



# QUORIDOR su LANDTIGER

Carlo MIGLIACCIO s332937

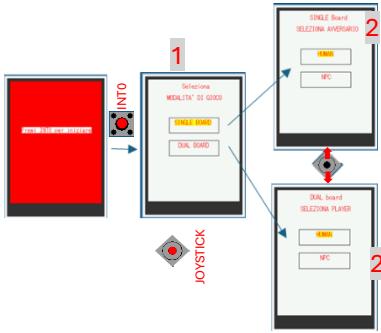
## APPLICATION NOTE

Obiettivo del presente documento è quello di spiegare brevemente – dando qualche dettaglio tecnico – le caratteristiche aggiunte all’implementazione del gioco **Quoridor** per la scheda **LandTiger**. Tale release, in particolare, amplia e rivede le feature di una “versione base” precedentemente sviluppata.

## 1) MENU

Permette di far scegliere all’utente:

- La modalità di gioco tra **SINGLE** e **MULTIPLAYER**;
- Il tipo di giocatore tra **HUMAN** e **NPC** (par. 3)



Le subroutine adibite alla gestione del menù, implementate sfruttando la libreria GLCD.h, sono:

❖ **void VisualizzaMenu(QualeMenu quale)** cambia, in base alla variabile **quale**, il tipo di schermata visualizzata: selezione *modalità di gioco* (1 in figura) o selezione *tipo giocatore* (2 in figura).

❖ **void CambiaEvidenziazione(int Cosa, int Quale)** usata insieme ai tasti direzionali del joystick per evidenziare la scelta dell’utente.

Dal momento che parte dei tasti del joystick sono utilizzati anche per scegliere le mosse, per differenziare le azioni da svolgere è stata introdotta una **variabile di stato** di tipo **STATO** che potesse monitorare l’attività del gioco (si parte in modalità **MODE**).

```
1 typedef enum {MODE, PLAYER_SEL, PLAY} STATO;
2 STATO StatoGioco=MODE;
```

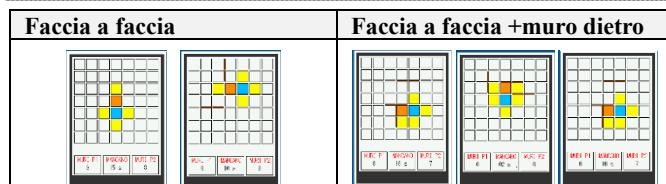
In particolare, le routine di gestione dei tasti UP, DOWN, SEL usano queste informazioni per decidere (tramite **switch-case**) quale compito indirizzare.

Le scelte (al passo 1 e 2) vengono invece salvate nelle variabili **game\_mode** (1) e **opposite/mine** (2) (a seconda della modalità scelta) e usate in modo opportuno dalle routine di gestione del gioco. Valore di init=-1.

```
1 typedef enum {SINGLE, DUAL} GAME_MODE;
2 typedef enum {HUMAN, NPC} PLAYER_TYPE;
3 GAME_MODE game_mode=-1;
4 PLAYER_TYPE opposite=-1;
5 PLAYER_TYPE mine=-1;
```

## 2) AGGIUNTA NUOVE MOSSE

*Nota: L’aggiunta di queste nuove mosse completa il set di regole del gioco originale<sup>1</sup>, solo parzialmente coperto nella versione precedente.*



Per poter supportare queste nuove mosse:

- ❖ È stata irrobustita la funzione **Check()**, che durante ogni turno evidenzia le mosse possibili, aggiungendo **nuove flag** (**UPRIGHT**, **UPLEFT**, **DOWNRIGHT**, **DOWNLEFT**) e controlli che limitano o garantiscono il movimento del token;
- ❖ Viene data la possibilità all’utente di muoversi con il joystick anche nelle **quattro direzioni diagonali** (quando consentito). A tale scopo l’ISR del RIT (timer che si occupa della gestione in polling del joystick) contempla i casi in cui due direzioni siano attivate contemporaneamente.

- ❖ Sono state aggiunte quindi le seguenti condizioni di verifica sui pin (attivo basso) della PORT1 associati alla periferica (vedi Schematici LandTiger<sup>2</sup>)

Nuova direzione	Condizioni da verificare (IRQ_RIT.c) <sup>3</sup>
UP RIGHT	(LPC_GPIO1->FICPIN & (1<<29)) == 0 && (LPC_GPIO1->FICPIN & (1<<28)) == 0
UP LEFT	(LPC_GPIO1->FICPIN & (1<<26)) == 0 && (LPC_GPIO1->FICPIN & (1<<27)) == 0
DOWN RIGHT	(LPC_GPIO1->FICPIN & (1<<25)) == 0 && (LPC_GPIO1->FICPIN & (1<<28)) == 0
DOWN LEFT	(LPC_GPIO1->FICPIN & (1<<26)) == 0 && (LPC_GPIO1->FICPIN & (1<<27)) == 0

Come ci si può aspettare sono state introdotte altrettante variabili di stato che gestiscono il polling delle quattro nuove direzioni.

**Problema riscontrato:** durante il test delle nuove funzionalità, si è notato che non sempre su tutte le schede si riesce a muovere facilmente il joystick nelle direzioni diagonali.

## 3) NPC (Player automatico)

È la situazione in cui si sceglie come *opponent* (*opposite*=NPC, *game\_mode*=SINGLE) o *proprio giocatore* (*mine*=NPC, *game\_mode*=DUAL) NPC; in questo contesto, la mossa in uno dei due turni viene fatta in modo automatico.

La funzioni preposte a tale compito sono:

- ❖ **void MossaNPC (\_typePLAYER\* whoPt)** genera in modo *pseudocasuale* uno tra i due tipi di mossa possibili (MOVE, WALL). A questo punto:
  - Se Modalità MOVE → Scegli Direzione
  - Se Modalità WALL → Scegli Centro Muro, Scegli Orientamento
- ❖ **void MossaNPC\_wrapper (void)** è una “funzione-involucro” che si occupa del cambio di turno, invoca **MossaNPC()**, controlla la vittoria e infine azzerà il Timer per far giocare l’opponent.

## 4) MULTIPLAYER MODE

Selezionata come modalità **DUAL BOARD**, le due schede prima che cominci il gioco si scambiano un **acknowledge** nel modo seguente:



Si aspettano 3s per rilevare l’ACK dell’altra scheda (viene usato a tale scopo il TIMER1 attivato dopo il SEL del joystick), si entra quindi nella modalità di scelta player, comincia infine la partita. Il gioco prosegue come di consueto, con qualche accorgimento:

- ❖ **Trasmissione mossa:** Quando viene fatta una mossa la si trasmette sul bus CAN<sup>4</sup> utilizzando il formato definito nelle specifiche (se ne occupa la funzione **void TrasmettitCAN\_Msg (int Mossa)** che usa **CAN\_wrMsg ()** di **CAN.c**)
- ❖ **Ricezione Mossa:** Il cambio di turno, oltre che dallo scadere del timer (TIMER0), viene gestito dall’ISR del CAN in cui:
  - 1) Leggendo il messaggio ricevuto, si esegue l’unpack della Mossa ricevuta e la si visualizza;
  - 2) Si ripristina il turno e nel caso in cui *mine*=NPC si esegue la mossa in automatico.

<sup>3</sup> <https://www.pololu.com/file/0J431/user.manual.lpc17xx.pdf>

<sup>4</sup> [https://it.wikipedia.org/wiki/Controller\\_Area\\_Network](https://it.wikipedia.org/wiki/Controller_Area_Network)