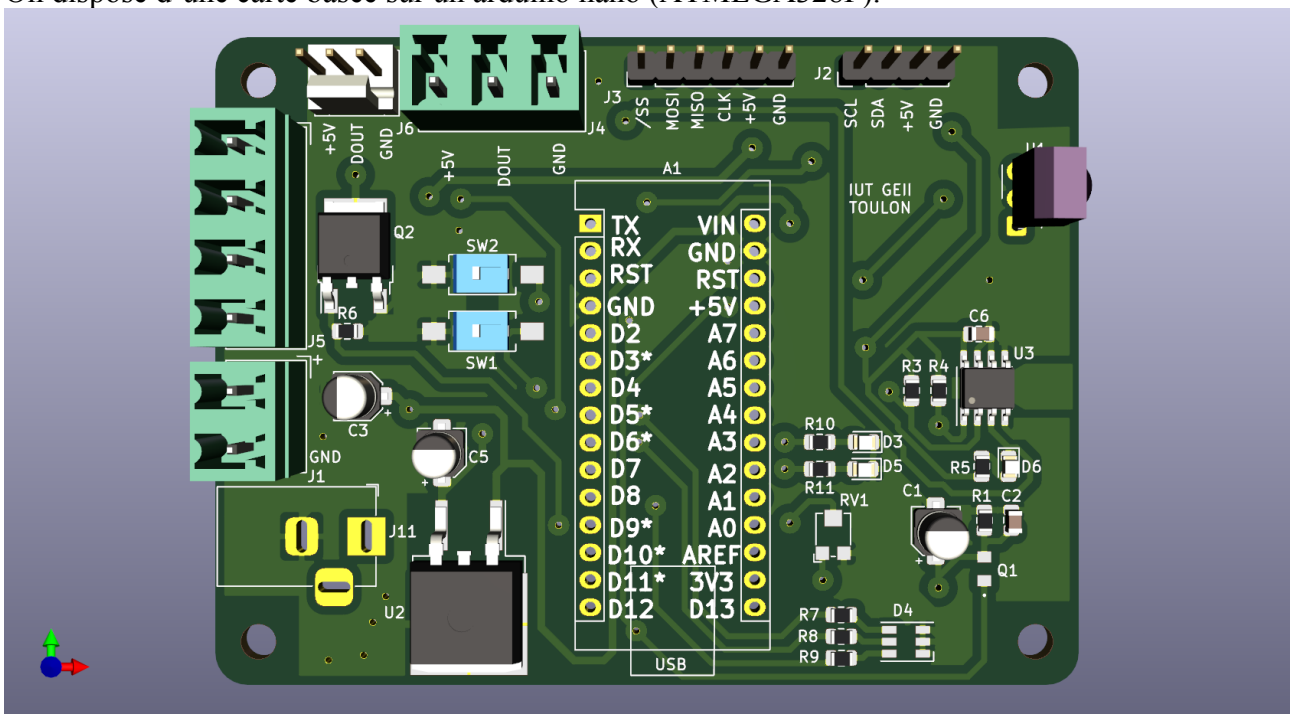
Mesures à l'oscilloscope

L'objectif est de savoir utiliser l'oscilloscope pour faire des mesures sur des cartes à microcontrôleur :

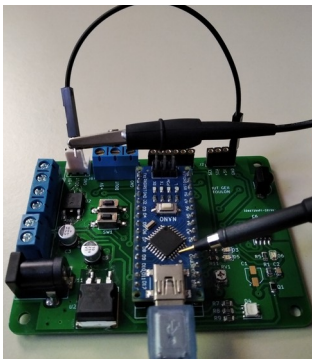
- mesures de tensions continues
- mesures de tensions et de durées sur des signaux périodique
- mesures de perturbations sur des tensions continues
- captures d'évènements non périodiques.

On dispose d'une carte basée sur un arduino nano (ATMEGA328P).



Le schéma est donné en annexe.

Toutes les mesures se font avec une sonde. Attention à ne pas faire de court-circuits. Connectez la pince crocodile de la sonde à la masse de la carte.



Pour chaque mesure demandée, prendre un copie d'écran de l'oscilloscope et la mettre dans votre compte rendu.

1. Réglages préliminaires

Avant toute mesure il faut vérifier les réglages de l'oscilloscope et de la sonde sur un signal connu. La plupart des oscilloscopes fournissent un signal carré 0/5V à 1kHz (PROBE COMP sur Tektronix).
Utilisez ce signal pour vérifier les réglages de la voie (atténuation, position du zéro, calibre,...). Si nécessaire réglez la compensation de la sonde pour obtenir un carré parfait[1].

2. Mesures de tensions continues

Alimentez la carte en USB.
Utilisez l'affichage "valeur moyenne".
Mesurez la tension 5V la tension 3.3V et la tension aux bornes de la LED PWR.
Régler le potentiomètre pour avoir 1.5V sur A0.

3. Mesures sur des signaux périodique.

Modifier le programme "**blink**" pour utiliser la led D5 de la carte et pour obtenir une durée d'éclairement de 100ms.
Mesurer la tension aux bornes de la Led.

Charger le programme "**pwm**" et relever les signaux de commande de la led RGB.
- Affichez les fréquences et les rapports cycliques
- Utilisez les "curseurs" pour mesurer les temps haut et bas et recalculez les fréquences et les rapports cycliques. Comparer aux valeurs théoriques.
- Mesurer le temps à l'état haut de la Led verte avec le maximum de résolution possible.

Charger le programme "**serial**"
- Mesurer avec la meilleure résolution possible la durée d'un bit.
- Comparer à la valeur théorique (9600 bit/s).
- Pourquoi avoir choisi le caractère ASCII `U`.
- Modifier le programme pour transmettre le caractère A toutes les 10ms.
- Sachant qu'il s'agit d'une transmission asynchrone 9600 bit/s 8 bits sans parité, retrouver le sens (LSB ou MSB first ?) et les bits du code ASCII de A.

4. Mesures de durées d'exécution

En utilisant une sortie logique, on peut mesurer la durée d'exécution d'une partie d'un programme : il suffit de faire changer la valeur de la sortie au début et à la fin de la séquence à mesurer.
Charger le programme "**execTime**".
Le calcul est mis en commentaire pour pouvoir mesurer la durée de la fonction digitalWrite().

- Mesurer la durée de la fonction digitalWrite() avec la meilleure résolution possible.

1 Le réglage dépend de la fréquence. Si on s'écarte notablement de 1kHZ, il faut faire le réglage avec un signal carré proche de la fréquence du signal à mesurer.

Il faudra maintenant retrancher cette valeur des mesures suivantes.

- Enlever le commentaire sur le calcul et mesurer la durée du calcul **v=4*t+5 ;**.

- Mesurer les durées des calculs suivant :

v=4*t+5.0 ;
v=4.0*t+5.0 ;
v=4.0*t+5 ;

- Conclusion.

Les variables sont déclarées ***volatile***[2] pour empêcher le compilateur d'optimiser le programme. Effet le compilateur s'aperçoit que ***t*** est constante et que ***v*** n'est pas utilisée par la suite. Il ne fait donc pas le calcul. Enlevez volatile pour voir l'effet de l'optimisation.

5. Mesures de perturbations sur des tensions continues

La consommation de courant par le microcontrôleur et par les périphériques connectés sur la carte (LED, transistors,..) perturbent la tension d'alimentation 5V. Une perturbation trop importante peut entraîner un dysfonctionnement du microcontrôleur (RESET intempestif).

Pour mesurer une faible variation sur une tension continue en utilise le mode de couplage AC de la voie d'entrée. Dans ce mode la composante continue est supprimée pour pouvoir "zoomer" sur les variations autour de la valeur moyenne.

- Charger le programme "**sleep"** et lire les messages sur le terminal. Le micro est en mode basse consommation (sleep). Un appui sur SW2 le réveille et alimente la led rouge en PWM. La consommation de la led perturbe (faiblement) la tension continue d'alimentation 5V.

- Relever les variations de la tension 5V (zoom en mode AC) lorsque le micro est endormi. Mesurer la valeur crête à crête.

-Lorsque la led est alimentée, relever les variations crête à crête de tension d'alimentation 5V en concordance avec le signal PWM commandant la led.

- Modifier le programme pour commander les deux autres couleurs[3] et refaire la mesure.

-Conclusion.

2 Le qualificatif *volatile* indique au compilateur que la valeur peut être modifiée par un moyen extérieur (sous programme d'interruption, thread, registre d'entrée/sortie,…) et donc aucune optimisation n'est appliquée.
3 Les PWM des trois couleurs ne sont pas toutes à la même fréquence. Cela complique un peu la forme observée.

6. Captures d'éventements fugitifs

- Charger le programme "**aleaCar"** qui émet à 115200 bits/seconde un caractère aléatoire au bout d'un temps aléatoire.

- Capturer le signal d'un caractère et décoder le.

- Charger le programme "**tempI2C"** qui lit périodiquement le circuit TMP75 sur le bus I2C. On retrouve les signaux I2C SCL et SDA sur le connecteur J2.

- Mesurer la fréquence de l'horloge SCL.

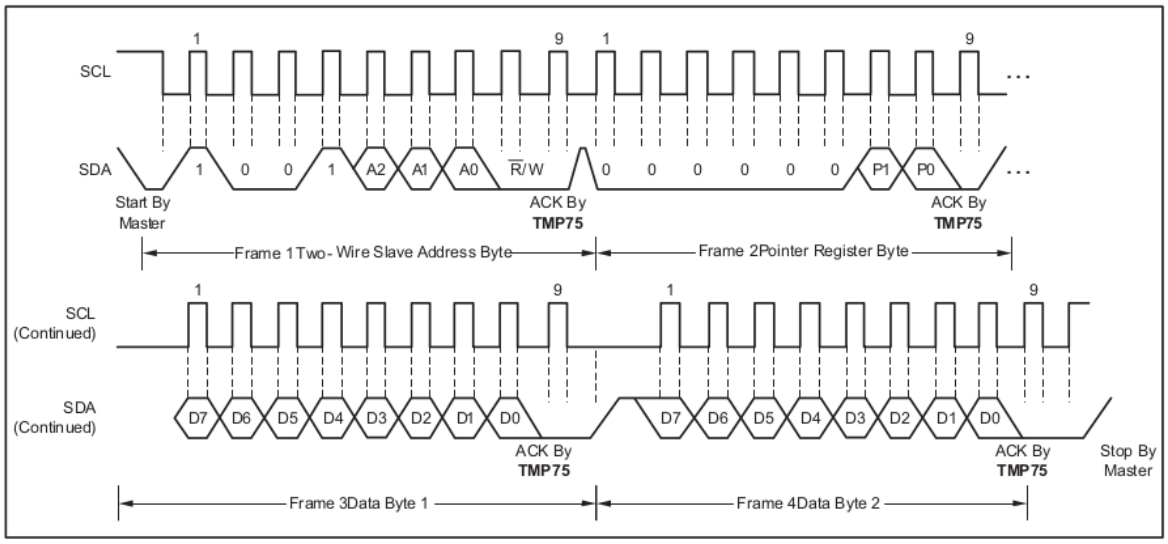


Figure 7. Two-Wire Timing Diagram for the TMP75 Write Word Format

- Afficher le début de l'échange jusqu'au bit ACK (voie SCL, voie SDA).

- Retrouver le bit de *start* et la valeur de l'adresse du circuit.