```
/*Programa 2 SRC
 * main.c
 *
 *  Created on: Feb 5, 2016
 *      Author: pedro
 */

#include "leon3_uart.h"    //Include del .h donde esta la funcion
#include "leon3_bprint.h"

int main(){
        //Programa 1
        //leon3_putchar('p');         //pasamos un p
        //leon3_putchar('2');         //pasamos un 2
        //leon3_putchar('\n');//pasamos un salto de linea

        //Programa 2
        char * pchar="cadena\n";
        leon3_print_string(pchar);
        leon3_print_uint8(155);

        return 0;
}

/*
 * leon3_bprint.c
 *
 *  Created on: Feb 12, 2016
 *      Author: pedro
 */

#include "leon3_bprint.h"//incluimos las definiciones propias
#include "leon3_uart.h"             //incluimos las funciones de uart

int8_t leon3_print_string(char* str){
        //variables aux que sera el inicio de recorrido
        int cont=0;

        while(str[cont]!='\0'){                         //recorre el arry de char hasta encontrar el \0 que indica el final
                leon3_putchar(str[cont]); //se envia para imprimir el char en el que nos encontramos
                cont++;                                                 //aumentamos el contador para que en la proxima
iteracion se use el proximo caracter
        }
        return 0;
}

int8_t leon3_print_uint8(uint8_t i){
        //variable contador inicializada a 0
        int aux=0;

        if (i<=9){          //caso de solo unidades numero menor a 9
                aux=i;                                  //sacamos las unidades
                leon3_putchar('0'+aux);   //usamos el cacter 0 como base y le sumamos el numero tambien so podria usar
directamente 'X'
        }else if(i>9 && i<=99){     //caso de decenas y unidades
                aux=((i/10)%10); //sacamos las decenas
                leon3_putchar('0'+aux);
                aux=i%10;                               //sacamos las unidades
                leon3_putchar('0'+aux);
        }else{              //caso de centenas decenas y unidades
                aux=((i/10)/10);   //sacamos las centenas
                leon3_putchar('0'+aux);
```

```
		aux=((i/10)%10); //sacamos las decenas
		leon3_putchar('0'+aux);
		aux=i%10;				//sacamos las unidades
		leon3_putchar('0'+aux);
	}
	leon3_putchar('\n');	//salto de linea para que sea mas legible al mostrarse el codigo
	return 0;
}

/*
 * leon3_uart.c
 *
 *  Created on: Feb 5, 2016
 *      Author: pedro
 */

//archivo en el que se encuentra la funcion especicicadas en el .h y usadas por el main

//Declaraciones del uso de lib y demas
#include "leon3_uart.h"

//Especificamos una mascara con la que compararemos para verificar si esta en un estado concreto un valor  de status
tambien se podria usar directamente el numero
#define LEON3_UART_TFF (0x200) //0x200en exa es 512 en decimal que en binario es 0..1000000000 que es el bit que
queremos obtener

//Declaracion del registro y su estructura ademas se especifica donde esta cada dato
//El atributo volatile que fuerza a que el compilador no optimice la comprobación de los valores que toman esos campos. Esto
permite que se puedan
//modificar vía hardware, y el software compruebe siempre el valor que tienen, sin considerar si acaba de asignarse a un valor
constante
struct UART_regs{
	/** \brief UART Data Register */
	volatile uint32_t Data; // 0x80000100
	/** \brief UART Status Register */
	volatile uint32_t Status; // 0x80000104
	/** \brief UART Control Register */
	volatile uint32_t Ctrl; // 0x80000108
	/** \brief UART Scaler Register */
	volatile uint32_t Scaler; // 0x8000010C
};

//Definicion del puntero a registro de la estructura y lo ubicamos en 0x80000100 direccion en la cual se encuentran los datos
struct	UART_regs * pLEON3_UART_REGS = 0x80000100;

//funcion que recibe un caracter y espera hasta que le permitan escribir o venza el tiempo
int8_t leon3_putchar(char c){
	///inicializa el tiemporizador a 0
	uint32_t write_timeout=0;

	//esperamos por permiso (pLEON3_UART_REGS->Status sea 0) mientras tengamos tiempo (write_timeout <
0xAAAAA)
	while(((LEON3_UART_TFF & pLEON3_UART_REGS->Status)==0x200) && (write_timeout < 0xAAAAA)){
		write_timeout++;		//incrementamos el contador
	}

	//Si no hemos llegado al limite de tiempo para poder escribir en data osea pLEON3_UART_REGS->Status era 0
	if(write_timeout < 0xAAAAA){
		pLEON3_UART_REGS->Data = (uint32_t)c;		//pasamos el valor a el registo en la posicion data
	}
	return (write_timeout == 0xAAAAA);	//salida en funcion de lo que haya pasado
}
```