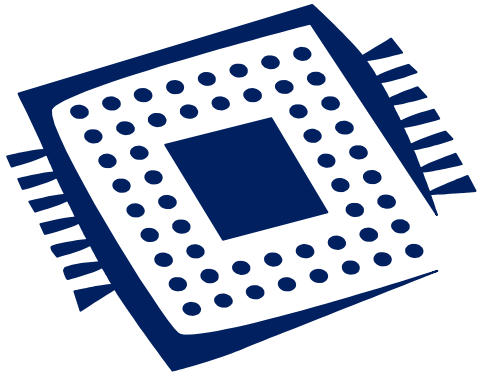


- COURS -

Bus de communication

Mohamed Chaker Bali
mohamed.chaker.bali@gmail.com

Janvier 2018

Iset'Com
Institut Supérieur des Etudes Technologiques
en Communications de Tunis

Organisation générale du cours

- 21h de cours (transparents) + TD
- 21h de TP
- DS et Examen finale

Contenu du cours

Partie 1. Introduction aux systèmes embarqués

- Définition, historique et applications
- Technologies des systèmes embarqués
- Les puces ARM STM32

Partie 2. Interfaçage local : capteur et modules

- Universal Asynchronous Receiver Transmitter (UART)
- Serial Peripheral Interface (SPI)
- Le bus I2C

Partie 3. Interfaçage système et bus externes

- Le bus USB
- Le bus CAN

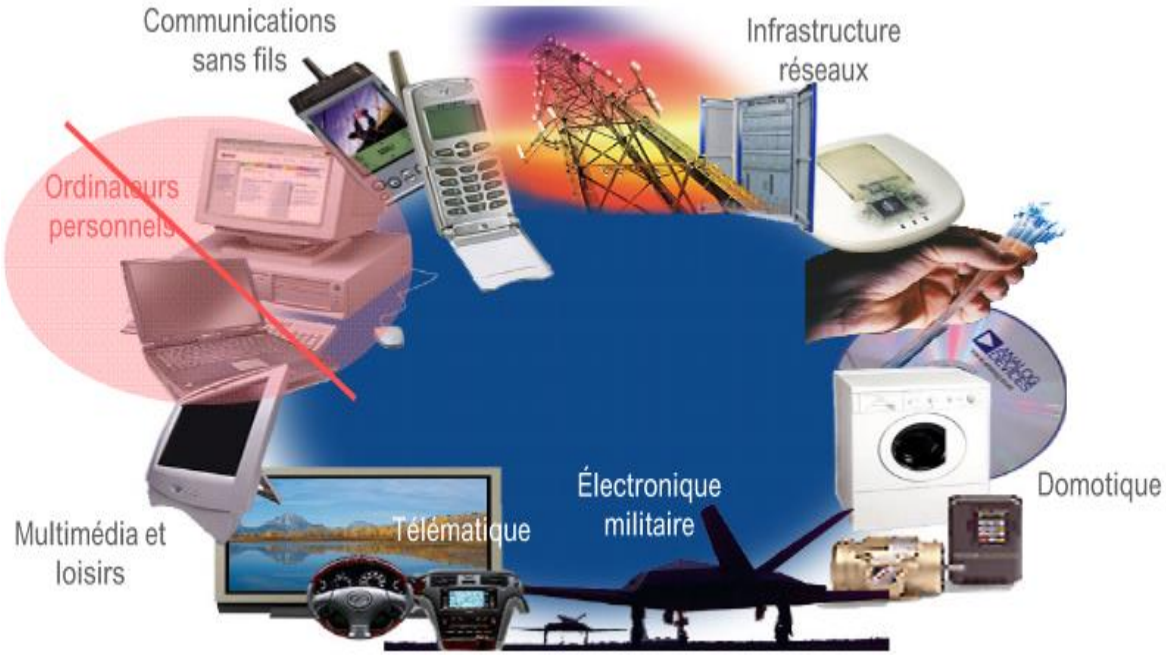
Partie 1.

Introduction aux systèmes embarqués

1. Définition, historique et applications
2. Technologies des systèmes embarqués
3. Les puces ARM STM32

Définition d'un système embarqué

- Tout système conçu pour résoudre un problème ou une **tache spécifique** mais n'est pas un ordinateur d'usage générale
- Système électronique et informatique **autonome** avec des entrées/sorties non standard (clavier ou écran du PC)
- Système utilisant **conjointement** du matériel (Hard) et du logiciel (Soft) pour mettre en œuvre une **fonction spécifique**
- Système matériel et application **étroitement liés**: non facilement **discernables** comme dans un système de type PC
- Système qui **interagit** fortement avec son environnement (contraintes de temps réel et de dynamique des phénomènes physiques)



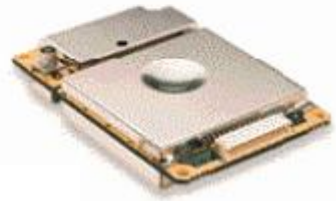
Caractéristiques d'un système embarqué

Caractéristiques générales :

- Dédié à une application spécifique
- Coût réduit,
- Maximisation rapport performance/prix
- Volume restreint (compact, pas modulaire)
- Capacité mémoire adaptée
- Capacité de calcul appropriée à l'application
- Exécution temps réel (souvent)
- Fiabilité et sécurité de fonctionnement
- Consommation d'énergie maitrisée (voir très faible en cas d'utilisation sur batterie)







Système embarqué vs PC

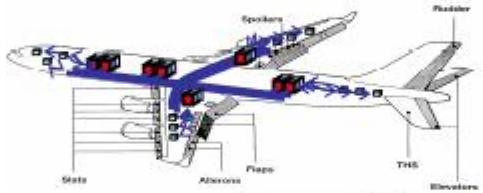
Système embarqué

- Forme et périphériques suivant les exigences et l'environnement de l'application
- Matériel et logiciel dédiés aux fonctionnalités de l'application
- Contraintes de consommation, de temps de traitement, d'encombrement, de robustesse et de coût

Micro-ordinateur PC

- Forme et périphériques standards
- Matériel ouvert à de nombreuses applications logicielles
- Dispositif qui s'adapte à différents types de travaux: bureautique, calcul scientifique, gestion,…
- Moins de contraintes sur les ressources logicielles et matérielles

Historique des systèmes embarqués



Historique des systèmes embarqués

- **1833**: Babbage invente le concept du calculateur programmable
- **1945**: Von Neumann introduit une nouvelle d'architecture de processeur programmable
- **1946**: ENIAIC, 1ère machine de calcul électronique (18000 tubes électroniques)
- **1948**: Invention du transistor aux Bell Labs (Bardeen,Shockley)
- Les premiers systèmes embarqués sont apparus en **1971** avec l'apparition du Intel 4004 (le début de l'informatique embarquée)
- **1980**: RTOS (Real Time OS) génériques
- **1985**: entrée en masse sur le marché des processeurs industriels des firmes: Motorola, Zilog, Texas Instruments, Analog Devices, Thomson, Philips,…
- **1990**: explosion du marché des téléphones portables et de l'Internet: nombreux nouveaux opérateurs,…
- **20xx**: marché du multimédia, domotique, transport intelligent, réalités virtuelles,…

Domaines d'application

- **Produits électroménagers**
  - Cafetière, machines à laver, fours à micro-onde,…
- **Électronique grand public**
  - Caméras et appareils photos numériques, décodeurs vidéo, téléphones portables, PDA,…
- **Automobile**
  - ABS, GPS, contrôle moteur, informatique de confort,…
- **Avionique, spatial, procédés industriels**
  - Systèmes de navigation aérienne et maritime, systèmes de contrôle des procédés industriels
- **Télécommunications et informatique**
  - Terminaux, nœuds de transfert, équipement de transmission, périphériques informatiques

Applications en télécommunications

- **LAN câblé**

RNIS, xDSL (30 – 500 MIPS), ATM, IP (3000 MIPS),…

- **LAN sans fils**

WLL (DECT: 30 MIPS), WLAN: (WiFi: 30 – 1000 MIPS), WiMax,… UWB,…

- **WAN**

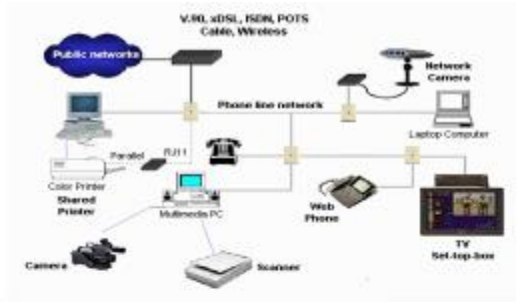
GSM (30 MIPS), UMTS (300 MIPS), MBS (3000 MIPS),….

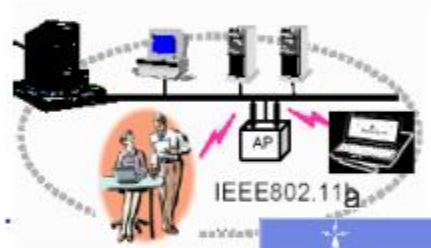
- **Applications multimédia**

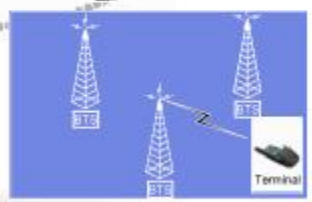
Communication: Visiophone, TV numérique, Web-phone, vidéosurveillance,…

Diffusion: Audio (DAB), Vidéo (DVB),…

Manipulation: services interactifs, gestion de multiples flots d'information (MPEG4),…





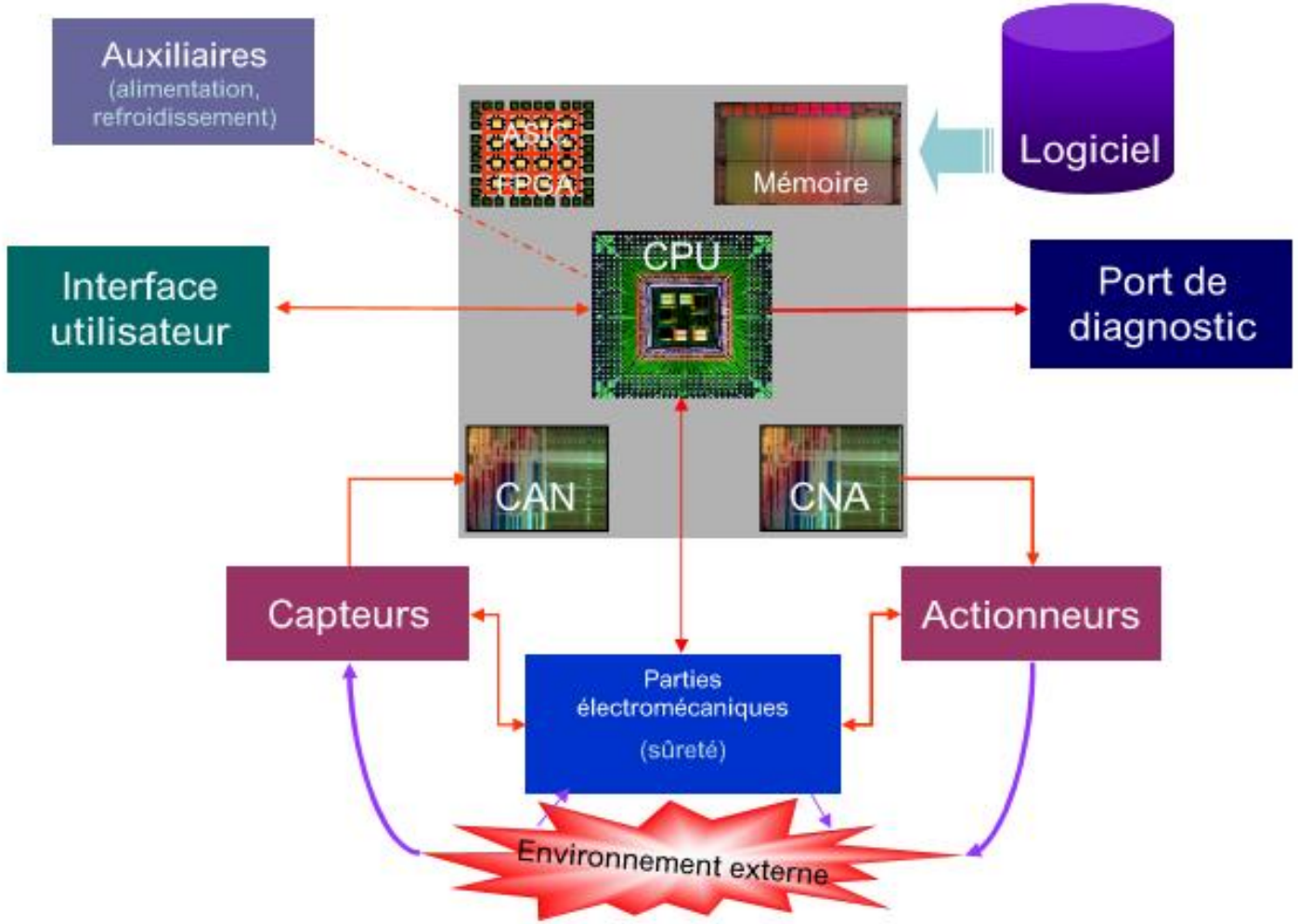


Partie 1.

Introduction aux systèmes embarqués

1. Définition, historique et applications
2. Technologies des systèmes embarqués
3. Les puces ARM STM32

Architecture d'un système embarqué



Classification des technologies

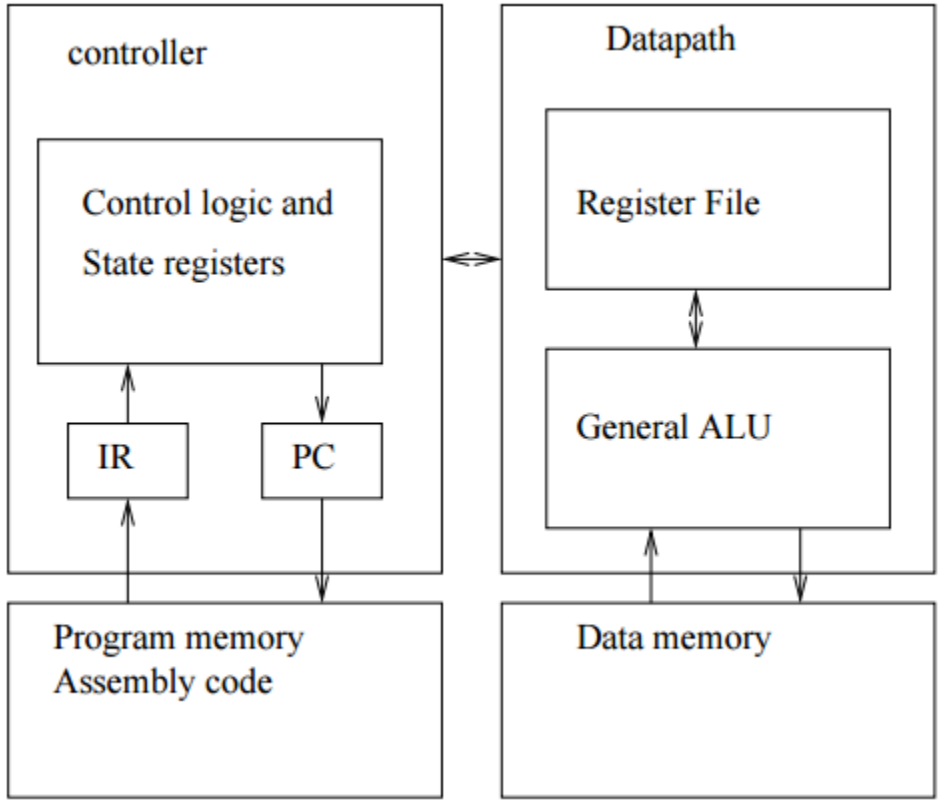
- **Technologie**: façon d'accomplir une tâche en faisant appel à des procédés techniques, des méthodes et du savoir faire
- **Trois technologies clés pour les systèmes embarqués**:
  - Technologie des processeurs: architecture basée sur un moteur de traitement utilisé pour implanter les fonctionnalités du système (GPP, ASP, SPP)
  - Technologie des circuits intégrés (IC): permet une implantation numérique (niveau portes) mappée dans un circuit intégré (Full-custom ASIC, Semi-custom ASIC, PLD)
  - Technologie de conception: règles d'optimisation de développement et d'intégration système, d'automatisation des étapes du flux de conception et d'optimisation du temps de développement (Co-Design, partionnement, …)

Technologie clé: Processeurs

- **Grande variété** d'architecture de processeurs
- Un processeur n'est pas nécessairement programmable
- On distingue généralement
  - Les processeurs à usage généraux (General Purpose Processor, GPP)
  - Les processeurs dédiés à une tache (Single Purpose Processor, SSP)
  - Les processeurs spécifiques à certaines applications (Application Specific Processor, ASP. ex: DSP)

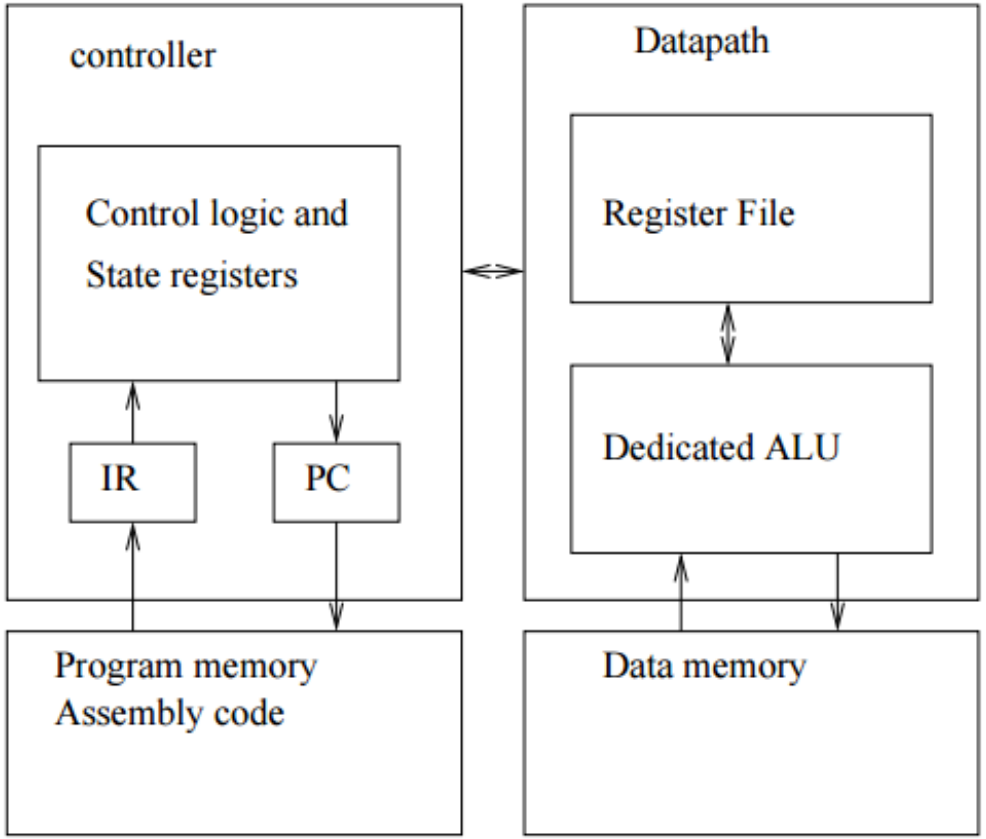
Processeurs à usage général (GPP)

- **Processeur programmable** utilisé pour de **nombreuses applications** (aussi appelé microprocesseur)
- Caractéristiques
  - Une mémoire pour le programme
  - Un chemin de donné (datapath) généraliste comprenant un unité arithmétique et logique (ALU) puissante et un gros ban de registre
- Intérêt:
  - Time to market et coût
  - flexibilité
- Exemple: Pentium, PowerPC, ARM, MIPS, etc.



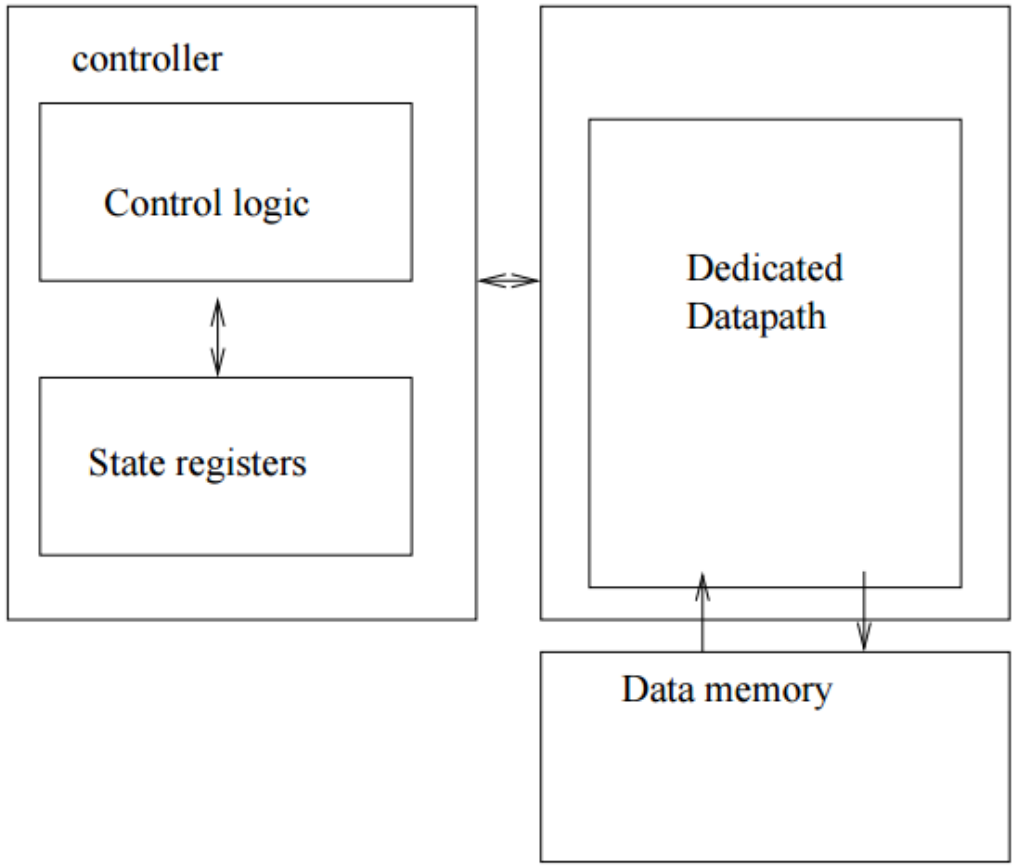
Processeurs spécifiques (ASP)

- **Processeur programmable optimisé** pour **une classe particulière d'applications** (ASP: Application Specific Processor).
- Caractéristiques:
  - Mémoire de programme
  - Chemin de donnée optimisé
  - Unités fonctionnelles spécifiques
- Intérêt:
  - Flexibilité
  - performances: surface, rapidité, consommation
- Exemple: DSP, micro-controleur (processeur 4bits, 8bits).



Processeurs dédiés (SPP)

- **Circuits intégrés** destinés à exécuter **exactement un programme**: coprocesseur, accélérateur matériel ou périphérique.
- Caractéristiques:
  - Contient seulement les composants nécessaires à l'exécution du programme concerné
  - en général pas de mémoire de programme
- Intérêt:
  - Rapidité
  - Faible consommation
  - Surface
- Exemple: unité de calcul flottant, contrôleur USB, PCMCIA, decoder



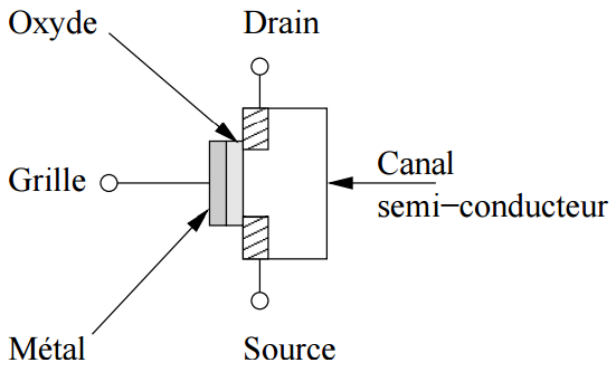
Principales caractéristiques des processeurs

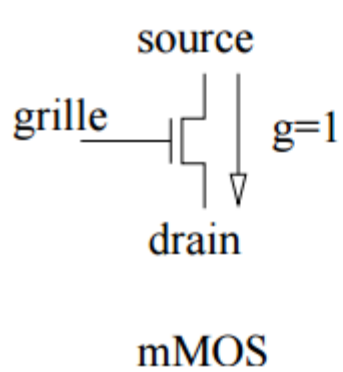
- **Largeur des registres internes** de manipulation de données (8, 16, 32, 64, 128) bits.
- **Cadence d'horloge** exprimée en MHz ou GHz.
- **Nombre de noyaux** de calcul (core).
- **Jeu d'instructions** (ISA: Instructions Set Architecture) dépendant de la famille (CISC, RISC, etc)
- **Finesse de gravure** exprimée en nm (nanomètres)
- **Famille d'architecture**:
  - CISC (Complex Instruction Set Computer): choix d'instructions aussi proches que possible d'un langage de haut niveau).
  - RISC (Reduced Instruction Set Computer) : choix d'instructions plus simples et d'une structure permettant une exécution très rapide).
  - VLIW (Very Long Instruction Word).
  - DSP (Digital Signal Processor): processeur conçu et adapté à certains types de calculs (3D, image, vidéo, son, etc.).

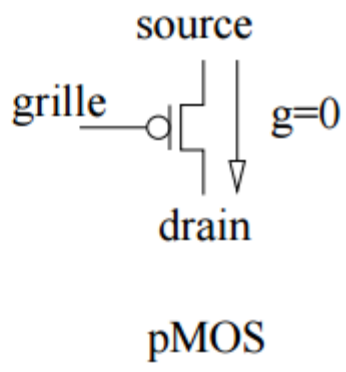
Technologie clé: circuits intégrés

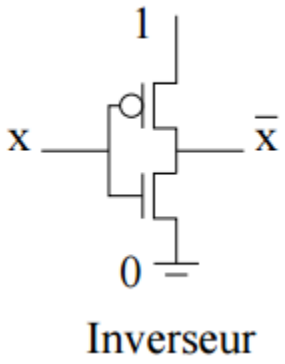
- **Les circuits intégrés (CI)** opèrent en logique booléenne (portes logiques ON/OFF)
- **Le transistor**: un robinet à électrons
- **Technologie CMOS** (Complementary Metal Oxide Semiconductor)
  - Niveaux logiques : 0=0V et 1=3V
- Deux types de portes
  - nMOS : conducteur si la grille=1
  - pMOS : conducteur si la grille=0
- Quelques portes de base: Inverseur, NAND, NOR

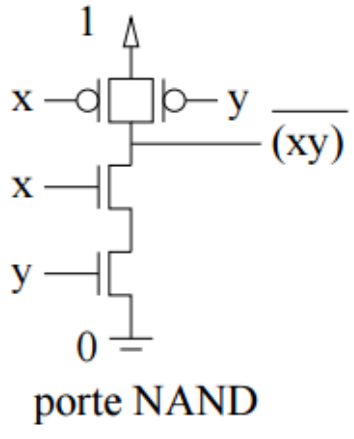
Technologie CMOS

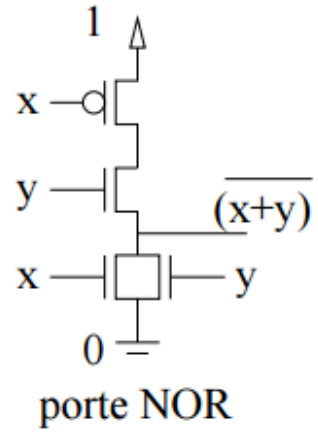






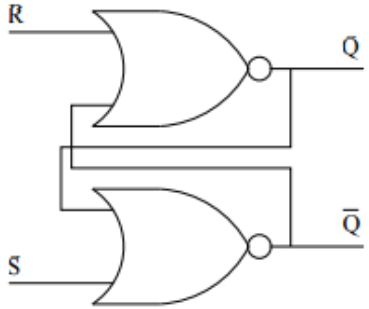




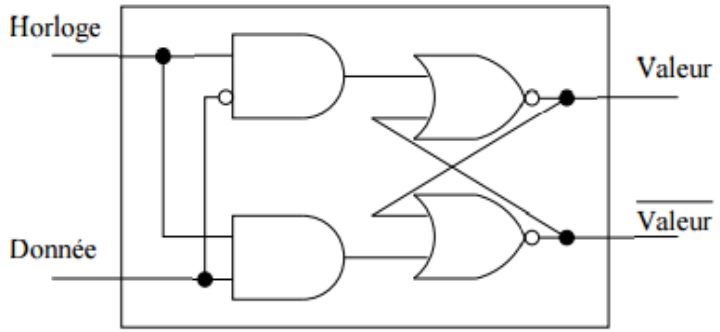


Circuits numériques

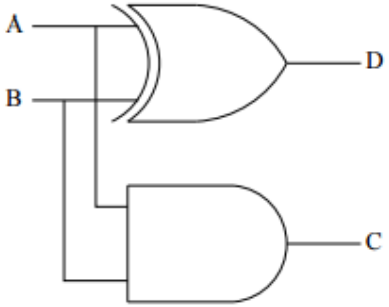
- **Les portes logiques** permettent de construire **n'importe quel circuit** par **assemblage**:
  - Mémorisation



Bascule RS



  - logique combinatoire



Additionneur 1 bit

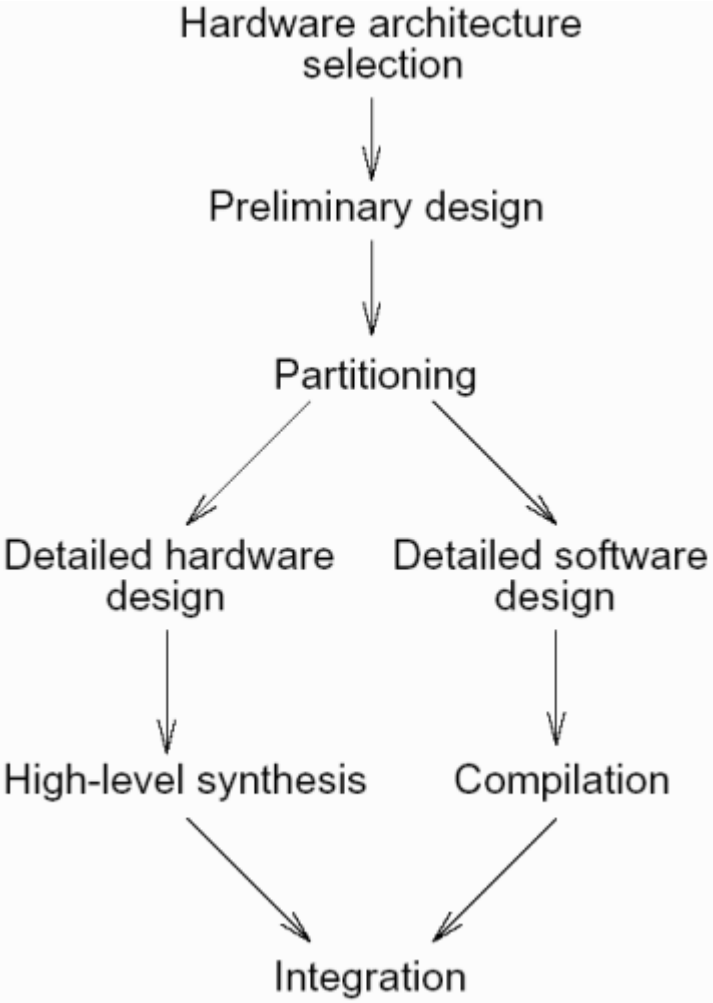
- Un circuit peut être construit comme une machine à états finie ou bien un circuit flot de données.

Circuits intégrés: choix possibles

- Par ordre décroissant de complexité de conception, on trouve les circuits intégrés:
  - **Full-custom**: tout est modifiable : transistors (type, caractéristiques), connexions. . .
  - **Semi-custom**: les éléments logiques sont choisis dans une bibliothèque de portes, les connexions sont libres
  - **Circuits logiques programmables**: les éléments logiques existent déjà physiquement sur le circuit, seules les connexions peuvent être définies
- Il y a un compromis entre la complexité de conception et les performances

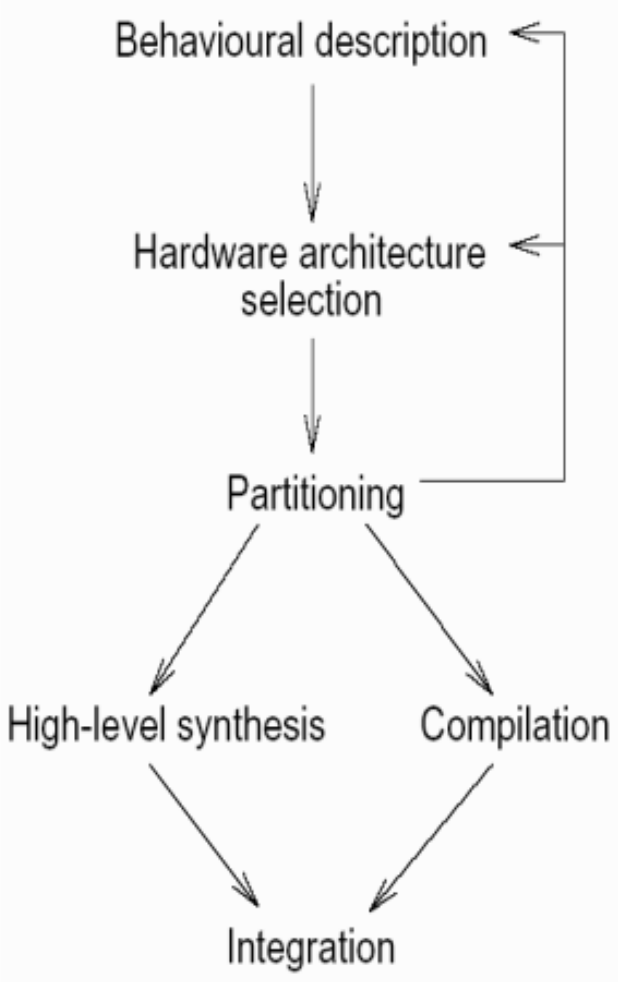
Conception d'un système embarqué

- 2 équipes qui travaillent (matériel et logiciel) séparément
- **1ére étape**: choix du matériel (composants électroniques, processeur,…) pour le système
- **2éme étape**: le système matériel conçu est transféré aux programmeurs
- **3éme étape**: les programmeurs réalisent un logiciel en n'exploitant que les ressources matériel imposées
- Inconvénients
  - Difficulté d'évaluer les ressources réellement nécessaires
  - Sous-systèmes souvent trop puissants par rapport aux besoins réels



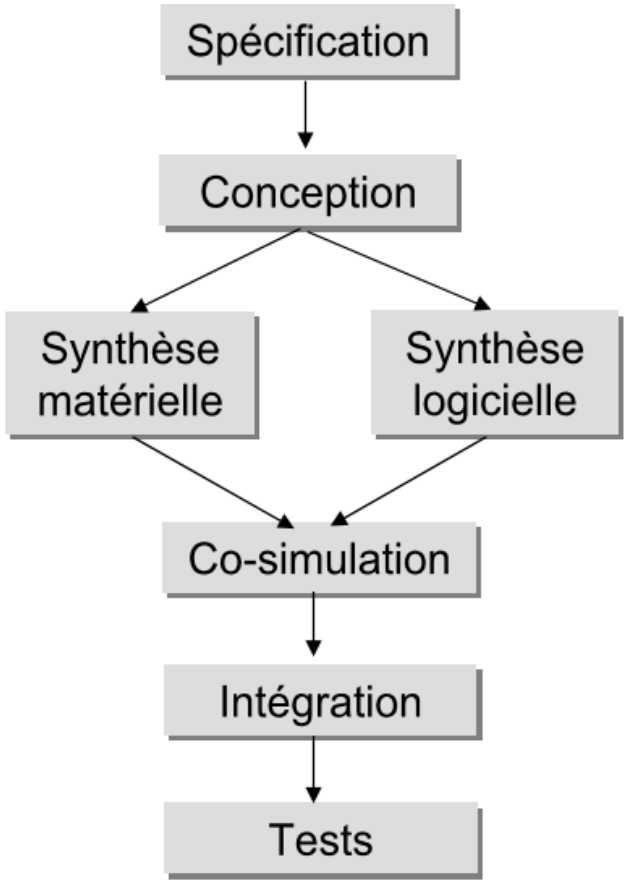
Définition du codesign

- Conception en même temps à la fois le matériel et le logiciel pour une fonctionnalité à implémenter
- Conception HW/SW réalisé par le même groupe d'ingénieurs en coopération
- Repousser le plus loin possible dans la conception du système les choix matériel
- Apports du codesign:
  - Parallélisme, algorithmes distribués, architecture spécialisée
  - Reconfiguration statique et dynamique
  - Indépendance vis-à-vis des évolutions technologiques



Étapes du codesign

1. **Spécification**: liste des fonctionnalités du système de façon abstraite
2. **Techniques de conception**: modélisation, partitionnement, synthèse de communication, ordonnancement,…
3. **Synthèse matérielle**: choix du matériel et CAO (Conception Assisté par Ordinateur) électronique
4. **Synthèse logicielle**: compilation
5. **Co-simulation**: validation de l'architecture et des fonctionnalités
6. **Intégration**: réalisation matérielle et intégration des modules logicielles
7. **Tests d'intégration**: tests expérimentaux et debugage



Partie 1.

Introduction aux systèmes embarqués

1. Définition, historique et applications
2. Technologies des systèmes embarqués
3. Les puces ARM STM32

Qu'est-ce qu'un microcontrôleur ?

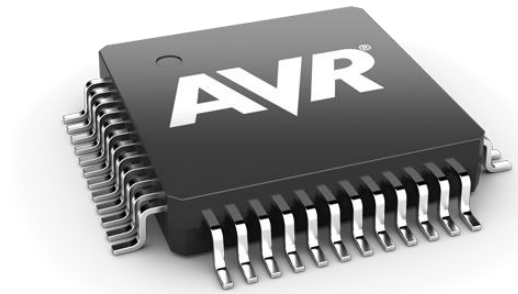
- **Microcontrôleur** (µC, uC, MCU) = circuit intégré qui rassemble les éléments essentiels d'un ordinateur :
  - unité de calcul et de contrôle
  - mémoire vive
  - périphériques (ex : port série, timers, DMA, flash. . .)
  - E/S analogiques (ex : convertisseurs analogique ↔ numérique (ADC/DAC))

- **Exemples d'utilisation**
  - Electroménager & Internet of Things (montres connectées, bracelets santé, capteurs domotiques, four, machine à laver, thermostat, smartphones, tablettes
  - Automobile, Aeronautique (une voiture moderne intègre ≈30 MCUs)

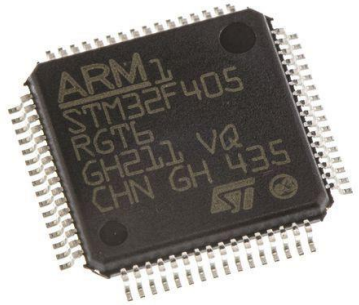
Quelques MCUs

- **Microchip PIC16** (1990, 8 bits, 1MHz, quelques registres)
- **Microchip PIC12**

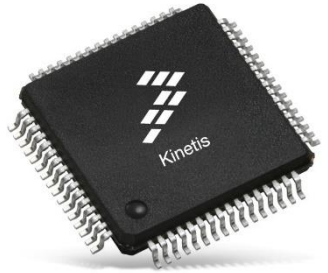
- **Atmel AVR** (2000, 8/16 bits, base des Arduino)



- **ARM (2000) :** Coeurs communs, différents constructeurs (ST,Texas Instruments, Microchip. . .)



- **SoC** Freescale, Broadcom, TI etc. (2010, Raspberry Pi)
  - 1GHz, 64 bits, RAM 512 Mo



Le MCU, un compromis

- **L'intérêt**
  - Forte intégration (une puce intègre toutes les fonctions)
  - Faible consommation électrique (1–100mW)
  - Faible coût (0.10–10€)
  - Généricité (par rapport au silicium dédié)

- **Les limitations**
  - peu de capacité de calcul (1–200 MHz)
  - très faible stockage (1–512 Ko RAM)

Protocoles de communication embarqués

- Toute communication entre le MCU et ses périphérique s'établit selon un protocole (une langue commune)
- **Interne**
  - bus de données : unité de calcul ↔ RAM ↔ DMA,
- **Externe**
  - communication avec puces/cartes externes ;
  - communication inter-MCU (réseau de capteurs)
- **Architectures**
  - symétrique, en réseaux, client-serveur. . .
  - en général : couches de nombreux protocoles
- **Exemple**
  - UART, SPI, I2C, CAN, USB, OSI. . .

Protocoles de communication embarqués

Définition

- « Un bus est un jeu de lignes partagées pour l'échange de mots numériques. » (Traité de l'électronique, Paul Horowitz & Winfield Hill)
- Un bus permet de faire transiter (liaison série/parallèle) des informations codées en binaire entre deux points.
- Typiquement les informations sont regroupés en mots : octet (8 bits), word (16 bits) ou double word (32 bits).

Caractéristiques

- nombres de lignes,
- fréquence de transfert.

La famille des ARM Cortex

ARM

- Architecture qui spécifie (notamment) un jeu d'instructions RISC, l'organisation de la mémoire, modèle d'exécution. . .
- Différentes versions du standard : ARMvX-M

Cortex

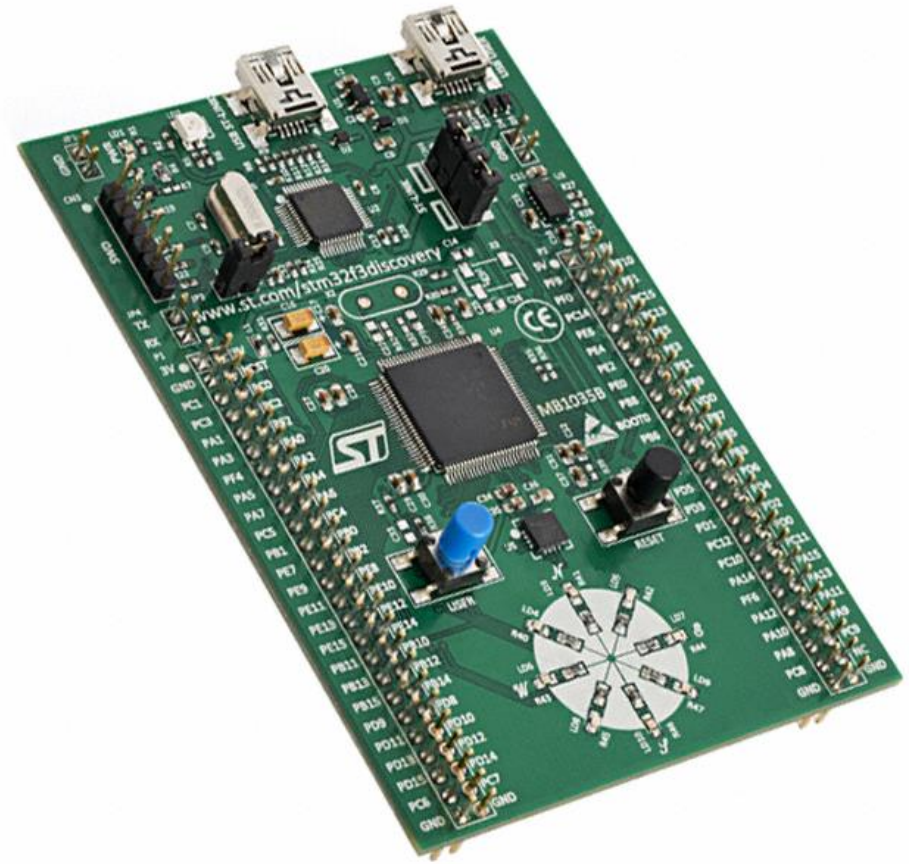
- Famille de cœurs physiques vendus aux fabricants de silicone
  - Cortex-A pour Application (ex : smartphones)
  - Cortex-R pour Real-time (temps d'exécution déterministe)
  - Cortex-M pour eMbedded (faible consommation)
- Licences Cortex-M4
  - Atmel, STMicroelectronics, NXP, Texas Instruments

La famille des MCU STMicroelectronics STM32

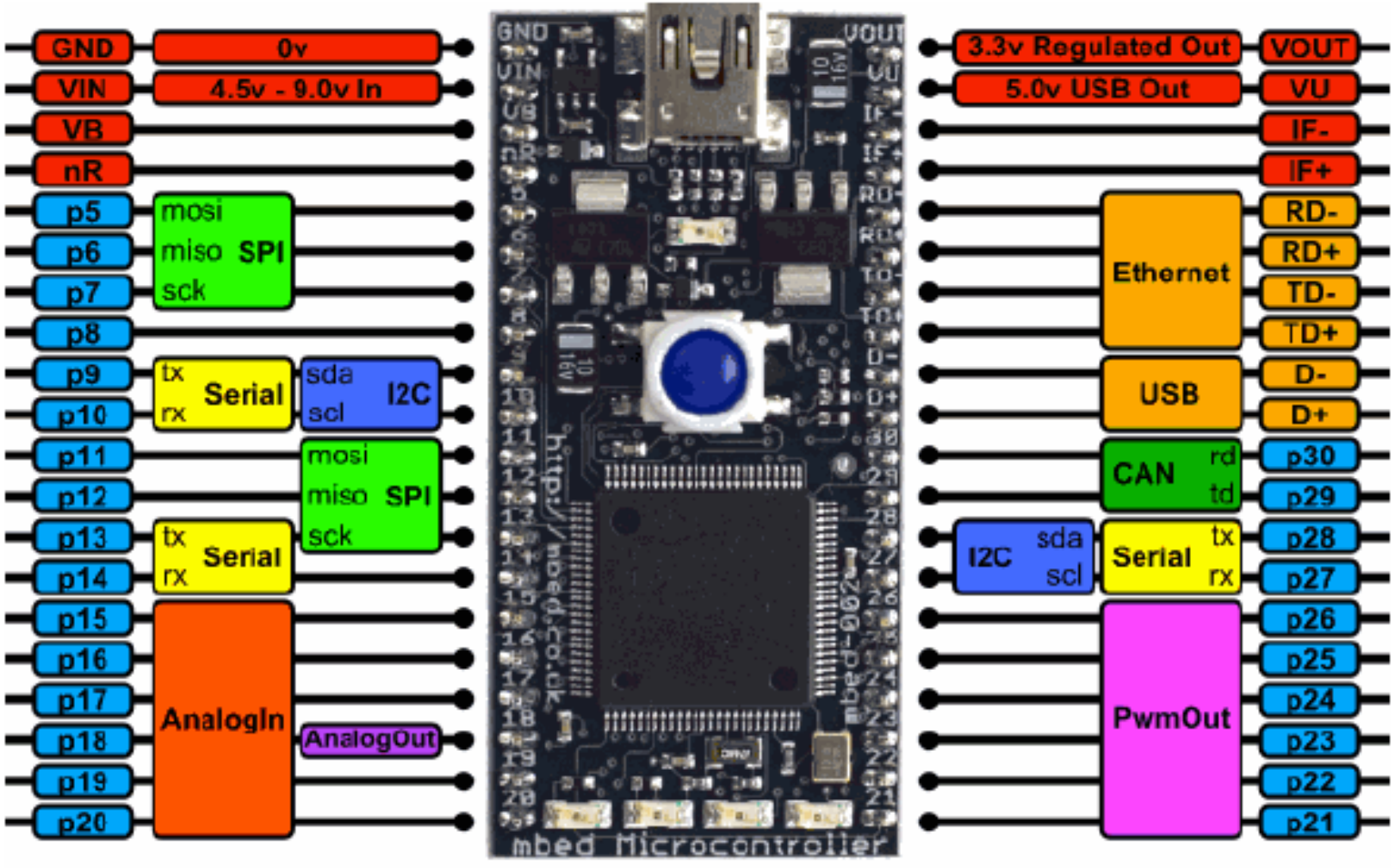
- **STM32L0 Cortex M0+**, 32MHz, 8Ko SRAM, 32-64Ko flash
- **STM32F1 Cortex M1**, 24-72MHz, 4-96Ko SRAM, 16-1024Ko
- **STM32F3 Cortex M4** + FPU, 72MHz, 16-40Ko SRAM, 64-256Ko
- **STM32F7 ARM Cortex-M7F**, 216MHz, 512-1024Ko RAM, . . .
- Chaque famille contient divers périphériques (E/S, calcul, . . .) :
  - 2xADC multiplexé, 2xDAC 12/16 bits
  - USART, SDIO, I2C, SPI, USB, . . .
  - EEPROM, flash, ROM. . .
  - DMA, timers, watchdogs, RTC, RNG. . .

La carte STM32F3-discovery

- STM32F303 : 72MHz, 48Ko RAM, 256Ko flash
- documentation et schéma
- programmeur intégré ST-Link (STM32F0) USB
- port USB utilisateur
- alimentation par USB ou externe (pile)
- accéléromètre/boussole 3D
- gyroscope
- 10 LEDs
- 1 bouton utilisateur



GPIO


GND
0v
VIN
4.5v - 9.0v In
VB
nR
p5
p6
p7
p8
p9
p10
p11
p12
p13
p14
p15
p16
p17
p18
p19
p20
mosi
miso SPI
sck
tx
rx Serial
sda
scl I2C
mosi
miso SPI
sck
tx
rx Serial
AnalogIn
AnalogOut
mbed Microcontroller
3.3v Regulated Out
5.0v USB Out
Ethernet
USB
CAN rd td
I2C sda scl
Serial tx rx
PwmOut
VOUT
VU
IF-
IF+
RD-
RD+
TD-
TD+
D-
D+
p30
p29
p28
p27
p26
p25
p24
p23
p22
p21

Partie 1.

Interfaçage local : capteur et modules

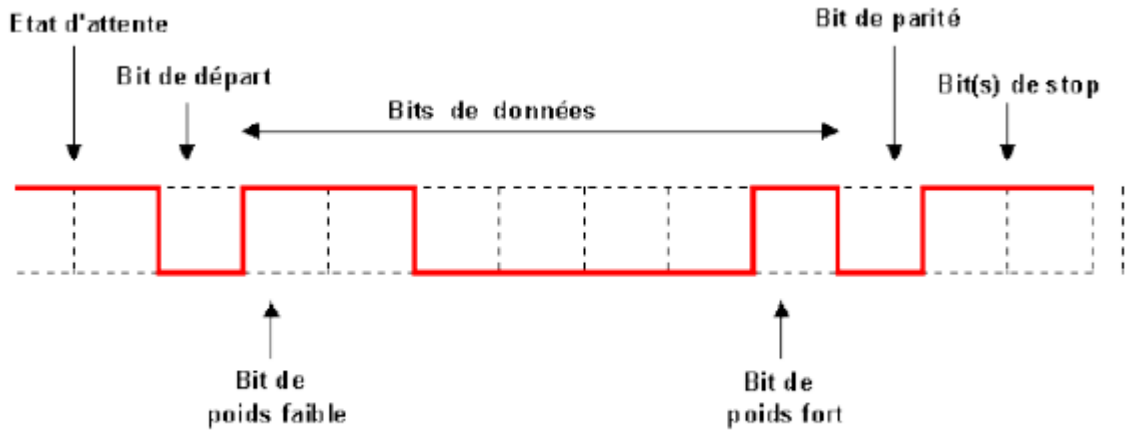
1. Universal Asynchronous Receiver Transmitter (UART)
2. Serial Peripheral Interface (SPI)
3. Le bus I2C

Le bus série UART

- Communication avec un unique circuit
- Communications full duplex et asynchrones

Format d'une liaison série asynchrone

- Les signaux d'une liaison série asynchrone doivent avoir le format ci-dessous:



Exemple: transmission de la lettre "C" (code ASCII: H '43')

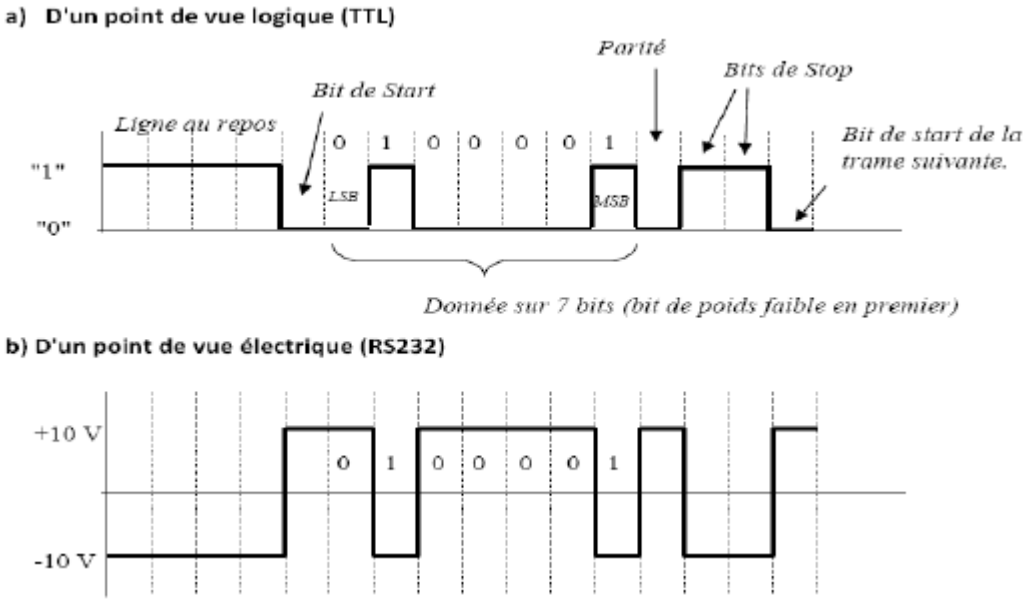
- La transmission s'effectue dans l'ordre suivant :
  - Etat d'attente (niveau logique 1)
  - Envoi d'un bit de départ (niveau logique 0)
  - Envoi des bits de données, on commence par le bit de poids faible, on termine par le bit de poids fort.
  - Eventuellement envoi d'un bit de parité paire ou impaire.
  - Envoi d'un ou de deux bits de stop (niveau 1) indiquant la fin d'émission du caractère.
  - La ligne se retrouve alors en état d'attente (niveau 1), le cycle peut recommencer avec l'envoi d'un nouveau caractère.

Protocole de transmission

- Afin que les éléments communicants puissent se comprendre, il est nécessaire d'établir un protocole de transmission.
- Ce protocole devra être le même pour les deux éléments afin que la transmission fonctionne correctement.
- **Paramètres rentrant en jeu :**
  - **Longueur des mots :** 7 bits (ex : caractère ascii) ou 8 bits
  - **La vitesse de transmission :** les différentes vitesses de transmission sont réglables à partir de 110 bits par seconde (bps) de la façon suivante : 110 bps, 150 bps, 300 bps, 600 bps, 1200 bps, 2400 bps, 4800 bps, 9600 bps …18,2 kbps …56 kbps
  - **Parité :** le mot transmis peut-être suivi ou non d'un bit de parité qui sert à détecter les erreurs éventuelles de transmission. Il existe deux types de parité :
    - **parité paire :** le bit ajouté à la donnée est positionné de telle façon que le nombre des états 1 soit paire sur l'ensemble données + bit de parité.
    - **parité impaire :** le bit ajouté à la donnée est positionné de telle façon que le nombre des états 1 soit impaire sur l'ensemble données + bit de parité.

Protocole de transmission

- Exemple :
- Soit à transmettre en parité paire, 7 bits de données, 2 bits de stop, le caractère "B" dont le codage ASCII est $(42)_{16}$ ou $(1000010)_2$;
- La trame sera la suivante :

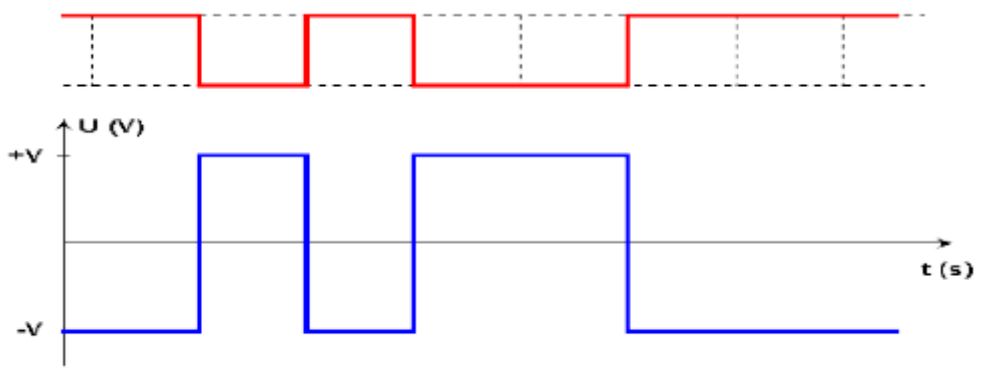


Protocole de mise en œuvre

- Pour que deux éléments d'un système puissent communiquer à l'aide d'une liaison série asynchrone, il faut que ces deux équipements soient configurés de la même manière.
- Cette configuration appelée protocole de communication doit pour une liaison série préciser :
  - le nombre de bits de données (datas).
  - l'utilisation ou non du bit de parité. S'il est utilisé, préciser si la parité est paire (even) ou impaire (odd).
  - le nombre de bits de stop (1 ou 2).
  - la vitesse de transmission en Bauds ou en bits par seconde.
- **NB:** le nombre total de bits pour l'envoi d'un caractère ne devra pas dépasser 11 (du bit de départ au bit de stop). Le protocole suivant est donc interdit : 1 bit de start, 8 bits de données, 1 bit de parité et 2 bits de stop.

Liaisons tension

- MODE ASYMETRIQUE
  - En mode asymétrique les états logiques sont transmis sur la ligne par deux niveaux de tension, l'un positif l'autre négatif.

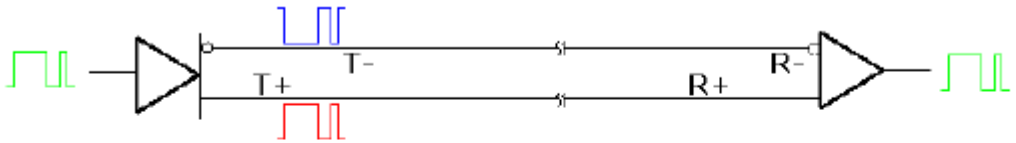


Etats logiques et niveaux de tension d'une liaison asymétrique

  - La liaison tension asymétrique la plus utilisée travaille en logique négative. Le niveau logique 1 est défini par une tension négative, le niveau 0 par une tension positive.
  - Les systèmes basés sur la transmission en mode asymétrique sont sensibles aux parasites induits. De ce fait le débit nominal maximum et la longueur maximum du câble sont de 20 kBauds et de 15 mètres.

Liaisons tension

- MODE SYMETRIQUE
  - La liaison symétrique ou différentielle permet de transmettre des données sur de grandes distances à des vitesses élevées. Elle est peu sensible aux parasites induits, ceux ci affectent les deux fils de la ligne et se trouvent inhibés par l'entrée différentielle du récepteur.
  - Sur une liaison différentielle (ou symétrique) les signaux (T+ et T-) sont transmis en opposition de phase Le récepteur réalise la différence de ces deux signaux (R+ et R-) pour obtenir le signal utile



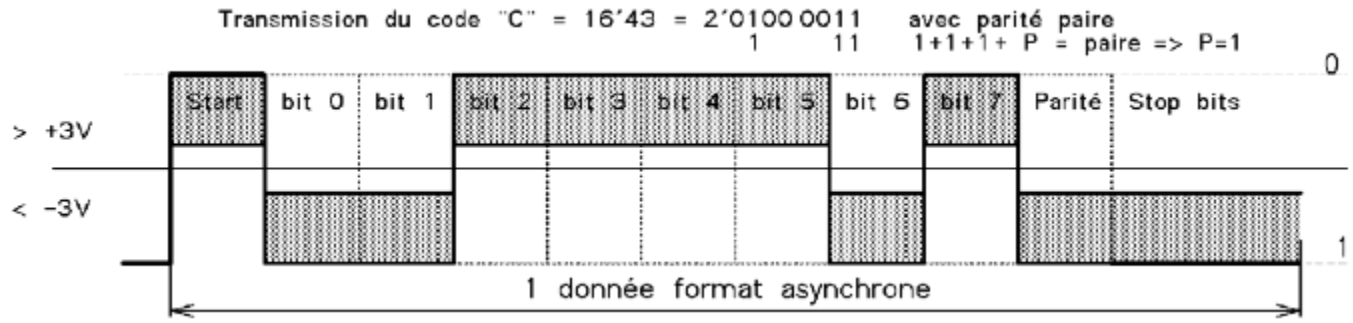
Si une **perturbation** se produit, elle se présente sur les deux fils avec la même polarité
Le récepteur réalise la différence des deux signaux : la perturbation n'est pas transmise au signal utile

Liaison RS-232

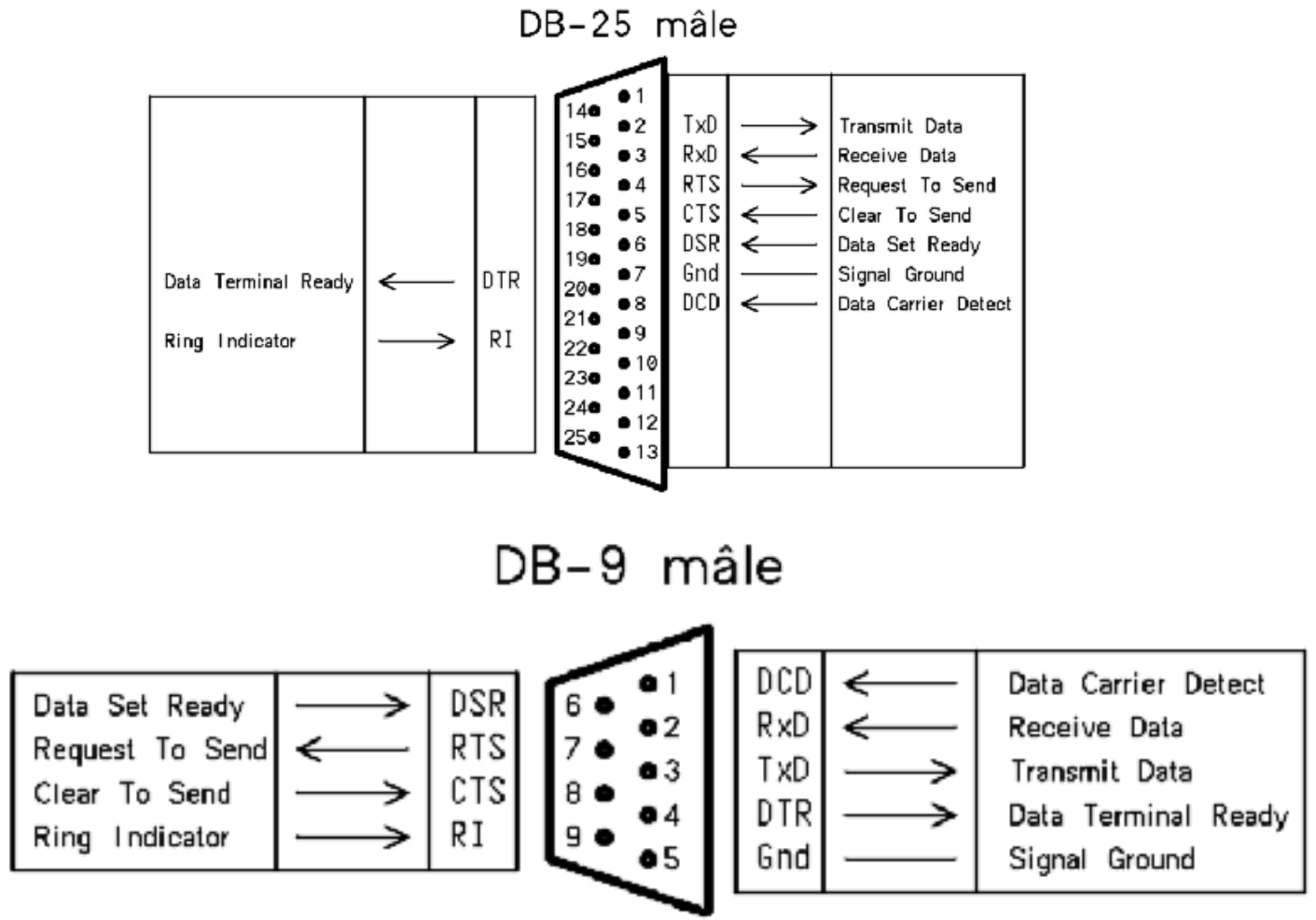
- La liaison RS-232 est issue de la norme du même nom qui permet l'envoi de données via une chaîne de niveaux logiques envoyés en série (d'où le nom du port du PC).
- Elle permet de faire dialoguer deux systèmes (et seulement deux) entre eux. Les données sont envoyées par trames de 5, 6, 7 ou 8 bits.
- Cette liaison est asymétrique : de type asynchrone, c'est-à-dire qu'elle n'envoie pas de signal d'horloge pour synchroniser les deux intervenants de la liaison : il est donc nécessaire que ces derniers soient configurés de la même manière (vitesse de transmission, nombre de bits par trame, etc.). La vitesse de transmission s'exprime en bauds (bps = bits par seconde) : les valeurs les plus courantes sont 2400, 4800 et 9600 bauds.
- Les niveaux logiques ont une grande marge d'erreur, ce qui permet à la liaison RS-232 de n'être que peu sensible aux perturbations, et donc de pouvoir être mise en place sur de longues distances. En effet, le niveau logique "zéro" est représenté par une tension comprise entre +3 et +15 V, et le niveau logique "un" est représenté par une tension comprise entre -3 et -15 V.
- Longueur maxi du câble est de 15m
- Liaison point à point.

Liaison RS-232

- Inconvénients :
  - faible débit
  - faible longueur
  - Fonctionnement asymétrique
  - liaison pt à pt
- Exemple, RS-232
- Transmission code "C", parité paire :
  - ASCII C 16'43 2' 0100 0011
  - LSb (bit 0) en premier
  - '0' > +3V
  - '1' < -3V



Connecteur RS-232, 9 pôles



Exemple, RS-232

- 3 signaux sont indispensables pour la communication:
  - **Gnd**: Terre de transmission (pin 7)
  - **TxD**: transmission de données
  - **RxD**: réception de données
- Viennent ensuite 2 signaux de gestion du contrôle de flux de transmision entre l´émetteur et le récepteur:
  - **RTS**: *Requets To Send*, l´émetteur désire émettre
  - **CTS**: *Clear To Send*, le récepteur autorise l´émetteur à émettre
- Un signal est utilisé pour indiquer que la communication est établie:
  - **DCD**: *Data Carrier Detect*, la porteuse est valide
- 2 signaux indiquent que les équipements sont prêts pour communiquer:
  - **DTR**: *Data Terminal Ready*, le terminal est prêt
  - **DSR**: *Data Set Ready*, le modem est prêt
- Un dernier signal utilisé avec certains modems:
  - **RI**: *Ring Indicator*, sonnerie

Caractéristiques des liaison tensions

- Le tableau ci dessous résume les principales caractéristiques des liaisons séries asynchrones en tension.

Paramètre	RS 232 D	RS 422 A	RS 485
Mode fonctionnement	Asymétrique	Symétrique différentiel	Symétrique différentiel
Nombre émetteurs	1	1	32
Nombre récepteurs	1	10	32
Longueur maximum (m)	15	1 200	1 200
Débit maximum (bauds)	20 K	10 M	10 M

UART

- Un UART, pour Universal Asynchronous Receiver Transmitter, est un émetteur-récepteur asynchrone universel.
- En langage courant, c'est le composant utilisé pour faire la liaison entre l'ordinateur et le port série . L'ordinateur envoie les données en parallèle (autant de fils que de bits de données).
- Il faut donc transformer ces données pour les faire passer à travers une liaison série qui utilise un même fil pour faire passer tous les bits de données.
- La communication peut être «full duplex» (à la fois envoyer et recevoir en même temps) ou «half duplex» (dispositifs à tour de rôle émission et de réception).
- En 2008, UART sont couramment utilisés avec RS-232 pour les communications des systèmes embarqués.
- Il est utile de communiquer entre les microcontrôleurs et également avec les PC.

Constitution d'une trame UART

- Une trame UART est constituée des bits suivants :

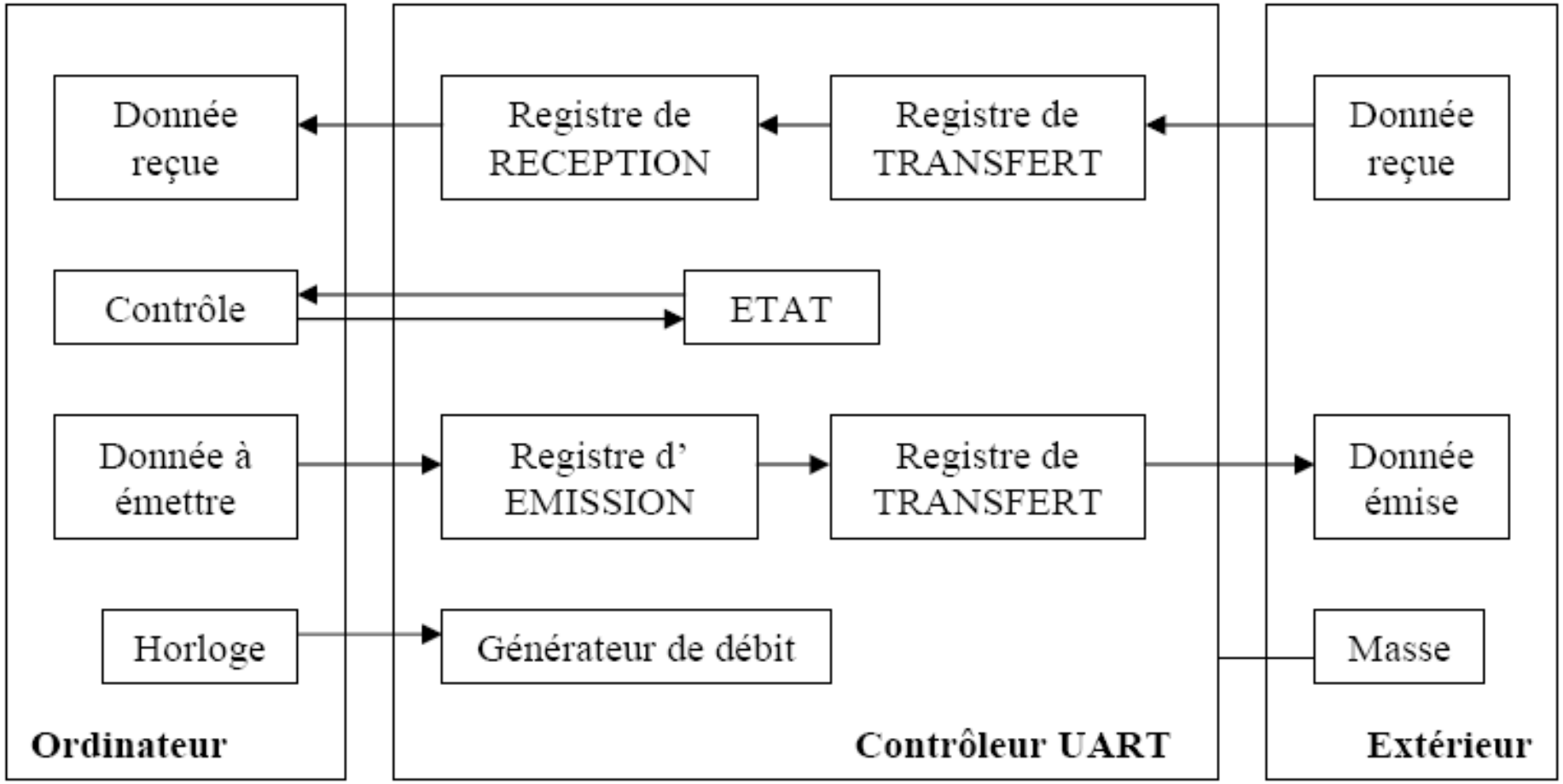
Constitution d'une trame UART

  - un bit de start toujours à 0 : servant à la synchronisation du récepteur
  - les données : la taille peut varier (généralement entre 5 et 9 bits)
  - éventuellement un bit de parité paire ou impaire
  - un bit de stop toujours à 1 (la durée peut varier entre 1, 1,5 et 2 temps bit)
  - Le niveau logique de repos est le 1.

Vitesse de transmission - UART

- Afin de faciliter l'interopérabilité entre périphériques (PC, microcontrôleur, modem, …) des vitesses de transmission sont normalisées, l'unité baud correspondant à un temps bit :
  - 110 bps
  - 300 bps
  - 1 200 bps
  - 2 400 bps
  - 4 800 bps
  - 9 600 bps
  - 19 200 bps
  - 38 400 bps
  - 57 600 bps
  - 115 200 bps
  - 230 400 bps (selon la fréquence d'oscillation employée)
  - 460 800 bps
  - 921 600 bps (« environ 1 mégabaud »)
  - 1 843 200 bps
  - 3 686 400 bps

Diagramme UART



UART

- Il faut au minimum trois fils pour établir une connexion :
  - TD (Transmit Data) : broche 2 du connecteur
  - RD (Receive Data) : Broche 3 du Connecteur
  - GR (Ground) : Broche 7 du connecteur

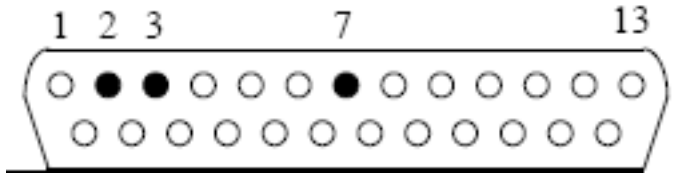
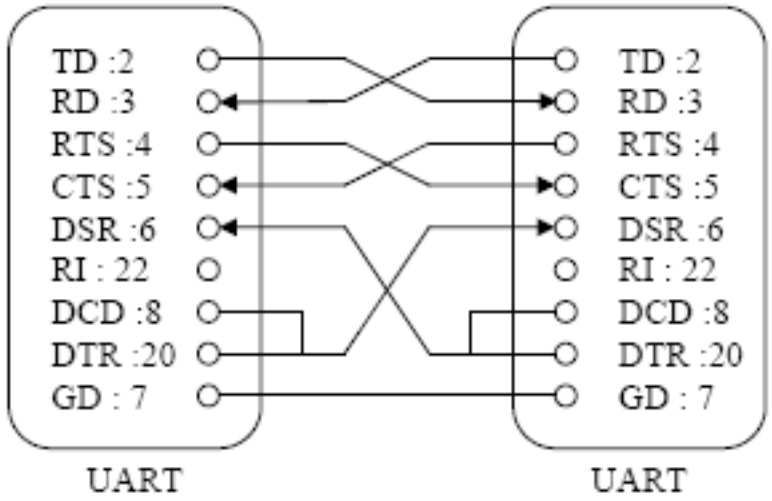


Fig. 17:

Connexion "Modem nul"
UART ↔ UART

Cette liaison permet de connecter deux ordinateurs sans modem en utilisant les signaux de contrôle des modems. Cette technique a pour avantage de simuler la présence d'un modem.



Exercice

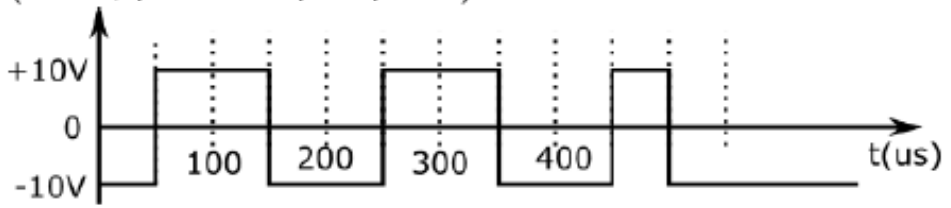
Exercices : page 1

1. On observe à l'oscilloscope le signal suivant sur une liaison asynchrone à 8 bits, sans parité ni bit de stop.
   Quel est l'octet transmis ?

2. On reçoit le signal suivant d'une liaison liaison asynchrone à 8 bits, sans bit de stop, mais avec parité paire.
   - Quel est l'octet transmis ?
   - Le bit de parité est-il correct ?

Exercices : page 2

- Sur une liaison RS232, on observe à l'oscilloscope le signal suivant (8 bits, pas de stop ni parité)

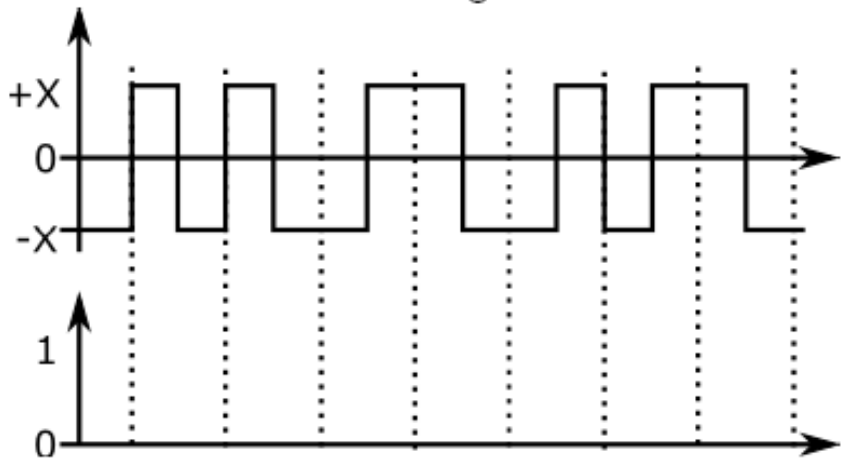


- Quel est l'octet transmis :
- Quel est le débit en Bauds de la liaison :
- Quel est le débit normalisé qui se rapproche le plus :
- Que vaut l'erreur (%) :

Exercice

Exercices : page 3

- On observe à l'oscilloscope le signal suivant sur une liaison à codage "Manchester". Dessiner le signal binaire encodé.



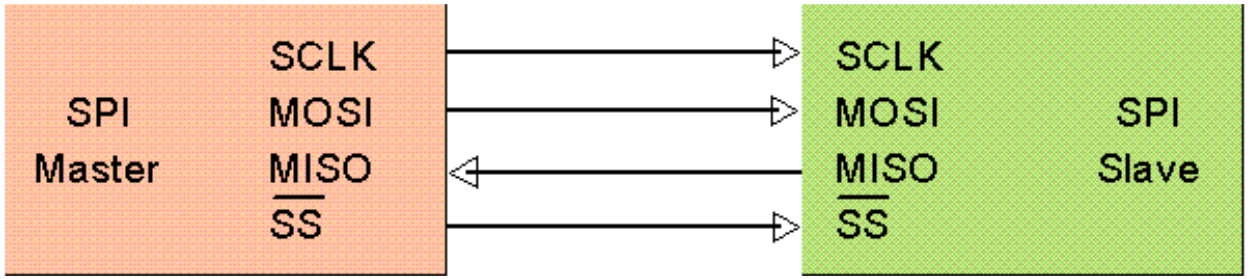
Partie 1.

Interfaçage local : capteur et modules

1. Universal Asynchronous Receiver Transmitter (UART)
2. Serial Peripheral Interface (SPI)
3. Le bus I2C

Communication série

- Une liaison **SPI** (pour Serial Peripheral Interface) est un bus de donnée série synchrone baptisé ainsi par Motorola, et qui opère en Full Duplex.
  - Les circuits communiquent selon un schéma maître-esclaves, où le maître s'occupe totalement de la communication.
  - Plusieurs esclaves peuvent coexister sur un bus, la sélection du destinataire se fait par une ligne dédiée entre le maître et l'esclave appelée chip select.



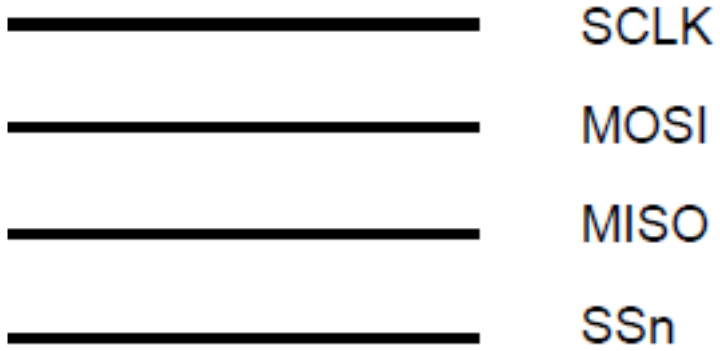
SIMO ou MOSI: **S**lave **I**nput –**M**aster **O**utput
SOMI ou MISO : **S**lave **O**utput –**M**aster **I**nput
SCLK ou UCLK: **U**sart SPI **CL**oc**K**
STE ou SS : **S**lave **T**ransmit **E**nable (aussi **S**lave **S**elect)

Communication série

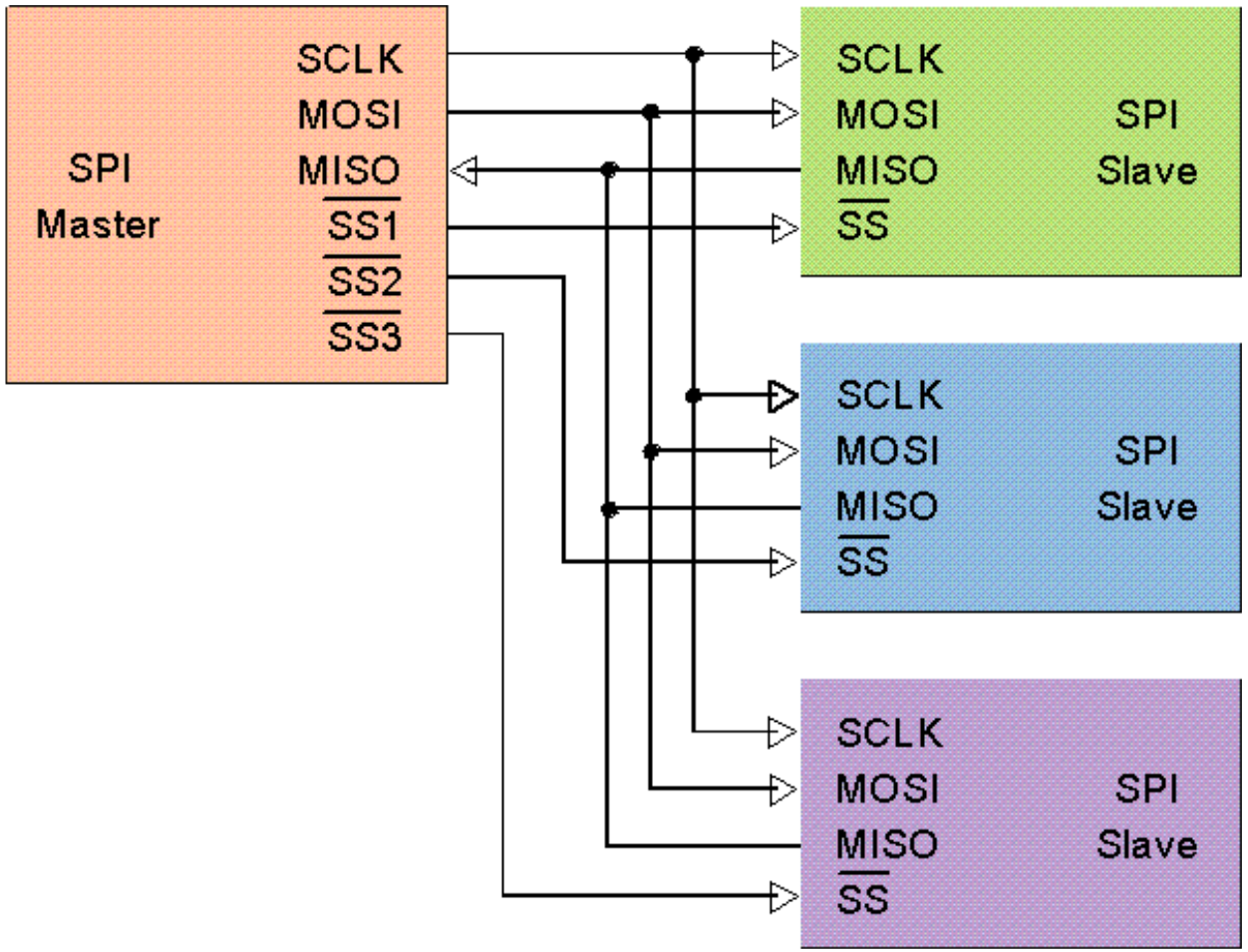
- Les caractéristiques du mode **SPI** comprennent :
  - des données de longueur de 7-bits ou 8-bits
  - interface nécessitant 3 ou 4 lignes
  - mode en **Master** (maître) ou en **Slave** (esclave)
  - registre à décalage de transmission et de réception indépendant
  - buffer de ligne de transmission et de réception séparés
  - polarité et contrôle de phase de l'horloge **UCLK**
  - fréquence programmable pour l'horloge **UCLK** en mode Master
  - source d'interruption indépendante pour la transmission et la réception

Le support physique

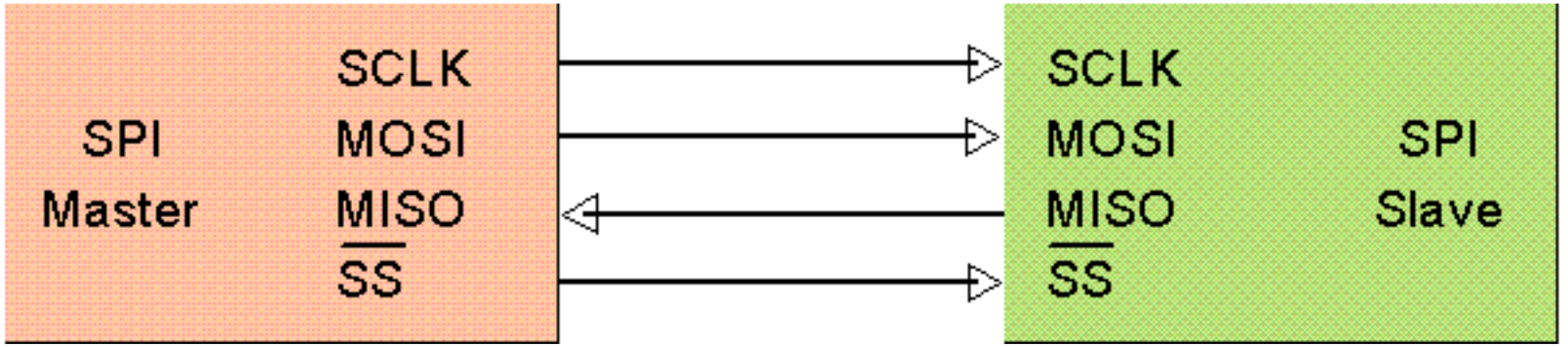
- Les données échangées sont des octets. La transmission s'effectue sur 2 fils monodirectionnels (nommés **MOSI**, **MISO**).
- Une horloge indépendante fixée par le maître synchronise les échanges (en général sur front).
- La fréquence de l'horloge de transmission est comprise entre 1 Mhz et 20 Mhz (selon les performances des circuits reliés au bus).
- Il n'y a pas d'adressage des esclaves (comme sur un bus **I2C** par exemple).
- L'esclave devient actif au moyen d'une ligne de sélection de boîtier dédiée (généralement active à l'état bas).
- La ligne est constituée de 3 fils auxquels il faut ajouter les fils de sélection d'esclave.



Liaison SPI avec un maître et trois esclaves

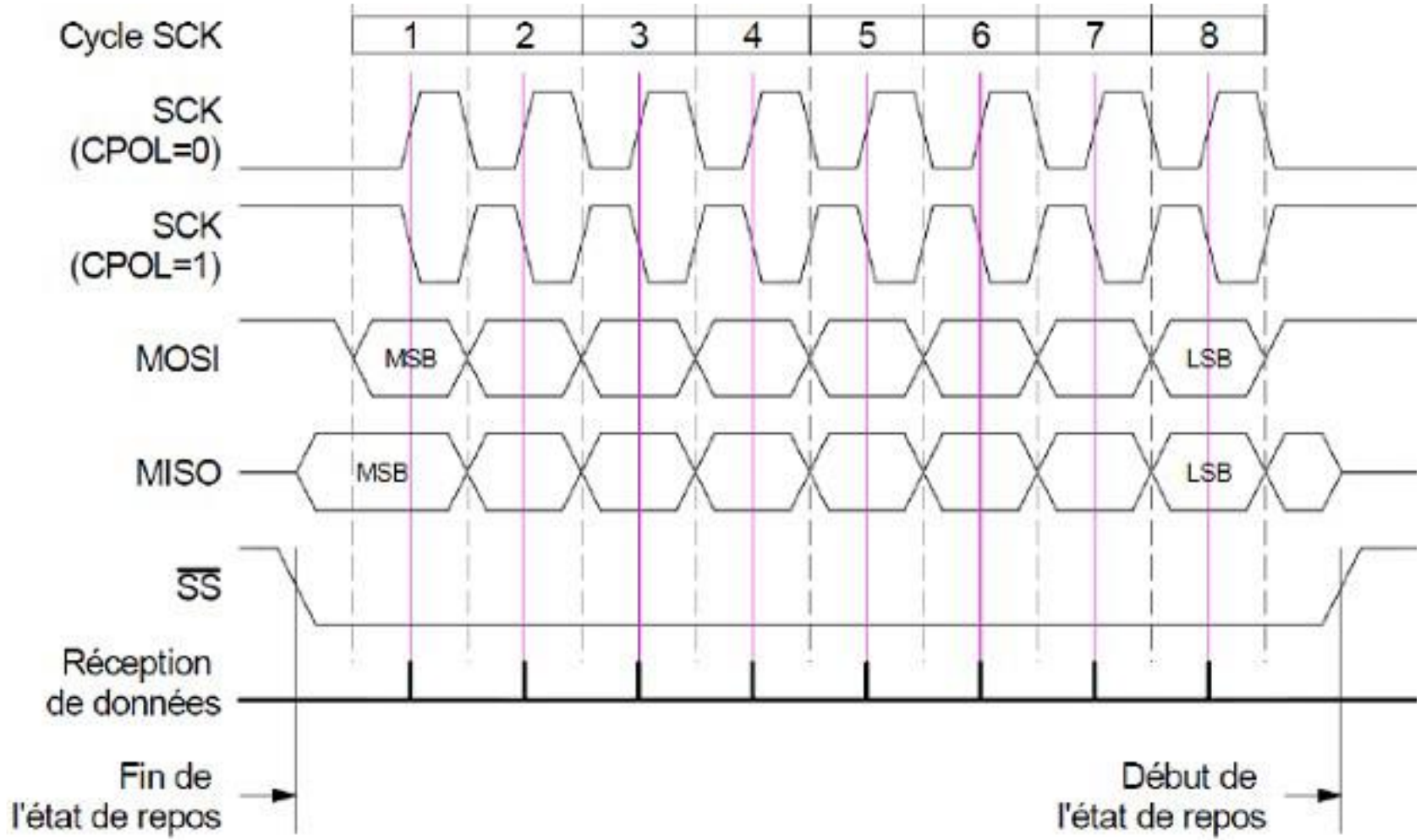


Fonctionnement



- Une transmission **SPI** typique est une communication simultanée entre un maître et un esclave.
  - **Le maître** génère l'horloge et sélectionne l'esclave avec qui il veut communiquer
  - **L'esclave** répond aux requêtes du maître
- A chaque coup d'horloge le maître et l'esclave s'échangent un bit.
- Après huit coups d'horloges le maître a transmis un octet à l'esclave et vice-versa.
- La vitesse de l'horloge est réglée selon des caractéristiques propres aux périphériques.

Fonctionnement



Avantages et Inconvénients du bus SPI

Avantages	Inconvénients
Communication en Full Duplex	Pas d'adressage possible
"Indépendant" du nombre de bits à transmettre	Utilisation sur très courte distance (même carte)
Pas de collision possible	Nécessite plus de fils que I²C
Les esclaves utilisent l'horloge du maître pas de problème de précision de quartz	Pas d'acquittement (le maître ne sait pas s'il est écouté)
Beaucoup plus rapide que I²C en mode standard	
Possibilité de configuration à plusieurs maîtres	

Partie 1.

Interfaçage local : capteur et modules

1. Universal Asynchronous Receiver Transmitter (UART)
2. Serial Peripheral Interface (SPI)
3. Le bus I2C

Bus I2C

- Bus sériel 2 fils pour données et contrôle
- Une ligne de données **SDA**
- Une ligne d'horloge **SCL**
- Conditions uniques de **start** et **stop**
- Possibilité d'avoir plusieurs maîtres (un seul à la fois)
- Plusieurs esclaves, sélectionnés par une adresse 7-Bit
- Transfert bidirectionnel de données
- Acknowledge après chaque octet transféré
- Pas de limite au nombre d'octet transférés

Familles de circuits avec I2C

- Microcontrôleurs
- Microprocesseurs
  - Périphériques généraux
  - Interfaces I2C-parallèle, I/O
  - Mémoires, Afficheur, DAC, ADC, Horloges (RTC)
- Périphériques pour applications spécifiques
  - Audio, Téléphonie, Vidéo

Caractéristiques

- Synchronisation d'horloge
- Procédure d'arbitration
- Vitesse de transmission jusqu'à 100Khz
- Longueur de bus jusqu'à 4 mètres
- Capacité pilotable jusqu'à 400pF
- Possibilité de placer des résistances en série pour des questions de protection des circuits
- Compatible avec la plupart des technologies (TTL, CMOS, etc…).

Définitions

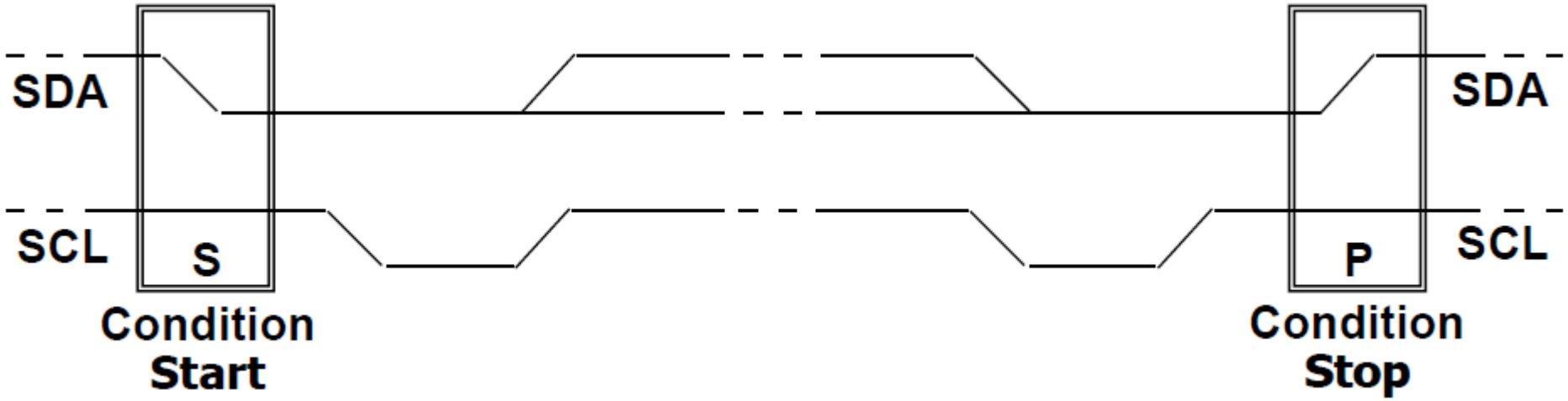
- **Maître** :
  - Initie un transfert en générant des conditions de **start** et **stop**
  - Génère l'horloge
  - Transmet l'adresse de l'esclave
  - Détermine le sens du transfert de données
- **Esclave** :
  - Répond seulement quand il est adressé
  - Le timing est contrôlé par la ligne d'horloge **SCL**

Aspects matériels

- Les circuits connectés sur le bus doivent avoir un accès “Drain ouvert” pour les deux lignes **SDA** (données) et **SCL** (horloge)
- Les circuits doivent être capables de détecter le niveau logique
- Tous les circuits doivent avoir une référence **GND** commune
- Les lignes **SCL** et **SDA** sont reliées à **VCC** via des résistances de pull-up
- A tout moment, le bus **I2C** est dans l’un des états suivants :
  - Au repos
  - En mode de transmission maître
  - En mode de réception esclave

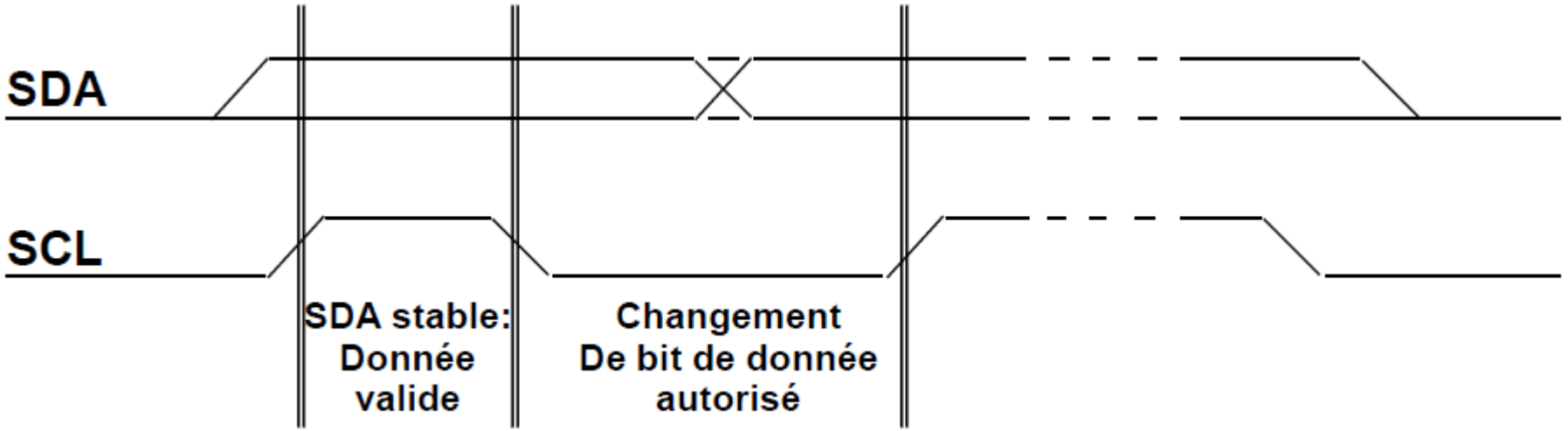
Conditions de Start et Stop

- Une transition sur **SDA**, alors que **SCL** est à '1', est définie comme une condition de **Start** ou **Stop**
- Ces deux conditions (**Start** et **Stop**) sont générées par le maître du bus
- Le **Bus** est occupé (busy) après une condition de **Start**



Transferts de données

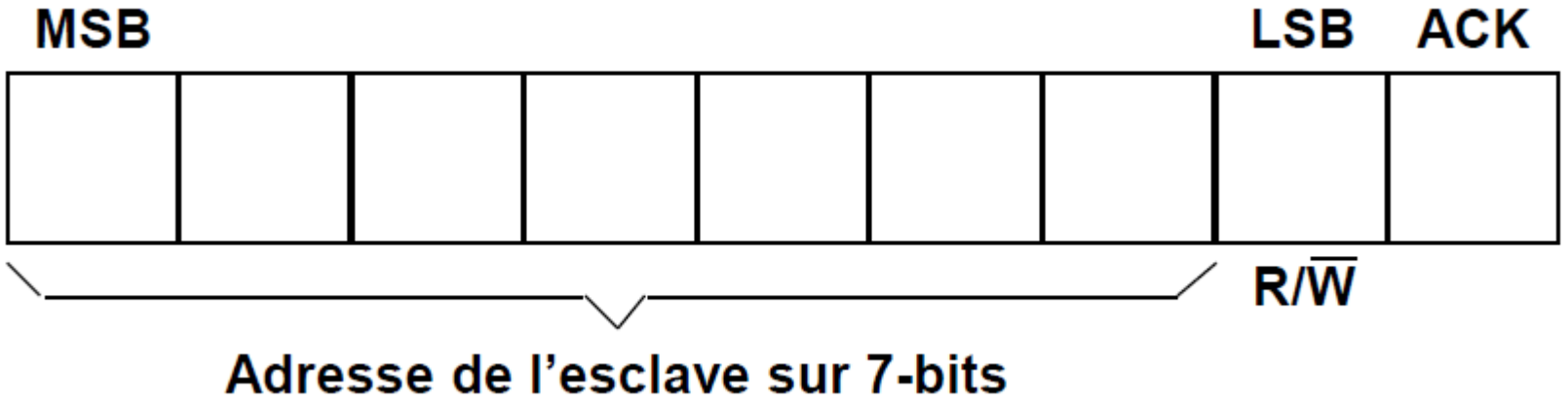
- En mode de transfert normal, la ligne de données **SDA** change d'état seulement lorsque **SCL** est à '0'



Adressage I2C

- Chaque nœud (périphérique) a une adresse unique sur 7 bits
- Généralement, les nœuds ont une partie fixe et une partie programmable de leur adresse
- Les adresses commençant par **0000** ou **1111** ont des fonctions spéciales
  - **0000000** est une adresse globale
  - **0000001** est une adresse sans objet (Null)
  - **1111xxxx** est réservée pour les expansions du bus

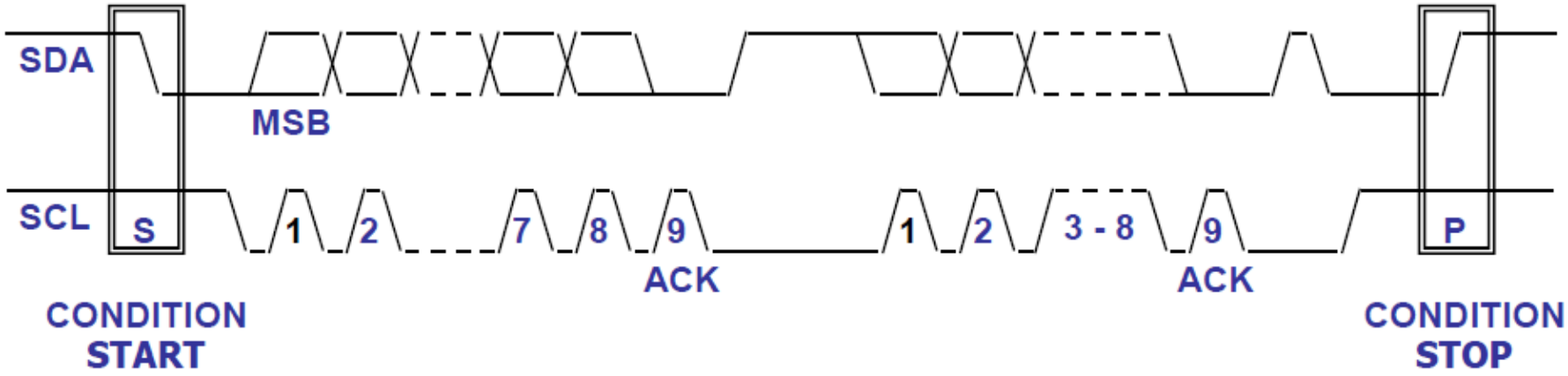
Premier octet transmis



- R/W :
  - **0** : Le maître écrit vers l'esclave
  - **1** : Le maître lit l'esclave

Acknowledgement

- Les récepteurs (**Maître** ou **Esclave**) tirent la ligne de donnée **SDA** à '0' durant une période d'horloge après réception d'un octet.
- Les récepteurs **Maître** laissent la ligne **SDA** à '1' après réception du dernier octet demandé.
- Les récepteurs **Esclave** laissent la ligne **SDA** à '1' sur l'octet qu'ils peuvent accepter.



Partie 1.

Interfaçage système et bus externes

1. Le bus USB
2. Le bus CAN

USB

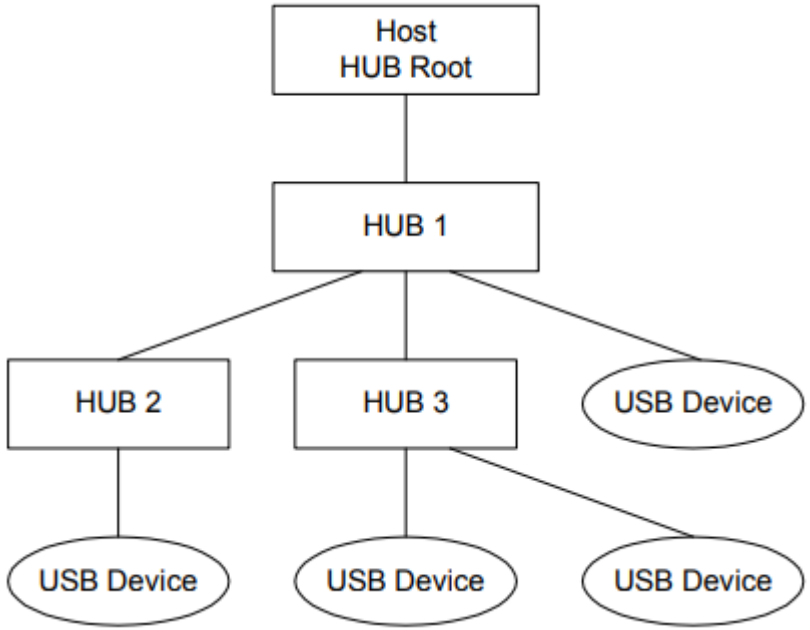
- **USB**: Universal Serial Bus
- Le USB a été conçu afin de remplacer le port série
- L'utilisation du USB implique des redevances. Être membre officiels des développeurs de USB coûte 2500$ USD par année.
- Avoir un Vendor_ID coûte 200$ USD par année…
- Il existe trois normes USB principales:
  - **USB 1.1** (1998)
  - **USB 2.0** (2000, révisé en 2002).
  - **USB 3.0** (2008)
- Dans le cours, nous allons nous concentrer sur USB 2.0 (avec quelques mentions de 3.0)

Versions de USB & vitesses de communication

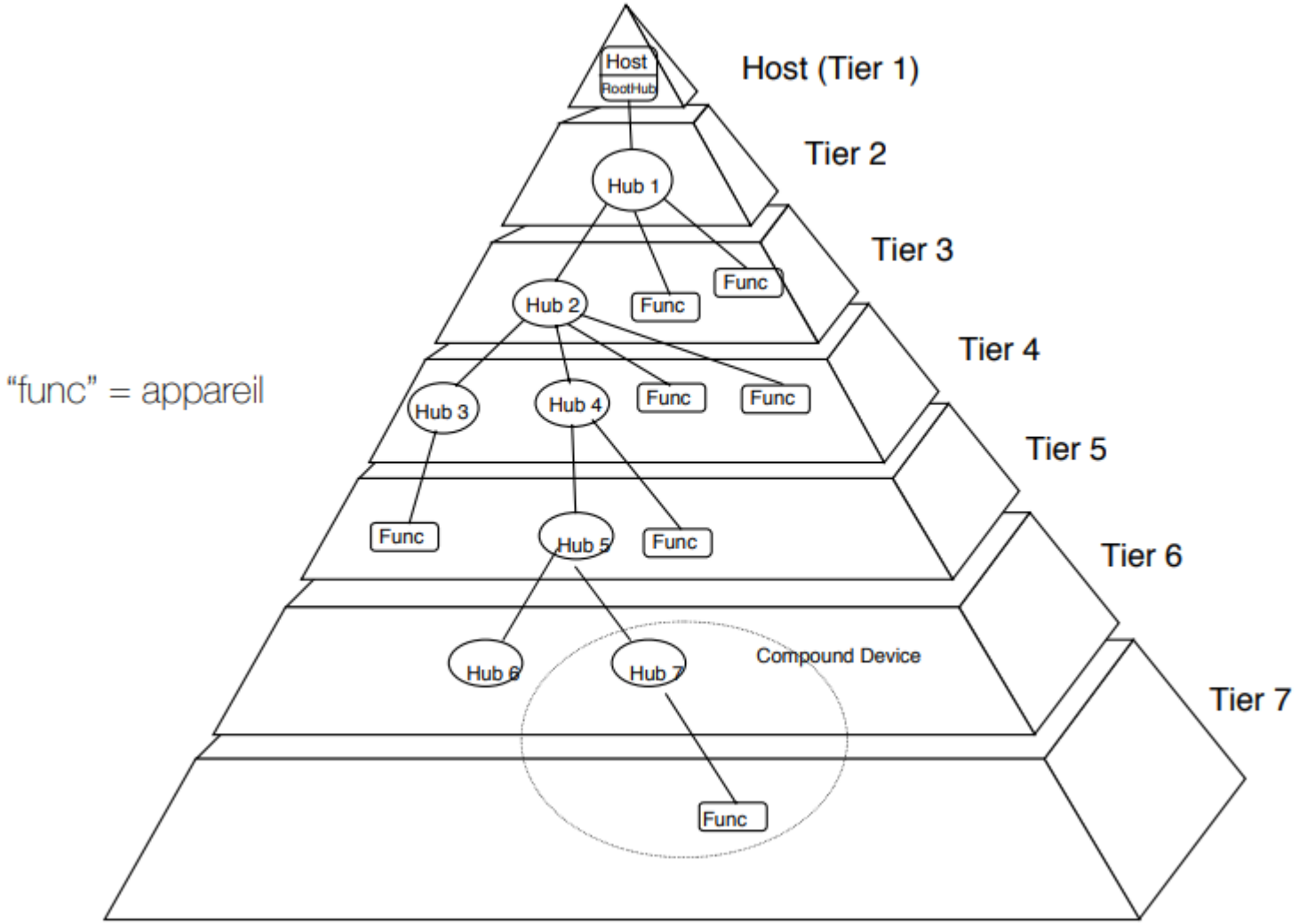
- 1.0 “Low Speed”: 1.5 Mb/s
  - claviers, souris, joystick
- 1.0 “Full Speed”: 12 Mb/s
  - appareils à gros débits comme les modems et téléphones.
- 2.0 “High Speed”: 480 Mb/s
  - appareils multimédia à très haut débit comme les caméras numériques
- 3.0 “SuperSpeed”: 5 Gb/s
- 3.1 “SuperSpeed+”: 10 Gb/s

Topologie d'un réseau USB

- Un **réseau USB** a une topologie en étoile.
- Le port USB est contrôlé entièrement par un contrôleur unique appelé hôte ("host"). Souvent le PC, il initie toutes les communications, et est le maître absolu du bus.
- Les "hubs" permettent de relier plusieurs appareils à un seul port USB.
  - Le rôle principal des hubs est de transférer les données de l'hôte aux périphériques.
  - Chaque hub contrôle ses ports afin de savoir si un appareil s'y connecte
  - Il peut y avoir 5 niveaux de hub en plus du hub racine.
- Il y a 127 appareils maximum dans un réseau USB.
- Chaque appareil a son adresse.



Topologie d'un réseau USB



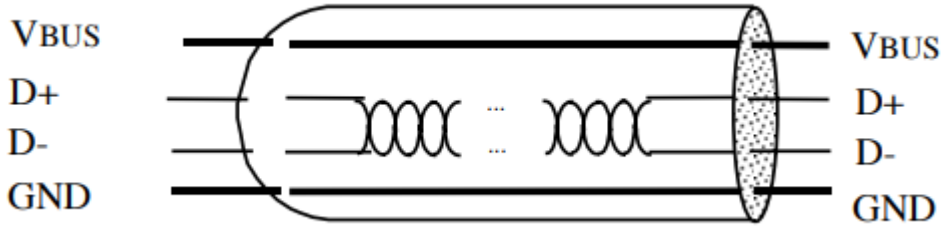
Classes d'appareils USB

- Tous les périphériques **USB** appartiennent à une classe.
- Les classes d'appareils sont:
  - Audio (Haut parleurs)
  - Communication (Modem)
  - Souris, Claviers
  - Écrans
  - Physical feedback devices (Force feedback joystick)
  - Alimentation (UPS—batterie de backup)
  - Imprimante
  - Disque dur
  - Hub!

Matériel — fils

- Le câble **USB** est constitué de 4 fils:
  - **Vbus** est l'alimentation 5Vdc (entre 4.75V et 5.25V)

  (peut alimenter les appareils branchés sur le bus!)
  - **D-** et **D+** servent au transport des données.
  - **GND** est la référence électrique



Matériel — connecteurs et mode différentiel

- Il y a deux types de connecteurs **USB**: **A** et **B**.
- Pour garantir la topologie étoile, les connecteurs:
  - **A** "pointent" toujours vers le haut, vers l'hôte
  - **B** "pointent" toujours vers le bas, vers les périphériques

- **USB** utilise le mode différentiel:
  - les données sont transmise par la différence entre **D+** et **D-**.
  - lorsque **D+** et **D-** sont forcés près de la masse par l'hôte, cela signifie une demande de reset.

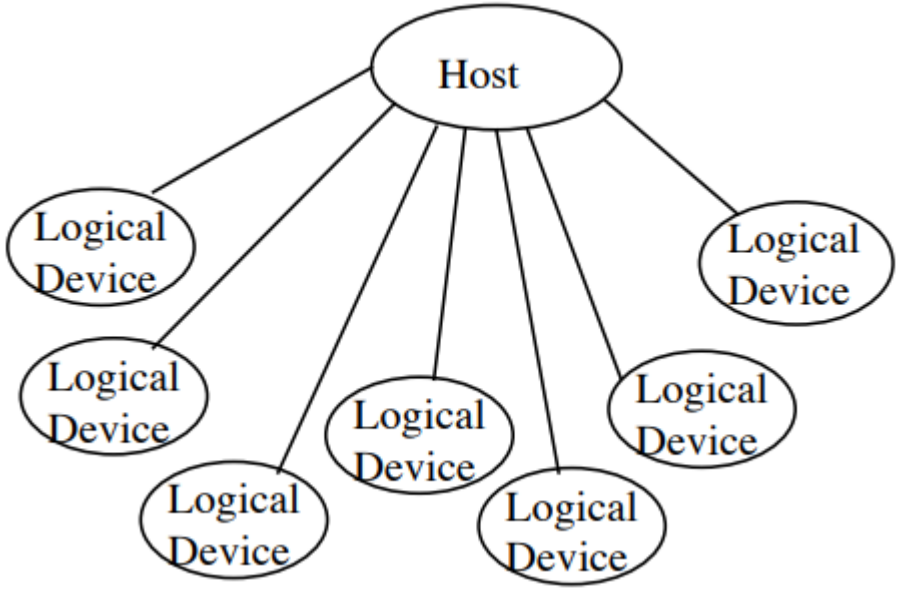
Matériel — NRZI

- **NRZI**: "Non Return to Zero Inverted"
- Habituellement, nous encodons les bits directement
  - "0" ou "1" logiques
- **NRZI** encode les bits par des changements d'états
  - changement d'état == "0"
  - état constant (pas de changement) == "1"

Bit à envoyer	État précédent	Nouvel état
0	0	1
0	1	0
1	0	0
1	1	1

Logiciel — structure logique

- Bien que les appareils soient branchés suivant une structure en étoile, l'hôte communique avec les appareils comme s'ils étaient branchés ensemble directement
- Si un **hub** est retiré, tous les appareils branchés au **hub** le sont aussi

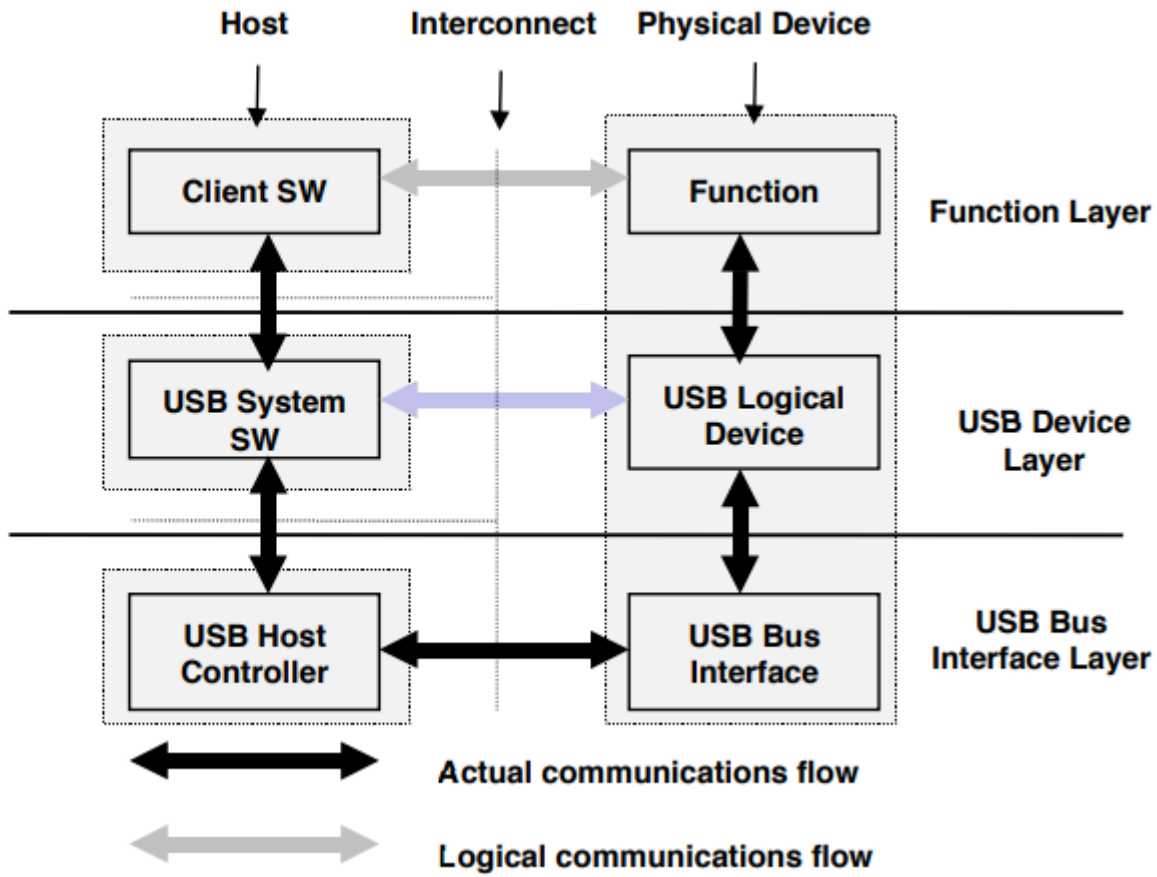


USB — matériel et logiciel

Couche haut-niveau
(fonctionnalités spécifiques)

Couche logique
(fonctionnalités générales)

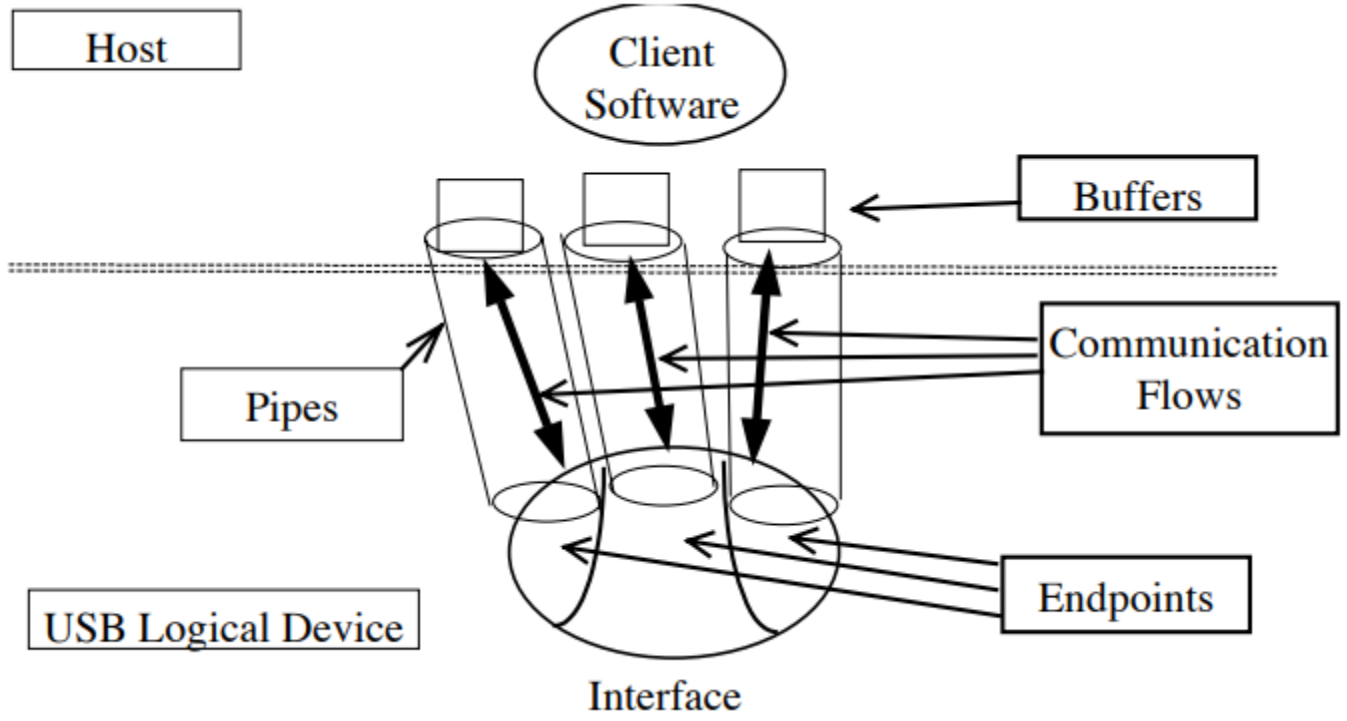
Couche physique
(encodage, paquets)



Logiciel — composantes logiques

- Chaque appareil USB contient une ou plusieurs connections logiques appelées terminaisons (“endpoints”).
- L’hôte envoie ou reçoit des données à partir d’une terminaison.
- Chaque appareil possède au moins une terminaison: la terminaison 0.
  - C’est avec cette terminaison que l’hôte communique avec d’obtenir des informations sur les configurations d’un appareil.
- Un groupe de terminaisons est appelé une interface.
- Un groupe d’interfaces est appelé une configuration.
- Un appareil peut avoir plusieurs interfaces, réparties dans plusieurs configurations. Toutefois, la plupart des appareils n’ont qu’une seule configuration, qui ne contient qu’une seule interface.
- Le lien entre l’hôte et une terminaison est appelé canal (“pipe”).

Logiciel — composantes logiques

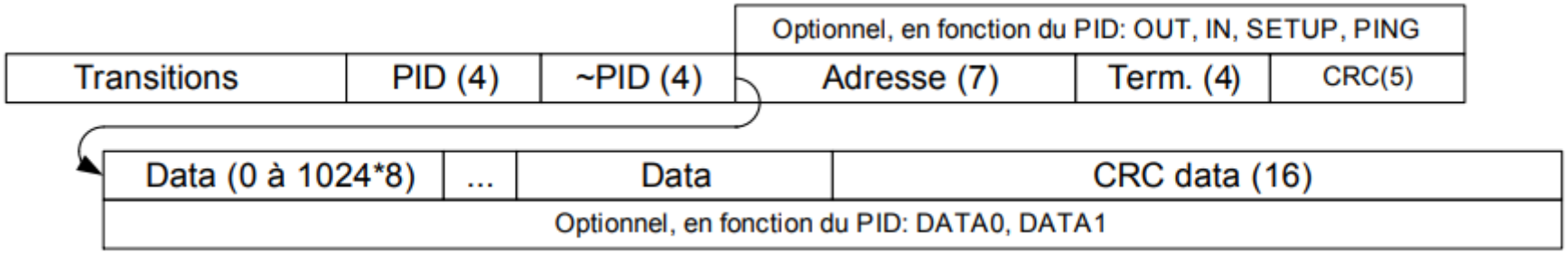


Logiciel — transactions

- Les transactions USB se font à l'aide de jetons, c'est-à-dire à l'aide de bytes transmis ayant des valeurs et significations précises.
- Les jetons et messages transitent dans des trames de 1ms ou de 125us (haute vitesse).
- Les appareils nouvellement branchés sur le port USB sont détectés automatiquement.

Logiciel — paquets

- Un paquet est divisé en plusieurs parties:
  - **Transition (“SYNC”):** 10101010 en succession rapide pour synchroniser les horloges
  - **“PID”:** “packet identifier”, qui indique le type de paquet. On ajoute aussi une version inversée de ce PID.
  - **Adresse:** appareil à qui s’adresse ce paquet, et la terminaison employée
  - **Données:** données à transmettre
  - **Correction d’erreur (“**Cyclic Redundancy Check”, CRC)



Types de paquets

PID Type	PID Name	PID<3:0>*	Description
Token	OUT	0001B	Address + endpoint number in host-to-function transaction
	IN	1001B	Address + endpoint number in function-to-host transaction
	SOF	0101B	Start-of-Frame marker and frame number
	SETUP	1101B	Address + endpoint number in host-to-function transaction for SETUP to a control pipe
Data	DATA0	0011B	Data packet PID even
	DATA1	1011B	Data packet PID odd
	DATA2	0111B	Data packet PID high-speed, high bandwidth isochronous transaction in a microframe (see Section 5.9.2 for more information)
	MDATA	1111B	Data packet PID high-speed for split and high bandwidth isochronous transactions (see Sections 5.9.2, 11.20, and 11.21 for more information)

PID Type	PID Name	PID<3:0>*	Description
Handshake	ACK	0010B	Receiver accepts error-free data packet
	NAK	1010B	Receiving device cannot accept data or transmitting device cannot send data
	STALL	1110B	Endpoint is halted or a control pipe request is not supported
	NYET	0110B	No response yet from receiver (see Sections 8.5.1 and 11.17-11.21)
Special	PRE	1100B	(Token) Host-issued preamble. Enables downstream bus traffic to low-speed devices.
	ERR	1100B	(Handshake) Split Transaction Error Handshake (reuses PRE value)
	SPLIT	1000B	(Token) High-speed Split Transaction Token (see Section 8.4.2)
	PING	0100B	(Token) High-speed flow control probe for a bulk/control endpoint (see Section 8.5.1)
	Reserved	0000B	Reserved PID

Logiciel — transactions

- Les transferts de données sont faits en mode half-duplex (USB 3.0: full-duplex)
- C'est l'hôte qui initie tous les transferts de données.
- La plupart des transactions nécessite l'envoi de 3 paquets:
  - 1. "Token packet": l'hôte envoie un paquet décrivant le type, et la direction de la transaction.

  Le paquet contient:

  • L'adresse de l'appareil USB

  • Le numéro de terminaison sur cet appareil

  - 2. "Data packet": L'appareil USB correspondant s'active en fonction de l'adresse reçue.

  Un deuxième paquet est envoyé:

  • Le paquet contient les données correspondant à la transaction demandée,

  • Le paquet est envoyé selon la direction de la transaction (hôte vers appareil, ou appareil vers hôte),

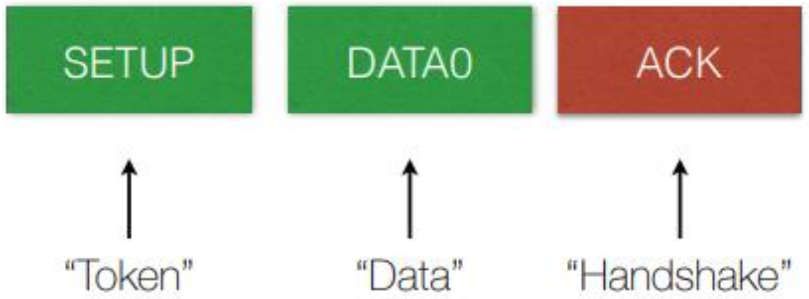
  - 3. "Handshake packet": l'appareil de destination indique si la transaction a été complétée avec succès ou non.

Logiciel — protocole de bus

- Exemples de transactions:
  - Transfert de données, du périphérique vers l'hôte

  - Transfert de données, de l'hôte vers le périphérique

  - Configuration, de l'hôte vers le périphérique



Types de transfert

- Le **USB** supporte 4 types de transfert:
  - **de contrôle:** sert à la configuration et à la commande d'un appareil. Il est effectué à partir de la terminaison 0.
  - **isochrone:** est un mode de transfert pour lequel les données sont transmise à vitesse constante, et garantie. Idéal pour les flux de données ("streaming").
  - **par interruption:** est utilisé par les appareils ayant peu de données à transmettre, mais ayant des données qui doivent être transmises rapidement (exemple: clavier ou souris). Ce ne sont pas de "vraies" interruptions: elles sont détectées par interrogation successives ("polling") de provenant l'hôte. La fréquence de ces interrogations est donnée par les descripteurs de l'appareil.
  - **par bloc:** permet de transférer des volumes importants de données lorsqu'il n'y a pas de contraintes temporelles (exemple: imprimante).

Partie 1.

Interfaçage système et bus externes

1. Le bus USB
2. Le bus CAN

Caractéristiques générales

- **CAN** : Control Area Network, BOSCH 1983
- Quelle est le domaine d'application?
  - Principalement l'automobile
- Débit (Dépend de la taille du réseau)
  - jusqu'à 1 Mbps (réseaux < 40 m)
  - 125 kbps (réseau < 500 m).
- Nombre de fils ?

Caractéristiques générales

- Asynchrone
- Bus Multi-Maîtres
- Bidirectionnel
- Half duplex
- Différentielle : moins sensible aux perturbations
- Byte oriented transmission

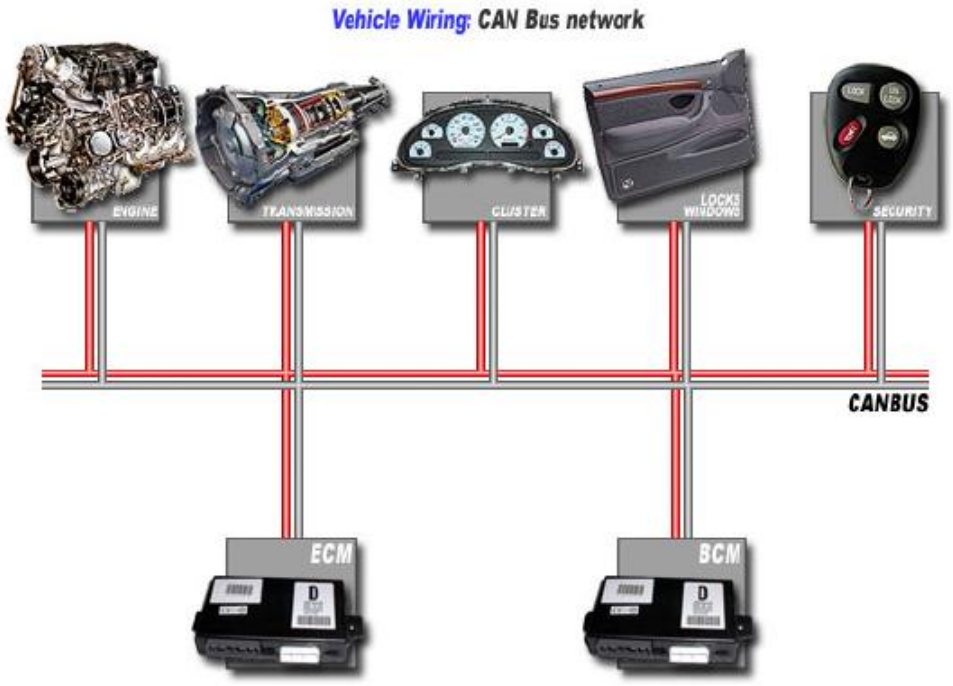
Origines

- Constat dans l'automobile :
  - 2000 m de câble 1800 connexions en 1995 >> La fiabilité et la sécurité sont menacées.
  - Les normes en matière de pollution et de consommation d'énergie multiplient les capteurs et actionneurs intelligents.
  - Le besoin de sécurité accrue (ABS, ESP, AIR-BAG…) et la demande de confort (mémorisation des réglages de conduite, climatisation régulée par passager, système de navigation…) ne font que renforcer cette tendance.
- Extension du bus CAN
  - Les composants CAN se démocratisent et investissent d'autres secteurs de l'industrie (engins travaux publique, agricole, médical, produits numériques, systèmes électrotechnique…)

Pourquoi un bus CAN?

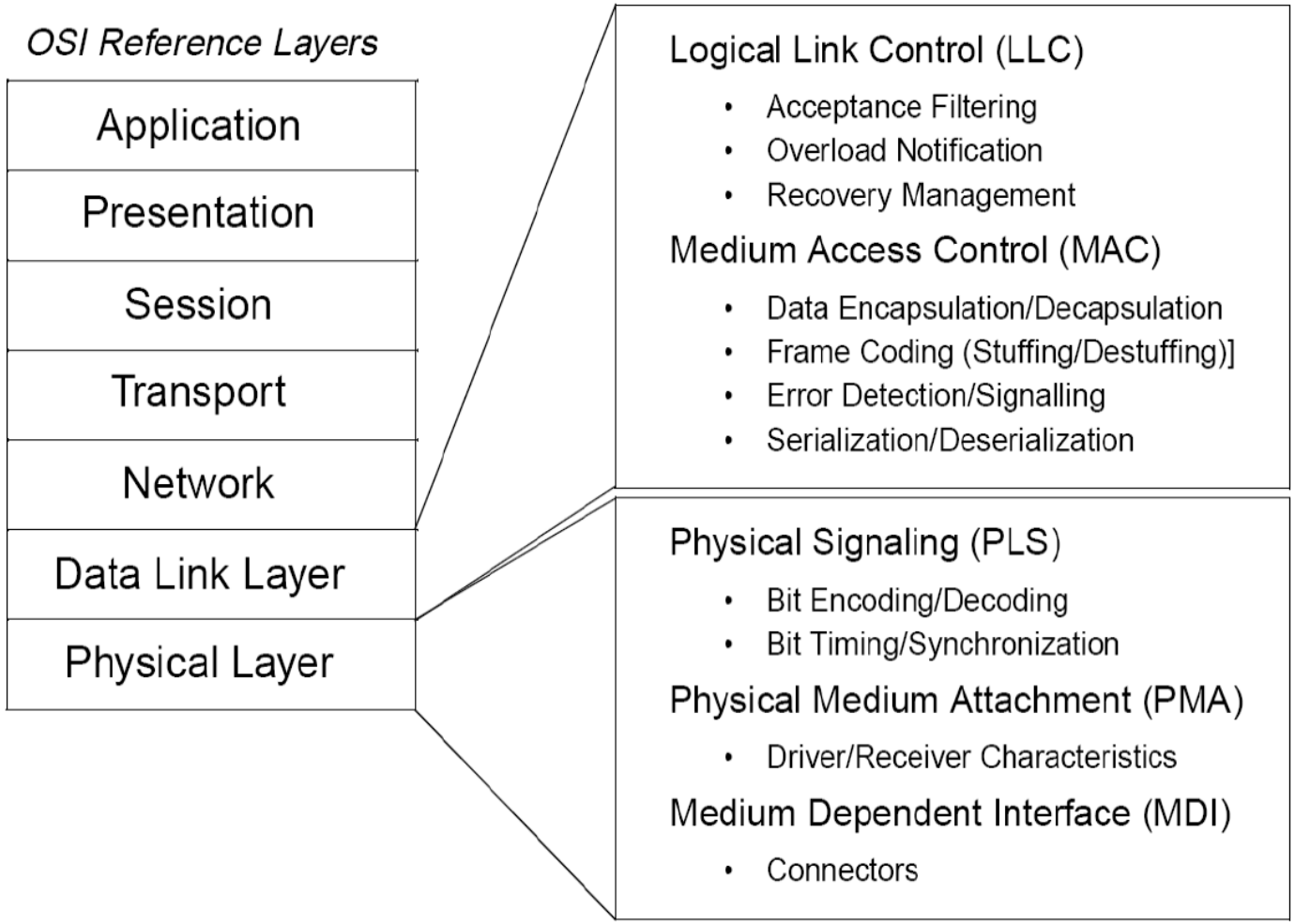
- Tous les nœuds reçoivent le message CAN. Chaque nœud récepteur décide de l'intérêt du message émit. Il est possible d'interroger un nœud particulier.

Câblage conventionnel



Câblage CAN

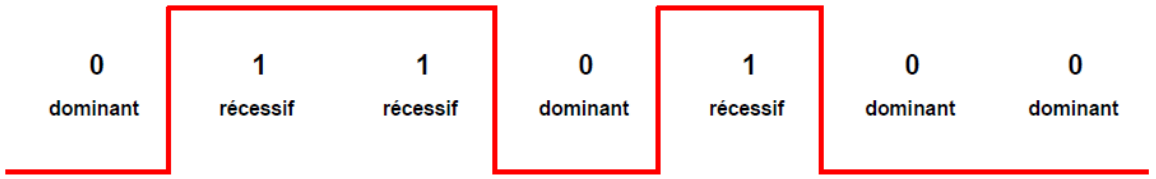
Les couches CAN



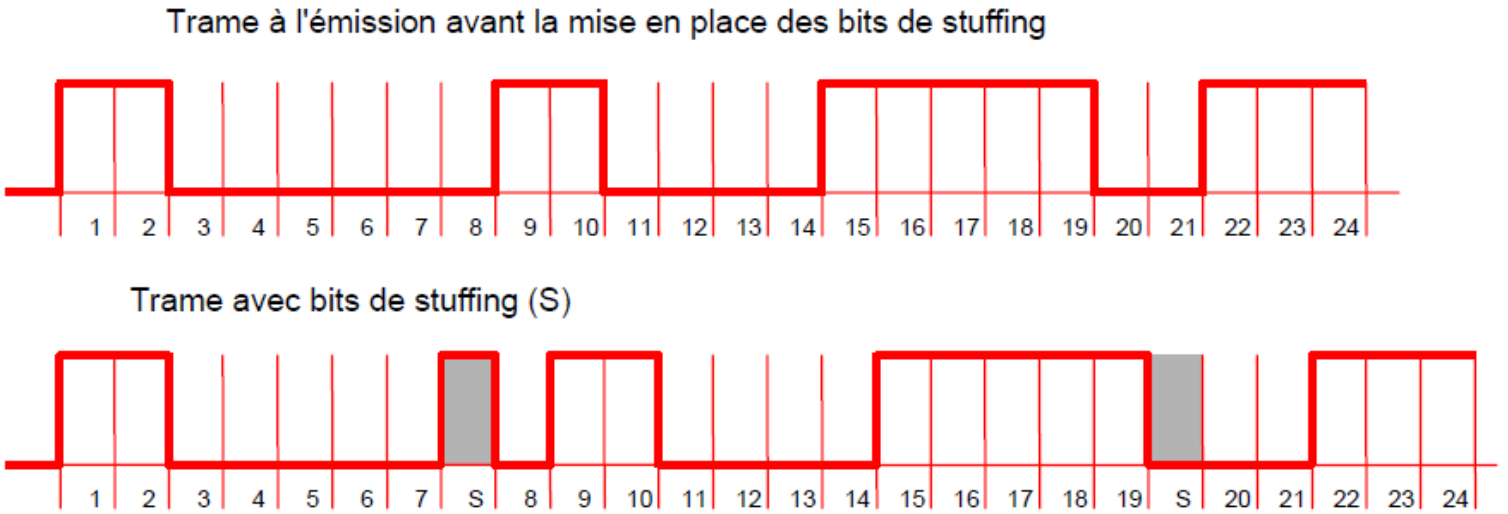
- Les sous couches LLC, MAC et PLS sont traitées par les circuits contrôleur de bus CAN (microcontrôleur, circuits spécialisés)

Couche physique

- Codage NRZ (Non Return To Zero). Le niveau de tension de la ligne est maintenu pendant toute la durée durant laquelle un bit est généré.

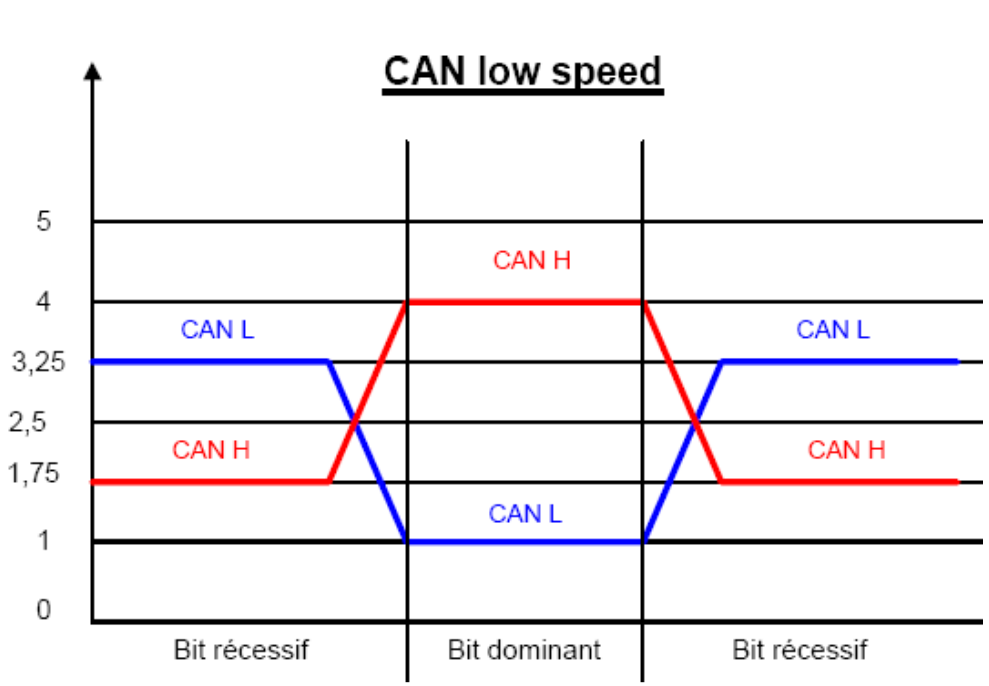


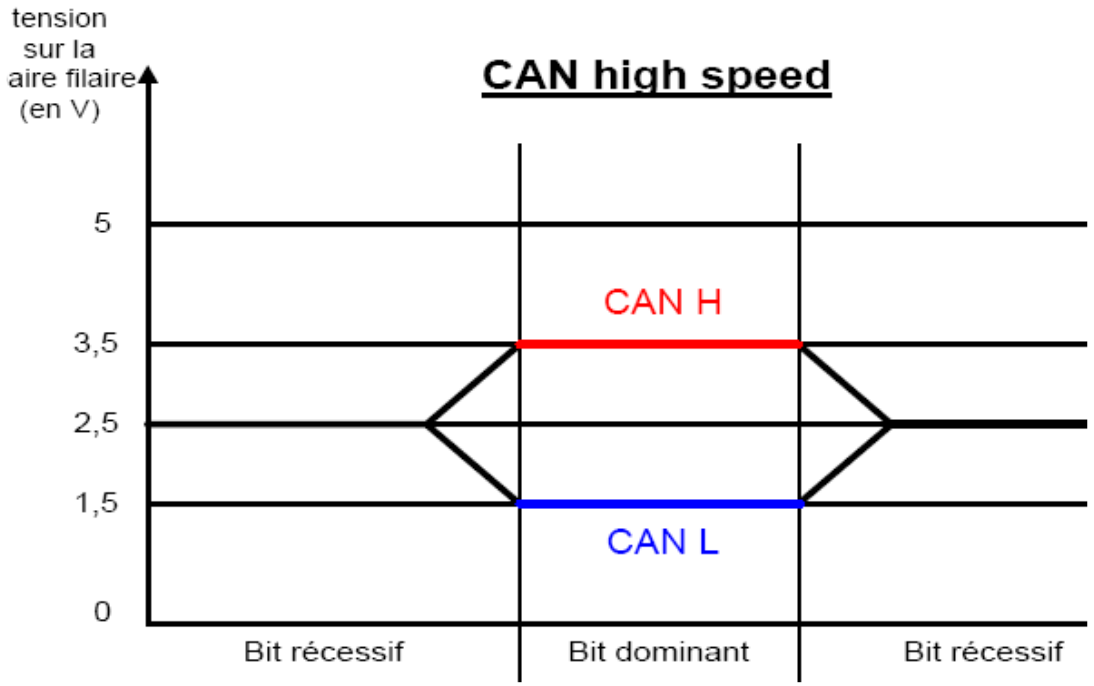
- La technique du Bit Stuffing impose au transmetteur d'ajouter automatiquement un bit de valeur opposée lorsqu'il détecte 5 bits consécutifs dans les valeurs à transmettre. Pourquoi ?



Couche physique

- La transmission des données s'effectue sur une paire par émission différentielle entre les deux lignes (CAN H et CAN L). La ligne du bus doit se terminer par des résistances de 120 Ohms.

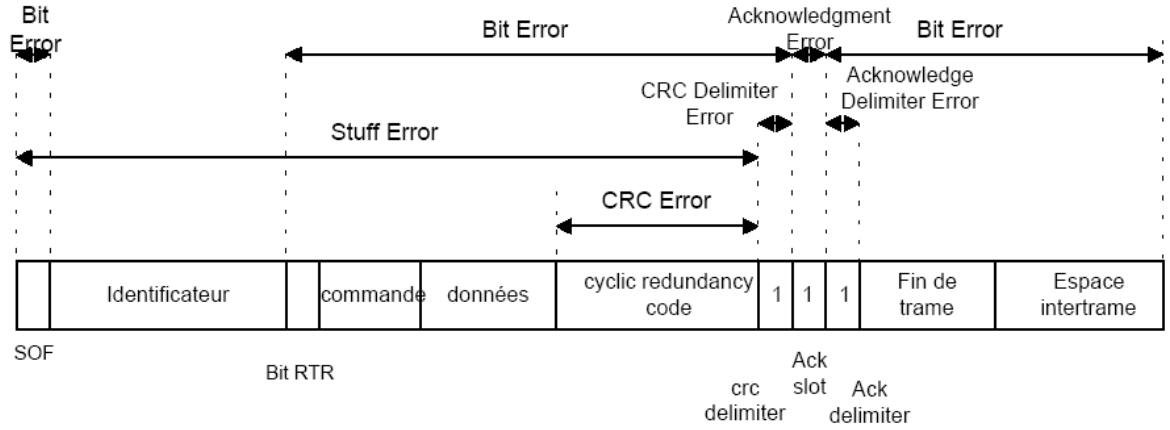




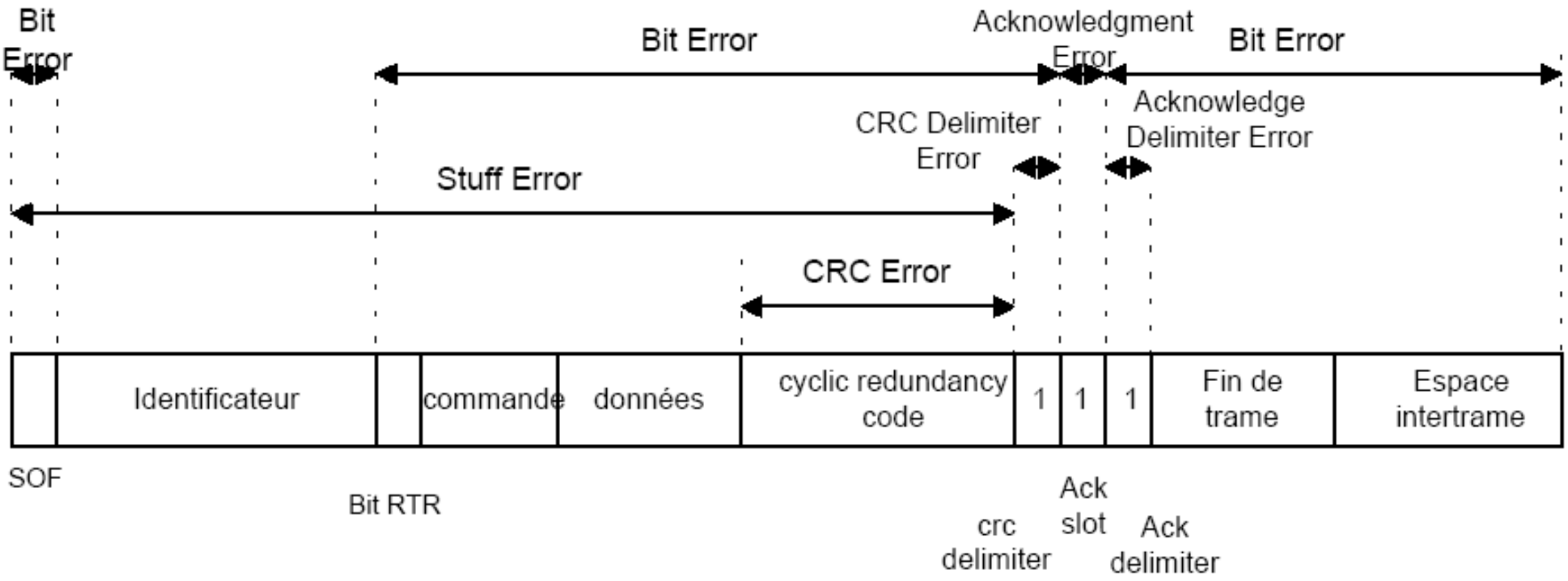
Format de la trame de donnée

- Il existe quatre types de trames spécifiques et d'un intervalle de temps les séparant :
  - Les trames de données
  - Les trames de requête (demande d'information à un nœud)
  - Trames d'erreurs (émises par n'importe quel nœud dès la détection d'une erreur)
  - Des trames de surcharge (ces trames correspondent à une demande d'un laps de temps entre les trames de données et de requête précédentes et successives).

Note : Il existe un espace intertrame de 3 bits récessifs entre les trames de données et de requête.



Format de la trame de donnée



Caractéristiques générales

Field name	Length (bits)	Purpose
Start-of-frame	1	Denotes the start of frame transmission
Identifier A	11	First part of the (unique) identifier for the data which also represents the message priority
Substitute remote request (SRR)	1	Must be recessive (1). Optional
Identifier extension bit (IDE)	1	Must be recessive (1). Optional
Identifier B	18	Second part of the (unique) identifier for the data which also represents the message priority
Remote transmission request (RTR)	1	Must be dominant (0)
Reserved bits (r0, r1)	2	Reserved bits (it must be set dominant (0), but accepted as either dominant or recessive)
Data length code (DLC)	4	Number of bytes of data (0–8 bytes)[a]
Data field	0–64 (0-8 bytes)	Data to be transmitted (length dictated by DLC field)
CRC	15	Cyclic redundancy check
CRC delimiter	1	Must be recessive (1)
ACK slot	1	Transmitter sends recessive (1) and any receiver can assert a dominant (0)
ACK delimiter	1	Must be recessive (1)
End-of-frame (EOF)	7	Must be recessive (1)

Terminologie

- **Nœud :** Sous-ensemble relié à un réseau de communication et capable de communiquer sur le réseau.
- **Valeurs du bus :** Les deux valeurs logiques définies ne sont pas le 0 et le 1 , mais des formes dites dominante et récessive.
- **Message :** Chaque information est véhiculée sur le bus à l'aide d'un message (trame de bits) de format défini mais de longueur variable (et limitée).
- **Routage des informations :** Des nœuds peuvent être ajoutés au réseau sans qu'il n'y ait rien à modifier. Chaque message possède un identificateur (identifier) qui n'indique pas la destination du message mais la signification des données du message. Ainsi tous les nœuds reçoivent le message, et chacun est capable de savoir grâce au système de filtrage si ce dernier lui est destiné ou non.

Terminologie

- **Trame de données :** Une trame de données (data frame) est une trame qui transporte, comme son nom l'indique, des données.
- **Demande d'une trame de données :** Un noeud peut demander à un autre noeud d'envoyer une trame de données, et pour cela il envoie lui même une trame de requête. La trame de données correspondant à la trame de requête initiale possède alors le même identificateur.
- **Débit bit :** Le débit bit peut varier entre différents systèmes, mais il doit être fixe et uniforme au sein d'un même système.
- **Priorités :** Les identificateurs de chaque message permettent de définir quel message est prioritaire sur un autre.
- **Fonctionnement multi-maître :** Lorsque le bus est libre, chaque nœud peut décider d'envoyer un message. Seul le message de plus haute priorité prend possession du bus.

Terminologie

- **Arbitrage :** Si deux noeuds ou plus tentent d'émettre un message sur un bus libre il faut régler les conflits d'accès. On effectue alors un arbitrage bit à bit (non destructif) tout au long du contenu de l'identificateur. Ce mécanisme garantit qu'il n'y aura ni perte de temps, ni perte d'informations. CSMA/CA (Carrier Sense Multiple Access - Collision Avoidance).
- **Points de connexion :** La liaison de communication série CAN est un bus sur lequel un nombre important d'unités peuvent êtreraccordées. En pratique le nombre total d'unités sera déterminé par les temps de retard (dus aux phénomènes de propagation) et/ou les valeurs des charges électriques que ces unités présentent sur le bus.
- **Acquittement :** Tous les récepteurs vérifient la validité d'un message reçu, et dans le cas d'un essage correct ils doivent acquitter en émettant un flag.

Fin...