My Project

Generated by Doxygen 1.8.10

Sat Sep 19 2015 00:46:04

Contents

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 ins_t Struct Reference

`#include <decoder.h>`

Data Fields

- char ** array

3.1.1 Field Documentation

3.1.1.1 char** array

The documentation for this struct was generated from the following file:

- decoder.h

3.2 instruction_t Struct Reference

`#include <decoder.h>`

Data Fields

- char mnemonic [10]
- char op1_type
- char op2_type
- char op3_type
- uint32_t op1_value
- uint32_t op2_value
- uint32_t op3_value

3.2.1 Field Documentation

3.2.1.1 char mnemonic[10]

3.2.1.2 char op1_type

3.2.1.3 uint32_t op1_value

3.2.1.4 char op2_type

3.2.1.5 uint32_t op2_value

3.2.1.6 char op3_type

3.2.1.7 uint32_t op3_value

The documentation for this struct was generated from the following file:

- decoder.h

Chapter 4

File Documentation

4.1 ALU.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "ALU.h"
#include <string.h>
```

Functions

- void ADDS (uint32_t *Rd, uint32_t Rm, uint32_t Rn, uint32_t *banderas)

  funcion que suma
- void ADD (uint32_t *Rd, uint32_t Rm, uint32_t Rn)

  funcion que suma pero no actualiza banderas
- void SUBS (uint32_t *Rd, uint32_t Rm, uint32_t Rn, uint32_t *banderas)

  funcion que resta y actualiza las banderas
- void SUB (uint32_t *Rd, uint32_t Rm, uint32_t Rn)

  funcion que resta
- void ANDS (uint32_t *Rd, uint32_t Rm, uint32_t Rn, uint32_t *banderas)

  funcion que producto logico a nivel de bit
- void ORRS (uint32_t *Rd, uint32_t Rm, uint32_t Rn, uint32_t *banderas)

  funcion OR
- void MOVS (uint32_t *Rm, uint32_t Rn, uint32_t *banderas)

  funcion que mueve datos de un registro a otro
- void MOV (uint32_t *Rm, uint32_t Rn)

  funcion que mueve datos de un registro a otro pero solo lo hace para actulizar banderas, no guarda este resultado
- void BICS (uint32_t *Rd, uint32_t Rm, uint32_t Rn, uint32_t *banderas)

  funcion para ejecutar una AND entre un registro y el negado del otro
- void EORS (uint32_t *Rd, uint32_t Rm, uint32_t Rn, uint32_t *banderas)

  funcion para ejecutar una exclusive OR bit a bit con los dos registros de entrada
- void MVNS (uint32_t *Rd, uint32_t Rm, uint32_t *banderas)

  funcion para ejecutar la operacion logica NOT a Rm
- void NOP ()

  funcion para no hacer nada
- void CMN (uint32_t Rm, uint32_t Rn, uint32_t *banderas)

  funcion sumar pero no guarda el resultado

- void CMP (uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion restar pero no guarda el valor solo actualiza banderas
- void TST (uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion realiza la operacion AND pero no guarda el valor solo actualiza banderas
- void ACTNZ (uint32_t ∗Rd, uint32_t ∗banderas)

  funcion que actualiza las banderas N y Z
- void ADCS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion que efectua la suma normal y suma el carry
- void SBCS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion que efectua la resta normal y resta el carry
- void RSBS (uint32_t ∗Rd, uint32_t Rm, uint32_t ∗banderas)

  funcion que efectua el complemento a dos de un registro
- void MULS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion que efectue el producto de dos registros sin signo

4.1.1 Function Documentation

4.1.1.1 void ACTNZ (uint32_t ∗ *Rd,* uint32_t ∗ *banderas*)

funcion que actualiza las banderas N y Z

Parameters

∗*Rd*	resultado
∗*banderas*	direccion de memoria de la bandera cero "N"
∗*op*	se identifica la operacion

Returns

no hay retorno

4.1.1.2 void ADCS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que efectua la suma normal y suma el carry

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro a sumar
Rn	registro a sumar
∗*banderas*	direccion de memoria de la bandera cero "N"
∗*op*	se identifica la operacion

Returns

no hay retorno

4.1.1.3 void ADD (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn*)

funcion que suma pero no actualiza banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado

Returns

no hay retorno

4.1.1.4 void ADDS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que suma

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado
∗*banderas*	

Returns

no hay retorno

4.1.1.5 void ANDS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que producto logico a nivel de bit

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.1.1.6 void BICS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion para ejecutar una AND entre un registro y el negado del otro

Parameters

Rn	Registro de entrada
Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado.
∗*banderas*	direccion de memoria de la bandera cero "N"
∗*op*	se identifica la operacion

Returns

no hay retorno

4.1.1.7 void CMN (uint32_t *Rm,* uint32_t *Rn,* uint32_t * *banderas*)

funcion sumar pero no guarda el resultado

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.1.1.8 void CMP (uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion restar pero no guarda el valor solo actualiza banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*banderas*	

Returns

no hay retorno

4.1.1.9 void EORS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion para ejecutar una exclusive OR bit a bit con los dos registros de entrada

Parameters

Rn	Registro de entrada
Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado.
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.1.1.10 void MOV (uint32_t ∗ *Rm,* uint32_t *Rn*)

funcion que mueve datos de un registro a otro pero solo lo hace para actulizar banderas, no guarda este resultado

Parameters

Rn	Registro que se va a mover
∗*Rm*	lugar donde se almacenaria el resultado.
∗*banderas*	

Returns

no hay retorno

4.1.1.11 void MOVS (uint32_t ∗ *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que mueve datos de un registro a otro

Parameters

Rn	Registro de entrada
∗*Rm*	lugar donde se almacenara el resultado.
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.1.1.12 void MULS (uint32_t ∗ *Rd*, uint32_t *Rm*, uint32_t *Rn*, uint32_t ∗ *banderas*)

funcion que efectue el producto de dos registros sin signo

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro a multiplicar
Rn	registro a multiplicar
∗*banderas*	direccion de memoria de las bandera

Returns

no hay retorno

4.1.1.13 void MVNS (uint32_t ∗ *Rd*, uint32_t *Rm*, uint32_t ∗ *banderas*)

funcion para ejecutar la operacion logica NOT a Rm

Parameters

Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado.
∗*op*	se identifica la operacion

Returns

no hay retorno

4.1.1.14 void NOP ( )

funcion para no hacer nada

Parameters

no	tiene parametros de entrada

Returns

no hay retorno

4.1.1.15 void ORRS (uint32_t ∗ *Rd*, uint32_t *Rm*, uint32_t *Rn*, uint32_t ∗ *banderas*)

funcion OR

Parameters

Rn	Registro de entrada
Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.1.1.16 void RSBS (uint32_t ∗ *Rd*, uint32_t *Rm*, uint32_t ∗ *banderas*)

funcion que efectua el complemento a dos de un registro

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro al que se le aplicara el complemento
∗*banderas*	direccion de memoria de las banderas

Returns

no hay retorno

4.1.1.17 void SBCS (uint32_t ∗ *Rd*, uint32_t *Rm*, uint32_t *Rn*, uint32_t ∗ *banderas*)

funcion que efectua la resta normal y resta el carry

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro a de entrada
Rn	registro a restar con el de entrada
∗*banderas*	direccion de memoria de las banderas

Returns

no hay retorno

4.1.1.18 void SUB (uint32_t ∗ *Rd*, uint32_t *Rm*, uint32_t *Rn*)

funcion que resta

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado

Returns

no hay retorno

4.1.1.19 void SUBS (uint32_t ∗ *Rd*, uint32_t *Rm*, uint32_t *Rn*, uint32_t ∗ *banderas*)

funcion que resta y actualiza las banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗Rd	lugar donde se almacenara el resultado
∗banderas	

Returns

no hay retorno

4.1.1.20 void TST (uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion realiza la operacion AND pero no guarda el valor solo actualiza banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗banderas	direccion de memoria de las banderas

Returns

no hay retorno

4.2 ALU.h File Reference

Functions

- void ADCS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion que efectua la suma normal y suma el carry
- void ADDS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion que suma
- void ADD (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn)
  funcion que suma pero no actualiza banderas
- void ANDS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion que producto logico a nivel de bit
- void BICS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion para ejecutar una AND entre un registro y el negado del otro
- void CMN (uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion sumar pero no guarda el resultado
- void CMP (uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion restar pero no guarda el valor solo actualiza banderas
- void EORS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion para ejecutar una exclusive OR bit a bit con los dos registros de entrada
- void MOV (uint32_t ∗Rm, uint32_t Rn)
  funcion que mueve datos de un registro a otro pero solo lo hace para actulizar banderas, no guarda este resultado
- void MOVS (uint32_t ∗Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion que mueve datos de un registro a otro
- void MULS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)
  funcion que efectue el producto de dos registros sin signo
- void MVNS (uint32_t ∗Rd, uint32_t Rm, uint32_t ∗banderas)
  funcion para ejecutar la operacion logica NOT a Rm

- void NOP ()

  funcion para no hacer nada
- void ORRS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion OR
- void RSBS (uint32_t ∗Rd, uint32_t Rm, uint32_t ∗banderas)

  funcion que efectua el complemento a dos de un registro
- void SBCS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion que efectua la resta normal y resta el carry
- void SUBS (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion que resta y actualiza las banderas
- void SUB (uint32_t ∗Rd, uint32_t Rm, uint32_t Rn)

  funcion que resta
- void TST (uint32_t Rm, uint32_t Rn, uint32_t ∗banderas)

  funcion realiza la operacion AND pero no guarda el valor solo actualiza banderas
- void ACTNZ (uint32_t ∗Rd, uint32_t ∗banderas)

  funcion que actualiza las banderas N y Z

4.2.1 Function Documentation

4.2.1.1 void ACTNZ (uint32_t ∗ *Rd,* uint32_t ∗ *banderas*)

funcion que actualiza las banderas N y Z

Parameters

∗*Rd*	resultado
∗*banderas*	direccion de memoria de la bandera cero "N"
∗*op*	se identifica la operacion

Returns

no hay retorno

4.2.1.2 void ADCS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que efectua la suma normal y suma el carry

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro a sumar
Rn	registro a sumar
∗*banderas*	direccion de memoria de la bandera cero "N"
∗*op*	se identifica la operacion

Returns

no hay retorno

4.2.1.3 void ADD (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn*)

funcion que suma pero no actualiza banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado

Returns

no hay retorno

4.2.1.4 void ADDS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que suma

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado
∗*banderas*	

Returns

no hay retorno

4.2.1.5 void ANDS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que producto logico a nivel de bit

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.2.1.6 void BICS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion para ejecutar una AND entre un registro y el negado del otro

Parameters

Rn	Registro de entrada
Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado.
∗*banderas*	direccion de memoria de la bandera cero "N"
∗*op*	se identifica la operacion

Returns

no hay retorno

4.2.1.7 void CMN (uint32_t *Rm,* uint32_t *Rn,* uint32_t * *banderas*)

funcion sumar pero no guarda el resultado

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.2.1.8 void CMP (uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion restar pero no guarda el valor solo actualiza banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*banderas*	

Returns

no hay retorno

4.2.1.9 void EORS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion para ejecutar una exclusive OR bit a bit con los dos registros de entrada

Parameters

Rn	Registro de entrada
Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado.
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.2.1.10 void MOV (uint32_t ∗ *Rm,* uint32_t *Rn*)

funcion que mueve datos de un registro a otro pero solo lo hace para actulizar banderas, no guarda este resultado

Parameters

Rn	Registro que se va a mover
∗*Rm*	lugar donde se almacenaria el resultado.
∗*banderas*	

Returns

no hay retorno

4.2.1.11 void MOVS (uint32_t ∗ *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que mueve datos de un registro a otro

Parameters

Rn	Registro de entrada
∗*Rm*	lugar donde se almacenara el resultado.
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.2.1.12 void MULS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que efectue el producto de dos registros sin signo

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro a multiplicar
Rn	registro a multiplicar
∗*banderas*	direccion de memoria de las bandera

Returns

no hay retorno

4.2.1.13 void MVNS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t ∗ *banderas*)

funcion para ejecutar la operacion logica NOT a Rm

Parameters

Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado.
∗*op*	se identifica la operacion

Returns

no hay retorno

4.2.1.14 void NOP ( )

funcion para no hacer nada

Parameters

no	tiene parametros de entrada

Returns

no hay retorno

4.2.1.15 void ORRS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion OR

Parameters

Rn	Registro de entrada
Rm	Registro de entrada
∗*Rd*	lugar donde se almacenara el resultado
∗*banderas*	direccion de memoria de la bandera cero "N"

Returns

no hay retorno

4.2.1.16 void RSBS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t ∗ *banderas*)

funcion que efectua el complemento a dos de un registro

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro al que se le aplicara el complemento
∗*banderas*	direccion de memoria de las banderas

Returns

no hay retorno

4.2.1.17 void SBCS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que efectua la resta normal y resta el carry

Parameters

∗*Rd*	registro donde se guarda el resultado
Rm	registro a de entrada
Rn	registro a restar con el de entrada
∗*banderas*	direccion de memoria de las banderas

Returns

no hay retorno

4.2.1.18 void SUB (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn*)

funcion que resta

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado

Returns

no hay retorno

4.2.1.19 void SUBS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion que resta y actualiza las banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*Rd*	lugar donde se almacenara el resultado
∗*banderas*	

Returns

no hay retorno

4.2.1.20 void TST (uint32_t *Rm,* uint32_t *Rn,* uint32_t ∗ *banderas*)

funcion realiza la operacion AND pero no guarda el valor solo actualiza banderas

Parameters

Rn	Registro de entrada
Rm	registro de entrada
∗*banderas*	direccion de memoria de las banderas

Returns

no hay retorno

4.3 branch.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "branch.h"
```

Functions

- void B (uint32_t ∗R, uint32_t b)

  funcion que aumenta o disminuye b posiciones el pc
- void BEQ (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en uno
- void BNE (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en cero
- void BCS (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno
- void BCC (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posisiones el pc si la bandera C esta en cero
- void BMI (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en uno
- void BPL (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en cero
- void BVS (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en uno
- void BVC (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en cero

- void BHI (uint32_t *R, uint32_t b, uint32_t *banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno y la Z en cero
- void BLS (uint32_t *R, uint32_t b, uint32_t *banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en cero o la Z en uno
- void BGE (uint32_t *R, uint32_t b, uint32_t *banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son iguales
- void BLT (uint32_t *R, uint32_t b, uint32_t *banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son diferentes
- void BGT (uint32_t *R, uint32_t b, uint32_t *banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero y N igual a V
- void BLE (uint32_t *R, uint32_t b, uint32_t *banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero 0 N diferente de V
- void BAL (uint32_t *R, uint32_t b, uint32_t *banderas)

  funcion que aumenta o disminuye b posiciones el pc
- void BL (uint32_t *LR, uint32_t *PC, uint32_t salto)

  funcion que llama una subrutina que esta en una direccion relativa al pc
- void BLX (uint32_t *LR, uint32_t *PC, uint32_t direccion)

  funcion que llama una subrutina que esta en la posicion direccion y RL avanza a la siguiente direccion
- void BX (uint32_t *PC, uint32_t direccion)

  funcion que salta a una direccion espesifica por un registro

4.3.1 Function Documentation

4.3.1.1 void B (uint32_t * *R*, uint32_t *b*)

funcion que aumenta o disminuye b posiciones el pc

Parameters

**R*	direccion del PC
b	valor que se va a cambiar el PC

Returns

no hay retorno

4.3.1.2 void BAL (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc

Parameters

**R*	direccion del PC
b	valor que va a cambiar el PC
**banderas*	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.3 void BCC (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posisiones el pc si la bandera C esta en cero

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.4 void BCS (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.5 void BEQ (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en uno

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.6 void BGE (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son iguales

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.7 void BGT (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero y N igual a V

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.8 void BHI (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno y la Z en cero

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.9 void BL (uint32_t ∗ *RL*, uint32_t ∗ *PC*, uint32_t *salto*)

funcion que llama una subrutina que esta en una direccion relativa al pc

Parameters

salto	el numero de direcciones que avanzara desde su posicion actual
∗PC	tomara la direccion de la subrutina ∗PC=Reg[13]
∗LR	tomara el valor de la siguiente direccion apartir de la posicion actual ∗LR=Reg[14]

Returns

no hay retorno

4.3.1.10 void BLE (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero 0 N diferente de V

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.11 void BLS (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en cero o la Z en uno

Parameters

∗*R*	direccion del PC
b	valor que va a cambiar el PC
∗*banderas*	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.12 void BLT (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son diferentes

Parameters

∗*R*	direccion del PC
b	valor que va a cambiar el PC
∗*banderas*	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.13 void BLX (uint32_t ∗ *RL*, uint32_t ∗ *PC*, uint32_t *direccion*)

funcion que llama una subrutina que esta en la posicion direccion y RL avanza a la siguiente direccion

Parameters

direccion	es un registro cuyo valor es la direccion de la subrutina
∗*PC*	tomara la direccion de la subrutina ∗PC=Reg[13]
∗*LR*	tomara el valor de la siguiente direccion apartir de la posicion actual ∗LR=Reg[14]

Returns

no hay retorno

4.3.1.14 void BMI (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en uno

Parameters

∗*R*	direccion del PC
b	valor que va a cambiar el PC
∗*banderas*	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.15 void BNE (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en cero

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.16 void BPL (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en cero

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.17 void BVC (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en cero

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.18 void BVS (uint32_t * *R*, uint32_t *b*, uint32_t * *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en uno

Parameters

$*R$	direccion del PC
b	valor que va a cambiar el PC
$*banderas$	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.3.1.19 void BX (uint32_t * *PC*, uint32_t *direccion*)

funcion que salta a una direccion espesifica por un registro

Parameters

direccion	es un registro cuyo valor es la direccion
∗PC	tomara la direccion de la subrutina ∗PC=Reg[13]

Returns

no hay retorno

4.4 branch.h File Reference

Functions

- void B (uint32_t ∗R, uint32_t b)

  funcion que aumenta o disminuye b posiciones el pc
- void BEQ (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en uno
- void BNE (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en cero
- void BCS (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno
- void BCC (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posisiones el pc si la bandera C esta en cero
- void BMI (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en uno
- void BPL (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en cero
- void BVS (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en uno
- void BVC (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en cero
- void BHI (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno y la Z en cero
- void BLS (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en cero o la Z en uno
- void BGE (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son iguales
- void BLT (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son diferentes
- void BGT (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero y N igual a V
- void BLE (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero 0 N diferente de V
- void BAL (uint32_t ∗R, uint32_t b, uint32_t ∗banderas)

  funcion que aumenta o disminuye b posiciones el pc
- void BL (uint32_t ∗RL, uint32_t ∗PC, uint32_t salto)

  funcion que llama una subrutina que esta en una direccion relativa al pc
- void BLX (uint32_t ∗RL, uint32_t ∗PC, uint32_t direccion)

  funcion que llama una subrutina que esta en la posicion direccion y RL avanza a la siguiente direccion
- void BX (uint32_t ∗PC, uint32_t direccion)

  funcion que salta a una direccion espesifica por un registro

4.4.1 Function Documentation

4.4.1.1 void B (uint32_t * *R*, uint32_t *b*)

funcion que aumenta o disminuye b posiciones el pc

Parameters

∗R	direccion del PC
b	valor que se va a cambiar el PC

Returns

no hay retorno

4.4.1.2 void BAL (uint32_t ∗ *R,* uint32_t *b,* uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.3 void BCC (uint32_t ∗ *R,* uint32_t *b,* uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posisiones el pc si la bandera C esta en cero

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.4 void BCS (uint32_t ∗ *R,* uint32_t *b,* uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.5 void BEQ (uint32_t ∗ *R,* uint32_t *b,* uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en uno

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.6 void BGE (uint32_t ∗ *R,* uint32_t *b,* uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son iguales

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.7 void BGT (uint32_t ∗ *R,* uint32_t *b,* uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero y N igual a V

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.8 void BHI (uint32_t ∗ *R,* uint32_t *b,* uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en uno y la Z en cero

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.9 void BL (uint32_t ∗ *RL,* uint32_t ∗ *PC,* uint32_t *salto*)

funcion que llama una subrutina que esta en una direccion relativa al pc

Parameters

salto	el numero de direcciones que avanzara desde su posicion actual
∗*PC*	tomara la direccion de la subrutina ∗PC=Reg[13]
∗*LR*	tomara el valor de la siguiente direccion apartir de la posicion actual ∗LR=Reg[14]

Returns

no hay retorno

4.4.1.10 void BLE (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z es cero 0 N diferente de V

Parameters

∗*R*	direccion del PC
b	valor que va a cambiar el PC
∗*banderas*	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.11 void BLS (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera C esta en cero o la Z en uno

Parameters

∗*R*	direccion del PC
b	valor que va a cambiar el PC
∗*banderas*	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.12 void BLT (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N y V son diferentes

Parameters

∗*R*	direccion del PC
b	valor que va a cambiar el PC
∗*banderas*	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.13 void BLX (uint32_t ∗ *RL*, uint32_t ∗ *PC*, uint32_t *direccion*)

funcion que llama una subrutina que esta en la posicion direccion y RL avanza a la siguiente direccion

Parameters

direccion	es un registro cuyo valor es la direccion de la subrutina
∗PC	tomara la direccion de la subrutina ∗PC=Reg[13]
∗LR	tomara el valor de la siguiente direccion apartir de la posicion actual ∗LR=Reg[14]

Returns

no hay retorno

4.4.1.14 void BMI (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en uno

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.15 void BNE (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera Z esta en cero

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.16 void BPL (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera N esta en cero

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.17 void BVC (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en cero

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.18 void BVS (uint32_t ∗ *R*, uint32_t *b*, uint32_t ∗ *banderas*)

funcion que aumenta o disminuye b posiciones el pc si la bandera V esta en uno

Parameters

∗R	direccion del PC
b	valor que va a cambiar el PC
∗banderas	Direccion de las banderas utilizada para evaluar la condicion

Returns

no hay retorno

4.4.1.19 void BX (uint32_t ∗ *PC*, uint32_t *direccion*)

funcion que salta a una direccion espesifica por un registro

Parameters

direccion	es un registro cuyo valor es la direccion
∗PC	tomara la direccion de la subrutina ∗PC=Reg[13]

Returns

no hay retorno

4.5 decoder.c File Reference

```
#include "decoder.h"
#include "ALU.h"
#include "registros.h"
#include "desplazamiento.h"
#include "curses.h"
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
#include "branch.h"
```

Functions

- void decodeInstruction (instruction_t instruction, uint32_t ∗Reg, uint32_t ∗banderas)
- instruction_t getInstruction (char ∗instStr)

Obtiene la instrucción separada por partes.

- int readFile (char *filename, ins_t *instructions)
- int countLines (FILE *fp)

4.5.1 Function Documentation

4.5.1.1 int countLines (FILE * *fp*)

4.5.1.2 void decodeInstruction (instruction_t *instruction,* uint32_t * *Reg,* uint32_t * *banderas*)

4.5.1.3 instruction_t getInstruction (char * *instStr*)

Obtiene la instrucción separada por partes.

Parameters

instrStr	cadena que contiene la instrucción.

Returns

instruction_t la instrucción separada por partes.

4.5.1.4 int readFile (char * *filename,* ins_t * *instructions*)

4.6 decoder.h File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
```

Data Structures

- struct ins_t
- struct instruction_t

Functions

- void decodeInstruction (instruction_t instruction, uint32_t *Reg, uint32_t *banderas)
- instruction_t getInstruction (char *instStr)

  Obtiene la instrucción separada por partes.
- int readFile (char *filename, ins_t *instructions)
- int countLines (FILE *fp)

4.6.1 Function Documentation

4.6.1.1 int countLines (FILE * *fp*)

4.6.1.2 void decodeInstruction (instruction_t *instruction,* uint32_t * *Reg,* uint32_t * *banderas*)

4.6.1.3 instruction_t getInstruction (char * *instStr*)

Obtiene la instrucción separada por partes.

Parameters

instrStr	cadena que contiene la instrucción.

Returns

instruction_t la instrucción separada por partes.

4.6.1.4 int readFile (char * *filename,* ins_t * *instructions*)

4.7 desplazamiento.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "desplazamiento.h"
#include "ALU.h"
```

Functions

- void LSLS (uint32_t *Rd, uint32_t Rm, uint32_t num, uint32_t *banderas)
  funcion para desplazar el dato Rm cierta cantidad de veces hacia la izquierda
- void LSRS (uint32_t *Rd, uint32_t Rm, uint32_t num, uint32_t *banderas)
  funcion para desplazar el dato Rm cierta cantidad de veces hacia la derecha
- void RORS (uint32_t *Rd, uint32_t Rm, uint32_t num, uint32_t *banderas)
  funcion para rotar hacia la derecha un dato
- void ASRS (uint32_t *Rd, uint32_t Rm, uint32_t num, uint32_t *banderas)
  funcion para hacer un desplazamiento aritmetico a la derecha
- void REV (uint32_t *Rd, uint32_t Rm)
  funcion para desplazar paquetes de 8 bits
- void REV16 (uint32_t *Rd, uint32_t Rm)
  funcion para desplazar paquetes de 16 bits
- void REVSH (uint32_t *Rd, uint32_t Rm)
  funcion para intercambiar los dos byts menos significativos.

4.7.1 Function Documentation

4.7.1.1 void ASRS (uint32_t * *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t * *banderas*)

funcion para hacer un desplazamiento aritmetico a la derecha

Parameters

*Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara
num	numero de bits que se desplazara

Returns

no hay retorno

4.7.1.2 void LSLS (uint32_t * *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t * *banderas*)

funcion para desplazar el dato Rm cierta cantidad de veces hacia la izquierda

Parameters

∗Rd	registro donde se guardara el dato desplazado
Rm	registro que se desplazara
num	numero de bits que se desplazara el dato Rm

Returns

no hay retorno

4.7.1.3 void LSRS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t ∗ *banderas*)

funcion para desplazar el dato Rm cierta cantidad de veces hacia la derecha

Parameters

∗Rd	registro donde se guardara el dato desplazado
Rm	registro que se desplazara
num	numero de bits que se desplazara el dato Rm

Returns

no hay retorno

4.7.1.4 void REV (uint32_t ∗ *Rd,* uint32_t *Rm*)

funcion para desplazar paquetes de 8 bits

Parameters

∗Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara

Returns

no hay retorno

4.7.1.5 void REV16 (uint32_t ∗ *Rd,* uint32_t *Rm*)

funcion para desplazar paquetes de 16 bits

Parameters

∗Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara

Returns

no hay retorno

4.7.1.6 void REVSH (uint32_t ∗ *Rd,* uint32_t *Rm*)

funcion para intercambiar los dos byts menos significativos.

Parameters

∗Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara

Returns

no hay retorno

4.7.1.7 void RORS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t ∗ *banderas*)

funcion para rotar hacia la derecha un dato

Parameters

∗Rd	variable donde se almacena la rotacion
Rm	registro que se rotara
num	numero de bits que se rotara el dato Rm

Returns

no hay retorno

4.8 desplazamiento.h File Reference

Functions

- void ASRS (uint32_t ∗Rd, uint32_t Rm, uint32_t num, uint32_t ∗banderas)
  funcion para hacer un desplazamiento aritmetico a la derecha
- void LSLS (uint32_t ∗Rd, uint32_t Rm, uint32_t num, uint32_t ∗banderas)
  funcion para desplazar el dato Rm cierta cantidad de veces hacia la izquierda
- void LSRS (uint32_t ∗Rd, uint32_t Rm, uint32_t num, uint32_t ∗banderas)
  funcion para desplazar el dato Rm cierta cantidad de veces hacia la derecha
- void REV (uint32_t ∗Rd, uint32_t Rm)
  funcion para desplazar paquetes de 8 bits
- void REV16 (uint32_t ∗Rd, uint32_t Rm)
  funcion para desplazar paquetes de 16 bits
- void REVSH (uint32_t ∗Rd, uint32_t Rm)
  funcion para intercambiar los dos byts menos significativos.
- void RORS (uint32_t ∗Rd, uint32_t Rm, uint32_t num, uint32_t ∗banderas)
  funcion para rotar hacia la derecha un dato

4.8.1 Function Documentation

4.8.1.1 void ASRS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t ∗ *banderas*)

funcion para hacer un desplazamiento aritmetico a la derecha

Parameters

∗Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara
num	numero de bits que se desplazara

Returns

no hay retorno

4.8.1.2 void LSLS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t ∗ *banderas*)

funcion para desplazar el dato Rm cierta cantidad de veces hacia la izquierda

Parameters

∗Rd	registro donde se guardara el dato desplazado
Rm	registro que se desplazara
num	numero de bits que se desplazara el dato Rm

Returns

no hay retorno

4.8.1.3 void LSRS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t ∗ *banderas*)

funcion para desplazar el dato Rm cierta cantidad de veces hacia la derecha

Parameters

∗Rd	registro donde se guardara el dato desplazado
Rm	registro que se desplazara
num	numero de bits que se desplazara el dato Rm

Returns

no hay retorno

4.8.1.4 void REV (uint32_t ∗ *Rd,* uint32_t *Rm*)

funcion para desplazar paquetes de 8 bits

Parameters

∗Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara

Returns

no hay retorno

4.8.1.5 void REV16 (uint32_t ∗ *Rd,* uint32_t *Rm*)

funcion para desplazar paquetes de 16 bits

Parameters

∗Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara

Returns

no hay retorno

4.8.1.6 void REVSH (uint32_t ∗ *Rd,* uint32_t *Rm*)

funcion para intercambiar los dos byts menos significativos.

Parameters

∗Rd	variable donde se almacena el desplazamiento
Rm	registro que se desplazara

Returns

no hay retorno

4.8.1.7 void RORS (uint32_t ∗ *Rd,* uint32_t *Rm,* uint32_t *num,* uint32_t ∗ *banderas*)

funcion para rotar hacia la derecha un dato

Parameters

∗Rd	variable donde se almacena la rotacion
Rm	registro que se rotara
num	numero de bits que se rotara el dato Rm

Returns

no hay retorno

4.9 main.c File Reference

```
#include "decoder.h"
#include "ALU.h"
#include "registros.h"
#include "desplazamiento.h"
#include "curses.h"
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include <stdint.h>
```

Functions

- int main (void)

4.9.1 Function Documentation

4.9.1.1 int main (void)

4.10 registros.c File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include "registros.h"
#include <curses.h>
```

Functions

- void mostrar_registros (uint32_t *registro)

  Funcion que toma los valores de los registros y los imprime en pantalla.
- void mostrar_banderas (uint32_t banderas[4])

  Funcion que toma los valores de las banderas y los imprime en pantalla.
- void mostrar_operacion (char *op)

  Funcion que muestra la operacion que se realizo.

Variables

- unsigned int i

4.10.1 Function Documentation

4.10.1.1 void mostrar_banderas (uint32_t *banderas[4]*)

Funcion que toma los valores de las banderas y los imprime en pantalla.

Parameters

banderas[4]	Arreglo que contiene cada 1 de las banderas

Returns

No tiene retorno

4.10.1.2 void mostrar_operacion (char * *op*)

Funcion que muestra la operacion que se realizo.

Parameters

op[5]	variable tipo caracter donde se gusrda el nombre de la operacion realizada

Returns

No tiene retorno

4.10.1.3 void mostrar_registros (uint32_t * *registro*)

Funcion que toma los valores de los registros y los imprime en pantalla.

Parameters

registro[13]	Arreglo que contiene cada 1 de los registros

Returns

No tiene retorno

4.10.2 Variable Documentation

4.10.2.1 unsigned int i

4.11 registros.h File Reference

Functions

- void mostrar_registros (uint32_t *registro)

  Funcion que toma los valores de los registros y los imprime en pantalla.
- void mostrar_banderas (uint32_t banderas[4])

  Funcion que toma los valores de las banderas y los imprime en pantalla.
- void mostrar_operacion (char *op)

  Funcion que muestra la operacion que se realizo.

4.11.1 Function Documentation

4.11.1.1 void mostrar_banderas (uint32_t *banderas[4]*)

Funcion que toma los valores de las banderas y los imprime en pantalla.

Parameters

banderas[4]	Arreglo que contiene cada 1 de las banderas

Returns

No tiene retorno

4.11.1.2 void mostrar_operacion (char * *op*)

Funcion que muestra la operacion que se realizo.

Parameters

op[5]	variable tipo caracter donde se gusrda el nombre de la operacion realizada

Returns

No tiene retorno

4.11.1.3 void mostrar_registros (uint32_t * *registro*)

Funcion que toma los valores de los registros y los imprime en pantalla.

Parameters

registro[13]	Arreglo que contiene cada 1 de los registros

Returns

No tiene retorno

4.12 test.c File Reference