

Emulador ARM Cortex M0

Generated by Doxygen 1.8.10

Sat Sep 19 2015 15:30:42

Contents

1	Practica #1 Emulador ARM Cortex -M0	1
2	Data Structure Index	3
2.1	Data Structures	3
3	File Index	5
3.1	File List	5
4	Data Structure Documentation	7
4.1	u32tobyte_t Union Reference	7
4.1.1	Field Documentation	7
4.1.1.1	"@1	7
4.1.1.2	byte0	7
4.1.1.3	byte1	7
4.1.1.4	byte2	7
4.1.1.5	byte3	7
4.1.1.6	data	7
5	File Documentation	9
5.1	Banderas.c File Reference	9
5.1.1	Detailed Description	9
5.1.2	Function Documentation	9
5.1.2.1	BANDERAS(uint32_t Rd, uint32_t Rn, uint32_t Rm, int *Banderas)	9
5.1.2.2	BANDERAS_DES(uint32_t Rd, int *Banderas)	9
5.2	Banderas.h File Reference	10
5.2.1	Detailed Description	10
5.2.2	Function Documentation	10
5.2.2.1	BANDERAS(uint32_t Rd, uint32_t Rn, uint32_t Rm, int *Banderas)	10
5.2.2.2	BANDERAS_DES(uint32_t Rd, int *Banderas)	10
5.3	Instrucciones.c File Reference	11
5.3.1	Detailed Description	11
5.3.2	Function Documentation	12
5.3.2.1	ADCS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	12

5.3.2.2	ADDS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	13
5.3.2.3	ANDS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	13
5.3.2.4	CMNS(uint32_t Rn, uint32_t Rm, int *flags)	13
5.3.2.5	CMPS(uint32_t Rn, uint32_t Rm, int *flags)	13
5.3.2.6	EORS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	14
5.3.2.7	MOVS(uint32_t *Rd, uint32_t Rn)	14
5.3.2.8	MULS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	14
5.3.2.9	NOP()	15
5.3.2.10	ORRS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	15
5.3.2.11	SBCS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	15
5.3.2.12	SUBS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	15
5.3.2.13	TST(uint32_t Rn, uint32_t Rm, int *flags)	15
5.4	Instrucciones.h File Reference	16
5.4.1	Function Documentation	16
5.4.1.1	ADCS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	16
5.4.1.2	ADDS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	17
5.4.1.3	ANDS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	17
5.4.1.4	CMNS(uint32_t Rn, uint32_t Rm, int *flags)	17
5.4.1.5	CMPS(uint32_t Rn, uint32_t Rm, int *flags)	17
5.4.1.6	EORS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	18
5.4.1.7	MOVS(uint32_t *Rd, uint32_t Rn)	18
5.4.1.8	MULS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	18
5.4.1.9	NOP()	19
5.4.1.10	ORRS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	19
5.4.1.11	SBCS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	19
5.4.1.12	SUBS(uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)	19
5.4.1.13	TST(uint32_t Rn, uint32_t Rm, int *flags)	19
5.5	Instrucciones_desplazamiento.c File Reference	20
5.5.1	Detailed Description	20
5.5.2	Function Documentation	21
5.5.2.1	ASR(uint32_t *Rdn, uint32_t Rm, int *flags)	21
5.5.2.2	BIC(uint32_t *Rdn, uint32_t Rm, int *flags)	22
5.5.2.3	LSLS(uint32_t *Rdn, uint32_t Rn, uint32_t Rm, int *flags)	22
5.5.2.4	LSRS(uint32_t *Rdn, uint32_t Rn, uint32_t Rm, int *flags)	22
5.5.2.5	MVN(uint32_t *Rdn, uint32_t Rm, int *flags)	22
5.5.2.6	REV(uint32_t *Rdn, uint32_t Rm, int *flags)	23
5.5.2.7	REV16(uint32_t *Rdn, uint32_t Rm, int *flags)	23
5.5.2.8	REVSH(uint32_t *Rdn, uint32_t Rm, int *flags)	23
5.5.2.9	ROR(uint32_t *Rdn, uint32_t Rm, int *flags)	23
5.5.2.10	RSB(uint32_t *Rdn, uint32_t Rm, int *flags)	24

5.6	Instrucciones_desplazamiento.h File Reference	24
5.6.1	Detailed Description	25
5.6.2	Function Documentation	25
5.6.2.1	ASR(uint32_t *Rdn, uint32_t Rm, int *flags)	25
5.6.2.2	BIC(uint32_t *Rdn, uint32_t Rm, int *flags)	25
5.6.2.3	LSLS(uint32_t *Rdn, uint32_t Rn, uint32_t Rm, int *flags)	25
5.6.2.4	LSRS(uint32_t *Rdn, uint32_t Rn, uint32_t Rm, int *flags)	25
5.6.2.5	MVN(uint32_t *Rdn, uint32_t Rm, int *flags)	26
5.6.2.6	REV(uint32_t *Rdn, uint32_t Rm, int *flags)	26
5.6.2.7	REV16(uint32_t *Rdn, uint32_t Rm, int *flags)	26
5.6.2.8	REVSH(uint32_t *Rdn, uint32_t Rm, int *flags)	26
5.6.2.9	ROR(uint32_t *Rdn, uint32_t Rm, int *flags)	27
5.6.2.10	RSB(uint32_t *Rdn, uint32_t Rm, int *flags)	27
5.7	Instrucciones_salto.c File Reference	27
5.7.1	Detailed Description	28
5.7.2	Function Documentation	28
5.7.2.1	B(uint32_t *registro, int Salto)	28
5.7.2.2	BCC(uint32_t *registro, int Salto, int *Banderas)	28
5.7.2.3	BCS(uint32_t *registro, int Salto, int *Banderas)	29
5.7.2.4	BEQ(uint32_t *registro, int Salto, int *Banderas)	29
5.7.2.5	BGE(uint32_t *registro, int Salto, int *Banderas)	29
5.7.2.6	BGT(uint32_t *registro, int Salto, int *Banderas)	29
5.7.2.7	BHI(uint32_t *registro, int Salto, int *Banderas)	30
5.7.2.8	BL(uint32_t *registro, int Salto)	30
5.7.2.9	BLE(uint32_t *registro, int Salto, int *Banderas)	30
5.7.2.10	BLS(uint32_t *registro, int Salto, int *Banderas)	30
5.7.2.11	BLT(uint32_t *registro, int Salto, int *Banderas)	30
5.7.2.12	BLX(uint32_t *registro, uint32_t Registro)	31
5.7.2.13	BMI(uint32_t *registro, int Salto, int *Banderas)	31
5.7.2.14	BNE(uint32_t *registro, int Salto, int *Banderas)	31
5.7.2.15	BPL(uint32_t *registro, int Salto, int *Banderas)	31
5.7.2.16	BVC(uint32_t *registro, int Salto, int *Banderas)	31
5.7.2.17	BVS(uint32_t *registro, int Salto, int *Banderas)	32
5.7.2.18	BX(uint32_t *registro, uint32_t Salto)	32
5.8	Instrucciones_salto.h File Reference	32
5.8.1	Function Documentation	33
5.8.1.1	B(uint32_t *registro, int Salto)	33
5.8.1.2	BCC(uint32_t *registro, int Salto, int *Banderas)	33
5.8.1.3	BCS(uint32_t *registro, int Salto, int *Banderas)	33
5.8.1.4	BEQ(uint32_t *registro, int Salto, int *Banderas)	34

5.8.1.5	BGE(uint32_t *registro, int Salto, int *Banderas)	34
5.8.1.6	BGT(uint32_t *registro, int Salto, int *Banderas)	34
5.8.1.7	BHI(uint32_t *registro, int Salto, int *Banderas)	34
5.8.1.8	BL(uint32_t *registro, int salto)	35
5.8.1.9	BLE(uint32_t *registro, int Salto, int *Banderas)	35
5.8.1.10	BLS(uint32_t *registro, int Salto, int *Banderas)	35
5.8.1.11	BLT(uint32_t *registro, int Salto, int *Banderas)	35
5.8.1.12	BLX(uint32_t *registro, uint32_t Registro)	36
5.8.1.13	BMI(uint32_t *registro, int Salto, int *Banderas)	36
5.8.1.14	BNE(uint32_t *registro, int Salto, int *Banderas)	37
5.8.1.15	BPL(uint32_t *registro, int Salto, int *Banderas)	37
5.8.1.16	BVC(uint32_t *registro, int Salto, int *Banderas)	37
5.8.1.17	BVS(uint32_t *registro, int Salto, int *Banderas)	37
5.8.1.18	BX(uint32_t *registro, uint32_t Salto)	38
5.9	main.c File Reference	38
5.9.1	Detailed Description	38
5.9.2	Function Documentation	38
5.9.2.1	main(void)	38
5.10	Registros.c File Reference	38
5.10.1	Detailed Description	39
5.10.2	Function Documentation	39
5.10.2.1	MostrarRegistro(uint32_t *registro)	39
5.11	Registros.h File Reference	39
5.11.1	Detailed Description	39
5.11.2	Function Documentation	39
5.11.2.1	MostrarRegistro(uint32_t *registro)	39

Chapter 1

Practica #1 Emulador ARM Cortex -M0

Documentacion pertinente del software realizado para realizar el Emulador del procesador ARM Cortex -M0

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

u32tobyte_t	7
---------------------------------------	---

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Banderas.c	Contiene las funciones para la correcta activacion de las banderas N,Z,C,V	9
Banderas.h	Archivo que contiene las definiciones de las funciones para la activacion de las bandera	10
Instrucciones.c	Libreria encargada de realizar las funciones aritmeticas	11
Instrucciones.h	16
Instrucciones_desplazamiento.c	Libreria encargada de realizar las funciones de desplazamiento	20
Instrucciones_desplazamiento.h	Archivo que contiene las definiciones de las funciones de desplazamiento	24
Instrucciones_salto.c	Libreria encargada de realizar las funciones saltos	27
Instrucciones_salto.h	32
main.c	Codigo en el cual se ejecutaran las funciones, utilizando todas las librerias creadas para cada una de ellas ademas de la libreria curses.h para la interfaz del emulador, y de la libreria decoder.h para obtener las instrucciones del documento de texto	38
Registros.c	Contiene la funcion para poder imprimir en pantalla los registros	38
Registros.h	Archivo que contiene la definicion de la funcion para imprimir en pantalla los registros	39

Chapter 4

Data Structure Documentation

4.1 u32tobyte_t Union Reference

Data Fields

- uint32_t [data](#)
 - struct {
 - uint8_t [byte0](#)
 - uint8_t [byte1](#)
 - uint8_t [byte2](#)
 - uint8_t [byte3](#)
- };

4.1.1 Field Documentation

4.1.1.1 struct { ... }

4.1.1.2 uint8_t byte0

4.1.1.3 uint8_t byte1

4.1.1.4 uint8_t byte2

4.1.1.5 uint8_t byte3

4.1.1.6 uint32_t data

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

- [Instrucciones_desplazamiento.c](#)

Chapter 5

File Documentation

5.1 Banderas.c File Reference

Contiene las funciones para la correcta activacion de las banderas N,Z,C,V.

```
#include "Banderas.h"
```

Functions

- void **BANDERAS** (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, int *Banderas)
Funcion donde se llevara a cabo la activacion de las banderas.
- void **BANDERAS_DES** (uint32_t *Rd*, int *Banderas)
Funcion donde se llevara a cabo la activacion de las banderas.

5.1.1 Detailed Description

Contiene las funciones para la correcta activacion de las banderas N,Z,C,V.

5.1.2 Function Documentation

5.1.2.1 void **BANDERAS** (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *Banderas*)

Funcion donde se llevara a cabo la activacion de las banderas.

Parameters

<i>Rd</i>	Resultado
<i>Rn</i>	Primer dato de la operacion realizada
<i>Rm</i>	Segundo dato de la operacion realizada
<i>Banderas</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.1.2.2 void **BANDERAS_DES** (uint32_t *Rd*, int * *Banderas*)

Funcion donde se llevara a cabo la activacion de las banderas.

Parameters

<i>Rd</i>	Resultado
<i>Banderas</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.2 Banderas.h File Reference

Archivo que contiene las definiciones de las funciones para la activacion de las bandera.

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

Functions

- void [BANDERAS](#) (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, int **Banderas*)
Funcion donde se llevara a cabo la activacion de las banderas.
- void [BANDERAS_DES](#) (uint32_t *Rd*, int **Banderas*)
Funcion donde se llevara a cabo la activacion de las banderas.

5.2.1 Detailed Description

Archivo que contiene las definiciones de las funciones para la activacion de las bandera.

5.2.2 Function Documentation

5.2.2.1 void BANDERAS (uint32_t *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *Banderas*)

Funcion donde se llevara a cabo la activacion de las banderas.

Parameters

<i>Rd</i>	Resultado
<i>Rn</i>	Primer dato de la operacion realizada
<i>Rm</i>	Segundo dato de la operacion realizada
<i>Banderas</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.2.2.2 void BANDERAS_DES (uint32_t *Rd*, int * *Banderas*)

Funcion donde se llevara a cabo la activacion de las banderas.

Parameters

<i>Rd</i>	Resultado
<i>Banderas</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3 Instrucciones.c File Reference

libreria encargada de realizar las funciones aritmeticas

```
#include "Instrucciones.h"
#include "Banderas.h"
```

Functions

- void **ADDS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion suma.
- void **ANDS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica AND, bit a bit.
- void **EORS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica X-OR, bit a bit.
- void **MOVS** (uint32_t *Rd, uint32_t Rn)
Funcion realiza una copia de un registro en otro.
- void **ORRS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica OR, bit a bit.
- void **SUBS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion resta.
- void **CMNS** (uint32_t Rn, uint32_t Rm, int *flags)
Funcion suma,pero solo modifica las banderas.
- void **CMPS** (uint32_t Rn, uint32_t Rm, int *flags)
Funcion resta, pero solo modifica las banderas.
- void **MULS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion multiplica, guarda 32 bits menos significativos.
- void **TST** (uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica AND, bit a bit pero solo modifica las banderas.
- void **NOP** ()
Funcion que no realiza ninguna operacion.
- void **ADCS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion suma con acarreo.
- void **SBCS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion resta con acarreo.

5.3.1 Detailed Description

libreria encargada de realizar las funciones aritmeticas

5.3.2 Function Documentation

5.3.2.1 void ADCS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion suma con acarreo.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.2 void ADDS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion suma.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.3 void ANDS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica AND, bit a bit.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Primer dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.4 void CMNS (uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion suma,pero solo modifica las banderas.

Parameters

<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.5 void CMPS (uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion resta, pero solo modifica las banderas.

Parameters

<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.6 void EORS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica X-OR, bit a bit.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer registro a relizar operacion
<i>Rm</i>	Primer registro a relizar operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.7 void MOVS (uint32_t * *Rd*, uint32_t *Rn*)

Funcion realiza una copia de un registro en otro.

Parameters

<i>*Rd</i>	Dato donde se realizara la copia
<i>Rn</i>	Dato al que se le realizara una copia

Returns

No hay retorno de la funcion

5.3.2.8 void MULS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion multiplica, guarda 32 bits menos significativos.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.9 void NOP ()

Funcion que no realiza ninguna operacion.

Returns

No hay retorno de la funcion

5.3.2.10 void ORRS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica OR, bit a bit.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer registro a relizar operacion
<i>Rm</i>	Primer registro a relizar operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.11 void SBCS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion resta con acarreo.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.12 void SUBS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion resta.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.3.2.13 void TST (uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica AND, bit a bit pero solo modifica las banderas.

Parameters

<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4 Instrucciones.h File Reference

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

Functions

- void **ADDS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion suma.
- void **ANDS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica AND, bit a bit.
- void **EORS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica X-OR, bit a bit.
- void **MOVS** (uint32_t *Rd, uint32_t Rn)
Funcion realiza una copia de un registro en otro.
- void **ORRS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica OR, bit a bit.
- void **SUBS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion resta.
- void **CMNS** (uint32_t Rn, uint32_t Rm, int *flags)
Funcion suma,pero solo modifica las banderas.
- void **CMPS** (uint32_t Rn, uint32_t Rm, int *flags)
Funcion resta, pero solo modifica las banderas.
- void **MULS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion multiplica, guarda 32 bits menos significativos.
- void **ADCS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion suma con acarreo.
- void **SBCS** (uint32_t *Rd, uint32_t Rn, uint32_t Rm, int *flags)
Funcion resta con acarreo.
- void **TST** (uint32_t Rn, uint32_t Rm, int *flags)
Funcion logica AND, bit a bit pero solo modifica las banderas.
- void **NOP** ()
Funcion que no realiza ninguna operacion.

5.4.1 Function Documentation

5.4.1.1 void ADCS (uint32_t * Rd, uint32_t Rn, uint32_t Rm, int * flags)

Funcion suma con acarreo.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.2 void ADDS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion suma.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.3 void ANDS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica AND, bit a bit.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Primer dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.4 void CMNS (uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion suma,pero solo modifica las banderas.

Parameters

<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.5 void CMPS (uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion resta, pero solo modifica las banderas.

Parameters

<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.6 void EORS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica X-OR, bit a bit.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer registro a relizar operacion
<i>Rm</i>	Primer registro a relizar operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.7 void MOVS (uint32_t * *Rd*, uint32_t *Rn*)

Funcion realiza una copia de un registro en otro.

Parameters

<i>*Rd</i>	Dato donde se realizara la copia
<i>Rn</i>	Dato al que se le realizara una copia

Returns

No hay retorno de la funcion

5.4.1.8 void MULS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion multiplica, guarda 32 bits menos significativos.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.9 void NOP ()

Funcion que no realiza ninguna operacion.

Returns

No hay retorno de la funcion

5.4.1.10 void ORRS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica OR, bit a bit.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer registro a relizar operacion
<i>Rm</i>	Primer registro a relizar operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.11 void SBCS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion resta con acarreo.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.12 void SUBS (uint32_t * *Rd*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion resta.

Parameters

<i>*Rd</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.4.1.13 void TST (uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Funcion logica AND, bit a bit pero solo modifica las banderas.

Parameters

<i>Rn</i>	Primer dato a realizar la operacion
<i>Rm</i>	Segundo dato a realizar la operacion
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5 Instrucciones_desplazamiento.c File Reference

libreria encargada de realizar las funciones de desplazamiento

```
#include "Instrucciones_desplazamiento.h"
#include "Banderas.h"
```

Data Structures

- union [u32tobyte_t](#)

Functions

- void [LSLS](#) (uint32_t *Rdn, uint32_t Rn, uint32_t Rm, int *flags)
Desplazamiento logico a la izquierda.
- void [LSRS](#) (uint32_t *Rdn, uint32_t Rn, uint32_t Rm, int *flags)
Desplazamiento logico a la izquierda.
- void [ROR](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Rotacion a la derecha.
- void [ASR](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Desplazamiento aritmetico a la derecha.
- void [BIC](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Operacion logica AND entre un dato y el complemento del otro dato.
- void [MVN](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Guardar el complemento de un dato.
- void [RSB](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Complemento a dos de un dato.
- void [REV](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Cambiar el orden de los bytes.
- void [REV16](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Cambiar el orden de los bytes en cada halfword de 16bits.
- void [REVSH](#) (uint32_t *Rdn, uint32_t Rm, int *flags)
Cambiar el orden de los bytes del halfword bajo.

5.5.1 Detailed Description

libreria encargada de realizar las funciones de desplazamiento

5.5.2 Function Documentation

5.5.2.1 void ASR (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Desplazamiento aritmetico a la derecha.

Parameters

<i>*Rdn</i>	Dato que se desplazara
<i>Rm</i>	Numero de desplazamientos
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.2 void BIC (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Operacion logica AND entre un dato y el complemento del otro dato.

Parameters

<i>*Rdn</i>	Primer operador de la operacion AND
<i>Rm</i>	Segundo operador de la operacion AND, al cual se le hara el complemento
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.3 void LSLS (uint32_t * *Rdn*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Desplazamiento logico a la izquierda.

Parameters

<i>*Rdn</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Dato a desplazar
<i>Rm</i>	Numero de desplazamientos
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.4 void LSRS (uint32_t * *Rdn*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Desplazamiento logico a la izquierda.

Parameters

<i>*Rdn</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Dato a desplazar
<i>Rm</i>	Numero de desplazamientos
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.5 void MVN (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Guardar el complemento de un dato.

Parameters

<i>Rdn</i>	Donde se almacenara el dato
<i>Rm</i>	Dato al que se le hara el complemento
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.6 void REV (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Cambiar el orden de los bytes.

Parameters

<i>Rdn</i>	Donde se almacenara el resultado
<i>Rm</i>	Dato que se le cambiara el orden de los bytes
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.7 void REV16 (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Cambiar el orden de los bytes en cada halfword de 16bits.

Parameters

<i>Rdn</i>	Donde se almacenara el resultado
<i>Rm</i>	Dato que se le cambiara el orden de los bytes
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.8 void REVSH (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Cambiar el orden de los bytes del halfword bajo.

Parameters

* <i>Rdn</i>	Donde se almacenara el resultado
<i>Rm</i>	Dato que se le cambiara el orden de los bytes
* <i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.5.2.9 void ROR (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Rotacion a la derecha.

Parameters

<i>*Rdn</i>	Dato que se rotara
<i>Rm</i>	Orden de la rotacion

Returns

No hay retorno de la funcion

5.5.2.10 void RSB (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Complemento a dos de un dato.

Parameters

<i>Rdn</i>	Donde se almacenara el dato
<i>Rm</i>	Dato al que se le hara el complemento a dos
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6 Instrucciones_desplazamiento.h File Reference

Archivo que contiene las definiciones de las funciones de desplazamiento.

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

Functions

- void **LSLS** (uint32_t **Rdn*, uint32_t *Rn*, uint32_t *Rm*, int **flags*)
Desplazamiento logico a la izquierda.
- void **LSRS** (uint32_t **Rdn*, uint32_t *Rn*, uint32_t *Rm*, int **flags*)
Desplazamiento logico a la izquierda.
- void **ROR** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Rotacion a la derecha.
- void **ASR** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Desplazamiento aritmetico a la derecha.
- void **BIC** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Operacion logica AND entre un dato y el complemento del otro dato.
- void **MVN** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Guardar el complemento de un dato.
- void **RSB** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Complemento a dos de un dato.
- void **REV** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Cambiar el orden de los bytes.
- void **REV16** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Cambiar el orden de los bytes en cada halfword de 16bits.
- void **REVSH** (uint32_t **Rdn*, uint32_t *Rm*, int **flags*)
Cambiar el orden de los bytes del halfword bajo.

5.6.1 Detailed Description

Archivo que contiene las definiciones de las funciones de desplazamiento.

5.6.2 Function Documentation

5.6.2.1 void ASR (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Desplazamiento aritmetico a la derecha.

Parameters

* <i>Rdn</i>	Dato que se desplazara
<i>Rm</i>	Numero de desplazamientos
* <i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.2 void BIC (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Operacion logica AND entre un dato y el complemento del otro dato.

Parameters

* <i>Rdn</i>	Primer operador de la operacion AND
<i>Rm</i>	Segundo operador de la operacion AND, al cual se le hara el complemento
* <i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.3 void LSLS (uint32_t * *Rdn*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Desplazamiento logico a la izquierda.

Parameters

* <i>Rdn</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Dato a desplazar
<i>Rm</i>	Numero de desplazamientos
* <i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.4 void LSRS (uint32_t * *Rdn*, uint32_t *Rn*, uint32_t *Rm*, int * *flags*)

Desplazamiento logico a la izquierda.

Parameters

<i>*Rdn</i>	Dato donde se almacenara el resultado
<i>Rn</i>	Dato a desplazar
<i>Rm</i>	Numero de desplazamientos
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.5 void MVN (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Guardar el complemento de un dato.

Parameters

<i>Rdn</i>	Donde se almacenara el dato
<i>Rm</i>	Dato al que se le hara el complemento
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.6 void REV (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Cambiar el orden de los bytes.

Parameters

<i>Rdn</i>	Donde se almacenara el resultado
<i>Rm</i>	Dato que se le cambiara el orden de los bytes
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.7 void REV16 (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Cambiar el orden de los bytes en cada halfword de 16bits.

Parameters

<i>Rdn</i>	Donde se almacenara el resultado
<i>Rm</i>	Dato que se le cambiara el orden de los bytes
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.8 void REVSH (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Cambiar el orden de los bytes del halfword bajo.

Parameters

<i>*Rdn</i>	Donde se almacenara el resultado
<i>Rm</i>	Dato que se le cambiara el orden de los bytes
<i>*flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.6.2.9 void ROR (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Rotacion a la derecha.

Parameters

<i>*Rdn</i>	Dato que se rotara
<i>Rm</i>	Orden de la rotacion

Returns

No hay retorno de la funcion

5.6.2.10 void RSB (uint32_t * *Rdn*, uint32_t *Rm*, int * *flags*)

Complemento a dos de un dato.

Parameters

<i>Rdn</i>	Donde se almacenara el dato
<i>Rm</i>	Dato al que se le hara el complemento a dos
<i>flags</i>	Arreglo donde se almacenaran las banderas

Returns

No hay retorno de la funcion

5.7 Instrucciones_salto.c File Reference

libreria encargada de realizar las funciones saltos

```
#include "Instrucciones_salto.h"
```

Functions

- void **BEQ** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si los datos son iguales.
- void **BNE** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si los datos no son iguales.
- void **BCS** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es mayor o igual a otro (sin signo)
- void **BCC** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es menor a otro (sin signo)

- void **BMI** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si el dato es negativo.
- void **BPL** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si el dato es positivo.
- void **BVS** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si hay sobreflujo.
- void **BVC** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si no hay sobreflujo.
- void **BHI** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es mayor que el otro (sin signo)
- void **BLS** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es mayor o igual a otro (sin signo)
- void **BGE** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es mayor o igual a otro (con signo)
- void **BLT** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es menor a otro (con signo)
- void **BGT** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es mayor a otro (con signo)
- void **BLE** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es menor o igual a otro (con signo)
- void **B** (uint32_t *registro, int Salto)
Funcion de salto, sin condicion.
- void **BL** (uint32_t *registro, int salto)
- void **BX** (uint32_t *registro, uint32_t Salto)
- void **BLX** (uint32_t *registro, uint32_t Registro)

5.7.1 Detailed Description

libreria encargada de realizar las funciones saltos

5.7.2 Function Documentation

5.7.2.1 void B (uint32_t * registro, int Salto)

Funcion de salto, sin condicion.

Parameters

<i>PC</i>	Intrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar

Returns

No retorna

5.7.2.2 void BCC (uint32_t * registro, int Salto, int * Banderas)

Funcion de salto, si un dato es menor a otro (sin signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.3 void BCS (uint32_t * registro, int Salto, int * Banderas)

Funcion de salto, si un dato es mayor o igual a otro (sin signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.4 void BEQ (uint32_t * registro, int Salto, int * Banderas)

Funcion de salto, si los datos son iguales.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.5 void BGE (uint32_t * registro, int Salto, int * Banderas)

Funcion de salto, si un dato es mayor o igual a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.6 void BGT (uint32_t * registro, int Salto, int * Banderas)

Funcion de salto, si un dato es mayor a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.7 void BHI (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es mayor que el otro (sin signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.8 void BL (uint32_t * *registro*, int *salto*)

5.7.2.9 void BLE (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es menor o igual a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.10 void BLS (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es mayor o igual a otro (sin signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.11 void BLT (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es menor a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.12 void BLX (uint32_t * *registro*, uint32_t *Registro*)

5.7.2.13 void BMI (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si el dato es negativo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.14 void BNE (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si los datos no son iguales.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.15 void BPL (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si el dato es positivo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.16 void BVC (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si no hay sobreflujo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.17 void BVS (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si hay sobreflujo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.7.2.18 void BX (uint32_t * *registro*, uint32_t *Salto*)

5.8 Instrucciones_salto.h File Reference

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

Functions

- void **BEQ** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si los datos son iguales.
- void **BNE** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si los datos no son iguales.
- void **BCS** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es mayor o igual a otro (sin signo)
- void **BCC** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si un dato es menor a otro (sin signo)
- void **BMI** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si el dato es negativo.
- void **BPL** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si el dato es positivo.
- void **BVS** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si hay sobreflujo.
- void **BVC** (uint32_t *registro, int Salto, int *Banderas)
Funcion de salto, si no hay sobreflujo.
- void **BHI** (uint32_t *registro, int Salto, int *Banderas)

- Funcion de salto, si un dato es mayor que el otro (sin signo)*
- void **BLS** (uint32_t *registro, int Salto, int *Banderas)
- Funcion de salto, si un dato es mayor o igual a otro (sin signo)*
- void **BGE** (uint32_t *registro, int Salto, int *Banderas)
- Funcion de salto, si un dato es mayor o igual a otro (con signo)*
- void **BLT** (uint32_t *registro, int Salto, int *Banderas)
- Funcion de salto, si un dato es menor a otro (con signo)*
- void **BGT** (uint32_t *registro, int Salto, int *Banderas)
- Funcion de salto, si un dato es mayor a otro (con signo)*
- void **BLE** (uint32_t *registro, int Salto, int *Banderas)
- Funcion de salto, si un dato es menor o igual a otro (con signo)*
- void **B** (uint32_t *registro, int Salto)
- Funcion de salto, sin condicion.*
- void **BL** (uint32_t *registro, int salto)
- void **BX** (uint32_t *registro, uint32_t Salto)
- void **BLX** (uint32_t *registro, uint32_t Registro)

5.8.1 Function Documentation

5.8.1.1 void B (uint32_t * registro, int Salto)

Funcion de salto, sin condicion.

Parameters

<i>PC</i>	Intrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar

Returns

No retorna

5.8.1.2 void BCC (uint32_t * registro, int Salto, int * Banderas)

Funcion de salto, si un dato es menor a otro (sin signo)

Parameters

<i>PC</i>	Intrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.3 void BCS (uint32_t * registro, int Salto, int * Banderas)

Funcion de salto, si un dato es mayor o igual a otro (sin signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.4 void BEQ (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si los datos son iguales.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.5 void BGE (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es mayor o igual a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.6 void BGT (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es mayor a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.7 void BHI (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es mayor que el otro (sin signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.8 void BL (uint32_t * *registro*, int *salto*)

5.8.1.9 void BLE (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es menor o igual a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.10 void BLS (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es mayor o igual a otro (sin signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.11 void BLT (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si un dato es menor a otro (con signo)

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.12 void BLX (uint32_t * *registro*, uint32_t *Registro*)

5.8.1.13 void BMI (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si el dato es negativo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.14 void BNE (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si los datos no son iguales.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.15 void BPL (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si el dato es positivo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.16 void BVC (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si no hay sobreflujo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.17 void BVS (uint32_t * *registro*, int *Salto*, int * *Banderas*)

Funcion de salto, si hay sobreflujo.

Parameters

<i>PC</i>	Instrucciones a la que se le hara el salto correspondiente
<i>Salto</i>	correspondientes a realizar
<i>Banderas</i>	Quien se analizara para ver si se realiza el salto

Returns

No retorna

5.8.1.18 void BX (uint32_t * *registro*, uint32_t *Salto*)

5.9 main.c File Reference

Codigo en el cual se ejecutaran las funciones, utilizando todas las librerias creadas para cada una de ellas ademas de la libreria curses.h para la interfaz del emulador, y de la libreria decoder.h para obtener las instrucciones del documento de texto.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <curses.h>
#include "Registros.h"
#include "Instrucciones.h"
#include "Instrucciones_desplazamiento.h"
#include "decoder.h"
```

Functions

- int [main](#) (void)

5.9.1 Detailed Description

Codigo en el cual se ejecutaran las funciones, utilizando todas las librerias creadas para cada una de ellas ademas de la libreria curses.h para la interfaz del emulador, y de la libreria decoder.h para obtener las instrucciones del documento de texto.

5.9.2 Function Documentation

5.9.2.1 int main (void)

5.10 Registros.c File Reference

Contiene la funcion para poder imprimir en pantalla los registros.

```
#include "Registros.h"
#include <curses.h>
```

Functions

- void [MostrarRegistro](#) (uint32_t *registro)
Funcion que muestra el registro.

5.10.1 Detailed Description

Contiene la funcion para poder imprimir en pantalla los registros.

5.10.2 Function Documentation

5.10.2.1 void MostrarRegistro (uint32_t * registro)

Funcion que muestra el registro.

Parameters

<i>*registro</i>	Arreglo que contiene los registros
------------------	------------------------------------

Returns

No hay retorno de la funcion

5.11 Registros.h File Reference

Archivo que contiene la definicion de la funcion para imprimir en pantalla los registros.

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

Functions

- void [MostrarRegistro](#) (uint32_t *registro)
Funcion que muestra el registro.

5.11.1 Detailed Description

Archivo que contiene la definicion de la funcion para imprimir en pantalla los registros.

5.11.2 Function Documentation

5.11.2.1 void MostrarRegistro (uint32_t * registro)

Funcion que muestra el registro.

Parameters

<i>*registro</i>	Arreglo que contiene los registros
------------------	------------------------------------

Returns

No hay retorno de la funcion

