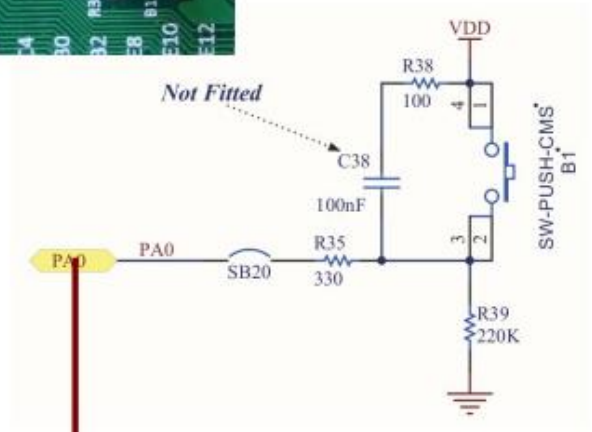
MICROS 32 BITS
STM - GPIO

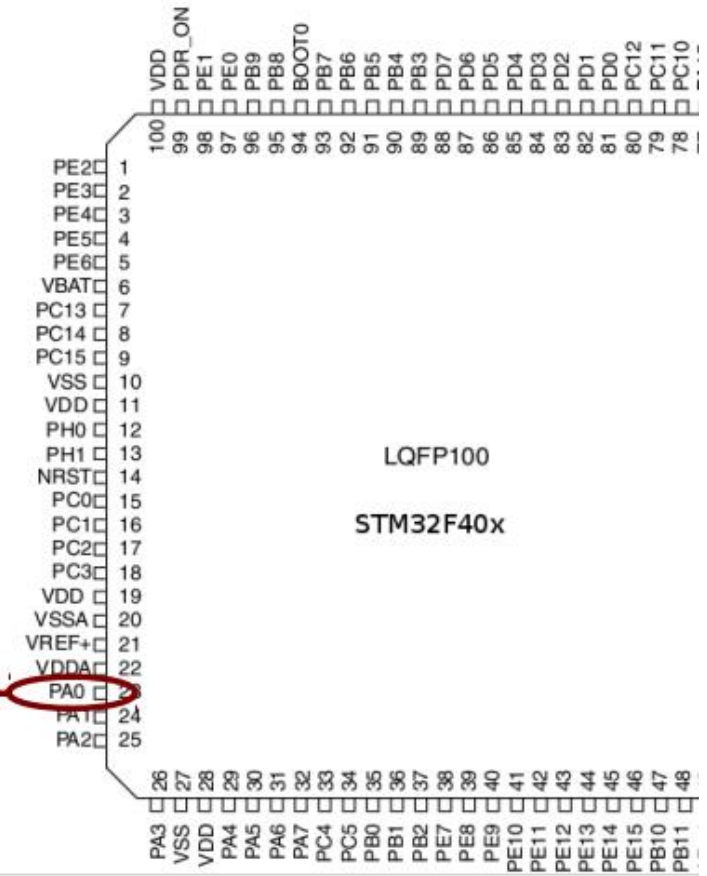
Realizada por: ROBINSON JIMENEZ MORENO

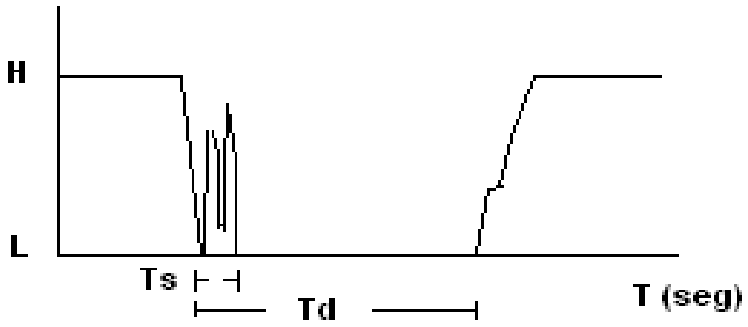
Profesora: LUISA FERNANDA GARCÍA

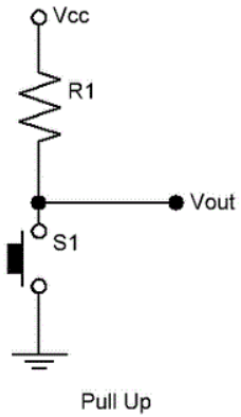
UNIVERSIDAD MILITAR NUEVA GRANADA

INGENIERÍA MECATRÓNICA UMNG

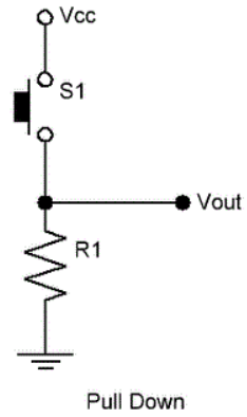




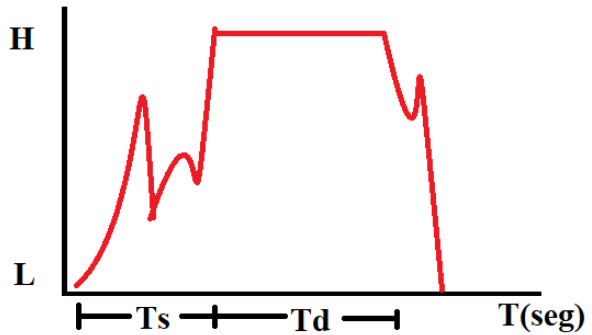




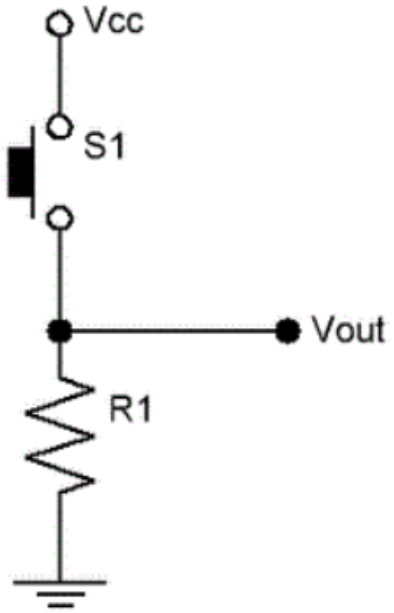
Pull Up



Pull Down

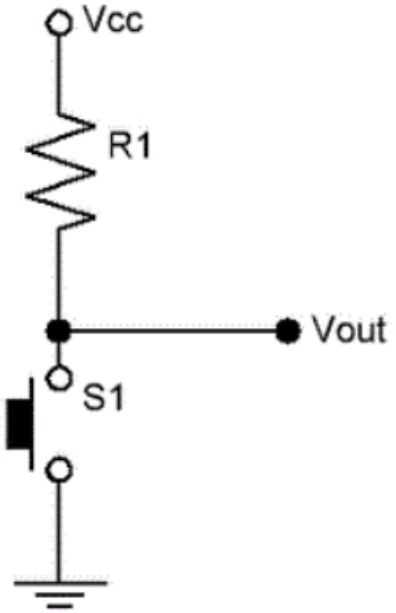


```c
#include "stm32f7xx.h"
int main(void){

  RCC -> AHB1ENR = 0X6; //HABILITAR LOS PUERTOS B Y C

  GPIOB -> MODER = 0X10004001; //
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 0x55555555; //VELOCIDAD MEDIA
  GPIOB -> PUPDR = 0X10004001; // NO PULL UP
  GPIOC -> MODER = 0; //


  while(1){
      if((GPIOC -> IDR&0X2000)==0X2000){
      for(int i=0;i< 100000; i++);
      GPIOB -> ODR ^=0x4081;
          }
    }
  }
```



Pull Down

```c
#include "stm32f7xx.h"
int main(void){

  RCC -> AHB1ENR = 0X6; //HABILITAR LOS PUERTOS B Y C

  GPIOB -> MODER = 0X10004001; //
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 0x55555555; //VELOCIDAD MEDIA
  GPIOB -> PUPDR = 0X10004001; // NO PULL UP
  GPIOC -> MODER = 0; //

  //RESISTENCIA EN PULL-DOWN
  while(1){
      if((GPIOC -> IDR&0X2000)==0){
      for(int i=0;i< 100000; i++);
      GPIOB -> ODR ^=0x4081;
          }
    }
  }
```

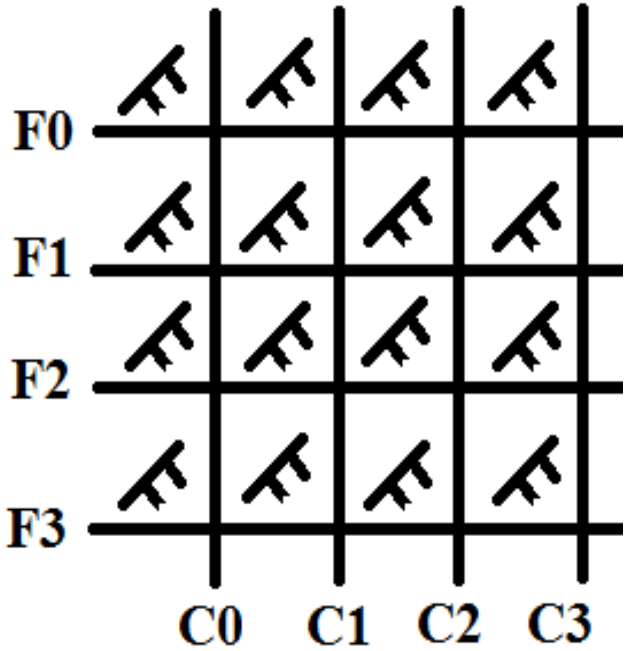


Pull Up

```c
#include "stm32f7xx.h"
int main(void){
  int a=0;
  RCC -> AHB1ENR = 0X6; //HABILITAR LOS PUERTOS B Y C

  GPIOB -> MODER = 0X10004001; //
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 0x55555555; //VELOCIDAD MEDIA
  GPIOB -> PUPDR = 0X10004001; // NO PULL UP
  GPIOC -> MODER = 0; //

  //RESISTENCIA EN PULL-DOWN
  while(1){
      while((GPIOC -> IDR&0X2000)==0X2000){
      for(int i=0;i< 100000; i++);
      a++;
          }
    }
  }
```

```c
#include "stm32f7xx.h"
int main(void){
  int a=0;
  RCC -> AHB1ENR = 0X6; //HABILITAR LOS PUERTOS B Y C

  GPIOB -> MODER = 0X10004001; //
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 0x55555555; //VELOCIDAD MEDIA
  GPIOB -> PUPDR = 0X10004001; // NO PULL UP
  GPIOC -> MODER = 0; //

  //RESISTENCIA EN PULL-DOWN
  while(1){
      if((GPIOC -> IDR&0X2000)==0X2000){
      for(int i=0;i< 100000; i++);
        while((GPIOC -> IDR&0X2000)==0X2000);
      a++;
          }
    }
  }
```

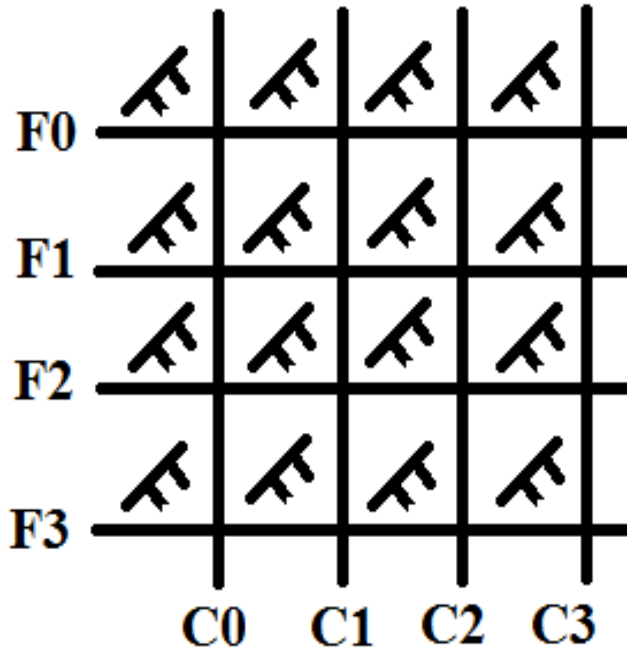
TECLADO MATRICIAL



Barrido matricial.

Evaluar celda por celda para determinar si se ha activado alguna.

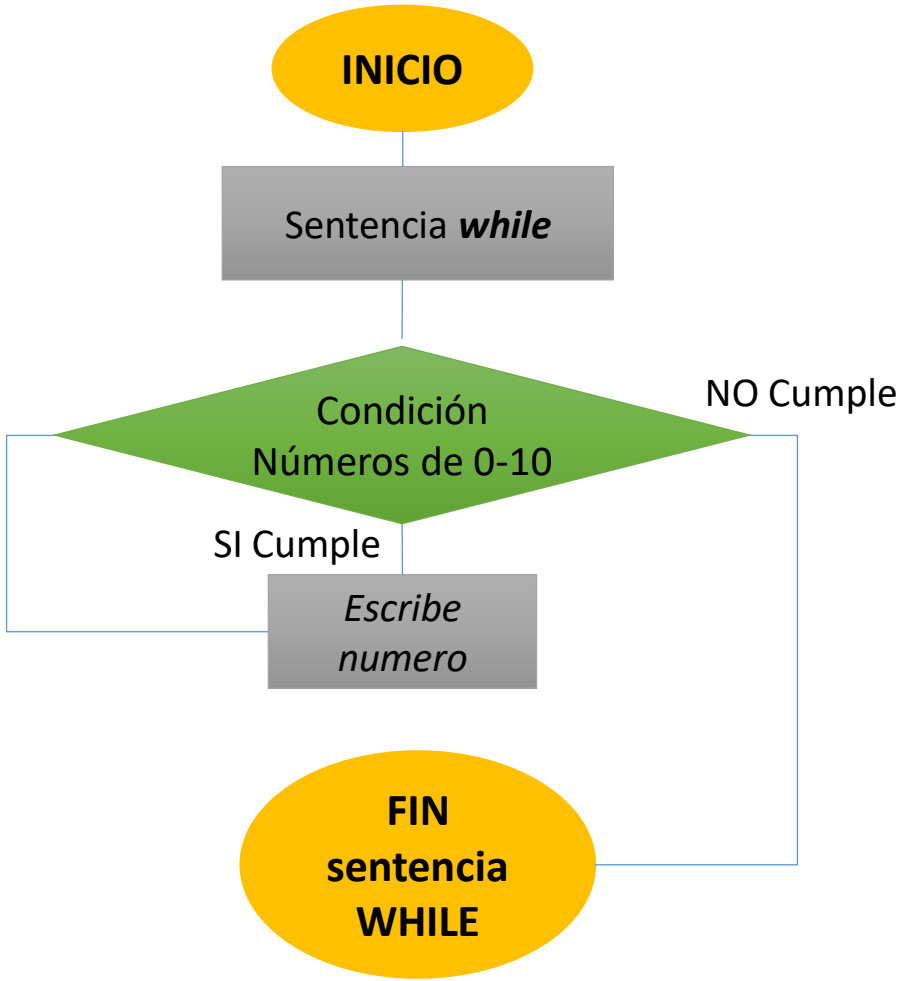
Ejemplo: F1C1 LHLL LHLL



Debemos hacer la tabla de relación de Filas Columnas y tecla

f0 f1 f2 f3 c0 c1 c2 c3	Código	Significado
1 0 0 0 0 0 0 0	128	act f0
0 1 0 0 0 0 0 0	64	act f1
0 0 1 0 0 0 0 0	32	act f2
0 0 0 1 0 0 0 0	16	act f3

```
#include <stdio.h>
#include "stm32f7xx.h"
int numeros[10]={126,48,109,121,51,91,95,112,127,123};
int a,cl;
int teclado(void){
GPIOB -> ODR  =128;
for(int i = 0; i < 100000; i++){};
cl=(GPIOB -> IDR)&0X000000FF;
if(cl==129){return ;}
else if(cl==130){return ;}
else if(cl==132){return ;}
else if(cl==136){return ;}
GPIOB -> ODR  =64;
for(int i = 0; i < 100000; i++){};
cl=(GPIOB -> IDR)&0X000000FF;
if(cl==65){return ;}
else if(cl==66){return ;}
else if(cl==68){return ;}
else if(cl==72){return ;}
GPIOB -> ODR  =32;
for(int i = 0; i < 100000; i++){};
cl=(GPIOB -> IDR)&0X000000FF;
if(cl==33){return ;}
else if(cl==34){return ;}
else if(cl==36){return ;}
else if(cl==40){return ;}
GPIOB -> ODR  =16;
for(int i = 0; i < 100000; i++){};
cl=(GPIOB -> IDR)&0X000000FF;
if(cl==13){return ;}
else if(cl==18){return ;}
else if(cl==20){return ;}
else if(cl==24){return ;}
}
```




```
int main(void){

  RCC -> AHB1ENR = 0X6; //HABILITAR LOS PUERTOS B Y C

  GPIOB -> MODER = 0X00005500; //
  GPIOB -> OTYPER = 0X0; //PUSH PULL
  GPIOB -> OSPEEDR = 00005555; //VELOCIDAD MEDIA
  GPIOB -> PUPDR = 0; // NO PULL UP

    GPIOC -> MODER = 0X00005555; //
  GPIOC -> OTYPER = 0X0; //PUSH PULL
  GPIOC -> OSPEEDR = 00005555; //VELOCIDAD MEDIA
  GPIOC -> PUPDR = 0; // NO PULL UP

  while(1){

      for( a = 0; a < 10; a++){};

        a=teclado();
        GPIOC-> ODR ^= numeros[a];

    }
  }
```

TAREA

Desarrollar un programa que sume o multiplique dos números ingresados por teclado matricial y muestre el resultado por un puerto en que hay 8 leds de visualización.