
PUCP

PROGRAMACIÓN DE MICROCONTROLADORES ARM

FABRICUM - PUCP

Sesión 5 - 16/07/2024:

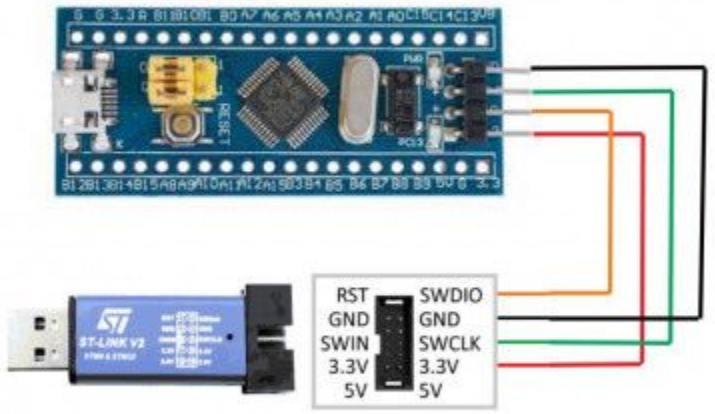
- Puertos de Propósito General
  - Descripción del módulo
  - Registros de configuración
  - Ejemplos de aplicación
- Módulo de comunicación UART
  - Descripción del funcionamiento
  - Registros de operación
  - Envío y recepción de caracteres
  - Envío de cadenas

Programación con punteros

Programación de microcontroladores ARM - Sesión 5

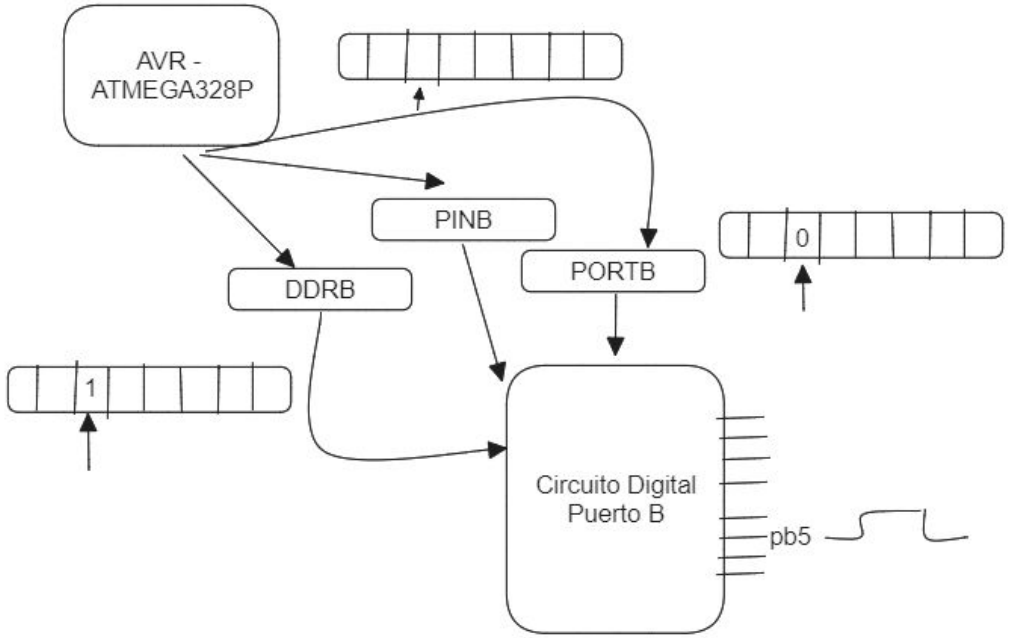
PUCP

Conexiones ST-LINK - STM32F103C8T6



De la clase anterior …

Los microcontroladores son circuitos digitales complejos y síncronos en donde la modificación de los registros de memoria alteran el comportamiento de los diferentes periféricos internos. El término "periféricos programables" se debe a que esta alteración del comportamiento es ordenado y sistemático a través de una serie de reglas bien documentadas en los "Datasheet"



Del ejemplo anterior …

Para el microcontrolador STM32F103C8T6 se verificó en la hoja de datos las direcciones en memoria creando los punteros correspondientes y modificando los bits necesarios para generar parpadeos en el puerto PC13:

- Memory Map
- Punteros apuntando a registros

```c
#include <stdint.h>

int main(void)
{
  //Creamos punteros a los registros:
  uint32_t* clk_reg = (uint32_t*) 0x40021018;
  uint32_t* puertoC_mode_reg = (uint32_t*) 0x40011004;
  uint32_t* puertoC_output_reg = (uint32_t*) 0x4001100C;

  *clk_reg |= (1 << 4);
  *puertoC_mode_reg &= 0xFF0FFFFF;
  *puertoC_mode_reg |= 0x00200000;
  *puertoC_output_reg |= 0x00002000;


    /* Loop forever */
  while(1)
  {
    for(int i = 0; i < 200000;i++);
    *puertoC_output_reg |= 0x00002000;
    for(int i = 0; i < 200000;i++);
    *puertoC_output_reg &= 0xFFFFDFFF;
  }
}
```

TM4C123GH6PM

Para el microcontrolador TM4C123GH6PM se sigue el mismo proceso, en este caso se tienen otros registros pero la lógica de programación sigue siendo la misma. Este es un acercamiento de bajo nivel a los microcontroladores permitiendo su generalización en la programación

10.3 Initialization and Configuration

The GPIO modules may be accessed via two different memory apertures. The legacy aperture, the Advanced Peripheral Bus (APB), is backwards-compatible with previous devices. The other aperture, the Advanced High-Performance Bus (AHB), offers the same register map but provides better back-to-back access performance than the APB bus. These apertures are mutually exclusive. The aperture enabled for a given GPIO port is controlled by the appropriate bit in the **GPIOHBCTL** register (see page 258). Note that GPIO can only be accessed through the AHB aperture.

To configure the GPIO pins of a particular port, follow these steps:

1. Enable the clock to the port by setting the appropriate bits in the **RCGCGPIO** register (see page 340). In addition, the **SCGCGPIO** and **DCGCGPIO** registers can be programmed in the same manner to enable clocking in Sleep and Deep-Sleep modes.
2. Set the direction of the GPIO port pins by programming the **GPIODIR** register. A write of a 1 indicates output and a write of a 0 indicates input.
3. Configure the **GPIOAFSEL** register to program each bit as a GPIO or alternate pin. If an alternate pin is chosen for a bit, then the `PMCx` field must be programmed in the **GPIOPCTL** register for the specific peripheral required. There are also two registers, **GPIOADCCTL** and **GPIODMACTL**, which can be used to program a GPIO pin as a ADC or µDMA trigger, respectively.
4. Set the drive strength for each of the pins through the **GPIODR2R**, **GPIODR4R**, and **GPIODR8R** registers.
5. Program each pad in the port to have either pull-up, pull-down, or open drain functionality through the **GPIOPUR**, **GPIOPDR**, **GPIOODR** register. Slew rate may also be programmed, if needed, through the **GPIOSLR** register.
6. To enable GPIO pins as digital I/Os, set the appropriate `DEN` bit in the **GPIODEN** register. To enable GPIO pins to their analog function (if available), set the `GPIOAMSEL` bit in the **GPIOAMSEL** register.

Memory map y direcciones

Para cualquier microcontrolador que se desee programar en un bajo nivel se debe tener en cuenta las hojas de datos y las direcciones de los diferentes registros. Los periféricos programables son circuitos digitales que tienen un espacio de memoria (registros) para la modificación de su comportamiento y así generar o leer señales electrónicas las cuales son interpretadas por el código del programador.

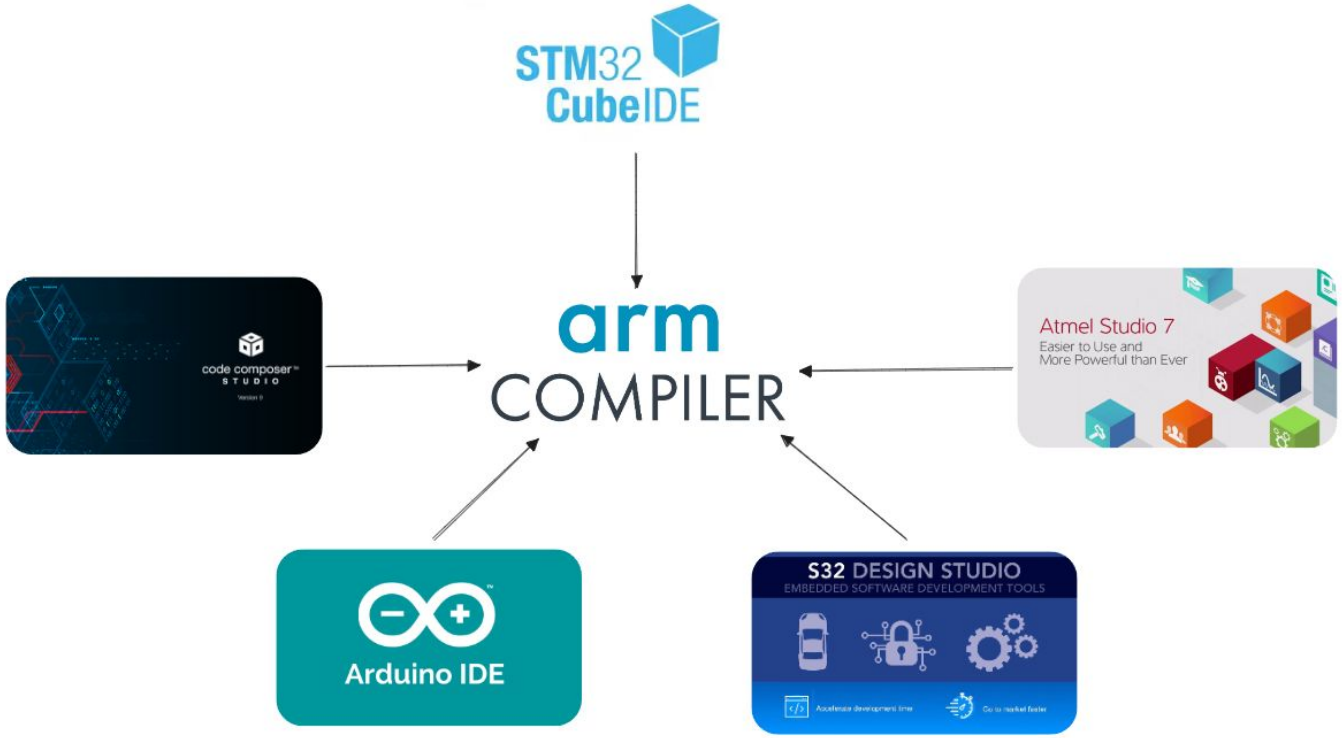
Table 2-4. Memory Map

Start	End	Description	For details, see page ...
Memory			
0x0000.0000	0x0003.FFFF	On-chip Flash	540
0x0004.0000	0x1FFF.FFFF	Reserved	-
0x2000.0000	0x2000.7FFF	Bit-banded on-chip SRAM	525
0x2000.8000	0x21FF.FFFF	Reserved	-
0x2200.0000	0x220F.FFFF	Bit-band alias of bit-banded on-chip SRAM starting at 0x2000.0000	525
0x2210.0000	0x3FFF.FFFF	Reserved	-
Peripherals			
0x4000.0000	0x4000.0FFF	Watchdog timer 0	776
0x4000.1000	0x4000.1FFF	Watchdog timer 1	776
0x4000.2000	0x4000.3FFF		
0x4000.4000	0x4000.4FFF		
0x4000.5000	0x4000.5FFF		

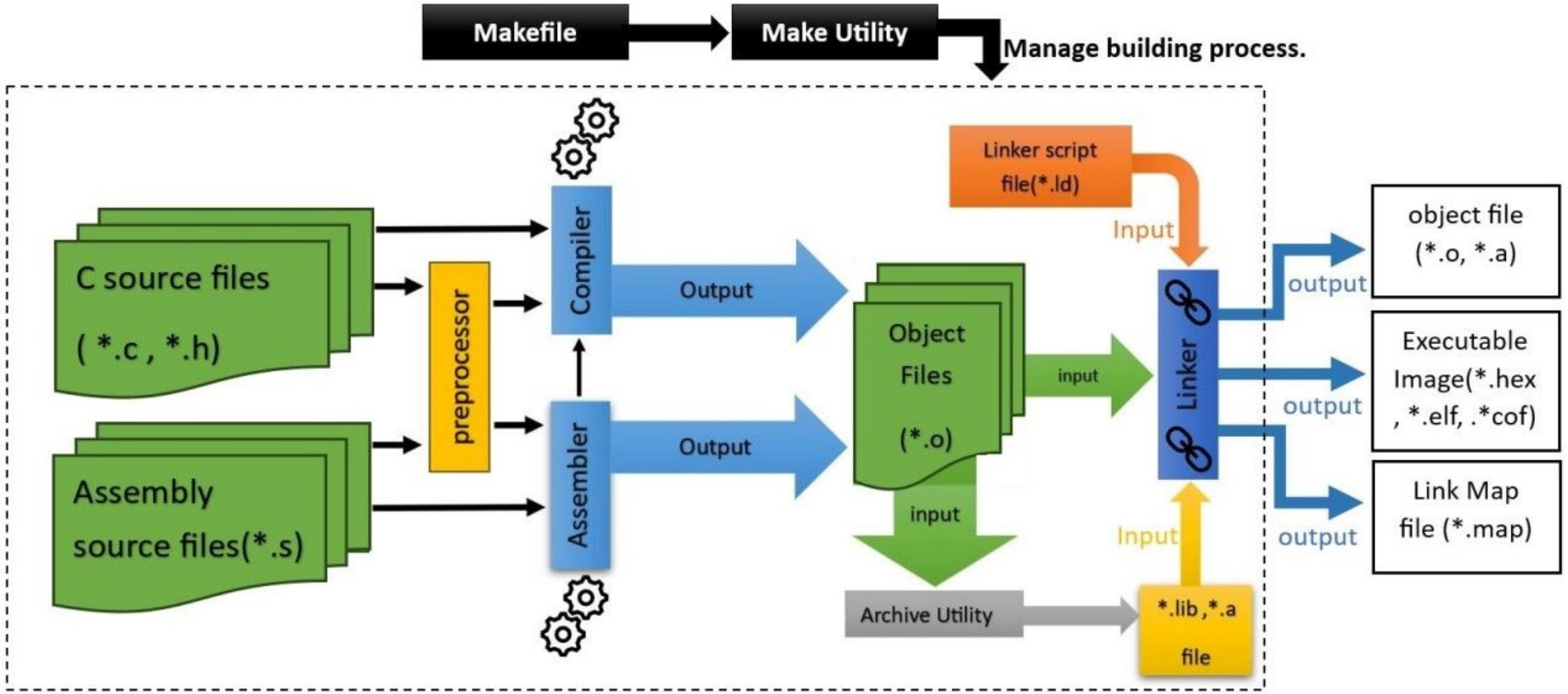
4.2.4 Instantiation

ID	Base address	Peripheral	Instance	Description	
0	0x40000000	CLOCK	CLOCK	Clock control	
0	0x40000000	POWER	POWER	Power control	
0	0x50000000	GPIO	GPIO	General purpose input and output	Deprecated
0	0x50000000	GPIO	P0	General purpose input and output, port 0	
0	0x50000300	GPIO	P1	General purpose input and output, port 1	
1	0x40001000	RADIO	RADIO	2.4 GHz radio	
2	0x40002000	UART	UART0	Universal asynchronous receiver/transmitter	Deprecated
2	0x40002000	UARTE	UARTE0	Universal asynchronous receiver/transmitter with EasyDMA, unit 0	
3	0x40003000	SPI	SPI0	SPI master 0	Deprecated
3	0x40003000	SPIM	SPIM0	SPI master 0	
3	0x40003000	SPIS	SPIS0	SPI slave 0	
3	0x40003000	TWI	TWI0	Two-wire interface master 0	Deprecated
3	0x40003000	TWIM	TWIM0	Two-wire interface master 0	
3	0x40003000	TWIS	TWIS0	Two-wire interface slave 0	
4	0x40004000	SPI	SPI1	SPI master 1	Deprecated
4	0x40004000	SPIM	SPIM1	SPI master 1	

Entornos de programación



Estructura de compilación y carga



Carga para STM32

Para cargar la información a un microcontrolador se realizan 3 pasos:

- Compilación a archivo .elf
- Generación del ejecutable .bin
- Carga al microcontrolador

Generación del archivo .elf

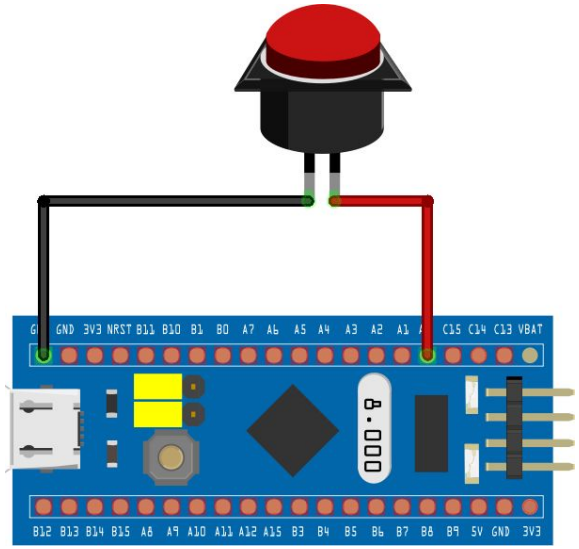
- arm-none-eabi-gcc
- -mcpu=cortex-m3
- -mthumb
- –specs=nosys.specs
- -T
- stm32f103c8tx_flash.ld
- startup_stm32f103c8tx.s
- parpadeo.c
- parpadeo.elf

Generación del ejecutable .bin

- arm-none-eabi-objcopy
- -O binary
- parpadeo.elf
- parpadeo.bin

Carga del archivo

- st-flash
- write
- parpadeo.bin
- 0x08000000

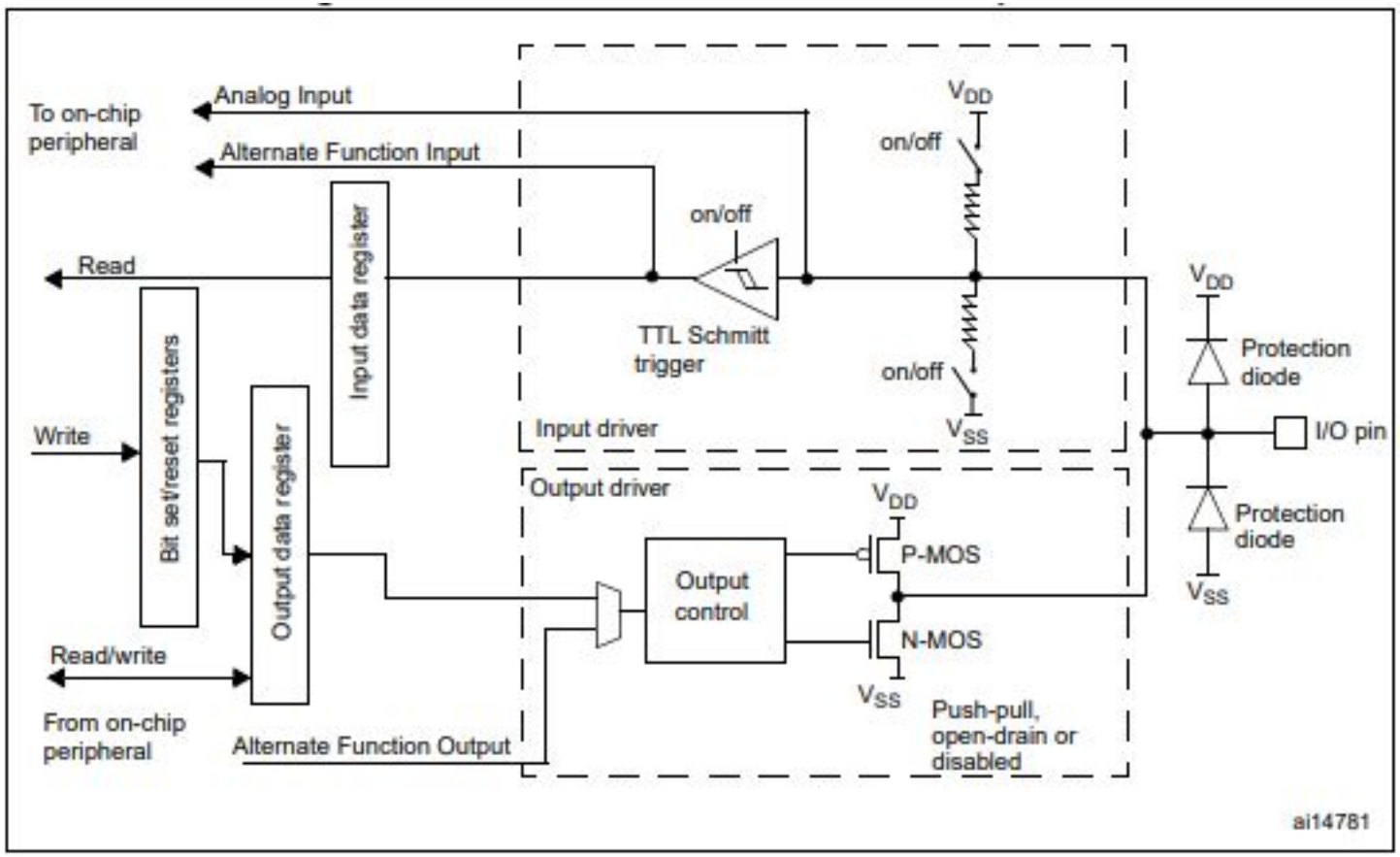


Puertos de propósito general

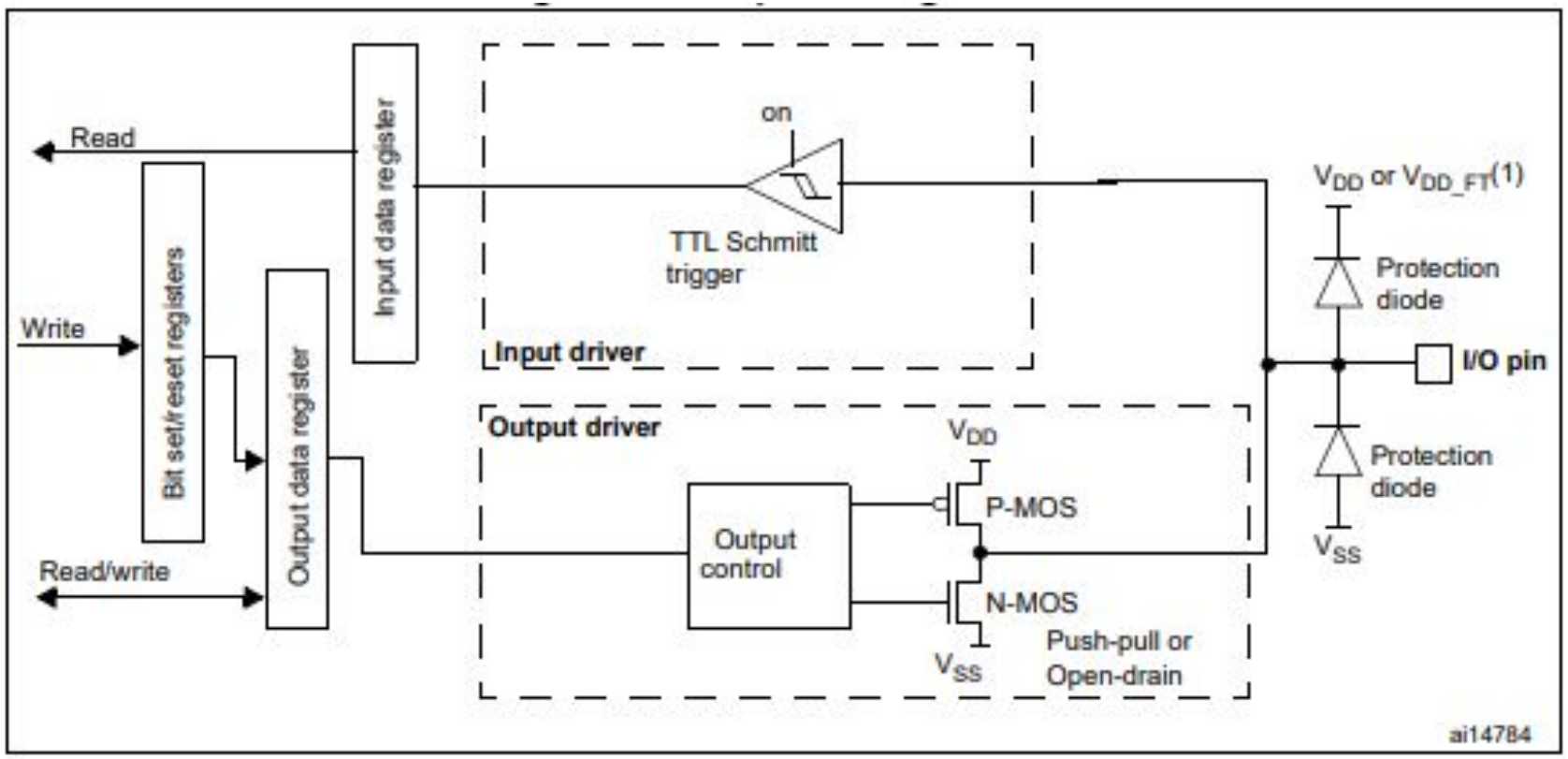
Programación de microcontroladores ARM - Sesión 5

PUCP

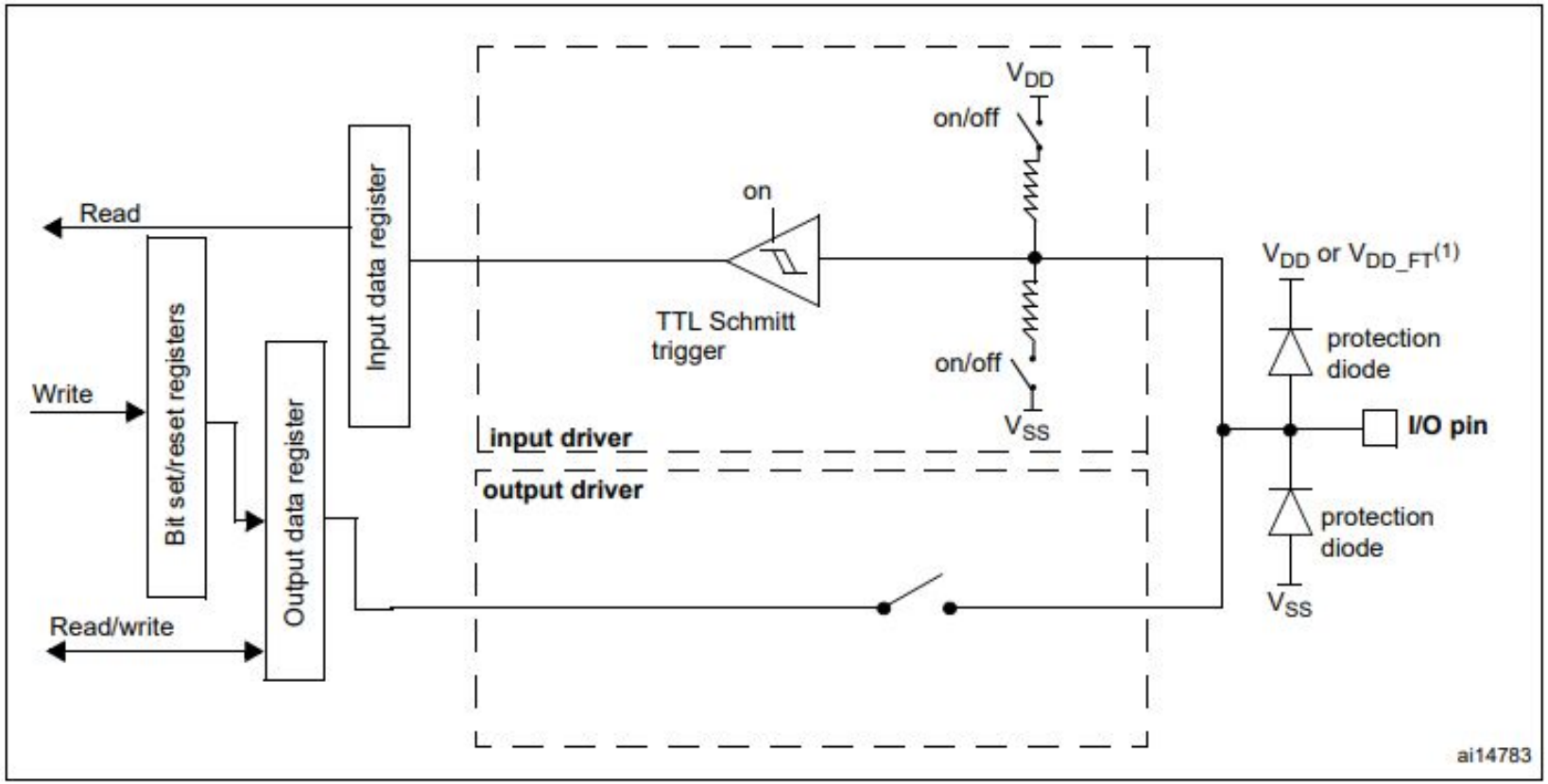
Módulo GPIO


To on-chip peripheral
Analog Input
Alternate Function Input
Read
Write
Read/write
From on-chip peripheral
Bit set/reset registers
Input data register
Output data register
Alternate Function Output
on/off
TTL Schmitt trigger
on/off
on/off
V_{DD}
V_{SS}
Input driver
Output driver
V_{DD}
P-MOS
Output control
N-MOS
V_{SS}
Push-pull, open-drain or disabled
V_{DD}
Protection diode
I/O pin
Protection diode
V_{SS}
ai14781

Configuración como salida



Configuración como entrada



Habilitación de la señal de reloj

7.3.7 APB2 peripheral clock enable register (RCC_APB2ENR)

Address: 0x18

Reset value: 0x0000 0000

Access: word, half-word and byte access

No wait states, except if the access occurs while an access to a peripheral in the APB2 domain is on going. In this case, wait states are inserted until the access to APB2 peripheral is finished.

Note: When the peripheral clock is not active, the peripheral register values may not be readable by software and the returned value is always 0x0.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										TIM11 EN	TIM10 EN	TIM9 EN	Reserved		
										rw	rw	rw			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADC3 EN	USART 1EN	TIM8 EN	SPI1 EN	TIM1 EN	ADC2 EN	ADC1 EN	IOPG EN	IOPF EN	IOPE EN	IOPD EN	IOPC EN	IOPB EN	IOPA EN	Res.	AFIO EN
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw		rw

Configuración del modo de operación

9.2.1 Port configuration register low (GPIOx_CRL) (x=A..G)

Address offset: 0x00

Reset value: 0x4444 4444

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNF7[1:0]		MODE7[1:0]		CNF6[1:0]		MODE6[1:0]		CNF5[1:0]		MODE5[1:0]		CNF4[1:0]		MODE4[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNF3[1:0]		MODE3[1:0]		CNF2[1:0]		MODE2[1:0]		CNF1[1:0]		MODE1[1:0]		CNF0[1:0]		MODE0[1:0]	
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Configuración del modo de operación

Bits 31:30, 27:26, 23:22, 19:18, 15:14, 11:10, 7:6, 3:2 **CNFy[1:0]:** Port x configuration bits (y= 0 .. 7)
These bits are written by software to configure the corresponding I/O port.
Refer to *Table 20: Port bit configuration table*.
In input mode (MODE[1:0]=00):
00: Analog mode
01: Floating input (reset state)
10: Input with pull-up / pull-down
11: Reserved
In output mode (MODE[1:0] > 00):
00: General purpose output push-pull
01: General purpose output Open-drain
10: Alternate function output Push-pull
11: Alternate function output Open-drain

Bits 29:28, 25:24, 21:20, 17:16, 13:12, 9:8, 5:4, 1:0 **MODEy[1:0]:** Port x mode bits (y= 0 .. 7)
These bits are written by software to configure the corresponding I/O port.
Refer to *Table 20: Port bit configuration table*.
00: Input mode (reset state)
01: Output mode, max speed 10 MHz.
10: Output mode, max speed 2 MHz.
11: Output mode, max speed 50 MHz.

Lectura de un puerto

9.2.3 Port input data register (GPIOx_IDR) (x=A..G)

Address offset: 0x08h

Reset value: 0x0000 XXXX

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDR15	IDR14	IDR13	IDR12	IDR11	IDR10	IDR9	IDR8	IDR7	IDR6	IDR5	IDR4	IDR3	IDR2	IDR1	IDR0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **IDRy:** Port input data (y= 0 .. 15)

These bits are read only and can be accessed in Word mode only. They contain the input value of the corresponding I/O port.

Escritura en un puerto

9.2.4 Port output data register (GPIOx_ODR) (x=A..G)

Address offset: 0x0C

Reset value: 0x0000 0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODR15	ODR14	ODR13	ODR12	ODR11	ODR10	ODR9	ODR8	ODR7	ODR6	ODR5	ODR4	ODR3	ODR2	ODR1	ODR0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bits 31:16 Reserved, must be kept at reset value.

Bits 15:0 **ODRy:** Port output data (y= 0 .. 15)

These bits can be read and written by software and can be accessed in Word mode only.

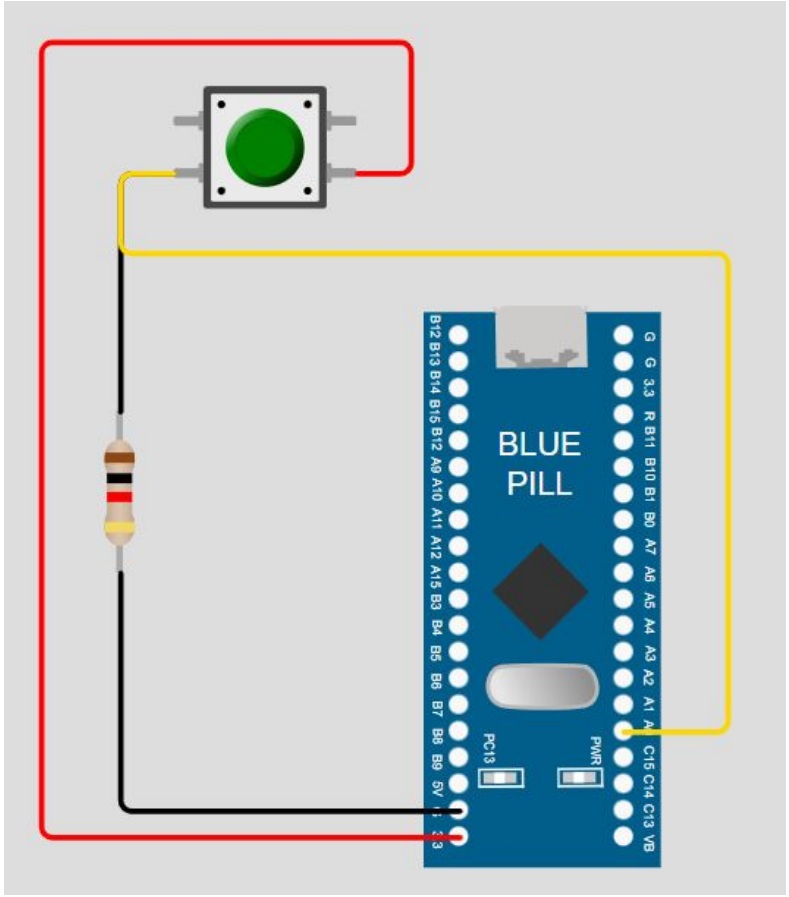
Note: For atomic bit set/reset, the ODR bits can be individually set and cleared by writing to the GPIOx_BSRR register (x = A .. G).

Ejemplos de aplicación

Estos ejemplos se desarrollaran utilizando punteros, hojas de datos y el microcontrolador STM32F103C8T6. Se usarán los archivos de configuración con estructuras.

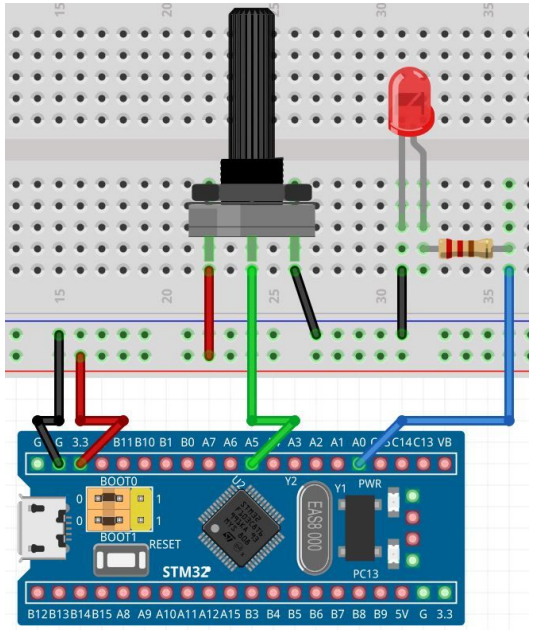
- Lectura de un pulsador y control de un Led
- Problema de aplicación

Lectura de un pulsador y control de un led



Problema de aplicación

Con el pulsador conectado en PA0 implementar un contador que cada 5 pulsaciones emite parpadeos incrementales empezando desde 1 hasta 4. El proceso debe ser cíclico; es decir luego de llegar a 4 se debe reiniciar a 1.



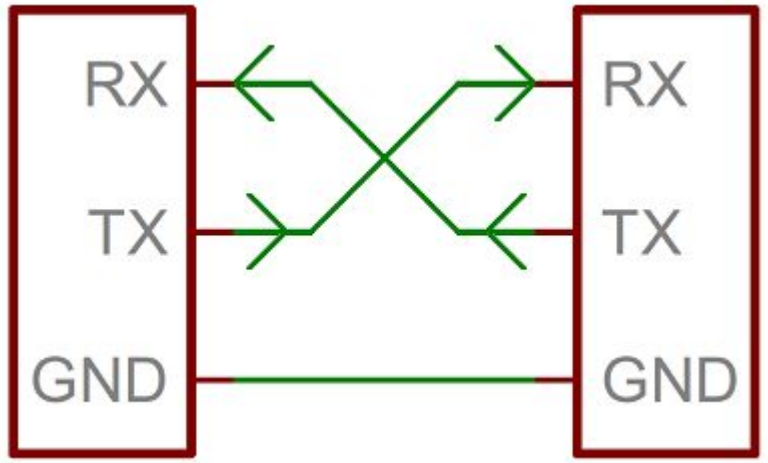
Módulo UART

Programación de microcontroladores ARM - Sesión 5

PUCP

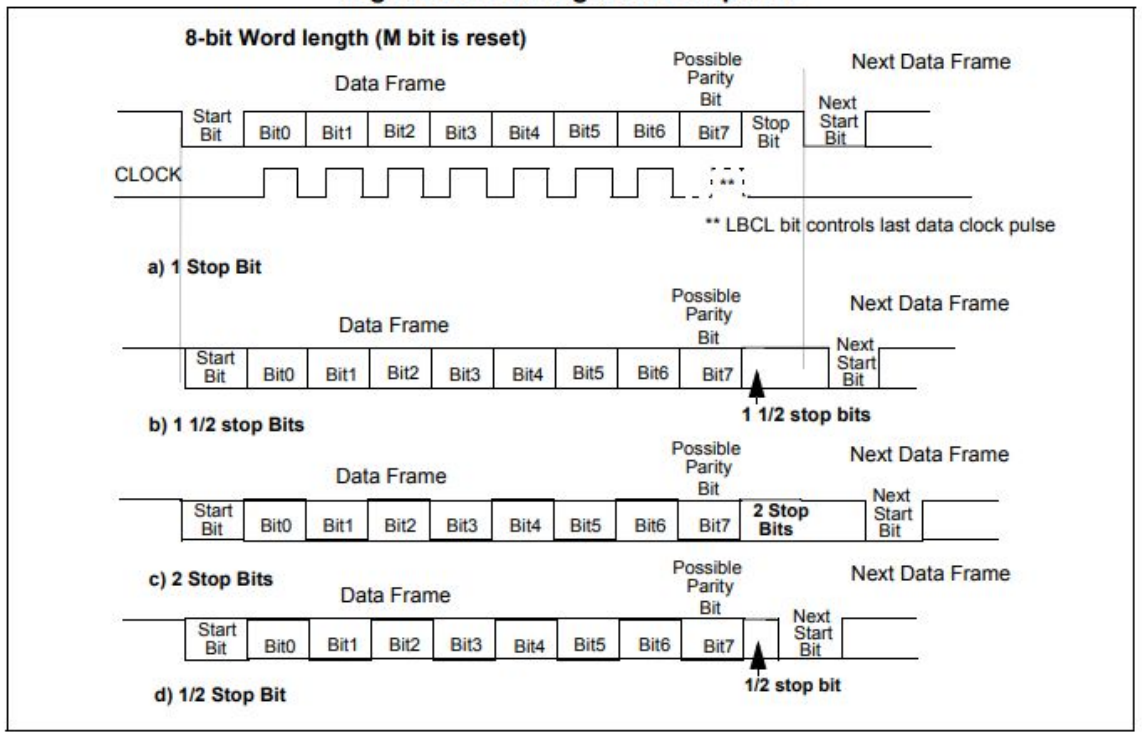
Comunicación serial

Los microcontroladores utilizan señales eléctricas para representar información, desde valores lógicos hasta caracteres, en este último caso se utiliza un formato definido de trama el cual es representado a través de 1's y 0's. El código ascii nos brinda la representación numérica de caracteres.



Código ASCII

Figure 281. Configurable stop bits



Velocidad de transmisión

Los parámetros de recepción y envío deben estar configurados tanto en el dispositivo de recepción como en el dispositivo de envío y deben tener la misma referencia. Un parámetro importante es la velocidad de transmisión el cual se define a partir de la velocidad de reloj del módulo.

$$\text{Tx/ Rx baud} = \frac{f_{CK}}{(16*\text{USARTDIV})}$$

Registro de estado del UART

27.6.1 Status register (USART_SR)

Address offset: 0x00
Reset value: 0x00C0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CTS	LBD	TXE	TC	RXNE	IDLE	ORE	NE	FE	PE
						rc_w0	rc_w0	r	rc_w0	rc_w0	r	r	r	r	r

Registro de control del UART

27.6.4 Control register 1 (USART_CR1)

Address offset: 0x0C

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		UE	M	WAKE	PCE	PS	PEIE	TXEIE	TCIE	RXNEIE	IDLEIE	TE	RE	RWU	SBK
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Registro de control 2 del UART

27.6.5 Control register 2 (USART_CR2)

Address offset: 0x10

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Res.	LINEN	STOP[1:0]		CLK EN	CPOL	CPHA	LBCL	Res.	LBDIE	LBDL	Res.	ADD[3:0]			
	rw	rw	rw	rw	rw	rw	rw		rw	rw	rw	rw	rw	rw	rw

Registro de datos

27.6.2 Data register (USART_DR)

Address offset: 0x04

Reset value: Undefined

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							DR[8:0]								
							rw	rw	rw	rw	rw	rw	rw	rw	rw

Registro de configuración de velocidad

27.6.3 Baud rate register (USART_BRR)

Note: The baud counters stop counting if the TE or RE bits are disabled respectively.

Address offset: 0x08

Reset value: 0x0000

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIV_Mantissa[11:0]												DIV_Fraction[3:0]			
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Ejemplos de aplicación

Para este módulo se desarrollarán los siguientes ejemplos

- Envio de carácter
- Envio de cadena
- Recepción de carácter
- Integración con otros módulos.