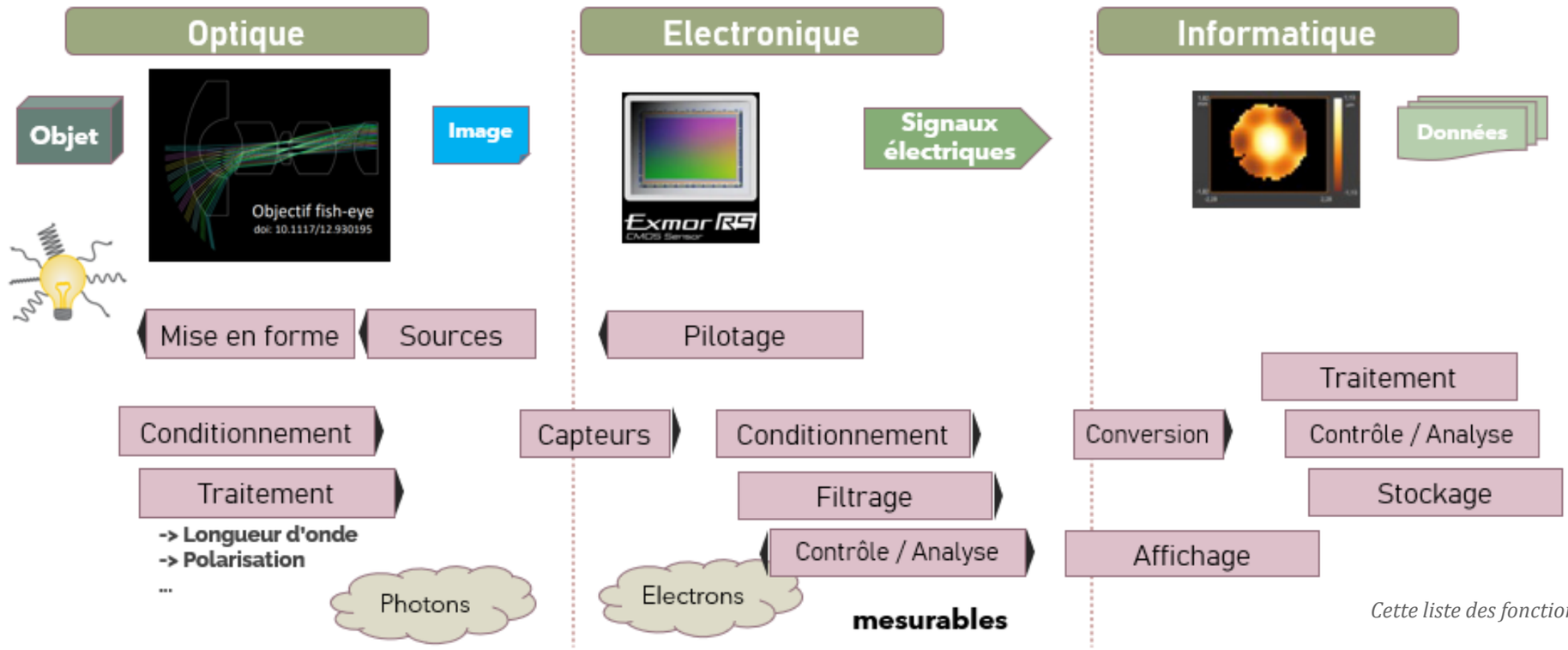
Traitement de l'Information



Cette liste des fonctionnalités n'est pas exhaustive...

http://**lense.institutoptique.fr/** outils-numeriques/ ceti/ ieti/ protis/

LEnsE

Fabienne BERNARD
Responsable pédagogique

Julien VILLEMEJANE
Co-Responsable pédagogique

Thierry AVIGNON
Co-Responsable pédagogique

Cédric LEJEUNE
Co-Responsable pédagogique

Semestre 5

ONIP-1
Outils Numériques
Sylvie LEBRUN
Julien VILLEMEJANE

CéTI
Conception Electronique
Julien VILLEMEJANE

Semestre 6

ONIP-2
Outils Numériques
Sylvie LEBRUN
Julien VILLEMEJANE

IéTI
Ingénierie Electronique
Julien VILLEMEJANE

Semestre 8

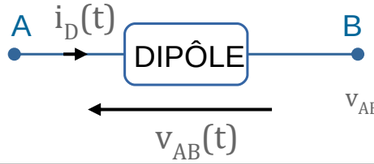
ProTIS
Procédés de **T**raitement
de l'**I**nformation et du **S**ignal
Fabienne BERNARD
Julien VILLEMEJANE

Fondamentaux / Dipôles et réseaux

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

DIPÔLES

Composant électrique à deux bornes

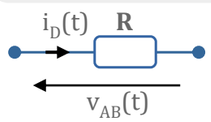


$i_D(t)$: **courant** traversant le dipôle

$v_{AB}(t)$: **différence de potentiel** aux bornes du dipôle

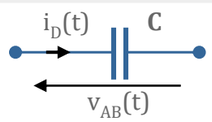
RÉCEPTEUR LINÉAIRE

Relation linéaire entre $i_D(t)$ et $v_{AB}(t)$



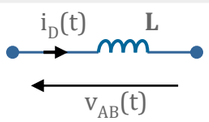
Résistance (Ω)

$v_{AB}(t) = R \cdot i_D(t)$



Condensateur (F)

$i_D(t) = C \cdot dv_{AB}(t)/dt$

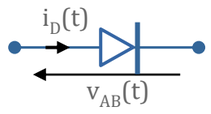


Inductance (H)

$v_{AB}(t) = L \cdot di_D(t)/dt$

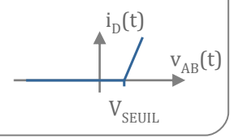
RÉCEPTEUR NON-LINÉAIRE

Relation non-linéaire entre $i_D(t)$ et $v_{AB}(t)$



Diode

$i_D(t) > 0$ si $v_{AB}(t) > V_{SEUIL}$

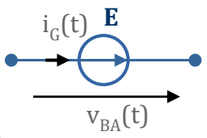


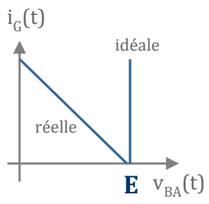
GÉNÉRATEURS

TENSION

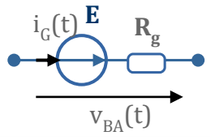
Source idéale

E = constante
quelque soit i_G





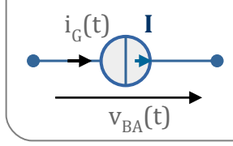
Source réelle

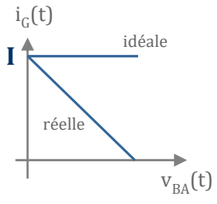


COURANT

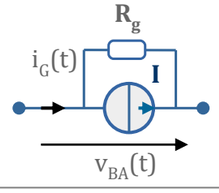
Source idéale

I = constante
quelque soit v_{BA}

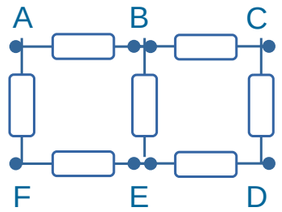




Source réelle



RÉSEAUX



Ensemble de dipôles reliés entre eux

BRANCHE

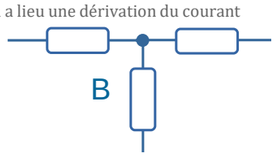
Ensemble de dipôles reliés en SÉRIE

Tous les dipôles d'une même branche sont parcourus par le même courant

NOEUD

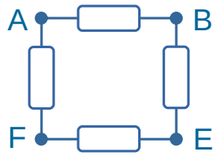
Point du réseau

Où a lieu une dérivation du courant



MAILLE

Tout chemin fermé du réseau

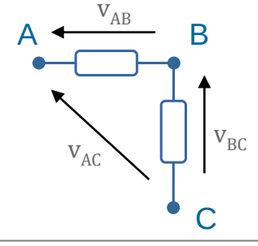


LOIS DE KIRCHHOFF

MAILLE : la tension aux bornes d'une branche d'un réseau est égale à la somme algébrique des tensions aux bornes de chacun des dipôles qui la composent

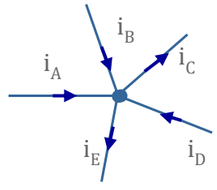
NŒUD : en un nœud, la somme des courants entrants est égale à la somme des courants sortants

LOI DES MAILLES



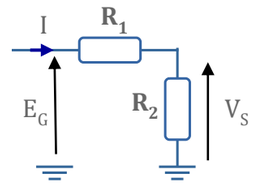
$$v_{AC} = v_{AB} + v_{BC}$$

LOI DES NŒUDS



$$i_A + i_B + i_D = i_C + i_E$$

DIVISEUR DE TENSION

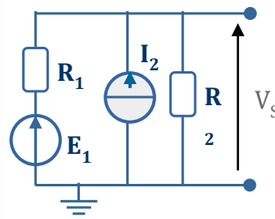


$V_S = R_2 \cdot I$ et $E_G = (R_1 + R_2) \cdot I$

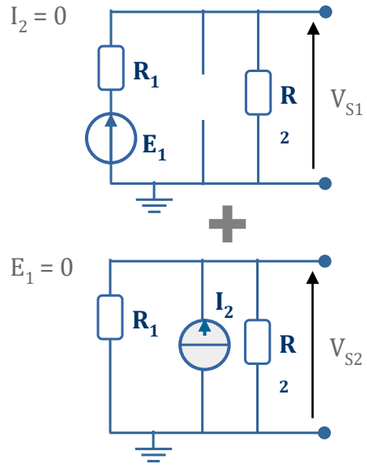
$$V_S = E_G \cdot \frac{R_2}{R_1 + R_2}$$

SUPERPOSITION

Sources indépendantes



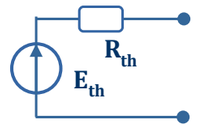
=



$$V_S = V_{S1} + V_{S2}$$

MODÈLES

THÉVENIN

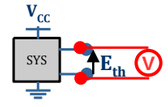


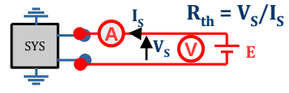
E_{th} : tension à vide du réseau

R_{th} : résistance équivalente du réseau

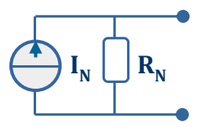
lorsqu'on éteint les générateurs indépendants

En pratique





NORTON

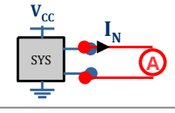


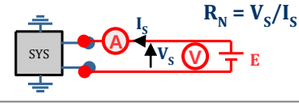
I_N : courant de court-circuit

R_N : résistance équivalente du réseau

lorsqu'on éteint les générateurs indépendants

En pratique

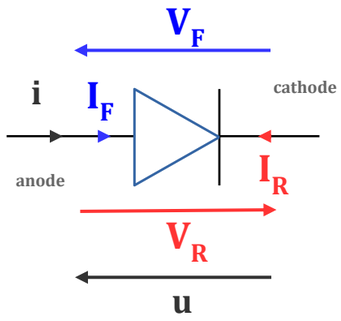




Diode / LED / Photodiode

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

DIODE



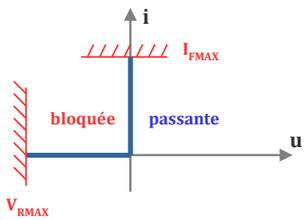
I_F : **courant direct**
souvent $I_F < I_{FMAX}$

V_F : **tension directe**
aussi appelée seuil

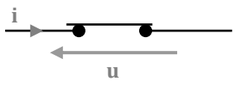
I_R : **courant inverse**

V_R : **tension inverse**
souvent $V_R < V_{RMAX}$

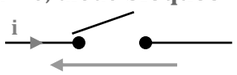
MODÈLE IDÉAL



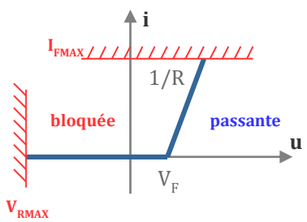
Si u > 0, diode **passante**



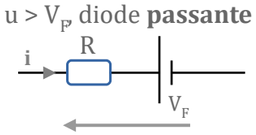
Si u < 0, diode **bloquée**



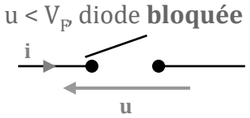
MODÈLE SIMPLE



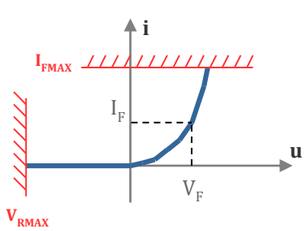
Si u > V_F, diode **passante**



Si u < V_F, diode **bloquée**



MODÈLE COMPLET



Si u > 0, diode **passante**

$$i = I_0 \left[\exp(u / n.V_0) - 1 \right]$$

loi exponentielle

V_0 : tension thermique

$V_0 = k.T / e$ — T : température (K), k : Constante de Boltzmann, e : charge d'un électron

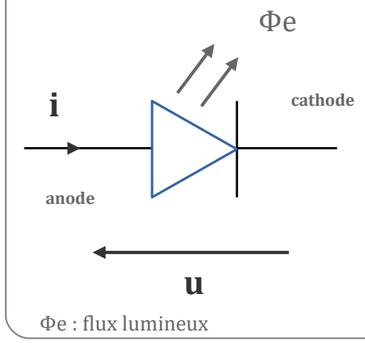
n : facteur de qualité

I_0 : constante spécifique à un type

$e = -1{,}602 \times 10^{-19}$ C

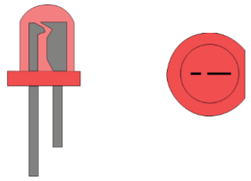
$k = 1{,}38064852 \cdot 10^{-23}$ J/K

LED

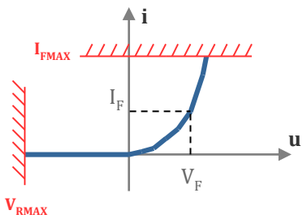


Φe : flux lumineux

LED : *Light-Emitting Diode*
DEL : Diode électroluminescente



CARACTÉRISTIQUES ÉLECTRIQUES



Si u > V_F diode **passante**

émission de photons

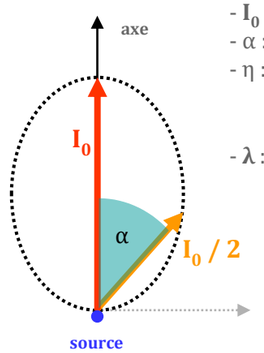
$$\Phi e = k \cdot i$$

V_F dépendant de la longueur d'onde

PARAMÈTRES IMPORTANTS :

- V_F ; I_{FMAX} ; V_{RMAX}
- P_T : puissance totale dissipable
- Bande-passante / temps de réponse
- Capacité (souvent parasite)

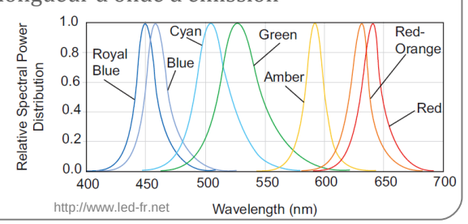
CARACTÉRISTIQUES OPTIQUES



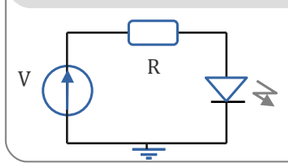
- I_0 : intensité lumineuse sur l'axe
- α : demi-angle (directivité)
- η : rendement de conversion

$$\eta = \frac{\text{Nb photons émis}}{\text{Nb électrons}}$$

- λ : longueur d'onde d'émission



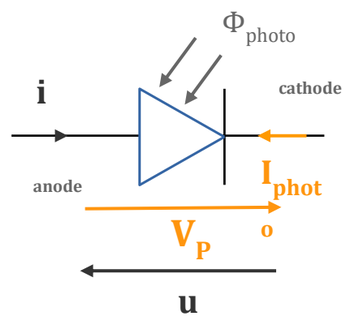
EN PRATIQUE



R : résistance de protection en courant

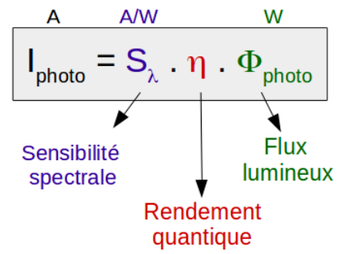
$$R_{MIN} = \frac{V_{MAX} - V_F}{I_{FMAX}}$$

PHOTODIODE

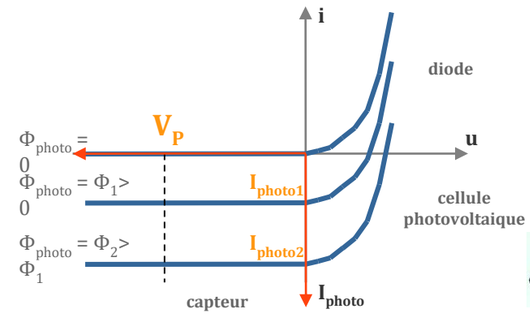


V_P : tension de polarisation

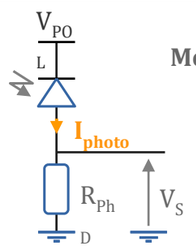
I_{PhD} : courant proportionnel au flux lumineux



CARACTÉRISTIQUES ÉLECTRIQUES



EN PRATIQUE



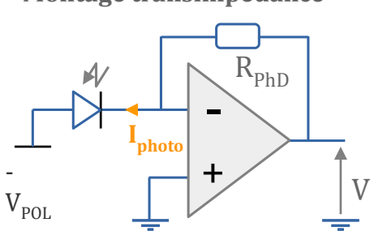
Montage simple

$$V_S = R_{PhD} \cdot I_{photo}$$

- Bande-passante limitée
  Capacité intrinsèque de la photodiode
- Sensible à l'impédance d'entrée
  du montage aval

Montage transimpédance

$$V_S = R_{PhD} \cdot I_{photo}$$



+ Bande-passante améliorée
+ Moins sensible à la capacité intrinsèque de la photodiode
- Apparition d'une résonance
  Gain-peaking / ALI

Capteur

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

GRANDEURS PHYSIQUES

MESURANDE

Grandeurs analogues à la grandeur physique à observer

- Température
- Force
- Position
- Luminosité
- Pression
- Débit
- ...

m(t) → **CAPTEUR** → s(t)

Transforme une grandeur physique observée (mesurande) **vers une autre grandeur physique utilisable** (électrique)

GRANDEURS ELECTRIQUES

SORTIE

Grandeurs mesurables analogiques ou numériques
(souvent électriques)

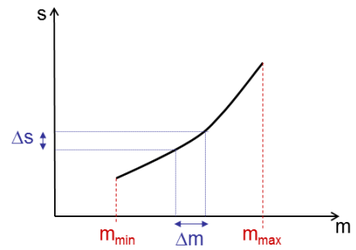
- Courant
- Tension
- Fréquence
- ...

PERFORMANCES

FONCTION DE TRANSFERT

Relation entre s(t) et m(t)

Cette relation peut être
- non-linéaire
- non continu
- par morceaux



SENSIBILITÉ

Pente de la tangente à la caractéristique entrée/sortie en un point donné

$$S(P) = \Delta S / \Delta m_P$$

Capteur	Etendue de mesure E.M.	Sensibilité S
Thermistance - semiconducteur - Platine (Pt)	 0 -> 100° C -100° C -> 1000° C	 3% / ° C 0,3% / ° C
Piezo - Quartz - PZT (Titano-Zirconate de Plomb)	0 -> 100 kN	 2,3 pC / N 110 pC / N
Photodiode	≈ 100 mW	1 A / W
µaccéléromètre ADXL202	2 g (g = 9,81 m s^{-2})	312 mV / g

ÉTENDUE DE MESURE

Plage dans laquelle le capteur répond aux spécifications

$$E.M. = m_{max} - m_{min}$$

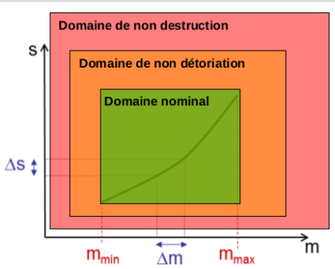
En dehors de cette plage de mesure, le constructeur ne garantit pas les performances de son système

DOMAINE D'UTILISATION

Domaine nominal
équivaut à l'étendue de mesure

Domaine de non détérioration
le capteur retrouve ses paramètres nominaux dans le domaine nominal

Domaine de non destruction
le capteur ne retrouve pas ses paramètres nominaux dans le domaine nominal mais il n'est pas détruit



En dehors de ces domaines spécifiés par le constructeur, il peut y avoir destruction du capteur

Ex : Capteur de force à jauges piézorésistives N556-1

Domaine	Mesurande	Température
Nominal	0-10 N	0°C à 60°C
Non-Détérioration	150 %	-20°C à 100°C
Non-Destruction	300 %	-50°C à 120°C

RÉSOLUTION

Plus petite variation de grandeur mesurable

LINÉARITÉ

Écart de sensibilité sur l'étendue de mesure

TEMPS DE RÉPONSE

Temps de réaction du capteur
Souvent lié à sa bande-passante

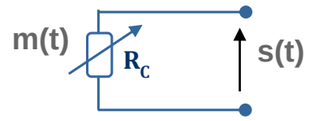
La sensibilité du capteur peut en effet dépendre de la fréquence à laquelle on souhaite l'utiliser*

** Voir aussi Régime Harmonique / Analyse Harmonique d'ordre 1 et 2*

TYPES DE CAPTEURS

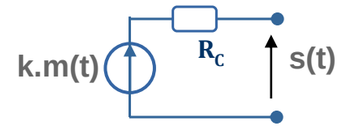
PASSIF

Impédance variable



Nécessite une alimentation externe

ACTIF



Transforme directement en grandeur électrique

ANALOGIQUE

Infinité de valeurs continues
Tension, courant…
Ex : Thermocouple

NUMERIQUE

Tout Ou Rien (TOR)
'0' ou '1' Ex : Fin de course

Intelligent / Smart
SPI/I2C Ex : Accéléro Num

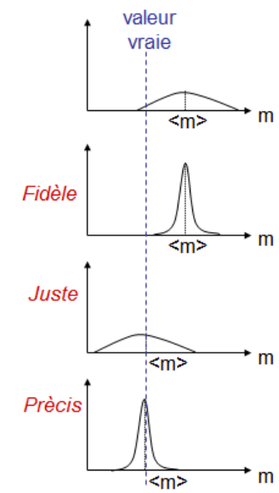
PRÉCISION

Aptitude du capteur à donner une mesure proche de la valeur vraie

Etude statistique sur n mesures

$$\langle m \rangle = \frac{\sum_i m_i}{n}$$

$$\sigma = \sqrt{\frac{\sum_i (m_i - \langle m \rangle)^2}{n-1}}$$



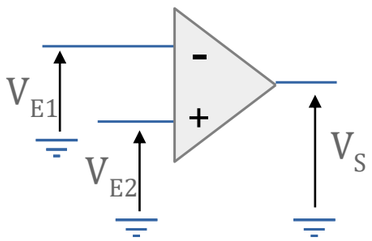
Un capteur **précis** est un capteur **fidèle** et **juste**

Amplificateur Linéaire Intégré / Principe et montages de base

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

MODE NON-LINÉAIRE

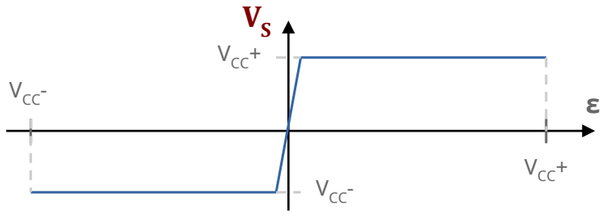
COMPARATEUR SIMPLE



$V_S = V_{CC}+$ Si $V_{E2} > V_{E1}$

$V_S = V_{CC}-$ Si $V_{E1} > V_{E2}$

Caractéristique Vs = f (ε) avec ε = V+ - V-



COLLECTEUR OUVERT / ÉMETTEUR OUVERT

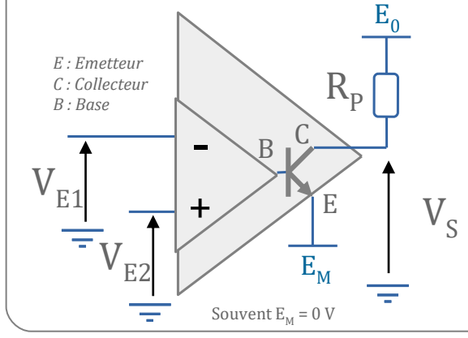
Comparateur associé à un transitor T :

I_B : courant entrant dans la base

I_C : courant entrant dans le collecteur

→ si $I_B > 0$ alors $I_C > 0$, T = interrupteur fermé

→ sinon $I_C = 0$, T = interrupteur ouvert



Souvent $E_M = 0$ V

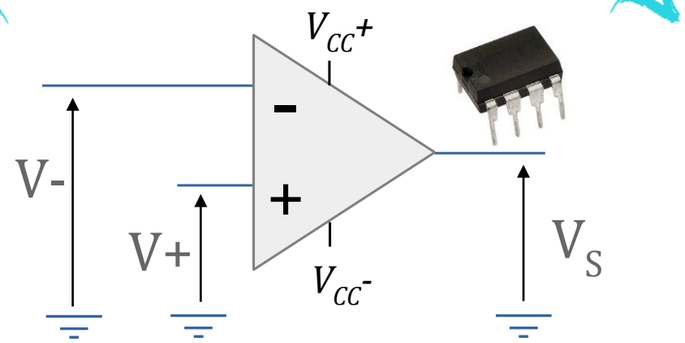
Si $V_{E2} > V_{E1}$ → $I_B > 0$ **$V_S = E_M$**

Si $V_{E1} > V_{E2}$ → $I_B = 0$ **$V_S = E_0$**

COMPOSANTS

- *LM311 : asymétrique, CO, EO*
- *LM339 : asymétrique, CO, 4 comparateurs*

NON — **CONTRE-RÉACTION NÉGATIVE ??** — OUI



FONCTION DE TRANSFERT

$$V_S = A . (V+ - V-)$$

avec $10^5 < A < 10^7$

Saturation à $Vs = V_{CC}+$

CARACTÉRISTIQUES

- **Slew Rate** (SR) en V/µs
- Produit **Gain Bande Passante** en MHz

G . BP = constante

- **Puissance** dissipable en W
- **Courant maximal** en sortie en A

ALIMENTATION

- **Symétrique** : $V_{CC}+ = +U$ et $V_{CC}- = -U$
- **Asymétrique** : $V_{CC}+ = +U$ et $V_{CC}- = 0V$
  - *avec 3 V < U < 18 V*

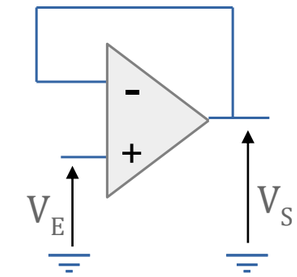
CHECK-LIST PRATIQUE

- *Vérifier les alimentations*
- *Vérifier le signal d'entrée* $V_{CC}- < V_E < V_{CC}+$
- *Vérifier que V+ = V- si mode linéaire*
- *Vérifier la tension de sortie, si Vs = $V_{CC}+$ ou $V_{CC}-$*
  - *modifier la tension d'entrée*
  - *modifier le gain du montage*

MODE LINÉAIRE

$$V- = V+$$

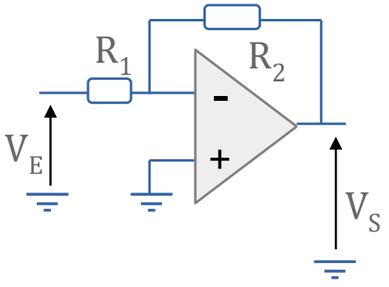
SUIVEUR



$V_S = V_E$

Adaptation d'impédance

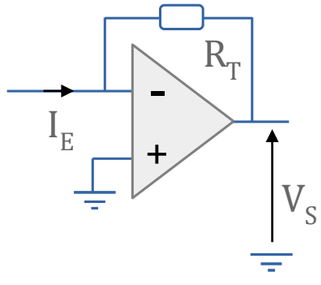
INVERSEUR



$$V_s = -\frac{R_2}{R_1} . V_E$$

Amplification

TRANSIMPEDANCE



$V_s = -R_T . I_E$

Conversion courant/tension

COMPOSANTS

- *TL071 / TL081 : symétrique, GBP = 3 MHz*
- *TL082 / TL084 = 2 x TL081 / 4 x TL081*
- *TLE2072 : symétrique, GBP = 9 MHz*
- *LM358 : asymétrique, GBP = 1 MHz*

Amplificateur Linéaire Intégré / Modélisation 1er ordre et rebouclage

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

MODÈLE DU PREMIER ORDRE

LIMITATION EN FRÉQUENCE

Les **amplificateurs linéaires intégrés**, comme beaucoup d'autres composants, ont un comportement fréquentiel non constant.
Ils se comportent comme un **filtre de type passe-bas**, que l'on peut modéliser par un **système du premier ordre**.

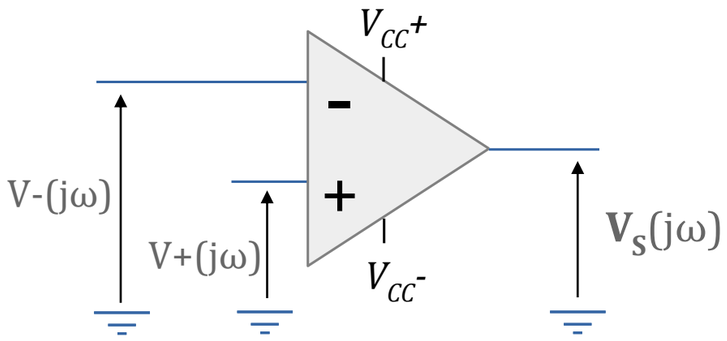
NB : la **limitation en tension de l'amplitude du signal de sortie** est toujours effective, elle dépend de la tension d'alimentation.

Le paramètre important à prendre en compte est le **gain unitaire**, aussi appelé **produit gain – bande-passante**.

Ce paramètre est donné en **Hz** et il est **constant**.

Exemple pour un produit gain – bande-passante GBW = 3 MHz
- pour une amplification de 1 du système, la bande-passante du système sera de 3 MHz (3 MHz / 1)
- pour une amplification de 1000 du système, la bande-passante du système sera de 3 kHz (3 MHz / 1000)

FONCTION DE TRANSFERT



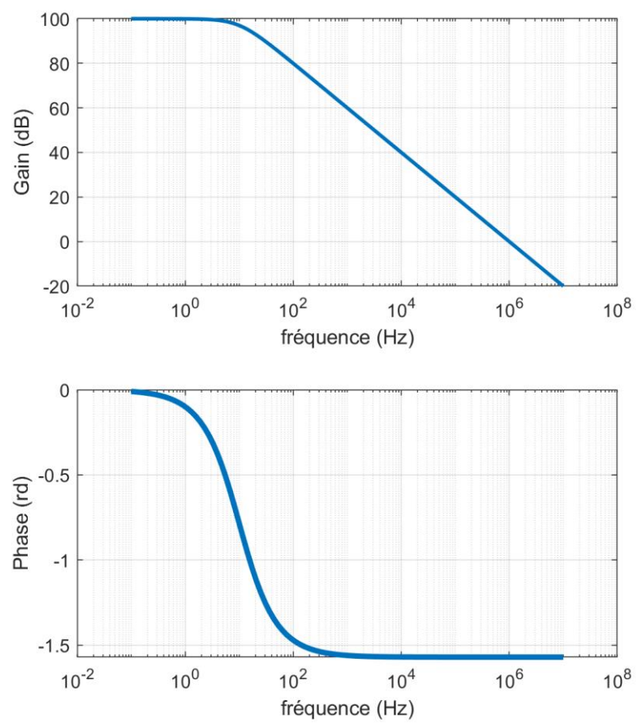
$$\mathbf{V_S(j\omega) = A(j\omega) \,.\, [V+(j\omega) \;-\; V-(j\omega)]}$$

Où $\underline{\mathbf{A}}\mathbf{(j\omega)} = \dfrac{\mathbf{Av}}{\mathbf{1 + j\,\dfrac{\omega}{\omega_C}}}$

Av : amplification différentielle

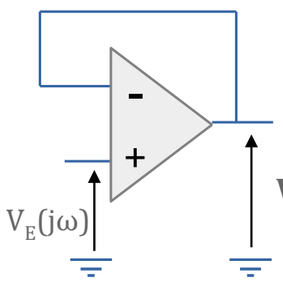
ω_C **= GBW / Av**

RÉPONSE EN FRÉQUENCE



Exemple d'un ALI ayant un produit gain – bande-passante GBW = 1 MHz et une amplification différentielle de 10^5

FONCTION DE TRANSFERT EN SUIVEUR



A partir de l'équation ci-contre, on obtient pour ce circuit (suiveur) :

$$\mathbf{V_S(j\omega) = A(j\omega) \,.\, [V_E(j\omega) \;-\; V_S(j\omega)]}$$

On obtient la fonction de transfert suivante :

$$\mathbf{T(j\omega) = \frac{V_S(j\omega)}{V_E(j\omega)} = \frac{A(j\omega)}{1 + A(j\omega)}}$$

REBOUCLAGE

INTÉRÊT DU REBOUCLAGE / SUIVEUR

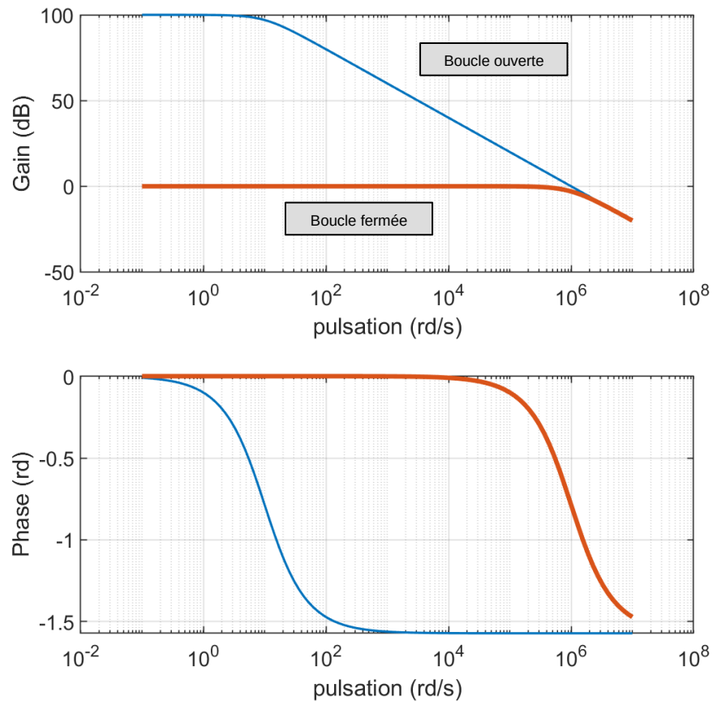
Le fait de **reboucler un système**, ou de le fermer, c'est-à-dire réinjecter une image de la valeur de sortie sur l'une de ses entrées (ici l'entrée négative), permet de **modifier son comportement fréquentiel**.

Un **ALI non rebouclé** a un **gain important** (minimum 100 dB) mais une **bande-passante très faible** (de l'ordre de la dizaine de Hz). *Ce fort gain entraîne malheureusement une saturation de la sortie assez rapidement.*

Un **ALI rebouclé** a une **meilleure bande-passante** (produit gain fois bande-passante constant) mais un **gain plus faible**.

RÉPONSE EN FRÉQUENCE

Exemple d'un ALI ayant un produit gain – bande-passante GBW = 1 MHz et une amplification différentielle de 10^5 (identique ci-contre) et le rebouclage en mode suiveur.

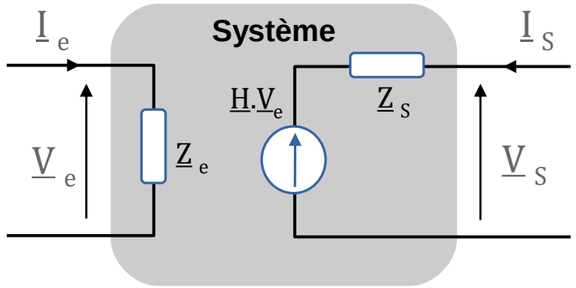


Systèmes linéaires / Superposition

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

MODÈLE

Regroupement de composants (dipôles ou autres) **régi par des équations linéaires** (pouvant être différentielles) dans sa **relation entre son entrée et sa sortie**, permettant le **transfert d'énergie** entre deux dipôles (ou systèmes)



$\underline{V}_e$ $\underline{I}_e$: tension / courant d'entrée

$\underline{V}_S$ $\underline{I}_S$: tension / courant de sortie

H : fonction de transfert

$\underline{Z}_e$: impédance d'entrée

$\underline{Z}_S$: impédance de sortie

CARACTÉRISTIQUES

GAIN EN TENSION

$$\underline{H} = \underline{V}_S / \underline{V}_E \quad \text{lorsque } I_S = 0$$

c'est à dire, lorsque la charge n'est pas connectée au système

Lorsque ce gain dépend de la fréquence* du signal d'entrée ($\omega = 2.\pi.f$), on parle alors de **fonction de transfert** : $\underline{T}(j\omega) = \underline{V}_S / \underline{V}_e$

Les impédances d'entrée et de sortie peuvent également dépendre de la fréquence du signal d'entrée appliqué

Voir également la fiche sur le **régime harmonique*

IMPÉDANCE D'ENTRÉE

Impédance vue par le générateur (ou le système placé en amont) lorsque le **système** à étudier est **chargé** (connecté à sa charge)

$$\underline{Z}_e = \underline{V}_e / \underline{I}_e$$

IMPÉDANCE DE SORTIE

Impédance associée au générateur parfait (gain en tension) **vue par la charge** en sortie du système lorsque $\mathbf{V_e = 0\ V}$

$$\underline{Z}_S = \underline{V}_S / \underline{I}_S$$

EN PRATIQUE

GAIN EN TENSION

CAS CONTINU :
- on déconnecte la charge Z_L
- on applique une tension $\mathbf{V_e}$ continue
- on mesure la tension $\mathbf{V_s}$
- $\mathbf{A = V_S / V_e}$

ANALYSE HARMONIQUE :
- on applique une tension sinusoïdale $\mathbf{V_e}$ d'amplitude constante
- on mesure l'amplitude de la tension $\mathbf{V_s}$ pour **diverses fréquences** de $\mathbf{V_e}$ (en vérifiant qu'elle soit toujours sinusoïdale)
- $\mathbf{A(\omega) = V_S(\omega) / V_e(\omega)}$
- On peut ensuite tracer l'évolution de A en fonction de ω (Bode)

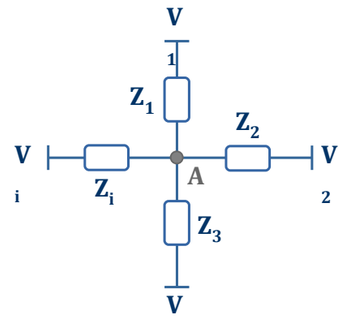
IMPÉDANCE D'ENTRÉE

CAS CONTINU :
- on connecte la charge Z_L au quadripole
- on applique une tension $\mathbf{V_e}$ continue en entrée
- on mesure le courant $\mathbf{I_e}$ entrant dans le quadripole
- $\mathbf{Z_e = V_e / I_e}$

IMPÉDANCE DE SORTIE

CAS CONTINU :
- on court-circuite l'entrée : $\mathbf{V_e = 0\ V}$
- on applique une tension $\mathbf{V_S}$ continue sur la sortie
- on mesure le courant $\mathbf{I_S}$ entrant dans le quadripôle, côté sortie
- $\mathbf{Z_S = V_S / I_S}$

SIMPLIFICATION DE MILLMAN



En un **nœud A** d'un réseau de branches en parallèle de générateurs de tension réels (source de tension et impédance)

la **tension au point A** vaut :

avec Y = 1/Z

$$V_A = \frac{Y_1.V_1 + Y_2.V_2 + Y_3.V_3 + Y_i.V_i}{Y_1 + Y_2 + Y_3 + Y_i}$$

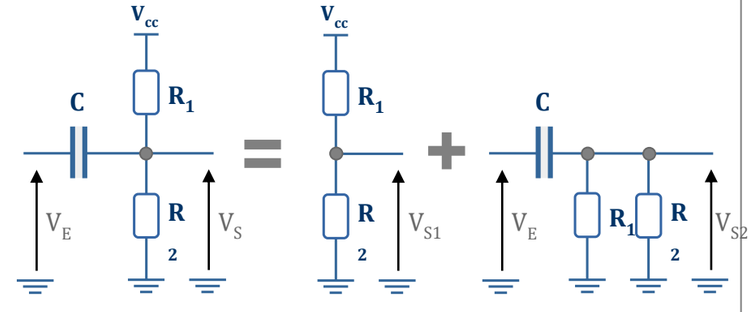
Attention !
Tous les potentiels doivent être référencés par rapport à un même potentiel, souvent noté masse.

Généralisation à N branches en parallèle

$$V_A = \frac{\sum_{k=1}^{k=N} Y_k.V}{\sum_{k=1}^{k=N} Y_k}$$

ASTUCE / VALEUR MOYENNE

Par superposition



V_E : composante fréquentielle
V_{CC} : composante continue

$V_E = 0$

POLARISATION

$V_{S1} = V_{CC} \cdot \frac{R_2}{R_1 + R_2}$

$V_{CC} = 0$

PETITS SIGNAUX

$V_{S2} = V_E \cdot \frac{j.R_E.C.\omega}{1 + j.R_E.C.\omega}$

Ce circuit permet de modifier la **valeur moyenne** d'un signal comportant des **composantes fréquentielles** supérieures à la fréquence de coupure donnée par la relation suivante

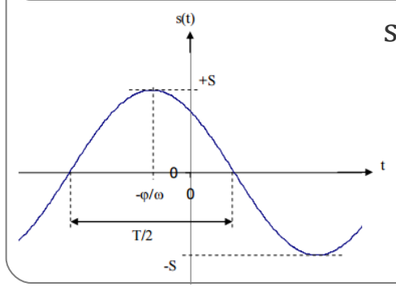
Passe-haut de fréquence

$f_c = \frac{1}{2.\pi.(R_1 // R_2).C.\omega}$

Régime Harmonique

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

REPRÉSENTATION TEMPORELLE



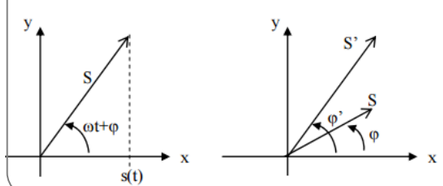
$$s(t) = S \cdot \cos(\omega t + \varphi)$$

S : amplitude du signal
ω : pulsation du signal (rd/s)
f : fréquence du signal (Hz)

$\omega = 2.\pi.f$ $\quad f = 1/T$

T : période du signal (s)
φ : déphasage du signal (rd)

REPRÉSENTATION DE FRESNEL



Représentation graphique des amplitudes et des phases

Vecteurs tournants à ω

En **régime harmonique**, **linéaire**, invariant, tous les signaux évoluent à la **même pulsation** ω

Pour des signaux plus élaborés, on décompose en **somme de signaux sinusoïdaux**, par application du théorème de superposition

REPRÉSENTATION COMPLEXE

$$s_1(t) = S \cdot \cos(\omega t + \varphi)$$

Projection sur y : $s_2(t) = S \cdot \sin(\omega t + \varphi)$

On pose : $\mathbf{s(t) = s_1(t) + j \cdot s_2(t)}$ avec : $j^2 = -1$

On a alors : $\mathbf{s(t) = S \cdot \exp(j(\omega t + \varphi))}$

$$\mathbf{s(t) = S \cdot \exp(j\varphi) \cdot \exp(j(\omega t))}$$

$$\mathbf{s(t) = \underline{S} \cdot \exp(j(\omega t))}$$

AMPLITUDE COMPLEXE
ne dépendant pas du temps

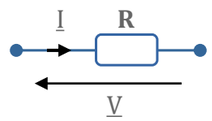
ANALYSE HARMONIQUE = COMPORTEMENT FRÉQUENTIEL

IMPÉDANCE COMPLEXE

En **régime harmonique :** v(t) et i(t) ont la même pulsation
Ainsi :

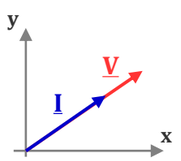
$$\frac{v(t)}{i(t)} = \frac{\underline{V}}{\underline{I}} = \underline{Z}$$

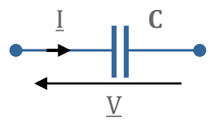
DIPÔLES LINÉAIRES



Résistance (Ω)

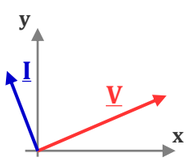
$\underline{Z} = R$

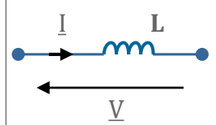




Condensateur (F)

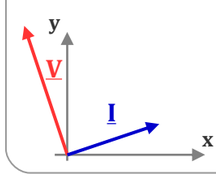
$\underline{Z} = 1 / (j C \omega)$



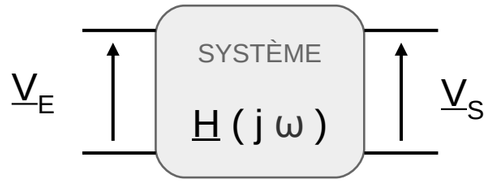


Inductance (H)

$\underline{Z} = j L \omega$



FONCTION DE TRANSFERT



Un système peut être caractérisé par sa **réponse en fréquence**, qu'on appelle aussi **fonction de transfert** H(jω)

$$\underline{V}_S(j\omega) = \underline{H}(j\omega) \cdot \underline{V}_E(j\omega) \quad \xrightarrow{TF}$$

$$\xrightarrow{TF^{-1}} \quad v_S(t) = h(t) * v_E(t)$$

convolution

Par application de la transformée de Fourier inverse, on obtient la **réponse impulsionnelle** du système notée h(t)

RÉPONSE IMPULSIONNELLE

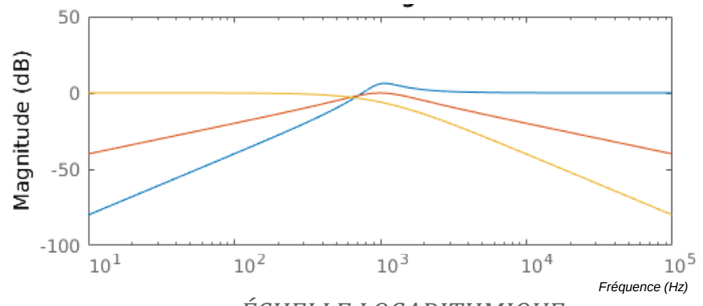
DIAGRAMME DE BODE

Un **diagramme de Bode** est une représentation graphique de l'évolution en fonction de la fréquence :
- du **gain de la fonction de transfert**, noté $\mathbf{G_{dB}(j\omega)}$

$$G_{dB}(j\omega) = 20 \cdot \log(|\underline{H}(j\omega)|)$$

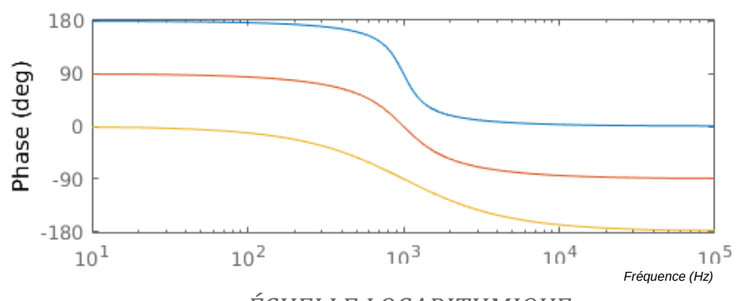
- de la **phase de la fonction de transfert**, notée $\mathbf{arg(\underline{H}(j\omega))}$

GAIN EN DECIBEL



ÉCHELLE LOGARITHMIQUE

PHASE

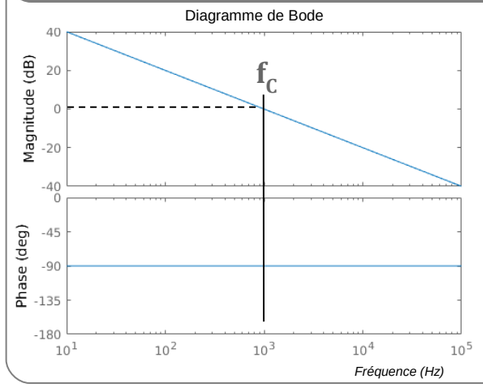


ÉCHELLE LOGARITHMIQUE

Filtrage / Analyse harmonique / Ordre 1

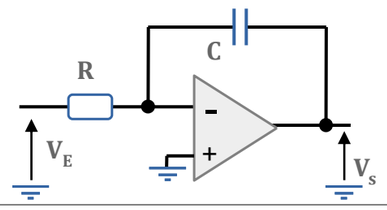
INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

INTÉGRATEUR PARFAIT



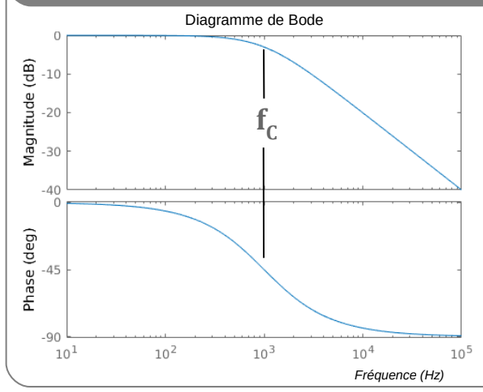
$$\underline{T}(j\omega) = \frac{1}{j\,\frac{\omega}{\omega_C}}$$

EN PRATIQUE



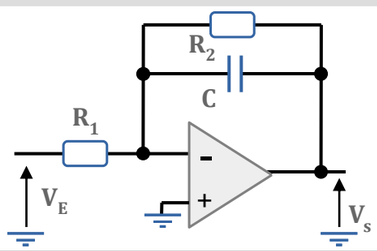
$$\omega_C = 1 / R.C$$

INTÉGRATEUR RÉEL



$$\underline{T}(j\omega) = \frac{A}{1 + j\,\frac{\omega}{\omega_C}}$$

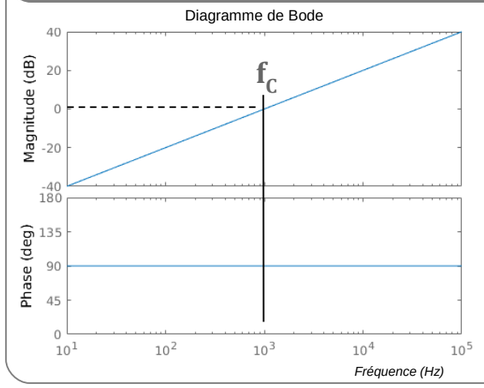
EN PRATIQUE



$$A = - R_2 / R_1$$

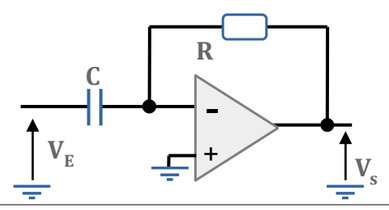
$$\omega_C = 1 / R_2.C$$

DÉRIVATEUR PARFAIT



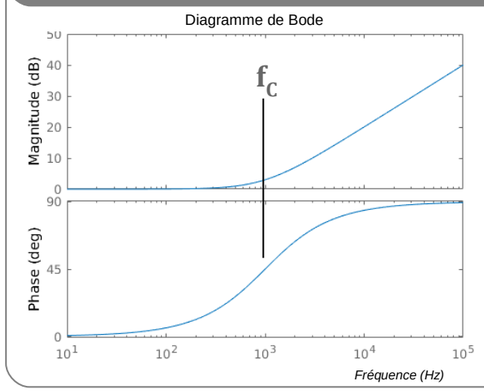
$$\underline{T}(j\omega) = \frac{j\,\omega}{\omega_C}$$

EN PRATIQUE



$$\omega_C = 1 / R.C$$

DÉRIVATEUR RÉEL



$$\underline{T}(j\omega) = \frac{1 + j\,\omega}{\omega_C}$$

CHECK-LIST PRATIQUE

- *Vérifier les **alimentations***
- *Vérifier le **signal d'entrée*** $V_{CC}^- < V_E < V_{CC}^+$
- *Vérifier que **V+ = V-** (mode linéaire)*
- *Vérifier la **tension de sortie**,*
  - *→ si $Vs = V_{CC}^+$ ou V_{CC}^-, modifier la tension d'entrée*
- *Vérifier le comportement **rapidement** par un **balayage en fréquence** du signal d'entrée (mode sweep)*

MISE EN SÉRIE / CASCADE

EXEMPLE

$$\underline{T}(j\omega) = K \cdot \frac{1 + j\,\omega/\omega_{C1}}{1 + j\,\omega/\omega_{C2}}$$

PASSAGE EN DECIBEL

$$T_{dB} = 20 \cdot \log(|\underline{T}(j\omega)|)$$

$$= 20 \cdot \log(| 1 + j\,\omega / \omega_{C1} |)$$

Modèle Dérivateur réel

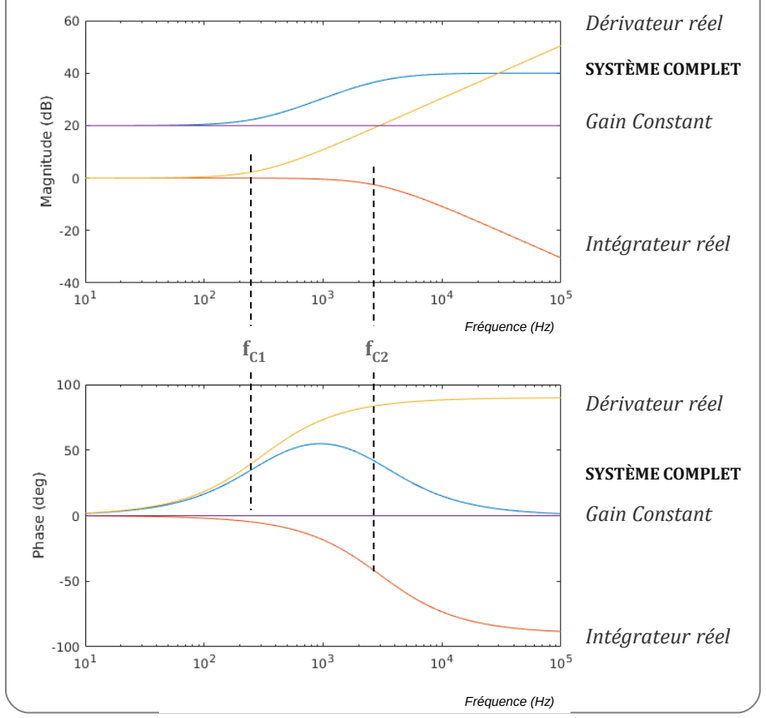
$$+ 20 \cdot \log(1 / | 1 + j\,\omega / \omega_{C2} |)$$

Modèle Intégrateur réel

$$+ 20 \cdot \log(|K|)$$

Gain Constant

DIAGRAMME DE BODE



Dérivateur réel

SYSTÈME COMPLET

Gain Constant

Intégrateur réel

Dérivateur réel

SYSTÈME COMPLET

Gain Constant

Intégrateur réel

Filtrage actif / Analyse harmonique / Ordre 2

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

FILTRE ORDRE 2 / FORMES CANONIQUES

PARAMÈTRES

A : amplification dans la bande passante
$\mathbf{f_C}$: fréquence caractéristique du filtre — $\omega = 2.\pi.f$
m : facteur d'amortissement
Q : facteur de qualité — $m = 1/2.Q$

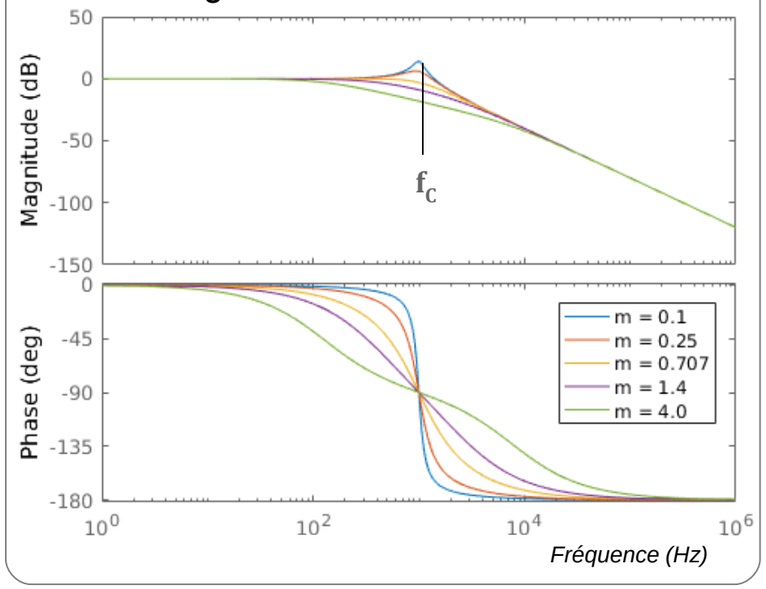
PASSE-HAUT

$$T_{HP}(j\omega) = \frac{\mathbf{A} \cdot \left(\mathbf{j}\frac{\omega}{\omega_c}\right)^2}{\mathbf{1} + \mathbf{2} \cdot \mathbf{m} \cdot \mathbf{j}\frac{\omega}{\omega_c} + \left(\mathbf{j}\frac{\omega}{\omega_c}\right)^2}$$

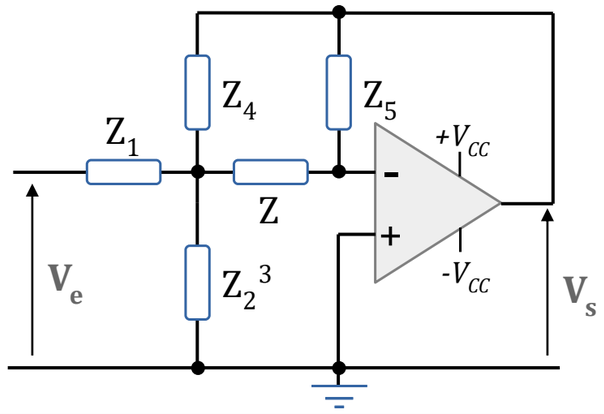
PASSE-BAS

$$T_{LP}(j\omega) = \frac{A}{\mathbf{1} + \mathbf{2} \cdot \mathbf{m} \cdot \mathbf{j}\frac{\omega}{\omega_c} + \left(\mathbf{j}\frac{\omega}{\omega_c}\right)^2}$$

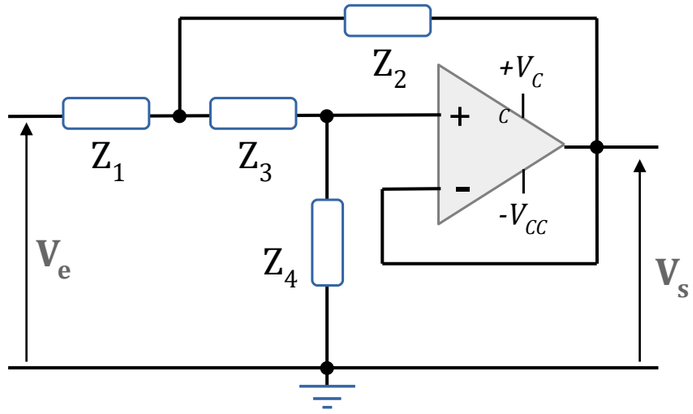
Diagramme de Bode / Passe-Bas



STRUCTURE DE RAUCH



STRUCTURE DE SALLEN-KEY



FONCTION DE TRANSFERT

$$T(j\omega) = \frac{Y_1 \cdot Y_3}{(Y_3 \cdot Y_4) + Y_5 \cdot (Y_1 + Y_2 + Y_3 + Y_4)}$$

FONCTION DE TRANSFERT

$$T(j\omega) = \frac{Y_1 \cdot Y_3}{(Y_1 + Y_2) \cdot (Y_3 + Y_4) + Y_3 \cdot (Y_4 - Y_2)}$$

TYPES / A = -1

Passe-bas : Z_1 : R / Z_2 : C_2 / Z_3 : R / Z_4 : R / Z_5 : C_5

$\omega_C = 1 / R\sqrt{C_2\, C_5}$ $\qquad$ $m = \frac{3}{2}\sqrt{\frac{C_5}{C_2}}$

Passe-haut : Z_1 : C / Z_2 : R_2 / Z_3 : C / Z_4 : C / Z_5 : R_5

$\omega_C = 1 / C\sqrt{R_2\, R_5}$ $\qquad$ $m = \frac{3}{2}\sqrt{\frac{R_2}{R_5}}$

TYPES / A = 1

Passe-bas : Z_1 : R_1 / Z_2 : C_2 / Z_3 : R_3 / Z_4 : C_4

$\omega_C = 1 / \sqrt{R_1\, R_3\, C_2\, C_4}$ $\qquad$ $m = \frac{C_4\,(R_1 + R_3)}{2\sqrt{R_1\, R_3\, C_2\, C_4}}$

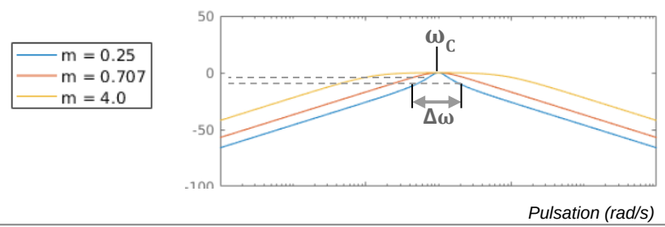
Passe-haut : Z_1 : C_1 / Z_2 : R_2 / Z_3 : C_3 / Z_4 : R_4

$\omega_C = 1 / \sqrt{R_2\, R_4\, C_1\, C_3}$ $\qquad$ $m = \frac{R_2\,(C_1 + C_3)}{2\sqrt{R_2\, R_4\, C_1\, C_3}}$

PASSE-BANDE

$$T_{BP}(j\omega) = \frac{A \cdot 2 \cdot m \cdot j\frac{\omega}{\omega_c}}{1 + 2 \cdot m \cdot j\frac{\omega}{\omega_c} + \left(j\frac{\omega}{\omega_c}\right)^2}$$

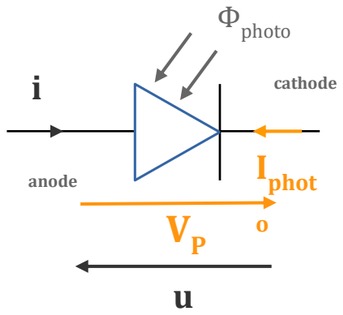
Largeur de la bande-passante (3 dB) $\qquad$ $\Delta\omega = 2\, m\, \omega_C$



Photodétection

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

PHOTODIODE = CAPTEUR



V_P : tension de polarisation

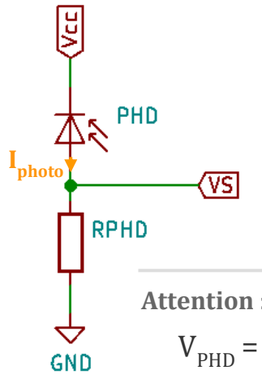
I_{PhD} : courant proportionnel au flux lumineux

A	A/W	W

$$I_{photo} = S_\lambda \cdot \eta \cdot \Phi_{photo}$$

Sensibilité spectrale — Rendement quantique — Flux lumineux

MONTAGE « SIMPLE »



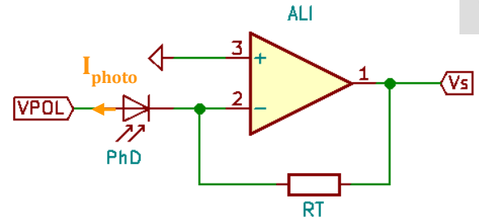
$$V_S = R_{PhD} \cdot I_{photo}$$

- Bande-passante limitée
  Capacité intrinsèque de la photodiode
- Sensible à l'impédance d'entrée
  du montage aval / système de mesure

Attention : tension de polarisation non constante !

$V_{PHD} = V_{CC} - V_S$ avec V_S dépendant du flux lumineux

MONTAGE TRANSIMPÉDANCE

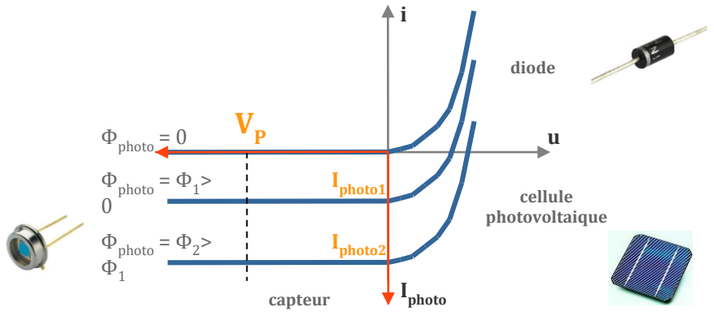


$$V_S = R_T \cdot I_{photo}$$

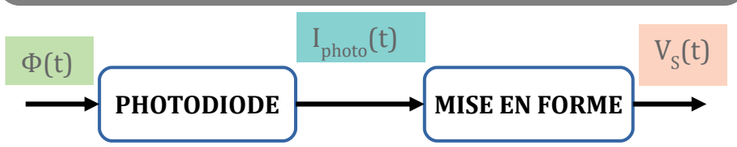
+ Bande-passante améliorée
+ Moins sensible à la capacité intrinsèque de la photodiode
+ Tension de polarisation fixe

- Montage plus « complexe »
  ALI et alimentations
- Apparition d'une résonance
  Gain-peaking / ALI

CARACTÉRISTIQUES ÉLECTRIQUES



SYSTÈME DE PHOTODÉTECTION



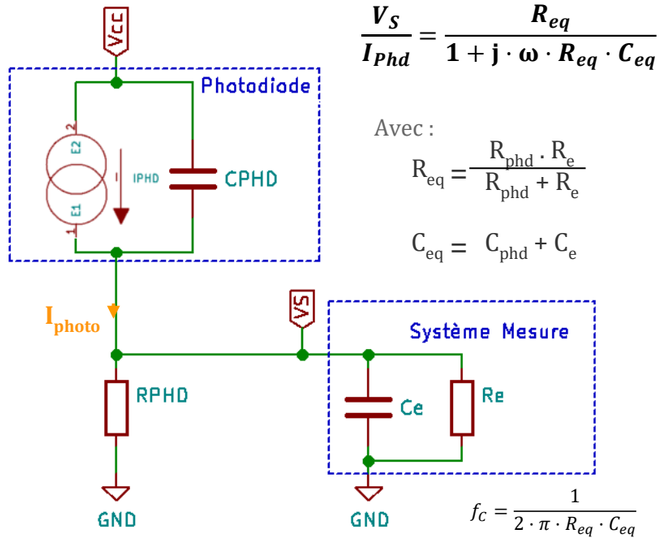
$\Phi(t) = \Phi_{cst} + \phi_{ph}(t)$ → $I(t) = I_{cst} + i_{ph}(t)$ → $V(t) = V_{cst} + v_{ph}(t)$

Φ_{cst} = flux constant
$\phi_{ph}(t)$ = flux dépendant du temps

Photodiode : capteur permettant de mesurer un flux lumineux et de le convertir en courant

Mise en forme : étage de conversion d'une grandeur électrique vers une autre grandeur électrique plus facilement mesurable (amplification, filtrage...)

MODÈLE DU SYSTÈME DE MESURE



$$\frac{V_S}{I_{Phd}} = \frac{R_{eq}}{1 + j \cdot \omega \cdot R_{eq} \cdot C_{eq}}$$

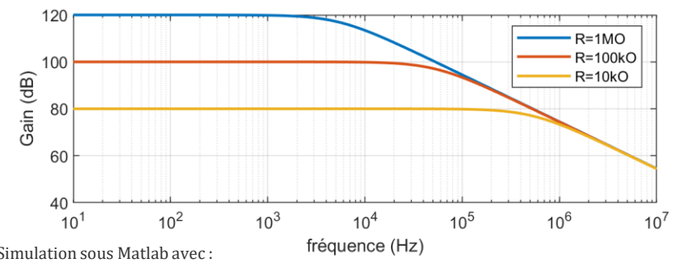
Avec :

$$R_{eq} = \frac{R_{phd} \cdot R_e}{R_{phd} + R_e}$$

$$C_{eq} = C_{phd} + C_e$$

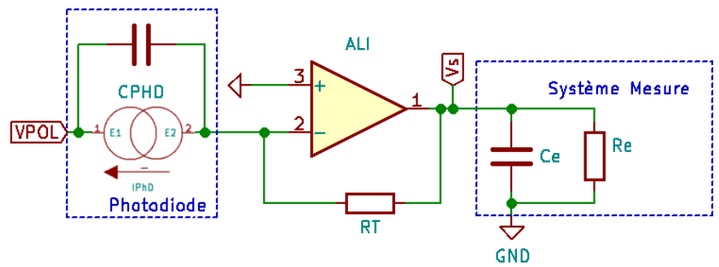
$$f_C = \frac{1}{2 \cdot \pi \cdot R_{eq} \cdot C_{eq}}$$

R_e : résistance d'entrée du système de mesure (oscilloscope, multimètre...)
C_e : capacité d'entrée du système de mesure (câble coaxial, oscilloscope...)



Simulation sous Matlab avec :
R_e = 100M / C_{phd} = 70pF / C_e = 120pF

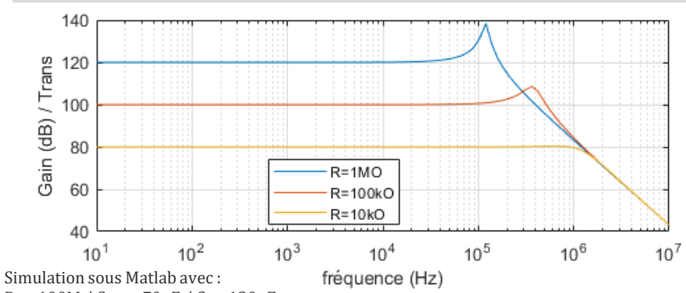
MODÈLE DU SYSTÈME DE MESURE



$$\frac{V_S}{I_{Phd}} = \frac{R_T \cdot A_0}{\left(1 + \frac{j \cdot \omega}{\omega_0}\right) \cdot \left(1 + \frac{j \cdot \omega}{\omega_c}\right) + A_0}$$

En utilisant le modèle du premier ordre pour l'amplificateur intégré (A_0, w_0)

Gain-peaking : $f_T = \sqrt{f_C . GBP}$ avec $f_C = \frac{1}{2 \cdot \pi \cdot R_{PhD} \cdot C_{PhD}}$



Simulation sous Matlab avec :
R_e = 100M / C_{phd} = 70pF / C_e = 120pF

Carte Nucléo-64 / STM32L476

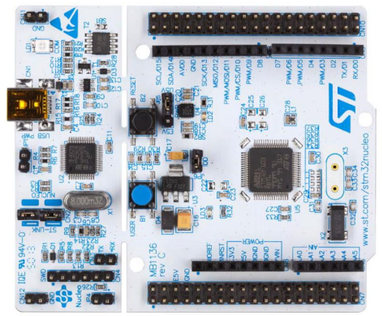
INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

MBED COMPILER / CODE SOURCE

MBED propose une **interface de développement en ligne**. L'écriture du code (une fois le projet créé) se fait en **langage C++** (proche du C dans le cas de l'embarqué).

Le code est constitué ainsi :

- Ressources / Prototypes
- Déclaration des entrées/sorties
- Variables globales
- MAIN
  - Initialisation
  - **BOUCLE INFINIE**
    - Lecture entrées
    - Traitement
    - Affichage / Pilotage
  - Appel à des fonctions
- Routines d'interruption
- Autres fonctions



life.augmented

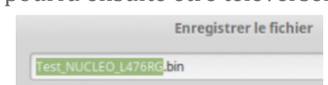
https://os.mbed.com/docs/v5.8

COMPILATION / TÉLÉVERSEMENT

Compile

Une étape de compilation permet de produire un fichier binaire qui pourra ensuite être téléverser

Enregistrer ce fichier sur votre disque dur



Une fois la carte connectée en USB à votre PC, elle est reconnue comme un espace mémoire. Il est possible alors de déplacer le fichier binaire dans cet espace mémoire. Le transfert vers la zone mémoire du microcontroleur s'effectue alors automatiquement.

A VOUS DE JOUER…

Des tutoriels sont disponibles sur
lense.institutoptique.fr/nucleo

ENTRÉES NUMÉRIQUES

Configurer la direction de la broche en entrée

```
DigitalIn mon_entree([nom_broche]);
```

Lire la valeur sur l'entrée correspondante

```
int a;
a = mon_entree ;
```

[nom_broche] = nom de la broche à configurer

```
DigitalIn mon_bp1(D10) ;
int a  = mon_bp1;
```

Ex : récupère dans la variable **a** la valeur de la broche D10

SORTIES NUMÉRIQUES

Configurer la direction de la broche en sortie

```
DigitalOut ma_sortie([nom_broche]);
```

Mettre la sortie à '0' (logique)

```
ma_sortie = 0 ;
```

Mettre la sortie à '1' (logique)

```
ma_sortie = 1 ;
```

[nom_broche] = nom de la broche à configurer

SORTIES PWM

Les sorties numériques notées **PWM** sur la carte, permettent de générer un **signal rectangulaire** de **fréquence** et de **rapport cyclique** paramétrables

Configurer la broche pour une utilisation en PWM

```
PwmOut ma_mli([nom_broche]) ;
```

Configurer la fréquence ou période (P = 1/F)

```
ma_mli.periode(double [temps_en_s]) ;
```

Configurer le rapport cyclique entre 0 et 1

```
ma_mli.write(double [rc_0_a_1]) ;
```

```
PwmOut ma_sortie(D3) ;
ma_sortie.period(0.01) ;   // F = 100 Hz
ma_sortie.write(0.4) ;     // RC = 40 %
```

ALIMENTATION

L'alimentation se fait par le port USB

(ainsi que le téléversement des programmes)

ATTENTION : les broches n'acceptent que des tensions comprises entre **0 et 3.3V / Pas de tensions négatives**

COMMUNICATION SÉRIE

Les notées **RX** et **TX** (ainsi que la liaison USB) sur la carte permettent de transmettre des données selon la norme **RS232**

Configurer la communication

```
Serial ma_liaison([broche_TX, broche_RX]);
```

Réglage de la vitesse de transfert

```
ma_liaison.baud(int [vitesse_en_bauds]);
```

NB : d'autres paramètres sont réglables comme la parité… fonction ***format****(…)*

Envoi d'un octet

```
ma_liaison.putc(char [octet_a_envoyer]);
```

Envoi d'une chaîne de caractères (utile pour le débogage)

```
ma_liaison.printf(char* [chaine_a_envoyer]);
```

```
Serial mon_periph(USBTX , USBRX);
mon_periph.baud(115200) ;
mon_periph.printf("Bonjour\r\n");
```

Ex : démarre une communication à 115200 bauds avec le PC et affiche : Bonjour (puis saute à la ligne)

ENTREES ANALOGIQUES

La carte possède des entrées analogiques (notées *Analog In*) reliées à un convertisseur analogique-numérique de 12 bits

Configurer la broche en entrée analogique

```
AnalogIn mon_en_an([nom_broche]) ;
```

Récupérer la donnée analogique (entre 0.0 et 1.0)

```
double val = mon_en_an.read() ;
```

```
AnalogIn mon_entree(A1) ;
double k = mon_entree.read() ;
mon_periph.printf("Vin=%lf V", k*3.3) ;
```

SORTIES ANALOGIQUES

La carte possède une sortie analogique (notées *Analog Out*) reliées à un convertisseur numérique-analogique de 12 bits

Configurer la broche en sortie analogique

```
AnalogOut mon_en_an([nom_broche]) ;
```

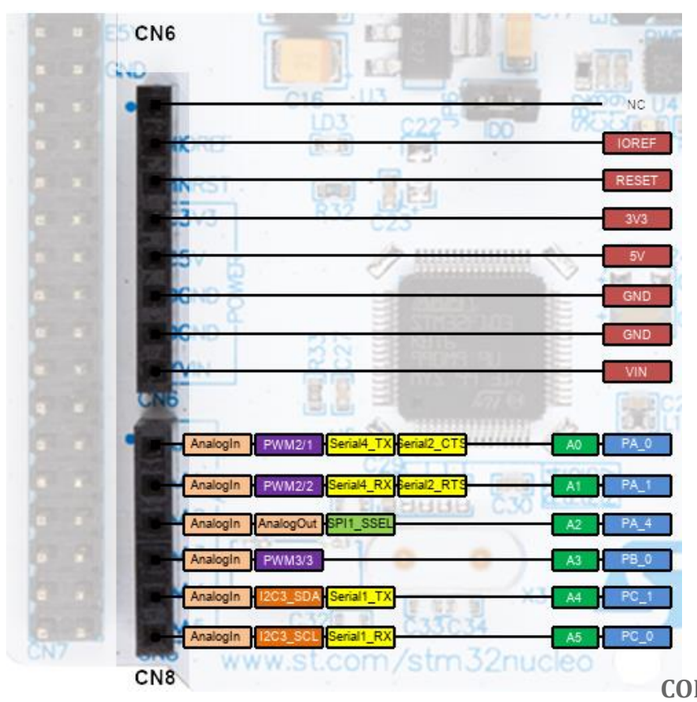
Ecrire la donnée analogique (entre 0.0 et 1.0)

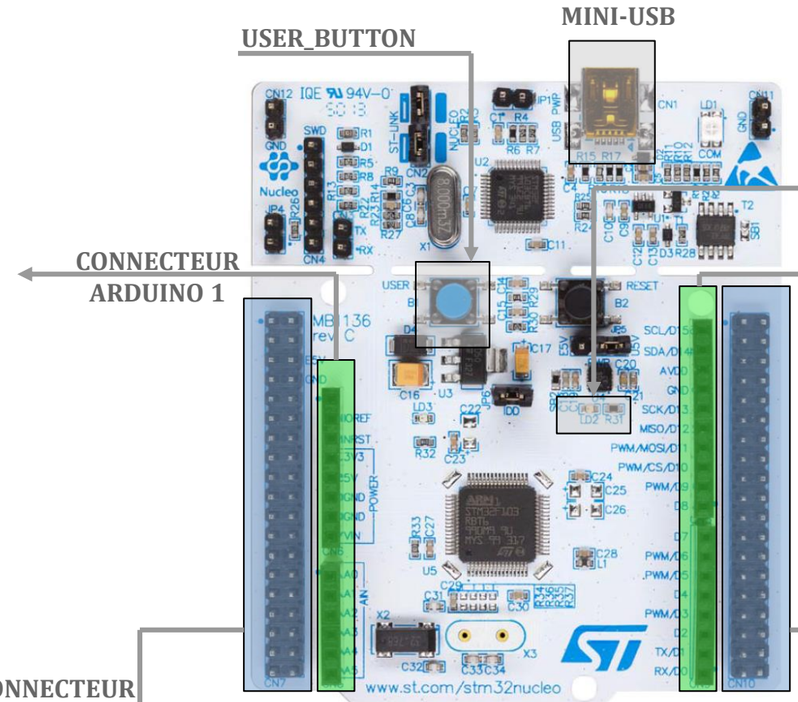
```
mon_en_an.write(double [val_0_a_1]) ;
```

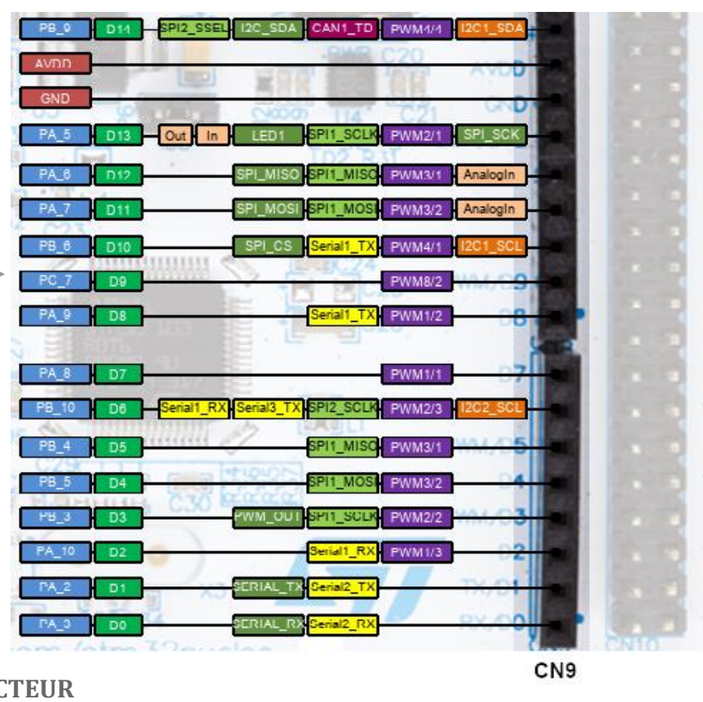
```
AnalogOut ma_sortie(A2) ;
ma_sortie.write(2.5 / 3.3) ;   // tension de 2,5V
```

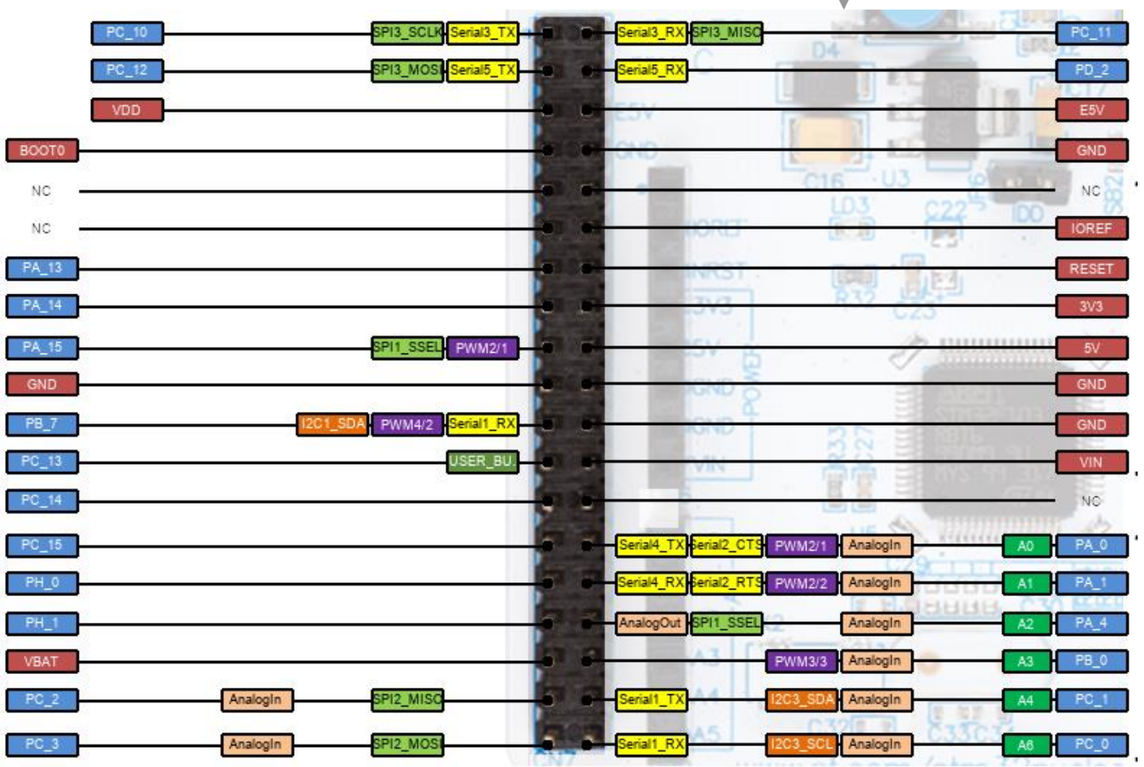
Carte Nucléo-64 / STM32L476 / Broches d'entrées-sorties

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental









Liste des composants disponibles au LEnsE

INSTITUT d'OPTIQUE GRADUATE SCHOOL ParisTech | LEnsE Laboratoire d'Enseignement Expérimental

RÉSISTANCES

- *10 Ω, 47 Ω, de 100 Ω à 1 MΩ (Série E12 – 1/4 W)*

CONDENSATEURS

- *De 1 nF à 1 µF (non polarisé – Série E6)*
- *4,7 µF, 10 µF, 47 µF, 100 µF, 220 µF, 1000 µF, 2200 µF (polarisé – filtrage alimentation)*

PROJETS 1A

- *Plateforme robotique / Foll'iogs the line*

PROJETS 2A

- *LED trichromes : Bivar R50RGB-F-0160*
- *Photodiodes trichromes : KPS-5130PD7C*
- *Montages :*
  - *Point de rosée (x3)*
  - *LED de puissance (x2)*
  - *Commande de Peltier (x8)*

AUDIO

- *Haut-Parleurs : 8Ω, 1W*
- *Ampli 25W + HP 30W - 4Ω*
- *Prises jack 3.5 / 6.5 mm / Male/Femelle*

PAILLASSE

- *Oscilloscope / GBF (x2)*
- *PC : capture oscilloscope, suite Office, accès internet*
- *Alimentation stabilisée / Multimètre*
- *Câbles avec fiche banane (différentes couleurs et tailles)*

MATÉRIEL COMMUN

- *Alimentation variable 3-12 V*
- *Câbles BNC-BNC et BNC-bananes*
- *Fils conducteurs (boite jaune)*
- *Pince / Sonde / Tournevis (boite verte)*

Des informations sur les maquettes sont disponibles sur GitHub **https://github.com/IOGS-LEnsE-embedded/Maquettes_LEnsE**

DIODES / LED

- *1N4148 : signal*
- *1N4001/2 : redressement*
- ***Zener** : 2,4V à 15V*
- ***LED** : R,B,V... **IR** : TSAL6100, **Fibre** : SFH756*
- ***PhD** : SFH206, **Fibre** : SFH250, **IR** : SFH205*

TRANSISTORS

- ***NPN** : 2N3904, 2N2222*
- ***PNP** : 2N3906*
- ***MOS N** : BS170, BS107*
- ***MOS N Power** : IRF540*

CIRCUITS INTÉGRÉS ANALOGIQUES

ALI / MODE LINÉAIRE

- *TL071 / **TL081** : symétrique, GBP = 3 MHz*
- *TL082 / TL084 = 2 x TL081 / 4 x TL081*
- *TLE2072 : symétrique, GBP = 9 MHz*
- ***LM358** : asymétrique, GBP = 1 MHz*

AUTRES

- *DG200/ 202 : interrupteur analogique commandable*
- *AD620 : amplificateur d'instrumentation*
- *AD633 : multiplieur analogique*
- *MCP1702 : Régulateur 3.3V – 100 mA*
- *L7805 : Régulateur 5V - 1A*

ALI / COMPARATEUR

- ***LM311** : asymétrique, CO, EO*
- *LM339 : asymétrique, CO, 4 comparateurs*

AMPLI AUDIO

- ***LM386** / LM380 : 1W / 2 .5 W*
- *LM833 : Double / 500 mW (casque audio)*

FILTRES ACTIFS

- *UAF42 : Filtre universel, 100 kHz*
- ***MF4** / MAX296 : Capacité commutée – Ordre 4 / 8*

CIRCUITS INTÉGRÉS NUMÉRIQUES

MICROCONTRÔLEURS

- *PIC12F1572 : 8 bits / ADC/3xPWM/USART*
- *PIC16F1503/1509 : 8 bits / ADC/4xPWM/SPI/I2C*
- *PIC16F1455/1459 : 8 bits / ADC/DAC/SPI/I2C/USB*
- *PIC18F26K22/46K22 : 8 bits / 64 MHz*
- *DsPIC30F2011 : 16 bits / ADC/DSP*

CONV. ANALOG. / NUM.

- *TLC549 : SPI / 8 bits*
- *MCP3001 : SPI / 10 bits*

CONV. NUM. / ANALOG.

- *AD7524 : Parallèle / 12 bits*
- *AD7303 : SPI / 8 bits*
- *MCP4921 : SPI / 12 bits*

LOGIQUE

- *74LS00 : 4 x NAND – 2 entrées*
- *74LS74 : 2 x Bascule D*
- *74LS90 : Compteur décimal*
- *74LS93 : Compteur 4 bits*
- *74LS191 : Compteur binaire / BCD 4 bits*
- *74LS624 : Oscillateur contrôlé en tension (VCO)*
- *4011 : 4 x NAND – 2 entrées*
- *4013 : 2 x Bascule D*
- *4018 : Compteur / Diviseur par 10*
- *4040 : Compteur 12 bits*
- *4046 : Boucle à verrouillage de phase (avec VCO)*
- *4051 / 4043 : Multiplexeur analogique (8V / 2x4V)*
- *4511 : Décodeur BCD / 7 segments*

AUTRES

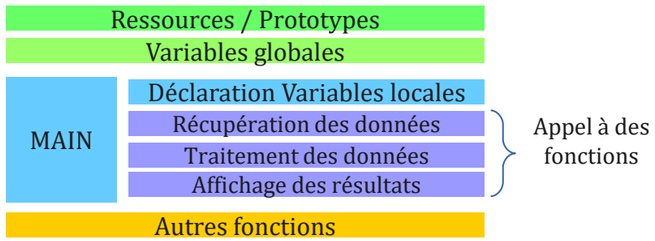
- *LM555 : Temporisation*
- *MCP6S92 : Ampli à Gain Programmable / SPI*
- *RAM 23LCV1024 : StaticRAM – 1Mbits / SPI*

Langage C

Notions de C

STRUCTURE D'UN CODE C

Le langage C est un **code très strict** dans sa façon de coder.
Le **code source** doit avoir la **structure suivante** :



TYPAGE DES VARIABLES

Une variable en C est caractérisée par :

- son **adresse** : endroit de stockage en mémoire
- son **type** : nature de l'information qu'elle contient
- son **nom** : nom donnée à la variable dans le programme
- sa **valeur** : contenu de la variable

```
int k = 5;    double y = 2.45;    char c = 'g';
```

Chaque type de variable est codé sur un **certain nombre d'informations binaires** (limitant la valeur maximale qu'elle peut contenir)

ENTIERS	**int**	32 bits – 4 octets	%d
	short	16 bits - 2 octets	%d
RÉELS	**double**	64 bits – 8 octets	%lf
CARACTÈRES	**char**	8 bits – 1 octet	%c

STRUCTURE D'UN PROJET C

Afin de mieux structurer les informations et, en particulier, gérer les différentes fonctions de l'application, on peut classer ces fonctions dans des **bibliothèques**.
Chaque bibliothèque est composée :

- d'un **fichier header** (extension .h) : rassemblant les prototypes des fonctions et les constantes propres
- d'un **fichier source** (extension .c) : rassemblant les définitions des fonctions

tableau1D.h

```
#ifndef TABLEAU1D_H_INCLUDED
#define TABLEAU1D_H_INCLUDED
#include <stdlib.h>
#include <stdio.h>

void afficheTab(int tab, int dim);
void triTabCroissant(int tab, int dim);

[...]

#endif // TABLEAU1D_H_INCLUDED
```

tableau1D.c

```
#include "tableau1D.h"
void afficheTab(int tab, int dim){
        int i;
        for(i = 0; i < dim; i++)
        ...
}
void triTabCroissant(int tab, int dim){
        ...
}
[...]
```

main.c

```
#include <stdlib.h>
#include <stdio.h>
#include "tableau1D.h"
int main(void){
        ...
afficheTab(montab, 10);
                        ...
}
```

FONCTION

Une **fonction** est un **regroupement d'instructions**.
Elle est caractérisée par :

- son **identificateur** (ou nom)
- son **type de retour** : type de donnée retournée (1 seule)

Pour pouvoir exécuter sa suite d'instructions, une fonction peut avoir besoin de **paramètres d'entrée** (ou **arguments**) typés.

```
type nom(type arg1, type arg2);
```

Ceci est le prototype de la fonction.

Les fonctions doivent ensuite **être écrites en dehors** de toute autre fonction. Elles seront ensuite **appelées dans** d'autres fonctions.

```
int somme(int a, int b){
        int c = a + b;
        return c;                }
```

AFFICHAGE CONSOLE

Pour **afficher dans la console**, on utilise la fonction **printf**.

```
printf("Exemple de texte avec saut de ligne\n");
int a = 32;  printf("a =%d\n", a);     // entier
double k = 5.3;  printf("k=%2.4lf\n", k);
// réel avec 2 chiffres significatifs avant la virgule et 4 chiffres après
```

LECTURE CLAVIER

Pour **lire des données au clavier**, on utilise la fonction **scanf**.

```
int b;          scanf("%d",&b);          // entier
double m;  scanf("%lf",&m);          // réel
```

STRUCTURES DE CONTRÔLE

```
if((x==2) && (y > 5)){
    k = 1;
}
else{
    k = 0;
}
```

Exécute le premier bloc d'instructions si la condition est vraie, sinon exécute le second bloc d'instructions

```
switch(x){
    case 1 :
        k = 5;
        break;
    case 15 :
        k = 2;
        break;
    default :
        k = 0;
}
```

Exécute le bloc d'instructions selon le cas validé

```
for(m = 5; m < 120 ; m = m + 10 ){
    printf("m =%d\n", m);
}
```

Exécute le bloc d'instructions pour m allant de 5 à 120 par pas de 10

```
while(z <= 0){
    k++;
    Z += 0.1;
}
```

Exécute le bloc d'instructions tant que la condition est vraie

CONSTANTES

Il est également possible de déclarer des **constantes**.

```
#define M 10    #define PI 3.14    #define NOM "julien"
```

TABLEAU 1D

Les tableaux sont des **regroupements indexés** de N variables d'un **même type**, l'indice allant de **0 à N-1**.
Ils sont caractérisés par leur **nom** et le **nombre de variables** qu'ils peuvent stocker. La taille du tableau est fixée statiquement.

```
int notes[5] = {10, 15, 12, 08, 13};
double suite[M];                        int i;
for(i = 0; i < M ; i++){    suite[i] = 2*i; }
```

Remplissage d'un tableau d'entier (notes) et d'un tableau de réels (suite)

```
for(i = 0; i < M ; i++)
    printf("case %d = %lf \n", i, suite[i]);
```

Affichage d'un tableau de réels (suite)

POINTEURS

Un pointeur est une variable qui **peut stocker l'adresse** d'une autre variable.

```
type *p_nom ;
```

On distingue un pointeur d'une variable standard dans sa déclaration par le symbole *
A sa déclaration, un pointeur n'adresse **aucune variable**. Il retourne la valeur **NULL**.

L'adresse d'une variable standard est donnée en ajoutant le symbole **&** devant le nom de la variable.

```
int i = 3;
int *p_i = &i;      //p_i contient l'adresse de la variable i
printf("i =%d - *p_i = %d \n", i, *p_i);
```

ALLOCATION DYNAMIQUE

L'intérêt principal du langage C est de pouvoir optimiser au mieux l'utilisation de la mémoire centrale du calculateur.
Pour cela, il existe des fonctions permettant d'**allouer la mémoire de manière dynamique**.

```
double *k;                  int m = 5 ;
k = malloc(m);
*(k+1) = 2.67;
printf("case k+1 = %lf \n", k[1]);
```

Création d'un pointeur de type double nommé k
Allocation de 5 espaces mémoires de double au pointeur k
Remplissage de l'adresse pointée par k+1 (c'est à dire la case mémoire de type double suivant celle pointée par k)
Affichage de cette valeur

Langage C

Chaîne de caractères

DÉCLARATION

Une chaîne de caractères est un **regroupement de caractères**.

```
char ma_chaine[128];
```

Création d'un tableau de caractères de 128 cases

INITIALISATION

Il existe plusieurs manières d'initialiser une chaîne de caractères

```
char ma_chaine[128] = "Vive le C" ;
```

Création d'un tableau de caractères de 128 cases dont 10 seront utilisées : 9 caractères de la chaîne et le caractère de fin de chaîne '\0'

```
char ma_chaine[] = "Vive le C" ;
```

Création d'un tableau de caractères de 10 cases : 9 caractères de la chaîne et le caractère de fin de chaîne '\0'

```
char *ma_chaine = "Vive le C" ;
```

Création d'un pointeur vers des caractères / Allocation de 10 cases mémoires : 9 caractères de la chaîne et le caractère de fin de chaîne '\0'

BIBLIOTHÈQUE <STRING.H>

Un ensemble de fonctions regroupé dans la bibliothèque **<string.h>** permet de gérer les chaînes de caractères.

Longueur d'une chaîne

```
int k = strlen(ma_chaine);  // k = 9
```

Compte le nombre de caractères jusqu'au caractère '\0'

Copie d'une chaîne

```
char *ch_source = "Vive le C";  char ch_dest[40];
strcpy(ch_dest, ch_source);
```

La chaîne source peut être une chaîne constante. Ce qui permet d'initialiser.

Comparaison de chaînes

```
int id = strcpy(ch_dest, ch_source);
```

Renvoie '0' si les deux chaînes sont identiques, une autre valeur sinon.

Concaténation de chaînes

```
strcat(ch_dest, ch_source);
```

Concatène les deux chaînes de caractères et stocke le résultat dans la première

Écriture formatée dans une chaîne

```
double k = 5.3;  sprintf(ch_dest, "k=%2.4lf\n", k);
```

Écrit le texte formaté dans la chaîne ch_dest

Langage C

Gestion de fichiers

Pour pouvoir lire ou écrire des données dans un fichier, il existe un certains nombres de fonctions contenues dans la bibliothèque :

<string.h>

TYPES DE FICHIERS

Il existe deux grands types de fichiers dans lesquels on peut stocker des données :

- des **fichiers texte ASCII**, lisibles par n'importe quel lecteur de fichier texte (NotePad par exemple)
- des **fichiers binaires** (propriétaires), non lisibles par un lecteur standard, mais nécessitant la connaissance préalable du format

POINTEUR VERS UN FICHIER

Les fichiers sont traités en C sous forme d'un pointeur de type :

```
FILE *mon_fichier;
```

FERMETURE D'UN FICHIER

Lorsqu'un fichier est utilisé par un programme, le système d'exploitation vérouille ce fichier pour tout autre programme. Il est donc essentiel de refermer un fichier une fois qu'il n'est plus utilisé. Pour cela, on utilise la fonction suivante :

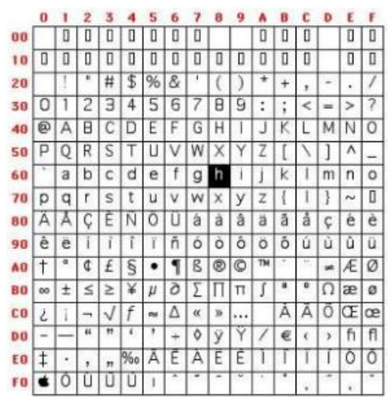
```
fclose(mon_fichier);
```

1 octet = 8 bits

2^10 = 1ki / 2^20 = 1Mi / 2^30 = 1Gi / 2^40 = 1Ti

CODE ASCII

Les variables de type **char** permettent de stocker des données sur 1 octet.
Le code **ASCII** (*American Standard Code for Information Interchange*) permet de coder l'alphabet anglais (incluant les chiffres) sur 7 bits.



```
ma_chaine[2] = 'h';
```

La case 2 de la chaine *ma_chaine* va contenir le code ASCII de la lettre h.

Il existe un **caractère spécial de fin de chaîne** : **'\0'**
Il permet de délimiter la fin de la chaîne.

OUVRIR UN FICHIER

Pour **ouvrir le fichier** nommé mon_fichier.txt en écriture ASCII, on utilise la fonction suivante :

```
mon_fichier = fopen("my_file.txt", "w" );
```

Cette fonction renvoie l'adresse de la première donnée récupérée du fichier, si l'ouverture s'est faite normalement.
Sinon, elle renvoie la valeur **NULL**. On peut ainsi tester l'ouverture d'un fichier par le test suivant :

```
if(mon_fichier != NULL){ /*ouverture réussie*/ }
```

Le premier paramètre est le **chemin** (ici relatif) **vers le fichier** à ouvrir (une chaîne de caractères) avec son extension.
Le second paramètre est le **mode d'ouverture** du fichier :

- **r** : *read* / lecture
- **w** : *write* / écriture (en écrasant le fichier)
- **a** : *append* / écriture (en ajoutant au fichier existant)

On peut ajouter l'option **b** (binary) pour des fichiers binaires.

```
fopen("mon_fichier.bin", "rb" );
```

Ex : pour ouvrir le fichier binaire ***mon_fichier.bin*** en lecture

ÉCRIRE DES DONNÉES / FICHIER TEXTE

L'écriture dans un fichier texte se fait par transmission de chaines de caractères, grâce à la fonction :

```
void fprintf(FILE *fic, char *format, var1, var2...)
```

Par exemple, pour écrire le texte : VARIABLE A =, suivi de la valeur de la variable a (type double par exemple), on peut utiliser l'instruction suivante :

```
fprintf(mon_fichier, "VARIABLE A =%lf \n", a);
```

LIRE DES DONNÉES / FICHIER TEXTE

La lecture dans un fichier texte se fait par récupération de chaines de caractères, grâce à la fonction :

```
void fscanf(FILE *fic, char *format, adr_var1, ...)
```

Par exemple, pour récupérer une variable entière et une chaîne de caractères dans un fichier, on peut utiliser l'instruction suivante :

```
fscanf(mon_fichier, "%d %s", a, ma_chaine);
```

Ici **a** est de type entier et **ma_chaine** est une chaîne de caractères.

Attention : la lecture d'une chaîne de caractères à l'aide de la commande ***fscanf*** est limitée au premier espace. On peut aussi utiliser la commande ***fgets*** pour contourner ce problème :

```
fgets(ma_chaine, 128, mon_fichier);
```

Le second paramètre correspond à la taille maximale de la chaîne de caractères à récupérer.

Il existe également un caractère spécifique de fin de fichier : **EOF**.