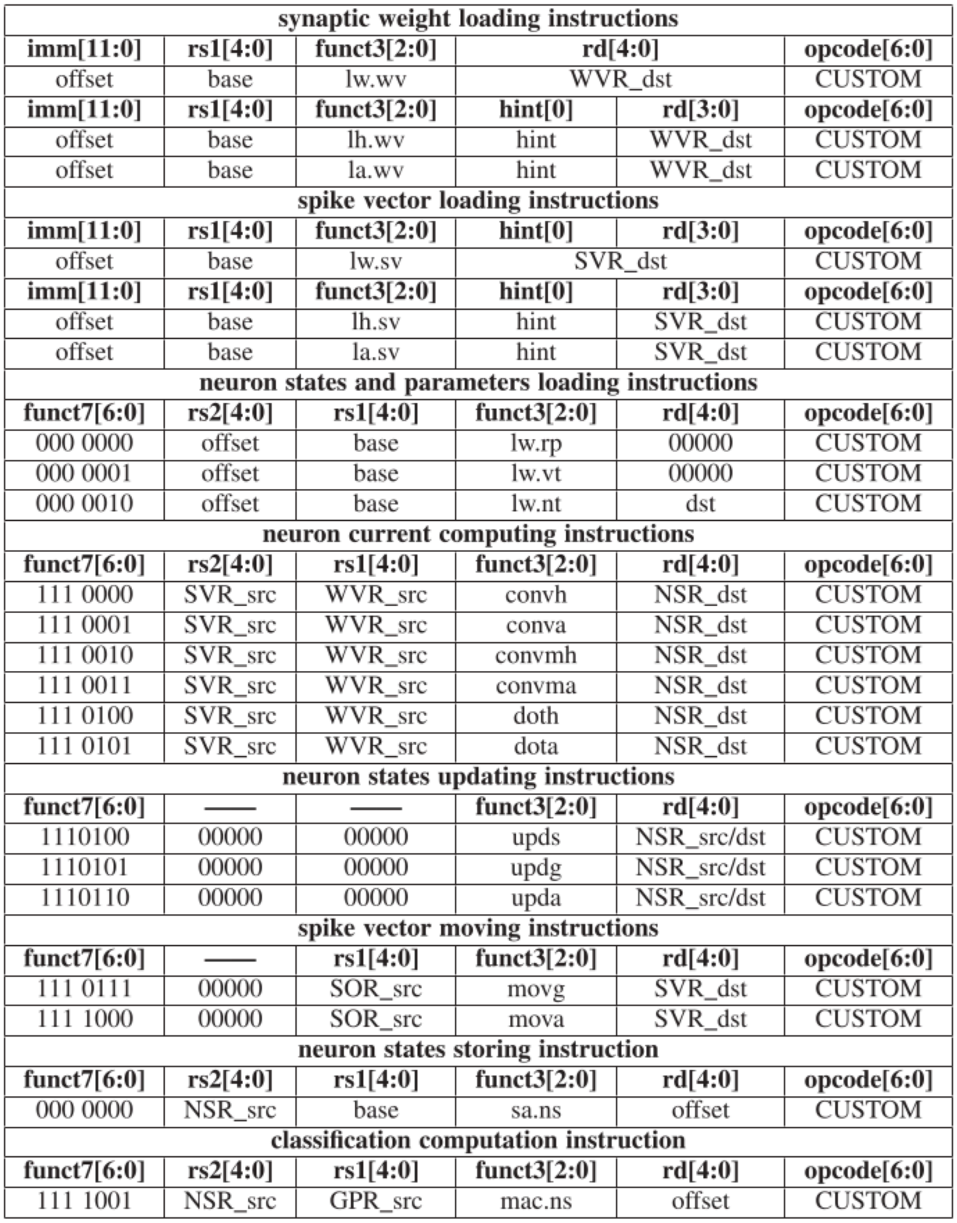
RVNE - SNN Core Mapping

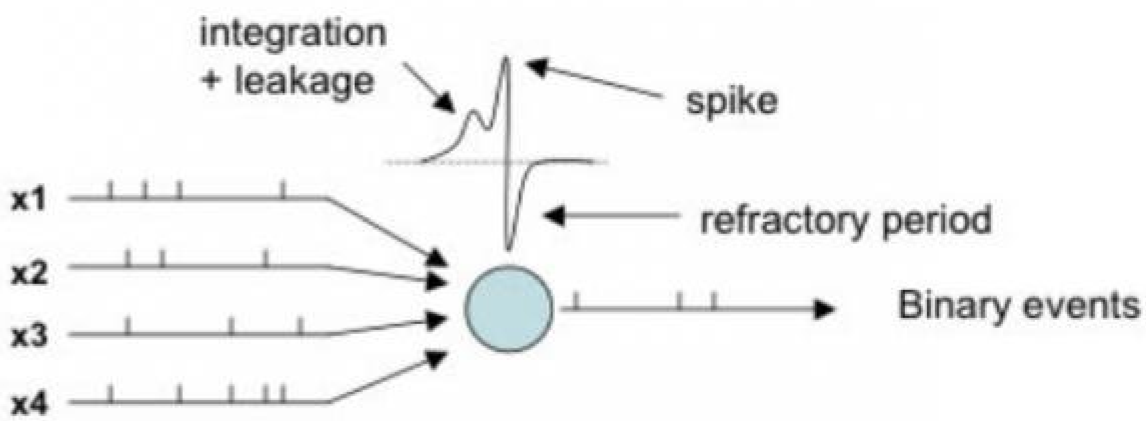
Introduction

The following operations mapping is done assuming a 32 neuron-layer and for a given timestamp and neuron. The RVNE ISA and sample code from the paper was used to learn about the ISA and its workings.



synaptic weight loading instructions					
imm[11:0]	**rs1[4:0]**	**funct3[2:0]**	**rd[4:0]**		**opcode[6:0]**
offset	base	lw.wv	WVR_dst		CUSTOM
imm[11:0]	**rs1[4:0]**	**funct3[2:0]**	**hint[0]**	**rd[3:0]**	**opcode[6:0]**
offset	base	lh.wv	hint	WVR_dst	CUSTOM
offset	base	la.wv	hint	WVR_dst	CUSTOM
spike vector loading instructions					
imm[11:0]	**rs1[4:0]**	**funct3[2:0]**	**hint[0]**	**rd[3:0]**	**opcode[6:0]**
offset	base	lw.sv	SVR_dst		CUSTOM
imm[11:0]	**rs1[4:0]**	**funct3[2:0]**	**hint[0]**	**rd[3:0]**	**opcode[6:0]**
offset	base	lh.sv	hint	SVR_dst	CUSTOM
offset	base	la.sv	hint	SVR_dst	CUSTOM
neuron states and parameters loading instructions					
funct7[6:0]	**rs2[4:0]**	**rs1[4:0]**	**funct3[2:0]**	**rd[4:0]**	**opcode[6:0]**
000 0000	offset	base	lw.rp	00000	CUSTOM
000 0001	offset	base	lw.vt	00000	CUSTOM
000 0010	offset	base	lw.nt	dst	CUSTOM
neuron current computing instructions					
funct7[6:0]	**rs2[4:0]**	**rs1[4:0]**	**funct3[2:0]**	**rd[4:0]**	**opcode[6:0]**
111 0000	SVR_src	WVR_src	convh	NSR_dst	CUSTOM
111 0001	SVR_src	WVR_src	conva	NSR_dst	CUSTOM
111 0010	SVR_src	WVR_src	convmh	NSR_dst	CUSTOM
111 0011	SVR_src	WVR_src	convma	NSR_dst	CUSTOM
111 0100	SVR_src	WVR_src	doth	NSR_dst	CUSTOM
111 0101	SVR_src	WVR_src	dota	NSR_dst	CUSTOM
neuron states updating instructions					
funct7[6:0]	**——**	**——**	**funct3[2:0]**	**rd[4:0]**	**opcode[6:0]**
1110100	00000	00000	upds	NSR_src/dst	CUSTOM
1110101	00000	00000	updg	NSR_src/dst	CUSTOM
1110110	00000	00000	upda	NSR_src/dst	CUSTOM
spike vector moving instructions					
funct7[6:0]	**——**	**rs1[4:0]**	**funct3[2:0]**	**rd[4:0]**	**opcode[6:0]**
111 0111	00000	SOR_src	movg	SVR_dst	CUSTOM
111 1000	00000	SOR_src	mova	SVR_dst	CUSTOM
neuron states storing instruction					
funct7[6:0]	**rs2[4:0]**	**rs1[4:0]**	**funct3[2:0]**	**rd[4:0]**	**opcode[6:0]**
000 0000	NSR_src	base	sa.ns	offset	CUSTOM
classification computation instruction					
funct7[6:0]	**rs2[4:0]**	**rs1[4:0]**	**funct3[2:0]**	**rd[4:0]**	**opcode[6:0]**
111 1001	NSR_src	GPR_src	mac.ns	offset	CUSTOM

The following diagram shows illustrates all important actions required for a SNN:



Loading in Weights Vector and Spike Input Vector

Assuming r1 has the base address for the weight vector, and r2 has the base address for the spike input vector

```
lw.wv x0, 0(r1)
lw.sv r3, 0(r2)
```

Loading in current neuron

Assuming the Spiking Output Vector is stored at base address 0

```
addi r4, x0, 0
```

Loading current neurons rp and rt

Assuming 200 is the address where refractory period is stored, and 204 is the address where voltage threshold is stored.

lw.rp loads the refractory period into RPR (Refractory Period Register)

lw.vt loads threshold into VTR (Voltage Threshold Register)

```
lw.rp 0 (x200)
lw.vt 0 (x204)
```

Cumulating the current

Performs the following arithmetic operation and stores in r4:

$$I(t) = \sum_{n=0}^{N} S_n * W_n$$

```
conva r4, r3
```

Updating states

This includes updating states including current, voltage, spike sums, and refractory period of all neurons indirectly indexed by r4. The subtraction of leaky term from membrane potential and refractory period is taken care of by the microarchitectural implementation.

```
upda r4
```

Moving spike vector to output vector.

Although not mandatory, moving the output vector (SOR) into a SVR is application dependent.

```
mova r4, r5
```