



UNIVERSITA' DEGLI STUDI DI BERGAMO
Facoltà di Ingegneria

Informatica Industriale

Prof. Davide Brugali

3.3 – CANBus

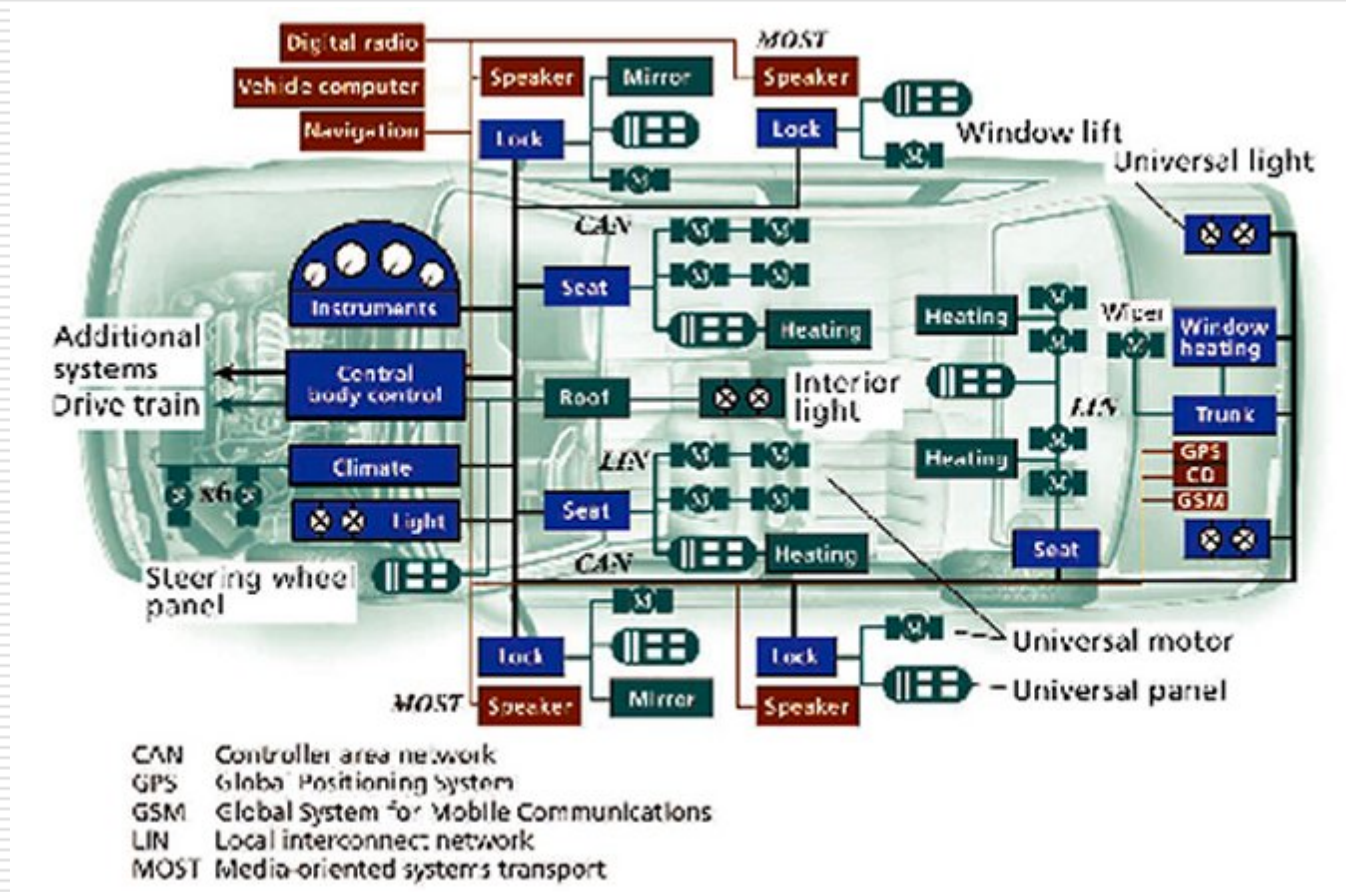
Controller Area Network

- ❑ CAN è un bus seriale di comunicazione dati progettato per applicazioni real-time
- ❑ Sviluppato da Robert Bosch nel 1986
- ❑ Consente la comunicazione tra i numerosi dispositivi elettronici presenti all'interno delle automobili, eliminando numerose linee dedicate e sensori ridondanti.

Caratteristiche

- ❑ Consente a controllori, sensori ed attuatori di comunicare a velocità fino 1Mbit/sec, offrendo:
 - ❑ bassi costi di progettazione e implementazione
 - ❑ operatività in condizioni critiche
 - ❑ facilità di configurazione e modifica
 - ❑ rilevamento automatico degli errori
 - ❑ diagnosi dei guasti
 - ❑ affidabilità e sicurezza

Controllo elettronico autovetture



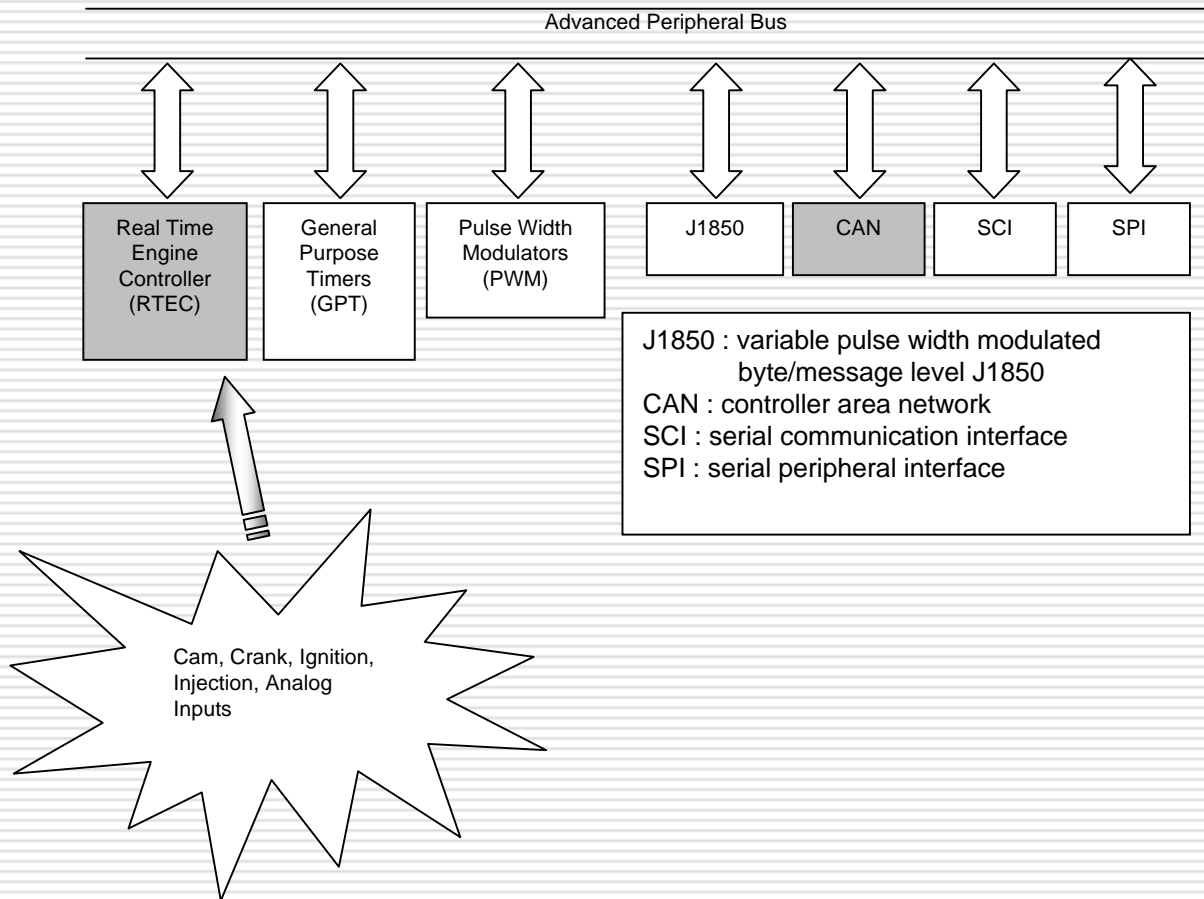
Controllo elettronico autovetture

- Sistemi di controllo elettronici altamente affidabili e *fault-tolerant*
 - ❑ assistenza alla guida
 - ❑ Sensortronic Brake System (SBS)
 - ❑ Adaptive Cruise Control (ACC)
 - ❑ Weight Classification System (WCS)

Powertrain Controller

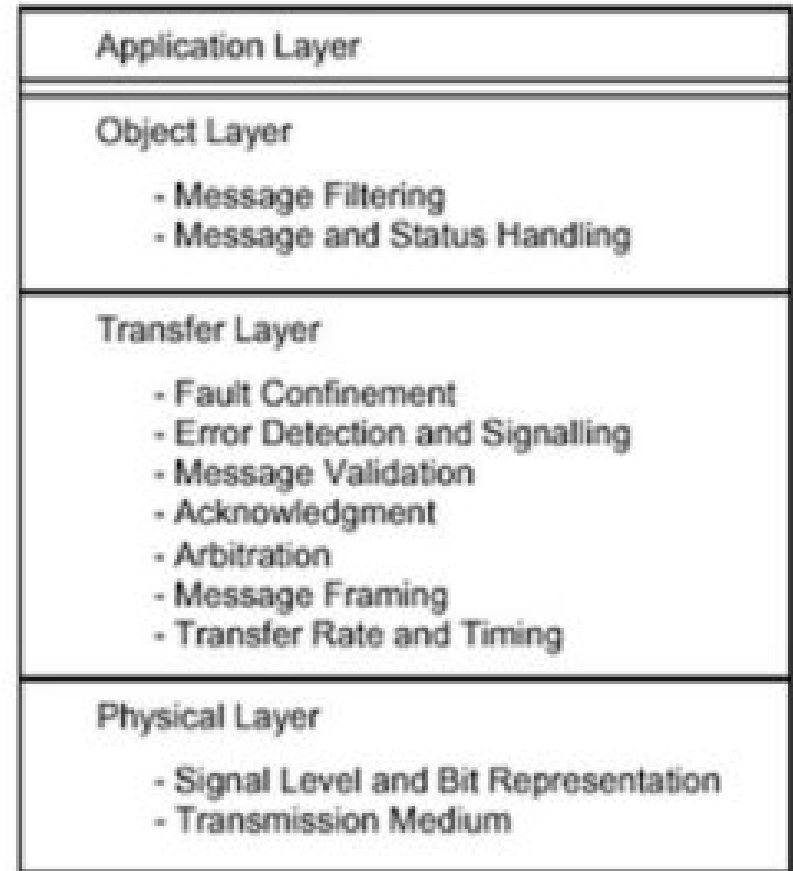
- ❑ **obiettivo** : soddisfare la richiesta di potenza (in base al segnale di input proveniente dal pedale), cercando di minimizzare il consumo di carburante e le emissioni di inquinanti e di ottimizzare le performance (es. guidabilità).
- ❑ Le diverse unità di controllo ed i sensori comunicano attraverso CAN bus.
- ❑ I μ controllori utilizzano molte informazioni: temperatura olio, densità e temperatura aria, pressione idraulica, voltaggio della batteria, velocità del veicolo...

Powertrain Controller



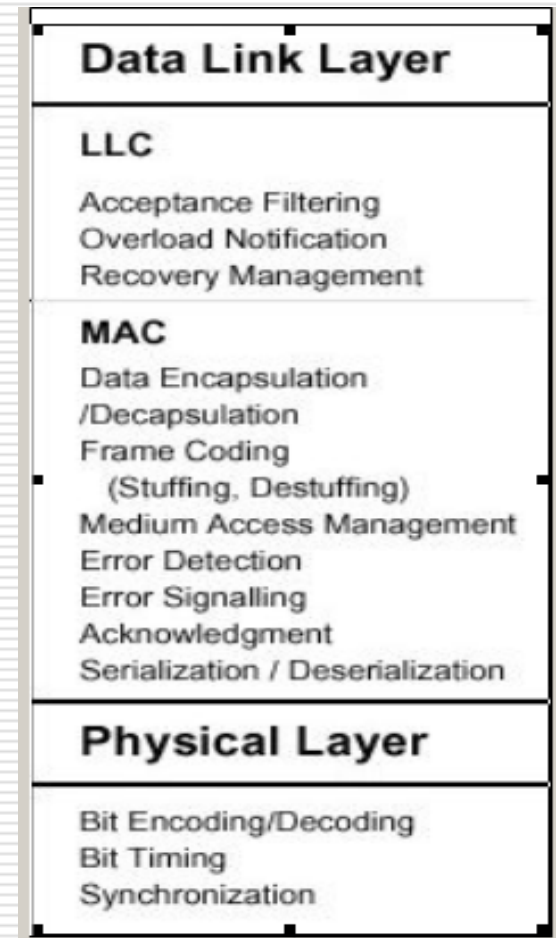
SPECIFICA CAN 2.0 – Parte A

- ❑ **Object Layer** : si occupa principalmente di individuare quali messaggi ricevere/trasmettere dal/al "transfer layer" e di fornire un'interfaccia all'"application layer".
- ❑ **Transfer Layer** : si occupa principalmente del protocollo di trasferimento (arbitrazione, segnalazione e gestione degli errori...). Fornisce i messaggi ricevuti all'"object layer" ed accetta i messaggi da trasmettere dall'"object layer".
- ❑ **Physical Layer** : si occupa del trasferimento dei bits tra i diversi nodi della rete. Ovviamente tutti i nodi presenti nella rete hanno lo stesso "Physical Layer"



SPECIFICA CAN 2.0 – Parte B

- **Data Link Layer**
 - **Logical Link Control (LLC)**
Sublayer : si occupa principalmente di stabilire quali messaggi accettare tra quelli ricevuti e di fornire servizi per il trasferimento dei dati.
 - **Medium Access Control (MAC)**
Sublayer : si occupa principalmente del protocollo di trasferimento (arbitrazione, segnalazione e gestione degli errori...)
- **Physical Layer** : si occupa principalmente del trasferimento dei bits tra i diversi nodi della rete. Ovviamente tutti i nodi presenti nella rete hanno lo stesso "Physical Layer"



Livello 1 – Physical Layer



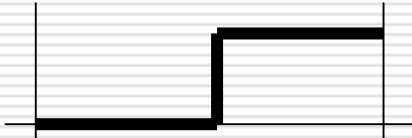
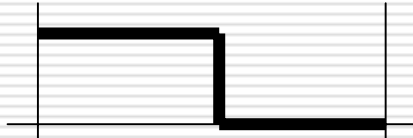
□ velocità di trasmissione:

■ 40m	1 Mbit/s
■ 100m	500kbit/s
■ 200m	250kbit/s
■ 500m	125kbit/s
■ 6km	10kbit/s

Livello 1 – Physical Layer

- ❑ Il bus può trovarsi in due stati: “dominante” (valore logico 0) e “recessivo” (valore logico 1).
- ❑ Questi stati corrispondono a tensioni elettriche che dipendono dal livello fisico adottato.
- ❑ Il livello fisico definisce le tensioni elettriche, lo schema di segnalazione sul bus, l'impedenza dei cavi...
- ❑ Questo livello non è incluso nelle specifiche CAN rilasciate da Bosch, ma viene definito dagli standard ISO11898 (CAN High Speed) ed ISO11519 (CAN Low Speed).

Bit representation

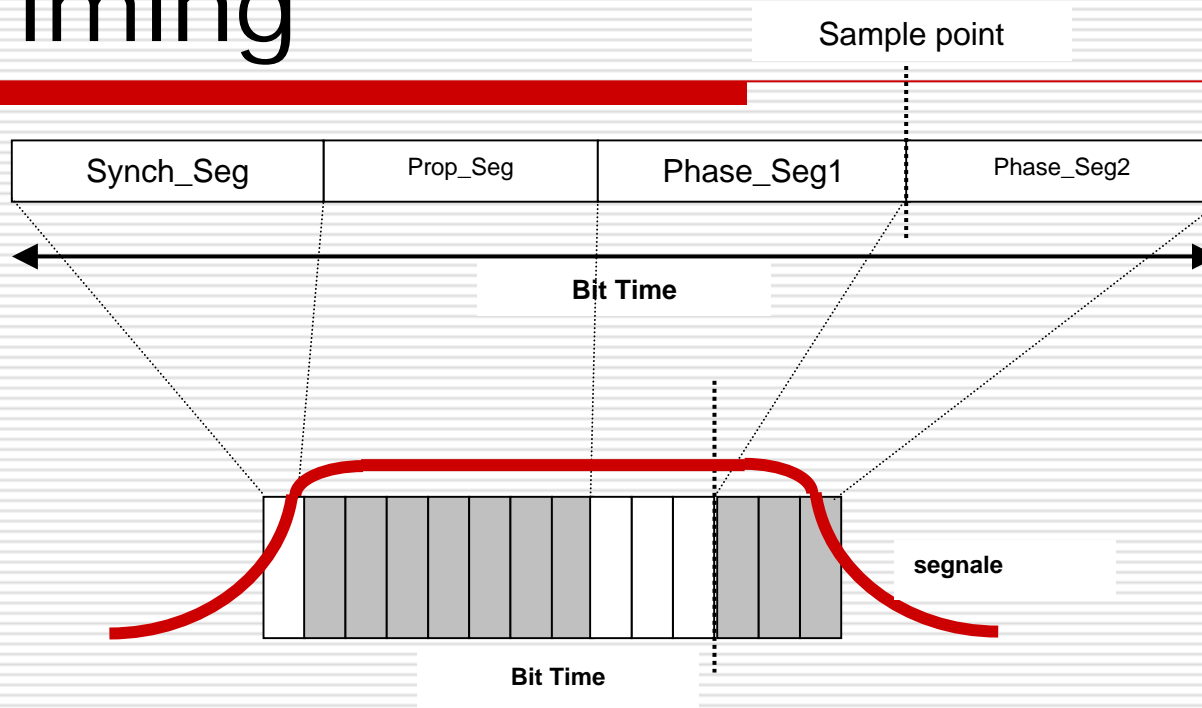
	0	1
NRZ		
Manchester		

Bit representation

□ Problema della sincronizzazione

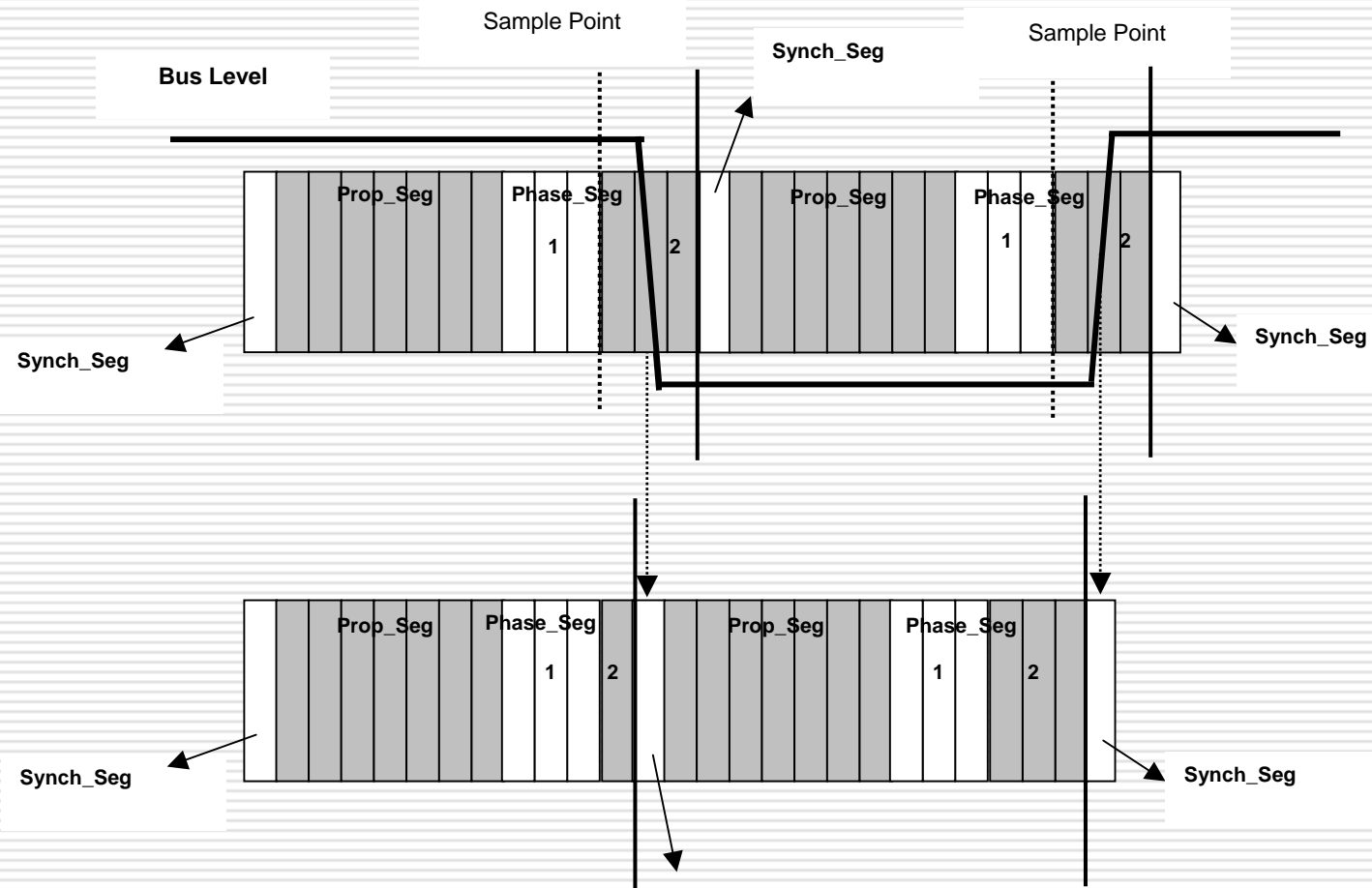


Bit Timing



- ☐ segmento di sincronizzazione (Synch_Seg)
- ☐ segmento di propagazione (Prop_Seg)
- ☐ segmento di fase 1 (Phase_Seg1)
- ☐ segmento di fase 2 (Phase_Seg2)

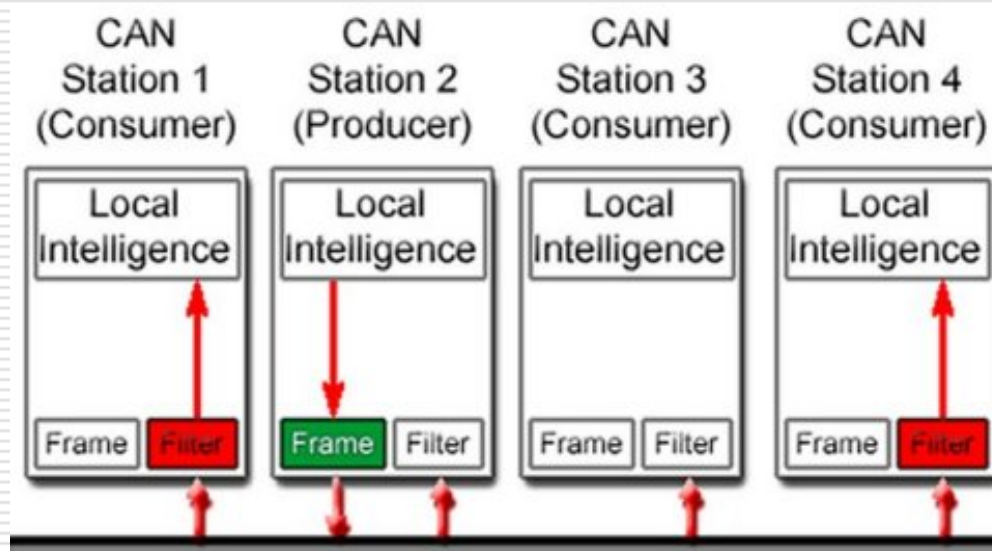
Sincronizzazione



Livello 2 – Data Layer

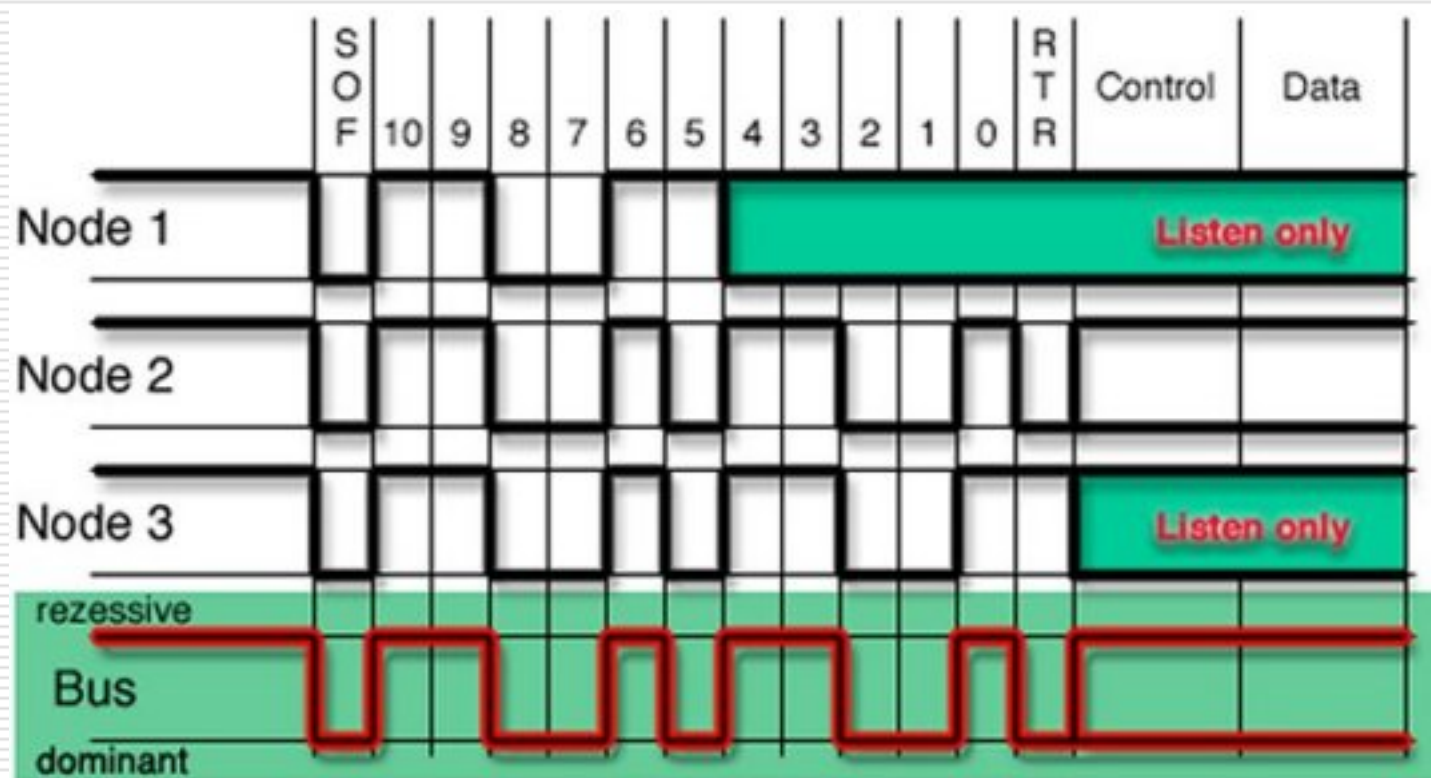
- ❑ Comunicazione di tipo *broadcast*
- ❑ I pacchetti non contengono indirizzi
- ❑ Il contenuto del messaggio è identificato tramite un identificatore, che definisce anche la priorità del messaggio.
- ❑ Ogni nodo “ascolta” tutto il traffico e filtra solo i messaggi di interesse

Comunicazione



- ❑ Station1 : tachimetro
- ❑ Station2 : nodo trasmettitore
- ❑ Station3 : unità di gestione aria condizionata
- ❑ Station4 : Engine Management System (EMS)

Bit-wise arbitration



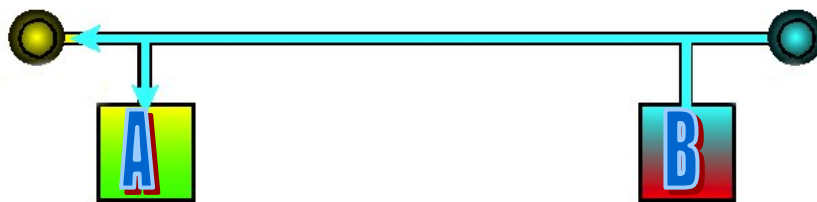
Bit-wise arbitration



Il nodo A trasmette il bit recessivo sul bus libero.



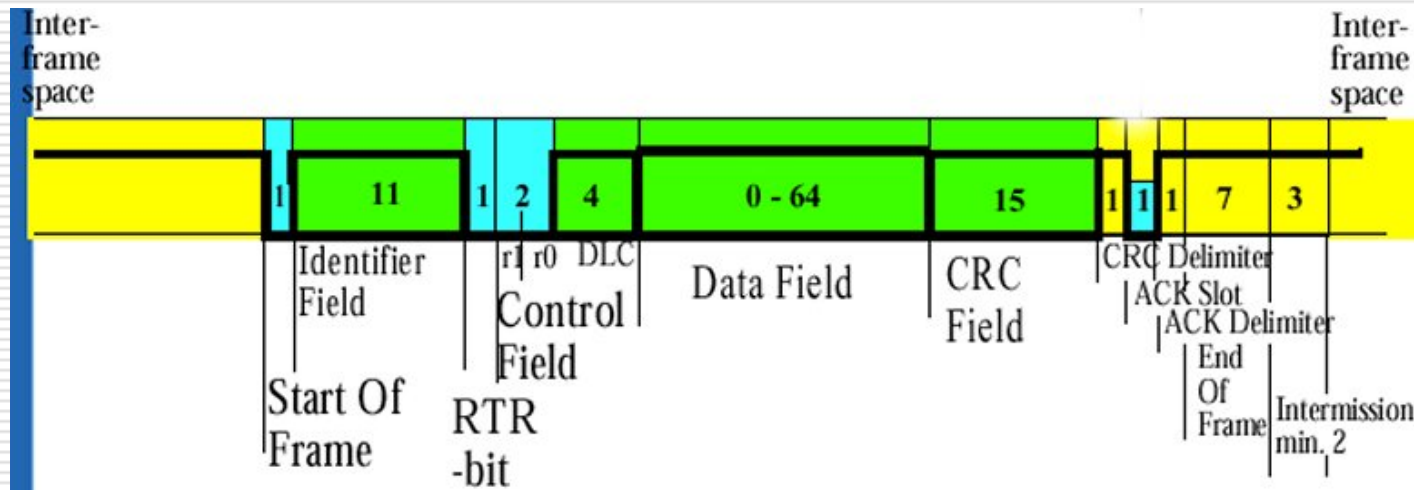
Dato che il bit trasmesso dal nodo A non ha ancora raggiunto B, quest'ultimo crede che il bus sia ancora libero e trasmette il bit dominante.



Quando il nodo A riceve il bit dominante trasmesso da B, capisce di aver priorità minore e si ritira dalla competizione. Il nodo B continua la trasmissione.

Messaggi

- ❑ Data Frame (standard / extended)
- ❑ Remote Frame (standard / extended)
- ❑ Error Frame
- ❑ Overload Frame



Formato Messaggi – Data Frame

- Un campo **Start Of Frame (SOF)**. E' un bit dominante (0 logico) che indica l'inizio di un message frame.
- Un **Arbitration Field** contenente:
 - **per CAN 2.0A, 11-bit** di *identificatore* ed il *Remote Transmission Request (RTR) bit*. Quest'ultimo quando settato a 0 indica che il frame è un Data Frame, altrimenti indica che è un Remote Frame.
 - **per CAN 2.0B, 29-bit** di *identificatore* ed il *Remote Transmission Request (RTR) bit*. Quest'ultimo quando settato a 0 indica che il frame è un Data Frame, altrimenti indica che è un Remote Frame.
L'identificatore è suddiviso in Base Identifier (primi 11-bit) ed Extension Identifier (restanti 18-bit), per garantire la compatibilità con la versione 2.0A

Formato Messaggi – Data Frame

- ❑ Un **Control Field** contenente 6 bit: 2 bit riservati per usi futuri e 4 bit di *Data Length Code* (DLC). Questo indica la lunghezza in byte del Data Field seguente.
- ❑ Un **Data Field** contenente da 0 a 8 byte di dati.
- ❑ Un **CRC Field** contenente 15-bit di *cyclic redundancy check code* ed un bit recessivo come *delimitatore*.
- ❑ Un **Ack Field** di 2 bit : il primo è lo *Slot Ack* che viene trasmesso come recessivo, ma sovrascritto con un bit dominante da ogni stazione che riceve correttamente il messaggio; il secondo bit è recessivo e svolge il compito di *delimitatore*.
- ❑ Un campo **End Of Frame** (EOF) che consiste di 7 bit recessivi.

Formato Messaggi – Remote Frame

- Si differenzia dal Data Frame poichè:
 - è esplicitamente marcato come Remote Frame (il bit RTR è recessivo)
 - non è presente il campo Data Field
- Viene usato per sollecitare la trasmissione del corrispondente Data Frame.
- Ad esempio, se il nodo A trasmette un Remote Frame con Arbitration Field posto a 234, allora il nodo B risponderà con un Data Frame avente Arbitration Field pari a 234.
- I Remote Frame possono essere usati per implementare una gestione del traffico di tipo richiesta-risposta; molti controllori CAN possono essere programmati per rispondere automaticamente al Remote Frame.

Rilevamento degli errori – Messaggio

- ❑ **Cyclic Redundancy Check (CRC)** : la sequenza CRC, contenuta all'interno di ogni messaggio, è il risultato di opportuni calcoli effettuati dal trasmettitore. Viene costruita a partire dai bit appartenenti a Start of Frame, Arbitration Field, Control Field e Data Field. Ogni ricevente calcola la sequenza CRC nello stesso modo con cui è stata calcolata dal trasmettitore. Se la sequenza CRC calcolata non corrisponde a quella ricevuta nel messaggio, viene segnalato un *CRC Error*.
- ❑ **Frame Check** : questo meccanismo controlla la struttura del frame trasmesso, verificando il formato e la dimensione dei campi. Eventuali errori vengono segnalati come *Format Errors*.
- ❑ **Acknowledgement Check** : ogni nodo che ha ricevuto correttamente il messaggio, sovrascrive il bit di acknowledgement con un bit dominante. Nel caso in cui il trasmettitore non rileva alcun riscontro al frame appena inviato, viene segnalato un *Ack error*.

Rilevamento degli errori – Bit

- ❑ **Monitoring** : ciascun trasmettitore monitora il livello del segnale trasmesso sul bus. Se il livello del bit attualmente letto differisce da quello trasmesso, viene segnalato un *Bit error*.
- ❑ **Bit Stuffing** : il trasmettitore inserisce dopo 5 bit consecutivi della stessa polarità, un bit di polarità opposta ("stuff bit"), che viene rimosso dal ricevente. Nel caso in cui un ricevente rileva più di 5 bit consecutivi dello stesso livello logico, segnala uno *Stuff Error*.

Isolamento dei guasti

- ❑ 2 *Error Count* : ricezione + trasmissione
- ❑ Stati possibili:
 - *Error Active* : entrambi contatori < 127
 - *Error Passive* : almeno un contatore > 127
 - *Bus Off* : un contatore > 255

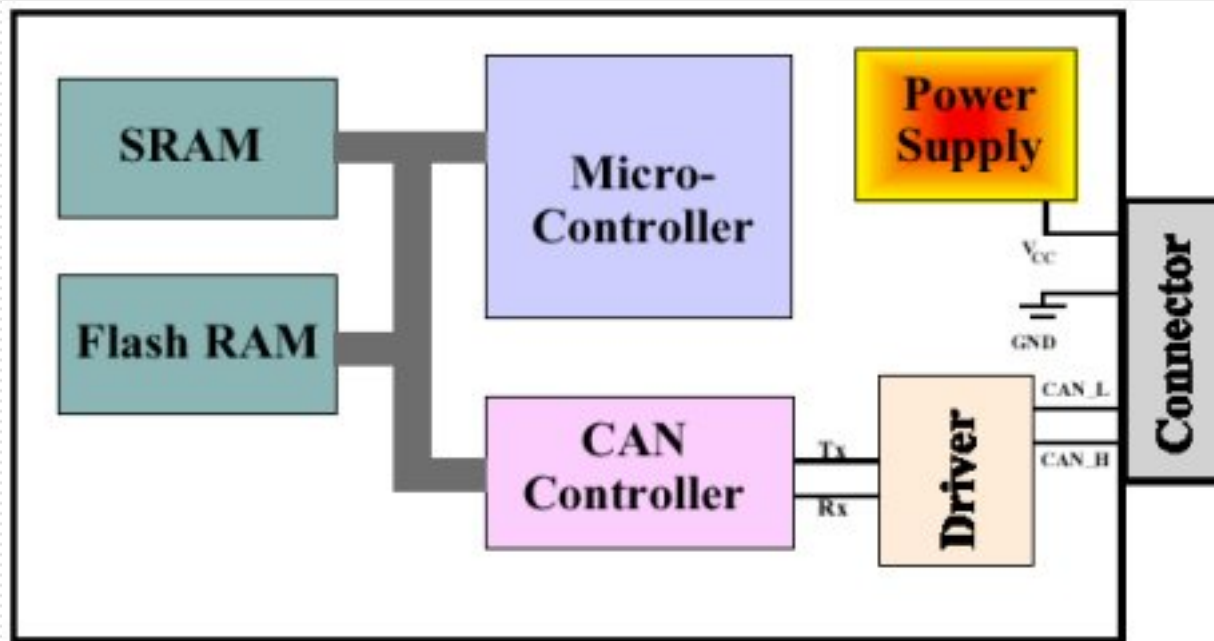


Nodo si stacca dal bus

Affidabilità

- E' stato calcolato che su una rete operante a 1Mbit/sec, con un'utilizzazione media del bus del 50%, una lunghezza media dei messaggi di 80 bit, ed un tempo di lavorazione di 8 ore al giorno per 365 giorni l'anno, un errore irrilevato capiterà ogni 1000 anni.

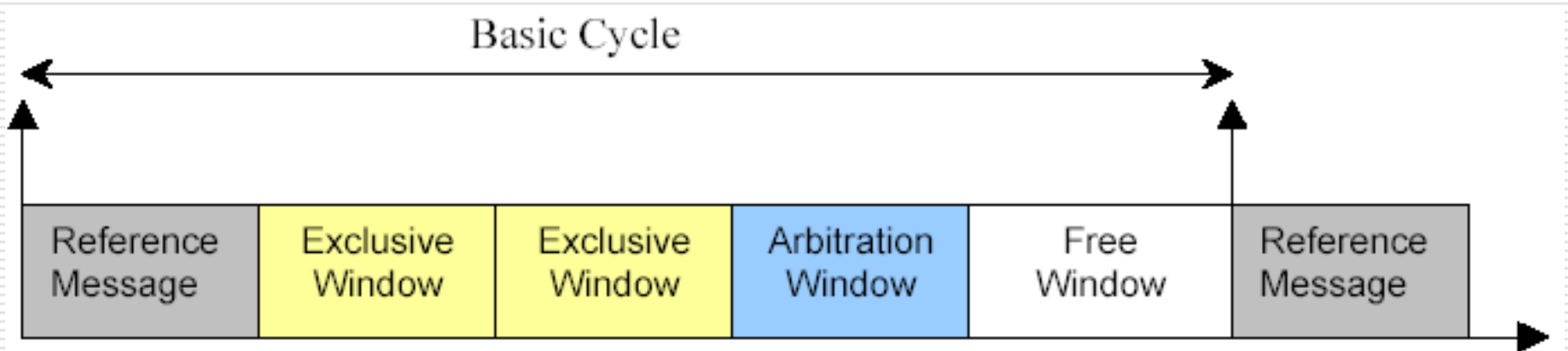
Modulo CAN



Layer 5 – Session → TTCAN

- ❑ **TTP – Time Triggered CAN**
- ❑ La comunicazione dipende da una temporizzazione predefinita al momento della progettazione del sistema
- ❑ L'accesso al mezzo trasmissivo è *conflict free*
- ❑ strategia TDMA (Time Division Multiple Access). Ad ogni nodo sono associati specifici slot temporali, durante i quali vengono trasmessi i messaggi.

Time Triggered CAN



Time Triggered CAN

- ❑ **Reference Message**: inviato dal Time Master Unit. Indica l'inizio del Basic Cycle
- ❑ **Exclusive Window** : intervallo di tempo per inviare messaggi periodici che non competono per il bus (una sola device alla volta, le altre sono disabilitate)
- ❑ **Arbitration Window** : intervallo di tempo per inviare messaggi "event-triggered" che competono per l'uso del bus
- ❑ **Free Window**: non utilizzata (future espansioni di TTP)

TTCAN Matrix cycle

Reference Message	Message A	Message B	Arbitration	Free	Message C
-------------------	-----------	-----------	-------------	------	-----------

Reference Message	Message A	Message R	Message M	Message S	Message C
-------------------	-----------	-----------	-----------	-----------	-----------

--	--	--	--	--	--

Reference Message	Message A	Message U	Arbitration	Free	Message C
-------------------	-----------	-----------	-------------	------	-----------

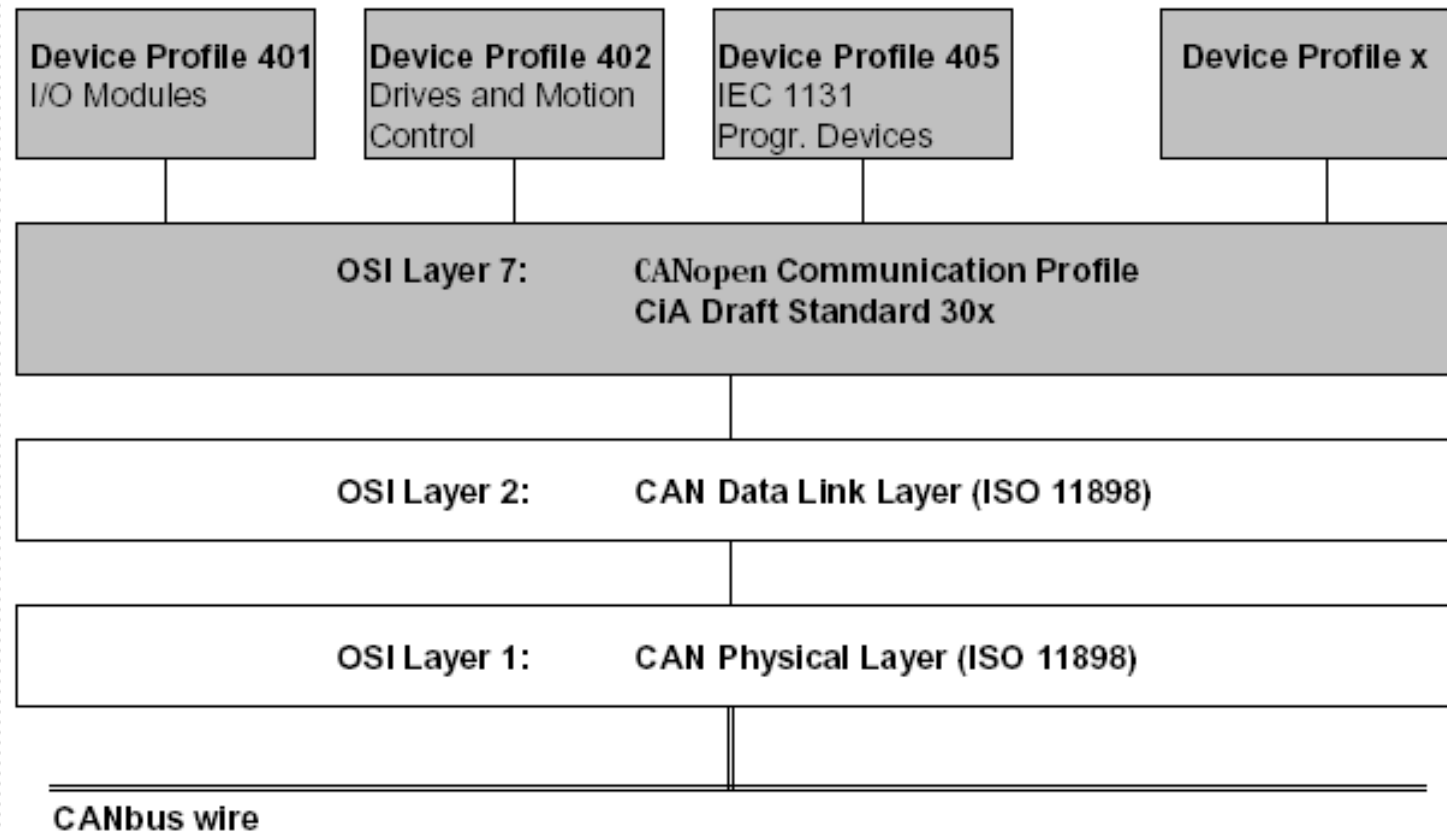
Time Triggered CAN

- ❑ Ogni TTCAN controller contiene una **MEDL (Message Descriptor List)** in cui sono racchiuse tutte le informazioni temporali relative alla trasmissione o ricezione di ogni singolo messaggio (es quando inviare un particolare messaggio)

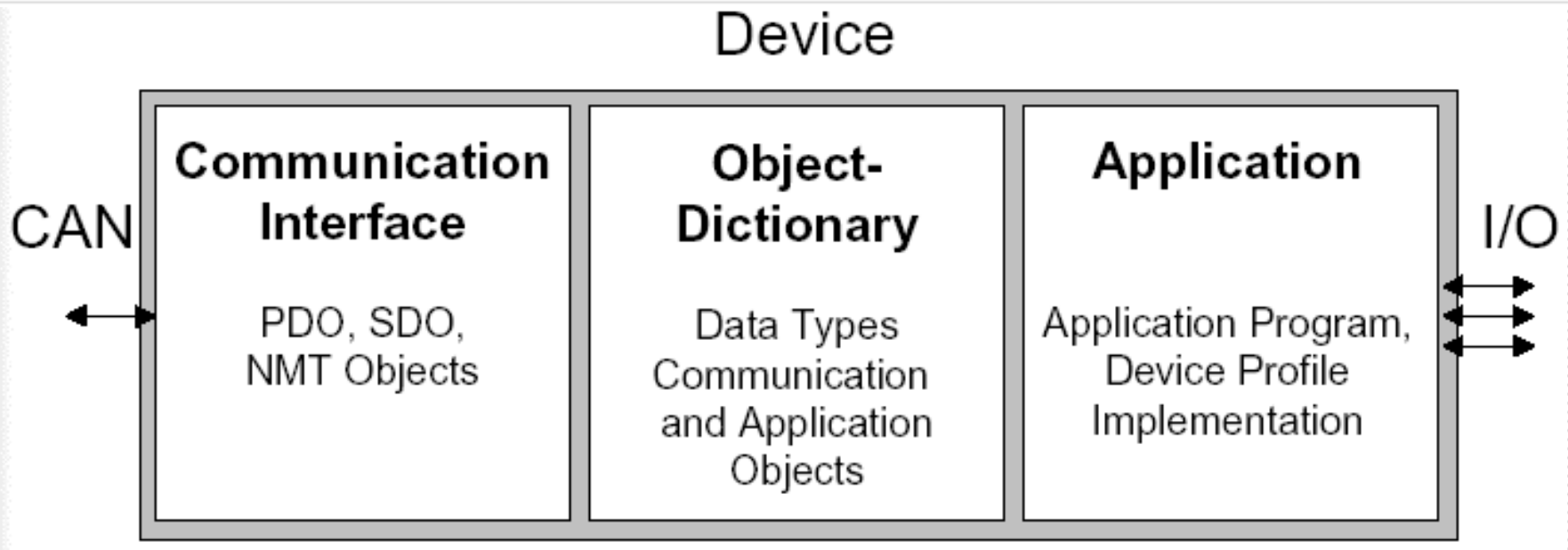
Layer 7 – CAN Application Layer

- ❑ Un messaggio di livello 2 può contenere solo 8 bytes
- ❑ Necessario un protocollo per la trasmissione di blocchi di dati
- ❑ Per favorire l'interoperabilità occorre descrivere le funzionalità delle periferiche

CANopen Reference Model



CANopen Device Model



CANopen Object Dictionary

<i>Index (hex)</i>	<i>Object</i>
0000	Not used
0001 - 001f	Static Data Types
0020 - 003f	Complex Data Types
0040 - 005f	Manufacturer Specific Data Types
0060 - 007f	Device Profile Specific Data Types
0080 - 009f	Device Profile Specific Complex Data Types
00a0 - 0fff	Reserved for further use
1000 - 1fff	Communication Profile Area
2000 - 5fff	Manufacturer Specific Profile Area
6000 - 9fff	Standardized Device Profile Area
a000 - ffff	Reserved for further user

- ❑ Una tabella di max 64K records che memorizza funzionalità e parametri di run-time di una specifica device
- ❑ A seconda del tipo di device i parametri possono essere letti/scritti

CANopen Communication Model

□ Quattro classi di oggetti comunicanti

- Process Data Objects (PDO)
- Service Data Objects (SDO)
- Network Management Object (NMT)
- Predefined Objects (Sync, Emergency-Message,
- Time-Stamp Message)

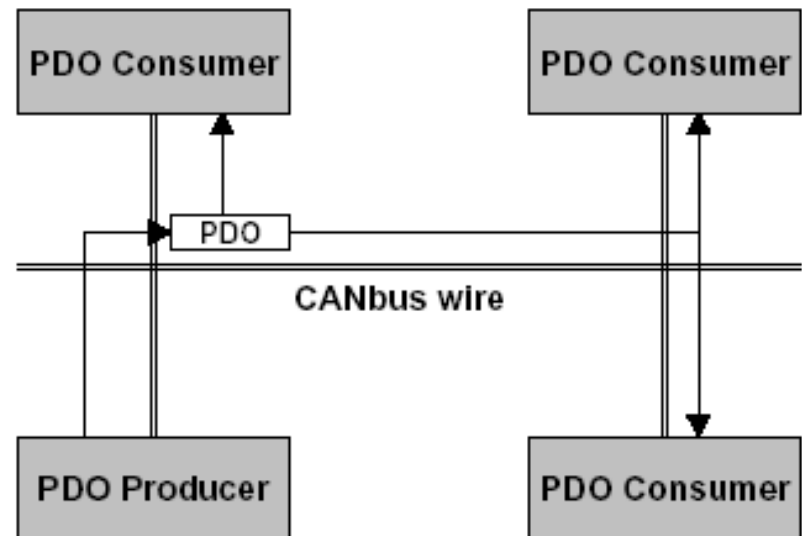
CANopen Process Data Objects

PDO communication is used for real-time, unconfirmed data transfer

Up to 8 bytes of application data

Messages can be sent in broadcast mode

PDOs can be transmitted (a)cyclic, (a)synchronous, or via remote frames

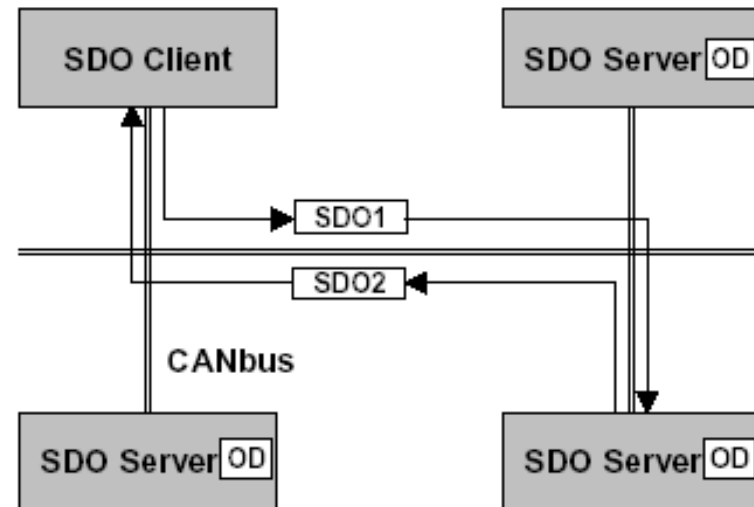


CANopen Service Data Objects

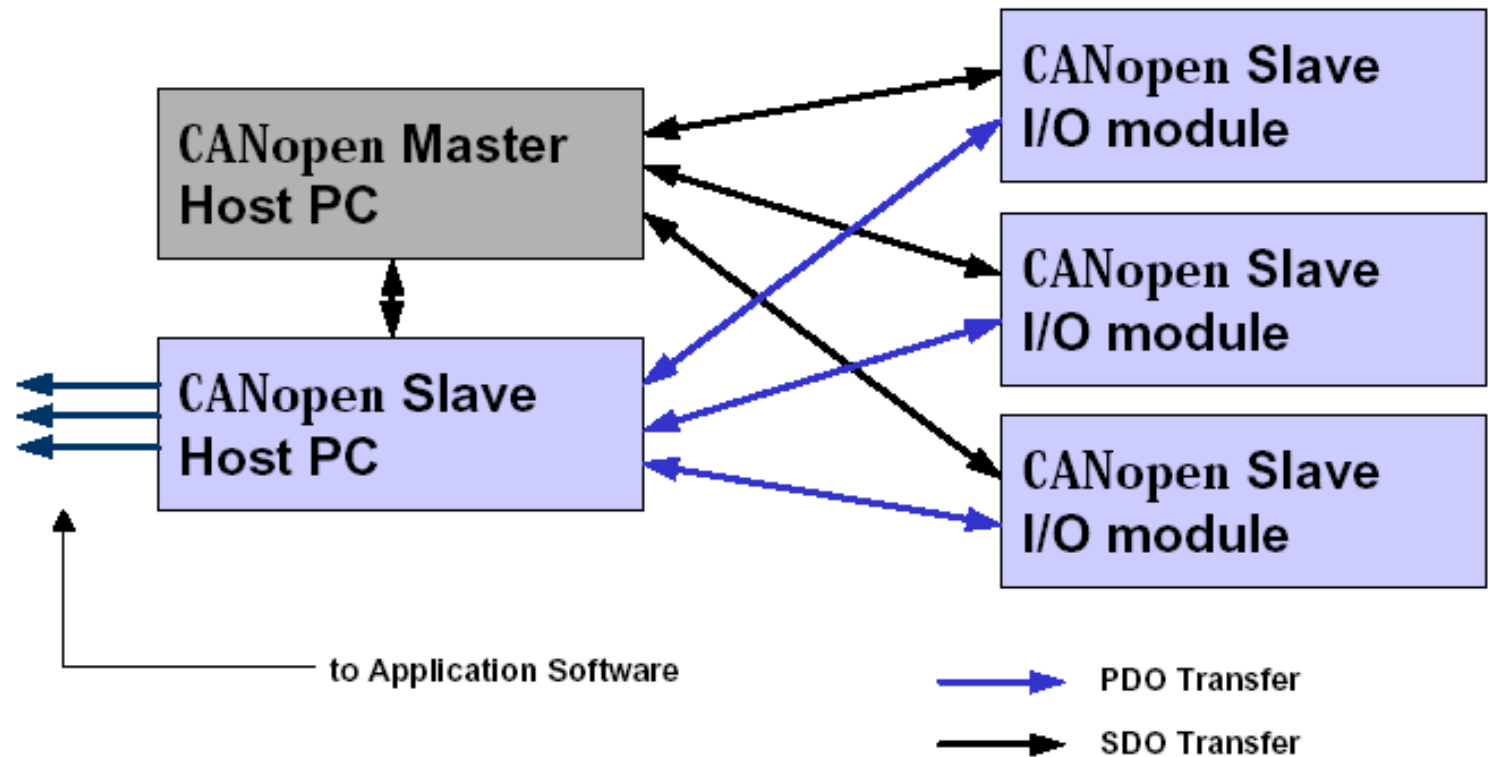
SDO communication is used to directly access the Object Dictionary and to transfer configuration data

Up to 4 bytes of data content for Expedited Transfer, unlimited for Segmented Transfers

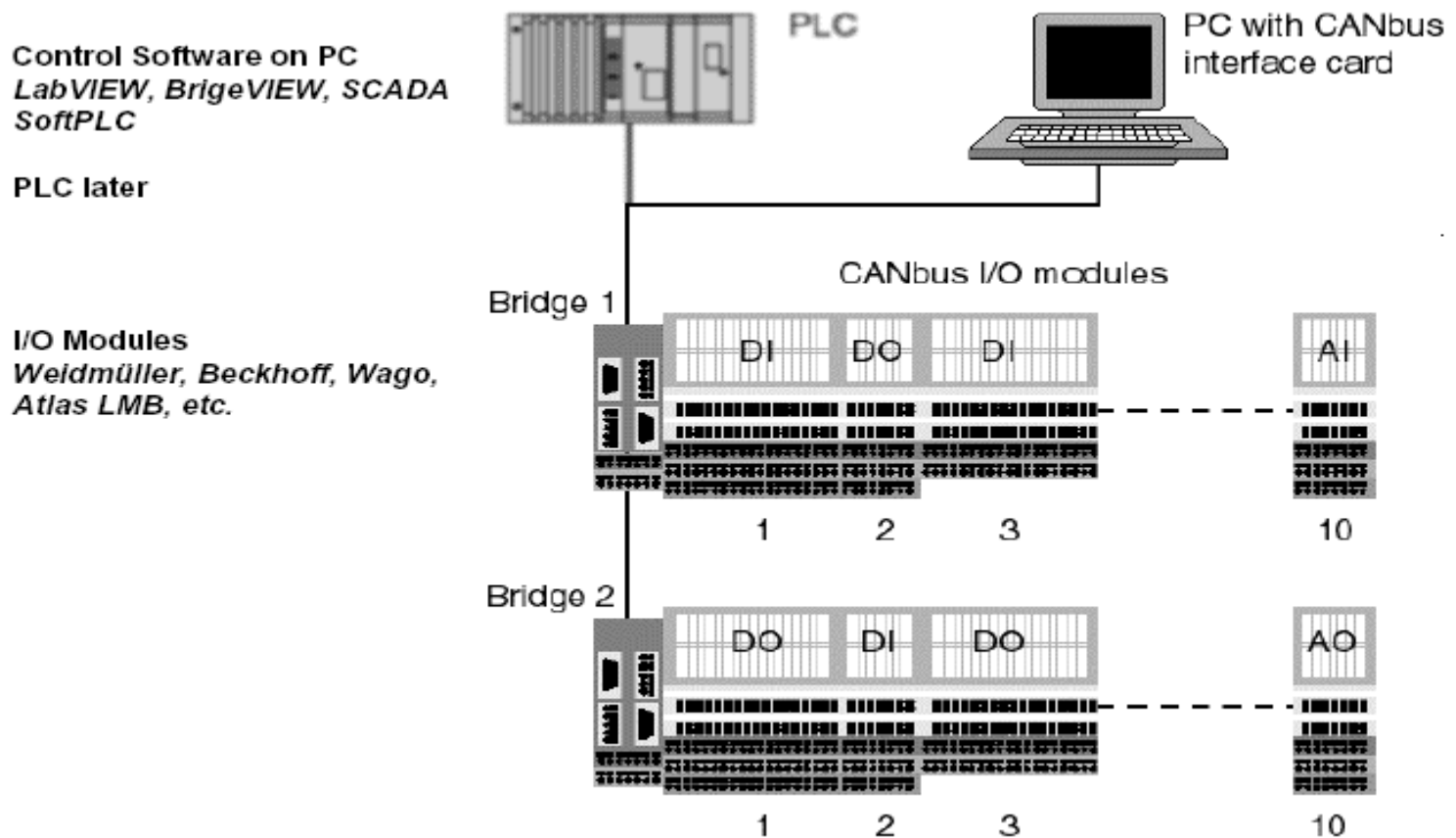
Peer to peer, confirmed communication



Functional Blocks - Software



CANopen Network



CANopen – basic services

- CAN Message Specification (CMS)
 - Vengono definite entità ad alto livello: variabili, eventi, domini accessibili via la rete
- Network Management (NMT)
 - Supporto alla configurazione, gestione e monitoraggio dei nodi di rete
- Identifier Distributor (DBT)
 - Allocazione dinamica degli identificatori di messaggio durante l'inizializzazione
- Layer Management (LMT)
 - Un nodo Master può inviare parametri di configurazione a nodi Slave