Teclados matriciales

Informática II - R2004
2021

Recordando lo desarrollado para un teclado de 4 teclas

```
void DriverTecladoSW(void)
{
    static uint8_t lastKey = NO_KEY;
    static uint8_t count;
    uint8_t key;

    key = DriverTecladoHW();

    if(lastKey == key && count < DEBOUNCE_COUNT)
    {
        count++;
        if(count == DEBOUNCE_COUNT)
        {
            Tecla = key;
        }
    }
    else if(lastKey != key)
    {
        count = 0;
    }
    lastKey = key;
}
```

```
uint8_t DriverTecladoHW( void )
{
    uint8_t Codigo = NO_KEY;

    if ( GetPIN ( KEY0 , BAJO ) )
        return SW1;

    if ( GetPIN ( KEY1 , BAJO ) )
        return SW4;

    if ( GetPIN ( KEY2 , BAJO ) )
        return SW7;

    if ( GetPIN ( KEY3 , BAJO ) )
        return SW10;

    return Codigo;
}
```

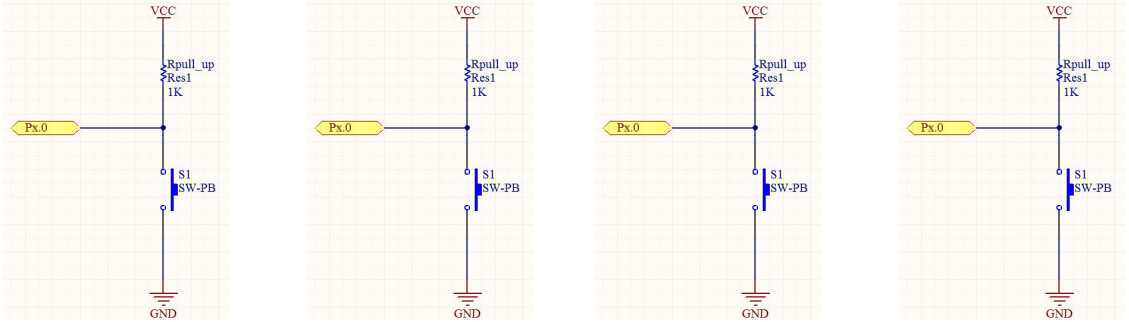
Diseñado de esta manera, el driver DriverTecladoSW realiza el antirebote del teclado, mientras que el DriverTecladoHW nos indica cuál es la tecla presionada. Si cambio de teclado deberé cambiar esta última función

Teclados matriciales

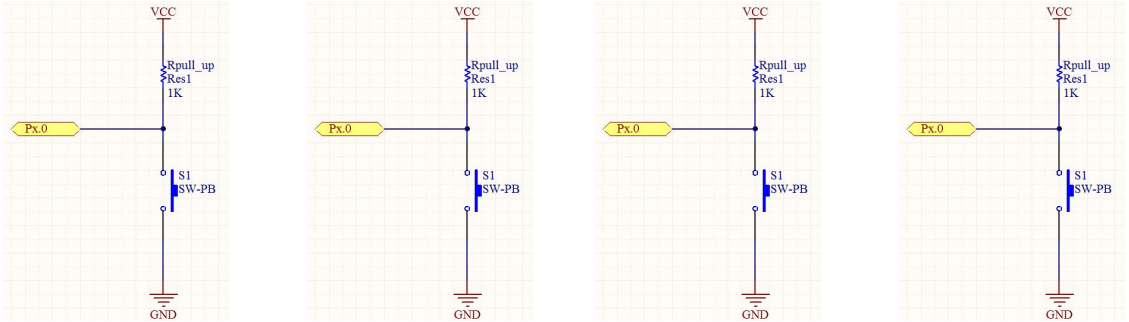
Cuando quiero aumentar la cantidad de teclas que utilizo, las opciones comerciales más comunes son la utilización de teclados matriciales. Los mismos se componen por un conjunto de pulsadores, cuya tecnología puede variar de un teclado a otro, y cuya conexión explicaremos a continuación:

Teclados - Esquema de atención a N teclas

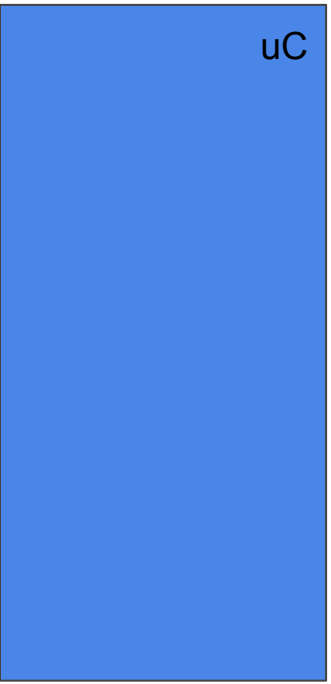
uC

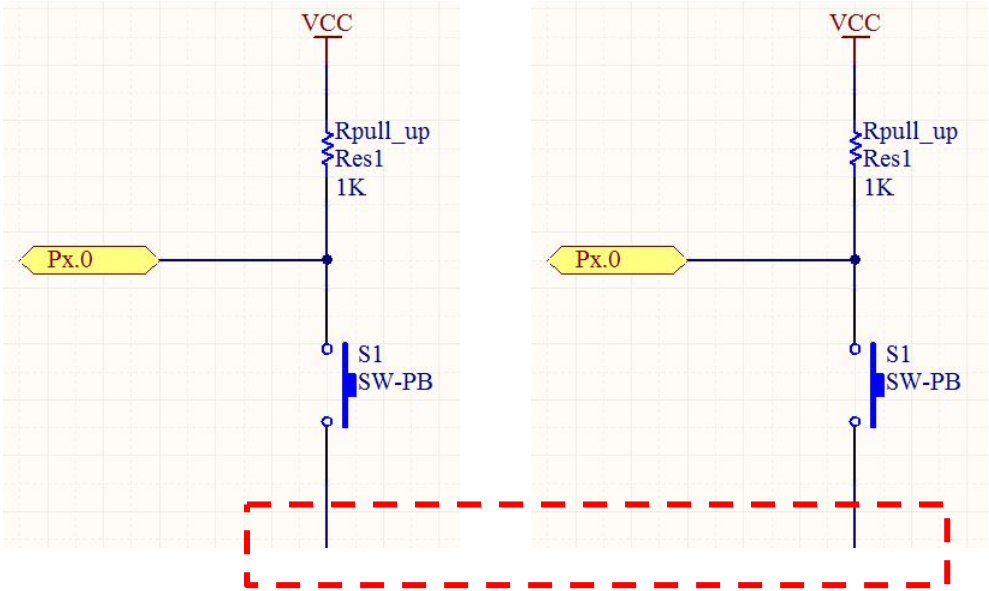


Con la conexión que utilizábamos para los pulsadores, cada tecla deberá tener un pin dedicado del micro para su lectura. No suele ser cómodo destinar tantos pines, considerando que sabemos que se presionará *DE A UNA TECLA POR VEZ* (cuando una entrada está activa, todo el resto están inactivas)



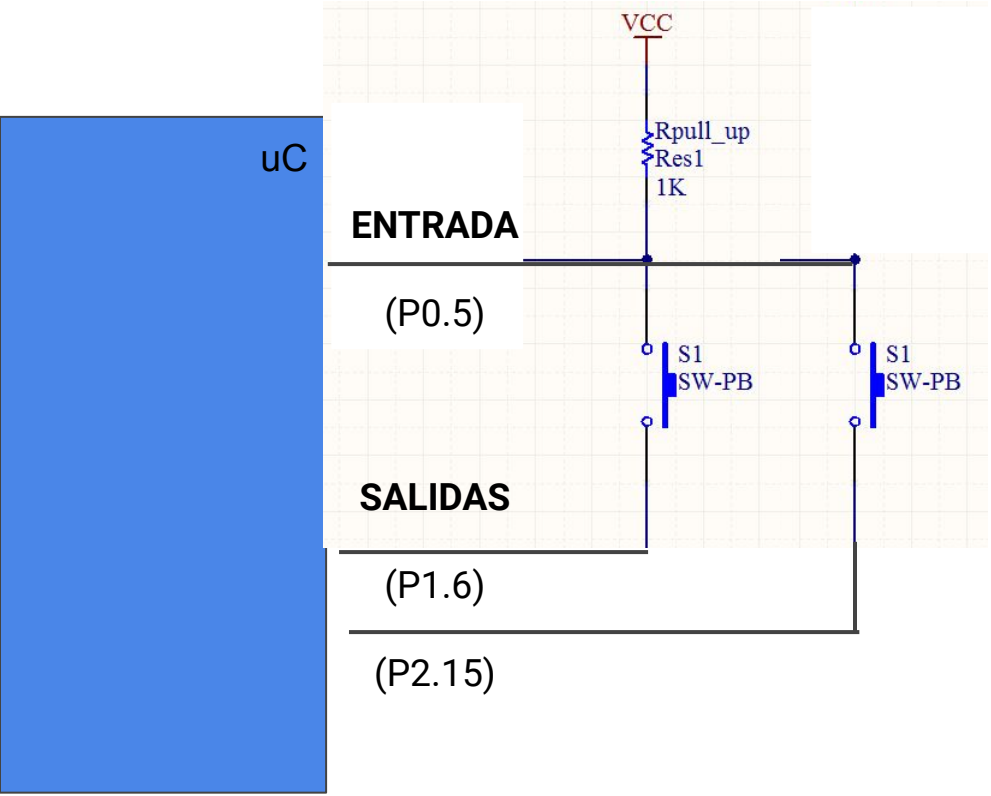
Lógica de selección y lectura de teclas





Si ponemos ahora nuestra atención en este pin del pulsador, vemos que en caso de conectarlo a masa (0 Volts - 0 lógico), el pulsador funciona normalmente. Ahora, si el mismo estuviese conectado a VCC (3.3Volts - 1 lógico), el pin Px.0 siempre estaría leyendo un 1, no importa si el pulsador está presionado o está suelto. Dicho de otra manera, el pulsador estaría ***DESACTIVADO***. ¿Qué pasaría si conecto entonces este pin a una salida del microcontrolador, y voy activando o desactivando cada pulsador a medida que quiero leerlo?

Lógica de selección y lectura de teclas

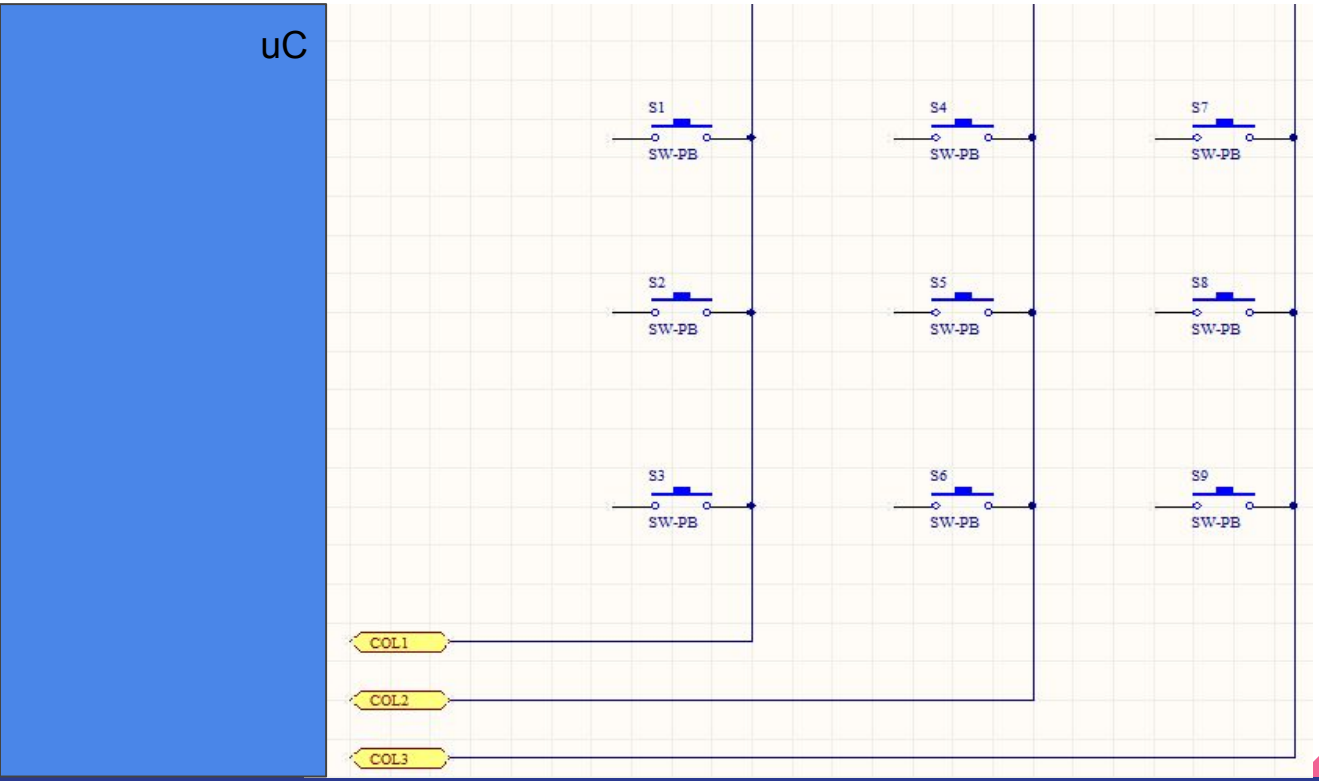


Si quiero leer el ***pulsador de la izquierda***, voy a poner un 0 en la salida P1.6 (Pulsador izq. activado) y un 1 en la salida P2.15 (Pulsador der. desactivado), y a continuación leo la entrada P0.5, para saber si el pulsador está presionado (leo un 0) o está suelto (leo un 1)

Si quiero leer el ***pulsador de la derecha***, voy a poner un 1 en la salida P1.6 (Pulsador izq. desactivado) y un 0 en la salida P2.15 (Pulsador der. activado), y a continuación leo la entrada P0.5, para saber si el pulsador está presionado (leo un 0) o está suelto (leo un 1)

Puedo desactivar ambos pulsadores poniendo un 1 en los dos, pero ***no puedo activar ambos pulsadores al mismo tiempo*** *(porque no sé que significa si leo un 0 en P0.5)*

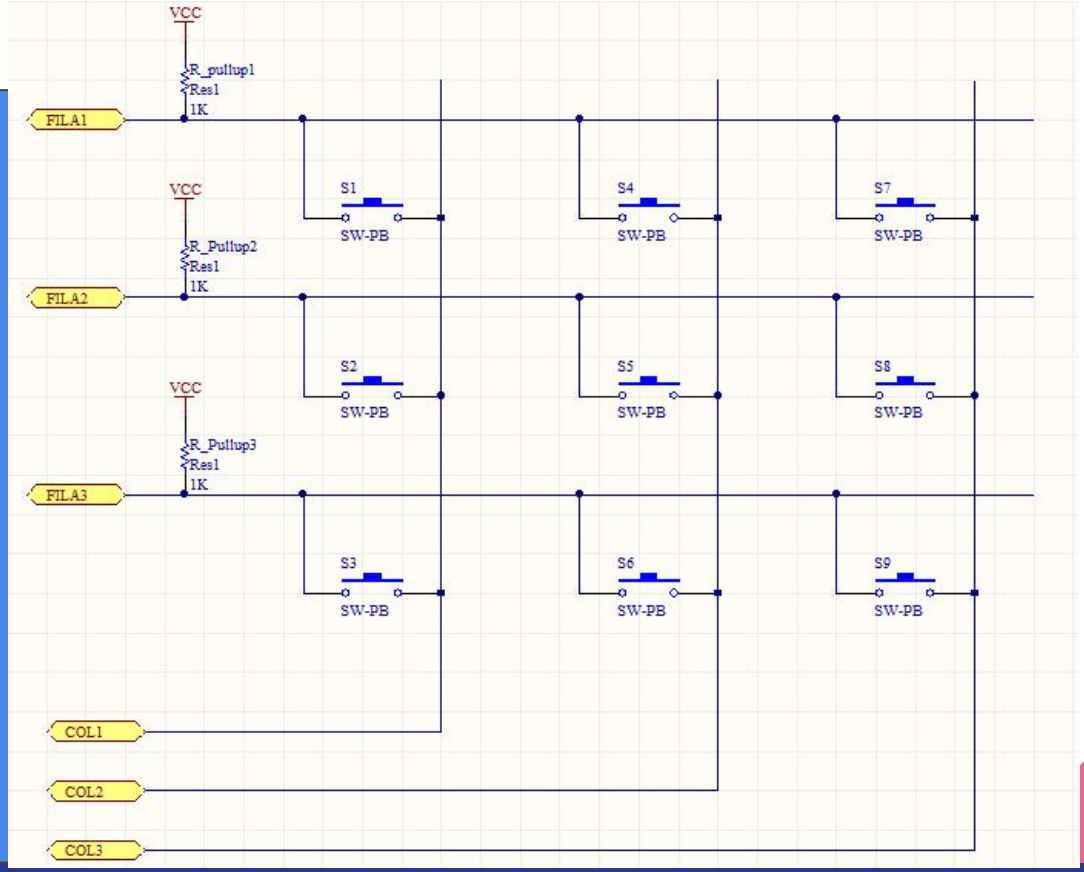
Teclados - Esquema matricial



Para que esto suponga un ahorro de pines, voy a usar la misma lógica de selección para elegir entre 3 COLUMNAS de pulsadores distintos. Recordemos que c***on un 0 ACTIVAMOS LA COLUMNA, y con un 1 la DESACTIVAMOS***

Teclados - Esquema matricial

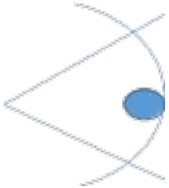
uC

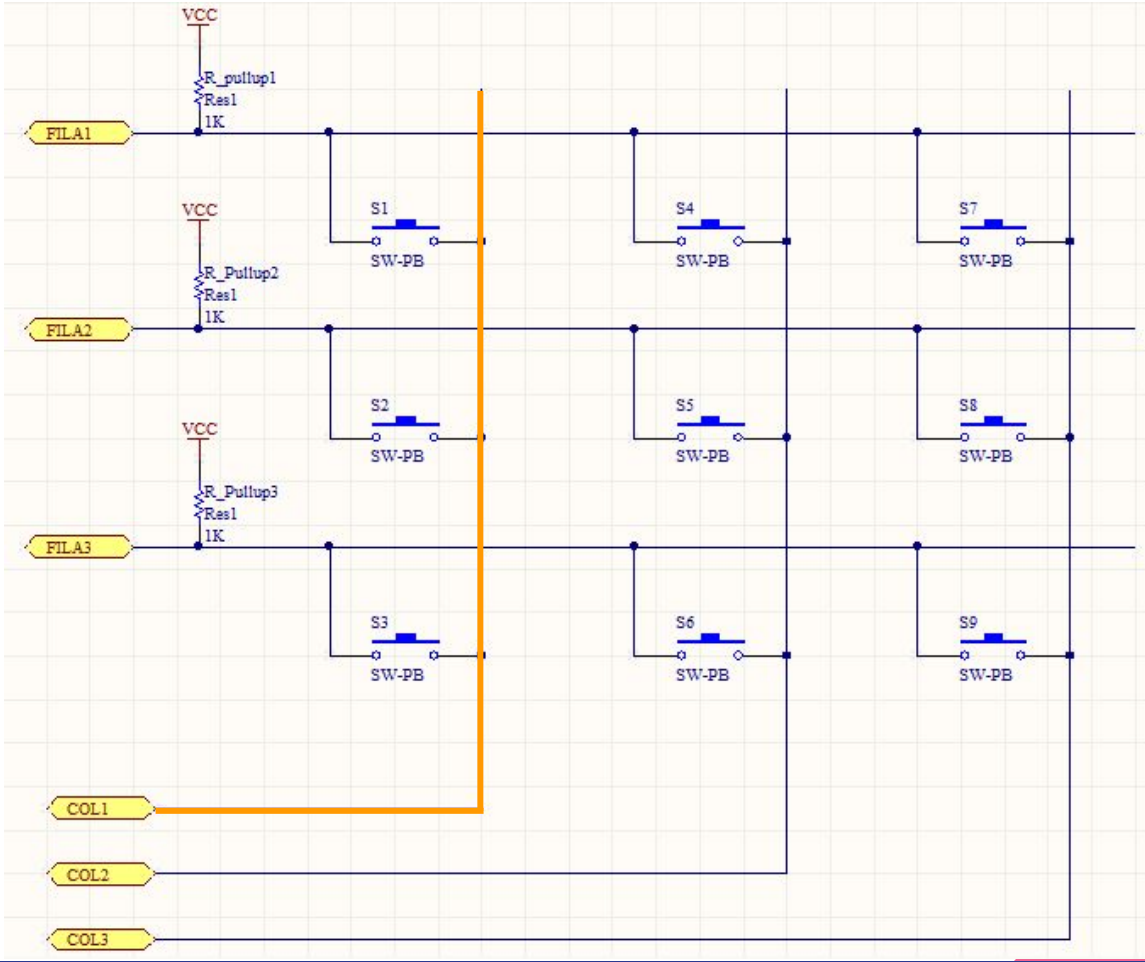


A su vez, ahora conectamos entre si A LA MISMA ENTRADA los 3 pulsadores de la primer fila, a OTRA ENTRADA los 3 de la segunda, y A UNA TERCERA ENTRADA los 3 pulsadores de la tercer fila.

¿Cómo funciona este esquema?

ENTRADAS





1

1

1

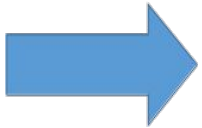
1. Selecciono la fila a leer

2. Me fijo si alguna tecla en esa fila esta presionada

3. Selecciono la siguiente fila.

4. Completo con todas las filas y vuelvo a empezar.

SALIDAS

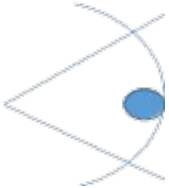


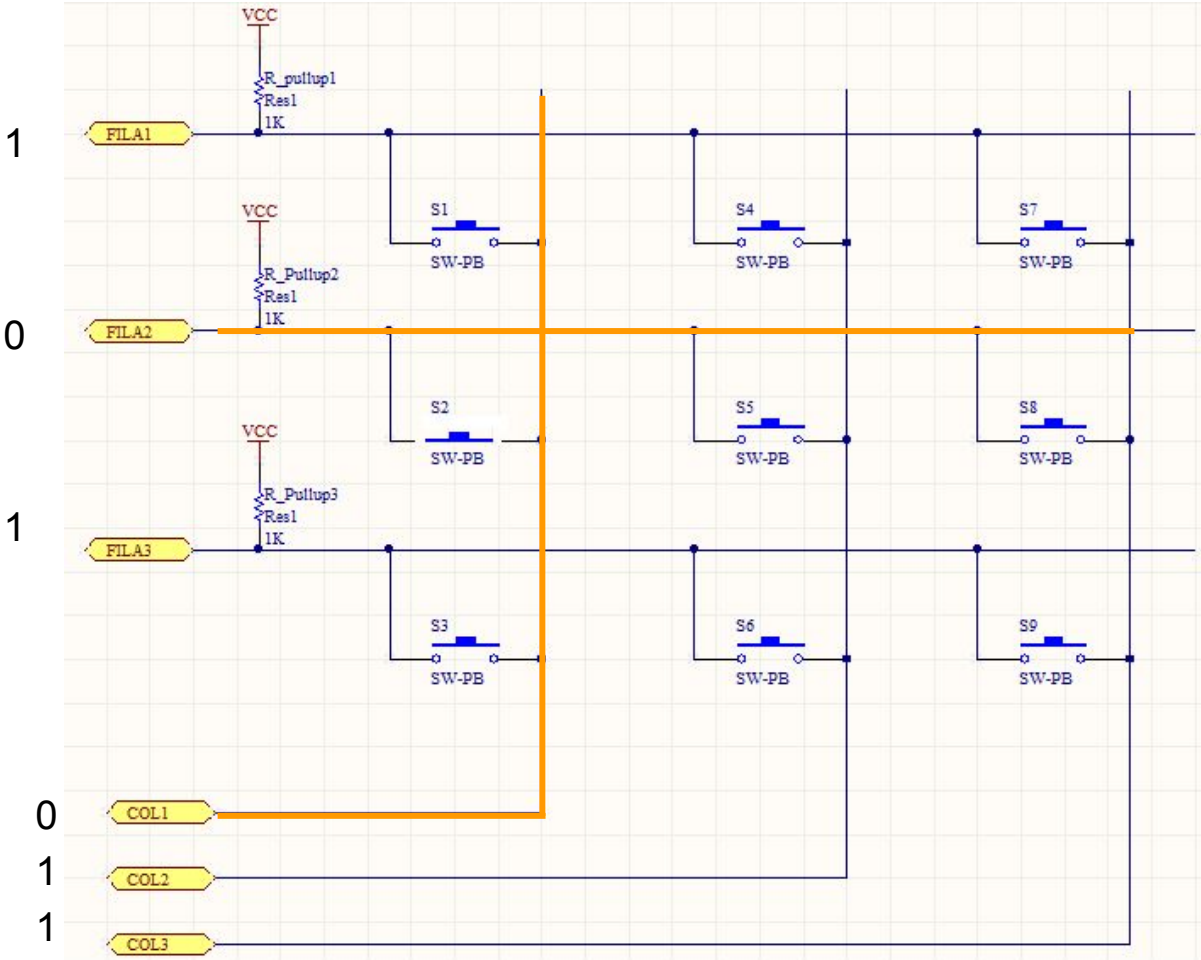
0

1

1

ENTRADAS





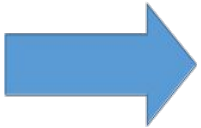
1.Selecciono la fila a leer

2. Me fijo si alguna tecla en esa fila esta presionada

3. Selecciono la siguiente fila.

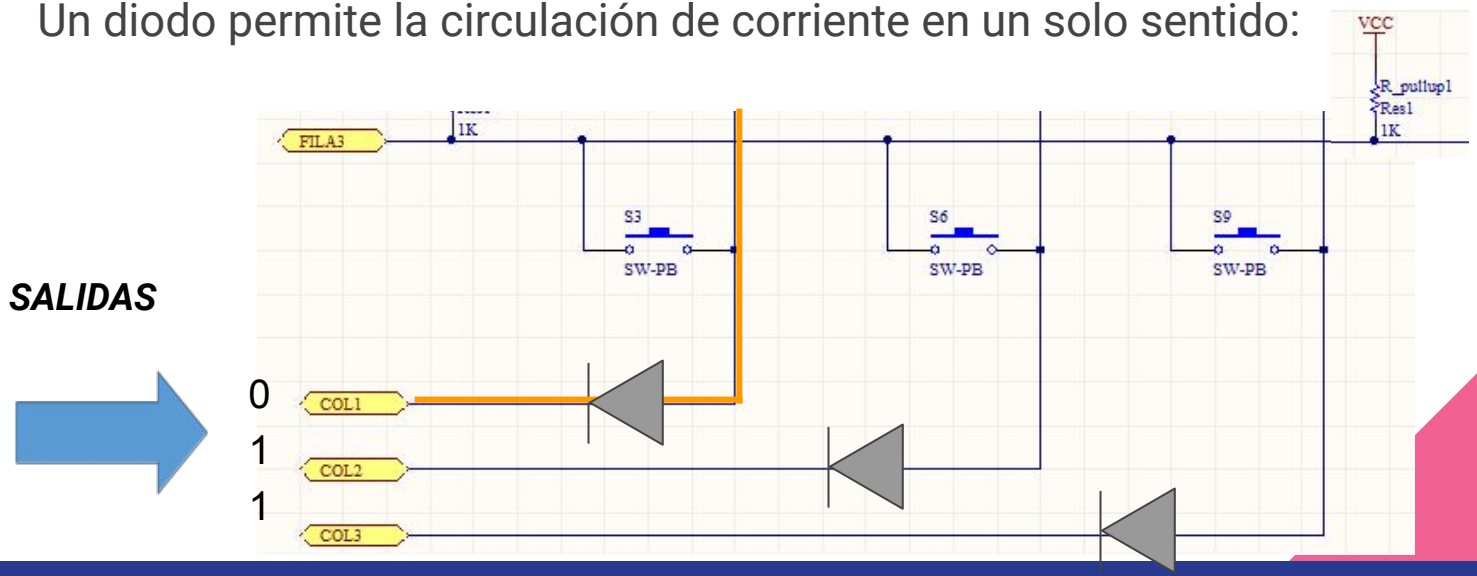
4. Completo con todas las filas y vuelvo a empezar.

SALIDAS



Algunas consideraciones:

Conectando los teclados de acuerdo a la imagen anterior, puede suceder que ante la opresión de 2 teclas en simultáneo se genere un cortocircuito entre los pines del microprocesador (estoy poniendo una SALIDA en 0V, y estoy poniendo OTRA salida en 3.3V, y las conecto entre sí. Esto, en la teoría, demanda una corriente infinita, y puede quemar un pin del puerto). Para evitar este problema, suelen ponerse diodos en las salidas digitales. Un diodo permite la circulación de corriente en un solo sentido:



Teclados matriciales - Ahorro de pines

Con este esquema, organizando los pulsadores en N filas y M columnas, necesito N + M pines. De esta manera, un teclado de 3x3 necesita 6 pines para manejar 9 pulsadores, uno de 4x4 necesita 8 pines para 16 pulsadores, y así sucesivamente.

Recordando las premisas originales, este esquema me sirve para teclados en los que se presione una tecla por vez, ya que no podré detectar la opresión de 2 teclas en simultáneo.

Ejemplos de este tipo de teclado se pueden encontrar en controles remotos, controles de acceso, tableros de control de máquinas o electrodomésticos y botoneras en general.

Desarrollamos el driver de teclado matricial

Recordando, nuestro driver debe:

- **Función InicializaciónTeclado():**

  Configurar los pines a donde conecto el teclado como GPIO, las columnas como salidas y las filas como entradas

- ***Función DriverTecladoHw():***
  a. Seleccionar una columna (poner un 0 en el pin correspondiente)
  b. Leer las diferentes filas para ver si hay alguna en 0. Si hay, devuelvo el valor de la tecla correspondiente.
  c. Seleccionar la columna siguiente
  d. Repetir la operación tantas veces como columnas tenga mi teclado.

- ***Función DriverTecladoSW():***
  Realiza el antirebote de la tecla seleccionada (ya está hecho)