RFC ls002.fmi v2 Floating-Point Load-Immediate </>

- Funded by NLnet under the Privacy and Enhanced Trust Programme, EU Horizon2020 Grant 825310, and NGI0 Entrust No 101069594

- https://libre-soc.org/openpower/sv/int_fp_mv/#fmvis
- https://libre-soc.org/openpower/sv/rfc/ls002.fmi/
- https://bugs.libre-soc.org/show_bug.cgi?id=1092
- https://git.openpower.foundation/isa/PowerISA/issues/87

Severity: Major

Status: New

Date: 05 Oct 2022 v3 TODO

Target: v3.2B

Source: v3.0B

Books and Section affected:

```
Book I Scalar Floating-Point 4.6.2.1
Appendix E Power ISA sorted by opcode
Appendix F Power ISA sorted by version
Appendix G Power ISA sorted by Compliancy Subset
Appendix H Power ISA sorted by mnemonic
```

Summary

```
Instructions added
fmvis - Floating-Point Move Immediate, Shifted
fishmv - Floating-Point Immediate, Second-half Move
```

Submitter: Luke Leighton (Libre-SOC)

Requester: Libre-SOC

Impact on processor:

```
Addition of two new FPR-based instructions
```

Impact on software:

```
Requires support for new instructions in assembler, debuggers,
and related tools.
```

Keywords:

```
FPR, Floating-point, Load-immediate, BF16, bfloat16, BFP32
```

Motivation

Similar to `lxvkq` but extended to a bfloat16 with one 32-bit instruction and a full FP32 in two 32-bit instructions these instructions always save a Data Load and associated L1 and TLB lookup. Even quickly clearing an FPR to zero presently needs Load.

Notes and Observations:

1. There is no need for an Rc=1 variant because this is an immediate loading instruction (an FPR equivalent to `li`)
2. There is no need for Special Registers (FP Flags) because this is an immediate loading instruction. No FPR Load Operations alter `FPSCR`, neither does `lxvkq`, and on that basis neither should these instructions.
3. `fishmv` as a FRT-only Read-Modify-Write (instead of an unnecessary FRT,FRA pair) saves five potential bits, making the difference between a 5-bit XO (VA/DX-Form) and requiring an entire Primary Opcode.

Changes

Add the following entries to:

- the Appendices of Book I
- Instructions of Book I as a new Section 4.6.2.1
- DX-Form of Book I Section 1.6.1.6 and 1.6.2
- Floating-Point Data a Format of Book I Section 4.3.1

Floating-Point Move Immediate </>

`fmvis FRT, D`

0-5	6-10	11-15	16-25	26-30	31	Form
Major	FRT	d1	d0	XO	d2	DX-Form

Pseudocode:

```
bf16 <- d0 || d1 || d2  # create bfloat16 immediate
bfp32 <- bf16 || [0]*16 # convert bfloat16 to BFP32
FRT <- DOUBLE(bfp32)    # convert BFP32 to BFP64
```

Special registers altered:

`None`

The value `D << 16` is interpreted as a 32-bit float, converted to a 64-bit float and written to `FRT`. This is equivalent to reinterpreting `D` as a `bfloat16` and converting to 64-bit float.

Examples:

```
fmvis f4, 0 # writes +0.0 to f4 (clears an FPR)
fmvis f4, 0x8000 # writes -0.0 to f4
fmvis f4, 0x3F80 # writes +1.0 to f4
fmvis f4, 0xBFC0 # writes -1.5 to f4
fmvis f4, 0x7FC0 # writes +qNaN to f4
fmvis f4, 0x7F80 # writes +Infinity to f4
fmvis f4, 0xFF80 # writes -Infinity to f4
fmvis f4, 0x3FFF # writes +1.9921875 to f4
```

Floating-Point Immediate Second-Half Move </>

`fishmv FRT, D`

DX-Form:

0-5	6-10	11-15	16-25	26-30	31	Form
Major	FRT	d1	d0	XO	d2	DX-Form

Pseudocode:

```
n <- (FRT)                      # read FRT
bfp32 <- SINGLE(n)              # convert to BFP32
bfp32[16:31] <- d0 || d1 || d2  # replace LSB half
FRT <- DOUBLE(bfp32)            # convert back to BFP64
```

Special registers altered:

`None`

An additional 16-bits of immediate is inserted into the low-order half of the single-format value corresponding to the contents of FRT.

This instruction performs a Read-Modify-Write on FRT. In hardware, `fishmv` may be macro-op-fused with `fmvis`.

Programmer's note: The use of these two instructions is strategically similar to how `li` combined with `oris` may be used to construct 32-bit Integers. If a prior `fmvis` instruction had been used to set the upper 16-bits from a BFP32 value, `fishmv` may be used to set the lower 16-bits. Example:

```
# these two combined instructions write 0x3f808000
# into f4 as a BFP32 to be converted to a BFP64.
# actual contents in f4 after conversion: 0x3ff0_1000_0000_0000
# first the upper bits, happens to be +1.0
fmvis f4, 0x3F80 # writes +1.0 to f4
# now write the lower 16 bits of a BFP32
fishmv f4, 0x8000 # writes +1.00390625 to f4
```

[[!tag opf_rfc]]

DX-Form </>

Add the following to Book I, 1.6.1.6, DX-Form

```
  |0    |6   |11   |16   |26   |31
  | PO  | FRT|   d1|   d0|   XO|d2
```

Add DX to FRT Field in Book I, 1.6.2

```
 FRT (6:10)
     Field used to specify an FPR to be used as a
     source.
     Formats: D, X, DX
```

bfloat16 definition </>

Add the following to Book I, 4.3.1:

The format may be a 16-bit bfloat16, 32-bit single format for a single-precision value...

The bfloat16 format is used as an immediate.

The structure of the bfloat16, single and double formats is shown below.

```
  |S |EXP| FRACTION|
  |0 |1 8|9      15|
```

Figure #. Binary floating-point half-precision format (bfloat16)

Appendices </>

```
Appendix E Power ISA sorted by opcode
Appendix F Power ISA sorted by version
Appendix G Power ISA sorted by Compliancy Subset
Appendix H Power ISA sorted by mnemonic
```

Form	Book	Page	Version	mnemonic	Description
DX	I	#	3.0B	fmvis	Floating-point Move Immediate, Shifted
DX	I	#	3.0B	fishmv	Floating-point Immediate, Second-half Move