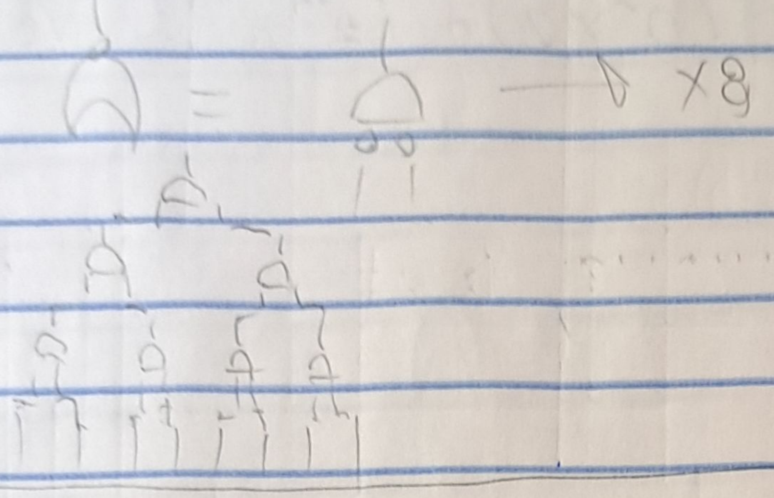
20

(scratch)



[one3] [two3]

A) 10111000 / 01000111 / 01000111 / 01001000 /

00000001

(Addr = 0-12)

(1) (1111) (1111 1111)

div op1 op2

division line which divides op1 by op2

$\boxed{4}$ $\boxed{4}$ $\boxed{4}$ $\boxed{4}$

$\boxed{4}$ $\boxed{4}$ $\boxed{4}$ $\boxed{4}$

for first and second word.

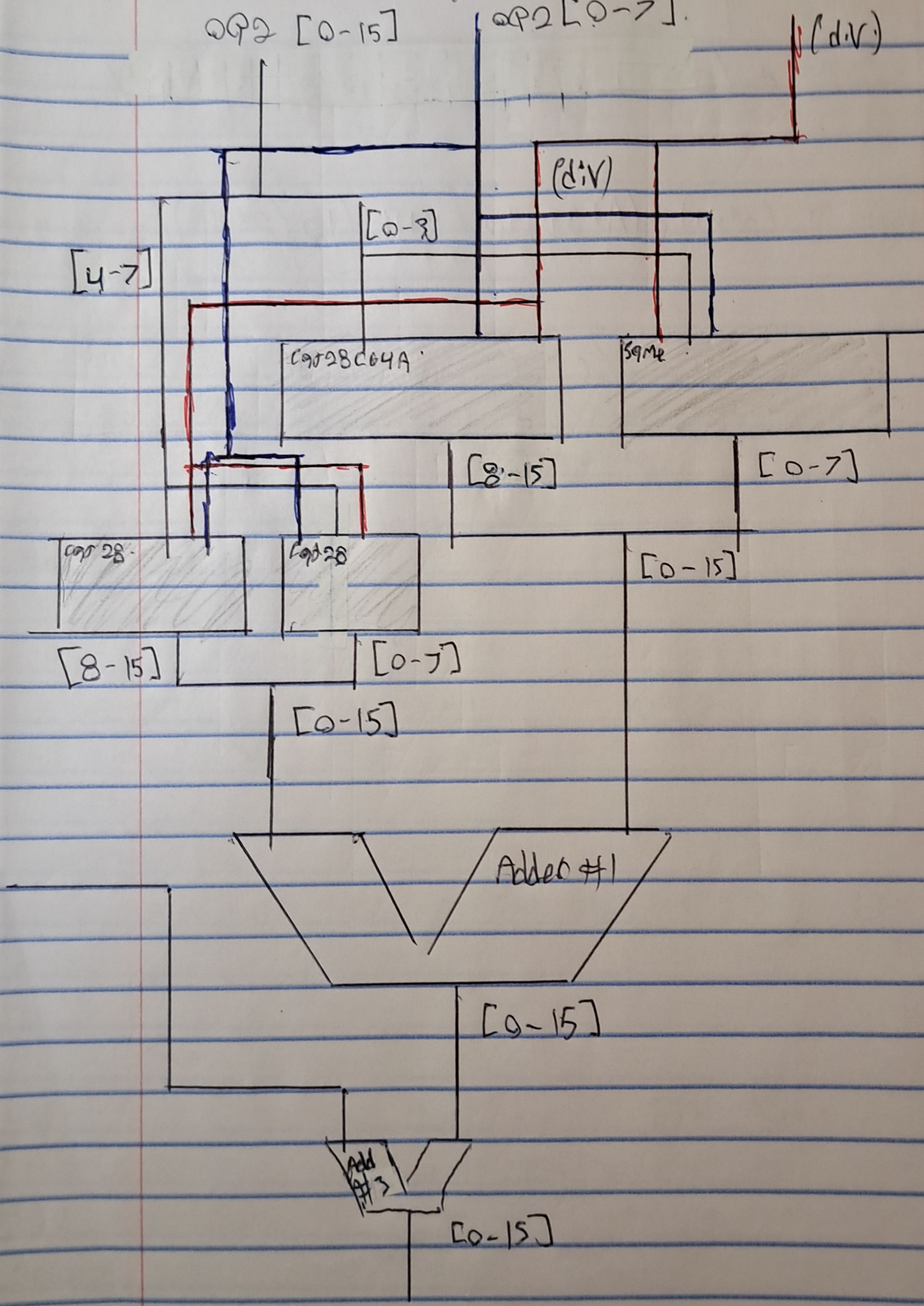
(Scloom)

A) $(2^{\emptyset} \times 1) + (2^{1} \times 1) + (2^{2} \times \emptyset) + (2^{3} \times 1) =$ Vq1 /

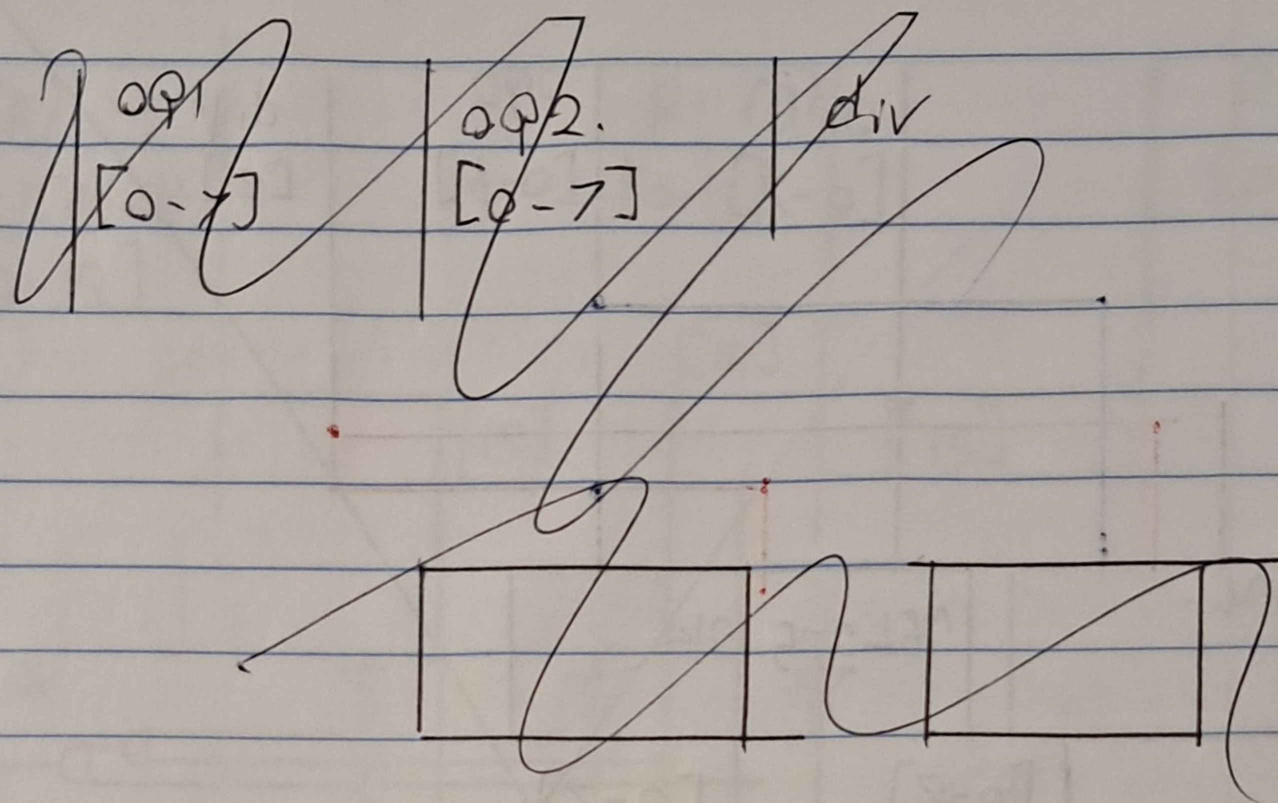
B) $(10 \times$ Vq1$) = 10(2^{\emptyset} \times 1) + ($A1BD$)$ $1\emptyset$ / B1B $= (10)(2^{\emptyset})(1)$ / $= 10$ /

C) $10 = (2^{3} \times 1) + (2^{1} \times 1)$ / C1 $= (1010)$ /

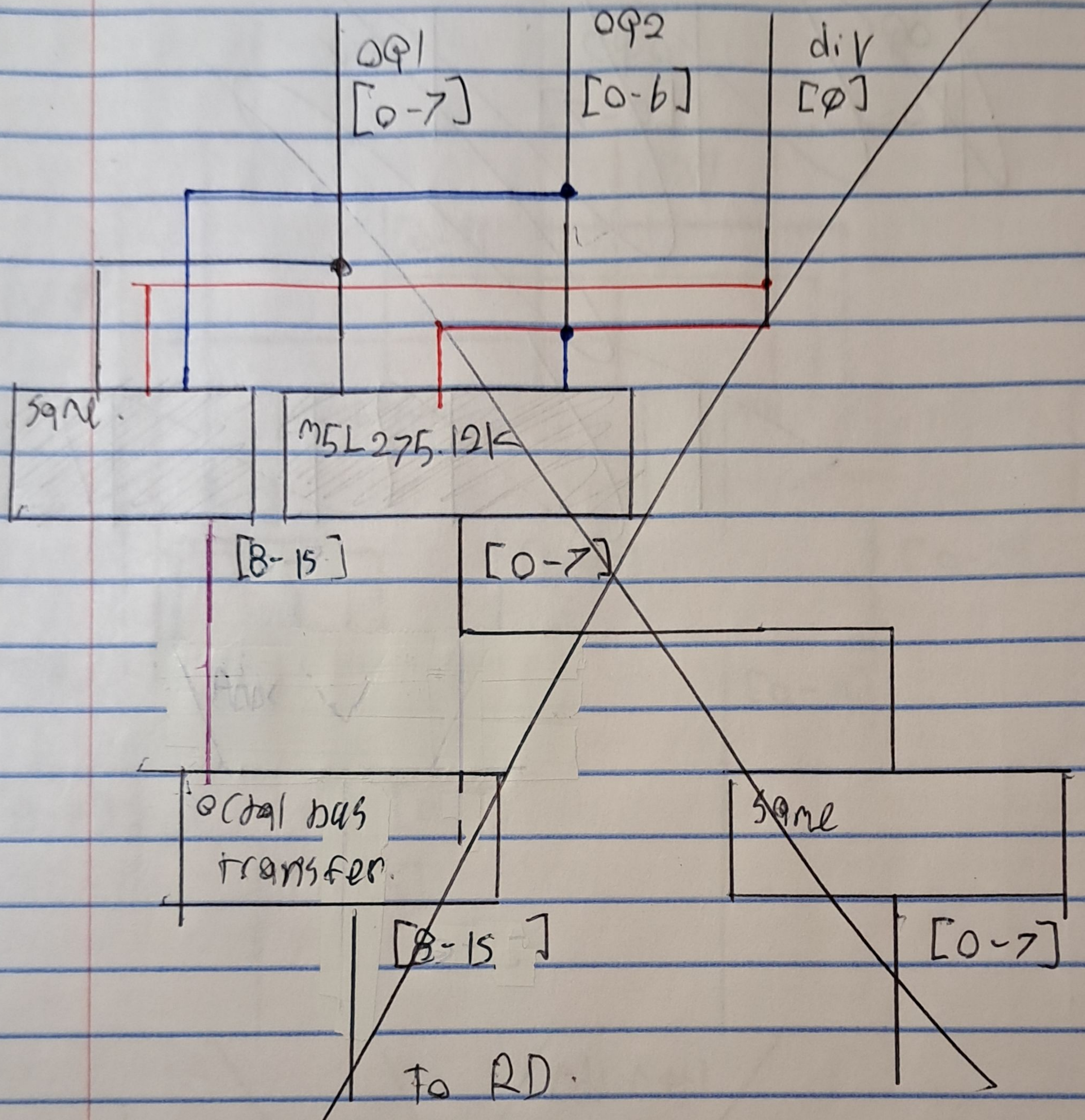
ALU multiplication module design.



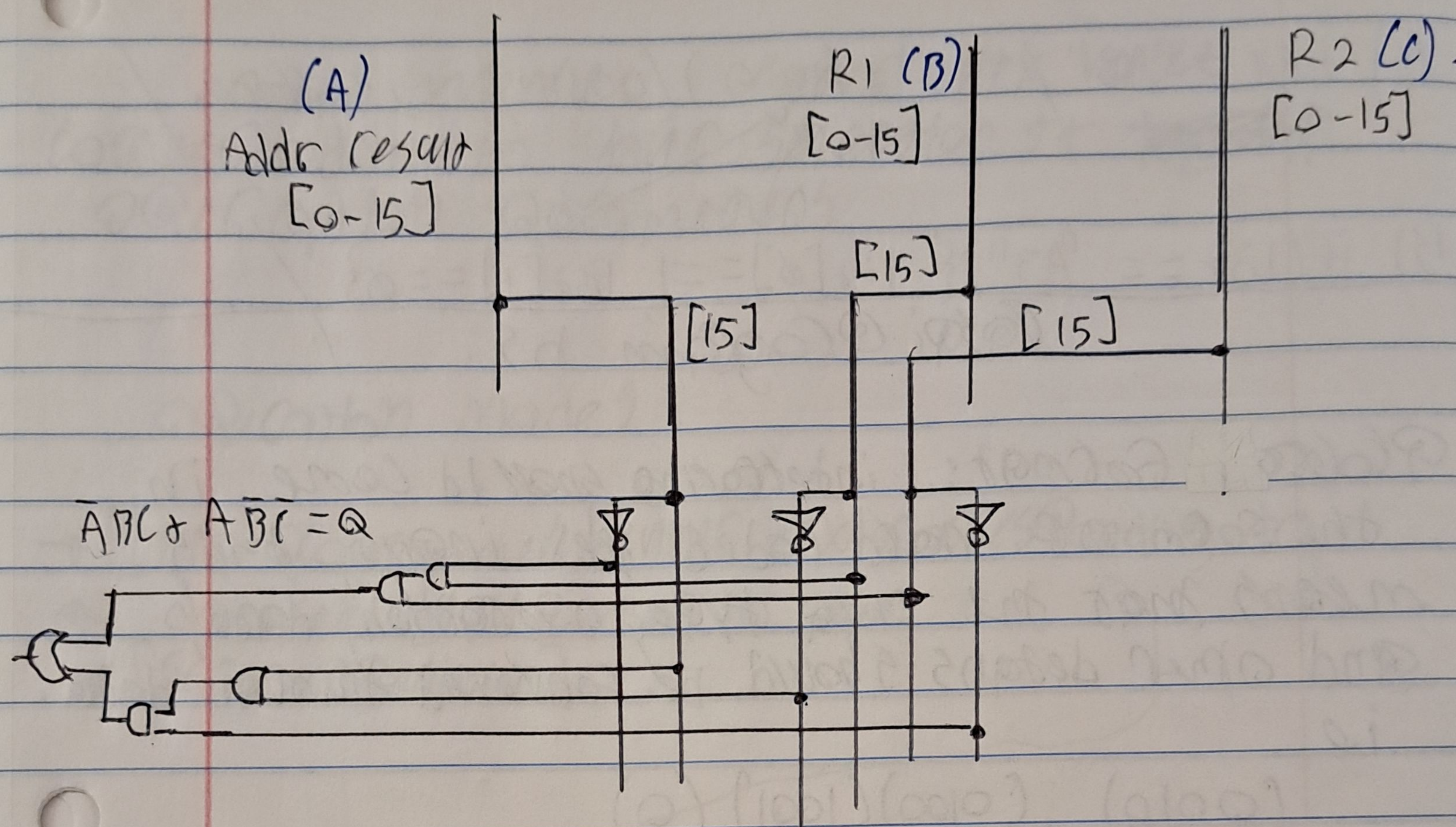
[New Alu multiplier design.]



New ALU multiplier function



overflow flag design:



Addr rslt.	R1	R2	Flag
0	0	0	Ø
0	0	1	Ø
0	1	0	Ø
0	1	1	1
1	0	0	1
1	0	1	Ø
1	1	0	Ø
1	1	1	Ø

or, A | B | C | Q |

$\bar{A}BC + A\bar{B}\bar{C} = Q$.

Raspi program SW.

A) Hex = "A3" / "A3" = (1010)(0011) / is correct. /

B) if(list == "A3"){ list[∅]==1, list[1]==∅}

Raspi program bs.

Program format: interfacing should come in the form of hard coded Hex input, this means that the chip type, operation, data, and other details should be conveyed through data.
i.e.

(0010) (0100)(1001) (0)

(what chip am I using) (data) (W,O,R)

(make sure to leave enough bits for this option.)

(We dont know How many chips there will be)

Casqi Program:

Main Menu: enter Hex value corresponding to your selection of ~~[illegible]~~ operation parameters.

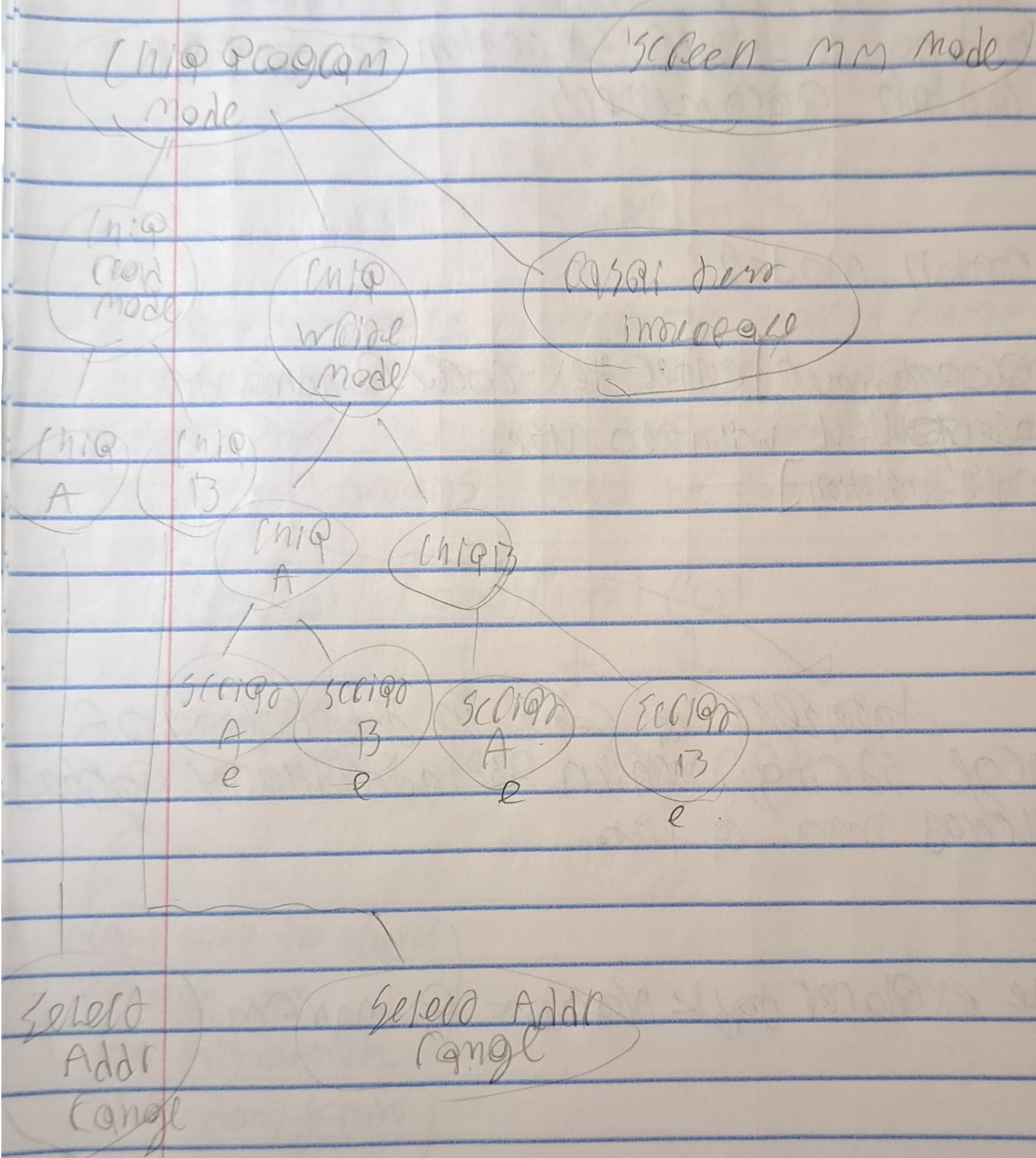
Operation modes:

→ [unit programmer] enter hex corresponding to dataset you wish to use.

→ [screen translation]

datasets come in the form of one large string, which is individually parsed and turned into a list.

> File explorer task Name = PcmanFM.


Screen MM mode
Select Addr range
Select Addr range

CarPi program todo.

- > Program needs to ~~start~~ Assign buttons and leds upon learning chip and operation. ☐
- > Add string parser function ☐
- > Add second data injector. ☑
- > Setup GitHub on Pi for easy file transfer. ☐

Things A wire can carry

- > Vcc
- > Ground (no info)
- > data (arithmetic).
- > data (instruction / operation based)
- > data Addresses
- > Control signals
- > Generic → bus lines
- >

- > Module input, > Vcc, inter-module connection
- > Module output, > Ground,
- >

- > to ALU
- > to CU
- > to interrupt module.
- > etc.

Space logistic issue brainstorm

1. remove tv from top, and put it in cabinet. Use the two layers to space part A and part B, have two bus plates with connection them, and control lines do similarly.

multiplication module brainstorm.

A) $(A+B)(C+D) = AC + AD + BC + BD \;/\; \left(\frac{x}{2} + \frac{x}{2}\right)\left(\frac{y}{2} + \frac{y}{2}\right) /$

(!)

B) $\cancel{\frac{xy}{4}} +$ / is not same value / $\frac{A}{C} + \frac{A}{D} + \frac{B}{C} + \frac{B}{D}$ /

[x] [y]

C) $= A\left(\frac{1}{C} + \frac{1}{D}\right) + B\left(\frac{1}{C} + \frac{1}{D}\right) / = (A+B)\left(\frac{1}{C} + \frac{1}{D}\right) /$

D) let $y = 15$, size = 4 bits / $\frac{1}{C} = \frac{1}{8+4} = \frac{1}{12}$ / $\frac{1}{D} = \frac{1}{2+1} = \frac{1}{3}$ /

e) $y = \frac{1}{12} + \frac{1}{3} = \frac{3}{36} + \frac{12}{36} / = \frac{15}{36} = 0.416667$ /

F) $\frac{1}{15} = 0.06667$ / $\Delta = 0.35$ / let $y^{-1} = 31$, size = 6 bits /

g) $\frac{1}{C} = \frac{1}{0+16+8} = \frac{1}{24}$ / $\frac{1}{D} = \frac{1}{4+2+1} = \frac{1}{7}$ / $\frac{1}{C} + \frac{1}{D} = 0.1845$ /

h) $\frac{1}{31} = 0.03225$ / $\Delta = 0.15226$ / $\frac{7}{168} + \frac{24}{168}$ / $\frac{31}{168} = \frac{1}{C} + \frac{1}{D}$ /

i) let $Y = 255$, size $= 8$ bits / $A = 0.0669$ /

j) $\frac{1}{255} = 0.003921$ / $\frac{1}{c} + \frac{1}{d} = 0.9708$ / $\frac{1}{c} + \frac{1}{d} = \frac{1}{240} + \frac{1}{15}$ /

k) values decrease by some factor, does the difference decrease by the same factor?

L) $\frac{31}{168}$ / $\frac{15}{36} = \frac{8+4+2+1}{32+4}$ / $\frac{31}{168} = \frac{16+8+4+2+1}{128+(32+8)}$ /

m) $\frac{15}{36} = \frac{N}{2^5+2^2}$ / $\frac{31}{168} = \frac{N}{2^7+2^5+2^3}$ /

N) maybe m13 form could still be used, if you change 1/c + 1/D to compensate for the difference and find the corresponding value which gets 1/y.

o) $J3 = \frac{255}{(240)(15)}$ / $= \frac{Y}{(C)(D)}$ / $\frac{31}{168} = \frac{Y}{(0+16+8)(4+2+1)}$ /

$[Z(Y)]$

q) with div, $(X)\left(\frac{Y}{(C)(D)}\right)$ / $= \left(X_{0-15}\right)\left(\frac{Y_{0-15}}{(C_{0-7})(D_{7-15})}\right) = \left(\frac{X}{Y}\right)$

(49)(16w)

j) $\dot{Z} = \dfrac{C+D}{(C)(D)}$ / $\left(\dfrac{1}{16}\right) = \dfrac{C+\dot{D}}{(C)(D)}$ / $D = 8$ /

k) $\dfrac{1}{16}(C)(8) = C+8$ / $\left(\dfrac{C}{2}\right) = C+8$ / $2C+18 = C$ /

L) $-18 = C$ / $\left(\dfrac{1}{8}\right)$ / $\dot{Z} = \dfrac{C+\dot{D}}{(C)(D)}$ / $\dot{z}(C)(\dot{D}) = C+\dot{D}$ /

m) $\dot{z}(\dot{D}) = 1 + \dfrac{\dot{D}}{C}$ / $\dot{z}\dot{D}(C) - (C) = \dot{D}$ / $= (C)(\dot{z}\dot{D}-1) = \dot{D}$ /

n) $C = \dfrac{\dot{D}}{(\dot{z}\dot{D}-1)}$ / = turns exponential, C becomes a fraction if D is a whole number.

o) $C = \dfrac{}{\left(\frac{1}{W}\right)D - 1}$ / $\dfrac{D}{W} - 1$ / $\dfrac{D}{W} - \dfrac{W}{W}$ / $\dfrac{D}{\frac{\dot{D}-W}{W}}$ / $\dfrac{W(\dot{D})}{\dot{D}-W} = C$ /

q) $\dfrac{(8)X}{X-(8)} = C$ / = is doable. / $X = 12$, $C = 24$

(37)

A) $Z(Y) = \dfrac{Y}{(C(Y))(D(Y))}$ / $(C)(D)(Z) = Y$ / $(C$...

B) $Z(Y)(C(Y))(D(Y)) = Y$ / $Z(C)(D) = Y$ / $Z(Y_{0-7})(Y_{7-15}) = Y(0-15)$ /

C) $(X)\left(\dfrac{Y}{(C)(D)}\right) = (A+B)\left(\dfrac{1}{C} + \dfrac{1}{D}\right) = (X)(\text{some fraction})$ /

D) $(\text{some fraction}) = Z = \dfrac{Y}{(C)(D)}$ /

e) $\dot{Z} = \dfrac{Y}{(C)(D)} = \dfrac{Y}{(Y_{0-7})(Y_{7-15})}$ / let $\dot{Z} = 50$ /

f) $50 = \dfrac{Y}{(7)(13)}$ / $Y = (50)(7)(13)$ / $Y = 4550$ /

g) $\dfrac{1}{50} = \dfrac{Y}{(7)(13)}$ / $\dfrac{(7)(13)}{50} = Y$ / $Y = 1.82$ /

H) $\dfrac{1}{16} = \dfrac{Y}{(2)(4)}$ / $\dfrac{8}{16} = Y$ / $\dfrac{1}{16} = \dfrac{Y}{(1.6)(2)}$ / $Y = 2$ /

$C + D = Y$ /

A) $y = 255$ / $= \dfrac{1}{240} + \dfrac{1}{15}$ / $= \dfrac{1}{2^7+2^6+2^5+2^4} + \dfrac{1}{2^3+2^2+2^1+2^0}$ /

B) let $y = y_{a-3}$ / $\dfrac{A+B+C+D}{(A+B)(C+D)} = \dfrac{A+B+C+D}{AC+AB+BC+BD}$ / = 4 terms.

$2^3 \quad 2^2 \quad 2^1 \quad 2^0$

C) $(A+B)(C+D+E)$ / $AC+AD+AE+BC+BD+BE$ = 6 terms /

D) $(A+B+C)(D+E+F)$ / = 9 terms $= 3^2$ / $(A+B)(C+D)$ = 4 terms $= 2^2$ /

E) $(A+B+C+D)(E+F+G+H)$ / $= 16 = 4^2$ / $(\text{div})^2$ = terms /

F) [B2] Assume value of 15, $\dfrac{2^3+2^2+2^1+2^0}{(2^3)(2^1)+(2^3)(2^2)+(2^2)(2^1)+(2^2)(2^0)}$ /

G) $= \dfrac{2^3+2^2+2^1+2^0}{2^4+2^5+2^3+2^2}$ / $\dfrac{2^3+2^2+2^1+2^0}{2^5+2^4+2^3+2^2}$ / $= \dfrac{2^{N+3}+2^{N+2}+2^{N+1}+2^N}{2^{N+5}+2^{N+4}+2^{N+3}+2^{N+2}}$ /

H) $= \dfrac{y}{(A+B)(C+D)}$ /

32

(36)

A) $\left(\frac{A}{C+D}\right) + \left(\frac{B}{C+D}\right)$ / 1st Bit value = 2 / $\frac{2^1+2^0}{2^3+2^2+2^1+2^0} + \frac{2^2+2^3}{2^3+2^2+2^1+2^0}$ /

B) $\frac{2^1}{2^3+2^2+2^1+2^0} + \frac{2^0}{2^3+2^2+2^1+2^0}$

C)

36

(35)

A_{0-8}

A) $(A+B)\left(\frac{1}{C+D}\right)$ / if $(C=D)$ { } ! $\frac{1}{2} = \frac{1}{4} + \frac{1}{4}$ / $(A+B)\left(\frac{1}{C} + \frac{1}{D}\right)$ /

B) $A_1 = \left(\frac{A}{C+D}\right) + \left(\frac{B}{C+D}\right)$ / $= \left(\frac{AA+AB}{C+D}\right) + \left(\frac{BA+BB}{C+D}\right)$

C) $= \frac{AA_{0-3}}{C_{0-7}+D_{0-7}} + \frac{AB_{0-3}}{C_{0-7}+D_{0-7}}$ + ect. /

D) ~~$0 = 2^0$ / $1 = (2^1 - 1)$ / $2 = 2^1$ / $3 = 2^1+1$ / $4 = 2^2$ / $5 = 2^2+1$ /~~

e) ~~$6 = (2 \times 3)$ / $= (2) \times (2+1) = 4+2 = 2^2+2$ /~~

last resort.

A) $(A_{0-7} + B_{7-15}) + \left(\frac{1}{Z_{0-7}}\right)$ / $= \frac{A_{0-7}}{Z_{0-7}} + \frac{B_{0-7}}{Z_{0-7}}$ /

B) could maybe do 16 bit value with two registers?
Zupper, Zlower?

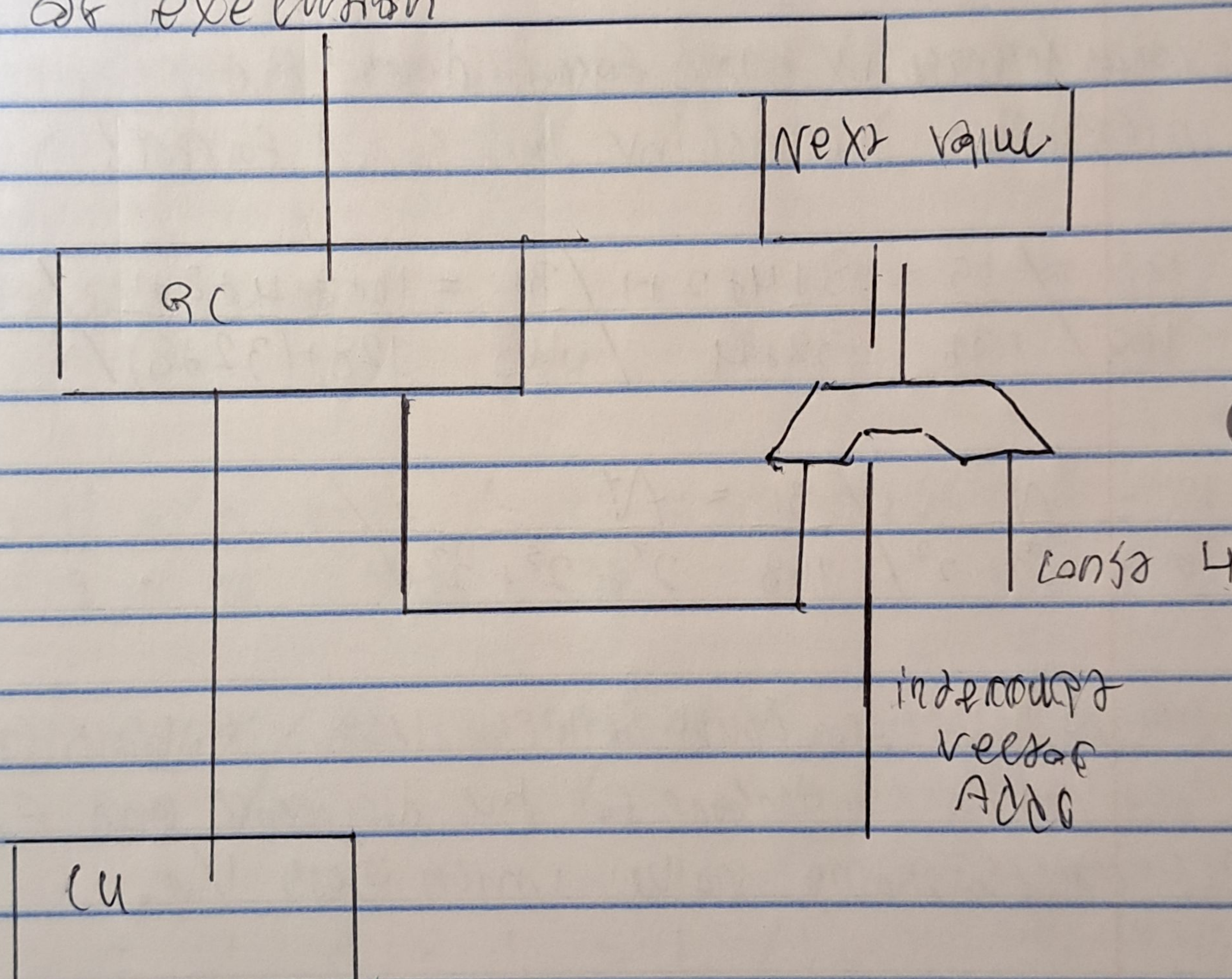
A) $\left(\frac{A}{C} + \frac{B}{C}\right) + \left(\frac{A}{D} + \frac{B}{D}\right) = A\left(\frac{1}{C} + \frac{1}{D}\right) + B\left(\frac{1}{C} + \frac{1}{D}\right)$

B) N[illegible]

C)

Possible better interrupt module design

instructions are executed in the order in which they appear to the PC, so if you have a "next PC Addr" register, which can be over written, you might be able to reduce some of the complexity involved in throwing a value in the middle of execution.



The interrupt Address would over write the next PC Addr if conditions permit, and to return back to that Next value, the next value reg could be saved somewhere.

Plummer method

Some operations occur at the speed at which chips propagate, for example the ALU in the 16 bit computer does not need to wait for an operand register to be written, as values sort of "fall" into place at the speed of the chips. I wonder if its possible to reduce the execution time of some instructions to one clock cycle if you design the entire computer to work in a similar fashion. This might be faster than pipelining as that takes multiple cycles per operation, albeit still in an optimized fashion. Of course some operations, such as writing back to a chip require that a value be present, so some sort of "value ready" signal would need to be implemented:

→ Not efficient, if stage takes up one full clock cycle, then removing the load and unload times would not result in faster speed than pipelining.

In the very least this might be used to do away with the clock, as finishing an instruction would notify the start of the next one.