# Final

Durée 1h30, documents autorisés : une feuille de notes.
Tout code doit être clairement commenté, le barème est indicatif.

## 1. Questions (8)

**1.** Que réalisent les programmes ASM ARM suivants :

**1.1.**

```
ldr     r4, [r7]
add     r4, r4, #1
str     r4,[r7]
```

**1.2.**

```
cmp     r0, #0
bgt     label1
```

**2.** Expliquez ce que font les deux instructions suivantes et leur utilité :

BL label
B label

**3.** Expliquez comment fonctionne la zone mémoire de pile (stack) et donnez les éléments que l'on peut retrouver mémorisé dans la zone de pile

**4.** En langage C, quel est l'utilité du mot 'volatile' devant la déclaration d'une variable (e.g. volatile int32_t a ;)

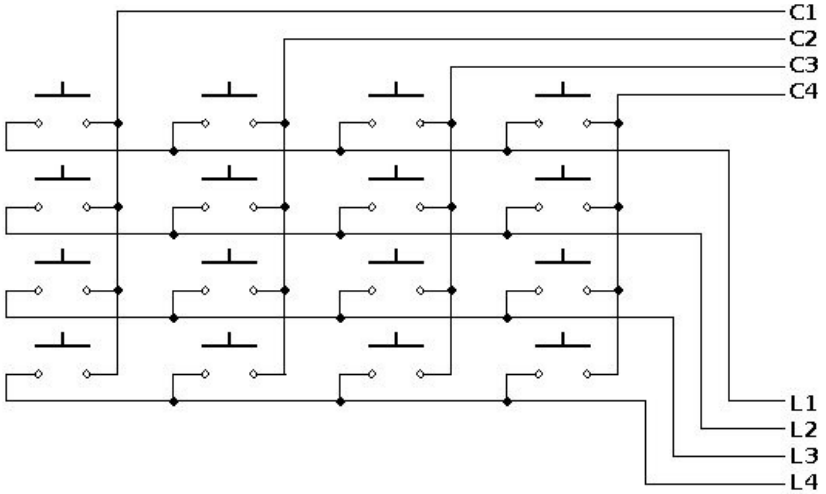
**5.** Sur un microcontrôleur cortex-M, l'adresse du vecteur d'exception de l'IRQ0 est 0x40. Expliquez ce qu'il se passe lorsqu'une requête d'interruption IRQ0 (non masquée) arrive sur ce microcontrôleur.

## 2. Utilisation du GPIO d'un microcontrolleur STM32 (12)

On souhaite interfacer à un microcontrôleur STM32 un clavier numérique à touche à l'aide d'un GPIO.

### 2.1. Description

Le clavier numérique 16 touches est constitué de 16 boutons poussoir permettant de mettre en contact une ligne avec une colonne. 8 connexions sont disponibles.
Clavier 16 touches :

Exemple l'appui sur la touche 8, interconnecte la ligne L3 avec la colonne C2.

Les broches du clavier 16 touches sont connectées aux 8 broches poids faible du GPIOA (PA7 à PA0) d'un microcontrôleur STM32 :

- Les lignes sont connectées aux 'pins' PA7 (L4) à PA4 (L1) et ces 4 ports seront configurés en sortie
- Les colonnes sont connectées aux 'pins' PA3 (C4) à PA0 (C1), ces 4 pins sont configurées en entrée avec des résistance de rappel haute (pull-up).

Les résistances de rappel permettent de maintenir les colonnes à l'état haut lorsque les boutons ne sont pas enfoncé (interrupteurs lignes/colonnes ouverts).
Ainsi les pins PA3 à PA0 (colonnes) seront au '1' logique lorsque aucune touche n'est enfoncée.

l'appuie sur une touche, mettra au niveau logique la ligne et la colonne correspondante. (e.g. l'appuie sur la touche 6 mettra la ligne L2 avec la colonne C3 au même niveau logique). En forçant sur la ligne le niveau logique 0, on peut détecter si le bouton est enfoncé par simple lecture des niveaux logiques des colonnes.

Par exemple, en forçant la ligne L2 à 0, si la touche 6 est appuyée, alors la colonne C3 sera à 0 dans le cas contraire elle sera à 1

Ainsi pour détecter si une des touche de ligne est enfoncé, il faut forcer cette ligne à 0 et laisser les autres lignes à la haute impédance et lire l'état des colonnes. En procéder ainsi pour chaque ligne.

Le principe pour détecter quelle touche est enfoncée, consiste à placer alternativement les lignes à 0 et à détecter l'état des colonnes en entrées.

## 2.2. Programmation

La programmation se fait en lange C. Le port GPIOA est à l'adresse 0x4000000. Vous utiliserez la structure définie ci-après (GPIO_TypeDef) pour accéder aux registres du GPIOA.
**Aucune constante n'est définie.**
La documentation simplifiée du GPIO est fournie en annexe

```
typedef struct
{
  volatile uint32_t MODER;       /*!< GPIO port mode register,    Address offset: 0x00*/
  volatile uint32_t OTYPER;      /*!< GPIO port output type register, Address offset: 0x04  */
  volatile uint32_t OSPEEDR;     /*!< GPIO port output speed register, Address offset: 0x08 */
  volatile uint32_t PUPDR; /*!< GPIO port pull-up/pull-down register, Address offset:0x0C*/
  volatile uint32_t IDR;  /*!< GPIO port input data register, Address offset: 0x10 */
  volatile uint32_t ODR;  /*!< GPIO port output data register, Address offset: 0x14 */
  volatile uint32_t BSRR; /*!< GPIO port bit set/reset register, Address offset: 0x18*/
  volatile uint32_t LCKR;  /*!< GPIO port configuration lock register, Address offset:0x1C*/
  volatile uint32_t AFR[2];/*!< GPIO alternate function registers, Address offset: 0x20-0x24*/
} GPIO_TypeDef;
```

### 2.2.1. Configurations des port

Donnez le code de configuration des pin PA7 à PA0 tel que :
PA3 à PA0 configurés en entrée avec pull-up (résistance de rappel haute)
PA7 à PA4 configurés en sortie 'open-drain' sans résistance de rappel

### 2.2.2. Détermination des touches

**1.** Donnez le code C permettant de placer la ligne L2 à 0 et les ligne L1, L3 L4 à la haute impédance

**2.** Donnez le code C permettant de déterminer si une la colonne C3 est à 0

**3.** Donnez le code d'une fonction ne prenant pas de paramètre et renvoyant le code ascii de la touche appuyée. On rappelle que le code ascii est donné par le caractère entre cote (e.g. var = 'A' , stocke le code ascii du A dans var).

Note : En mode open-drain, lorsqu'un 0 est placé sur le bit i du registre ODR, la pin i correspondante est à 0, si c'est un1 qui est placé sur ce bit, la pin correspondante est à la haute impédance (équivalent à un interrupteur ouvert, i.e. pas de potentiel défini par le microcontrôleur)

## 2.3. Annexes

### 8.4.1 GPIO port mode register (GPIOx_MODER) (x = A..E and H)

Address offset: 0x00

Reset values:

- 0x0C00 0000 for port A
- 0x0000 0280 for port B
- 0x0000 0000 for other ports

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| MODER15[1:0] | | MODER14[1:0] | | MODER13[1:0] | | MODER12[1:0] | | MODER11[1:0] | | MODER10[1:0] | | MODER9[1:0] | | MODER8[1:0] | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| MODER7[1:0] | | MODER6[1:0] | | MODER5[1:0] | | MODER4[1:0] | | MODER3[1:0] | | MODER2[1:0] | | MODER1[1:0] | | MODER0[1:0] | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bits 2y:2y+1 **MODERy[1:0]:** Port x configuration bits (y = 0..15)
These bits are written by software to configure the I/O direction mode.
00: Input (reset state)
01: General purpose output mode
10: Alternate function mode
11: Analog mode

### 8.4.2 GPIO port output type register (GPIOx_OTYPER) (x = A..E and H)

Address offset: 0x04

Reset value: 0x0000 0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| OT15 | OT14 | OT13 | OT12 | OT11 | OT10 | OT9 | OT8 | OT7 | OT6 | OT5 | OT4 | OT3 | OT2 | OT1 | OT0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **OTy**: Port x configuration bits (y = 0..15)
These bits are written by software to configure the output type of the I/O port.
0: Output push-pull (reset state)
1: Output open-drain

### 8.4.4 GPIO port pull-up/pull-down register (GPIOx_PUPDR) (x = A..E and H)

Address offset: 0x0C

Reset values:

- 0x6400 0000 for port A
- 0x0000 0100 for port B
- 0x0000 0000 for other ports

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| PUPDR15[1:0] | | PUPDR14[1:0] | | PUPDR13[1:0] | | PUPDR12[1:0] | | PUPDR11[1:0] | | PUPDR10[1:0] | | PUPDR9[1:0] | | PUPDR8[1:0] | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| PUPDR7[1:0] | | PUPDR6[1:0] | | PUPDR5[1:0] | | PUPDR4[1:0] | | PUPDR3[1:0] | | PUPDR2[1:0] | | PUPDR1[1:0] | | PUPDR0[1:0] | |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bits 2y:2y+1 **PUPDRy[1:0]:** Port x configuration bits (y = 0..15)
These bits are written by software to configure the I/O pull-up or pull-down
00: No pull-up, pull-down
01: Pull-up
10: Pull-down
11: Reserved

### 8.4.5 GPIO port input data register (GPIOx_IDR) (x = A..E and H)

Address offset: 0x10

Reset value: 0x0000 XXXX (where X means undefined)

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| IDR15 | IDR14 | IDR13 | IDR12 | IDR11 | IDR10 | IDR9 | IDR8 | IDR7 | IDR6 | IDR5 | IDR4 | IDR3 | IDR2 | IDR1 | IDR0 |
| r | r | r | r | r | r | r | r | r | r | r | r | r | r | r | r |

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **IDRy**: Port input data (y = 0..15)
These bits are read-only and can be accessed in word mode only. They contain the input value of the corresponding I/O port.

### 8.4.6 GPIO port output data register (GPIOx_ODR) (x = A..E and H)

Address offset: 0x14

Reset value: 0x0000 0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Reserved | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| ODR15 | ODR14 | ODR13 | ODR12 | ODR11 | ODR10 | ODR9 | ODR8 | ODR7 | ODR6 | ODR5 | ODR4 | ODR3 | ODR2 | ODR1 | ODR0 |
| rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw | rw |

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **ODRy**: Port output data (y = 0..15)
These bits can be read and written by software.
*Note: For atomic bit set/reset, the ODR bits can be individually set and reset by writing to the GPIOx_BSRR register (x = A..E and H).*

### 8.4.7 GPIO port bit set/reset register (GPIOx_BSRR) (x = A..E and H)

Address offset: 0x18

Reset value: 0x0000 0000

| 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| BR15 | BR14 | BR13 | BR12 | BR11 | BR10 | BR9 | BR8 | BR7 | BR6 | BR5 | BR4 | BR3 | BR2 | BR1 | BR0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| BS15 | BS14 | BS13 | BS12 | BS11 | BS10 | BS9 | BS8 | BS7 | BS6 | BS5 | BS4 | BS3 | BS2 | BS1 | BS0 |
| w | w | w | w | w | w | w | w | w | w | w | w | w | w | w | w |

Bits 31:16 **BRy:** Port x reset bit y (y = 0..15)

These bits are write-only and can be accessed in word, half-word or byte mode. A read to these bits returns the value 0x0000.

0: No action on the corresponding ODRx bit

1: Resets the corresponding ODRx bit

*Note: If both BSx and BRx are set, BSx has priority.*

Bits 15:0 **BSy:** Port x set bit y (y= 0..15)

These bits are write-only and can be accessed in word, half-word or byte mode. A read to these bits returns the value 0x0000.

0: No action on the corresponding ODRx bit

1: Sets the corresponding ODRx bit