UNIVERSITÀ
DEGLI STUDI
DI PADOVA

DIPARTIMENTO
DI INGEGNERIA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

"SISTEMI MICROELETTRONICI E MICROINFORMATICI PER LA GESTIONE DELL'AUTOMOBILE"

Relatore: Prof. Zanoni Enrico

Laureando: Boscolo Meneguolo Luca

ANNO ACCADEMICO 2021 – 2022

17/03/2022

Indice

Introduzione

La prima autovettura della storia fu la Benz Patent Motorwagen, ingegnerizzata da Karl Benz e prodotta nel 1886 dalla casa tedesca Benz & Cie. All'apparenza è simile a una bicicletta a tre ruote, ma ha una particolarità che la differenzia da qualsiasi creazione dell'epoca: la presenza di un motore a scoppio. Si tratta di un motore monocilindrico da 577 cm^3 erogante una potenza di 0.75 CV, che permetteva al veicolo di raggiungere la velocità di 16 km/h. Questo veicolo fu un capolavoro della meccanica e tecnica; tuttavia, per i nostri standard appare sicuramente poco sicuro e pratico.

Le automobili sono cambiate radicalmente nel corso dei decenni, diventando sempre più ricche di funzioni e comfort. Le prime automobili fino agli anni '40 non disponevano di componenti elettrici, e tutte le funzioni erano ottenute mediante semplici sistemi meccanici o pneumatici. Questo portava ad avere componenti semplici, ma anche poco affidabili e precisi.

Successivamente vennero introdotti i primi dispositivi elettrici classici, come tergicristalli e finestrini elettrici. Tuttavia, questi componenti non giocavano un ruolo fondamentale nelle funzioni primarie del veicolo, ma si limitavano a compiere funzioni di contorno.

La vera rivoluzione è dovuta all'avvento dell'elettronica a microcontrollore: nei veicoli vengono installati dei veri e propri computer che regolano i parametri del funzionamento del motore. Inizialmente i veicoli montavano uno o in genere pochi controllori; successivamente il numero di controllori aumentò a scapito dell'efficienza del sistema. Si passò dunque ad architetture ottimizzate, sfruttando anche la maggior potenza computazionale data dai controllori multicore.

Il progresso tecnologico e la ricerca di maggiore comfort promuovono l'uso di componenti elettronici in moltissimi ambiti nuovi, e questo comporta potenziali vulnerabilità per la sicurezza. I dispositivi informatici devono essere pensati per far fronte alla possibilità di attacchi informatici, manomissioni e malfunzionamenti dell'hardware, adottando controlli di integrità e protocolli specifici per la condivisione di dati in maniera sicura.

Capitolo 1

La comparsa dell'elettronica in campo automobilistico

Un modello estremamente basilare di un veicolo, dotato solamente di motore, ruote e controlli, non necessita quasi per nulla di componenti elettronici: tutte le parti essenziali possono essere gestite meccanicamente e pneumaticamente. Unica eccezione è la candela, un componente elettrico situato nelle testate dei cilindri che ha l'importante scopo di accendere la miscela di aria e combustibile.

A partire dagli anni '40 cominciarono a diffondersi fra le autovetture di diversi produttori nuovi accessori elettrici. Questi dispositivi non erano essenziali al funzionamento intrinseco del veicolo, bensì erano da considerarsi degli optional per migliorare l'esperienza di guida.

1.1 Elettrificazione

La prima evoluzione elettronica nelle automobili fu la cosiddetta *elettrificazione* che, a partire dagli anni '40, portò all'utilizzo di circuiti elettrici per svolgere alcune funzioni finora svolte con strumenti meccanici. Non essendo ancora disponibili componenti elettronici (l'invenzione del transistor bipolare risale al 1947) si utilizzano componenti elettromeccanici: l'interruttore o il relè aziona direttamente l'attuatore.

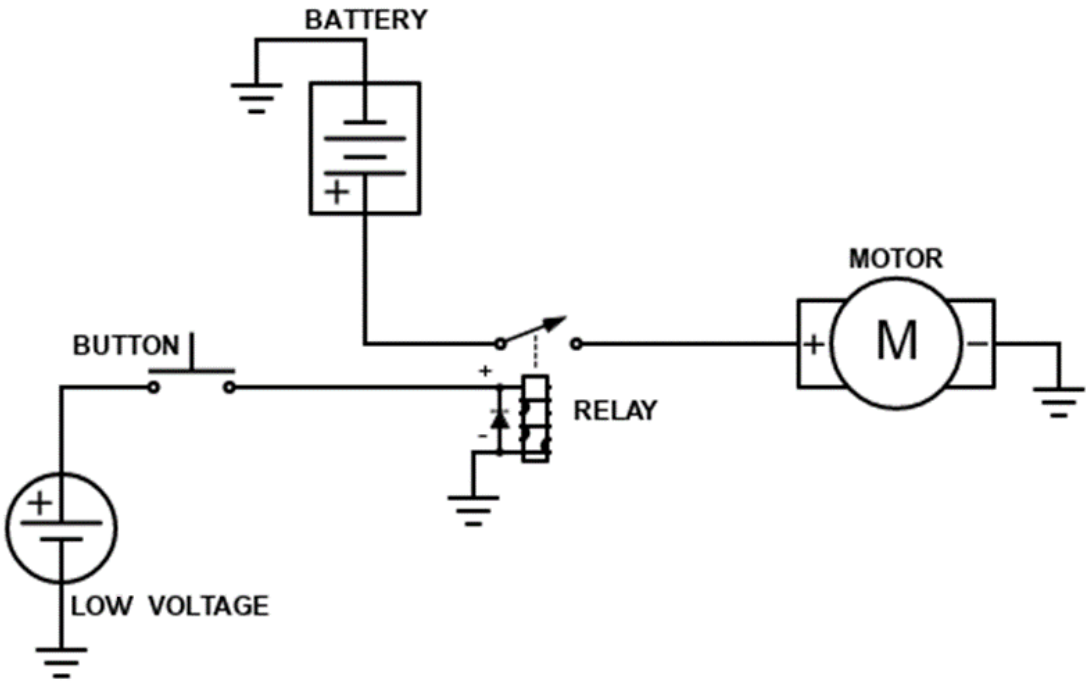

Figura 1. Esempio di un circuito elettromeccanico. Un pulsante comanda un relè, che attiva il motorino elettrico dell'alzacristalli.

Questo tipo di approccio è molto semplice e robusto. Dal momento che l'interruttore è connesso direttamente al componente da azionare è infatti impossibile che avvengano interruzioni del funzionamento, a meno che qualche componente non subisca danni irreparabili. Tutti i circuiti sono cablati dalla fabbrica, le funzioni degli interruttori sono decise dai progettisti e non possono essere riprogrammate in seguito.

Ad esempio, consideriamo un sistema di climatizzazione elettromeccanico. In un tipico sistema AC elettromeccanico è presente una manopola che controlla la velocità della ventola, una manopola per controllare la direzione del flusso e una manopola per la temperatura. La manopola per la velocità della ventola è un semplice interruttore circolare a più posizioni che, tramite dei relè, pilota la ventola a diverse tensioni, ottenendo diverse velocità. La manopola per variare la posizione dei flussi è un componente puramente meccanico in quanto è connessa tramite un albero a delle paratie che dirigono i flussi su diversi condotti. Ogni condotto porta l'aria verso destinazioni diverse (parabrezza, bocchettoni superiori ecc.). La manopola per controllare la temperatura controlla anch'essa una paratia in maniera meccanica, che permette di occludere parzialmente o totalmente l'afflusso di aria calda. Un sistema elettromeccanico simile a questo non offre automatismi.

1.2 Il progresso verso l'elettronica a microcontrollore

In un periodo in cui le automobili sono quotidianamente utilizzate in tutto il mondo, si manifesta la necessità di renderle sicure e pulite. Le emissioni di anidride carbonica hanno raggiunto livelli intollerabili e una buona percentuale deriva dagli scarichi delle automobili. Con l'aumento del traffico, questo problema rischia di peggiorare, in quanto sempre più veicoli attraversano le strade, inquinando l'atmosfera. Si manifesta la necessità di adottare scelte intelligenti per ridurre i consumi e le emissioni ed è evidente che l'elettronica convenzionale non dispone della flessibilità richiesta.

L'evoluzione elettronica ha dunque permesso di implementare circuiti a microcontrollore in cui il software gioca un ruolo fondamentale per ridurre al minimo i consumi. Successivamente, i circuiti a microcontrollore sono stati adottati anche per altri scopi, tra cui la sicurezza stradale e il comfort a bordo veicolo.

Capitolo 2

Elettronica a microcontrollore

Elemento fondamentale dell'elettronica per l'automobile è l'ECU, acronimo di *Electronic Control Unit*, unità di controllo elettronico. Si tratta di dispositivi elettronici a microcontrollore che gestiscono ciascuno una specifica funzione dell'automobile e sono utilizzati in un'architettura decentralizzata. Una moderna automobile possiede anche più di cento ECU diversi, ognuno posizionato strategicamente vicino al componente che controlla.

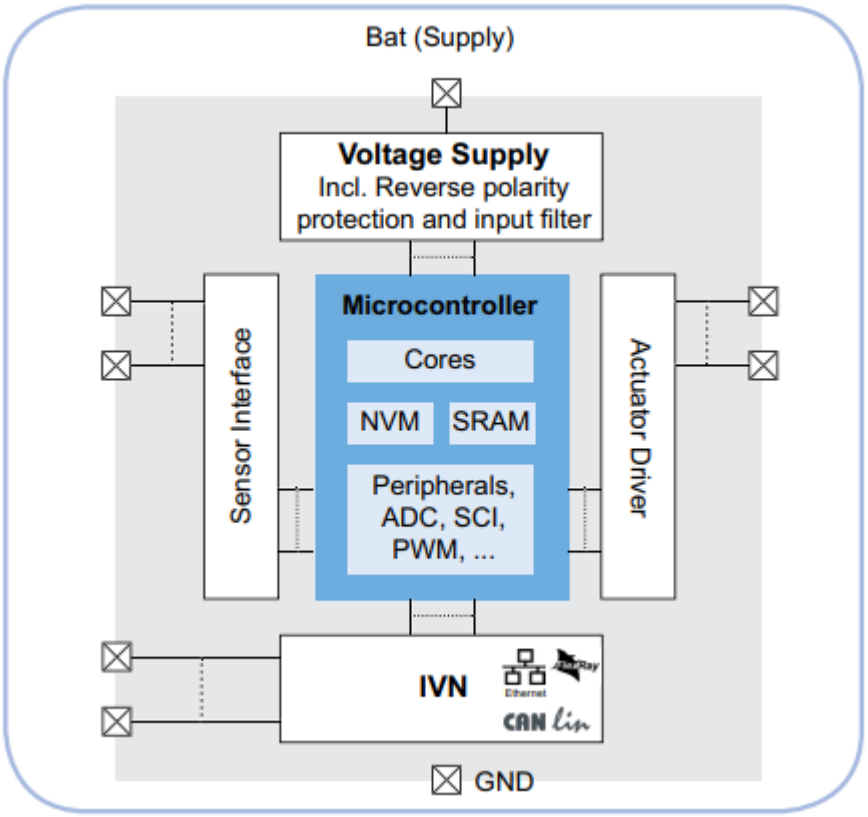

Figura 2. Modello di un generico ECU.
https://www.nxp.com/docs/en/white-paper/FVNECUA4WP.pdf

Lo schema di un ECU è visualizzato in figura 2. Il microcontrollore è il componente più importante, realizzato con un'architettura del tutto comparabile a un elaboratore. Sono presenti il processore e la memoria RAM, e tutte le periferiche necessarie. L'interfaccia dei Sensori costituisce l'input. I sensori forniscono dati relativi ai parametri in oggetto (temperature, posizioni, fasature ecc.) che verranno poi interpretati dal controllore. Il controllore comunica con la periferica da controllare tramite il Driver Attuatore. Infine, nelle versioni più recenti, i vari ECU sono interconnessi formando la cosiddetta In-Vehicle Network. Al giorno d'oggi esistono diversi protocolli, ognuno pensato per un utilizzo specifico.

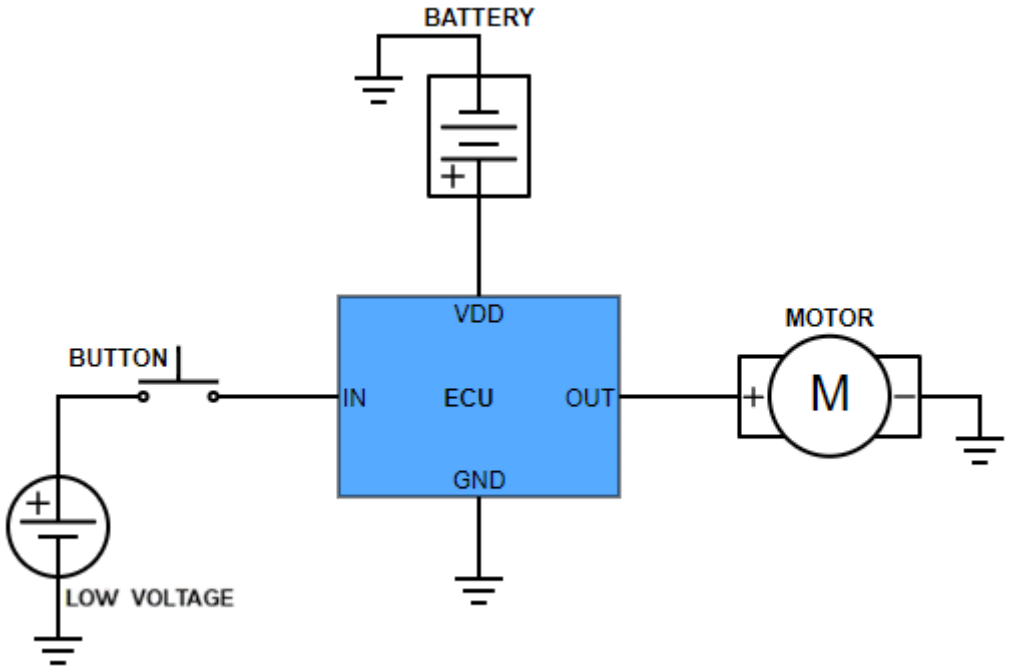

Figura 3. Esempio di un circuito con ECU. Un pulsante è input del controllore, il quale manda un impulso all'attuatore.

I controllori dispongono di un firmware che racchiude tutta la logica operativa che il componente deve seguire. Si tratta di un aspetto fondamentale perché tale software è quasi sempre aggiornabile dal produttore per aggiungere funzioni nuove o correggere eventuali problemi. In un processo industriale che punta alla standardizzazione delle parti, inoltre, è possibile produrre il medesimo hardware e personalizzarlo a seconda del mercato o del modello attraverso il software. I modelli di automobile più costosi avranno così più funzionalità dei modelli base, pur condividendo lo stesso hardware, e questo riduce notevolmente i costi di ingegnerizzazione.

I primi veicoli disponevano di pochi ECU, per le funzioni principali. Storicamente uno dei primi è stato l'*Engine Control Unit*, per controllare i parametri della combustione del motore.

2.1 Engine Control Unit

In un motore termico, la corretta combustione del carburante dovrebbe essere sempre garantita, al fine di ridurre i consumi e le emissioni. Il rapporto stechiometrico ideale fra aria e carburante, tuttavia, non è costante e varia a seconda delle condizioni atmosferiche in cui il motore sta operando. Una miscela troppo grassa, ovvero composta da una percentuale eccessiva di carburante, potrebbe danneggiare il motore e aumentare l'usura. Inoltre, una miscela grassa comporta un consumo di carburante maggiore e non sempre giustificato, in quanto non è garantito che tutta la quantità immessa venga combusta a pieno. Al contrario, una miscela magra diminuisce la lubrificazione delle parti in movimento. Si manifesta dunque la necessità di scegliere l'esatta quantità di carburante e aria da immettere nel motore.

Inizialmente questo compito era delegato al carburatore, un componente statico che assicurava il giusto rapporto stechiometrico. Il valore del rapporto può essere modificato, ma non è una pratica agevole: vengono molto spesso testati in fabbrica da personale esperto, che agisce su degli ugelli per aumentare o diminuire i flussi di carburante e aria attraverso il componente. Le regolazioni vengono provate su banco, facendo operare il motore a determinate andature e condizioni. Successivamente vengono analizzati i dati del banco, e vengono effettuate modifiche nel componente per ottenere i risultati cercati. Una volta calibrato, il carburatore non deve essere modificato dall'utilizzatore.

Il carburatore, dunque, non è pensato per adattarsi alle condizioni ambientali quali temperatura dell'aria e pressione, ma fornisce una regolazione valida in tutte le circostanze. In circostanze ambientali normali, e a seguito di una adeguata calibrazione, il carburatore è una valida soluzione per assicurare un sufficiente rapporto stechiometrico carburante-ossigeno, ma a temperature diverse in modo significativo dalle temperature di normale operatività, l'efficienza del motore diminuisce inevitabilmente.

Il controllo dell'acceleratore è puramente meccanico: la pressione del pedale agisce sull'apertura della valvola che controlla l'immissione del carburante nel carburatore.

Con questo sistema, puramente meccanico, la regolazione dei vari organi che costituivano il motore era fissa, e non erano possibili variazioni dipendenti dalle condizioni d'uso o dalle condizioni ambientali.

A partire dagli anni '70, in concomitanza con lo sviluppo tecnologico, alcuni produttori di auto cominciarono a dotare i propri veicoli di un dispositivo elettronico con lo scopo di porre rimedio a questa grossa limitazione. Il dispositivo in questione, chiamato Engine Control Unit (da non confondere con Electronic Control Unit), inizialmente controllava solo la quantità di carburante che gli iniettori dovevano rilasciare nel cilindro. L'ECU analizza i dati in ingresso derivanti dai vari sensori e stabilisce in tempo reale l'esatto rapporto stechiometrico. L'acceleratore non è più connesso meccanicamente alle valvole, ma è elettronico e composto da un sensore di posizione. Esso appare all'ECU come un sensore in input, e contribuisce all'output. L'ECU stabilisce quanto carburante erogare in base a delle tavole chiamate in gergo *mappe*, ovvero degli array multidimensionali. Infine, mediante degli attuatori, eroga le quantità calcolate. Successive evoluzioni hanno reso l'ECU un dispositivo molto complesso, in grado di controllare molti parametri della combustione, tra cui la fasatura delle valvole e dell'accensione.

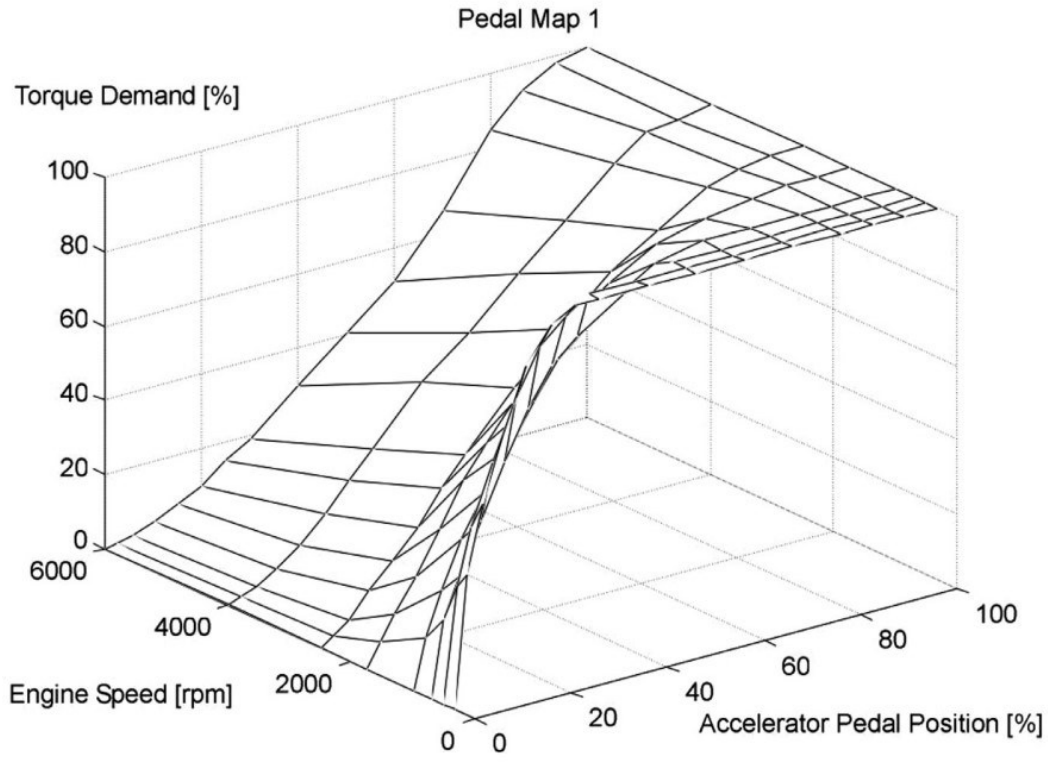

Figura 4. Una generica mappa a due variabili. Due input (rotazioni del motore e pressione dell'acceleratore) forniscono un output (coppia motrice)
https://www.sciencedirect.com/science/article/abs/pii/S1367578816300086

2.2 La decentralizzazione nelle prime architetture ECU

Mentre il progresso tecnologico continuò, vennero progettati sempre nuovi ECU per rendere elettrificate e intelligenti altre funzioni del veicolo. Questo progressivo ampliamento dell'utilizzo di ECU, però, si mosse verso una soluzione decentralizzata, in cui vennero man mano aggiunte funzionalità mediante circuiti separati, pensati appositamente per lo scopo. Questo rende la progettazione molto più semplice perché ogni ECU può funzionare indipendentemente dagli altri. La manutenzione è semplice: basta individuare il componente che causa un malfunzionamento e, data la separazione tra le varie funzioni, ciò è piuttosto agevole.

Con la crescita esponenziale del numero degli ECU si manifesta la necessità di creare una forma di rete di interconnessione. Molto spesso ECU diversi, ma legati da un simile scopo, hanno la necessità di comunicare reciprocamente il loro stato; ciò viene effettuato attraverso diversi standard di bus dati. Aggiungere un nuovo ECU al sistema è relativamente semplice: è necessario verificare che i bitrate delle interconnessioni siano sufficienti a supportare il bitrate massimo del nuovo controllore.

Dal momento che ogni ECU controlla una o poche funzioni, le singole unità di elaborazione possono disporre di ridotta potenza computazionale, e utilizzare nel caso più comune anche un solo core e poca memoria RAM.

I difetti che le architetture decentralizzate possiedono, però, limitano gli sviluppi futuri in campo tecnologico. Al crescere delle funzioni implementate via software, cresce la complessità e la mole delle interconnessioni. Tanti circuiti separati richiedono una mole di dati notevole sui bus, e gli stessi bus richiedono interconnessioni che rendono il circuito complicato. Inoltre, la presenza di connessioni fisiche tra i nodi limita lo spazio per il layout dell'architettura. Gli standard più utilizzati per l'interconnessione sono FlexRay, CAN (Controller Area Network), LIN (Local Interconnect Network) e MOST (Media Oriented System Transport). Questi protocolli sono stati sufficienti per i primi decenni, ma considerando la curva dell'aumento delle funzionalità software negli ultimi anni, è facile intuire che il volume di dati e delle interconnessioni fra ECU diventerà presto insostenibile.

I produttori di automobili devono garantire il rispetto delle norme di sicurezza del Paese in cui il veicolo verrà venduto. Molto spesso, Paesi diversi adottano norme diverse, e garantire una copertura totale richiede la personalizzazione del software per i vari mercati. Poiché le funzioni

interessate risiedono in controllori diversi, un sistema decentralizzato richiederebbe la personalizzazione di decine di ECU.

2.3 Progresso verso la centralizzazione

Fino ad ora, la complessità software degli ECU è stata sufficientemente bassa da rendere l'architettura decentralizzata lo standard nel settore automobilistico. I vantaggi in termini di costi superano gli svantaggi e ciò è stato un disincentivo alla ricerca di soluzioni migliori.

Il progresso tecnologico spinge a software più complessi e l'architettura decentralizzata si manifesta inadeguata. Alcuni produttori, tra cui Bosch, stanno cominciando la transizione verso architetture E/E (elettriche/elettroniche) centralizzate, mentre altri fanno ancora affidamento a soluzioni del passato. L'adozione di una nuova architettura è un investimento per un'azienda, compensata da una riduzione stimata fino al 20% del numero totale di controllori e fino al 10% del costo totale dei materiali.

I controllori non gestiscono più un unico componente, ma sono *cross-domain*, cioè operano su più zone del veicolo. Le architetture sono formate da pochi e potenti computer che gestiscono più componenti del veicolo. I controllori sono interconnessi da protocolli ad alta velocità. Ad oggi si è affermata Automotive Ethernet, che è in grado di supportare grandi quantitativi di dati e comunicazioni con requisiti di sicurezza. Non solo le comunicazioni interne sono più veloci ed efficienti: in una transizione verso veicoli interconnessi, un'architettura centralizzata sarebbe l'unica via per ottenere le prestazioni sperate.

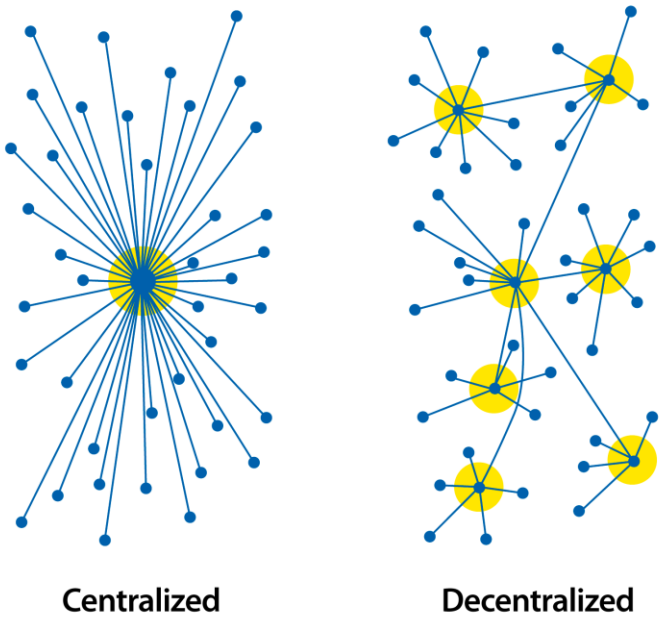

Figura 5. Differenza fra architettura centralizzata e decentralizzata https://www.designnews.com/electronics-test/centralized-or-decentralized-autonomous-vehicles-are-forcing-key-architectural

I pregi di un'architettura centralizzata sono esposti in seguito.

- Latenze ridotte. Avendo tutta la logica integrata in un unico componente, non è necessaria la comunicazione tra più ECU, rendendo le operazioni più veloci. Inoltre, i messaggi da trasmettere nel bus non devono contendersi le priorità fra diversi ECU in quanto questi messaggi sono stati trasmessi da un solo ECU.
- Cavi ridotti. L'unione di più funzioni in un unico controllore riduce il numero di interconnessioni fra gli ECU.
- Robustezza ai cambiamenti di design. Implementare una funzionalità via software è molto più semplice e non richiede grossi investimenti. Le aggiunte di sensori o attuatori non richiedono modifiche architetturali.

L'industria dell'aviazione è stata pioniera nel passaggio da architetture decentralizzate ad architetture centralizzate. Negli aeromobili sono già in utilizzo da decenni soluzioni centralizzate perché i benefici prestazionali sono notevoli, in quanto l'elettronica è preponderante nella sicurezza. Le latenze inferiori, il peso inferiore delle connessioni e i benefici architetturali premettono infatti agli aeromobili di operare in maggiore sicurezza.

Ci sono diverse realizzazioni pratiche possibili a partire da un'architettura centralizzata. Le più importanti sono architettura *domain-oriented*, *cross-domain* e *zone-oriented*.

2.3.1 Domain-Oriented

Vengono riconosciuti quattro domini fondamentali: abitacolo (luci interne e comfort), infotainment (display informativi e intrattenimento), sicurezza e movimento del veicolo (telaio e assistenze alla guida) e gruppo propulsore (motore e gas di scarico). Ad ogni dominio viene assegnato un cluster, composto da un *domain control unit* (DCU) e opzionali ECU di *sub-domain*. Gli eventuali ECU sono connessi soltanto al corrispettivo DCU. Lo scopo del DCU è offrire la maggior parte delle elaborazioni richieste dal dominio, e creare un livello di astrazione che semplifichi la comunicazione fra i vari DCU.

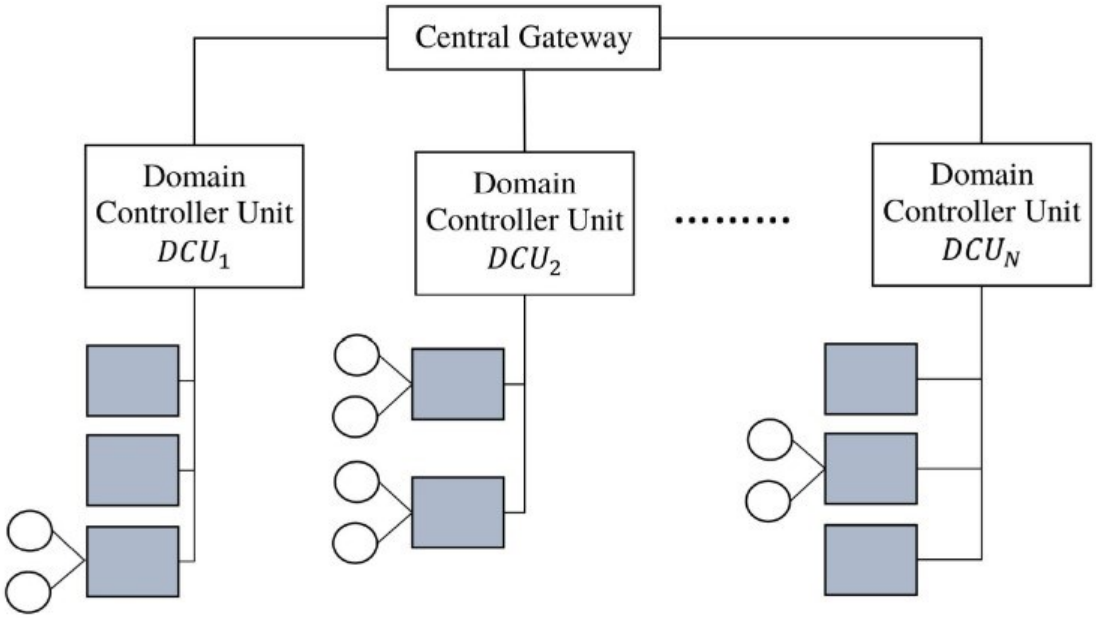

Figura 6. Schema di un'architettura Domain Oriented
https://ieeexplore.ieee.org/document/9337216

I DCU sono interconnessi da linee ad alta velocità, per via della grande quantità di dati che devono scambiare. Standard come Automotive Ethernet sono opportuni a garantire le prestazioni necessarie. Gli ECU di *sub-domain*, non dovendo generalmente scambiare grosse quantità di dati, sono connessi al DCU del proprio dominio tramite connessioni a minor bitrate come le tradizionali CAN, LIN o FlexRay, o Ethernet nel caso esse non dovessero essere sufficienti in termini prestazionali.

Tuttavia, questa architettura solleva due problemi. Con la conseguente implementazione di sempre più ADAS (Advanced Driver Assistance Systems), aumentano i domini interessati che devono scambiarsi i dati, e questo rende più difficile l'individuazione di un dominio come una zona del veicolo in cui risiede una particolare funzione. Inoltre, si sospetta che con l'aumentare delle connessioni fra DCU, i sistemi di connessione non saranno più sufficienti.

2.3.2 Cross-Domain Oriented

Le funzioni di più domini sono implementate in un unico ECU chiamato *cross-domain control unit* (CDCU). Questo permette di raggruppare più domini in un unico ECU.

2.3.3 Zone Oriented

La vettura è divisa in zone e ogni zona ha un *zone control unit* (ZCU). Questo potrebbe sembrare simile all'architettura domain-oriented, con la differenza che le zone non sono divise per funzione ma per posizione nel veicolo. Inoltre, le ZCU non prendono parte al calcolo computazionale, ma hanno lo scopo di raccogliere i dati dei sensori e trasmetterli a server centrali molto potenti, i quali eseguiranno poi le operazioni.

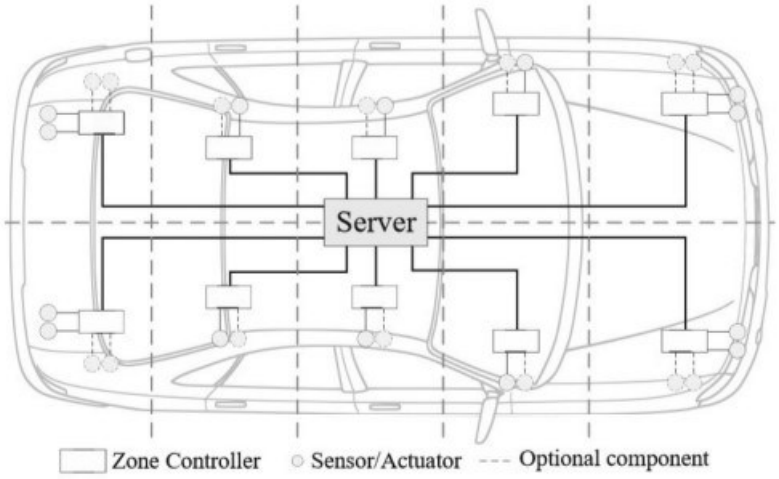

Figura 7. Schema di un'architettura Zone Oriented
https://ieeexplore.ieee.org/document/9337216

Questa architettura potrebbe inoltre offrire, in congiunta con le reti 5G ad elevata velocità, un server completamente cloud, che elabora i dati senza la necessità di essere fisicamente presente nel veicolo.

Tuttavia, un server completamente cloud non è stato ancora usato in campo pratico, in quanto non è garantita l'operatività in ogni circostanza. La rete 5G non è sufficientemente estesa per garantire un servizio sicuro, e in caso di mancanza di segnale non è definito il comportamento del sistema E/E del veicolo.

2.4 AUTOSAR Classic

La centralizzazione di un'architettura è un tema complesso e, senza alcuno standard su cui basarsi, il processo di integrazione rischia di diventare troppo frammentato. L'utilizzo di componenti provenienti da diversi produttori richiede uniformità nell'architettura software. Al giorno d'oggi si è affermato AUTOSAR, acronimo di *AUTomotive Open System ARchitecture*. AUTOSAR offre un framework di sviluppo per sistemi embedded ad uso automobilistico. Sviluppato da un consorzio di produttori di veicoli ed elettronica, ha come scopo quello di definire uno standard per facilitare la portabilità e la modularità del software.

La denominazione *classic* è subentrata dopo l'anno 2017, quando è stata proposta l'evoluzione diretta del framework, chiamata *Adaptive*.

L'architettura di AUTOSAR prevede 3 layer: *Application Layer*, *Runtime Environment*, e *Basic Software*.

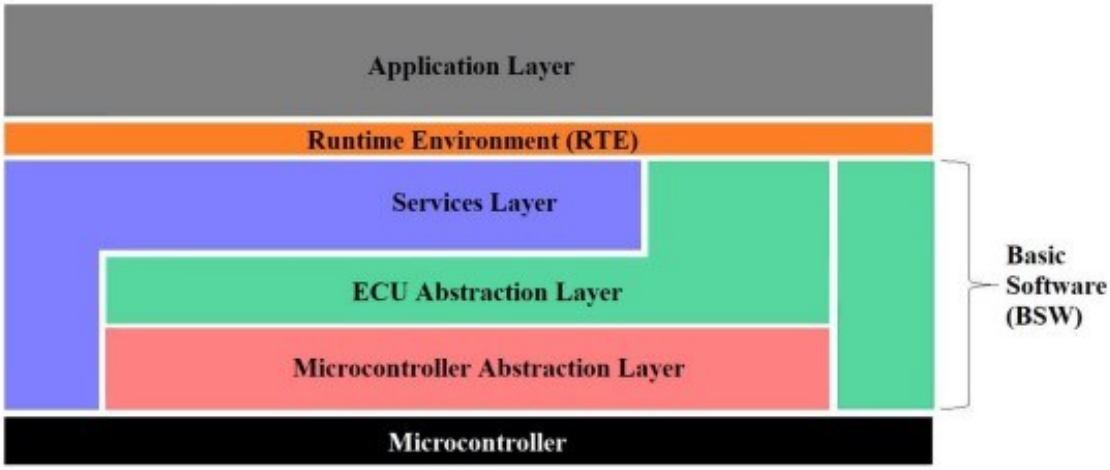

Figura 8. Layer dell'architettura AUTOSAR
https://ieeexplore.ieee.org/document/9337216

L'Application Layer è il livello più alto, che implementa le funzionalità di un ECU sotto forma di componenti software indipendenti dall'hardware. Il RTE è il mezzo di comunicazione dell'architettura, fornisce un gateway con cui il software accede alle funzionalità dell'ECU. Il BSW fornisce, attraverso diversi layer di astrazione sempre di più basso livello, il collegamento fra hardware e RTE.

L'architettura a strati permette di supportare la centralizzazione, separando i componenti software dall'hardware.

Dal punto di vista della sicurezza, AUTOSAR prevede dei sistemi che facilitano l'individuazione e la risoluzione di eventuali errori, come richiesto dallo standard ISO 26262. La sicurezza in hardware viene garantita mediante dei controlli nel core e nella RAM. I controlli nel core permettono di rilevare errori nelle elaborazioni della CPU che risulterebbero in output sfalsati. I controlli nella RAM permettono di rilevare errori nella memorizzazione dei dati nella memoria volatile.

2.5 AUTOSAR Adaptive

Nato nel 2017, è la diretta evoluzione di AUTOSAR Classic, pur non rimpiazzandolo. È pensato per i sistemi ad alta prestazione e dalla bassa criticità, cioè con connessioni dall'elevato bitrate. Gli standard Classic e Adaptive supportano la coesione in un unico sistema automotive. AUTOSAR Classic rimane un valido protocollo per i domini legati al telaio e al gruppo propulsore. La versione Adaptive viene invece utilizzata sui domini che richiedono maggiori bitrate come l'infotainment e la connettività. Anche nelle applicazioni per implementare la guida autonoma si utilizza la versione Adaptive, perché garantisce prestazioni migliori.

Capitolo 3

Evoluzione dal single-core al multicore

I controllori single core diventano sempre meno adatti a gestire la mole di operazioni. Attualmente vengono utilizzati controllori multicore dall'elevata potenza. Infineon ha iniziato a sperimentare processori dotati di 3 core già dal 1999, e ad oggi è arrivata alla sesta iterazione.

Queste architetture supportano il parallelismo, che è necessario per soddisfare i requisiti dello standard ISO 26262.

Realizzare un'architettura multicore partendo dalle fondamenta sarebbe troppo costoso e richiederebbe troppo tempo. È decisamente vantaggioso studiare delle procedure per migrare da un'architettura single-core a una multicore. Durante il processo di migrazione è necessario tenere in considerazione alcuni problemi che non compaiono negli ECU single-core: protezione dei dati, bilanciamento del carico e ottimizzazione dei task.

3.1 Protezione dei dati

La condivisione dei dati avviene quando un dato, memorizzato da un core in una cella di memoria, viene utilizzato da un altro core. I processori multicore sfruttano il parallelismo e questo può rendere alcune operazioni "unsafe" ovvero non sicure.

Consideriamo un semplice esempio: un dato viene salvato da CPU0 su una cella di memoria, pronto per essere utilizzato. CPU1 sovrascrive il dato prima che CPU0 lo abbia utilizzato. CPU0 utilizza il dato modificato da CPU1, causando inevitabili errori logici. Una simile vulnerabilità non è permessa, perché gli ECU devono anche gestire importanti funzioni di sicurezza.

Una soluzione tradizionale a un simile problema avviene mediante l'utilizzo di un semaforo. Si tratta di un elemento di programmazione che permette di sincronizzare diversi flussi di codice concorrenti, cioè eseguiti simultaneamente ma in diversi core. Per poter realizzare un semaforo, è necessario scrivere del codice di protezione, il cui scopo è di gestire la sincronizzazione dei flussi. Una soluzione del genere trova difficile applicazione in un ECU, dove il numero di dati condivisi è notevolmente grande. Al crescere dei dati condivisi, il codice di protezione cresce esponenzialmente, rendendo la soluzione impraticabile.

Gli ECU basati su architettura AUTOSAR dispongono di un sistema di gestione dei dati attraverso runnable impliciti. I runnable hanno accesso a un buffer protetto, che viene copiato dalla RAM prima della loro esecuzione. I runnable possono modificare solo le variabili all'interno del buffer e, una volta terminato il codice, copiano il contenuto del buffer nella RAM. Questo garantisce la protezione dei dati, in quanto il runnable utilizzerà sempre un set coerente di variabili.

Tuttavia, si instaura un nuovo problema: la race condition. Una race condition avviene quando due processi tentano di accedere alla stessa risorsa, la quale non può essere condivisa contemporaneamente. Se non gestita correttamente può portare al deadlock, o stallo, in cui i processi attendono indefinitamente che la risorsa diventi disponibile. I problemi di race condition possono essere risolti attraverso dei metodi di protezione, che devono essere particolarmente ottimizzati in quanto i dati condivisi sono molti.

Tra i vari metodi di gestione della coerenza, ne è necessario uno che supporti lettura e scrittura, e che non abbia impatti sulla memoria, dato che la moltitudine di dati condivisi rischierebbe di saturarla. Il metodo lock-key, basato sulla tecnica dello spinlock, è una valida soluzione al problema della coerenza dei dati. Il metodo modifica il nome di un dato condiviso, assegnandone uno univoco per il proprio task. Una volta creata una copia locale della variabile, comincia la parte critica del task, ovvero la manipolazione dei dati, che è sicura perché protetta dallo spinlock e dalla disabilitazione degli interrupt. Alla fine della sezione di copia critica, va inserita la funzione di rilascio dello spinlock, e gli interrupt riabilitati.

3.2 Bilanciamento del carico

Il bilanciamento del carico è un ulteriore problema da ottimizzare in un'architettura multicore. In un sistema ideale, ai diversi core vengono attribuiti i task in maniera equa, in modo tale che vengano eseguiti nel minor tempo possibile e che non se ne accumulino mai. Inoltre, permette ai core di eseguire sempre delle operazioni evitando stati di idle prolungati, che andrebbero a intaccare le prestazioni generali del componente.

Per poter definire una strategia di ottimizzazione è utile suddividere i task in tre tipologie: periodici, non periodici e irregolari. I task periodici sono invocati a fissi intervalli di tempo, mentre i non periodici sono invocati con frequenza sconosciuta. I task irregolari sono simili ai task non periodici, in quanto non hanno un periodo fissato, ma dipendente dalle rotazioni del motore (RPM).

È necessario inoltre considerare il carico aggiuntivo dovuto alla gestione della protezione dei dati, in quanto per grandi quantità di dati condivisi, rende elevata la differenza.

Un valido protocollo per ottenere l'ottimizzazione cercata è definito Load Balancing of Migration.

1) Viene analizzata la velocità di ogni core, espressa in MHz. In generale ogni core può disporre di velocità diverse. Inoltre, è importante conoscere il numero di core per considerare i casi in cui i task debbano essere allocati a specifici core.
2) Vengono integrate le informazioni dei task irregolari, periodici e non periodici. Per semplificazione si assume che i task irregolari siano periodici.
3) Vengono analizzate le operazioni di lettura/scrittura dei task. Per questa fase è necessario assumere che le operazioni di lettura/lettura non siano considerate condivise, poiché non possono avvenire errori logici.
4) Vengono distinti task di allocazione e task non periodici.
5) Viene calcolato il tempo di esecuzione totale del task.
6) Viene considerato il caso migliore ottenuto.

3.3 Ottimizzazione dei task tramite Offset Allocation

L'allocazione ritardata è una tecnica per migliorare le prestazioni dello scheduling dei task, e si basa sul ritardare l'inizio di un task.

Lo standard AUTOSAR prevede un algoritmo che controlla periodicamente la percentuale di CPU utilizzata dai task in un determinato slot temporale prefissato. All'arrivo di un nuovo task non periodico, l'algoritmo deve valutare la presenza di slot liberi, e successivamente assegnare il task. Tuttavia, questa operazione potrebbe non avere riscontri favorevoli, perché i task periodici potrebbero occupare molta percentuale di CPU per ogni slot.

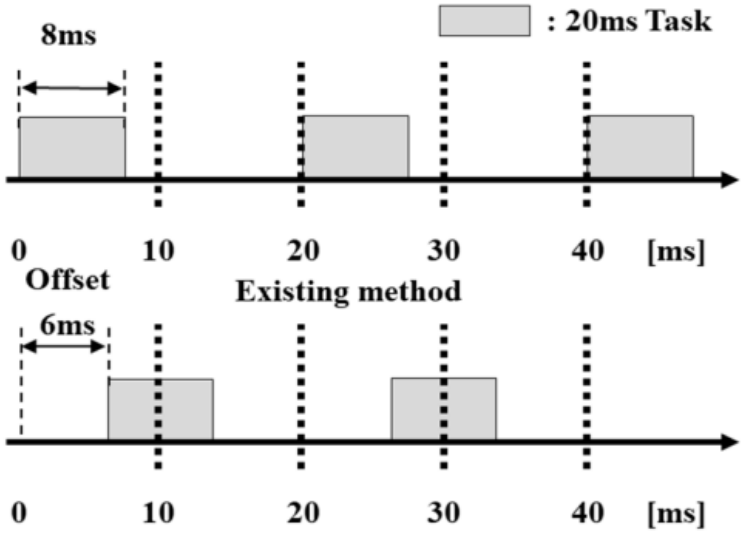

Figura 9. Esempio di Offset Allocation
https://ieeexplore.ieee.org/document/9398676

La soluzione è di ritardare ogni task periodico della giusta quantità di cicli CPU, tale che il task venga eseguito esattamente fra due slot temporali. In questa situazione il tempo di carico della CPU risulta minore, e possono essere inseriti task imminenti. I task periodici rimangono tali, poiché vengono soltanto posticipati di un'unità di tempo fissa. Con questo accorgimento, invece, i task non periodici hanno un tempo di attesa generalmente minore.

3.4 La simulazione dei chip

Una volta progettato il sistema E/E di un veicolo, deve essere popolato di software. Alcuni domini, come ad esempio il dominio infotainment, richiedono pochi raffinamenti, in quanto operano in maniera simile su tutti i veicoli. Non si può dire lo stesso del gruppo propulsore, che richiede fini calibrazioni ai parametri del motore. Di uno stesso modello di veicolo, vengono generalmente prodotte diverse versioni, aventi tecnologie diverse per quanto riguarda il cambio e il motore. Per ogni versione andrebbe creato un software a parte, che tiene conto delle caratteristiche tecniche del gruppo propulsore. Questo procedimento richiede molto tempo, perché è necessario seguire un protocollo molto dispendioso.

1) Preparare un software per gli ECU;
2) Caricare il software sul veicolo;
3) Testare il comportamento del software nel banco motore o in situazioni pratiche;
4) Analizzare i dati dei sensori.

Se i dati raccolti dai sensori non sono ottimali o si discostano da quelli teorizzati, è necessario analizzare la causa, ingegnerizzare una soluzione e reiterare il processo.

Questo protocollo è indispensabile per calibrare correttamente tutti i parametri dell'ECU, tuttavia ha numerosi svantaggi. Un test sul banco richiede molto tempo: il processo avviene in tempo reale e può richiedere diverse ore. Generalmente si vanno a verificare tutte le condizioni operative per garantire un funzionamento ottimale in ogni circostanza, e questo rende laborioso il processo. Il protocollo è inoltre costoso e servono investimenti dalla casa produttrice. È necessario coprire i costi di manutenzione del banco, carburante e operai specializzati.

Questo processo è da considerarsi *trial and error*, e generalmente richiede almeno un'iterazione, il che significa che raramente si ottengono i risultati sperati direttamente al primo test. Il buon esito in una (o poche) iterazioni dipende esclusivamente dall'esperienza e dalle conoscenze tecniche del personale incaricato.

Per sopperire questi punti sfavorevoli, al giorno d'oggi si effettua la calibrazione tramite simulazioni al computer. Esistono alcuni software specializzati nella simulazione di ECU, che vengono utilizzati per velocizzare le operazioni di test su banco. Le simulazioni offrono i risultati in tempi brevi, e di conseguenza la modifica di alcuni parametri ha riscontri quasi immediati.

Figura 10. Silver, un software per simulare gli ECU
https://cf-images.us-east-1.prod.boltdns.net/v1/static/5748441669001/11c03685-e914-44e0-a270-9113b13bc932/7b90282a-dcd0-456f-b768-b25963a39547/1920x1080/match/image.jpg

Inoltre, è possibile ricercare automaticamente i valori ottimi dei parametri, sempre in tempo quasi reale. In questa fase il simulatore esegue automaticamente le modifiche ai parametri usando intelligenti algoritmi di predizione e determina la configurazione ottima, cioè la migliore possibile. Questo punto è molto importante, perché non è sempre garantito che con il metodo manuale si ottenga un risultato di tale accuratezza. Generalmente, con il metodo manuale, si stabilisce una certa soglia oltre la quale si considera il risultato accettabile, ma difficilmente si ottiene il risultato ottimo, cosa estremamente facile con le simulazioni.

Capitolo 4

Standard di comunicazione

In questa sezione verranno analizzati i più importanti protocolli di trasmissione dei dati tra ECU.

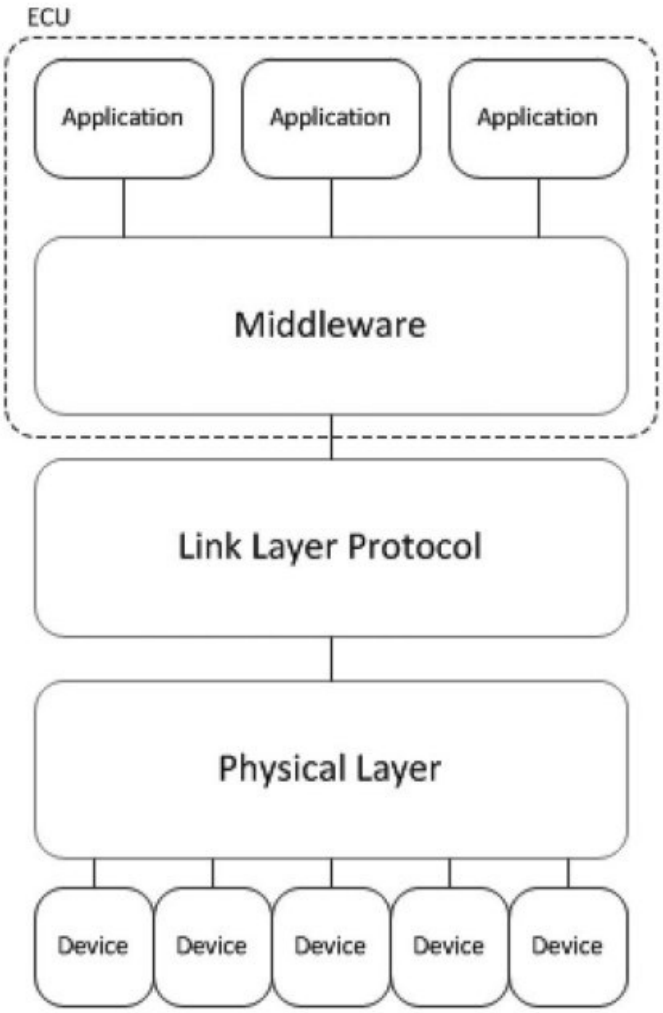

Figura 11. Hardware stack di un'architettura automotive.

Per decine di anni sono stati utilizzati protocolli come Controller Area Network (CAN, FlexRay, Local Interconnect Network (LIN), Media Oriented Systems Transport (MOST), Low Voltage Differential Signaling (LVDS) e Firewire. Questi protocolli sono basati su tecnologie diverse, e pertanto offrono bitrate diversi.

Protocol	Bitrate	Medium	Protocol
LIN	19.2 Kbps	Single Wire	Serial
CAN	1 Mbps	Twisted Pair	CSMA/CR
FlexRay	20 Mbps	Twisted Pair/Optical Fibre	TDMA
MOST	150 Mbps	Optical Fibre	TDMA
LVDS	655 Mbps	Twisted Pair	Serial/Parallel

4.1 LIN

LIN è un protocollo seriale sviluppato alla fine degli anni ’90. Utilizza un singolo cavo, e ciò rende il cablaggio semplice ed economico. Nella trasmissione di un dato si identifica un master, che instaura una comunicazione tramite un header. Lo slave riconosce l’identificativo e inizia a trasmettere il messaggio di risposta, composto da 1 byte fino ad un massimo di 8 byte e un ulteriore byte per effettuare il checksum.

4.2 CAN

CAN è stato sviluppato da Bosch all’inizio del 1980 e rilasciato nel 1986. Utilizza una coppia di cavi intrecciati per una velocità di trasmissione massima pari a 1 Mb/s fino a 40 metri. La comunicazione è asincrona, e i messaggi sono incapsulati in un frame di massimo 64 bit. Al giorno d’oggi è molto utilizzato soprattutto per le operazioni di diagnostica del motore. Nel tentativo di aumentare il bitrate è stato introdotto CAN FD, dove FD sta per Flexible Data-Rate. Utilizza principalmente due tecniche per aumentare la quantità di dati: aumenta il bitrate del payload o aumenta il numero di byte del payload.

4.3 FlexRay

FlexRay è nato nel 2006 da un consorzio, il cui obiettivo era creare un protocollo più veloce e affidabile di CAN. Il consorzio è stato abbandonato nel 2009, tuttavia alcuni veicoli successivi hanno comunque adottato il protocollo. Offre latenza e jitter costanti, grazie alla sincronizzazione a clock.

4.4 MOST

MOST è stato sviluppato principalmente come protocollo multimediale, in quanto flussi video possono richiedere elevati bitrate. Lo standard più performante, MOST150, offre 150 Mb/s, ed è più indicato di CAN nei contenuti multimediali.

4.5 LVDS

LVDS non è stato progettato specificatamente per il settore automotive; tuttavia, l'elevato bitrate (655 Mb/s) lo ha reso appetibile nel progressivo aumento della quantità di dati da scambiare.

4.6 Automotive Ethernet

Ethernet è un bus di comunicazione molto utilizzato nel modo di internet, per il basso costo, l'elevato bitrate e la flessibilità. Le vecchie tecnologie come CAN e MOST sono state pensate specificatamente per l'utilizzo nel settore automotive. Sono perciò avvantaggiate dal fatto che offrono soluzioni su misura pensando alla comunicazione a bordo veicolo. Al momento del lancio offrivano prestazioni eccellenti; i bitrate offerti erano sufficienti a supportare le applicazioni per cui erano destinati. Attualmente ciò non è più valido.

Ethernet ha già soppiantato CAN per l'interfaccia con strumenti di diagnostica del veicolo. Un aggiornamento del firmware di ridotta portata occupava anche più di dieci ore con l'interfaccia CAN, ora con Ethernet richiede pochi minuti.

Capitolo 6

Sicurezza

Nella maggior parte delle comunicazioni interne è necessario un certo livello di sicurezza end-to-end. Tutte le informazioni trasmesse devono essere visibili solo ai destinatari interessati, e i componenti non autorizzati non devono prendere parte alla comunicazione. Le attuali tecniche permettono segretezza, prevenzione alla manipolazione e autenticazione.

6.1 Sicurezza elettronica e informatica a livello architetturale

Prima dell'avvento dell'elettronica digitale nei veicoli, i sistemi di sicurezza erano piuttosto deboli e consistevano principalmente in dispositivi meccanici, come il blocco dello sterzo e la chiusura delle porte con la chiave. Alcuni veicoli più costosi disponevano di un basilare allarme; in ogni caso questi dispositivi proteggevano il veicolo da attacchi fisici come furti o manomissioni. Con l'avvenire di un'elettronica sempre più integrata in tutte le funzioni del veicolo, si sono evoluti anche i sistemi di sicurezza, che si trasformarono in complessi dispositivi corredati da software. Questi dispositivi, però, sono più vulnerabili a potenziali attacchi, con conseguenti implicazioni per la sicurezza del guidatore e dei passeggeri. I sistemi di sicurezza meccanici, inoltre, erano vulnerabili solo ad attacchi fisici, in cui si presuppone che l'attaccante entri a contatto diretto con il mezzo. Con le connettività wireless delle automobili di ultima generazione, si introducono involontariamente nuove tecniche di attacco che non richiedono il contatto fisico dell'attaccante e dunque possono essere "silenti".

Attaccante	Difficoltà	Accesso	Obiettivo
Furto	varia	wireless/fisico	furto
Hacker	medio – alta	wireless	notorietà
Criminale	media – molto alta	wireless/fisico	nuocere ai passeggeri
Tuner	media – molto alta	fisico	modificare i parametri
Competitor (sett. automobilistico)	alta – molto alta	fisico	studiare l'architettura

6.1.1 Controller Authentication

Ogni controllore necessita di un certificato di autentificazione, che ne determina la validità come mittente. I mittenti di ogni messaggio necessitano di tale autenticazione per accertare che solo i controllori validi possano comunicare con il sistema. I messaggi non autorizzati possono essere separati e processati con le dovute accortezze, o semplicemente scartati. Un certificato è composto da tre campi: ID, PK e Auth. Il campo ID contiene l'identificativo del controllore, e deve essere una stringa univoca. Il campo PK contiene la chiave privata, e Auth le autorizzazioni del componente. Il gateway, cioè il componente che intercetta i messaggi nel bus dati, contiene una lista di chiavi pubbliche PK_{OEM} di tutti i dispositivi accreditati per il rispettivo veicolo. Il certificato di un dispositivo viene firmato dal produttore con una chiave segreta SK_{OEM}. Il gateway utilizza queste informazioni per verificare l'integrità del certificato, e a processo concluso aggiunge il dispositivo alla lista dei controllori verificati.

6.1.2 Comunicazione criptata

La crittazione di tutte le trasmissioni di dati è fondamentale per garantire la sicurezza. Esistono due tecniche principali di cifratura: crittografia simmetrica e asimmetrica. Date le limitazioni in termini di potenza, capacità e timing, è necessario utilizzare entrambi le tecniche a seconda dell'applicazione.

In seguito verranno analizzate nel dettaglio.

Crittografia asimmetrica

Si tratta di un sistema a coppia di chiavi, cioè sono richieste due chiavi per completare il ciclo di trasmissione dell'informazione. Una chiave è definita pubblica, ed è la chiave che viene utilizzata per cifrare il messaggio da trasmettere. Il messaggio in questo stadio è cifrato, dunque appare illeggibile a qualsiasi tentativo di attacco. Alla chiave pubblica è associata una privata, che solo il titolare adibito alla ricezione del messaggio è tenuto a conoscere. Tale chiave deve essere tenuta rigorosamente segreta, poiché permette la decifrazione del messaggio. La chiave pubblica, invece, ha tale nome perché può essere tranquillamente divulgata senza compromettere la sicurezza del sistema, dal momento che essa dipende dalla segretezza della chiave privata.

Questo tipo di crittografia utilizza algoritmi matematici complessi e che non ammettono soluzioni efficienti. L'elevato peso computazionale rende questi algoritmi particolarmente indicati per piccoli messaggi. Inoltre, non è definito un canale sicuro per la trasmissione della chiave privata al ricevitore del messaggio.

Crittografia simmetrica

Questo sistema utilizza una sola chiave, che per ovvi motivi di segretezza deve essere privata. La chiave di crittazione e la chiave di decrittazione sono identiche. È molto più semplice da implementare della crittografia asimmetrica, e offre prestazioni migliori. Anche in questo caso la chiave non può essere trasmessa in modo sicuro, e deve essere conosciuta da entrambi le parti prima dell'invio del messaggio.

	Asimmetrica	**Simmetrica**
Prestazioni	basse	alte
Sicurezza	molto alta	alta

Nei sistemi automotive, così come in altre applicazioni, si utilizza una combinazione delle due tecniche. La crittografia asimmetrica viene impiegata per la distribuzione sicura della chiave privata. Tale chiave viene impiegata per la decrittazione dei veri e propri messaggi, codificati con una più prestante crittografia simmetrica. La chiave privata viene periodicamente modificata, come ulteriore misura precauzionale.

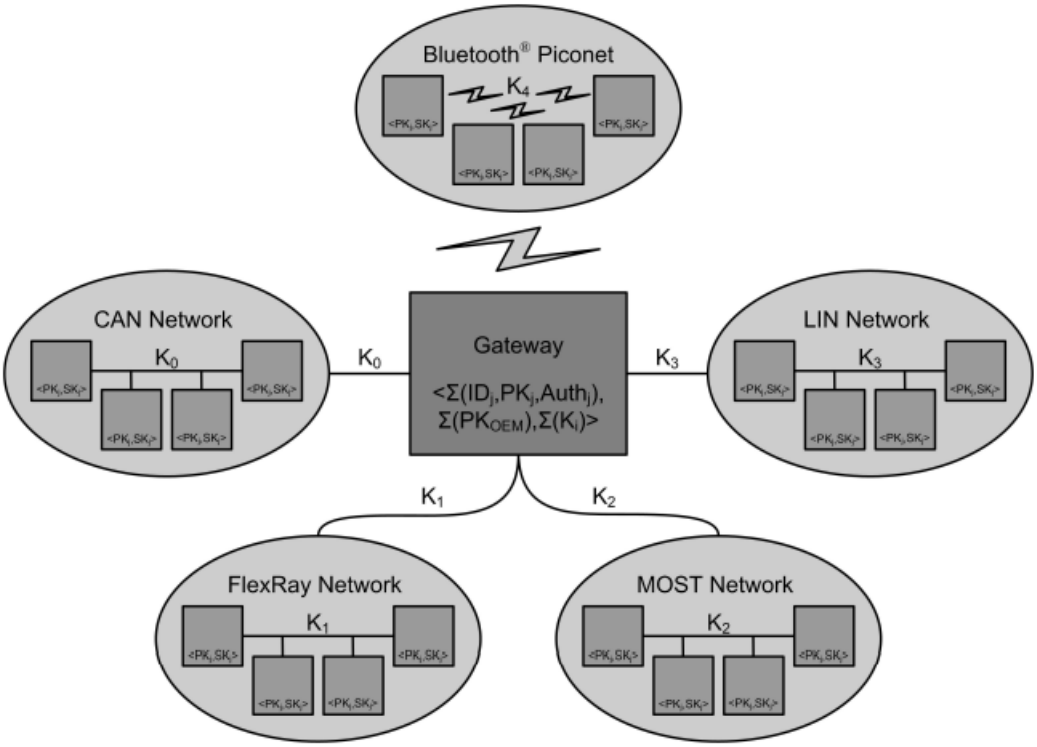

Figura 12. Criptazione e gateway
https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.728&rep=rep1&type=pdf

Nell'esempio, un gateway centrale interconnette i protocolli di bus fra loro. Il gateway è di conseguenza un tramite obbligatorio nel caso si debbano instaurare comunicazioni che interessano due bus diversi. Inoltre, il gateway possiede un'area di memoria protetta, che contiene le chiavi private e una lista dei controllori validi con le rispettive autorizzazioni Auth. Ogni controllore di un bus possiede la chiave simmetrica di gruppo K_i e la coppia asimmetrica PK_j e SK_j. Inoltre possiede la chiave pubblica del gateway PK_G.

6.1.3 Gateway Firewall

L'ultimo livello di sicurezza di un gateway viene realizzato con un firewall. Si tratta di un componente software (può anche essere accompagnato da hardware dedicato) che consente la protezione informatica di una rete.

Se i controller implementano le firme digitali o gli indirizzi MAC, il firewall è basato sulle autorizzazioni fornite nei certificati di ogni controller. Il firewall intercetta ogni richiesta e permette solo ai controllori accreditati di comunicare fra i bus.

Infine, le funzioni di diagnostica del veicolo, nonché le interfacce di diagnostica, che sono fondamentali in fase di produzione e revisione, devono essere filtrati dal firewall durante l'utilizzo finale.

6.2 Batteria

Ogni veicolo è dotato di una batteria, che serve per fornire l'energia necessaria ad accendere il motore. Queste batterie sono generalmente di piccole dimensioni, commisurate al carico che il veicolo richiede. I veicoli elettrici invece richiedono batterie agli ioni di litio di grosse capacità, in alcuni casi anche superiori a 100 kW/h. Queste batterie sono molto delicate: una foratura potrebbe cortocircuitare gli strati e causare un rapido incendio. Anche la ricarica impropria è possibile causa di incendio, pertanto ogni batteria è dotata di un circuito di protezione chiamato Battery Management System, che regola le tensioni e le correnti di carica. Il BSM monitora la tensione, la temperatura e il flusso di corrente dalla batteria e verso la batteria, ed è responsabile delle strategie di carica per prevenire la prematura usura delle celle.

Controllare i parametri del BMS potrebbe garantire a un attaccante il pieno controllo della batteria, tra cui disabilitare le funzioni di protezione. In questo modo possono essere favorite condizioni elettrodinamiche pericolose alla chimica della batteria e, nei casi gravi, persino l'autoignizione di una o più celle. Per questo motivo le batterie di recente costruzione sono dotate di valvole di sfogo che consentono a un'eventuale pressione accumulata di essere rilasciata al di fuori delle celle. I BMS più avanzati hanno una tecnologia che implementa dei sistemi di controllo hardware che sconnettono i contatti delle batterie in caso di tensioni o temperature al di fuori dai limiti sicuri.

6.3 Il sistema di ricarica come punto di intrusione

A differenza dei veicoli tradizionali, quelli elettrici necessitano di una porta per la ricarica delle batterie. I primi modelli offrivano un semplice porta elettrica; successivamente si sono affermati numerosi standard che implementano, oltre all'erogazione della corrente, anche protocolli di scambio dati tra BMS e la torretta di ricarica. Alcuni standard, come il giapponese CHAdeMO, utilizzano il bus CAN per la trasmissione dei dati. Purtroppo, alcuni gateway CAN-CAN utilizzati oggi non offrono il filtro dei messaggi. Questo potrebbe permettere a un hacker svariate azioni criminose, tra cui riprogrammare una chiave, o addirittura gli ECU. I sistemi di ricarica del futuro prevedono aggiornamenti firmware e servizi aggiuntivi come lo streaming multimediale. Queste feature rendono ancora maggiori le possibilità teoriche di attacco.

Queste tipologie di criticità possono essere sfruttate da un malvivente con la tecnica del man-in-the-middle. La tecnica consiste nell'interporre una presa modificata fra la presa di ricarica e il connettore della torretta, all'insaputa dell'utente. A una rapida occhiata, la presa appare perfettamente autentica, ma in realtà l'utente interagisce con il componente modificato. Tali componenti possono essere programmati in molti modi, con lo scopo principale di alterare o creare pacchetti dati da trasmettere al bus.

6.4 Drive by wire

Questa funzionalità non è esclusiva alle auto elettriche, ma si sta diffondendo anche alle auto a propulsione tradizionale. I comandi di guida, come acceleratore, freno e sterzo non sono meccanicamente connessi al componente, ma sono dei potenziometri che pilotano degli attuatori. Nonostante ciò porti dei benefici, introduce anche vulnerabilità potenzialmente pericolose per la salute dei passeggeri.

Conclusioni

Il settore automobilistico è da sempre molto proficuo, e di conseguenza rimane costantemente al passo con lo sviluppo tecnologico. In passato sono state regolarmente aggiunte innovazioni e ci sarà da aspettarsi altrettanto nel futuro.

L'obiettivo futuro è la creazione di un ampio ecosistema IoT in cui veicoli e infrastrutture stradali possono scambiare reciprocamente informazioni. Per supportare una simile impresa sono necessarie reti mobili a bassa latenza ed elevato bitrate. Le prestazioni delle reti 5G sono promettenti e dovrebbero riuscire gestire un ecosistema simile; tuttavia, non si dispone ancora di una copertura territoriale sufficiente, e ciò rende impossibile una realizzazione pratica.

Una rete così formata potrebbe anche fornire una base per un sistema affidabile di guida autonoma. Le auto potrebbero scambiare le posizioni reciproche, rendendo il software di guida autonoma consapevole di tutti i pericoli nei dintorni.

Come ogni nuova tecnologia che possiede più automazioni della precedente, si corre il rischio di introdurre possibili vulnerabilità. In questo caso si tratterà di guida autonoma, di conseguenza nessuna vulnerabilità andrà tollerata.

Le tecnologie sono nuove e ci sono numerosi scogli da superare, ma sicuramente è dove il settore automobilistico andrà a dirigersi.

Sitografia

- *A history of engine management systems according to Denso*
  https://www.denso-am.eu/media/corporate-news/2017/march-2017-newsletter-a-history-of-engine-management-systems-according-to-denso/
- *AURIX™ 32-bit microcontrollers for automotive and industrial applications*
  https://www.infineon.com/dgdl/Infineon-TriCore_Family_BR-ProductBrochure-v01_00-EN.pdf?fileId=5546d4625d5945ed015dc81f47b436c7
- *Future Vehicle Networks and ECUs Architecture and Technology considerations*
  https://www.nxp.com/docs/en/white-paper/FVNECUA4WP.pdf
- *Making the Case for Centralized Automotive E/E Architectures*
  https://ieeexplore.ieee.org/document/9337216
- *Vehicle-centralized, zone-oriented E/E architecture with vehicle computers*
  https://www.bosch-mobility-solutions.com/en/mobility-topics/ee-architecture/
- *Load Balancing across ECUs in Automotives*
  https://ieeexplore.ieee.org/abstract/document/5076882
- *The Migration of Engine ECU Software From Single-Core to Multi-Core*
  https://ieeexplore.ieee.org/document/9398676
- *Security in Automotive Bus Systems*
  https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.92.728&rep=rep1&type=pdf
- *Security challenges in automotive hardware/software architecture design*
  https://ieeexplore.ieee.org/document/6513548
- *Chip simulation of automotive ECUs*
  https://www.synopsys.com/content/dam/synopsys/verification/presentations/chip-simulation-automotive-sides.pdf

- *Benz Patent Motorwagen*
  https://en.wikipedia.org/wiki/Benz_Patent-Motorwagen
- *Full Virtualization of Renault's Engine Management Software and Application to System Development*
  https://arxiv.org/ftp/arxiv/papers/1802/1802.06841.pdf
- *Centralized or Decentralized? Autonomous Vehicles Are Forcing Key Architectural Decisions*
  https://www.designnews.com/electronics-test/centralized-or-decentralized-autonomous-vehicles-are-forcing-key-architectural
- *Basics of In-Vehicle Networking (IVN)*
  https://www.onsemi.com/pub/Collateral/TND6015-D.PDF
- *A review on control system architecture of a SI engine management system*
  https://www.sciencedirect.com/science/article/abs/pii/S1367578816300086