Università degli Studi di Salerno

UNIVERSITÀ DEGLI STUDI
DI SALERNO

Comunication Protocoll

Gruppo 2

Membro 1: Battipaglia Valerio
Membro 2: Caso Antonio
Membro 3: Dell'Orto Giuseppe Maria
Membro 4: De Stefano Sabatino

1 Protocoll

1. **Inizializzazione**: Il **Master** inizializza la struttura dati *Global_Struct_Master* con all'interno tutte le variabili lette dai vari sensori e aggiunge un campo checksum utilizzando la funzione *panda_comunication_init_checksum*. Questo passaggio garantisce l'integrità dei dati che verranno trasmessi.

2. **Trasmissione Master**: Il **Master** tenta di trasmettere *Global_Struct_Master* allo **Slave**. Per ogni ciclo di controllo si esegue una comunicazione, durante questa fase il master prova a trasmettere i dati un certo numero di volte:*MAX_NUMBER_OF_ATTEMPTS* .Questo valore nel caso in esame è stato impostato a 2 , questa scelta è stata fatta considerando che fra due comunicazioni non passa un tempo molto significativo e dunque si è preferito alleggerire il protocollo evitando di ritrasmettere troppe volte durante il singolo ciclo. Se la comunicazione non dovesse andare a buon fine durante un ciclo di controllo si incrementerà una variabile *counter_checksum* , se al prossimo ciclo si riscontrano nuovamente errori si andrà a disabilitare lo slave ipotizzando un malfunzionamento dello slave e/o del canale di comunicazione.Questo meccanismo di ritrasmissione aiuta a gestire eventuali errori di trasmissione e comportamenti incoerenti dello slave. Mentre se tutto dovesse andare a buon fine si passa al prossimo step.

3. **Ricezione Slave**: Se la trasmissione ha successo, lo **Slave** riceve i dati in Global_Struct_From_Slave. A questo punto verifica che il *checksum* associato ai dati sia corretto, se così non fosse lo Slave invierà un messaggio di **WRONG** al Master e si incrementa una variabile chiamata *counter_stop_master* . Se il checksum è valido, lo Slave fonde Global_Struct_Master e Global_Struct_Slave in Global_Struct. Questo passaggio consente di combinare i dati del master e dello slave in un'unica struttura.

4. **Trasmissione Slave**: Lo Slave tenta di trasmettere Global_Struct al Master. Se la trasmissione fallisce dopo *MAX_NUMBER_OF_ATTEMPTS* volte, lo Slave incrementa *counter_stop_master*. Se questo valore raggiunge una certa soglia significa che il Master è morto e dunque si riavvia il master e ci si porta nella fase di *Setup* aspettando che il master tenti di ricomunicare con lo Slave. Altrimenti, se tutto va bene lo slave resetta *counter_stop_master* a 0.

5. **Recezione del Master** : Se il master non riceve nulla in un certo periodo si presuppone che ci sia stato un problema e si incrementa *counter_checksum*. Se invece il valore ricevuto dovesse avere un *checksum* sbagliato o il messaggio contenga **WRONG** si ignora questo pacchetto ricevuto e si incrementa *counter_checksum* questo incremento indica che il canale è disturbato. Se il canale è disturbato per più comunicazioni di fila si presuppone che ci sia un problema sulla rete e si disabilita lo Slave per sicurezza e le comunicazioni cessano. Nel caso andasse tutto bene il master prende le informazioni da Global_Struct e le utilizza per la propria routine.

6. **Slave gestione di timeout**:Lo slave inoltre si accorge grazie alla comunicazione che il master è morto , infatti se non riceve nessun messaggio in 3 cicli di controllo ipotizza un malfunzionamento del master , lo disabilita e porta il rover in uno stato sicuro.

Il protocollo così descritto garantische le seguenti proprietà:

- **Gestione degli errori**: Il protocollo prevede meccanismi di ritrasmissione e di gestione degli errori, come il conteggio dei tentativi falliti e l'uso di checksum per garantire l'integrità dei dati.
- **Risposta agli errori**: Se si verificano errori ripetuti, sia il master che lo slave hanno la possibilità di passare a una modalità di setup o di stato degradato, rispettivamente. Questo può aiutare a prevenire ulteriori problemi e a mantenere la stabilità del sistema.
- **Deadlock Free**:Sia il master che lo slave hanno la capacità di capire se l'analogo è vivo e gestire di conseguenza la propria esecuzione.
- **Combinazione dei dati**: Il protocollo permette di combinare i dati del master e dello slave in un'unica struttura, facilitando la gestione dei dati.

2 Sequence Diagram

Per avere una maggiore comprensione del protocollo sono riportati i sequence diagram che mettono in risalto le situazioni descritte pocanzi.

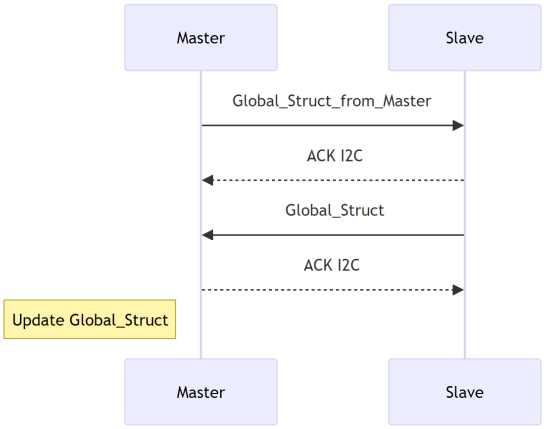

Figure 1: Situazione senza errori.

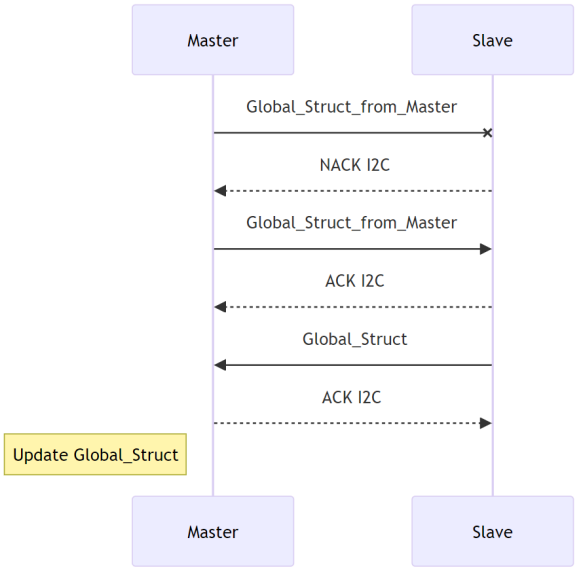

Figure 2: Master fallisce 1 trasmissione.

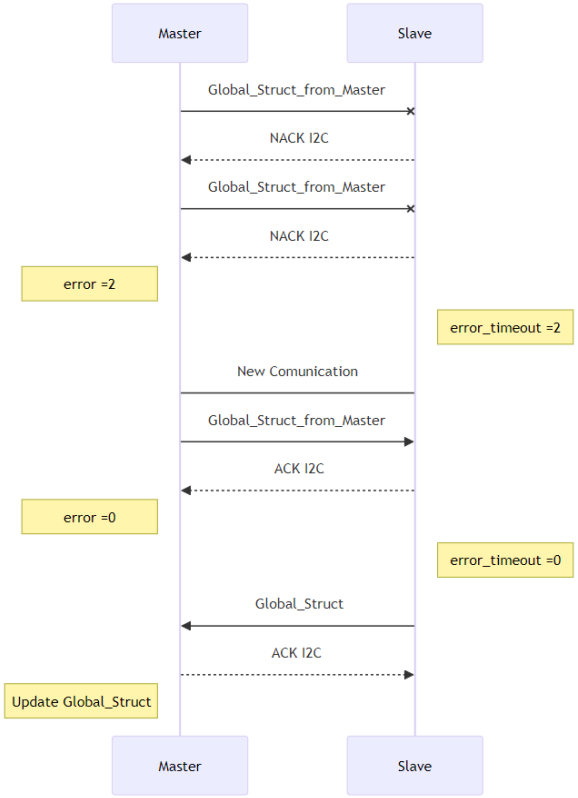

Figure 3: Master fallisce 2 trasmissioni.

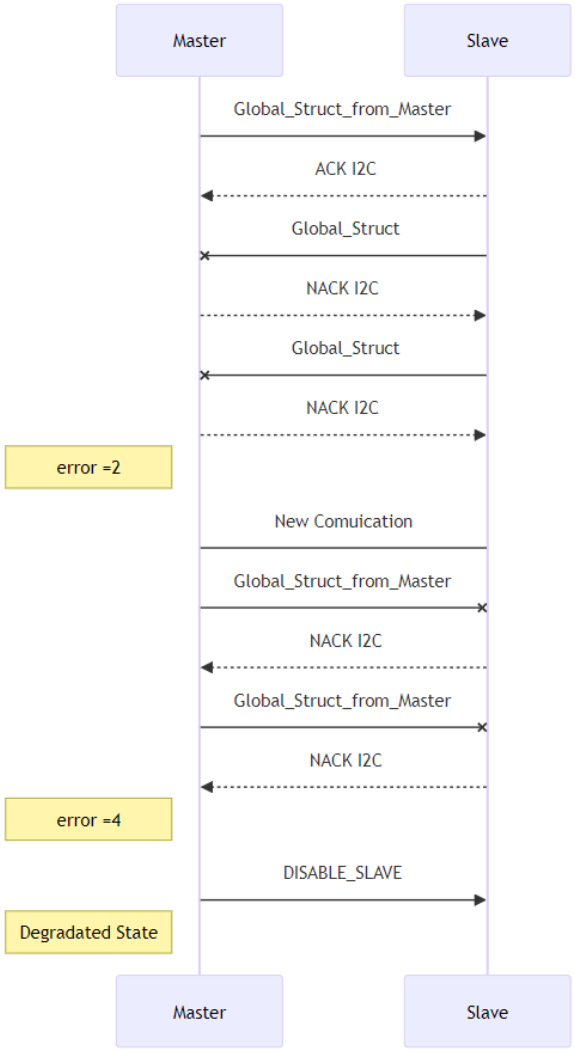

Figure 4: Fallite 3 trasmissioni Totali.

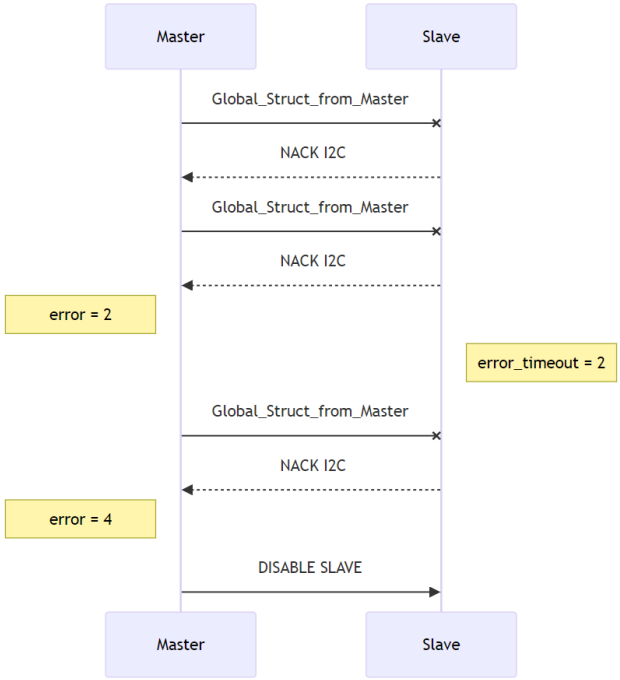

Figure 5: Canale distrurbato o slave rotto.

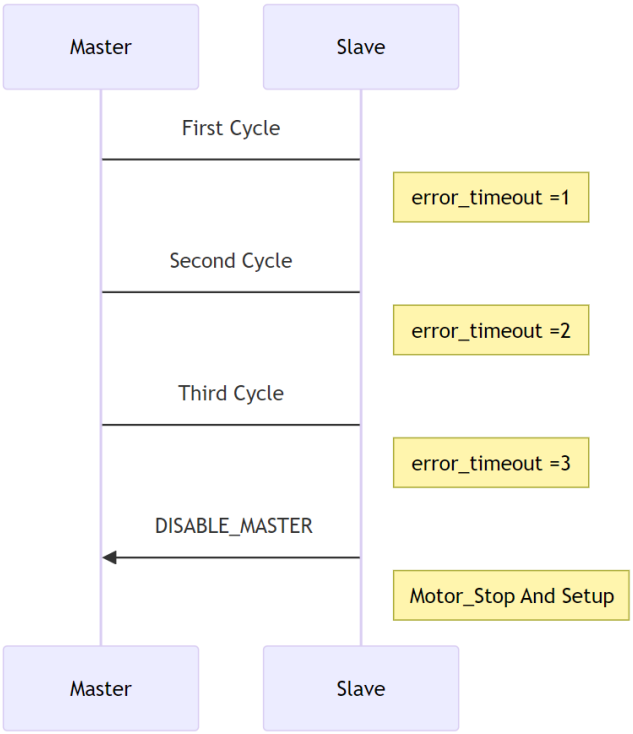

Figure 6: Master non comunica MAI.

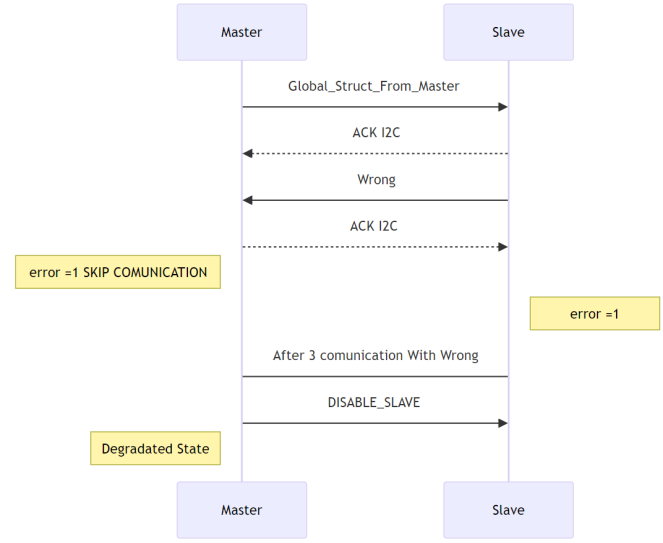

Figure 7: Checksum wrong.

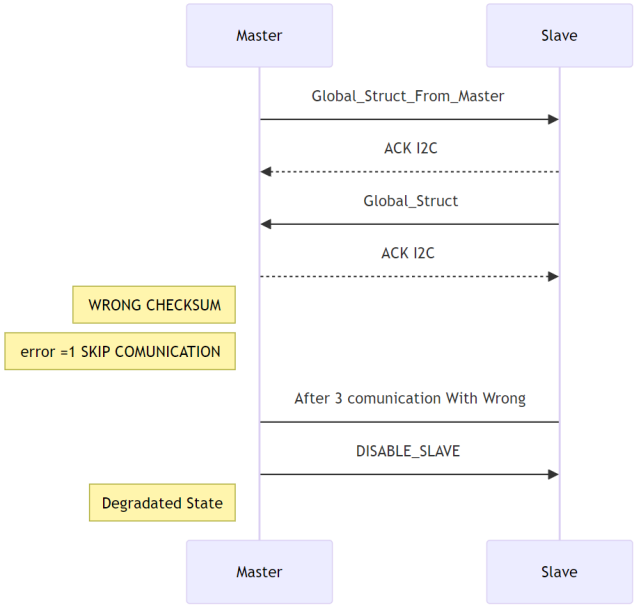

Figure 8: Checksum wrong.