. Izgled i elementi User interface-a biće predstavljeni u sledećem odeljku, dok će ovde biti objašnjeno kako pojedini događaji utiču na dešavanja u igrici.

Najznačajnija polja klase su:

*QPushButton **buttons_p, QPushButton **buttons_c:*

označavaju matricu dugmića tj. ćelija u okviru mreža igrača i računara

BattleShips player, BattleShips computer:

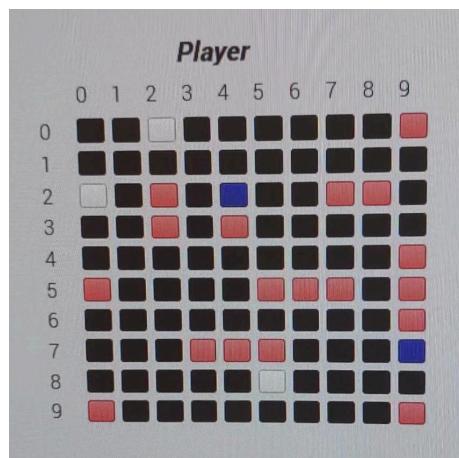
instance klase *BattleShips* koje čuvaju informacije o igračima

Tok programa

Kreiranjem objekta klase *Dialog* generišu se ćelije matrica 10x10 koje su predstavljene kao dugmići *QPushButton*.

Korisnik na početku markira ćeliju gde želi da mu budu početne koordinate broda, bira tip broda koji želi da postavi. Pritiskom na taster *Place Ship* prvo se proverava izbor tipa broda (na osnovu indeksa odabrane stavke iz *comboBox-a*), zatim se proverava da li je ćelija dostupna za postavljanje broda i najzad se brod postavlja u mrežu bojanjem dugmića u plavu boju. Postoji i opcija Undo koja briše odabrani brod iz mreže. Prilikom svakog postavljanja broda menja se broj preostalih ćelija koje možemo zauzeti (prema pravilu igre - 20). “Bitka” može započeti tek kad brodovi ispune svih 20 ćelija. Pritiskom na *START BATTLE* vrši se provera i startuje igra tako što se pojavljuje kartica sa mogućnošću unosa koordinata za pogađanje protivničkog broda.

Svakim pritiskom na dugme *Attack!* proverava se da li je u pitanju pogodak, dugmić se boji u crveno ili je reč o promašaju - dugmić se boji u crno. Ukoliko je brod “pogođen” moguće je ponovo birati koordinate, u suprotnom je protivnik na potezu. Protivnik - računar nasumično bira koordinate (funkcija *generateRandomNumbers* u okviru klase *BattleShips*), nakon čega se proverava stanje o broju preostalih brodova oba igrača što se ispisuje na LCD displeju (*lcdPrintf* funkcija).



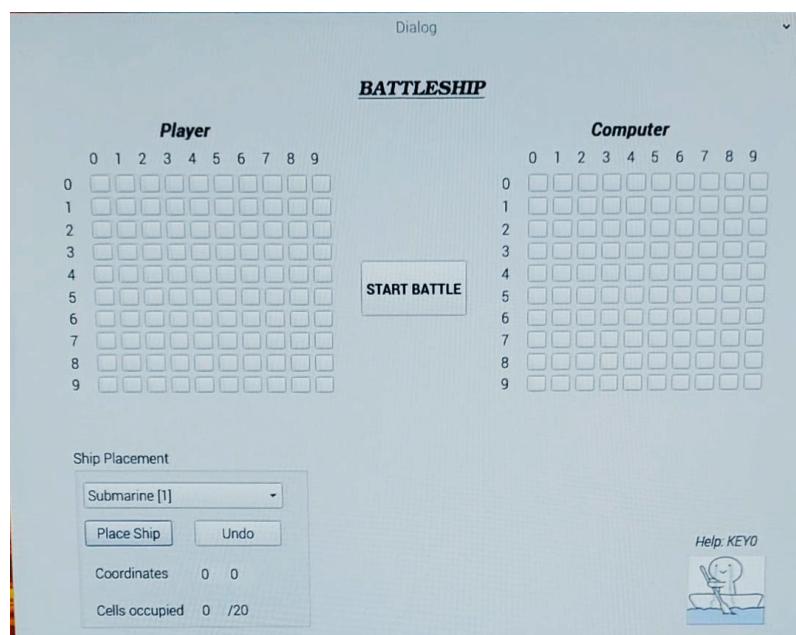
Slika 2 - Izgled igračeve mreže nakon više pogodaka i promašaja ćelija

Ako bar jedan od igrača izgubio sve živote prikazuje se informacija o pobedniku i iskače prozor gde se korisnik pita o započinjanju nove igre. U slučaju *Yes* resetuju se svi parametri i izgled dijaloga je isti kao na početku igre. U slučaju *No* blokiraju se sve kartice sa mogućnošću unosa i tada se može zatvoriti prozor.

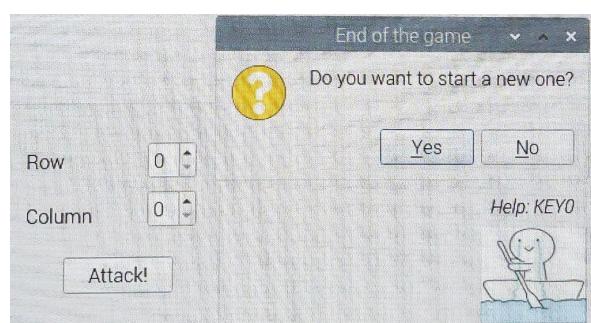
Postoji opcija da se u toku igrice može otvoriti prozor koji prikazuje pravila igre pritiskom na prvi taster DVK512 pločice. Realizovano je uz pomoć objekta klase *Timer* gde se stanje pina na koji je povezan taster očitava vrlo često nakon isteka kratkog vremenskog intervala koje je podešeno.

Grafički korisnički interfejs

U nastavku sledi izgled *User interface*-a kao i elementi koji su korišćeni.



Slika 3 - Izgled GUI-ja



Slika 4 - *QSpinBox* u okviru kartice sa leve strane i *QMessageBox* za dijalog sa korisnikom

Korišćene klase Qt elemenata:

- *QPushButton* - detaljno objašnjene funkcije dugmadi u prethodnom odeljku
- *QGridLayout* - radi lakšeg i preglednijeg pozicioniranja mreže dugmića
- *QVBoxLayout*, *QHoriyontalLayout* - radi lakšeg pozicioniranja indeksa redova i kolona
- *QGroupBox* - grupisanje funkcionalnosti - delova igre, postavljanje brodova i pogađanje
(Kartica za drugi deo igre pojavljuje se nakon startovanja, prikazana na slici 4)
- *QLabel* - nezaobilazne za lakše snalaženje i prikaz informacija o koordinatama u fokusu i broja preostalih brodova
- *QComboBox* - padajući meni, nudi moguće opcije za izbor brodova koje postavljamo
- *QSpinBox* - za unos koordinata reda i kolone, ograničava opseg unosa brojeva
- *QMessageBox* - za obaveštavanje, upozoravanje i pitanja upućena korisniku

Zaključak

Sve uvezši u obzir, može se reći da je projekat uspešno završen. Cilj zadatka bio je primeniti znanja iz predmeta Računarska elektronika kao i drugih njemu bliskih kako bi se hardver i softver osposobili za željenu aplikaciju.

Projekat bi se mogao unaprediti malo “življim” GUI-jem i dodatnim hardverom. Međutim, ovde je akcenat više na ispunjavanju funkcionalnosti i primeni osnovnih koncepata vezanih za *QtCreator*.

U prilogu se nalaze kodovi (osnovni c++ algoritam i *Qt* projekat), kao i demonstrativni video koji pokazuje kako prethodno opisano zapravo izgleda.

Literatura

- [1] Materijali sa predavanja i vežbi iz predmeta računarska elektronika:
<https://www.elektronika.ftn.uns.ac.rs/racunarska-elektronika/specifikacija/specifikacija-predmeta/>
(31.8.2024.)
- [2] Qt dokumentacija: <https://doc.qt.io/qt-6/>
(1.9.2024.)
- [3] Dokumentacija za Raspberry Pi:
<https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#raspberry-pi-2-model-b>
(1.9.2024.)