Université Sorbonne Paris Nord

# COMPTE RENDU :
# TP3 ET TP4

SAE13

Mme Touba

Dreano Lucas
BUT R&T1

Mr. LEE

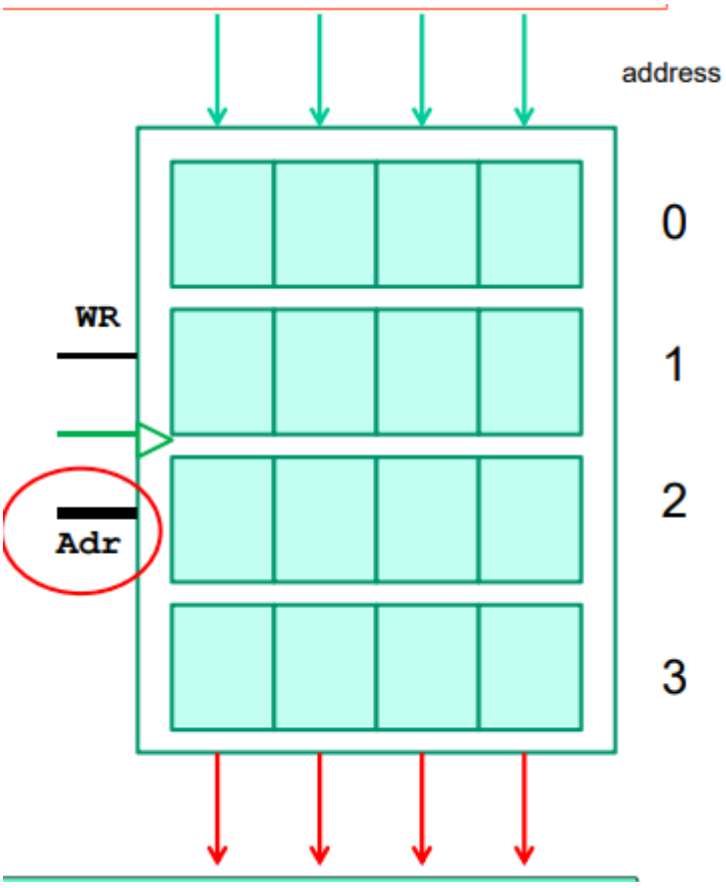
# TABLE DES MATIERES

## Table des matières

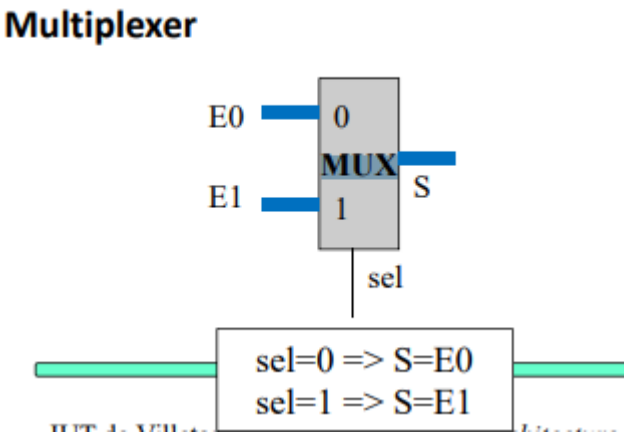
# INTRODUCTION

Un microprocesseur est composé de 4 composants principaux avec chacun un rôle bien précis,

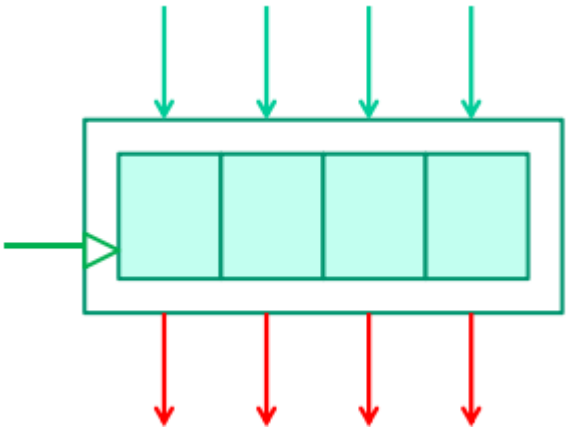
1. Le banc de registre : Il est composé de 4 registre 6 bits. Il sert à enregistrer des valeurs sur 4 bits avec 2 bits d'adresses donc de 00 à 11 pour les adresses et de 0000 à 1111 pour les valeurs contenues. On utilisera le paramètres WR pour écrire ou lire dans le registre et le paramètres ADR pour sélectionner sur quelle adresse on ecrira/lira



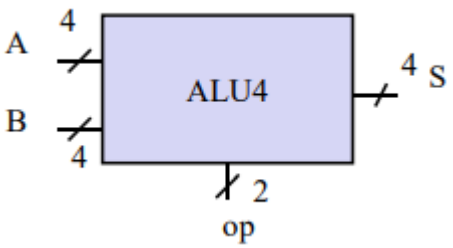
2. Le multiplexeur 2:1 : Il permet d'envoyer vers l'ALU soit une valeur d'une adresse du banc de registre soit une valeur définie par un signal de control. Il contient le paramètre Sel qui est utilisé pour sélectionner l'entrée E0 ou E1 du MUX.



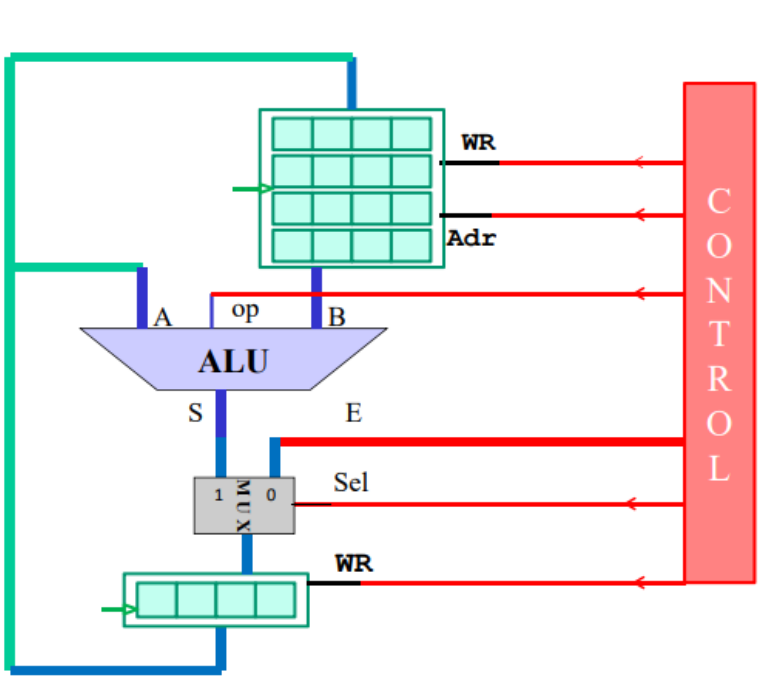

3. Le registre accumulateur : Il permet de charger des nombres depuis l'ALU ou encore d'écrire des résultats d'opération. Il possède le paramètre WR.



4. Unité Arithmétique et Logique: Son but est de réaliser des opérations selon son paramètres op, si op=0 on réalise une addition si op=1 alors on réalise une soustraction, ces opérations se réalise soit sur la valeur du registre accumulateur et une valeur du multiplexeur. Il y'a 2 valeurs 4 bits en entrée, un paramètre OP sur 2 bits et une sortie sur 4 bits




Alors on retrouve la structure complète du microprocesseur 4 bits :



# PARTIE I : TP3

## A. Ecriture des valeurs dans le banc de registre

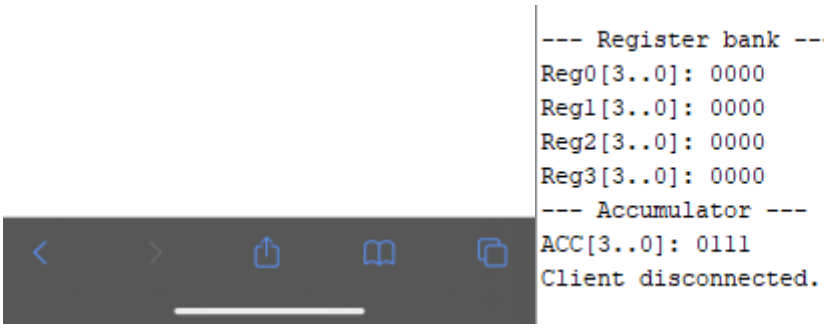
Tout d'abord pour réaliser les actions d'additions nous allons former un tableau de commande pour réaliser les opérations

| Code opératoire | Paramètre | WR (banc) | WR(ACC) | Adr | OP | Sel |
|---|---|---|---|---|---|---|
| 11 | 0111 | 0 | 1 | X | X | 0 |
| 10 | XXXX | 1 | 0 | 00 | X | X |
| 11 | 0110 | 0 | 1 | X | X | 0 |
| 10 | XXXX | 1 | 0 | 10 | X | X |
| 11 | 0011 | 0 | 1 | X | X | 0 |
| 10 | XXXX | 1 | 0 | 01 | X | X |
| 11 | 0111 | 0 | 1 | X | X | 0 |

| | | | | | | |
|---|---|---|---|---|---|---|
| 01 | X | 0 | 1 | 10 | 1 | 1 |
| 00 | X | 0 | 1 | 01 | 0 | 1 |

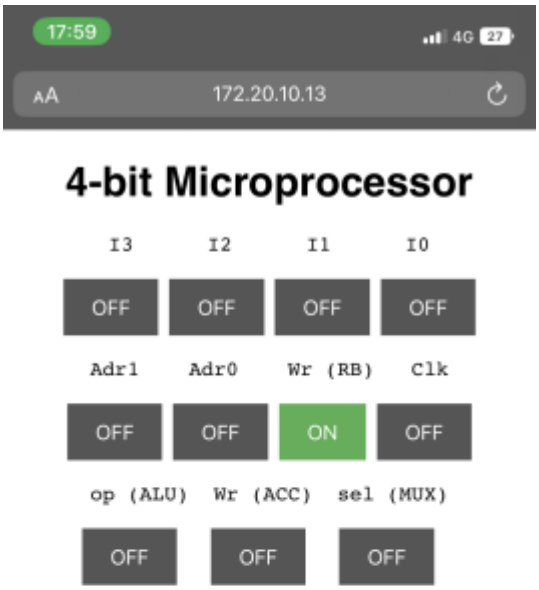
L'objectif est de réaliser 7-6+3 :

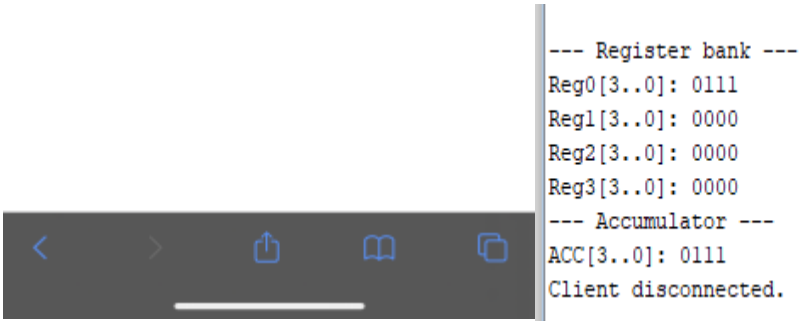
On écrit sur le registre accumulateur le nombre 7 donc 0111

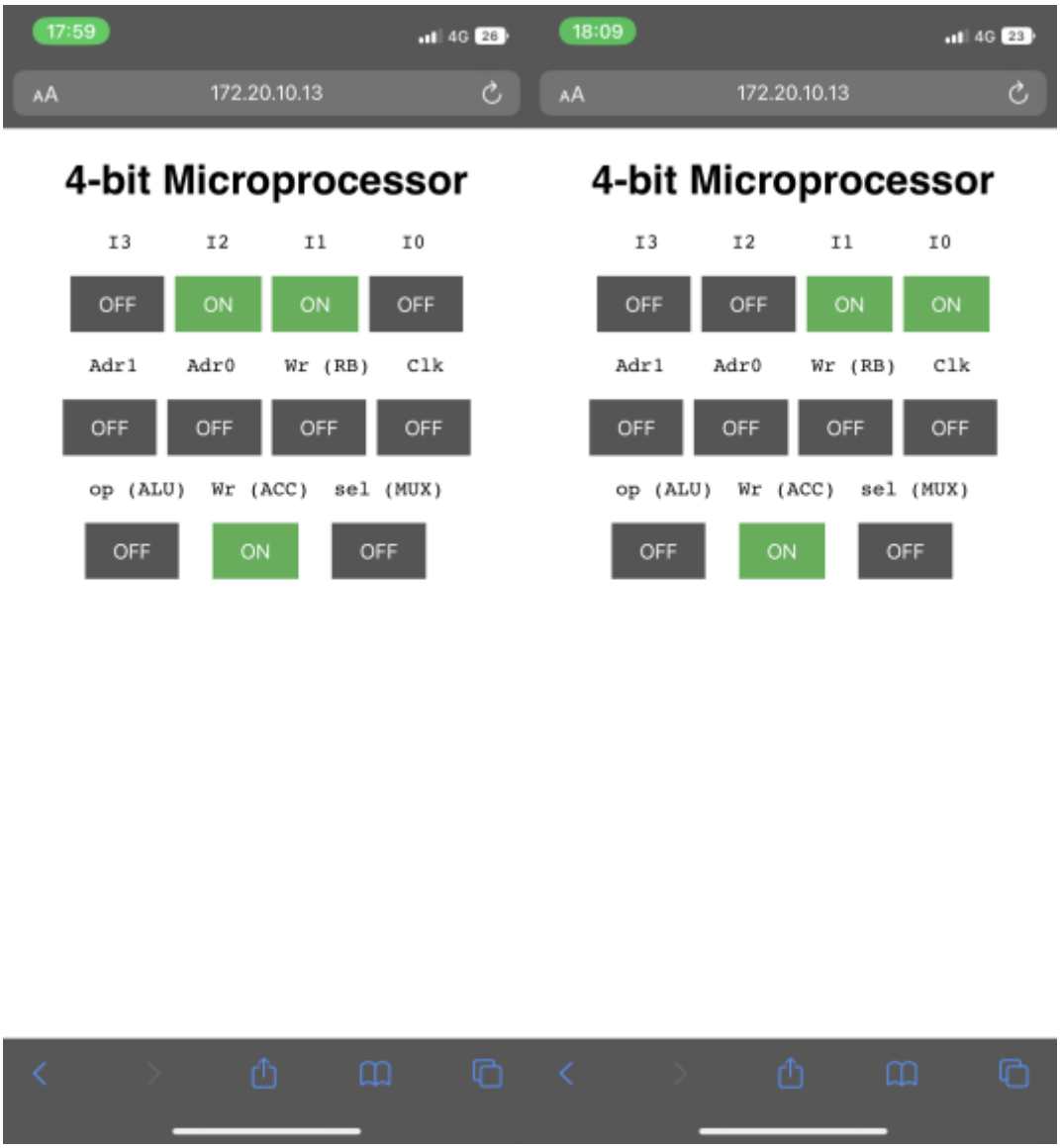

On obtient alors bien 7 dans le registre accumulateur sur le moniteur série.

On copie ensuite cette meme  valeur à l'adresse 00 du registre accumulateur




On écrit maintenant 6 donc 0110 sur le registre accumulateur puis on copie cette valeur à l'adresse 10, on réitère avec 3, 0011 à l'adresse 01 du banc de registre :




Toutes les valeurs sont maintenant ecrites correctement. Nous pouvons donc passer au calcul de ces valeurs.

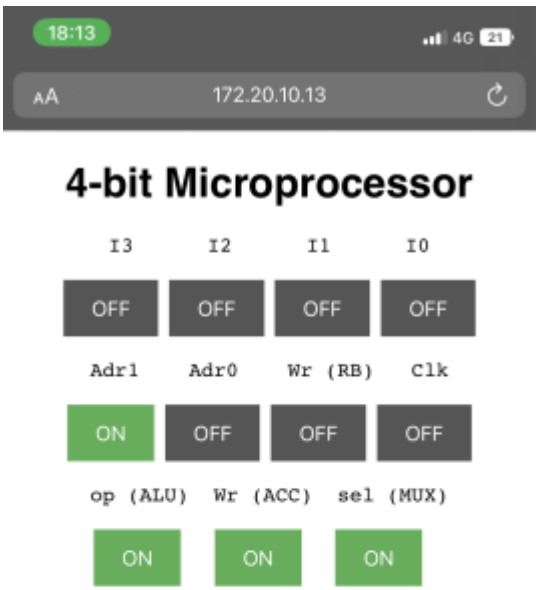
```
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0011
Reg2[3..0]: 0110
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0111
Client disconnected.
```

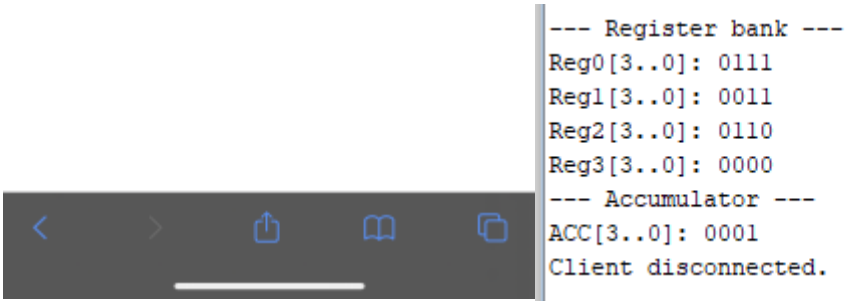
## B. Réalisation des calculs

On réalise maintenant l'opération 7-6, en mettant d'abord 7 dans le registre accumulateur donc WR(ACC)=1 I1=1 I2=1

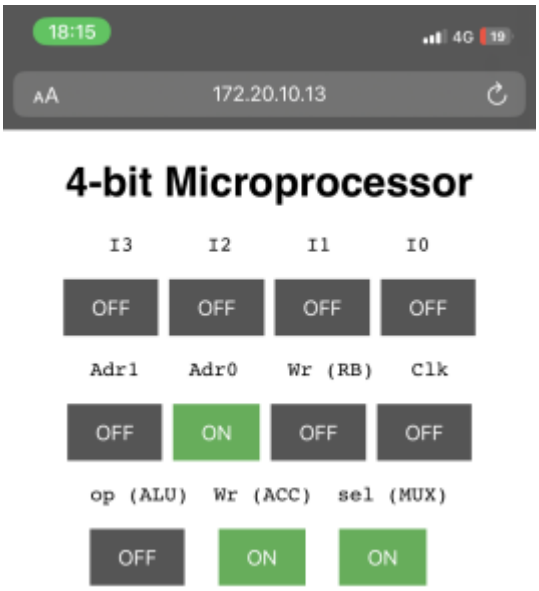
I0=1

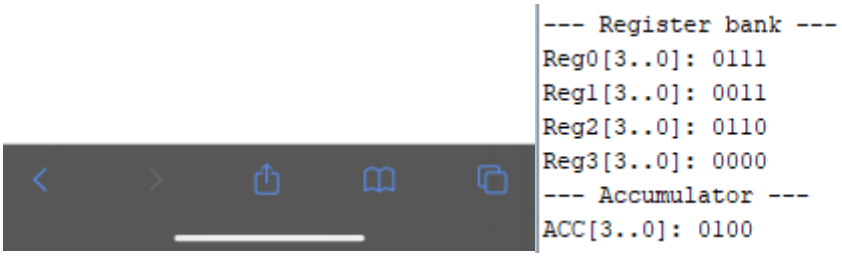
Puis on soustrait alors 6 écrit à l'adresse 10 du banc de registre et on obtient donc 0001, pour cela on active le sel=1 pour pouvoir prendre une valeur du banc de registre et OP(ALU)=1 pour soustraire :



On ajoute maintenant 3 donc on configure OP=0 pour additionner et on sélectionne l’adresse  01 :




On obtient donc 0100 le résultat de 7-6+3 c’est-à-dire 4.

# PARTIE II : TP 4

## A. Ecriture des valeurs dans le banc de registre

Pour ce TP on envoie directement des instructions en binaires sur 6 bits compose de 2 bits de codes opératoires et 4 bits de paramètres, le paramètre selon le code opératoire pourra influer sur l'adresse choisi, le nombre à écrire dans le registre accumulateur ou encore sur l'opération à effectuer dans le meme cadre que le tp 3 nous allons realiser l'opération 7-6+3

Rappel des codes operatoires:

- 00 : Addition
- 01 : soustraction
- 10 : Copie chargement accumulateur banc de registre
- 11 : Chargement accumulateur

Nous allons tout d'abord charger 7 dans le registre accumulateur avec le code operatoire 11 et le paramètre 0111

On envoie donc l'instruction 110111. Nous allons maintenant copier cette valeur à l'adresse 00 du registre accumulateur avec le code opératoire 10 l'adresse 00, on envoie donc l'instruction 1000.

```
New instruction received: 110111

Instruction: 110111
I[3..0]: 0111
Wr_ACC: 1
Wr_RB: 0
Adr: 01
op: 1
sel: 0
--- Register bank ---
Reg0[3..0]: 0000
Reg1[3..0]: 0000
Reg2[3..0]: 0000
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0111
```

```
New instruction received: 1000

Instruction: 100011
I[3..0]: 0011
Wr_ACC: 0
Wr_RB: 1
Adr: 00
op: 0
sel: 0
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0000
Reg2[3..0]: 0000
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0111
```

On répète l'opération pour 6 à l'adresse 01 et 3 à l'adresse 10 avec dans l'ordre les instructions suivantes:

- 110110
- 1001
- 110011
- 1010

```
New instruction received: 110110

Instruction: 110110
I[3..0]: 0110
Wr_ACC: 1
Wr_RB: 0
Adr: 01
op: 1
sel: 0
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0000
Reg2[3..0]: 0000
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0110
```

```
New instruction received: 1001

Instruction: 100110
I[3..0]: 0110
Wr_ACC: 0
Wr_RB: 1
Adr: 01
op: 0
sel: 0
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0110
Reg2[3..0]: 0000
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0110
```

```
Instruction: 110011
I[3..0]: 0011
Wr_ACC: 1
Wr_RB: 0
Adr: 00
op: 1
sel: 0
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0110
Reg2[3..0]: 0000
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0011
```

```
New instruction received: 1010

Instruction: 101011
I[3..0]: 1011
Wr_ACC: 0
Wr_RB: 1
Adr: 10
op: 0
sel: 0
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0110
Reg2[3..0]: 0011
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0011
```

## B. Réalisation des calculs

Nous pouvons maintenant commencer les calculs, tout d'abords avec 7-6

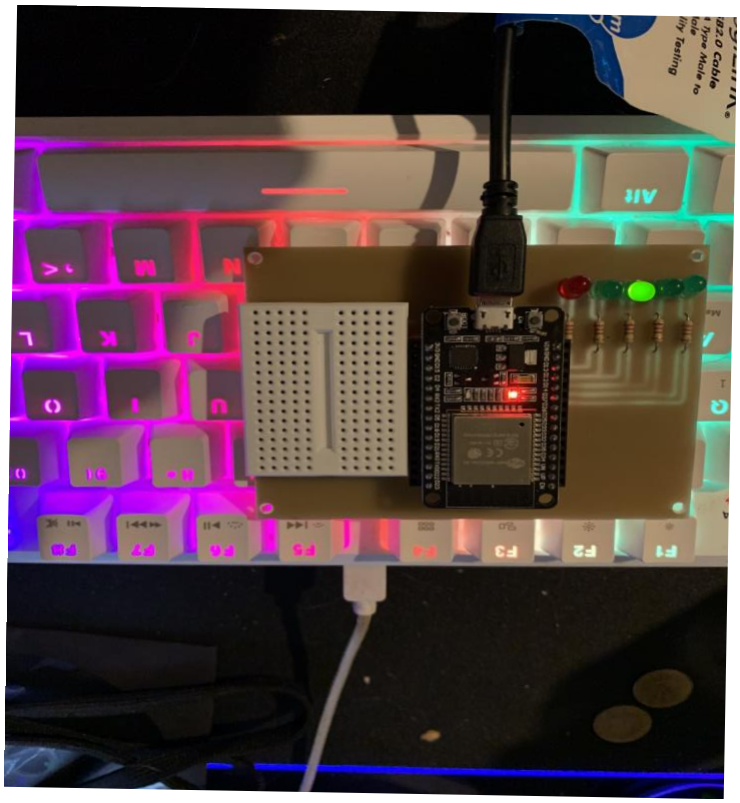
Nous allons recharger 7 dans l'accumulateur avec l'instruction 110111 puis effectuer une soustraction (01) avec 6 (adr 01) et donc envoyer l'instruction 0101


```
New instruction received: 0101

Instruction: 010111
I[3..0]: 0111
Wr_ACC: 1
Wr_RB: 0
Adr: 01
op: 1
sel: 1
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0110
Reg2[3..0]: 0011
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0001
```

On obtient donc bien 1 avec une LED allumée.

Enfin on envoie l'instruction 0010 pour additionner 1 et 3 et obtenir 4.

```
New instruction received: 0010

Instruction: 001011
I[3..0]: 1011
Wr_ACC: 1
Wr_RB: 0
Adr: 10
op: 0
sel: 1
--- Register bank ---
Reg0[3..0]: 0111
Reg1[3..0]: 0110
Reg2[3..0]: 0011
Reg3[3..0]: 0000
--- Accumulator ---
ACC[3..0]: 0100
```



# CONCLUSION

| Code opératoire | Paramètre | WR (banc) | WR(ACC) | Adr | OP | Sel |
|---|---|---|---|---|---|---|
| 11 | 0111 | 0 | 1 | X | X | 0 |
| 10 | XXXX | 1 | 0 | 00 | X | X |
| 11 | 0110 | 0 | 1 | X | X | 0 |
| 10 | XXXX | 1 | 0 | 10 | X | X |
| 11 | 0011 | 0 | 1 | X | X | 0 |
| 10 | XXXX | 1 | 0 | 01 | X | X |
| 11 | 0111 | 0 | 1 | X | X | 0 |
| 01 | X | 0 | 1 | 10 | 1 | 1 |
| 00 | X | 0 | 1 | 01 | 0 | 1 |

En somme nous avons pu réaliser ces instructions sous deux formes tout d'abords avec des boutons on/off pour chacun des paramètres et ensuite avec des instructions directement sous format binaire où il suffit de rentrer un code operatoire suivi de ses paramètres selon le code opératoire :

- Dans le cas d'un chargement accumulateur le nombre à charger

- Dans le cas d'une copie de l'accumulateur au banc de registre l'adresse où copier la valeur
- Dans le cas d'une soustraction/addition l'adresse du nombre à soustraire/additionner