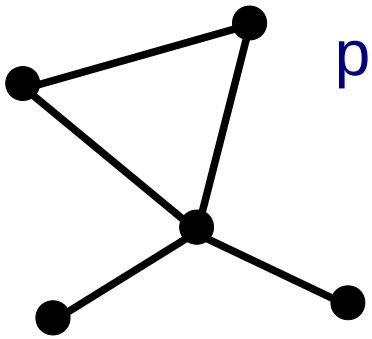
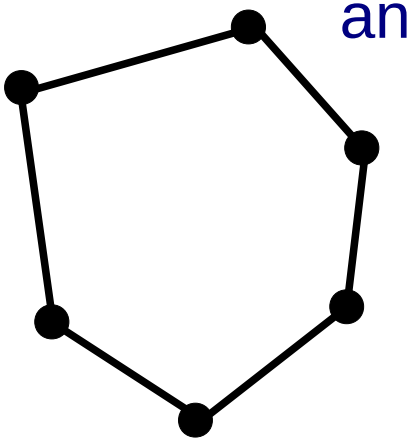
Bus de communication

introduction

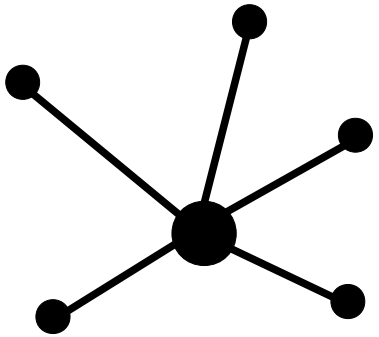
- bus de communications avec des systèmes extérieurs à la machine)



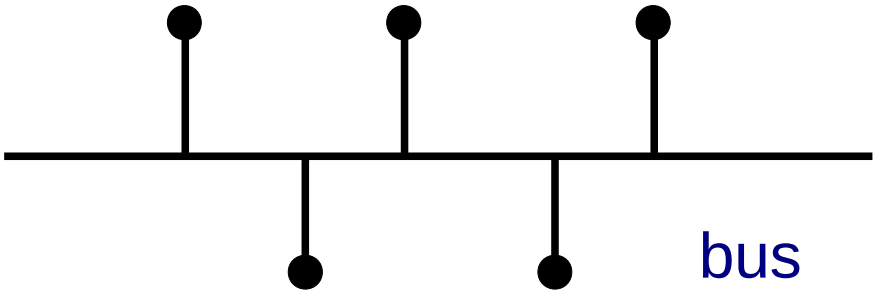
point à point



anneau



étoile



bus

introduction

- bus de communications avec des systèmes extérieurs à la machine
  - on ne traitera pas les bus PCI, ISA, PCMCIA ou SCSI (périphériques)
  - ni les bus série comme RS232 ou USB
  - PC104 et PC104+ sont deux bus que l'on rencontre beaucoup dans les applications TR. Ils sont équivalents à ISA et PCI
- bus de terrain (CAN)
- bus industriel (VME)

Bus CAN

introduction

- Controller Area Network présenté en 1983 par la société Bosch (équipementier pour l'automobile)
  - http://www.can-cia.org

introduction

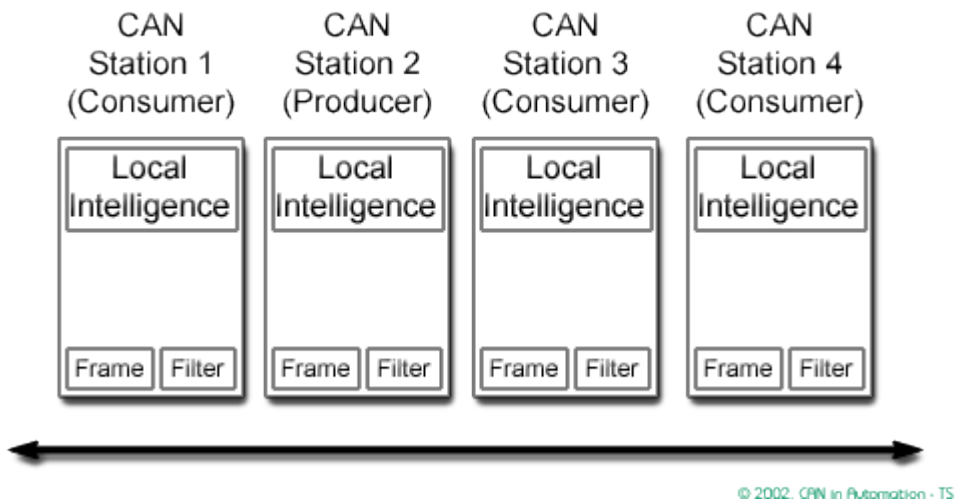
- Controller Area Network présenté en 1983 par la société Bosch (équipementier pour l'automobile)
  - http://www.can-cia.org
- protocole de communication défini par Bosch et Intel en 1985
- standardisation par l'ISO en 1986
- premier contrôleur (82526 par Intel) en 1987
- première voiture multiplexée par CAN en 1991

introduction

- système basé sur un bus série
  - hiérachisation des messages
  - garantie des temps de latence
  - souplesse de configuration
  - réception de multiples sources avec synchronisation temporelle (CSMA/CA)
  - fonctionnement multi-maître
  - détection et signalisation des erreurs
  - retransmission automatique des messages altérés dès que le bus est à nouveau au repos
  - distinction des erreurs temporaires ou de non-fonctionnalité permanente
  - déconnexion automatique des noeuds défectueux

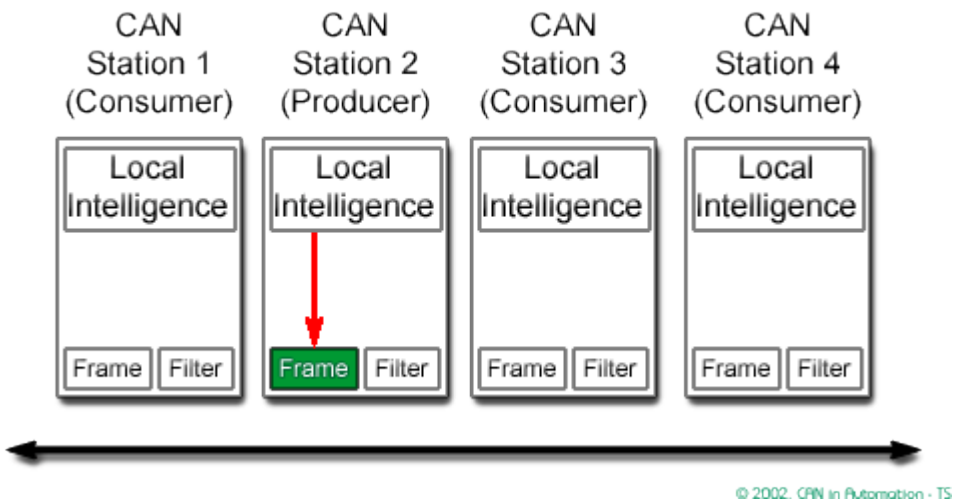
protocole de communication

- ISO 11898
- mécanisme de communication par diffusion (broadcast)



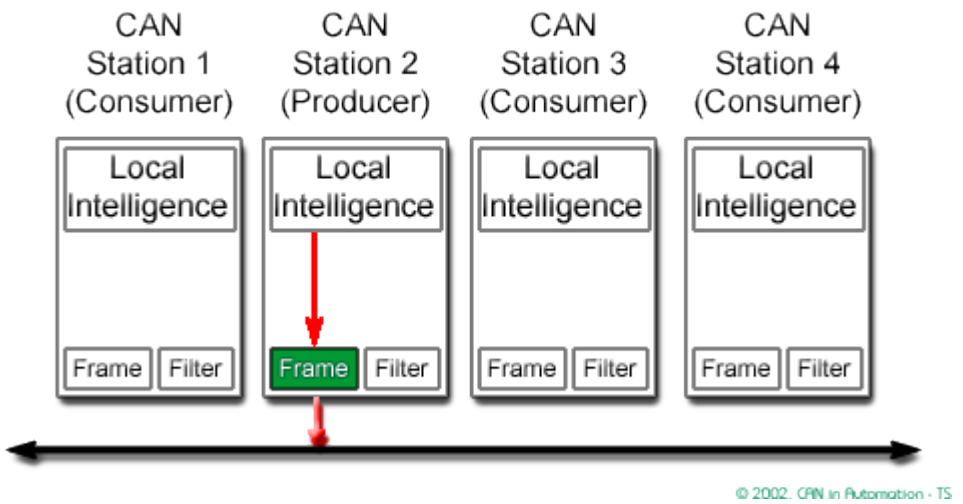
protocole de communication

- ISO 11898
- mécanisme de communication par diffusion (broadcast)



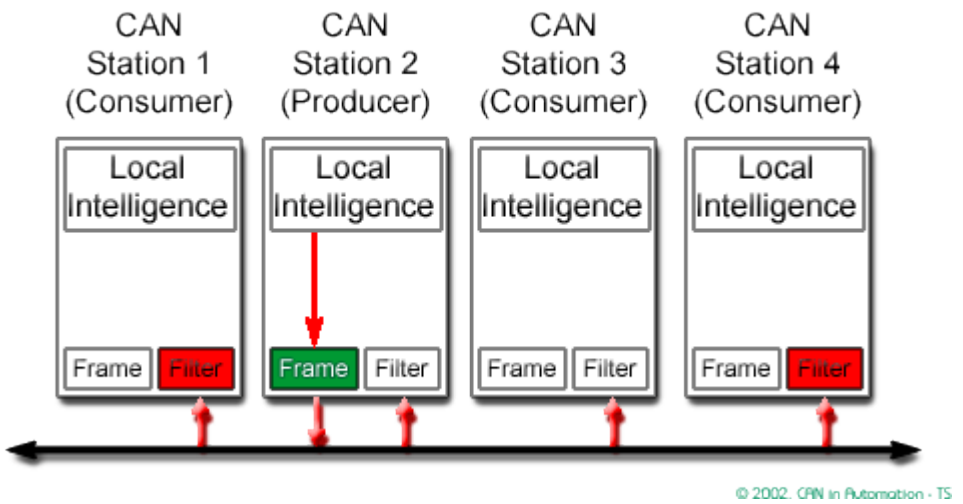
protocole de communication

- ISO 11898
- mécanisme de communication par diffusion (broadcast)



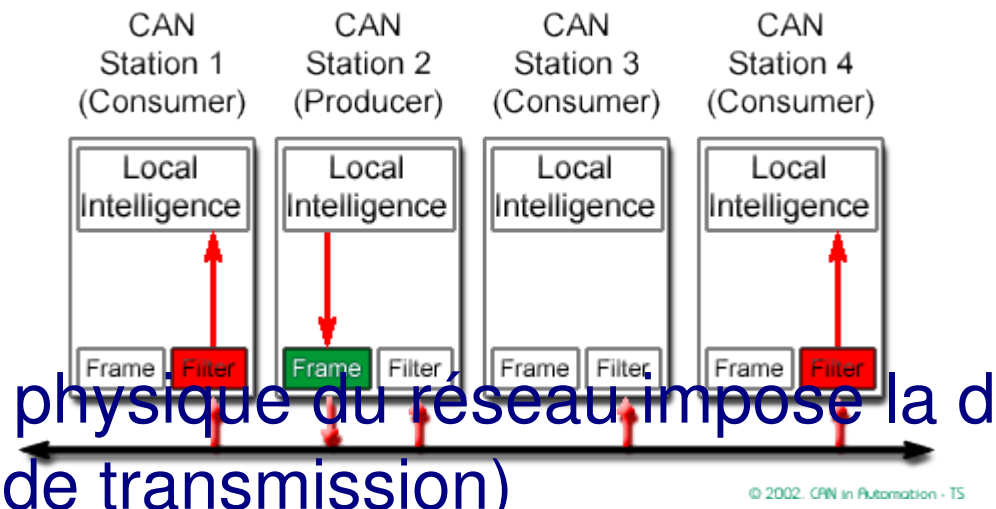
protocole de communication

- ISO 11898
- mécanisme de communication par diffusion (broadcast)



protocole de communication

- ISO 11898
- mécanisme de communication par diffusion (broadcast)

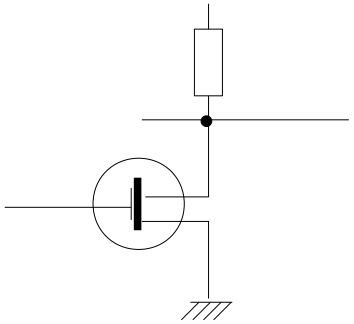


la longueur physique du réseau impose la durée du bit (fréquence de transmission)

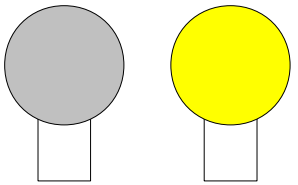
- vitesse de propagation ~ 5ns/m

protocole de communication

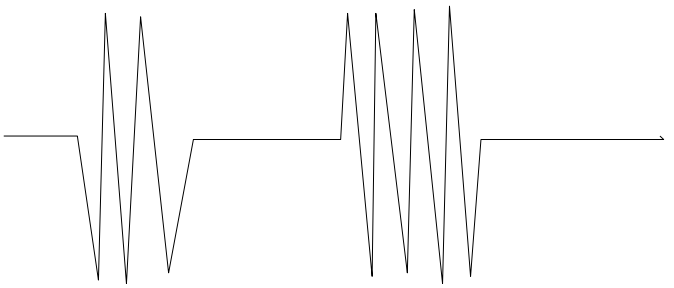
- valeur du bus
  - indépendante du support physique (fil de cuivre, infra-rouge, hertzienne, fibre optique, ...)
  - 2 valeurs logiques : "dominant" ou "récessif"
  - en cas de transmission simultannée d'un bit dominant et d'un bit récessif, la valeur résultante du bus sera "dominant"



Liaison filaire
Récessif : rappel de potentiel
Dominant : conducteur à la masse



Fibre optique
Récessif : absence de lumière
Dominant : présence de lumière



Liaison hertzienne
Récessif : absence de porteuse
Dominant : présence de porteuse

protocole de communication

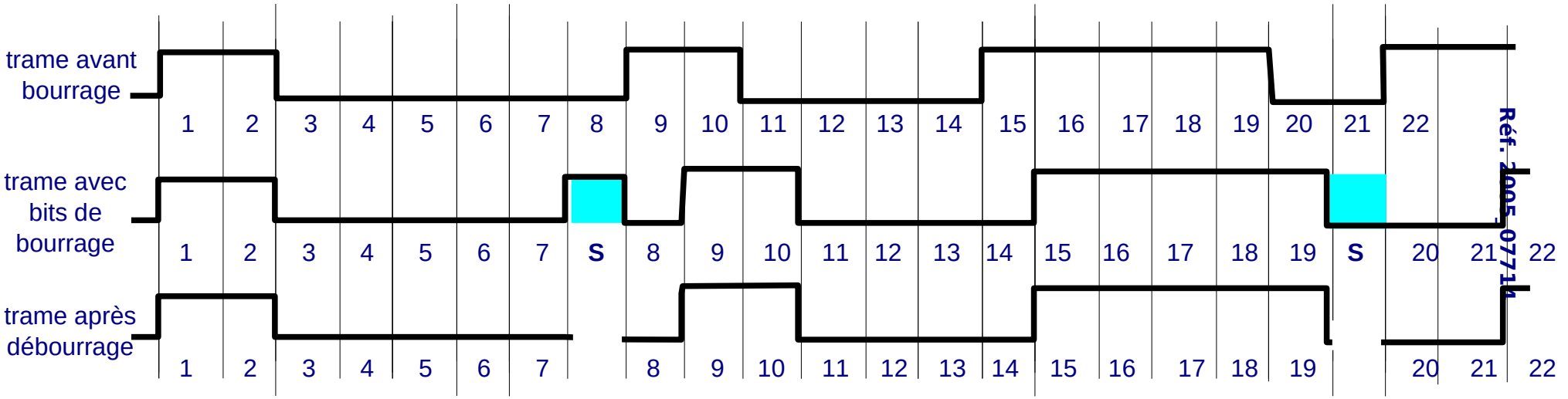
- procure une grande flexibilité (ajout de stations de réception)
- arbitrage à partir de l'identificateur du message
  - comparaison bit à bit
  - l'identificateur de plus petite valeur est le plus prioritaire
  - défini au moment du design et ne peut pas être changé dynamiquement

protocole de communication

- format des messages standardisé
  - DATA FRAME
    trame de données, transportant l'information d'un émetteur vers le récepteur
  - REMOTE FRAME
    trame de requête, pour demander la transmission d'un DATA FRAME avec le même identificateur
  - ERROR FRAME
    trame d'erreur transmise par une unité lorsqu'elle détecte une erreur de bus
  - OVERLOAD FRAME
    trame de surcharge utilisée pour générer un retard supplémentaire entre DATA FRAME et REMOTE FRAME

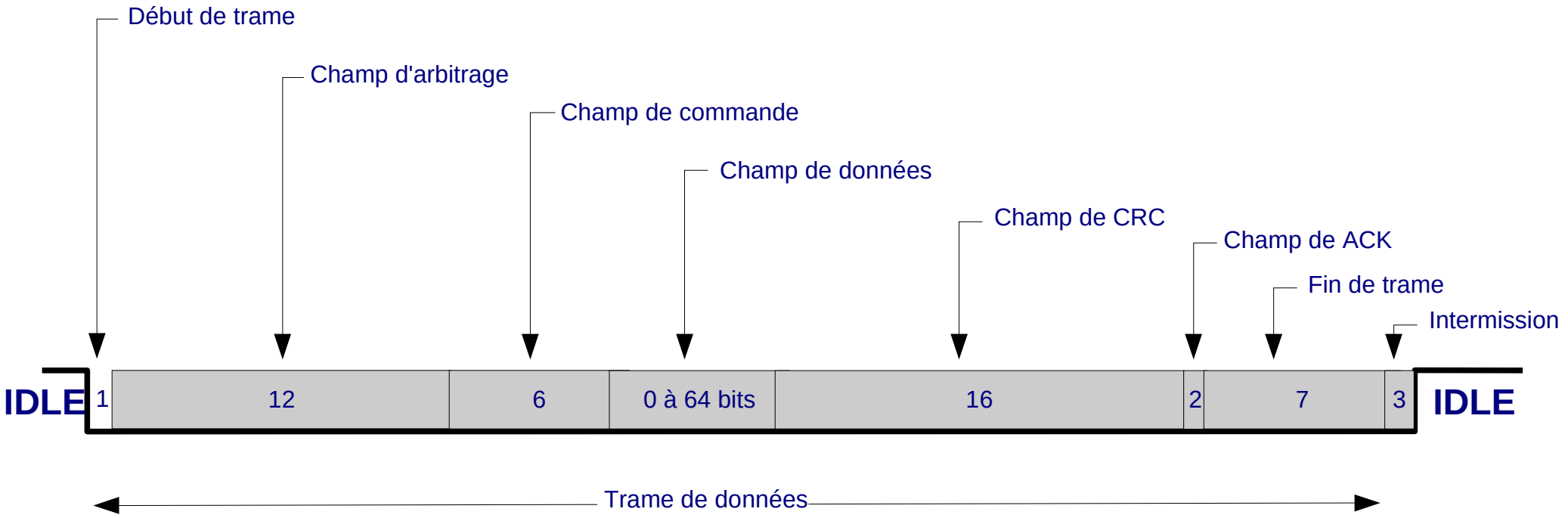
protocole de communication

- bus série : défini par sa durée (horloge)
- codage NRZ (Non Return to Zero) : pendant la durée du bit, sa valeur reste constante, qu'il soit dominant ou récessif
- bit stuffing : après le passage de 5 bits à la même valeur, on émet un bit supplémentaire de valeur opposée



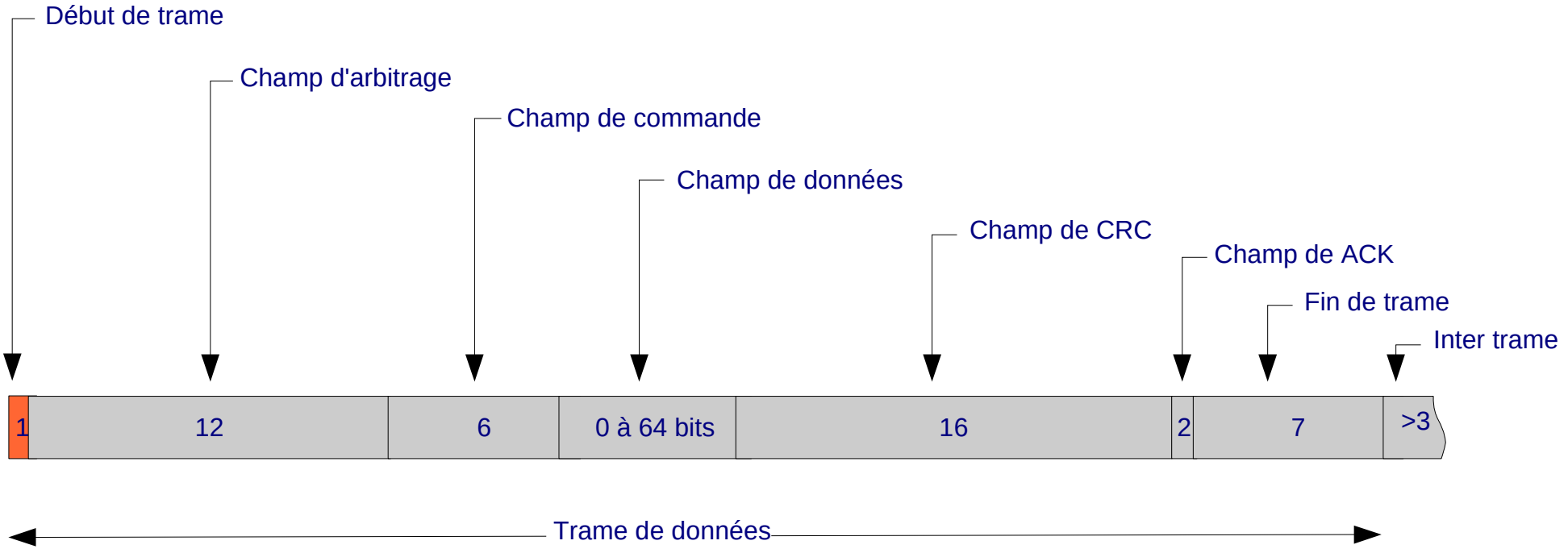
protocole de communication

- Data Frame



protocole de communication

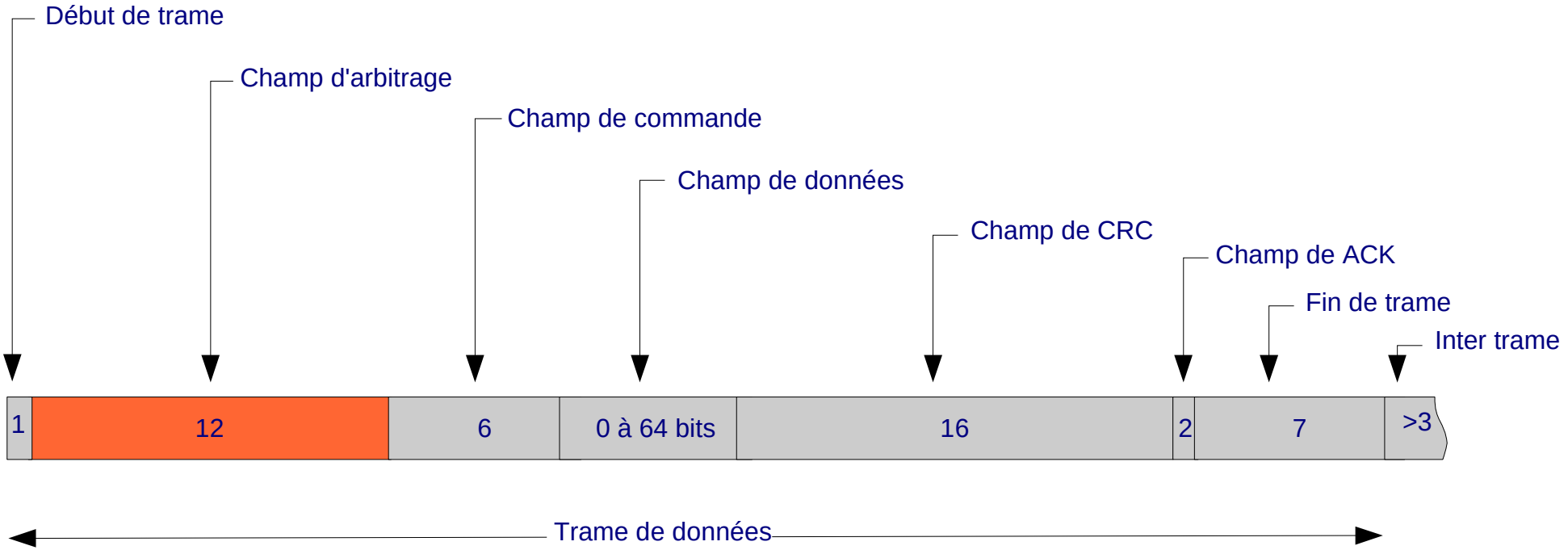
- Data Frame



  - ➢ début de trame :
    - ✔ 1 bit dominant signalant à tous le début d'un échange

protocole de communication

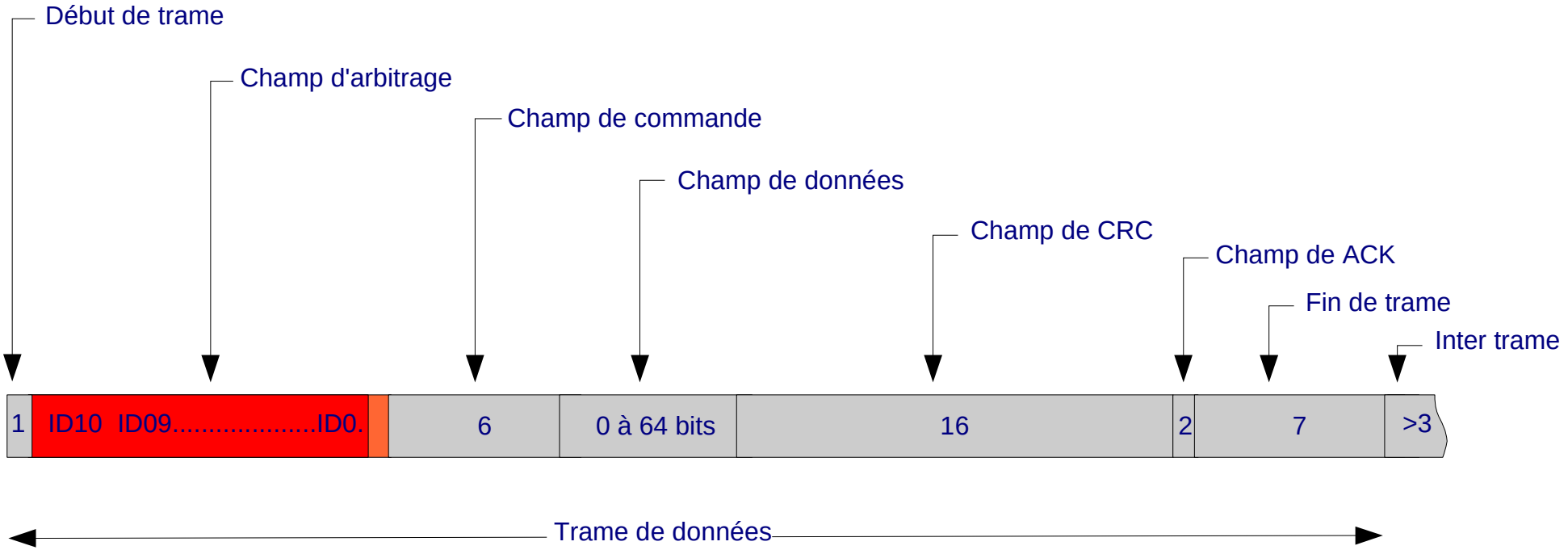
- Data Frame



  - Champ d'arbitrage :

protocole de communication

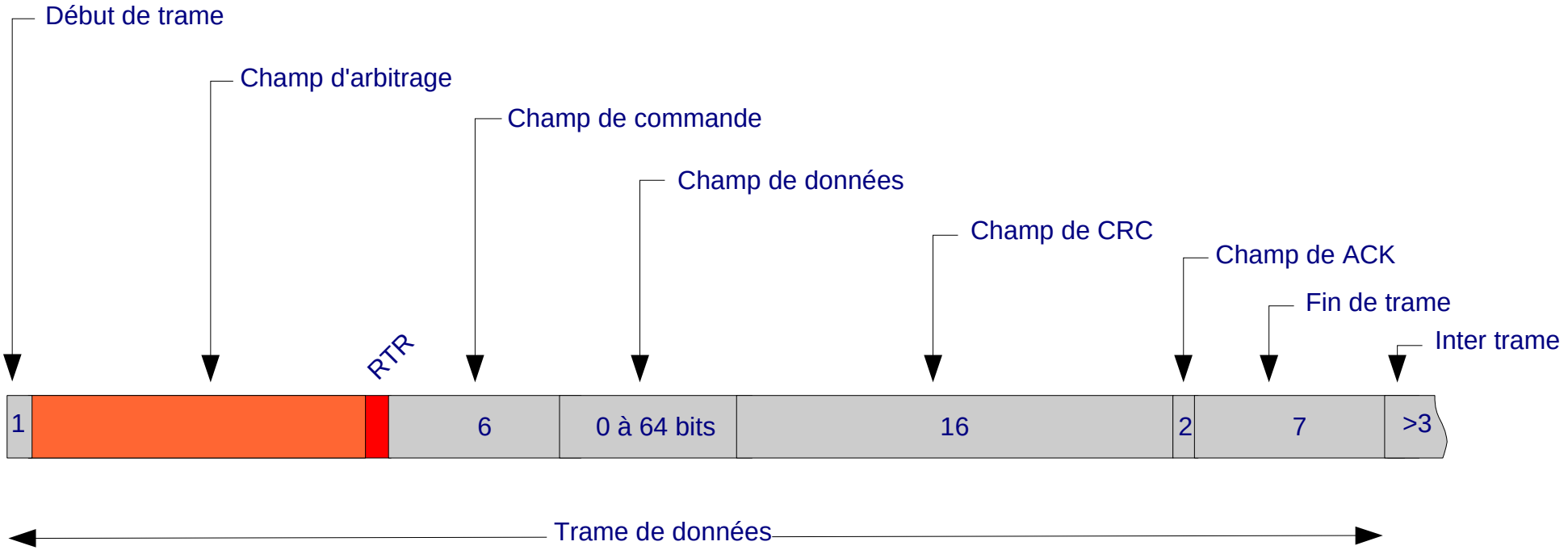
- Data Frame



- Champ d'arbitrage :
  - 11 bits pour l'identificateur
    - transmis de ID10 à ID0 (ID0 étant le bit le moins significatif)
    - les 7 bits les plus significatifs ne peuvent être tous récessifs

protocole de communication

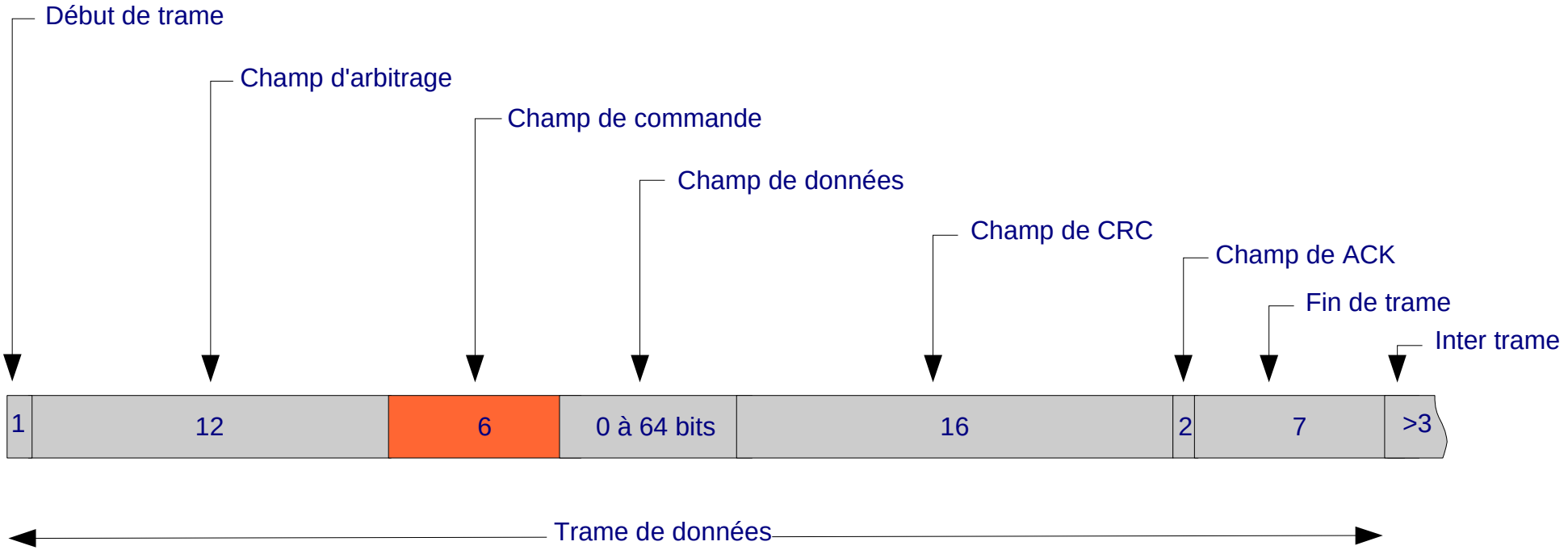
- Data Frame



  - Champ d'arbitrage :
    - 11 bits pour l'identificateur
    - 1 bit RTR (Remote Transmission Request)
      - dominant pour les trames de données
      - récessif pour les trames de requêtes

protocole de communication

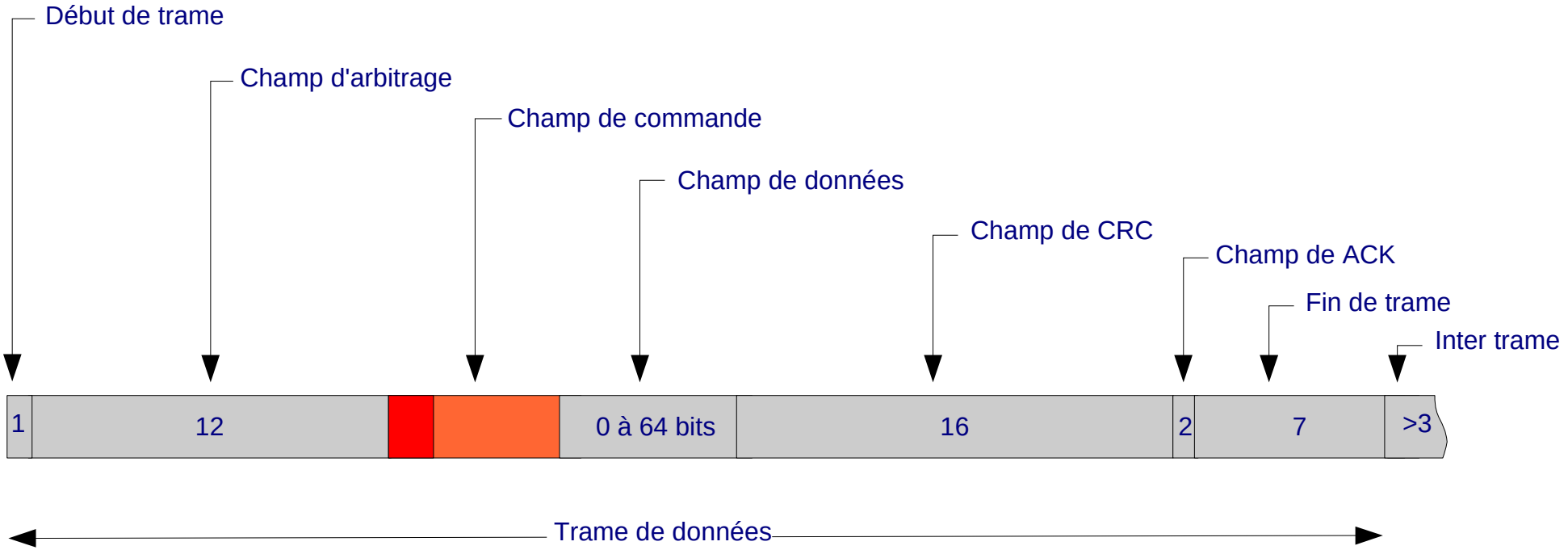
- Data Frame



  - Champ de commande :

protocole de communication

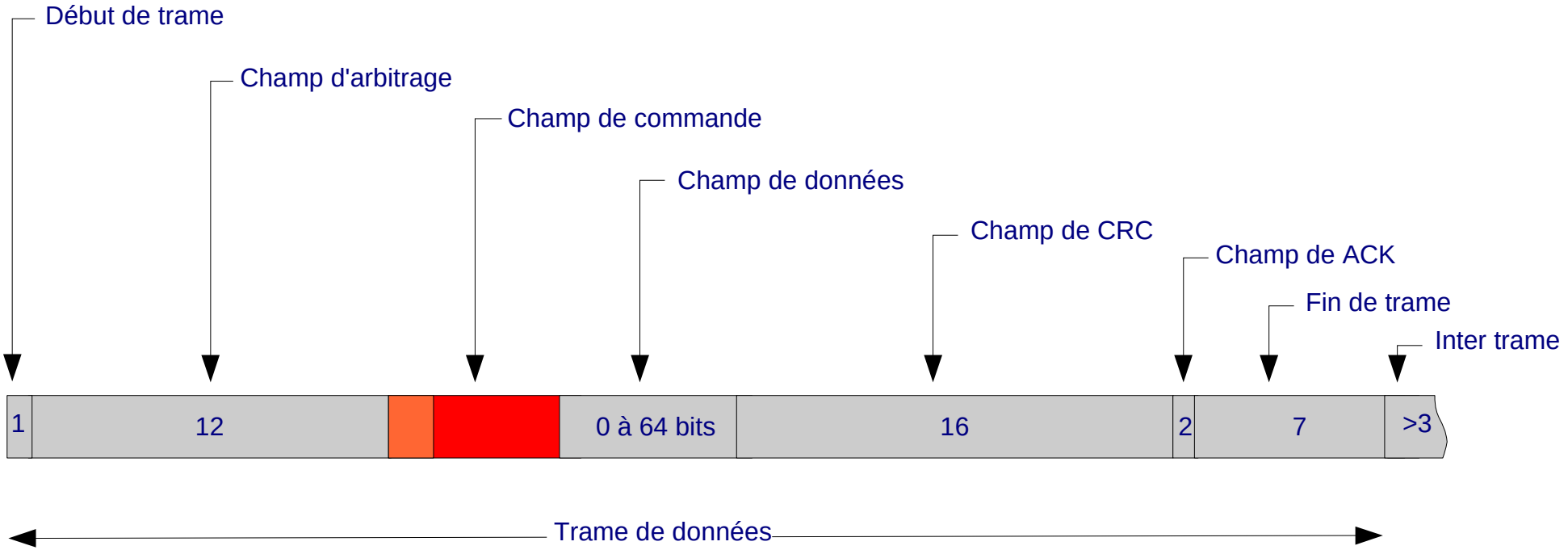
- Data Frame



  - Champ de commande :
    - 2 bits de réserve (compabilités avec la trame CAN2.0B et avec les futurs développements)

protocole de communication

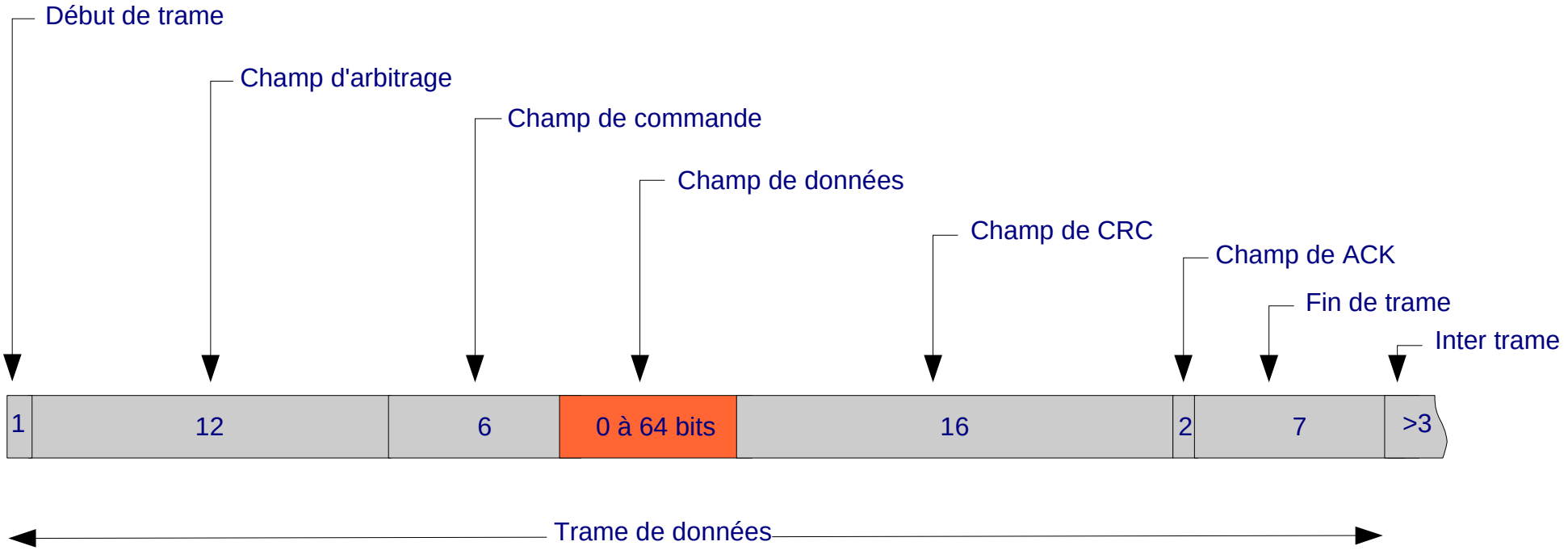
- Data Frame



  - Champ de commande :
    - 4 bits donnant le nombre d'octets contenus dans le champs de données

protocole de communication

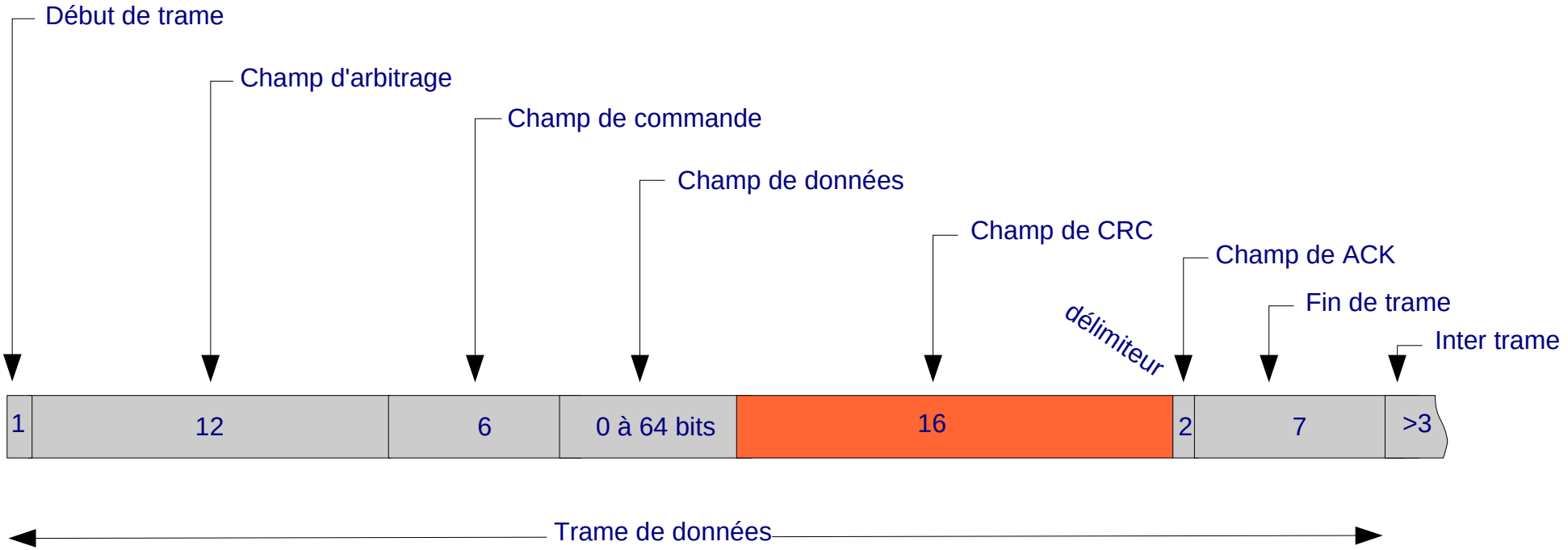
- Data Frame



  - Champ de données :
    - données utiles
    - constituées au plus de 8 octets
    - pour chaque octet, le bit le plus significatif est transmis le premier

protocole de communication

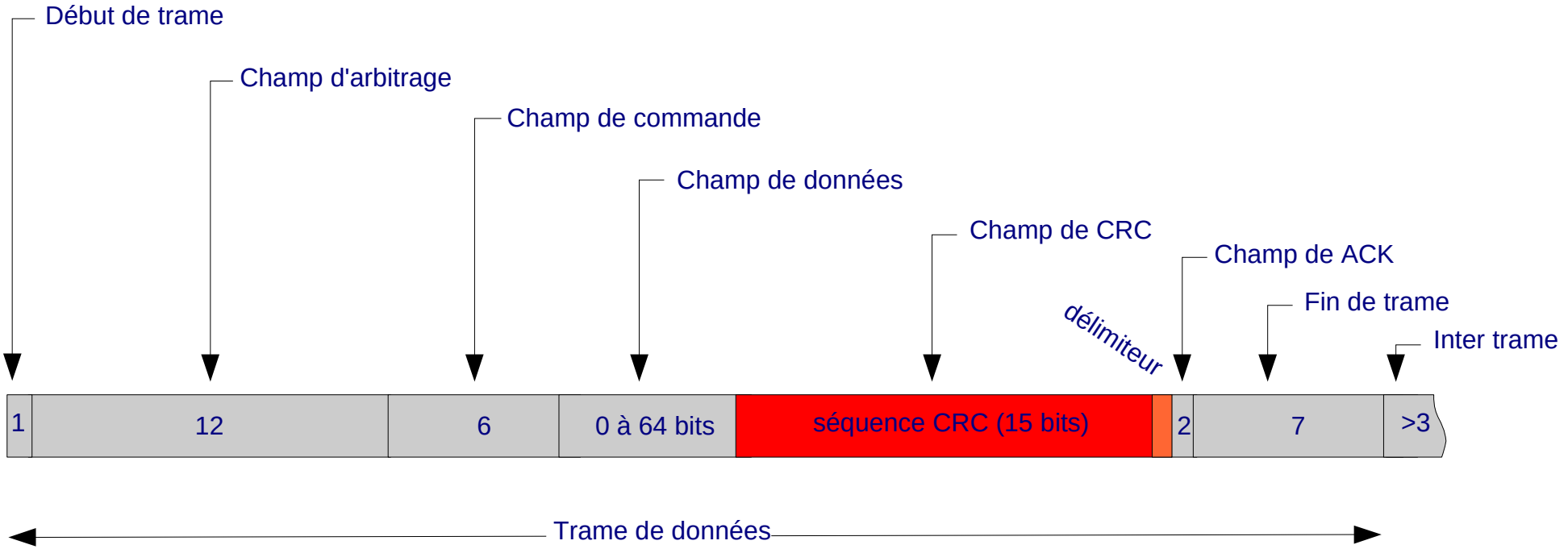
- Data Frame



  - Champ de CRC :

protocole de communication

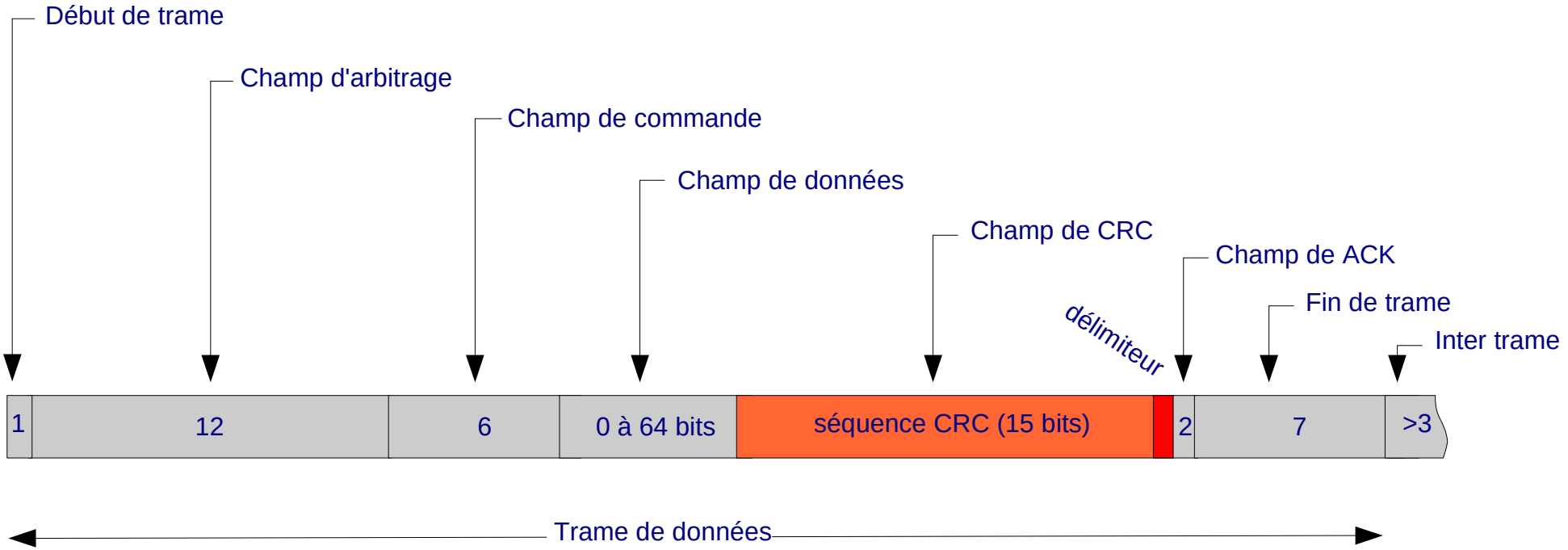
- Data Frame



  - Champ de CRC : séquence de 15 bits

protocole de communication

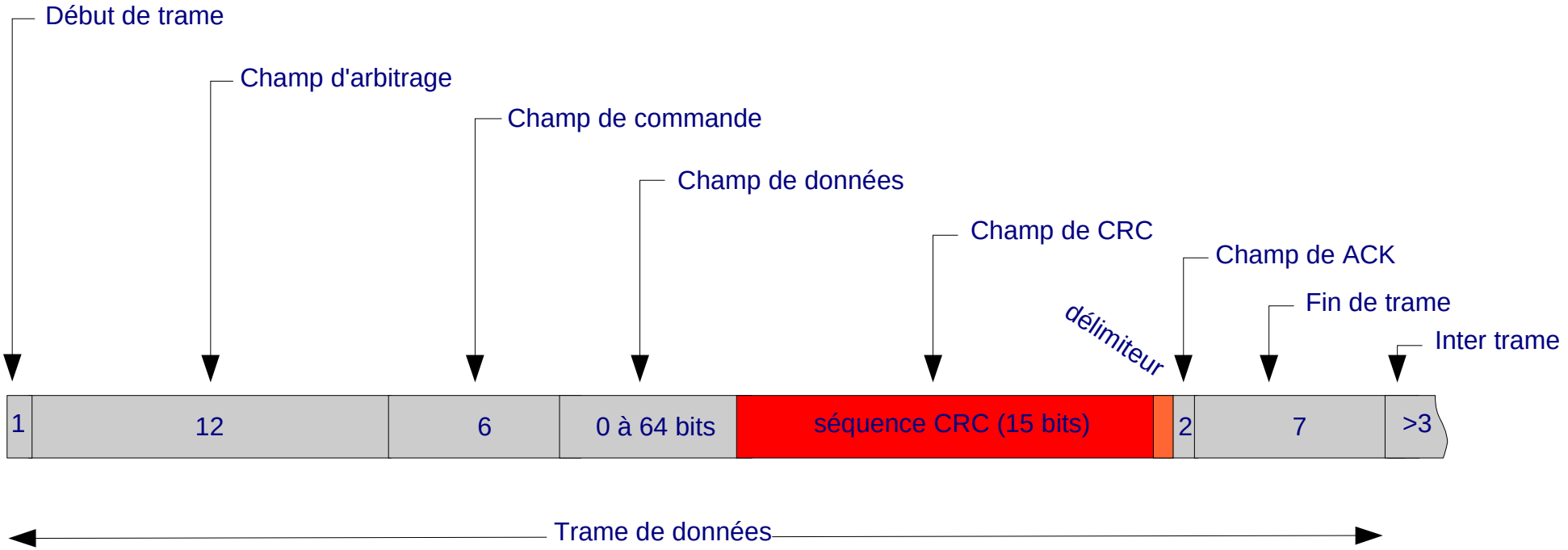
- Data Frame



- Champ de CRC : 15 bits + 1 bit délimiteur (récessif)

protocole de communication

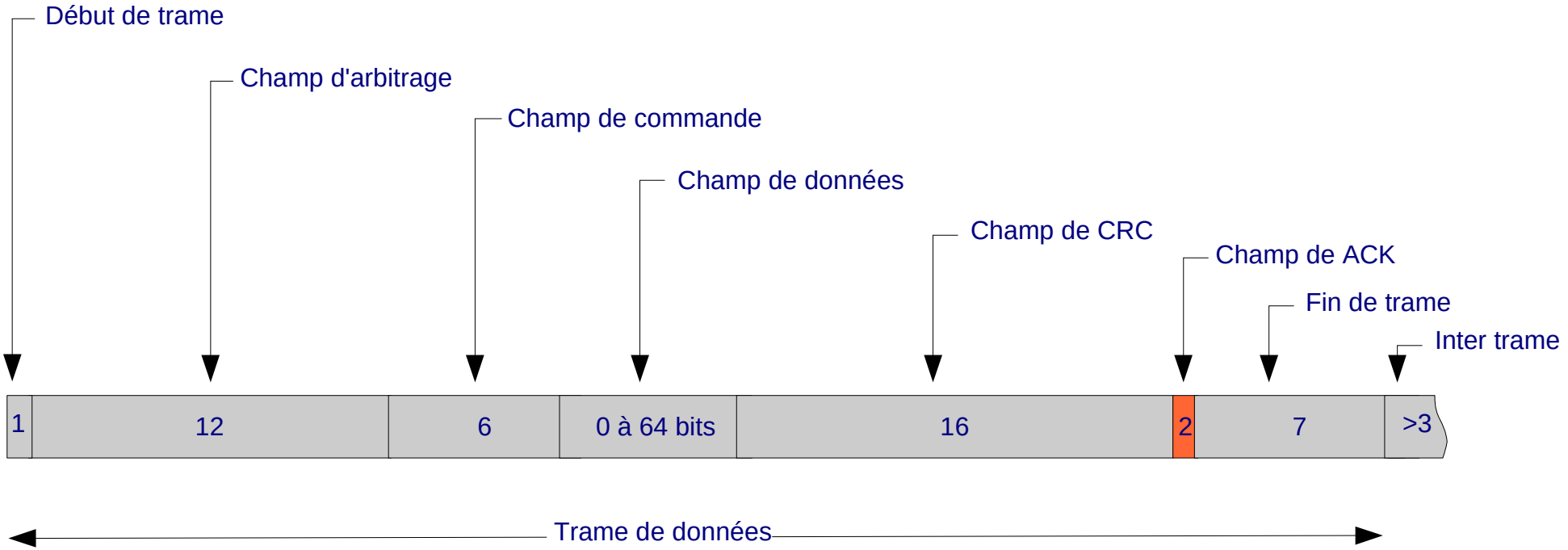
- Data Frame



- Champ de CRC : 15 bits + 1 bit délimiteur (récessif)
  - vérification de la validité du message transmis par un CRC suivant le code BCH
  - 5 erreurs de bits indépendantes détectables (probabilité d'erreur résiduelle : $2^{-15} \sim 3 \times 10^{-5}$

protocole de communication

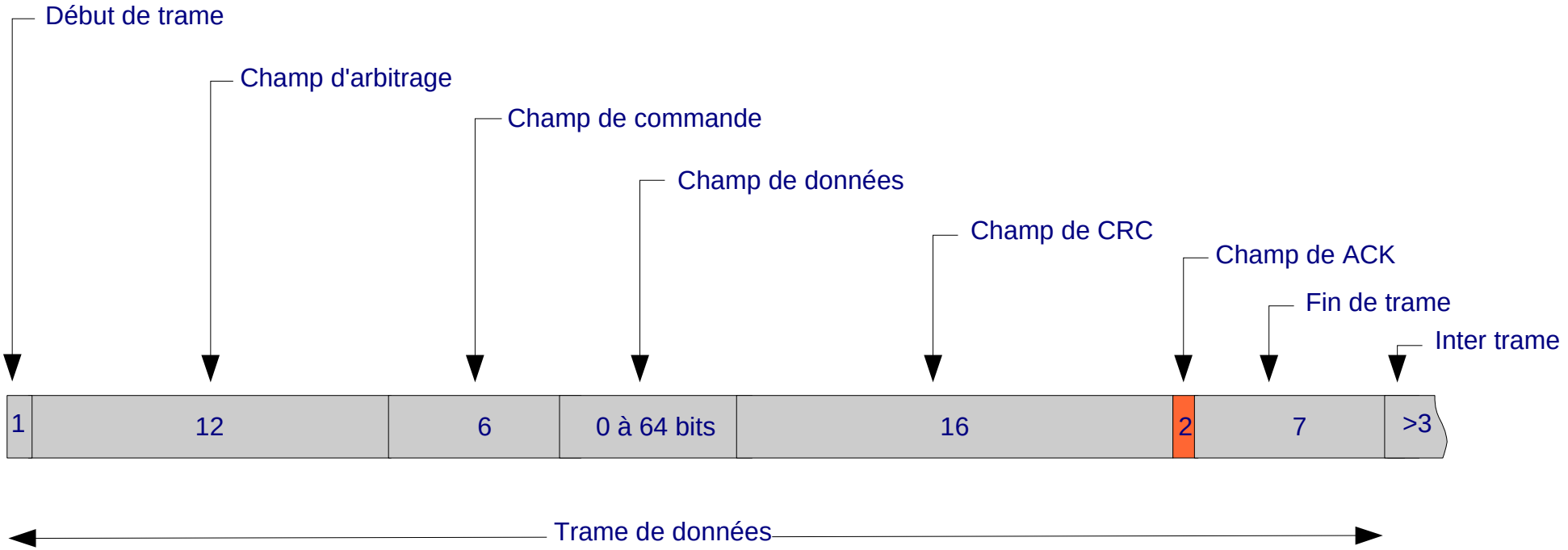
- Data Frame



- Champ de ACK : 1 bit ACK slot + 1 bit délimiteur
  - 2 bits récessifs émis par l'unité émettrice
  - toute unité qui a reçu correctement le message (CRC) superpose un bit dominant sur le bit ACK slot
    - Si OK, sur le bus : 1 bit dominant (ACK slot) entouré de 2 bits récessifs (CRC delimiter et ACK delimiter)

protocole de communication

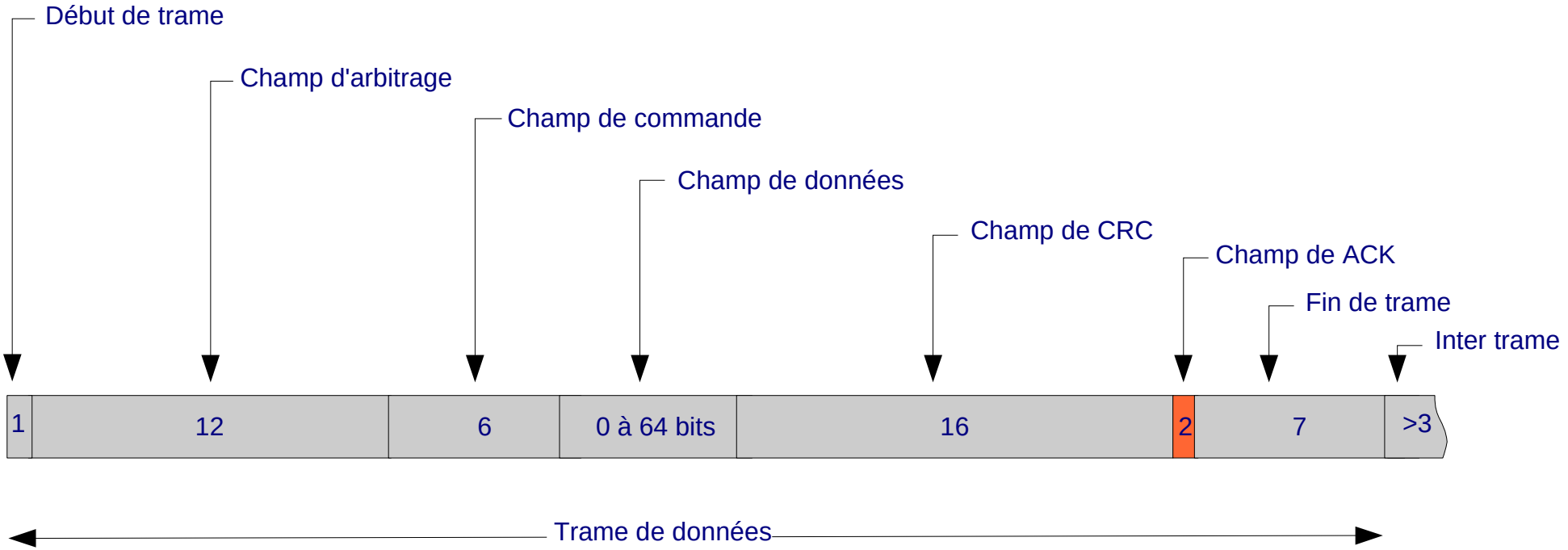
- Data Frame



- Champ de ACK : 1 bit ACK slot + 1 bit délimiteur
  - l'envoi de l'ACK ne signifie pas que le récepteur est intéressé par le message

protocole de communication

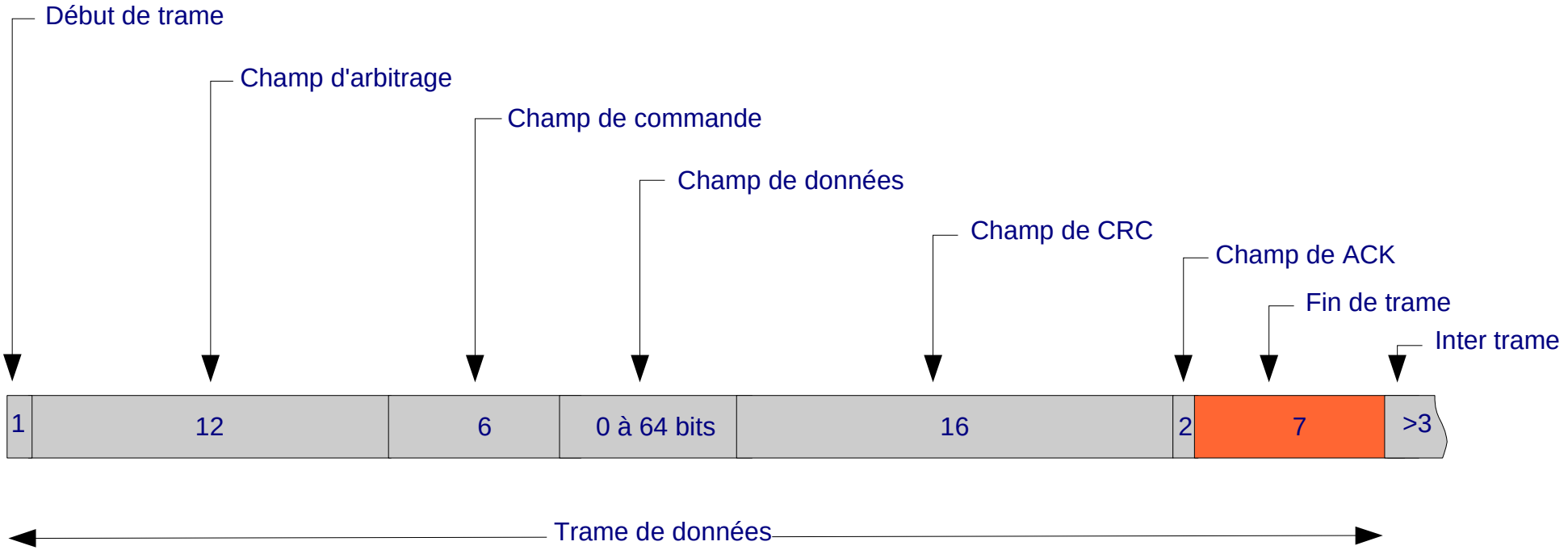
- Data Frame



- Champ de ACK : 1 bit ACK slot + 1 bit délimiteur
  - tout récepteur qui détecte une erreur et donc n'envoie pas d'acquitement, doit ensuite envoyer une trame d'erreur

protocole de communication

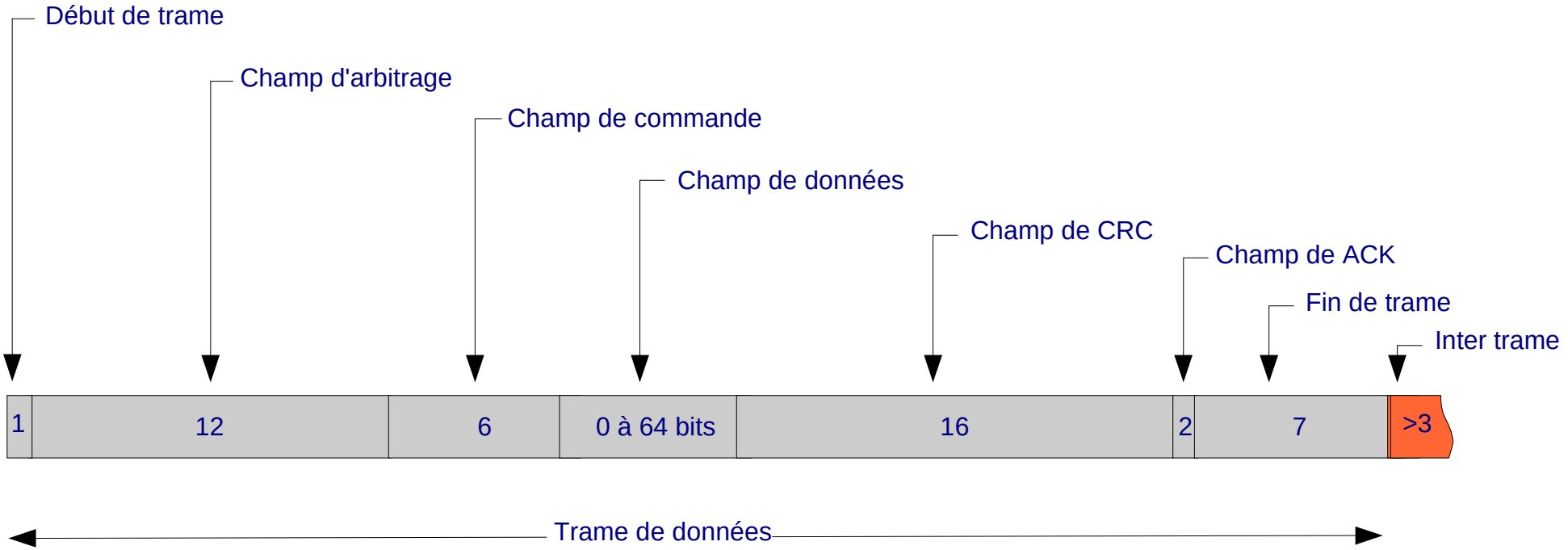
- Data Frame



- Champ de fin de trame :
  - 7 bits récessifs sans bit stuffing

protocole de communication

- Data Frame



  - Champ inter trame :
    - structure un peu complexe, pour améliorer la détection des erreurs ou des surcharges

protocole de communication

- Remote frame
  - pour demander la transmission de données par un autre nœud
  - comprend :
    - début de trame
    - champ d'arbitrage
    - champ de commande
    - champ de CRC
    - champ d'acquitement
    - fin de trame

protocole de communication

- Remote frame
  - à la différence de la trame de données, le bit RTR du champ d'arbitrage est récessif
    - à identificateur identique, la trame de donnée est prioritaire sur la trame de requête

  - les 4 derniers bits du champ de commande contiennent le nombre d'octets que devra contenir la trame de donnée satisfaisant la requête

protocole de communication

- détection des erreurs directement au niveau du message
  - utilisation d'un mécanisme de CRC (Cyclic Redundancy Check)
  - vérification de la structure globale de la trame
  - l'envoi d'un bit ACK par tous les receveurs qui fait qu'une erreur est détectée par l'émetteur si aucun ACK n'est reçu

protocole de communication

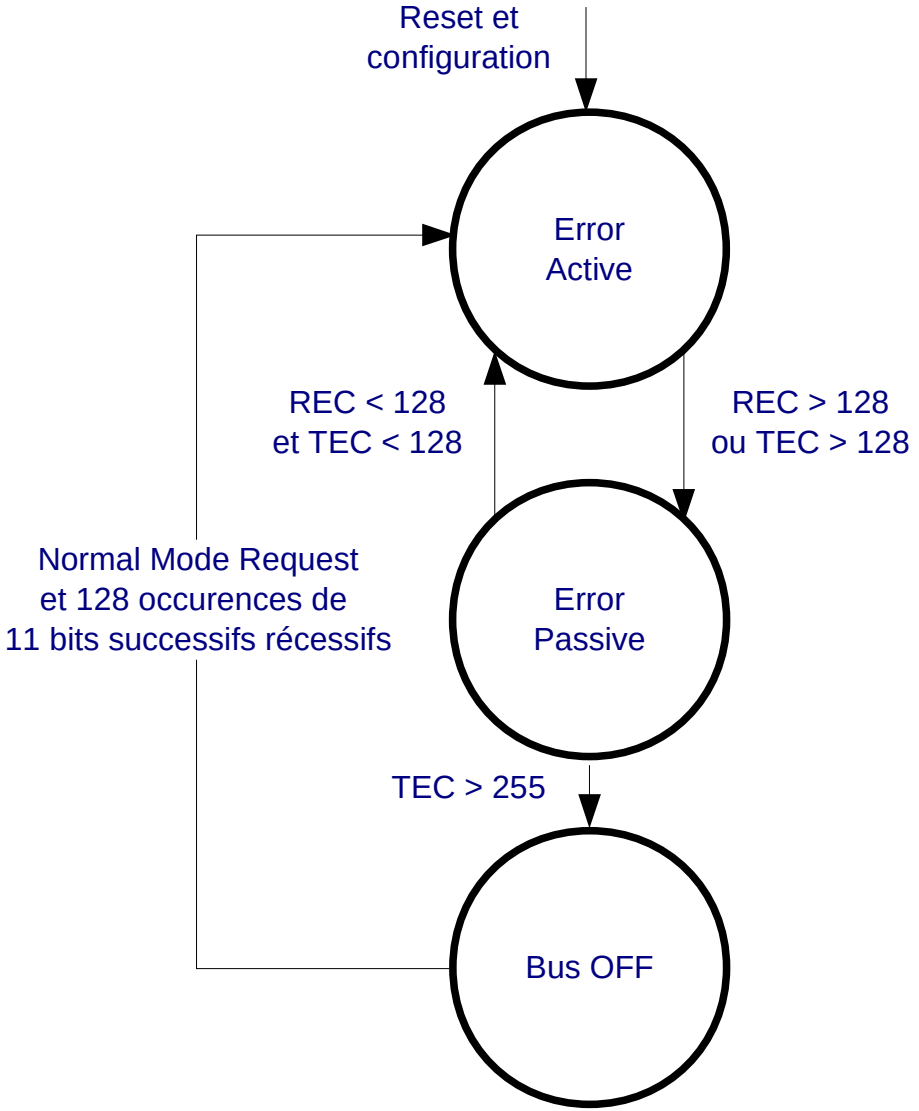
- détection des erreurs directement au niveau du message
- ou au niveau du bit
  - en monitorant au niveau des émetteurs les niveaux sur le bus et en comparant les bits émis et les bits reçus (permet en outre de vérifier la capacité de l'émetteur à détecter les erreurs)

protocole de communication

- détection des erreurs directement au niveau du message
- ou au niveau du bit
- les nombres d'erreurs détectées lors d'une transmission ou d'une réception sont stockés dans des compteurs (Transmit Error Counter et Receive Error Counter)
  - si le message est transmis ou reçu correctement, le compteur décroît
  - si une erreur est détectée, le compteur correspondant croît
  - une pondération de 8 est appliquée en défaveur de la décroissance des compteurs

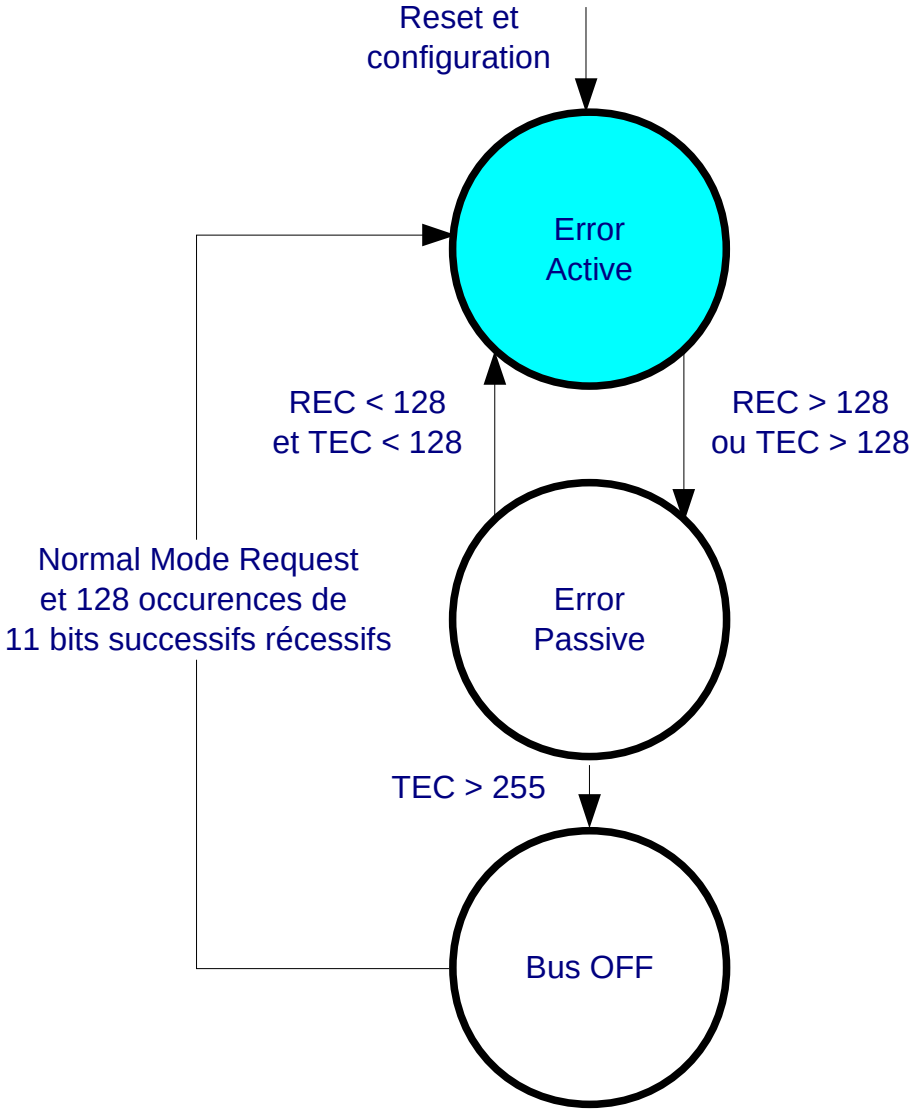
protocole de communication

- analyse des erreurs détectées



protocole de communication

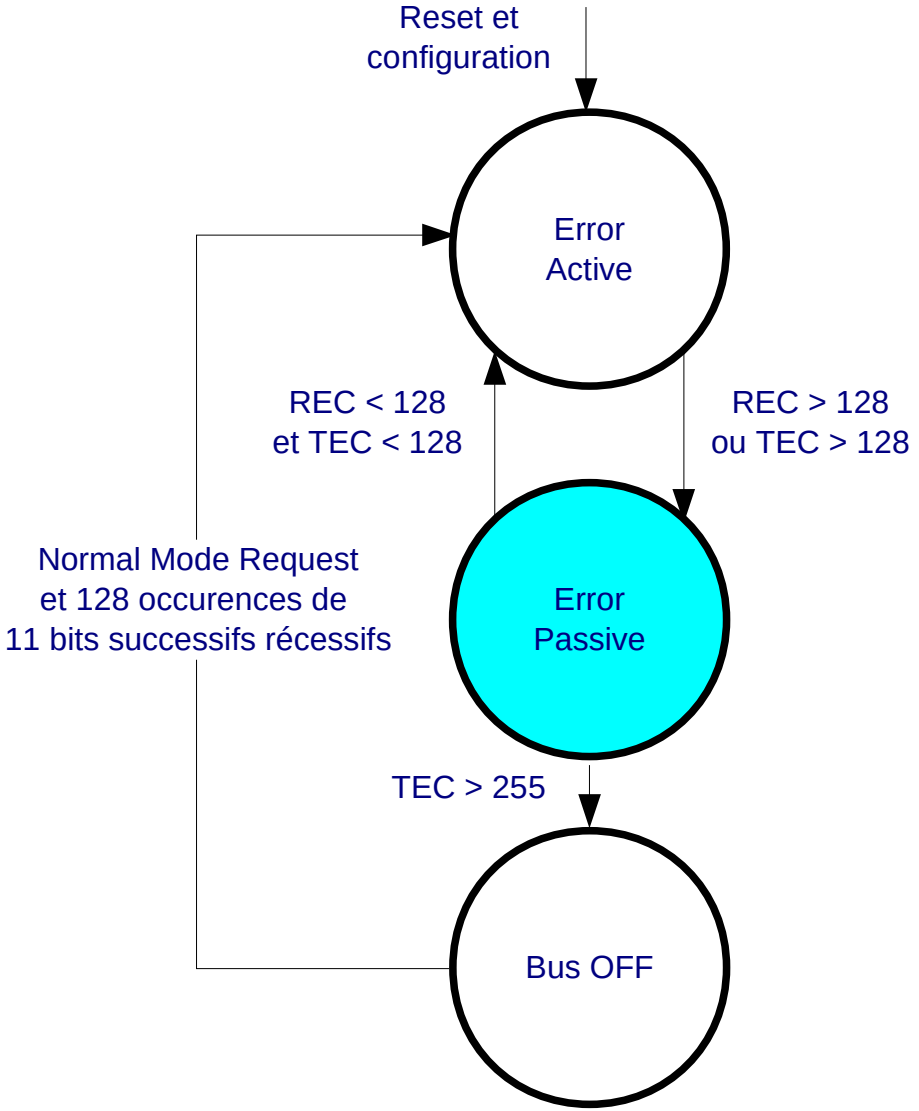
- analyse des erreurs détectées



- un nœud en état d'erreur active
  - ✔ continue de recevoir et d'émettre normalement
  - ✔ transmet un "active error flag" quand une erreur est détectée

protocole de communication

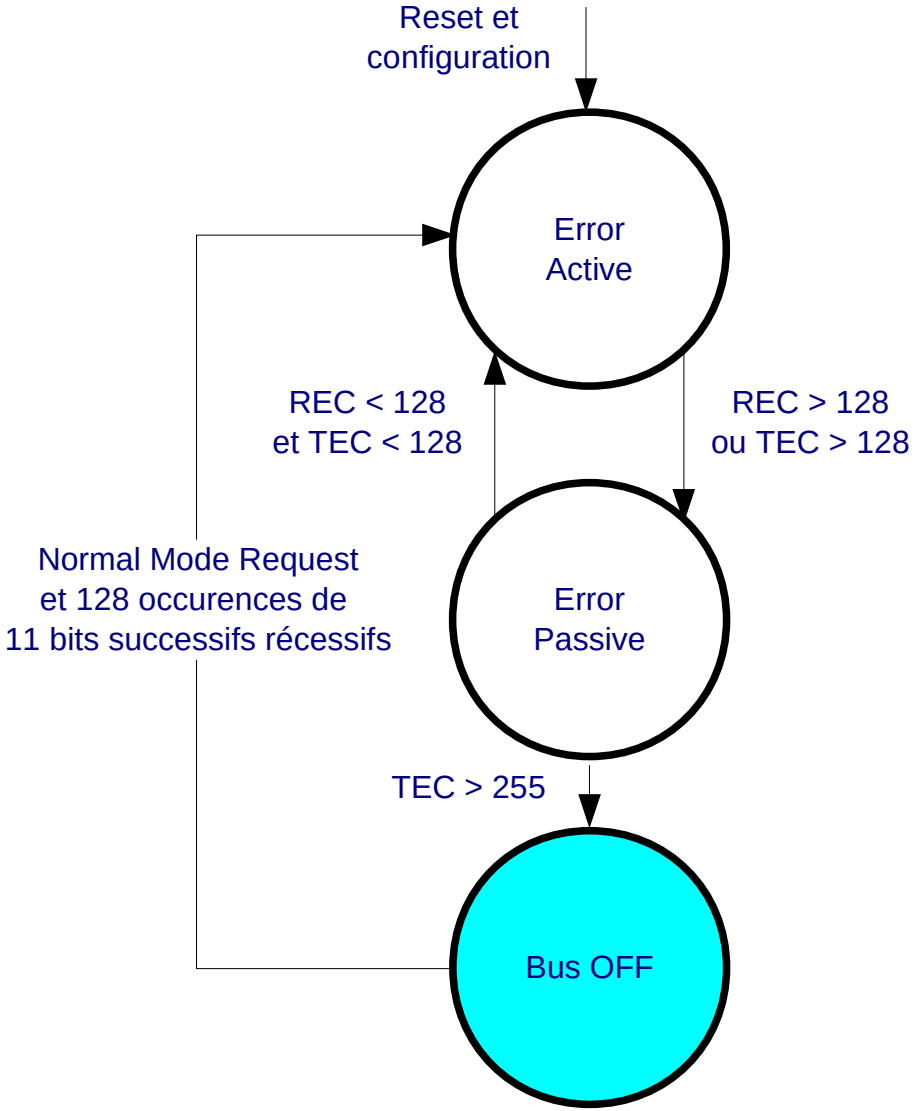
- analyse des erreurs détectées



- un nœud en état d'erreur passive
  - ✔ continue de recevoir et d'émettre normalement
  - ✔ transmet uniquement un "passive error flag" quand une erreur est détectée

protocole de communication

- analyse des erreurs détectées



- un nœud en état bus OFF
  - n'est plus autorisé à avoir une influence ou une activité sur le bus et cesse de recevoir et d'émettre normalement
  - les étages de commande du bus (drivers) doivent être électroniquement déconnectés
  - revient en mode error active quand 128 occurences de messages avec ACK delimiter, EoF et intermission OK sont passés

protocole de communication

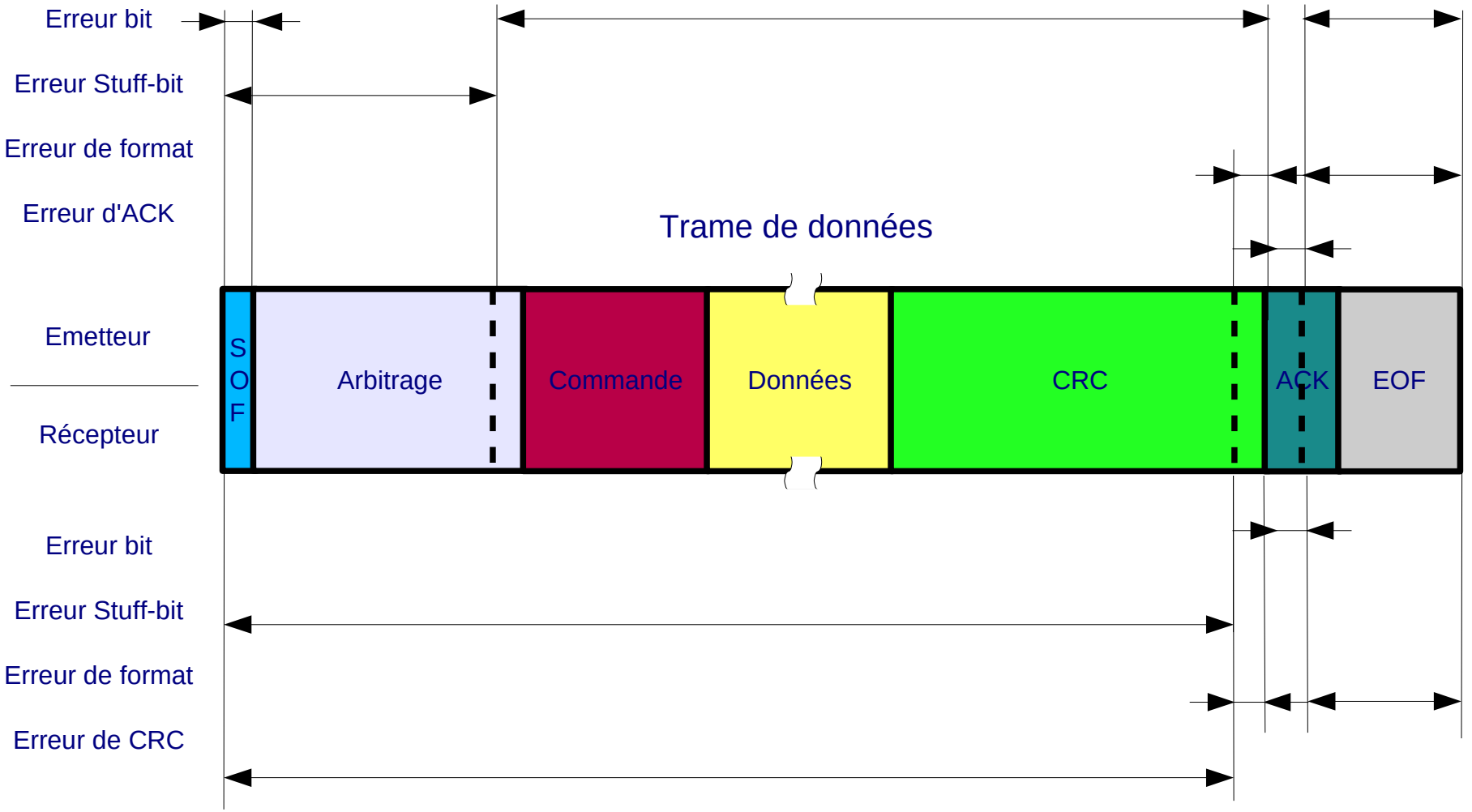
- analyse des erreurs détectées
  - il existe un certain nombre de règles du jeu pour le comptage et le décomptage des erreurs selon que c'est un nœud émetteur ou récepteur qui détecte la faute
  - ces règles font qu'un nœud fautif verra ses compteurs augmenter beaucoup plus vite que les autres noeuds et sera facilement repérable sur le réseau

protocole de communication

- détection des erreurs
  - bit entaché d'erreur (comparaison du bit à émettre et du bit émis)
  - erreur de bit stuffing
  - erreurs d'acquittement
  - erreurs de CRC
  - erreurs de structure (observation d'un bit dominant là où on attendait un bit récessif)
    - CRC delimiter
    - ACK delimiter
    - EoF
    - error delimiter
    - overload delimiter

protocole de communication

- détection des erreurs

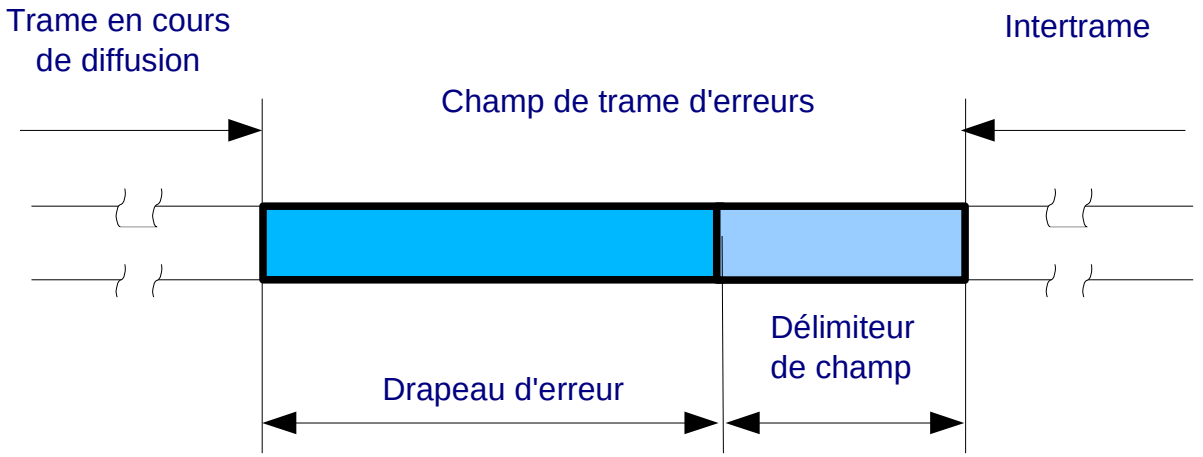


protocole de communication

- signalisation des erreurs
  - pour informer les autres participants de la détection
  - pour envoyer un état des compteurs locaux (émission d'un active – ou passive – error flag)
- mécanisme
  - émission d'une trame d'erreurs par la station détectrice
    - dès le bit suivant si l'erreur est due à un bit, un stuffing bit, une erreur de structure, une erreur d'acquittement
    - après le ACK delimiter si c'est une erreur de CRC

protocole de communication

- trame d'erreurs



  - champ des drapeaux d'erreurs
    - superposition des drapeaux d'erreurs auxquels les différentes stations présentes sur le bus ont contribué
  - active error flag : 6 bits dominants consécutifs
  - passive error flag : 6 bits récessifs consécutifs (éventuellement écrasés par des bits dominants provenant d'autre nœuds
  - délimiteur : 8 bits récessifs consécutifs

protocole de communication

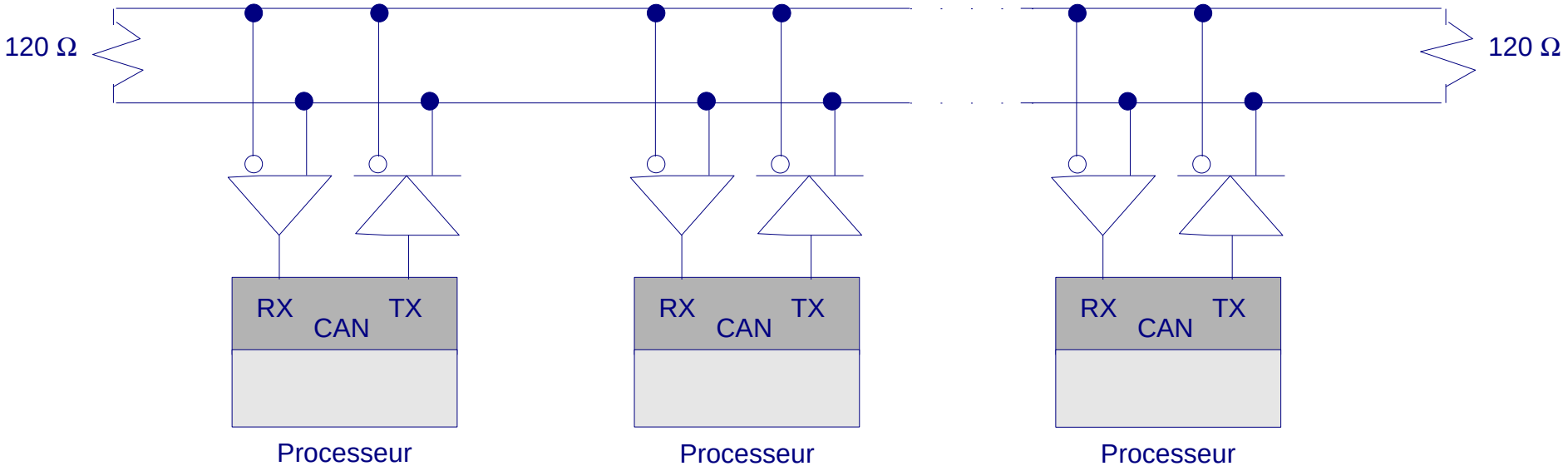
- récupération des erreurs
  - par réémission de la trame fautive
    - jusqu'à ce qu'elle passe correctement et qu'il n'y ait plus de message d'erreur
  - les stations déficientes seront automatiquement éliminées lorsque leur compteur d'erreur les mettra en bus off
  - si toutes les stations sont en bus off → système bloqué → reset manuel !

couche physique de communication

- CAN suit le modèle OSI et implémente directement (au niveau du matériel) les 2 couches inférieures
  - couche physique
  - couche de liens
- 2 options pour la couche physique
  - haute vitesse (ISO 11898-2)
  - basse vitesse, tolérante aux fautes (ISO 11898-3)

couche physique de communication

- option à haute vitesse
  - de 1 Mb/s pour une longueur maximale de 40 m
  - à ~ 50 kb/s pour une longueur maximale de 1 km
  - sur un bus en paire torsadée terminée par des résistances de charge de 120 Ω



couche physique de communication

- option à basse vitesse, tolérante aux fautes
  - plusieurs versions
    - ISO 11519-2
    - ISO 11992
    - ISO 11898-3
  - résistance de terminaison ~ 100 Ω (pas moins)
  - entre 20 et 32 stations sur un bus de 40 m maximum à 125 kb/s maximum

extensions de CAN

- TTCAN : time triggered CAN
  - décrit dans la norme ISO 11898-4
  - pour gérer l'ordonnancement de messages dont l'envoi sur un bus CAN est déclenché par le temps ou par des événements
    - applications de contrôle
    - amélioration des performances temps réel (diminution de la gigue dans la latence de transmission des messages, déterminisme du schéma de communication, tolérance aux fautes)

extensions de CAN : TTCAN

- basé sur l'utilisation d'un message de référence envoyé périodiquement par un maître du temps
- l'envoi du message de référence démarre un "cycle de base" qui comporte plusieurs fenêtres temporelles de tailles différentes durant lesquelles des messages peuvent être envoyés
  - fenêtres "exclusives" pour les messages périodiques, au début desquelles un message prédéfini est envoyé
  - fenêtres "arbitrées" pour les messages non périodiques (spontanés)
  - fenêtres "libres" pour éventuellement étendre le réseau
  - le pattern des fenêtres est défini au moment du design

extensions de CAN : TTCAN

- tolérance aux fautes grâce à la présence de plusieurs "maîtres du temps"
  - 1 seul maître effectif
  - plusieurs maîtres potentiels
  - accès à la maîtrise suivant la priorité du message de référence
  - tentative de prise de la maîtrise à l'occurence d'un timeout sur la réception du message de référence
- possibilité de changer dynamiquement la base de temps, ou même de passer temporairement à une structure apériodique (event driven) grâce à un message de référence spécial

extensions de CAN : CANopen

- couche de haut niveau construite au dessus de CAN
- initialement un projet ESPRIT, contrôlé depuis 1995 par le consortium CiA (CAN in Automation)
- facilite le travail des concepteurs en proposant
  - des objets pour la communication (Process Data Objects)
  - des objets pour la configuration (Service Data Objects)
  - des objets pour la gestion du temps (réel)
  - des objets pour la gestion du réseau
  - des couches applicatives
- permettant de ré-utiliser le software et le hardware

autres extensions de CAN

- développements spécifiques
  - CAN Kingdom (contrôle de machines)
    http://www.cankingdom.org/
  - DeviceNet (automatisation d'usines)
    http://www.odva.org/

Bus VME

le bus VME

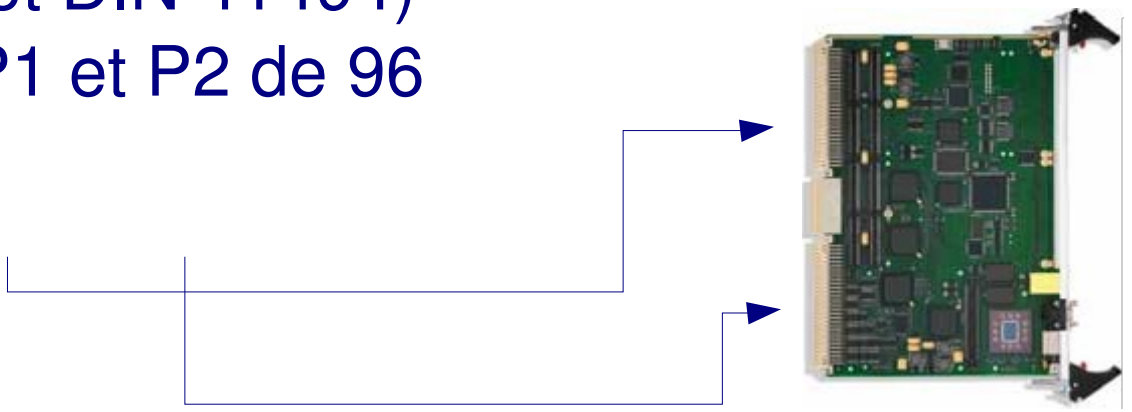
- Versa Module Eurocard
- standard industriel né au début des années 1980
  - au début pour les cartes basées sur des processeurs Motorola 68000
- très utilisé dans les applications mettant en jeu l'acquisition et le traitement des données
- norme passée dans le domaine public (pas de licence pour un bus propriétaire)
- VME International Trade Union http://www.vita.com

le standard VME

- défini dans la norme
  IEEE 1014-87 ou IEC 821-297
- conçu par des informaticiens industriels pour des informaticiens industriels
- objectifs :
  - communication entre deux unités sans gêner le fonctionnement interne des autres
  - système dont les performances dépendent essentiellement des unités et pas du bus lui-même
  - grande latitude dans le choix des moyens d'optimisation des coûts et des performances sans nuire à la compatibilité

le standard VME

- bus multiprocesseur asynchrone:
  - plusieurs cartes maîtresses peuvent accéder à des ressources communes (cartes mémoire, modules d'E/S, etc...)
  - cadencé par un mécanisme de handshake
- cartes au format "double Europe" (norme DIN 41612 et DIN 41494) sur 2 connecteurs P1 et P2 de 96 broches



le standard VME

- Connecteur P1
  - un bus de données de 16 bits
  - un bus d'adresses de 24 bits
  - 6 lignes de modification d'adresses
  - plusieurs lignes d'alimentation et de contrôle

le standard VME

Pin Assignment for the VMEbus P1/J1 Connector			
Pin	**Row a**	**Row b**	**Row c**
1	D00	BBSY*	D08
2	D01	BCLR*	D09
3	D02	ACFAIL*	D10
4	D03	BG0IN*	D11
5	D04	BG0OUT*	D12
6	D05	BG1IN*	D13
7	D06	BG1OUT*	D14
8	D07	BG2IN*	D15
9	GND	BG2OUT*	GND
10	SYSCLCK	BG3IN*	SYSFAIL*
11	GND	BG3OUT*	BERR*
12	DS1*	BR0*	SYSRESET*
13	DS0*	BR1*	LWORD*
14	WRITE*	BR2*	AM5
15	GND	BR3*	A23
16	DTACK*	AM0	A22

Pin Assignment for the VMEbus P1/J1 Connector			
Pin	**Row a**	**Row b**	**Row c**
17	GND	AM1	A21
18	AS*	AM2	A20
19	GND	AM3	A19
20	IACK*	GND	A18
21	IACKIN*	GND	A17
22	IACKOUT*	GND	A16
23	AM4	GND	A15
24	A07	IRQ7*	A14
25	A06	IRQ6*	A13
26	A05	IRQ5*	A12
27	A04	IRQ4*	A11
28	A03	IRQ3*	A10
29	A02	IRQ2*	A09
30	A01	IRQ1*	A08
31	-12 VDC	+5 VSTBY	+12 VDC
32	+5 VDC	+5 VDC	+5 VDC

* indique que le signal est activé quand le niveau est bas

- voir http://www.vita.com/vmefaq/vmepins.html

le standard VME

- Connecteur P2
  - extensions du bus de données (D15-D31) et du bus d'adresses (A24-A31)
  - extensions définies par l'utilisateur
    - VSB
    - PCI
    - autres...
- une extension du bus pour les processeurs 64 bits a été standardisée
  - utilise des connecteurs P1 et P2 à 160 broches
  - plus un connecteur optionnel P0 dédié aux E/S

organisation du bus VME

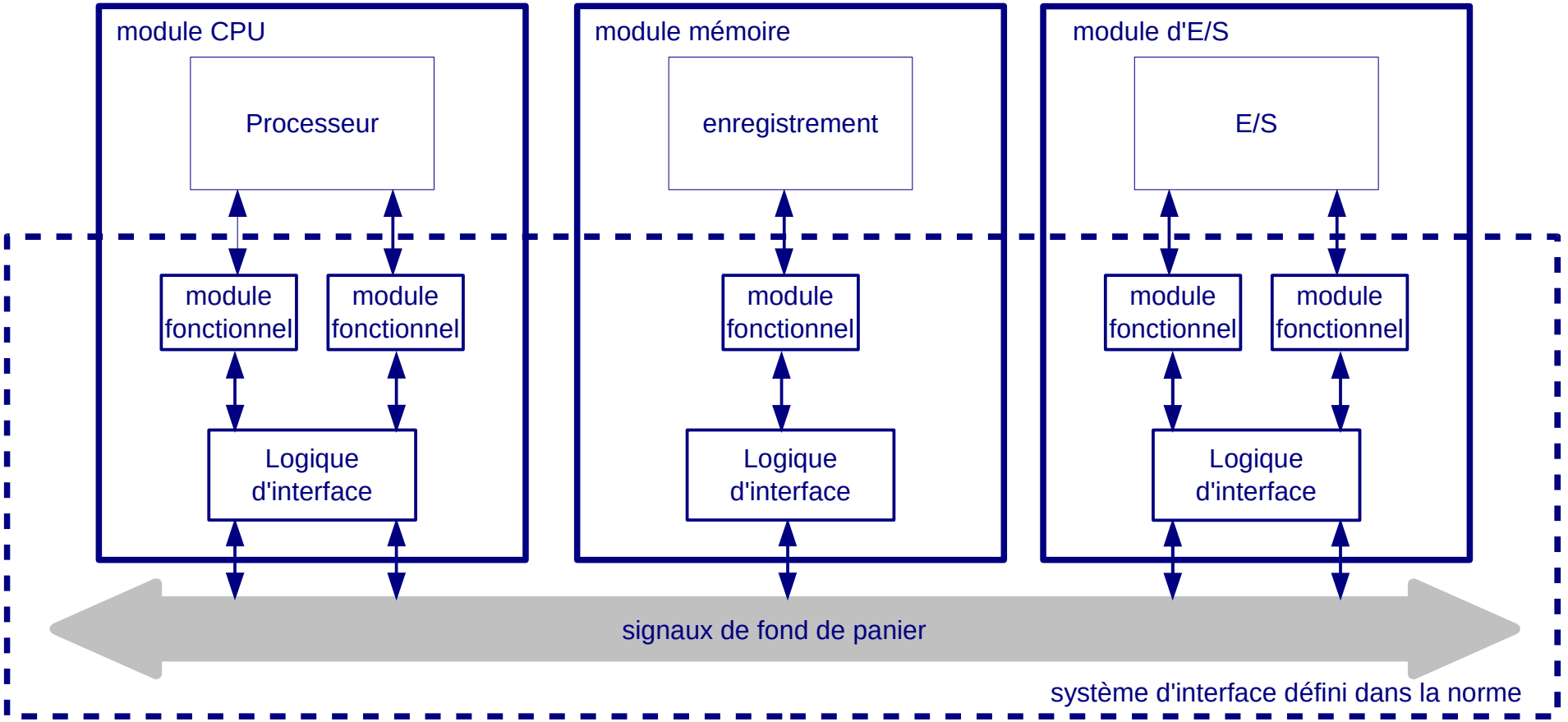
- transfert de données en mode maître-esclave
- asynchrone
- non multiplexé
- transferts sur des largeurs de mots de 8, 16 ou 32 bits
- 7 niveaux de priorité d'interruption
- 4 niveaux d'arbitrage du bus de données
- lignes spécialisées pour
  - détection rapide de défauts
  - contrôle de bus
  - contrôle de l'alimentation électrique

organisation du bus VME

- bus VME logiquement organisé en 5 sous-bus
  - Data Transfer Bus
  - Data Transfer Arbitration Bus
  - Priority interrupt Bus
  - Utility bus
  - Serial Bus

organisation du bus VME

- décrit en termes de modules fonctionnels qui servent à gérer l'utilisation du bus de transfert de données (DTB)
  - conceptuels, parfois associés à du hardware



organisation du bus VME

- fonctions d'interfaces
  - transfert de données
    - les modules maîtres peuvent initier des transferts de données sur le DTB
    - les modules esclaves peuvent détecter les transferts et y participer
    - une carte peut posséder à la fois un module maître et un module esclave
  - arbitrage du DTB
    - plusieurs maîtres peuvent coexister dans un système VME, un seul à la fois peut avoir le contrôle du bus
    - module pour les fonctions de demande (requester)
    - module pour les fonctions d'arbitrage (arbiter)

organisation du bus VME

- gestion des interruptions
  - une ressource peut demander une interruption de l'activité normale
  - module pour la génération de l'interruption (interrupter)
  - module pour le traitement (interrupt handler)
- utilitaires
  - horloge
  - (ré)initialisation
  - défauts du système
  - défaut d'alimentation
  - lignes série de communication

transfert des données

- par l'intermédiaire d'un bus spécialisé (DTB)
  - bus d'adresses
    - bus de 32 bits A0 à A31
    - adressage sur 16, 24 ou 32 bits
    - espace mémoire adressable minimum : 1 octet
  - bus de données
    - 3 largeurs de mots : 8/16/32 bits (D8/D16/D32)
  - bus de modification d'adresse
    - AM0 à AM5
  - lignes de contrôle de transfert par le maître
  - lignes d'état contrôlées par l'esclave

transfert des données

- lignes de modification d'adresse
  - pour permettre au maître d'envoyer des informations supplémentaires lors d'un transfert :
    - configuration dynamique du système, en imposant à l'esclave de ne réagir qu'à un seul code
    - sélection dynamique de l'emplacement de l'esclave dans l'espace d'adressage du maître
    - sélection du domaine d'adressage (*court* sur 16 bits, *standard* sur 24 bits, *long* sur 32 bits)
    - modification dynamique des privilèges nécessaires pour accéder à certains niveaux d'exécution de l'esclave
    - modification du type de transfert (par blocs)

transfert des données

- modificateurs

AM5	AM4	AM3	AM2	AM1	AM0	fonctions	défini par
1	1	1	1	1	1	Accès ascendant en mode superviseur standard	Spécifié bus VME
1	1	1	1	1	0	Accès au programme en mode superviseurs standard	Spécifié bus VME
1	1	1	1	0	1	Accès aux données en mode superviseur standard	Spécifié bus VME
1	1	1	1	0	0	non défini	réservé
1	1	1	0	1	1	Accès ascendant en mode utilisateur standard	Spécifié bus VME
1	1	1	0	1	0	Accès au programme en mode utilisateur standard	Spécifié bus VME
1	1	1	0	0	1	Accès aux données en mode utilisateur standard	Spécifié bus VME
1	1	1	0	0	0	Non défini	réservé
1	1	0	X	X	X	Non défini	réservé
1	0	1	1	1	1	Non défini	réservé
1	0	1	1	1	0	Non défini	réservé
1	0	1	1	0	1	Accès entrée/sortie en mode superviseur court	Spécifié bus VME
1	0	1	1	0	0	Non défini	réservé
1	0	1	0	1	1	Non défini	réservé
1	0	1	0	1	0	Non défini	réservé
1	0	1	0	0	1	Accès entrée/sortie en mode utilisateur court	Spécifié bus VME
1	0	1	0	0	0	Non défini	réservé
1	0	0	x	x	x	Non défini	réservé
0	1	x	x	x	x	Non défini	utilisateur
0	0	1	1	1	1	Accès ascendant en mode superviseur étendu	Spécifié bus VME
0	0	1	1	1	0	Accès au programme en mode superviseur étendu	Spécifié bus VME
0	0	1	1	0	1	Accès aux données en mode superviseur étendu	Spécifié bus VME
0	0	1	1	0	0	Non défini	réservé
0	0	1	0	1	0	Accès ascendant en mode utilisateur étendu	Spécifié bus VME
0	0	1	0	1	0	Accès au programme en mode utilisateur étendu	Spécifié bus VME
0	0	1	0	0	1	Accès aux données en mode utilisateur étendu	Spécifié bus VME
0	0	1	0	0	0	Non défini	réservé
0	0	0	x	x	x	Non défini	réservé

transfert des données

- lignes de contrôle et d'état
  - 4 lignes de contrôle pour le module maître
    - $\overline{AS}$: address strobe
    - $\overline{DS0}$: data strobe 0 : sélection l'octet de poids faible
    - $\overline{DS1}$: data strobe 1 : sélection l'octet de poids fort
    - $\overline{LWORD}$: accès à un mot de 32 bits
    - $\overline{WRITE}$: direction du transfert (niveau bas : écriture maître $\rightarrow$ esclave)
  - 2 lignes d'état contrôlées par le module esclave
    - $\overline{DTACK}$: data acknowledge (transfert réussi)
    - $\overline{BERR}$: erreur dans le transfert en cours

transfert des données

- bus d'adresses
  - pour accéder à un octet de mémoire par le module maître
    - valeur de l'adresse sur le bus d'adresse A02-A32
    - utilisation des lignes DS0, DS1, LWORD et A01 pour le mode d'accès
- lignes de données
  - sur les lignes D0 à D15
  - les lignes DS0, DS1, A01 et LWORD permettent de choisir l'octet, la longueur du transfert et l'alignement

transfert des données

- exemple de lecture

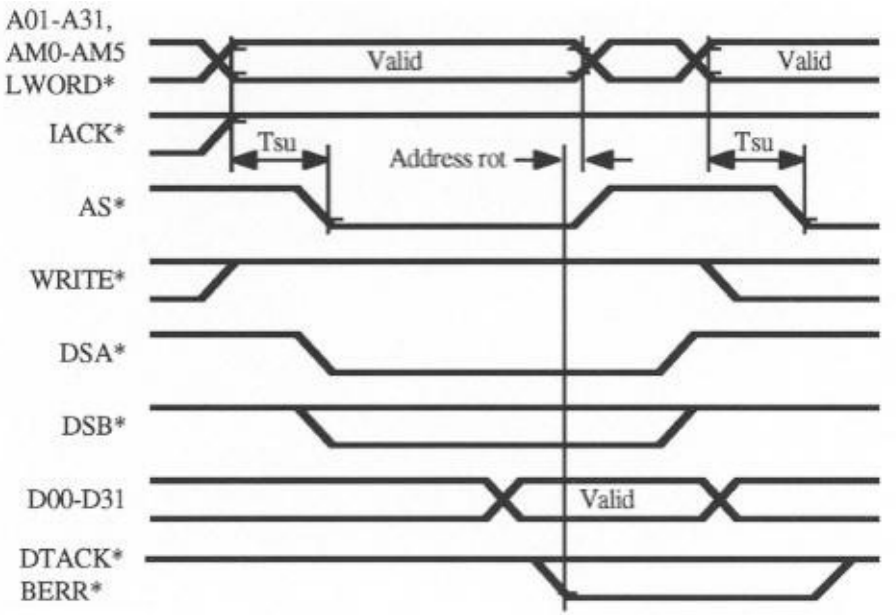
1) le module maître utilise les lignes A01-31, AM0-AM5 et LWORD pour choisir l'esclave et le mode

2) le maître désactive IACK pour indiquer que ce n'est pas un cycle en réponse à un interrupt

3) le maître active AS pour valider l'adresse

4) le maître désactive WRITE (niveau haut) pour indiquer un cycle de lecture

5) le maître active les lignes DS0 et/ou DS1 pour indiquer où (sur D0 à D31) il veut lire les données



6) après un certain temps, l'esclave active les lignes de données D0 à D31

7) quand les lignes D0-D31 sont stables, l'esclave active DTACK (ou BERR) pour signaler le succès (ou l'échec) du transfert

8) après un certain temps, le maître lit les lignes de données

9) le maître désactive les lignes DS0 et DS1 pour indiquer que les données ont été lues

10) l'esclave désactive DTACK pour indiquer qu'il a terminé avec le cycle

arbitrage pour l'accès au bus

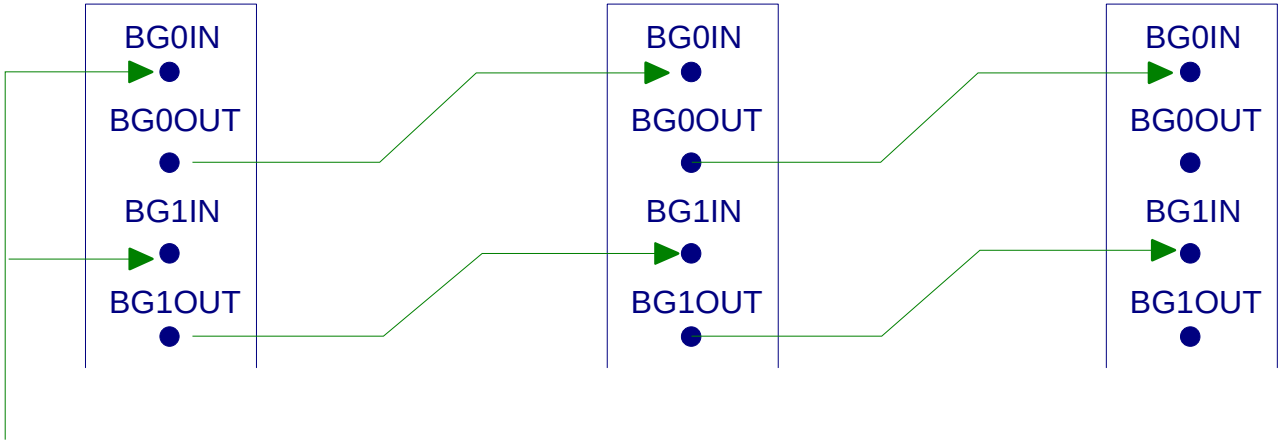
- plusieurs modules maîtres peuvent coexister dans le même chassis
- un seul maître peut posséder le contrôle du DTB
- la logique d'accès est implantée au niveau du hardware
  - accès exclusif d'un maître au DTB
  - gestion des demandes d'accès (BUS REQUEST)
    - 3 modes :
      - PRI-ARBITER : priorité décroissante de BR3 à BR0
      - RRS-ARBITER : Round robin de BR(n) à BR(n-1)
      - SGL-ARBITER : requêtes de BR3 seulement acceptées

arbitrage pour l'accès au bus

- lignes d'arbitrage du bus DTB
  - 6 lignes de bus et 4 lignes en daisy-chain
    - 4 lignes de requêtes BR0 à BR3
    - 1 ligne BBSY (Bus Busy) pour indiquer que le bus est contrôlé par un maître
    - 1 ligne BCLR (Bus Clear) pour indiquer qu'une requête de priorité supérieure est présente en mode PRI-ARBITER
    - des lignes d'allocation de requête BG[0-3]IN et BG[0-3]OUT qui doivent être transmises d'un module à l'autre par une daisy chain (ou par des jumpers si il n'y a pas de module présent ou si le module n'utilise pas ce niveau)
  - un module demandant le contrôle du bus doit donc gérer 1 ligne de demande (BR[0-3]), 1 ligne d'allocation (BG[0-3]OUT) et la ligne de bus busy BBSY

arbitrage pour l'accès au bus

- le module assurant le rôle d'arbitre DOIT occuper le slot 1 dans le chassis VME
- le positionnement géographique des modules procure implicitement un niveau de priorité supplémentaire



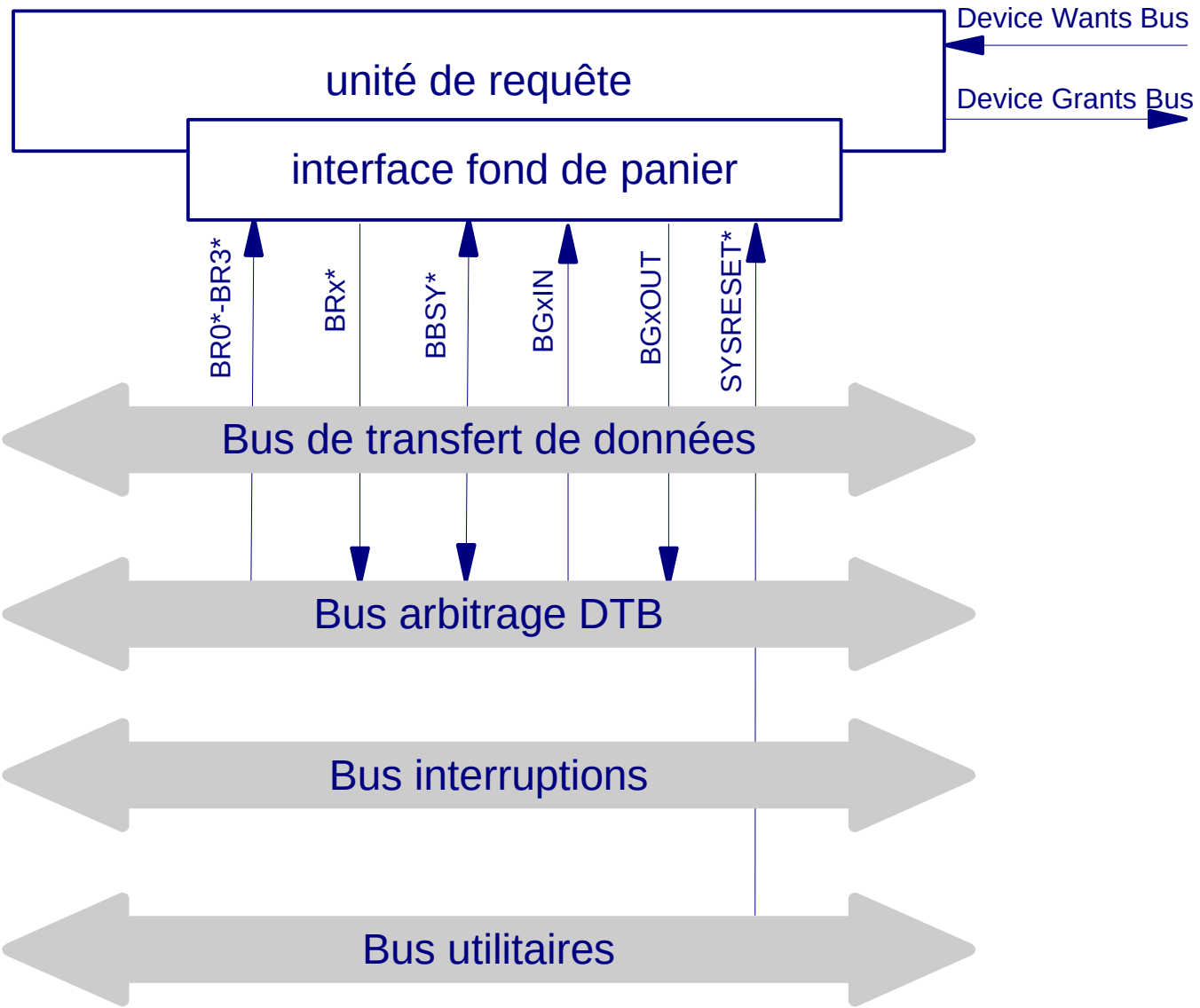
arbitrage pour l'accès au bus

- générateur de requête
  - gère le signal DEVICE_WANT_BUS du maître de son module VME, ou l'unité de gestion des interrupts de ce même module
  - génère un Bus Request (BR[0-3])
  - détecte le signal BG[0-3]IN et le transmet à BG[0-3]OUT si le bus n'est pas demandé par le maître de son module
  - si l'unité maître demande le DTB et que la ligne BG[0-3]IN est en niveau bas, le générateur indique la disponibilité du bus par un signal DEVICE_GRANTED_BUS et met la ligne BBSY en niveau bas

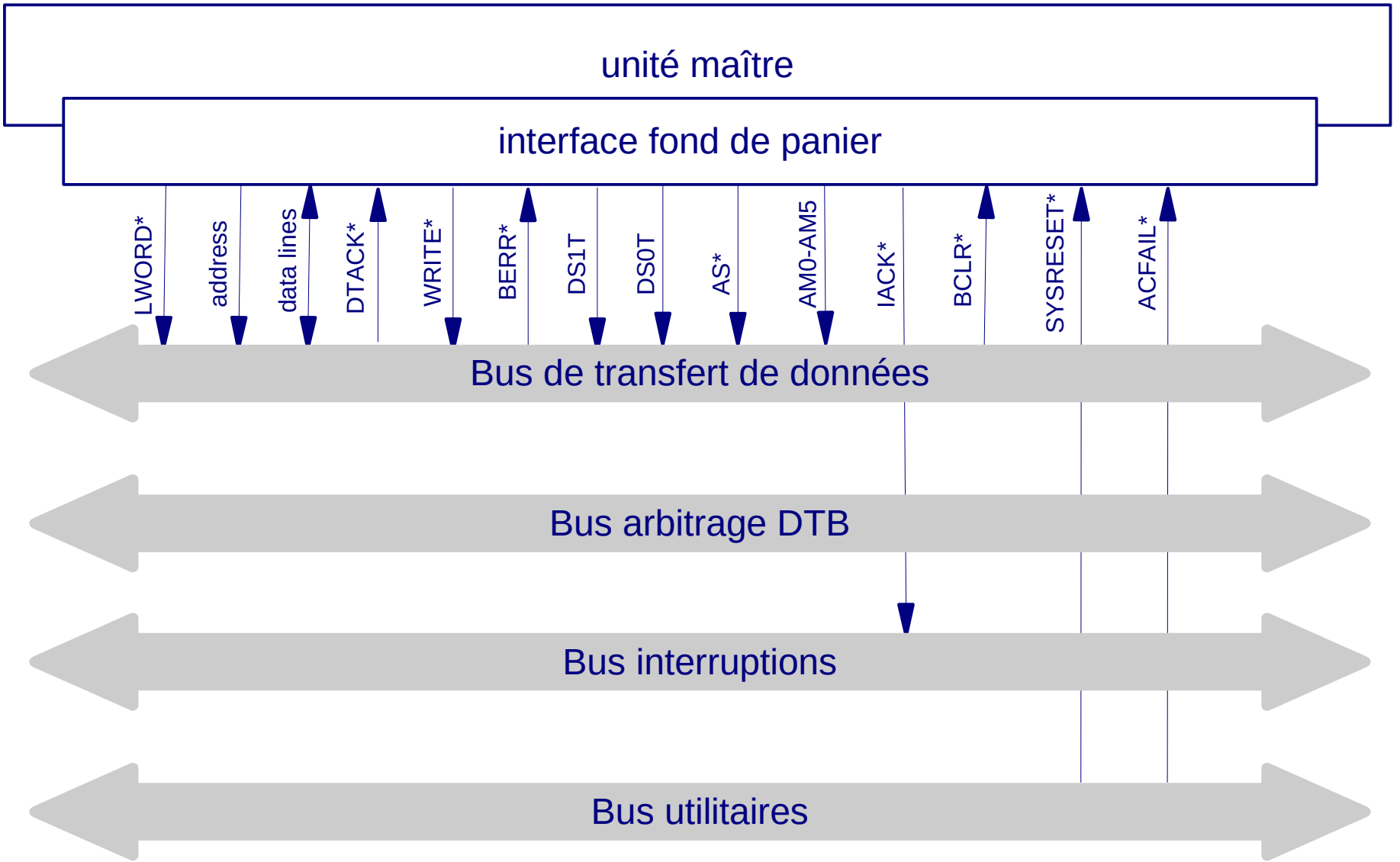
arbitrage pour l'accès au bus

- 3 modes de fonctionnement pour l'unité de requête de bus
  - RWD (Release When Done)
    BSSY est remise au niveau haut dès que la demande de contrôle est annulée
  - ROR (Release On Request)
    BSSY n'est remise au niveau haut que quand la demande de contrôle est annulée et qu'une autre requête BR[0-3] est présente (évite la surcharge du bus)
  - FAIR
    pas de requête de bus tant que des requêtes sont encore en cours de traitement

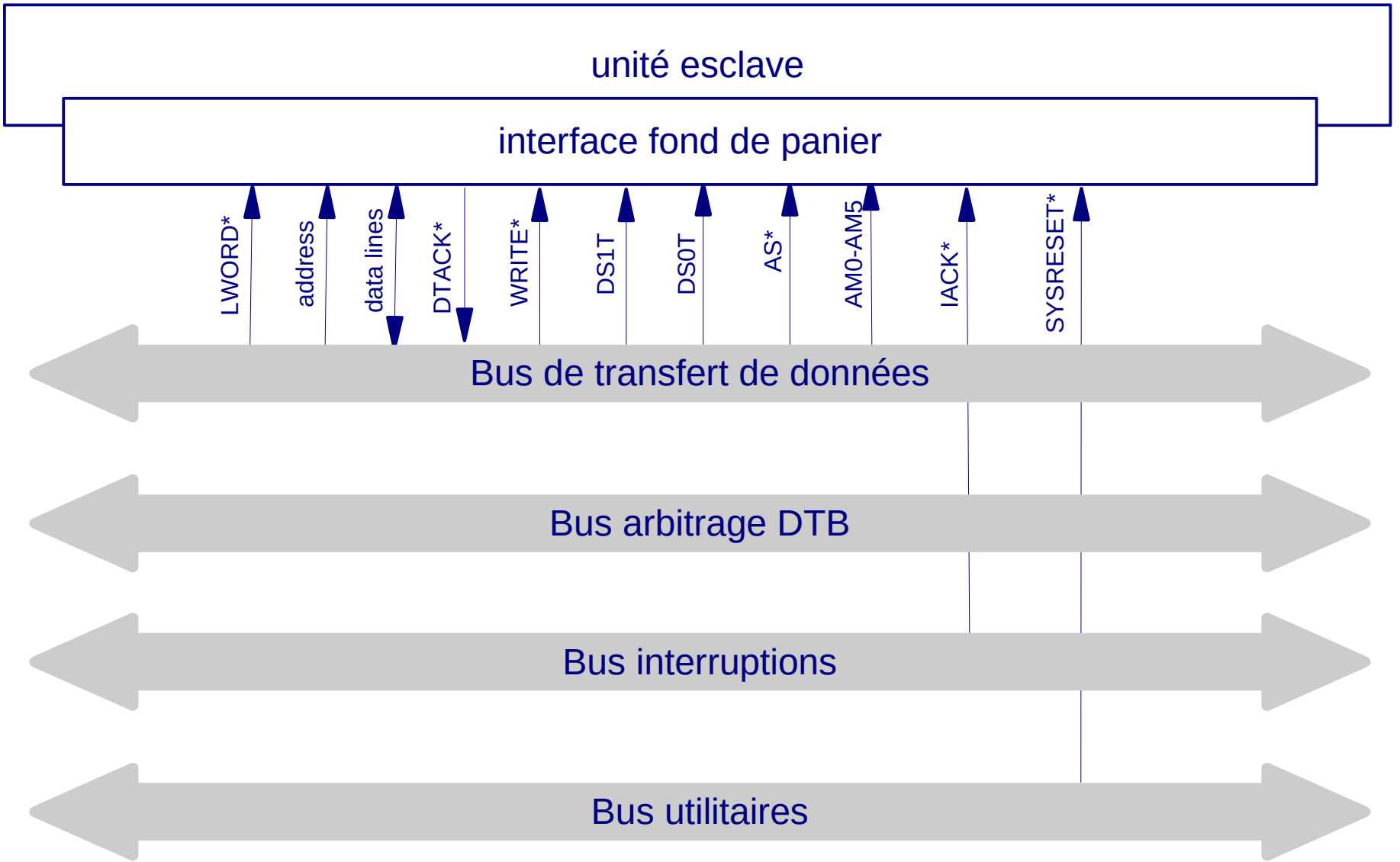
bloc diagramme du module de requête



bloc diagramme du module maître

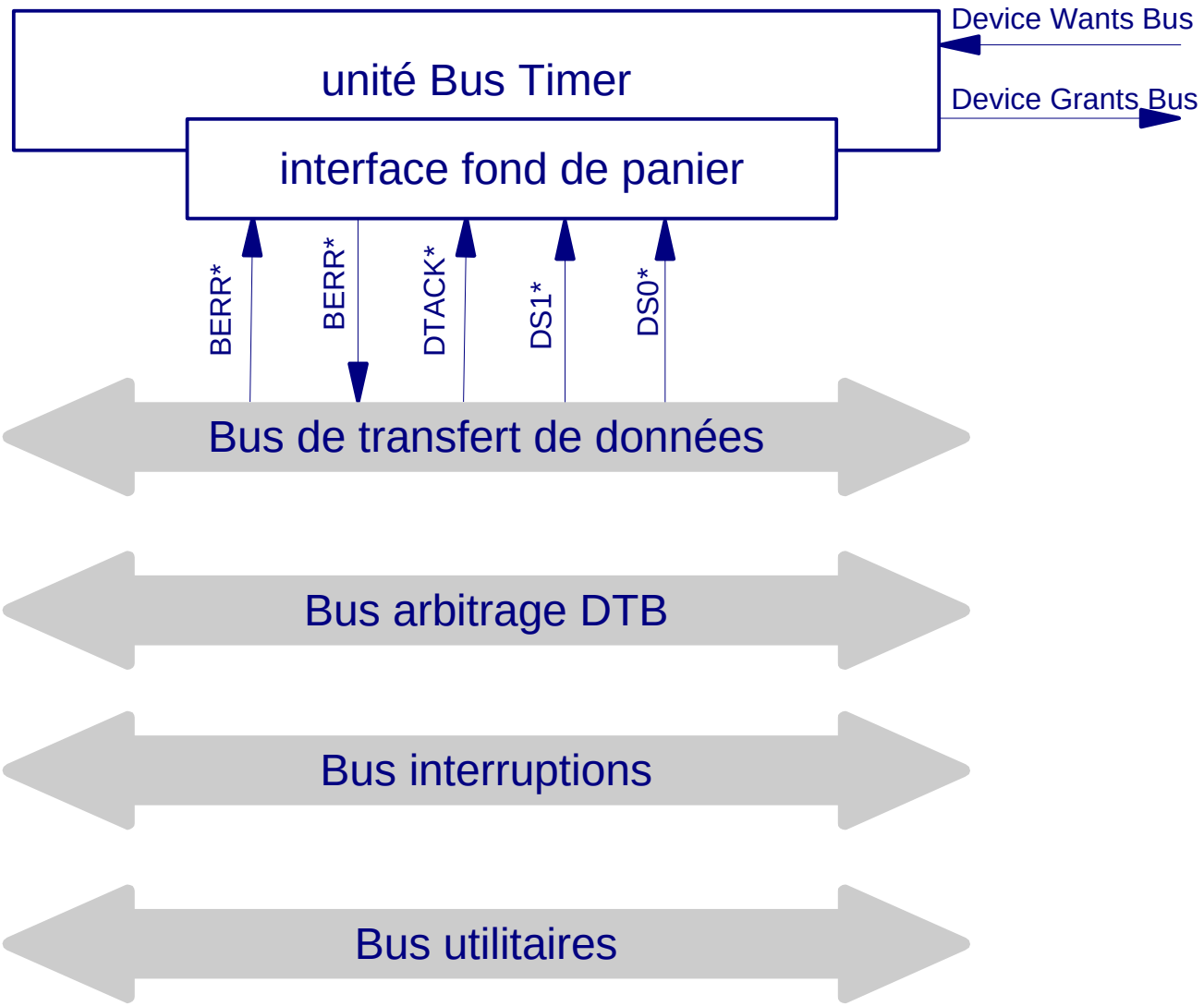


bloc diagramme du module esclave



bloc diagramme du module bus timer

- mesure du temps mis par les cycles sur le DTB



les modes de transfert de données

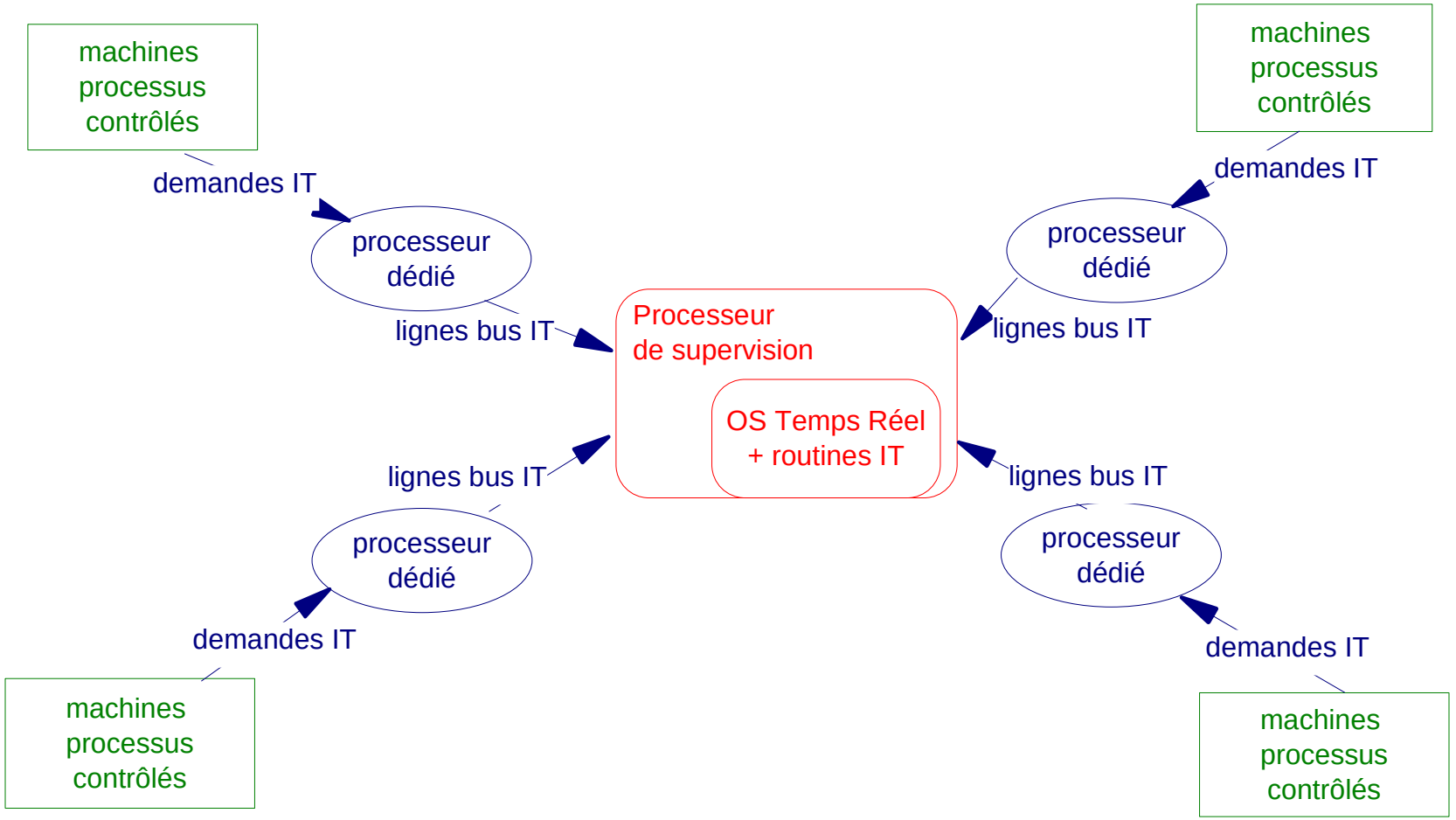
- cycle VME : séquence d'opérations qui permet la communication entre deux modules sur le bus VME, en suivant les protocoles définis dans la norme
- une fois qu'un cycle est démarré, et jusqu'à ce qu'il se termine, les autres cartes du chassis n'ont plus accès au contrôle
- les transferts de données peuvent se faire en modes 8 bits (octet et octet étendu), 16 bits et 32 bits

les modes de transfert de données

- 7 types de cycles :
  - lecture : transfert esclave $\in$ maître
  - écriture : transfert maître $\in$ esclave
  - lecture par blocs : seule l'adresse de départ est fournie et le maître ne rend le bus qu'à la fin
    256 blocs maximum
  - écriture par blocs : analogue à la lecture par blocs
  - lecture-modification-écriture, sans rendre le bus pour émuler l'atomicité de l'accès à une variable
  - adressage pour positionner simplement les adresses sur le bus
  - acquittement d'interruption pour confirmer la réception d'un interrupt et le transfert du vecteur

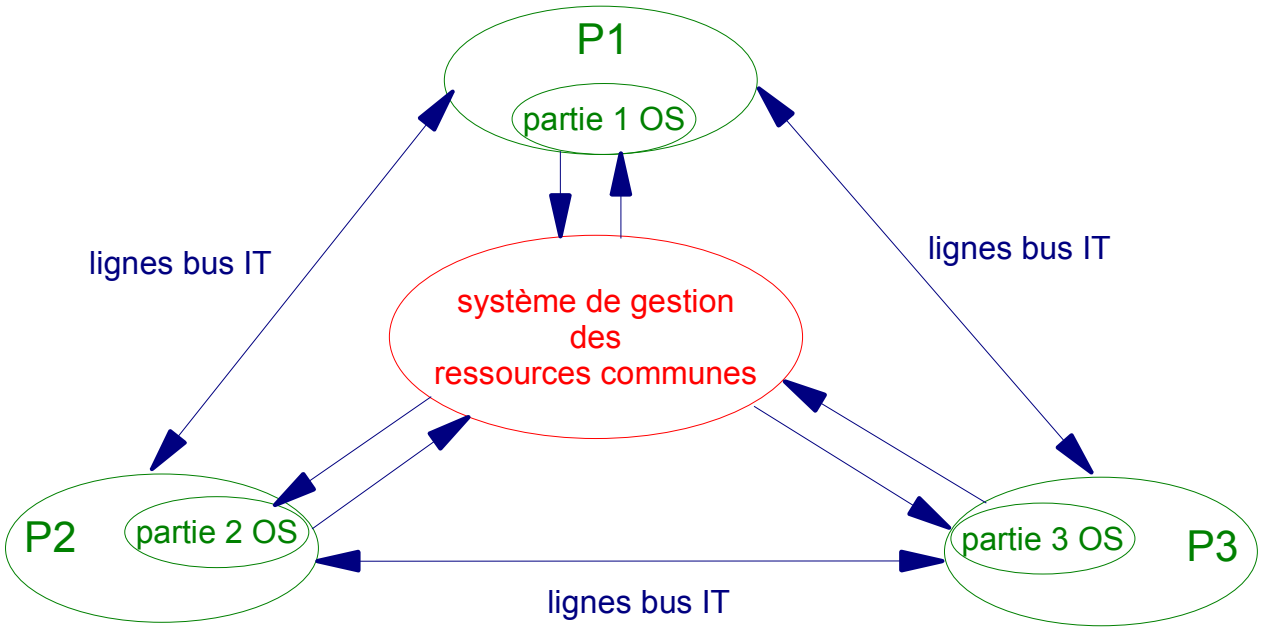
bus hiérarchisé d'interruptions

- pour générer et traiter les interruptions
- 2 catégories de systèmes de gestion :
  - système centralisé où un seul processeur reçoit et traite les interruptions



bus hiérarchisé d'interruptions

- système décentralisé où chaque processeur ne traite que les interruptions qui lui sont destinées. L'allocation de la prochaine tâche se fera sur le premier processeur qui se libère. Chaque processeur exécute une partie du système d'exploitation et a accès aux ressources globales



bus hiérarchisé d'interruptions

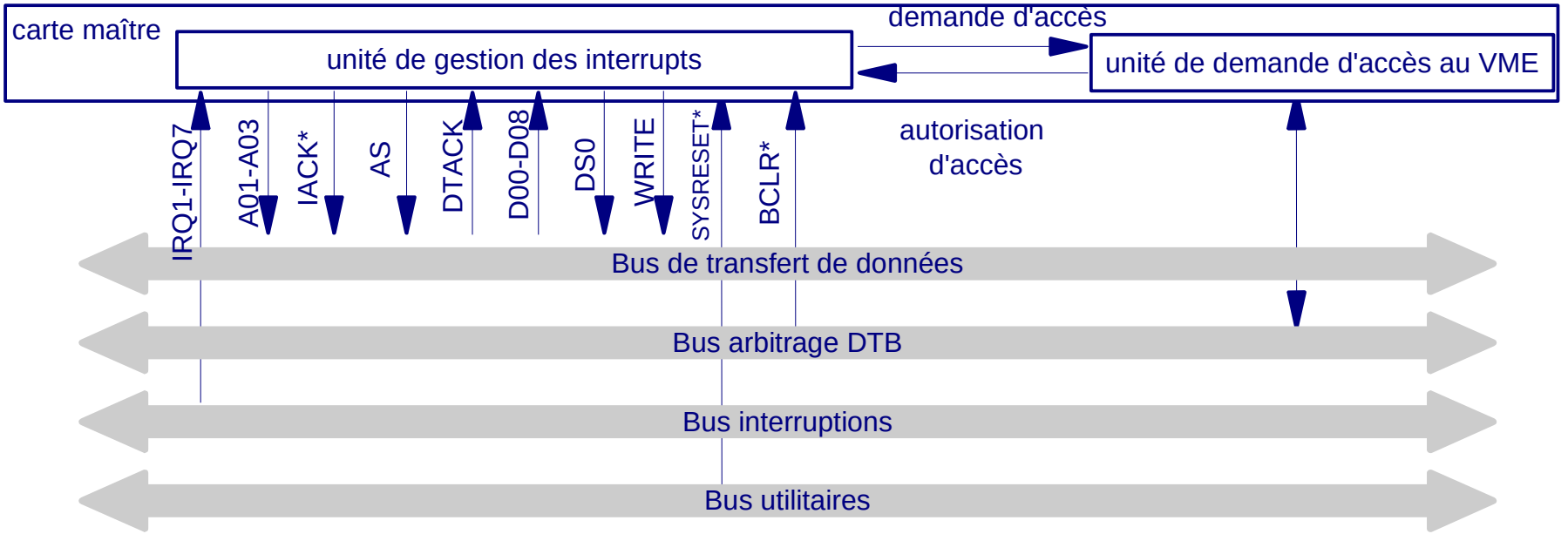
- lignes de gestion des interruptions
  - 7 lignes de demande IRQ1 à IRQ7
    - chaque ligne peut être activée par un niveau bas de l'unité de demande d'interruption (interrupter)
    - IRQ7 est la plus prioritaire dans un système centralisé
  - 1 ligne de prise en compte IACK
  - 2 lignes IACKIN et IACKOUT pour faire un chaînage des interruptions
    - la continuité des lignes doit être assurée par des cavaliers si un slot n'est pas occupé

bus hiérarchisé d'interruptions

- unité de gestion des interruptions, pour
  - déterminer parmi les demandes celle qui a la priorité la plus élevée
  - demander le contrôle du bus via l'unité de demande de contrôle du bus et générer alors le signal IACK de prise en compte de l'interrupt sur le bus
  - lire le vecteur d'interruption et initialiser la séquence de traitement

bus hiérarchisé d'interruptions

- unité de gestion des interruptions

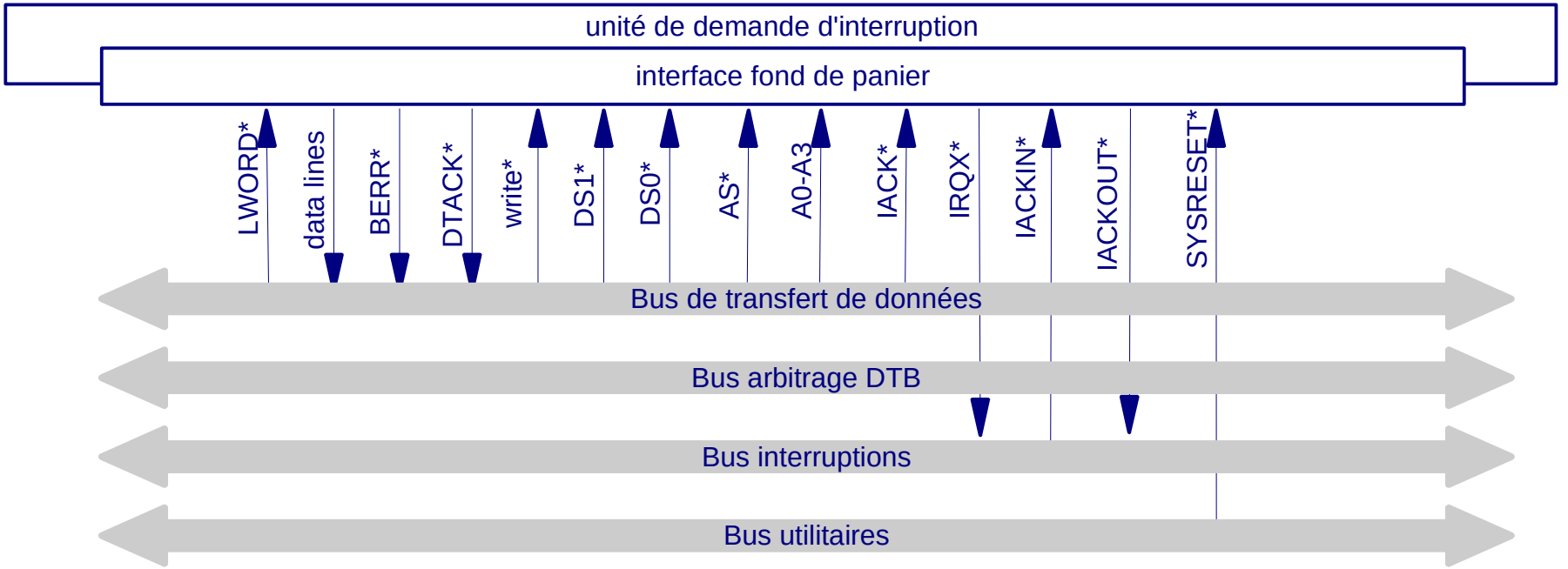


bus hiérarchisé d'interruptions

- unité de demande des interruptions
  - adresse une demande à l'unité de gestion associée à la ligne de demande utilisée
  - fournit un vecteur d'interruption quand elle reçoit le signal de prise en compte
  - transmet le signal de prise en compte si elle n'est pas la source

bus hiérarchisé d'interruptions

- unité de demande des interruptions



bus utilitaires

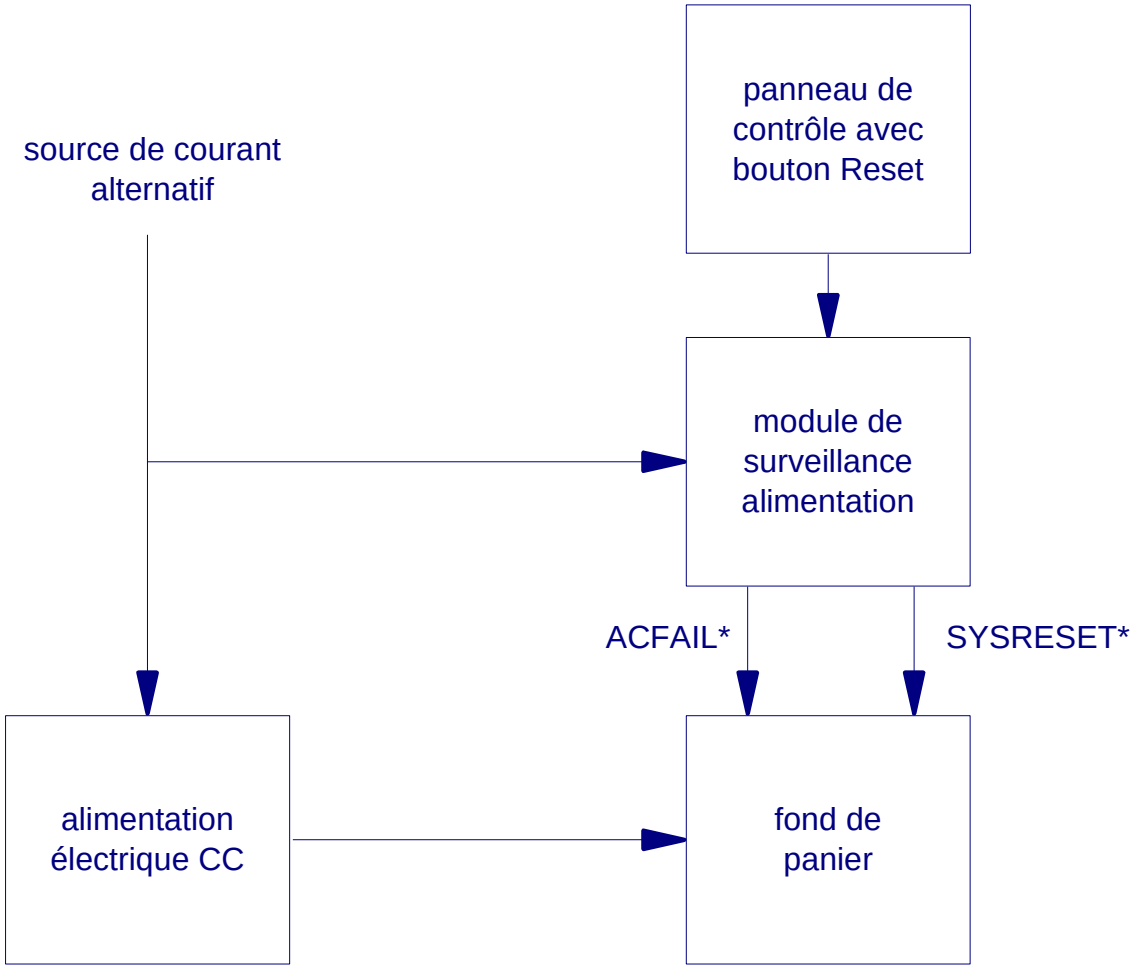
- lignes utilitaires :
  - SYSCLK horloge système
  - SERCLK horloge série
  - SERDAT données série
  - ACFAIL* panne d'alimentation
  - SYSRESET initialisation du système (prise en compte optionnelle)
  - SYSFAIL panne système

bus utilitaires

- modules utilitaires :
  - générateur de l'horloge système
    - autonome, fréquence 16 MHz, dans le module dans le slot #1
    - base de temps pour certains systèmes
  - horloge série
    - forme suivant IEEE P1132
    - pour des transferts série, en utilisant également les 2 lignes de transfert série

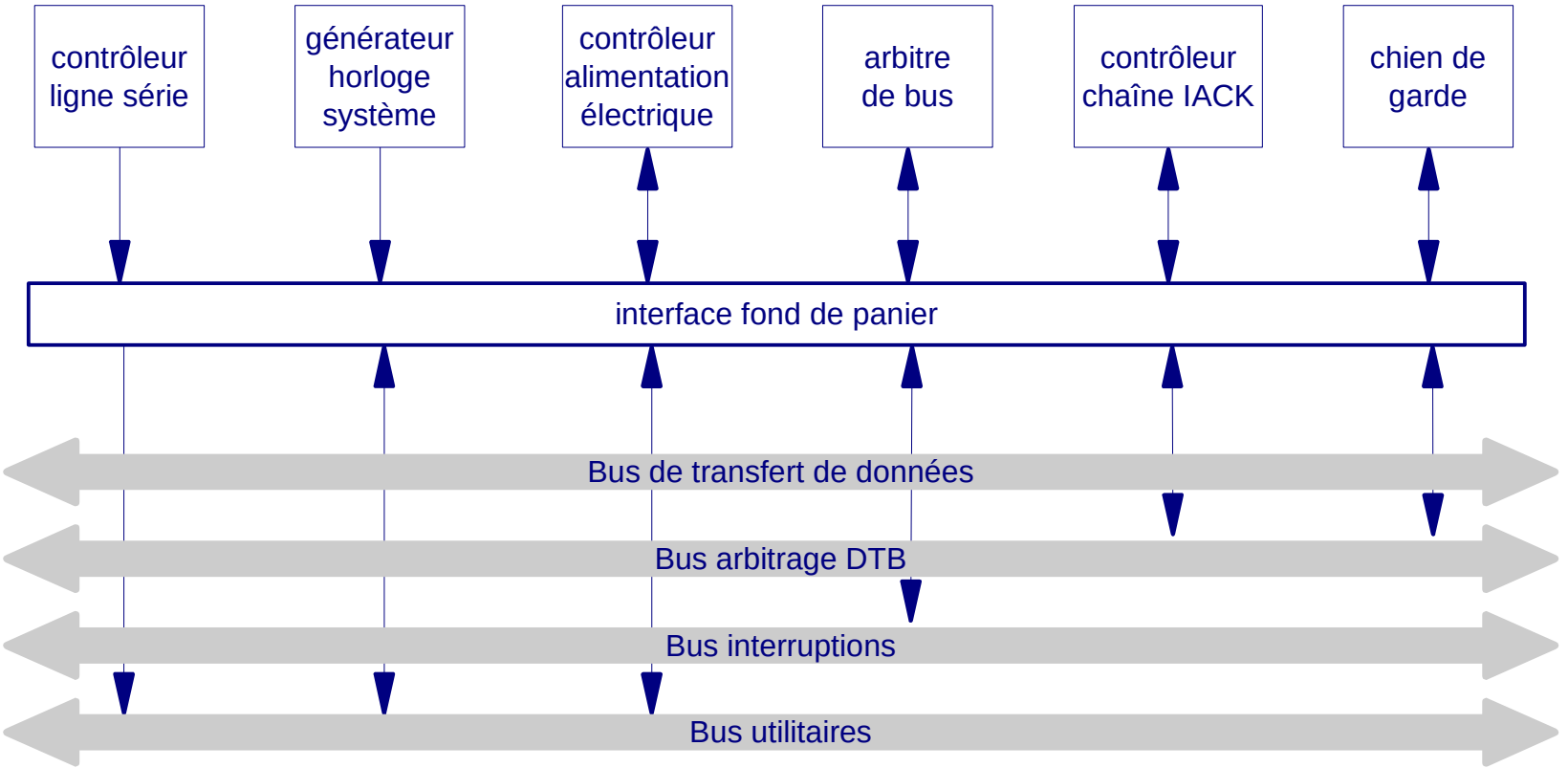
bus utilitaires

- surveillance de l'alimentation électrique



contrôle du système

- fonction assurée par le premier module dans le chassis



les principales cartes VME

- cartes processeurs
  - assurent généralement aussi les fonctions de maître
- cartes mémoires
  - de moins en moins utilisées avec l'augmentation de la mémoire disponible sur les cartes processeurs
- modules d'entrées-sorties
  - liaison avec le monde extérieur
  - prise en compte des interruptions extérieures
  - liaison avec d'autres types de bus