External Signals

- reset_l
- go
- update
- correct
- done (output)
- class (output)
- sync (output)

* Overflow detection will be external

* Weights can be read arbitrarily (see datapath)

* Based on the assumption that during CPU is clocking faster

reset_l → (Init) ⟲ ~go/

Init → go/en_w0, ext_w0, sync → (w0) ⟲ ~go/

w0 → go/en_w1, ext_w1, sync → (w1) ⟲ ~go/

w1 → go/en_w2, ext_w2, sync → (w2) ⟲ ~go/

w2 → go/en_n, sync → (n) ⟲ ~go/

n → go/en_x1, sel_add_A = 00, sel_add_B = w0, en_add, sync → (x1) ⟲ ~go/

x1 → go/en_x2, sel_mult_A = w1, sel_mult_B = x1, en_mult, sync → (x2)

x2 → / sel_add_A = mult_out_reg, sel_add_B = add_out_reg, en_add, sel_mult_A = w2, sel_mult_B = x2, en_mult → (Comp I)

~update/done, sel_add_A = mult_out_reg, sel_add_B = add_out_reg, en_add → (n)

(count == 2)/done → (n)

correct == class/done → (n)

Comp I → update/ sel_add_A = mult_out_reg, sel_add_B = add_out_reg, en_add, clr_count → (Check)

(count != 2) & (correct != class)/ sel_mult_A = n, sel_mult_B = x_i, en_mult → (T1)

T1 → / sel_mult_A = mult_out_reg, sel_mult_B = d, en_mult → (T2)

T2 → /en_count, sel_add_A = mult_out_reg, sel_add_B = w_i, en_w_i → (Check)

← determined by val of count, do w0, w1, w2

Picked based off of count's value

* only update weights for incorrect predictions

* Need to wait extra cycle in training b/c class must be computed

* $x_i = 1$ for $x0$