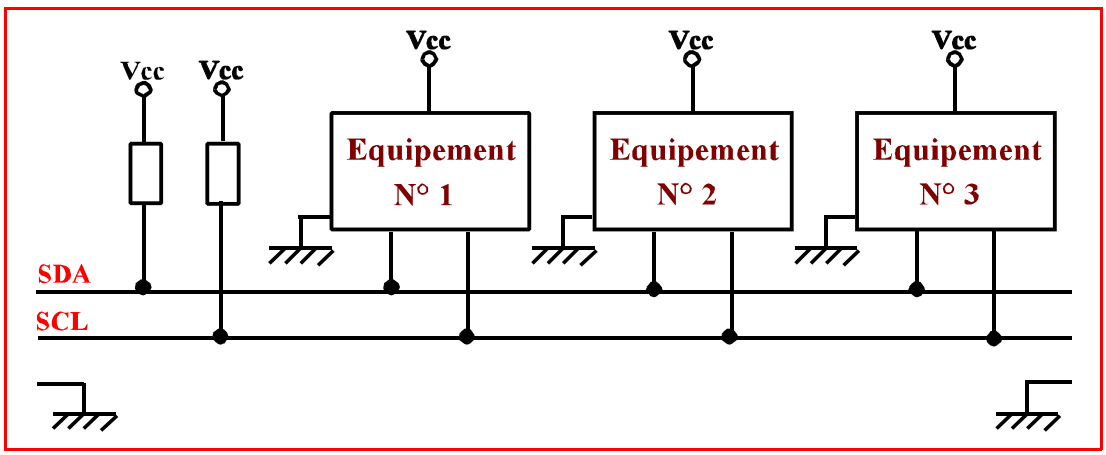
Le Bus I2C
(Inter-Integrated Circuit)

I) Présentation du projet :

II) Généralités sur le bus I2C :

Depuis qu'existent les circuits intégrés à grande échelle, les microprocesseurs et les micro-contrôleurs, le nombre d'appareils "intelligents" n'a cessé d'augmenter dans l'industrie. Malgré la miniaturisation constante, les concepteurs de systèmes sont toujours confrontés au problème de nombre croissant des liaisons entre les différents circuits intégrés. Le fabricant Phillips, gros producteur d'appareils de grande consommation, s'est inventé une solution astucieuse avec le bus I2C, qui permet de relier des circuits intégrés entre eux au moyen de deux fils seulement. Ce bus est devenu rapidement un standard.



- **adressage des circuits sur 7 bits**
- **vitesse plafonnée à 100 kbit/s**
- **compatibilité avec le CBUS (ancêtre de l'I2C chez Philips)**

Quelques circuits disponibles sur le marché :

- Extension à huit entrées-sorties parallèle (entrée/sortie 8 bits)
- Convertisseur A/N et N/A
- Mémoire Eeprom
- Mémoire Ram statique
- Horloge-calendrier en temps réel
- Récepteur infra-rouge (télécommande RC5)
- Capteur de température
- Décodeur télétexte
- Micro-contrôleurs à interface I2C intégrée (80C552, 80C652..)

Face à l'explosion du nombre de circuits I2C disponibles et au trafic en très forte augmentation, Philips a développé de nouvelle spécification de l'I2C : mode rapide et extension du protocole.

- compatibilité totale avec l'ancien I2C, mais abandon de la compatibilité CBUS
- adressage étendu sur 10 bits
- vitesse montée à 400 kbit/s

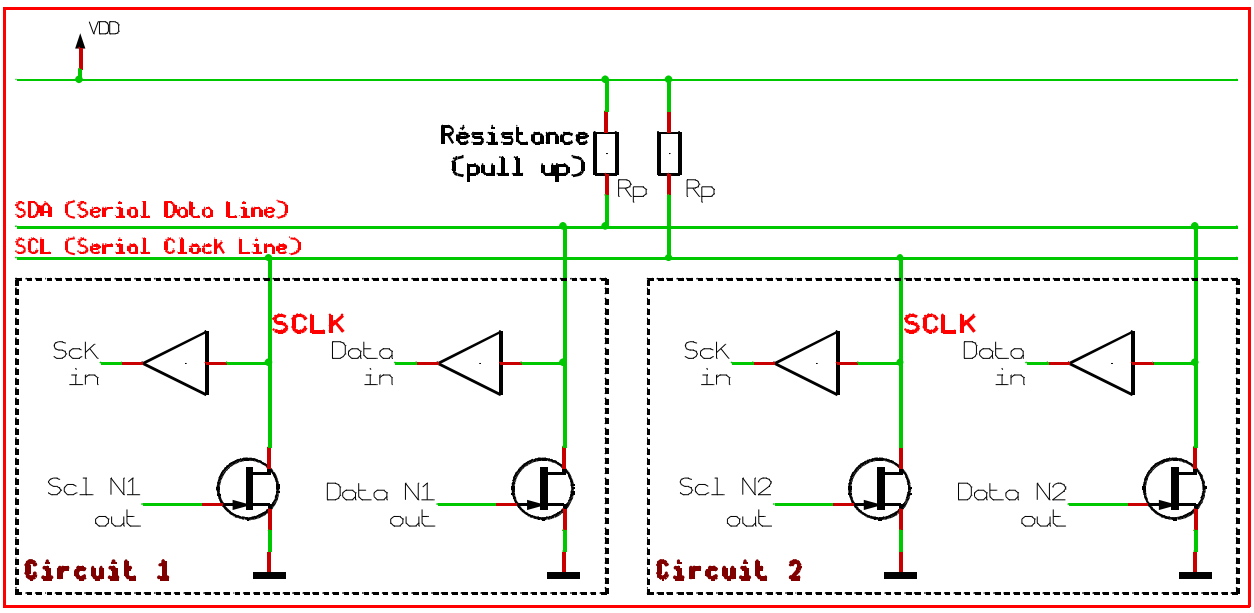
Le bus I2C a encore aujourd'hui le vent en poupe car il est de plus en plus utilisé dans l'électronique grand-public, parfois déguisé sous une norme pour des besoins particuliers comme le SMBus qui est implanté dans tous les nouveaux PC à base de chipset Intel 430TX ou postérieur, ou encore le fameux DDC qui équipe tous les moniteurs et cartes vidéo récents.

Très récement, Philips a introduit une nouvelle extension de la norme I2C qui étend sa vitesse à 3.4 Mbits/s, Aucune info pour le moment. A surveiller sur le site de Philips.

III) Spécifications du bus I2C.

1) Caractéristiques générales.

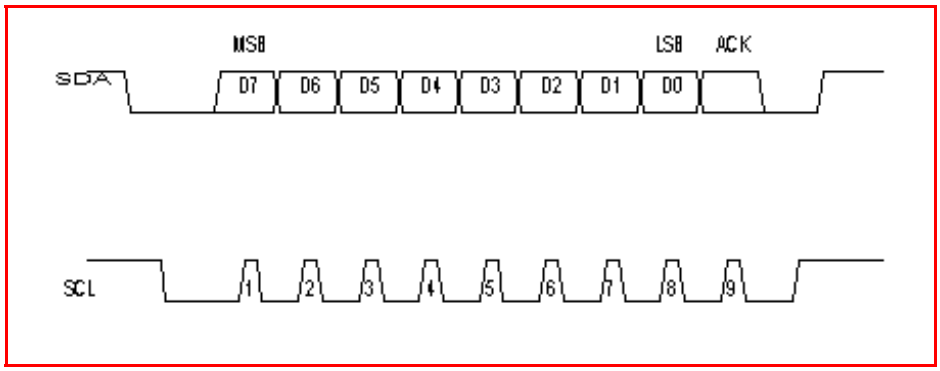
Les deux lignes du bus, SDA et SCL, sont bidirectionnelles. Elles sont toujours maintenues au niveau haut par une résistances de rappel *(pullup)*. Les deux lignes se trouvent donc au niveau haut quand le bus est libre. Les circuits connectés au bus doivent avoir une interface à collecteur ouvert pour les bipolaires, à drain ouvert pour les MOS. Cette configuration permet à un circuit quelconque de ramener à la masse le niveau d'une ligne, pour réaliser la fonction ET câblé. Le débit maximal dans la version standard du bus PC est de l00 kilobits par seconde; dans le mode rapide, une extension du protocole, il peut atteindre 400 kilobits par seconde. Le nombre de points de connexion n'est limité que par la capacité maximale de 400 pF sur le bus.



2) Protocole de communication.

Le protocole du bus I2C définit la succession des états possibles pour les signaux SDA et SCL. Le protocole défini aussi comment doivent réagir les nœuds raccordés au bus en cas de conflit.

Avant d'initier un dialogue sur le bus, un nœud doit s'assurer que le bus I2C est libre. Pour ce faire le nœud vérifiera si les lignes SDA et SCL sont au repos (à l'état haut) pendant un intervalle de temps suffisant. Pour transmettre des données sur un bus I2C, un nœud doit surveiller deux conditions particulières: la condition de départ " START " et la condition d'arrêt " STOP "



a) Condition de départ : START

Si le bus I2C est au repos, un nœud indiquera qu'il souhaite en prendre le contrôle en mettant tout d'abord la ligne SDA au niveau bas. Puis, après le temps défini par les spécifications du bus I2C, le circuit placera aussi la ligne SCL au niveau bas. Cette situation détermine la condition de départ, appelée "START". A partir de cet instant les autres nœud savent que le bus est occupé, et ils ne devraient pas tenter d'en prendre le contrôle. Les nœuds qui partagent le bus doivent continuer de scruter le bus I2C pour attendre la condition de fin appelée " STOP ". Ceci leur sera nécessaire avant de conclure que le bus I2C est réellement libre (le fait que les lignes SDA et SCL soient à l'état haut depuis longtemps ne signifie pas forcement que le bus est libre).

Lorsqu'un nœud prendre le contrôle du bus il en devient le "Maître". C'est toujours le maître qui produira le signal d'horloge, quelque soit le sens du transfert. Avant de placer sur la ligne SDA les bits qui forment l'octet à transmettre, le maître doit placer la ligne d'horloge SCL à 0. Pendant ce temps la ligne SDA peut changer de niveau.

Après la condition "START " le maître peut modifier l'état de la ligne pour débuter le transfert des données. Lorsque la ligne SDA est dans l'état requis le maître placera la ligne SCL au niveau haut. Ensuite, tant que la ligne SCL est au niveau haut, la ligne SDA doit rester dans un état stable. Après un temps minimum défini par les spécifications du bus I2C, le maître replacera la ligne SCL au niveau bas. Le maître peut alors enchaîner avec la transmission du bit suivant. Le cycle recommencera huit fois pour transmettre les huit bits de donnée. Notez que c'est le bit de poids fort qui est transmis en premier.

b) Acquittement.

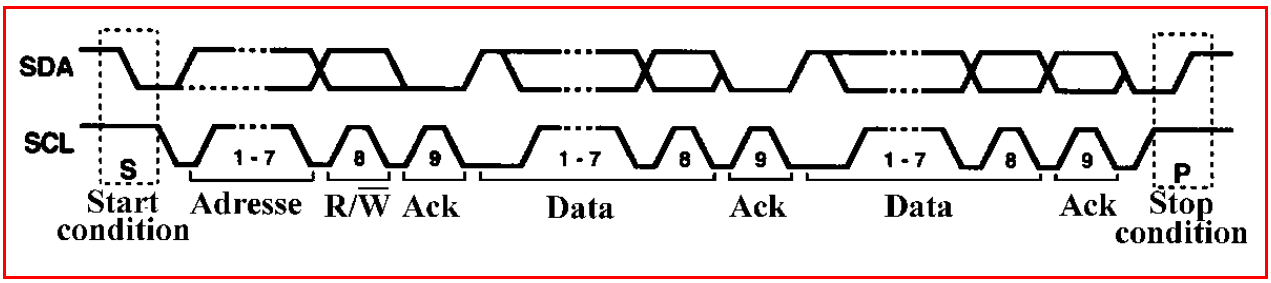
Pour chaque octet transmis sur le bus I2C, le circuit qui a reçu les données doit informer le maître que tout c'est bien passé. Pour cela le nœud récepteur doit imposer un bit d'acquittement (ACK) sur la ligne SDA. Pour ce faire, après la transmission d'un octet et pendant que la ligne SCL est encore au niveau bas, le maître place sa sortie au niveau haut (donc en haute impédance). Le maître génère ensuite un cycle d'horloge supplémentaire tandis que le nœud récepteur place sa sortie au niveau bas (ACK). Lorsque la ligne SCL passe au niveau haut le maître lit l'état la ligne SDA imposé par l'esclave. Si la valeur lue pour le bit " ACK " est un niveau bas, cela signifie que l'esclave s'est bien acquitté de l'octet reçu. Sinon c'est qu'il y a une erreur de transmission et le maître doit générer la condition arrêt, pour libérer le bus I2C.

c) Condition de STOP.

Habituellement la ligne SDA doit rester dans un état stable pendant que le signal SCL est à l'état haut. Cependant pour générer la condition d'arrêt "STOP" la ligne SDA doit passer du niveau bas vers le niveau haut, pendant que la ligne SCL est au niveau haut. Ensuite le signal SCL repasse lui aussi à l'état haut. Ensuite, si les lignes SDA et SCL restent au repos pendant un temps suffisant, le bus pourra être considéré comme libre.

d) Mode d'adressage.

Pour l'instant nous avons vu comment le maître prend le contrôle du bus I2C pour transmettre un octet. Le maître doit pouvoir choisir à quel nœud s'adresser. Dans ce but, le premier octet que transmet le maître correspond à l'adresse I2C du nœud choisit. Le nœud en question est appelé l'esclave ou la cible.



Le format d'une adresse I2C est un peu particulier. Le bit de poids faible (D0) est réservé pour indiquer si le maître demande une lecture à l'esclave ou bien au contraire si le maître impose une écriture à l'esclave. Il ne reste donc que 7 bits utiles pour former l'adresse de la cible. Il n'y a donc que 128 adresses distinctes possible pour un bus I2C. En réalité il existe un mode d'adressage étendu (adressage sur 10 bits) mais nous n'aborderons ici.

Chaque nœud connecté au bus I2C répond à une adresse qui doit être unique. Souvent l'adresse associée à un circuit est définie par des broches de sélection qui sont portées à VCC ou à la masse. Une partie de l'adresse est définie en usine par un "masque" qui dépendra de la fonction du circuit. Pour des nœuds équipés d'un microcontrôleur, l'adresse de réponse sur le bus I2C (Own Adresse) sera configurée par programme. Lors de la conception d'un système articulé autour d'un bus I2C il faut veiller à l'unicité des adresses attribuées aux différents composants.

Une fois que l'adresse est transmise sur le bus I2C, l'esclave concerné indique qu'il est présent en plaçant le bit ACK à 0. Si le bit ACK vaut 1 le maître comprend qu'il y a une erreur de sélection ou de transmission, et il génère la condition d'arrêt.

Si le bit D0 (R/W) de l'adresse cible, est à 0 cela signifie que le maître veut réaliser une opération d'écriture. Dans ce cas, la transmission des octets indiqués se répète autant de fois que nécessaire. Une seule condition arrêt survient, à la fin du transfert. Après chaque octet transmis, le maître vérifiera si l'esclave s'est bien acquitté de l'octet reçu. Si le bit ACK est valide le maître peut continuer d'envoyer des octets à l'esclave, ou bien, il peut décider de terminer le dialogue en envoyant la condition " STOP "

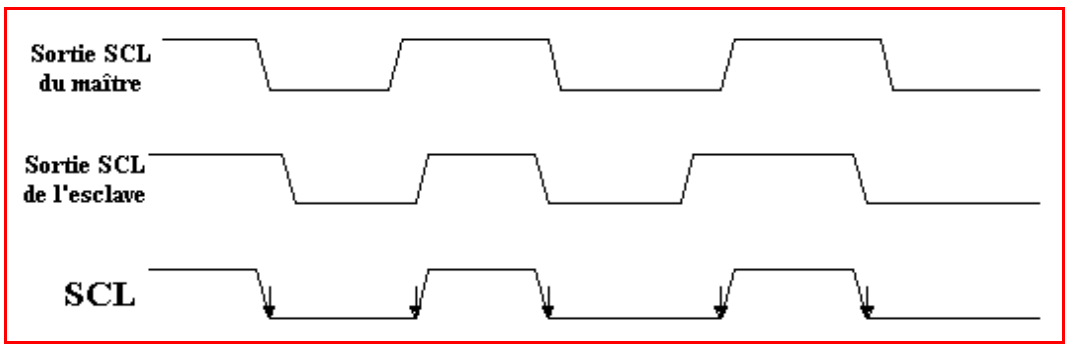
e) Répétition de Start (Repeated start).

Il est possible d'enchaîner écriture et lecture sans avoir à passer par une condition d'arrêt. Pour cela, le maître enverra une condition de nouveau départ (REPEATED START)

Cette possibilité présente un intérêt dans le cas où le bus I2C est partagé par plusieurs maîtres potentiels. Grâce à cette possibilité, le maître est assuré de garder le contrôle du bus aussi longtemps que nécessaire. Par contre si le maître libère le bus (suite à une condition d'arrêt), rien ne lui garantit qu'il obtiendra à nouveau le contrôle du bus immédiatement. Selon les opérations à effectuer cela peut être gênant. Cette possibilité est souvent utilisée lorsque la cible nécessite le chargement d'un pointeur avant de commencer une lecture.

f) Synchronisation.

Il est possible de ralentir la transmission en cours sur le bus I2C. Si un circuit esclave a besoin de ralentir les échanges avec le bus il lui suffit de maintenir la ligne SCL à état bas le temps nécessaire. Le circuit maître scrute en permanence la ligne SCL pour la comparer avec l'état qu'il souhaite lui même imposer. Quand le circuit maître détecte un niveau bas tandis qu'il vient de placer sa sortie SCL à état haut, il passe dans une boucle d'attente. L'attente se poursuit jusqu'à ce que la ligne SCL soit libérée par le circuit esclave.



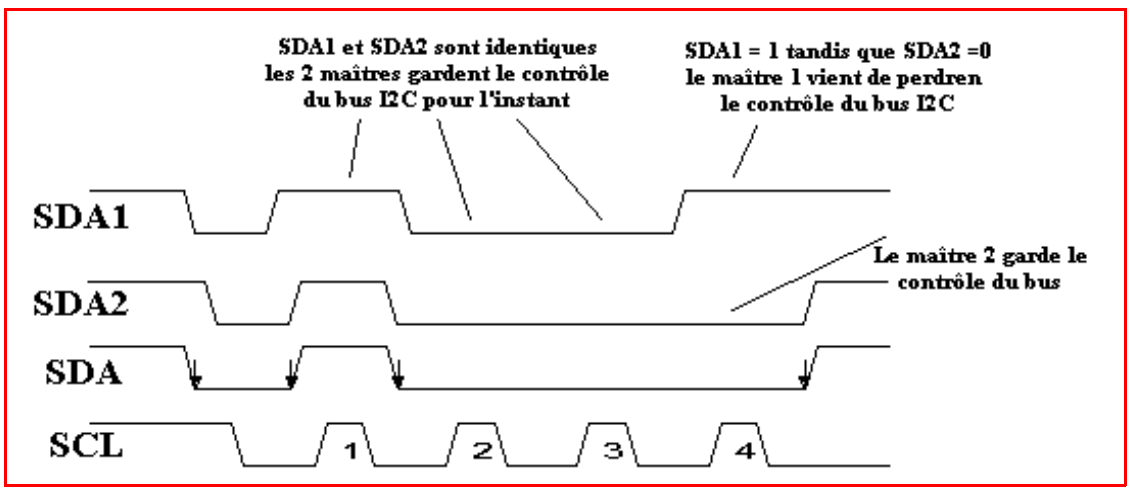
g) Arbitrage des échanges sur un bus I2C.

Imaginons le cas où plusieurs circuits tentent d'imposer la condition " START " aux lignes SDA et SCL, quasiment au même moment. Au départ les lignes SDA et SCL étant au niveau haut, tous les circuits voient le bus dans l'état de repos. Chaque circuit va placer la ligne SDA au niveau bas, à quelques nanosecondes d'écart, de sorte que chacun aura l'impression qu'il contrôle effectivement la ligne. Il en va de même pour la ligne SCL. Tant que tous les circuits imposent les mêmes états sur les lignes SDA et SCL il n'y a aucun conflit visible.

Pour déterminer qu'il y a un conflit les circuits raccordés au bus I2C doivent comparer en permanence l'état qu'ils demandent avec l'état effectif sur le bus. Dès que l'état diffère le nœud perd le contrôle du bus. Il doit alors devenir passif. Selon le moment où ils a perdu le contrôle du bus le nœud réagira différemment. Si le circuit a perdu le contrôle du bus avant la fin de la transmission de l'adresse cible il doit continuer à lire la ligne SDA au cas où il seraient lui-même la cible. Par contre s'il a perdu le contrôle du bus après la transmission de l'adresse cible ils se contentera d'attendre la condition d'arrêt.

Enfin, si les différents maîtres en compétitions tentent d'adresser la même cible, pour la même opération (R/W), la ligne SDA peut être toujours au niveau requis. La procédure d'arbitrage va alors continuer avec les données à transmettre. Dans le cas d'une opération de lecture, passé ce cap, il ne devrait plus y avoir de conflit puisque c'est le circuit esclave qui prend ensuite le contrôle de la ligne SDA. Les différents maîtres vont donc lire les mêmes données ensemble. Dans le cas d'une écriture vers la cible, si les différents maîtres écrivent les mêmes données, le conflit n'aura pas lieu d'exister (puisque tout les nœuds imposeront les mêmes états en même temps). La procédure d'arbitrage va donc se poursuivre jusqu'à ce que l'un des maîtres demande un niveau différent des autres.

En poussant le raisonnement à l'extrême on peut imaginer le cas où tous les nœuds demandent les mêmes états jusqu'à la condition d'arrêt. La probabilité d'un tel cas est faible. Quoi qu'il en soit, si cet événement se produit, la procédure d'arbitrage garantie que le résultat final sera correct. Chaque maître aura réalisé la même opération, exactement en même temps.



h) Appelle général (General call)

Lorsque l'adresse de la cible I2C est égale à la valeur zéro, il s'agit de la fonction "General call ". Dans ce cas, tous les circuits connectés au bus devraient, théoriquement, répondre par un acquittement (ACK). Dans la pratique les dispositifs simples ignoreront cette fonction car il s'agit de dispositifs qui se comportent essentiellement en esclaves. Le deuxième octet qui est transmis après l'adresse zéro détermine l'action à accomplir par les dispositifs qui auront répondu par un acquittement.

3) Caractéristiques électriques des lignes SDA et SCL.

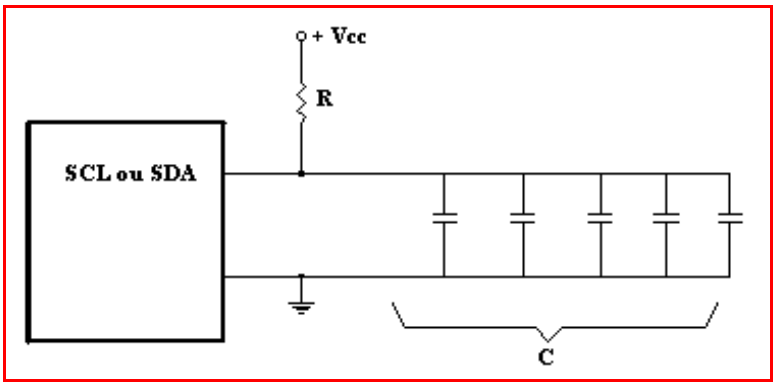
Contrairement aux signaux logiques de technologie TTL, les niveaux électriques des lignes SDA et SCL ne sont pas figés. Comme l'indiquent les spécifications du bus I2C, les niveaux dépendent de la tension d'alimentation, comme pour les circuit CMOS. Le niveau haut est défini pour une tension dépassant 0,7 x VCC tandis que le niveau bas est défini pour une tension qui reste inférieure à 0,3 x VCC..

Le courant maximum circulant dans les étages de sorties des lignes SDA et SCL est fixé à 3mA. Cette caractéristique a une incidence sur la valeur des résistances de rappel à VCC des lignes SDA et SCL, et par voie de conséquence sur les temps de montés des signaux. Le bus I2C supporte deux vitesses de transmission: le mode standard et le mode rapide.

Les spécifications du bus I2C indiquent que le temps de monté des signaux doit rester inférieur à 1us dans le mode standard, et 300ns dans le mode rapide. Par ailleurs la capacité maximum qui charge les lignes SDA et SCL ne doit pas dépasser 400pF, pour respecter les temps de descente.

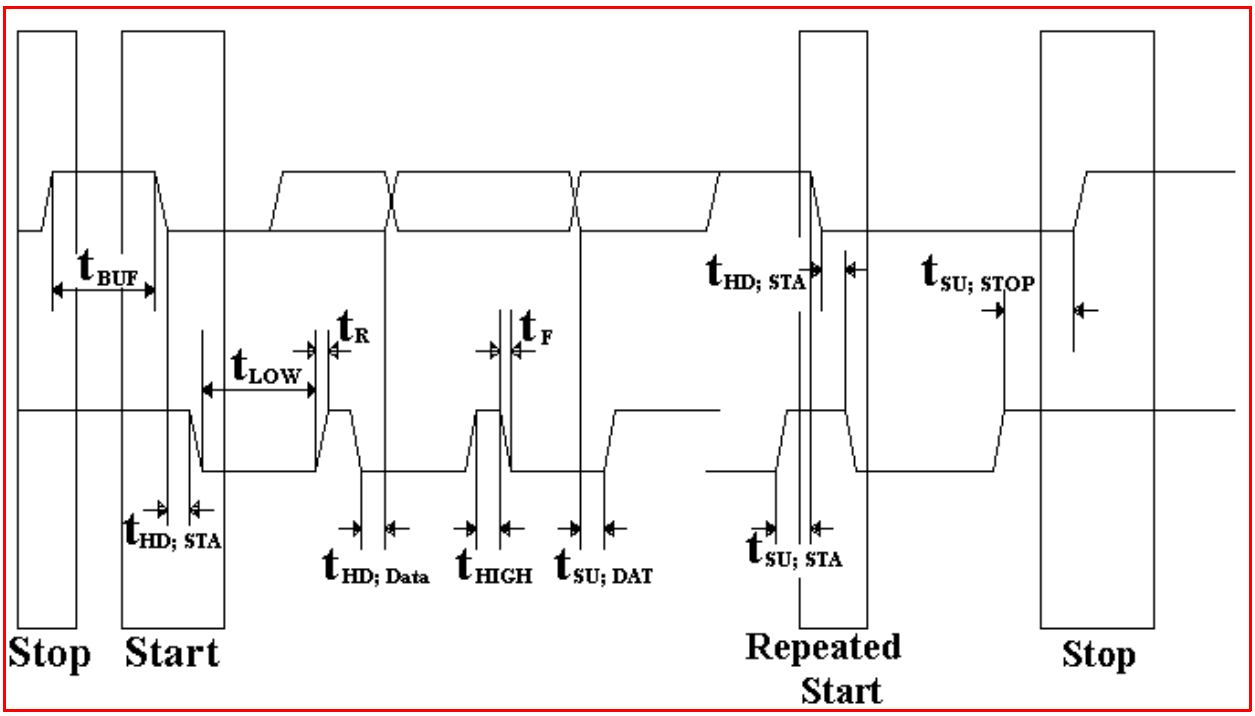
Le schéma représente le cas le plus simple pour piloter les lignes SDA et SCL. Dans ce cas de figure, pour ne pas dépasser le courant maximum que peuvent absorber les étages de sortie, la résistance aura pour valeur minimale

R = (Vcc-VSat) / Imax
soit environ 1,5K• pour Vcc = 5V.



Le temps de monté sera dépendant du réseau RC qui apparaît clairement sur le schéma. La capacité C sera le total des capacités d'entrées des différents circuits connectés au bus, et de la capacité répartie du câble de liaison. L'évolution de la tension, pour un front montant, suit la loi exponentielle suivante $V=Vcc(1 - e^{-t/RC})$. Le temps de monté s'établit alors à environ Tr = 2,2RC. Le temps de monté ne devant pas dépasser 1uS (dans le mode standard), la capacité totale de charge sur la ligne ne devra donc pas dépasser 303pF.

4) Les Spécifications essentielles du bus I2C.



V_{IL} Tension de définition du niveau bas

V_{IH} Tension de définition du niveau haut

V_{OL} Tension résiduelle de la sortie pour I=3mA

T_{of} Temps de descente de VIH min à VIL max avec une capacité de charge de 10pF à 400pF

C_{in} Capacité des entrées SDA et SCL

I_i Courant dans les lignes SDA et SCL, en tant qu'entrées

f_{SCL} Fréquence de l'horloge SCL

t_{BUF} Temps qui sépare une condition " STOP " d'une condition " START "

$t_{HD;\ SDA}$ Temps de maintient après la condition " START " ou " REPEATED START "

t_{LOW} Temps à l'état bas du signal SCL

t_{HIGH} Temps à l'état haut du signal SCL

$t_{SU;\ STA}$ Temps de setup avant la condition " START "

$t_{SU;\ DAT}$ Temps de setup avant lecture de la ligne SDA

t_R Temps de monté des signaux SDA et SCL

t_F Temps de descente des signaux SDA et SCL

$t_{SU;\ STOP}$ Temps de setup avant la condition " STOP "

C_b Capacité de charge pour une ligne SDA ou SCL

Tableau de caractéristiques:

Paramètres	Mode	Standard	Mode	Rapide	Unité
	Min	Max	Min	Max	
V_{IL}	- 0,5	0,3 Vcc	- 0,5	0,3 Vcc	V
V_{IH}	0,7 Vcc	Vcc + 0,5	0,7 Vcc	Vcc + 0,5	V
V_{OL} (Imax = 3mA)	0	0,4	0	0,4	V
T_{of}		250	$20 + 0,1C_b$	250	ns
C_{in}		10		10	pf
I_i	-10	10	-10	10	uA
f_{SCL}	0	100	0	400	kHz
t_{BUF}	4,7		1,3		µs
t_{HD} ; t_{SDA}	4,0		0,6		µs
t_{LOW}	4,7		1,3		µs
t_{HIGH}	4,0		0,6		µs
$t_{SU\ ;\ STA}$	4,7		0,6		µs
$t_{SU;\ DAT}$	250		100		ns
t_R		1000	$20+0,1C_b$	300	ns
t_F		300	$20+0,1C_b$	300	ns
$t_{SU\ ;\ STO}$	4,0		0,6		µs
C_b		400		400	pf

IV) Présentation de quelques circuits I2C.

1) PCF8574: Extension à huit entrées-sorties parallèles.

2) PCF8591: Convertisseur AN/NA.

3) ST24C08: Mémoire Eeprom 8K0 (4 x 256 x 8)

4) PCF8570: Mémoire Ram statique 128 x 8 bits / 256 x 8bits

5) 80C552: Microcontrôleur à interface I2c intégrée

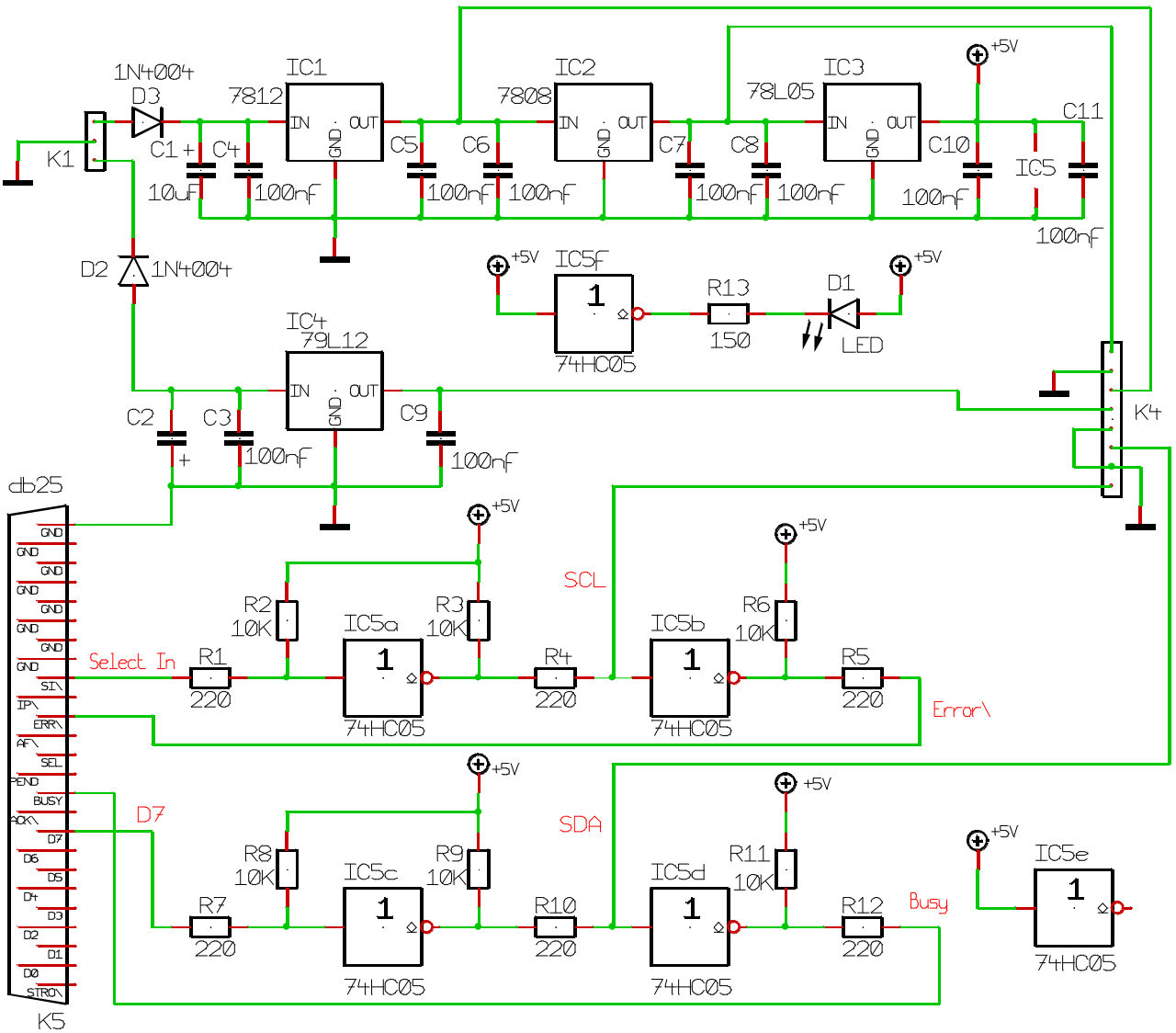
6) PCF8573: Horloge-calendrier en temps réel

V) Interface PC / Bus I2c universelle.

1) Le principe de fonctionnement.

L'interface proposée permet d'émuler un bus I2C à partir du port parallèle d'un PC. Cette interface contrôle le bus en mode multi-maîtres. Ici, le PC pourra soit être le maître, soit être un simple esclave en attente d'un message I2C. Cette interface simple assure des niveaux électriques compatibles avec ceux d'un vrai bus I2C.

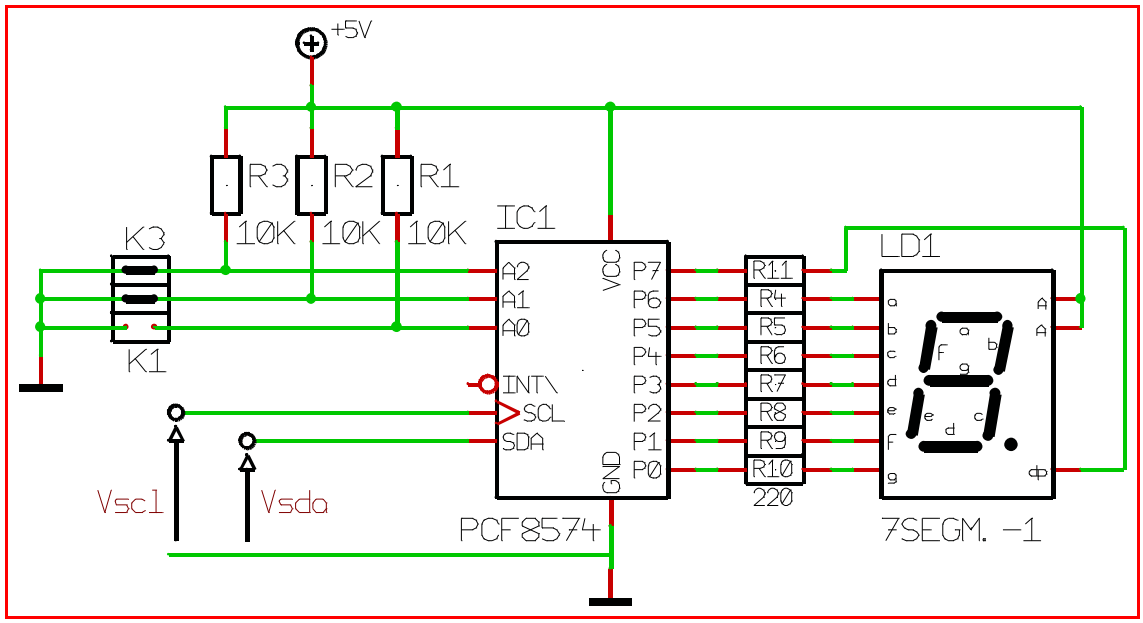
2) Schéma structurel.



Le montage est alimenté en + ou - 15V par une alimentation régulée et filtrée, et protégé par 2 diodes D2 et D3. La led rouge permet de visualiser la présence du +5V. Les broches 9 (D7) et 17 (Select In) servent à générer les lignes SDA et SCL. Les signaux sont inversés, par portes non (74HC05) sortie à collecteur ouvert. Les résistances R3 et R9 font office de Pull-up sur les lignes SDA et SCL. Celles-ci sont donc maintenue à l'état haut lorsque le bus est au repos. Lorsque les lignes D7 et Select In reçoivent un niveau haut, les lignes SDA et SCL sont amenés au niveau bas. Les entrées 11 (Busy) et 15 (Error\ ) du port centronic servent de retour aux niveaux présents sur les lignes SDA et SCL. Les résistances R1, R5, R7 et R12 servent à limiter le courant sur les entrées du port //.

VI) TD d'application

1) Soit le montage suivant:



a) A partir de la documentation constructeur, déterminer la valeur de l'adresse du circuit Ic1.

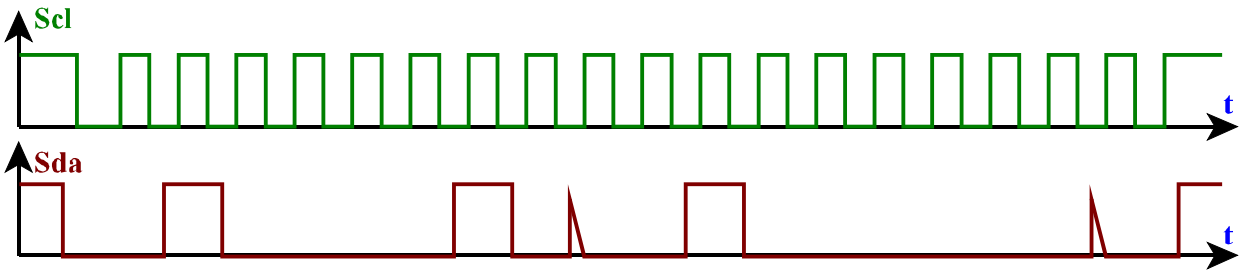
b) Indiquer les niveaux logiques nécessaires en sortie de IC1 (P0 à P7) pour visualiser la valeur 3 sur l'afficheur.

c) Calculer le courant passant dans les segments 'a' et 'f' de l'afficheur. (valeur affichée 3) (segment allumé: Vseg =1,6v).

d) A partir de la documentation constructeur, tracer les chronogrammes de commande de Vscl et Vsda en correspondance qui permettent de visualiser la valeur 3 sur l'afficheur.

e) Donner la procédure des codes hexadécimal à envoyer par la ligne 'SDA' qui permet de décompter de 9 à 0 en 10 secondes.

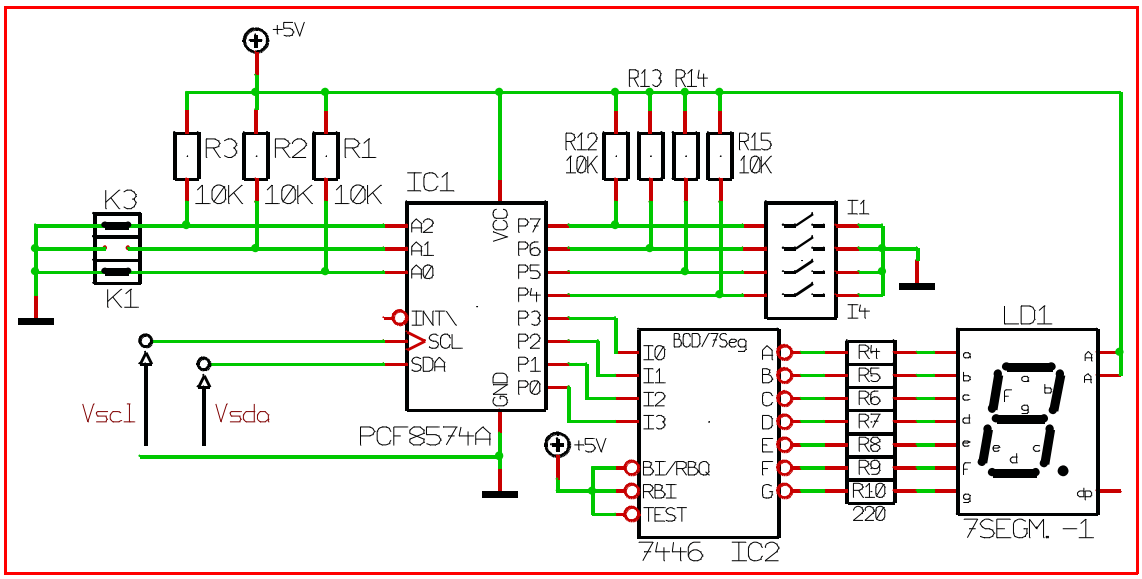
f) Analyser le chronogramme suivant:



- Repérer la condition de start, les bits d'adressage, le bit R/W, l'accusé de réception, les bits de la donnée et la condition de stop

- Que réalise l'envoie de ces deux chronogrammes sur le montage.

2) Soit le montage suivant:



a) A partir de la documentation constructeur, déterminer la valeur de l'adresse du circuit Ic1.

b) Indiquer les niveaux logiques nécessaires en sortie de IC1 (P0 à P7) pour visualiser la valeur 5 sur l'afficheur.

c) A partir de la documentation constructeur, tracer les chronogrammes de commande de Vscl et Vsda en correspondance qui permettent de visualiser la valeur 5 sur l'afficheur.

d) A partir de la documentation constructeur, tracer les chronogrammes de commande de Vscl et Vsda en correspondance qui permettent de lire la position des interrupteurs I1 à I4 sachant que I1, I2 sont fermés et I3, I4 sont au repos.

e) Donner la séquence de transmission des lignes Scl et Sda qui permet de lire la valeur des interrupteurs et commande l'afficheur tel que:

I4	I3	I2	I1	AFF
on	on	on	off	1
on	on	off	off	3
on	off	off	on	6
off	on	on	off	9

VII) Travaux Pratiques.

- Réaliser le montage du schéma 2 du Td d'application à l'aide des blocs logiques. On utilisera les sorties LED du bloc logique pour visualiser les lignes Sda et Scl.
- Simuler la trame de transmission des lignes Sda et Scl à l'aide des interrupteurs du bloc logique pour afficher la valeur '5'.Tracer les chronogrammes de cette transmission.
- Simuler la trame de transmission des lignes Sda et Scl à l'aide des interrupteurs du bloc logique pour lire l'état des interrupteurs I1 à I4 du montage. Tracer les chronogrammes de cette transmission.