RTL Design & Verification.

ICA (Independent Component Analysis):

$X = AS$ $\quad$ X - 7 mixed signal vectors

S - independent source signal vector.

$A = 7\times7$

$S = 7\times7$ $\quad$ (we are taking 7, 7D signals)

$X = 7\times7$

$X = \{x_{i,j}\} \rightarrow i = 1$ to 7, $j = 1$ to 16 (we are taking length of frame = 16)

$S = \{s_{i,j}\} - i = 1$ to 7, $j = 1$ to 16 $\quad$ 16 samples.

Z - whitened signal (taken as input)

s^{est} - is computed of original sources by finding 7×7 unmixing matrix W

$$S^{est}_{7\times16} = W^T_{7\times7}\, Z_{7\times16}$$

$$W = \begin{bmatrix} w_{1,1} & w_{2,1} & w_{3,1} & w_{4,1} & w_{5,1} & w_{6,1} & w_{7,1} \\ w_{1,2} & w_{2,2} & w_{3,2} & w_{4,2} & w_{5,2} & w_{6,2} & w_{7,2} \\ w_{1,3} & w_{2,3} & w_{3,3} & w_{4,3} & w_{5,3} & w_{6,3} & w_{7,3} \\ w_{1,4} & w_{2,4} & w_{3,4} & w_{4,4} & w_{5,4} & w_{6,4} & w_{7,4} \\ w_{1,5} & w_{2,5} & w_{3,5} & w_{4,5} & w_{5,5} & w_{6,5} & w_{7,5} \\ w_{1,6} & w_{2,6} & w_{3,6} & w_{4,6} & w_{5,6} & w_{6,6} & w_{7,6} \\ w_{1,7} & w_{2,7} & w_{3,7} & w_{4,7} & w_{5,7} & w_{6,7} & w_{7,7} \end{bmatrix}$$

$= [\bar{w}_1, \bar{w}_2, \bar{w}_3, \bar{w}_4, \bar{w}_5, \bar{w}_6, \bar{w}_7] \rightarrow$ 7 vectors in 7D

$\bar{w}_1 \rightarrow 1$, 7D vector.

$$s^{est}_{7\times16} = W^T_{7\times7} \cdot Z_{7\times16}$$

$$= \begin{bmatrix} \bar{w}_1 \\ \bar{w}_2 \\ \bar{w}_3 \\ \bar{w}_4 \\ \bar{w}_5 \\ \bar{w}_6 \\ \bar{w}_7 \end{bmatrix}_{7\times7} \begin{bmatrix} z_{1,1} & z_{1,2} & z_{1,3} & z_{1,4} & \cdots & z_{1,16} \\ z_{2,1} & z_{2,2} & z_{2,3} & z_{2,4} & \cdots & z_{2,16} \\ z_{3,1} & z_{3,2} & z_{3,3} & z_{3,4} & \cdots & z_{3,16} \\ z_{4,1} & z_{4,2} & z_{4,3} & z_{4,4} & \cdots & z_{4,16} \\ z_{5,1} & z_{5,2} & z_{5,3} & z_{5,4} & \cdots & z_{5,16} \\ z_{6,1} & z_{6,2} & z_{6,3} & z_{6,4} & \cdots & z_{6,16} \\ z_{7,1} & z_{7,2} & z_{7,3} & z_{7,4} & \cdots & z_{7,16} \end{bmatrix}_{7\times16}$$

$$= \begin{bmatrix} \bar{w}_1 \\ \bar{w}_2 \\ \bar{w}_3 \\ \bar{w}_4 \\ \bar{w}_5 \\ \bar{w}_6 \\ \bar{w}_7 \end{bmatrix} \left\{ [z_{i,1}, z_{i,2}\ z_{i,3} - z_{i,4}\ z_{i,5} - \cdots - z_{i,16}]_{7\times16} \right.$$

$S^{est}_{7\times 16} = [$

$w^T \cdot z$

$$= \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} & w_{15} & w_{16} & w_{17} \\ w_{21} & w_{22} & w_{23} & w_{24} & w_{25} & w_{26} & w_{27} \\ w_{31} & w_{32} & w_{33} & w_{34} & w_{35} & w_{36} & w_{37} \\ w_{41} & w_{42} & w_{43} & w_{44} & w_{45} & w_{46} & w_{47} \\ w_{51} & w_{52} & w_{53} & w_{54} & w_{55} & w_{56} & w_{57} \\ w_{61} & w_{62} & w_{63} & w_{64} & w_{65} & w_{66} & w_{67} \\ w_{71} & w_{72} & w_{73} & w_{74} & w_{75} & w_{76} & w_{77} \end{bmatrix}_{7\times 7} \begin{bmatrix} z_{11} & z_{12} & z_{13} & z_{14} & z_{15} & z_{16} & \cdots & z_{1,16} \\ z_{21} & z_{22} & z_{23} & z_{24} & z_{25} & z_{26} & \cdots & z_{2,16} \\ z_{31} & z_{32} & z_{33} & z_{34} & z_{35} & z_{36} & \cdots & z_{3,16} \\ z_{41} & z_{42} & z_{43} & z_{44} & z_{45} & z_{46} & \cdots & z_{4,16} \\ z_{51} & z_{52} & z_{53} & z_{54} & z_{55} & z_{56} & \cdots & z_{5,16} \\ z_{61} & z_{62} & z_{63} & z_{64} & z_{65} & z_{66} & \cdots & z_{6,16} \\ z_{71} & z_{72} & z_{73} & z_{74} & z_{75} & z_{76} & \cdots & z_{7,16} \end{bmatrix}_{7\times 16}$$

$$= \begin{bmatrix} \boxed{w_{11}z_{11} + w_{12}z_{21} + w_{13}z_{31} + w_{14}z_{41} + w_{15}z_{51} + w_{16}z_{61} + w_{17}z_{71}}^{\,1,1} & w_{11}z_{12} + \cdots & \cdots \\ w_{21}z_{11} + \cdots & \cdots & \cdots \\ \vdots & & \vdots \end{bmatrix}_{7\times 16}$$

to make w_i (weight vector) independent → normalize, orthogonalize, contrast function (update).

$$w_i^{new} = E[z(w_i^T z)^3] - 3w_i$$

$$w_i^{new} = w_i^{new} - \sum_{j=0}^{i-1} (w_i \cdot w_j^c) w_j^c$$

$$w_i^{new} = \frac{w_i^{new}}{\| w_i^{new} \|}$$ → check to w_i^{new} is converged to w_i within ε (convergence threshold)

→ till $i = n$ → repeated

then S^{est} is found by matrix multiplication of W_{new} & Z to get ~~updated~~ & estimated sources $S^{est}_{7\times 16}$.

→ Inputs are:

$Z_{7\times 16} = [$ ~~1200 3400 4500 600 2100 5700 3900 1800~~

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1200	3400	4000	600	2100	5700	3900	1800	900	4200	6100	2800	3300	1500	2200	5400
2100	4000	5300	1700	3200	6500	4100	2900	1100	5000	3800	4600	2300	1400	6000	3700
3300	2200	1100	4500	5600	800	3700	6200	2400	1900	3500	4800	5700	6300	2500	1600
4200	5800	2400	3100	6000	1300	4900	2600	3900	1200	5400	900	6100	4300	3200	2700
5500	1600	3800	6300	2700	4400	1900	5100	3300	4000	2500	1000	3600	6400	2800	5900
900	6100	3000	4900	5300	2100	6500	4600	2800	3700	1400	2200	5700	1800	4200	3500
600	2500	1800	5200	4100	3600	2900	6400	3800	5000	1700	4500	800	2300	6200	3100

$$
w_{7\times 7} = \begin{array}{c|ccccccc}
 & w_1 & w_2 & w_3 & w_4 & w_5 & w_6 & w_7 \\
\hline
1D & 1000 & 1000 & 1500 & 2000 & 1000 & 2500 & 3000 \\
2D & 1000 & 2000 & 3000 & 4000 & 2000 & 3200 & 4200 \\
3D & 1000 & 3000 & 2000 & 1000 & 3000 & 2000 & 4400 \\
4D & 1000 & 4000 & 3000 & 2000 & 1000 & 3100 & 5000 \\
5D & 1000 & 5000 & 4000 & 3000 & 2000 & 1000 & 3000 \\
6D & 1000 & 6000 & 5000 & 4000 & 3000 & 1200 & 1600 \\
7D & 1000 & 7000 & 1000 & 3000 & 4000 & 1800 & 1600
\end{array}
$$

Process: for $i=1, k=1$

① Take w_1 ($k=1$ – no G(s), only normalize) $\longrightarrow \underline{w}_1\,(7\times 1)$ ($i=1$ - direct update)

$$w_{1,new} = E[z(w^T z)^3] - 3w_k \quad (i = i+1)$$

$$\left[\frac{z^1_{7\times1}(w^T_{1\,1\times7}\cdot z^1_{7\times1})^3 + z^2_{7\times1}(w^T_{1\,1\times7}\cdot z^2_{7\times1})^3 + \cdots z^{16}_{7\times1}(w^T_{1}\cdot z^{16}_{7\times1})^3}{16}\right] - 3\underline{w}_1 = w_{1,new}$$

$w_{1,new} \longrightarrow \underline{w}_{1,new}$ (normalize) $\longrightarrow$ as $i=2$ $\quad w_{diff} = \underline{w}_{1,new} - \underline{w}_1 < 16'd10$

$\varepsilon = 0.001 = 16'd10$ (lets take $\varepsilon = 10^{-3}$) later we { desire number of iterations convergence }

elements if $< 16'd10$ → (we got 1st ~~column~~ row of $W^T_{1\times7}$th estimation) – similarly we got other columns in next 6 steps & then do matrix multiplication with $Z_{7\times16}$.

again iteratively.

Math:

$w_1 = \{1000, 1000, 1000, 1000, 1000, 1000, 1000\}$ ✓

$\underline{w} = \{378, 378, 378, 378, 378, 378, 378\}$ ✓

$\underline{w}^T z^1 =$ (scalar product)

$\underline{w}^T z^1 \;(0.378)[(1.2) + (9.1) + (3.3) + (4.2) + (5.5) + (0.9) + (6)]$

$= (0.378)(23.2) = 8769$

$\underline{w}^T z^2 = 0.378[25.6] = 9676$ $\qquad$ 3 → $0.378(21.9) = 8278$

4 → $0.378(26.3) = 9941$ $\qquad$ 5 → $0.378(29) = 10962$

6 → $0.378(24.4) = 9223$ $\qquad$ 7 → $0.378(27.7) = 10470$

8 → $0.378(29.6) = 11188$ $\qquad$ 9 → $0.378(18.2) = 6879$

10 → $0.378(24.9) = 9412$ $\qquad$ 11 → $0.378(24.4) = 9223$

12 → $0.378(20.8) = 7862$ $\qquad$ 13 → $0.378(26.9) = 10168$

14 → $0.378(24) = 9072$ $\qquad$ 15 → $0.378(27.1) = 10243$

16 → $0.378(25.9) = 9790$

$E((w^T z)^3 z) \rightarrow$

$z_{1}(w^T z_{i})^3 = 1.2\,(9.676)^3 = 1087 +$ $/16 =$

$z_{2}(w^T z_{i})^3 = 2.1\,(9.676)^3 = 1902 +$ $/16 =$

$z_{3}(w^T z_{i})^3 = 3.3\,(9.676)^3 = 2989 +$ $/16 =$

$z_{4}(w^T z_{i})^3 = 4.2\,(9.676)^3 = 3804 +$ $/16 =$

$z_{5}(w^T z_{i})^3 = 5.5\,(9.676)^3 = 4982 +$ $/16 =$

$z_{6}(w^T z_{i})^3 = 0.9\,(9.676)^3 = 815 +$ $/16 =$

$z_{7}(w^T z_{i})^3 = 6.0\,(9.676)^3 = 5435 +$ $/16 =$

(8.769)

$8769 \rightarrow w^T z'$ $\quad (w^T z')^3 = (8.769)^3 = \underline{674.295}$

↳ The thing should be 48 bit right!?

So, take z values also as normalized ones - then there would be no problem!! → if not check how?

$z_{7\times16}$ = (normalized values -

$z_1 = \{119, 209, 328, 417, 545, 89, 595\}$
$z_2 = \{327, 378, 208, 548, 159, 578, 236\}$
$z_3 = \{497, 585, 121, 265, 420, 331, 199\}$

checked in code

$z_4 = \{054, 153, 404, 279, 566, 440, 467\}$
$z_5 = \{181, 276, 483, 528, 233, 458, 354\}$
$z_6 = \{534, 609, 075, 122, 412, 197, 337\}$
$z_7 = \{353, 371, 335, 425, 172, 588, 202\}$
$z_8 = \{149, 241, 515, 216, 424, 382, 532\}$
$z_9 = \{120, 147, 320, 520, 440, 374, 507\}$
$z_{10} = \{415, 495, 188, 419, 395, 366, 494\}$
$z_{11} = \{598, 372, 343, 529, 245, 137, 167\}$
$z_{12} = \{315, 518, 540, 101, 113, 248, 506\}$
$z_{13} = \{295, 205, 455, 545, 321, 509, 071\}$
$z_{14} = \{442, 132, 596, 407, 605, 170, 217\}$
$z_{15} = \{200, 545, 227, 291, 254, 381, 563\}$
$z_{16} = \{516, 354, 153, 258, 564, 335, 296\}$

next segment ↓ the values can be normal also.

$w^T z_1 \quad w^T z_{16} \rightarrow (k_{16})^3$

$\underline{\qquad\qquad}$

7

$w^T z' = 0.378\{2.303\}^3 = 870$ eigenvector corresponding to $w_{1,1}z_{11} + w_{12}z_{21} + \cdots w_{17}z_{71}$

2 $0.378\{2.418\}^3 = 914$

3 $0.378\{2.419\}^3 = 914$

4 $0.378\{2.363\}^3 = 893$

5 $0.378\{2.503\}^3 = 946$

6 $0.378\{2.287\}^3 = 864$

7 $0.378\{2.507\}^3 = 947$

8 $0.378\{2.458\}^3 = 929$

9 $0.378\{2.429\}^3 = 918$

10 $0.378\{2.471\}^3 = 933$

11 $0.378\{2.391\}^3 = 903$

12 $0.378\{2.341\}^3 = 884$

13 $0.378\{2.401\}^3 = 907$

14 $0.378\{2.269\}^3 = 857$

15 $0.378\{2.461\}^3 = 930$ all 16 'k' values

16 $0.378\{2.477\}^3 = 936$ → scalar product

$w_{1,1}z_{1,16} + w_{12}z_{2,16} + \cdots w_{17}z_{7,16}$

$0.870 \times 0.870 = 0.7569$

$0.870 \times 0.870 \times 0.870 = 0.658$ → $\underline{658} = (w^T z')^3$

$z \times (w^T z')^3 = (0.119) \times (0.658) = 0.0783 = 78$

$k_1^3 \times z_{2,1} = (0.209) \times (0.658) = 0.1375 = 137$

$k_1^3 \times z_{3,1} = (0.328) \times (0.658) = 0.2158 = 215$

$k_1^3 \times z_{4,1} = (0.417) \times (0.658) = 0.2743 = 274$

$k_1^3 \times z_{5,1} = (0.546) \times (0.658) = 0.3592 = 359$

$k_1^3 \times z_{6,1} = (0.089) \times (0.658) = 0.0585 = 58$

$k_1^3 \times z_{7,1} = (0.596) \times (0.658) = 0.392 = 392$ working

$k_1 z_{11} + k_2 z_{12} + k_3 z_{13} + \cdots k_{16} z_{1,16} / 16 - 3(0.378$

Normalization ✓ 980 ✓

Update:

$k[15:0]$ k - scalar product $(w^T z^i)$

~~k[1]~~ $z_1[15:0]$, $z_2[15:0]$ ---- . $z_7[15:0]$

$k[15:0] = [870, 914, 914, 893, 946, 864, 947, 929, 918, 933,$
$903, 884, 907, 857, 930, 936]$

$E_{14}(z \times k^3) = \dfrac{k^3[0]\, z_1[0] + k^3[1]\, z_1[1] + \dots\, k^3[15]\, z_1[15]}{16}$

$\therefore w_{new,1,1} = E_{1,1}(z \times k3) - 3\, w_{1,1}$

an - $E(z \times k3)$

$E_2(z \times k3) = k^3[0]\, z_2[0] + k^3[1]\, z_2[1] + \dots\, k^3[15]\, z_2[15]$

$(870)^3 \times 119 + (914)^3 \times 321 + (914)^3 \times 497 + (893)^3 \times 54 + (946)^3 \times 181$
$+ (864)^3 \times 534 + (947)^3 \times 353 + (929)^3 \times 149 + (918)^3 \times 120 + (933)^3 \times 415 \times$
$(902)^3 \times 598 + (884)^3 \times 315 + (907)^3 \times 295 + (857)^3 \times 142 + (930)^3 \times 200$
$+ (936)^3 \times 516 = \cancel{4378164 \times 10^6} = \cancel{4378164/10^3} = \cancel{4378}$

$= 0.078 + 0.245 + 0.379 + 0.038 + 0.153 + 0.344$
$+ 0.299 + 0.119 + 0.092 + 0.337 + 0.438 + 0.217$
$+ 0.220 + 0.000087 + 0.1608 + 0.423 = 3.162187 = \cancel{3162}$

3630 ✓

$= \cancel{\dfrac{3162}{16}} = \cancel{527}\ \dfrac{3630}{16} = 226$ ✓

$E_{11} = 226$, $E_{12} = 263$ $E_{13} = 246$ $E_{14} = 262$

$E_{15} = 271$ $E_{16} = 268$ $E_{17} = 272$

$$\begin{bmatrix} w_{new,11} \\ w_{new,12} \\ w_{new,13} \\ 4 \\ 5 \\ 6 \\ 7 \end{bmatrix} = \begin{bmatrix} 226 - 378 \times 3 \\ 263 - 378 \times 3 \\ 246 - 378 \times 3 \\ 262 - 378 \times 3 \\ 271 - 378 \times 3 \\ 268 - 378 \times 3 \\ 272 - 378 \times 3 \end{bmatrix} =$$

16×

~~226×1~~

Sum =

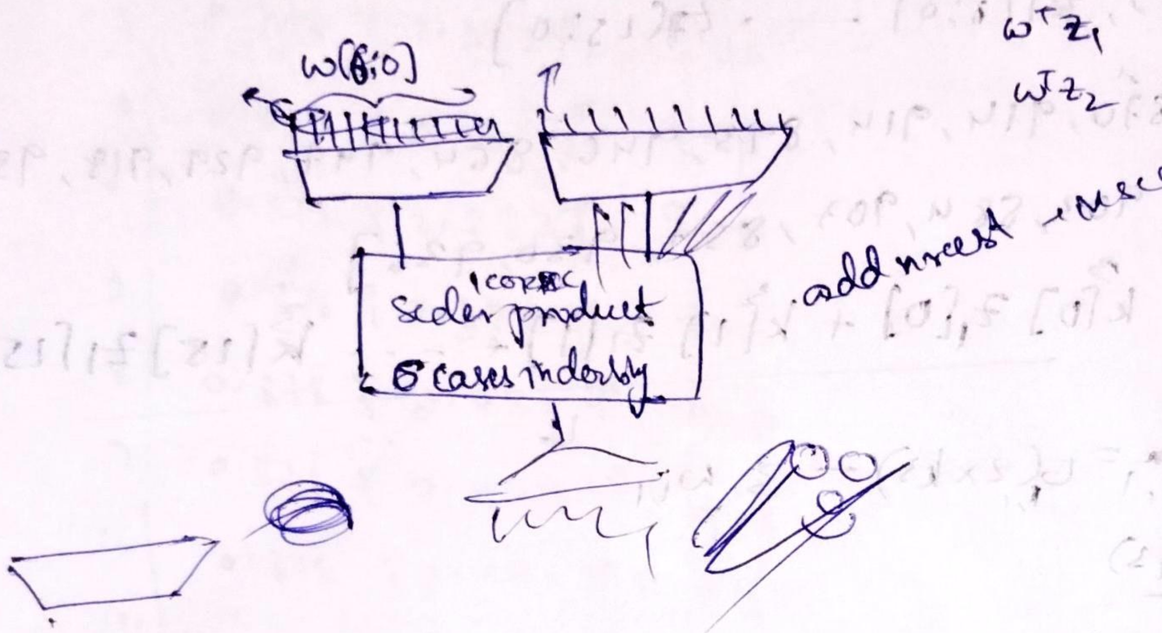
Q.e

5.

21

~~write both n~~ for w_1 → normalization → update (till $w_{diff} < \varepsilon = 16'd10$)
i → no. of updates for each vector (i this made 1 after 1 vector updation is done)

Update block:



~~And the thing~~

And the thing I was talking about...

Case-1: In testbench we are giving 1 as 1000.
In this case
$1 \equiv 1000$

$$\frac{1}{\sqrt{7}} = 0.378 = 378$$

(we are getting this only in normalization ✓)

Case-2: If we give 1 as 1024

$1 \equiv 1024$

$$\frac{1}{\sqrt{7}} = 0.378 \equiv 378 = \cancel{\frac{1024}{\sqrt{7}}} \quad 387$$

In this case we are dividing both of them

$$\frac{1024}{\sqrt{(1.024)^2 \times 7}} = 378 ✓ \qquad = \frac{1024 \cdot 1}{1024 \times \sqrt{7}}$$

basically normalizing, so it should not matter but see

But in G80 when we are doing

taking $1 \equiv 1000$ 2 = 2000 ... 7 = 7000

we get, 868, 1865, 2729, 3424, 3670,

$1 = 1024$, $2 = 2048$ ... $7 = 7 \times 1024$

we get → 886, 1910, 2795, 3507, 3760, 2747, −1940

case there's error because it is ~~not~~ taking it as 1, instead taking it as 1.024 only.

as if we do math using 1 we get the values that we are getting with 1000 only.

when we are taking 1 as 1000 we aren't changing any math just shifting the decimal point

But if we consider 1 as 1024 we are changing the whole math at places. where it should do cubes

$1^3 = 1$ $(1000)^3 = 1000000000$

$(1024)^3 = 1048680876$

↳ by shifting we can get it back to 1024 but when we do multiplication we won't be shifting so there will be error from math.

$1 \times 1 = 1$

$1000 \times 1000 = 1000$

$1024 \times 1024 = 2048$

like whole math at places,

$a_1 \cos\theta + a_2 \sin\theta \rightarrow 1\cos\theta + 1\sin\theta = \frac{1}{\sqrt{2}} + \frac{1}{\sqrt{2}} = \sqrt{2}$

↳ 1.414

$1000\left(\frac{1}{\sqrt{2}}\right) + 1000\left(\frac{1}{\sqrt{2}}\right) = \frac{2000}{\sqrt{2}} = 1414$

$1024\left(\frac{1}{\sqrt{2}}\right) + 1024\left(\frac{1}{\sqrt{2}}\right) = 1448 \neq 1414.$

↳ This is equal to 1.414 in that convention but my whole point is the math isn't matching like the numbers.

Rn we don't have problem with using 1024 also cause the error is relatively small only but again when we want to compare with our math values they might not match. eg we can think more on this

Nice

gro / rot / up

Normalization → clk, k, enable-norm, nreset-norm,
done-norm, norm-in [6:0], theta-norm [5:0]
k = 0 & i = 0
take from memory
norm-out [6:0].

gso → clk, i, k, nreset-gso, enable-gso,
theta1 [5:0], theta2 [5:0] - - thetac [5:0], wnew [6:0], gso-out [6:0]
done-gso.

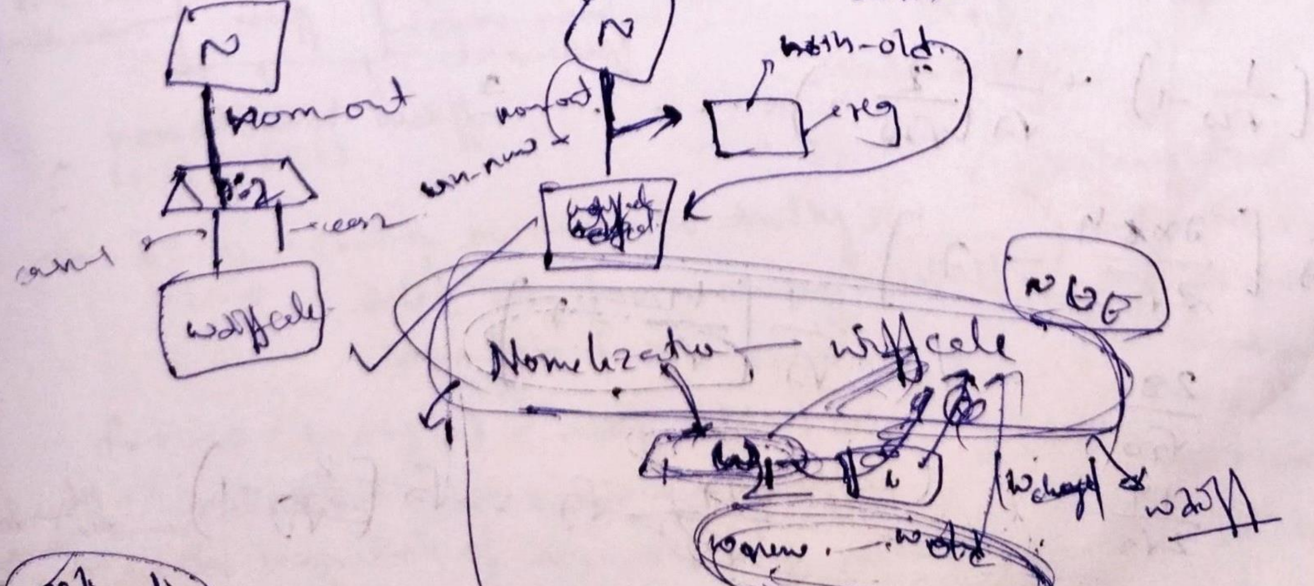
Update → clk, k, enable-update, nreset-update,
done-update, w in [6:0], wnew [6:0].

wdiff calc ~~converge~~ → clk, k, nreset-converge, epsilon,
enable-converge, done-converge, wdiff
~~win-old~~, ~~win-new~~ win-diff,

~~Normal-converge → clk, k, i, enable-norm, nreset-norm~~
~~epsilon; done-norm, norm-in, theta-norm,~~
~~for update norm-out, converge~~ if compared for gso.
~~wdiff~~ if wdiff < ε
put in shift register & next norm

doubt → if nrest-norm is applied then value stored in shift registers will become 0 right?

NOTE:
normalization → norm-out is stored in a register & connected to demux



estimation

Normalization_wdiff]

clk, i, k, enable_norm, nrest_norm, norm_in,
norm_out, theta_out done_norm, ~~epsilon~~,

find gso → gso_vectors_7D_rot

find normalization → normalization_7D_rol

find normalization_wdiff → normalization_wdiff_7D

extra:

wm_old

wm_diff [6:0] , ~~epsilon, converge~~ wdiff till inputs done

add block of wdiff calc

calculate difference outside & gives as input

w_{new} $\frac{1}{\sqrt{140}}$ $\frac{2}{\sqrt{140}}$ $\frac{3}{\sqrt{140}}$ $\frac{4}{\sqrt{140}}$ $\frac{5}{\sqrt{140}}$ $\frac{6}{\sqrt{140}}$ $\frac{7}{\sqrt{140}}$

w_{old} $\frac{1}{\sqrt{7}}$ $\frac{1}{\sqrt{7}}$ $\frac{1}{\sqrt{7}}$ $\frac{1}{\sqrt{7}}$ $\frac{1}{\sqrt{7}}$ $\frac{1}{\sqrt{7}}$ $\frac{1}{\sqrt{7}}$

$\sqrt{20 \times 7}$

$$\frac{1}{\sqrt{20}}\left(\frac{1}{\sqrt{7}}-\frac{1}{\sqrt{7}}\right)+\frac{1}{\sqrt{20}}\left(\frac{2}{\sqrt{7}}-\frac{1}{\sqrt{7}}\right) \longrightarrow$$

$$= \frac{1}{\sqrt{20}}\left[0 + \frac{1}{\sqrt{7}} + \frac{2}{\sqrt{7}} + \cdots \frac{6}{\sqrt{7}}\right]$$

$\cancel{3} \; 3 \; 6 \times 7$

$2\sqrt{7}$

$\frac{21}{\sqrt{7}} \times \frac{1}{\sqrt{20}}$

$\frac{3\sqrt{7}}{\sqrt{20}}$ 1.7748

$$\left(\frac{1}{\sqrt{20}} \times \frac{1}{\sqrt{7}} - \frac{1}{\sqrt{7}}\right)$$

$$\frac{1}{\sqrt{7}}\left(\frac{1}{\sqrt{20}}-1\right) + \frac{1}{\sqrt{7}}\left(\frac{2}{\sqrt{20}}-1\right) + \cdots \quad 7$$

$$\frac{1}{\sqrt{7}}\left[\frac{7 \times 8}{2\sqrt{20}} - 7\right]$$

$\frac{28}{\sqrt{20}}$

$\frac{1}{\sqrt{7}}\left[\frac{14}{\sqrt{5}} - 7\right]$

$\frac{14}{2\sqrt{5}}$

$$\frac{2\sqrt{7}}{\sqrt{5}} - \sqrt{7} = \sqrt{7}\left[\frac{2}{\sqrt{5}} - 1\right]$$

$\frac{1}{\sqrt{7}}\left(\frac{1}{\sqrt{20}}-1\right), \frac{1}{\sqrt{7}}\left(\frac{2}{\sqrt{20}}-1\right), \frac{1}{\sqrt{7}}\left(\frac{3}{\sqrt{40}}-1\right) \cdots \frac{1}{\sqrt{2}}\left(\frac{2}{\sqrt{20}}+0\right)$

norm – $\frac{1}{\sqrt{7}}\left[\left(\frac{1}{\sqrt{40}}-1\right)^2+\left(\frac{2}{\sqrt{40}}-1\right)^2+ \ldots \left(\frac{A}{\sqrt{40}}-1\right)^2\right]^{1/2} \approx 0.459$

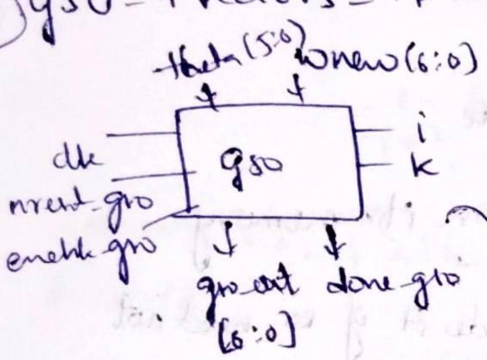
(perfectly working)

– scalar product – taking thetam as input & make estimation block directly!

~~GSO:~~

Final ICA modules:

1) GSO-7 vectors_7D_rot: clk, nreset_gso, enable_gso, k, i, theta1, theta2, theta3, theta4, theta5, theta6, wnew, gso_out, done_gso.
(input)

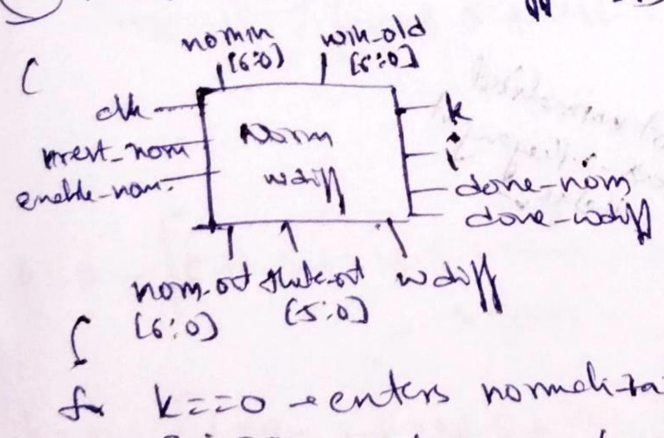


for k==0 (doesn't enter gso)
for i==0 → takes input from memory (for any k)
for any i ∈ 1 to 15, k ∈ 1 to 6 → takes wnew as input.
takes theta as input

thetas are stored in demux outputs of normalization

- 6 ← for k=2 → theta1(5:0)
- 12 ← k=3 → theta2(5:0), theta1(5:0)
- 18 ← k=4 → theta3(5:0), theta2(5:0), theta1(5:0)
- 24 ← k=5 → $\theta_4^{(5:0)}, \theta_3^{(5:0)}, \theta_2^{(5:0)}, \theta_1(5:0)$
- 30 ← k=6 → $\theta_5(5:0), \theta_4(5:0), \theta_3(5:0), \theta_2(5:0), \theta_1(5:0)$
- 36 ← k=7 → $\theta_6, \theta_5, \theta_4, \theta_3, \theta_2, \theta_2, \theta_1$ – each with 5:0

2) Normalization_wdiff_7D: clk, nreset_norm, enable_norm, k, i, norm_in (6:0), wm_old, norm_out (6:0), theta_out (5:0), wdiff, done_norm, done_wdiff.



for k==0 & i==0 – enters normalization directly so take wm from memory, rest all cases take from norm_in.

for i==0 (for any k) – wdiff shouldn't be calculated – we are assigning wdiff as the magnitude of norm_out only; rest all cases doing difference of norm_out with wm_old & finding magnitude – but in control block make it not enter.

already found – extra clk cycles

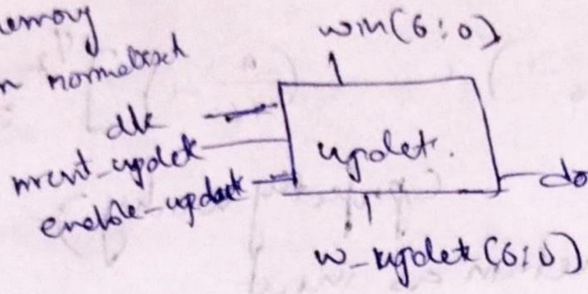
we might not need to add the i==0 case in code as in control signal we can regulate like if i==0 then if done_norm do nreset (don't wait till done_wdiff) – doubts.

but we added code for self code!

(3) Update_ICA_~~mem~~mem: clk, nreset_update, enable_update, done_update, win[6:0], w_updated[6:0]

i & k are not reg.

z from memory
win from normalizati



w_updated.

$$E\left[z\,(w_{in}^{T}z)^3\right] - 3\,w_{in}$$

(4) Estimate_ICA : clk, nreset_estimate ~~X~~, enable_estimate, done_estimate, theta_norm(5:0), ~~z~~ z_estimated[15:0]

16 x 122 clkcycles

after 16 x 122 ~~clkcycles~~

shows w_final

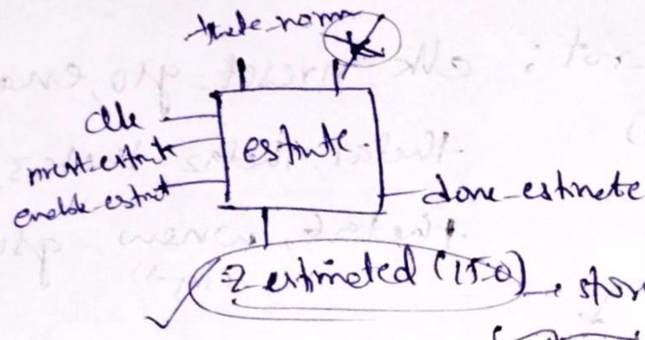
each row it calculates once & sends to memory so ~~nreset~~ reg.

z ~~in~~ it takes from memory

i & k ~~are~~ is not reg

k is required - as to send to memory

if k=1 - it will send to one location till k=7 it will send to other locations



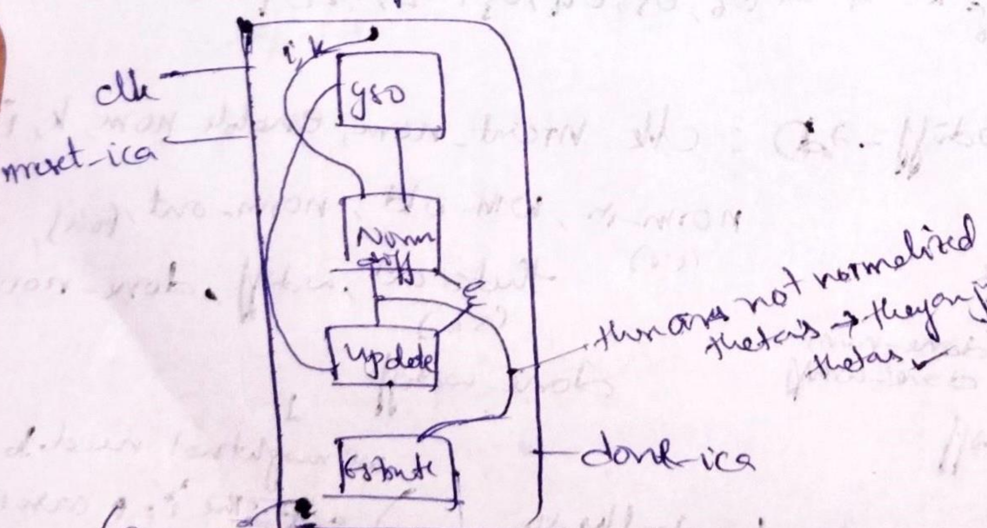
store it in memory.

check how to do it if so, need not send it as an output.

check this: if zestimated is not given out as output & stored in memory then k should be given as input. (otherwise if we are storing outside then k ~~should~~ ~~given~~ need not be given & zestimated should be given as output)

lets add to the memory outside

(5) ICA_ICA_final : (method-1) as of now.



these are not normalized thetas → they are just thetas

(everything stored in memory)

(6) NUE_ICA - Mix all Normalization_diff, update & estimation

This we can do later (using single hardware) as of now lets use individual modules only.

clock cycles: (for one run)

gso → k = 0 → 2.5
(i ≠ 0) k = 1 → 224.5
k = 2 → 448.5
k = 3 → 671.5
k = 4 → 894.5
k = 5 → 1117.5
k = 6 → 1338.5

Normalization with wdiff (any k) → i = 0 → 216.5 (no wdiff calc)
i ≠ 0 → 325.5

update → 122×16 = 1952 (for 16 samples).
any i
any k

Estimate → 1912. (119.5×16) → for 16 samples 1 vector.

whole ICA clk cycles! clk cycle = 10ns. (2.84757×10n = 0.00284 = 2.84ms)

k=0, i=0 → Normalization (216.5) → update (1952)
↓ i=1
. . . ← update (1952) i=2 ← Normalize (325.5) ← ~~gso~~ (2.5)

imax = 15 → [(216.5 + 1952) + 15 (1952 + 325.5)]
for k=0
= 36,331 → estimate (1912) → 38243 (max)

k=1, i=0 → gso (224.5) → Nor (216.5) → update (1952)
↓ i=1
. . . ← update (1952) i=2 ← Nor (325.5) ← gso (224.5)

imax = 15 → [(224.5 + 216.5 + 1952) + 15 (224.5 + 325.5 + 1952)]
= 39923 → Estimat (1912) → 41835 (max).

k=2 → 448.5 → 216.5 → 1952 $\xrightarrow{i=1}$ 448.5 → 325.5
↓
1952

imax = 15 → [448.5 + 216.5 + 1952 + 15 (448.5 + 325.5 + 1952)] → esti (1912)
= 43507
↓
45419 (max)

k=3 → [(671.5 + 216.5 + 1952) + 15 (671.5 + 325.5 + 1952)]
= 47075 → esti (1912) → 48987 (max)

k=4 → [(894.5 + 216.5 + 1952) + 15 (894.5 + 325.5 + 1952)]
= 50643 → esti (1912) → 52555 (max)

k=5 → [(1117.5 + 216.5 + 1952) + 15 (1117.5 + 325.5 + 1952) + 1912
= 56123 (max)

k=6 → 1338.5 + 216.5 = 1555 + 1952 = 3507
memory efficiency can be saved for the data

Total: 38243 + 41835 + 43507 + 48987 + 52555 + 56123 + 3507 = 284757 clk cycles.

done-norm   done-wdiff e funds (NUE)

win [6:0]

thetas-norm[5:0] → $\boxed{\text{thetas\_norm}^{[5:0]} <= \text{theta\_norm}^{[3:0]}}$ from normlz

to estimate

$\boxed{\text{win}[6:0] <= \text{norm\_out}[6:0]}$ ← when you get done-norm as 1

to updat   from normlzt

		Normlize	Updat	Estmate
Normelizelm	Vec-xin	mux 16:1	mux 8:1	—
Updat	Vec-yin	mux 16:1	mux 8:1	—
Estmat	Rot-xin	~~mux~~	mux 8:1	mux 8:1
	Rot-yin	~~demx~~	mux 8:1	mux 8:1
	Vec-nout	demx 1:16	dem 1:8	—
	vec-angle-out	dem 1:8	dem 1:8	—
	Rot-xout	demx 1:8	demx 1:8	demx 1:8
	Rot-yout	demx 1:8	—	~~demx~~ —

2   2   3

norml 8:1   wdiff 8:1

16:1 + 8:1

sel → 5bits

16:1 → normly

24:17 → updat

32:25 → estmt

mux 32:1 → Vec-xin, Vec-yin, Rot-xin, Rot-yin

~~16:1 e Rot~~

norml, wdiff

16:1 → nout

24:17 → updat

32:18 → estmt

demx 1:32 → vec-nout, vec-angle-out, Rot-xout, Rot-yout.

1111

1111

10000

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16

32:1 mux

1:32 dem

16:1 mux } up

1:16 demx } Estm

sel-m3-norm <= 1

32:1

25

1

5 →

0-7

8-15

16-23

24-31

01111

10001

don't chng } sel-mux-updt

m3-upd

sel-d-updat

4 bits only.

2nd clk } sel m1-estmt

m1

sel-destm

4 bits