Università degli Studi di Ferrara

DOTTORATO DI RICERCA IN

SCIENZE DELL'INGEGNERIA

CICLO XXVI

COORDINATORE Prof. Trillo Stefano

Nuove architetture di controllo distribuito per automazione di macchine da lavoro e agricole

Settore Scientifico Disciplinare ING-INF/05

Dottorando

Dott. Dian Massimo

Tutore

Prof. Ruggeri Massimiliano

Anni 2011/2013

1 SOMMARIO

2 PREMESSA

La tesi tratta di ricerche sviluppate durante il dottorato, svolto presso l'istituto di ricerca per le macchine agricole e movimento terra (IMAMOTER) del CNR. IMAMOTER è un centro di ricerca le cui ricerche sono incentrate su diversi aspetti relativi alle macchine agricole, macchine movimento terra e, in generale, ai veicoli heavy duty. Il personale di ricerca dell'Istituto è formato da gruppi di ingegneri e fisici specializzati nella meccanica, nello studio delle vibrazioni, sull'acustica e, in particolare, una eccellenza nazionale ed Europea per la parte di oleodinamica e di controllo elettronico dei componenti oleodinamici, ovvero della meccatronica legata alle applicazioni elettroidrauliche. Come nota storica si riporta che l'istituto IMAMOTER è nato come un istituto dedicato alla meccanica, seguendo le tecnologie legate alle macchine agricole e movimento terra, e in esso la tecnologia elettronica è stata introdotta negli ultimi quindi anni, e col tempo ha trovato il suo spazio e si è conquistata uno ruolo sempre più importante, riducendo la dipendenza delle applicazioni dalla sola parte meccanica e, al contempo, contribuendo a migliorare la precisione e il guadagno dei sistemi di controlli progettati. L'avanzare dell'elettronica, con la possibilità di utilizzare potenze di calcolo sempre superiori, ha aperto la strada alle tecnologie informatiche e ai controlli automatici, cioè alla possibilità di utilizzare logiche programmabili per ottenere controlli ancora più avanzati e validi per diverse applicazioni. Con l'avanzare delle tecnologie e dei protocolli di comunicazione si è aperta la prospettiva di poter integrare sistemi, originariamente concepiti come stand-alone, in sistemi a controllo distribuito, formati da più unità di controllo elettronico (ECU), ciascuna delle quali esegue una funzione specifica, aumentando la scalabilità e la riusabilità di ciascun componente.

Negli ultimi anni, con l'avanzare delle tecnologie wireless e i protocolli di rete wired utilizzate nelle reti ad alto throughput, sono stati resi possibili scenari di sistemi di ECU o macchine che funzionano in maniera collaborativa o cooperativa, in cluster, portando quindi l'automazione e automatizzazione di lavorazioni su campo a livelli solo pochi anni fa impensabili. L'aumento della automazione delle lavorazioni e la crescente complessità dei sistemi ha portato infine ad una attenta valutazione della componente di sicurezza funzionale che tali sistemi possono offrire.

Per meglio comprendere l'ambito delle ricerche e le tematiche trainanti, è utile ricordare che il personale dell'Istituto IMAMOTER è coinvolto attivamente in comitati internazionali e progetti che riguardano la sicurezza dei mezzi, sia dal punto di vista meccanico che elettronico. Inoltre è attivo in organismi di normazione per standard di sicurezza e di comunicazione come ISO – International Organization for Standardization, e SAE – Society of Automotive Engineering.

L'ambito delle ricerche sviluppate durante il triennio di dottorato è inquadrato all'interno dei trend di ricerca relativi alle tipologie di macchine studiate presso l'Istituto IMAMOTER. I trend di ricerca per le macchine del futuro ad elevate automazione si possono riassumere in alcuni macro-argomenti:

- La progettazione di architetture di sistema intrinsecamente sicure
- Lo sviluppo di tecnologie di fleet-management per una maggiore reliability dei sistemi e virtual fencing delle macchine per implicazioni di sicurezza
- Lo sviluppo di protocolli di comunicazione wireless real-time per il lavoro cooperativo di più macchine
- Lo sviluppo di protocolli di comunicazione wired scalabili ad alto thorughput e real-time per controllo distribuito di ECU di diversi produttori all'interno di un unico sistema
- La automazione delle macchine e dei cluster di macchine cooperative.

Dato il ruolo del dipartimento Sistemi di Produzione del CNR di cui IMAMOTER è parte, molto legato al mondo industriale, l'ambito di ricerca all'interno di IMAMOTER non può essere considerata ricerca di base, proprio per il mondo a cui si dedica, bensì è ricerca applicata e innovazione con lo scopo di ottenere tecnologie e prodotti affidabili e prontamente fruibili nel medio periodo dalle aziende.

Da queste premesse, il dottorato è stato svolto sia su ricerche industry-driven, cioè sviluppate per poter dare risposta a necessità contingenti di aziende che lavorano nel mondo off-road, sia su ricerche technology-driven, cioè l'applicazione di tecnologie esistenti per creare architetture valide per i sistemi del futuro (medio-lungo periodo). Del primo tipo di ricerca, cioè industry-driven, fa parte la ricerca su una architettura generale a sicurezza intrinseca per sistemi di ECU off-road, richiesta sempre maggiormente dalle industrie produttrici di macchine off-road, per poter rispettare i requisiti di sicurezza delle nuove normative vigenti. Del secondo tipo di ricerca, technology-driven, fa parte la ricerca per la realizzazione di un nuovo standard per protocolli di comunicazione ad alto throughput per il controllo distribuito all'interno di un sistema dinamico formato da ECU di diversi produttori.

Per comodità i due filoni di ricerca, il primo, con lo scopo di arrivare ad un'architettura fruibile e applicabile direttamente su delle macchine e il secondo, con lo scopo di arrivare ad un'architettura scalabile e validata per il futuro, sono stati trattati in due capitoli separati, anche se sono stati portati avanti parallelamente durante l'arco dei tre anni di dottorato.

Come si potrà vedere nel corso dei prossimi capitoli, ci si è focalizzati molto sullo studio di architetture di sistema, fondamentali nelle macchine off-road, piuttosto che su una visione specifica sul singolo componente: ciò è dovuto al fatto che ciascuna macchina off-road è il risultato di un'integrazione di sistemi

diversi di diversi produttori (system integration) per realizzare controlli molto avanzati. Si può pensare alle macchine del presente come a sistemi distribuiti su internet, dove ci sono diverse funzionalità e servizi, forniti da diversi produttori. La differenza principale è che mentre sulla rete globale si ha a disposizione farm-server, il real-time non è sempre richiesto, come per la safety, su una macchina off-road ci sono limitate risorse computazionali, è richiesto un alto determinismo e la sicurezza funzionale (functional safety) è fondamentale.

L'approccio architetturale diventa quindi fondamentale in questo contesto, oltre che la possibilità di creare uno standard di comunicazione intrinsecamente sicuro e scalabile, che possa sostituire la tecnologia in uso attualmente, mantenendone comunque la compatibilità. La ricerca si è concentrata su un attento studio di architetture e tecnologie già esistenti, ma utilizzate in ambiti applicativi differenti, che con gli opportuni adattamenti possono essere utilizzate per lo scopo finale, al fine di produrre un risultato applicabile utilizzando al meglio tecnologie provate e validate. Ciò è avvenuto sfruttando sia codice sorgente open source, sia protocolli già sviluppati per ambiti di applicazione diversi, ma convenientemente personalizzabili per lo scopo delle ricerche svolte durante il dottorato.

3 NUOVI PROTOCOLLI DI COMUNICAZIONE AD ALTO THROUGHPUT SU MACCHINE AGRICOLE E MOVIMENTO TERRA

3.1 Introduzione

Il mondo delle macchine agricole ha avuto, con l'avvento dell'elettronica, un grande sviluppo per quanto riguarda le nuove possibilità di controllo introdotte. Con la nascita della norma SAE J1939, che standardizza le comunicazioni fra ECU in macchine da lavoro, movimento terra, off-road e agricola per tutto ciò che concerne le reti powertrain, si è potuto iniziare ad avere uno standard solido per poter far comunicare diverse ECU di diversi produttori sulla stessa macchina. Un'evoluzione specializzata di questa norma è nata nel 2002, lo standard ISO 11783, che si basa sulla stessa architettura di basso livello della norma SAE J1939 (fino al livello ISO OSI di trasporto), ma con lo scopo di standardizzare tutte le applicazioni agricole, realizzando di fatto uno standard per eseguire lavorazioni e automazioni di precisione (precision farming) e il fleet management. A differenza di J1939, dove ciascuna ECU occupa un posto ben preciso all'interno della macchina operatrice, deciso dal costruttore della macchina (system integrator), la filosofia di ISO 11783 è di tipo plug-n-play, cioè supporta la interconnessione a run-time di ECU di diversi costruttori, che si auto-configurano all'interno della rete per poter eseguire le funzionalità per cui sono state progettate.

Con l'aumento e l'affinamento delle funzionalità di precision farming e del conseguente aumento della mole di dati scambiati per il corretto funzionamento del sistema e l'introduzione di nuove funzionalità, quali ad esempio la guida autonoma, e l'aumentare del numero di ECU all'interno di una rete di un moderno trattore, si è arrivati a saturare la banda disponibile sulla rete dei trattori più avanzati, di fatto raggiungendo il limite della tecnologia su cui tutto ciò si basa, cioè il bus CAN (Controller Area Network, specificato da Bosch Automotive nel 1992 con le specifiche CAN 2.0 A e 2.0 B che sono poi state recepite da ISO nello standard ISO 11898).

3.2 La norma ISO 11783 e il comitato ISO

La norma ISO 11783, che disciplina lo standard di comunicazione su macchine agricole e movimento terra, nasce nel 2002 all'interno del comitato ISO TC 23/SC 19, Agriculture Electronics. Per il suo sviluppo fu creato il Working Group 1, Mobile Equipment, che da allora ha espanso la norma portandola alla sua quarta revisione nel 2010. I membri del Working Group 1 appartengono sia al mondo dell'industria, rappresentando i maggiori costruttori internazionali di macchine agricole e attrezzi (come John Deere, Claas, Kverneland, AGCO, Dickey-John, CNH, Same Deutz Fahr, Grimme, Muller), sia al mondo dell'università e della ricerca e di enti di normazione nazionale, (come IMAMOTER, UniMORE, UNACOMA per l'Italia, Techinsche Universität München, DLG, CCI, VDMA per la Germania, OSU per gli Stati Uniti, NARO per il Giappone).

Questo standard, inizialmente era nato come “porting” dello standard SAE J1939 per macchine heavy-duty e off-road, definisce il protocollo e il layer fisico per la comunicazione tra Unità di Controllo Elettronico (ECU) di diversi produttori all’interno delle macchine, favorendo la cooperazione e il riutilizzo di componenti elettronici [1].

ISO/OSI		SAE J1939		ISO 11783	IEC 61162 (NMEA 2000)	ISO 15765 (KWP2000)
Application Layer	7	J1939 Part 7x, 81		ISO 11783 Part 6, 9, 10, 13, 14	NMEA2000	ISO 15765-3
Presentation Layer	6	J1939 Part 71		ISO 11783 Part 7, 11		
Session Layer	5			ISO 11783 Part 6, 7, 10, 14		
Transport Layer	4	J1939 Part 21	⊆	ISO 11783 Part 3, 6	IEC61162-3	
Network Layer	3	J1939 Part 31	⊆	ISO 11783 Part 3, 5	J1939 Part 31	ISO 15765-2 – J1939 Part 31
Data Layer	2	SAE J 1939 Part 21 ISO 11898	⊆	ISO 11898, ISO 11783 Part 2, 3	ISO 11898	ISO 11898
Physical layer	1	SAE J 1939 Part 1x	⊇	ISO 11783 Part 2		

Figura 3.1 Mappa basata su stack ISO/OSI delle normative sui protocolli di comunicazione su veicoli off-road

La norma J1939, seppur piuttosto completa nei livelli ISO OSI fino al livello di trasporto, non è mai riuscita a standardizzare delle applicazioni specifiche ad-interim, cosa che invece è stata fatta dalla ISO 11783, rendendo necessario l’ampliamento dei livelli di trasporto per grandi moli di dati. Nonostante abbiano preso strade parallele ma diverse, entrambe mantengono una completa compatibilità all’interno dello stesso layer fisico, su cui possono coesistere, assieme ad altri protocolli di comunicazione per la diagnostica (ISO 15765) e per le comunicazioni GPS (NMEA 2000).

3.3 Lo Standard ISO 11783

Prima di esporre quali siano le nuove funzionalità richieste e i nuovi requisiti delle reti agricole del futuro è necessario soffermarsi su alcuni aspetti specifici della normativa, per poterne capire i meccanismi principali, al fine di comprendere le decisioni prese durante la ricerca. Si possono considerare i prossimi capitoli un’introduzione, comunque non esaustiva, della normativa, che consta di migliaia di pagine, ma che comunque possono dare un’idea della complessità della rete e delle funzionalità definite all’interno dello standard.

3.3.1 La Struttura

Lo standard ISO 11783, ad ora, si divide in quattordici parti:

- ISO 11783 Part 1: General standard
- ISO 11783 Part 2: Physical layer
- ISO 11783 Part 3: Data link layer
- ISO 11783 Part 4: Network layer
- ISO 11783 Part 5: Network management layer
- ISO 11783 Part 6: Virtual terminal
- ISO 11783 Part 7: Implement message application layer
- ISO 11783 Part 8: Power train messages
- ISO 11783 Part 9: Tractor ECU
- ISO 11783 Part 10: Task controller and management information system interchange
- ISO 11783 Part 11: Mobile data element dictionary
- ISO 11783 Part 12: Diagnostics services
- ISO 11783 Part 13: File Server
- ISO 11783 Part 14: Sequence control

Ciascuna parte non copre un livello specifico ISO-OSI, anche se i titoli possono trarre in inganno. Per questo motivo la trattazione dei temi specifici verrà divisa in base ai tre temi fondamentali:

- Parte fisica che copre il livello 1, dando alcune informazioni sulla struttura fisica della rete
- Parte di comunicazione che copre quanto la norma definisce a proposito dei Livelli 2-4 della pila ISO OSI
- Funzionalità che copre i livelli 5-7, dove verranno descritte brevemente come lo standard definisce le funzionalità all'interno della rete.

3.3.2 Parte fisica

Le specifiche sulla parte fisica si trovano all'interno della parte 2 della norma. Il layer fisico si basa sullo standard CAN 2.0B descritto all'interno della norma ISO 11898, con alcune modifiche introdotte dalla norma SAE J1939.

La velocità della rete è di 250 Kbit/s, per poter ottenere un buon trade-off tra throughput, affidabilità e lunghezza totale di un segmento di rete che può arrivare fino a 40 metri.

Le terminazioni della rete, al contrario della norma ISO 11898 che prevede delle resistenze passive da 120 Ω, sono attive e sono fornite da dei circuiti detti TBC [2] (Terminated Bias Circuit). Questi circuiti hanno il compito di diminuire gli effetti del degrado dei segnali dovuti a disturbi esterni e presenza di molte ECU. Per questo motivo la struttura del bus di comunicazione è formata da 4 cavi twisted pair [3], su cui passano i segnali differenziali CAN_H e CAN_L sia le linee di alimentazione fornite dai TBC, TBC_PWR e TBC_GND. Vista la necessità di posizionare le terminazioni attive ai due estremi del segmento di rete e al contempo la necessità di essere hot-pluggable, vengono definiti dei particolari tipi di connettori, detti break-away, che si comportano in maniera diversa a seconda che siano o no collegati:

- Se sono collegati, fanno passare tutti i segnali verso l'esterno e permettono di fatto di allungare le reti
- Se sono scollegati attivano i circuiti di terminazione attiva.

Ciascun segmento di rete può ospitare al massimo 30 ECU, in modo da non degradare troppo il segnale e limitare al contempo la mole di dati che può viaggiare sul Bus.

3.3.3 Parte di comunicazione

Per l'accesso al canale di comunicazione ISO11783 definisce un subset di quello specificato dallo standard CAN 2.0b a 29 bit. L'accesso è sempre CSMA/BA (Carrier Sense Multiple Access With Bit Arbitration) basato su arbitraggio del campo ID 29 bit del CAN; la differenza principale sta nel fatto che, mentre su CAN si possono utilizzare tutti e 29 bit dell'ID per realizzare 2^{29} livelli di priorità e tipicamente il discriminante del contenuto informativo è contenuto in tutto il campo ID, su ISO 11783 vengono utilizzati solo i 3 bit di maggior peso, cioè i bit 29-27, in modo da poter utilizzare il resto dei bit per realizzare un tipo di comunicazione punto-punto o punto-multi punto.

Al momento sono utilizzate solo la priorità 3 per i messaggi di controllo, 6 per tutto il resto (si noti che a maggior numero corrisponde una minore priorità, per definizione del CAN).

Il meccanismo di comunicazione è basato sull'uso di indirizzi logici a 8 bit. Ciascuna ECU possiede un proprio indirizzo logico e può comunicare con altre ECU ponendo come indirizzo di destinazione il loro indirizzo logico.

Gli indirizzi disponibili sono 254, cioè dallo 0 al 253 (0xFD). L'indirizzo 0xFE è utilizzato come NULL_ADDRESS, cioè è l'indirizzo logico sotto cui vengono raccolte tutte le ECU che non sono riuscite a reclamare un indirizzo logico. L'indirizzo 0xFF è l'indirizzo globale o BROADCAST_ADDRESS, utilizzato per comunicare a tutti i nodi.

3.3.3.1 Address claiming

L'assegnamento di indirizzi logici avviene attraverso una procedura, detta di Address Claiming [4], operazione eseguita ad ogni accensione del sistema o all'inserimento di un nuovo nodo. La procedura è stata studiata affinché non ci sia bisogno di una ECU master che distribuisca gli indirizzi, bensì come una procedura distribuita, in modo da essere sempre applicabile. La procedura si basa sul fatto che ciascuna ECU possiede un proprio indirizzo preferito, impostato dal costruttore, in base alla specifica funzionalità svolta dalla stessa e inoltre possiede un identificativo di 64 bit, detto NAME [4] che la rende unica all'interno della rete. All'interno del NAME ci sono alcuni parametri di configurazione della ECU, come il codice costruttore (assegnato dall'organo SAE per tutto il mondo), il numero seriale ed altre informazioni sul tipo di ECU (per l'agricoltura, la marina, heavy-duty ecc.)

Visto che ciascuna ECU ha un proprio indirizzo predefinito, impostato dal costruttore, deve essere possibile poterlo cambiare nel caso ce ne sia un'altra che lo usa. Inoltre alla fine di tale procedura l'ECU dovrebbe memorizzare il suo nuovo indirizzo in memoria non volatile in modo che al riavvio successivo della rete non si debba rifare la stessa procedura.

Come già detto, la procedura inizia allo start-up o all'inserimento di un nuovo nodo. Ciascuna ECU manda il messaggio Address Claimed, reclamando il proprio indirizzo. Se due nodi inviano nello stesso istante il messaggio reclamando lo stesso indirizzo si potrebbe verificare un errore del CAN, poiché entrambi userebbero lo stesso ID CAN. Per questo motivo ciascuna ECU deve inserire un tempo di ritardo pseudo-casuale per diminuire la probabilità di collisione. Se entro 1,25 secondi nessuna ECU reclama lo stesso indirizzo, il nodo ha reclamato con successo l'indirizzo logico, che userà per tutte le comunicazioni successive.

Nel caso in cui due nodi reclamino lo stesso indirizzo, il protocollo definisce che quello con NAME minore ha la precedenza sull'indirizzo. Pertanto quello con NAME maggiore proverà a reclamare un altro indirizzo. Il tempo massimo di convergenza per questo algoritmo nel caso peggiore è pari a 1,25 secondi per numero di ECU di un segmento CAN, cioè 37,5 secondi per 30 ECU.

È disponibile anche un messaggio broadcast per richiedere lo stato dell'indirizzamento degli altri nodi. Utilizzando questo messaggio una ECU può richiedere un indirizzo libero senza dover risolvere alcun contenzioso con i nodi che hanno già reclamato l'indirizzo.

Ciascuna ECU che riesce ad ottenere il proprio indirizzo logico lo salva in memoria non volatile in modo da ridurre il tempo di setup della rete al prossimo startup, avendo già una configurazione stabile.

3.3.3.2 Comunicazione fra nodi

Il meccanismo di comunicazione punto-punto e punto-multi punto è realizzato attraverso l'utilizzo di indirizzi logici, il cui ottenimento è descritto nel capitolo precedente. Tali indirizzi possono essere considerati di livello 3 all'interno dello stack ISO/OSI, in quanto i messaggi codificati attraverso questi indirizzi possono passare attraverso bridge CAN.

I rimanenti 26 bit dell'ID CAN sono suddivisi in due parti:

- Source Address, di 8 bit, che indica l'indirizzo logico della sorgente del messaggio. Tale campo può essere nullo (cioè 0xFE) sono nel caso in cui il nodo non sia riuscito a reclamare un indirizzo e risponda al messaggio specifico che esegue la richiesta dei nodi non identificati. Non è consentito l'uso dell'indirizzo globale come sorgente di un messaggio
- PGN, di 18 bit, che indica il tipo di informazione contenuta all'interno del messaggio, oltre che la semantica di comunicazione.

Tabella 3.1 Struttura del campo ID di un messaggio ISO 11783

ID 25	ID 24	ID 23	ID 22	ID 21	ID 20	ID 19	ID 18	ID 17	ID 16	ID 15	ID 14	ID 13	ID 12	ID 11	ID 10	ID 9	ID 8
EDP	DP	PDU Format								PDU Specific							
PGN (18 Bit)																	

ID 7	ID 6	ID 5	ID 4	ID 3	ID 2	ID 1	ID 0	
Source Address (8 bit)								

Il PGN [5] (la cui struttura è rappresentata dalla **Errore. L'origine riferimento non è stata trovata.**) a sua olta può essere diviso nei seguenti campi:

- PDU (Program Data Unit) Format (8 bit): definisce il contenuto informativo del messaggio, cioè la funzionalità a cui deve assolvere o il tipo di messaggio. Inoltre definisce se il tipo di messaggio è broadcast, oppure unicast. Ci sono 256 PDU disponibili così suddivisi:
    - 0-239: Messaggi punto-punto
    - 240-255: Messaggi punto-multipunto.
- PDU Specific: ha valore diverso in base al valore del campo PDU Format:
    - PDU Format 0-239, detto PDU1: è l'indirizzo logico di destinazione del messaggio punto-punto. L'indirizzo logico di destinazione può essere GLOBAL_ADDRESS per trasmettere il messaggio in Broadcast
    - PDU Format 240-255, detto PDU2: diventa anch'esso parte della definizione del contenuto informativo del messaggio
- Extended Data Page e Data Page: hanno una duplice funzionalità:
    - Discernere il protocollo ISO 11783 dal protocollo ISO 15765 (Protocollo di diagnostica su macchine agricole)
    - Fornire la possibilità di un bit in più per raddoppiare il numero degli identificativi di messaggio disponibili.

Tabella 3.2 - Significato dei bit di Data Page ed Extended Data Page

EDP	DP	Tipo di messaggio
0	0	ISOBUS Page 0
0	1	ISOBUS Page 1
1	1	ISOBUS Reserved
1	0	ISO 15765-3

I PGN effettivamente disponibili, per le osservazioni fatte sopra, sono $2 * (240 + (16 * 256) = 8672$ PGN di cui alcuni già assegnati a identificativi di altre norme come la SAE J1939 e IEC 61162. 480 PGN indicano un contenuto informativo con semantica unicast o broadcast, a seconda dell'indirizzo di destinazione, gli altri 8192 sono PGN multicast.

3.3.3.3 Protocolli di trasporto

La rete CAN fornisce solo messaggi con contenuto informativo a 8 byte. Questo limite è insufficiente per le applicazioni agricole, soprattutto quelle che richiedono il trasferimento di ingenti moli di dati di configurazione (quali Virtual Terminal, Task Controller e Sequence Controller), oppure il semplice trasferimento di pacchetti di diagnosi.

J1939 definisce un protocollo di trasporto, detto Transport Protocol (TP) in grado di trasportare fino a 1785 byte in una sessione in una comunicazione punto-punto. ISO 11783 utilizza tale protocollo ma, per soddisfare le funzionalità descritte, ne definisce altri due:

- BAM: Broadcast Address Message, appena accennato sulla norma J1939, è stato ridefinito per esigenze specifiche.
- ETP: Extended Transport Protocol con cui si riescono a trasferire enormi moli di dati, superiori ai 100 MByte per sessione

Ciascun protocollo di trasporto ha un proprio PGN, e sia durante la fase di handshaking, sia nel contenuto trasportato è presente il PGN dei dati trasportati, affinché essi possano essere inviati ai livelli ISO/OSI superiori (le funzionalità) in maniera completamente trasparente.

In base al contenuto informativo di un PGN esso può quindi essere trasportato:

- Con un singolo messaggio se contiene 8 byte di dati
- Con un TP o un BAM se contiene da 9 a 1785 byte
- Con un ETP se contiene da 1786 a 117440512 byte.

Nei prossimi capitoli ci sarà una trattazione incentrata sui meccanismi di base dei vari protocolli di trasporto, per poter meglio comprendere le scelte fatte durante lo sviluppo del nuovo protocollo di comunicazione.

3.3.3.3.1 Transport Protocol

Il Transport Protocol [5], preso dalla norma SAE J1939, viene utilizzato per comunicazioni punto-punto da 9 a 1785 byte. Esso include la divisione in pacchetti di un unico PDU e il suo riassemblaggio, con alcune tecniche di controllo di flusso.

Vengono definiti due PGN per il protocollo di trasporto:

- TP_CM - Transport Protocol Control Message: che, a seconda del primo byte può essere
    - Request To Send (TP.CM_RTS): messaggio di controllo per iniziare una connessione peer to peer, ove il richiedente (o mittente) dichiara la dimensione in byte del messaggio che vuole inviare, il PGN, e il numero massimo di pacchetti per sessione (che deve essere almeno 1)
    - Clear To Send (TP.CM_CTS): messaggio di controllo inviato dal ricevente del protocollo di trasporto, utilizzato sia in risposta al RTS, sia fra una sessione e l'altra. Attraverso questo

messaggio il ricevente dichiara il prossimo frammento del messaggio che si aspetta di ricevere e il numero di messaggi della sessione

- End of Message Acknowledge (TP.CM_EOMA): messaggio di controllo inviato dal ricevente per indicare al richiedente la corretta ricezione di tutti i frammenti del messaggio;
- Abort (TP.CM_ABORT): messaggio di controllo che può essere inviato da qualunque delle due parti per annullare la comunicazione, in cui viene specificato il motivo (risorse insufficienti o time-out della comunicazione

- TP_DT - Transport Protocol Data Transfer da:
  - 1 byte per il Sequence Number, utilizzato per il riassemblaggio e riordinamento dei frammenti
  - 7 byte di dati, che rappresentano un frammento del messaggio completo

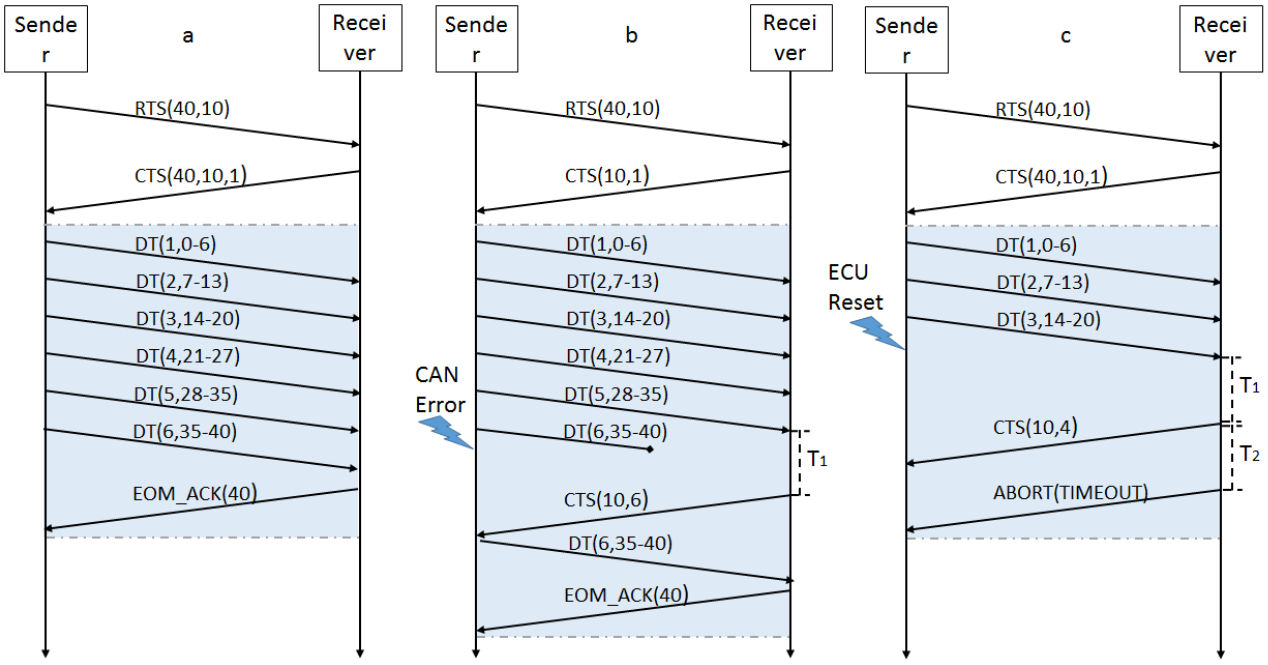


Figura 3.2 - Esempi di sessioni di TP, esempio "a" senza errori, esempio "b" con errore e ritrasmissione, esempio "c" con errore e Abort. In azzurro è evidenziata la parte in cui la connessione è considerata attiva

La connessione viene dichiarata instaurata quando all'invio di un RTS da parte di un nodo si ha la risposta di un CTS. Da questo momento il nodo mittente invierà i pacchetti rispettando il numero di frammenti per sessione imposto dal ricevente, che potrà mettere in pausa la connessione o dimensionare la sessione (o finestra) per attuare politiche di controllo di flusso. La chiusura della connessione può avvenire con un EOMA in caso di successo o un ABORT, in caso di fallimento.

Nel caso in cui ci siano errori di trasmissione (**Errore. L'origine riferimento non è stata trovata.**, esempio b) l protocollo applica una politica di ritrasmissione. Nel caso descritto dall'esempio l'ultimo pacchetto DT viene perso, pertanto il receiver, aspettato un Time-out T_1, invia un CTS richiedendo il pacchetto mancante. Il mittente ritrasmette il pacchetto richiesto e la sessione si conclude. Ci sono molti altri casi analizzabili, come ad esempio la perdita di un frammento all'interno della sessione e lo standard lascia liberi gli implementatori per quanto riguarda la strategia di ritrasmissione, che possono adottare politiche di recupero di tipo Go Back N o Selective Repeat, in base al messaggio CTS inviato.

Nel caso in cui ci sia un errore più grave, come ad esempio il reset della ECU o il suo scollegamento fisico dalla rete, (**Errore. L'origine riferimento non è stata trovata.**, esempio c) il protocollo cerca di riprendere la onnessione inviando il CTS, come nell'esempio precedente. Allo scadere di un secondo Time-out T_2, il ricevente ritiene la connessione fallita inviando un Abort per Time-out.

3.3.3.3.2 Broadcast Address Message

Un altro protocollo di trasporto definito all'interno della norma è il Broadcast Address Message [5] (BAM). Questo tipo di protocollo è unreliable e non esegue controllo di flusso, poiché serve per inviare un messaggio più lungo di 8 byte broadcast. È utilizzato ad esempio per l'invio di diagnosi periodico da parte di un nodo. Il numero di diagnosi attive non è predicibile a priori e il nodo è tenuto a pubblicare le proprie diagnosi per poter informare i nodi interessati del proprio stato.

Il BAM utilizza lo stesso PGN del TP, ma vengono utilizzati solo due tipi di messaggi:

- TP_CM - Transport Protocol Control Message:
    - Broadcast Address Message (TP.BAM): messaggio di controllo per avvertire tutti i nodi della rete che si sta iniziando una sessione di trasporto formata da N frammenti con un determinato PGN
- TP_DT - Transport Protocol Data Transfer da:
    - 1 byte per il Sequence Number, utilizzato per il riassemblaggio e riordinamento dei frammenti
    - 7 byte di dati, che rappresentano un frammento del messaggio completo

Non essendoci controllo di flusso la norma impone solo un tempo minimo tra un pacchetto e l'altro di 50 ms, per assicurarsi che il messaggio riesca ad attraversare il massimo numero di bridge consentito.

3.3.3.3.3 Extended Transport Protocol

La dimensione massima trasportabile attraverso TP è insufficiente per applicazioni che richiedono la configurazione completa di una ECU (ad esempio l'Object Pool, trattato successivamente). Per questo motivo ISO 11783 introduce un nuovo protocollo di trasferimento, detto Extended Transport Protocol [6] (ETP), in grado di trasportare messaggi fino a 117440512 byte. ETP è un super set di TP, in quanto utilizza lo stesso meccanismo di base, compresi i time-out.

ETP utilizza lo stesso PGN per i messaggi di controllo, utilizzando dei nuovi Control Byte per il trasferimento di messaggi più grandi di 1785 byte, mentre per il pacchetto di dati usa un PGN diverso:

- ETP_CM - Transport Protocol Control Message: che, a seconda del primo byte può essere
  - Extended Request To Send (ETP.CM_RTS): messaggio di controllo per iniziare una connessione peer to peer, ove il richiedente dichiara la dimensione in byte del messaggio che vuole inviare e il PGN
  - Extended Clear To Send (ETP.CM_CTS): messaggio di controllo inviato dal ricevente del protocollo di trasporto, utilizzato sia in risposta al RTS, sia fra una sessione e l'altra. Attraverso questo messaggio il ricevente dichiara il prossimo frammento del messaggio che si aspetta di ricevere e il numero di messaggi della sessione
  - Extended Data Packet Offset (ETP.CM_DTO): messaggio di controllo inviato dal richiedente al ricevente, all'arrivo di un ETP.CM_CTS, per definire l'offset di pacchetto da cui parte il Sequence Number dei messaggi TP_DT che seguono. In questa maniera il ricevente conosce l'offset assoluto del frammento sommando l'offset del ETP.CM_DTO al Sequence Number del TP_DT.
  - Extended End of Message Acknowledge (TP.CM_EOMA): messaggio di controllo inviato dal ricevente per indicare al richiedente la corretta ricezione di tutti i frammenti del messaggio;
  - Abort (TP.CM_ABORT): messaggio di controllo che può essere inviato da qualunque delle due parti per annullare la comunicazione, lo stesso utilizzato da TP
- ETP_DT – Extended Transport Protocol Data Transfer da:
  - 1 byte per il Sequence Number, utilizzato per il riassemblaggio e riordinamento dei frammenti
  - 7 byte di dati, che rappresentano un frammento del messaggio completo

Si può pensare a ETP come a un insieme di tanti piccoli DT. Inoltre la separazione dei due protocolli a livello di PGN per i dati e a livello di Control Byte per i messaggi di controllo permette l'istanziazione di due connessioni, una TP e una ETP, contemporanee fra due nodi.

3.3.3.4 Infrastrutture di rete

La norma definisce le seguenti infrastrutture di rete per l'interconnessione di diversi segmenti, dette Network Interconnect Unit [7] (NIU):

- Repeater: una ECU che ripete i messaggi su CAN. Può essere utile per aumentare la lunghezza massima del Bus, però non deve introdurre ritardi sul tempo di bit maggiori del 10%. L'abuso di Repeater può provocare il fallimento della procedura di Bit-Arbitration, dovuta ai ritardi di propagazione;
- Bridge: una ECU di interconnessione che immagazzina messaggi e fa il forwarding su una o più porte in base alle regole impostate. Essa può avere politiche di filtering su base ID CAN;
- Router: una ECU di interconnessione che estende il bridge, attuando politiche di instradamento basate su indirizzi logici di livello 3. Può eseguire politiche di traslazione di indirizzi logici e filtering dei PGN;
- Gateway: una ECU di interconnessione che è in grado di eseguire ripacchettizzazioni dei messaggi, oltre a tutto ciò che è in grado di fare il router;
- Tractor ECU: ECU di interconnessione tra rete Power-Train J1939 e rete ISO 11783. Oltre a fare da NIU, esegue molti altri compiti, descritti nei capitoli successivi.

3.3.4 Parte di applicazione

Lo standard ISO 11783 definisce diverse applicazioni standard. Alcune di esse possono essere viste come servizi, dove c'è uno o più master che possono fornire lo stesso servizio e uno o più client che fruiscono di tale servizio. Tipicamente i master sono a bordo del trattore, mentre le ECU all'interno degli *implement* (attrezzi) possono essere client di più servizi. Ciascun servizio viene definito Control Function. Nella norma vengono definiti i seguenti servizi o applicazioni:

- TECU
- Virtual Terminal
- Auxiliary Input/Function
- Task Controller
- Sequence Controller
- File Server

A questi si può aggiungere il GPS, non definito nella norma ISO 11783, bensì attraverso la norma IEC 61131 i cui messaggi possono co-esistere con una rete ISO 11783 ed essere utilizzati dalle ECU che richiedono tali informazioni.

3.3.4.1 Virtual Terminal

Il *Virtual Terminal* [6] (VT) è una delle prime applicazioni, assieme alla TECU, ad essere stata introdotta nello standard ISO 11783. Esso consiste in un'interfaccia uomo macchina formata da un display (negli ultimi anni anche Touch-screen, anche se ci sono problemi di sicurezza) e da un determinato numero di tasti funzionali riprogrammabili (detti Soft Key), che svolgono diverse funzioni in base alla schermata visualizzata. Tale display è montato all'interno del trattore e, attenendosi alla semantica del Plug And Play, portata avanti della norma ISO 11783, riesce ad identificare qualunque attrezzo che si connetta e rappresentarne graficamente le pagine di configurazione, visualizzazione e comando. Dalla terza revisione della normativa possono coesistere più Virtual Terminal, pertanto l'architettura di questa applicazione è di tipo Multi-Master Multi-Slave. Il master della comunicazione è il Virtual Terminal, mentre i client sono detti Working Set Master [8].

Figura 3.3 - Esempio di Virtual Terminal che visualizza l'Object Pool di uno sprayer

Alla connessione fisica o all'accensione ciascuna ECU Working Set Master montata su un attrezzo o a bordo del trattore si identifica sul Bus con la procedura di Address Claiming descritta nel capitolo **Errore. L'origine iferimento non è stata trovata.** e ricerca, attraverso un meccanismo di discovery basato su Request a Global Address il o i Virtual Terminal disponibili. In base alle preferenze espresse precedentemente dall'utente oppure per impostazioni di default ciascun Working Set stabilisce una connessione con un VT, tenuta viva da un meccanismo di Keep Alive. Una volta stabilita la connessione può richiedere le capability del Virtual Terminal (Touchscreen, dimensione della pagina grafica, numero di tasti riconfigurabili, lingue) e

inviare al master un Object Pool, cioè una struttura ad albero formata da oggetti vettoriali che descrivono tutte le pagine grafiche. Si può paragonare l'Object Pool a un insieme di file HTML in formato binario, dove ciascun TAG è un oggetto e ha un ID unico all'interno dell'OP, attraverso cui il Working Set Master riesce a riconoscerne l'interazione con l'utente. A differenza di HTML, gli oggetti sono descritti da strutture binarie e vengono inseriti all'interno di una pagina grafica attraverso puntatori, per ottimizzare l'occupazione e diminuire i tempi di setup, visto che la banda disponibile per il trasferimento rappresenta il collo di bottiglia. Un'interfaccia grafica complessa, con molte stringhe e oggetti raster può occupare centinaia di kilobyte.

Con l'evoluzione dei Virtual Terminal e della norma sono stati aggiunti nuovi tipi di oggetto in grado di rappresentare in maniera standard dati real-time o visualizzarli in formato grafico.

I tipi di oggetto presenti in un Object Pool si possono dividere in diverse categorie

- Oggetti contenitore: sono oggetti che contengono altri oggetti o puntatori che possono puntare ad altri oggetti, come le pagine grafiche (Data Mask), pagine di allarme (Alarm Mask), contenitori nascondibili (Container)
- Oggetti di interazione: pulsanti latched o non latched (Button), pulsanti riconfigurabili a lato del Virtual Terminal (Soft Key)
- Oggetti di input: campi di testo con o senza validazione (Input String), campi numerici validati (Input Number), checkbox (Input Boolean), combo box (Input List)
- Oggetti di output: numeri (Output Number), liste (Output List), stringhe (Output String), Barre a riempimento (Linear Bar Graph), Barre a riempimento circolari (Arched Bar Graph), contatori (Meter),
- Oggetti puramente grafici: linee, ellissi, immagini, poligoni, rettangoli

Come detto precedentemente attraverso Virtual Terminal si possono inviare comandi all'attrezzo e ciò introduce notevoli problemi dal punto di vista della sicurezza. Quando è stato concepito il Virtual Terminal, non erano ancora entrate in vigore normative che disciplinassero la sicurezza su sistemi di controllo distribuito in ambito agricolo e, ad oggi, la norma non fa alcuna differenza fra comandi che potrebbero essere Safety-Relevant da operazioni di inserimento dati all'interno di pagine di configurazione. Un altro problema sollevato all'interno del comitato WG1 sono le cosiddette Unattended Activation, cioè l'attivazione involontaria di un pulsante, più probabile in terminali Touch, che potrebbero armare un sistema elettro-meccanico o idraulico lato attrezzo, costituendo un rischio per la salute.

3.3.4.2 Auxiliary Input/Function

Attraverso il Virtual Terminal è possibile configurare un attrezzo e comandarlo, sia attraverso comandi impartiti direttamente dal terminale, sia configurando l'uso di interfacce di comando quali joystick, pulsanti, roller e leve presenti sulla trattrice, detti Auxiliary Input [6]. Anche queste serie di dispositivi di comando sono componenti a norma ISOBUS e sono dispositivi configurabili, associabili in modo completamente libero alle funzionalità dei diversi Implement che di volta in volta siano collegati alla trattrice. Tali Interfacce di comando azionabili dall'operatore, non sono altro che dispositivi di comando il cui azionamento genera un messaggio che è pubblicato sulla rete e che viene recepito dalla unità di controllo elettronico, che è stata associata a quel comando e che deve esplicitare l'azione di controllo corrispondente.

Figura 3.4 Auxiliary input montati su plancia completa di Virtual Terminal, su cui è visualizzata la schermata di assgnazione Auxiliary Input - Auxiliary Function

Si ha quindi una remotazione dei comandi in rete attraverso l'azionamento di dispositivi di comando azionabili manualmente, completamente configurabili attraverso una serie di pagine speciali residenti nel Virtual Terminal. In particolare esistono pagine nelle quali si può realizzare tale associazione, simili a quella mostrata in **Errore. L'origine riferimento non è stata trovata.**, dove nella colonna in giallo sono visibili le unzionalità della trattrice azionabili da remoto, e pagine di controllo e verifica globale dei settaggi realizzati.

Tale struttura costituisce un vero e proprio comando di tipo by-wire, perché pilota attuatori di tipo diverso attraverso messaggi inviati in rete CAN, nel caso particolare utilizzando il protocollo definito dallo standard per la trasmissione del comando.

In altri tipi di veicoli un sistema di questo tipo può utilizzare protocolli di tipo diverso, ad esempio CAN Open, ma la sostanza non cambia: si tratta di un sistema di pilotaggio di tipo Actuation By-Wire, nel quale una attuazione comandata direttamente è sostituita da un sistema di unità di controllo elettronico distribuite in rete e dislocate in punti diversi del veicolo, che si occupano di interpretare un comando e passarne il valore interpretato alla unità responsabile della attuazione.

Tale struttura, al di là delle problematiche di sicurezza che inevitabilmente devono essere affrontate, offre una grande versatilità e anche una semplificazione dei sistemi più complessi, oltre all'ovvio vantaggio immediato di non dover installare niente in cabina, a tutto guadagno della ergonomia, della sicurezza, della rapidità di installazione di un attrezzo e anche del colpo d'occhio.

3.3.4.3 Il controllo degli attuatori

Fino a ora si è parlato genericamente di attuatori, comprendendo ogni tipo possibile di comando impartito agli attrezzi da parte dei componenti di controllo e comando sulla rete ISOBUS. Non fanno eccezione le valvole elettroidrauliche e in particolare le valvole proporzionali ausiliarie di cui sono dotate tutte le trattrici agricole. Già nelle ultime versioni della normativa SAE J 1939 (versione 2006) erano comparsi i messaggi per la gestione delle valvole ausiliarie, proprio per l'avvento sul mercato di distributori e valvole a controllo elettronico via CAN. Lo standard ISO 11783 si è immediatamente adeguato a questa possibilità, introducendo la possibilità di pilotare le valvole ausiliarie della trattrice attraverso una serie di messaggi che possono essere impartiti sia dall'operatore che da attrezzi programmabili; tali implement sono in grado di regolare in autonomia il proprio funzionamento, definendo in modo automatico la portata idraulica desiderata o il livello di pressione necessario per il proprio corretto funzionamento.

Data la struttura della topologia delle reti di cui è dotata una trattrice agricola, i messaggi necessari al comando delle valvole ausiliarie della trattrice devono essere passati dalla rete degli attrezzi, nella quale è utilizzato il protocollo ISOBUS, alla rete Powertrain, dove invece sia adotta lo standard SAE J1939.

L'informazione di stato delle valvole effettuerà invece il percorso contrario, essendo generata dalle unità di controllo della attuazione e della regolazione della apertura delle valvole ausiliarie connesse alla rete Powertrain e dovendo essere passata alla rete ISOBUS, in modo da permettere all'operatore o all'attrezzo di verificare l'effettiva apertura delle valvole della quantità desiderata, come chiusura del loop di controllo.

Il passaggio delle informazioni è regolato da una speciale unità di controllo elettronico standard descritta dalla norma ISO 11783, fulcro del sistema, che è responsabile di compiti fondamentali per la sicurezza di marcia e di uso della trattrice, la cosiddetta Tractor ECU (Electronic Control Unit).

3.3.4.4 TECU

La particolarità delle Auxiliary Valve è quella di essere normalmente controllate sotto la rete Powertrain e come tali essere considerate "elementi di sicurezza", come quasi tutti i componenti che sono collegati alla rete Powertrain. Anche nel mondo delle trattrici si è adottata la filosofia già standardizzata nel mondo Truck & Bus: la rete Powertrain è una rete di sicurezza in cui ogni unità elettronica afferente ad essa deve essere approvata dal costruttore; la rete non può essere modificata in alcun modo rispetto alla sua struttura di progetto dopo l'uscita del veicolo dal luogo di produzione. Nel mondo Truck & Bus è addirittura nato un protocollo (Protocollo FMS) che vede la creazione di un gateway - detto centralina FMS - alla quale deve collegarsi chiunque voglia osservare dati relativi alla rete Powertrain e/o inviare messaggi su detta rete. A nessuno è quindi consentito aggiungere unità sulla rete Powertrain anche per sola osservazione dei dati circolanti.

Anche nel mondo delle trattrici e anche nel protocollo ISOBUS è stato previsto un gateway per proteggere la rete Powertrain dal traffico generato dagli attrezzi che potrebbe compromettere il buon funzionamento della rete che è responsabile della marcia della trattrice. Tale gateway è la Tractor ECU, normalmente detta TECU, che è a tutti gli effetti un filtro delle richieste del mondo delle applicazioni al mondo Powertrain.

La TECU è il fulcro della rete, poiché è responsabile della gestione dell'alimentazione sia di tutte le ECU attaccate al BUS sia di tutti gli attuatori elettrici per un minimo di 65 A. Attraverso la TECU, tutte le ECU possono accedere a informazioni del trattore come unità di misura, lingua, numeri seriali, ore di lavoro, velocità, stato del sollevatore e possono richiedere l'alimentazione anche durante lo spegnimento del trattore da chiave per eseguire operazioni di salvataggio dati.

Essa è standardizzata in tre versioni, ciascuna delle quali si adatta alla classe della trattrice su cui è istallata. Le classi si intendono come livelli di automatizzazione delle lavorazioni e delle funzionalità effettivamente presenti ed erogabili dalla trattrice. Trattrici di gamma alta, capaci di guida autonoma o semiautonoma, con possibilità di pilotaggio da remoto di tutti i dispositivi di trasmissione della potenza e di regolazione del

funzionamento e della marcia, saranno dotati di una TECU in grado di pilotare e controllare tali caratteristiche.

Nello specifico, la TECU, o Standard Tractor Network Interconnection Unit, è responsabile del trasferimento di dati e comandi tra la rete Powertrain e la rete ISOBUS nei due sensi, in base alla classe della applicazione. Un attrezzo collegato, può richiedere attraverso un'operazione di discovery quali capability la TECU fornisce e la sua classe.

Ogni classe identifica un minimo set di messaggi e funzionalità che la trattrice deve fornire per permettere la corretta connessione degli implement all'Implement Bus (ISOBUS) e il loro corretto funzionamento. Ogni costruttore di trattrici ha poi la facoltà di completare il set di messaggi con messaggi appartenenti alla classe superiore.

Per fornire una panoramica completa delle possibilità offerte dal protocollo ISOBUS nelle diverse versioni conviene riportare la lista delle funzionalità offerte dalle diverse classi di Trattrici.

3.3.4.4.1 Tractor ECU di Classe 1 – Basic Functionality & Status

La TECU di Classe 1 è da intendersi per trattrici già in campo e ne è sconsigliato l'utilizzo in trattrici di nuova progettazione; tale componente è definito nello spirito di riportare sulla rete attrezzi solo le informazioni di base dello stato della trattrice, per collegare ad essa un attrezzo in classica configurazione stradale o poco più.

Sono trasferiti dalla rete Powertrain alla rete attrezzi i messaggi contenenti le informazioni di:

- velocità delle ruote e velocità rispetto al terreno,
- il regime motore,
- velocità della PTO posteriore e suo stato (connesso/disconnesso),
- Lo stato delle luci di base (quelle per la marcia stradale) della trattrice e delle frecce,
- la lingua con cui sono visualizzabili i dati della TECU,
- lo stato della chiave.

L'unica effettiva richiesta che può venire dal lato Implement, è la cosiddetta Maintain Power Request, ovvero la richiesta di mantenimento della potenza elettrica alla rete attrezzi e all'attrezzo, per permettere la chiusura di tutte le operazioni di cui è responsabile l'elettronica dell'attrezzo, in caso di richiesta di spegnimento della macchina da parte dell'operatore.

3.3.4.4.2 Tractor ECU di Classe 2 – Advanced Tractor Status

Oltre a tutte le informazioni di base già elencate per la classe 1, la TECU di classe 2 fornisce anche informazioni addizionali sulla Marcia della trattrice:

- Distanza percorsa e direzione calcolate sulla base dei sensori alle ruote,
- Distanza percorsa e direzione calcolate sulla base dei sensori relativi al terreno,
- Lo stato di tutte le luci della trattrice,
- Lo stato delle valvole ausiliarie.

Le informazioni sulle valvole ausiliarie sono disponibili solo allo scopo di monitorarne lo stato, non è possibile però comandarne il funzionamento. Per ogni valvola ausiliaria (fino a un massimo di 64 valvole) sono pubblicati sulla rete:

- La portata misurata o stimata in uscita o in ingresso,
- Lo stato della valvola (block, extend, retract, float)
- La pressione alle porte (Extend / Retract / Return),
- Il Failsafe Mode (Block/Float)
- Rear Draft (stimato),
- Nominal Implement Position.

3.3.4.4.3 Tractor ECU di Classe 3 – Implement Commands

La classe 3 è la più completa ed è anche quella di maggior interesse per il controllo attivo delle valvole proporzionali per via remota, attraverso la rete CAN. Oltre a tutti i messaggi presenti nelle classi 1 e 2, nella classe 3 sono presenti i messaggi di comandi degli Implement, che possono comandare gli attuatori presenti nella trattrice, qualora la TECU ritenga che lo stato di marcia e di funzionamento della trattrice lo consentano senza che si ingenerino condizioni di pericolo.

Nello specifico sono presenti i messaggi per il controllo della posizione dell'attacco a tre punti posteriori, e i comandi della Presa di Forza posteriore della trattrice:

- attivazione/disattivazione della PTO,
- osservazione della velocità della PTO in RPM (1/min),
- modalità di funzionamento richiesta, ovvero richiesta di funzionamento a 540/540E/1000/1000E RPM.

Per quanto riguarda le Valvole Ausiliarie (EHR) proporzionali, invece sono presenti I messaggi:

- di portata desiderata,
- di variazione dello stato (block, extend, retract, float),
- di comando del Failsafe Mode (block/float).

Esistono poi altere due classificazioni che sono degli Addendum uno per il controllo automatico di Hitch e PTO e l'altro per I sistemi di navigazione basati su GPS per i sistemi dotati di sistemi di autoguida.

Quindi il controllo "remoto" della trattrice può essere eseguito su:

- Speed Control della macchina attraverso un sistema di Cruise Control comandato dagli attrezzi,
- Controllo del motore diesel in velocità (minima limite) o in coppia,
- Un sistema di Slip Control che può agire sui componenti:
  - Rear/Front Hitch
  - Rear/Front PTO governate a velocità costante o in coppia
  - Auxiliary Valve (EHR)
- Un controllo di Massimo Draft sulle Valvole Ausiliarie.

Dal punto di vista della sicurezza, la possibilità che un attrezzo possa automaticamente attivare la PTO oppure pilotare la velocità del trattore o delle valvole ausiliarie porta a grandi problematiche sia nel caso in cui l'attrezzo erroneamente comandi una di queste funzionalità rappresentando un pericolo per l'uomo e per l'ambiente. Allo stesso tempo, queste funzionalità portano un'altissima automatizzazione delle lavorazioni agricole, basti pensare ad un aratro od ad una seminatrice che regola la velocità del trattore e la velocità della PTO per ottimizzare la lavorazione per ottenere una migliore aratura in base alla resistenza del terreno od una semina più uniforme.

3.3.4.5 Task Controller

L'agricoltura di precisione, oltre ad essere data da elevate automazioni a livello attrezzo, è basata sulle lavorazioni guidate attraverso GPS e mappe di prescrizione, che permettono una precisione di lavorazione del singolo metro quadro o della singola fila. Nella norma ISO 11783 questa funzionalità viene standardizzata dall'applicazione Task Controller [9]. Il Task Controller è l'equivalente del Virtual Terminal per quanto concerne l'agricoltura di precisione, poiché anch'esso è in grado di riconoscere gli attrezzi collegati e di configurarsi in base alla struttura degli stessi, calibrando i comandi inviati in base alla dimensione dell'attrezzo, alla posizione degli attuatori e dell'attrezzo stesso. Inoltre la norma standardizza il formato delle mappe di prescrizione, per poter permettere al Task Controller di incrociare i dati delle mappe con le strutture dati provenienti dagli attrezzi per eseguire compiti di agricoltura di precisione.

Figura 3.5 - Esempio di task controller sviluppato per SAME, con mappa di prescrizione per defogliazione su vigneto

Il Task controller tiene conto del posizionamento GPS e sulla velocità del trattore per inviare i comandi e raccogliere i dati da tutti gli attuatori per poter creare parcelle parziali o totali dei quantitativi di prodotti utilizzati. Con questa ottimizzazione si hanno risparmi di costi sia per l'ambiente sia per l'agricoltore, senza tenere conto dell'aumento di produttività del terreno che può essere lavorato ad un'elevata precisione. La potenza e la novità dello strumento sta nel fatto che supporta una semantica del Plug-n-Play: l'agricoltura di precisione esiste già da alcuni anni, come molti programmi di Farm Management, oppure terminali GPS, ma uno strumento che riesca a coprire le necessità di ciascuna applicazione e al contempo essere standard e retro-compatibile rappresenta una novità assoluta nell'ambito applicativo agricolo.

Il procedimento con cui gli attrezzi si identificano è lo stesso descritto nel capitolo del Virtual Terminal, cioè ciascuna ECU che implementa la funzionalità client esegue una fase di address claim, una fase di discovery ed una di configurazione, durante la quale viene inviata una struttura dati ad albero formata da oggetti detti descrittori. Tale struttura è definita Task Controller Object Pool (TCOP), ove sono contenuti diversi tipi di oggetti e descrittori, sia grafici sia funzionali.

In base alle funzionalità fornite il Task Controller Master (TCM) si divide in tre classi, ciascuna delle quali comprende le funzionalità della classe precedente:

- Classe 1: Il Task Controller Master può raccogliere i dati di lavorazione e salvarli su memoria non volatile per statistiche o fatture
- Classe 2: Il Task Controller Master è equipaggiato di GPS, o associato a un GPS esterno su rete ISOBUS, ed è in grado di leggere i file delle mappe di prescrizione, scaricabili attraverso supporti quali chiavette USB e utilizzarli con gli attrezzi opportuni
- Classe 3: Il Task Controller Master è in grado di gestire i client utilizzando la cosiddetta Section Control, o controllo di sezione. Attraverso questo meccanismo il TCM tiene conto delle aree già lavorate o lavorate parzialmente, ed è in grado di attivare e disattivare parti degli attrezzi, dette sezioni, come ad esempio parti della barra che alimenta gli ugelli di uno sprayer, utilizzando le informazioni contenute nel TCOP.

Se da un certo punto di vista, la possibilità di poter controllare ogni singolo attuatore in maniera standard e riconfigurabile porta dei vantaggi, da un altro punto di vista esistono anche degli svantaggi dovuti alla mole di traffico generata, che dipende sia dal numero degli attuatori, sia dal fatto che ogni messaggio CAN può comandare un solo attuatore. Si pensi ad applicazioni che prevedono uno o più sprayer a catena di decine di metri di larghezza con decine di attuatori (un iniettore ogni 15 – 20 cm su barre di 40 o 50 metri). Oltre a problemi dovuti alla lunghezza della rete, si rilevano oggi anche problemi dal punto di vista del throughput del layer fisico che risulta insufficiente, argomento che verrà approfondito nei capitoli successivi.

3.3.4.6 Sequence Controller

Il Sequence Controller è l'ultima funzionalità nata nel 2010 all'interno dello standard ISO 11783, e, al contrario delle funzionalità di Virtual Terminal e Task Controller, è completamente nuova. Il Sequence Controller permette la registrazione, modifica e ripetizione di sequenze manuali eseguite dall'utente su diversi tipi di attuatori, che vanno dagli attuatori dell'attrezzo fino allo sterzo, l'acceleratore, i freni e le valvole ausiliarie del trattore. L'esempio tipico di questa funzionalità è la manovra di fine campo in cui l'operatore deve rallentare, alzare il sollevatore, spegnere la PTO o eventuali valvole ausiliarie che pilotano i motori idraulici dell'attrezzo, iniziare la manovra di fine campo, riallinearsi alla fila successiva e ripetere al contrario tutte le operazioni sopra elencate. Queste sequenze richiedono molta esperienza da parte dell'operatore e molto spesso sono, proprio perché eseguite da operatori, eseguite in modo non corretto e poco efficiente. Il Sequence Control può registrare e modificare parte o totalità di queste azioni per poterle poi ripetere all'occorrenza, esattamente come delle macro di un programma informatico.

3.4 Motivazioni della ricerca

Grazie ad IMAMOTER e a CUNA (Commissione tecnica di unificazione nell'autoveicolo) si ha avuto la possibilità di poter partecipare attivamente ai meeting ISO del TC23/SC19 Working Group 1 – Mobile Equipment, incaricato della scrittura e manutenzione della norma ISO 11783. All'interno di questo Working Group sono nati gli input e gli stimoli per la ricerca del presente Dottorato, costituiti dalle nuove necessità dell'ambiente agricolo - per poter raggiungere livelli e obiettivi di agricoltura di precisione sempre più elevati nel medio-lungo periodo e migliorare sia la qualità e la uniformità dei prodotti sia la redditività delle attività legate alle lavorazioni agricole.

In particolare, ad una riunione del 2010, sono sorte richieste da parte di alcuni membri di nuove funzionalità quali:

- Sicurezza dei comandi su CAN, derivante dalla armonizzazione della norma ISO 25119 (Part 3 - Annex B) che disciplina la sicurezza nei sistemi di comunicazione che inviano comandi rilevanti per la sicurezza,
- Maggiore frequenza di ripetizione dei comandi degli Auxiliary Input, che ad ora possono inviare comandi ad una frequenza massima di 5 Hz, insufficienti per alcuni tipi di applicazione, automatizzata,
- Maggiore frequenza dei comandi per i Task Controller Client per una maggiore precisione delle lavorazioni e per la riduzione della dimensione della area minima di controllo,
- Maggiore throughput per aggiornamento firmware direttamente da Bus fisico,
- Maggiore connettività verso server esterni per scopi di Fleet Management e raccolta dati di lavorazione, oltre che per poter inviare via Wireless o rete dati Cellulari i dati di lavorazione e le mappe di prescrizione,
- Necessità di autenticazione per operazioni rischiose, la cui responsabilità in caso di danni a persone o cose non è ancora ben definita dalle normative di sicurezza, come ad esempio i comandi al trattore di TECU Classe 3,
- Possibilità di avere flussi video standardizzati all'interno della rete e sovrapposti ai normali segnali di controllo ISOBUS, sia per il controllo di zone cieche durante le manovre sia per il controllo visivo da parte dell'operatore della lavorazione in run time.

Tutte queste nuove funzionalità si scontrano con uno dei colli di bottiglia sollevato proprio all'interno del Working Group 1: il layer fisico, cioè il Bus CAN a 250 kbaud. Tale bus infatti si sta rilevando insufficiente per la gestione di trattori complessi che comprendono tutte le funzionalità sopra descritte, in quanto la sua banda viene completamente saturata, comportando il non invio di alcuni pacchetti. Si

ricorda che a pari priorità e funzione (indicata dal valore del PGN), in caso di occupazione totale del Bus sono inviati i comandi che hanno rispettivamente Destination Address e Source Address più bassi, provocando lo Starvation di comandi ad uguale priorità ma con Destination Address e Source Address più alti. Il traffico offerto a 250 Kbit/s, i pacchetti di 8 byte e il CRC a 15 bit sono quindi limitativi e obsoleti per la rete del futuro, poiché non permettono quella scalabilità richiesta per poter implementare queste nuove funzionalità.

3.5 Requisiti della rete del futuro

Per raggiungere queste funzionalità e garantirne comunque un'estensione plausibile delle stesse nel medio-lungo periodo sono stati identificati i seguenti requisiti:

- Alto throughput
    - Per l'invio simultaneo di più stream video di telecamere,
    - Per l'affinamento dei controlli basati su Auxiliary Function/Input, ora insufficienti,
    - Per l'invio di sempre maggiori moli di dati di diagnosi e di registrazione dei parametri sensibili delle lavorazioni (inquinanti, pesticidi, concimi, sementi e in generale tutti i dati interessanti per la qualità del prodotto e per la gestione finanziaria della farm),
    - Per l'aggiornamento di firmware e di mappe di prescrizione e programmi di lavoro in tempi compatibili con le esigenze di lavorazione,
- Sicurezza e autenticazione
    - Per le comunicazioni tra ECU e TECU che hanno implicazioni di safety. Si prenda ad esempio un attrezzo che per ottimizzare il lavoro necessita di una certa funzionalità e richieda alla TECU il mantenimento di una velocità costante,
    - Per la comunicazione tra Sequence Control Master e TECU, che può richiedere l'utilizzo completo dello sterzo, freni e acceleratore,
    - Comunicazione fra ECU di macchine diverse (Macchine cooperative in cluster).
- Scalabilità
    - Per poter gestire nuove funzionalità in una prospettiva di lungo periodo,
    - Per poter gestire un numero sempre crescente di nodi della rete.
- Retro-compatibilità
    - Per poter comunque supportare le ECU basaste su CAN, poiché il tempo di vita di trattori ed attrezzi è molto elevato, così come il costo.
- Safety
    - Per poter allineare il protocollo di comunicazione agli standard ISO25119 e ISO13849/ISO15998.

- Openness
    - Uno standard è facilmente adottabile se utilizza architetture aperte e non proprietarie,
    - L'utilizzo di altri standard aperti e Well Tried, assieme a componenti COTS può migliorare la diffusione dello standard e contribuisce alla riduzione dei costi anche su piccole serie di macchine.
- Integrazione con "Internet of Things"
    - Per scopi di fleet management, diagnosi automatiche, service in tempo reale, e per la possibilità di interazione tra il veicolo e il Farm Management System, ove si possono impartire in real-time i comandi e le mappe di prescrizione.

La ricerca della rete del futuro si deve basare su tecnologie già esistenti, poiché il problema della inadeguatezza delle funzionalità implementabili su CAN è già stata affrontata negli ambiti industriali, Automotive e dell'aeronautica. In questi ambienti la migrazione verso altri standard è in corso o è già stata fatta e si possono trovare molte similitudini, ma anche molte differenze, visto che i driver e le esigenze dell'ambito agricolo presentano molte particolarità e una maggiore complessità e dinamica.

Uno dei denominatori comuni della migrazione è il fatto di utilizzare un layer fisico diverso, cioè Ethernet con alcune modifiche per quanto riguarda il metodo di accesso al canale, la gestione della comunicazione e, talvolta, creando un protocollo proprietario basato su hardware proprietario. Alcuni utilizzano hardware proprietari (ad esempio Ethercat), altri utilizzano hardware standard ma protocolli proprietari (BMW).

Nei prossimi capitoli saranno analizzate alcune soluzioni basate su CAN e su Ethernet che sono state prese in considerazione durante la ricerca, con analizzati i punti di forza e le debolezze per i requisiti posti dalle reti agricole del futuro.

3.6 Differenze fra CAN ed Ethernet

Lo standard ISO 11898 e lo standard Ethernet (sottoinsieme della norma IEEE 802.3) hanno caratteristiche molto diverse, dovute alle differenti applicazioni per cui sono stati pensati.

Tabella 3.3 - Differenze principale tra CAN ed Ethernet

	CAN 2.0B	**Ethernet (100BaseT)**
Banda	Fino a 1 MBit/s	100 MBit/s
Lunghezza della rete	40 m a 1 M Teoricamente infinita con Repeater	100 Metri Teoricamente infinita con Hub
Contenuto informativo di un pacchetto	8 byte	1500 byte
Codifica	Differenziale	A blocchi 5b4
Modalità di accesso al mezzo	CSMA/BA	CSMA/CD
Tipo di cavo	Twisted Pair (Quad Twister Pair per J1939 e ISO 11783)	Quad Twisted Pair
Terminazioni	120 Ω	100 Ω
Topologie	Bus	Stella e/o Anello (con Switch) Bus (con Hub)
Conformità EMC per Automotive	Dipende dal Transceiver (PHY)	Dipende dal Transceiver (PHY)
CRC	15 bit	32 bit
Prioritizzazione	Basata su ID	Nessuna

La Tabella 3.3 riassume le principali differenze tra lo standard CAN e quello Ethernet. Per il confronto è stato preso lo standard IEEE 802.3 più diffuso, cioè Ethernet 100BaseTX. A prima vista si nota subito il vantaggio principale di Ethernet su CAN, la larghezza di banda, che è superiore di due ordini (o tre, se si considera il Gigabit) di grandezza al massimo consentito dal protocollo CAN. A questo si aggiunge che l'unità minima, cioè il pacchetto base può contenere una quantità di dati 100 volte maggiore, feature molto utile per invio di diagnosi o di comandi a più attuatori con un unico pacchetto, inoltre l'overhead nel caso di protocolli di trasporto viene molto ridotto rispetto al Transport Protocol delle reti CAN based. La lunghezza di un singolo segmento di rete è molto maggiore su Ethernet, quindi in grado di coprire interamente il backbone di un trattore con molti implement in cascata. Infine si ha un CRC molto più robusto rispetto a quello implementato in hardware su CAN, riducendo così la probabilità di errore residuo sul canale di comunicazione.

Il problema fondamentale di Ethernet è il metodo di accesso che, a differenza del CAN, è distruttivo:

- Nel protocollo CAN il metodo d'accesso Carrier Sense Multiple Access with Bus Arbitration, che assicura che i pacchetti non vengano distrutti, oltre a fornire una prioritizzazione dei messaggi basata sull'ID, con conseguente controllo di congestione gestito direttamente in hardware;

- Ethernet utilizza, invece, un metodo Carrier Sense Multiple Access with Collision Detection, che non preserva il pacchetto dalla distruzione nel caso in cui due nodi accedano al BUS contemporaneamente.

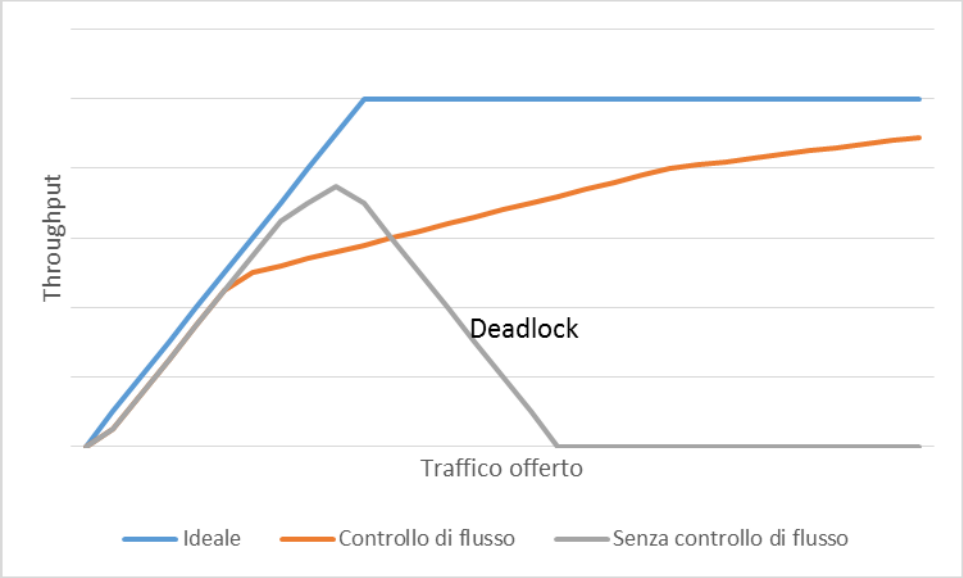


Figura 3.6 - Trend del throughput rispetto al traffico offerto di un sistema CSMA/CD

Sulle reti Ethernet il meccanismo di Collision Detection, oltre alla distruzione del pacchetto, porta con sé la segnalazione di un messaggio ICMP (se si usa lo stack TCP/IP) per segnalare l'errore e costringe entrambi i nodi ad una ritrasmissione. Dal punto di vista della reliability della rete questo è un grosso limite, poiché all'aumentare del traffico offerto si può verificare una situazione di deadlock, dove le ritrasmissioni intasano la rete (il cui trend qualitativo è mostrato in Figura 3.6) riducendo il throughput offerto. Per questo motivo è necessario un diverso accesso al canale o un meccanismo di controllo di flusso o di controllo di congestione che possa eliminare questo fenomeno, che deve necessariamente essere implementato nei livelli superiori del modello ISO-OSI.

3.7 Competitors - CAN

Prima di parlare di soluzioni basate su layer fisico Ethernet è doveroso ricordare che esistono anche dei protocolli e standard basati su un'evoluzione del CAN, che ne aumentano il massimo throughput raggiungibile e/o la sicurezza.

3.7.1 Flexray

Il primo standard preso in considerazione è FlexRay o ISO 17458 che permette un troughput fino a 10 Mbit/s, garantendo alti livelli di affidabilità. Di contro utilizza hardware molto costosi e non consente funzionalità quali il plug & play, data la staticità della configurazione della mappa di divisione del tempo del beacon. Inoltre ha gli stessi problemi della lunghezza del bus, dovuta principalmente alla necessità di sincronizzazione dei clock delle periferiche, gestito da complessi bus guardian residenti nei controller. Il suo utilizzo è stato confinato nel mondo automotive per il quale è stato espressamente concepito, in particolare in alcune vetture BMW, per alcuni apparati della vettura, tipicamente per funzionalità safety relevant. Gli elevati costi di periferiche di questo genere non si confanno alle richieste del mercato e il throughput massimo disponibile non è sufficiente per garantire la scalabilità richiesta nel medio-lungo periodo.

3.7.2 CAN-FD

Il secondo standard analizzato è quello che potrebbe essere considerato il successore del CAN 2.0, detto CAN Flexible Data Rate (CAN-FD) sviluppato all'interno di Bosch e pubblicato per la prima volta nel 2011. Esso si pone come compatibile con lo standard ISO 11898 e in grado di supportare throughput superiori, utilizzando pacchetti dati fino a 64 byte contenuti nello stesso spazio dei vecchi pacchetti a 8 byte.

La bitrate è dinamica, in quanto la fase di arbitraggio e di acknowledge rimane identica al protocollo CAN 2.0, mentre all'interno del campo dati e checksum, essa può aumentare in base alla dimensione dei dati da inviare. In questo modo nello spazio di 8 byte + checksum si possono inviare pacchetti di 16,32 o 64 byte, con checksum a 17 o 21 bit. Lo svantaggio principale è che, nonostante sia stata definita come compatibile, i nodi equipaggiati con la periferica CAN classica non riconoscono i dati e falliscono il calcolo del checksum, di fatto riempiendo la rete di Error Frame. Quindi in realtà, per poter creare una rete ibrida di nodi CAN 2.0 e nodi CAN-FD viene consigliato di settare i nodi legacy in Listen Only Mode, per poi passare alla modalità trasmissione solo quando devono inviare un pacchetto. Tale tecnica è difficilmente realizzabile in una rete dinamica; inoltre questo influisce sul meccanismo di gestione degli errori, che ha reso il CAN molto utilizzato in ambienti automotive ed industriali. Infine la retro-compatibilità è garantita solo se si interviene sui nodi legacy con queste modifiche, rimanendo incompatibile con nodi legacy in funzionamento normale. I limiti di banda rimangono gli stessi, come i limiti di lunghezza del Bus, aumentando il throughput teorico fino ad 8 volte. Il massimo throughput raggiungibile rimane comunque inferiore agli 8Mbit con banda base della rete a 1Mbit che permette lunghezze del bus inferiori ai 10 metri.

Usare questo tipo di bus su una rete a norma ISO 11783, significherebbe poter aumentare il troughput al massimo di 8 volte, quindi a meno di 2MBit, insufficienti per le nuove funzionalità da implementare nel

medio-lungo periodo. Questo, aggiunto alla scarsa retro-compatibilità ha reso non adottabile questa soluzione.

3.8 Competitors - Ethernet fieldbuses

3.8.1 Ethercat

Ethercat è un protocollo di comunicazione sviluppato da Beckoff nel 2003 per l'ambiente industriale in grado di realizzare un controllo distribuito di vari device, garantendo il determinismo della comunicazione. È molto utilizzato anche nell'ambio di applicazioni safety-relevant, nella sua variante "Safety over EtherCAT", in grado di soddisfare i requisiti SIL3 per la parte di comunicazione [10] .

Figura 3.7 - Percorso di un pacchetto EtherCAT

Il protocollo utilizza Ethernet 100BaseTX o 100BaseFX come layer fisico e si basa su pacchetti Ethernet particolari, contraddistinti da un tipo *Ethertype* diverso da quello usato per gli altri protocolli (Es IP). Il tipo di comunicazione realizzata è di tipo Master-Slave, in cui esiste un master che invia comandi o richieste e unità slave che rispondono a tali richieste. La topologia tipica è a bus, ma può essere modificata all'occorrenza in stella.

Il determinismo viene garantito dal meccanismo di comunicazione, dove ogni pacchetto inviato dal master viene processato da uno slave e passato al successivo slave, fino ad arrivare alla fine del bus, da cui torna di nuovo al master per lo stesso percorso inverso, in uno schema logical-ring. Ciascuno slave ha un hardware proprietario che permette la lettura del pacchetto e la modifica on-the-fly della area di pacchetto a lui riservata, riducendo il tempo di trasmissione grazie all'annullamento di problemi di arbitraggio e alla riduzione e annullamento dei tempi inter-slot tra risposte delle diverse unità afferenti alla rete.

I tempi di ciclo ottenibili sono molto bassi (al di sotto dei 250 µs con 100 devices [11]), permettendo controlli dell'ordine di 10 KHz.

Il protocollo EtherCAT è proprietario e per poterlo utilizzare è necessario entrare a far parte del consorzio EtherCAT. Il master non richiede una interfaccia fisica particolare, bensì solo lo stack Ethercat software (che può essere scaricato e utilizzato dai membri), mentre ciascuno Slave richiede l'impiego di un hardware

proprietario per il corretto funzionamento del meccanismo di modifica On-The-Fly dei pacchetti. Esiste una versione open-source dello stack Ethercat che però impone il limite di utilizzo per soli scopi didattici [12].

3.8.2 Powerlink

Il protocollo Ethernet Powerlink (EPL) è stato per la prima volta introdotto nel 2001 dalla compagnia austriaca B&R, per il mondo delle automazioni industriali. Per ottenere determinismo e reliability Powerlink agisce a livello 2 della pila ISO/OSI, attuando delle modifiche al metodo di accesso al canale. Powerlink forza un accesso di tipo Time Division Multiple Access in grado di eliminare i problemi di deadlock e di fallimenti sull'invio di pacchetti.

Questo meccanismo è assicurato dall'esistenza di un master della comunicazione (detto management node, MN) che dà le tempistiche attraverso l'invio di token broadcast. Ciascuno slave comunica nel proprio slot temporale assegnato dal master dopo una fase di autenticazione e discovery delle unità presenti nella rete. In questo modo si evita la possibilità di accessi simultanei al bus, aumentando la reliability del sistema.

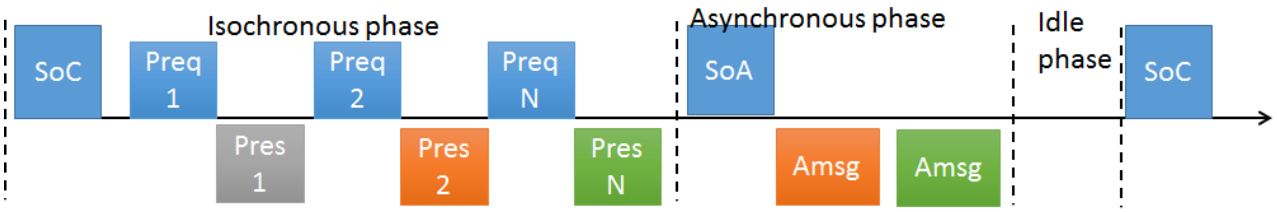


Figura 3.8 - Esempio di ciclo di comunicazione su Powerlink

Un ciclo di comunicazione su powerlink si divide in tre fasi:

- Fase isocrona: Inizia con l'invio di un pacchetto broadcast di sincronizzazione da parte del master detto SoC (Start of Cycle). In base alla configurazione della rete il master invia un pacchetto unicast a ciascun nodo (Poll Request Frame). Il nodo risponde con un pacchetto multicast (Poll Request Response) che informa tutti gli altri nodi dell'inizio del proprio slot. Alla fine del tempo a disposizione dello slot, il master invia un altro Poll Request Frame al nodo successivo.
- Fase asincrona: Inizia con l'invio di un pacchetto broadcast da parte del master detto SoA (Start of Asynchronous). In questa fase torna il funzionamento CSMA/CD, dove tutti i nodi possono comunicare liberamente in maniera asincrona, ed utilizzare tutte le features del protollo TCP/IP standard, fino all'arrivo di un nuovo pacchetto di sincronizzazione di tipo Start Of Cycle
- Fase idle: questa fase non è sempre presente, bensì è quella fase tra la parte asincrona e l'inizio di un nuovo ciclo dove nessun nodo comunica.

Per migliorare l'utilizzo della banda alcuni nodi possono condividere lo stesso slot, avendo così slot di dimensione diversa. Questo protocollo garantisce reliability e determinismo, con tempi di ciclo bassi, fino a 200 µS [13]. Il numero massimo di nodi consentito dal protocollo è di 240. La configurazione non è statica e può essere modificata in qualsiasi momento, ad esempio per l'introduzione di un nuovo nodo. Le richieste di associazione del nuovo nodo avvengono nella fase asincrona in modo da non disturbare la comunicazione isocrona.

3.8.3 TTEthernet

TTEthernet [14] è un protocollo di comunicazione commercializzato da TTTech Computertechnik AG e standardizzato come SAE AS6802 (standard certificato anche per l'avionica), per comunicazioni che richiedono un hard real-time. Esso si basa sullo stesso principio di powerlink, cioè sulla modifica del metodo d'accesso al canale su TDMA. Come Powerlink definisce degli stadi della comunicazione in cui ciascun nodo partecipante invia un certo tipo di traffico. Rispetto a Powerlink fornisce tre tipologie di traffico:

- Messaggi time-triggered: messaggi che richiedono un hard real-time e per questo sono i più prioritari
- Messaggi rate-constrained: messaggi che richiedono di essere inviati ad un certo rate, ma che hanno richieste in termini di tempo inter-pacchetto meno stringenti, cioè possono supportare un jitter limitato
- Messaggi best-effort: messaggi che non hanno requisiti di determinismo e possono appartenere anche a protocolli diversi rispetto a TTEthernet

A differenza di Powerlink il determinismo e l'hard real-time è garantito dal meccanismo di sincronizzazione dei clock basato su IEEE 1588 e dall'utilizzo di interfacce di rete proprietarie (switch TTEthernet) in grado di limitare al massimo i fenomeni di jitter e trattando in maniera diversa i diversi tipi di traffico.

Ha prestazioni nettamente superiori rispetto a Powerlink, però richiede hardware proprietario e la configurazione di rete è statica, cioè decisa a priori.Tale limitazione lo rende quindi inutilizzabile per le reti agricole del futuro.

3.8.4 Stack TCP/IP

Lo Stack TCP/IP fornisce numerose funzionalità e servizi, è molto utilizzato e stabile e affronta il problema della congestione in maniera diversa rispetto ad un Ethernet fieldbus. La semantica dello Stack TCP/IP è di tipo best-effort con controllo di congestione per aumentare la reliability, a scapito però del determinismo.

Uno dei punti di forza dello stack TCP/IP è che è stato concepito per supportare una moltitudine di servizi molto differenziata ed è molto malleabile dal punto di vista delle semantiche. Inoltre lascia una grande libertà d'azione nel poter creare dei protocolli ad-hoc per gli scopi più svariati. Molte funzionalità, come il CRC, i protocolli di trasporto, i protocolli di sicurezza e di autenticazione, video e audio real-time, QoS, VLAN, si interfacciano in maniera trasparente sulla suite TCP/IP. Questo ne fa uno strumento ideale, inoltre il suo possibile utilizzo sia su reti wired che wireless costituisce un vantaggio enorme, per quanto riguarda l'integrazione del trattore su reti di macchine cooperative o comunque raggiungibili da remoto attraverso servizi internet. Infine, una delle caratteristiche più potenti e flessibili è sicuramente il meccanismo delle porte, ciascuna delle quali può fornire un servizio od una funzionalità e può essere assegnata a reti private virtuali (VPN).

La suite TCP/IP fornisce di base due protocolli di trasferimento a livello 4, che hanno semantiche completamente diverse di seguito descritti brevemente. Entrambi si basano su IP a layer 3 della definizione di stack di ISO /OSI.

3.8.4.1 TCP

Il Transmission Control Protocol (TCP) è un protocollo di trasporto reliable, a controllo di congestione e orientato alla connessione, di solito utilizzato per comunicazioni di grandi moli di dati. Esso implementa una connessione virtuale tra due peer, in grado di garantire una trasmissione affidabile di pacchetti, e in grado di occuparsi del loro riordinamento ed di implementare un controllo di congestione che abbassa la velocità, aumentando così la latenza e la reliability. Il TCP può essere in teoria usato in applicazioni sensibili alla latenza, ma andrebbero considerati diversi fattori. Ad esempio TCP rimane a bassa latenza solo se i ricevitori del traffico sono sempre 'attivi', e se il canale non è congestionato, in pratica la latenza è bassa finché non si riscontrano significativi ostacoli che abilitano il controllo di flusso, perdendo così il determinismo della trasmissione. Nelle reti ISO 11783 tale protocollo potrebbe sostituire i protocolli di trasporto descritti in **Errore. L'origine riferimento non è stata trovata.**, ma poco si confà all'invio di acchetti temporizzati.

Sarebbe comunque possibile migliorare il comportamento del protocollo TCP, mediante alcune modifiche e/o configurazione di parametri dello stesso. Ad esempio:

- Usare una bufferizzazione intelligente o socket non bloccanti per poter 'droppare' alcuni pacchetti a bassa priorità in situazioni di congestione pur di mantenere una bassa latenza,
- Disabilitare l'algoritmo di Nagle attivando l'opzione TCP_NODELAY in setsockopt(anche se questa modifica realizza un miglioramento per quanto concerne la latenza ma a scapito dell'efficienza),
- Impostare un'alta priorità al traffico che deve avere bassa latenza (QoS),

- Mantenere il buffer delle socket ad una grandezza minima cosicché non vi sia introduzione di latenza a causa della bufferizzazione ma piuttosto una perdita di dati.

3.8.4.2 UDP

L'User Datagram Protocol è un protocollo leggero, senza connessione e non reliable. Attraverso questo protocollo si possono mandare pacchetti fino a 65Kbyte, di cui però non si ha la garanzia di arrivo. Esso non prevede l'instanziazione e la negoziazione di una connessione e non tiene conto della congestione. Tale protocollo è pensato per la flessibilità a costo di non avere alcuna garanzia di reliability. È utilizzato come base per molti altri protocolli, poiché si possono implementare all'occorrenza meccanismi quali acknowledge espliciti e/o running number per creare protocolli ad-hoc a livelli più alti. Un esempio è il Realtime Protocol (RTP) utilizzato per le comunicazioni audio-video che si basa su UDP: esso non ha bisogno di reliability, bensì ha vincoli stringenti in termini di real-time. La reliability è demandata all'applicazione, che provvederà a sostituire i pacchetti mancanti con pacchetti vuoti o aggiungendo del rumore.

Dal punto di vista della ricerca, è molto interessante proprio per la sua alta flessibilità e la possibilità di customizzazione in base alle esigenze del protocollo implementato.

3.9 La sintesi della tecnologia

	Openess	Hardware availability/cost	Bandwidth	Hotplug Capability	Topology	Safety (certification)	Automotive Compliancy (EMI/EMC, temp, humidity, ...)	Semantics	Realtime
Stack TCP/IP + Ethernet	YES	high availability, COTS and industrial solutions	100/1000Mbit	YES	Every	NO (possible with some modification)	PHY dependent	Multi-Master Multi-Slave	NO
Ethernet Powerlink	YES	High availability, COTS and industrial solutions	100/1000Mbit	YES	Every	N/A	PHY dependent	Master-Slave (for timing) MM-MS is possible	Soft Realtime
TTEthernet	mostly YES	Expensive proprietary network infrastructures	100/1000Mbit	NO	Every	N/A (but used for Steer By Wire)	PHY dependent	Master-Slave (for timing) MM-MS is possible	Hard Realtime
Ethercat	Closed Standard	Annual fee Proprietary hardware slaves	100/1000Mbit	NO	Logical Ring	IEC-61508 SIL 3 certified layer	PHY dependent	Master-Slave	Hard Realtime
Flexray	Closed standard	Expensive controllers	up to 10Mbit	NO (attempts were made to enable feature)	Physical bus	N/A	YES	Master-Slave	Hard Realtime
CAN-FD	Yes	ISO11898-2/6 transceivers	2Mbit for ISOBUS theoretical	YES	Physical bus	N/A	PHY dependent	Multi-Master Multi-Slave	Hard Realtime

Figura 3.9 – Tabella riassuntiva delle facilities dei protocolli analizzati

I protocolli fin ora analizzati hanno pro e contro, a cui però bisogna dare un diverso peso. Infatti, per avere un hard realtime, si vanno incontro a problemi di costo e di chiusura dello standard. Inoltre la maggior parte degli Ethernet Fieldbuses riescono a funzionare e a garantire un certo livello di prestazioni e di real-

time a scapito della configurabilità del sistema, che diventa staticamente decisa dal system integrator all'atto della sua costruzione. La capacità di auto-configurarsi è fondamentale se si vuole mantenere la capacità Plug-N-Play portata avanti dallo Standard ISO 11783. Se a questo si aggiunge il problema dei componenti proprietari necessari al funzionamento della rete, che limitano l'adozione di tale tecnologia all'utilizzo di componenti di uno o pochi costruttori, si può desumere che questi protocolli non si confanno alle esigenze della rete ISO 11783 del futuro.

Il CAN-FD potrebbe rappresentare una soluzione, se non fosse che comunque richiederebbe la costruzione di un gateway di separazione tra la vecchia rete ISOBUS e la nuova rete, avendo comunque un benefit limitato a soli 2 Mbit/s teorici, troppo pochi per garantire una scalabilità di medio-lungo periodo.

I due candidati per la rete del futuro, che più si avvicinano ai requisiti imposti sono Powerlink e lo stack TCP/IP. Essi possono avere una configurazione dinamica, supportando l'inserimento di nuovi nodi. Powerlink fornisce già tutta l'infrastruttura per ottenere una rete reliable ed avere un soft real time, a scapito però della retrocompatibilità con le reti già esistenti, il cui accesso al canale è basato su CSMA. Lo Stack TCP/IP può essere plasmato in modo da poter rimpiazzare trasparentemente tutti i livelli ISO/OSI e fare in modo che il vecchio possa parlare con il nuovo in maniera completamente trasparente. La soluzione Powerlink, per poter essere retrocompatibile, richiede la realizzazione di un gateway molto complesso, che da una parte deve implementare il master della comunicazione TDMA e dall'altra parte deve "simulare" tutte le ECU lato Powerlink. Essendo le due semantiche diverse, il gateway dovrebbe prendersi carico di intere parti di protocollo ISO11783 da rimodulare per le esigenze del TDMA, con bufferizzazione dei messaggi e possibili ritardi.

Lo Stack TCP/IP, con opportuni accorgimenti e modifiche, è in grado di fornire determinismo e soft real time, come dimostrato in un'esperienza di dottorato precedente sulle reti Wireless fatta all'interno del WG5 per le comunicazioni Machine To Machine [15]. Le differenze con questa esperienza sono molte, in quanto il traffico M2M è basato su reti Wireless con diversi metodi di accesso al canale e per scopi diversi da quelli della presente ricerca, anche se si tratta comunque di controllo distribuito con requisiti di safety e real-time. Si può trovare un denominatore comune tra l'esperienza maturata precedentemente per le reti Wireless e le reti Wired del futuro, sfruttando le conoscenze e gli strumenti acquisiti..

Date tutte queste considerazioni emerge quindi la scelta fatta, che consiste nell'arricchire lo Stack TCP/IP per poter dare una rete del futuro scalabile, completamente retrocompatibile, deterministica, sicura e con alta integrazione al mondo dei servizi, del cloud e della comunicazione wireless per cluster di macchine.

La ricerca del presente Dottorato, è partita quindi dall'analisi di prestazioni dello Stack TCP/IP in condizioni che simulano un utilizzo tipico all'interno di una rete ISOBUS, per ricavarne informazioni utili sul comportamento totale del sistema e testarne i limiti.

3.10 Task 0 – Analisi di prestazioni con Stack TCP/IP

3.10.1 Scopo

Lo Stack TCP/IP è il protocollo di comunicazione maggiormente utilizzato per un'alta varietà di applicazioni, il che garantisce una più semplice adottabilità da parte dei costruttori, per il gran numero di periferiche COTS e le numerose suite disponibili (Windows Socket, Berkley Socket, uIP, iwIP, OpenTCP, FNET Embedded TCP/IP, ecc..) per vari target, dagli embedded 16 bit ai System On Chip ARM, fino ai PC.

Lo scopo di questa prima fase è di pura investigazione sul comportamento dello Stack TCP/IP per tipologie di traffico *rate-constrained*, che rappresentano la maggior parte del traffico all'interno di una rete ISOBUS durante il funzionamento. Queste misurazioni verranno fatte in base alle diverse topologie di reti, tempi di ciclo, dimensione del pacchetto, numero degli host e traffico di disturbo per poter verificare appunto quale sia la migliore configurazione per questa tipologia di traffico.

Il sistema simulato è un sistema master-multislave dove i flussi sono rivolti tutti in un'unica direzione. I messaggi inviati sono marcati con running number, time stamp e contengono dei dati pseudo-casuali che simulano comandi o stati degli slave. I pacchetti vengono inviati utilizzando il protocollo UDP, per ridurre al minimo il ritardo di trasmissione e l'overhead di comunicazione. La rete utilizzata è basata su un'Ethernet 100BaseTx con cavi CAT5, una delle configurazioni più comuni in circolazione per reti LAN.

Di seguito verranno descritti i parametri utilizzati per la creazione dei test.

3.10.1.1 Lunghezza del pacchetto

La lunghezza del pacchetto è fondamentale, in quanto costituisce il trade-off tra la parte di header e la parte di dato. Pacchetti troppo piccoli potrebbero ad un contenuto informativo troppo limitato rispetto all'overhead, facendo calare il throughput reale del sistema. Pacchetti troppo lunghi vanno comunque evitati, poiché forzerebbero i livelli inferiori alla frammentazione del pacchetto, impedendo il raggiungimento del target di real time della informazione trasferita. Un pacchetto Ethernet può portare un contenuto informativo che va da 46 a 1500 byte di dati. La varietà di periferiche su cui il pacchetto passa, potrebbero avere come Maximum Transmission Unit inferiore alla dimensione massima di un pacchetto Ethernet, fatto dovuto al passaggio su layer fisici diversi. Questo problema viene risolto dal protocollo IP, che nella risoluzione dei percorsi riesce a scoprire l'MTU dell'intero percorso. Se a IP arrivano pacchetti più grandi del MTU del percorso, essi vengono frammentati e riassemblati. Il procedimento di frammentazione IP, che può avvenire sia perché si inviano pacchetti Ethernet più grandi di 1500 byte, oppure perché si inviano pacchetti IP maggiori del MTU del percorso, porta ad un calo di determinismo, poiché la frammentazione gestita da IP non è controllabile a priori nelle normali implementazioni del protocollo.

Oltre a ciò, bisogna tenere conto che non tutti gli host sono in grado di gestire pacchetti di 1500 byte a livello IP, pertanto la dimensione del pacchetto andrebbe fatta tenendo conto del limite minimo di dati gestibile da un nodo IP, cioè 576 byte [15].

3.10.1.2 Transmission rate

Il rate di trasmissione o tempo inter-pacchetto definisce la granularità del controllo. In una rete ISO 11783 i pacchetti di controllo o di comando non sono diretti (ad esempio non pilotano direttamente un PID o un'elettrovalvola), bensì utilizzano valori target che poi vengono rielaborati. Ad ora su una rete ISO 11783 i comandi vengono inviati da una frequenza minima di 1 Hz fino ad un massimo dii 50 Hz, per evitare il più possibile fenomeni di saturazione della rete.

3.10.1.3 Topologia della rete

La topologia della rete è un parametro chiave, in quanto può modificare sostanzialmente il comportamento della rete stessa. Con gli hub si ha un bus logico, mentre gli switch realizzano una topologia a stella. I tempi di ritardo e la varianza (che determina il determinismo), come la possibilità di distruzione di pacchetti cambia molto fra l'utilizzo di uno o dell'altro

Gli Hub sono dei repeater L1 che copiano il segnale di ciascuna porta TX su tutte le altre porte in RX, creando quindi un bus logico. Se due host comunicano nello stesso istante si può avere una collisione, riducendo la reliability. Allo stesso tempo l'hub introduce un tempo di ritardo (per i Classe I) di 1,4 µs [16], ovvero il tempo per trasmettere i primi 140 bit, che servono per il meccanismo di Collision Detection.

Gli switch invece sono dei bridge L2 (o alcuni più avanzati L3), che instradano i pacchetti in base all'indirizzo MAC di destinazione. Al loro interno hanno una memoria che registra, per ciascuna porta, una tabella di indirizzi MAC collezionati. All'arrivo di un pacchetto ad un MAC sconosciuto, lo invia su tutte le porte. A differenza degli Hub, gli Switch hanno dei meccanismi di bufferizzazione dei pacchetti in diverse code, una per ciascuna porta. Questo porta a eliminare il problema delle collisioni all'interno di una rete puramente switched, ma allo stesso tempo introduce ritardi di propagazione e quindi latenze molto più elevate di quelle di un hub, e tali latenze potrebbero degradare il real-time della comunicazione. Esistono 4 meccanismi di forwarding che possono essere utilizzati sugli switch:

- Store and forward: Lo switch bufferizza ciascun pacchetto prima di trasmetterlo. In questo modo viene introdotto un ritardo pari al tempo di trasmissione, non calcolabile quindi a priori, ma al più 16 µs. La bufferizzazione serve per controllare la validità del pacchetto, distruggendolo se non valido e diminuire il traffico dovuto ad eventuali segnalazioni di errori. È uno dei metodi più utilizzati negli switch commerciali.

- Cut through: Lo switch inizia la trasmissione del pacchetto alla lettura del destination address, se la porta è libera, altrimenti esegue il meccanismo di Store And Forward. Nel caso in cui venga trasmesso il pacchetto immediatamente si ha un ritardo di soli 1,6 µs, senza però eseguire la validazione del pacchetto, quindi col potenziale rischio di trasmettere un pacchetto non valido.
- Fragment free: Questo metodo è un ibrido che cerca di prendere I vantaggi dei due metodi precedenti, attraverso il quale il pacchetto Ethernet inizia ad essere inviato dopo aver ricevuto i primi 64 byte. Questa dimensione assicura che il pacchetto ricevuto non è un frammento, cioè un pacchetto distrutto da una collisione. Non viene fatta alcuna assunzione sulla validità del pacchetto, poiché per fare ciò servirebbe verificare i 4 byte di trailer dello stesso. Anche qui, se la coda è piena il pacchetto viene comunque bufferizzato. Al minimo si ha un ritardo di trasmissione di 6,4 µs.
- Adaptive switching: Un metodo in cui lo switch sceglie in maniera automatica quale strategia attuare delle precedenti, per il forwarding dei pacchetti.

3.10.1.4 Numero degli host

La variazione sul numero degli host non dovrebbe portare grosse differenze dal punto di vista del determinismo su reti a stella, a patto che non si raggiungano condizioni di traffico limite. Ci si aspetta un comportamento proporzionale tra l'aumento del numero degli host e la diminuzione del tempo di ciclo.

3.10.1.5 Traffico di disturbo

Inserire del traffico di disturbo serve a testarne gli effetti sul determinismo del sistema in condizioni di carico dovuto a traffico che poco c'entra con la comunicazione in corso (un esempio potrebbe essere l'inserimento di un nodo o un'operazione di scarico diagnosi su un file server). Esso non dovrebbe inficiare la comunicazione dei comandi sulla rete, ma può essere tollerato un aumento limitato di jitter.

3.10.2 Testbench

Per la realizzazione del banco di test sono stati utilizzati 5 PC, con CPU a 3GHz e 512 MByte di RAM, connessi via rete Wifi per la sincronizzazione e il planning dei test da remoto. Un altro PC viene utilizzato per lanciare i test e raccogliere i dati degli stessi. Tutti i PC montano una distribuzione Red Hat di Linux, con kernel patchato con la patch RT_PREEMPT, in modo da poter utilizzare alcune facilities soft real-time, come i timer ad alta precisione, i task a priorità ed i segnali real-time, in modo da abbattere I tempi di latenza dovuti al sistema operativo. Tutte le configurazioni ACPI sono state disabilitate, in modo da non avere riduzioni di clock non previste.

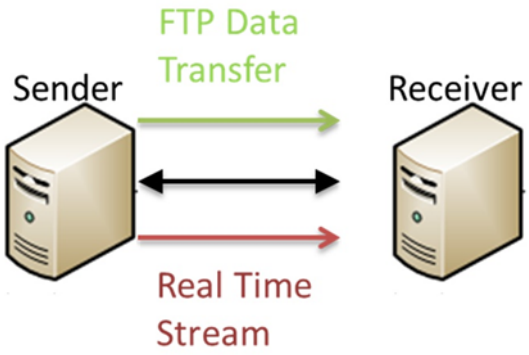


Figura 3.10 - Layout del testbench "crossed"

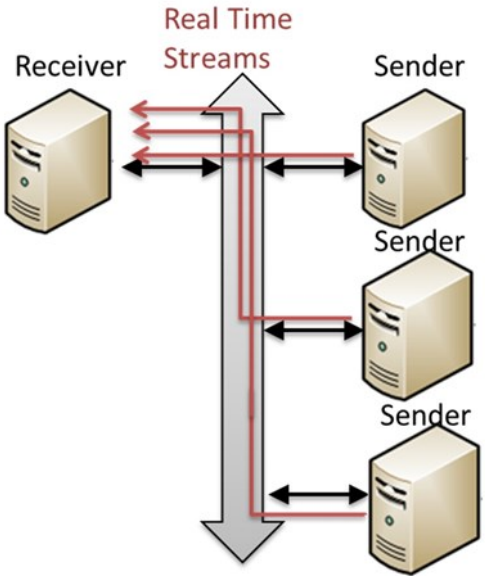


Figura 3.11 - Layout del testbench "logical bus"

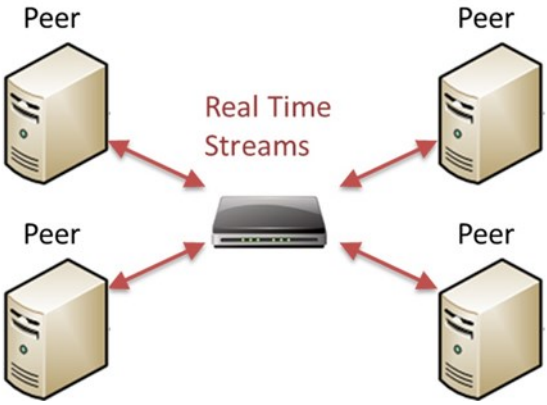


Figura 3.12 - Layout del testbench "star"

Tutti i PC sono collegati attraverso Wi-Fi ad un altro PC, di controllo, il quale apre su ciascuno di essi una shell SSH per richiamare i programmi di test. L'infrastruttura di test è completata dall'infrastruttura di rete scelta di volta in volta. L'hardware utilizzato per i tre tipi di topologia sono:

1. Cavo UTP Cat5 crossato, per I test sulle massime performance tra soli due host (Figura 3.10)
2. Un Hub 3-COM 3C16750B 10/100 Mbps, per simulare un'architettura a bus logico (Figura 3.11)
3. Uno Switch 3COM 3C16791A 10/100 Mbps Full-Duplex con strategia Store-and-Forward per creare l'architettura a stella (Figura 3.12)

Vengono utilizzati diversi programmi per l'invio, la ricezione dei pacchetti, oltre a script in shell per lanciare tutti i test, raccogliere i risultati in directory, processarli e trasformarli in grafici e statistiche.

Nonostante la ricerca sia rivolta alle prestazioni su dispositivi embedded, si è scelto di utilizzare macchine X86 con Linux per poter creare una piattaforma di test che non risenta di limitazioni dovute alle prestazioni dell'hardware,, bensì dalla rete analizzata, oltre che per avere a disposizione strumenti utili di remotazione e la patch RT_PREEMP più stabile che non su sistemi Linux embedded. È stato eseguito un lavoro di tuning delle applicazioni e del sistema operativo per diminuire al massimo l'impatto del sistema operativo sulle operazioni di invio e ricezione pacchetti, per poter ricondursi alle stesse tempistiche di un Embedded, che ha sicuramente prestazioni inferiori in termini di velocità di clock, ma sicuramente ci sono meno passaggi tra la chiamata a funzione di invio pacchetto e l'effettivo arrivo sul PHY.

I programmi di esecuzione dei test sono scritti in GNU C, con utilizzo massiccio di system call POSIX, applicando molti accorgimenti, tenendo conto della struttura interna del kernel, per minimizzare il tempo di latenza e mascherare eventuali eventi asincroni sulle macchine [18]. Sia il processo di sender che quello di receiver, grazie alla patch RT e alle modifiche delle impostazioni di sistema, riescono a girare alla più alta priorità che ci possa essere su Linux, con nice -5 e real time priority a 89. Un altro accorgimento, ad esempio, è quello di precaricare tutto il programma in RAM, in modo che non ci siano chiamate del sistema operativo al disco fisso, che potrebbero creare disturbo non voluto all'interno dei test, dando dei risultati errati.

I programmi di esecuzione dei test sono 2, entrambi da riga di comando e facilmente personalizzabili per la varietà di opzioni con cui possono essere chiamati.

Il primo programma, sender, invia il numero di pacchetti ad una certa frequenza, entrambi parametri impostabili da linea di comando, come datagrammi UDP verso una particolare destinazione. Questi pacchetti UDP hanno il campo Type Of Service settato a Low Delay, per garantire una priorità maggiore rispetto a traffico TCP di distrurbo iniettato per test in condizioni limite, un campo di timestamp inserito appena prima dell'invio ed un running number, utilizzato per contare il numero di pacchetti arrivati.

Il secondo programma, receiver, è costituito invece da due thread:

- uno ad alta priorità, in ascolto sulla porta, che svuota prontamente i buffer del sistema operativo su buffer pre-allocati di dimensione superiore a quella necessaria per il raccoglimento di tutti i flussi, apponendo un timestamp ad ogni pacchetto ricevuto.
- uno a bassa priorità, che scrive i dati ricevuti su disco rigido, per l'analisi dei dati e statistiche.

All'interno della suite software ci sono anche gli script shell che raccolgono automaticamente I dati dai nodi sender e receiver e li organizzano in cartelle con data e ora di inizio del test.

Infine è stato creato uno script scritto in Octave, chiamato spider, che analizza in maniera ricorsiva il file system alla ricerca dei dati dei test, ordinandoli e processandoli, per ricavarne statistiche, quali media e varianza sul tempo inter-partenza e inter-arrivo dei pacchetti e sulla percentuale di pacchetti persi, oltre a generare grafici per le valutazioni del caso.

3.10.3 Test

Le sessioni di test condotte, servono a mettere in luce particolari aspetti delle reti Ethernet in diverse configurazioni. Ogni test è formato da più sessioni, ciascuna delle quali differisce dalle altre per alcuni parametri, come il rate dei pacchetti, il numero degli host e la dimensione dei pacchetti. Ciascuna sessione è composta da un invio di 10.000 pacchetti ad un rate prestabilito, per poter dare una valenza statistica all'esperimento. I tre tipi di test che sono stati fatti sono:

- Test "crossed" (la cui topologia è schematizzata in Figura 3.10), il cui scopo è di testare la massima capacità di invio dati real-time su Ethernet. Inoltre serve per determinare l'effetto di un traffico best-effort su questa configurazione
- Test "Multiple Hosts With Hub" (la cui topologia è schematizzata in Figura 3.11), in cui si testa il comportamento di una rete con topologia bus nel caso di accesso concorrente di uno o più host che inviano pacchetti rate constrained. Questi test non prevedono l'utilizzo contemporaneo di traffico di disturbo best effort.
- Test "Performance switch" (la cui topologia è schematizzata in Figura 3.12), in cui si testano le prestazioni dello switch con 2 host che inviano flussi di pacchetti verso una destinazione. I test si concentrano sull'aumento del payload al fine di saturare la banda e verificare le prestazioni massime di throughput in una rete di questo tipo.

Nelle sessioni di test vengono creati flussi di dati anche con payload superiori sia alla minima dimensione supportata dagli host (576 byte), sia al massimo MTU definito da Ethernet [17] , in modo da testare gli effetti dell'IP Fragmentation sia su Hub che su Switch.

Tabella 3.4 Risultati (sub)ottimali del test del test "crossed"

Payload	FTP	Interdeparture time [µs]	Avg [µs]	Var	Arrived
800	No	110	111.03	0.00674	100%
500	No	50	81.69	0.00010	100%
500	No	85	85.11	0.00019	100%
500	Yes	85	84.57	0.10038	100%
50	No	80	80.00	0.00002	100%
50	No	50	75.27	0.00165	51.5%
50	Yes	80	79.81	0.04453	100%

Tabella 3.5 – Risultati (sub)ottimali del test "multiple host hub"

Payload	Hosts present	Interdeparture time [µs]	Avg [µs]	Var	Arrived
50	2	140	144,84	0.002179	96%
50	2	150	149.99	0.000063	100%
500	2	140	167,16	0.003741	81%
500	2	150	149.99	0.000076	100%
800	2	170	169.96	0.000034	100%
1200	2	220	219.99	0.000031	100%
1600	2	340	340.00	0.033576	100%
2000	2	390	389.99	0.00447	100%
50	3	210	209.98	0.000259	100%
500	3	240	239.98	0.000357	100%
800	3	260	260.00	0.0005	100%
1200	3	340	344.43	2.515014	99.5%
1600	3	510	510.93	3.288083	99.8%
2000	3	620	620.06	3.777209	99.9%
50	4	260	259,94	0.00419	100%
500	4	300	299,97	0.001252	100%
800	4	340	339,93	0.000963	100%

Tabella 3.6 - Risultati (sub)ottimali del test "performance switch"

Payload	Interdeparture time [µs]	Avg [µs]	Var	Bandwidth [Mbps]
800	110	110,44	0.00002	63,64
1200	140	140.46	0.00007	72,86
1600	180	180.41	0.00005	74,44
2000	210	212.24	0.00005	79,05

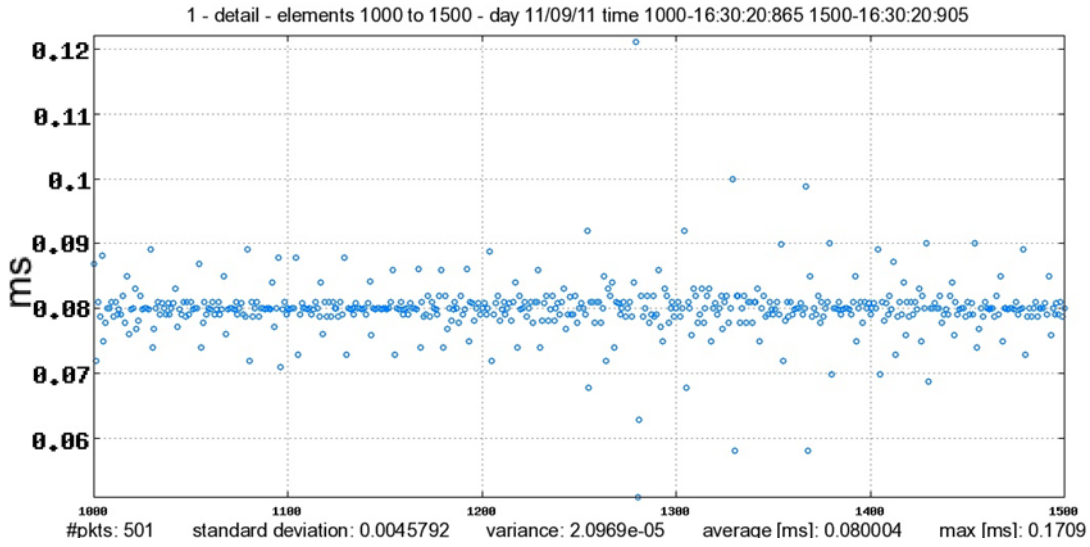


Figura 3.13 - Dettaglio di una sessione del test "crossed" sui tempi dei pacchetti di 50 byte ricevuti ad un intervallo di 80 µS

Le statistiche e i risultati dei test si basano sulla misurazione del tempo al ricevitore tra la ricezione del pacchetto i-esimo e i+1-esimo, misurandone il valor medio e la varianza, che assumono valore statistico, visto l'elevato numero di pacchetti su cui è eseguito il test. Vengono inoltre creati dei grafici per la visualizzazione della distribuzione dei pacchetti nel tempo, basandosi sul tempo inter-arrivo, che aiutano nella valutazione di irregolarità e di eventuali ritardi (i cui esempi sono in Figura 3.13 e Figura 3.14).

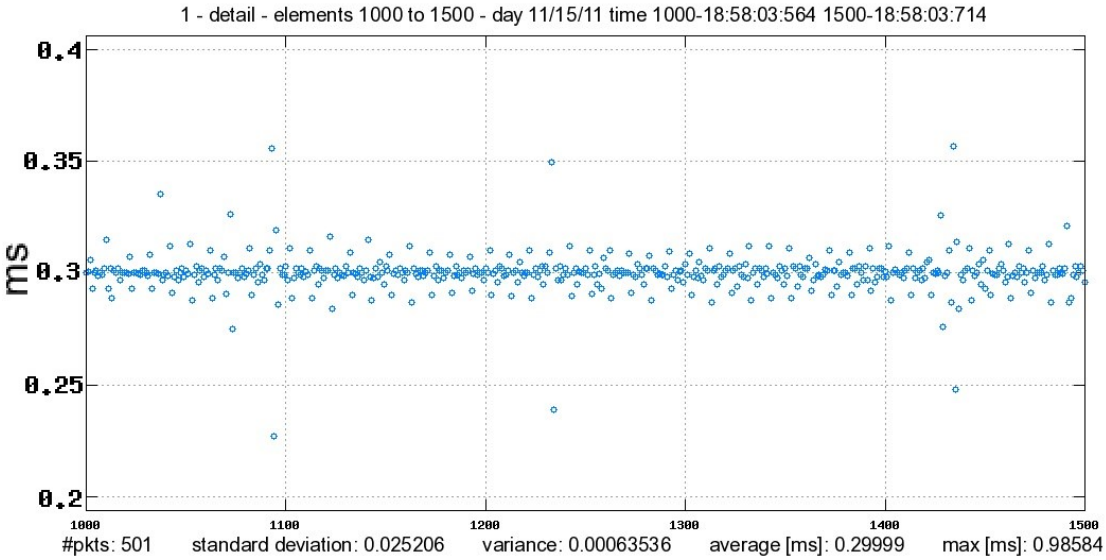


Figura 3.14 - Dettaglio di una sessione del test "multiple host hub" con 4 sender a 500 byte inviati ad intervalli di 300 µs

Le tabelle Tabella 3.4, Tabella 3.5 e Tabella 3.6 mostrano solo una parte dei risultati di tutti i test, omettendo alcuni risultati ottenuti con test che utilizzavano parametri fuori dai range, per cui la ricezione dei pacchetti era inferiore al 100%, ed evidenziando i casi limite, dove la differenza di microsecondi tra un pacchetto e l'altro può significare la non ricezione completa di tutto lo stream di dati.

3.10.4 Risultati – test "Crossed"

Il test Crossed è stato fatto per determinare le massime prestazioni di Ethernet con TCP/IP, i cui risultati verranno poi utilizzati per scopi di comparazione. I risultati più importanti di questo test sono riportati in Tabella 3.4, dimostrando, con questa semplice topologia, le grandi capacità di Ethernet.

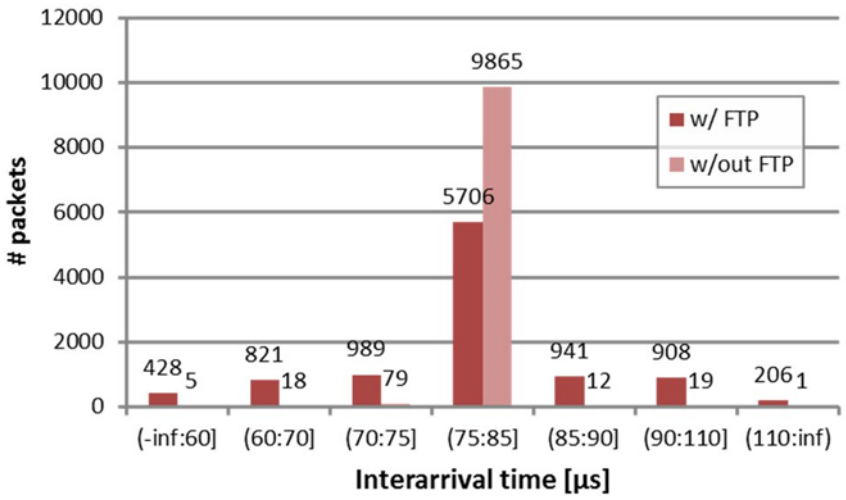


Figura 3.15 - Comparazione tra i flussi dati con o senza traffico FTP di disturbo.

In questa situazione ideale, l'unica causa di perdita di pacchetti può essere trovata nelle code hardware del controller ethernet o del buffer software presente nel driver di basso livello incluso nella distribuzione di Linux, ed è dovuta quindi ad un limite hardware della periferica o a un limite di dimensione del buffer software.

La Figura 3.15 mostra la distribuzione dei pacchetti da 50 byte inviati a intervalli di 80 µS da cui si può dedurre:

- Le performance molto elevate dal punto di vista del determinismo, visto che più del 98% dei pacchetti viene ricevuto entro un intervallo di ±5µs;
- Il traffico non regolato FTP (che potenzialmente può occupare l'intera banda) aumenta la varianza dell'arrivo dei pacchetti, diminuendone il determinismo. Rimane comunque un buon risultato, visto che il 76% dei pacchetti viene ricevuto entro un intervallo di ±10µs.

3.10.5 Risultati – test "Multiple host hub"

Nel test "multiple host hub" è stata testata una topologia a bus logico, con un numero crescente di flussi, uno per host, che si intersecano sulla rete.

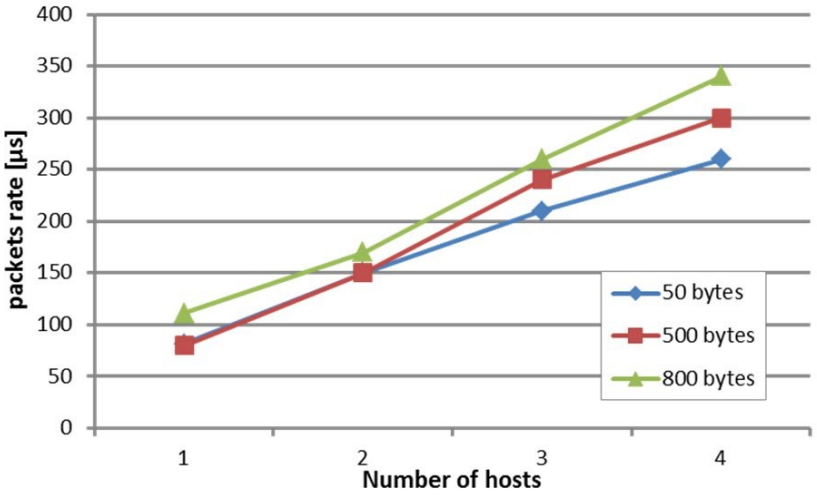


Figura 3.16 - Trend per numero di flussi sul test "Multiple Host Hub"

La Tabella 3.5 mostra che al di sotto di una certa soglia, i 140 µs, le spedizioni comunque falliscono, mentre se si rimane al di sotto del MTU del sistema (1500 byte) si può notare un trend quasi lineare in base al numero di flussi e al tempo di inter partenza medio dei pacchetti (come si vede in Figura 3.16), mantenendo fissa la dimensione. Al di sopra di questa linea le trasmissioni avvengono correttamente, mentre al di sotto abbiamo il fallimento della comunicazione; ciò è dovuto alla perdita di pacchetti causati da un tuning sbagliato della rete, cioè da un tempo inter partenza dei pacchetti troppo basso rispetto al numero degil host. La dimensione del pacchetto trasla sull'asse y il tempo minimo di inter partenza dei pacchetti, fino ad arrivare alla soglia del MTU.

Aumentando la dimensione del pacchetto oltre l'MTU si ha un degrado delle prestazioni, in termine di varianza, che aumenta di 4 ordini di grandezza, e in termine di tempo minimo inter-pacchetto per avere una trasmissione senza perdite. L'aumento della varianza è appunto dovuto all'IP Fragmentation che invia più pacchetti su Ethernet, aumentando la probabilità di collisione e diminuendo quindi il determinismo.

3.10.6 Risultati – test "Performance Switch"

Nel test "Performance Switch" lo scopo è diverso rispetto agli altri, poiché all'interno di una rete a stella è assente il fenomeno della collisione dei pacchetti. In questo test due host (con due flussi) testano le caratteristiche dello Switch, spingendosi fino al limite della banda occupabile. La Tabella 3.6 mostra che le

performance sono superiori, se comparate agli stessi risultati ottenuti con due flussi su hub, sia in termini di minimo tempo inter-pacchetto, sia in termini di varianza.

La Banda in Tabella 3.6 è calcolata usando la seguente formula, dove P è il payload del pacchetto, H_{Eth}, H_{IP}, H_{UDP} e H_{RT} sono rispettivamente le dimensioni degli header Ethernet (28 Byte), IP (20 Byte), UDP (8 Byte) e del protocollo (5 Byte), $\bar{\tau}$ è la media del tempo inter-arrivo dei pacchetti e S il throughput.

Equazione 3.1 - Calcolo del througput

$$S = \frac{((P + H_{Eth} + H_{IP} + H_{UDP} + H_{RT}) * 8)}{\bar{\tau}}$$

Qui la banda è importante e sembra crescere linearmente con il traffico offerto, fino a raggiungere la capacità massima dello switch, che è inferiore da quella teorica di Ethernet, che è comunque ragguardevole, poiché si attesta intorno agli 80 Mbps.

3.10.7 Conclusioni

I test “Crossed” hanno fornito le performance di Ethernet in una situazione quasi ideale: diminuendo il tempo inter-pacchetto sotto il limite del 80 µs si causa una perdita di pacchetti, probabilmente dovuta a problemi nelle code hardware della periferica Ethernet o del buffer del driver di basso livello di Linux, mentre negli altri casi si nota che il tempo medio di arrivo inter-pacchetto è molto vicino a quello della trasmissione. Confrontando le prestazioni con una rete CAN a 1 MBit (inutilizzabile per un’applicazione reale, per i vincoli di lunghezza massima del segmento di rete), è un ottimo risultato: su Ethernet si possono inviare messaggi di comando o di stato con payload di 50 bytes ad una frequenza di 12,5 kHz, supportando nel contempo un traffico FTP di disturbo non limitato in banda, contro un pacchetto di 8 byte ad una frequenza massima di circa 4 kHz del CAN, senza alcun disturbo.

Aumentando il numero dei flussi si è in grado di avvicinarsi ad una situazione realistica e anche in questo caso Ethernet riesce a dare prestazioni superiori per quanto riguarda il throughput (ovviamente) ma rimanendo con un determinismo elevato. Sui protocolli ISO 11783 il massimo rate di messaggi consentito per pochissimi messaggi è di 50 Hz, mentre la maggior parte è a 10 Hz, mentre su Ethernet si riescono a inviare 4 flussi a 3 KHz, con un trend quasi lineare, dove all’aumentare il numero dei flussi cala proporzionalmente la massima rate dei messaggi. Se si vuole invece un maggiore determinismo ed immunità alle perdite di pacchetti gli switch possono essere un’ottima soluzione.

3.11 Task 1 Proof of concept

3.11.1 Introduzione

Partendo dalle conclusioni del task precedente, si può pensare allo Stack TCP/IP come una possibile soluzione per le reti del futuro, visto l'ampio margine di miglioramento e la capacità di gestire flussi a frequenza molto elevate, fino a due ordini di grandezza superiori a quelle gestite da una rete ISO 11783 odierna.

I risultati ottenuti sono promettenti, però, affinché essi possano essere validati in modo certo, è necessario fare ulteriori test, questa volta con casistiche reali e una vera rete ISO 11783 ricercando anche condizioni di *worst case*. Per questo motivo sono stati creati dei test ad-hoc come proof-of-concept per dimostrare la realizzabilità di una rete ibrida CAN-Ethernet su cui gira il protocollo ISO 11783. Lo scopo di questo task è quindi di verificare che l'introduzione di un nuovo layer fisico, a fianco di uno già esistente e testato, non porti a malfunzionamenti dovuti a ritardi e nel contempo verificarne la bontà dell'implementazione.

La situazione di particolare interesse, dal punto di vista del proof of concept, è l'analisi di prestazioni della rete ibrida durante un trasferimento dati effettuato con l'Extended Transport Protocol, descritto nel capitolo **Errore. L'origine riferimento non è stata trovata.**.

A questo stadio della ricerca non è ancora stato realizzato un porting completo dello stack, affrontato nei task successivi, bensì è stato realizzato un incapsulamento dei messaggi CAN all'interno di pacchetti UDP, al fine di testare che i meccanismi di base del protocollo continuino a funzionare correttamente, anche in presenza di reti ibride, e poter generare delle statistiche. Tale approccio semplifica la fase implementativa e non sfrutta a pieno le caratteristiche di flessibilità di dimensione dei datagrammi e il routing dei pacchetti offerto del protocollo.

3.11.2 Test bench

Il banco di test deve riprodurre una rete reale e un vero caso d'uso, pertanto si sono utilizzati dei componenti commerciali e certificati ISO 11783 di progetto del dottorando, e quindi modificabili nelle modalità che saranno descritte nei paragrafi seguenti.

Figura 3.17 - Foto del Virtual Terminal V2 certificato e successivamente modificato per il test-bench

All'interno di IMAMOTER, è stato sviluppato dal dottorando un Virtual Terminal V.2 basato su architettura iMX38 di Freescale ARM11 a 400 MHz con distribuzione Linux con kernel 2.6.23 per la azienda CO.BO. Tale Virtual Terminal ha ottenuto la certificazione ISOBUS dall'ente di certificazione tedesco DLG nel 2010, ed è stato commercializzato a partire dall'anno successivo. Esso ha a disposizione due interfacce CAN, un'interfaccia Ethernet 10/100 e due porte seriali UART. Vista la partecipazione di IMAMOTER nello sviluppo di tale applicazione si è potuto modificarne le chiamate a funzione dei driver CAN, reindirizzandole ad un componente interno realizzato ad-hoc, con funzioni di proxy trasparente. Il proxy incapsula i pacchetti CAN in uscita in pacchetti UDP e sta in ascolto su una porta UDP per elaborare i pacchetti ricevuti e trasformarli in finti messaggi CAN, per inviarli infine ai livelli superiori. In questo modo l'applicazione non ha percezione del cambio di layer fisico, e si può utilizzare il proxy sia per dimostrare la fattibilità di una rete di questo tipo, sia per poter ottenere statistiche senza dover modificare l'applicativo, né lo stack ISO11783. Non avendo il pieno controllo del dispositivo, come ad esempio i sorgenti dei driver CAN e del kernel, non è stato possibile patchare il kernel per avere a disposizione i timer ad alta risoluzione per poter avere delle statistiche avanzate. Per questo motivo tali statistiche sono state raccolte attraverso strumenti di sniffing su Ethernet montati su altre unità della rete.

Figura 3.18 - Foto del Gateway CAN-Ethernet

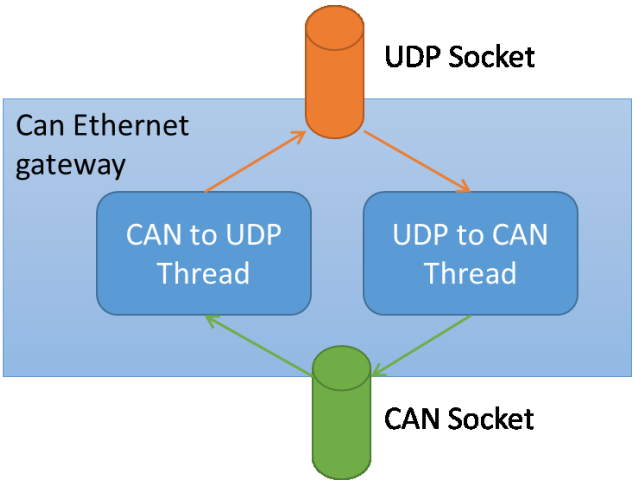


Figura 3.19 - Schema di funzionamento del Gateway CAN-Ethernet

Il secondo elemento fondamentale, costruito ad-hoc per il test bench è il gateway CAN-Ethernet, realizzato attraverso una ECU automotive grade (Figura 3.18) basata su:

- ARM Cortex A8 Architecture
- AM3505@500Mhz Cpu
- Linux Ångström v 2.6.32
- SocketCAN
- 2 Can e 1 Ethernet 10/100
- 4 UART

L'architettura scelta è comoda sia per poter testare la velocità di una distribuzione Linux per Embedded alle prese con messaggi real-time, sia per alcune features come le socket CAN, una patch del kernel che aggiunge la famiglia AF_CAN fra le famiglie di socket e permette di manipolare il CAN-BUS come una socket *raw*, quindi con tutte le facitilites fornite da questa struttura fondamentale, quali read, write, operazioni bloccanti, utilizzo di select, il tutto da user space.

Il gateway è stato realizzato come programma in user space, con priorità RT. Il programma è stato scritto in GNU C, con ampio uso delle system call di linux e delle socket, oltre ad appoggiarsi alla liberia *Pthread*. Tutte le priorità sono state settate per diminuire al massimo le latenze dovute al sistema operativo.

La struttura del programma, dato il compito di gateway assegnato alla unità, è snella ma comunque guidata da scelte mirate dalle necessità di real-time e di latenza dell'applicazione. Esso è costituito di due thread e

due socket, una CAN ed una UDP. Un thread utilizza una chiamata bloccante di lettura su socket CAN, incapsula il messaggio e lo scrive attraverso la primitiva *Sendmsg* su UDP; l'altro rimane in stato di listen bloccante su una porta su UDP, fa l'unmarshalling del pacchetto CAN e lo invia con write su socketCAN.

Il terzo elemento è il Working Set Master, cioè la ECU su CAN che si interfaccia come slave del Virtual Terminal, inviando il proprio Object Pool (**Errore. L'origine riferimento non è stata trovata.**) attraverso un xtended Transport Protocol. Questo elemento è stato realizzato attraverso il plug-in ISO 11783 della suite CANoE, che permette di simulare intere rete CAN e implementa al suo interno ECU simulate che comunicano in base alla norma J1939 e/o ISO 11783.

La suite CANoE e l'hardware CanCaseXL di Vector Informatik è l'unico strumento professionale esistente per il test di reti ISOBUS ed è ampiamente utilizzata dai maggiori costruttori di veicoli agricoli per sviluppare e eseguire il debug di una rete ISO 11783 o SAE J1939.

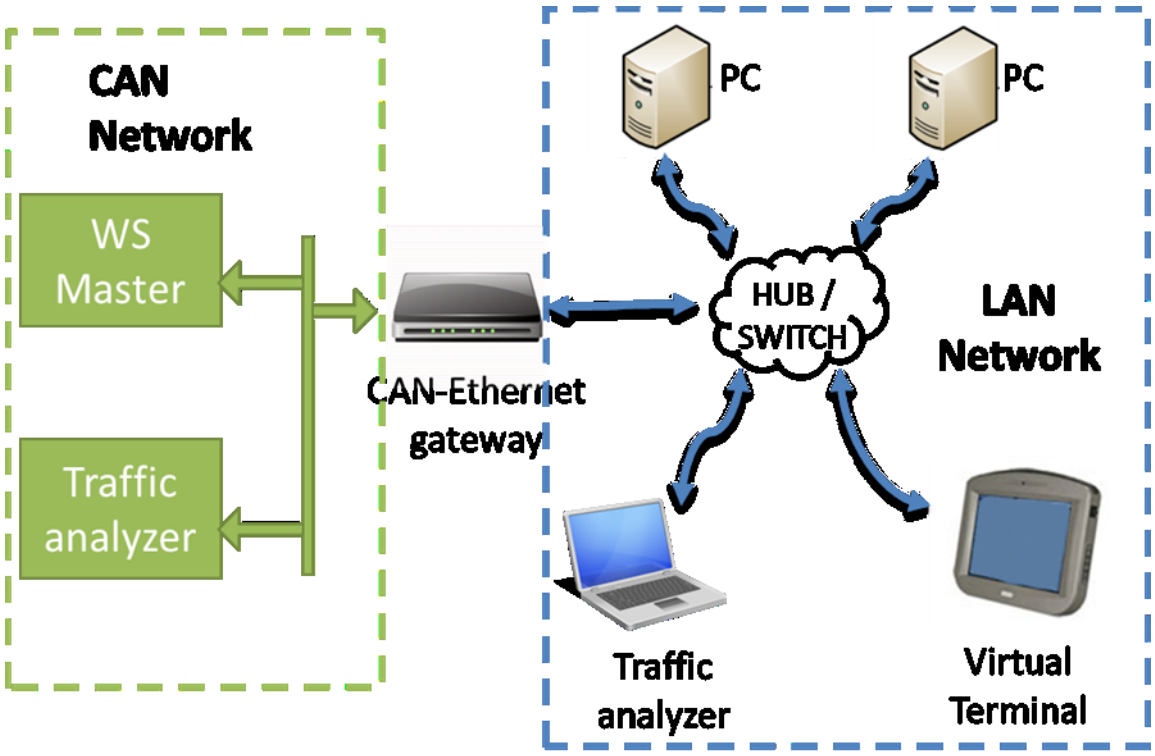


Figura 3.20 - Struttura generale del test-bench

Grazie a questa suite è possibile creare degli script personalizzati in linguaggio CAN Access Programming Language (CAPL), per poter modificare il comportamento delle ECU simulate. Il suo utilizzo nel banco di test è stato fondamentale, in quanto è stato possibile creare sessioni automatiche di test che avrebbero altrimenti richiesto continui riavvii manuali di tutto il sistema o il taglio della rete, per terminare in maniera forzata la connessione Virtual Terminal – Working Set. La prima fase dei test, si concentra sul protocollo di

trasporto, che viene eseguito tipicamente una volta sola, per ottimizzare l'occupazione di banda e i tempi di caricamento del sistema, come spiegato nei capitoli precedenti.

Sono stati inoltre utilizzati due analizzatori di rete:

- Un PC con distribuzione Linux Debian con *WireShark* in modalità promiscua per la parte di rete Ethernet
- Un PC con hardware CAN-Sniffer Kvaser Memorator-Pro a due canali CAN e software E-Tracing Omnitracer, software professionale per lo sniffing di reti CAN e LIN.

L'uso degli analizzatori nelle due reti connesse al gateway ha reso possibile l'analisi comparativa dei dati presenti nelle due reti e delle prestazioni relative in ogni test.

Gli elementi di interconnessione dei nodi ethernet sono gli stessi del Task 0 e cioè:

- Un Hub 3-COM 3C16750B 10/100 Mbps, per simulare un'architettura a bus logico
- Uno Switch 3COM 3C16791A 10/100 Mbps Full-Duplex con strategia Store-and-Forward per creare l'architettura a stella

Gli ultimi elementi del banco di test sono gli elementi di disturbo, cioè due PC che in alcuni test genereranno un traffico FTP di disturbo, a banda non limitata, durante le comunicazioni CAN, per vederne gli effetti sulla comunicazione. L'architettura generale è schematizzata in Figura 3.20.

3.11.3 Test su trasferimento di Object Pool

Lo scopo di questi test è di analizzare le performance della rete ibrida in diverse condizioni e topologie. In particolare si vuole testare gli effetti della rete ibrida sul protocollo di trasporto più oneroso e band-consuming della norma ISO 11783, l'Extended Transport Protocol, utilizzato per l'invio dell'Object Pool di un Working Set ad un Virtual Terminal. Grazie alla suite CANoE si è creato appunto un Working Set Master in grado di eseguire tutte le operazioni necessarie per forzare l'invio automatico di Object Pool, cosa non possibile con una ECU reale, se non agendo manualmente sul Virtual Terminal per cancellare l'Object Pool ogni volta e riavviando la rete ad ogni sessione. Inoltre è stato possibile forzare le temporizzazioni del protocollo di trasporto alla massima velocità per portare al limite la rete CAN, fino alla saturazione; condizione peggiorativa (worst case) di quel che succede in realtà, poiché la maggior parte delle ECU ISO 11783 invia messaggi CAN ad un rate non superiore ai 200 Hz all'interno di protocolli di trasporto (quindi a priorità bassa), per non mandare in starvation altre comunicazioni allo stesso livello di priorità.

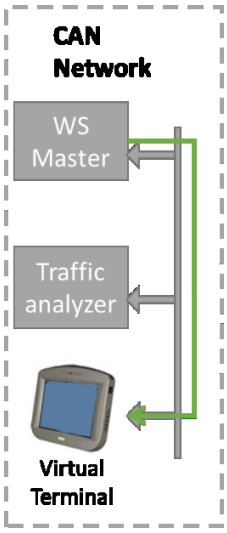


Figura 3.21 - Topologia "Only-CAN"

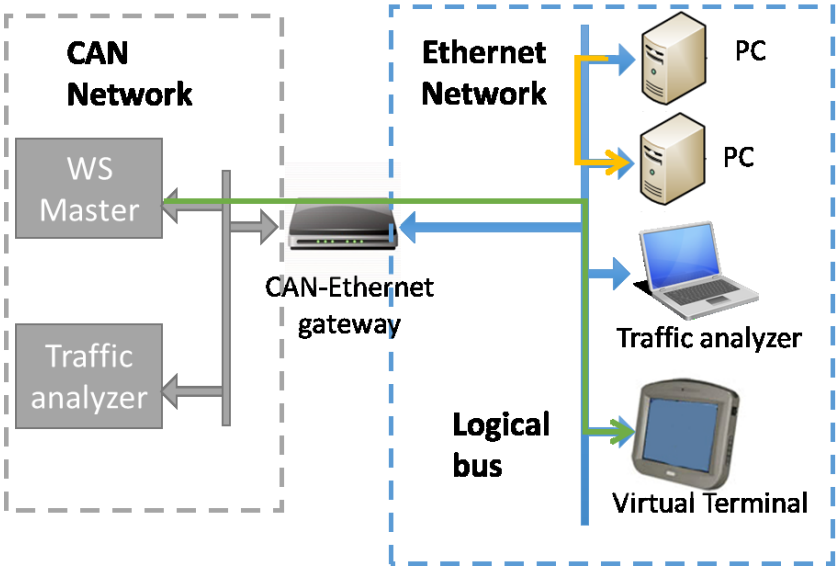


Figura 3.22 - Topologia "Hybrid Hub Network"

I test sono stati divisi in 3 sessioni così definite:

1. Test “Only CAN”, la cui topologia è schematizzata in Figura 3.21. In verde si vede il flusso dati da Working Set Master a Virtual Terminal. Lo scopo di questo test è di fornire i risultati di riferimento in condizione di rete unica in termini di tempi di invio dell’Object Pool.
2. Test “Hybrid Hub Network”, la cui topologia è schematizzata in Figura 3.22. Lo scopo di questo test è di verificare la bontà di una topologia di questo genere, che permette ritardi inferiori rispetto alla configurazione 3, e verificarne il comportamento in caso ideale e con traffico di disturbo (rappresentato in giallo sulla figura).

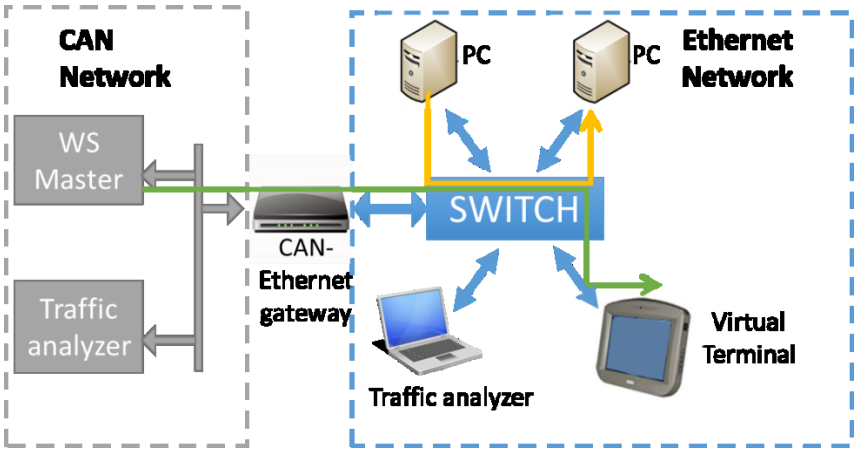


Figura 3.23 - Topologia "Hybrid Switch Network"

3. Test “Hybrid Switch Network”, la cui topologia è schematizzata in Figura 3.23. Lo scopo di questi test è la verifica del comportamento dello Switch, che introduce inevitabili ritardi sulla comunicazione, per verificare se essi possono provocare problemi ai meccanismi del protocollo e valutare le performance in caso di traffico di disturbo.

Il traffico di disturbo è stato creato con un trasferimento di 1 GByte di dati attraverso il protocollo SCP tra due host, per creare condizioni di stress della rete.

3.11.4 Risultati e conclusioni

Tabella 3.7 - Risultati dei Test del trasferimento di un Object Pool

	AVERAGE TRANSFER TIME [S]	RELATIVE STANDARD DEVIATION [%]	VARIANCE
CAN ONLY	9,2428	3,580	0,0013
ETHERNET (HUB)	9,4650	5,91%	0,0035
ETHERNET (SWITCH)	9,3286	3,86%	0,0015
ETHERNET (SWITCH + DATA TRANSFER)	9,3532	7,41%	0,0055

I risultati ottenuti in Tabella 3.7 sono le medie dei 100 trasferimenti di Object Pool provate nelle varie configurazioni. L’object pool trasferito è la configurazione di uno Sprayer commerciale, di dimensione 107 kbyte. In tutti i test si è spinta la rete CAN al limite, raggiungendo il 90% di banda occupata come mostrato in Figura 3.24.

Il test su CAN ha evidenziato un tempo medio di trasferimento di 9,24 secondi. Tale tempo è stato calcolato prendendo la differenza dei timestamp del primo pacchetto di RTS e il pacchetto EOMA. Tutti i test con rete ibrida hanno evidenziato un leggero aumento di tale tempo, dovuto ai ritardi introdotti da Gateway e Proxy su Virtual Terminal, comunque inferiore al 2,5%.

I test su HUB si sono rivelati peggiori rispetto a quelli su switch, per l’accesso concorrenziale al Bus dei pacchetti applicativi del Virtual Terminal e del Working Set Master durante il trasferimento. Questo è dovuto al fatto che l’accesso non regolato sul bus logico Ethernet ha prodotto delle collisioni casuali e delle conseguenti ritrasmissioni, fatto dimostrato da una maggiore deviazione standard dei risultati su HUB. Inoltre la topologia a bus logico si è rivelata assolutamente non deterministica in caso di traffico di disturbo concorrente. Infatti, in questa situazione, l’aumento delle collisioni ha portato il protocollo di trasferimento ad un Abort per timeout, facendolo quindi fallire.

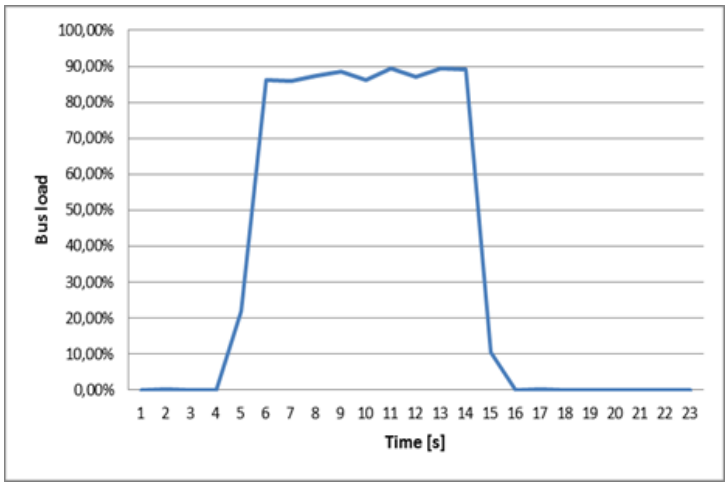


Figura 3.24 - Grafico occupazione banda su CAN durante il trasferimento

Viceversa i test su topologia a stella sono stati i migliori, in quanto in questo tipo di tipologia non ci possono essere collisioni di pacchetto, rendendo la comunicazione più deterministica ed introducendo solo un tempo di ritardo dovuto alla ritrasmissione del pacchetto. Il ritardo totale è dovuto in parte allo switch, quantificabile nell'ordine di decine di microsecondi, in parte al Virtual Terminal, a causa dell'uso di un Kernel non patchato RT_PREEMPT e il processo "Virtual Terminal" che quindi non è schedulato a priorità elevata, di fatto rendendolo sensibile alle latenze del sistema operativo e alla de-schedulazione del processo.

Un altro buon risultato è il tempo medio di trasferimento su topologia a stella in condizioni di traffico non regolato di disturbo che occupa il 90% della banda Ethernet, che aumenta di soli 10 millisecondi, rispetto alla rete in condizioni iniziali. La varianza è aumentata, indicando una perdita di determinismo, però rimanendo comunque a livelli accettabili.

3.11.5 Conclusioni

Nonostante questa prova sia stata eseguita in condizioni peggiorative rispetto ad una condizione reale, dai risultati si evince che si può creare una rete ibrida CAN-Ethernet. Questo proof-of-concept non è l'arrivo ma l'inizio di un vero e proprio porting dello stack ISO 11783 su TCP/IP. Nelle prove eseguite non sono state utilizzate tutte le accortezze usate nel Task 0, ma nonostante ciò il comportamento della rete è corretto e di fatto la comunicazione è perfettamente funzionante. I ritardi inseriti dalla rete sono trascurabili rispetto al tempo di trasferimento di un pacchetto CAN e rispetto al ritardo introdotto da ECU non regolate a dovere (ad esempio il Virtual Terminal). Nonostante il protocollo implementato sulla rete Ethernet non sia ottimale e abbia molti margini di miglioramento, il funzionamento della rete non è stato compromesso dall'introduzione di traffico di disturbo sovrapposto alla normale comunicazione funzionale.

3.12 Task 2 – Porting stack ISO 11783 over Ethernet

3.12.1 Introduzione

I task precedenti sono stati utili per dare l'indicazione che la strada intrapresa è corretta. Con il task 0 si è verificata la scalabilità e il determinismo necessari per sostituire il layer fisico CAN con il layer fisico Ethernet, mentre con il Task 1 si è verificata la possibilità di avere una rete eterogenea e capace di essere retro-compatibile con i sistemi legacy. Lo scopo di questo task è di effettuare un porting dei livelli bassi del protocollo ISO11783 su un nuovo protocollo basato su TCP/IP che possa fornire sia le prestazioni viste nel Task 0, sia la retrocompatibilità vista nel Task 1, privilegiando comunque un obiettivo di trasparenza per i livelli applicativi, l'aumento della sicurezza delle comunicazioni, la scalabilità e l'interfacciamento con reti e servizi che ad ora non sono standardizzati dalla norma.

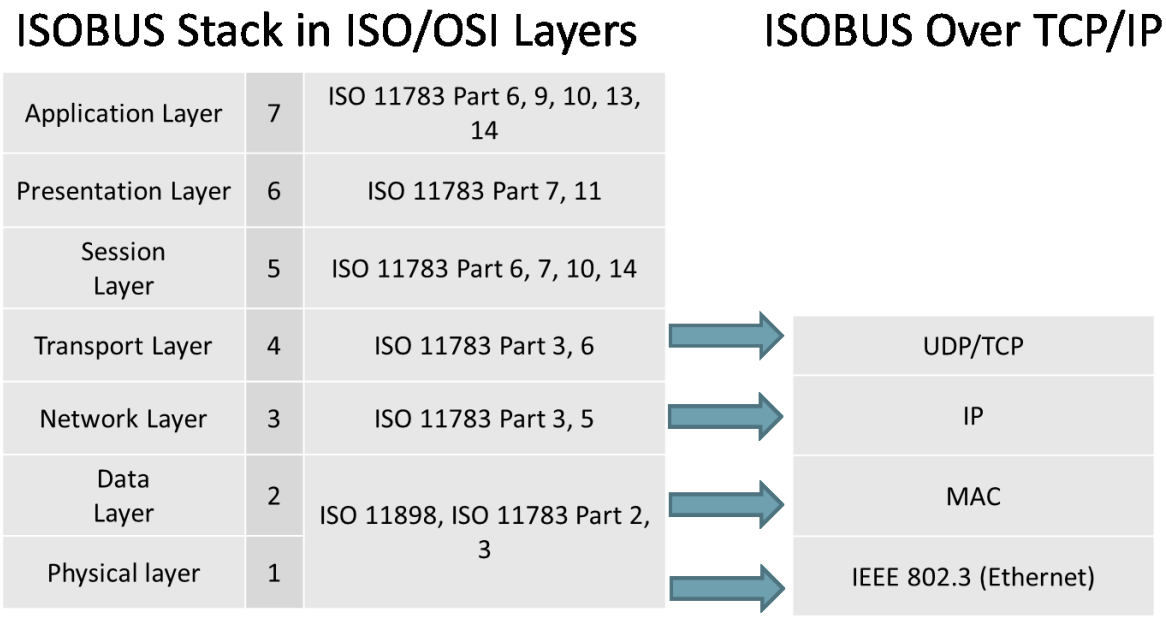


Figura 3.25 - Porting dello stack ISO11783 su Ethernet e TCP/IP

I problemi affrontati in questo task saranno correlati al fatto che i due protocolli, pur avendo caratteristiche apparentemente comuni come protocolli di trasporto, semantica e indirizzamento, hanno differenze sostanziali nei meccanismi in cui essi operano che devono essere colmate per fornire la necessaria trasparenza ai livelli superiori. In Figura 3.25 è presente una schematizzazione delle metodologie utilizzate per sostituire alcuni livelli dello Stack ISO 11783.

Lo scopo finale di questo task è di creare una rete eterogenea, dove tutti i vecchi servizi siano disponibili e al contempo introdurne alcuni di nuovi, pertanto i test eseguiti alla fine saranno puramente funzionali, al

fine di validarne l'architettura. Nei prossimi capitoli verranno affrontati tutti i problemi derivanti dal porting e saranno evidenziate le scelte fatte.

3.12.2 Gateway ISOBUS-IP

In una rete ibrida di questo tipo, l'unità fondamentale è la NIU, che si occupa del routing dei messaggi, re-impacchettamento e traslazione e assegnamento degli indirizzi. Il NIU oggetto della ricerca è stato battezzato come "Isobus-IP Gateway", in quanto esegue diverse operazioni non facilmente riconducibili ad una unità di interconnessione IP (repeater, bridge, router, gateway, proxy) o ISO 11783. Il nome Gateway non deve essere confuso né con quello utilizzato per definire un Gateway su Stack TCP/IP né con quello utilizzato su ISO 11783.

3.12.3 Indirizzamento

La norma ISO 11783 definisce i possibili indirizzi di livello 3 tra 0 e 253. 253 viene utilizzato come NULL Address, mentre 255 come Broadcast Address.

Tabella 3.8 - Tipi di reti nel protocollo IP

Class	Range of addresses	Number of addresses
A	10.0.0.0-10.255.255.255	2^{24} = 16777216 (-2)
B	172.16.0.0–172.31.255.255	2^{20} = 1048576 (-2)
C	192.168.0.0–192.168.255.255	2^{16} = 65536 (- 2)

Nel protocollo IP, si hanno a disposizione diverse classi di rete, definite nello standard RFC1466 [18] (Tabella 3.8). Tali classi di rete sono state standardizzate per poter arginare il problema di esaurimento degli IP pubblici, e definire delle classi di comodo in base al numero di nodi che la rete può ospitare. Ad esempio, nella classe C si possono definire 65536 reti di 65533 nodi.

Visto il numero di nodi che una rete di un trattore può sopportare è limitato a 30 per segmento, è lecito pensare che avere 253 nodi o ECU a bordo di un unico sistema trattore sia un'ipotesi non restrittiva per una rete del futuro (basti pensare ai nuovi progetti in cui si ipotizza un indirizzo per ogni ugello di una barra di uno sprayer). Per questo motivo l'architettura di rete pensata sta in una sottorete di Classe C a 24 bit di maschera, in grado di contenere fino a 253 indirizzi di livello 3 validi. Le altre sottoreti di Classe C potranno essere utilizzate per nodi non appartenenti allo standard ISO 11783, oppure per eseguire il routing di pacchetti Machine To Machine, in un'ottica di integrazione con reti Wireless ad-hoc.

Nel protocollo IP sono definiti due indirizzi particolari, uno equivalente alla subnet mask della rete, detto indirizzo di rete, ed uno broadcast, utilizzato per inviare un messaggio a tutti i nodi della rete.

Nel Gateway si rende quindi necessario eseguire una traslazione di indirizzo così descritta:

$$Address_{IP} = Address_{CAN} + 1 + Address_{NET}$$

3.12.4 Routing

Il gateway è responsabile del routing dei pacchetti, pertanto deve tenere due tabelle, una per gli indirizzi sulla rete Ethernet, un'altra per gli indirizzi su rete CAN.

Dalla parte CAN è impossibile pensare di poter utilizzare filtri hardware, dato il loro numero limitato e il numero elevato di combinazioni possibili di PGN. Il filtro viene fatto a livello software, utilizzando delle policy di routing per fare passare solo i pacchetti che sono indirizzati alla rete IP e viceversa.

Da lato Ethernet, si è scelto di utilizzare la feature di *IP aliasing*, cioè la possibilità di poter essere registrati con più indirizzi IP utilizzando lo stesso MAC address. Gli *aliased IP* corrispondono ai valori traslati degli indirizzi di livello 3 su rete ISO11783. Il vantaggio di utilizzare un simile approccio è il poter sfruttare i meccanismi del routing IP e dello Stack TCP/IP, senza dover fare modifiche. Utilizzando gli aliased IP inoltre si possono gestire i pacchetti a User Level, per semplificare il meccanismo di ripacchettizzazione da e verso CAN.

Su ISO 11783 ci sono due tipi di messaggi broadcast: i messaggi PDU1 con destination address uguale al global address e i messaggi PDU2. Mentre i primi si possono considerare broadcast senza riserve, i secondi in realtà, da un punto di vista semantico sono dei messaggi multicast. I messaggi PDU2 hanno solo un indirizzo sorgente e il PGN a 18 bit che ne identifica un servizio. Tale servizio può essere di interesse di alcune ECU mentre di altre no: su una ECU dalla limitata capacità computazionale vengono usati dei filtri hardware CAN su questi tipi di messaggi, in modo da gestire solo quelli che portano un contenuto informativo utile. Un esempio può essere l'informazione della velocità del trattore, pubblicata su un messaggio PDU2, che può essere utilizzata da una ECU che esegue una lavorazione di precisione basata sulla velocità (come ad esempio una macchina seminatrice - *Seeder*) mentre essere completamente ignorato da una ECU montata su un aratro. Per questo motivo i messaggi PDU2 possono essere considerati a pieno titolo dei messaggi Multicast su IP. Pur essendo una conseguenza logica, l'interpretazione dei PDU2 come Multicast presenta dei problemi in una rete ibrida, in quanto il gateway dovrebbe essere a conoscenza di quali ECU nella rete CAN sono interessate ad un particolare servizio multicast, cosa non possibile, visto che i filtri hardware sono interni a ciascuna periferica CAN e non è noto a priori quale

tipologia di unità di controllo sarà interessata alle diverse tipologie di informazioni immesse e circolanti nella rete. Per questo motivo i PDU2 vengono trattati come messaggi broadcast a tutti gli effetti dal gateway.

Per mantenere la comunicazione in entrambi i sensi, il gateway deve applicare le seguenti policy di routing:

- Se il messaggio ricevuto da CAN ha un PGN che identifica un pacchetto PDU1 e il destination address coincide con uno degli indirizzi della tabella degli indirizzi Ethernet, il pacchetto viene processato e trasmesso come pacchetto UDP verso l'host indicato come messaggio unicast
- Se il messaggio ricevuto da CAN ha un PGN che identifica un pacchetto PDU1 e il destination address è l'indirizzo globale (GLOBAL_ADDRESS o BROADCAST_ADDRESS), oppure il PGN è di tipo PDU2 (broadcast), il messaggio viene trasmesso come pacchetto UDP con indirizzo di destinazione broadcast.
- Se viene ricevuto un pacchetto da UDP vuol dire che il destination address si trova su CAN o è Broadcast, pertanto viene immesso un messaggio CAN con l'indirizzo IP traslato come source address, e a seconda del tipo di PGN ricevuto viene mandato un messaggio PDU1 o PDU2.

3.12.5 Address Claiming

L'*address claiming* su rete ISO11783 è un meccanismo di gestione degli indirizzi distribuito, che contrasta col modello di gestione degli indirizzi all'interno di una rete IP dove gli indirizzi possono essere reclamati (impostando un IP statico) oppure ottenuti in "prestito" da un DHCP server, in maniera centralizzata.

I due meccanismi sono molto diversi sia per la modalità, sia perché i fondamenti su cui si basano sono profondamente diversi. Mentre su ISO 11783 c'è la contesa di un indirizzo logico attraverso l'Address Claim, dove il contendente con il NAME più basso (più vecchio e quindi più prioritario) vince l'indirizzo, in una rete IP, nel caso in cui due host reclamino lo stesso indirizzo si ha una contesa non risolta, rendendo l'indirizzo IP inutilizzabile finchè uno dei due host cambia IP volontariamente.

Questo problema potrebbe essere risolto creando un protocollo di livello 2-3 ad-hoc, basato sui MAC address, che rappresentano un numero seriale identificativo unico di una periferica di rete. Per quanto il NAME, essendo un identificativo univoco di ciascuna ECU, indipendente dalla rete, possa ricordare molto la definizione del MAC di una scheda di rete, essi presentano le seguenti differenze:

- L'indirizzo MAC su IPv4 è di 48 bit, mentre il NAME è di 64 bit

- L'indirizzo MAC è diviso in due parti, la parte di maggior peso di 24 bit detta OUI (Organization Unique Idientifier), mentre gli altri 24 bit costituiscono il numero seriale. Il NAME possiede un campo simile a OUI, detto Manufacturer Code, di soli 11 bit e il numero seriale di 21 bit
- I primi bit del byte di maggior peso dell'indirizzo MAC vengono utilizzati per identificare se il MAC è unico o assegnato localmente e se rappresenta un MAC speciale per multicast. Il bit di maggior peso del NAME indica la possibilità di cambiare indirizzo.
- Il NAME viene utilizzato esclusivamente per l'Address Claiming (il collante fra il livello 2 e il livello 3), mentre il MAC può essere utilizzato per comunicazioni broadcast attraverso la sequenza FF:FF:FF:FF:FF:FF e per comunicazioni punto-punto in una rete *switched*.
- Il NAME identifica il tipo di ECU, a che classe appartiene (Marina, Agricoltura, Macchine Off-Road ecc..) e la sua funzionalità.

Risulta quindi evidente che l'unica funzione che hanno in comune è quella di rappresentare un numero identificativo unico della ECU o host all'interno della rete.

Poiché il contenuto informativo in termini di unicità del NAME è di 32 bit (numero seriale e manufacturer code), si potrebbe pensare di comprimere il NAME in un MAC, creando una funzione che raccolga i campi necessari al funzionamento. Se questo è possibile per una rete IP, tale compressione potrebbe portare a problemi di incompatibilità dovuti alla funzione inversa che trasforma il MAC in NAME, perdendo alcuni campi che non sono strettamente necessari a garantire l'unicità, ma vengono utilizzati da altre ECU per poter discriminare una ECU con servizi interessanti da una che non li offre. Per esempio, se una ECU è un GPS, all'interno del NAME è specificata tale funzione, per cui le altre ECU della rete interessate alla posizione GPS, possono impostare i filtri CAN Hardware basati su indirizzo sorgente già all'address claim.

Visto che la via della funzione di trasformazione 1:1 MAC-NAME non è percorribile, se non col rischio di perdere qualche funzionalità sulla rete legacy, è necessario cambiare approccio. Creare un protocollo ad hoc livello 2 basato su MAC per la contesa dell'indirizzo IP basandosi su NAME potrebbe essere una sfida interessante, però ci si allontanerebbe troppo dallo standard TCP/IP, e non si privilegerebbe la riusabilità di servizi e strutture già esistenti. La ricerca è proseguita su un metodo che potesse sfruttare le tecnologie esistenti applicando delle modifiche, in modo da poter essere compatibile sia con la rete eterogenea ISO 11783 sia con una rete standard IP.

Tralasciando la configurazione manuale degli indirizzi IP, la cui strada non è assolutamente percorribile, la sfida quindi è di realizzare attraverso un metodo standard un assegnamento degli indirizzi centralizzato che mantenga la coerenza con una parte di rete ove l'assegnamento degli indirizzi è distribuito.

Il protocollo DHCP [19] è molto complesso, poiché attraverso un solo servizio di discovery, in base alle features gestite dal DHCP server, è possibile conoscere indirizzi IP di active directory, NFS, stampanti, DNS gateway per una configurazione completamente automatica dell'host appena entrato. Una delle cose interessanti del protocollo DHCP, è la possibilità di utilizzare delle DHCP_OPTIONS, alcune standard, altre proprietarie. In questo modo si mantiene la compatibilità con host non capaci di comunicazioni ISO 11783 su IP, avendo la possibilità di assegnarvi un indirizzo IP su un'altra sottorete, e al contempo riconoscere gli host capaci di comunicazioni ISO 11783.

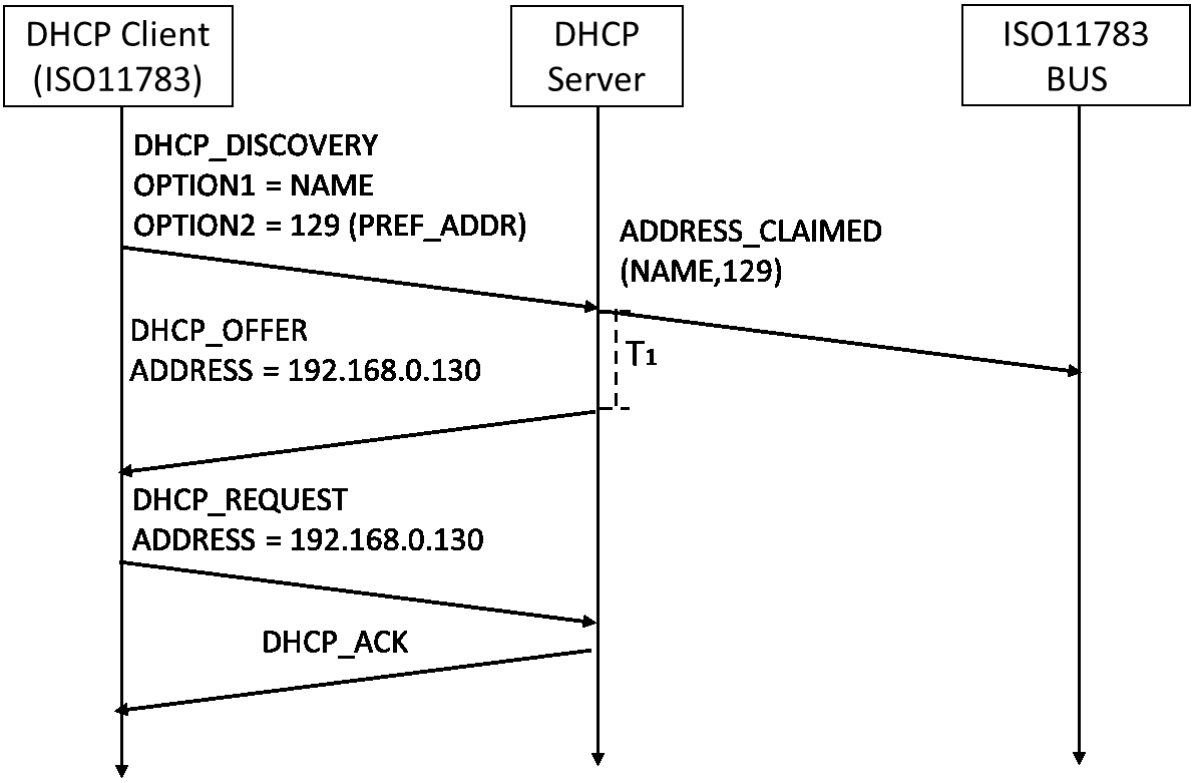


Figura 3.26 - Esempio di Address Claiming su rete ibrida, senza contese

L'idea di base è quella di utilizzare le DHCP_OPTIONS di tipo Vendor Specific, supportate dallo standard, per poter passare il NAME e l'indirizzo preferito di livello 3, all'interno del pacchetto DHCP_Discover al DHCP Server situato sul Gateway, il quale si preoccuperà di eseguire la procedura di address claiming all'interno della rete ISO 11783, per conto dell'host (Figura 3.26). Se il DHCP server ha successo risponde con una DHCP_Offer con l'indirizzo IP traslato (di cui si è parlato nel capitolo precedente). L'host si salverà tale indirizzo ottenuto come Preferred Address, in modo che alle riaccensioni successive richieda un indirizzo in precedenza libero.

La procedura di Address Claim richiede 1,25 secondi senza contese affinché un indirizzo possa essere considerato assegnato. Il DHCP Server si prenderà il carico di eseguire la procedura completa, fino a trovare

un indirizzo libero. Tale procedura potrebbe richiedere decine di secondi, ma ciò non inficia il protocollo DHCP, che supporta tempi lunghi di timeout. Tale modalità è quella scelta e sarà descritta nel capitolo relativo alla implementazione del sistema.

3.12.6 Protocolli di trasporto e incapsulamento

I protocolli di trasporto disponibili su ISO 11783 sono 3, due uni-cast ed uno broadcast. Per l'invio di messaggi uni-cast senza protocollo di trasporto è stato scelto di utilizzare pacchetti UDP con dimensione IP pari a 576 byte , per due motivi:

- È inferiore al MTU di Ethernet, che è di 1500. In questo modo si evita la frammentazione IP, dannosa per applicazioni in cui determinismo e real-time sono fondamentali (si veda il Task 0)
- È la minima dimensione che deve poter essere gestita da un host [20], aumentando così la compatibilità con sistemi embedded in cui la RAM è una risorsa scarsa.

Bit Byte	0 - 7		8 – 15	16 - 23	24-31
0 – 3	Version	IHL = 5	TOS	Length <= 576	
4 – 7	Identification			Flags + Fragment Offset	
8 – 11	TTL		Protocol = UDP	Header Checksum	
12 – 15	Source Address				
16 – 19	Destination Address				
20 – 23	Source Port			Destination Port	
24 – 31	Length			Checksum	
32 – 35	BAM-ID		PGN		
36 – 40	Unix timeval (s)				
41 – 44	Unix timeval (us)				

Figura 3.27 - Struttura di un pacchetto ISO1173-UDP generico

La struttura del pacchetto ISO11783-UDP (Figura 3.27) è formata da un normale pacchetto UDP, a cui si aggiunge un header per la gestione del protocollo ISO 11783. La lunghezza massima del pacchetto totale è di 576 byte + header IP. Impostando nessun opzione IP e contando l'header di UDP e di ISO 11783, si ha che la massima quantità di dati trasmissibile attraverso un singolo messaggio è di 532 byte.

All'interno dell'header ISO11783-UDP sono stati inseriti i campi:

- PGN, per poter redirezionare i pacchetti al livello applicativo in base al contenuto informativo;
- *Timeval*, con secondi e microsecondi per la validazione di messaggi real-time
- BAM-ID, per ricostruire i pacchetti del protocollo broadcast

Il gateway deve quindi impacchettare tutti i dati provenienti da Transport Protocol o no in pacchetti ISO11783-UDP e viceversa. Con 532 byte si coprono tutte le necessità di pacchetti di comando del vecchio protocollo. Infatti sia i messaggi TP che ETP non richiedono real-time, bensì sono dei veri e propri trasferimenti di dati di configurazione. In realtà alcuni di essi, come ad esempio alcuni messaggi generati dal VT, sono messaggi di controllo o di configurazione ma non richiedono TP maggiori di 260 byte.

Per quanto riguarda i protocolli di trasporto unicast TP maggiori di 536 byte e ETP, il gateway fa da proxy, cioè apre una connessione TCP ad un nodo IP, mentre dall'altra parte gestisce il TP o ETP. Il TCP è altamente affidabile, ma non garantisce il real-time, cosa che non è richiesta nemmeno dai protocolli di trasporto su ISO 11783.

Il protocollo BAM (*Broadcast Announcement Message*), cioè il protocollo di trasporto broadcast, non ha requisiti di affidabilità né di correttezza, pertanto è stato tradotto attraverso l'invio di una serie di pacchetti ISO11783-UDP, utilizzando il campo BAM-ID per la ricostruzione del messaggio. Se è richiesto un numero di byte inferiori a 532, viene utilizzato un normale pacchetto ISO11783-UDP broadcast.

3.12.7 Priorità

La prioritizzazione all'interno della norma ISO 11783 è definita da 3 bit ed è gestita direttamente in hardware dal controller CAN. Nel pacchetto IP esiste il campo Type Of Service, formato da 8 bit. Tale campo, inizialmente utilizzato per definire 8 livelli di priorità con i 3 bit di minor peso, è stato sostituito dal campo Diffentiated Services (DiffServ) a 6 bit più due bit per il controllo di congestione (ECN). DiffServ utilizza 3 soli bit per definire la classe di servizio, lasciando liberi i 3 bit di priorità per retrcompatibilità.

La prioritizzazione di ISO 11783 viene quindi mappata sui 3 bit di priorità eseguendo il complemento a 1, visto che su IP a numeri di TOS più elevati corrispondono priorità più elevate, al contrario del CAN.

3.12.8 Nuovi messaggi ad alto throughput

Per verificare la possibilità di utilizzare l'alto throughput garantito dalla rete Ethernet sono stati introdotti dei nuovi messaggi di controllo distribuito, non presenti nello standard ISO 11783. A questo proposito è stato usato un PGN libero che solo le nuove ECU possono utilizzare, in quanto violerebbe il message rate massimo di 20 Hz, imposto dallo standard ISO 11783. Inoltre, grazie alla dimensione maggiore del pacchetto, è possibile introdurre altri campi richiesti dalla normativa ISO 15998 sulla safety dei protocolli di comunicazione per macchine agricole.

Il tipo di messaggio che si vuole aggiungere è un messaggio di controllo distribuito tra Auxiliary Input e Auxiliary Function. Ogni ECU può avere uno o più Auxiliary Input e per ciascuno di essi deve inviare un

Auxilary Input Status Message che ha un message rate di 1 Hz, fino a raggiungere un massimo di 5 Hz nel caso in cui il valore dell'Auxiliary input cambi spesso. Per alcune applicazioni questo vincolo non permette un controllo distribuito efficace, dovuto al basso numero di messaggi di stato, oltre che rappresentare un problema di sicurezza nel caso in cui si perda anche un solo messaggio: la perdita di un messaggio viene riconosciuta dall'auxiliary function tre secondi dopo, cioè il timeout definito dalla norma ISO 11783.

Tabella 3.9 - Tabella delle contro-misure per errore di trasmissione definita dallo standard ISO 15998

Transmission error	Measures per message						
	Running number	Time stamp	Time expiration	Reception Acknowledgement	Identification for sender and receiver	Data integrity assurance	Redundancy with cross check
Repetition	x	x					x
Loss	x			x			x
Insertion	x			x[a]	x[b]		x
Incorrect sequence	x	x					x
Message falsification				x		x	x[d]
Retardation		x	x[c]				
Coupling of SR and non-SR information				x[a]	x		

a depends on application
b only for sender identification. Detects only insertion of an invalid source
c Required in all cases and can implemented in the sink of messages in case of a static map of messages to be received
d This measure is only comparable with a high quality data assurance mechanism if by calculation it can be proved that the residual error r
in Paragraph "Data Integrity Assurance" if two messages are sent through independent networks or network addresses and hardware.

In questo particolare caso, avere la possibilità di inviare messaggi ad una velocità di ripetizione molto più elevata porta indubbiamente ad un controllo più fine da parte della Auxiliary Function, oltre che il riconoscimento tempestivo di errori dovuti al canale di comunicazione o al malfunzionamento dell'Auxiliary Input, riducendo quello che è chiamato *response time* o FRT (*Function Reaction Time*), nella normativa di functional safety specifica per le macchine agricole [22]. Infine la maggiore capacità di dati trasportabile

con un singolo pacchetto consente di migliorare l'efficienza globale del protocollo, permettendo l'invio dello stato di tutti gli Auxiliary input di una ECU in un unico pacchetto.

Viste le implicazioni di safety che le Auxiliary Function possono avere, i nuovi messaggi scambiati dovrebbero avere un livello di robustezza maggiore rispetto a quelli utilizzati all'interno dello standard ISO 11783. Per questo motivo, per la realizzazione di tale messaggio unicast Auxiliary Input – Auxiliary Function si sono introdotte le contro-misure consigliate dalla norma ISO 15998 (riassunte in Tabella 3.9) e riprese successivamente nella norma ISO 25119 part 3 Annex B, Table B.2. Per questa tipologia di messaggio, oltre ad inviare i tipici campi richiesti da ISO 11783, vengono aggiunti il Running Number, Timestamp e un Checksum a 32 bit dei dati, mentre il timestamp con la granularità di un µs è già compreso nel protocollo ISOBUS Over IP.

Questo messaggio viene inviato da Auxiliary input alla Auxiliary Function associata, con un rate di 1 millisecondo, in modo da ottenere un controllo a 1 KHz e dare all'Auxiliary Function la possibilità di reagire più tempestivamente ad errori.

3.12.9 Realizzazione

È stato realizzato un prototipo di sistema in grado di poter gestire una rete ibrida CAN-Ethernet in cui i protocolli ISO 11783 e TCP/IP coesistono. In tale prototipo non sono state realizzate tutte le possibili funzionalità di ISO 11783, bensì solo quelle necessarie per dimostrarne il funzionamento. Le ECU utilizzate per il proof of concept sono le seguenti:

- Gateway Isobus-IP
- ECU ISO11783 con Auxiliary Input su Ethernet
- ECU ISO11783 con Auxiliary Function su Ethernet
- Virtual Terminal su CAN
- ECU ISO 11783 su CAN

3.12.9.1 Architettura del Gateway Isobus-IP

Il Gateway Isobus-IP è stato realizzato sulla stessa piattaforma ARM Cortex A8 descritta in 3.11.2 su cui è stata implementata una diversa architettura software.

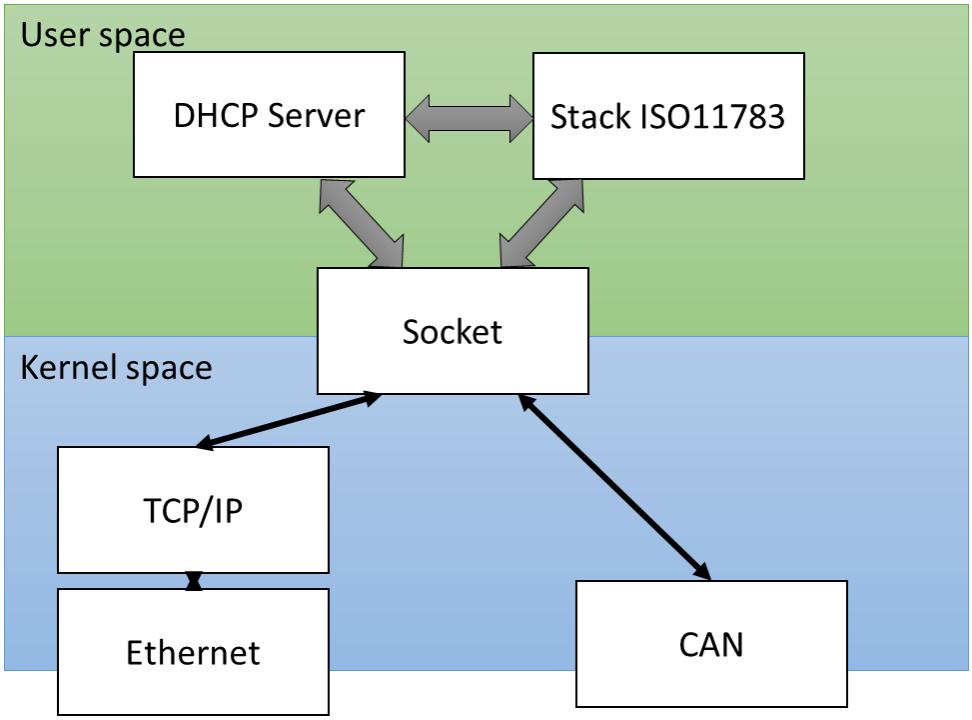


Figura 3.28 - Schema di base del Gateway ISOBUS IP

Il gateway (Figura 3.28**Errore. L'origine riferimento non è stata trovata.**) è realizzato attraverso l'uso di due rocessi separati collegati attraverso una doppia socket in localhost:

- DHCP Server, il cui compito è quello di assegnare gli indirizzi dalla parte Ethernet
- Stack ISO11783, che svolge i diversi compiti di:
    - Gestione delle tabelle di routing fra indirizzi IP e indirizzi ISO 11783
    - Ripacchettamento e spacchettamento dei messaggi per i protocolli di trasporto
    - Marshalling e unmarshalling dei messaggi
    - Gestione degli alias IP dalla parte Ethernet
    - Eseguire le operazioni di address claiming

Il primo processo, il DHCP server, gira a priorità più bassa e deve gestire le Vendor Options ISO 11783, oltre che comunicare col processo ISO 11783 che si occupa di eseguire la procedura di address claiming. Per modificare il comportamento del DHCP Server si è scelto di modificarne uno già pronto. La modifica è stata fatta partendo dai sorgenti del DHCP Server *udhcp* fornito da *BusyBox*, una toolsuite di servizi e applicazioni shell in formato ridotto per embedded.

Il secondo processo, Stack ISO 11783, che gira a priorità più elevata, simula le ECU presenti su CAN su IP e viceversa. Esso mantiene due tabelle ove sono contenuti gli indirizzi di livello logico su ISO 11783 e gli

indirizzi IP su Ethernet, per poter applicare le politiche di routing necessarie per la comunicazione, ad esempio ignorando i messaggi uni-cast dove sorgente e destinazione si trovano su rete CAN (mentre su IP tale comportamento è garantito intrinsecamente dalla rete). Lo Stack, inoltre, per mantenere gli IP "simulati" gestisce gli alias IP sulla rete attraverso socket Netlink [21], che permettono di manipolare le impostazioni di rete di un host Linux da programmi user space, con particolari permessi di esecuzione. Attraverso la comunicazione in localhost, notifica al server DHCP eventuali indirizzi già acquisiti, e si occupa di creare le richieste di address claim, il cui risultato viene comunicato al DHCP server.

3.12.9.2 Architettura delle ECU su Ethernet

Per realizzare delle ECU in grado di supportare Ethernet nativamente, cioè utilizzando uno stack software per la gestione dello stack TCP/IP, sono state utilizzate delle deveopment board con micro ARM Cortex M4.

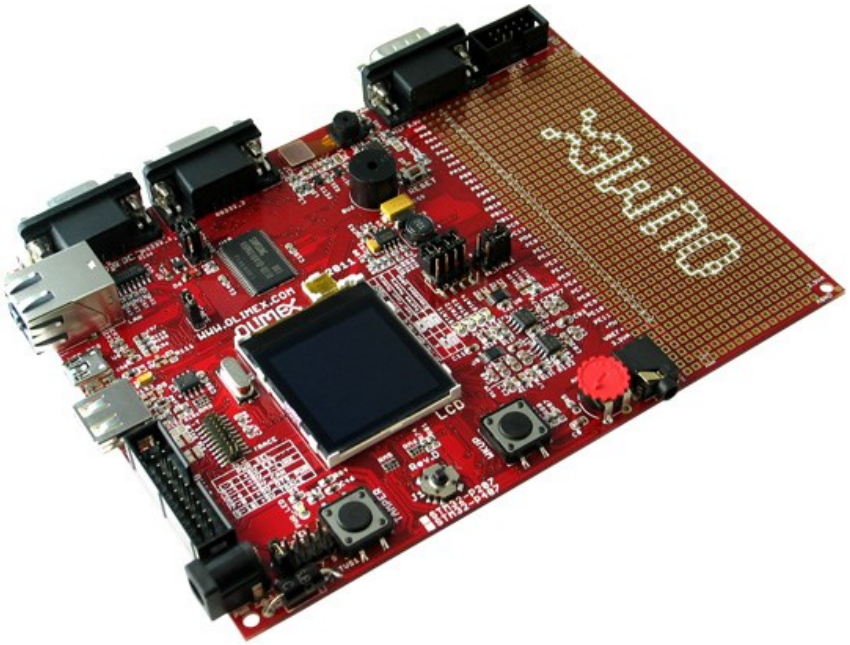


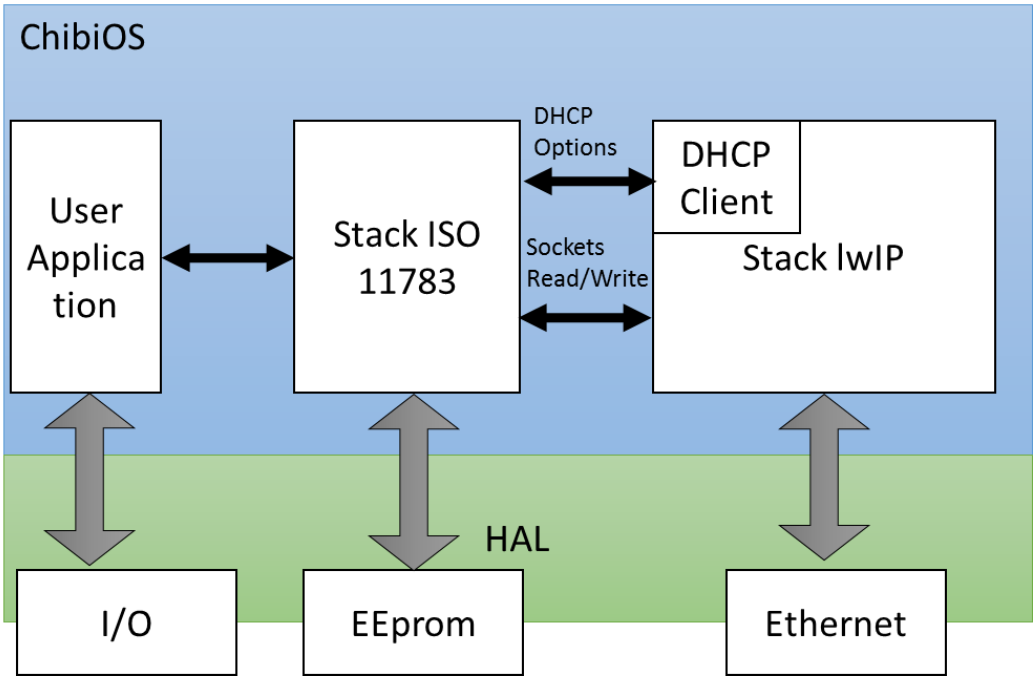
Figura 3.29 - Foto della scheda Development Board Olimex STM32-407

In particolare è stata utilizzata una development board di Olimex STM32-P407 basata su ARM Cortex M4 della ST, completa di PHY Ethernet.

Le caratterisitche peculiari della scheda sono:

- Cortex M4@168 MHz
- 1 MByte Flash

- 196 Kbyte RAM
- 1 Ethernet
- 2 CAN
- Joystick



Sul microcontrollore è stato montato il sistema operativo real-time ChibiOs, con lo stack TCP/IP *lwIP*. L'utilizzo di sistema operativo è quasi d'obbligo, vista la complessità dello Stack e per poter creare la separazione completa tra i task di Stack da quelli di controllo. Lo Stack TCP/IP è stato modificato per utilizzare le *vendor options* per il DHCP Client per la comunicazione ISOBUS-IP. Inoltre è stato integrato lo Stack ISO 11783, con le opportune modifiche per quanto riguarda la parte di comunicazione. Il sostrato di ISO 11783 è stato mantenuto per permettere il facile porting di applicazioni legacy, che si basano su chiamate a libreria dello Stack, lasciandole all'oscuro del cambiamento dei livelli 1-4 della pila ISO-OSI.

Lo Stack ISO 11783 rimane in ascolto sulla porta 10.000, sia UDP che TCP, utilizzate all'interno del protocollo ISOBUS over IP, per inviare i messaggi real-time e il traffico di configurazione non real-time. I messaggi vengono interpretati e generano eventi gestiti dal thread dell'applicazione finale.

3.12.10 TestBench e risultati

È stato realizzato un banco di test di una rete prototipale mista CAN-Ethernet, per poter verificare l'effettivo funzionamento del porting del protocollo ISO 11783 su Ethernet con Stack TCP/IP.

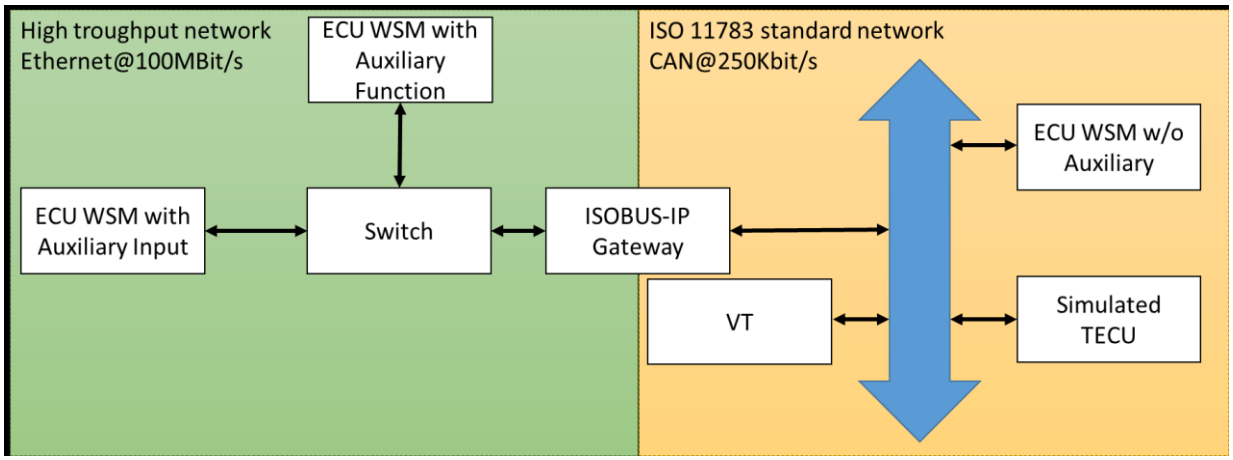


Figura 3.30 - Schema del sistema prototipale a doppia rete ISO 11783

Il banco di test prototipale (il cui schema generale è in Figura 3.30) è formato da ECU commerciali e certificate, oltre che da alcune costruite ad-hoc, per poter dimostrare che le modifiche introdotte garantiscono comunque l'interoperabilità del nuovo con il vecchio.

Il sistema di test è formato dalle seguenti ECU:

1. ECU Working Set Master con Auxiliary Function su Ethernet basata su architettura Cortex M4 e Stack ISOBUS over IP, basato su applicazione commerciale
2. ECU Working Set Master con Auxiliary Input su Ethernet basata su architettura Cortex M4 e Stack ISOBUS over IP, basato su applicazione commerciale
3. ISOBUS over IP Gateway, basato su architettura Cortex A8, affacciato sia su CAN che su Ethernet
4. Virtual Terminal commerciale già certificato
5. ECU Working Set Master commerciale già certificata
6. Simulatore di TECU, creato mediante l'applicazione PC CANoE

Attraverso questo banco di test si intende verificare tutte le modifiche effettuate nel porting e descritti dai punti precedenti. Il solo funzionamento del sistema è valido come test globale e come proof of concept, in quanto durante il setup del sistema vengono automaticamente utilizzati tutti i meccanismi descritti in precedenza, in particolare:

- L'address claiming, senza di esso i 3 Working Set Master non si potrebbero identificare con il Virtual Terminal

- Il routing, senza di esso le ECU dal lato CAN non vedrebbero le ECU lato Ethernet e viceversa, non permettendo quindi ai 2 Working Set Master su Ethernet di vedere il Virtual Terminal su CAN
- L'indirizzamento, senza il quale i messaggi non arriverebbero a destinazione
- I protocolli di trasporto, senza di essi i Working Set Master non riuscirebbero ad inviare correttamente la propria configurazione al Virtual Terminal
- I nuovi messaggi ad alto troughput, senza i quali uno dei due Working Set invierebbe un warning al Virtual Terminal

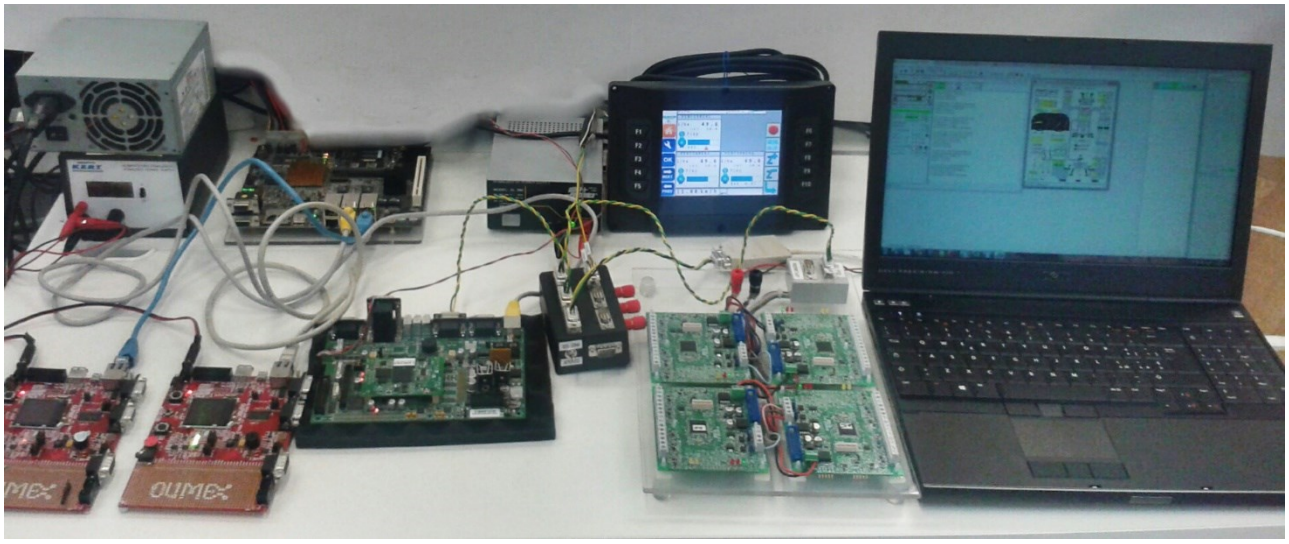


Figura 3.31 - Foto del sistema di test

Il proof of concept è stato validato dal caricamento corretto degli object pool sul Virtual Terminal e dalla possibilità di associare Auxiliary Function ad Auxiliary input. È stato possibile visualizzare l'object pool di tutti e tre i WSM, pur risiedendo su reti diverse. Le ECU su Ethernet visualizzano correttamente i dati provenienti dalla TECU simulata, mentre non sono stati rilevati pacchetti persi tra Auxiliary Input e Auxiliary Function, cosa che non sorprende visti i risultati del Task 0. Inoltre grazie ai messaggi ad alto throughput, una eventuale disconnessione di una delle due ECU può essere notata in maniera tempestiva.

Tale risultato prova la corretta esecuzione dei task nelle tempistiche richieste dal Virtual Terminal su rete ISOBUS, che prova che sia lo switch sia il gateway non introducono ritardi significativi, mentre consentono la creazione di una rete ad alto throughput compatibile con la parte di rete legacy.

3.12.11 Conclusioni

Questo task ha dimostrato la possibilità di utilizzare una rete ad alto throughput come rete del futuro per le comunicazioni fra ECU all'interno del veicolo, mantenendo comunque la retrocompatibilità con la rete pre-esistente.

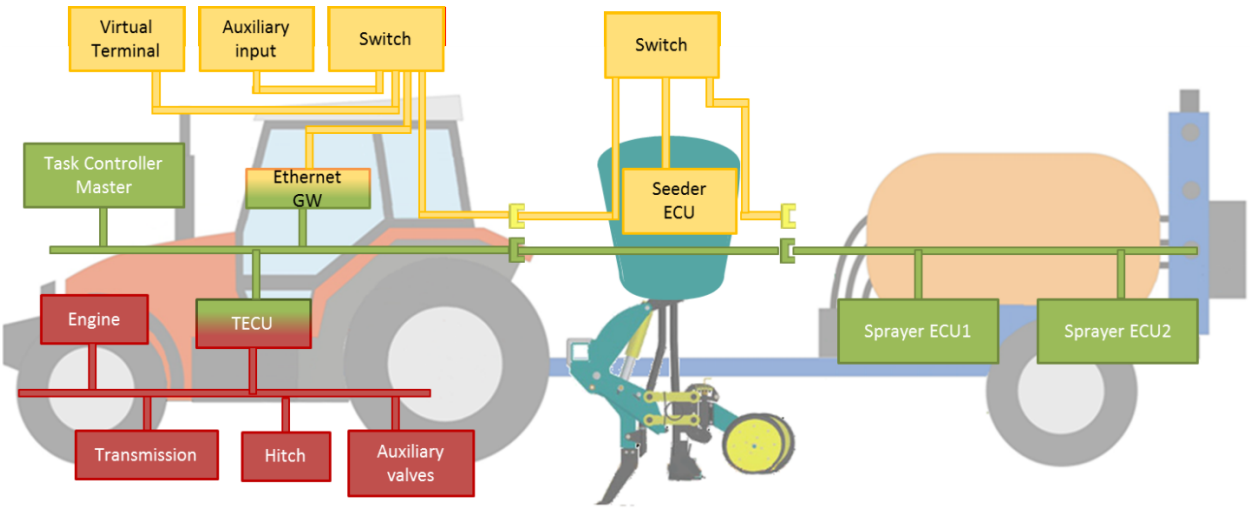


Figura 3.32 - Esempio di rete del futuro, con due backbone, uno CAN, uno Ethernet

La retro compatibilità è fondamentale nell'ambiente delle macchine agricole, visto il lungo tempo di vita degli attrezzi agricoli e delle trattrici. Con la validazione del proof of concept si è aperta la strada a nuove topologie di reti ibride, ad esempio con due backbone (come in Figura 3.32), una Ethernet ed una CAN, per poter gestire sia ECU ed attrezzi del futuro, con capacità di automazione e sicurezza molto elevate, sia ECU del presente, con ridotte capacità di automazione e di sicurezza, dettate dai limiti di throughput e dai vincoli di occupazione di banda del layer fisico. L'alto throughput ottenuto apre la strada a nuove applicazioni di sicurezza, che possono basarsi su meccanismi di ridondanza software e maggiore velocità di reazione a guasti.

Inoltre lo standard utilizzato permette nuove integrazioni con il mondo esterno, come ad esempio l'integrazione con una rete wireless ad-hoc per lavorazioni cooperative di più mezzi che rispettano lo standard ISO 11783, oppure l'integrazione della macchina all'interno della rete Internet, con servizi di remotazione di comandi, informazioni e diagnostica.

Questo lavoro è stato presentato ai membri della commissione WG1 del comitato ISO, che visti i risultati e l'interesse dimostrato da diverse nazioni ai risultati di questa ricerca, ha deciso di creare una task force di studio per la creazione di una nuova normativa per reti ad alta velocità, guidata da IMAMOTER.

4 SAFETY NELLE MACCHINE AGRICOLE E MOVIMENTO TERRA A LIVELLO DI ECU

4.1 Introduzione

Il mondo delle macchine movimento terra ha subito negli anni ’90 la prima rivoluzione, cioè l’introduzione dell’elettronica all’interno delle macchine per sostituire i controlli prima prettamente idraulici o meccanici di tutto il sistema. Negli ultimi anni, in particolare con l’entrata in vigore della norme ISO 13849 e ISO 25119, e la nuova direttiva macchine (Direttiva 2006/42/EC, di fatto entrata in vigore alla fine del 2011 complice la crisi economica che ne ha ritardato la introduzione), tutti i costruttori di macchine movimento terra hanno dovuto far fronte ai nuovi requisiti di sicurezza funzionale all’interno di sistemi elettronici ed elettromeccanici. Infatti attraverso queste norme sono stati introdotti dei requisiti di sicurezza funzionale più stringenti e definiti rispetto al passato e alla precedente norma armonizzata EN 954, mutuandoli da mondi già regolamentati quali l’avionica e, in tempi più recenti, il mondo automotive. Dall’introduzione delle norme ne è uscita la necessità di rivalutare completamente l’architettura interna e di comunicazione delle ECU che pilotano organi elettromeccanici, di fatto classificando come insicuro qualunque singolo componente elettronico, sia discreto sia a logica programmabile. I requisiti di sicurezza non sono applicati ad un singolo componente, bensì a tutto il canale responsabile della esecuzione di una funzione, e ciascun canale, in base alle funzionalità che esegue, può avere requisiti di safety diversi; questa diversa classificazione in base al rischio associato a ogni operazione rende di fatto difficile creare un approccio general purpose che possa essere riutilizzato per più macchine. Visto il tipo di mercato, caratterizzato da una enormità di diverse funzioni per numeri piuttosto piccoli (esattamente il contrario dell’automotive) la ricerca di un architettura generale in grado di soddisfare i requisiti di sicurezza e al contempo più generale possibile è diventata di fondamentale importanza. Dato il contesto in cui il CNR opera, ovvero come guida delle imprese italiane, e dato che tali imprese hanno tra i punti di forza la capacità di differenziare la produzione in base alle richieste dei clienti creando di fatto, molti tipi di macchine diverse, la mia ricerca è volta alla progettazione di un’architettura hardware e software in grado di poter coprire la maggior parte di applicazioni possibili rispettandone i criteri di sicurezza. La sintesi di un progetto facilmente adattabile e consistente in un nucleo non modificabile da ogni singola applicazione e responsabile della attuazione delle funzioni di sicurezza, rende più agevole alle aziende che lo adottino l’accesso alla certificazione.

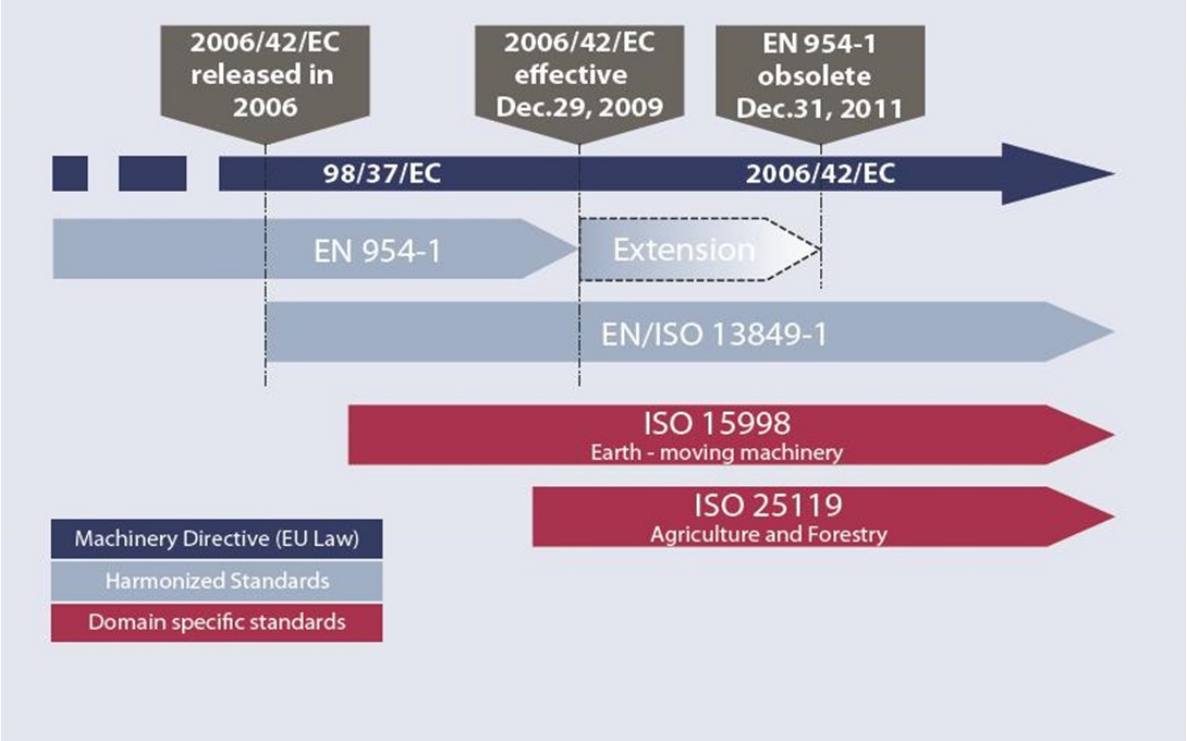
4.2 Safety e safety systems

La *Safety* è libertà dall’inaccettabile rischio di lesioni fisiche o di danni alla salute delle persone che si verifichino, direttamente o indirettamente, a seguito di danni alle cose o all’ambiente.

Sono definiti *Safety Systems*, l'insieme dei sistemi di protezione contro danni causati da agenti esterni non controllabili dall'utente o dalla disattenzione dello stesso. Questi sistemi devono essere in grado di poter garantire la safety, che può coincidere col portare l'intero sistema in uno stato *safe*, in cui non possono essere causati danni a persone o cose, oppure se non esistente, devono essere in grado di fornire le stesse funzionalità o funzionalità ridotte (dette di recovery) in condizioni di sicurezza. In base alla tipologia di recovery che viene attuata a fronte di un guasto, un sistema è detto fail silent, failure operational o fault tolerant.

4.3 Normative di riferimento

Prima di parlare di criteri di safety e progettazione di un sistema safety critical per veicoli off-road è necessaria un'introduzione alla notevole quantità di normative che disciplinano la functional safety nei componenti elettronici per le macchine. Esistono varie tipologie di norme, e la classificazione di tali norme va dalle Direttive (comunitarie), che indicano i requisiti che devono soddisfare le macro categorie di macchine, alle norme generali (dette di tipo A), che definiscono principi non sempre facilmente applicabili in un sistema reale, sia per mancanza di metodologie, sia per la specificità di un sistema i cui requisiti di sicurezza non possono essere coperti da una norma generale.



Esistono quindi le normative specifiche, sia per classi di veicoli (dette di tipo B), sia per specifiche tipologie di veicoli (norme di tipo C).

Dal 2011, la nuova Direttiva Macchine (2006/42/EC) ha reso obsoleto l'uso della norma EN 954-1, che disciplinava i requisiti di sicurezza delle macchine a favore di norme più restrittive e specifiche, quali la norma ISO 13849 per le "macchine" in generale (quindi norma di tipo B) e normative di tipo specifico per categorie di macchinari, quali la ISO 15998 per le macchine movimento terra e la ISO 25119 per le macchine agricole, norme di tipo C.

Un costruttore di veicoli, all'atto della progettazione di un nuovo sistema che può avere caratteristiche rilevanti dal punto di vista della sicurezza, deve quindi soddisfare dei requisiti di sicurezza (EHSR), per poter essere conforme alla Direttiva Macchine.

Esistono quindi tre tipologie di norme:

- Tipo A: Norme generali, applicabili a tutti i tipi di macchine
- Tipo B: Divise in due tipi:
    - Tipo B1: copre alcuni aspetti particolari di safety ed ergonomia delle macchine
    - Tipo B2: sono relative a componenti di sicurezza e di protezione
- Tipo C: coprono particolari classi di macchine; sono le normative specifiche .

Vista la difficoltà di poter certificare un sistema completo, quasi tutte le normative di riferimento definiscono delle guideline per le procedure che devono essere seguite per la progettazione, lo sviluppo e la manutenzione di un sistema E/E/PES (Electrical/Electronic/Programmale Electronic System).

Ove esista una normativa di tipo C armonizzata alla Direttiva (in questo caso alla Direttiva Macchine o alla Direttiva Trattrici agricole), tale normativa, se seguita dal costruttore, dà la presunzione di conformità alla Direttiva e quindi alla omologazione del macchinario. In mancanza di una normativa di tipo C specifica per una determinata tipologia di macchine, viene analizzata la normativa di tipo B, più generale, che si applica – in questo caso specifico a tutte le macchine da lavoro.

Tutte le normative riguardanti la sicurezza funzionale applicata alle macchine, fanno riferimento a modelli di valutazione:

1. della funzionalità della parte hardware del sistema elettronico che è coinvolto nella funzione di sicurezza, della struttura delle unità elettroniche coinvolte e delle loro interconnessioni,
2. della parte software e in particolare della sua qualità secondo modelli quali, ad esempio, il CMM (Capability Maturity Model),
3. della copertura diagnostica,
4. della robustezza del progetto dal punto di vista sistemistico,

5. della qualità dei componenti.

Tutti gli aspetti di progetto dei sistemi elettronici concorrono a formare il grado complessivo di performance del sistema dal punto di vista della sicurezza funzionale, ma la analisi necessariamente parte dagli aspetti hardware e sistemistici e scompone i sistemi elettronici coinvolti in serie di funzioni.

Ogni funzione è analizzata nel suo flusso informativo e di esecuzione ed è associata a una struttura ingresso – logica – uscita, che deve essere ricondotta a modelli classificati, che permettono di ricondurre il sistema a una griglia di valutazione delle performance che fornisce un dato oggettivo singolo, detto Performance Level (PL), Agricultural Performance Level (AgPL) o Safety Integrity Level (SIL), a seconda della normativa seguita per la conduzione della analisi.

Oltre alla necessaria precisazione sulla equivalenza tra i PL e i gradi SIL, sancita da un parametro nelle normative, grazie alla quale da una classificazione si può ricavare agevolmente l'omologa nelle altre norme, va sottolineato che la valutazione dei progetti non è generale, bensì è profondamente calata nella funzione della applicazione specifica. Ovvero il giudizio sulla adeguatezza di un sistema non è assoluto ma è vincolato alla specifica applicazione analizzata.

Se questo approccio, mutuato anche dalla norma ISO 26262 che standardizza la functional safety in ambito automotive, non crea problemi ai costruttori di auto (passenger car), è invece molto oneroso per tutti i costruttori di veicoli off road da lavoro, siano essi agricoli o da cantiere, in quanto i costi delle analisi e delle certificazioni sono da assorbire su piccole serie e a volte quasi su pezzi unici.

La opportunità di generare un archetipo progettuale con caratteristiche che lo rendano facilmente adattabile a diverse applicazioni, senza la necessità di ripercorrere l'intero processo di analisi per la classificazione in termini di functional safety, è stata giudicata di fondamentale importanza dalla Unione Costruttori macchine Agricole e Movimento Terra (UNACOMA) di Confindustria, di cui l'Istituto IMAMOTER del CNR è partner per lo sviluppo e la ricerca applicata. Tale esigenza è stata recepita e ha attivato presso IMAMOTER uno studio per la definizione di una architettura adattabile e flessibile, che avesse le caratteristiche di garantire un elevato Performance Level dal punto di vista della sicurezza funzionale, utilizzando il più possibile elementi non direttamente dipendenti dalla particolare applicazione, ma solo dalle modalità di utilizzo delle risorse hardware.

Il progetto ha una sua caratterizzazione sistemistica e ne viene presentata la genesi dal punto di vista di approccio funzionale e normativo. Ne vengono esposti i punti di forza in relazione alla sua adattabilità e, infine ne vengono descritte alcune applicazioni e adattamenti di successo, che sono oggi nella fase di industrializzazione.

4.4 Un caso di studio

L'applicazione delle normative sopra descritte è molto difficile se non si specifica ove il sistema deve operare. L'approccio imposto dalle normative per lo sviluppo di un sistema safety-relevant è vincolato alla realizzazione hardware del sistema. L'approccio scelto per la ricerca è stato di tipo architetturale, cioè si è cercato di rimanere il più staccati possibile dai calcoli di MTTF, Diagnostic Coverage (DC) e CCF (Common Cause Failure), per ragionare ad un livello più elevato. I calcoli potranno essere fatti di volta in volta che l'architettura generale sarà applicata ad un sistema reale. Ovviamente la bontà del modello e dell'architettura proposta può essere validato solo nell'utilizzo di un caso reale, e sarà maggiore se si potranno riutilizzare gli stessi schemi architetturali hardware e software con minime modifiche, che riguardano le configurazioni dei canali e della diagnostica, su diverse applicazioni, anche diverse tra loro. L'approccio scelto quindi si scontra con quello imposto dalle normative, che richiedono che il progetto sia validato esclusivamente su una applicazione specifica, ma la ricerca presentata mira a dimostrare che tale approccio è valido e che si è in grado di definire un archetipo architetturale, che rende conformi tutte le applicazioni relative a definiti ambiti applicativi, nella ipotesi di rispettare delle regole nella definizione delle risorse utilizzate per la realizzazione dei canali funzionali.

4.5 Ipotesi

Lo sviluppo di un modello architetturale, per quanto l'approccio sia generale, deve comunque partire da delle ipotesi e da alcuni vincoli per poter essere sviluppato, ed eventualmente generalizzato, ove possibile.

L'ambiente su cui la ricerca è iniziata è quello delle macchine off-road, in particolare macchine di movimentazione carichi sospesi o strutture mobili con uomini a bordo. In questa tipologia di applicazione si hanno dei requisiti di sicurezza per quanto concerne il pilotaggio delle valvole proporzionali o ON-OFF che mettono in movimento i braccii meccanico del veicolo, poiché potrebbero causare danni a persone o cose per un comando inatteso od errato, oppure provocare il ribaltamento del mezzo.

Un aspetto fondamentale è lo stato di sicurezza, o safe state, in cui l'intero sistema in condizioni di guasto, minimizza il rischio di provocare danni, e che deve essere poter raggiunto anche in presenza di accumulo più guasti. Tale stato non sempre esiste o presenta una minimizzazione accettabile: si pensi ad esempio ad un sistema *Steer by Wire*, affetto da un guasto che non permetta di leggere correttamente il sensore sul volante, non esiste un modo per poter definire uno stato sicuro, a meno di non prevedere un sistema completamente ridondato e in logica di major voting simile alle architetture utilizzate nel mondo avionico. Tali sistemi devono quindi essere molto robusti e di tipo fault-tolerant, cioè essere in grado di eseguire comunque la loro funzione, magari con prestazioni ridotte o limitate, anche in caso di guasto multiplo.

Nella movimentazione dei carichi fortunatamente i sistemi non devono essere fault-tolerant, bensì si può generalmente definire un safe state, e per questo sono definiti fail-safe. Lo stato sicuro corrisponde alla stasi dei carichi, e la fault recovery potrebbe essere attivata per errori accumulati su tutta la catena di informazioni oppure per bloccare un movimento che potrebbe fare uscire il baricentro del mezzo dalla base, provocandone il ribaltamento. Si possono prevedere delle operazioni di recovery, di solito attuate agendo manualmente o meccanicamente sui circuiti idraulici per poter, ad esempio, abbassare un braccio o far scendere gli uomini a bordo della piattaforma di sollevamento.

Esistono però condizioni e sistemi che non sono di così semplice classificazione e sempre più spesso nascono richieste per la ideazione di sistemi di tipo *fail operational*, ovvero in grado di offrire recovery funzionali alternative in condizioni di guasto, proprio a causa della impossibilità di considerare la stasi come uno stato di sicurezza.

L'ultimo parametro per l'inizio della ricerca è l'identificazione dell'obiettivo in termini di Performance Level che il sistema deve poter fornire, da cui dipende tutta la struttura del sistema.

Tabella 4.1 - Tabella (da ISO 13849) dei Perfomance Level in base a DC, MTTF, Hardware Category

Category	**B**	**1**	**2**	**2**	**3**	**3**	**4**
DC_{avg}	**none**	**none**	**low**	**medium**	**low**	**medium**	**high**
$MTTF_d$ of each channel							
Low	a	Not covered	a	b	b	c	Not covered
Medium	b	Not covered	b	c	c	d	Not covered
High	Not covered	c	c	d	d	d	e

Il performance level richiesto dalle nuove normative per dei canali che comandano le attuazioni nelle macchine off road è tipicamente un PL C, raggiungibile attraverso diverse strategie (si veda la Tabella 4.1), sia hardware, che software che architetturali. Ad esempio si può ottenere un Performance Level C, utilizzando un Hardware Category 1 a 1 microcontrollore, ma ciò richiede l'utilizzo di componenti di canale costosi e difficilmente reperibili, oltre che a una certificazione del software. La certificazione del software è un'operazione onerosa, sia dal punto di vista della documentazione richiesta, sia per quanto riguarda lo sviluppo.

Inoltre, a corollario, si precisa che normalmente non è possibile considerare i componenti elettronici programmabili come componenti well tried, per cui è consigliabile sempre ideare architetture ridondanti in

ogni sistema in cui sia presente un microcontrollore, un DSP o una Logica programmabile, a meno di non utilizzare sistemi particolari che offrono già canali ridondanti on chip.

Per poter agilmente ottenere un Performance Level C si possono utilizzare architetture hardware più complesse come un Hardware Category 2 con Diagnostic Coverage medio, senza dover ricorrere a certificazioni software che impongono analisi quantitative e definizione di specifiche con metodi formali e semi-formali, che imporrebbero alle aziende modifiche onerose nei processi di sviluppo. Viceversa in caso di architetture ridondanti il PL C richiede analisi di qualità del software abbastanza limitate e normalmente di tipo qualitativo.

4.6 Hardware Category

La scelta della categoria hardware è fondamentale, poiché da essa si parte per costruire l'architettura del sistema. Sono definite cinque categorie hardware, dalla più semplice (la B, per base) a quella più complessa e sicura (la 4). La categoria hardware viene modellata per canale, pertanto una stessa ECU può avere diverse Categorie Hardware in base a come è realizzato il canale specifico che realizza una particolare funzione.

La categorie B e 1, hanno una architettura dove c'è una parte di logica a canale singolo, a cui è collegata una parte di Input e di Output, senza nessun altro elemento esterno. La differenza tra un categoria B e 1 è la copertura diagnostica (assente in B) e l'utilizzo di well tried components per le parti hardware (non richiesto nella categoria B).

Per quanto detto precedentemente, in presenza di un microcontrollore, è normalmente impossibile sancire una categoria 1, mentre si deve dichiarare una Categoria B, che rende impossibile utilizzare tale sistema per applcazioni safety critical nel mondo dei veicoli off road.

Le categorie hardware più interessanti dal punto di vista della ricerca, che permettono di avere Performance Level elevati sono le categorie 2, 3 e 4, mostrate in Figura 4.1 e Figura 4.2. Un hardware di categoria 2 ha una parte di Logica Programmabile (il blocco L) che esegue le trasformazioni del segnale S_i già condizionato da componenti discreti (blocco I), facendolo diventare un segnale di uscita S_o che piloterà un'uscita attraverso un blocco di circuiteria di amplificazione che si occuperà di pilotare l'attuatore (blocco O). Allo schema classico è aggiunto un componente di Supervisione detto Test Equipment (blocco TE) che monitora ingresso, logica ed uscita, e invia un segnale di consenso ad un blocco che dà il consenso (blocco OTE, Output Test Equipment).

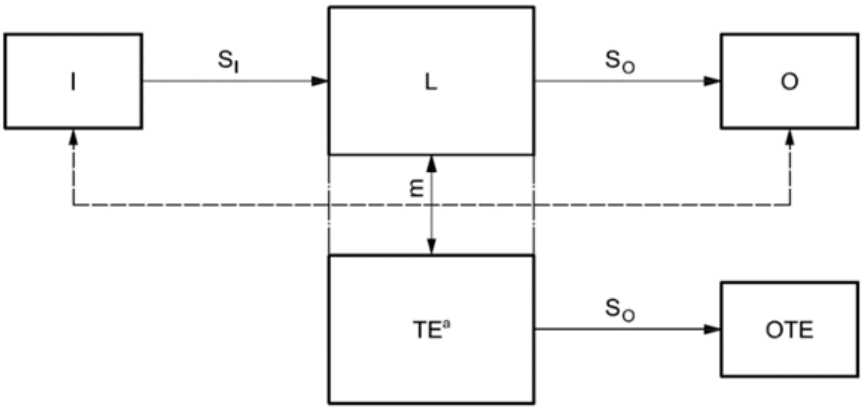


Figura 4.1 - Schema di un Hardware Category 2

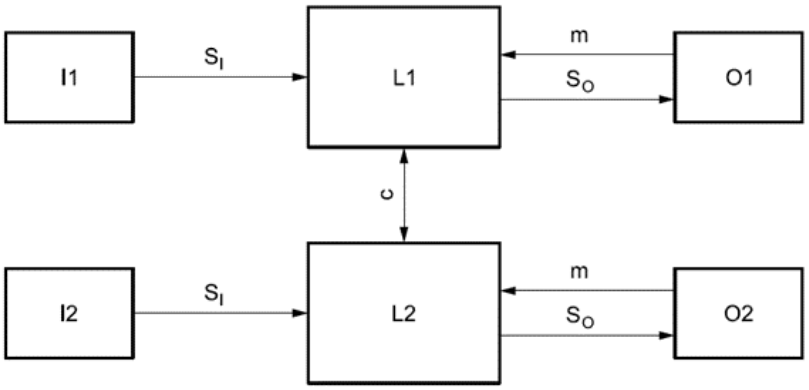


Figura 4.2 - Schema di un Hardware Category 3-4

In Figura 4.2 è invece schematizzato un hardware Categoria 3-4, tipicamente utilizzato per sistemi fault tolerant. Questo tipo di schema equipaggia due logiche differenti con uscite con feedback. La differenza tra un categoria 3 ed un categoria 4 è la copertura della diagnosi e l'MTTF dei componenti utilizzati sul canale

La ricerca si è concentrata sulla realizzazione di una architettura che sia un mix delle categorie hardware descritte e che dia i benefici del controllo di un supervisore (tipica di una architettura HC 2), dando comunque una ridondanza ed una *diversity* agli ingressi e ai consensi (tipica di una architettura HC 4).

4.7 Architettura hardware

Nei prossimi capitoli verranno illustrate tutte le caratteristiche pregnanti dell'architettura hardware oggetto della ricerca, e di tutti gli accorgimenti utilizzati per poter soddisfare sia i requisiti di un category 2,

sia di un category 4. Si noti che l'architettura verrà descritta in maniera generale, tenendo conto solo dei canali che richiedono un Performance Level C o superiore. La realizzazione di tale architettura potrebbe variare in base alla disponibilità di risorse del sistema, rispettando però i criteri generali

4.7.1 Doppio microcontrollore

Nell'architettura vengono definiti due tipi di microcontrollore, collegati attraverso una periferica seriale sincrona, al fine di verificare molti aspetti dello stato interno di ciascun processore da parte dell'altro, attraverso un canale di comunicazione che impone un throughput che è adeguato al FRT delle funzionalità da realizzare nella applicazione specifica (tipicamente attorno ai 100 – 200 ms), e così definiti:

1. Main Microcontroller (MMC): è il processore principale, sui cui sono presenti tutte le logiche di canale. Esso contiene al proprio interno tutto il codice per gestire canali safe e non safe e può essere modificato dall'utente finale per implementare le proprie strategie. È il master del controllo e slave della linea di supervisione seriale.
2. Safety Microcontroller (SMC): è il processore di supervisione e di ridondanza. Al suo interno sono presenti alcune logiche di base di massima per la gestione dei canali safety relevant. È master della comunicazione su linea di supervisione seriale e gestisce le diagnosi dei canali, oltre che il campionamento sincronizzato dei canali. Può avere delle performance inferiori rispetto al MMC, in quanto non necessità di particolare velocità di calcolo.

La diversity dei due micro è garantita dai seguenti principi:

- Diversità dei produttori
- Diversità delle architetture hardware (ad esempio si può usare come MMC un ARM Cortex M3 a 32 bit e come SMC un PIC18 a 8 bit o dsPic30F a 16 bit)
- Diversità dei compilatori, garantita dalla diversa architettura hardware, che richiede compilatori diversi (ad esempio GNU GCC Toolchain per ARM e Microchip C18 o C30 per PIC/dsPic)
- Diversi programmatori
- Presenza di sistema operativo, che su MMC può essere presente proprio per la maggior logica di controllo che si vuole implementare, mentre su SMC deve essere assente, visti i semplici compiti.
- Diversità della realizzazione della abilitazione dei carichi (ad esempio con "pompe di carica" o con porte logiche statiche).

Tipicamente, il linguaggio di programmazione per eccellenza su dispositivi Embedded è il C, pertanto sarà difficile garantire una diversity a livello di linguaggio, se non per particolari realizzazioni descritte in futuro.

I microcontrollori sono collegati da diverse linee di I/O e di comunicazione per la gestione delle funzioni di safety, descritte nei capitoli successivi.

4.7.2 Ingressi ridondati

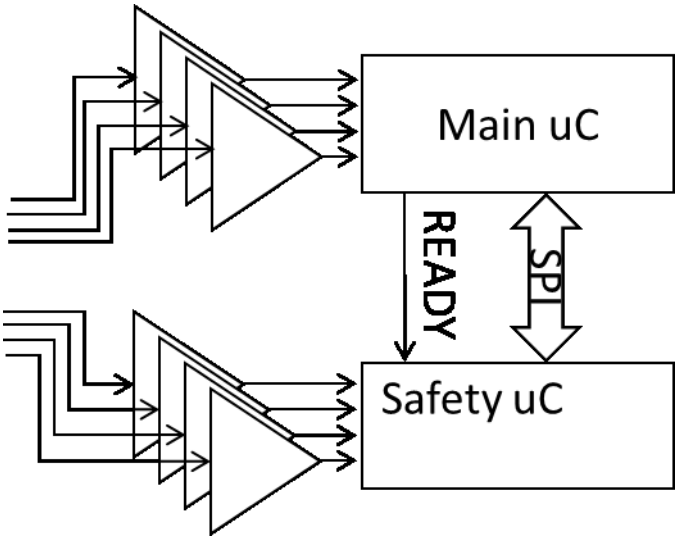


Figura 4.3 - Schema degli ingressi ridondati

Gli ingressi sono ridondati come un Hardware Category 4, cioè ciascun microcontrollore campiona i segnali in ingresso con il proprio ADC. Affinché le misure siano sincronizzate, e quindi validate, attraverso la linea di comunicazione sincrona di Safety tra i due microcontrollori, il SMC richiede il campionamento periodico al MMC di ciascun canale.

Il campionamento di un canale su MMC è triggerato dalla ricezione di un comando su SPI, mentre il campionamento su SMC è triggerato all'invio del comando su SPI, pertanto i campionamenti sono sufficientemente sincronizzati.

4.7.3 Uscite con feedback

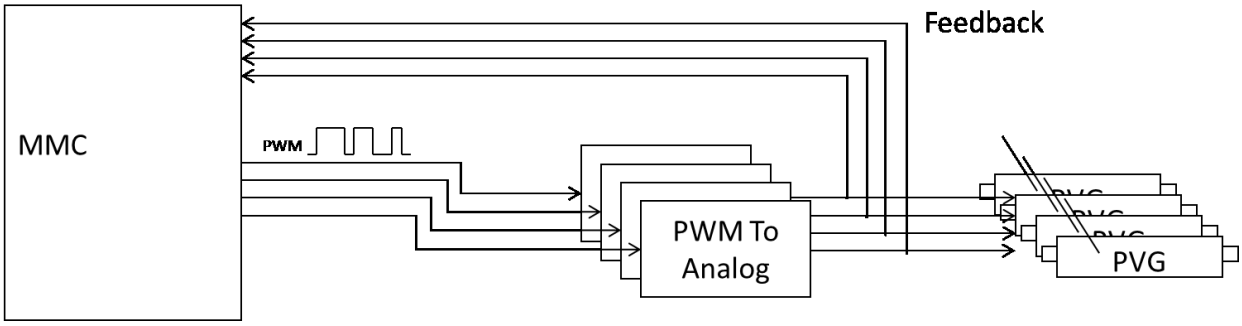


Figura 4.4 - Schema degli stadi di uscita

Tutte le uscite dei canali Safety-Relevant hanno un meccanismo di feedback per verificare che il comando sia effettivamente stato eseguito. Questo permette di diagnosticare eventuali fault sulle linee di uscita, dovuti o allo scollegamento dei carichi o a errori della rete di uscita, offrendo una Diagnostic Coverage Medium o High in funzione alla contemporanea presenza di sensori o segnali di stato sulle linee controllate. Questa tipologia di uscita è tipica degli hardware category 4, ed è l'unica che offra un'alta copertura diagnostica dei guasti.

In Figura 4.4 si vede lo schema con uscite analogiche, ma lo stesso principio è utilizzato per le uscite digitali, utilizzate per pilotare elettrovalvole di tipo ON-OFF.

4.7.4 Energizzazione a doppia conferma

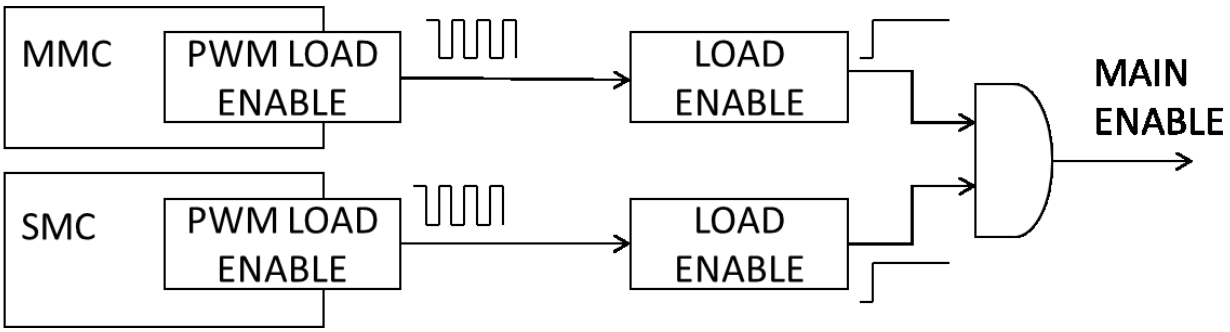


Figura 4.5 - Schema del meccanismo di doppia conferma

Nel caso in cui uno dei microcontrollori non riesca a diagnosticare un eventuale errore, deve essere data la possibilità all'altro di poter intervenire. Per questo motivo, tutte le uscite marcate come safety relevant, che pilotano carichi, vengono messe in AND logico da un segnale di abilitazione carichi. Su molti componenti discreti di sicurezza c'è sempre un ingresso di Enable, a cui viene appunto collegato questo segnale. Il segnale di abilitazione carichi rappresenta lo stato del sistema. Quando tutto va bene è alto e i

carichi si comportano come da logica del MMC; quando invece viene rilevato un guasto il micro abbassa il proprio Enable. In questo modo, avendo due microcontrollori diversi, si aumenta la sicurezza. Infine, un altro accorgimento è che il segnale di abilitazione di ciascuno dei due microcontrollori è una PWM generata via software a ciclo di programma, che va a caricare una pompa di carica (LOAD ENABLE): se uno dei microcontrollori si blocca per qualunque motivo, i carichi vengono quindi disabilitati, portando il sistema in un safe state di stasi degli attuatori.

4.7.5 Reset Gerarchico

Nel caso in cui il MMC esegua delle operazioni non valide, il SMC può eseguire il reset del MMC per ripristinare una situazione corretta. Il problema è che anche SMC, essendo un elemento non sicuro potrebbe eseguire delle operazioni non valide.

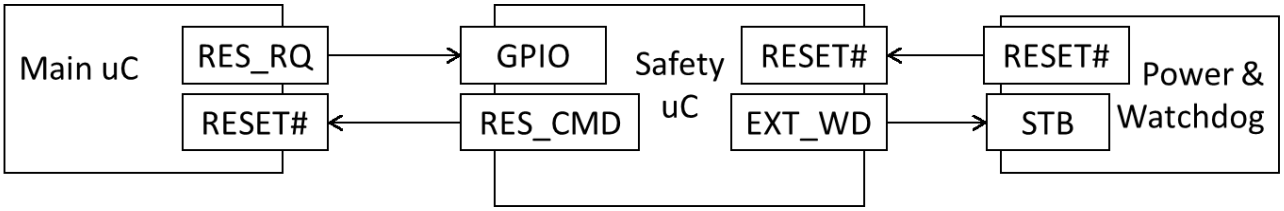


Figura 4.6 Schema dell'architettura di reset

La soluzione utilizzata nell'architettura (in Figura 4.6) realizza un sistema di reset che dà la possibilità ad entrambi i micro di essere resettati, nel caso di malfunzionamenti. Questo meccanismo di reset evita il deadlock del sistema nel caso entrambi i microcontrollori si blocchino per un qualsiasi motivo.

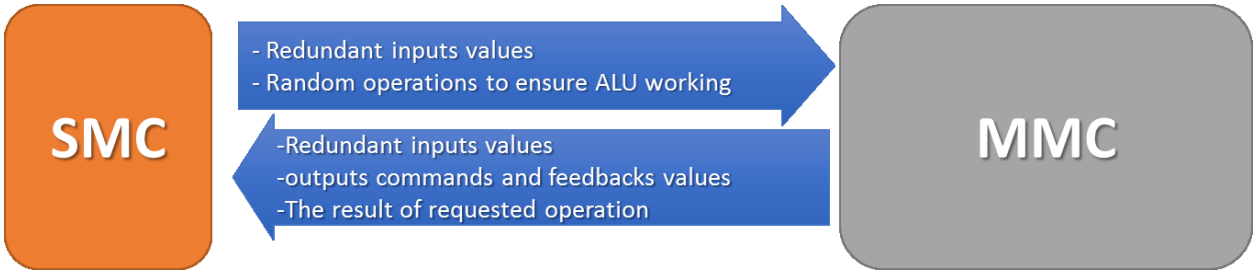
Il microcontrollore di safety può resettare direttamente il micro principale, mentre il contrario non è possibile, in quanto il micro principale può generare una richiesta su un pin di I/O, ma non può resettare il SMC, onde evitare possibili deadlock dove entrambi i micro si tengano resettati l'un l'altro. Il reset del SMC, in caso sia richiesto, è eseguito da un Watchdog esterno pilotato da un segnale ad onda quadra via software.

4.7.6 Periferiche di comunicazione

L'ultima parte di architettura trattata è la ridondanza delle periferiche di comunicazione. Anch'esse possono essere veicolo di informazioni o comandi safety relevant. Per questo motivo dovrebbero essere ridondate. Il SMC non deve comunicare su tali periferiche, bensì essere in grado di poter leggere messaggi e controllare con opportune funzioni che la periferica di comunicazione del MMC sia funzionante.

4.8 Protocollo di comunicazione fra SMC e MMC

Il protocollo di comunicazione fra SMC e MMC è fondamentale per l'architettura hardware descritta. L'hardware da solo non basta per poter ottenere dei Performance Level elevati, ma con delle funzioni di supervisione piuttosto robuste e che offrano una buona copertura delle funzionalità critiche del microcontrollore principale, si è in grado di aumentare le prestazioni del sistema, alleggerendo il peso di eventuali certificazioni sul software che esegue fisicamente il controllo dei canali.



Il protocollo di supervisione si basa su domande periodiche eseguite da SMC a MMC. Queste domande hanno uno schema preciso e servono per verificare il corretto funzionamento delle periferiche interne di MMC, oltre che fornire comunque le informazioni sugli ingressi ridondati per poter applicare strategie da parte del MMC (media dei due ingressi, piuttosto che prendere l'ingresso minore).

Tabella 4.2 - Guasti rilevati dal protocollo di comunicazione MMC-SMC

Guasto	Soluzione attuata nel protocollo
Guasto ALU	Richiesta di operazioni matematiche semplici casuali (addizioni, sottrazioni, moltiplicazioni, divisioni) e confronto col risultato
Errore su SPI	CRC16 di ogni messaggio e verifica
Guasto clock	Richiesta del valore di un Free Running Timer
Guasto periferica CAN	Richiesta del numero di messaggi ricevuti
Inconsistenza su ingressi ridondati	Richiesta dei valori degli ingressi ridondati
Inconsistenza sul feedback degli output	Richiesta dei valori di feedback degli output e dei valori di output

In Tabella 4.1 sono elencati alcuni dei guasti più comuni che il protocollo è in grado di identificare. Inoltre attraverso questi messaggi periodici vengono inviati e ricevuti lo stato di tutti gli ingressi ridondati e le uscite, comprese quelle di abilitazione.

4.9 Gestione dei fault

Lo scopo principale del protocollo di comunicazione tra i due processori e di strategie di controllo locali che si basano su ridondanze di ingressi sia analogici che di stato e su funzioni di plausibilità (che però sono davvero molto strettamente dipendenti dalle specifiche applicazioni e che quindi sono implementate solo nello stadio di personalizzazione della architettura) è quello di individuare i malfunzionamenti, che però non devono portare immediatamente al blocco del sistema. Infatti devono poter essere tollerate solamente delle situazioni anomale che possono verificarsi per un tempo determinato e poi non verificarsi più. Tali situazioni generano degli errori sporadici che non devono essere confusi con guasti, altrimenti si rischia che il sistema diventi inutilizzabile al primo errore di lettura. Le anomalie possono capitare e devono essere distinte dai guasti.

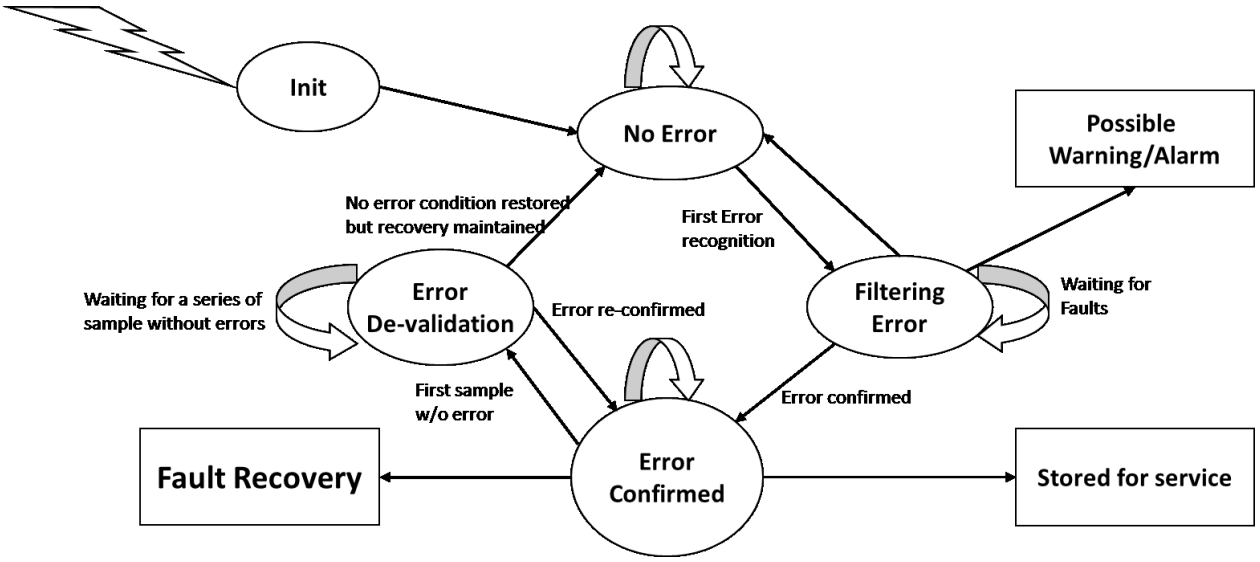


Figura 4.7 - Schema della FSM di diagnostica

Per questo motivo, all'interno dell'architettura è stata creata una macchina a stati per ciascuna linea di diagnosi. Ogni errore trovato da SMC, o relativo a un canale hardware locale a uno dei processori, può provenire da diverse fonti ed essere scorrelato da errori di altro tipo. Ad esempio, il fatto di avere un errore su un ingresso e contemporaneamente uno su CAN non deve portare all'erronea conclusione che ci sia un problema generale, bensì che ci sono due problemi concorrenti, magari generati dalla stessa causa. La diagnosi è fatta per linee, ciascuna delle quali identifica un tipo di guasto. SI prenda ad esempio un ingresso ridondato: esso può avere diverse tipologie di guasti, e per poter avere una diagnostica accurata devono essere trattati con diverse linee:

- Underflow Sensore, quando su un sensore con uscita in tensione controllata diagnosticabile [0.5-4.5] V, si ha uno 0

- Overflow, quando su un sensore 0-5-4.5 si ha 5
- Inconsistenza del segnale ridondato, quando su un sensore si ha un valore diverso rispetto al suo ridondato

In questo esempio, per un solo ingresso sono necessarie cinque linee di diagnosi e i guasti non sono tutti uguali, poiché alcuni possono essere tollerati, dipendentemente dalla funzione che essi realizzano, mentre altri possono portare al raggiungimento forzato di uno stato safe.

Per questo motivo si è creata una struttura generale di macchina a stati standard, configurabile attraverso il passaggio di funzioni personalizzate che gestiscono:

- Il guasto temporaneo,
- il guasto confermato,
- la recovery,
- il guasto de-confermato.

La macchina a stati si divide in 5 stati:

- *Init*, lo stato iniziale di inizializzazione delle strutture
- *No errors*, stato in cui non sono ancora avvenuti errori
- *Fault Recognition*, stato in cui è stato rilevato almeno un errore
- *Fault Confirmed*, stato in cui l'errore è confermato
- *Fault De-Confirmation*, stato in cui, dopo che l'errore è stato confermato, si registra almeno un campione senza errori

Il riconoscimento dei guasti avviene attraverso una funzione statistica, che incrementa un numero per ogni campione errato e lo decrementa per ogni campione senza errori. Per poter eseguire un tuning fine della funzione bisogna tenere conto del tempo massimo di reazione del sistema ad un guasto (FRT), per definire il limite statistico per cui un errore è confermato e il valore di incremento da dare sia per un campione errato sia per uno corretto. Tipicamente un valore corretto decrementa con magnitudo inferiore rispetto all'incremento di un valore errato, proprio per permettere che un sistema che accumula troppi errori sui campioni entri comunque in uno stato di guasto confermato e per limitare il rischio di non riconoscere guasti intermittenti.

Una volta confermato il guasto vengono chiamate delle funzioni di gestione delle recovery e della memorizzazione del guasto stesso. La fase di de-conferma, si comporta esattamente come la fase di conferma, con due differenze: alla fine della de-conferma vengono eseguite altre operazioni per il ripristino

del sistema (ad esempio eliminando delle limitazioni alle uscite) e la deconferma non è uno stato sempre presente. Ci sono alcuni tipi di guasto che richiedono il riavvio completo del sistema (key cycle) per poter essere de-confermati.

4.10 Applicazioni

L'architettura sopra definita ha una valenza di reference design, e la sua bontà si può misurare solo applicandola a casi reali. Durante il presente dottorato essa è stata applicata in due applicazioni, una specifica ed una di valenza generale che verranno descritte nei prossimi capitoli.

4.10.1 Prima applicazione: movimentazione di un braccio telescopico

La prima applicazione di questa architettura generale è stata progettata per realizzare una ECU in grado di controllare un braccio telescopico con cestello su un veicolo off-road da lavoro, un Telehandler.

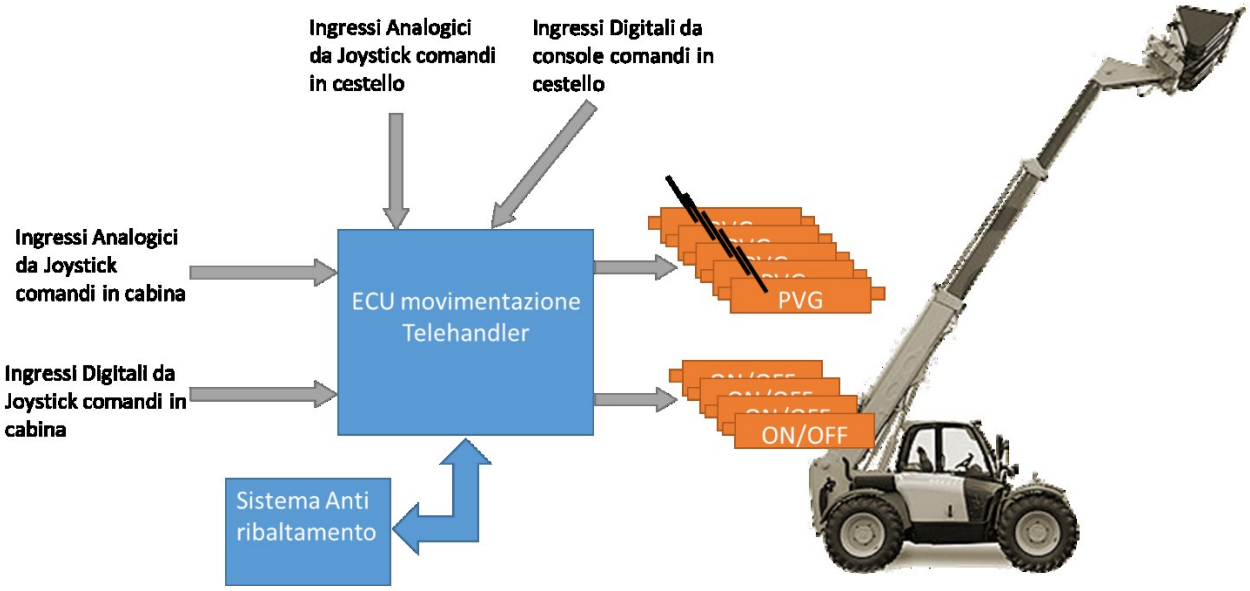


Figura 4.8 - Schema di base della ECU per il controllo di un braccio telescopico

La ECU (il cui schema di base è in Figura 4.8) deve essere in grado di ricevere comandi analogici da joystick e/o pulsanti digitali sia provenienti dalla cabina, sia dal cestello sospeso e inviare comandi a valvole proporzionali e/o a valvole ON-OFF per far eseguire il movimento meccanico al braccio. All'interno del sistema può essere presente un'altra ECU, che ha lo scopo di rilevare lo spostamento del baricentro della macchina in posizioni di pericolo per quanto concerne il ribaltamento. Tale ECU invia un comando di abilitazione ON/OFF, in base alle condizioni di equilibrio del veicolo.

La movimentazione di un braccio telescopico con cestello che può portare persone, richiede almeno un livello di sicurezza classificato come Performance Level C. Inoltre è stato identificato il safe state come stasi degli attuatori, in quanto esiste una recovery a prestazioni ridotte completamente meccanica e idraulica. Attraverso questa recovery meccanica e idraulica è possibile eseguire il by-pass del controllo elettronico, facendo defluire l'olio dalle elettrovalvole e riportare a terra eventuali operatori che si trovano sul cestello ad altezze importanti.

I canali safety relevant sono tutti i segnali analogici da cabina e da cestello in ingresso, e in uscita sono gli attuatori proporzionali. Inoltre, per la possibile presenza di un sistema anti-ribaltamento su CAN, anche la cella CAN e in generale lo stato della rete e della comunicazione devono essere controllate.

4.10.1.1 Realizzazione

Per la realizzazione di questa ECU è stato applicato tutto quanto descritto nei capitoli 4.7, 4.8 e 4.9, potendo contare su una progettazione completa per un'applicazione specifica. La ECU è formata da due microcontrollori, MMC è Fujitsu a 32 bit mentre SMP è un Microchip PIC18 a 8 bit, entrambi programmati in C, però da programmatori diversi.

MMC esegue tutte le funzioni di decodifica dei segnali analogici e digitali provenienti da cabina o cestello e pilota gli attuatori. Su SMC, vista la progettazione molto specifica, sono stati implementati degli algoritmi con strategie decisionali qualitative, che riescono ad analizzare la plausibilità dei comandi di uscita del MMC basandosi sugli ingressi; per fare un esempio di strategia qualitativa descritta in linguaggio naturale: se tutti gli ingressi sono in posizione di stasi, gli output dovrebbero essere anche essi in condizioni di stasi. Quindi, oltre a tutte le diagnosi basate su canale è possibile identificare guasti dovuti alla logica di MMC.

Figura 4.9 - Foto della ECU per il controllo di un braccio telescopico

La ECU realizzata (Figura 4.9) ha le seguenti caratteristiche tecniche:

- 37 ingressi digitali ridondati (37+37)
- 15 ingressi analogici ridondati (15+15)
- 16 uscite digitali di potenza (3 A) a MOSFET protette contro il corto circuito, sovraccarico,inversioni di polarità, sovratensioni. Feedback sul uC del segnale sul connettore.
- 6 uscite analogiche con feedback al uC del segnale sul connettore
- 4 uscite di potenza PWM (3 A) a MOSFET protette contro il corto circuito, sovraccarico,inversioni di polarità, sovratensioni.
- 2 ingressi frequenziali
- 2 uscite open collector
- 5 ingressi digitali
- 2 linee CAN, di cui una ridondata
- 2 linee RS232

Si può notare l'elevato numero di ingressi e uscite disponibili, per poter essere più general purpose possibile e coprire una vasta gamma di applicazioni che eseguono però sempre la stessa classe di funzioni. Inoltre, per ciascuna applicazione è necessario riprogettare il firmware di entrambi i microcontrollori, pertanto ogni applicazione avrà il suo firmware specifico.

4.10.2 Un caso generale

Nel mondo delle macchine movimento terra, essendo caratterizzato da bassi numeri, è preferibile realizzare delle ECU general purpose, in grado di coprire la più vasta gamma di applicazioni possibile. Creare una ECU general purpose è difficile, se poi a questo si aggiunge che certi canali sono safety-relevant, per quanto detto prima, il lavoro si complica.

Una volta validato il modello progettuale, si deve passare a una sua generalizzazione che lo renda applicabile a un più ampio parco di applicazioni.

Il passo successivo della ricerca è quindi stato la realizzazione fisica dell'architettura di safety per ottenere un sistema general purpose, modificabile dall'utente finale. Come vedremo nei prossimi capitoli la realizzazione di tale sistema ha dovuto far fronte a dei compromessi, per poter dare la libertà al programmatore di creare le proprie funzioni e al contempo fare in modo che i canali e il sistema intero siano adeguatamente coperti dalla diagnostica dei guasti.

Per creare un sistema general purpose è necessario fornire all'utente finale la possibilità di poter modificare il codice per creare i propri controlli che si adattano alla macchina su cui il sistema verrà installato. Il fatto di avere un'architettura sufficientemente robusta da non dover richiedere certificazioni del software, semplifica la customizzazione della ECU, ma dipende, oltre che dall'hardware, dalle funzioni di supervisione software descritte precedentemente. Se l'utente riuscisse a bypassare queste funzioni di supervisione, tutta l'architettura perderebbe la sua sicurezza intrinseca.

Quindi, il meccanismo di customizzazione è fondamentale in questa tipologia di ECU e si possono utilizzare diversi approcci:

1. Dare a disposizione all'utente finale il codice sorgente, completo di funzioni di safety, per la customizzazione
2. Rilasciare una libreria chiusa, contenente le funzioni di safety, all'utente finale
3. Scrivere il codice customizzato per l'utente finale
4. Fornire un framework di sviluppo che "guidi" l'utente finale allo sviluppo dei propri controlli.

La prima soluzione è la più semplice, però implica la fornitura del codice sorgente al cliente, esponendo il proprio know how. Inoltre lo svantaggio principale di questo approccio è che se l'utente non ha familiarità con il codice C e ha a disposizione il codice sorgente delle funzioni di safety, potrebbe utilizzarle in maniera sbagliata o addirittura modificarle, compromettendo la sicurezza di tutta l'architettura.

La seconda soluzione è simile alla prima, senza l'esposizione del codice sorgente, ma richiede da parte del fornitore la costruzione di un HAL per l'interfacciamento corretto delle proprie funzioni di safety. In questo modo il sistema è più protetto da malfunzionamenti e bug dovuti alla scarsa qualità del codice di controllo dell'utente finale. Si ha comunque a che fare con un codice di basso livello, il C, che non può proteggere la libreria, ad esempio, da accessi non autorizzati all'hardware.

La terza soluzione è la più sicura, poiché si basa sul fatto che l'utente finale conosce il controllo che vuole implementare, mentre il fornitore della ECU conosce a fondo l'architettura di safety. Questo però porta a costi elevati di personalizzazione per l'utente finale, che si deve appoggiare completamente al costruttore per vedere realizzato il proprio controllo. Questo porta a una dipendenza del cliente verso il fornitore, oltre a una esposizione del know how del cliente e a tempi di sviluppo maggiori, dovuto dall'overhead delle comunicazioni tra aziende.

La quarta soluzione richiede maggiori sforzi da parte del produttore che per una ECU deve creare un vero e proprio framework con un IDE per fare in modo che l'utente possa programmare con un linguaggio sicuro e litato. Tale linguaggio dovrebbe essere semplice, in grado di implementare i più comuni controlli di automazione, e al contempo limitare la possibilità che l'utente possa creare errori o eludere le funzioni di safety. Sicuramente questo è l'approccio migliore per poter realizzare un sistema safety general purpose poiché lascia liberta agli utenti di personalizzare il controllo, mentre il costruttore fornisce tutte le funzioni di diagnosi e di sicurezza richieste dalle normative ISO 13849 / ISO15998, ISO 25119 e, in generale, IEC 61508.

La costruzione di un framework con pseudo-linguaggio è costosa in termini di tempo e di know-how, oltre che probabilmente inutile: esistono già molti framework con linguaggi adatti per lo sviluppo di controlli usati in automazione industriali, di cui i più famosi sono Codesys, PCWorx e Step 7.

Questi framework utilizzano i linguaggi definiti nella norma IEC 61131-3, per applicazioni industriali e PLC. Tali linguaggi vengono descritti nelle normative di safety come preferibili [22] [23] per la realizzazione di funzioni di controllo. Essi sono definiti nella norma IEC 61511-1 come LVL (Limited Variability Language), in quanto limitano alcune strutture di programmazione tipiche di linguaggi a basso livello con maggiore espressività quali C, C++, definiti FVL (Full Variability Language).

Un linguaggio di programmazione LVL non permette alcuni costrutti come i costrutti ricorsivi o l'utilizzo di puntatori, eliminando la possibilità di, rispettivamente, funzioni non deterministiche che possano generare overflow dello stack o creare errori non predicibili derivanti dall'uso scorretto dei puntatori.

Dal punto di vista della presente ricerca, cioè dell'applicazione dell'architettura di safety per un sistema general purpose, i framework industriali forniscono le stesse caratteristiche. Il framework scelto è Codesys di 3S-Smart Software Solutions GMBH, per la sua ampia diffusione nel mondo industriale, la possibilità di poter integrare il run-time all'interno delle proprie ECU (con il pagamento di una fee e di royalties) e per i costi ridotti rispetto ad altre soluzioni.

4.10.2.1 L'ambiente Codesys

Il framework Codesys si divide in due parti:

- una parte PC, che fornisce l'IDE di sviluppo, compilatori, debugger, e parser dei linguaggi IEC 61131-3,
- una parte embedded, che è formata da un sistema operativo real-time (chiamato Codesys Run-Time Enviornement) in grado di eseguire il codice compilato a partire dai linguaggi IEC 61131-3.

L'IDE Codesys 2.6 fornisce i seguenti linguaggi di programmazione IEC 61131-3:

- IL - Instruction List, un linguaggio di programmazione testuale, con sintassi simil-Assembler. È un linguaggio FVL.
- ST – Structured Text, un linguaggio di programmazione testuale, con sintassi che ricorda il Pascal.
- FDB – Function Block Diagram, un linguaggio di programmazione grafico, dove ogni funzione è rappresentata da un blocco, simile al linguaggio G di Labview.
- SFC – State Function Chart, un linguaggio di programmazione grafico, dove ciascun blocco rappresenta uno stato del sistema, utile per la programmazione di PLC a macchine a stati.
- LD – Ladder Diagram, un linguaggio di programmazione grafico, che utilizza la sintassi elettrica per definire il comportamento del sistema.

La parte di run-time è scritta in C ed è disponibile compilata per diverse piattaforme [24] e compilatori. Essa implementa un sistema operativo con componenti personalizzabili, quali protocolli di comunicazione (come CANOpen, J1939, TCP/IP, Ethercat, TTEthernet), o moduli HMI. Il sistema operativo consta di uno scheduler in grado di agganciarsi a task compilati all'interno dell'IDE e di un *Hardware Abstraction Layer* che riporta le periferiche di qualunque microcontrollore come variabili di ingresso o di uscita, eseguendo i necessari controlli di validità.

I programmi in Codesys sono costituiti di diverse *Program Organization Unit* (POU), che non sono altro che task con priorità e frequenza di esecuzione personalizzabili. Ciascun task può essere scritto in qualunque dei linguaggi di programmazione disponibili, consentendo all'utente finale un'ampia personalizzazione e

modellazione del proprio sistema. Inoltre tutte le periferiche di I/O, essendo mappate come variabili, possono essere utilizzate come tali, lasciando l'utente fuori da tutto ciò che riguarda i driver di basso livello, come la configurazione di registri o altre procedure *error-prone*.

4.10.2.2 Limiti di progettazione

L'astrazione introdotta dal Codesys RTE, come qualsiasi altro framework, si paga in termini di performance massimi del dispositivo embedded: con un micro a 100 MHz il tempo minimo di task è sui 2 millisecondi, mentre per controlli comuni tale tempo sale a 10 ms. Per controlli più veloci, o magari personalizzati, la soluzione con Codesys dovrebbe essere evitata, proprio per i costi dei microcontrollori richiesti [25].

Inoltre per poter implementare le funzioni di sicurezza "nascoste" all'utente finale è necessario integrarle sui sorgenti del Codesys RunTime, implementando tali funzioni come task privilegiati, non modificabili né evitabili dall'utente. Infine attraverso tale integrazione le funzioni di safety possono accedere ai registri del microcontrollore e delle periferiche per ottenere un'elevata sincronizzazione tra MMP e SMP, oltre che accedere a periferiche, come ad esempio la SPI, normalmente non supportate in User Space.

Per poter modificare il Codesys RunTime è necessario acquistare il codice sorgente, modificarlo e creare un hardware ad hoc, che per piccole produzioni è anti-economico. Inoltre tutti i plug-in offerti per protocolli di comunicazione, come J1939 o CANOpen, dovrebbero essere re-implementati aumentando ulteriormente i tempi di realizzazione e il debug. L'utilizzo di System On Module già con a bordo il Codesys RTE con diversi plug-in è economicamente più conveniente, senza contare che l'utilizzo di moduli COTS presenti sul mercato da un certo tempo, possono essere considerati come sistemi già ampiamente testati, e quindi con probabilità di errori hardware o software molto minori, rispetto all'implementare un'architettura da 0.

Si è inoltre voluta testare la robustezza dell'architettura studiata, provandone una implementazione che comprendesse anche queste limitazioni e quindi utilizzando componenti COTS con Codesys integrato; si è operato quindi sull'architettura generale del sistema, per ottenere comunque delle funzioni di safety non eludibili né modificabili dall'utente finale.

4.10.2.3 Realizzazione

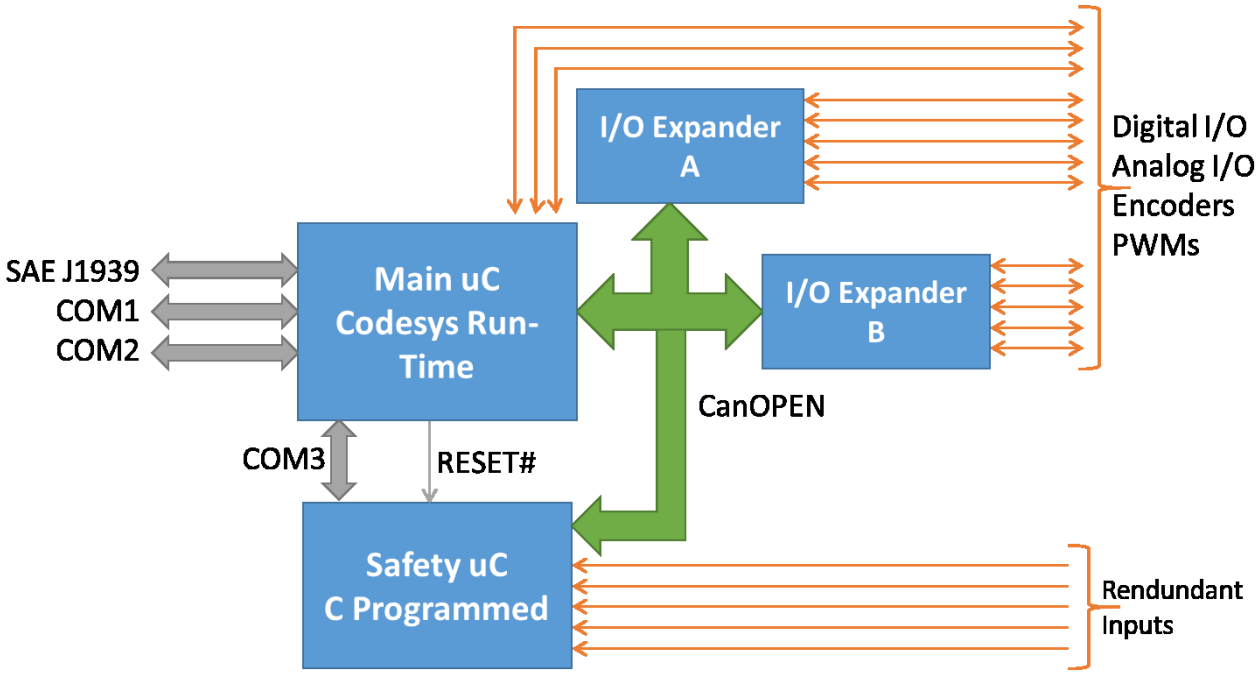


Figura 4.10 - Struttura di base della ECU General Purpose con canali safety

La ECU general purpose (Figura 4.10) realizzata è formata dai seguenti componenti:

- MMC – SoM con
  - 16 Bit ST10F273 CPU
  - 2 MByte program data
  - 512 kByte RAM and 16 kByte EEPROM data memory
  - 2 CAN
  - 4 UART
  - 1 SPI MASTER
  - CanOPEN Master plug-in
  - J1939 plug-in
  - External Watchdog
- SMC – PIC18F8680,
- 2 nodi di espansione CANOpen Slave con 32 I/O riconfigurabili come
  - Max 16 ingressi digitali
  - Max 16 uscite digitali
  - Max 8 ingressi analogici
  - Max 8 uscite PWM
  - Watchdog interno

Il microcontrollore principale (MMC), grazie ai due I/O Expander è in grado di gestire più di 80 linee di I/O, molte delle quali riconfigurabili. Per piccole produzioni di ECU diverse vuol dire alta versatilità per la costruzione della scheda, che può prevedere alcuni ingressi e uscite riconfigurabili, attraverso la depauperazione di alcune linee piuttosto che di altre. Grazie a Codesys con il plug-in CanOpen, l'accesso agli I/O Expander, a livello utente è completamente trasparente, poiché essi vengono mappati in variabili (in scrittura/lettura per le uscite, in lettura per gli ingressi) come se si trovassero sul microcontrollore principale.

Le funzioni di safety e di diagnostica sono implementate all'interno del SMC, affinché non possano essere modificate. Su SMC è stato implementato il protocollo descritto in 4.8, ridotto però di alcune informazioni.

Mentre nell'architettura progettata si pensava ad un meccanismo di sincronizzazione basato su una linea di I/O e SPI, su questa tipologia di ECU non è possibile avere alcuna sincronizzazione di questo genere per due motivi:

- non è chiaro il l'istante di tempo in cui vengono campionati tutti gli ingressi, poiché ciascun POU può eseguire a rate diversi. Tale limitazione è dovuta all'HAL introdotto da Codesys,
- gli ingressi campionati si potrebbero trovare su I/O Expander, peggiorando le cose.

È possibile però sfruttare il fatto che gli I/O Expander si trovino su CAN e utilizzino il protocollo CanOpen per poter ottenere un sincronismo maggiore e bypassare i problemi derivanti dall'HAL di Codesys. Per questo motivo sul processore di sicurezza SMC è stato implementato una parte del protocollo CanOpen al fine di poter leggere i PDO (Process Data Object) dei comandi (Master a Slave) e delle letture (da Slave a Master). Inoltre, il protocollo CanOpen può utilizzare dei messaggi di SYNC per la lettura e scrittura dei PDO, permettendo quindi un livello di sincronismo determinabile a priori.

La configurazione CanOpen non può essere cambiata dall'utente finale poiché tutti gli I/O Expander hanno un indirizzo determinato dalla configurazione di alcuni PIN, così come la velocità del CAN. Inoltre le porte di I/O, per quanto possano essere configurabili, rimangono attaccate ad un hardware che condiziona il segnale in un determinato modo (non si può pensare di configurare un pin di I/O come analogico, se la rete hardware sottostante è collegata ad un'uscita).

Il SMC regola le proprie acquisizioni sul messaggio SYNC del master, in modo da essere il più sincronizzati possibile con le acquisizioni degli slave CanOPEN di MMC. Attraverso il protocollo CANOPen il SMC è in grado di eseguire le seguenti diagnosi:

- underflow ingressi,

- overflow ingressi,
- valore discordante ingressi ridondati,
- assenza di messaggio di heartbeat degli slave,
- assenza di messaggio di heartbeat del master,
- underflow uscite (se analogiche),
- overflow uscite (se analogiche),
- feedback non coerente con le uscite,
- errori di configurazione del canOpen.

Per controllare le periferiche interne del microcontrollore il SMC si deve appoggiare ad un meccanismo di comunicazione asincrono, basato su messaggi. MMC implementa la parte “slave” di questo protocollo e risponde in maniera sincrona alle richieste del SMC.

Lo “slave” su MMC è stato implementato in linguaggio ST, come *parser* a macchina a stati, che raccoglie le informazioni richieste da SMC e invia le risposte con il proprio stato interno. Il SMC attraverso il protocollo seriale può diagnosticare i guasti su periferiche interne al microcontrollore quali:

- guasti all’ALU, mediante richiesta di operazioni matematiche
- guasti ai timer, mediante richiesta del valore di un free running timer ad una frequenza prestabilita

Infine viene reso disponibile, attraverso il protocollo il valore degli ADC campionati ridondati per poter dare modo all’utente finale su Codesys di attuare strategie di controllo adeguate basandosi su un doppio segnale (media, maggiore dei due, minore dei due).

Tutte le uscite di sicurezza sono energizzate col meccanismo di doppia abilitazione descritta in 4.7.4.

Lo schema a reset gerarchico è assicurato dal watchdog esterno montato sul SoM, il cui Feed viene fatto attraverso una funzione di libreria su Codesys. Tale funzione è inserita nel task con priorità più bassa a livello utente.

I canali di sicurezza sono predefiniti e non modificabili, dato uno specifico hardware, pertanto la configurazione di tali canali e dei parametri delle macchine a stati delle diagnosi è tenuta all’interno della memoria non volatile del SMC, quindi non accessibile all’utente finale.

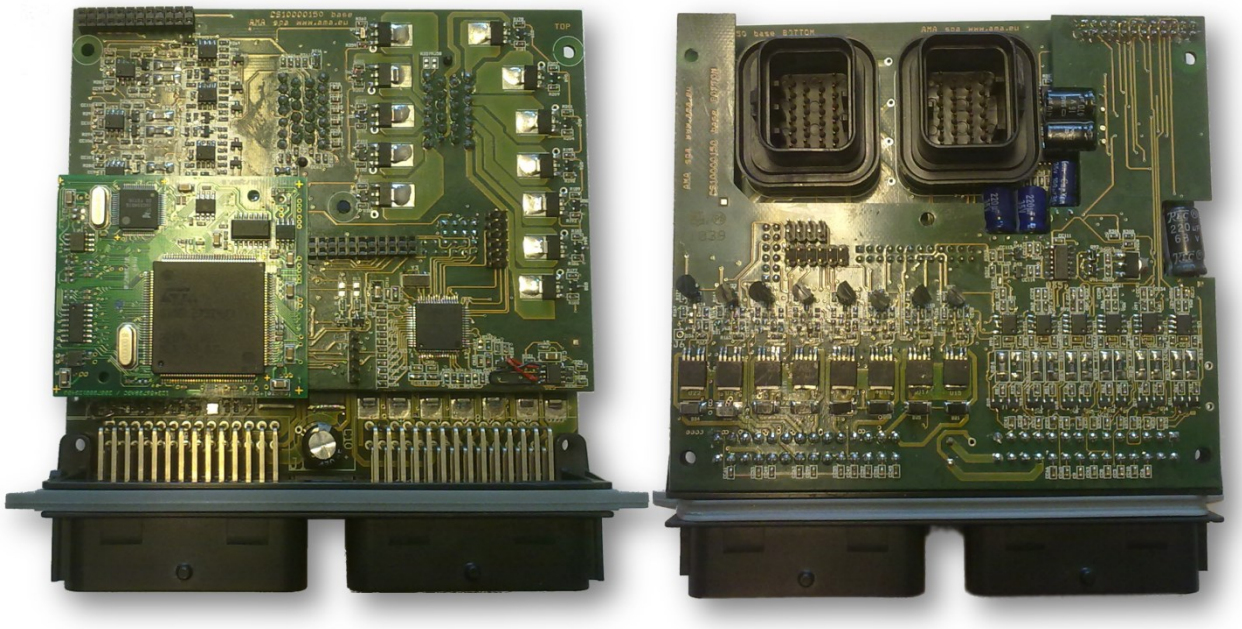


Figura 4.11 - Foto della scheda protipale di ECU General Purpose con canali safety-relevant

La ECU realizzata (Figura 4.11), con due I/O Expander presenta le seguenti caratteristiche di I/O

- 8 Uscite analogiche 3A con Feedback su uC (Safety),
- 16 Uscite digitali 3° con Feedback su uC (Safety),
- 8 Ingressi analogici ridondati (Safety),
- 16 Ingressi digitali protetti ridondati (Safety),
- 1 CAN con protocollo J1939,
- 2 seriali UART,
- 2 ingressi in frequenza,
- 1 ingresso encoder,
- 1 uscita a mezzo ponte H.

4.11 Conclusioni

L'architettura di safety sviluppata si è rivelata facilmente adattabile sia per applicazioni specifiche, sia per applicazioni general purpose. Nonostante l'architettura general purpose presenti alcune limitazioni, come ad esempio l'impossibilità di un sincronismo perfetto sui segnali in ingresso e l'improteggibilità delle funzioni di safety lato MMC, ha moltissimi vantaggi dal punto di vista della personalizzazione da parte dell'utente finale, come l'utilizzo di linguaggi a variabilità limitata e la semplice integrazione di funzioni di controllo. Inoltre l'architettura si è adattata in modo ottimale alla configurazione con I/O Expander,

bypassando in parte il problema del sincronismo e al contempo è in grado di poter dare al sistema la scalabilità necessaria per gestire un numero maggiore di I/O di safety, mantenendo la stessa struttura di base. Infine, nonostante la parte di comunicazione slave sia disponibile all'utente finale, essa non può essere modificata né bypassata, pena il passaggio immediato allo stato safe, garantito dal meccanismo di doppia conferma. L'alta copertura diagnostica fornita dal protocollo di comunicazione fra SMC e MMC è in grado di fornire all'utente finale un sistema personalizzabile e a elevato grado di sicurezza.

La validazione finale di tale architettura viene dal mondo dell'industria, per cui queste ECU sono state realizzate. La prima ECU è stata certificata per un'applicazione di sollevamento gabbie con persone a bordo con Performance Level C, mentre la seconda è stata definita capace di arrivare ad un Performance Level D.

5 CONCLUSIONI FINALI

Entrambi i filoni di ricerca, portati avanti parallelamente durante il dottorato, hanno portato a risultati diversi, dovuti al tipo diverso tipo di obiettivo, ma positivi. I risultati della ricerca su architetture di rete ha portato alla creazione di una nuova rete scalabile, ad alto throughput, con maggiore sicurezza e compatibile con lo standard attuale. Tale rete lascia grande spazio per poter realizzare i *deliverable* richiesti dalla commissione ISO TC23/SC19/WG1 ed è stata giudicata positivamente dalla commissione stessa, che ha avanzato la volontà di voler proseguire il lavoro, costituendo una task force dedicata, con i lunghi tempi che caratterizzano il più grande ente di normazione mondiale. La ricerca industry-driven ha portato diversi risultati, sia in termini di applicabilità, sia in termini di effettiva realizzazione dell'architettura hardware-software a sicurezza intrinseca, utilizzata non solo per le applicazioni descritte in questa tesi, bensì applicata anche a due applicazioni reali, una di controllo sollevatore ed una di controllo acceleratori, freno e PTO, che il dottorando ha avuto modo di sviluppare per una azienda nell'ambito delle attività di dottorato.

Le ricerche presentate offrono una via di sviluppo possibile per architetture di controllo distribuito per i veicoli off-road di futura generazione, ma che potrebbe essere applicata anche al mondo automotive e al mondo dei *commercial vehicle*, completamente basata su software open source e su componentistica esistente.

Tali ricerche dimostrano inoltre, che il collegamento diretto tra il mondo della ricerca e quello della industria è in grado di sintetizzare risultati, non solo validi, ma anche immediatamente applicabili a sistemi reali.

6 RINGRAZIAMENTI

Un sincero ringraziamento va al tutor di questa tesi, il prof. Massimiliano Ruggeri, per la sua disponibilità, la pazienza e la fiducia accordata nei miei confronti, permettendomi di partecipare attivamente in prima persona all'interno del comitato WG1, dove ho presentato questi lavori, e nei comitati tecnici AEF di certificazione e safety, di cui ho fatto parte come esperto tecnico.

Questa tesi è stata svolta all'interno dell'istituto IMAMOTER, pertanto vorrei ringraziare il direttore, l'Ing Roberto Paoluzzi, per la disponibilità e l'incoraggiamento a proseguire nelle ricerche sulle reti agricole del futuro, e per gli utili consigli.

Ringrazio i miei colleghi e amici, aumentati in questi anni, Giorgio Malaguti, Carlo Ferraresi e Luca Dariz, per le animate discussioni e scambi di idee sulle reti del futuro, sulla safety e sulle architetture di sistema, che mi hanno fornito sempre nuovi stimoli e spunti per il dottorato e per il lavoro.

Un ultimo ringraziamento a Ilaria Laurenti, per la pazienza dimostrata nei miei confronti e il supporto fornitomi da sempre, ma soprattutto in questo periodo, in cui la mia attenzione è stata assorbita completamente dalla scrittura della tesi, tralasciando qualunque cosa.

7 ABBREVIAZIONI

- E/E/PE: Electrical/Electronic/Programmable Electronic
- ECU: Electronic Controller Unit
- EHSR: Requisiti di sicurezza per la direttiva machine (Essential health and safety requirements)
- VT: Virtual Terminal
- TECU: Tractor ECU
- TC: Task Controller
- SC: Sequence Controller
- OP: Object Pool
- TCOP: Task Controller Object Pool
- SCOP: Sequence Controller Object Pool
- WSM: Working Set Master
- PGN: Parameter Group Number
- PDU: Program Data Unit
- PTO: Power Take Off
- SA: Source Address
- DA: Destination Address
- AUXI: Auxiliary Input
- AUXF: Auxiliary Function
- NIU: Network Interconnection Unit
- OUI: Organization Unique Identifier
- CSMA/CD: Carrier Sense Multiple Access with Collision Detection
- CSMA/CA: Carrier Sense Multiple Access with Collision Avoidance
- CSMA/BA: Carrier Sense Multiple Access with Bit Arbitration

8 BIBLIOGRAFIA

[1] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 1: General standard for mobile data and communication,* ISO, 2007.

[2] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 2: Physical layer,* ISO, 2012.

[3] SAE, *J1939-11 - Physical Layer, 250K bits/s, Twisted Shielded Pair.,* SAE, 2006.

[4] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 5: Network Management,* ISO, 2011.

[5] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 3: Data link layer,* ISO, 2007.

[6] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 6: Virtual Terminal,* ISO, 2010.

[7] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 4: Network layer,* ISO, 2011.

[8] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 7: Implement messages appplication layer,* ISO, 2009.

[9] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 10: Task Controller and management information system data interchange,* ISO, 2009.

[10] Beckhoff GMBH, «EtherCAT for Safety,» 2007. [Online]. Available: http://www.ethercat.org/pdf/english/pcc0107_safety_over_ethercat_e.pdf. [Consultato il giorno 20 Dicembre 2013].

[11] G. Prytz, «A performance analysis of EtherCAT and PROFINET IRT,» in *13th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), pp 408-415*, Hamburg, 2008.

[12] BerliOs, «Open Ethercat,» [Online]. Available: http://soem.berlios.de/. [Consultato il giorno 2013].

[13] M. G. W. K. Miroslaw Wlas, «The Ethernet POWERLINK Protocol for Smart Grids Element Integration,» in *IEEE International Symposium on Industrial Eletronics (ISIE)*, Gdansk, 2011.

[14] TTTech, «TTEthernet - A powerful network solution for all purposes,» 2010. [Online]. Available: http://www.tttech.com/fileadmin/content/general/secure/TTEthernet/TTTech-TTEthernet-Scalable_Real-Time_Ethernet_Platform-Whitepaper.pdf. [Consultato il giorno 13 Dicembre 2013].

[15] B. F. A. M. R. W. Richard Stevens, Unix Network Programming, Volume 1: The Sockets Networking API (3rd Edition), Addison-Wesley Professional Computing Series, 2003.

[16] Intel, «What is the Difference Between Class I and Class II Hubs?,» 31 Marzo 2008. [Online]. Available: http://www.intel.com/support/express/hubs/sb/CS-020686.htm. [Consultato il giorno 12 Gennaio 2013].

[17] R. Braden, *Requirements for internet hosts communication layers,* RFC-1122, IETF Network Working Group, 1989.

[18] E. Gerich, «RFC 1466 - Guidelines for Management of IP Address Space,» 1993. [Online]. Available: http://www.rfc-editor.org/rfc/rfc1466.txt. [Consultato il giorno 2013].

[19] Droms, «RFC 2131 - Dynamic Host Configuration Protocol,» 1997. [Online]. Available: http://tools.ietf.org/html/rfc2131. [Consultato il giorno 2013].

[20] Information Sciences Institute, «RFC791 - INTERNET PROTOCOL,» 1981. [Online]. Available: http://tools.ietf.org/html/rfc791. [Consultato il giorno 2013].

[21] N. Horman, «Understanding And Programming With Netlink Sockets,» 2004. [Online]. Available:

http://people.redhat.com/nhorman/papers/netlink.pdf. [Consultato il giorno 2013].

[22] K. Toon, «IEC 61131-3 in safety applications,» in *The Application of IEC 61131 in Industrial Control (Ref. No. 2002/060), IEE*, 2002.

[23] IEC, *Functional safety of eletrical/electronic/programmable elctronic safety-related systems,* IEC, 2010.

[24] 3S-Smart Software Solutions GmbH, «CODESYS Control Runtime Toolkit,» [Online]. Available: http://www.codesys.com/products/codesys-runtime/runtime-toolkit.html. [Consultato il giorno 2013].

[25] M. M. V. F. Massimiliano Ruggeri, «CoDeSys vs Embedded Approach to Electronic Control Design for Small Production Series: a Case Study,» in *10th Scandinavian International Conference on Fluid Power, SICFP*, Tampere, 2011.

[26] IEC, *Functional safety of eletrical/electronic/programmable elctronic safety-related systems,* IEC, 2010.

[27] IEC, *Functional safety of eletrical/electronic/programmable elctronic safety-related systems,* IEC, 2010.

[28] ISO, *ISO 15998 Earth-moving machinery - Machine-control systems (MCS) using electronic components,* ISO, 2008.

[29] Bosch GMBH, «CAN with Flexible Data-Rate - White Paper - v1.1,» [Online]. Available: http://www.bosch-semiconductors.de/media/pdf_1/canliteratur/can_fd_spec.pdf. [Consultato il giorno 01 02 2014].

[30] G. Mazzini, Reti di telecomunicazioni, Bologna: Pitagora Editrice, 2002.

[31] A. Revenaz, *NUOVI PROTOCOLLI PER GESTIONE DATI E COMUNICAZIONE REAL-TIME IN AMBITO AGRICOLO,* Ferrara, 2010.

[32] M. R. V. T. Alfredo Revenaz, «Low Latency WI-FI Real-Time Protocol for Agricultural Machines Synchronization Using Linux RT Kernel,» in *20th International Symposium on Industrial Electronics*

(ISIE), Gdansk, 2011.

[33] ISO, *ISO 25119 - Tractors and machinery for agriculture - Safety-related parts of control systems. Part 1: General principles for design and development,* ISO, 2010.

[34] ISO, *ISO 25119 - Tractors and machinery for agriculture - Safety-related parts of control systems. Part 3: Series development, hardware and software,* ISO, 2010.

[35] ISO, *ISO 25119 - Tractors and machinery for agriculture - Safety-related parts of control systems. Part 4: Production, operation, modification and supporting processes,* ISO, 2010.

[36] ISO, *ISO 25119 - Tractors and machinery for agriculture - Safety-related parts of control systems. Part 2: Concept phase,* ISO, 2010.

[37] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 3: Data link layer,* ISO.

[38] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 8: Power train messages,* 2006.

[39] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 9: Tractor ECU,* ISO, 2012.

[40] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 11: Mobile data element dictionary,* ISO, 2011.

[41] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 12: Diagnostics services,* ISO, 2009.

[42] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 13: File Server,* ISO, 2011.

[43] ISO, *ISO 11783 - Tractors and machinery for agriculture and forestry - Serial control and communications data network - Part 14: Sequence Control,* ISO, 2013.

9 ARTICOLI PUBBLICATI IN CONFERENZE

[1] Ruggeri M., Fracassi M., Martelli M., Dian M., "Fixed Point Versus Floating Point Mathematics In Embedded System Programming For Fluid Power Mechatronic Components Control: A Real Case Study", Proceedings of the 8th JFPS international Symposium on Fluid Power, Okinawa (Japan), 2011.

[2] Ruggeri M., Fracassi M., Martelli M., Dian M., "Two Stage Flow Regulation Valve Control Optimization By Software Techniques And Mathematics Of Digital Systems Approach", Proceedings Of 8th International Fluid Power Conference (8.IFK). Group E - Control Of Fluid Power Systems Paper E7; Pp. 477-489. March 26-28, 2012 - Dresden, Germany.

[3] Malaguti G., Dian M., Ruggeri M., "Real Time Distributed Control On Machines Over Quasi Deterministic Ethernet", Proceedings Of 7th FPNI PhD Symposium Om Fluid Power; University Of Modena And Reggio Emila - June 27-30, 2012, Pp. 249-263. ISBN 978-88-7559-069-7.

[4] Dian M., Malaguti G., Ruggeri M., "A Real-Time Wireless Protocol Proposal For Remote Agricultural Machine Control", Proceedings Of 7th FPNI PhD Symposium Om Fluid Power; University Of Modena And Reggio Emila - June 27-30, 2012, Pp. 209-224. ISBN 978-88-7559-069-7.

[5] Ferraresi C., Dian M., Malaguti G., Ruggeri M., "Isobus Over Ethernet: A First Implementation", Proceedings Of 7th FPNI PhD Symposium Om Fluid Power; University Of Modena And Reggio Emila - June 27-30, 2012, Pp. 751-766. ISBN 978-88-7559-069-7.

[6] Dian M., Malaguti G., Ruggeri M., "A Safety Compliant Universal Machine Control Unit using Codesys", The 13th Mechatronics Forum International Conference, September 17 - 19, 2012, Linz, Austria PP 122-129. ISBN 978-3-99033-042-5

[7] Ruggeri M., Malaguti G., Dian M., Ferraresi C., "Quasi Isochronous Wireless Communication Protocol For Co-Operative Vehicle Clusters", 12th European Regional Conference of the International Society for Terrain-Vehicle Systems – September 24-27, 2012, Pretoria, South Africa

[8] Ruggeri M., Malaguti G., Dian M., "Real Time Ethernet For Heavy-Duty Vehicle Powertrain Control", 12th European Regional Conference of the International Society for Terrain-Vehicle Systems – September 24-27, 2012, Pretoria, South Africa

[9] Ruggeri M., Dian M., Malaguti G., "SAE J 1939 Over Real Time Ethernet: the Future of Heavy Duty Vehicle Networks", SAE 2012 Commercial Vehicle Engineering Congress

[10] Malaguti G., Dian M., Ruggeri M. – "UDP Based Inter/Intra Task Communication For Processes Independence In Safety Critical Embedded Applications“ - Proceedings Of "The 2013 IEEE International Symposium On Industrial Electronics (ISIE 2013)"

[11] Malaguti G., Dian M., Ferraresi C., Ruggeri M. - “Comparison On Technological Opportunities For In-Vehicle Ethernet Networks” - Proceedings Of INDIN 2013, IEEE International Conference On Industrial Informatics Fascicolo 1, Vol.1/2013. 28-30 Luglio 2013. Bochum, Germany

[12] Malaguti G., Dian M., Ruggeri M. - "Performance Comparison On Traffic Control Methods For In-Vehicle Ethernet Networks" - SAE International Papers. SAE COMVEC 2013.

10 ARTICOLI SU RIVISTA

[1] M. Ruggeri, G. Malaguti, M. Dian - "The Future of Heavy-Duty Networking" - Off HighWay Engineering Journal, SAE International, USA, Vol. 4, n. 12, 13-12-2012. (ISSN 1939-6686)