



# INSTITUTO TECNOLÓGICO SUPERIOR DE ATLIXCO

## INGENIERÍA MECATRÓNICA MICROCONTROLADORES

### PORAFOLIO DE EVIDENCIAS CATEDRÁTICO

Dra. Mariana Natalia Ibarra Bonilla

#### INTEGRANTES

IM140763 COYOTL TAPIA RAQUEL ARELY

## Microcontroladores

### Primer parcial

Examen parcial escrito	50%
Prácticas con reporte y/o video	40%
Participaciones en clase (programa)	10%
	100%

### Segundo y Tercer parcial

Examen parcial escrito	30%
Prácticas con reporte y/o video	50%
participaciones en clase (programa)	20%

100%

### Cuarto parcial

proyecto Integrador con reporte y video	80%
participaciones en el desarrollo del proyecto	10%
portfolio de evidencias completo (disco con todas las prácticas)	10%

### Exámenes parciales

- parcial 1. 15 de Septiembre del 2017
- parcial 2. 13 de octubre del 2017
- parcial 3. 10 de Noviembre del 2017
- parcial 4. 8 de Diciembre del 2017

### Complementaria

13 de Diciembre del 2017

Tarjetas de desarrollo con ARM cortex M4:

ARM® Cortex® M4 cookbook  
Dr. Mark Fisher

Cómo programar ctt  
M. Deitel & paul j. Deitel  
Ed. prentice Hall

Documentación proporcionada por INTESC  
Página principal

[http://www.intesc.mx/productos/ophyra/  
tutoriales](http://www.intesc.mx/productos/ophyra/tutoriales)

<http://www.intesc.mx/recursos-ophyra/>

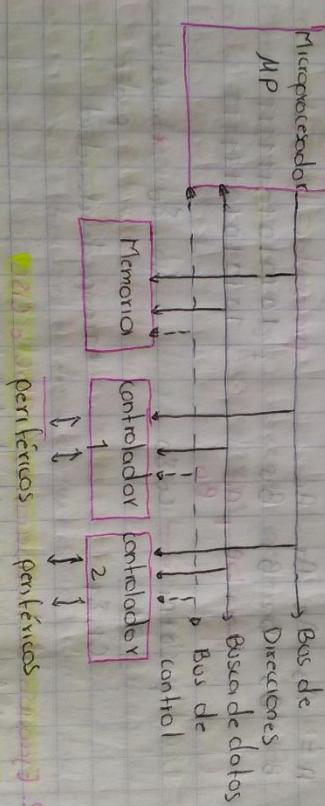
Software recomendado

<http://www.intesc.mx/software-recomendado/>

**Microcontrolador** es un circuito programable, de reducido tamaño que contiene todos los componentes de una computadora. Es un computador dedicado. En su memoria sólo reside un programa destinado a ejecutar una aplicación determinada.

Es un computador completo, aunque de limitadas prestaciones que está contenida en un chip de un circuito integrado y se destina a gobernar una sola tarea.

**Un microprocesador** es un circuito integrado que contiene la unidad central de proceso (CPU), también llamada procesador, de un computador. (laptop)



Microprocesador en una aplicación real, se debe conectar con componentes tales como memoria o buses de transmisión de datos.

Es un sistema abierto con el que puede construirse un computador con las características que se desee, cumpliendo las necesidades.

Un microcontrolador es un sistema cerrado que contiene

on computador completo y de prestaciones limitadas que

-

sólo salen al exterior las líneas que gobiernan los periféricos

J Cómo elegir un microcontrolador?

por el tamaño de palabra

Define el número de bits de los datos con los que trabaja la CPU. suelen ser 4 bits, 8 bits, 16 bits, 32 bits o 64 bits

procesador de 8 bits

los bytes (8 bits)

El número de 32 bits se almacena en 4 bytes

A = A<sub>3</sub> A<sub>2</sub> A<sub>1</sub> A<sub>0</sub> procesador de 32 bits

B = B<sub>3</sub> B<sub>2</sub> B<sub>1</sub> B<sub>0</sub> + 101010111111

suma 150 + 10

B<sub>0</sub>

+ cotar → 0,80

(0,80 \* habla de alta velocidad)

B<sub>1</sub> \* Tiempo de velocidad

C<sub>1</sub>, E<sub>1</sub>

## 2. Ejecución de instrucciones: RISC e CISC

- computador con conjunto de instrucciones Reducidas, RISC (Reduced Instruction Set Computer). El conjunto de instrucción que el procesador es capaz de interpretar y ejecutar es pequeño, además de ser simples y rápidas (PIC)

- computador con conjunto de instrucciones complejas, CISC (Complex Instruction Set Computer). El conjunto de instrucción es grande por ser amplio y permitir operaciones complejas Intel 8051

## 3. Ampliación para intercambio de datos entre los CPU y memoria

Arquitectura denominada

- von Neumann (tradicional)

- Harvard



El procesador ejecuta las instrucciones máquina que están en la memoria principal

Microcontrolador Von Neumann

memoria

de programa

Almacena: Von Neumann

se destaca por tener la memoria de datos compartida

- E) Harvard se caracteriza por tener la memoria de datos independientes bases

- Almel es uno de los mayor fabricantes de microcontrolador más populares son los AVR y ARM

- Intel micro 8051

Microcontroladores RISC

Atmel, National Instruments, XILINX, intel

Solo diseño y licencia a terceras personas

ARM

TI

National Semiconductor

STMicroelectronics

Microchip

Motorola

Philips

Siemens

NEC



micro de como flotante (FPU)

10M - 8V5Cv

**Cortex-M** Esta diseñado para satisfacer la creciente categoría de aplicaciones flexibles específicamente dirigidas al control de motores, automotrices, industriales, etc.

**Cortex-N4** (des)procesamiento digital de señales se desarrolla para dirigirse a mercados de control de señales digitales que demandan una menor fluencia y facilizar el uso de las capacidades de control y procesamiento de señales.

Requerimientos del microcontrolador ARM STM32F407VETx  
(Cortex-M4) 32 bits

- Núcleo de 32 bits ARM Cortex®-M4
- 168 MHz de reloj + instrucciones DSP + unidad de como flotante (FPU)
- 1 Mbyte de memoria flash
- Diferentes interfaces de comunicación: USB, I2C, USART, SPI, LAN
- DMA

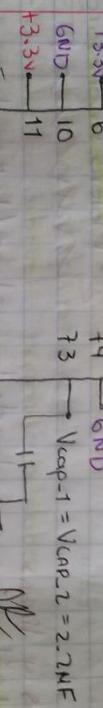
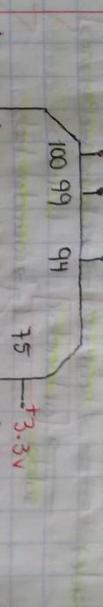
El procesador

- Decodificador de instrucciones parte que desencadena las instrucciones del programa y activa otras unidades basándose en constante
- Unidad aritmética lógica: realiza todas las operaciones matemáticas (suma, resta, multiplicación) y las operaciones lógicas (and, or, not) ALU multidigitos
- Acumulador o registro de trabajo: Es un registro SFR (registro de funciones especiales). Es utilizado para almacenar los datos a realizar de operaciones. 1000 → 8 bits

Encapsulado LQFP  
LQFP (low-profile quad flat package)

En el micro el mismo se carga el programa en la memoria en Boot loader (primero resetear)

(circuito mínimo  
boot loader)



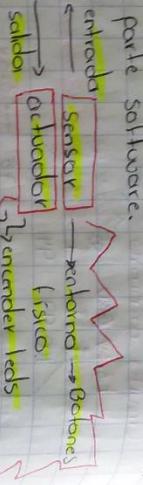
GND +3.3V  
Vcap-1 +3.3V  
Vcap-2  
2.7nF

## Entradas y salidas digitales

25-Agosto-2017

Entrada salida digital es un mecanismo básico de cualquier microcontrolador que permite escribir y leer cualquier señal en dos posibles estados binarios equivalentes a 1 o a 0 en la parte software.

Micromotor



### puertos y líneas

Al subsistema encargado de la escritura/lectura de señales digitales se le denomina general purpose input-output (I/O) y los son agrupaciones de 8, 16 o 32 líneas en función del modelo de microcontrolador.

- Los agrupaciones son de 16 líneas que corresponden internamente a una palabra binaria de 16 bits.
- Los puertos se numeran con letras (A,B,C,D,...) y las líneas con un número entre 0 y el número de líneas por puerto
- 16 pines asociados

### Características principales de los GPIO

- Estados de salida: push-pull, open drain + resistencia de pull-up/down
- (Volviendo el pin)
- Estados de entrada: flotante, pull-up, pull-down, inverso
- Velocidad de lectura/escritura seleccionable
- Bloqueo de GPIO
- Selección de funciones alternativas
- Tolerancia a SV (En las salidas)
- Todos los GPIO de la serie STM32 puede ser configurados como fuente de interrupción externa.

25-Agosto-2017

- Solo tiene un propósito cada pin
- Pwm están en casi todos los pins se encuentra

- Solo Pwm están en casi todos los pins se encuentra

- Pwm

- Esto

- Esto

En cada puerto de salida hay un driver de salida digital (salida analógica y una posible conexión a una función alternativa digital).

### 1: Configuración GPIO como entrada

- El buffer de salida es deshabilitado.
- La entrada Schmitt Trigger es activada.
- Las resistencias de pull-up o pull-down están disponibles para ser activadas.
- pull-up: un resistor conectado a un nivel '1'.
- pull-down: un resistor conectado a un nivel bajo.

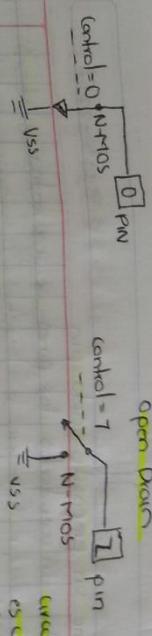
### 2: Configuración GPIO como salida

- El buffer de salida es habilitado y se puede configurar en modo push-pull o open-drain
- La entrada Schmitt Trigger es activada
- Pueder leer el valor presente en el GPIO
- Pueder escribir el valor escrito en el GPIO
- Puede leer el último valor escrito en el GPIO
- Las resistencias pull-up o pull-down están disponibles para ser activadas.

### Modo open drain

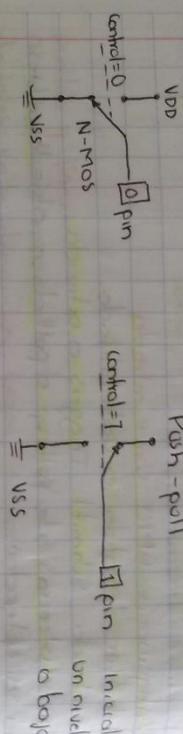
- Este modo el pin de salida está en alta impedancia (z)
- Un "0" en el registro de salida activa el NMOS
- Un "1" en el registro de salida deja el puerto en alta impedancia (el NMOS nunca se activa).

## open Drain



## Modo push-pull

- un "0" en el registro de salida activa el N-MOS
- un "1" en el registro de salida activa el P-MOS



## 3: Configuración de función alternativa

- El buffer de salida puede ser configurado como open-drain o como push-pull
- se puede leer el valor presente en el GPIO
- el buffer es controlado por la salida del periferico

## 4: Configuración entrada analógica

- No es tolerante a su el voltaje máximo soportado es 3.3V.

- La entrada Schmitt Trigger es desactivada y se fuerza un valor de "0" a la salida

## - Funciones alternativas de un GPIO

Hasta multiplexadas hasta 16 funciones alternativas

## GPIO (general purpose input/output)

- Entrada/salida de propósito General

## Gestión de pines

STM32 cube MX Vamos a configurar entradas y salidas Atolice truemudio Vamos a programar código Flash loader donde vamos a cargarle el programa a la tarjeta.

### 1: Configurar el hardware usando STM32cubeMX

- 1.1 Nuevo proyecto "New project" y seleccionamos el microcontrolador de código STM32F407VET6

### 1.2 Seleccionar los pines a utilizar

- 1.2.1 Se presiona clic sobre el pin y seleccionamos GPIO\_Input o GPIO\_Output según corresponda

### 1.3 Recordemos que cada pin tiene hasta 16 funciones, para el

- 1.3.1 Recordemos que cada pin tiene hasta 16 funciones, para el momento únicas vamos a configurar una de las posibles.

### 1.4 Clickderecho sobre el pin y seleccionando "Enter user label"

- 1.4.1 A continuación se desplegará el espacio donde se escribirá el nombre que deseas.

### 1.5 Despues de escribir el nombre se presiona la tecla Enter y automáticamente se coloca el nombre en nuestra pin.

### 1.6 Configurar las resistencias

- 1.6.1 En las opciones de configuración es posible habilitar las resistencias pull-up o pull-down

### 1.7 Configuración de entrada y salidas del microcontrolador

- 1.7.1 Una vez seleccionado GPIO se desplegará la ventana correspondiente

### PC2 Configuration

### GPIO mode

- Input mode
- Output mode
- Input pull-up/poll-down
- Output pull-up
- Button

- Se genera automáticamente cuando se renombra los pines

+Vcc

30-Agosto-2017



\* Configuración para pines de salida

PEO (configuration)

High

output push pull

GPIO mode

No pull-up and no pull-down

GPIO pull/pulldown

Low

Maximum output speed

Low

User label

Led Rojo

\* GPIO output level: nivel lógico inicial en el pin de salida

- High: nivel lógico alto

- Low: nivel lógico bajo

\* GPIO mode

- Output push-pull: genera búsquedas lógicas (voltajes) a través del pin

- Output open drain: recibe de una excitación externa (pull-up o externo) para que los niveles lógicos se reflejen en el pin.

\* GPIO pull up/pull-down: "No pull-up and no pull-down"

debido a que no es necesario habilitar ninguna de estas

resistencias del microcontrolador, pues el pin solo se usará

como salidas.



1.4 Generar el código fuente.

Se selecciona el menú project → Generar code o dar clic en el icono de un chip.

- 1: Se coloca un nombre del proyecto (project name)
- 2: project location (workspace para Atollic TrueStudio).

### 3: Instrucciones

Sin toxos

HAL\_GPI0 - ReadPin()  
HAL\_GPI0 - ReadPin(GPI0\_TypeDef \*GPI0X, uint16\_t GPI0P)

Paso 1. presionar sin soltar Botón Reset.

Paso 2. presionar sin soltar Boot.

Paso 3. soltar

Paso 4. soltar

Paso 5. soltar

Paso 6. soltar

Paso 7. soltar

Paso 8. soltar

Paso 9. soltar

Paso 10. soltar

Paso 11. soltar

Paso 12. soltar

Paso 13. soltar

Paso 14. soltar

Paso 15. soltar

Paso 16. soltar

Paso 17. soltar

Paso 18. soltar

Paso 19. soltar

Paso 20. soltar

Paso 21. soltar

Paso 22. soltar

Paso 23. soltar

Paso 24. soltar

Paso 25. soltar

Paso 26. soltar

Paso 27. soltar

Paso 28. soltar

Paso 29. soltar

Paso 30. soltar

Paso 31. soltar

Paso 32. soltar

Paso 33. soltar

Paso 34. soltar

Paso 35. soltar

Paso 36. soltar

Paso 37. soltar

Paso 38. soltar

Paso 39. soltar

Paso 40. soltar

Paso 41. soltar

Paso 42. soltar

Paso 43. soltar

Paso 44. soltar

Paso 45. soltar

Paso 46. soltar

4: Descarga el programa \* .Hex ophyro

→ Next

Com16 modo arranque-programación (bootLOADER)

Paso 7. modo arranque-programación (bootLOADER)

para hacer esto presionamos el botón Reset (reinicio).

Paso 1. presionar sin soltar Botón Reset.

Paso 2. presionar sin soltar Botón Reset.

Paso 3. presionar sin soltar Botón Reset.

Paso 4. presionar sin soltar Botón Reset.

Paso 5. presionar sin soltar Botón Reset.

Paso 6. presionar sin soltar Botón Reset.

Paso 7. presionar sin soltar Botón Reset.

Paso 8. presionar sin soltar Botón Reset.

Paso 9. presionar sin soltar Botón Reset.

Paso 10. presionar sin soltar Botón Reset.

Paso 11. presionar sin soltar Botón Reset.

Paso 12. presionar sin soltar Botón Reset.

Paso 13. presionar sin soltar Botón Reset.

Paso 14. presionar sin soltar Botón Reset.

Paso 15. presionar sin soltar Botón Reset.

Paso 16. presionar sin soltar Botón Reset.

Paso 17. presionar sin soltar Botón Reset.

Paso 18. presionar sin soltar Botón Reset.

Paso 19. presionar sin soltar Botón Reset.

Paso 20. presionar sin soltar Botón Reset.

Paso 21. presionar sin soltar Botón Reset.

Paso 22. presionar sin soltar Botón Reset.

Paso 23. presionar sin soltar Botón Reset.

Paso 24. presionar sin soltar Botón Reset.

Paso 25. presionar sin soltar Botón Reset.

Paso 26. presionar sin soltar Botón Reset.

Paso 27. presionar sin soltar Botón Reset.

Paso 28. presionar sin soltar Botón Reset.

Paso 29. presionar sin soltar Botón Reset.

Paso 30. presionar sin soltar Botón Reset.

Paso 31. presionar sin soltar Botón Reset.

Paso 32. presionar sin soltar Botón Reset.

Paso 33. presionar sin soltar Botón Reset.

Paso 34. presionar sin soltar Botón Reset.

Paso 35. presionar sin soltar Botón Reset.

Paso 36. presionar sin soltar Botón Reset.

Paso 37. presionar sin soltar Botón Reset.

Paso 38. presionar sin soltar Botón Reset.

Paso 39. presionar sin soltar Botón Reset.

Paso 40. presionar sin soltar Botón Reset.

Paso 41. presionar sin soltar Botón Reset.

Paso 42. presionar sin soltar Botón Reset.

Paso 43. presionar sin soltar Botón Reset.

Paso 44. presionar sin soltar Botón Reset.

Paso 45. presionar sin soltar Botón Reset.

Paso 46. presionar sin soltar Botón Reset.

Paso 47. presionar sin soltar Botón Reset.

## Programa Aprender y Pasear on LED

```

while(1)
{
    /* user code and while */
    /* user code begin 3 */
    HAL_GPIO_WritePin(GPIOE, Led_Rojo_Pin, GPIO_PIN_RESET);
    HAL_Delay(1000);
    HAL_GPIO_WritePin(GPIOE, Led_Rojo_Pin, GPIO_PIN_SET);
    HAL_Delay(1000);
    HAL_GPIO_WritePin(GPIOE, Led_Azul_Pin, GPIO_PIN_RESET);
    HAL_Delay(1000);
    HAL_GPIO_WritePin(GPIOE, Led_Azul_Pin, GPIO_PIN_SET);

    /* Abrir el proyecto hecho */
    - [C] - SRC
    - main.c
    - proyecto 2
    * cuando modificas solo hay que compilar
    * Boton (pin configuration) → GPIO → pull-up
}

If - else
if ((A==10) {
    if (B>20) {
        C=15;
    }
} else {
    C=5;
}

```

6-Sep-2014

Scribe

no permite regresar el if  
break;  
El comando break provoca la salida de switch al final de las sentencias; contrario se ejecuta case

no permite saltar el if  
break;

### Ejemplo Switch-case

```

For
For es modo para crear loops
de interacciones
Sintaxis
for (inicialización; condición de
finalización; incremento)
{
    sentencias;
}

```

```

switch (A)
{
    case 0:
        B = 1;
    case 1:
        B = 2;
    case 2:
        B = 3;
    default:
        B = 4;
}

```

### Ejemplo Switch-case

```

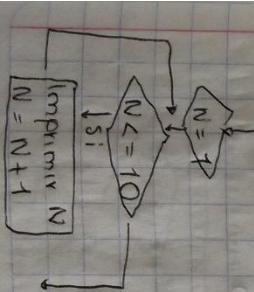
For
For es modo para crear loops
de interacciones
Sintaxis
for (inicialización; condición de
finalización; incremento)
{
    sentencias;
}

```

```

switch (A)
{
    case 0:
        B = 1;
    case 1:
        B = 2;
    case 2:
        B = 3;
    default:
        B = 4;
}

```



### Ejemplo

```

for (N=1; N<=10; ++N)
{
    printf ("%d", N);
}

```

case constante 1:  
sentencias;  
break;  
case constante 2:  
sentencias;

[Imprimir N  
N = N+1]

#### 4: while / Do-while

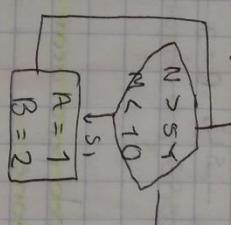
while se utiliza para repetir sentencias. La expresión se evalúa y la sentencia se ejecuta mientras la expresión es verdadera cuando es falsa se salta del while.

Sintaxis  
while (expresión)  
{     sentencias;  
}

while (N>5 && M<10)  
{  
    A = 1;  
    B = 2;  
}

#### 5: Do-while

en la condición de finalización (la cual se evalúa al final) del ciclo, lo cual evalúa desde el final del ciclo, por las sentencias se ejecutan al menos una vez



El while (1) el ciclo se está generando  
quedarme hay

#### Actividad

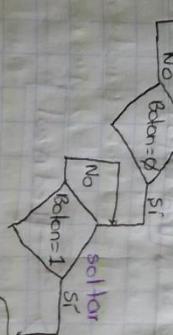
6 - Septiembre - 2014

160 MHz Trabaja el reloj del microcontrolador.

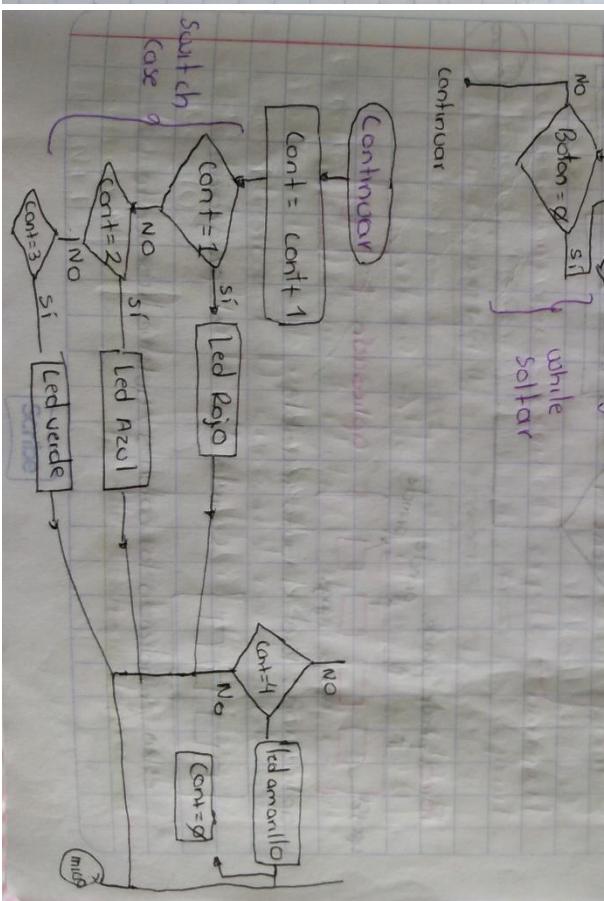
Secuencia Anti-Rebote (latch)



para implementación con ciclos WHILE



Vamos a generar un contador para poder incrementar

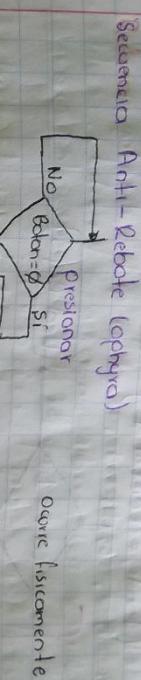


## Actividad

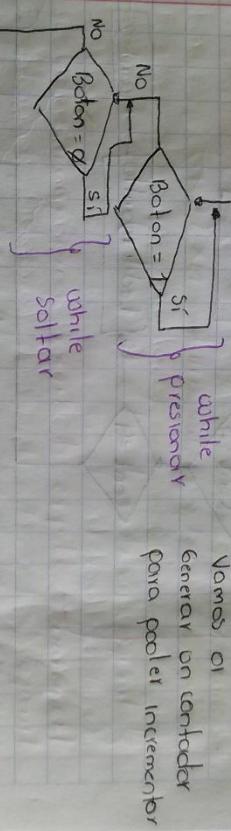
6- Septiembre - 2017

160 MHz Trabaja el reloj del microcontrolador.

Sección Anti-Rebote (ladrillo)

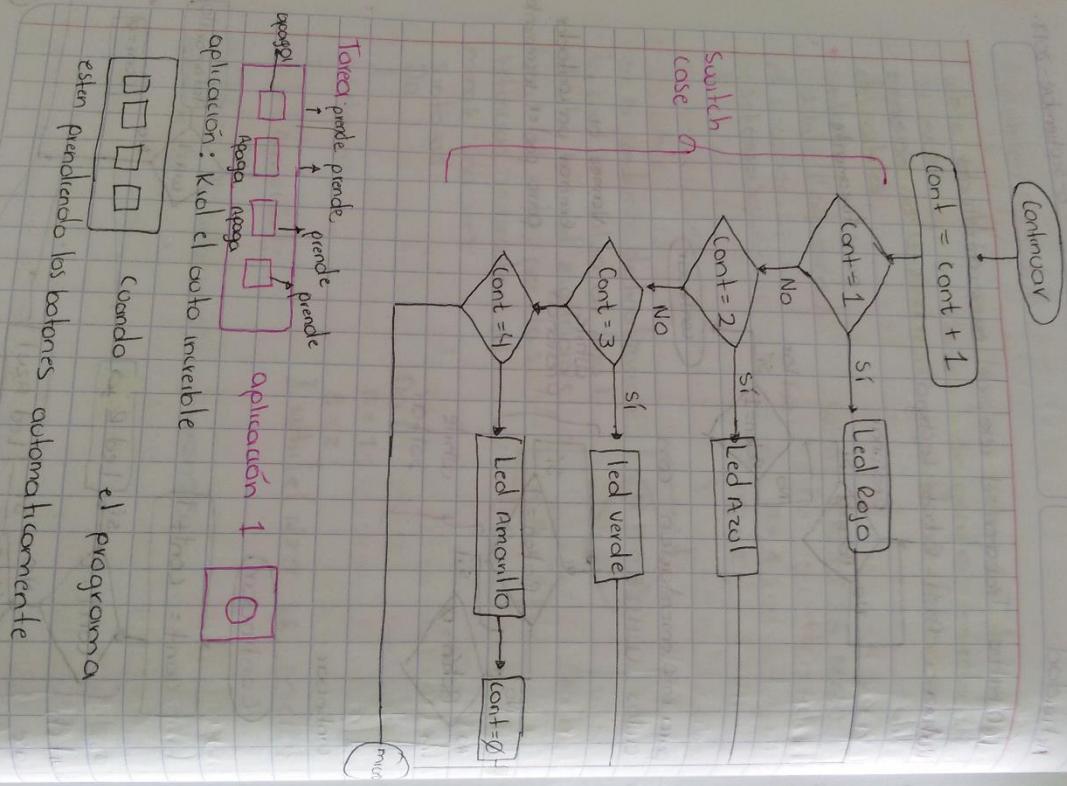
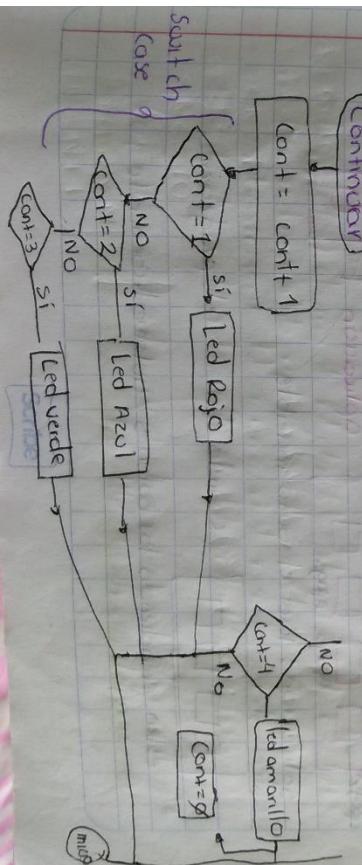


para implementación con ciclos WHILE



Nomos el

generar un contador para poder incrementar



están prendiendo los botones automáticamente

## Segundo parcial

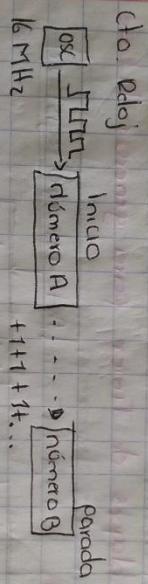
18-septiembre-2014

### Timers y su interrupción

Microcontrolador necesita contar eventos o generar retardos de gran precisión, no es recomendable ya que el CPU se queda esperando.

Los temporizadores o timers son puentes hardware que suplen este efecto, desencadenando de un trabajo poco grato al micro. Permiten medir ancho de pulso de señales, generar señales digitales, contar impulsos.

Temporizadores (timers), son puentes hardware que permiten medir el tiempo transcurrido entre dos eventos, basados en contar los pulsos generados por el oscilador principal (ciclo de reloj).



$$\text{Tiempo transcurrido} = (\beta - \alpha) \text{ seg}$$

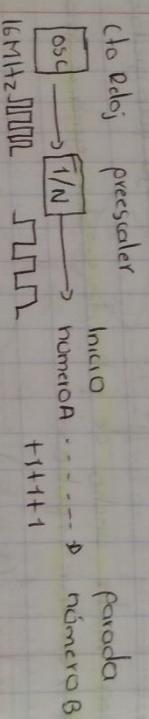
\* Cuenta 14 temporizadores (12 Timers con resolución de 16 bits y 2 Timers con 32 bits).

32-1 Se le resta uno porque inicio en cero. Prescaler acepta cualquier número entre 11 a 6535

## ¿Qué es prescaler?

Un prescaler es un divisor de frecuencia programable. Para el timer, el uso de un prescaler implica quitar para el timer 1, 2, 3, 4, etc. o más ciclos de reloj a su entrada para generar un pulso a la salida. Se utiliza cuando es necesario medir los períodos de tiempo más largos con un periodo más grande.

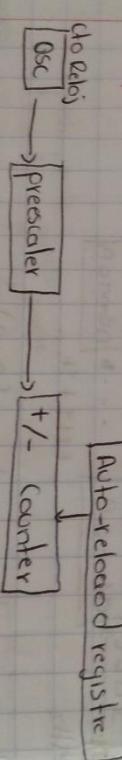
- una señal con un periodo de reloj (contar períodos o ciclos - períodos de mi señal de reloj). Timer



$$\text{Tiempo transcurrido} = N \times (\beta - \alpha) \text{ seg}$$

El tiempo que dura el conteo (prescaler)

Diagrama a bloques de Timer para generar la base de tiempo.



Auto-reload registro: contador de periodo (establece el número de ciclos que debe ejecutar el timer)

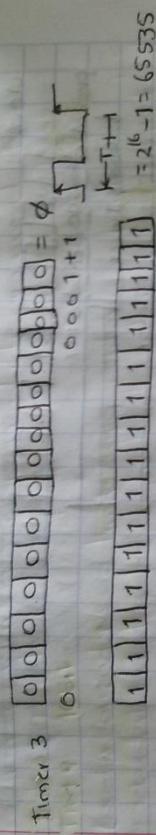
Prescaler: divisor de frecuencia del temporizador.

Counter: contador del temporizador de n bits de resolución

\* Configuración en número de incrementos (no en tiempo) es un registro. Auto-reload

Delay para el CPU \* Contador es un registro del número de incrementos

Ejercicio: Utiliza el Timer 3 y calcula el valor del prescaler y del contador de periodo para generar un reloj de tiempo de 1 Segundo



Interrupción  
consiste en señales que informan a la CPU de que hay que atender algo.  
\* permite a cualquier servicio interrumpir o exterior interrupción ejecución del programa principal en cualquier momento.

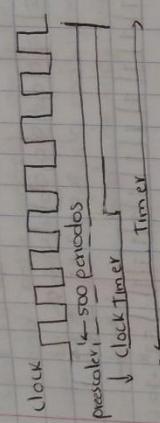
- Interrupción provoca que el procesador cancelle temporalmente el trabajo en curso también llamado "programa principal"
- pasa a ejecutar otra tarea programada llamada "pieza de tratamiento de la interrupción" realizar el trabajo que tenga que ver con este evento y continuar finalmente, con lo que estaba antes

1 Int main (void)

3  
2  
1  
\* Se presenta interrupción  
void Timx\_IRQHandler(void)

$$\begin{aligned} \text{Contador periodo} &= 32000 \\ (\text{prescaler} = 500) &= 32000 \end{aligned}$$

$$\text{Tiempo} = (32000) (31.25 \mu\text{s}) = 1 \text{ ms}$$



Interrupción  
consiste en señales que informan a la CPU de que hay que atender algo.  
\* permite a cualquier servicio interrumpir o exterior interrupción ejecución del programa principal en cualquier momento.

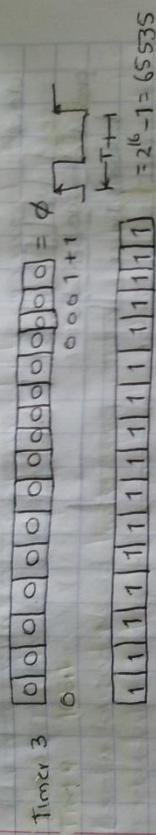
- Interrupción provoca que el procesador cancelle temporalmente el trabajo en curso también llamado "programa principal"
- pasa a ejecutar otra tarea programada llamada "pieza de tratamiento de la interrupción" realizar el trabajo que tenga que ver con este evento y continuar finalmente, con lo que estaba antes

1 Int main (void)

3  
2  
1  
\* Se presenta interrupción  
void Timx\_IRQHandler(void)

Delay para el CPU \* Contador es un registro del número de incrementos

Ejercicio: Utiliza el Timer 3 y calcula el valor del prescaler y del contador de periodo para generar un reloj de tiempo de 1 Segundo



\* Durante tiempo le lleva al Timer 3 realizar todo la cuenta

$\phi \rightarrow 65535$

$F_{clock} = 16 \text{ MHz}$

$T_{clock} = \frac{1}{16 \text{ MHz}} = 62.5 \text{ ns}$

$Tiempo_{max\_contador} = 62.5 \text{ ns} * 65535 = 4.09 \text{ ms}$

\* ¿Cuántas N veces debemos aumentarlo?

prescaler=N =  $\frac{1.5 \text{ seg}}{4.09 \text{ ms}} = 384$

N=prescaler/máximo

a) prescaler = 244  
contador periodo = 65535

\* b) Deudimos un prescaler de 500

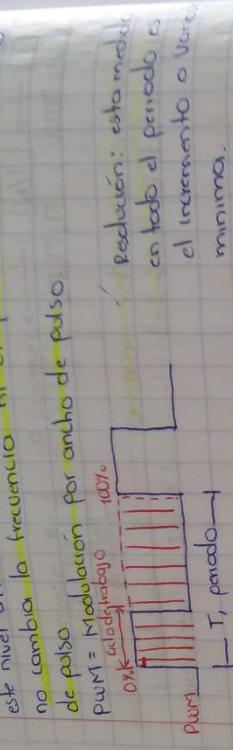
- Tiempo de incremento de Timer =  $\frac{1}{16 \text{ MHz}}$

1 incremento  $\rightarrow 31.25 \mu\text{s} / 1000 = 31.25 \mu\text{s}$

## funcionamiento de PWM usando TIMERS

funcionamiento de PWM usando TIMER:

Duty cycle = el ciclo de trabajo, una señal cuadrada usada para controlar la duración en que la señal está en alto este nivel alto, la duración en que la señal está en bajo no cambia la frecuencia ni el periodo solo el ancho de pulso.



$$\text{configuración:} \\ \text{prescaler} = \left[ \frac{\text{Frecuencia Reloj}}{(\text{Frec. Pwm}) (\text{Resolución Pwm})} \right] - 1$$

$$\text{Frec Reloj} = 16 \text{ MHz}$$

- Frec Pwm =
- \* dividir todo el periodo en intervalos para ser la resolución dependiendo de lo que se desee.

STM(cubo)

- \* prescaler
- \* Resolución (Counter period) (32 bits) quita en 32 bits

\* Iniciar el PWM

HAL-TIM\_Pwm\_Start(puntero, lana)

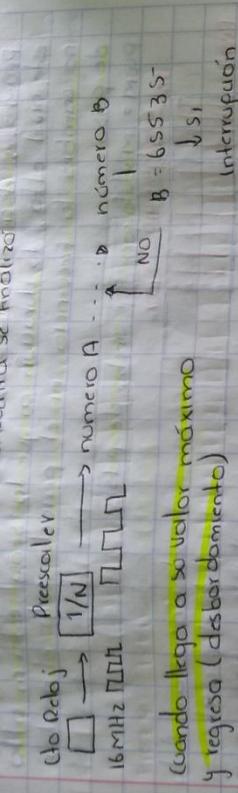
\* establecer el ciclo de trabajo

HAL-TIM\_Set - COMPARAR (Puntero, canal, valor de trabajo deseado 2 guiones)

ejemplo 100 intervalos solo que 90 50% var SO.

18-septiembre-2017

Timer: Pueden prever un sedoso, una interrupción cada N segundos \* 1 bit se activa cuando la cuenta se finaliza



Timer STM32 cube → para usar los TIMERS

- Se configura el periodo
- Se configura el contador periodo en el software (No hay hardware la interrupción por Timer programación)

atollic

Interrupción por timer, además debe ser configurado el software para inicializarla en el programa

HAL-TIM\_Base\_Start\_IT(8htim1);

STM32 cube

Paso 1 TIM3

Configurar la fuente de reloj - clock source (Internal clock)

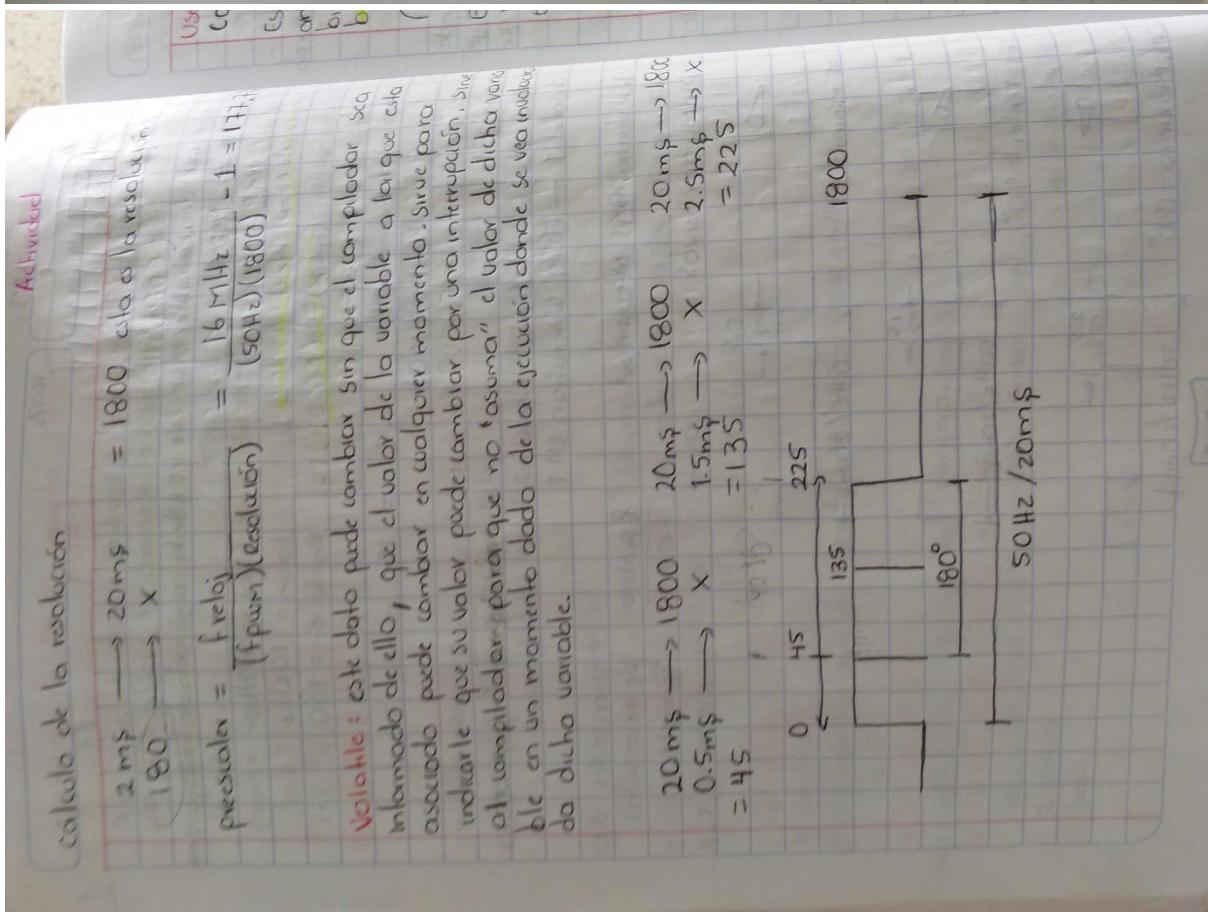
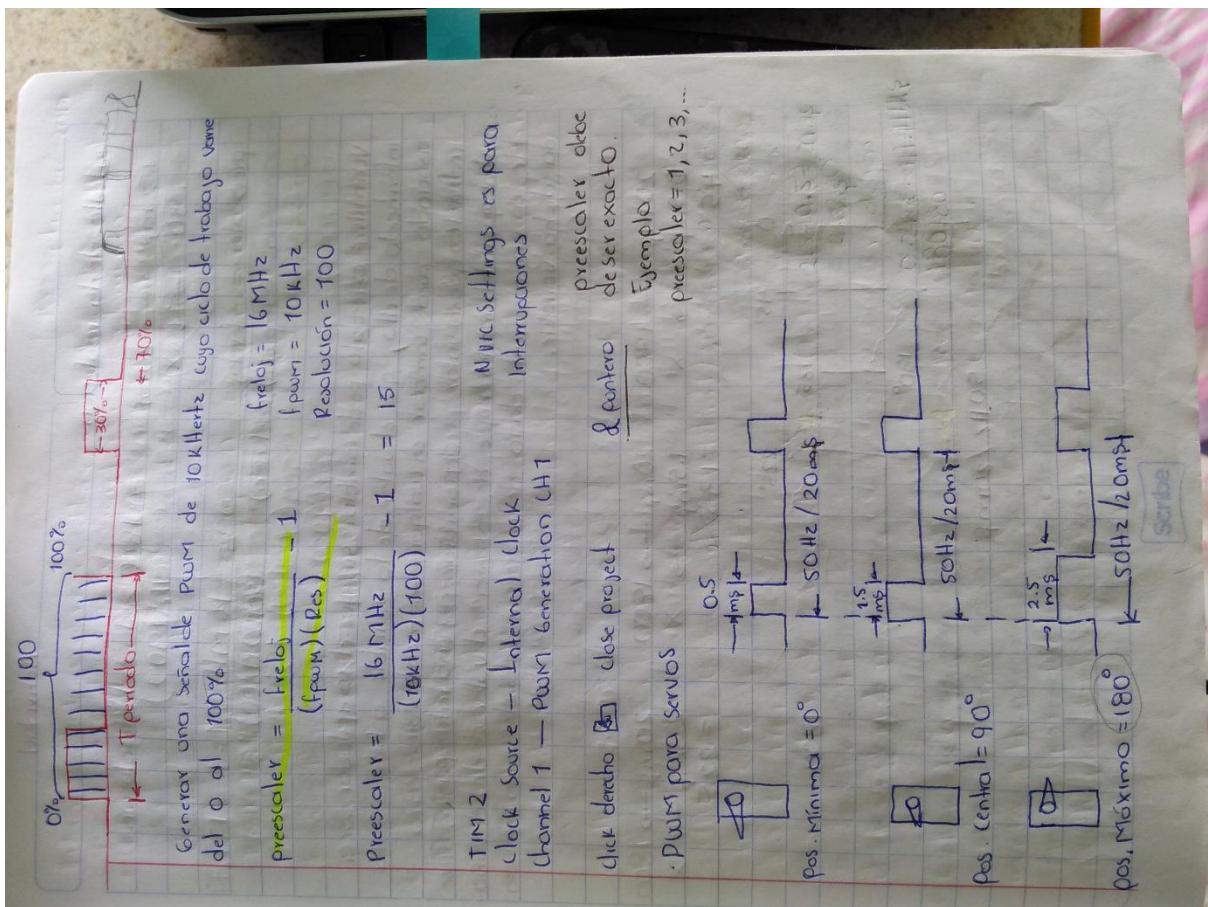
clock configuration

Clock se activa para la interrupción

configuration

- TIM3 seleccionar counter settings
- prescaler (psc - 16 bit width) = 500
- Counter Period (AutoReload) = 32000

Enabled



9-oct-2017

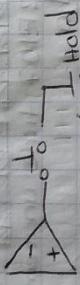
### Uso del convertidor - analógico digital ADC

Convertidor analógico a valor digital  
Es un circuito que se encarga de convertir un valor analógico de voltaje a su correspondiente combinación binaria. convierte un número real en un número binario.

ADC 0101

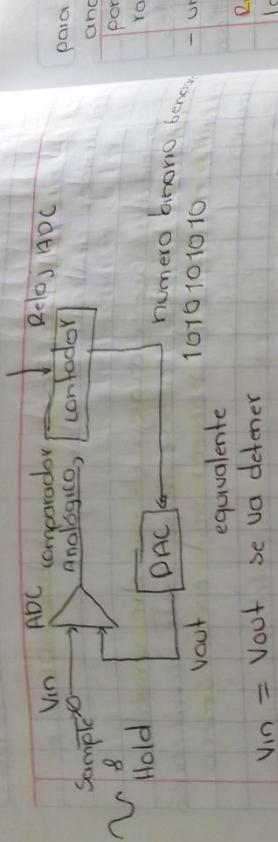
El módulo ADC que utiliza hace un muestreo y retención (sample and hold) con un condensador y después utiliza el módulo de conversión.

$V_{out} = V_{out}$  → valor muestreo y retención.



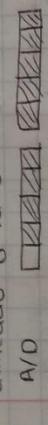
El convertidor ADC convierte una señal de entrada en un resultado binario de 12 bit utilizando el método de aproximación sucesiva.

Aproximaciones sucesivas: se basa en realizar sucesivas comparaciones de forma ascendente o descendente hasta encontrar un valor digital que iguale la tensión entregada por el conversor D/A (Digital analógico) y la tensión de entrada.

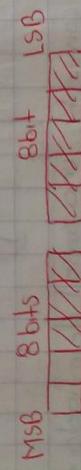


La resolución mínima o calidad de conversión se ajusta a diferentes necesidades al seleccionar Voltaje de referencia. Ver y Ver y la cantidad de bits que representan el resultado de la conversión.

La ADC de n bits puede representar hasta  $2^n$  valores digitales alineados a la derecha o la izquierda.



El STM32F407 tiene un convertidor de 12 bits alineado a la derecha.



$2^{12}-1$  almacenamiento - Derecho.



$$2^{12}-1 = (4095)_0 \\ = (FFF)_{16}$$

### STM32 - Cube

**Resolution** Selecciona la resolución para la conversión del ADC.

**Data Alignment** selecciona la alineación de los datos leídos por el ADC.

**Continuous Conversion Mode** modo de conversión continuo de los canales a convertir leídos por el ADC.

**Number of conversion** número de canales a convertir leídos por el ADC.

```
    Inicial el ADC
    HAL_ADC_Start(&hadc1);
    Lee el valor de la lectura del ADC
    HAL_ADC_GetValue(&hadc1);
```

### Ejemplo ADC:

Leer el canal 0 del ADC1 y controlar el color de un LED RGB de acuerdo al voltaje analógico presente en el canal de tal manera que:

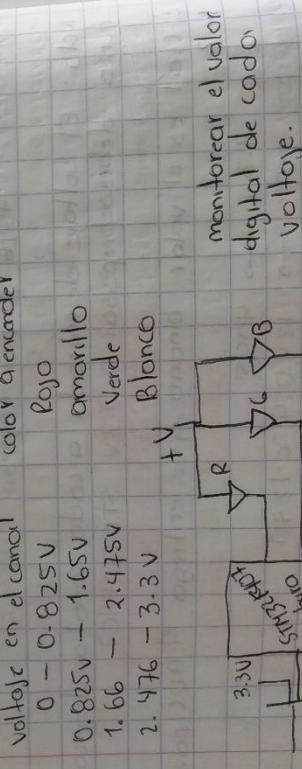
voltaje en el canal | color a encender

Rojo

Amarillo

Verde

Blanco



3.3  
0.8  
1.6  
2.4  
0.8  
1.6  
2.4

para que el ADC pueda realizar una conversión del valor analógico, se tiene que agregar un voltaje de referencia porque éste es el que indica precisamente cuál es el rango de operación de la entrada del ADC

- un ADC de n bits puede representar hasta  $2^n$  valores

### Resolución del ADC:

la diferencia entre dos valores análogicos correspondientes a dos valores digitales consecutivos se define como la resolución de voltaje de ADC

$$\text{Resolución} = \frac{(V_{ref+} - V_{ref-})}{2^{n-1}}$$

¿Cuál es la resolución del ADC 12 bits con  $V_{ref+} = 3.3V$  y  $V_{ref-} = 0V$ ?

$$\text{Resolución} = \frac{(3.3 - 0)}{4096 - 1} = \frac{3.3}{4095} = 0.805mV$$

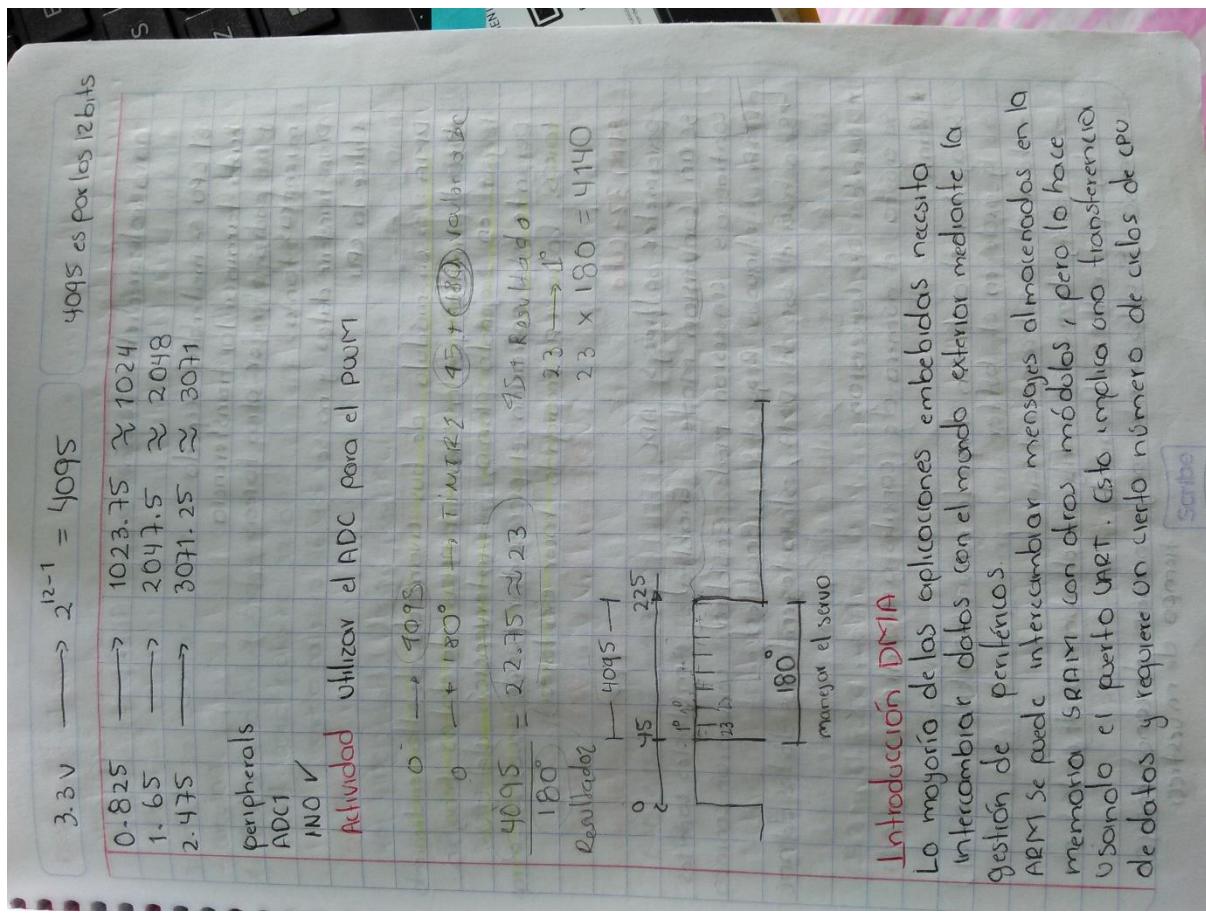
Nota: El voltaje de entrada a cada canal analógico nunca debe superarse a 3.3V STM32F407

¿Cuál es el valor binario que entrega el ADC para 1V?

$$3.3V \rightarrow 4095$$

$$1V \rightarrow 1240.9 \approx 1241$$





para realizar dicha acción

El LPU se mantendrá ocupado hasta terminar los transferencias y no podrá atender otras tareas mientras tanto.

DMA transfirió datos y no se utilizó la CPU

STM32: es un módulo de bus avanzado de alto rendimiento que permite que las transferencias de datos tengan lugar en segundo plano, sin intervención del CPU. Durante esta operación, el procesador puede ejecutar otras tareas. (otro CPU encargado hacer esa tarea).

**STM32 - DMA**

parameters settings ADC multiples canales

- \* Scan Conversion Mode Enabled
- \* continuous conversion mode Enabled
- \* DMA continuous Requests Enabled

Se activa el uso del DMA, estara leyendo los datos del ADC

\* Number of conversions Se ajusta el número de canales a convertir

\* Rank Se ajusta el orden en el que la conversión (de cada canal) se guarda en el buffer

- channel Se selecciona el canal

- Sampling time. Se configura el número de ciclos de reloj para leer el canal.

minimo 15 cycles - entre mayor numero de canal mayor tiempo de muestra

16-oct-2017

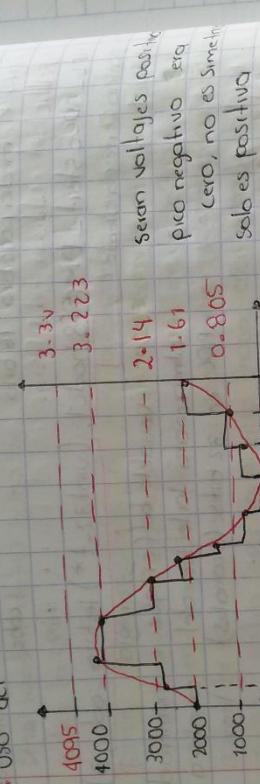
### DMA settings

```
/* User code begin */
HAL_TIM_PWM_Start(Bkpin2, Tim_Channel_1);
para inicio y d ciclo de trabajo
```

18-Oct-2017

### DAC ( Digital to Analog converters )

\* uso del DAC con TIMER



$$FFF = 4095$$

$$\text{Sen} = \left[ \sin \left( \frac{2\pi}{n_p} t \right) + 1 \right] \left[ \frac{0 \times FFF + 1}{2} \right]$$

offset

2048

El microcontrolador STM32F407V6 cuenta con dos canales DAC que pueden trabajar independientes y ser controlados por los timers 6 y 7. Ambos canales pueden tener peticiones Software DMA haciendo que en general el rendimiento y el procesamiento de datos sea mayor, la resolución del DAC es de 12 bits

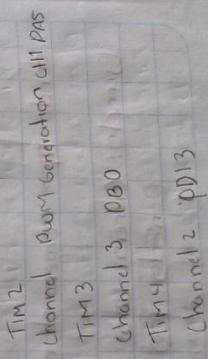
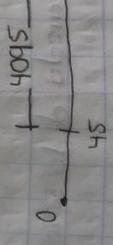
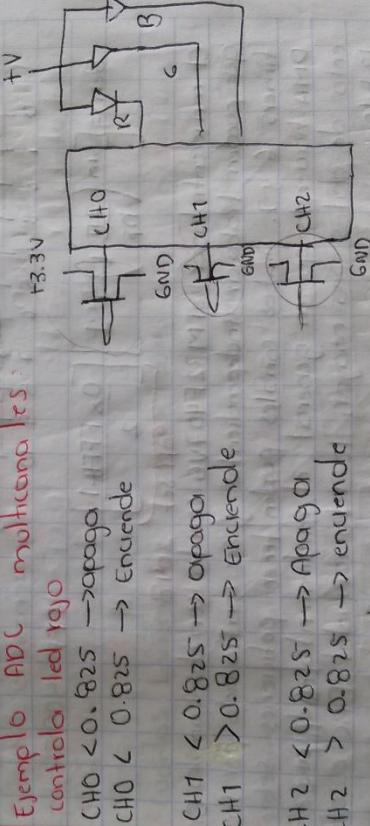
dónde  $n_p$  = número de muestras en un periodo de lo señal

Ejemplo: Generar una señal senoidal a 100 Hz y un periodo 100 muestras

- 1: add click
- 2: DMA Request se selecciona la opción ADC
- 3: modo: seleccionar ADC1 (en azul) y seleccionar en modo circular.
- 4: Data width: se mantiene la opción por default Half word una palabra 32 bits 12 bits media palabra media palabra cable serial

- HAL\_ADC\_Start\_DMA(&hadc1, (uint32\_t\*) lectura\_adc, 3),
- ptrero correspondiente al adc1
- Variable buffer que almacenará la lectura del ADC en 32 bits. (3 canales)

### Ejemplo ADC multicanal res:



18-oct-2017

```

    int oport redondear enteros
    if (resultado < 4095) {
        datos[K] = resultado
    } else {
        datos[K] = 4095
    }

```

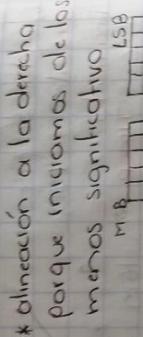
Trigger Event Selection update event

23-septiembre - 2017

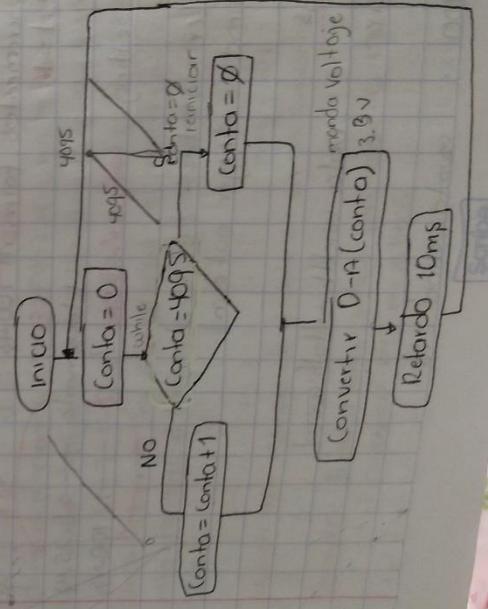
### Dac (Convertidor digital analógico)

Generara una señal diente de sierra. Se debe tener en cuenta que esta señal

DAC configuration  
output buffer Enable None  
Trigger None



\* alineación a la derecha  
porque inviertemos de los  
mismos significativa

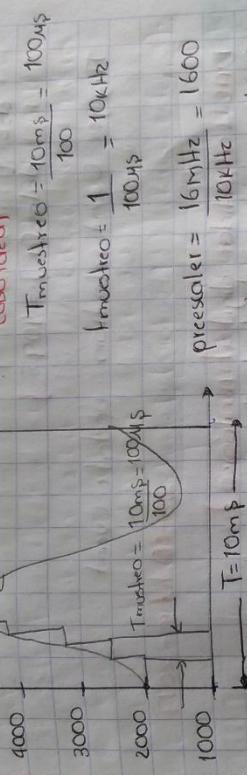


18-oct-2017

Timer va estar contando el tiempo de maestro en que va estar generando este señal. Intervalos donde va estar generando cada muestra.

Timer va contar 100us

caso ideal



$$T_{muestreo} = \frac{10\text{ms}}{100} = 100\mu\text{s}$$

$$f_{muestreo} = \frac{1}{100\mu\text{s}} = 10\text{kHz}$$

18-oct-2017

```

    if (resultado < 4095) {
        datos[K] = resultado
    } else {
        datos[K] = 4095
    }

```

Trigger Event Selection update event

23-septiembre - 2017

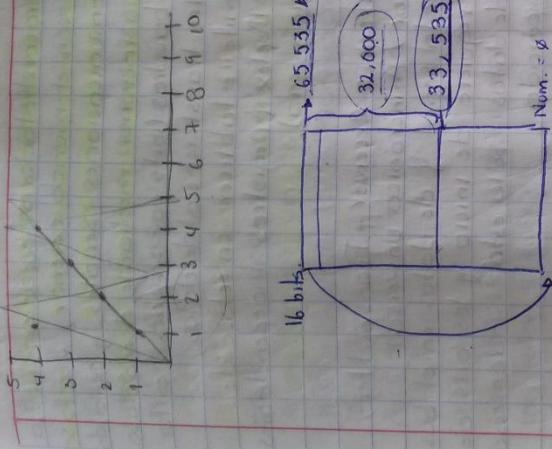
### Dac (Convertidor digital analógico)

Generara una señal diente de sierra. Se debe tener en cuenta que esta señal

DAC configuration  
output buffer Enable None  
Trigger None

-

### Tercer parcial



# de Incrementos.

### Tercer parcial

1 - Noviembre - 2017

**Investigar sobre el USART y UART**

USART Tiene la capacidad de actuar de forma síncrona o asíncrona dependiendo que los datos de velocidad de reloj. El reloj es recuperado a partir de los datos en sí o se envía como una señal externa. Los datos son regulares y los bits de sincronización con la señal del reloj. No arranca y parada de bits utilizados. Esto permite una mayor velocidad en baudios cuando operan sincrónicamente, ya que pocos intervalos de tener una cierta garantía y más bits de puede utilizar para datos en lugar de como encabezados.

- USART (Universal Synchronous Asynchronous Receiver Transmitter) Receptor / Transmisor Síncrono / Asíncrono Universal

1 - Noviembre - 2017

**UART (Universal Asynchronous Receiver - Transmitter, Receptor / Transmisor / Asíncronico Universal)**

Se tiene un reloj interno de la señal y de datos en el bus serie. Puede tener algo más, requiere de inicio y los bits de datos sólo están sincronizados con el inicio y bits de parada.

**UART (Transmisor - Receptor Serie Síncrono - Asíncrono Universal)**

Transmitir o recibir datos en serie (Interfaz de comunicación serie)

**Comunicación Serie**

- Síncrono (half-duplex) utiliza una señal de reloj y una línea de datos. Se suele utilizar cuando la distancia entre el emisor y receptor es pequeña.
- Protocolos I<sub>2</sub>C - SPI - I<sub>2</sub>S
- Asíncrono (full-duplex)
- No se envía señal de reloj, por lo que el emisor y receptor deben tener relojes con la misma frecuencia y fase. Se suele utilizar cuando la

UART

Vel

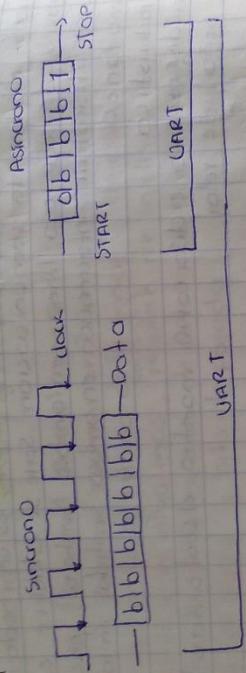
o

re

## Transmisión Asíncrona UART

1-Nov-2017

- Estado Inactivo la linea de datos permanece en alto
- Estado alto cada transmisión de datos comienza con un bit de anaque (start), el cual siempre es cero.
- cada dato tiene un ancho de 9 bits (frame)
- cada bit menos significativo LSB se transmite el bit menor de dato determinado con un bit cada transmisión de dato (stop) el cual siempre es uno.



El más utilizado es RS-232

### Uso del protocolo RS-232

Tiene pellizcos hacia el DMA (Acceso Directo a Memoria) haciendo que el proceso de lectura o escritura sea muy eficiente.

#### STM32cube 1. Activar el USART 3

- Mode : Asynchronous
- Hardware Flow control (RS-232). Disable
- \* Se activan los pinos correspondientes a Rx (Receptor) Tx (Transmisión) parámetro se configura Baud Rate 115200 Bits/s aumentamos la velocidad de transmisión

## Protocolo de red

### Protocolo de red

atollo  
funcional  
HAL\_UART  
UART

HAL\_UART  
UART3  
size  
Timeout  
PortID  
Parity  
HAL\_UART  
UART3  
strlen

Ejemplo  
Descripción  
Transmisor  
A  
B

### UART

No  
funciona

1) #

2) #

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

-

Scribo

Scribo

Scribo

Scribo

Scribo

Scribo

## Transmisión Síncrona, USART

distancia entre el emisor y receptor es grande  
Protocolo - RS-232 - USB - Con Bus,

comunicación serie consiste en enviar los datos bit a bit a través de un línea común.

Transmisión Síncrona, USART

Modo síncrono se permite la transmisión continua de datos y no existe un límite de tamaño, es un modo half-duplex la comunicación Serie se establece a través de una sola linea, en ambas sendidas, por lo que puede transferir información en ambas sendidas en forma simultánea

- un dispositivo actúa como maestro, el cual genera la señal de reloj

modo síncrono

HAL\_UART

UART

size

Timeout

PortID

Parity

HAL\_UART

UART3

strlen

Ejemplo

Descripción

Transmisor

A

B

### USART

Se emplea relojes tanto en el emisor y el receptor. Ambos relojes deben ser igual frecuencia y debe estar en fase

O sincronizados

Cada trama de datos tiene un formato fijo y fijo (start) y un bit final) un bit de inicio o arranque (start) y un bit final) o de paro (stop) que permite realizar la sincronización

La transmisión es modo full-duplex, se utilizan dos líneas.

- se puede transmitir y recibir información en forma simultánea

Scribo

Scribo

Scribo

Scribo

Scribo

Scribo

Scribo

Scribo

Scribo

1-nov-2017

### atollie principales funciones

```
HAL_UART_Peceive();  
HAL_UART_Receive(&uart, uint8_t *pData, uint16_t size,  
uint32_t Timeout)  
{  
    // Recibe  
    // Señal de cantidad de datos a recibir  
    // Tiempo de duración que se va esperar el dato  
    // Pdato : de 8bit - el dato que se va enviar  
    // para transmitir  
    HAL_UART_Transmit();  
}
```

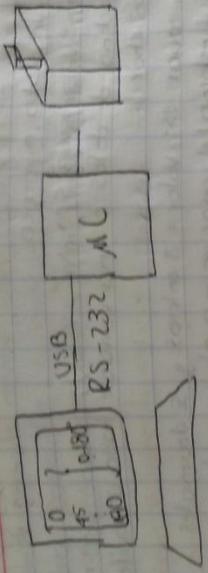
1-nov-2017

### Programa UART Asincrono

```
/* USER BEGIN includes */  
// librerias para manipular datos salida/entrada para  
puertos  
#include <stdio.h>  
#include <String.h>  
/* USER CODE BEGIN 2 */  
// Se declara la variable y debe ser tipo buffer vector  
uint8_t Data_Recibido[1];  
// Puede ser char o uint8_t es un arreglo de 30 bytes  
char buffer[30]; // Solo tenemos para 30 caracteres  
float Valor = 3.14156; // El valor numerico flotante a enviar  
  
while  
/* USER CODE BEGIN 3 */  
// Instrucción para recibir datos por el puerto  
HAL_UART_Receive(&uart3, Data_Recibido, (uint16_t)1, (uint32_t)  
// puntero, variable que contiene el dato, cantidad de bytes a recibir  
if (Data_Recibido[0] == 'A') {  
    HAL_GPIO_WritePin(lcd_azul_GPIO_Port, lcd_azul_Pin, GPIO_PIN_SET);  
}  
else if (Data_Recibido[0] == 'R') {
```

- 2) Declaración de las variables locales
  - Variable o apuntador que contendrá los datos provenientes del puerto (el carácter que se recibe desde la PC)
  - Del puerto (el carácter que se recibe desde la PC)
  - Tipo buffer
  - Variable o apuntador que contiene los datos a ser enviados, tipo char.

**práctica** Enviar el ángulo de posición para el servo



1. ¿Qué variable hace mapear el acelerómetro y el giroscopio?  
2. En qué unidades entrega el acelerómetro y el giroscopio la medición?

3. Escribe el comando y la dirección de los registros en los que se entregan las mediciones del acelerómetro, giroscopio y sensor de temperatura.  
4. ¿Cuáles son los rangos de escala completa y sensibilidad con los que opera el acelerómetro y el giroscopio?  
5. ¿Cuál es la expresión para convertir a grados Celsius la medida entregada por el sensor de temperatura?

10 - NOV - 2017

Velocidad angular (giroscopio)  
aceleración (acelerómetro)

$$2 \text{ giroscopio es grado/Segundo} \Rightarrow \frac{\theta}{s}$$

$$g \rightarrow \text{gravedad es una aceleración } 9.8 \text{ m/s}^2$$

$$q = 9.8 \text{ m/s}^2$$

desplazamiento angular es un giro.  
y esto hablando de una velocidad

HAL - 6PIO - writePort (led\_on) - 6PIO - Port, led\_on - Pin, 6PIO, Pin\_Set;  
HAL - 6PIO - writePin (led\_top - 6PIO - Port, led\_top - Pin, 6PIO, Pin\_RESET);

```

3 // para enviar un dato numérico
spints (buffer, "Valor= %3.5f \n\r", Valor);
// envia una cadena de datos o una variable
// como un buffer o
// arreglo de tipo char
// la cadena o variable es: Valor = 3 enteros y 5 decimales
// \n\r inter y reborro (el cursor regresa al inicio de la linea)

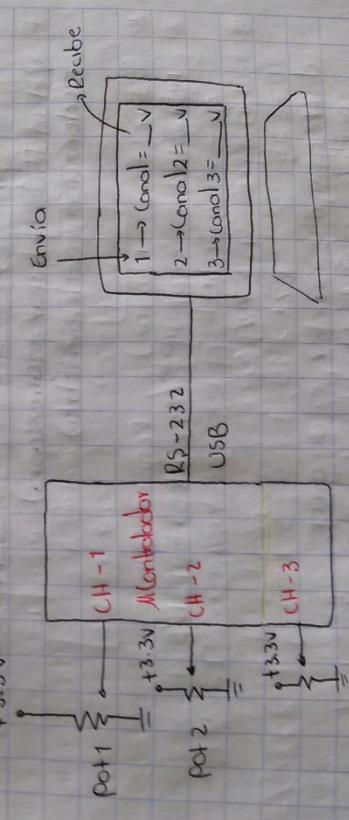
```

**instrucción para transmitir**

HAL\_UART\_Transmit (8uart), 3, (uint8\_t\*) buffer, (uint16) strlen(buffer),
(uint32\_t) 100);
// strlen = calculo cuantos datos hay almacenados en el arreglo
// buffer y devuelve el valor
// argumentos de entrada
// 1) puntero, variable que contiene los datos a enviar, cantidad de
// caracteres contenidos en buffer

Actividad

+3.3V



6 - Noviembre - 2017

Accelerometro		Giroscopio		
	AFS_SEL	Full Scale Range	FS_SEL	Full Scale Range
4:	0	$\pm 2g$	0	$\pm 250^{\circ}/s$
	1	$\pm 4g$	1	$\pm 500^{\circ}/s$
	2	$\pm 8g$	2	$\pm 1000^{\circ}/s$
	3	$\pm 16g$	3	$\pm 2000^{\circ}/s$

Sensibilidad del Accelerómetro      Sensibilidad del Giroscopio

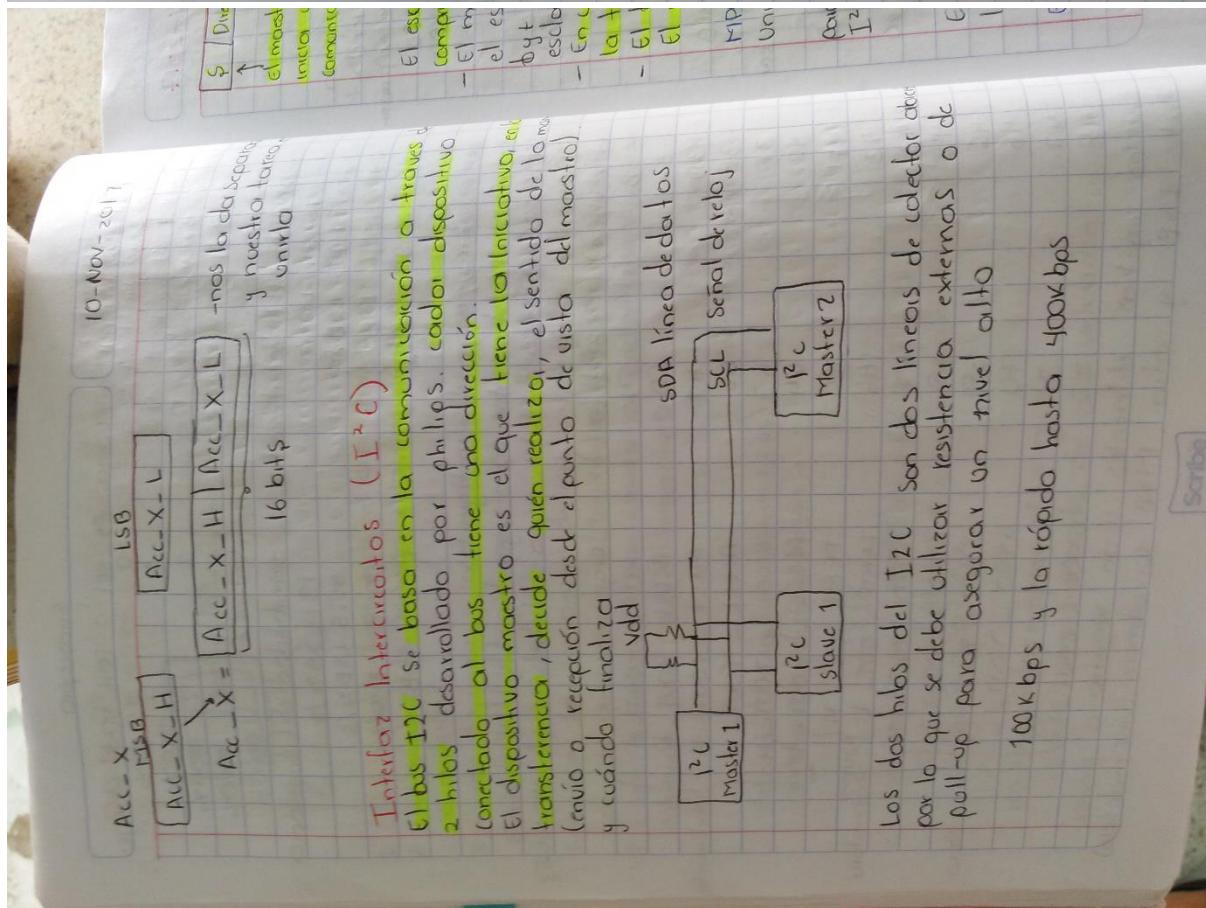
16384	LSB/g	131	LSB%/fs
8192	LSB/g	65.5	LSB%/fs
4096	LSB/g	32.8	USB%/fs
2048	LSB/g	16.4	LSB%/fs

5- Temperatura en grados centigrados  
 (TEMP\_OUT register value as a signed quantity) / 340 + 36.53  
 (TEMP\_OUT register de valor como cantidad Entera) 340 + 36.53

6- Registro nombre

51	ACCEL_XOUT_H
60	ACCEL_YOUT_H
61	ACCEL_ZOUT_H
62	GYRO_XOUT_H
63	GYRO_YOUT_H
64	GYRO_ZOUT_H
65	TEMP_OUT_H
66	TEMP_OUT_L
67	GYRO_XOUT_L
68	GYRO_YOUT_L
69	GYRO_ZOUT_L
70	GYRO_RZOUT_L
71	GYRO_RYOUT_L
72	GYRO_RXOUT_L

En dos registros diferentes y un registro o paro  
 bits LSB y otro MSB conteneno (un en uno solo)



10 - NOV - 20

### 1. I<sup>2</sup>C dirección

I<sup>2</sup>C - Master - Transmisor (escuchar)

I<sup>2</sup>C - Master - Escuchar - Leer)

- en automático adiciona y cambian el bit ADD, para que el maestro realizar un comando a la memoria o dispositivo (obligado)
- Dirección (obligada)
- Registro de dirección
- el bit ADD
- para leer

0x00 x,

- tiene un bit de último bit lo va poner la instrucción
- la dirección es el hexadecimal D0
- Esclavo 0x00 para recibir y transmitir.

### 2. power Management

el registro power management, su dirección es 0x6B

- permite al usuario configurar el modo de energía y la fuente de relé. También proporcionar un bit para restar todo el dispositivo (Device RESET) y un bit para deshabilitar o sensor de temperatura (TEMP\_OIS).

- Es desactivar el bit SLEEP ( $SLEEP = 0$ )
- se diracción 6B accedemos a estos registros

### 3. Frecuencia de muestreo

Sample Rate Dividez especifica la tasa de muestreo para el Mpu - 6axo. La dirección de este registro es 0x10

10 - NOV - 20

### Register

(Hex)

1q

Todos los

de

4:

Medio

Aleatorio

6mos

M

M

10 - NOV - 20

### Eslavo

1 / Dirección

R/W

ACK

Datos

ACK

Datos

R/W

P

S

Z

N

el ultimo bit para escribir y ser 0. Si stop

primer tronante

la dirección del dispositivo.

conectual se quiere

comunicar

El esclavo genera un bit de reconocimiento ACK , el caca

compacta que la dirección concuerda con la suya

- El maestro también puede recibir datos en ese caso

el es quien genera la señal de reconocimiento ACK por cada

bit recibido, menos para el último en este caso el

esclavo libera la línea de datos y master genera un STOP

- en caso de que el maestro no reciba el ACK se detiene

la transmisión

- El bit indica la acción del maestro (el último bit)

El maestro indica al esclavo

MPU - 6050

Unidad de medida

1 canal

(IMU)

3 - ejes

Dirección del dispositivo (data sheet)

parametros

condiciones

min

typ

max

bits

notes

I<sup>2</sup>C ADDRESS

ADD = 0

1101000

A DO = 1

1101001

Dirección (0x5)

DO = 0

0110 1000

0x68

DO = 1

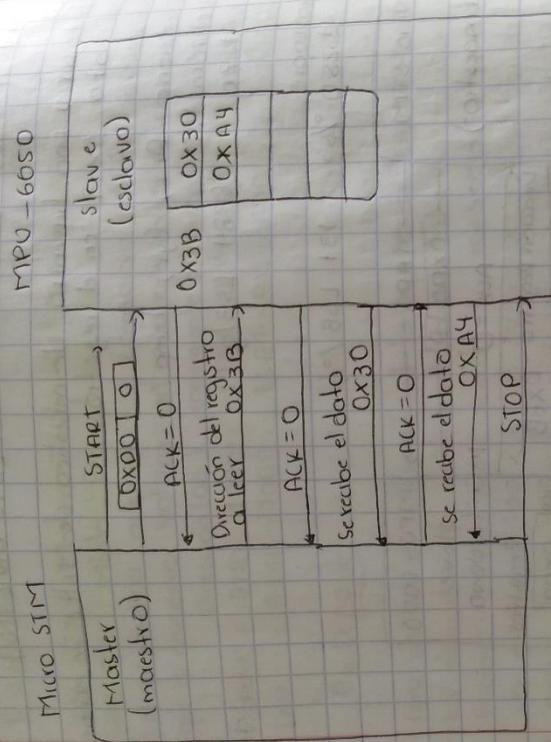
0110 1001

0x69

El maestro va a escribir o leer. va mandar la dirección

va en el último bit.

10-Noviembre-17  
Leer 2 datos almacenados en los registros a partir  
dirección CS Ox 3B



Siempre debe enviar un ACK para todos los demás datos para el último. A través de una dirección un banco de registros tiene una dirección

Atollie,  
HAL\_I2C\_Master\_Transmit  
(I2C\_Handle, uint16\_t DevAddress, uint8\_t \*PData,  
uint16\_t Size, uint32\_t Timeout)

HAL\_I2C\_Master\_Receive

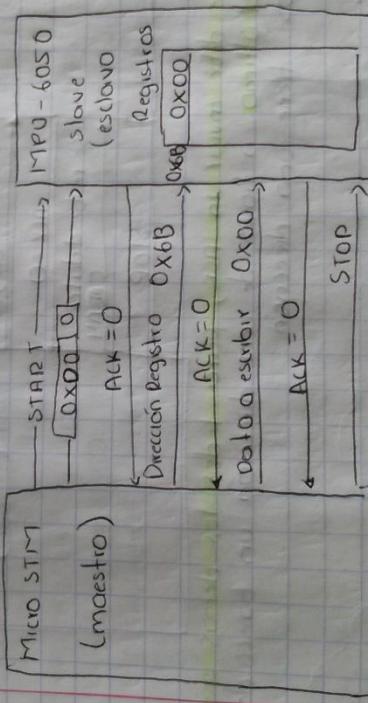
float - valor - 0;

Register (Hex)	Register (Decimal)	B17	B16	B15	B14	B13	B12	B11	B10	SMPLRT - Div [7:0]
19 25										

Trabajar a una frecuencia de muestreo de 1kHz se escribe el dato 0xa7 en el registro.

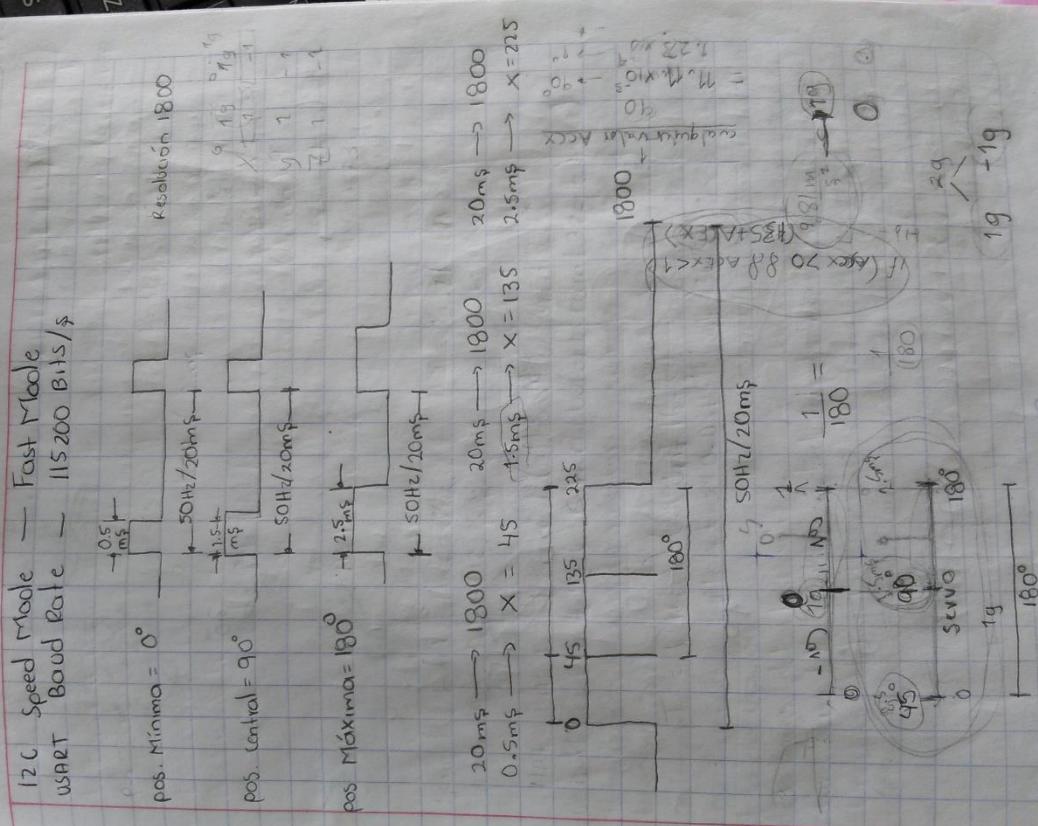
4: Mediciones Acc, Gyr, Temp  
Acelerómetro  
± 2g 16384 LSB/g

Giroscopio  
± 250 °/s 131 LSB / °/s



Dirección del dispositivo  
Dirección del registro de lo cual va escribir 6B  
Dirección del dato

parameter



Escribir / Leer en la EEPROM

4 / Dic / 2017

Memoria EEPROM

4 kB

32 bits

Dirección  
0XA0

almacenar 4000 espacios  
de memorias

A0 → Hexadecimal  
comunicación es por I<sup>2</sup>C

Activar I<sup>2</sup>C  
USART 3

Elas-2019-08  
 HAL\_I2C\_Mem\_Write(8h12c1, ADDMEMORY, 0x00,  
 32, datos, 10, 50)

Tiempo de espera  
 50ms

puntero

Dirección de la memoria

cada dato que se va ir guardando

para escribir (guardas el dato de la memoria)

HAL\_I2C\_Mem\_Write(8h12c1, ADDMEMORY, 0x00,  
 32, datos, 10, 50);

es útil arranca todo inicialmente

- para leer (de manera infinita)

HAL\_I2C\_Mem\_Read(8h12c1, ADDMEMORY, 0x00, 32,  
 datos\_in, 10, 50);

3: la dirección a partir de la cual voy a leer  
 4: 32 bits establece la memoria  
 5: variable que esta guardando los datos que voy leer

Projecto

vaso  
 tequilero  
 (sin tequila)

ophyro

Bluetooth HC05

Serial

A

B

C

D

Scribe

- Si el robot se del proyecto va
 - Si solo logra
 - Si avanza d
 - Si hace lo aplicación
 - Todo par

Lunes 11  
 Lunes 18  
 Miércoles  
 11:00 a

Paso 1  
 Activar  
 Mode : T  
 PA5 -  
 PA7 -

Paso 2  
 Activar  
 PD7  
 PD6  
 PA15  
 PA7

Paso 3  
 geopic  
 SRC  
 Inc

Paso  
 conf

- Si el robot se lleva equilibrar con el vaso su calificación del proyecto va ser 70

- Si solo logra avanzar con el vaso 2 metro 80
- Si avanza 1 metro y de reversa regresa 85
- Si hace lo mismo con el punto 3 pero con la aplicación Bluetooth 90
- Todo por Bluetooth si caer el vaso 100

Lunes 11 Diciembre la estructura.

Lunes 18 Diciembre Avances

Miercoles 20 Diciembre 2017 Hora de la clase.  
11:00 am (límite)

### PASO 1

Activar el SPI1

Mode : Transmit only Master Solo va recibir.

PA5 — SPI1\_MOSI — PB5 salida de datos SPI1\_MOSI

PA7 — SPI1\_BCK — PB3 reloj

SPI1\_SCK

### PASO 2

Activar como salidas

PD7 — TFT\_Reset GPIO\_OUTPUT

PD6 — TFT\_RS

PA15 — SPI1\_CS

PA7 — TFT\_BckLight controlar la luz de la pantalla

### PASO 3

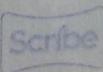
describir y pegar los archivos correspondientes

SRC — carpeta

Inc — carpeta

### PASO 4

configurar el proyecto



configurar el proyecto de forma normal, agregar lib.  
estándar.

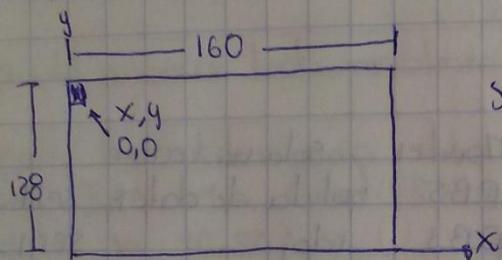
- C Linker
- libraries - Add → :libst7735-Driver.a - indicando la librería que se va incluir
- Add → "\${workspace\_loc:/\${ProjName}}/src"

Inc - MapPIN-Ophyro

RGB tiene 255

U32 → 32 bits

FF - 1 byte - 8bit - color rojo



Siempre contemplado

$$\text{prescaler} = \frac{16\text{MHz}}{(15\text{kHz})(100)} - 1 = 9.5 \approx 10$$

resolución = 100

comunica Bluetooth = 9600 B.t/s

$$\frac{9.5}{50} = 0.19$$

$$\frac{9.5}{80} = 0.11875$$