Contenido

1 Implementacion del MIPS

1.1 Intrucciones implementadas

1.1.1 Tipo R

1. Descripcion

Instruccion	Descripcion
SLL	Shiftea hacia la izquierda un registro (T) una cantidad listada en la instruccion (H) y lo almacena en otro registro (D)
SRL	Shiftea hacia la derecha un registro (T) una cantidad listada en la instruccion (H) y lo almacena en otro registro (D)
SRA	Shiftea aritmetico hacia la derecha un registro (T) una cantidad listada en la instruccion (H) y lo almacena en otro registro (D)
SLLV	Shiftea hacia la izquierda un registro (T) una cantidad que esta en otro registro (S) y lo almacena en otro registro (D)
SRLV	Shiftea hacia la derecha un registro (T) una cantidad que esta en otro registro (S) y lo almacena en otro registro (D)
SRAV	Shiftea aritmetico hacia la derecha un registro (T) una cantidad que esta en otro registro (S) y lo almacena en otro registro (D)
ADDU	Add unsigned, suma dos registros y almacena el resultado en un registro
SUBU	Substract unsigned, resta dos registros y almacena el resultado en un registro
AND	Bitwise AND dos registros y almacena el resultado en un registro
OR	Bitwise OR dos registros y almacena el resultado en un registro
XOR	Bitwise XOR dos registros y almacena el resultado en un registro
NOR	Bitwise NOR dos registros y almacena el resultado en un registro
SLT	Si S < T setea D a 1 else 0
JR	Jumpea a la addr del registro S
JALR	Jumpea a la addr del registro S y guarda PC + 4 en D

2. Señales

Instruccion	Codigo	2LUT	Branch	BEQ/BNE	JRS	JINM	RA
SLL	0000_00SS_SSST_TTTT_DDDD_DHHH_HH00_0000	1	0	0	0	0	0
SRL	0000_00SS_SSST_TTTT_DDDD_DHHH_HH00_0010	1	0	0	0	0	0
SRA	0000_00SS_SSST_TTTT_DDDD_DHHH_HH00_0011	1	0	0	0	0	0
SLLV	0000_00SS_SSST_TTTT_DDDD_DHHH_HH00_0100	1	0	0	0	0	0
SRLV	0000_00SS_SSST_TTTT_DDDD_DHHH_HH00_0110	1	0	0	0	0	0
SRAV	0000_00SS_SSST_TTTT_DDDD_DHHH_HH00_0111	1	0	0	0	0	0
ADDU	0000_00SS_SSST_TTTT_DDDD_D000_0010_0001	1	0	0	0	0	0
SUBU	0000_00SS_SSST_TTTT_DDDD_D000_0010_0011	1	0	0	0	0	0
AND	0000_00SS_SSST_TTTT_DDDD_D000_0010_0100	1	0	0	0	0	0
OR	0000_00SS_SSST_TTTT_DDDD_D000_0010_0101	1	0	0	0	0	0
XOR	0000_00SS_SSST_TTTT_DDDD_D000_0010_0110	1	0	0	0	0	0
NOR	0000_00SS_SSST_TTTT_DDDD_D000_0010_0111	1	0	0	0	0	0
SLT	0000_00SS_SSST_TTTT_DDDD_D000_0010_1010	1	0	0	0	0	0
JR	0000_00SS_SSST_TTTT_DDDD_D000_0000_1000	1	0	0	1	0	0
JALR	0000_00SS_SSST_TTTT_DDDD_D000_0000_1001	1	0	0	1	0	0

Instruccion	ALU	B/I	S/U	SHAMT	RE	WE	S/U	DATA_SIZE	REG_WE	MEM/ALU	DATA/PC
SLL	0111	0	0	1	0	0	0	00	1	1	0
SRL	0110	0	0	1	0	0	0	00	1	1	0
SRA	1000	0	0	1	0	0	0	00	1	1	0
SLLV	0111	0	0	0	0	0	0	00	1	1	0
SRLV	0110	0	0	0	0	0	0	00	1	1	0
SRAV	1000	0	0	0	0	0	0	00	1	1	0
ADDU	0000	0	0	0	0	0	0	00	1	1	0
SUBU	0001	0	0	0	0	0	0	00	1	1	0
AND	0010	0	0	0	0	0	0	00	1	1	0
OR	0011	0	0	0	0	0	0	00	1	1	0
XOR	0100	0	0	0	0	0	0	00	1	1	0
NOR	0101	0	0	0	0	0	0	00	1	1	0
SLT	1010	0	0	0	0	0	0	00	1	1	0
JR	0000	0	0	0	0	0	0	00	0	0	0
JALR	0000	0	0	0	0	0	0	00	1	0	1

1.1.2 Tipo I

1. Descripcion

Instruccion	Descripcion
LB	Load byte, carga un byte a un registro (T) desde MEM(S + I)
LH	Load half word, carga half word a un registro (T) desde MEM(S + I)
LW	Load word, carga word a un registro (T) desde MEM(S + I)
LBU	Load byte unsigned, carga unsigned byte a un registro (T) desde MEM(S + I)
LHU	Load half word unsigned, carga unsigned half word a un registro (T) desde MEM(S + I)
LWU	Load word unsigned, carga unsigned word a un registro (T) desde MEM(S + I)
SB	Store byte, carga el byte menos significativo desde un registro (T) a MEM(S + I)
SH	Store half word, carga el half word menos significativo desde un registro (T) a MEM(S + I)
SW	Store word, carga el word desde un registro (T) a MEM(S + I)
ADDI	Suma un registro (S) con el inmediato (I) y lo guarda en otro registro (T)
ANDI	Bitwise AND un registro (S) con el inmediato (I) y lo guarda en otro registro (T)
ORI	Bitwise OR un registro (S) con el inmediato (I) y lo guarda en otro registro (T)
XORI	Bitwise XOR un registro (S) con el inmediato (I) y lo guarda en otro registro (T)
LUI	El valor inmediato (I) es shifteado a la izquierda 16 bits y guardado en el registro (T)
SLTI	If S < I => T = 1, else T = 0
BEQ	Branchea a PC + I*4 si ambos registros (S y T) equalean
BNE	Branchea a PC + I*4 si ambos registros (S y T) no equalean

2. Señales

Instruccion	Codigo	2LUT	Branch	BEQ/BNE	JRS	JINM	RA
LB	1000_00SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
LH	1000_01SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
LW	1000_11SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
LBU	1001_00SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
LHU	1001_01SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
LWU	1001_11SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
SB	1010_00SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
SH	1010_01SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
SW	1010_11SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
ADDI	0010_01SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
ANDI	0011_00SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
ORI	0011_01SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
XORI	0011_10SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
LUI	0011_11SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
SLTI	0010_10SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	0	0	0	0	0
BEQ	0001_00SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	1	0	0	0	0
BNE	0001_01SS_SSST_TTTT_IIII_IIII_IIII_IIII	0	1	1	0	0	0

Instruccion	ALU	B/I	S/U	SHAMT	RE	WE	S/U	DATA_SIZE	REG_WE	MEM/ALU	DATA/PC
LB	0000	1	0	0	1	0	0	00	1	0	0
LH	0000	1	0	0	1	0	0	01	1	0	0
LW	0000	1	0	0	1	0	0	10	1	0	0
LBU	0000	1	0	0	1	0	1	00	1	0	0
LHU	0000	1	0	0	1	0	1	01	1	0	0
LWU	0000	1	0	0	1	0	1	10	1	0	0
SB	0000	1	0	0	0	1	0	00	0	0	0
SH	0000	1	0	0	0	1	0	01	0	0	0
SW	0000	1	0	0	0	1	0	10	0	0	0
ADDI	0000	1	0	0	0	0	0	00	1	1	0
ANDI	0010	1	1	0	0	0	0	00	1	1	0
ORI	0011	1	1	0	0	0	0	00	1	1	0
XORI	0100	1	1	0	0	0	0	00	1	1	0
LUI	1011	1	1	0	0	0	0	00	1	1	0
SLTI	1010	1	0	0	0	0	0	00	1	1	0
BEQ	0000	1	0	0	0	0	0	00	0	0	0
BNE	0000	1	0	0	0	0	0	00	0	0	0

1.1.3 Tipo J

1. Descripcion

Instruccion	Descripcion
J	PC = (PC & 0xF0000000) OR (I « 2)
JAL	Jumpea a la direccion calculada [PC = (PC & 0xF0000000) OR (I « 2)] y storea la direccion de retorno en RA = PC + 4

2. Señales

Instruccion	Codigo	2LUT	Branch	BEQ/BNE	JRS	JINM	RA
J	0000_10II_IIII_IIII_IIII_IIII_IIII_IIII	0	0	0	0	1	0
JAL	0000_11II_IIII_IIII_IIII_IIII_IIII_IIII	0	0	0	0	1	1

Instruccion	ALU	B/I	S/U	SHAMT	RE	WE	S/U	DATA_SIZE	REG_WE	MEM/ALU	DATA/PC
J	0000	0	0	0	0	0	0	00	0	0	0
JAL	0000	0	0	0	0	0	0	00	1	0	1

1.2 Señales de control

1.2.1 DEC

- RA
- SHAMT

1.2.2 EX

- ALU [4bits]
  - 0000 → ADD
  - 0001 → SUB
  - 0010 → AND
  - 0011 → OR
  - 0100 → XOR
  - 0101 → NOR
  - 0110 → SRL
  - 0111 → SLL
  - 1000 → SRA
  - 1001 → SLA
  - 1010 → SLT
  - 1011 → LUI
- B/I
- S/U

1.2.3 MEM

- RE
- WE

- S/U
- DATA_SIZE [2bits]
  - `00` → `byte`
  - `01` → `half-word`
  - `10` → `word`
  - `11` → `NO!`

1.2.4 WB

- REG_WE
- MEM/ALU
- DATA/PC

1.3 Estructura de bits de control

1.3.1 EX (7 bits)

6	2	1	0
ALUCTRL	B/I	S/U	SHAMT

1.3.2 MEM (5 bits)

4	3	2	1-0
RE	WE	S/U	DSIZE

1.3.3 WB (8 bits)

7	2	1	0
DEST	REG_WE	MEM/ALU	DATA/PC

1.4 Register File

Register Number	Conventional Name	Usage
$0	$zero	Hard-wired to 0
$1	$at	Reserved for pseudo-instructions
$2 - $3	$v0, $v1	Return values from functions
$4 - $7	$a0 - $a3	Arguments to functions - not preserved by subprograms
$8 - $15	$t0 - $t7	Temporary data, not preserved by subprograms
$16 - $23	$s0 - $s7	Saved registers, preserved by subprograms
$24 - $25	$t8 - $t9	More temporary registers, not preserved by subprograms
$26 - $27	$k0 - $k1	Reserved for kernel. Do not use.
$28	$gp	Global Area Pointer (base of global data segment)
$29	$sp	Stack Pointer
$30	$fp	Frame Pointer
$31	$ra	Return Address

1.5 Interfaz MIPS - MicroBlaze

1.5.1 MicroBlaze → Interfaz

1. Tipo CTRL

NAME	31	25	15
START	0000_01	VXXX	XXXX
RESET	0000_10	VXXX	XXXX
MODE_GET	0010_00	VXXX	XXXX
MODE_SET_CONT	0010_01	VXXX	XXXX
MODE_SET_STEP	0010_10	VXXX	XXXX
STEP	1000_00	VXXX	XXXX
GOT_DATA	1001_00	VXXX	XXXX
GIB_DATA	1001_01	VXXX	XXXX

2. Tipo LOAD

NAME	31	25	15
LOAD_INSTR_LSB	0001_00	VAAA	DDDD
LOAD_INSTR_MSB	0001_01	VAAA	DDDD

3. Tipo REQ

NAME	31	25	15
REQ_MEM_DATA	0000_11	V0_0000_0001	AAAA
REQ_MEM_INSTR	0000_11	V0_0000_0010	AAAA
REQ_REG	0000_11	V0_0000_0100	AAAA
REQ_REG_PC	0000_11	V0_0000_0101	XXXX
REQ_LATCH_FETCH_DATA	0000_11	V0_0000_1000	XXXX
REQ_LATCH_FETCH_CTRL	0000_11	V0_0000_1001	XXXX
REQ_LATCH_DECO_DATA	0000_11	V0_0001_0000	XXXX
REQ_LATCH_DECO_CTRL	0000_11	V0_0001_0001	XXXX
REQ_LATCH_EXEC_DATA	0000_11	V0_0010_0000	XXXX
REQ_LATCH_EXEC_CTRL	0000_11	V0_0010_0001	XXXX
REQ_LATCH_MEM_DATA	0000_11	V0_0100_0000	XXXX
REQ_LATCH_MEM_CTRL	0000_11	V0_0100_0001	XXXX

1.5.2 Interfaz → MicroBlaze

NAME	31	25	15
OK	0000_11	XXXX	XXXX
NOK	0000_10	XXXX	XXXX
EOP	0001_00	XXXX	XXXX
MODE_CONT	0010_01	XXXX	XXXX
MODE_STEP	0010_10	XXXX	XXXX