École de Génie Électrique et Informatique
Université d'Ottawa

uOttawa

CEG 3536– Architecture des ordinateurs II
Automne 2023

Laboratoire 2 :
L'interface au matériel - le clavier

Soumis par :
Elam Olame **Mugabo** 300239792
Decaho **Gbegbe** 300094197

Professeur : Mohamed Ali **Ibrahim**, PhD.ing
Assistant à l'Enseignement : Joel **Simweray**

Date de soumission : 21 octobre 2023

1.Objectifs

L’objectif principal de ce laboratoire était de nous introduire à l'interface matérielle du Motorola 9S12DG256 et créer une application de clavier numérique.

2. Équipement et composantes utilisées

- Windows PC
- Carte d’entraînement Dragon-12
- VMware Horizon Client

3.Expérience

Dans le cadre du laboratoire 2, la tâche consiste à élaborer un logiciel capable de lire les entrées du clavier de la carte Dragon-12 connectée au microcontrôleur par le biais du port parallèle, tout en intégrant votre module dans le logiciel du système d’alarme. Afin d'assurer un fonctionnement optimal, l'implémentation d'une fonction delay.ms et de keyPad.asm est nécessaire.

3.1. Partie I : Module delay.asm

Le module Delay du laboratoire 1 comprend une fonction du sous-programme "void delayms (int num)" qui génère un délai de "num" millisecondes. Le paramètre "num" est transmis à la fonction "delayms" via le registre D.

a. Code C

delayms.c 1 | lab2.cs | Untitled-1 | keyPad.cs

C: > Users > jeannette > Desktop > delayms.c > ...

```
1
2   /*#include <stdio.h>[]*/
3
4   void delayms (int num) {
5       setDelay (num) {
6       do {
7           complete = pollDelay ();
8           while (complete != TRUE );
9       }
10      }
11  }
12
```

Figure 1 : Capture de l'écran du code C

b. Code ASM

keyPad.cs | delayms.cs | delay.asm | keyPad.asm

C: > Users > jeannette > Desktop > AlarmAssembler > delay.asm > ...

```
; Some definitions

  SWITCH code_section

;-------------------------------
; Subroutine delayms
; Parameters: num - number of milliseconds to delay - in D
; Returns: nothing
; Description: Delays for num ms.
;-------------------------------
delayms:
    pshd ; push d
    pshx ; push x
    bsr setDelay ();

; correspond à notre boucle do
  boucle_delayms :
    bsr polldelay; branchement à polldelay
    bne boucle_fini; exécute 3 fois
    bra boucle_delayms ;saut relatif depuis la position actuelle

  boucle_fini ;
    pulx ; pull x
    puld ; pull d

  rts

;-------------------------------------------------------
; Subroutine setDelay
```

Figure 2 : Capture de l'écran du code ASM

3.2. Partie II

Ci-dessous, le sous-programme "keyPad.asm" a pour objectif d'initialiser le matériel (Port A) connecté au clavier, ainsi qu'un sous-programme dédié à la lecture d'une touche du clavier, renvoyant le code ASCII de la touche tapée.

a. Code C

keyPad.cs | delay.asm | keyPad.asm

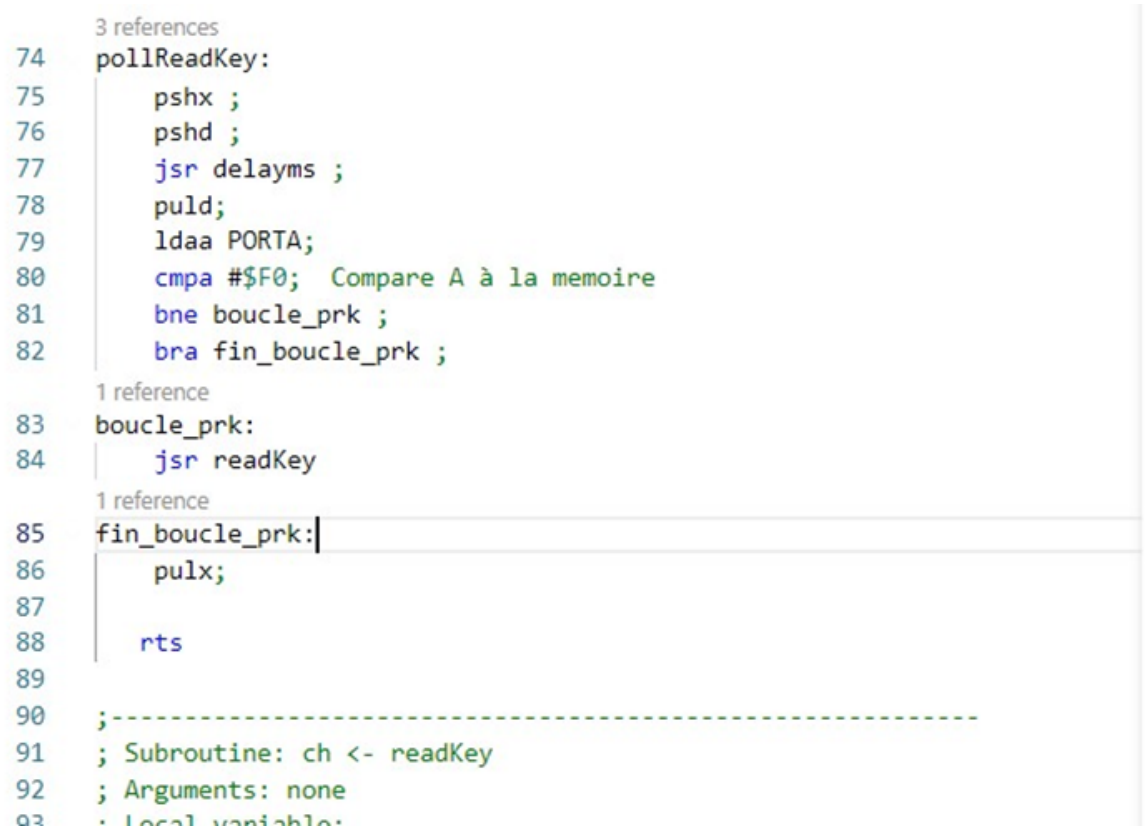
C: > Users > jeannette > Desktop > Lab2 > keyPad.cs

```
#include  <stdio.h>
#include  <time.h>

void initKeyPad (){
    PORTA = 0 ;   initialise la valeur du portA à 0 mais en soit ça devrait etre 0x0f
}

char readKey () {
    scanf ("%c", &ch) ; scan la valeur char de la clé
    byte ch;
    return ch ; valeur de la clé

}

char pollReadKey () {
    char ch = NOKEY ; pas de clé
    PORTA = 0x0f;
    int count = POLLCOUNT ;
    do {
        if (PORTA =! 0x0f ) {
            delayms (1) ; delay de 1ms
            if (PORTA != 0x0f) {
                ch = readKey () ; complète un rebond et traduit au clavier
            }
            break;
        }
        count --;
    } while (count !=0)
    return ch; retourne la clé
}
```

Figure 3 : Capture de l'écran du code C

b. ASM

- Pour le sous-programme pollReadKey :



```
3 references
74  pollReadKey:
75      pshx ;
76      pshd ;
77      jsr delayms ;
78      puld;
79      ldaa PORTA;
80      cmpa #$F0;  Compare A à la memoire
81      bne boucle_prk ;
82      bra fin_boucle_prk ;
1 reference
83  boucle_prk:
84      jsr readKey
1 reference
85  fin_boucle_prk:
86      pulx;
87
88    rts
89
90  ;-------------------------------------------------------------
91  ; Subroutine: ch <- readKey
92  ; Arguments: none
93  ; Local variable:
```

Figure 4 : Capture de l'écran du code ASM

- Pour le sous-programme readKey :

```
;  translate to get the corresponding ASCII code.
;-------------------------------------------------------------
; Stack Usage
    OFFSET 0  ; to setup offset into stack

readKey:
    ldaa PORTA;
    cmpa #$F0; compare A à la mémoire
    beq debounce;
    staa $3000 ;
    ldaa #$F0;
    staa Ddra;
    pshd;
    ldd #2;
    jsr delayms;
    puld;
    ldaa PORTA;
    staa $3001; moins significatif
    ldab $3000;
    orab $3001; op.log b dans la mémoire
    ldaa #$0F;
    staa Ddra;

debounce:
    pshd;
    ldd #15;
    jsr delayms ;
    puld;
    ldaa PORTA;
    cmpa #$F0 ;
    bne debounce ;

    rts
```

Figure 5 : Capture de l'écran du code ASM

- Pour le sous-programme initKeyPad :

```
;-------------------------------------------------------------
; Subroutine: initKeyPad
;
; Description:
;   Initiliases PORT A
;-------------------------------------------------------------
1 reference
initKeyPad:
;initiales le port registre a
    ldaa #$01  ;
    staa Ddra ;
    ldaa #$0F ;
    staa Pucr ;
    rts

;-------------------------------------------------------------
```

Figure 6 : Capture de l'écran du code ASM

4.Conclusion

Au cours de cette expérience, le premier défi auquel nous avons été confrontés était le module delay.asm, en raison d'erreurs détectées dans celui du laboratoire 1. Après avoir identifié ces erreurs, la conversion du code C en langage assembleur est devenue plus aisée.

Cependant, notre démonstration n'a pas abouti en raison de l'absence d'implémentation de la fonction readKeyCode. Cette lacune résulte du fait que la personne en charge de la section keyPad.asm n'a pas pris en considération cet aspect. Dans l'ensemble, l'objectif du laboratoire a été atteint avec succès, car une application de clavier numérique a été élaborée.