Electrónica Digital
Clase 5

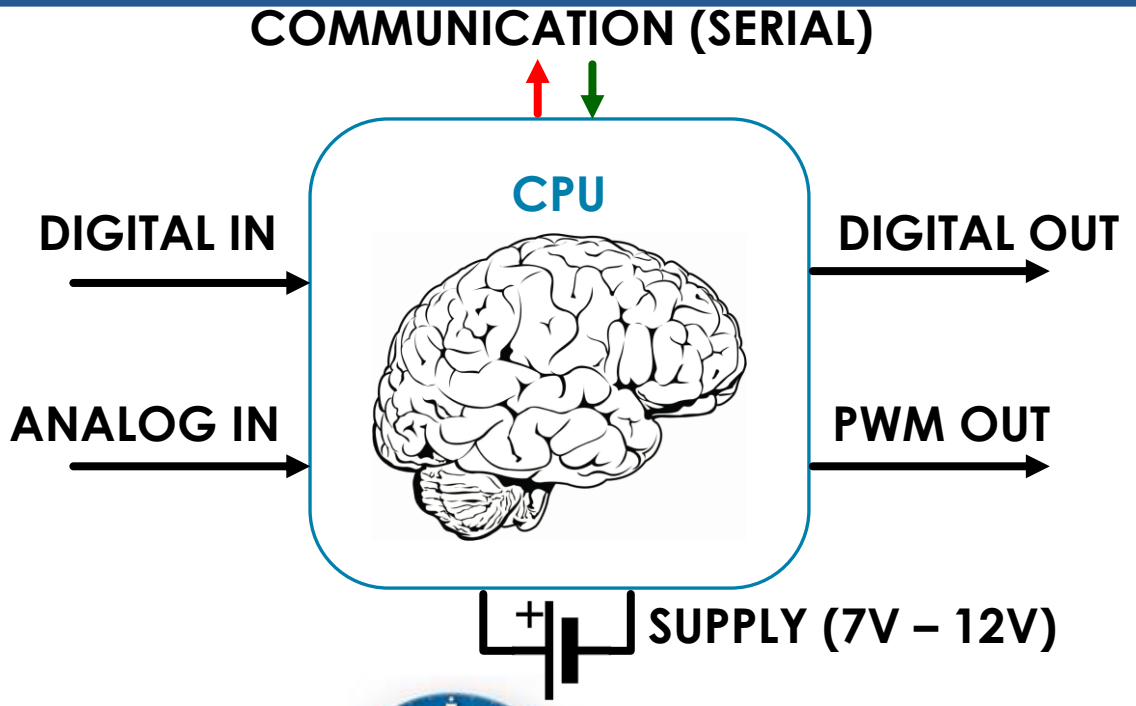
Introducción al Arduino.

Arduino basico, análogo y serial.

UNIVERSIDAD EAFIT®

Introducción al Arduino

- ¿Qué es?
  - Plataforma de electrónica abierta.
  - Posee la estructura completa de un microprocesador (CPU, RAM, ROM y periféricos de I/O).
  - Permite la creación de prototipos electrónicos.
  - Basada en software de código abierto.
  - Fácil de usar.
  - Permiten realizar secuencias y operaciones matemáticas.
  - Poseen alta velocidad de procesamiento.
  - Sirven para automatizar.



- Tipos de Arduino


Arduino Uno


Arduino Leonardo


Arduino Due


Arduino Yún


Arduino Micro


Arduino Robot

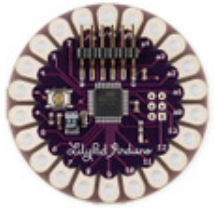

Arduino MKR1000


Arduino Esplora


Arduino Mega 2560


Arduino Ethernet

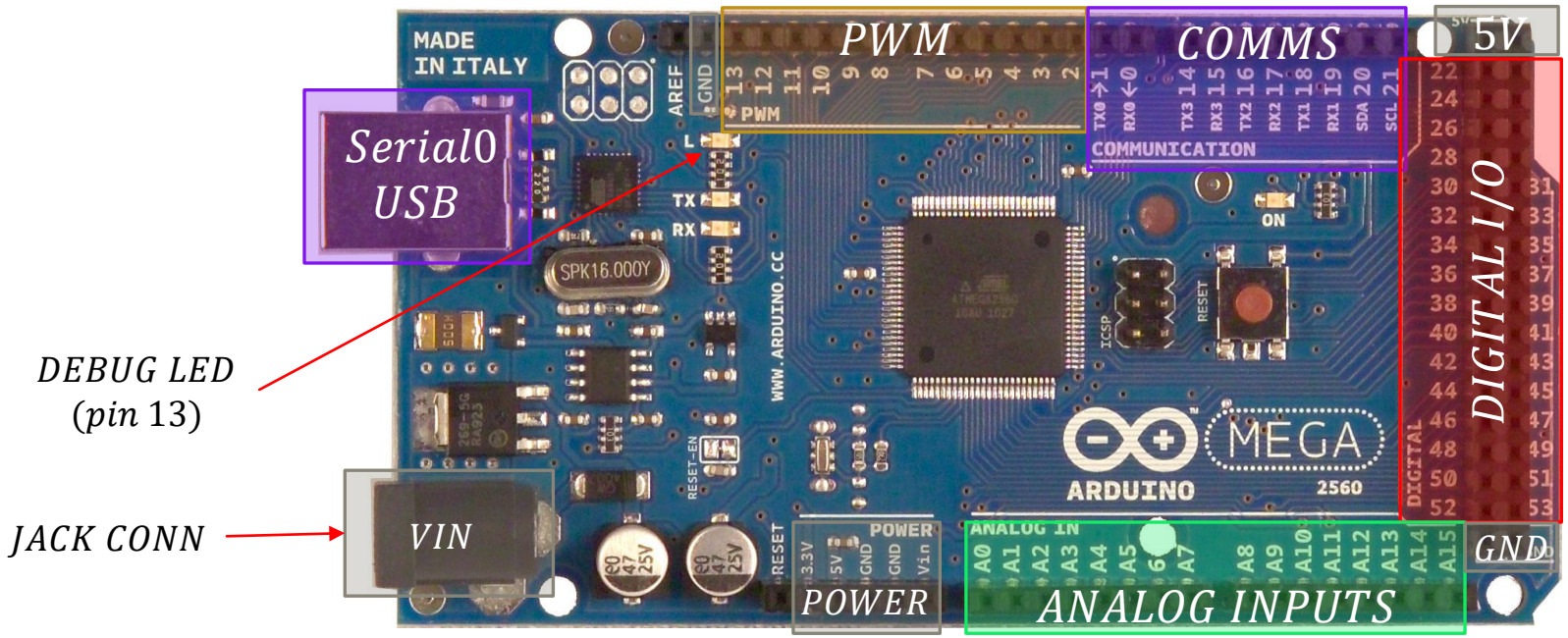

Arduino Mini


LilyPad Arduino


Arduino Nano

IoT

Arduino MEGA 2560 Pinout



- Basado en el microcontrolador **ATMEGA 2560**.
- **Alimentación:** Por conector Jack desde **7V hasta 12V**. También se puede alimentar por USB pero para bajos consumos.
- **Corriente Máxima por pin de I/O digital:** 40 mA.
- **Corriente Máxima para el pin de 3.3V:** 50 mA.
- **Entradas y salidas digitales:** 54 pines (**0V ó 5V**).
- **Entradas Análogas:** 16 pines (desde **0V hasta 5V** de voltaje análogo).
- **Salidas por PWM:** 15 pines (están contemplados dentro de los 54 de I/O).
- **Velocidad del oscilador:** 16 MHz.
- **Comunicación:** Conexión USB CDC (Serial) + I2C + SPI.

Ficha Técnica Arduino Mega 2560

Microcontrolador	ATmega2560
Voltaje de funcionamiento	5V
Voltaje de entrada (recomendado)	7-12V
Voltaje de entrada (los limites)	6-20V
Pines de I/O digital	54 (de los cuales 15 proporcionan una salida PWM)
Pines de entrada análoga	16
Corriente DC I/O Pin	40 mA
corriente DC para 3.3V Pin	50 mA
Memoria flash	256 KB de los cuales 8 KB utilizadas por gestor de arranque
SRAM	8 KB
EEPROM	4 KB
Velocidad del cristal	16. MHz

Tipos de Variables para Arduino

NOMBRE	SINTAXIS	TAMAÑO	RANGO		EJEMPLO
			SIN SIGNO	CON SIGNO	
Booleano	`boolean`	1 bit	false True	N/A	`boolean estado = false;`

Caracter[1]	`char`
	`unsigned char`
		8 bits
		(1 byte)
			0 a 255	-128 a 127	`char micaracter = 'A';`
					`char micaracter = 65;`
					Ambos son equivalentes

Byte	`byte`	8 bits
		(1 byte)
			0 a 255	N/A	`byte hola = B00000111;`
					byte hola = 7;
					B indica que se escribirá en notación binaria.
					El B00000111 es igual a 7 en decimal.

Entero	`int` `unsigned int`	16 bit (2 bytes)	0 a 65535	-32768 a 32767	`unsigned int contador = 0;`
Doble	`long` `unsigned long`	32 bit	0 a 4,294,967,295	-2,147,483,648 a 2,147,483,647	`Unsigned long numero = 20000;`
Flotante[2]	`float`	32 bit	N/A	-3.4028235E+38 a 3.4028235E+38	`float temperatura = 88.5;`

[1]Consultar tabla ASCII para ver correspondencia (http://www.asciitable.com/index/asciifull.gif)
[2]Consultar documentación de ARDUINO para mayor información (http://arduino.cc/en/Reference/Float#.UxOT7_l5Njl)

EQUIVALENCIAS	
word	unsigned int
short	int

Operadores típicos

	SIMBOLO	DESCRIPCIÓN
ARITMÉTICOS	=	Asignación
	+	Suma
	-	Resta
	*	Multiplicación
	/	División
	%	Módulo (Residuo)
COMPARACIÓN	==	**Igualdad:** $x == y$ es equivalente a: x es igual a y?
	!=	**Desigualdad:** $x! = y$ es equivalente a: x es distinto a y?
	<	Menor que
	>	Mayor que
	<=	Menor o igual
	>=	Mayor o igual
BOOLEANOS	&&	AND
	ll	OR
	!	Negación (NOT)
ACUMULADORES	++	**Incremento:** $y = x + +$ es equivalente a: $y = x + 1$
	- -	**Decremento:** $y = x - -$ es equivalente a: $y = x - 1$
	+=	**Suma y acumulación:** $y += x$ es equivalente a: $y = y + x$
	-=	**Resta y acumulación:** $y -= x$ es equivalente a: $y = y - x$
	*=	**Multiplicación y acumulación:** $y *= x$ es equivalente a: $y = y * x$
	/=	**División y acumulación:** $y/= x$ es equivalente a: $y = y/x$

Arquitectura de un programa en Arduino

Declaración de librerías (Ej: `#include <SFEMP3Shield.h>`)

Definición de pines (Ej: `#define ledPin 13`)

Declaración de constantes (Ej: `const unsigned int contMax = 10;`)

Declaración de variables (Ej: `float temperatura = 0;`)

Declaración de subrutinas o funciones:

Ejemplo subrutina:

```
void leer() {                        //Ejemplo de una subrutina que lee el valor análogo de 0 a 1023 y lo convierte de 0
                                     //a 100 grados guardándolo en la variable flotante temperatura.
     y = analogRead(1);              //Lectura análoga del pin A1.
     temperatura = y*100.0/1023.0; //Conversión a flotante y en grados celcius
}
```

Ejemplo función:

```
int sumar(int x, int y) {  //Ejemplo de una función que suma dos números "x" y "y".
     return x + y;
}
```

Configuración de puertos y limpieza de puertos:

```
void setup() {
     //CONFIGURACIÓN: Decir que es entrada y que es salida"pinMode(PIN,OUTPUT o INPUT);" sin comillas.
     //LIMPIEZA: por seguridad, es correcto limpiar las salidas a utilizar con el fin de que estén apagadas al comienzo
     //del programa. Se utiliza la instrucción "digitalWrite(PIN,LOW);" sin //comillas.
     //COMUNICACIONES: Por ejemplo para comunicaciones con el PC se utiliza la instrucción "Serial.begin(BAUDIOS);" sin
     //comillas.
}
```

Ciclo infinito (Programa Principal):

```
void loop() {
     //Programa principal
}
```

Comandos mas usados

- **pinMode**
  - Configura el pin especificado como entrada o salida.
  - Sintaxis: `pinMode(pin, mode);`
    - pin: El # del PIN que se desea configurar según el Arduino
    - mode: Determina si el pin es entrada ó salida. Recibe `INPUT` ó `OUTPUT`.

- **analogRead**
  - Lee y devuelve el valor (value) en que se encuentra en un pin análogo.
  - Sintaxis: `analogRead(pin)`
    - Pin: El # del PIN de entrada que se desea leer su valor.
    - Devuelve `0` a `1023` (10 bit) dependiendo del valor en que se encuentre el pin de entrada leído.

- **digitalWrite**
  - Escribe a un pin de salida digital un valor ALTO (5V) ó un BAJO (0V).
  - Sintaxis: `digitalWrite(pin, value);`
    - pin: El # del PIN que se escribirle un valor.
    - value: `HIGH` ó `LOW`.

- **digitalRead**
  - Lee y devuelve el valor (value) en que se encuentra un pin de entrada digital.
  - Sintaxis: `digitalRead(pin)`
    - Pin: El # del PIN de entrada que se desea leer su valor.
    - Devuelve `HIGH` ó `LOW` dependiendo del valor en que se encuentre el pin de entrada leído.

- **delay**
  - Pausa el programa por un determinado tiempo (en milisegundos).
  - No es muy recommendable utilizarla debido a que frena del todo el programa y luego despues del tiempo continua ejecutandose.
  - Sintaxis: `delay(ms);`
    - ms: El numero de milisegundos que se desea pausar el programa (tipo `unsigned long`).

Función if

- Utilizada en conjunto con un operador de comparación.
- Prueba si una condición se cumple y en caso de cumplirse ejecuta determinadas acciones, luego continua con el programa.
- Sintaxis:

```
if (condición) {
     //Hago algo aqui
}
else if (otra condición) {
     //Hago algo aqui si hay otra condicion
}
else {
     //Hago algo aqui si no se cumple nada de lo anterior
}
```

- Ejemplo con pines digitales de entrada

```
if (digitalRead(pin) == HIGH) {
     //Hago algo aqui si el pin esta en ALTO
}
```

- Ejemplo con variables internas

```
if (temperatura > 25) {
     //Hago algo aqui si temperatura es mayor
     //a 25 grados
}
```

Función switch

- Permite realizar diferentes acciones dependiendo de una variable que puede tener varias posibilidades.
- Es como hacer varios if por la misma variable pero diferentes valores (switch ahorra mas tiempo);
- Cada case es el posible valor que puede tomar la variable y este case se finaliza con break;
- La variable por la que se pregunta en lo posible debe ser de tipo entero.
- Sintaxis:

```
switch (var) {
     case 0:
          //Hago algo aquí si var es igual a cero
     break;
     case 1:
          //Hago algo aquí si var es igual a uno
     break;
     case 2:
          //Hago algo aquí si var es igual a dos
     break;
}
```

- Es posible también preguntar por etiquetas (label) predefinidas al comienzo de un programa con la instrucción #define.

```
switch (var) {
     case label1:
          //Hago algo aquí si var es igual a la etiqueta label1
     break;
     case label2:
          //Hago algo aquí si var es igual a la etiqueta label2
     break;
}
```

Función for

Sintaxis de la estructura FOR

```
for (inicialización, condición, expresión)
{
  //Instrucciones
}
```

Ejemplo: Encender y apagar una alarma 3 veces cada medio segundo.

```
for (i = 0; i < 3;i++)
{
  //Instrucciones
}
```

Función while

Estructuras de control repetitivas

Las estructuras de control repetitivas o iterativas, también conocidas como "bucles" Algunas podemos usarlas cuando conocemos el número de veces que deben repetirse las operaciones. Otras nos permiten repetir un conjunto de operaciones mientras se cumpla una condición.

Bucle while

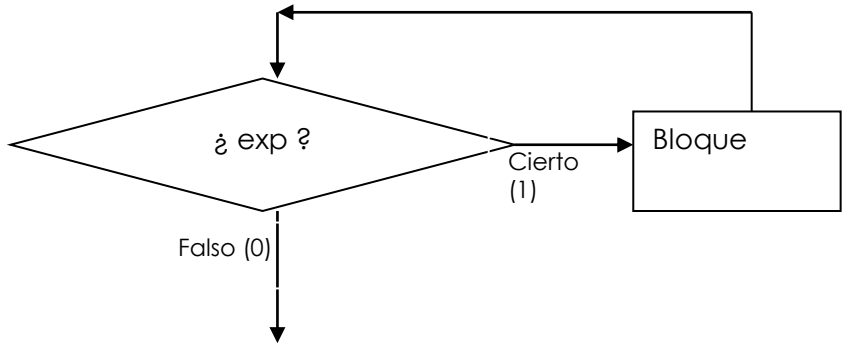
Ejecuta una instrucción o un bloque de instrucciones mientras la condición sea verdadera

Sintaxis

```
while (condición){
  //Sentencia;
}
```

Bloque de instrucciones

```
while (condición){
  //Sentencia 1;
  //Sentencia 2;
  //.. ..
  //Sentencia n;
}
```



Por lo general, dentro de la proposición ó del bloque de ellas, se modifican términos de la expresión condicional, para controlar la duración de la iteración.

Depuración de programas

Para depurar los programas es muy útil usar el monitor serial del programa Arduino. Con el monitor serial se puede indicar lo que esta ocurriendo en el programa. Cuando se activa o desactiva una entrada o salida.
Para configurar el monitor serial solo es necesario incluir en el setup del programa la siguiente instrucción:

```
Serial.begin(9600);          //Configura velocidad de la comunicación serial a 9600.
```

En el programa puede enviar cualquier mensaje con la instrucción:

```
Serial.println ("Led encendido");   //Imprime el texto Led encendido
```

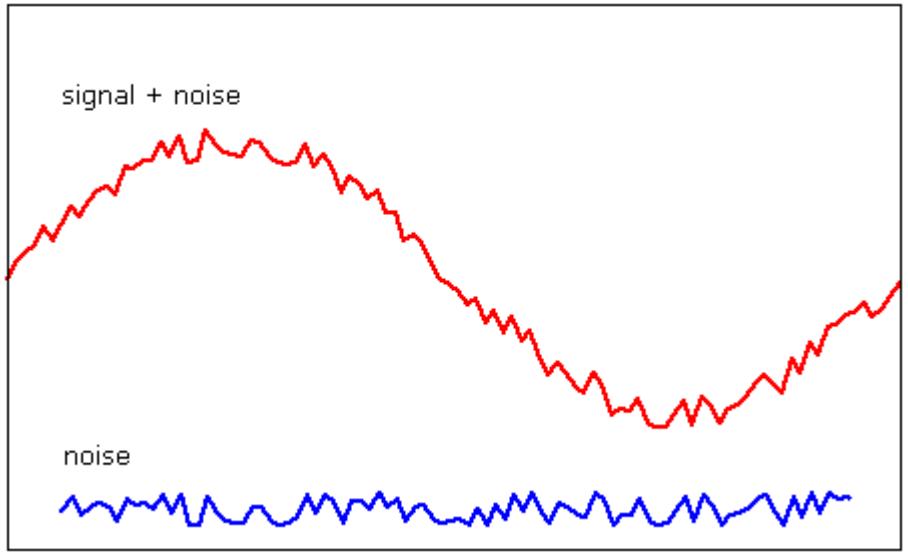
También se puede imprimir el valor de una variable con la instrucción:

```
Serial.println(variable);          //Imprime el contenido de la variable
```

Nota: Al incluir el monitor serial en el programa la ejecución la primera vez es mas demorada de lo normal.

Señales análogas

- Son señales que **varían** en el tiempo.
- Se utiliza el comando en arduino
  - `var = analogRead(PINANALOGO);`
- Un ejemplo es una onda sinusoidal.
- Los sensores en su gran mayoría entregan señales análogas.
- Pueden ser escalizadas según se requiera en la programación
- Es común encontrar **ruido electrónico** en una señal análoga.

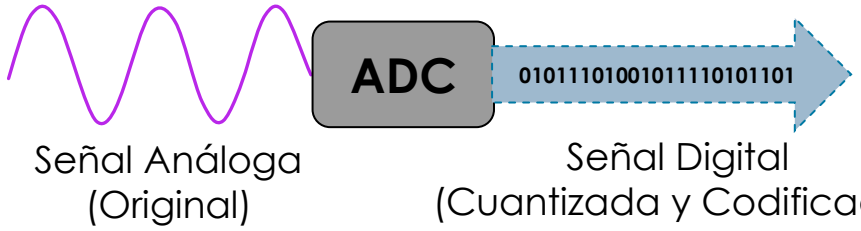



Analog Signal

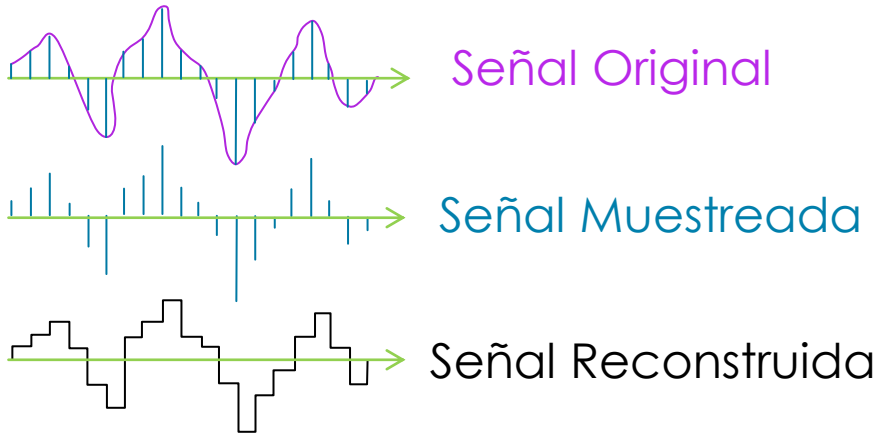

Digital Signal

CONVERSIÓN ANALOGA-DIGITAL (ADC)

- Nos permite **LEER** en un microcontrolador un valor de una variable continua en el tiempo, convirtiéndola a digital por medio de un **MUESTREO, CUANTIZACIÓN Y CODIFICACIÓN:**



A continuación una gráfica de este proceso:



- Resolución: Es la precisión que se puede lograr y depende de los bits (n) a los que se hace la ADC. *Es el mínimo valor que puede distinguir de la señal original.*

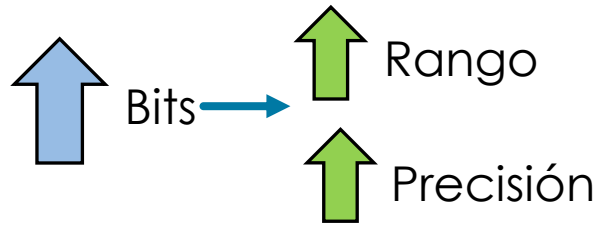
$$Resolución = \frac{V_{DD}}{2^n - 1}$$

Donde

V_{DD}: *Voltaje de alimentación del procesador*
n: *Número de bits de la conversión ADC*

- Rango: Son los limites superior e inferior, es decir, el máximo y el mínimo que tomará el valor cuando se realice la conversión.

$$Rango_{\text{sin } signo} = [0 \; a \; 2^n - 1]$$



Ejemplo Monitor serial

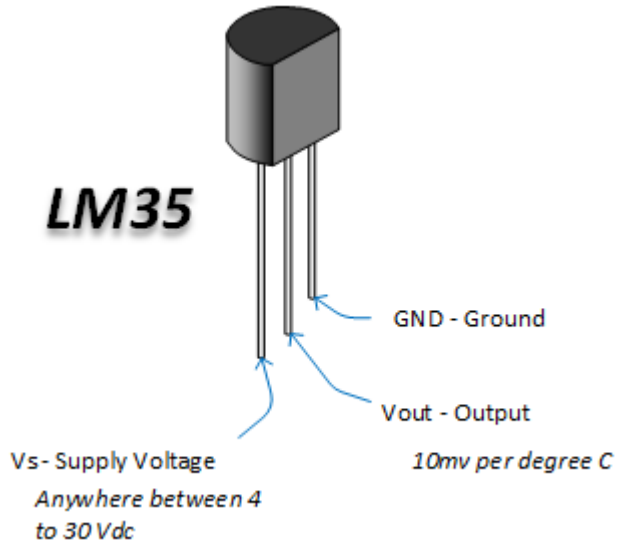
Mostrar en el monitor serial el valor numérico correspondiente a la señal análoga que entra por A0 (entrada análoga 0).

```
//Definicion de pines de I/O
#define POT 0 //Potenciometro conectado en el pin A0

//Definicion de variables
unsigned int medicion = 0;  //Variable para almacenar la medicion de la entrada A0

//Configuracion
void setup() {
  //Comunicaciones
  Serial.begin(9600); //Inicio comunicaciones con el PC (Serial0) a 9600 bauds
}

//Ejecucion
void loop() {
  medicion = analogRead(POT); //Realizo una lectura análoga por el pin POT (A0) y la almaceno en medicion
  Serial.print("VALOR POT: ");
  Serial.println(medicion);
  delay(500); //Retardo de 500 ms
}
```

Seguimiento

- Implemente un circuito que permita monitorear en el Serial el valor de la temperatura del sensor LM35.

Nota: Este sensor genera 10mV por cada grado centígrado.



Tips para la conversión

$ADC\ (10\ bit)$

$5V \rightarrow 1023$

$10\ mV \rightarrow 1℃$

Fin

MUCHAS GRACIAS