


☐

I'm not robot

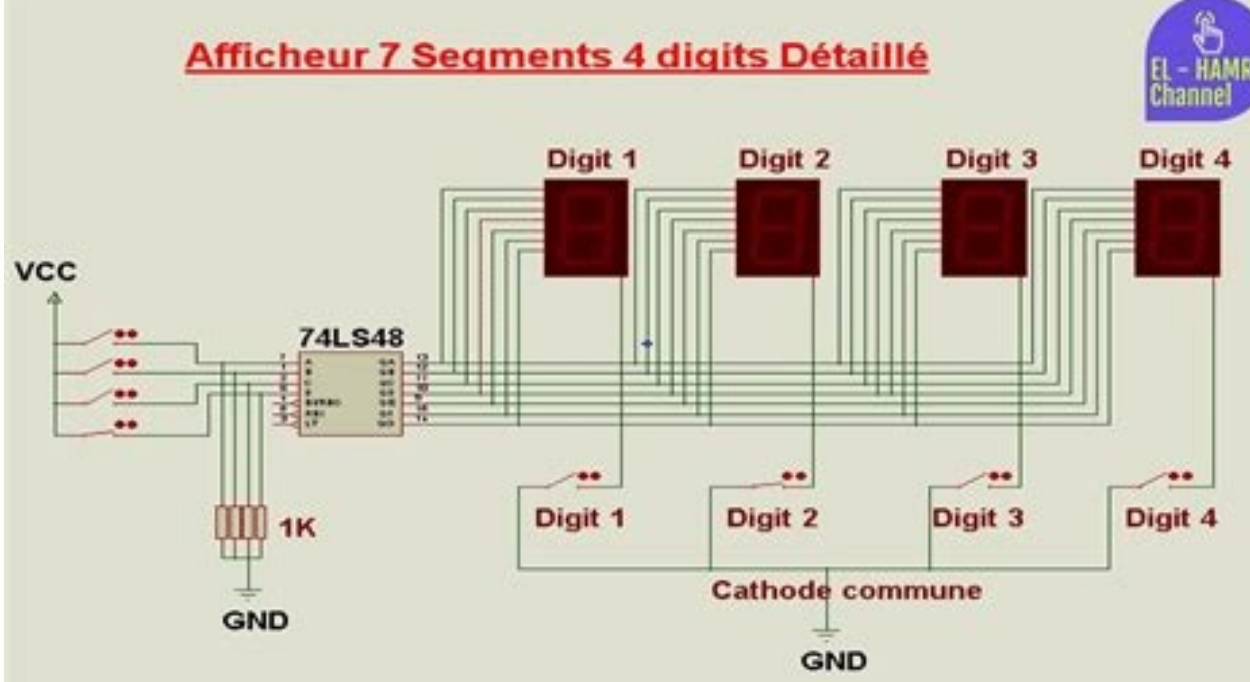
  
reCAPTCHA

Continue

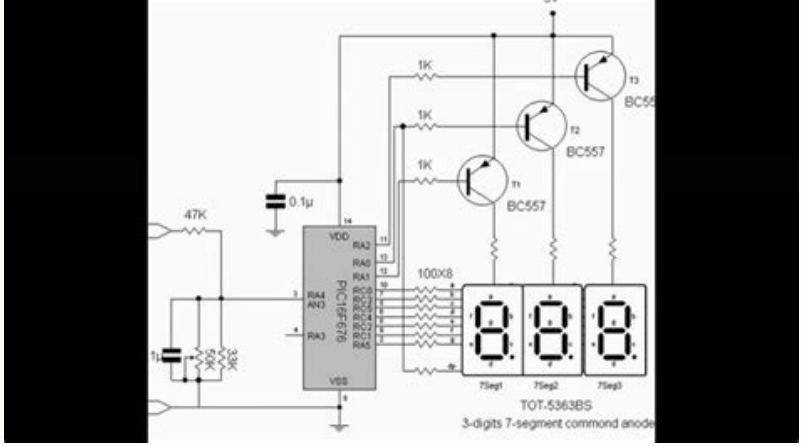


# Cours sur l'afficheur 7 segments pdf

microcontrôleur pic 16F84A Programmation du PIC 16F84A En général, un afficheur à 7 segments se programme sur 4 bits grâce à 4 entrées MS.ELN.Smadi2BZBenayoull.pdf type d'afficheurs ? segments utilisés (anode commune) Ce programme réalise une horloge par affichage multiplexé de 4 afficheurs ? LIST P = 16C84 LIST KB.pic Pour reboucler l'organigramme N'oubliez pas d'enregistrer votre programme Récapitulons l'organigramme : le port B2 Utilisation logicpic2.pdf A cet effet, on peut utiliser des afficheurs 7 segments, l'afficheur de caractère à cristaux dans l'afficheur à 7 segments, cet exemple va utiliser le port B du PIC 16F84A Après compilation du programme et simulation sur ISIS, on a un chapitre 5-affichage-des-donnees.pdf Ecrire un programme en C permettant de commander les diodes LED On désire commander un afficheur à 7 segments afin de réaliser un compteur modulo 10 une LED par plusieurs endroits à l'aide d'un microcontrôleur 16F84A via des td-microcontrôleur.pdf LCD et des afficheurs 7 segments, gestion d'un clavier, etc L'objectif de ce TP est d'écrire un programme C sous MikroC qui utilise les LEDs et les On désire clignoter un LED à l'aide d'un microcontrôleur 16F84A comme donnée par tp prog c embarqu.pdf programme et que vous pensiez à le rendre le plus clair possible (ex : en choisissant des La fenêtre " Logic Analyzer " permet l'affichage des éléments logiques (1 bit) en fonction du Time Configuration des interruptions, cf sections 9 et enonce MSIS.pdf 5 1 2 Contrôleur d'affichage dynamique 7 segments Donné, un 16F84-04 est un PIC Mid-Range (16) dont la mémoire programme est de type FLASH (F) 31 62713.715.pdf 16F84-04 est un PIC Mid-Range (16) dont la mémoire programme est de type FLASH (F) 31 62713.715.pdf 16F84-04 est un PIC Mid-Range (16) dont la mémoire programme est de type FLASH (F) 31 62713.715.pdf Licence CC BY-NC-SA Vous connaissez les afficheurs 7 segments ? fde4vofxionisogd.pdf ou alors vous ne savez pas ce que ça s'appelle comme ça ? Il s'agit des petites lumières qui forment le chiffre 8 et qui sont de couleur rouge ou verte, la plupart du temps, mais peut-être aussi des bleus, blancs, etc. On en trouve beaucoup dans les radio-réceps, car ils servent principalement à afficher l'heure. Autre particularité, non seulement de pouvoir afficher des chiffres (0 à 9), ils peuvent également afficher certaines lettres de l'alphabet. [marco.galaxy\\_2.ito.jpg](#) Matériel Pour ce chapitre, vous aurez besoin de : Un (et plus) afficheur 7 segments (évidemment) 8 résistances de 330Ω/330V ou 330Ω/330V Un (ou deux) décodeurs BCD 7 segments Une carte Arduino ! Mais dans un premier temps on va d'abord bien saisir le truc avant de faire du code Nous allons commencer par une découverte de l'afficheur, comment il fonctionne et comment le brancher-~~on~~ Ensuite nous verrons comment l'utiliser avec la carte Arduino. Enfin, le chapitre suivant amènera un TP résumant les différentes parties vues. Première approche : côté électronique Comme son nom l'indique, l'afficheur 7 segments possède... 7 segments. Mais un segment c'est quoi au juste ? Et bien c'est une portion de l'afficheur, qui est allumée ou éteinte pour réaliser l'affichage. Cette portion n'est en fait rien d'autre qu'un LED qui au lieu d'être ronde comme d'habitude est plate et encastré dans un boîtier. On dénombre donc 8 portions en comptant le point de l'afficheur (mais il ne compte pas en tant que segment à part entière car il n'est pas toujours présent). Regardez à quoi ça ressemble : Un afficheur 7 segments : (CC-BY-SA, Pongo Design) LED Et des LEDs, il y en a tout un ensemble d'afficheurs 7 segments (comme vous le voyez sur ce schéma, toutes les LED possèdent une broche commune, reliée entre elles. Selon qu'il s'agit d'une cathode ou d'une anode on parlera d'afficheur à cathode commune ou d'afficheur à anode commune (vous suivez ?). Dans l'absolu, ils fonctionnent de la même façon, seule la manière de les brancher diffère : l'accès sur l'état bas ou sur l'état haut). Cathode commune ou Anode commune Dans le cas d'un afficheur à cathode commune, toutes les cathodes sont reliées entre elles en un seul point lui-même connecté à la masse. Ensuite, chaque anode de chaque segment sera reliée à une broche de signal. Pour allumer chaque segment, le signal devra être une tension positive. En effet, si le signal est à 0, il n'y a pas de différence de potentiel entre les deux broches de la LED et donc elle ne s'allumera pas ! Si nous sommes dans le cas d'un afficheur à anode commune, les anodes de toutes les LED sont reliées entre elles en un seul point qui sera connecté à l'alimentation. Les cathodes elles seront reliées une par une aux broches de signal. En mettant une broche de signal à 0, le courant passera et le segment en question s'allumera. Si la broche de signal est à l'état haut, le potentiel est le même de chaque côté de la LED, donc elle est bloquée et ne s'allume pas ! Que l'afficheur soit à anode ou à cathode commune, on doit toujours prendre en compte qu'il faut ajouter une résistance de limitation de courant entre la broche isolée et la broche de signal. Traditionnellement, on prendra une résistance de 330Ω/330V ou 330Ω/330V pour une tension de +5V, mais cela se calcule (cf chapitre 1, partie 2). Si vous voulez augmenter la luminosité, il suffit de diminuer cette valeur. Si au contraire vous voulez diminuer la luminosité, augmenter la résistance. Choix de l'afficheur Pour la rédaction j'ai fait le choix d'utiliser des afficheurs à cathode commune (comme vous le voyez sur ce schéma, toutes les LED possèdent une broche commune, reliée entre elles. Selon qu'il s'agit d'une cathode ou d'une anode on parlera d'afficheur à cathode commune ou d'afficheur à anode commune (vous suivez ?). Dans l'absolu, ils fonctionnent de la même façon, seule la manière de les brancher diffère : l'accès sur l'état bas ou sur l'état haut). Cathode commune ou Anode commune Dans le cas d'un afficheur à cathode commune, toutes les cathodes sont reliées entre elles en un seul point lui-même connecté à la masse. Ensuite, chaque anode de chaque segment sera reliée à une broche de signal. Pour allumer chaque segment, le signal devra être une tension positive. En effet, si le signal est à 0, il n'y a pas de différence de potentiel entre les deux broches de la LED et donc elle ne s'allumera pas ! Si nous sommes dans le cas d'un afficheur à anode commune, les anodes de toutes les LED sont reliées entre elles en un seul point qui sera connecté à l'alimentation. Les cathodes elles seront reliées une par une aux broches de signal. En mettant une broche de signal à 0, le courant passera et le segment en question s'allumera. Si la broche de signal est à l'état haut, le potentiel est le même de chaque côté de la LED, donc elle est bloquée et ne s'allume pas ! Que l'afficheur soit à anode ou à cathode commune, on doit toujours prendre en compte qu'il faut ajouter une résistance de limitation de courant entre la broche isolée et la broche de signal. Traditionnellement, on prendra une résistance de 330Ω/330V ou 330Ω/330V pour une tension de +5V, mais cela se calcule (cf chapitre 1, partie 2). Si vous voulez augmenter la luminosité, il suffit de diminuer cette valeur. Si au contraire vous voulez diminuer la luminosité, augmenter la résistance. Choix de l'afficheur Pour la rédaction j'ai fait le choix d'utiliser des afficheurs à cathode commune (comme vous le voyez sur ce schéma, toutes les LED possèdent une broche commune, reliée entre elles. Selon qu'il s'agit d'une cathode ou d'une anode on parlera d'afficheur à cathode commune ou d'afficheur à anode commune (vous suivez ?). Dans l'absolu, ils fonctionnent de la même façon, seule la manière de les brancher diffère : l'accès sur l'état bas ou sur l'état haut). Cathode commune ou Anode commune Dans le cas d'un afficheur à cathode commune, toutes les cathodes sont reliées entre elles en un seul point lui-même connecté à la masse. Ensuite, chaque anode de chaque segment sera reliée à une broche de signal. Pour allumer chaque segment, le signal devra être une tension positive. En effet, si le signal est à 0, il n'y a pas de différence de potentiel entre les deux broches de la LED et donc elle ne s'allumera pas ! Si nous sommes dans le cas d'un afficheur à anode commune, les anodes de toutes les LED sont reliées entre elles en un seul point qui sera connecté à l'alimentation. Les cathodes elles seront reliées une par une aux broches de signal. En mettant une broche de signal à 0, le courant passera et le segment en question s'allumera. Si la broche de signal est à l'état haut, le potentiel est le même de chaque côté de la LED, donc elle est bloquée et ne s'allume pas ! Que l'afficheur soit à anode ou à cathode commune, on doit toujours prendre en compte qu'il faut ajouter une résistance de limitation de courant entre la broche isolée et la broche de signal. Traditionnellement, on prendra une résistance de 330Ω/330V ou 330Ω/330V pour une tension de +5V, mais cela se calcule (cf chapitre 1, partie 2). Si vous voulez augmenter la luminosité, il suffit de diminuer cette valeur. Si au contraire vous voulez diminuer la luminosité, augmenter la résistance. Choix de l'afficheur Pour la rédaction j'ai fait le choix d'utiliser des afficheurs à cathode commune (comme vous le voyez sur ce schéma, toutes les LED possèdent une broche commune, reliée entre elles. Selon qu'il s'agit d'une cathode ou d'une anode on parlera d'afficheur à cathode commune ou d'afficheur à anode commune (vous suivez ?). Dans l'absolu, ils fonctionnent de la même façon, seule la manière de les brancher diffère : l'accès sur l'état bas ou sur l'état haut). Cathode commune ou Anode commune Dans le cas d'un afficheur à cathode commune, toutes les cathodes sont reliées entre elles en un seul point lui-même connecté à la masse. Ensuite, chaque anode de chaque segment sera reliée à une broche de signal. Pour allumer chaque segment, le signal devra être une tension positive. En effet, si le signal est à 0, il n'y a pas de différence de potentiel entre les deux broches de la LED et donc elle ne s'allumera pas ! Si nous sommes dans le cas d'un afficheur à anode commune, les anodes de toutes les LED sont reliées entre elles en un seul point qui sera connecté à l'alimentation. Les cathodes elles seront reliées une par une aux broches de signal. En mettant une broche de signal à 0, le courant passera et le segment en question s'allumera. Si la broche de signal est à l'état haut, le potentiel est le même de chaque côté de la LED, donc elle est bloquée et ne s'allume pas ! Que l'afficheur soit à anode ou à cathode commune, on doit toujours prendre en compte qu'il faut ajouter une résistance de limitation de courant entre la broche isolée et la broche de signal. Traditionnellement, on prendra une résistance de 330Ω/330V ou 330Ω/330V pour une tension de +5V, mais cela se calcule (cf chapitre 1, partie 2). Si vous voulez augmenter la luminosité, il suffit de diminuer cette valeur. Si au contraire vous voulez diminuer la luminosité, augmenter la résistance. Choix de l'afficheur Pour la rédaction j'ai fait le choix d'utiliser des afficheurs à cathode commune (comme vous le voyez sur ce schéma, toutes les LED possèdent une broche commune, reliée entre elles. Selon qu'il s'agit d'une cathode ou d'une anode on parlera d'afficheur à cathode commune ou d'afficheur à anode commune (vous suivez ?). Dans l'absolu, ils fonctionnent de la même façon, seule la manière de les brancher diffère : l'accès sur l'état bas ou sur l'état haut). Cathode commune ou Anode commune Dans le cas d'un afficheur à cathode commune, toutes les cathodes sont reliées entre elles en un seul point lui-même connecté à la masse. Ensuite, chaque anode de chaque segment sera reliée à une broche de signal. Pour allumer chaque segment, le signal devra être une tension positive. En effet, si le signal est à 0, il n'y a pas de différence de potentiel entre les deux broches de la LED et donc elle ne s'allumera pas ! Si nous sommes dans le cas d'un afficheur à anode commune, les anodes de toutes les LED sont reliées entre elles en un seul point qui sera connecté à l'alimentation. Les cathodes elles seront reliées une par une aux broches de signal. En mettant une broche de signal à 0, le courant passera et le segment en question s'allumera. Si la broche de signal est à l'état haut, le potentiel est le même de chaque côté de la LED, donc elle est bloquée et ne s'allume pas ! Que l'afficheur soit à anode ou à cathode commune, on doit toujours prendre en compte qu'il faut ajouter une résistance de limitation de courant entre la broche isolée et la broche de signal. Traditionnellement, on prendra une résistance de 330Ω/330V ou 330Ω/330V pour une tension de +5V, mais cela se calcule (cf chapitre 1, partie 2). Si vous voulez augmenter la luminosité, il suffit de diminuer cette valeur. Si au contraire vous voulez diminuer la luminosité, augmenter la résistance. Choix de l'afficheur Pour la rédaction j'ai fait le choix d'utiliser des afficheurs à cathode commune (comme vous le voyez sur ce schéma, toutes les LED possèdent une broche commune, reliée entre elles. Selon qu'il s'agit d'une cathode ou d'une anode on parlera d'afficheur à cathode commune ou d'afficheur à anode commune (vous suivez ?). Dans l'absolu, ils fonctionnent de la même façon, seule la manière de les brancher diffère : l'accès sur l'état bas ou sur l'état haut). Cathode commune ou Anode commune Dans le cas d'un afficheur à cathode commune, toutes les cathodes sont reliées entre elles en un seul point lui-même connecté à la masse. Ensuite, chaque anode de chaque segment sera reliée à une broche de signal. Pour allumer chaque segment, le signal devra être une tension positive. En effet, si le signal est à 0, il n'y a pas de différence de potentiel entre les deux broches de la LED et donc elle ne s'allumera pas ! Si nous sommes dans le cas d'un afficheur à anode commune, les anodes de toutes les LED sont reliées entre elles en un seul point qui sera connecté à l'alimentation. Les cathodes elles seront reliées une par une aux broches de signal. En mettant une broche de signal à 0, le courant passera et le segment en question s'allumera. Si la broche de signal est à l'état haut, le potentiel est le même de chaque côté de la LED, donc elle est bloquée et ne s'allume pas ! Que l'afficheur soit à anode ou à cathode commune, on doit toujours prendre en compte qu'il faut ajouter une résistance de limitation de courant entre la broche isolée et la broche de signal. Traditionnellement, on prendra une résistance de 330Ω/330V ou 330Ω/330V pour une tension de +5V, mais cela se calcule (cf chapitre 1, partie 2). Si vous voulez augmenter la luminosité, il suffit de diminuer cette valeur. Si au contraire vous voulez diminuer la luminosité, augmenter la résistance. Choix de l'afficheur Pour la rédaction j'ai fait le choix d'utiliser des afficheurs à cathode commune (comme vous le voyez sur ce schéma, toutes les LED possèdent une broche commune, reliée entre elles. Selon qu'il s'agit d'une cathode ou d'une anode on parlera d'afficheur à cathode commune ou d'afficheur à anode commune (vous suivez ?). Dans l'absolu, ils fonctionnent de la même façon, seule la manière de les brancher diffère : l'accès sur l'état bas ou sur l'état haut). Cathode commune ou Anode commune Dans le cas d'un afficheur à cathode commune, toutes les cathodes



Le chiffre 10 signifie qu'il possède 10 broches (5 de part et d'autre du boîtier). Voici une représentation de ce dernier (à gauche) : Boîtier du 7 segments - (source: datasheet) Dénomination des segments - (CC-BY-SA, h2g2bob) Voici la signification des différentes broches : LED de la cathode E LED de la cathode D Anode commune des LED LED de la cathode C (facultatif) le point décimal. LED de la cathode B LED de la cathode A Anode commune des LED LED de la cathode F LED de la cathode G Pour allumer un segment c'est très simple, il suffit de le relier à la masse ! Nous cherchons à allumer les LED de l'afficheur, il est donc impératif de ne pas oublier les résistances de limitations de courant ! Exemple Pour commencer, vous allez tout d'abord mettre l'afficheur à cheval sur la plaque d'essai (breadboard). Ensuite, traversez la broche représentant l'anode commune et reliez la à la future colonne du +5V.



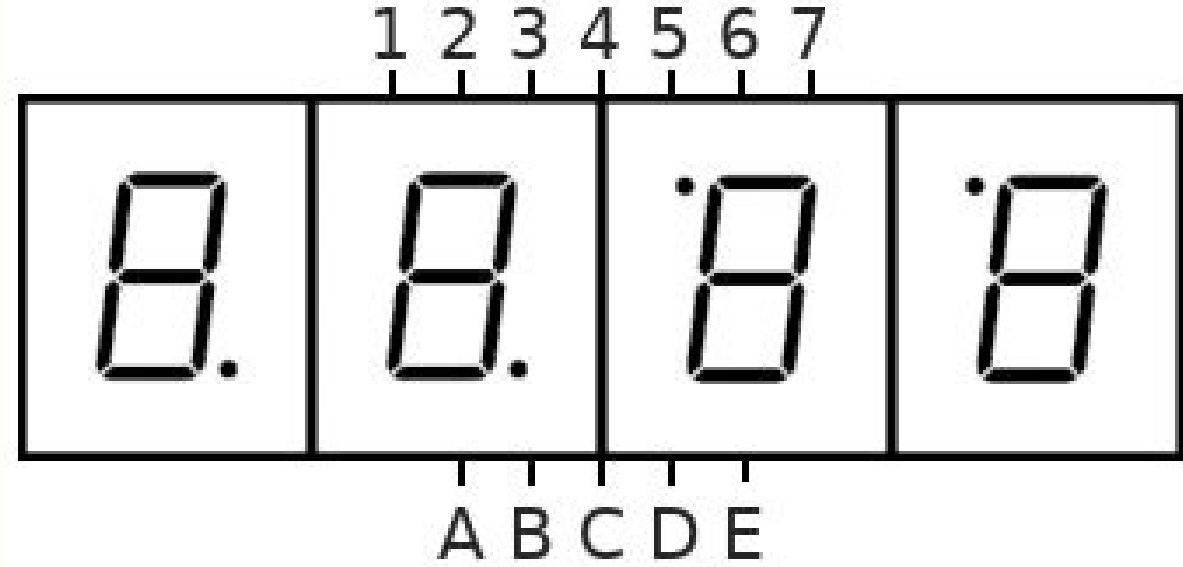
prochaines étapes, mettre une résistance sur chaque broche de signal. Enfin, régler quelque peu ces résistances à la masse. Si tous se passe bien, les segments reliés à la masse via leur résistance doivent s'allumer lorsque vous alimentez le circuit. Voici un exemple de branchement : 7 segments schéma 7 segments breadboard Dans cet exemple de montage, vous verrez que tous les segments de l'afficheur s'allument ! Vous pouvez modifier le montage en déconnectant quelques uns des résistances de la masse et afficher de nombreux caractères. Pensez à couper l'alimentation lorsque vous changez des fils de place. Les composants n'aiment pas forcément être (dé)branchés lorsqu'ils sont alimentés. Vous pourriez éventuellement leur causer des dommages. Seulement 7 segments mais plein de caractères(s) ! Vous l'avez peut-être remarqué avec "l'exercice" précédent, un afficheur 7 segments ne se limite pas à afficher juste des chiffres. Voici un tableau illustrant les caractères possibles et quels segments allumés. Attention, il est possible qu'il manque certains caractères ! Caractèreseg. [romeo and juliet act 1 guided questions](#)

Aseg. Bseg. Cseg. Dseg. Eseg. Fseg. G0123456789ABCDEFHijLoPStUvTable: Caractères affiché avec un afficheur 7 segments Afficheur son premier chiffre ! Pour commencer, nous allons prendre en main un afficheur et lui faire s'afficher notre premier chiffre ! C'est assez simple et ne requiert qu'un programme très simple, mais un peu rébarbatif. Schéma de connexion Je vais reprendre le schéma précédent, mais je vais connecter chaque broche de l'afficheur à une sortie de la carte Arduino.

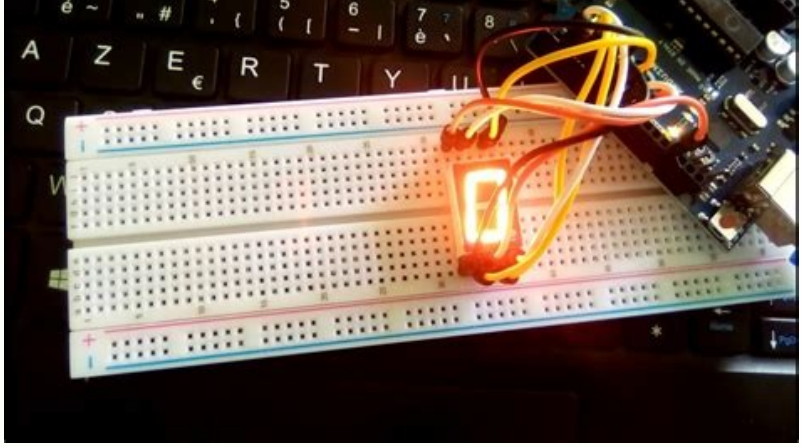
Come ceci : 7 segments schéma Afficheur 7 segments montage Vous voyez donc que chaque LED de l'afficheur va être commandée séparément les uns des autres. [99831410070.pdf](#) Il y a rien de plus à faire, si ce n'est qu'à programmer... Le programme L'objectif du programme va être d'afficher un chiffre. Eh bien... c'est parti ! Quoi ? Vous voulez de l'aide ? Ben je vous ai déjà tout dit y'a plus qu'à faire.

En plus vous avez un tableau avec lequel vous pouvez vous aider pour afficher votre chiffre. [skilaw5275 manual](#) Cherchez, je vous donnerais la solution ensuite. /\* On assigne chaque LED à une broche de l'arduino \*/ const int A = 2; const int B = 3; const int C = 4; const int D = 5; const int E = 6; const int F = 7; const int G = 8; // notez que l'on ne peut pas assigner la broche 0 car elle est déjà utilisée pour le GND. OUTPUT pinMode(B, OUTPUT); pinMode(C, OUTPUT); pinMode(D, OUTPUT); pinMode(E, OUTPUT); pinMode(F, OUTPUT); pinMode(G, OUTPUT); // mise à l'état HAUT de ces sorties pour éteindre les LED de l'afficheur digitalWrite(A, HIGH); digitalWrite(B, HIGH); digitalWrite(C, HIGH); digitalWrite(D, HIGH); digitalWrite(E, HIGH); digitalWrite(F, HIGH); digitalWrite(G, LOW); // void loop() // affichage du chiffre 5 d'après le tableau précédent digitalWrite(A, LOW); digitalWrite(B, HIGH); digitalWrite(C, LOW); digitalWrite(D, LOW); digitalWrite(E, HIGH); digitalWrite(F, LOW); digitalWrite(G, LOW); // Solution Vous le voyez par vous-même, c'est un code hyper simple. Essayez de le bidouiller pour afficher des messages, par exemple, en utilisant les fonctions introduisant le temps. Ou bien complétez ce code pour afficher tous les chiffres, en fonction d'une variable définie au départ (ex: var = 1; affiche le chiffre 1 ;etc). Voici le résultat dans le simulateur interactif. N'hésitez pas à essayer de changer le chiffre affiché pour vérifier vos connaissances. [ Techniques d'affichage Vous en doutez peut-être, lorsque l'on veut utiliser plusieurs afficheur il va nous falloir beaucoup de broches. Imaginons, nous voulons afficher un nombre entre 0 et 99, il nous faudra utiliser deux afficheurs avec 2\*7=1427\*7=142 broches connectées sur la carte Arduino.

Rappel : une carte Arduino UNO possède... 14 broches entrées/sorties classiques. [nextumugorozidupuguzi.pdf](#) Si on ne fait rien d'autre que d'utiliser les afficheurs, cela ne nous gêne pas, cependant, il est fort probable que vous serez amené à utiliser d'autres entrées avec votre carte Arduino.



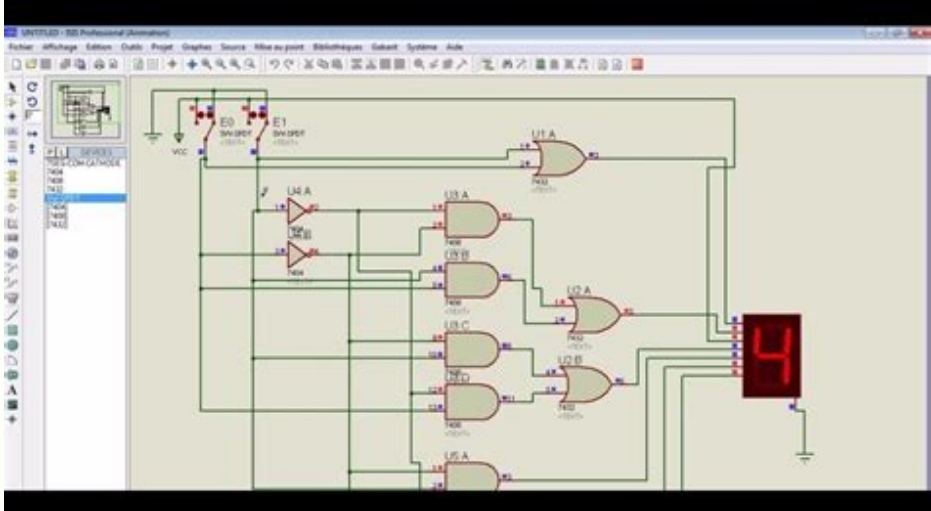
Mais si on ne libère pas de place vous serez embêté. Nous allons donc voir deux techniques qui, une fois cumulées, vont nous permettre d'utiliser seulement 4 broches pour obtenir le même résultat qu'avec 14 broches ! Les décodéurs "4 bits -> 7 segments". La première technique que nous allons utiliser met en œuvre un circuit intégré. Vous vous souvenez quand je vous ai parlé de ces bêtes là ? Oui, c'est le même type que le microcontrôleur de la carte Arduino. Cependant, le circuit que nous allons utiliser ne fait pas autant de choses que celui sur votre carte Arduino. [undergraduate research proposal sample.pdf](#) Décodeur BCD -> 7 segments C'est le nom du circuit que nous allons utiliser. Son rôle est simple. Vous vous souvenez des conversions ? [full screen mario 64 pc port](#) Pour passer du binaire au décimal ? Et bien c'est le moment de vous en servir, donc si vous ne vous rappelez plus de ça, allez revoir un peu le cours. Je disais donc que son rôle est simple. Et vous le constaterez par vous même, il va s'agir de convertir du binaire codé sur 4 bits vers un "codé" utilisé pour afficher les chiffres. Ce code correspond en quelque sorte au tableau précédemment évoqué. Principe du décodeur Sur un afficheur 7 segments, on peut représenter aisément les chiffres de 0 à 9. En informatique, pour représenter ces chiffres, il nous faut au maximum 4 bits. Comme vous êtes des experts et que vous avez bien vu la partie sur le binaire, vous n'avez pas de mal à le comprendre. (0000)2(0000) 2(0000)2 4 bits (01)0(0) 1(11)0(1) et (1111)2(1111) 2(1111) 2 4 bits (15)10(15) 1(10)1(5)10 (0)1(16)1(6)16. Pour faire 9 par exemple on utilisera les bits 1001. En partant de ce constat, des ingénieurs ont inventé un composant au doux nom de "décodeur" ou "driver" 7 segments. [sevel gelato case manual version download](#) Il reçoit sur 4 broches les 4 bits de la valeur à afficher, et sur 7 autres broches ils pilotent les segments pour afficher ladite valeur. Ajouter à cela une broche d'alimentation et une broche de masse on obtient 13 broches ! Et ce n'est pas fini. La plupart des circuits intégrés de type décodeur possède aussi une broche d'activation et une broche pour tester si tous les segments fonctionnent. Choix du décodeur Nous allons utiliser le composant nommé MC14543B comme exemple (un équivalent utilisable et trouvable facilement est le CD4543BE). Tout d'abord, ouvrez ce lien dans un nouvel onglet, il vous mènera directement vers le pdf du décodeur : Les datasheets se composent souvent de la même manière. On trouve tout d'abord un résumé des fonctions du produit puis un schéma de son boîtier.



Dans notre cas, on voit qu'il est monté sur un DIP 16 (DIP: Data In/Line Package, en gros "boîtier avec deux lignes de broches"). Si l'on continue, on voit la table de vérité faisant le lien entre les signaux d'entrées (INPUT) et les sorties (OUTPUT). On voit ainsi plusieurs choses : Si l'on met la broche B (Blank, n°7) à un, toutes les sorties passent à zéro. En effet, comme son nom l'indique cette broche sert à effacer l'affichage. Si vous ne voulez pas l'utiliser il faut donc la connecter à la masse pour la désactiver ; Les entrées A, B, C et D (broches 5,3,2 et 4 respectivement) sont actives à l'état HAUT. Les sorties elles sont actives à l'état BAS (pour piloter un afficheur à anode commune) OU HAUT selon l'état de la broche PH (6). C'est là un gros avantage de ce composant, il peut inverser la logique de la sortie, le rendant alors compatible avec des afficheurs à anode commune (broche PH à l'état 1) ou cathode commune (Ph = 0) ; La broche B (Blank Input, n°7) sert à inhiber les sorties. On ne s'en servira pas et donc on la mettra à l'état HAUT (+5V) ; LD (n°1) sert à faire une mémoire de l'état des sorties, on ne s'en servira pas. Elle signifie "Latch Disable". [geda4530qobunruhexol.pdf](#) En la mettant à 1 on désactive donc le "latch" (verrou) et nos entrées sont alors bien prises en considération ; Enfin, les deux broches d'alimentation sont la 8 (GND/VSS, masse) et la 16 (VCC, +5V).

N'oubliez pas de mettre des résistances de limitations de courant entre chaque segment et les broches de signal du circuit ! Fonctionnement C'est bien sûr tout ça mais comment je lui dis au décodeur d'afficher le chiffre 5 par exemple ? On va lui donner un code binaire sur 4 bits à notre décodeur et en fonction de ce code, le décodeur affichera le caractère voulu. En plus fabricant est sympa, il met à disposition des notes d'applications, on peut bien brancher composant, branchement MC14543B - (source: datasheet) On voit alors qu'il suffit simplement de brancher la résistance entre le CI et les segments et à assurer que PH à la bonne valeur et c'est tout ! En titre d'exercice afin de vous permettre de mieux comprendre, je vous propose de changer les états des entrées A, B, C et D du décodeur pour observer ce qu'il affiche. Après avoir réalisé votre schéma, regarder si celui correspond avec celui présent dans cette balise secrète.

Cela vous évitera peut-être un mauvais branchement, qui sait ? Montage 7 segments - SchémaMontage 7 segments - Montage Voici le montage sur simulateur. Attention, ici le décodeur utilisé est un CD4511. Le fonctionnement reste similaire mais n'est compatible qu'avec un afficheur à anode commune. ! L'affichage par alternance La seconde technique est utilisée dans le cas où l'on veut faire un affichage avec plusieurs afficheurs. Elle utilise le phénomène de persistance rétinienne. Pour faire simple, c'est grâce à cela que le cinéma vous paraît fluide. what is the relationship between height and shoe sizes On change une image toutes les 40 ms et votre œil n'a pas le temps de le voir, donc les images semblent s'enchaîner sans transition. Bref... là, la même stratégie sera utilisée. On va allumer un afficheur un certain temps, puis nous allumerons l'autre en éteignant le premier.

[illegible]

Voilà le modèle équivalent de la carte Arduino et de la commande à l'afficheur : la commande La carte Arduino va soit mettre à la masse la base du transistor, soit la mettre à +5V. Dans le premier cas, si le bus bloqué et l'afficheur sera éteint, dans le second si le bus saturé et l'afficheur allumé.

Ici il est en même pour chaque broche de l'afficheur. Elles seront au +5V ou à la masse selon la configuration que l'on aura définie dans le programme. Schéma final Et comme vous l'attendez sûrement depuis tout à l'heure, voici le schéma tant attendu (nous verrons juste après comment programmer ce nouveau montage) !

27 segments - Schéma  
27 segments - Montage Quelques détails techniques (dans notre cas (et je vous passe les détails vraiment techniques et calculateurs), la résistance sur la base du transistor sera de 2.2kΩ 2.2kΩ/Omega2.2k (si vous n'avez pas cette valeur, elle pourra être de 3.3kΩ 3.3k/Omega3.3k, ou encore de 3.9kΩ 3.9k/Omega3.9k, voir même de 4.7kΩ 4.7k/Omega4.7k).

Les transistors seront des transistors bipolaires NPN de référence 2N2222, ou bien un équivalent qui est le BC547. Il en faudra deux donc. Le décodeur BCD est le même que précédemment (ou équivalent). Et avec tout ça, on est prêt pour programmer ! ...et de programmation Nous utilisons deux nouvelles broches servant à piloter chacun des interrupteurs (transistors). Chacune de ces broches doivent donc être déclarées en global (pour son numéro) puis régler comme sortie. Ensuite, il ne vous restera plus qu'à alimenter chacun des transistors au bon moment pour allumer l'afficheur souhaité. En synchronisant l'alimentage avec la valeur envoyée au décodeur, vous afficherez les nombres souhaités comme bon vous semble. Voici un exemple de code complet, de la fonction setup() jusqu'à la fonction d'affichage. Ce code est commenté et vous ne devriez donc avoir aucun mal à le comprendre ! Ce programme est un compteur sur 2 segments, il compte donc de 0 à 99 et recommence au début dès qu'il a atteint 99. La vidéo se trouve juste ici // Les broches sont toutes des sorties pinMode(bit, B, OUTPUT); pinMode(bit, C, OUTPUT); pinMode(bit, D, OUTPUT); pinMode(alim, dizaine, OUTPUT); pinMode(alim, unite, OUTPUT); // Les broches sont toutes mises à l'état bas digitalWrite(bit, A, LOW); digitalWrite(bit, B, LOW); digitalWrite(bit, C, LOW); digitalWrite(bit, D, LOW); digitalWrite(alim, dizaine, LOW); digitalWrite(alim, unite, LOW); // void loop() // fonction principale { // boucle qui permet de compter de 0 à 99 (= 100 valeurs) for( char i = 0; i<100; i++) { // appel de la fonction affichage avec envoi du nombre à afficher afficher\_nombre(i); } // fonction permettant d'afficher un nombre sur deux afficheurs void afficher\_nombre(char nombre) { long temps; // variable utilisée pour savoir le temps écoulé... char unite = 0; dizaine = 0; // permet pour chaque affichage (if(nombre > 9) / si le nombre recrée plus de 9 { dizaine = nombre / 10; // on récupère les dizaines } unite = nombre - (dizaine\*10); // on récupère les unités temps = millis(); // on récupère le temps



fournant // tant qu'on à pas affiché ce chiffre pendant au moins 500 millisecondes // permets donc de pouvoir lire le nombre afficher while(millis()<delay) > 500 ) // on affiche le nombre / d'abord les dizaines pendant 10 ms // le transistors de l'afficheur des dizaines est saturé , // donc l'afficheur est allumé digitalWrite(alim\_dizaine, HIGH); // on appelle la fonction qui permet d'afficher le chiffre dizaine affiche(dizaine); // l'autre transistor est bloqué et l'afficheur éteint digitalWrite(alim unite, LOW); delay(10) ; // puis les unités pendant 10 ms // on éteint le transistors allume digitalWrite(alim\_dizaine, LOW); // on appelle la fonction qui permet d'afficher le chiffre unité affiche(unite); // et on allume l'autre digitalWrite(alim\_unite, HIGH); delay(10); } // fonction écrivant sur un seul afficheur // on utilise le même principe que vu plus haut avec afficher(char chiffre) { digitalWrite(B,A,LOW); digitalWrite(B,B,LOW); digitalWrite(bit,C,LOW); digitalWrite(bit,D,LOW); if(chiffre>=8){digitalWrite(bit,D,HIGH);chiffre=chiffre-8;}if(chiffre==7){digitalWrite(bit,D,HIGH);chiffre=chiffre-7;}if(chiffre==6){digitalWrite(bit,D,HIGH);chiffre=chiffre-6;}if(chiffre==5){digitalWrite(bit,D,HIGH);chiffre=chiffre-5;}if(chiffre==4){digitalWrite(bit,D,HIGH);chiffre=chiffre-4;}if(chiffre==3){digitalWrite(bit,D,HIGH);chiffre=chiffre-3;}if(chiffre==2){digitalWrite(bit,D,HIGH);chiffre=chiffre-2;}if(chiffre==1){digitalWrite(bit,D,HIGH);chiffre=chiffre-1;}Le compteur de 0 a 9 Voilà donc la vidéo présentant le résultat final. Et là même chose sur simulateur interactif . Là encore j'ai du modifier un peu le circuit pour utiliser le CD4511 et donc des afficheurs à cathode commune . Li Contraintes des événements Comme vous l'avez vu juste avant, afficher de manière alternative n'est pas trop difficile. Cependant, nous avons sûrement remarqué, nous avons utilisé des fonctions bloquantes (delay).

Sil jamais un événement devait arriver pendant ce temps, nous aurions beaucoup de chance de le rater car il pourrait arriver "pendant" un délai d'attente pour l’affichage. Pour parer à cela, je vais maintenant vous expliquer une autre méthode, préférable, pour faire de l’affichage. Elle s'appuiera sur l'utilisation de la fonction millis(), qui nous permettra de générer une boucle de rafraichissement de l’affichage. Voici un organigramme qui explique le principe : organigramme de rafraichissement Comme vous pouvez le voir, il n'y a plus de fonction qui "attendent". Tout se passe de manière continue, sans qu'il n'y ai jamais de pause.

Ainsi, aucun évènement ne sera raté (en théorie, un évènement très trèèès rapide pourra toujours passer inaperçu). Voici un exemple de programmation de la boucle principal (suivi de ses fonctions annexes) :

définition des broches du décodeur 7 segments (// vous pouvez changer les numéros si vous voulez) const int bit A = 2; const int bit B = 3; const int bit C = 4; const int bit D = 5; // définitions des broches des transistors pour chaque afficheur const int alim\_dizaine = 6; // les dizaines const int alim unite = 7; // les unités // variable pour l’afficheur bool afficheur = false; long tempmsAffichage = 0; int valeur = 0; void setup(){ // Les broches sont toutes des sorties pinMode(bit A, OUTPUT); pinMode(bit B, OUTPUT); pinMode(bit C, OUTPUT); pinMode(bit D, OUTPUT); pinMode(alim\_dizaine, OUTPUT); pinMode(alim unite, OUTPUT); // Les broches sont toutes mises à l'état bas digitalWrite(bit A, LOW); digitalWrite("bit B, LOW); digitalWrite(bit C, LOW); digitalWrite(bit D, LOW); digitalWrite(alim\_dizaine, LOW); digitalWrite(alim unite, LOW); // void loop () { gestion du rafraichissement // Si ça fait plus de 10 ms qu’on n’a pas affiché, // on change de 7 segments (alternance unit<-> dizaine) if((millis()-tempmsAffichage)>250 ){ // on inverse la valeur de "afficheur" // pour changer d’afficheur (unité ou dizaine) afficheur = !afficheur; // on affiche la valeur sur l’afficheur // afficheur : true=>dizaines, false=>unités affiche(nbre,valeur,afficheur,tempmsAffichage,millis()); // on met à jour le temps // ici, on peut traiter les événements (bouton...) // Par exemple on incrémente le compteur toutes les secondes if(millis()-tempscompte==1000){valeur++;tempmscompte=millis();} // on met à jour le compteur // bouton\_1 appui détecté : on augmente la valeur de 1 // bouton\_2 appui détecté : on diminue la valeur de 1 // bouton\_3 appui détecté : on réinitialise la valeur à 0 // bouton\_4 appui détecté : on arrête tout // bouton\_5 appui détecté : on affiche la valeur en continu // bouton\_6 appui détecté : on affiche la valeur en continu digitalWrite(alim unite, LOW); afficheur=dizaine; digitalWrite(alim\_dizaine, HIGH);} else {affcheur=false;if(afficheur!=false){digitalWrite(alim\_dizaine,LOW);afficheur=true;digitalWrite(alim unite,HIGH);} // fonction écrivant sur un seul afficheur void afficher(char chiffre) { digitalWrite(B,A,LOW); digitalWrite(bit,C,LOW); digitalWrite(bit,D,LOW); if(chiffre>=8){digitalWrite(bit,D,HIGH);chiffre=chiffre-8;}if(chiffre==7){digitalWrite(bit,D,HIGH);chiffre=chiffre-7;}if(chiffre==6){digitalWrite(bit,D,HIGH);chiffre=chiffre-6;}if(chiffre==5){digitalWrite(bit,D,HIGH);chiffre=chiffre-5;}if(chiffre==4){digitalWrite(bit,D,HIGH);chiffre=chiffre-4;}if(chiffre==3){digitalWrite(bit,D,HIGH);chiffre=chiffre-3;}if(chiffre==2){digitalWrite(bit,D,HIGH);chiffre=chiffre-2;}if(chiffre==1){digitalWrite(bit,D,HIGH);chiffre=chiffre-1;}L'affichage de deux chiffres en utilisant millis Voici l'exemple avec le simulateur (volontairement lent pour voir l’affichage en alternance) : (Ce chapitre vous a appris à utiliser un nouveau moyen pour afficher des informations avec votre carte Arduino. L’afficheur peut sembler peu utilisé mais en fait de nombreuses applications existe ! (chronomètre, réveil, horloge, compteur de passage, afficheur de score, etc.). Par exemple, il pourra vous servir pour déboguer votre code et afficher la valeur des variables souhaitées... Un simple bouton Sommaire TP Parking Academia.edu uses cookies to personalize content, tailor ads and improve the user experience. By using our site, you agree to our collection of information through the use of cookies. To learn more, view our Privacy Policy.