



Universidad de Costa Rica  
Facultad de Ingeniería  
Escuela de Ingeniería Eléctrica  
**IE-0624 Laboratorio de Microcontroladores**

**EIE**

Escuela de  
Ingeniería Eléctrica

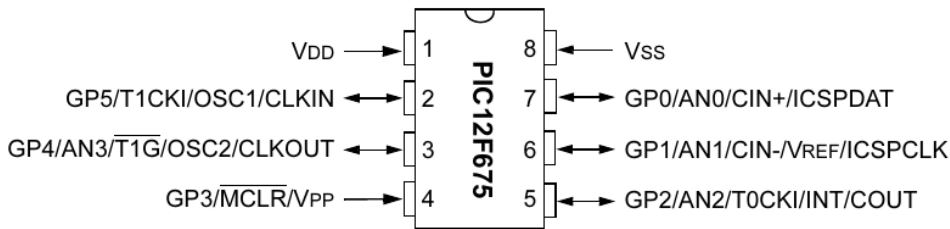
# GPIOs y el PIC12F629

MSc. Marco Villalta Fallas - `marco.villalta@ucr.ac.cr`

II Ciclo 2022

## Qué son los GPIOs?

- GPIO: *General Purpose Input Output*
- La gran mayoría de los pines en un MCU son GPIOs
- Pines que no son GPIOs: Vdd, Vss, MCLR (Reset), entradas de reloj, etc.
- Pines asignados a GPIOs comparten otras funciones
- Se agrupan en puertos (En el caso del PIC12F675 solo tiene un puerto)



## Como funciona un GPIO?

- Se operan muy similar entre ellos
- Generalmente se configura el funcionamiento de un GPIO a través de un registro
- Dependiendo del funcionamiento del MCU puede ser necesario modificar mas de un registro de configuración (**Importante: Leer la documentacion**)
- Configuración de registro permite indicar cuales son entradas o salidas.
- Si se configura como entrada se realiza la lectura con polling o por medio de interrupciones(este se vera mas adelante)
- Si se configura como salida, se maneja el estado en el programa (1:Alto/0:Bajo)

## Como se programa un GPIO en el PIC12F675?

En el PIC12F675 se utilizan varios registros para operarlos digitalmente.

- TRISIO: En este registro donde se indica el modo de operación de cada pin, un bit en alto(:1) lo pone como entrada y un bit en bajo(:0) lo pone como salida
- Con el registro GPIO se lee el estado del pin y se escribe al latch de salida
- Los registros ANSEL y/o CMCON debe de inicializarse para configurar un canal analógico como entrada digital.
- Dependiendo del pin se debe configurar el registro CONFIG.

**FIGURE 2-2: DATA MEMORY MAP OF THE PIC12F629/675**

File Address		File Address	
Indirect addr. <sup>(1)</sup>	00h	Indirect addr. <sup>(1)</sup>	80h
TMR0	01h	OPTION_REG	81h
PCL	02h	PCL	82h
STATUS	03h	STATUS	83h
FSR	04h	FSR	84h
GPIRQ	05h	TRISIO	85h
	06h		86h
	07h		87h
	08h		88h
	09h		89h
PCLATH	0Ah	PCLATH	8Ah
INTCON	0Bh	INTCON	8Bh
PIR1	0Ch	PIE1	8Ch
	0Dh		8Dh
TMR1L	0Eh	PCON	8Eh
TMR1H	0Fh		8Fh
T1CON	10h	OSCCAL	90h
	11h		91h
	12h		92h
	13h		93h
	14h		94h
	15h	WPU	95h
	16h	IDC	96h
	17h		97h
	18h		98h
CMCON	19h	VRCON	99h
	1Ah	EEDATA	9Ah
	1Bh	EADAR	9Bh
	1Ch	EECON1	9Ch
	1Dh	EECON2 <sup>(1)</sup>	9Dh
ADRESH <sup>(2)</sup>	1Eh	ADRESL <sup>(2)</sup>	9Eh
ADCON0 <sup>(2)</sup>	1Fh	ANSEL <sup>(2)</sup>	9Fh
	20h		A0h
General Purpose Registers 64 Bytes		accesses 20h-5Fh	
	5Fh		DFh
	60h		E0h
	7Fh		FFh
Bank 0		Bank 1	

■ Unimplemented data memory locations read as 0

# Registro TRISIO

Registro para configurar flujo de datos

**REGISTER 3-2: TRISIO — GPIO TRISTATE REGISTER (ADDRESS: 85h)**

U-0	U-0	R/W-x	R/W-x	R-1	R/W-x	R/W-x	R/W-x	
—	—	TRISIO5	TRISIO4	TRISIO3	TRISIO2	TRISIO1	TRISIO0	
bit 7								bit 0

bit 7-6: **Unimplemented:** Read as '0'

bit 5-0: **TRISIO<5:0>:** General Purpose I/O Tri-State Control bit

1 = GPIO pin configured as an input (tri-stated)

0 = GPIO pin configured as an output.

**Note:** TRISIO<3> always reads 1.

# Registro GPIO

Registro donde se guardo el estado del puerto GPIO

**REGISTER 3-1: GPIO — GPIO REGISTER (ADDRESS: 05h)**

U-0	U-0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
—	—	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
bit 7		bit 0					

bit 7-6: **Unimplemented:** Read as '0'

bit 5-0: **GPIO<5:0>:** General Purpose I/O pin.

1 = Port pin is >V<sub>IH</sub>

0 = Port pin is <V<sub>IL</sub>

# Registro CMCON

Registro donde se configura el comparador analógico, se debe modificar si se utilizan los pines GP0,GP1 y GP2 como entradas.

REGISTER 6-1: CMCON — COMPARATOR CONTROL REGISTER (ADDRESS: 19h)

U-0	R-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	COUT	—	CINV	CIS	CM2	CM1	CM0
bit 7			bit 0				

bit 7 **Unimplemented:** Read as '0'

bit 6 **COUT:** Comparator Output bit

When CINV = 0:

1 =  $V_{IN+} > V_{IN-}$

0 =  $V_{IN+} < V_{IN-}$

When CINV = 1:

1 =  $V_{IN+} < V_{IN-}$

0 =  $V_{IN+} > V_{IN-}$

bit 5 **Unimplemented:** Read as '0'

bit 4 **CINV:** Comparator Output Inversion bit

1 = Output inverted

0 = Output not inverted

bit 3 **CIS:** Comparator Input Switch bit

When CM2:CM0 = 110 or 101:

1 =  $V_{IN-}$  connects to  $CIN+$

0 =  $V_{IN-}$  connects to  $CIN-$

bit 2-0 **CM2:CM0:** Comparator Mode bits

Figure 6-2 shows the Comparator modes and CM2:CM0 bit settings

# Registro ANSEL

Registro donde se configura conversión y selección analógica, se debe modificar si se utiliza cualquier pin como entrada.

REGISTER 7-2: ANSEL — ANALOG SELECT REGISTER (ADDRESS: 9Fh)

U-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1	R/W-1
—	ADCS2	ADCS1	ADCS0	ANS3	ANS2	ANS1	ANS0
bit 7							bit 0

bit 7 **Unimplemented:** Read as '0'.

bit 6-4 **ADCS<2:0>:** A/D Conversion Clock Select bits

000 =  $F_{osc}/2$

001 =  $F_{osc}/8$

010 =  $F_{osc}/32$

×11 = FRC (clock derived from a dedicated internal oscillator = 500 kHz max)

100 =  $F_{osc}/4$

101 =  $F_{osc}/16$

110 =  $F_{osc}/64$

bit 3-0 **ANS3:ANS0:** Analog Select bits

(Between analog or digital function on pins AN<3:0>, respectively.)

1 = Analog input; pin is assigned as analog input<sup>(1)</sup>

0 = Digital I/O; pin is assigned to port or special function

**Note 1:** Setting a pin to an analog input automatically disables the digital input circuitry, weak pull-ups, and interrupt-on-change. The corresponding TRISIO bit must be set to Input mode in order to allow external control of the voltage on the pin.



# Registro CONFIG

- El PIC12F675 inicia por defecto con el watchdog timer habilitado (WDT).
- GPIO3 por defecto inicia configurado como MCLR (reset).
- El registro CONFIG se encuentra fuera del espacio de memoria del programa, pertenece al espacio de configuración de memoria.
- Típicamente el espacio de configuración de memoria se accede durante la programación.

REGISTER 9-1: CONFIG — CONFIGURATION WORD (ADDRESS: 2007H)

R/P-1	R/P-1	U-0	U-0	U-0	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1	R/P-1
B01	B00	---	---	---	CPD	CP	BOREN	MCLR	PWRT	WDTE	FOSC2	FOSC1	FOSC0
bit 13													bit 0

- bit 13-12 **B01:B00** Bandgap Calibration bits for BOD and POR voltage<sup>(1)</sup>  
00 = Lowest bandgap voltage  
11 = Highest bandgap voltage
- bit 11-9 **Unimplemented:** Read as '1'
- bit 8 **CPD** Data Code Protection bit<sup>(1)</sup>  
1 = Data memory code protection is disabled  
0 = Data memory code protection is enabled
- bit 7 **CP** Code Protection bit<sup>(1)</sup>  
1 = Program Memory code protection is disabled  
0 = Program Memory code protection is enabled
- bit 6 **BOREN** Brown-out Detect Enable bit<sup>(1)</sup>  
1 = BOD enabled  
0 = BOD disabled
- bit 5 **MCLR** GP3/MCLR pin function select<sup>(1)</sup>  
0 = GP3/MCLR pin function is MCLR  
1 = GP3/MCLR pin function is digital I/O, MCLR internally tied to V<sub>DD</sub>
- bit 4 **PWRT** Power-up Timer Enable bit  
1 = PWRT disabled  
0 = PWRT enabled
- bit 3 **WDT** Watchdog Timer Enable bit  
1 = WDT enabled  
0 = WDT disabled
- bit 2-0 **FOSC2:FOSC0** Oscillator Selection bits  
111 = RC oscillator: CLKOUT function on GP4/OSC2/CLKOUT pin, RC on GP5/OSC1/CLKIN  
110 = RC oscillator: I/O function on GP4/OSC2/CLKOUT pin, RC on GP5/OSC1/CLKIN  
101 = INTOSC oscillator: CLKOUT function on GP4/OSC2/CLKOUT pin, I/O function on GP5/OSC1/CLKIN  
100 = INTOSC oscillator: I/O function on GP4/OSC2/CLKOUT pin, I/O function on GP5/OSC1/CLKIN  
011 = EC: I/O function on GP4/OSC2/CLKOUT pin, CLKIN on GP5/OSC1/CLKIN  
010 = HS oscillator: High speed crystal/resonator on GP4/OSC2/CLKOUT and GP5/OSC1/CLKIN  
001 = XT oscillator: Crystal/resonator on GP4/OSC2/CLKOUT and GP5/OSC1/CLKIN  
000 = LP oscillator: Low power crystal on GP4/OSC2/CLKOUT and GP5/OSC1/CLKIN

# Configuración de CONFIG

- Es posible configurar el registro CONFIG con instrucciones de preprocesador (macros o pragmas).
- Típicamente estas macros están definidas en los encabezados del microcontrolador utilizadas por el compilador.
- Se debe revisar documentación del archivo de encabezado, del compilador y la hoja de datos del microcontrolador.
- En el caso del compilador sdcc se puede definir al principio del código fuente (por fuera del main() o en otro archivo de encabezado) la configuración de este registro:

```
typedef unsigned int word;  
word __at 0x2007 __CONFIG = ( macro1 & macro2 & etc ..... );
```

## Como se configuran los GPIOs?

- Antes de usar un pin se debe configurar como entrada o salida.
- Es posible configurar los pines bit a bit o todos a la vez

```
TRISIO = 0x00; //Se configuran todos como salidas  
GPIO = 0x00; //Se ponen todas las salidas en bajo
```

## Escritura de bits/registros en C?

Existen varios métodos para escribir registros en C:

- Método de sobreescritura de registro. No es muy recomendado porque puede sobrecribir configuraciones previas realizadas en el mismo programa, propenso a provocar pulgas

```
TRISIO = 0b00000000;
```

- Método de enmascarado de bits. Buena forma para no modificar bits previamente configurados.

```
GPIO &= (0b11111000); //Se limpian los 3 bits mas bajos sin afectar
```

- Método por campo de bit. Se modifica el bit o bits uno a uno

```
GPIObits.GP0 = 0 //Tambien se puede hacer con GP0 = 0; //Puede dar p
```

Se recomienda realizar la siguiente lectura sobre operaciones básicas a nivel de bit: [https://www.electronicshub.org/](https://www.electronicshub.org/bitwise-operators-in-microcontroller-programming/)

[bitwise-operators-in-microcontroller-programming/](https://www.electronicshub.org/bitwise-operators-in-microcontroller-programming/)

# Escritura/lectura de E/S digitales

- Luego de configurar los pines se pueden leer/escribir sus estados, lo que requiere agregar componentes para garantizar un comportamiento estable.
- Siempre es importante revisar las capacidades eléctricas de los puertos.

## 12.0 ELECTRICAL SPECIFICATIONS

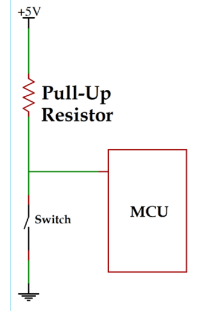
### Absolute Maximum Ratings†

Ambient temperature under bias .....	-40 to +125°C
Storage temperature .....	-65°C to +150°C
Voltage on VDD with respect to VSS .....	-0.3 to +6.5V
Voltage on MCLR with respect to VSS .....	-0.3 to +13.5V
Voltage on all other pins with respect to VSS .....	-0.3V to (VDD + 0.3V)
Total power dissipation <sup>(1)</sup> .....	800 mW
Maximum current out of VSS pin .....	300 mA
Maximum current into VDD pin .....	250 mA
Input clamp current, I <sub>IK</sub> (V <sub>I</sub> < 0 or V <sub>I</sub> > VDD) .....	± 20 mA
Output clamp current, I <sub>OK</sub> (V <sub>O</sub> < 0 or V <sub>O</sub> > VDD) .....	± 20 mA
Maximum output current sunk by any I/O pin .....	25 mA
Maximum output current sourced by any I/O pin .....	25 mA
Maximum current sunk by all GPIO .....	125 mA
Maximum current sourced all GPIO .....	125 mA

# Escritura/lectura de E/S digitales

## Lectura digital - Pull-up

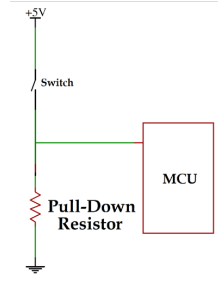
- Un pin puede estar en Alto, Bajo o flotando, no se recomienda (usar pull o/y down resistors)
- En configuracion pull-up el estado es en alto hasta que ocurre un evento que lo pone en bajo
- El estado logico del pin es 1 hasta que el boton se presiona. Se conoce como entrada con logica negativa.



# Escritura/lectura de E/S digitales

## Lectura digital - Pull-down

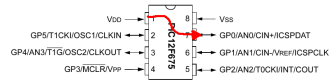
- En configuración pull-down el estado es en bajo hasta que ocurre un evento que lo pone en alto
- El estado lógico del pin es 0 hasta que el botón se presiona. Se conoce como entrada con lógica positiva.
- En los MCU algunos pines vienen con estas resistencias configurables con un registro. Se debe revisar bien el valor y la carga que pueden llevar.



# Escritura/lectura de E/S digitales

## Escritura digital - Current sourcing

- Corriente va de Vdd (+5V), pasa por pin y llega a GND.
- Cuando pines de salida estan como current source el pin esta como fuente (+5V) manejando la carga
- En un MCU los pines pueden manejar cargas pequeñas (20mA)

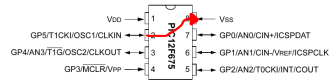




# Escritura/lectura de E/S digitales

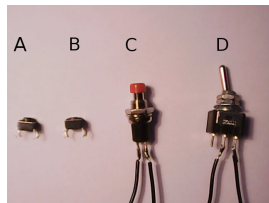
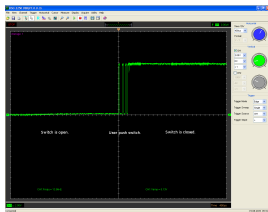
## Escritura digital - Current sinking

- Corriente va de Vss (0V), pasa por pin y llega a Vss.
- Cuando pines de salida están como current sink el pin esta como drenaje (GND) manejando la carga
- En un MCU los pines pueden manejar cargas pequeñas (20mA)



# Escritura/lectura de E/S digitales

## Manejo de rebotes



- Switches tienen efecto de rebote
- Rebotes generan falsas lecturas
- Deben filtrarse por HW o SW
- <https://www.allaboutcircuits.com/technical-articles/switch-bounce-how-to-deal-with-it/>

# Flujo de un programa de un MCU

Usualmente un programa para un microcontrolador esta compuesto por dos partes:

- Inicializacion de periféricos y MCU: Se establecen los modos de operación del microcontrolador y sus periféricos antes de usarse.
- Ejecución de acciones: Son las instrucciones que realiza el microcontrolador de forma cíclica.

# Generación de números aleatorios

Existen varios métodos con los microcontroladores para generar números aleatorios:

- Con una unidad RNG.
- Lectura de convertidores A/D.
- Con lfsr ( linear-feedback shift register )
- Algoritmo Blum-Blum-Shub (BBS)
- Por software con contadores

# Hola PIC

```
#include <pic14/pic12f675.h>
typedef unsigned int word;
word __at 0x2007 __CONFIG = (_BODEN_OFF);
void delay (unsigned int tiempo);
void main(void)
{
    TRISIO = 0x00;//Poner todos los pines como salidas
    GPIO = 0x00;//Poner todos en bajo
    unsigned int time = 100;

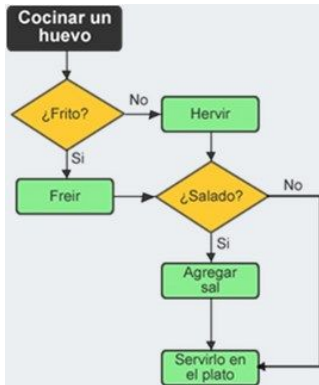
    //Loop forever
    while ( 1 ){
        GPIO0 = 0x00;//Esto se puede hacer tmb con GP0=0x00(no recomendado) o enmascarando
        delay(time);
        GPIO0 = ~GPIO0;//Esto es con el metodo de campo de bit
        delay(time);
    }
}

void delay(unsigned int tiempo)
{
    unsigned int i;
    unsigned int j;

    for(i=0;i<tiempo;i++)
        for(j=0;j<1275;j++);
}
```

# Diagrama de flujo

- El diagrama de flujo o diagrama de actividades es la representación gráfica del algoritmo o proceso.
- Unico punto de inicio y final.



## Recomendaciones para el laboratorios

- Leer con calma la hoja de datos del microcontrolador
- Leer documentación de librería/archivos de encabezado
- Ir paso por paso
- Preguntar