

Rapport de laboratoire

Ecole supérieure
Électronique

Laboratoire MINF
Salle R110

TP1 PWM A/D

Réalisé par :

Etienne De Oliveira

A l'attention de :

Serge Castoldi
Philippe Bovey

Dates :

Début du laboratoire : 28 novembre 2024
Fin du laboratoire : 9 janvier 2025

Table des matières :

TP1 PWM A/D.....	1
1 Cahier des charges.....	5
2 Réglage Timer et OC	5
2.1 Timer.....	5
2.1.1 Timer 1	5
2.1.2 Timer 2	5
2.1.3 Timer 3	6
2.1.4 Timer 4	6
2.2 OC.....	7
2.2.1 OC 2	7
2.2.2 OC 3	8
2.3 Schéma de mesure	8
2.3.1 Méthode de mesure	8
2.4 Résultats.....	9
2.5 Conversion AD→PWM.....	9
2.5.1 Code pour PWM 1 du moteur DC.....	9
2.5.1.1 Appliqué sur OC2	9
2.5.2 Code pour PWM 2 servomoteur.....	10
2.5.2.1 Appliqué sur OC3	10
3 Code	10
4 Niveau priorité	11
4.1 Explication mesure	11
4.2 Schéma de mesure	11
4.2.1 Méthode de mesure	11
4.3 Niveau 4 et 7.....	11
4.4 Niveau 4.....	12
5 PWM OC2 et Soft.....	12
5.1 Explication mesure	12
5.2 Schéma de mesure PWM OC2	12
5.2.1 Méthode de mesure :	13
5.3 Schéma de mesure PWM Soft	13
5.3.1 Méthode de mesure	13
5.4 Résultats.....	13
5.4.1 PWM OC2.....	13
5.4.1.1 Analyse.....	14
5.4.2 PWM Soft.....	14
5.4.2.1 Analyse.....	14
5.4.3 Passage PWM à 0	14
5.4.3.1 Analyse.....	14
6 Servomoteur	15
6.1 Schéma de mesure	15
6.1.1 Méthode de mesure :	15
6.2 Résultats.....	15
6.3 Analyse	15
7 Inversion sens moteur	16
8 Conclusion	17
9 Annexes.....	18

9.1 Cahier des charges	18
9.2 Liste du matériel	19
9.3 Mesure Timers	19
9.4 PWM OC2	20
9.5 PWM Soft	22
9.6 Passage PWM à 0	24
9.7 Servomoteur.....	25
9.8 Extrait cours ch.5 p.9	27
9.9 Feuille de contrôle	27

1 Cahier des charges

Voir en annexe 9.1.

2 Réglage Timer et OC

2.1 Timer

2.1.1 Timer 1

Période demandée de 20ms.

J'ai réglé le prescaler à 256.

Interruption de niveau 4. (Demandé CDC voir 9.1)

Formule pour Timer period :

$$TimerPeriod = \left(\frac{t \cdot f_{\mu C}}{prescaler} \right) - 1 = \left(\frac{20 \cdot 10^{-3} \cdot 80 \cdot 10^6}{256} \right) - 1 = 6'249$$

Réglage dans Harmony :

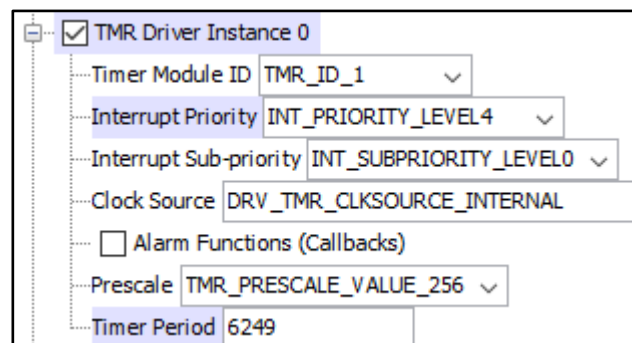


Figure 1 Réglage Harmony Timer1

2.1.2 Timer 2

Fréquence demandée de 40 kHz. Période, $t = \frac{1}{40 \cdot 10^3} = 25\mu s$

J'ai réglé le prescaler à 1.

Interruption de niveau 1. (Suivi exemple p. 2 et 3, chapitre 5 MINF TP)

Formule pour Timer period :

$$TimerPeriod = \left(\frac{t \cdot f_{\mu C}}{prescaler} \right) - 1 = \left(\frac{25 \cdot 10^{-6} \cdot 80 \cdot 10^6}{1} \right) - 1 = 1'999$$

Réglage dans Harmony :

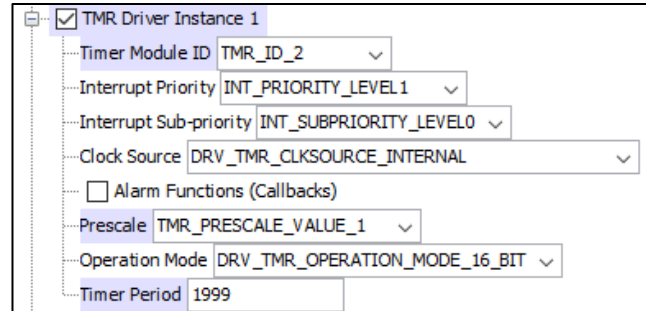


Figure 2 Réglage Harmony Timer12

2.1.3 Timer 3

Période demandée de 7ms.

J'ai réglé le prescaler à 64.

Interruption de niveau 1. (Suivi exemple p. 2 et 3, chapitre 5 MINF TP)

Formule pour Timer period :

$$TimerPeriod = \left(\frac{t \cdot f_{\mu c}}{prescaler} \right) - 1 = \left(\frac{7 \cdot 10^{-3} \cdot 80 \cdot 10^6}{64} \right) - 1 = 8'749$$

Réglage dans Harmony :

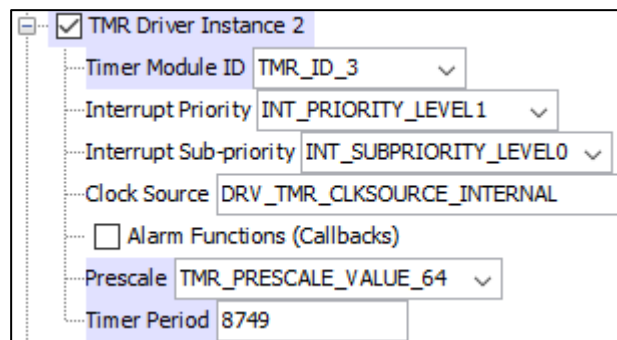


Figure 3 Réglage Harmony Timer3

2.1.4 Timer 4

Période demandée de 35μs.

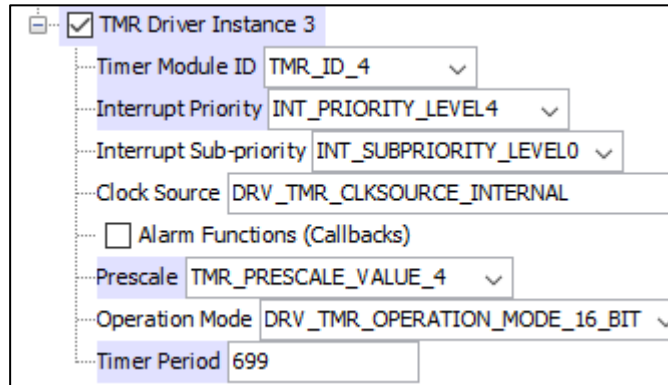
J'ai réglé le prescaler à 4.

Interruption de niveau 4. (Demandé CDC voir 9.1)

Formule pour Timer period :

$$TimerPeriod = \left(\frac{t \cdot f_{\mu c}}{prescaler} \right) - 1 = \left(\frac{35 \cdot 10^{-6} \cdot 80 \cdot 10^6}{4} \right) - 1 = 699$$

Réglage dans Harmony :



Configuration for TMR Driver Instance 3:

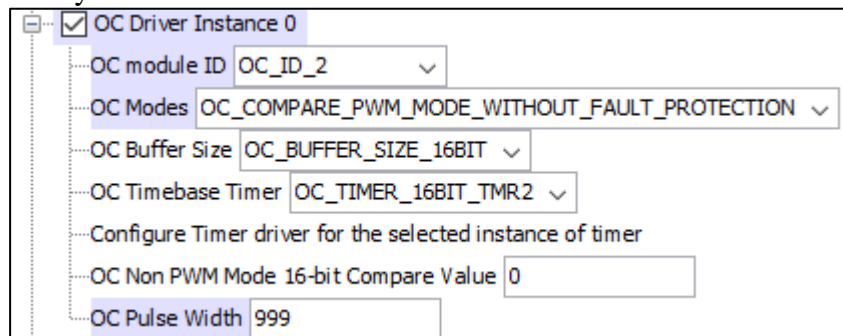
- ☒ TMR Driver Instance 3
- Timer Module ID: TMR_ID_4
- Interrupt Priority: INT_PRIORITY_LEVEL4
- Interrupt Sub-priority: INT_SUBPRIORITY_LEVEL0
- Clock Source: DRV_TMR_CLKSOURCE_INTERNAL
- ☐ Alarm Functions (Callbacks)
- Prescale: TMR_PRESCALE_VALUE_4
- Operation Mode: DRV_TMR_OPERATION_MODE_16_BIT
- Timer Period: 699

Figure 4 Réglage Harmony Timer4

2.2 OC

2.2.1 OC 2

Réglage dans Harmony :



Configuration for OC Driver Instance 0:

- ☒ OC Driver Instance 0
- OC module ID: OC_ID_2
- OC Modes: OC_COMPARE_PWM_MODE_WITHOUT_FAULT_PROTECTION
- OC Buffer Size: OC_BUFFER_SIZE_16BIT
- OC Timebase Timer: OC_TIMER_16BIT_TMR2
- ☒ Configure Timer driver for the selected instance of timer
- OC Non PWM Mode 16-bit Compare Value: 0
- OC Pulse Width: 999

Figure 5 Réglage Harmony OC2

Pour faire ces réglages, je me suis basé sur p.4 chapitre 5 MINF TP.

J'ai décidé de mettre au départ le Pulse Width à 50% ce qui correspond à 999. Car la période du Timer 2 est de 1'999. Cette valeur n'est pas très importante, parce que c'est la valeur d'initialisation. Elle va changer ensuite grâce au programme.

2.2.2 OC 3

Réglage dans Harmony :

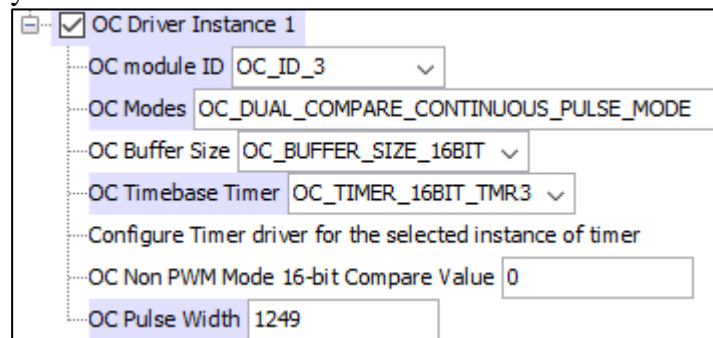


Figure 6 Réglage Harmony OC3

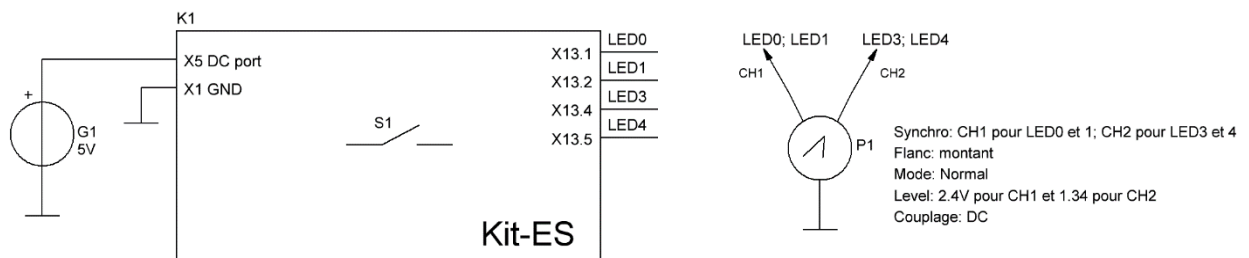
Pour faire ces réglages, je me suis basé sur p.4 chapitre 5 MINF TP.

J'ai décidé de mettre au départ le Pulse Width à 1'249 ce qui correspond à une impulsion de 1ms. Car la période du Timer 3 est de 7ms, qui correspond à 8'749.

$$PulseWidth = \frac{TimerPeriod}{t_{Timer3}} = \frac{8749}{7} = 1'249$$

Cette valeur n'est pas très importante, parce que c'est la valeur d'initialisation. Elle va changer ensuite grâce au programme.

2.3 Schéma de mesure



2.3.1 Méthode de mesure

1. Alimenter la carte
2. Programmer la carte si cela n'est pas fait.
3. Brancher les sondes de l'oscilloscope, d'après le schéma.
4. Régler l'oscilloscope comme sur le schéma.
5. Appuyer sur le bouton reset S1
6. Relever l'oscillogramme.

La liste du matériel se trouve en annexe 9.2.

2.4 Résultats

Timer n°	Période théorique	Période mesurée
1	20ms	20ms
2	25µs	25µs
3	7ms	7ms
4	35µs	35µs

Je peux constater qu'il n'y a pas de différence entre la valeur théorique et mesurée.
Les screens des mesures se trouve en annexe 9.3.

2.5 Conversion AD→PWM

2.5.1 Code pour PWM 1 du moteur DC

```

64 // Lecture des valeurs du convertisseur analogique-numérique (ADC)
65 appData.AdcRes = BSP_ReadAllADC(); // Récupération des valeurs ADC
66 lastValue0 = appData.AdcRes.Chan0; // Stockage de la dernière valeur du canal 0
67 sum0 -= adcValues0[index0]; // Soustraction de la valeur précédente du tableau
68 adcValues0[index0] = lastValue0; // Stockage de la nouvelle valeur dans le tableau
69 sum0 += adcValues0[index0]; // Ajout de la nouvelle valeur à la somme
70 index0 = (index0 + UN) % DIX; // Mise à jour de l'index pour le tableau
71 uint16_t averageValue0 = sum0 / DIX; // Calcul de la moyenne des valeurs du canal 0
72
73 // Mise à l'échelle de la valeur pour obtenir une plage de 0 à 198
74 uint16_t scaledValue0 = (averageValue0 * ORDONEEPRG) / MAXVALAD;
75
76 // Calcul de la vitesse signée variant de -99 à 99
77 PWMData.SpeedSetting = scaledValue0 - OFFSETORIG;
78
79 // Calcul de la vitesse absolue variant de 0 à +99
80 if (scaledValue0 >= OFFSETORIG)
81 {
82     PWMData.absSpeed = PWMData.SpeedSetting; // Vitesse positive
83 }
84 else
85 {
86     PWMData.absSpeed = abs(PWMData.SpeedSetting); // Vitesse négative
87 }

```

Figure 7 Extrait code lecture AD

ORDONEEPRG correspond à 198 plage maximum, MAXVALAD à 1023 et OFFSETORIG à 99.

Les commentaires sont assez explicites pour comprendre le raisonnement. J'ai créé un tableau buffer de 10 cases, pour stocker les 10 dernière valeur de l'AD.

A la ligne 70 le %10, me sert pour revenir à la première case une fois arrivé au bout du tableau.

2.5.1.1 Appliqué sur OC2

```

// Calcul de la valeur du nombre d'impulsions pour OC2
PLIB_OC_PulseWidth16BitSet(OC_ID_2, ((PWMData.absSpeed * MAXVALAD) / OFFSETORIG)*DEUX);

```

Figure 8 Calcul OC2

D'abord je prends ma valeur absolue et je la multiplie par 1023 pour mettre à l'échelle la vitesse pour qu'elle corresponde à la plage de valeurs que le PWM peut accepter.

Ensuite de divise par 99 pour ajuster la valeur mise à l'échelle pour qu'elle soit proportionnelle à la plage de vitesses.

Pour finir fois 2 car sinon j'obtiens la moitié PWM.

2.5.2 Code pour PWM 2 servomoteur

```
89 // Traitement pour le canal 1
90 lastValue1 = appData.AdcRes.Chan1; // Stockage de la dernière valeur du canal 1
91 sum1 -= adcValues1[index1]; // Soustraction de la valeur précédente du tableau
92 adcValues1[index1] = lastValue1; // Stockage de la nouvelle valeur dans le tableau
93 sum1 += adcValues1[index1]; // Ajout de la nouvelle valeur à la somme
94 index1 = (index1 + UN) % DIX; // Mise à jour de l'index pour le tableau
95 uint16_t averageValue1 = sum1 / DIX; // Calcul de la moyenne des valeurs du canal 1
96
97 // Calcul de l'angle absolu variant de 0 à 180
98 PWMData.absAngle = (averageValue1 * ANGLE_ABS) / MAXVALAD;
99
100 // Calcul de l'angle signé variant de -90 à 90
101 PWMData.AngleSetting = PWMData.absAngle - MAXANGLE;
102 }
```

Figure 9 Extrait code lecture AD

ANGLE_ABS correspond à 180 plage maximum, MAXANGLE à 90 et MAXVALAD toujours à 1023.

Les commentaires sont assez explicites pour comprendre le raisonnement. J'ai créé un tableau buffer de 10 cases, pour stocker les 10 dernière valeur de l'AD.

2.5.2.1 Appliqué sur OC3

```
// Calcul de la valeur du nombre d'impulsions pour OC3
PLIB_OC_PulseWidth16BitSet(OC_ID_3, ((PWMData.absAngle + MAXANGLE) * MAXVALAD / ANGLE_ABS)*DEUX);
```

Figure 10 Calcul OC3

D'abord je prends ma valeur absolue et je rajoute 90. Ensuite je la multiplie par 1023 pour mettre à l'échelle l'angle pour qu'elle corresponde à la plage de valeurs que le PWM peut accepter.

Ensuite de divise par 180 pour ajuster la valeur mise à l'échelle pour qu'elle soit proportionnelle à la plage d'angles.

Pour finir fois 2 car sinon j'obtiens la moitié du PWM.

3 Code

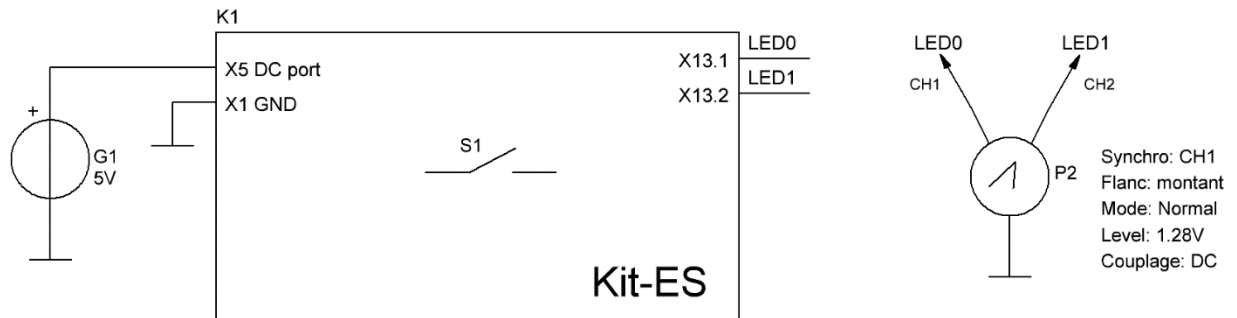
Le code complet se trouve sur Git.

4 Niveau priorité

4.1 Explication mesure

Cette mesure a pour but d'observer l'effet du niveau de priorité des Timers.

4.2 Schéma de mesure



4.2.1 Méthode de mesure

1. Alimenter la carte
2. Programmer la carte si cela n'est pas fait.
3. Brancher les sondes de l'oscilloscope, d'après le schéma.
4. Régler l'oscilloscope comme sur le schéma.
5. Appuyer sur le bouton reset S1
6. Relever l'oscillogramme.
7. Refaire les points 2 à 6 en changeant le niveau de priorité

4.3 Niveau 4 et 7

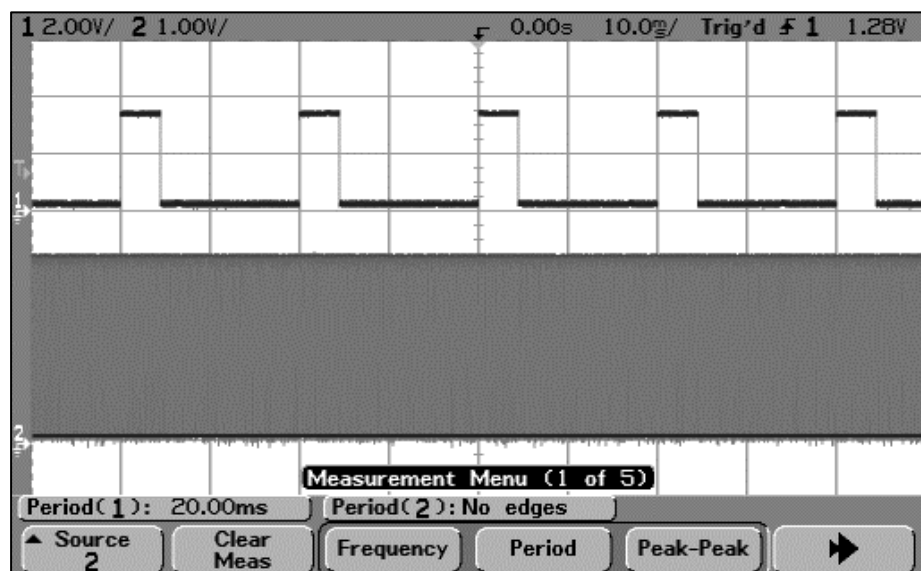


Figure 11 Timer 1 et 4

Lorsque le niveau de priorité est différent, le Timer 1 n'a pas d'influence sur le Timer 4. Lorsque qu'une interruption survient sur le Timer 1.

La période du Timer 1 est bien de 20ms.

4.4 Niveau 4

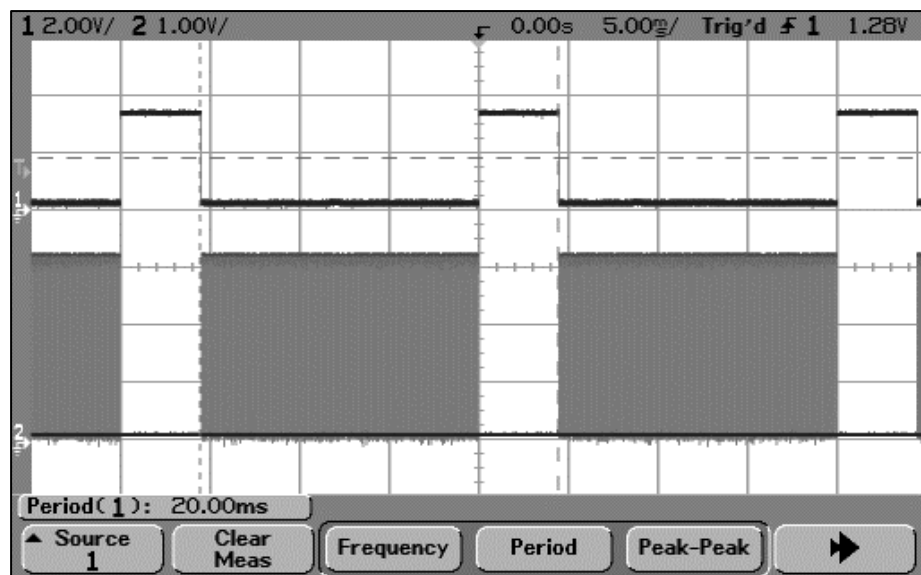


Figure 12 Timer 1 et 4 même niveau

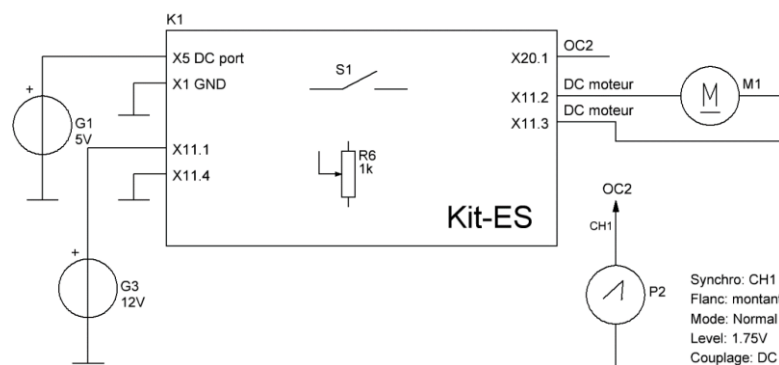
Lorsque les 2 Timers ont le même niveau de priorité. Le Timer 4 se fait interrompre durant l'interruption du Timer 1. C'est normal car même avec les mêmes niveaux le Timer 1 reste prioritaire sur le reste. (Voir annexe 9.8 extrait ch.5 cours p.9)

5 PWM OC2 et Soft

5.1 Explication mesure

Démontrer le fonctionnement des signaux PWM pour la Led et l'OC2.

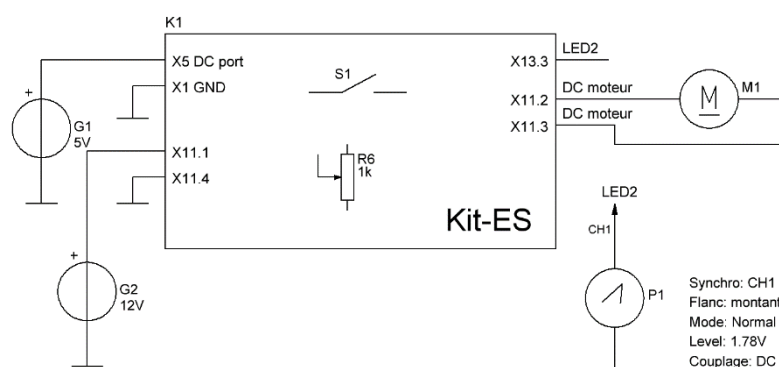
5.2 Schéma de mesure PWM OC2



5.2.1 Méthode de mesure :

1. Alimenter la carte
2. Programmer la carte si cela n'est pas fait.
3. Brancher les sondes de l'oscilloscope, d'après le schéma.
4. Régler l'oscilloscope comme sur le schéma.
5. Appuyer sur le bouton reset S1
6. Ensuite faite tourner le pot. R6 pour changer le rapport cyclique.
7. Relever l'oscillogramme.

5.3 Schéma de mesure PWM Soft



5.3.1 Méthode de mesure

1. Alimenter la carte
2. Programmer la carte si cela n'est pas fait.
3. Brancher les sondes de l'oscilloscope, d'après le schéma.
4. Régler l'oscilloscope comme sur le schéma.
5. Appuyer sur le bouton reset S1
6. Ensuite faite tourner le pot. R6 pour changer le rapport cyclique.
7. Relever l'oscillogramme.

5.4 Résultats

5.4.1 PWM OC2

Taux PWM	Durée high théorique [μs]	Durée high mesurée [μs]
5%	1.25	1.25
-5%	1.25	1.25
50%	12.5	12.95
-50%	12.5	12.95
95%	23.75	24.55
-95%	23.75	24.55

Calcul pour valeur théorique :

$$t = 25 \mu s$$

$$t_{on} = t \cdot Taux\%$$

5.4.1.1 Analyse

Je peux constater que mes valeurs sont assez proches de la théorie. Pour 5 et 50% les valeurs sont quasiment les mêmes. Pour 95 %, il y a un peu plus de 1µs de décalage. La principale cause peut être l'oscilloscope. J'ai fait cette mesure durant les vacances avec mon oscilloscope. Cet oscilloscope date 2001 et la dernière fois qui l'a été calibré fut en 2006 lorsqu'il appartenait encore à l'ETML. Cette valeur reste tout de même acceptable c'est moins de 5% d'erreur. Entre les valeurs positive et négative du PWM, il n'y a pas de différence. C'est normal car un taux ne peut pas être négatif. C'est uniquement avec les entrées du pont en H que l'on va pouvoir faire tourner le moteur dans l'autre sens.

Les screens des mesures se trouvent en annexe 9.4.

5.4.2 PWM Soft

Taux PWM	Durée high théorique [ms]	Durée high mesurée [ms]
5%	0.175	0.2099
-5%	0.175	0.2099
50%	1.75	1.785
-50%	1.75	1.785
95%	3.325	3.36
-95%	3.325	3.36

Calcul pour valeur théorique :

$$t = 3.5 \text{ ms}$$

$$t_{on} = t \cdot \text{Taux}\%$$

5.4.2.1 Analyse

Je peux constater que mes valeurs sont assez proches de la théorie. Pour 50 et 95% les valeurs sont quasiment les mêmes. Pour 5 %, il y a un peu plus de 30µs de décalage. La principale cause est du taux PWM sur l'affichage il était de 5% mais la valeur réelle est de 5,94% → 6 %. Cela correspond à un t_{on} de 210µs ce que j'obtiens.

Les screens des mesures se trouvent en annexe 9.5.

5.4.3 Passage PWM à 0

Taux PWM [0%]	Tension théorique [V]	Tension mesurée [V]
OC2	0	160m
Led	3.3	3.31

5.4.3.1 Analyse

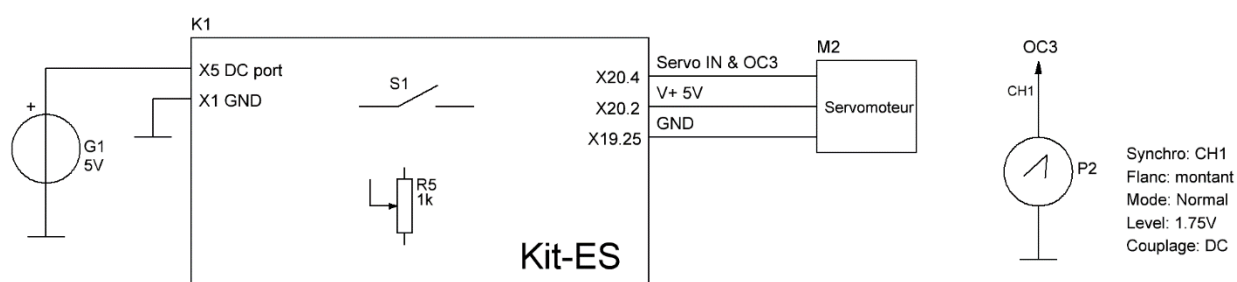
Pour l'OC2, je devrais avoir 0, c'est normal car un taux à 0 signifie que le signal est toujours bas. Avec la mesure j'obtiens 160mV, cette différence provient de mon oscilloscope. (Voir explication point 5.4.1.1.)

En ce qui concerne la Led je devrais avoir Vcc (3V3) car ce sont des Leds actives bas. C'est correct car j'ai 3.31 V. si les Leds auraient été actives hautes, j'aurais obtenu 0V.

Les screens sont en annexe 9.6.

6 Servomoteur

6.1 Schéma de mesure



6.1.1 Méthode de mesure :

1. Alimenter la carte
2. Programmer la carte si cela n'est pas fait.
3. Brancher les sondes de l'oscilloscope, d'après le schéma.
4. Régler l'oscilloscope comme sur le schéma.
5. Ensuite faite tourner le pot. R5 pour changer le rapport cyclique.
6. Relever l'oscillogramme.

6.2 Résultats

Angle [°]	Durée high théorique [ms]	Durée high mesurée [ms]
90	2.4	2.44
0	1.5	1.62
-90	0.6	0.8

Calcul pour valeur théorique pour 0 :

$$t = \frac{(t_{\max} + t_{\min})}{2} = \frac{(2.4m + 0.6m)}{2} = 1.5ms$$

6.3 Analyse

Mes valeurs mesurées sont assez proches des valeurs théoriques. La valeur à 90° est parfaite mais les 2 autres il y a un léger décalage. Ces différences peuvent provenir de mon oscilloscope. (Voir explication point Analyse.) mais dans l'ensemble il n'y a rien d'alarmant.

Les screens sont en annexe Servomoteur.

7 Inversion sens moteur

Pour pouvoir réaliser le code, je me suis basé sur le tableau du datasheet du pont en H. Disponible dans le cahier des charges.

Input				Output		
IN1	IN2	PWM	STBY	OUT1	OUT2	Mode
H	H	H/L	H	L	L	Short brake
L	H	H	H	L	H	CCW
		L	H	L	L	Short brake
H	L	H	H	H	L	CW
		L	H	L	L	Short brake
L	L	H	H	OFF (High impedance)		Stop
H/L	H/L	H/L	L	OFF (High impedance)		Standby

Figure 13 Extrait datasheet Hbridge

Pour aller dans le sens anti-horaire, c'est CCW. Et pour le sens horaire, c'est CW. Ensuite si on veut arrêter le moteur il faut regarder stop.

Dans le code voici ce que ça donne :

```

118 // Execution PWM et gestion moteur # partir des info dans structure
119 void GPWM_ExecPWM(S_pwmSettings *pData)
120 {
121     STBY_HBRIDGE_W = UN;
122     // Gestion de la direction du pont en H
123     if (PWMDData.SpeedSetting > ZERO)
124     {
125         AIN1_HBRIDGE_W = UN;
126         AIN2_HBRIDGE_W = ZERO;
127         // Avancer
128     }
129     else if (PWMDData.SpeedSetting < ZERO)
130     {
131         AIN1_HBRIDGE_W = ZERO;
132         AIN2_HBRIDGE_W = UN;
133         // Reculer
134     }
135     else
136     {
137         AIN1_HBRIDGE_W, AIN2_HBRIDGE_W = ZERO;
138         // Arrêter
139     }

```

Figure 14 Extrait code gestion pont en H

Je fais juste des tests pour savoir si la valeur est > ou < que 0. Sinon le moteur s'arrête.

8 Conclusion

La feuille de contrôle du fonctionnement se trouve en annexe 9.9.

Pour conclure ce laboratoire, j'ai pu approfondir ma compréhension des concepts de PWM, de conversion A/D et de gestion des interruptions dans un microcontrôleur. Le travail pratique m'a permis de réaliser des réglages fins sur les timers, et de constater que les périodes mesurées correspondaient parfaitement aux valeurs théoriques, ce qui m'a assuré que la mise en place des paramètres était correcte. La conversion des signaux analogiques en PWM, notamment pour le moteur à courant continu et le servomoteur, a été un exercice très intéressant, car il m'a permis de mieux saisir l'importance de l'échelle des valeurs et de l'ajustement du rapport cyclique.

Le point sur les priorités des interruptions ont également été un point intéressant, et j'ai pu observer l'impact des différents niveaux de priorité sur l'exécution des timers. Les tests sur le signal PWM, tant en mode matériel qu'en mode logiciel, ont montré des résultats très proches de la théorie, ce qui m'a conforté dans l'idée que mes réglages étaient bien réalisés. Cependant, certains petits écarts, comme ceux observés lors des mesures avec l'oscilloscope, m'ont rappelé que l'équipement peut parfois introduire de petites erreurs.

Lausanne, 15 janvier 2025

Signature :



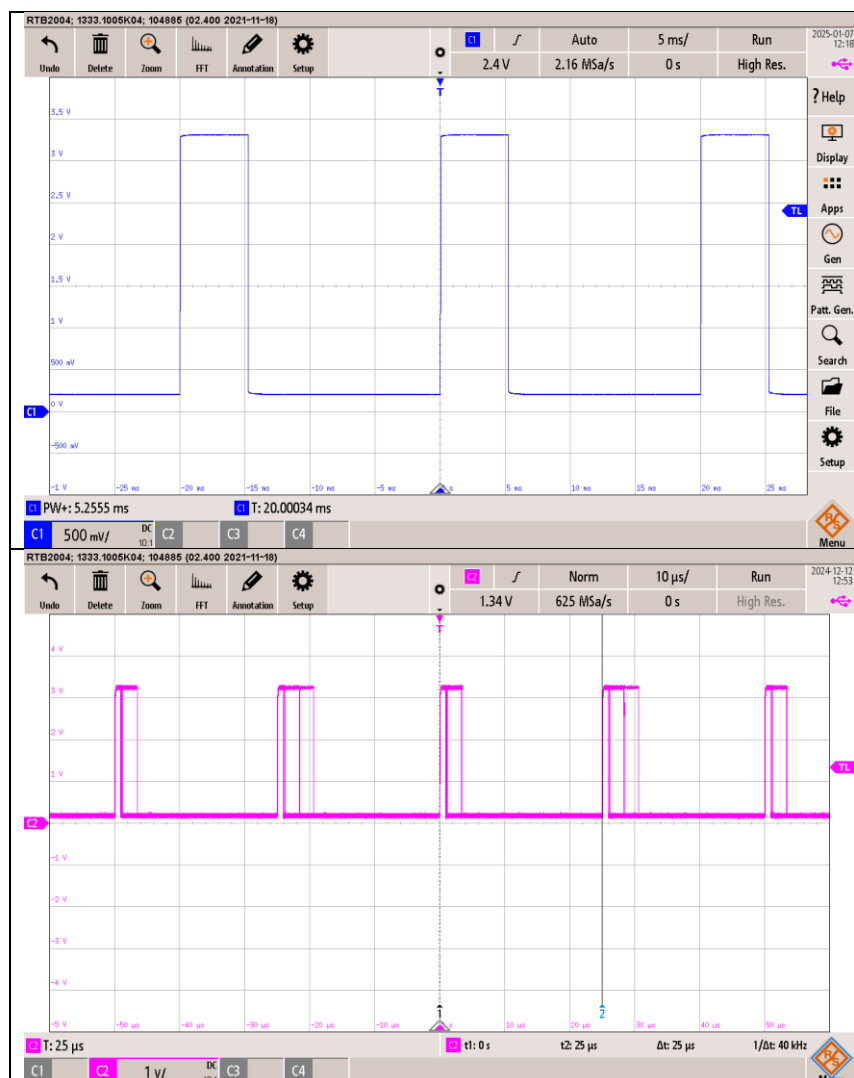
9 Annexes

9.1 Cahier des charges

9.2 Liste du matériel

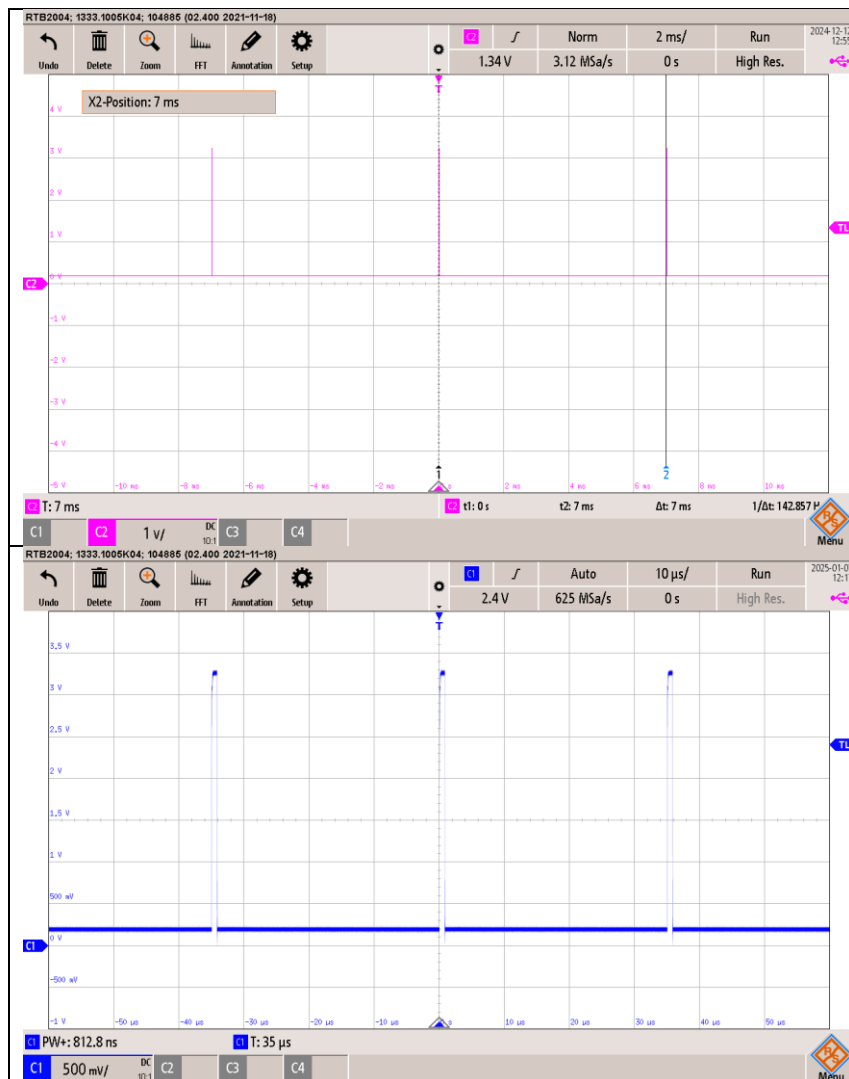
Désignation	Marque	Type	Caractéristiques	N° inventaire
G1	-	PC	Port USB	-
G2	SEFRAM	6330	Alimentation	ES.SLO2.00.00.24
P1	Rohde&Schwarz	RTB2004	Oscilloscope 2,5GS/s	ES.SLO2.05.01.12
G3	Ningbo FTZ Hopwell	DF1730SL5A	0-30V/0-5A	LO.SEV.01.04.06
P2	Agilent	54621A	60MHz/200MSa/s	LO.SEV.01.00.03
M1	-	Moteur	12V	-
M2	-	Servomoteur	-	-
K1	ETML	Carte	Kit-ES	ES.SLO2.00.05.36

9.3 Mesure Timers



Je peux voir la période ainsi que la durée d'impulsion.

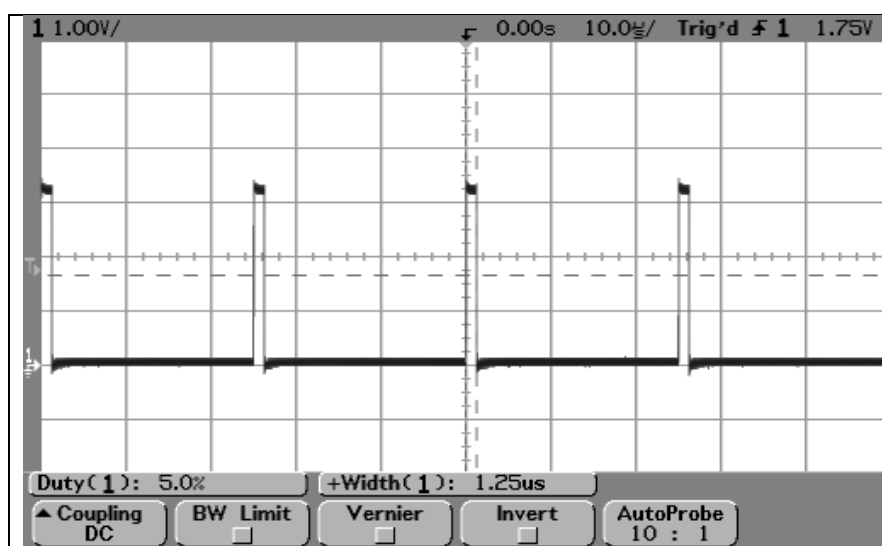
Je peux voir la période.



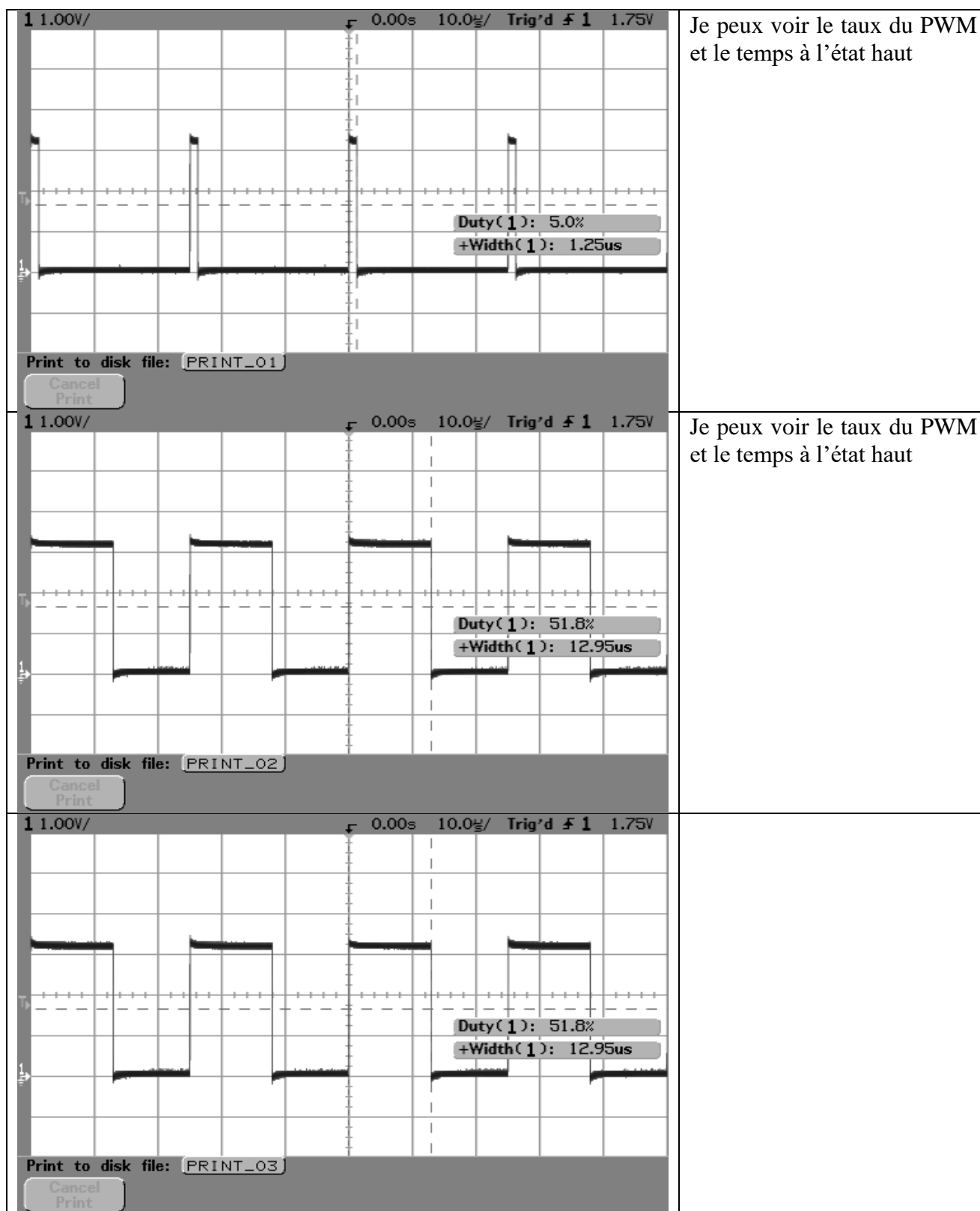
Je peux voir la période.

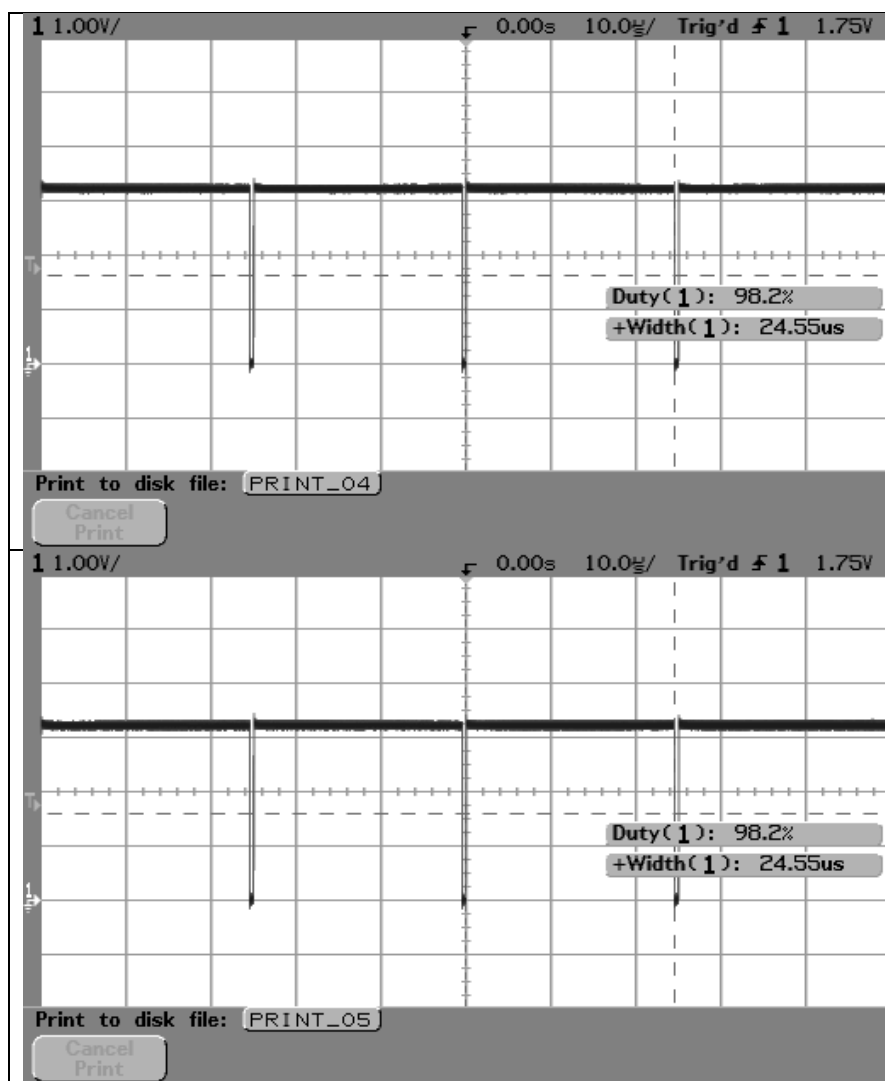
Je peux voir la période ainsi que la durée d'impulsion.

9.4 PWM OC2



Je peux voir le taux du PWM et le temps à l'état haut

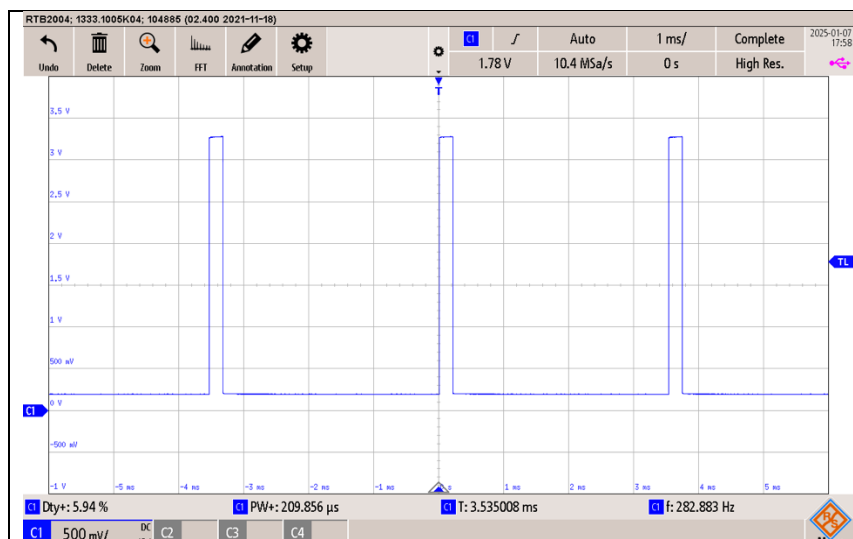




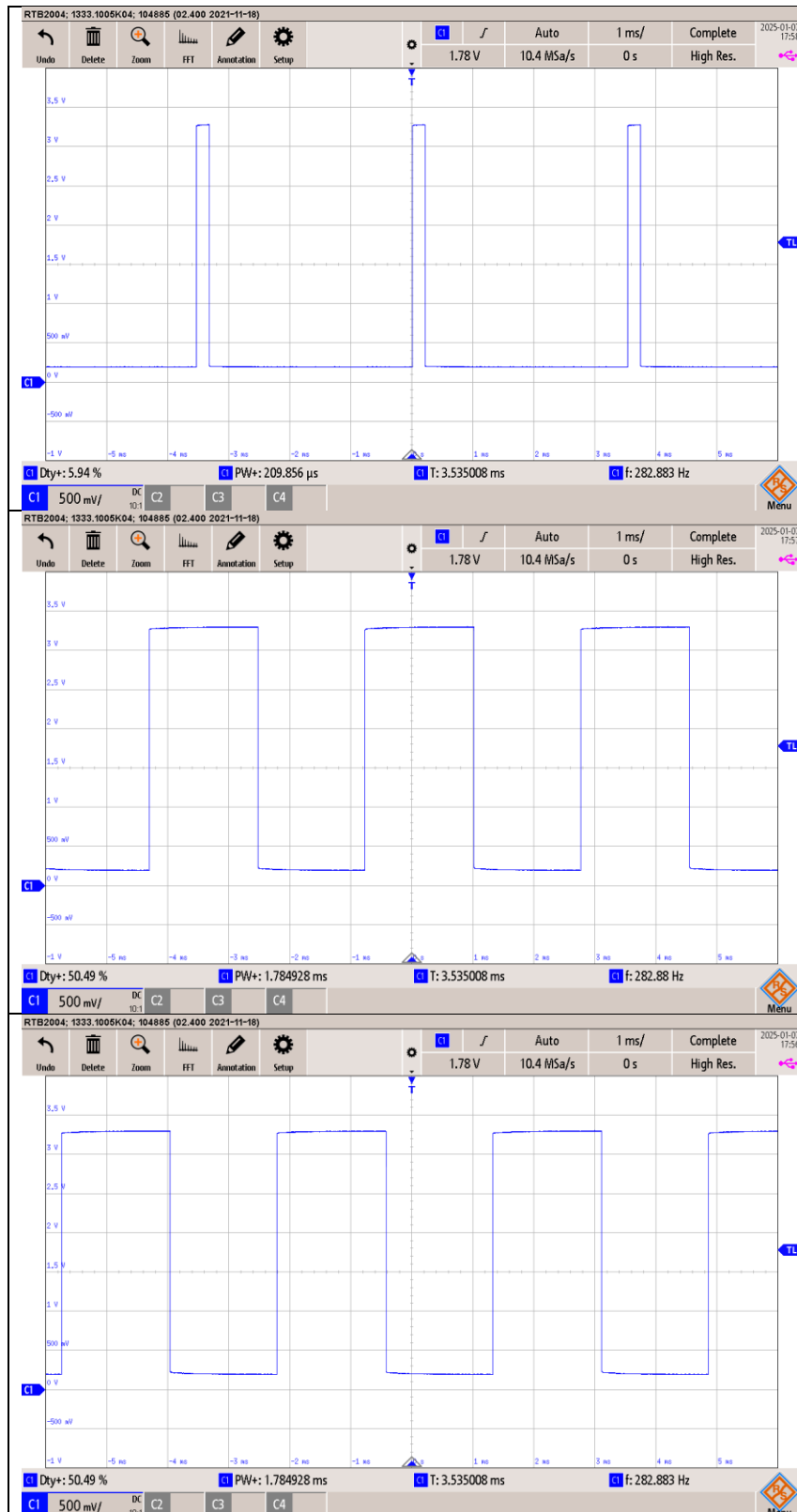
Je peux voir le taux du PWM et le temps à l'état haut

Je peux voir le taux du PWM et le temps à l'état haut

9.5 PWM Soft



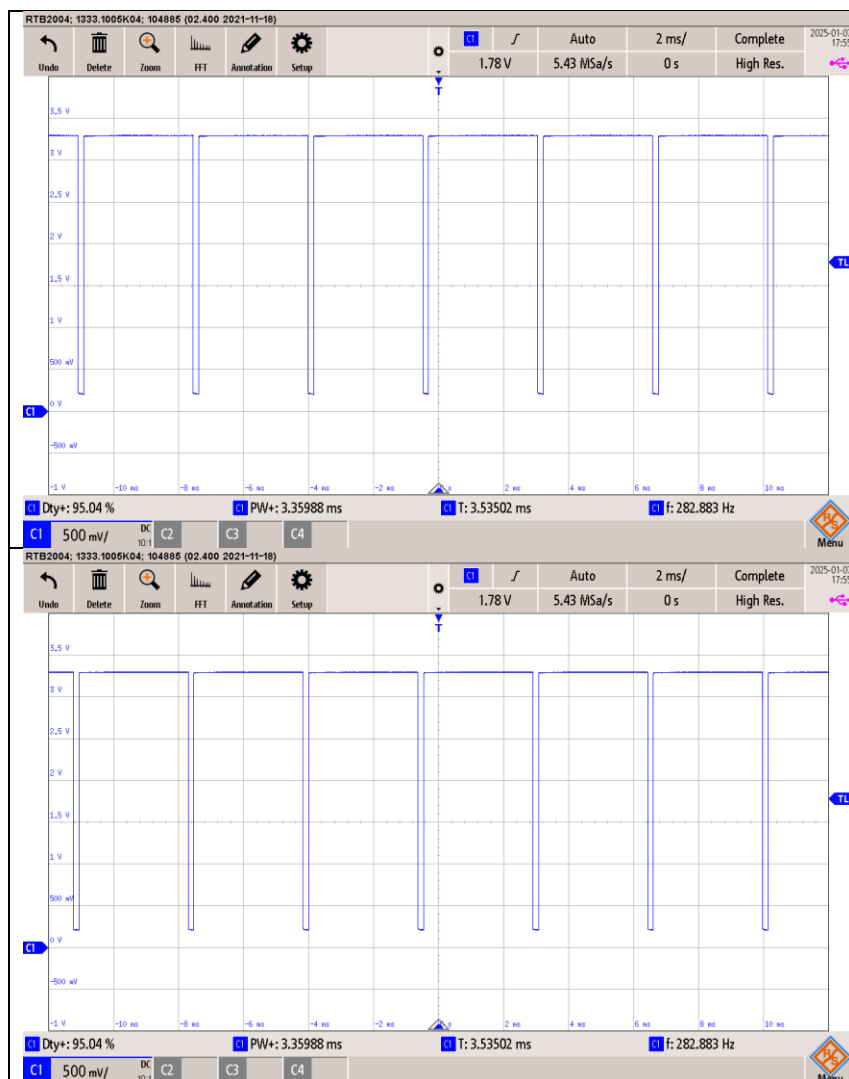
Je peux voir le taux du PWM et le temps à l'état haut. Ainsi que la période. Plus la fréquence.



Je peux voir le taux du PWM et le temps à l'état haut. Ainsi que la période.

Je peux voir le taux du PWM et le temps à l'état haut. Ainsi que la période.

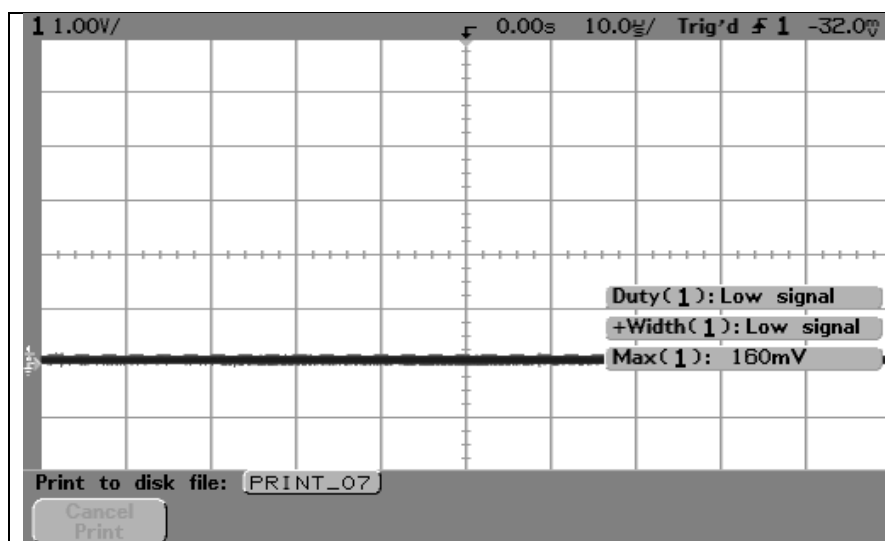
Je peux voir le taux du PWM et le temps à l'état haut. Ainsi que la période.



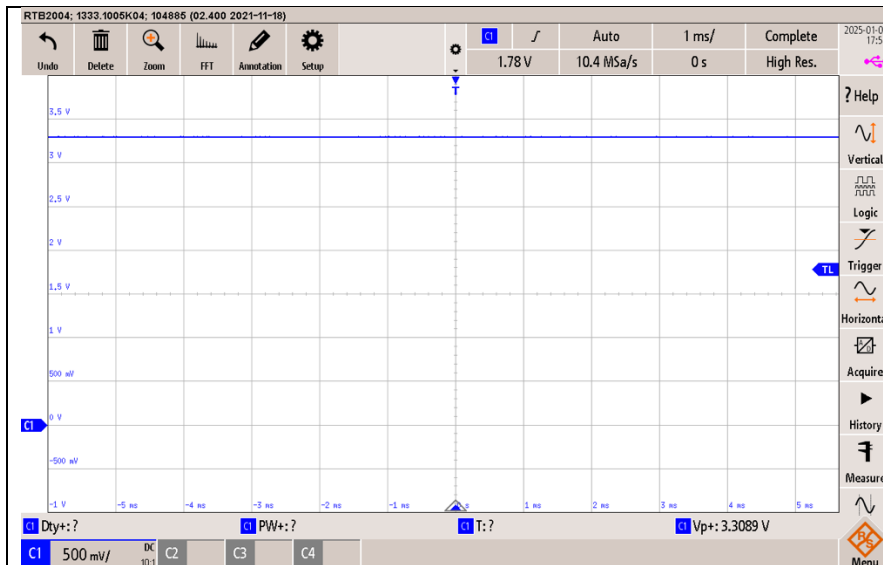
Je peux voir le taux du PWM et le temps à l'état haut. Ainsi que la période.

Je peux voir le taux du PWM et le temps à l'état haut. Ainsi que la période.

9.6 Passage PWM à 0

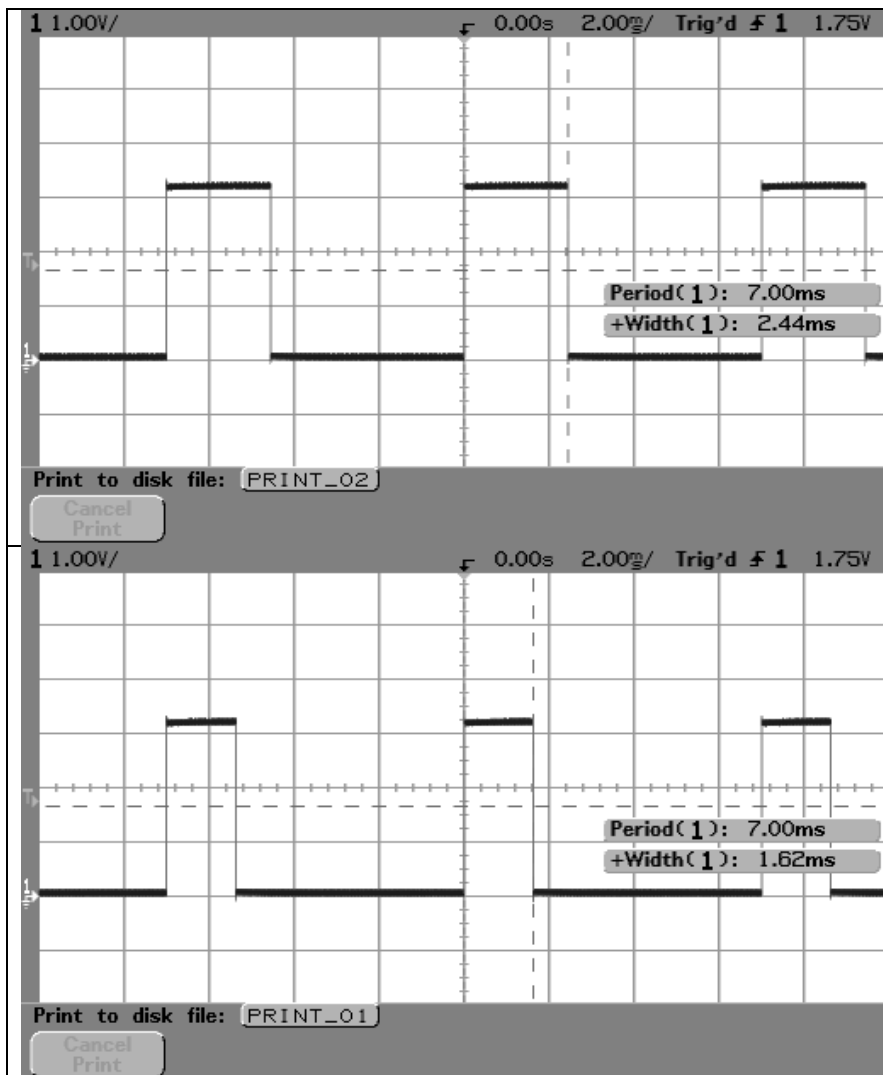


On peut voir le signal à 0V.



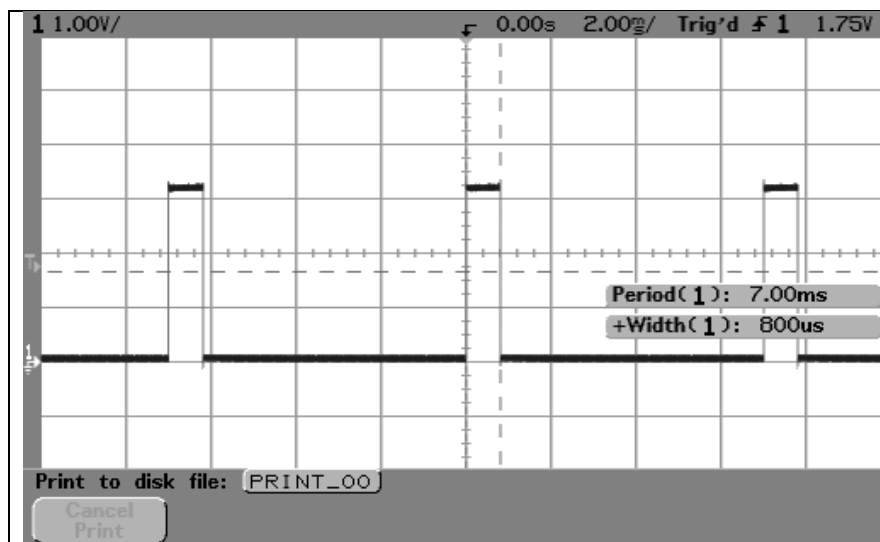
On peut voir le signal à 3.3V.

9.7 Servomoteur



Cas 90°: On peut voir la période du signal. Ainsi que la durée de l'état haut.

Cas 0°: On peut voir la période du signal. Ainsi que la durée de l'état haut.



Cas -90° : On peut voir la période du signal. Ainsi que la durée de l'état haut.

9.8 Extrait cours ch.5 p.9

5.2.4. RELATION ENTRE SOURCES D'INTERRUPTION ET REGISTRES

TABLE 7-1: INTERRUPT IRQ, VECTOR AND BIT LOCATION

Interrupt Source ⁽¹⁾	IRQ	Vecto Number	Interrupt Bit Location			
			Flag	Enable	Priority	Sub-Priority
Highest Natural Order Priority						
CT – Core Timer Interrupt	0	0	IFS0<0>	IEC0<0>	IPC0<4:2>	IPC0<1:0>
CS0 – Core Software Interrupt 0	1	1	IFS0<1>	IEC0<1>	IPC0<12:10>	IPC0<9:8>
CS1 – Core Software Interrupt 1	2	2	IFS0<2>	IEC0<2>	IPC0<20:18>	IPC0<17:16>
INT0 – External Interrupt 0	3	3	IFS0<3>	IEC0<3>	IPC0<28:26>	IPC0<25:24>
T1 – Timer1	4	4	IFS0<4>	IEC0<4>	IPC1<4:2>	IPC1<1:0>
IC1 – Input Capture 1	5	5	IFS0<5>	IEC0<5>	IPC1<12:10>	IPC1<9:8>
OC1 – Output Compare 1	6	6	IFS0<6>	IEC0<6>	IPC1<20:18>	IPC1<17:16>
INT1 – External Interrupt 1	7	7	IFS0<7>	IEC0<7>	IPC1<28:26>	IPC1<25:24>
T2 – Timer2	8	8	IFS0<8>	IEC0<8>	IPC2<4:2>	IPC2<1:0>
IC2 – Input Capture 2	9	9	IFS0<9>	IEC0<9>	IPC2<12:10>	IPC2<9:8>
OC2 – Output Compare 2	10	10	IFS0<10>	IEC0<10>	IPC2<20:18>	IPC2<17:16>
INT2 – External Interrupt 2	11	11	IFS0<11>	IEC0<11>	IPC2<28:26>	IPC2<25:24>
T3 – Timer3	12	12	IFS0<12>	IEC0<12>	IPC3<4:2>	IPC3<1:0>
IC3 – Input Capture 3	13	13	IFS0<13>	IEC0<13>	IPC3<12:10>	IPC3<9:8>
OC3 – Output Compare 3	14	14	IFS0<14>	IEC0<14>	IPC3<20:18>	IPC3<17:16>
INT3 – External Interrupt 3	15	15	IFS0<15>	IEC0<15>	IPC3<28:26>	IPC3<25:24>
T4 – Timer4	16	16	IFS0<16>	IEC0<16>	IPC4<4:2>	IPC4<1:0>
IC4 – Input Capture 4	17	17	IFS0<17>	IEC0<17>	IPC4<12:10>	IPC4<9:8>
OC4 – Output Compare 4	18	18	IFS0<18>	IEC0<18>	IPC4<20:18>	IPC4<17:16>
INT4 – External Interrupt 4	19	19	IFS0<19>	IEC0<19>	IPC4<28:26>	IPC4<25:24>
T5 – Timer5	20	20	IFS0<20>	IEC0<20>	IPC5<4:2>	IPC5<1:0>
IC5 – Input Capture 5	21	21	IFS0<21>	IEC0<21>	IPC5<12:10>	IPC5<9:8>
OC5 – Output Compare 5	22	22	IFS0<22>	IEC0<22>	IPC5<20:18>	IPC5<17:16>
SPI1E – SPI1 Fault	23	23	IFS0<23>	IEC0<23>	IPC5<28:26>	IPC5<25:24>
SPI1RX – SPI1 Receive Done	24	23	IFS0<24>	IEC0<24>	IPC5<28:26>	IPC5<25:24>
SPI1TX – SPI1 Transfer Done	25	23	IFS0<25>	IEC0<25>	IPC5<28:26>	IPC5<25:24>
U1E – UART1 Error	26	24	IFS0<26>	IEC0<26>	IPC6<4:2>	IPC6<1:0>
SPI3E – SPI3 Fault						
I2C3B – I2C3 Bus Collision Event						
U1RX – UART1 Receiver						
SPI3RX – SPI3 Receive Done	27	24	IFS0<27>	IEC0<27>	IPC6<4:2>	IPC6<1:0>
I2C3S – I2C3 Slave Event						
U1TX – UART1 Transmitter						
SPI3TX – SPI3 Transfer Done						
I2C3M – I2C3 Master Event	28	24	IFS0<28>	IEC0<28>	IPC6<4:2>	IPC6<1:0>
I2C1B – I2C1 Bus Collision Event						
I2C1S – I2C1 Slave Event						
I2C1M – I2C1 Master Event						
CN – Input Change Interrupt	32	26	IFS1<0>	IEC1<0>	IPC6<20:18>	IPC6<17:16>
AD1 – ADC1 Convert Done	33	27	IFS1<1>	IEC1<1>	IPC6<28:26>	IPC6<25:24>

Note 1: Not all interrupt sources are available on all devices. See Table 1, Table 2 and Table 3 for the list of available peripherals.

9.9 Feuille de contrôle