

RV32I BASE INSTRUCTION SET

Tipo U

1. LUI

31	12	11 7	6 0	
Inmediato[31:12]		RDest	0110111	RDest = se(Inmediato[31:12] << 12)

	Formato:	LUI RDest, Inmediato
	Propósito:	Cargar un valor Inmediato de 20 bits extendido en signo en la parte alta del registro RDest, y poner en 0 los 12 bits de la parte baja.
	Excepciones:	No produce excepción alguna.

1. AUIPC

31	12	11 7	6 0	
Imm[31:12]		RDest	0010111	RDest = pc + se(Imm[31:12] << 12)

	Formato:	AUIPC RDest, Inmediato
	Propósito:	Sumar el Inmediato con signo extendido de 20 bits, corrido a la izquierda por 12 bits, al pc, y guardar el resultado en RDest.
	Excepciones:	No produce excepción alguna.

Tipo S

1. SB

31 25	24 20	19 15	14 12	11 7	6 0	
Imm[11:5]	RFuente2	RFuente1	000	Imm[4:0]	0100011	Dirección de Memoria = RFuente1 + Imm[4:0]

	Formato:	SB RFuente2, Inmediato (RFuente1)
	Propósito:	Almacenar los 8-bits menos significativos de RFuente2 en la localidad especificada por la Dirección de Memoria.
	Excepciones:	Las excepciones que pueden ocurrir: 1) Relleno de TLB, 2) Entrada invalida de TLB, 3) Entrada modificada de TLB, 4) Error de bus y 5) Error de dirección.

1. SH

31 25	24 20	19 15	14 12	11 7	6 0	
Imm[11:5]	RFuente2	RFuente1	001	Imm[4:0]	0100011	Dirección de Memoria = RFuente1 + Imm[4:0]

	Formato:	SH RFuente2, Inmediato (RFuente1)
	Propósito:	Almacenar los 16-bits menos significativos de RFuente2 en la localidad especificada por la Dirección de Memoria.
	Excepciones:	Las excepciones que pueden ocurrir: 1) Relleno de TLB, 2) Entrada invalida de TLB, 3) Entrada modificada de TLB, 4) Error de bus y 5) Error de dirección.

1. SW

31 Imm[11:5]	25 RFuente2	24 20 RFuente1	19 15 010	14 12 Imm[4:0]	11 7 0100011	6 0 0	Dirección de Memoria = RFuente1 + Inmediato
-----------------	----------------	-------------------	--------------	-------------------	-----------------	----------	--

	Formato:	SW RFuente2, Inmediato (RFuente1)
	Propósito:	Almacenar los 32-bits menos significativos de RFuente2 en la localidad especificada por la Dirección de Memoria.
	Excepciones:	Las excepciones que pueden ocurrir: 1) Relleno de TLB, 2) Entrada invalida de TLB, 3) Entrada modificada de TLB, 4) Error de bus y 5) Error de dirección.

Tipo R

1. SLLI

31 0000000	25 Shamt	24 20 RFuente1	19 15 001	14 12 RDest	11 7 0010011	6 0 0	RDest = RFuente1 << Shamt
---------------	-------------	-------------------	--------------	----------------	-----------------	----------	---------------------------

	Formato:	SLLI RDest, RFuente1, Shamt
	Propósito:	Desplazar a la izquierda el valor RFuente1 por el valor en bits de Shamt, los bits liberados son reemplazados por ceros, y guardar el resultado en RDest.
	Excepciones:	

1. SRRI

31 0000000	25 Shamt	24 20 RFuente1	19 15 101	14 12 RDest	11 7 0010011	6 0 0	RDest = RFuente1 >> Shamt
---------------	-------------	-------------------	--------------	----------------	-----------------	----------	---------------------------

	Formato:	SRLI RDest, RFuente1, Shamt
	Propósito:	Desplazar a la derecha el valor RFuente1 por el valor en bits de Shamt, los bits liberados son reemplazados por ceros, y guardar el resultado en RDest.
	Excepciones:	

1. SRAI

31 0100000	25 Shamt	24 20 RFuente1	19 15 101	14 12 RDest	11 7 0010011	6 0 0	RDest = RFuente1 >> Shamt
---------------	-------------	-------------------	--------------	----------------	-----------------	----------	---------------------------

	Formato:	SRAI RDest, RFuente1, Shamt
	Propósito:	Desplazar a la derecha el valor RFuente1 por el valor en bits de Shamt, los bits liberados son reemplazados por copias del bit más significativo de RFuente1, y guardar el resultado en RDest.
	Excepciones:	

1. ADD

31	25	24	20	19	15	14	12	11	7	6	0	
0000000		RFuente2		RFuente1		000		RDest		0110011		RDest = RFuente1 + RFuente2

	Formato:	ADD RDest, RFuente1, RFuente2
	Propósito:	Sumar RFuente1 con RFuente2 y guardar el resultado en RDest.
	Excepciones:	Si ocurre sobre flujo en la operación, este se ignora y el resultado se escribe en RDest

2. SUB

31	25	24	20	19	15	14	12	11	7	6	0	
0100000		RFuente2		RFuente1		000		RDest		0110011		RDest = RFuente1 - RFuente2

	Formato:	SUB RDest, RFuente1, RFuente2
	Propósito:	Restar RFuente2 de RFuente1 y guardar el resultado en RDest.
	Excepciones:	Si ocurre sobre flujo en la operación, este se ignora y el resultado se escribe en RDest

1. SLL

31	25	24	20	19	15	14	12	11	7	6	0	
0000000		RFuente2		RFuente1		001		RDest		0110011		RDest = RFuente1 << RFuente2

	Formato:	SLL RDest, RFuente1, RFuente2
	Propósito:	Desplazar a la izquierda el valor RFuente1 por el valor en bits de RFuente2, los bits liberados son reemplazados por ceros, y guardar el resultado en RDest.
	Excepciones:	No produce excepción alguna.

1. SLT

31	25	24	20	19	15	14	12	11	7	6	0	
0000000		RFuente2		RFuente1		010		RDest		0110011		RDest = RFuente1 < RFuente2

	Formato:	SLT RDest, RFuente1, RFuente2
	Propósito:	Comparar (menor que) el contenido de RFuente1 con RFuente2 como valores enteros con signo, y guardar el resultado en RDest, si el resultado es verdadero (1), de otra forma (0)
	Excepciones:	No produce excepción alguna.

1. SLTU

31 0000000	25 RFuente2	20 RFuente1	19 15 011	14 12 RDest	11 7 0110011	6 0 $RDest = RFuente1 < RFuente2$
						$RDest = RFuente1 < RFuente2$

Formato:	SLT RDest, RFuente1, RFuente2
Propósito:	Compara (menor que) el contenido de RFuente1 y RFuente2 como valores enteros sin signo, y guarda el resultado booleano de la comparación en RDest, si el resultado es verdadero (1), de otra forma (0)
Excepciones:	No produce excepción alguna.

1. XOR

31 0000000	25 RFuente2	20 RFuente1	19 15 100	14 12 RDest	11 7 0110011	6 0 $RDest = RFuente1 \oplus RFuente2$
						$RDest = RFuente1 \oplus RFuente2$

Formato:	XOR RDest, RFuente1, RFuente2
Propósito:	Realizar una operación lógica XOR bit a bit de RFuente1 con RFuente2 y guardar el resultado en RDest.
Excepciones:	No produce excepción alguna.

1. SRL

31 0000000	25 RFuente2	20 RFuente1	19 15 101	14 12 RDest	11 7 0110011	6 0 $RDest = RFuente1 \gg RFuente2$
						$RDest = RFuente1 \gg RFuente2$

Formato:	SRL RDest, RFuente1, RFuente2
Propósito:	Desplazar a la derecha el valor RFuente1 por el valor en bits de RFuente2, los bits liberados son reemplazados por ceros, y guardar el resultado en RDest.
Excepciones:	No produce excepción alguna.

1. SRA

31 0100000	25 RFuente2	20 RFuente1	19 15 101	14 12 RDest	11 7 0110011	6 0 $RDest = RFuente1 \gg RFuente2$
						$RDest = RFuente1 \gg RFuente2$

Formato:	SRA RDest, RFuente1, RFuente2
Propósito:	Desplazar a la derecha el valor RFuente1 por el valor en bits de RFuente2, los bits liberados son reemplazados por copias del bit más significativo de RFuente1, y guardar el resultado en RDest.
Excepciones:	No produce excepción alguna.

1. OR

31	25	24	20	19	15	14	12	11	7	6	0	
0000000	RFuente2		RFuente1		110		RDest		0110011			RDest = RFuente1 RFuente2

	Formato:	OR RDest, RFuente1, RFuente2
	Propósito:	Realizar una operación lógica OR bit a bit de RFuente1 con RFuente2 y guardar el resultado en RDest.
	Excepciones:	No produce excepción alguna.

1. AND

31	25	24	20	19	15	14	12	11	7	6	0	
0000000	RFuente2		RFuente1		111		RDest		0110011			RDest = RFuente1 & RFuente2

	Formato:	AND RDest, RFuente1, RFuente2
	Propósito:	Realizar una operación lógica AND bit a bit de RFuente1 con RFuente2 y guardar el resultado en RDest.
	Excepciones:	No produce excepción alguna.

RV64I BASE INSTRUCTION SET (in addition to RV32I)

Tipo S

1. SD

31	25	24	20	19	15	14	12	11	7	6	0	
Imm[11:5]	RFuente2		RFuente1		011		Imm[4:0]		0100011			Dirección de Memoria = RFuente1 + Inmediato

	Formato:	SD RFuente2, Inmediato (RFuente1,)
	Propósito:	Almacenar la doble palabra (64 bits) del valor almacenado en RFuente1 a la localidad especificada por Dirección de Memoria.
	Excepciones:	

Tipo R

1. SLLI

31 0000000	25 Shamt	24 20 RFuente1	19 15 001	14 12 RDest	11 7 0010011	6 0 $RDest = RFuente1 \ll Shamt$

	Formato:	SLLI RDest, RFuente1, Shamt
	Propósito:	Desplazar a la izquierda el valor de RFuente1 por los bits de Shamt y guardar el resultado en RDest
	Excepciones:	Genera una instrucción ilegal si el bit Imm[5] ≠ 0

1. SRLI

31 0000000	25 Shamt	24 20 RFuente1	19 15 101	14 12 RDest	11 7 0010011	6 0 $RDest = RFuente1 \gg Shamt$

	Formato:	SRLI RDest, RFuente1, Shamt
	Propósito:	Desplazar a la derecha el valor de RFuente1 por los bits de Shamt y guardar el resultado en RDest
	Excepciones:	Genera una instrucción ilegal si el bit Imm[5] ≠ 0

1. SRAI

31 0100000	25 Shamt	24 20 RFuente1	19 15 101	14 12 RDest	11 7 0010011	6 0 $RDest = RFuente1 \gg RFuente2$

	Formato:	SRAI RDest, RFuente1, Shamt
	Propósito:	Desplazar aritméticamente a la derecha el valor de RFuente1 por el valor en bits de Shamt, duplicando el bit de signo (bit 31) en los espacios superiores vacíos y guardar el resultado en RDest
	Excepciones:	Genera una instrucción ilegal si el bit Imm[5] ≠ 0

1. SLLIW

31 0000000	25 Shamt	24 20 RFuente1	19 15 001	14 12 RDest	11 7 0010011	6 0 $RDest = RFuente1 \ll RFuente2$

	Formato:	SLLIW RDest, RFuente1, Shamt

	Propósito:	Desplazar a la izquierda los 32 bits del valor de RFuente1 por los bits de Shamt y produce un signo extendido de 32 bits para completar los 64-bits y guardar el resultado en RDest					
	Excepciones:	Genera una instrucción ilegal si el bit Imm[5] ≠ 0					

1. SRLIW

31 25	24 20	19 15	14 12	11 7	6 0	RDest = RFuente1 >> RFuente2
0000000	Shamt	RFuente1	101	RDest	0010011	

	Formato:	SRLIW RDest, RFuente1, RFuente2
	Propósito:	Desplaza a la derecha los 32 bits del valor de RFuente1 por los bits de Shamt y produce un valor de signo de 32 bits para completar los 64-bits y guardar el resultado en RDest
	Excepciones:	Genera una instrucción ilegal si el bit Imm[5] ≠ 0

1. SRAIW

31 25	24 20	19 15	14 12	11 7	6 0	RDest = RFuente1 >> RFuente2
0100000	Shamt	RFuente1	101	RDest	0010011	

	Formato:	SRAIW RDest, RFuente1, RFuente2
	Propósito:	Desplazar aritméticamente a la derecha los 32 bits del valor de RFuente1 por los bits de Sham, duplicando el bit de signo (bit 31) en los espacios superiores vacíos y produce un valor de signo de 32 bits para completar los 64 bits y guardar el resultado en RDest
	Excepciones:	Genera una instrucción ilegal si el bit Imm[5] ≠ 0

1. ADDW

31 25	24 20	19 15	14 12	11 7	6 0	RDest = se(RFuente1 + RFuente2)
0000000	RFuente2	RFuente1	000	RDest	0111011	

	Formato:	ADDW RDest, RFuente1, RFuente2
	Propósito:	Sumar valores de 32 bits de RFuente1 con RFuente2 produciendo resultados con signo extendido de 32 bits y guardar el resultado en RDest
	Excepciones:	Si ocurre sobre flujo en la operación, este se ignora y el resultado se escribe en RDest con signo extendido de 32 bits

2. SUBW

31 0100000	25 RFuente2	20 RFuente1	19 15 000	14 12 RDest	11 7 0111011	6 0 $RDest = se(RFuente1 - RFuente2)$
						$RDest = se(RFuente1 - RFuente2)$

Formato:	SUBW RDest, RFuente1, RFuente2					
Propósito:	Restar valores de 32 bits de RFuente1 con RFuente2 produciendo resultados con signo extendido de 32 bits y guardar el resultado en RDest					
Excepciones:	Si ocurre sobre flujo en la operación RDest no se modifica y se dispara una excepción, si no ocurre excepción de sobreflujo el resultado se escribe en RDest.					

1. SLLW

31 0000000	25 RFuente2	20 RFuente1	19 15 001	14 12 RDest	11 7 0111011	6 0 $RDest = se(RFuente1 << RFuente2)$
						$RDest = se(RFuente1 << RFuente2)$

Formato:	SLLW RDest, RFuente1, RFuente2					
Propósito:	Desplazar a la izquierda los 32 bits más bajos de RFuente1 por el valor en bits de RFuente2, los bits liberados son reemplazados por ceros, y guardar el resultado con signo extendido en RDest					
Excepciones:	No produce excepción alguna.					

1. SRLW

31 0000000	25 RFuente2	20 RFuente1	19 15 101	14 12 RDest	11 7 0111011	6 0 $RDest = se(RFuente1 >> RFuente2)$
						$RDest = se(RFuente1 >> RFuente2)$

Formato:	SRLW RDest, RFuente1, RFuente2					
Propósito:	Desplazar a la derecha los 32 bits más bajos de RFuente1 por el valor en bits de RFuente2, los bits liberados son reemplazados por ceros y guardar el resultado con signo extendido en RDest					
Excepciones:	No produce excepción alguna.					

1. SRAW

31 0100000	25 RFuente2	20 RFuente1	19 15 101	14 12 RDest	11 7 0111011	6 0 $RDest = RFuente1 >> RFuente2$
						$RDest = RFuente1 >> RFuente2$

Formato:	SRAW RDest, RFuente1, RFuente2					
Propósito:	Desplazar aritméticamente a la derecha los 32 bits más bajos de RFuente1 por los bits de RFuente2, duplicando el bit de signo y guardar el resultado en RDest					

	Excepciones:	No produce excepción alguna.
--	--------------	------------------------------

RV32M STANDARD EXTENSION

1. MUL

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 000	11 7 RDest	6 0 0110011	RDest = RFuente1 * RFuente2
---------------	----------------	----------------------------	--------------	---------------	----------------	-----------------------------

Formato:	MUL RDest, RFuente1, RFuente2
Propósito:	Multiplicar enteros con signo de 32 bits de RFuente1 y RFuente2 y guardar el resultado en RDest
Excepciones:	No produce excepción alguna.

2. MULH

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 001	11 7 RDest	6 0 0110011	RDest = RFuente1 * RFuente2
---------------	----------------	----------------------------	--------------	---------------	----------------	-----------------------------

Formato:	MULH RDest, RFuente1, RFuente2
Propósito:	Multiplicar enteros con signo de 32 bits para producir un resultado de 64bits, la palabra de 32 bits más significativa es puesta en el registro RDest
Excepciones:	No produce excepción alguna.

1.

MULHSU

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 011	11 7 RDest	6 0 0110011	RDest = RFuente1 * RFuente2
---------------	----------------	----------------------------	--------------	---------------	----------------	-----------------------------

Formato:	MULHSU RDest, RFuente1, RFuente2
Propósito:	Multiplicar enteros con signo y con enteros sin signo para producir un resultado de 64bits, la palabra de 32 bits más significativa es puesta en el registro RDest
Excepciones:	No produce excepción alguna.

1. MULHU

31	25	24 20	19 15	14 12	11 7	6 0
----	----	-------	-------	-------	------	-----

0000001	RFuente2	RFuente1	011	RDest	0110011	$RDest = RFuente1 * RFuente2$
---------	----------	----------	-----	-------	---------	-------------------------------

	Formato:	MULHU RDest, RFuente1, RFuente2
	Propósito:	Multiplicar enteros sin signo para producir un resultado de 64bits, la palabra de 32 bits más significativa es puesta en el registro RDest
	Excepciones:	No produce excepción alguna.

1. DIV

31 25	24 20	19 15	14 12	11 7	6 0	$RDest = RFuente1 / RFuente2 =$
0000001	RFuente2	RFuente1	100	RDest	0110011	

	Formato:	DIV RDest, RFuente1, RFuente2
	Propósito:	Dividir enteros de 32 bit con signo y guardar el resultado en RDest.
	Excepciones:	No produce excepción alguna.

1. DIVU

31 25	24 20	19 15	14 12	11 7	6 0	$RDest = RFuente1 / RFuente2$
0000001	RFuente2	RFuente1	101	RDest	0110011	

	Formato:	DIVU RDest, RFuente1, RFuente2
	Propósito:	Dividir enteros de 32 bit sin signo y guardar el resultado en RDest.
	Excepciones:	No produce excepción alguna.

1. REM

31 25	24 20	19 15	14 12	11 7	6 0	$RDest = RFuente1 \% RFuente2$
0000001	RFuente2	RFuente1	110	RDest	0110011	

	Formato:	REM RDest, RFuente1, RFuente2
	Propósito:	Proveer el residuo con signo de la división de RFuente1 entre RFuente2 y guardarlo en RDest.
	Excepciones:	

1. REMU

31 25	24 20	19 15	14 12	11 7	6 0	$RDest = RFuente1 \% RFuente2$
0000001	RFuente2	RFuente1	111	RDest	0110011	

						RFuente2
	Formato:	REMU RDest, RFuente1, RFuente2				
	Propósito:	Proveer el residuo sin signo de la división de RFuente1 entre RFuente2 y guardarlo en RDest .				
	Excepciones:	Si ocurre sobre flujo en la operación RDest no se modifica y se dispara una excepción, si no ocurre excepción de sobreflujo el resultado se escribe en RDest .				

RV64M STANDARD EXTENSION (in addition to RV32M)

1. MULW

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 000	11 7 RDest	6 0 0111011	RDest = se (RFuente1 * RFuente2)
---------------	-----------------------	-----------------------------------	--------------	----------------------	----------------	---

	Formato:	MULW RDest, RFuente1, RFuente2
	Propósito:	Multiplicar RFuente1 por RFuente2 , tomar la palabra menos significativa de 32 bits y guardar el resultado con signo extendido en RDest .
	Excepciones:	Overflow aritmético ignorado

1. DIVW

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 100	11 7 RDest	6 0 0111011	RDest = se (RFuente1 / RFuente2)
---------------	-----------------------	-----------------------------------	--------------	----------------------	----------------	---

	Formato:	DIVW RDest, RFuente1, RFuente2
	Propósito:	Dividir la palabra menos significativa de 32 bits de RFuente1 por la palabra menos significativa de 32 bits de RFuente2 tratando el resultado como entero con signo y guardar el cociente de 32 bits con signo extendido en RDest .
	Excepciones:	

1. DIVUW

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 101	11 7 RDest	6 0 0111011	RDest = se (RFuente1 / RFuente2)
---------------	-----------------------	-----------------------------------	--------------	----------------------	----------------	---

	Formato:	DIVUW RDest, RFuente1, RFuente2
	Propósito:	Dividir la palabra menos significativa de 32 bits de RFuente1 por la palabra menos significativa de 32 bits de RFuente2 tratando el resultado como entero sin signo y guardar el cociente de

		32 bits son signo extendido en RDest.
	Excepciones:	

1. REMW

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 110	11 7 RDest	6 0 0111011	$RDest = se(RFuente1 \% RFuente2)$
---------------	----------------	----------------------------	--------------	---------------	----------------	------------------------------------

Formato:	REMW RDest, RFuente1, RFuente2
Propósito:	Proveer el residuo con signo extendido de la división de RFuente1 entre RFuente2 y guardarlo en RDest.
Excepciones:	

1. REMUW

31 0000001	25 RFuente2	24 20 19 15 RFuente1	14 12 111	11 7 RDest	6 0 0111011	$RDest = se(RFuente1 \% RFuente2)$
---------------	----------------	----------------------------	--------------	---------------	----------------	------------------------------------

Formato:	REMUW RDest, RFuente1, RFuente2
Propósito:	Proveer el residuo sin signo de la división de RFuente1 entre RFuente2 y guardarlo en RDest.
Excepciones:	

RV32A STANDARD EXTENSION

RV64A STANDARD EXTENSION (in addition to RV32A)

RV32F STANDARD EXTENSION

Tipo S

1. FSW

31 Imm[11:5]	25 RFuente2	24 20 19 15 RFuente1	14 12 010	11 7 Imm[4:0]	6 0 0100111	Dirección de Memoria = RFuente1 + Inmediato
-----------------	----------------	----------------------------	--------------	------------------	----------------	--

Formato:	FSW RFuente2, Inmediato (RFuente1)
Propósito:	Almacenar el número de punto flotante de precisión simple de RFuente2 en la localidad especificada por la Dirección de Memoria.
Excepciones:	

Tipo R-4

1. FMADD.S

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	00	RFuente2	RFuente1	Rm	RDest	1000011		$RDest = RFuente1 * RFuente2 + RFuente3$

	Formato:	FMADD.S RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplicar los valores de punto flotante de RFuente1 por RFuente2 después sumar RFuente3 al resultado y guardararlo en RDest.
	Excepciones:	

1. FMSUB.S

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	00	RFuente2	RFuente1	Rm	RDest	1000111		$RDest = RFuente1 * RFuente2 - RFuente3$

	Formato:	FMSUB.S RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplica los valores de punto flotante de RFuente1 por RFuente2 después restar RFuente3 al resultado y guardararlo en RDest
	Excepciones:	

1. FNMSUB.S

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	00	RFuente2	RFuente1	Rm	RDest	1001011		$RDest = -RFuente1 * RFuente2 + RFuente3$

	Formato:	FNMSUB.S RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplica los valores de punto flotante de -(RFuente1) por RFuente2 después sumar RFuente3 al resultado y guardararlo en RDest.
	Excepciones:	

1. FNMADD.S

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	00	RFuente2	RFuente1	Rm	RDest	1001111		$RDest = -RFuente1 * RFuente2 - RFuente3$

	Formato:	FNMADD.S RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplica los valores de punto flotante de -(RFuente1) por RFuente2 después restar RFuente3 al resultado y guardararlo en RDest
	Excepciones:	

Tipo R

1. FADD.S

31	25	24	20	19	15	14	12	11	7	6	0	
0000000		RFuente2		RFuente1		Rm		RDest		1010011		$RDest = RFuente1 + RFuente2$

	Formato:	FADD.S RDest, RFuente1, RFuente2
	Propósito:	Sumar los valores de precisión simple de punto flotante de RFuente1 con RFuente2 y guardar el resultado en RDest.
	Excepciones:	

1. FSUB.S

31	25	24	20	19	15	14	12	11	7	6	0	
0000100		RFuente2		RFuente1		Rm		RDest		1010011		$RDest = RFuente1 - RFuente2$

	Formato:	FSUB.S RDest, RFuente1, RFuente2
	Propósito:	Restar los valores de precisión simple de punto flotante de RFuente1 con RFuente2 y guardar el resultado en RDest.
	Excepciones:	

1. FMUL.S

31	25	24	20	19	15	14	12	11	7	6	0	
0001000		RFuente2		RFuente1		Rm		RDest		1010011		$RFuente1 * RFuente2$

	Formato:	FMUL.S RDest, RFuente1, RFuente2
	Propósito:	Multiplicar los valores de precisión simple de punto flotante de RFuente1 con RFuente2 y guardar el resultado en RDest.
	Excepciones:	

1. FDIV.S

31	25	24	20	19	15	14	12	11	7	6	0	
0001100		RFuente2		RFuente1		Rm		RDest		1010011		$RDest = RFuente1 / RFuente2$

	Formato:	FDIV RDest, RFuente1, RFuente2
	Propósito:	Dividir los valores de precisión simple de punto flotante de RFuente1 con RFuente2 y guardar el cociente redondeado de precisión simple en RDest.
	Excepciones:	

1. FSQRT.S

31 0101100	25 00000	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = Sqrt(RFuente1)
---------------	-------------	-------------------	-------------	----------------	-----------------	-------------------------------

Formato:	FSQRT RDest, RFuente1
Propósito:	Calcular la raíz cuadrada de RFuente1 guardar el resultado en RDest
Excepciones:	

1. FSGNJ.S

31 0010000	25 RFuente2	24 20 RFuente1	19 15 000	14 12 RDest	11 7 1010011	6 0 RDest ={RFuente2[31], RFuente1[30:0]}
---------------	----------------	-------------------	--------------	----------------	-----------------	--

Formato:	FSGNJ.S RDest, RFuente1, RFuente2
Propósito:	Construir un número nuevo de punto flotante de precisión simple del exponente y significado de RFuente1, su signo lo determina el bit de signo de RFuente2 y guardar el resultado en RDest.
Excepciones:	

1.FSGNJS.N

31 0010000	25 RFuente2	24 20 RFuente1	19 15 001	14 12 RDest	11 7 1010011	6 0 RDest ={~RFuente2[31], RFuente1[30:0]}
---------------	----------------	-------------------	--------------	----------------	-----------------	---

Formato:	FSGNJS.N RDest, RFuente1, RFuente2
Propósito:	Construir un número nuevo de punto flotante de precisión simple del exponente y significado de RFuente1, su signo lo determina el bit de signo opuesto de RFuente2 y guardar el resultado en RDest.
Excepciones:	

1.FSGNJS.S

31 0010000	25 RFuente2	24 20 RFuente1	19 15 010	14 12 RDest	11 7 1010011	6 0 RDest = {RFuente1[63] ^ RFuente2[63], RFuente1[62:0]}
---------------	----------------	-------------------	--------------	----------------	-----------------	--

Formato:	FSGNJS RDest, RFuente1, RFuente2
Propósito:	Construir un número nuevo de punto flotante de precisión simple del exponente y significado de RFuente1, su signo lo determina el XOR de RFuente1 con RFuente2 y guardar el resultado

		en RDest.
	Excepciones:	

1. FMIN.S

31 0010100	25 RFuente2	24 20 RFuente1	19 15 000	14 12 RDest	11 7 1010011	6 0 RDest = min (RFuente1, RFuente2)
---------------	----------------	-------------------	--------------	----------------	-----------------	--

	Formato:	FMIN.S RDest, RFuente1, RFuente2
	Propósito:	Encontrar el valor mínimo de los registros fuente y guardarlos en RDest
	Excepciones:	

1. FMAX.S

31 0010100	25 RFuente2	24 20 RFuente1	19 15 001	14 12 RDest	11 7 1010011	6 0 RDest = max (RFuente1, RFuente2)
---------------	----------------	-------------------	--------------	----------------	-----------------	--

	Formato:	FMAX.S RDest, RFuente1, RFuente2
	Propósito:	Encontrar el valor máximo de los registros fuente y guardarlos en RDest
	Excepciones:	

1.FCVT.W.S

31 1100000	25 00000	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = se (s32(RFuente1))
---------------	-------------	-------------------	-------------	----------------	-----------------	-----------------------------------

	Formato:	FCVT.W.S RDest, RFuente1
	Propósito:	Convertir el valor de punto flotante de RFuente1 a un entero de 32 bits con signo y guardar el resultado con signo extendido en RDest
	Excepciones:	

1.FCVT.W.U.S

31	25	24	20	19	15	14	12	11	7	6	0	
1100000	00000	RFuente1	Rm	RDest	1010011	RDest = se (u32(RFuente1))						

	Formato:	FCVT.W.U.S RDest, RFuente1
	Propósito:	Convertir el valor de punto flotante de precisión simple de RFuente1 en un entero de 32 bits sin signo y guardar el resultado con signo extendido en RDest
	Excepciones:	

1.FMV.X.W

31	25	24	20	19	15	14	12	11	7	6	0	
1110000	00000	RFuente1	000	RDest	1010011	RDest = se (RFuente1)						

	Formato:	FMV.X.W RDest, RFuente1
	Propósito:	Mover un valor de precisión simple de punto flotante del registro y guardar el resultado con signo extendido en RDest
	Excepciones:	

1. FEQ.S

31	25	24	20	19	15	14	12	11	7	6	0	
1010000	RFuente2	RFuente1	010	RDest	1010011	RDest = (RFuente1 == RFuente2)						

	Formato:	FEQ.S RDest, RFuente1, RFuente2
	Propósito:	Poner en 1 RDest si los registros fuente de punto flotante de precisión simple son iguales, y 0 si son diferentes.
	Excepciones:	

1. FLT.S

31	25	24	20	19	15	14	12	11	7	6	0	
1010000	RFuente2	RFuente1	001	RDest	1010011	RDest = RFuente1 < RFuente2						

	Formato:	FLT.S RDest, RFuente1, RFuente2
	Propósito:	Poner en 1 RDest si el valor de punto flotante de precisión de RFuente1 es menor que el de RFuente2, y 0 si no.
	Excepciones:	

1. FLE.S

31	25	24	20	19	15	14	12	11	7	6	0	
1010000	RFuente2	RFuente1	001	RDest	1010011	RDest = RFuente1 <= RFuente2						

	Formato:	FLE.S RDest, RFuente1, RFuente2
	Propósito:	Poner en 1 RDest si el valor de punto flotante de precisión de RFuente1 es menor o igual que el de RFuente2, y 0 si no.
	Excepciones:	

1.

FCLASS.S

31 1110000	25 00000	24 RFuente1	20 001	19 RDest	15 1010011	14 RDest = classify(RFuente1)	12 0	11 7	10 6	9 0	

	Formato:	FCLASS.S RDest, RFuente1, RFuente2																						
	Propósito:	Indicar en RDest la clase del número flotante de precisión simple de RFuente1 de acuerdo con la siguiente tabla:																						
	Excepciones:	<table border="1"> <thead> <tr> <th>Bit en RDest</th> <th>Significado</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>RFuente1 es $-\infty$</td> </tr> <tr> <td>1</td> <td>RFuente1 es un número negativo normal</td> </tr> <tr> <td>2</td> <td>RFuente1 es un número negativo subnormal</td> </tr> <tr> <td>3</td> <td>RFuente1 es -0</td> </tr> <tr> <td>4</td> <td>RFuente1 es +0</td> </tr> <tr> <td>5</td> <td>RFuente1 es un número positivo subnormal</td> </tr> <tr> <td>6</td> <td>RFuente1 es un número positivo normal</td> </tr> <tr> <td>7</td> <td>RFuente1 es $+\infty$</td> </tr> <tr> <td>8</td> <td>RFuente1 es un NaN señalizado</td> </tr> <tr> <td>9</td> <td>RFuente1 es un NaN simple</td> </tr> </tbody> </table>	Bit en RDest	Significado	0	RFuente1 es $-\infty$	1	RFuente1 es un número negativo normal	2	RFuente1 es un número negativo subnormal	3	RFuente1 es -0	4	RFuente1 es +0	5	RFuente1 es un número positivo subnormal	6	RFuente1 es un número positivo normal	7	RFuente1 es $+\infty$	8	RFuente1 es un NaN señalizado	9	RFuente1 es un NaN simple
Bit en RDest	Significado																							
0	RFuente1 es $-\infty$																							
1	RFuente1 es un número negativo normal																							
2	RFuente1 es un número negativo subnormal																							
3	RFuente1 es -0																							
4	RFuente1 es +0																							
5	RFuente1 es un número positivo subnormal																							
6	RFuente1 es un número positivo normal																							
7	RFuente1 es $+\infty$																							
8	RFuente1 es un NaN señalizado																							
9	RFuente1 es un NaN simple																							

1.FCVT.S.W

31 1101000	25 00000	24 RFuente1	20 Rm	19 RDest	15 1010011	14 RDest = f32(RFuente1)	12 0	11 7	10 6	9 0	

	Formato:	FCVT.S. W RDest, RFuente1, RFuente2
	Propósito:	Convertir el entero de 32 bits de complemento a dos de RFuente1 a un número de punto flotante de precisión simple y guardar el resultado en RDest
	Excepciones:	

1.FCVT.S.WU

31 1101000	25 00001	24 RFuente1	20 rm	19 RDest	15 1010011	14 RDest = f32(RFuente1)	12 0	11 7	10 6	9 0	

	Formato:	FCVT.S. WU RDest, RFuente1
	Propósito:	Convertir el entero de 32-bits sin signo de RFuente1 a un número de punto flotante de

		precisión simple y lo guarda en RDest
	Excepciones:	

1.FMV.W.X

31 1111000	25 00000	24 20 RFuente1	19 15 000	14 12 RDest	11 7 1010011	6 0 RDest = RFuente1
		Formato: FMV.W.X RDest, RFuente1				
		Propósito: Mover el valor de punto flotante de precisión simple de RFuente1 a RDest				
		Excepciones:				

RV64F STANDARD EXTENSION (in addition to RV32F)

1.FCVT.L.S

31 1100000	25 00010	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = s64(RFuente1)
		Formato: FCVT.L.S RDest, RFuente1				
		Propósito: Convertir el valor de punto flotante de precisión simple de RFuente1 a un número entero de 64 bits complemento a dos y guardarlo en RDest				
		Excepciones:				

1.FCVT.LU.S

31 1100000	25 00011	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = u64(RFuente1)
		Formato: FCVT.LU.S RDest, RFuente1,				
		Propósito: Convertir el valor de punto flotante de precisión simple de RFuente1 a un número entero de 64 bits sin signo y guardarlo en RDest				
		Excepciones:				

1.FCVT.S.L

31	25	24	20	19	15	14	12	11	7	6	0	
1101000		00010		RFuente1		Rm		RDest		1010011		RDest = f32(RFuente1)

	Formato:	FCVT.S.L RDest, RFuente1
	Propósito:	Convertir el valor entero de 64 bits de complemento a dos de RFuente1 en un número flotante de precisión simple y lo guarda en RDest
	Excepciones:	

1. FCVT.S.LU

31	25	24	20	19	15	14	12	11	7	6	0	
1101000		00011		RFuente1		Rm		RDest		1010011		RDest = f32(RFuente1)

	Formato:	FCVT.S.LU RDest, RFuente1, RFuente2
	Propósito:	Convertir el valor entero de 64-bits sin signo de RFuente1 en un número flotante de precisión simple y lo guarda en RDest
	Excepciones:	

RV32D STANDARD EXTENSION

Tipo S

1. FSD

31	25	24	20	19	15	14	12	11	7	6	0	
Imm[11:5]		RFuente2		RFuente1		011		Imm[4:0]		0100111		Dirección de Memoria = RFuente1 + SE(Imm)

	Formato:	FSD RFuente2, Inmediato (RFuente1)
	Propósito:	Almacenar el valor de punto flotante de precisión simple de RFuente2 a la Dirección de Memoria
	Excepciones:	

Tipo R-4

1. FMADD.D

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	01	RFuente2	RFuente1	Rm	RDest	1000011		$RDest = RFuente1 * RFuente2 + RFuente3$

	Formato:	FMADD.D RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplicar los valores de punto flotante de precisión doble de RFuente1 por RFuente2 después suma RFuente3 al resultado sin redondear de punto flotante de precisión doble y guarda el resultado redondeado de precisión doble en RDest
	Excepciones:	

1. FMSUB.D

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	01	RFuente2	RFuente1	Rm	RDest	1000111		$RDest = RFuente1 * RFuente2 - RFuente3$

	Formato:	FMSUB.D RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplicar los valores de punto flotante de precisión doble de RFuente1 por RFuente2 después resta RFuente3 al resultado sin redondear de punto flotante de precisión doble y guarda el resultado redondeado de precisión doble en RDest
	Excepciones:	

1. FNMSUB.D

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	01	RFuente2	RFuente1	Rm	RDest	1001011		$RDest = -RFuente1 * RFuente2 + RFuente3$

	Formato:	FNMSUB.D RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplicar los valores de punto flotante de precisión doble de RFuente1 por RFuente2, niega el resultado, después suma RFuente3 al resultado sin redondear de punto flotante de precisión doble y guarda el resultado redondeado de precisión doble en RDest
	Excepciones:	

1. FNMADD.D

31	27	26 25	24 20	19 15	14 12	11 7	6 0	
RFuente3	01	RFuente2	RFuente1	Rm	RDest	1001111		$RDest = -RFuente1 * RFuente2 - RFuente3$

	Formato:	FNMADD.D RDest, RFuente1, RFuente2, RFuente3
	Propósito:	Multiplicar los valores de punto flotante de precisión doble de RFuente1 por RFuente2, niega el resultado, después resta RFuente3 al resultado sin redondear de punto flotante de precisión doble y guarda el resultado redondeado de precisión doble en RDest
	Excepciones:	

Tipo R

1. FADD.D

31	25	24	20	19	15	14	12	11	7	6	0	
0000001	RFuente2		RFuente1		Rm		RDest		1010011		RDest = RFuente1 + RFuente2	

	Formato:	FADD RDest, RFuente1, RFuente2
	Propósito:	Sumar los valores de punto flotante de precisión doble de RFuente1 con RFuente2 y guardar la suma redondeada de precisión doble en RDest
	Excepciones:	

1. FSUB.D

31	25	24	20	19	15	14	12	11	7	6	0	
0000101	RFuente2		RFuente1		Rm		RDest		1010011		RDest = RFuente1 - RFuente2	

	Formato:	FSUB.D RDest, RFuente1, RFuente2
	Propósito:	Restar los valores de punto flotante de precisión doble de RFuente1 con RFuente2 y guardar la diferencia de precisión doble en RDest
	Excepciones:	

1. FMUL.D

31	25	24	20	19	15	14	12	11	7	6	0	
0001001	RFuente2		RFuente1		Rm		RDest		1010011		RDest = RFuente1 * RFuente2	

	Formato:	FMUL.D RDest, RFuente1, RFuente2
	Propósito:	Multiplicar los valores de punto flotante de precisión doble de RFuente1 con RFuente2 y guardar el producto redondeado de precisión doble en RDest
	Excepciones:	

1. FDIV.D

31 0001101	25 RFuente2	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = RFuente1 / RFuente2
---------------	----------------	-------------------	-------------	----------------	-----------------	---------------------------------------

	Formato:	FDIV.D RDest, RFuente1, RFuente2
	Propósito:	Dividir los valores de punto flotante de precisión doble de RFuente1 entre RFuente2 y guardar el cociente redondeado de precisión doble en RDest
	Excepciones:	

1. FSQRT.D

31 0101101	25 00000	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = sqrt(RFuente1)
---------------	-------------	-------------------	-------------	----------------	-----------------	-------------------------------

	Formato:	FSQRT.D RDest, RFuente1, RFuente2
	Propósito:	Calcular la raíz cuadrada del valor de punto flotante de precisión doble de RFuente1 y guardar el resultado redondeado de precisión doble en RDest
	Excepciones:	Si ocurre sobre flujo en la operación RDest no se modifica y se dispara una excepción, si no ocurre excepción de sobreflujo el resultado se escribe en RDest.

1. FSGNJ.D

31 0010001	25 RFuente2	24 20 RFuente1	19 15 000	14 12 RDest	11 7 1010011	6 0 RDest = {RFuente2[63], RFuente1[62:0]}
---------------	----------------	-------------------	--------------	----------------	-----------------	--

	Formato:	FSGNJ.D RDest, RFuente1, RFuente2
	Propósito:	Construir un nuevo número de punto flotante de precisión doble con el valor de RFuente1, pero con el signo de RFuente2 y guardarlo en RDest
	Excepciones:	

1.FSGNJD.N.D

31 0010001	25 RFuente2	24 20 RFuente1	19 15 001	14 12 RDest	11 7 1010011	6 0 RDest = {~RFuente2[63], RFuente1[62:0]}
---------------	----------------	-------------------	--------------	----------------	-----------------	---

	Formato:	FSGNJN.D RDest, RFuente1, RFuente2
	Propósito:	Construir un nuevo número de punto flotante de precisión doble con el valor de RFuente1, pero con el signo opuesto de RFuente2 y guardarlo en RDest
	Excepciones:	

1. FSGNJX.D

31 0010001	25 RFuente2	24 20 19 15 RFuente1	14 12 010	11 7 RDest	6 0 1010011	RDest = {RFuente1[63] ^ RFuente2[63], RFuente1[62:0]}
---------------	----------------	----------------------------	--------------	---------------	----------------	---

	Formato:	FSGNJX.D RDest, RFuente1, RFuente2
	Propósito:	Construir un nuevo número de punto flotante de precisión doble con el valor de RFuente1, pero con el signo de la operación XOR de RFuente1 con RFuente2 y guardarlo en RDest
	Excepciones:	

1. FMIN.D

31 0010101	25 RFuente2	24 20 19 15 RFuente1	14 12 000	11 7 RDest	6 0 1010011	RDest = MIN (RFuente1, RFuente2)
---------------	----------------	----------------------------	--------------	---------------	----------------	----------------------------------

	Formato:	FMIN.D RDest, RFuente1, RFuente2
	Propósito:	Copiar el número menor de punto flotante de precisión doble de RFuente1 y RFuente2, y guardararlo en RDest.
	Excepciones:	

1. FMAX.D

31 0010101	25 RFuente2	24 20 19 15 RFuente1	14 12 001	11 7 RDest	6 0 1010011	RDest = MAX (RFuente1, RFuente2)
---------------	----------------	----------------------------	--------------	---------------	----------------	----------------------------------

	Formato:	FMAX.D RDest, RFuente1, RFuente2
	Propósito:	Copiar el número mayor de punto flotante de precisión doble de RFuente1 y RFuente2, y guardarlo en RDest.
	Excepciones:	

1.FCVT.S.D

31 0100000	25 00001	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = f32 (RFuente1)
---------------	-------------	-------------------	-------------	----------------	-----------------	-------------------------------

	Formato:	FCVT.S.D RDest, RFuente1, RFuente2
	Propósito:	Convertir el número de punto flotante de precisión doble de RFuente1 en un número de punto flotante de precisión simple y guardarla en RDest.
	Excepciones:	

1.FCVT.D.S

31 0100001	25 00000	24 20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = f64 (RFuente1)
---------------	-------------	-------------------	-------------	----------------	-----------------	-------------------------------

	Formato:	FCVT.D.S RDest, RFuente1
	Propósito:	Convertir el número de punto flotante de precisión simple de RFuente1 en un número de punto flotante de precisión doble y guardarla en RDest
	Excepciones:	

1. FEQ.D

31 1010001	25 RFuente2	24 20 RFuente1	19 15 010	14 12 RDest	11 7 1010011	6 0 RDest = (RFuente1 == RFuente2)
---------------	----------------	-------------------	--------------	----------------	-----------------	---------------------------------------

	Formato:	ADD RDest, RFuente1, RFuente2
	Propósito:	Poner en 1 RDest si el número de punto flotante de precisión doble de RFuente1 es igual al número de RFuente2, y 0 si son diferentes.
	Excepciones:	

1. FLT.D

31	25	24	20	19	15	14	12	11	7	6	0	
1010001	RFuente2	RFuente1	001	RDest	1010011							$RDest = (RFuente1 < RFuente2)$

	Formato:	FLT.D RDest, RFuente1, RFuente2
	Propósito:	Poner en 1 RDest si el número de punto flotante de precisión doble de RFuente1 es menor que el número de RFuente2, y 0 si son diferentes.
	Excepciones:	

1. FLE.D

31	25	24	20	19	15	14	12	11	7	6	0	
1010001	RFuente2	RFuente1	000	RDest	1010011							$RDest = (RFuente1 \leq RFuente2)$

	Formato:	ADD RDest, RFuente1, RFuente2
	Propósito:	Poner en 1 RDest si el número de punto flotante de precisión doble de RFuente1 es menor o igual que el número de RFuente2, y 0 si son diferentes.
	Excepciones:	

1. FCLASS.D

31	25	24	20	19	15	14	12	11	7	6	0	
1110000	00000	RFuente1	001	RDest	1010011							$RDest = classify(RFuente1)$

	Formato:	FCLASS.D RDest, RFuente1																						
	Propósito:	Indicar en RDest la clase del número flotante de precisión doble de RFuente1 de acuerdo con la siguiente tabla:																						
		<table border="1"> <thead> <tr> <th>Bit en RDest</th><th>Significado</th></tr> </thead> <tbody> <tr><td>0</td><td>RFuente1 es $-\infty$</td></tr> <tr><td>1</td><td>RFuente1 es un número negativo normal</td></tr> <tr><td>2</td><td>RFuente1 es un número negativo subnormal</td></tr> <tr><td>3</td><td>RFuente1 es -0</td></tr> <tr><td>4</td><td>RFuente1 es +0</td></tr> <tr><td>5</td><td>RFuente1 es un número positivo subnormal</td></tr> <tr><td>6</td><td>RFuente1 es un número positivo normal</td></tr> <tr><td>7</td><td>RFuente1 es $+\infty$</td></tr> <tr><td>8</td><td>RFuente1 es un NaN señalizado</td></tr> <tr><td>9</td><td>RFuente1 es un NaN simple</td></tr> </tbody> </table>	Bit en RDest	Significado	0	RFuente1 es $-\infty$	1	RFuente1 es un número negativo normal	2	RFuente1 es un número negativo subnormal	3	RFuente1 es -0	4	RFuente1 es +0	5	RFuente1 es un número positivo subnormal	6	RFuente1 es un número positivo normal	7	RFuente1 es $+\infty$	8	RFuente1 es un NaN señalizado	9	RFuente1 es un NaN simple
Bit en RDest	Significado																							
0	RFuente1 es $-\infty$																							
1	RFuente1 es un número negativo normal																							
2	RFuente1 es un número negativo subnormal																							
3	RFuente1 es -0																							
4	RFuente1 es +0																							
5	RFuente1 es un número positivo subnormal																							
6	RFuente1 es un número positivo normal																							
7	RFuente1 es $+\infty$																							
8	RFuente1 es un NaN señalizado																							
9	RFuente1 es un NaN simple																							
	Excepciones:																							

1.FCVT.W.D

31	25	24	20	19	15	14	12	11	7	6	0	
1100001	00000	RFuente1	Rm	RDest	1010011							$RDest = se(s32(RFuente1))$

	Formato:	FCVT.W.D RDest, RFuente1
	Propósito:	Convertir el número de punto flotante de precisión doble de RFuente1 en un entero de 32bits complemento a dos y guardar el resultado con signo extendido en RDest
	Excepciones:	

1.FCVT.WU.D

31 1101001	25 00001	20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest =se(u32(RFuente1))
---------------	-------------	----------------	----------------	-------------------	--------------------	---------------------------------------

	Formato:	FCVT.WU.D RDest, RFuente1
	Propósito:	Convertir el número de punto flotante de precisión doble de RFuente1 en un entero de 32-bits sin signo y guardar el resultado con signo extendido en RDest
	Excepciones:	

1.FCVT.D.W

31 1101001	25 00000	20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = f64(RFuente1)
---------------	-------------	----------------	----------------	-------------------	--------------------	---------------------------------

	Formato:	FCVT.D.W RDest, RFuente1
	Propósito:	Convertir el entero de 32-bits de complemento a dos de RFuente1 en un número de punto flotante de precisión doble y guardarlo en RDest
	Excepciones:	

1.FCVT.D.WU

31 1101001	25 00001	20 RFuente1	19 15 Rm	14 12 RDest	11 7 1010011	6 0 RDest = f64(RFuente1)
---------------	-------------	----------------	----------------	-------------------	--------------------	---------------------------------

	Formato:	FCVT.D.WU RDest, RFuente1
	Propósito:	Convertir el entero de 32-bits sin signo de RFuente1 en un número de punto flotante de precisión doble y guardarlo en RDest
	Excepciones:	

RV64D STANDARD EXTENSION (in addition to RV32D)

1.FCVT.L.D

31 25	24 20	19 15	14 12	11 7	6 0	RDest = s64(RFuente1)
1100001	00010	RFuente1	Rm	RDest	1010011	

	Formato:	FCVT.L.D RDest, RFuente1
	Propósito:	Convertir el número de punto flotante de precisión doble en el registro RFuente1 en un entero de 64-bits de complemento a dos y guardarlo en RDest
	Excepciones:	

1.FCVT.LU.D

31 25	24 20	19 15	14 12	11 7	6 0	RDest = u64(RFuente1)
1100001	00011	RFuente1	Rm	RDest	1010011	

	Formato:	FCVT.LU.D RDest, RFuente1
	Propósito:	Convertir el número de punto flotante de precisión doble en el registro RFuente1 en un entero de 64-bits sin signo y guardarlo en RDest
	Excepciones:	

1.FMV.X.D

31 25	24 20	19 15	14 12	11 7	6 0	RDest = RFuente1
1110001	00000	RFuente1	000	RDest	1010011	

	Formato:	FMV.X.D RDest, RFuente1
	Propósito:	Copiar el número de punto flotante de precisión doble de RFuente1 a RDest
	Excepciones:	

1.FCVT.D.L

31 25	24 20	19 15	14 12	11 7	6 0	RDest = f64(RFuente1)
1101001	00010	RFuente1	Rm	RDest	1010011	

	Formato:	FCVT.D.L RDest, RFuente1
	Propósito:	

	Propósito:	Convertir el entero de 64-bits de complemento a dos de RFuente1 en un número de punto flotante de precisión doble y guardarlo en RDest
	Excepciones:	

1.FCVT.D.LU

31 25	24 20	19 15	14 12	11 7	6 0	
1101001	00011	RFuente1	Rm	RDest	1010011	$RDest = f64(RFuente1)$

	Formato:	FCVT.D.LU RDest, RFuente1
	Propósito:	Convertir el entero de 64-bits sin signo de RFuente1 en un número de punto flotante de precisión doble y guardarlo en RDest
	Excepciones:	

1.FMV.D.X

31 25	24 20	19 15	14 12	11 7	6 0	
1101001	00000	RFuente1	Rm	RDest	1010011	$RDest = RFuente1$

	Formato:	FMV.D.X RDest, RFuente1
	Propósito:	Copiar el número de punto flotante de precisión doble de RFuente1 a RDest
	Excepciones:	