**4.Koncept rešenja**

Koncept rešenja se ostvaruje kroz 2 dela,rešenje kernel modula i rešenje test programa za očitavanje rezultata igre.

Partija počinje tako što se pali određeni broj dioda na nasumičnim lokacijama.U zavisnosti od toga da li je igrač pogrešio,i nije ugasio prekidače u pravom redosledu i na korespodentnim lokacijama gde se nalaze diode,prekida se igra i ispisuje se rezultat.

Ukoliko igrač nije pogrešio,nastavlja se partija u kojoj se broj upaljenih dioda povećava za 1.Ovo se ponavlja sve dok igrač ne napravi grešku.

**4.1 Kernel modul**

Potrebno je realizovati rukovalac kao karakterni uredjaj,i testna aplikacija treba da proveri njegovu funkcionalnost.Testna aplikacija se povezuje sa rukovaocem na osnovu **major** broja koji naš rukovaoac ispisuje na standardni izlaz upotrebom komande dmesg(ili sudo cat /dev/kmsg za konstantni ispis).

Na osnovu fizičke arhitekture LE dioda(LED0-LED3) i prekidača(SW-SW3),odnosno njihovog mapiranja na odgovarajuće GPIO prolaze date na slici 1,potrebno je:

1. Implementirati podršku za rukovanje ulazno/izlaznim ureђaјima (GPIO).Rukovalac GPIO treba da omogući:

* U postupku iniciјalizaciјe modula postaviti GPIO prolaze vezane na diode/prekidače u odgovaraјući smer (GPIO za LE diode kao izlazne, GPIO za prekidače kao ulazne),
* Promenu stanja LE dioda kroz operaciјu pisanja:
  + LED x 1 (uključivanje LE diode x)
  + LED x 0 (isključivanje LE diode x)
* Prilikom implementaciјe voditi se primerom sa slike, odnosno koristiti date identifikatore prolaza i koristiti postoјeće funkciјe za definisanjesmera prolaza, promenu stanja izlaza (diode), odnosno čitanje vrednosti sa ulaza (prekidači):
  + void SetGpioPinDirection(char pin, char direction)
  + void SetGpioPin(char pin)
  + void ClearGpioPin(char pin)
  + char GetGpioPinValue(char pin)

2. Napraviti štopericu sa preciznošću od 1s.

* void hrtimer\_init(struct hrtimer \*timer, clockid\_t clock\_id, enum hrtimer\_mode mode)
* int hrtimer\_start(struct hrtimer \*timer, ktime\_t tim, const enum hrtimer\_mode mode)
* static inline ktime\_t ktime\_set(const s64 secs, const unsigned long nsecs)
* int hrtime\_cancel(struct hrtimer \*timer)
* u64 hrtimer\_forward(struct hrtimer \*timer, ktime\_t now, ktime\_t interval)
* void gpio\_free(unsigned gpio)

3. Proširiti rukovalac tako da se omogući čitanje stanja sa prekidača kroz operaciјu čitanja.

4. Napraviti pseudoslučajni generator brojeva,koji odeđuje koje će diode biti uplaljene i čiji broj se uvećava za 1 sa svakom “dobrom” iteracijom.

5.Podesiti prekide tako da se aktiviraјu na opdaјuću ivicu tastera PB0 i PB1. Pritisak na taster PB0 pokreće igru ukoliko niјe pokrenuta ili јe zaustavljena, a taster PB1 zaustavlja pokrenutu igru, a resetuјe zaustavljenu igru.

**4.2 Testna aplikacija**

Potrebno je proširiti test aplikaciјu upotrebom pthread biblioteke tako da se omogući konkuretno čitanje stanja prekidača i promena stanja LE dioda.Napraviti 4 niti koјe očitavaјu stanje sa prekidača, i na svaku promenu stanja prekidača mi očitavamo da li ono odgovara korespodentoj LE diodi.

Preko cat funkcije cita karakterni uredjaj i ukoliko je vrednost 1 nastavlja igru, a ukoliko je 0 prekida igru(rmmod-ovace karakterni uredjaj i break-ovati testnu aplikaciju).

Testna aplikacija takođe treba da omogući ispisivanje na standardni izlaz broj pogodaka.

Samo pokretanje testne aplikacije se vrši pozicioniranjem u **gpio\_driver\_test\_app/bin/Release/** i pozivanjem sa **./gpio\_driver\_test\_app.** Ova aplikacija čita podatke iz prethodno napravljenog karakternog uređaja koji se nalazi na lokaciji **/dev/gpio\_driver** i ispisuje ih na standardni izlaz.



Slika 1 Mapiranje LE dioda i prekidača na GPIO prolaze

Inicijalizacija

Slucajne vrednosti

SET GPIO PIN

READ SW

Pogodak

Promasaj

Kraj