Documentacion Proyecto

Generado por Doxygen 1.8.10

Viernes, 18 de Septiembre de 2015 14:57:25

Índice general

Capítulo 1

Índice de estructura de datos

1.1. Estructura de datos

Lista de estructuras con una breve descripción:

Capítulo 2

Indice de archivos

2.1. Lista de archivos

Lista de todos los archivos con descripciones breves:

Capítulo 3

Documentación de las estructuras de datos

3.1. Referencia de la Estructura ins_t

`#include <decoder.h>`

Campos de datos

- char ** array

3.1.1. Documentación de los campos

3.1.1.1. char** array

La documentación para esta estructura fue generada a partir del siguiente fichero:

- decoder.h

3.2. Referencia de la Estructura instruction_t

`#include <decoder.h>`

Campos de datos

- 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. Documentación de los campos

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

La documentación para esta estructura fue generada a partir del siguiente fichero:

- decoder.h

Capítulo 4

Documentación de archivos

4.1. Referencia del Archivo alu.c

```
#include "alu.h"
#include "registros.h"
```

Funciones

- void ADD (uint32_t *Ra, uint32_t *Rb, uint32_t *Rc)
  funcion que suma
- void SUB (uint32_t *Ra, uint32_t *Rb, uint32_t *Rc)
  funcion que resta
- void MUL (uint32_t *Ra, uint32_t *Rb, uint32_t *Rc)
  funcion que multiplica dos registros
- void AND (uint32_t *Ra, uint32_t *Rb, uint32_t *Rc)
  funcion de producto logico
- void OR (uint32_t *Ra, uint32_t *Rb, uint32_t *Rc)
  funcion de suma logica
- void EOR (uint32_t *Ra, uint32_t *Rb, uint32_t *Rc)
  funcion de or exclusiva
- void MOV (uint32_t *Ra, uint32_t *Rb)
  funcion que escribe un valor de un registro en otro registro
- void MOVS (uint32_t *Ra, uint32_t inmediato)
  Copia el valor del inmediato en el registro.
- void ADDS (uint32_t *Ra, uint32_t *Rb, uint32_t inmediato)
  Suma un registro con un numero.
- void SUBS (uint32_t *Ra, uint32_t *Rb, uint32_t inmediato)
  Resta un registro con un numero.
- void MULS (uint32_t *Ra, uint32_t *Rb, uint32_t inmediato)
  Multiplica un registro con un numero.

4.1.1. Documentación de las funciones

4.1.1.1. void ADD (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t * *Rr*)

funcion que suma

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

no retorna nada

4.1.1.2. void ADDS (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t *inmediato*)

Suma un registro con un numero.

Parámetros

Rd,guarda	el resultado
Rm	operando 1
inmediato	operando 2

Devuelve

no retorna nada

4.1.1.3. void AND (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion de producto logico

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.1.1.4. void EOR (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion de or exclusiva

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.1.1.5. void MOV (uint32_t * *Rd,* uint32_t * *Rm*)

funcion que escribe un valor de un registro en otro registro

Parámetros

Rd	guarda el resultado
Rm	operando 1

Devuelve

un entero de 32 bits con el resultado

4.1.1.6. void MOVS (uint32_t * *Rd,* uint32_t *inmediato*)

Copia el valor del inmediato en el registro.

Parámetros

Rd	guarda el resultado
inmediato	operando 2

Devuelve

no retorna nada

4.1.1.7. void MUL (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion que multiplica dos registros

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.1.1.8. void MULS (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t *inmediato*)

Multiplica un registro con un numero.

Parámetros

Rd,guarda	el resultado
Rm	operando 1
inmediato	operando 2

Devuelve

no retorna nada

4.1.1.9. void OR (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion de suma logica

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.1.1.10. void SUB (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion que resta

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.1.1.11. void SUBS (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t *inmediato*)

Resta un registro con un numero.

Parámetros

Rd	guarda el resultado
Rm	operando 1
inmediato	operando 2

Devuelve

no retorna nada

4.2. Referencia del Archivo alu.h

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

Funciones

- void ADDS (uint32_t *Rd, uint32_t *Rm, uint32_t inmediato)
  Suma un registro con un numero.
- void MOVS (uint32_t *Rd, uint32_t inmediato)
  Copia el valor del inmediato en el registro.
- void SUBS (uint32_t *Rd, uint32_t *Rm, uint32_t inmediato)
  Resta un registro con un numero.
- void MULS (uint32_t *Rd, uint32_t *Rm, uint32_t inmediato)

Multiplica un registro con un numero.

- void ADD (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion que suma
- void SUB (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion que resta
- void AND (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion de producto logico
- void OR (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion de suma logica
- void EOR (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion de or exclusiva
- void MOV (uint32_t *Rd, uint32_t *Rm)

  funcion que escribe un valor de un registro en otro registro
- void MUL (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion que multiplica dos registros

4.2.1. Documentación de las funciones

4.2.1.1. void ADD (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t * *Rr*)

funcion que suma

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

no retorna nada

4.2.1.2. void ADDS (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t *inmediato*)

Suma un registro con un numero.

Parámetros

Rd,guarda	el resultado
Rm	operando 1
inmediato	operando 2

Devuelve

no retorna nada

4.2.1.3. void AND (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t * *Rr*)

funcion de producto logico

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.2.1.4. void EOR (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t * *Rr*)

funcion de or exclusiva

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.2.1.5. void MOV (uint32_t * *Rd*, uint32_t * *Rm*)

funcion que escribe un valor de un registro en otro registro

Parámetros

Rd	guarda el resultado
Rm	operando 1

Devuelve

un entero de 32 bits con el resultado

4.2.1.6. void MOVS (uint32_t * *Rd*, uint32_t *inmediato*)

Copia el valor del inmediato en el registro.

Parámetros

Rd	guarda el resultado
inmediato	operando 2

Devuelve

no retorna nada

4.2.1.7. void MUL (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t * *Rr*)

funcion que multiplica dos registros

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.2.1.8. void MULS (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t *inmediato*)

Multiplica un registro con un numero.

Parámetros

Rd,guarda	el resultado
Rm	operando 1
inmediato	operando 2

Devuelve

no retorna nada

4.2.1.9. void OR (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion de suma logica

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.2.1.10. void SUB (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion que resta

Parámetros

Rd	guarda el resultado
Rm	operando 1
Rr	operando 2

Devuelve

un entero de 32 bits con el resultado

4.2.1.11. void SUBS (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t *inmediato*)

Resta un registro con un numero.

Parámetros

Rd	guarda el resultado
Rm	operando 1
inmediato	operando 2

Devuelve

no retorna nada

4.3. Referencia del Archivo banderas.c

```
#include "banderas.h"
#include "registros.h"
```

Funciones

- void banderas (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion que registran los estados del microprocesador

4.3.1. Documentación de las funciones

4.3.1.1. void banderas (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t * *Rr*)

funcion que registran los estados del microprocesador

Parámetros

Rd	operando 1
Rm	operando 2
Rr	operando 3

Devuelve

no retorna nada

Definir una estructura, declarar dentro un vector de banderas, retornar la estructura a los registros, estos la retornan al main y se imprime

4.4. Referencia del Archivo banderas.h

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

Funciones

- void banderas (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion que registran los estados del microprocesador

4.4.1. Documentación de las funciones

4.4.1.1. void banderas (uint32_t * *Rd,* uint32_t * *Rm,* uint32_t * *Rr*)

funcion que registran los estados del microprocesador

Parámetros

Rd	operando 1
Rm	operando 2
Rr	operando 3

Devuelve

no retorna nada

Definir una estructura, declarar dentro un vector de banderas, retornar la estructura a los registros, estos la retornan al main y se imprime

4.5. Referencia del Archivo decoder.c

```
#include "decoder.h"
#include "alu.h"
#include "InstruccionesDesplazamiento.h"
```

Funciones

- void decodeInstruction (instruction_t instruction, uint32_t ∗Rd[ ], uint32_t ∗Rm[ ], uint32_t ∗Rr[ ])

4.5.1. Documentación de las funciones

4.5.1.1. void decodeInstruction (instruction_t *instruction,* uint32_t ∗ *Rd[ ],* uint32_t ∗ *Rm[ ],* uint32_t ∗ *Rr[ ]*)

4.6. Referencia del Archivo decoder.h

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

Estructuras de datos

- struct ins_t
- struct instruction_t

Funciones

- void decodeInstruction (instruction_t instruction, uint32_t ∗Rd[ ], uint32_t ∗Rm[ ], uint32_t ∗Rr[ ])
- instruction_t getInstruction (char ∗instStr)

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

4.6.1. Documentación de las funciones

4.6.1.1. int countLines (FILE * *fp*)

4.6.1.2. void decodeInstruction (instruction_t *instruction,* uint32_t * *Rd[ ],* uint32_t * *Rm[ ],* uint32_t * *Rr[ ]*)

4.6.1.3. instruction_t getInstruction (char * *instStr*)

Obtiene la instruccion separada por partes.

Parámetros

instrStr	cadena que contiene la instruccion.

Devuelve

instruction_t la instruccion separada por partes.

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

4.7. Referencia del Archivo InstruccionesDesplazamiento.c

```
#include "InstruccionesDesplazamiento.h"
#include "registros.h"
```

Funciones

- void LSL (uint32_t *Ra, uint32_t *Rb, uint32_t inmediato)
  funcion de desplazamiento logico a la izquierda
- void NOP ()
  funcion que no hace ninguna operacion solo aumenta el valor del pc
- void LSR (uint32_t *Ra, uint32_t *Rb, uint32_t inmediato)
  funcion de desplazamiento logico a la derecha
- void BIC (uint32_t *Ra, uint32_t *Rb)
  funcion que realiza una AND entre un registro y el complemento de otro
- void MVN (uint32_t *Ra, uint32_t *Rb)
  *funcion que gua*Raa el complemento de un numero*
- void RSBS (uint32_t *Ra, uint32_t *Rb)
  funcion que obtiene el complemento a dos de un numero
- void ASRS (int32_t *Ra, int32_t *Rb)
  funcion de desplazamiento aritmetico a la derecha
- void ROR (uint32_t *Ra, uint32_t *Rb)
  funcion de rotacion a la derecha

4.7.1. Documentación de las funciones

4.7.1.1. void ASRS (int32_t * *Ra,* int32_t * *Rb*)

funcion de desplazamiento aritmetico a la derecha

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.7.1.2. void BIC (uint32_t * *Ra,* uint32_t * *Rb*)

funcion que realiza una AND entre un registro y el complemento de otro

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.7.1.3. void LSL (uint32_t * *Ra,* uint32_t * *Rb,* uint32_t *inmediato*)

funcion de desplazamiento logico a la izquierda

Parámetros

Ra	operando 1, ademas guarda el resultado
Rb	operando 2
inmediato	operando 3

Devuelve

no retorna nada

4.7.1.4. void LSR (uint32_t * *Ra,* uint32_t * *Rb,* uint32_t *inmediato*)

funcion de desplazamiento logico a la derecha

Parámetros

Ra	operando1, ademas guarda el resultado
Rb	operando 2
inmediato	operando 3

Devuelve

no retorna nada

4.7.1.5. void MVN (uint32_t * *Ra,* uint32_t * *Rb*)

funcion que gua*Raa el complemento de un numero

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.7.1.6. void NOP ( )

funcion que no hace ninguna operacion solo aumenta el valor del pc

4.7.1.7. void ROR (uint32_t * *Ra*, uint32_t * *Rb*)

funcion de rotacion a la derecha

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.7.1.8. void RSBS (uint32_t * *Ra*, uint32_t * *Rb*)

funcion que obtiene el complemento a dos de un numero

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.8. Referencia del Archivo InstruccionesDesplazamiento.h

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

Funciones

- void LSL (uint32_t *Ra, uint32_t *Rb, uint32_t inmediato)

  funcion de desplazamiento logico a la izquierda
- void LSR (uint32_t *Ra, uint32_t *Rb, uint32_t inmediato)

  funcion de desplazamiento logico a la derecha
- void BIC (uint32_t *Ra, uint32_t *Rb)

funcion que realiza una AND entre un registro y el complemento de otro

- void MVN (uint32_t *Ra, uint32_t *Rb)

  *funcion que gua*Raa el complemento de un numero*

- void RSBS (uint32_t *Ra, uint32_t *Rb)

  funcion que obtiene el complemento a dos de un numero

- void NOP ()

  funcion que no hace ninguna operacion solo aumenta el valor del pc

- void ASRS (int32_t *Ra, int32_t *Rb)

  funcion de desplazamiento aritmetico a la derecha

- void ROR (uint32_t *Ra, uint32_t *Rb)

  funcion de rotacion a la derecha

4.8.1. Documentación de las funciones

4.8.1.1. void ASRS (int32_t * *Ra,* int32_t * *Rb*)

funcion de desplazamiento aritmetico a la derecha

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.8.1.2. void BIC (uint32_t * *Ra,* uint32_t * *Rb*)

funcion que realiza una AND entre un registro y el complemento de otro

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.8.1.3. void LSL (uint32_t * *Ra,* uint32_t * *Rb,* uint32_t *inmediato*)

funcion de desplazamiento logico a la izquierda

Parámetros

Ra	operando 1, ademas guarda el resultado
Rb	operando 2
inmediato	operando 3

Devuelve

no retorna nada

4.8.1.4. void LSR (uint32_t * *Ra,* uint32_t * *Rb,* uint32_t *inmediato*)

funcion de desplazamiento logico a la derecha

Parámetros

Ra	operando1, ademas guarda el resultado
Rb	operando 2
inmediato	operando 3

Devuelve

no retorna nada

4.8.1.5. void MVN (uint32_t * *Ra,* uint32_t * *Rb*)

funcion que gua*Raa el complemento de un numero

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.8.1.6. void NOP ( )

funcion que no hace ninguna operacion solo aumenta el valor del pc

4.8.1.7. void ROR (uint32_t * *Ra,* uint32_t * *Rb*)

funcion de rotacion a la derecha

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.8.1.8. void RSBS (uint32_t * *Ra,* uint32_t * *Rb*)

funcion que obtiene el complemento a dos de un numero

Parámetros

Ra	operando 1
Rb	operando 2

Devuelve

no retorna nada

4.9. Referencia del Archivo main.c

```
#include "registros.h"
#include "alu.h"
#include "decoder.h"
#include "InstruccionesDesplazamiento.h"
#include <curses.h>
```

Funciones

- int main (void)

4.9.1. Documentación de las funciones

4.9.1.1. int main (void)

4.10. Referencia del Archivo registros.c

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

Funciones

- void Registros (uint32_t *Rd, uint32_t *Rm, uint32_t *Rr)

  funcion que muestra los registros

4.10.1. Documentación de las funciones

4.10.1.1. void Registros (uint32_t * *Rd*, uint32_t * *Rm*, uint32_t * *Rr*)

funcion que muestra los registros

Parámetros

Rd	operando 1
Rm	operando 2
Rr	operando 3

Devuelve

no retorna nada

4.11. Referencia del Archivo registros.h

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

Funciones

- void Registros (uint32_t ∗Rd, uint32_t ∗Rm, uint32_t ∗Rr)

  funcion que muestra los registros

Variables

- int pc

4.11.1. Documentación de las funciones

4.11.1.1. void Registros (uint32_t ∗ *Rd,* uint32_t ∗ *Rm,* uint32_t ∗ *Rr*)

funcion que muestra los registros

Parámetros

Rd	operando 1
Rm	operando 2
Rr	operando 3

Devuelve

no retorna nada

4.11.2. Documentación de las variables

4.11.2.1. int pc

4.12. Referencia del Archivo saltos.c

4.13. Referencia del Archivo saltos.h

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

4.14. Referencia del Archivo test.c