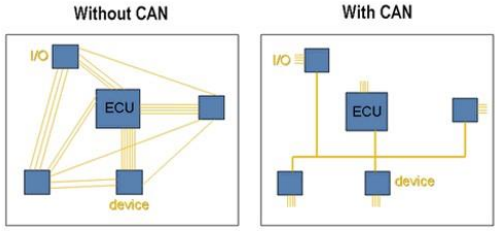
Content:

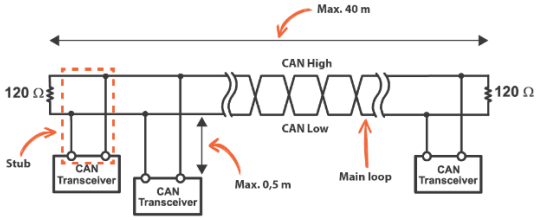
1- CAN basic concepts
   a. Old network vs CAN
   b. 2 standards
      i. CAN HIGH (ISO 11898 – 2)
      ii. CAN LOW (ISO 11898 – 3)
   c. Controller area network
   d. 2 wire bus
      i. Resistor termination
      ii. Bus logic
         1. CAN high logic
         2. CAN low logic
      iii. Transceiver
      iv. 40 meter twisted paired
   e. Event triggered
   f. Message ID
   g. Multi master
   h. Retriggering
   i. Bus off
2- CAN frame
   a. Data frame
      i. Bus idle
      ii. Start of frame
      iii. Identifier
      iv. Remote transmission request
      v. Identifier extension bit
         1. Standard format
         2. Extended format
         3. Substitute remote request
      vi. Reserved
      vii. Data length code
      viii. Data field
      ix. Cyclic redundant check
      x. CRC delimiter
      xi. Acknowledgement
      xii. Ack delimiter
      xiii. End of frame
   b. Remote frame
   c. Error frame
   d. Overload frame
   e. Message VS signal
3- CAN frame prioritization
4- CAN error detection
   a. Bit monitoring
   b. Frame format checking
   c. CRC
   d. ACK
   e. Bit stuffing
      i. Hard synchronization (with SOF)
      ii. Re-synchronization
5- CAN error handling
   a. Error frame
      i. Primary flags
      ii. Secondary flags
      iii. Error delimiter
   b. Bit monitoring error scenario
6- CAN error states
   a. TEC transmission error counter
   b. REC receiving error counter
   c. Error active
   d. Error passive
   e. Bus off
7- AUTOSAR CAN
   a. PDU router
   b. CAN TP
   c. CAN NM , generic NM
   d.

1- CAN basic concepts

- الهدف : انى بدل ما اوصل كل ال ECUs فى العربيه بباقى ال ECU و ال network تبقى complex اوى (غير ان مش كل ال sensor readings بتبقى shared) ... بالتالى ال CAN سهل كل دا



- ال CAN فى منه 2 standards
  - CAN low speed (125K)
    - ذى ال lighting مثلا
  - CAN high speed (1M)
    - ذى ال suspension مثلا
- اختصار controller area network
- 2 wire bus

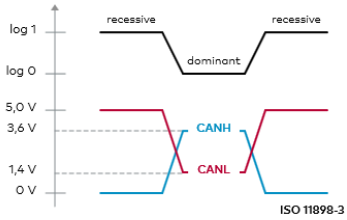


  - CAN high
  - CAN low

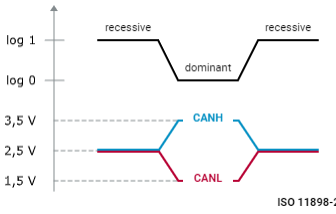
و لازم اعمل termination فى الاخر بمقاومه 120Ω عشان ال reflection
ازاى يعرف ال 1 , 0 ؟ ... بال difference بين ال CAN high , low عشان لو فى noise هتاثر فى الاتنين

Bus logic

الاسلاك دى twisted paired



لو شغال low speed can
1 if difference = 5V
0 if difference = 2V



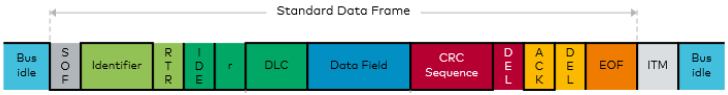
لو شغال high speed CAN
1 if diffrence = 0V
0 if diffrence = 2V

بالتالى لازم استخدم transceiver عشان يحول من ال controller logic لل can bus level

- اخره 40 meter ... لو هتزود ال length لازم تقلل ال rate والا هحتاج repeater
- Event triggered بيبعت ال frame لما يحب يبعت (بس لازم ال bus يبقى idle) ... لان بما انه multi master يقدر اى حد يبعت
- Message ID based بالتالى لما ازود node مش هيأثر على باقى ال bus (على حسب ال message ID ممكن واحد يستقبل ال frame و التانى ميعملش حاجه)
- Infinite node (theoretically)
- Multi master اى device يقدر ينزل frame
- بيقدر يعمل error detection قوى و يقدر ي re-trigger ال frame تانى لو اكتشف error
- فيه BUS OFF ... ايه دا ؟
  معناه ان لو فى node فيها مشكله هتاثر على ال bus بيفصل نفسه عن ال bus

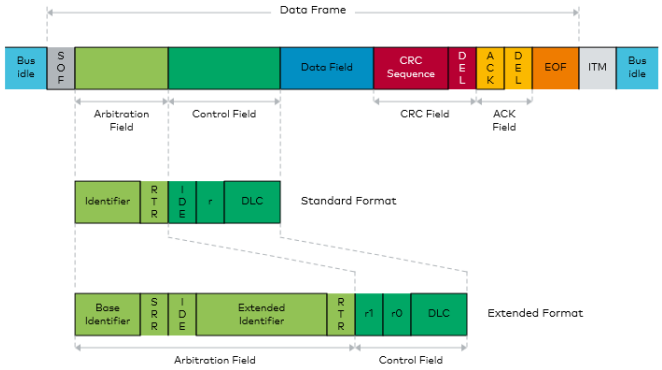
2- CAN frame

Data frame



2.1. Bus idle (1 bit = 1)
2.2. SOF Start of frame (1 bit = 0)
2.3. Identifier (message ID)
2.4. RTR (remote transmission request) (1 bit)
- 0 at data frame
- 1 at remote frame

2.5. IDE (identifier extension bit) (1 bit)
- 0 for standard format ذى الرسمه الى فوق
  فى ال standard format ال message ID = 11 bit
- 1 for extended format
  فى ال extended format ال message ID = 29 bit
  طب ليه محتاج حاجه ذى دى ؟ عشان لو عدد ال messages زاد اوى
  ال ID بيتقسم نصين 29 = 18 bit , 11 bit



2.5.1. SRR substitute remote request
هيا bit بديله لل RTR و نقل ال RTR فى الاخر بعد ال identifier
قيمتها 1
فايدتها : عشان يبقى ال standard ليه priority اعلى من ال extended فلو فى 2 frame على ال bus واحد standard و واحد extended و الاتنين ليهم نفس ال ID بتاع اول 11 bit ال standard هوا الى يكسب لان هوا الى هيطلع اول 0 بالتالى فى ال extended حطينا SRR = 1
بالظبط ذى ال data arbitration الى فى ال I2C

2.6. r reserved (1 bit = 0)
2.7. DLC data length code (4 bits)
بيتحكم عايز ابعت كام data byte فى ال frame دا
ممكن ابعت من 8 byte data – 0 بالتالى 4 bits
2.8. Data field (0 byte up to 8 byte)
2.9. CRC cyclic redundancy check (15 bit)
2.10. DEL CRC delimiter (1 bit = 1)
2.11. ACK acknowledgement (1 bit)
لما واحد يبعت ال frame لازم عالاقل واحد او اكتر يرد عليه ب ACK
و يطلع 0 على ال bus
2.12. DEL ack delimiter (1bit = 1)
2.13. EOF end of frame (7bits = 0b1111111)

Remote frame

هوا هوا ال data frame بس فيه اختلافين

1- RTR = 0
2- No data (DLC = 0) number of data byte = 0
لو ECU1 عايزه حاجه من ECU2 ... فمفيش data بيعتهاله اصلا دا
مستنى منه data فبيبعتله frame عشان ECU2 يشوف ال message ID فيقوم يرد بقى عليه بال data
ذى مثلا ال dashboard لما بتسال ال Motor ECU عن درجه الحراره ... فال dash board مش محتاجه تبعت data بس عايزه data من ال temp sensor

error frame

overload frame

ذى ال error frame
فايدته : لو الى بيستقبل عايز يعرف الى بيبعتله انه بيبعت ب data rate عالى
انا كدا كدا بظبط ال configurations كلها عشان كله يشتغل بنتفس ال speed بس دا
لو انا غلطت فى ال configuration

طب لو جه مثلا اكتر من 5 ones ... ال CAN هيحط بعد ال 5 ... bit ب 0
و لو جه اكتر zero هيحط بعد كل 5 متشابهين 1 = bit
ال sender بيحطها لوحده و ال receiver بيشيلها لوحده
ليه ؟ .... طريقه ال synchronization فى ال CAN هيا

1- Hard synchronization

الى هيا start of frame فلازم كل ال nodes تبص على ال bus مع اول falling edge فى ال SOF

2- Re synchronization

من ال falling , rising edges الى فى ال frame فلو قعدت فتره اكبر من 5 bits مجاليش ولا falling or rising edge يبقى انا كدا السرعه بتاعتى بطيئه مقارنه بال transmitter فمحتاج اعمل resynchronization

فلو جالى 6 bits متشابهين ورا بعض من 5 ذى ما ال CAN standard بيقول ... يبقى كدا فى error و الى بيكتشفه ال receiver

5- Error handling

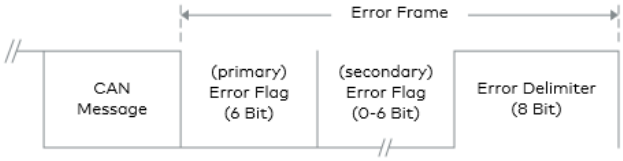
- اى حد بيكتشف اى error من ال 5 ... بينزل error frame
- لما ال error frame ينزل على ال bus الى كان باعت ال message ببعتها تانى

Error type	Node that detect that error
Bit monitoring	Sender
Frame format	Receiver
CRC	Receiver
ACK	Sender
Bit stuffing	Receiver

Error frame

دا ال frame الى بينزل على ال bus لو اى حد detect error
بيتكون من :

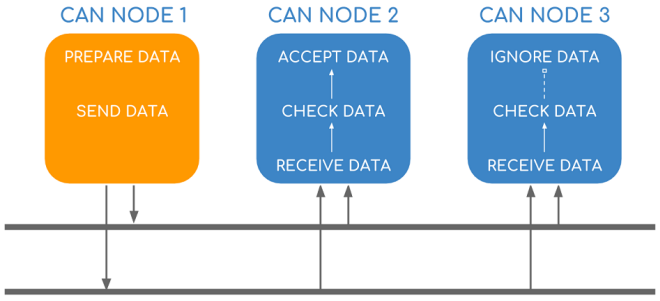
1- Primary flags (6 bits = 000 000)
2- Secondary flags (6 bits = 000 000)
3- Error delimiter (8 bits = 1111 1111)



١- لاقيت error انا هبعت error frame

٢- انا سمعت ان فى error و انا الى كنت باعت ال data انا هبعت ال data frame تانى

٢- انا سمعت ان فى error انا همسح ال data frame الى جالى



Scenario :

١- CAN node1 بعتت data frame
٢- CAN node2 اكتشفت ان فى error (اى نوع من ال 5)
٣- CAN node2 هتنزل على error frame على ال bus
٤- كل ال nodes الى استقبلت ال data frame دا لما تشوف بعده ال error frame بترمى ال data frame الى استقبلته
٥- CAN node1 لما تشوف ان ال data frame الى نزلته جه بعده error frame تقوم منزله نفس ال data frame تانى على ال bus

Message vs signal

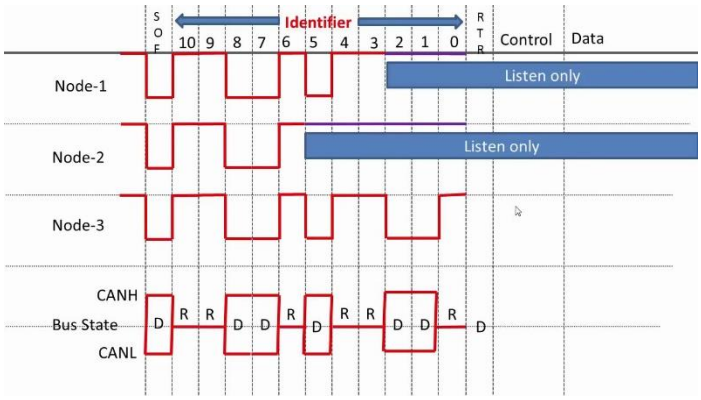
لو مثلا باب العربيه ال state بتاعته عباره عن 1 bit و حاله الازاز بتتمثل فى 3 bit و ال door lock بيتمثل فى 1 bit ... فمش منطقى انى ابعت CAN frame كامل (140 bit) عشان ابعت بس حاله الباب (1 bit) و بعديه CAN frame تانى عشان ابعت 3 bit window state كدا انا بضيع ال bandwidth
طب هعمل ايه ؟
هنبعت message جواها كذا signal يعنى ال data byte فى ال message هنحط جواه ال

- 1 bit door state
- 3 bits window state
- 1 bit lock state

كلهم فى message واحده
فممكن ال message يبقى جواها اكتر من signal

3- CAN frame prioritization

لو فى 2 ECU بعتو 2 messages فى نفس الوقت ؟
ذى فى ال I2C ... الى هيكسب اول واحد يطلع اول 0 او صاحب ال message ID الاقل (لان كل ما قل الرقم كل ما ال ID اوله بقى فيه اصفار كتير)



4- CAN error detection

- ال can يقدر ي check على 5 types of errors

- Bit monitoring

- ال sender هوا الى مسؤل انه يكتشفه
- ال sender بيشوف الى هوا مطلعه هوا الى محطوط على ال bus ولا لا

- Frame format checking

- ال receiver هوا الى بيعمل check على الحاجات دى
- بيعمل check على شكل ال frame بيبص على ال
  - CRC delimiter
  - ACK delimiter
  - EOF
- ال bits دى كلها ب 1 لو اى حاجه ب 0 يبقى اكيد ال timing مثلا مش مظبوط

- CRC

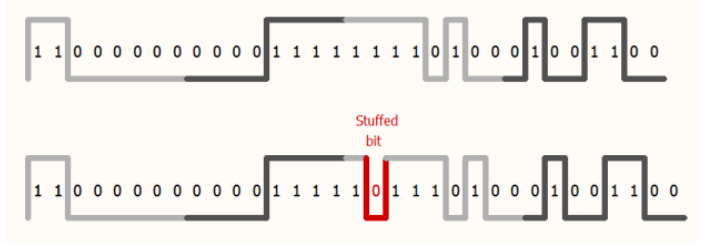
- ال receiver هوا الى بيعمل check على ال error دا
- هيستقبل ال data و يطلع ال check sum بتاعتها و يقارنها بالى جايله من برا

- ACK

- الى بي check ال error دا هوا ال sender
- ال sender بعد ما بيبعت ال CRC , data field بيستنى ack ... فلازم واحد عالاقل يرد عليا ب ACK (مانا مش بكلم نفسى) لو محدش بعت ACK دا يبقى error

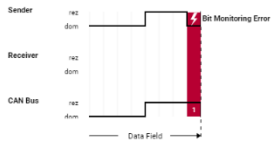
- Bit stuffing

- ال CAN مبيسمحش باكتر من 5 bits ورا بعض بنفس القيم



This error scenario is from
https://elearning.vector.com/mod/page/view.php?id=358
Bit monitoring error scenario

١- ال sender بيبعت frame (start bit , ID , ...) و هوا بيبعت جه يبعت 0 لاقى على bus = 1 ال

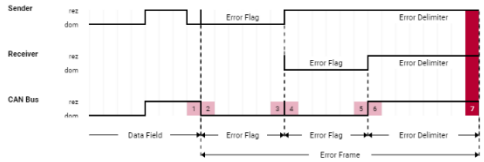


٢- ال sender هيحس بال error دا ... فيبتدى يعرف الناس ان حصل error عن طريق انه يبعت error frame ... فهيبتدى يبعت (6bits = 000 000) primary flag

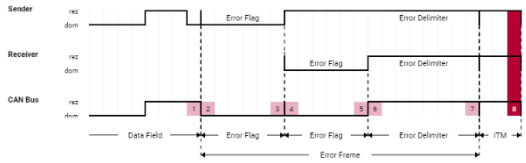
٣- دا هيعمل bit stuffing error عند ال receiver لانه شاف 6 zeros ورا بعض

٤- لما يحصل bit stuff error عند ال receiver يقوم باعت 000 000 هوا كمان الى احنا بنسميها secondary flag

٥- بعد ما ال receiver يبعت 6 bits ب zeros بيبتدى يبعت فى ال delimiter الى هما 8bits = 1111 1111 و كدا كدا ال sender كان بادئ فى ال delimiter من بدرى ف الى هيطلع على ال bus وحايد



٦- بعد كدا كله بيستنى فتره اسمها INM intermit ion ... دا الوقت بين كل frame و التانى (لازم ال bus يبقى idle بين كل frame و التانى عالاقل 3 bits) فبيطلع 111



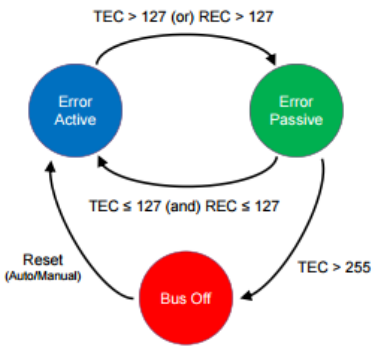
٧- بعد كدا ال sender بيبعت ال frame تانى

6- CAN error states

جوا ال can controller ... 2 error counters

1- TEC transmission error counter
لو بعت frame غلط يزود ال TEC

2- REC receiving error counter
لو استقبل frame غلط يزود ال REC



١- فى الاول خالص كل ال can nodes بتبقى فى ال error active طول ما ال
receive error ≤ 127
transmit error ≤ 127

٢- لو ال node هيا الى بعتت ال frame الغلط ... هتزود ال transmit error + 8 (ال TEC بيزيد ب 8)
لو انا استقبلت صح بس واحد تانى استقبل غلط ... هيزود ال receive counter + 1 (ال REC بيزيد ب 1)
لو انا الى استقبلت و انا الى detect ال error بيقى هزود ال REC + 8
من الاخر الى يخصنى هيزودنى ب ٨ و الى يخص غيرى يزودنى ب ١

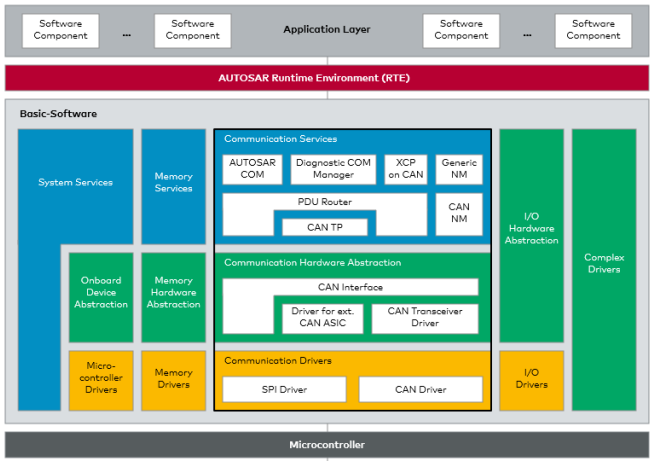
٣- لو ال TEC or REC زادو عن ال 127 هروح لل error passive
عقابى هنا ان ال error frame بتاعى بقى 111 111 6 bits = فبقى مالوش اى لازمه اكنى قفلتله ال errors عشان ميقرفنيش

بيزود فتره ال intermission time ITM تبقى بتساوى 8 bits

٤- لو وصل ال TEC بقى اكبر من 255 ساعتها هيروح لل bus off و يفصل نفسه عن ال bus و الحل
- انى اعمل RESET لل ECU
- انى اعمل re initialization لل can driver فى الكود
- ان ال driver يستنى 128 مره يحصل فيها 1 = 11 bits ورا بعض

- خلى بالك ... لو ال CAN بعت frame صح بيقلل ال counter ب ١
- و لو استقبلت frame صح بنقص ال counter ب ١

7- AUTOSAR CAN



1- PDU router

ال application و ال RTE ميعرفوش هما هيبعتو ال message دى على انهى protocol فشغالين ب message ID مش خاصه باى حاجه ... يعنى مثلا
ID 200 -> CAN
ID 201 -> LIN
ID 202 -> SPI

الى بيظبط الحاجات دى هيا ال PDU router
هوا عارف ان ال ID 200 هيتبعت بال CAN و بال CAN message ID = 3 مثلا و هكذا و بيبعتها لل CAN IF
لو مثلا هبعت 8 byte ف ال PDU هيكلم ال CAN IF على طول
بس لو عايز ابعت مثلا 16 byte يبقى انا كدا محتاج ابعته على مرتين فيبعته لل TP و ال TP يدى frame frame لل CAN IF

2- CAN TP (CAN transport layer)

عشان ي handle ال data الكبيره و يبعتها واحده واحده لل CAN IF

3- CAN NM , generic NM

وظيفتهم يبعتو ذى dummy frames عشان يقدرو يعملو management لل network و ميخلوش ال busses ... idle