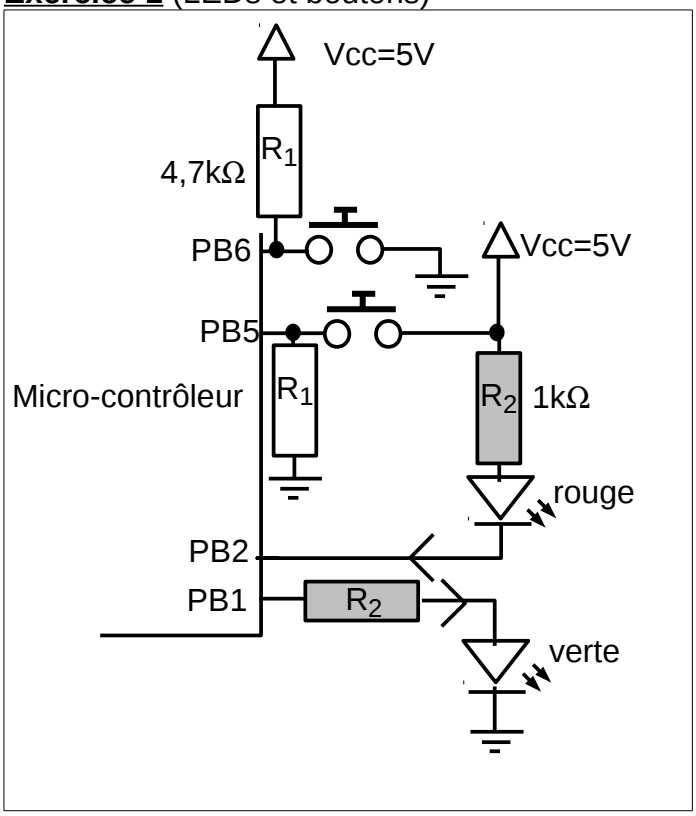
NOM :
Prénom :
Groupe :

Devoir Surveillé M2103

Exercice 1 (LEDs et boutons)



On donne ci-contre le schéma du montage que l'on désire utiliser pour cet exercice. PB6, PB5, PB2 et PB1 sont les bits utilisés du PORTB.

1°) Comment configurer le PORTB pour utiliser l'ensemble des bits présentés. Les bits PB7, PB4, PB3 et PB0 seront considérés comme des entrées. Réponse en hexadécimal et en binaire

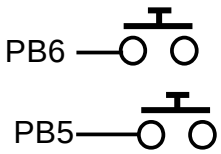
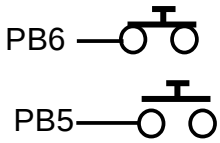
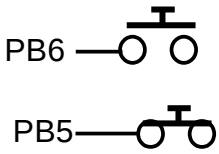
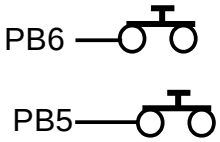
Réponses :

DDRB =0x ; //hexadecimal

DDRB =0b ; //binaire

2°) Dans un programme C, on trouve :
inters=PINB & 0x60;
Comment déclareriez-vous la variable inters ?
Réponse :

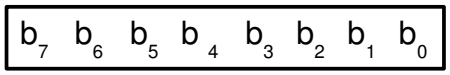
3°) Que donne l'instruction inters=PINB & 0x60; pour le schéma ci-dessus et pour les 4 possibilités des positions de l'interrupteur ?
Réponses :

			
0x	0x	0x	0x
0b	0b	0b	0b

4°) Écrire une boucle qui attend tant que tous les interrupteurs sont appuyés, et qui utilise l'instruction : inters=PINB & 0x60;

Réponse :

Exercice 2 (Expressions en C)
1°) Si une variable p de type char (8 bits signés) est déclarée écrire les expressions en C utilisant des masques permettant de :



- mettre à 1 les bits b4 et b6
- mettre à 0 les bits b3 et b2
- inverser le bit b5 (se fait facilement avec un ou exclusif)
- mettre à 1 le bit b7 et à 0 le bit b2

Réponses :

2°) Construire des expressions booléennes sur les tests suivants : expression vraie si
- le bit b6 est à 1 et le bit b4 est à 0 (de la variable p1)
- le bit b7 est à 1 ou le bit b1 est à 0 (de la variable p1)

Réponses :

3°) Donner toutes les valeurs possibles de l'expression :

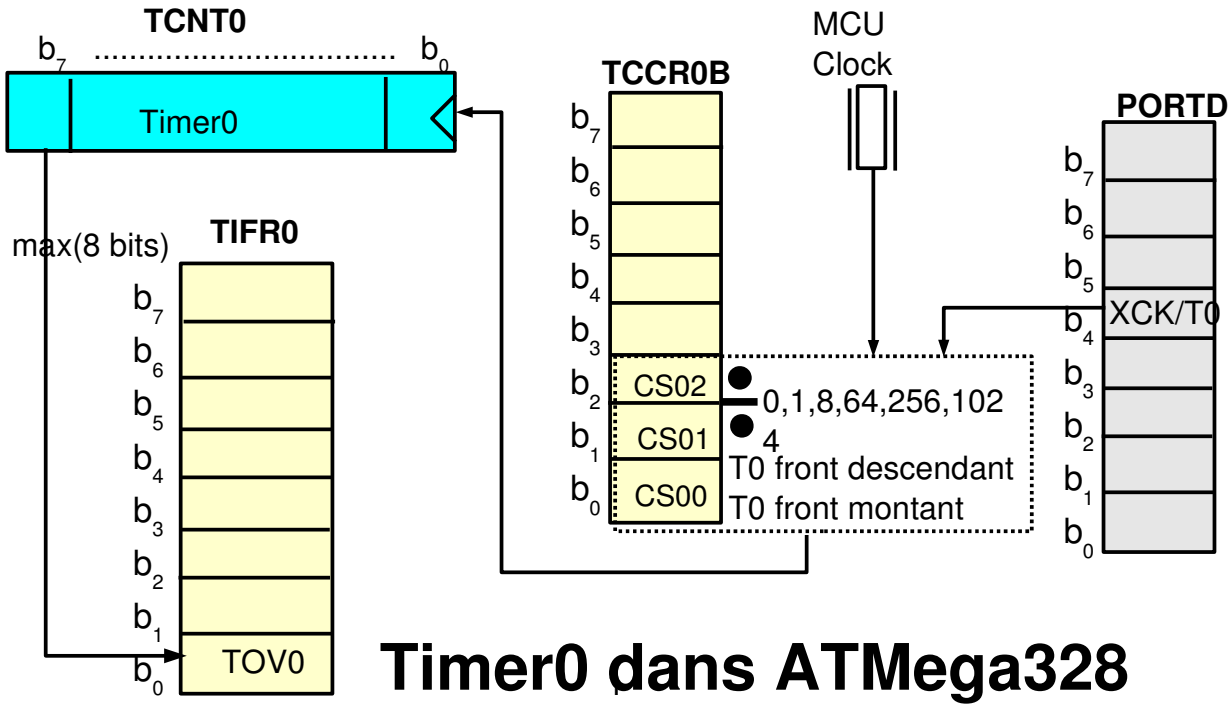
((p1 & 0x40) >>6) ^((p1 & 0x04)>>2)

pour toutes les valeurs de p1. Il y a 256 valeurs possibles pour p1, si vous ne trouvez aucune astuce de calcul vous y passerez quelques heures, soit bien plus que la durée du DS !

Réponses :

NOM :
Prénom :

Exercice 3 Analyse de programme C



Le quartz est choisi à 16 MHz dans ce premier problème.
1°) On donne le programme suivant concernant le timer 0 :

```
1         void main(void){
2         // initialisation du timer
3                 TCCR0B |= 0x04; // prescaler ??? , entrée sur quartz
4                 TCNT0 = 0x00; // RAZ de  tmr0
5         // bit PB1 du PORTB en sortie
6                 DDRB |= 0x02;
7                 while(1) {
8                         while ((TIFR0 & 0x01) == 0);
9                         TCNT0 = 32;
10                        TIFR0 |= 0x01;
11                        PORTB ^= 0x02 ; //  : basculement bit
12                }
13        }
```

Pouvez-vous donner la fréquence et la période du bit b_1 du **PORTB** avec quelques explications ?
Réponses :

2°) On vous donne maintenant un programme utilisant une interruption avec le timer0.

```
1         volatile unsigned char nb;
2
3         ISR(TIMER0_OVF_vect) {
4           nb++;
5           if (!(nb % 5)) // ou ((nb%5)==0)
6               PORTB = PORTB ^ 0x02 ; // ^ est un ou exclusif
7           TCNT0 = 0x0F;
8         } // fin ISR
9         main(void) {
10          DDRB = 0xFF; // 8 sorties pour B
11          TCCR0B |= 0x05; // prescaler ??? , entree sur quartz
12          TIMSK=(1<<TOIE0);  // autorise l'interruption timer
13          PORTB = 0x00; // aucune diode allumee
14          TCNT0 = 0x00 ;
15          nb=0;
16          TIFR0 |= 1<<TOV0 ; // clear TOV0
17          sei() ;
18          while(1) {
19           // on ne fait rien et c'est bien reposant...
20            }
21        }
```

Comme vous pouvez le constater, "nb" est incrémenté sans arrêt à chaque interruption. Compléter le tableau ci-dessous pour trouver la valeur de l'expression "(!(nb % 5))" (ou de **((nb%5)==0)**) en temps qu' expression booléenne : autrement dit la troisième ligne de ce tableau doit être remplie par des VRAI et FAUX !

Réponse :

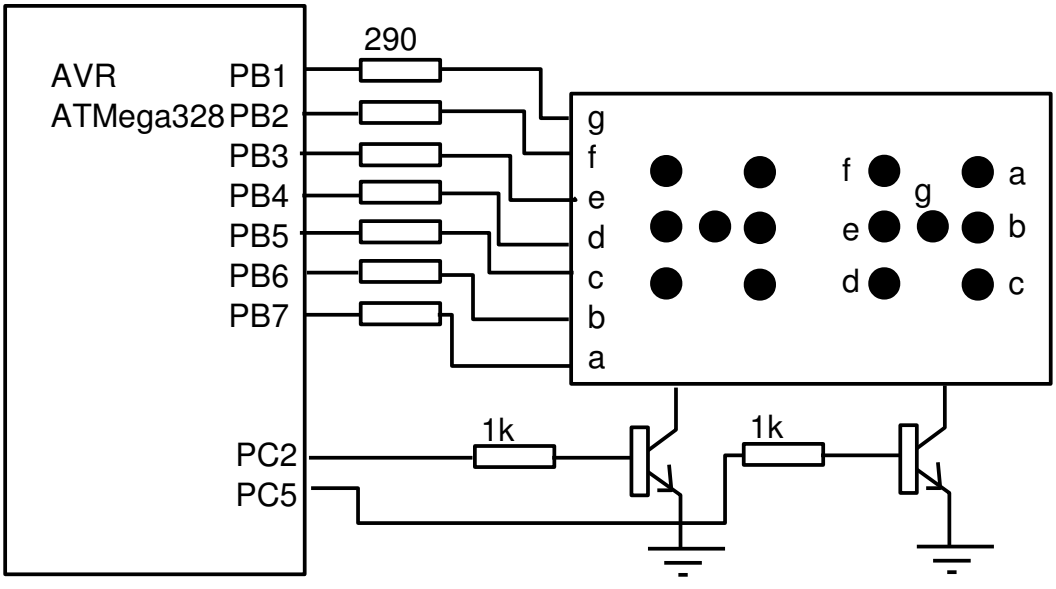
nb	0	1	2	3	4	5	6	7	8	9	10	11
nb%5												
((nb%5)==0)												
PORTB	0x02											

Pouvez-vous en déduire ce qui se passe sur le **PORTB** et à quelle fréquence (avec quartz à 16 MHz)?

Réponse :

Exercice 4

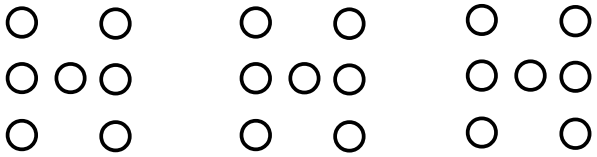
Une partie matérielle est constituée de deux dés (de 7 leds) multiplexés. Les leds sont commandées par le **PORTB**, tandis que les commandes d'affichages sont réalisées par les bits b_2 et b_5 du **PORTC**. Un schéma de principe est donné ci-après.



NOM :
Prénom :

Les leds s'allument avec un "1" logique, comme en TP M1103. Seuls les chiffres 1, 2, 3, 4, 5 et 6 peuvent être affichés, comme sur un dé de la vraie vie !

1°) Les trois premières valeurs d'un tableau de transcodage sont données :
`"unsigned char val_du_de[] = {0x02,0x90,0x92,...};"`
Dessinez ce qui sera affiché sur le dé pour ces trois valeurs sorties sur le **PORTB**.
Réponses :



2°) Compléter une fonction responsable du transcodage au plus simple (pas de gestion des erreurs si « no » est différent de 0, 1, 2 3, 4 et 5, et si 0 affiche 1 ... et 5 affiche 6 !):

```
1         unsigned char Display(unsigned char no) {
2             unsigned char Pattern;
3             unsigned char val_du_de[] = {0x02,0x90,0x92,
```

Réponse : (ci-dessus pour les valeurs manquantes du tableau et ci-dessous pour la suite du sous-programme)

3°) On vous donne le morceau de code :

```
1         unsigned char pseudoAleat(int Lim) {
2           unsigned char result;
3           static unsigned int Y=1;
4           Y = (Y * 32719 + 3) % 32749;
5           Result = ((Y % Lim));//
6           return Result;
7         }
```

Ce code est appelé chaque fois que l'on réalise un lancé de dé avec la valeur Lim=6. D'après les lignes 5 et 6, quelles sont alors les différentes valeurs possibles retournées par ce sous-programme.
Réponses :

4°) Donner en décimal la première valeur retournée par ce sous-programme.
Réponse :

L'interruption qui réalise l'alternance de l'affichage des deux dés est donnée maintenant. Elle utilise trois variables globales mux, de1 et de2 de type char :

```
1         ISR(TIMER0_OVF_vect) {
2             TCNT0 = 100;
```

```
3           // poids faible
4             if (mux == 0x20) { mux = 0x04;  PORTB = Display(de1); }
5           // poids fort
6             else { mux = 0x02; PORTB = Display(de2); }
7             PORTC = mux; // mux variable globale initialisée à 0x04
8         }
```

5°) Expliquez les valeurs 0x20 et 0x04 de la variable mux (unsigned char) qui sort sur le PORTC si allumer un dé consiste à rendre son transistor passant. Préciser pour chaque valeur de mux le(s) dé(s) concerné(s).
Réponse :

6°) Une entrée est ajoutée pour lancer le dé. Dessinez-là sur le dessin du dé en choisissant un pull-up externe ou interne.

7°) Quelle est, avec le choix de la question 6°), l'instruction qui attend que le bouton de lancé de dé soit appuyé.

Réponse :

8°) Comment généreriez-vous une valeur aléatoire dépendant du temps d'appui (sans timer) utilisant le sous-programme de la question 3°).

Réponse :

Exercice 5

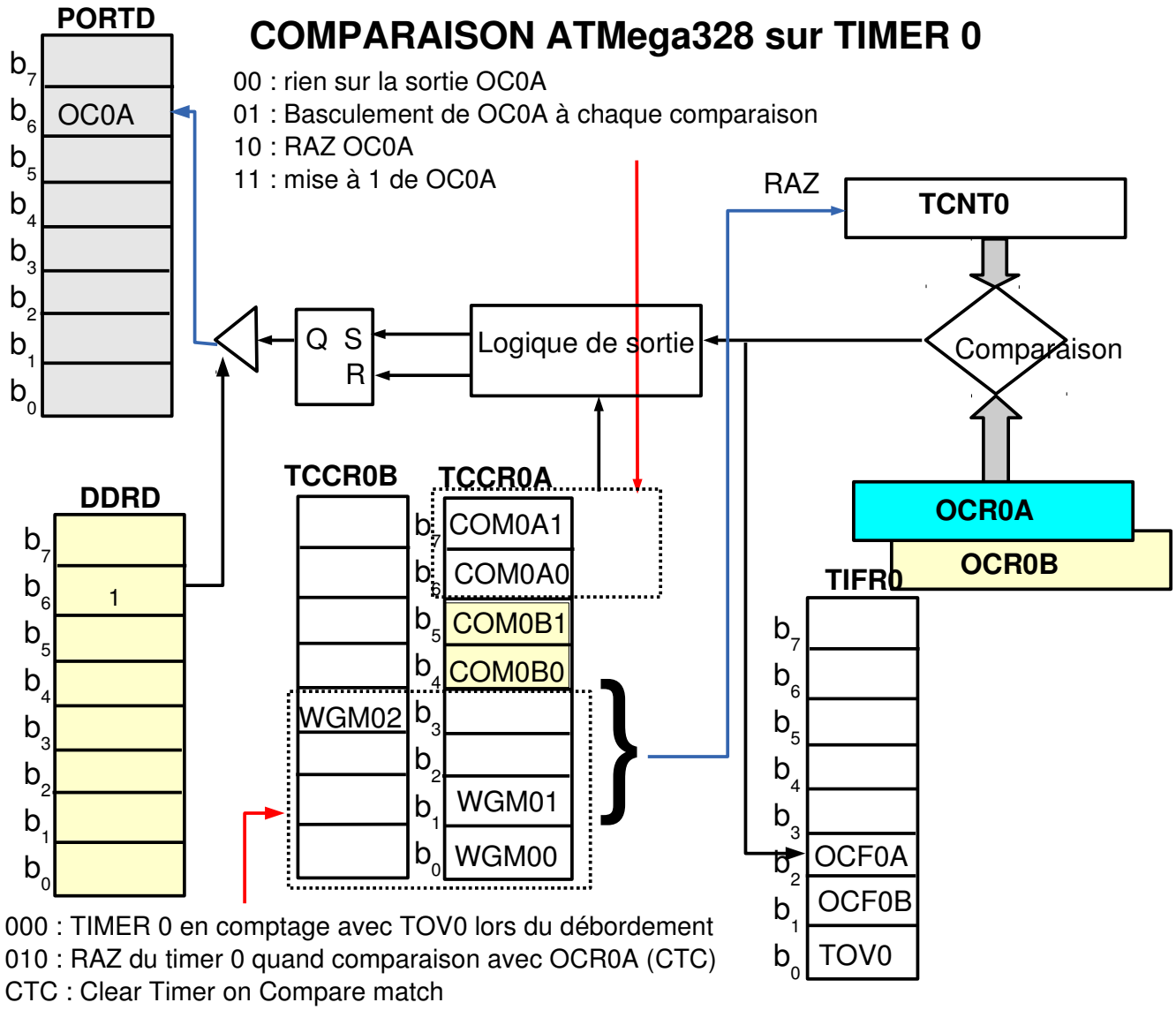
Écrire un programme complet qui réalise une fréquence de 39 Hz sur le bit b6 du PORTD en utilisant le mode CTC (Clear Timer on Compare match) que l'on traduirait par Reset du Timer quand Comparaison.

Ce programme sera utilisé dans une carte Arduino UNO caractérisée par un processeur ATMega328p et un quartz de 16 MHz.

Ce programme pourra être écrit en C pur ou en langage Arduino. On rappelle que ce genre de programme prend une dizaine de lignes tout au plus !

Si vous choisissez le langage C, on vous demande d'omettre la partie inclusion (autrement dit les #include...)

NOM :
Prénom :



Réponse : (en langage C ou Arduino)