


|          |   |  |                         |                |
|----------|---|--|-------------------------|----------------|
| BTS S.N. | <div>Enseignement Catholique</div> <div>Lédonien</div> <div></div> <div>JEANNE D'ARC • LA SALLE • SAINTE-MARIE</div> | « Systèmes Numériques »                | 14/03/2019              |                |
|          |   | Projet porte de poulailler automatique |                         |                |
|          |   | Année : 2018 / 2019                    | 1 <sup>ière</sup> Année | Samuel LITZLER |

**Objectif :**

Le but de ce **TP** est de réaliser la programmation d'un **afficheur 2x16 caractères** ainsi que d'un **clavier 16 touches**. Les deux seront reliés à un microcontrôleur **ATMEGA32**.

**Prérequis :**

Cours : « Le microcontrôleur ATMEGA32 »

Cours : « Programmation du microcontrôleur ATMEGA32 »

Cours : « Gestion des interruptions sur ATMEGA32 »

**Documents constituant le dossier :**

Page de garde (page 1)

TP de

**Remarque :**

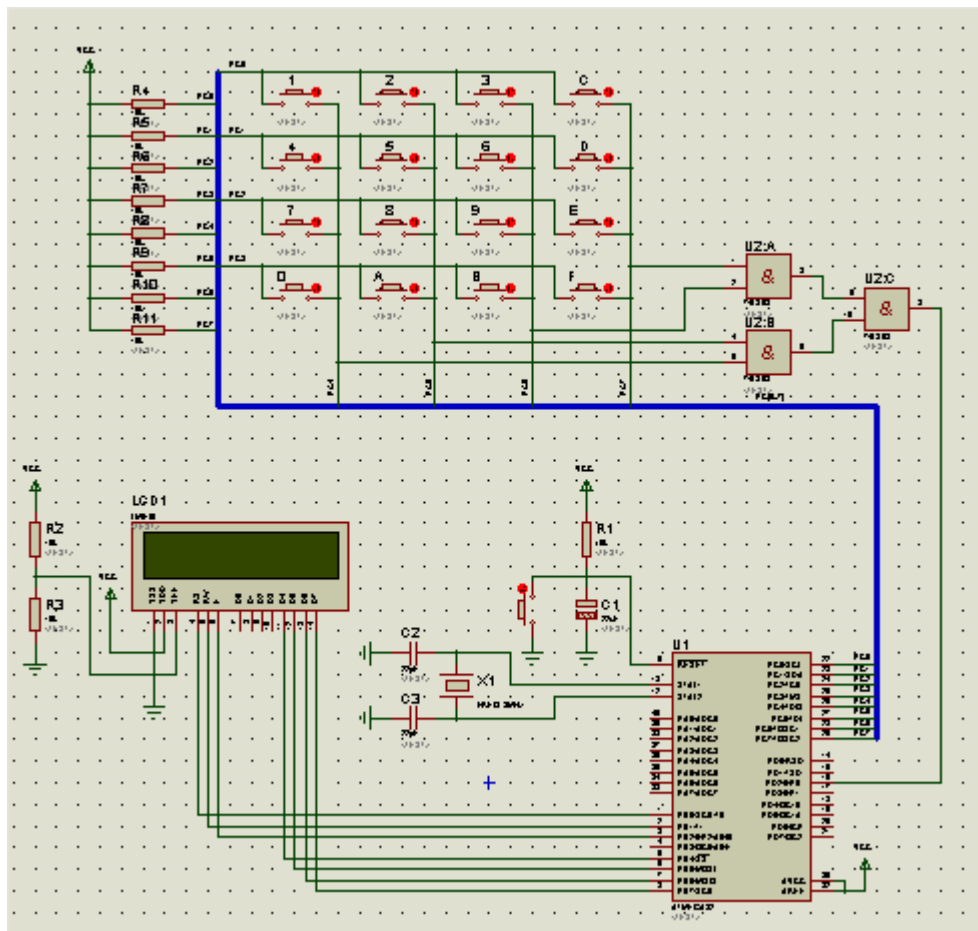
Nous utiliserons le logiciel **ISIS** pour faire les tests des programmes. Le programme sera fait avec le logiciel **CodeVisionAVR**.

## 1 / Utilisation et mise en œuvre d'un afficheur 2 x 16 caractères

Nous avons besoin d'une bibliothèque pour gérer des afficheurs LCD donc nous devons ajouter la bibliothèque « `alcd.h` ».

```
// Alphanumeric LCD functions
#include <alcd.h>
```

## 2 / Utilisation et mise en œuvre d'un afficheur 2 x 16 caractères



### 3 / Détection d'une touche

- La partie du port où se trouve les lignes est le port D.
- La partie du port où se trouve les colonnes est le port C.

- Le rôle des portes logiques est de signaler quand la touche est enclenchée.
- Le **PORT C de 0 à 3** sont initialement instanciés à **1** et le **PORT C de 4 à 7** sont eux initialement à **0**.  
Lorsque qu'aucune pression n'est effectuée sur une des touche, **PC4 à PC7** passe à **1** et de **PC4 à PC7** à **0** c'est-à-dire que **PC4, PC5, PC6 et PC7** reste à **0** et l'information qu'une touche a été pressé est envoyé.
- Nous initialisons deux variables **lecture** et **touche** ainsi que deux tableau **ligne1** et **ligne2** pour l'afficheur **LCD**.

```
#include <mega32.h>      // bibliothèque pour utiliser l'ATMEGA32
#include <delay.h>        // bibliothèque pour utiliser la temporisation

// Alphanumeric LCD functions
#include <alcd.h>         // bibliothèque pour utiliser l'afficheur

// Declare your global variables here
unsigned char lecture = 0x00;    // déclaration de la variable touche en l'initialisant à 0
unsigned char touche;           // déclaration de la variable touche
char ligne1[17] = "code touche : "; // il faudra afficher la valeur sur les positions 14 et 15
char ligne2[17] = "Touche      : "; // il faudra afficher la valeur sur la position 14
```

Le programme suivant permet de positionner le clavier dans une position d'attente d'appuis sur une touche.

```
delay_ms(10);           // attente de 10 ms pour éviter les rebonds
lecture = PINC;          // la variable lecture prend l'état du PINC
DDRC = 0xF0;            // le port C de 0 à 3 est ouvert et de 4 à 7 il est fermé
PORTC = lecture;        // on récupère la l'état du PINC via la variable lecture dans le PORTC
touche = PINC + lecture; // association de l'emplacement d'ela touche
DDRC = 0x0F;            // le port C de 0 à 3 est fermé et de 4 à 7 il est ouvert
PORTC = 0x00;           // on reinitialise le PORTC
delay_ms(100);          // attente de 10 ms pour éviter les rebonds
```

Nous aurons besoin aussi donc d'initialiser les ports de **PC0 à PC3** sur **1** et les ports de **PC4 à PC7** sur **0**.

```
// Port C initialization
// Function: Bit7=In Bit6=In Bit5=In Bit4=In Bit3=Out Bit2=Out Bit1=Out Bit0=Out
DDRC=(0<<DDC7) | (0<<DDC6) | (0<<DDC5) | (0<<DDC4) | (1<<DDC3) | (1<<DDC2) | (1<<DDC1) | (1<<DDC0);
// State: Bit7=T Bit6=T Bit5=T Bit4=T Bit3=0 Bit2=0 Bit1=0 Bit0=0
PORTC=(0<<PORTC7) | (0<<PORTC6) | (0<<PORTC5) | (0<<PORTC4) | (0<<PORTC3) | (0<<PORTC2) | (0<<PORTC1) | (0<<PORTC0);
```

- Voici ensuite donc le programme d'interruption agissant sur INT0 qui permet de détecter une touche enfoncée, qui stocke son code dans une variable, et qui initialise le port afin de positionner à nouveau le clavier dans une position d'attente d'appuis sur une touche.

```
#include <mega32.h>      // bibliothèque pour utiliser l'ATMEGA32
#include <delay.h>        // bibliothèque pour utiliser la temporisation

// Alphanumeric LCD functions
#include <alcd.h>         // bibliothèque pour utiliser l'afficheur

// Declare your global variables here
unsigned char lecture = 0x00;      // déclaration de la variable touche en l'initialisant à 0
unsigned char touche;              // déclaration de la variable touche
char ligne1[17] = "code touche : "; // il faudra afficher la valeur sur les positions 14 et 15
char ligne2[17] = "Touche       : "; // il faudra afficher la valeur sur la position 14

// External Interrupt 0 service routine
interrupt [EXT_INT0] void ext_int0_isr(void)
{
    // Place your code here
    delay_ms(10);                // attente de 10 ms pour éviter les rebonds
    lecture = PINC;               // la variable lecture prend l'état du PINC
    DDRC = 0xF0;                 // le port C de 0 à 3 est ouvert et de 4 à 7 il est fermé
    PORTC = lecture;             // on récupère la l'état du PINC via la variable lecture dans le PORTC
    touche = PINC + lecture;      // association de l'emplacement d'ela touche
    DDRC = 0x0F;                 // le port C de 0 à 3 est fermé et de 4 à 7 il est ouvert
    PORTC = 0x00;                // on reinitialise le PORTC
    delay_ms(100);               // attente de 10 ms pour éviter les rebonds
}
```

---

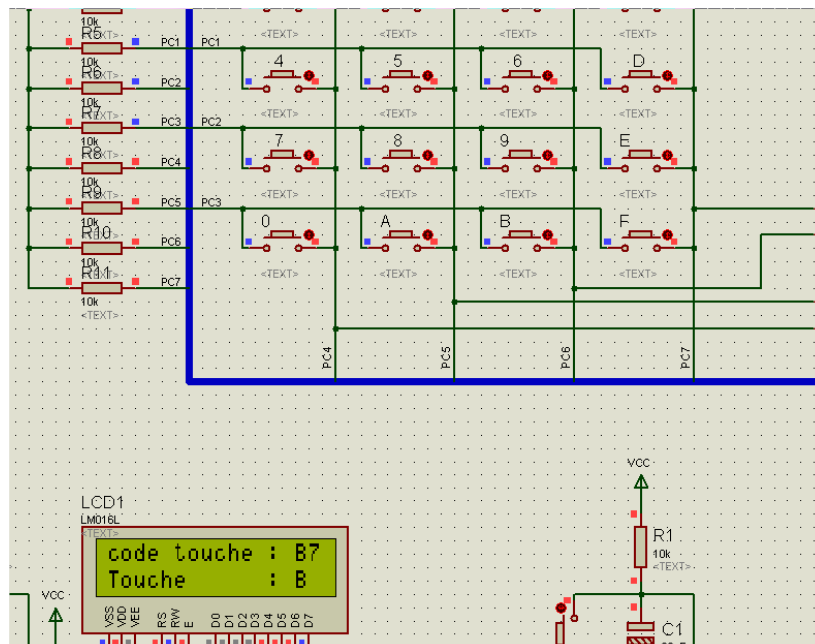
## 4 / Utilisation du clavier et de l'afficheur

---

Nous devons réaliser un programme qui permet de réaliser les fonctions suivantes :

- Initialisation du système (clavier + afficheur)
- Détection de l'appui sur une touche du clavier
- Affichage sur l'écran du code de la touche appuyée, ainsi que le numéro de la touche appuyée

**Exemple :**      Code touche : B7  
                   Touche        : B



```

lcd_init(16);           // initialisation de l'afficheur
lcd_gotoxy(0,0);        //          afficher le tableau
lcd_puts(ligne1);       // ligne1 sur la première ligne de l'afficheur

lcd_gotoxy(0,1);        //          afficher le tableau
lcd_puts(ligne2);       // ligne2 sur la deuxième ligne de l'afficheur

// Globally enable interrupts
#pragma asm("sei")

while (1)
{
    // Place your code here
    PORTA = touche; // lire la touche appuyée

    // touche 1
    if(touche == 0xEE) // condition si la touche 1 en appuyée
    {
        // alors
        lcd_gotoxy(14,0); // aller à la ligne d'abscisse 14 et d'ordonnée 0
        lcd_putchar('E'); // puis afficher E
        lcd_gotoxy(15,0); // aller à la ligne d'abscisse 15 et d'ordonnée 0
        lcd_putchar('E'); // puis afficher E
        lcd_gotoxy(14,1); // aller à la ligne d'abscisse 14 et d'ordonnée 1
        lcd_putchar('1'); // puis afficher 1
    }

    // Le principe est donc le même pour toutes les autres touches

```

```

// touche 2
if(touche == 0xDE)
{
    lcd_gotoxy(14,0);
    lcd_putchar('D');
    lcd_gotoxy(15,0);
    lcd_putchar('E');
    lcd_gotoxy(14,1);
    lcd_putchar('2');
}

```

```

// touche 8
if(touche == 0xDB)
{
    lcd_gotoxy(14,0);
    lcd_putchar('D');
    lcd_gotoxy(15,0);
    lcd_putchar('B');
    lcd_gotoxy(14,1);
    lcd_putchar('8');
}

```

```
// touche 3
if(touche == 0xBE)
{
    lcd_gotoxy(14,0);
    lcd_putchar('B');
    lcd_gotoxy(15,0);
    lcd_putchar('E');
    lcd_gotoxy(14,1);
    lcd_putchar('3');
}
```

```
// touche 4
if(touche == 0xED)
{
    lcd_gotoxy(14,0);
    lcd_putchar('E');
    lcd_gotoxy(15,0);
    lcd_putchar('D');
    lcd_gotoxy(14,1);
    lcd_putchar('4');
}
```

```
// touche 5
if(touche == 0xDD)
{
    lcd_gotoxy(14,0);
    lcd_putchar('D');
    lcd_gotoxy(15,0);
    lcd_putchar('D');
    lcd_gotoxy(14,1);
    lcd_putchar('5');
}
```

```
// touche 6
if(touche == 0xBD)
{
    lcd_gotoxy(14,0);
    lcd_putchar('B');
    lcd_gotoxy(15,0);
    lcd_putchar('D');
    lcd_gotoxy(14,1);
    lcd_putchar('6');
}
```

```
// touche 7
if(touche == 0xEB)
{
    lcd_gotoxy(14,0);
    lcd_putchar('E');
    lcd_gotoxy(15,0);
    lcd_putchar('B');
    lcd_gotoxy(14,1);
    lcd_putchar('7');
}
```

```
// touche 8
if(touche == 0xDB)
{
    lcd_gotoxy(14,0);
    lcd_putchar('D');
    lcd_gotoxy(15,0);
    lcd_putchar('B');
    lcd_gotoxy(14,1);
    lcd_putchar('8');
}
```

```
// touche 9
if(touche == 0xBB)
{
    lcd_gotoxy(14,0);
    lcd_putchar('B');
    lcd_gotoxy(15,0);
    lcd_putchar('B');
    lcd_gotoxy(14,1);
    lcd_putchar('9');
}
```

```
// touche 0
if(touche == 0xE7)
{
    lcd_gotoxy(14,0);
    lcd_putchar('E');
    lcd_gotoxy(15,0);
    lcd_putchar('7');
    lcd_gotoxy(14,1);
    lcd_putchar('0');
}
```

```
// touche B
if(touche == 0xB7)
{
    lcd_gotoxy(14,0);
    lcd_putchar('B');
    lcd_gotoxy(15,0);
    lcd_putchar('7');
    lcd_gotoxy(14,1);
    lcd_putchar('B');
}
```

```
// touche C
if(touche == 0x7E)
{
    lcd_gotoxy(14,0);
    lcd_putchar('7');
    lcd_gotoxy(15,0);
    lcd_putchar('E');
    lcd_gotoxy(14,1);
    lcd_putchar('C');
}
```

```
// touche D
if(touche == 0x7D)
{
    lcd_gotoxy(14,0);
    lcd_putchar('7');
    lcd_gotoxy(15,0);
    lcd_putchar('D');
    lcd_gotoxy(14,1);
    lcd_putchar('D');
}
```

```
// touche E
if(touche == 0x7B)
{
    lcd_gotoxy(14,0);
    lcd_putchar('7');
    lcd_gotoxy(15,0);
    lcd_putchar('B');
    lcd_gotoxy(14,1);
    lcd_putchar('E');
}
```

```
// touche F
if(touche == 0x77)
{
    lcd_gotoxy(14,0);
    lcd_putchar('7');
    lcd_gotoxy(15,0);
    lcd_putchar('7');
    lcd_gotoxy(14,1);
    lcd_putchar('F');
}
```

## 5 / Compte Rendu

Nous avons donc réalisé un programme qui utilise un **afficheur 2x16 caractères** ainsi qu'un **clavier 16 touches**, les deux reliés à un **microcontrôleur ATMEGA32**.

L'afficheur nous donne les informations de l'interruption de la touche en fonction de ce que l'on a mis dans les conditions **des boucles if**.