HW4 Systolic Array Mapping & Mapping of Vector Quantization

VLSI DSP HW4

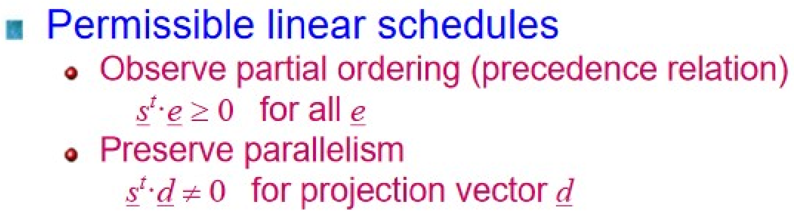
Shun-Liang Yeh, NCHU Lab612

4/29 2023

INDEX

I. Dependence graph mapping

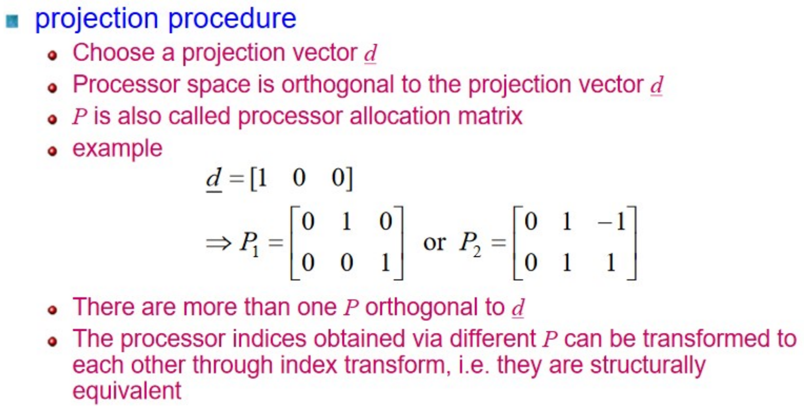
Linear schedule Checking



- Permissible linear schedules
  - Observe partial ordering (precedence relation)
    $\underline{s}^t \cdot \underline{e} \geq 0$ for all $\underline{e}$
  - Preserve parallelism
    $\underline{s}^t \cdot \underline{d} \neq 0$ for projection vector $\underline{d}$

- From the selected Scheduling Vector and Projection vector, check two conditions.
- Check whether the inner product of scheduling vector s and projection vector d is not equal to zero.
- Check whether the inner product of scheduling vector and every edges on DG is equal or greater then zero.

Projection Procedure



- projection procedure
  - Choose a projection vector $\underline{d}$
  - Processor space is orthogonal to the projection vector $\underline{d}$
  - P is also called processor allocation matrix
  - example

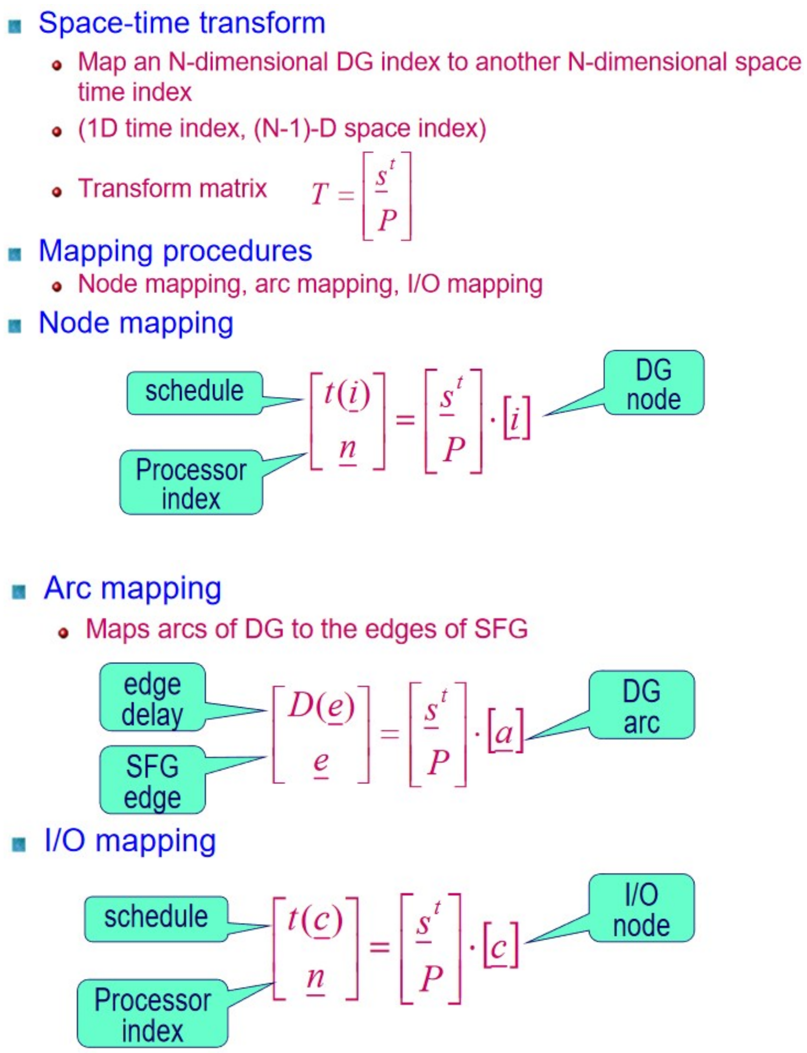
$$\underline{d} = [1 \quad 0 \quad 0]$$

$$\Rightarrow P_1 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \text{ or } P_2 = \begin{bmatrix} 0 & 1 & -1 \\ 0 & 1 & 1 \end{bmatrix}$$

  - There are more than one P orthogonal to $\underline{d}$
  - The processor indices obtained via different P can be transformed to each other through index transform, i.e. they are structurally equivalent

- Select a suitable processor vector P s.t. the inner product of processor vector P and projection vector d is 0.

Space time transform



- Space-time transform
  - Map an N-dimensional DG index to another N-dimensional space time index
  - (1D time index, (N-1)-D space index)
  - Transform matrix $T = \begin{bmatrix} \underline{s}^t \\ P \end{bmatrix}$
- Mapping procedures
  - Node mapping, arc mapping, I/O mapping
- Node mapping

$$\begin{bmatrix} t(\underline{i}) \\ \underline{n} \end{bmatrix} = \begin{bmatrix} \underline{s}^t \\ P \end{bmatrix} \cdot [\underline{i}]$$

schedule
Processor index
DG node

- Arc mapping
  - Maps arcs of DG to the edges of SFG

$$\begin{bmatrix} D(\underline{e}) \\ \underline{e} \end{bmatrix} = \begin{bmatrix} \underline{s}^t \\ P \end{bmatrix} \cdot [\underline{a}]$$

- I/O mapping

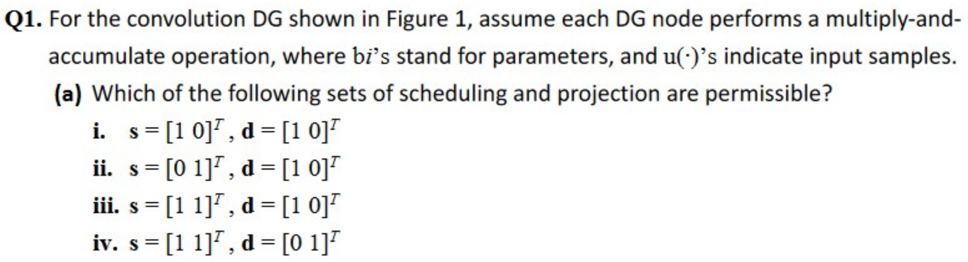
$$\begin{bmatrix} t(\underline{c}) \\ \underline{n} \end{bmatrix} = \begin{bmatrix} \underline{s}^t \\ P \end{bmatrix} \cdot [\underline{c}]$$

- From the derived s , d , P derive a transform matrix T for the N-th dimension DG graph.
- Start mapping every edges on the dependence graph according to the transform matrix T to get its space-time mapping results.
- From the mapping results of N-1 space-time dimension, sketch the edges on this new transformed space according to the mapping results.
- For more step by step algorithmic mapping examples, I suggest watching the lecture by Parhi, the video link is in the references.

II. Problem & Solution

Q1

a)



Q1. For the convolution DG shown in Figure 1, assume each DG node performs a multiply-and-accumulate operation, where b*i*'s stand for parameters, and u(·)'s indicate input samples.

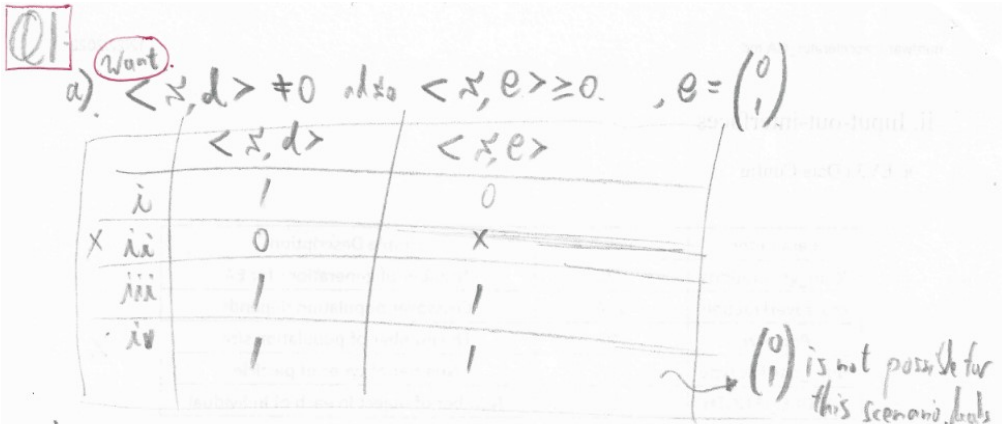
(a) Which of the following sets of scheduling and projection are permissible?

i. $\mathbf{s} = [1\ 0]^T$, $\mathbf{d} = [1\ 0]^T$

ii. $\mathbf{s} = [0\ 1]^T$, $\mathbf{d} = [1\ 0]^T$

iii. $\mathbf{s} = [1\ 1]^T$, $\mathbf{d} = [1\ 0]^T$

iv. $\mathbf{s} = [1\ 1]^T$, $\mathbf{d} = [0\ 1]^T$



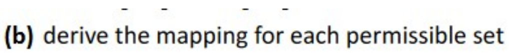
Q1

a) Want. $<s,d> \neq 0$ also $<s,e> \geq 0$. , $e = \begin{pmatrix}0\\1\end{pmatrix}$

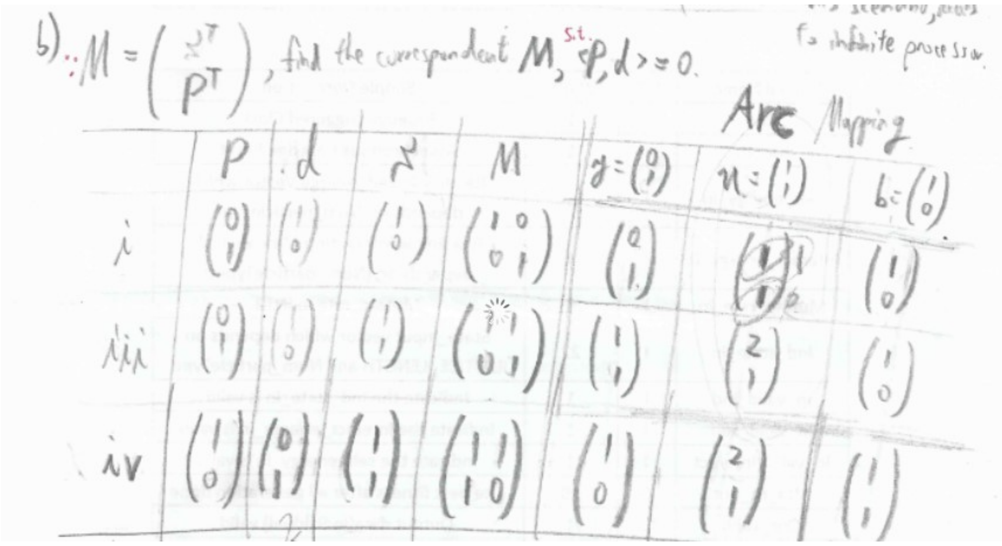
	$<s,d>$	$<s,e>$
i	1	0
X ii	0	X
iii	1	1
iv	1	1

→ $\begin{pmatrix}0\\1\end{pmatrix}$ is not possible for this scenario, leads

b)



(b) derive the mapping for each permissible set

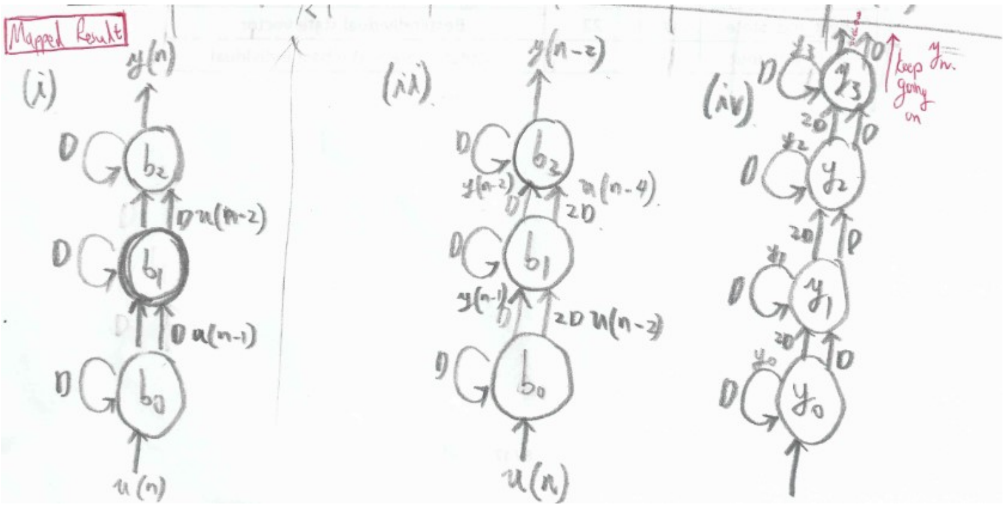


... scenario, leads to infinite processor.

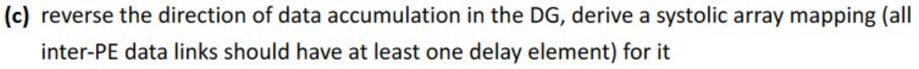
b) ∵ $M = \begin{pmatrix}s^T\\p^T\end{pmatrix}$, find the correspondent M, s.t. $<p,d> = 0$.

Arc Mapping

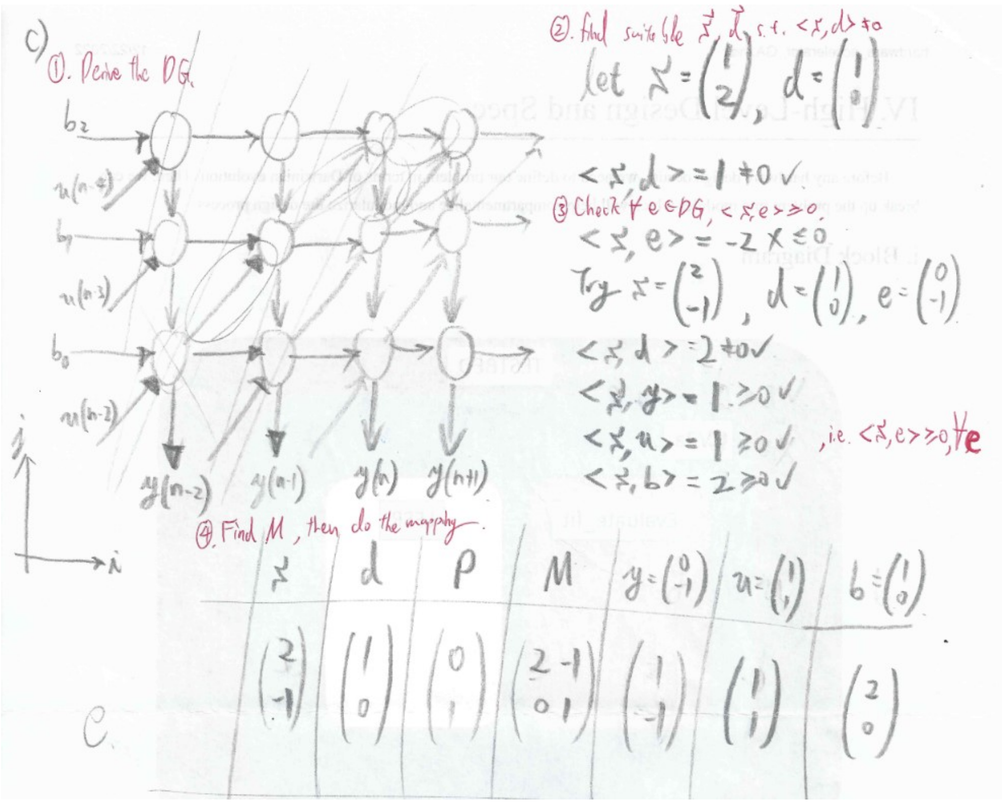
	P	d	s	M	$y=\begin{pmatrix}0\\1\end{pmatrix}$	$n=\begin{pmatrix}1\\1\end{pmatrix}$	$b=\begin{pmatrix}1\\0\end{pmatrix}$
i	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}1&0\\0&1\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$	$\cancel{\begin{pmatrix}1\\1\end{pmatrix}}\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$
iii	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}1&1\\0&1\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}2\\1\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$
iv	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}0\\1\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$	$\begin{pmatrix}1&1\\1&0\end{pmatrix}$	$\begin{pmatrix}1\\0\end{pmatrix}$	$\begin{pmatrix}2\\1\end{pmatrix}$	$\begin{pmatrix}1\\1\end{pmatrix}$



c)



(c) reverse the direction of data accumulation in the DG, derive a systolic array mapping (all inter-PE data links should have at least one delay element) for it



②. find suitable $\vec{s}, \vec{d}$, s.t. $\langle s, d\rangle \neq 0$

let $\vec{s} = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$, $d = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$

$\langle \vec{s}, d\rangle = 1 \neq 0$ ✓

③ Check $\forall e \in DG$, $\langle \vec{s}, e\rangle \geq 0$.

$\langle \vec{s}, e\rangle = -2$ ✗ ≤ 0

Try $\vec{s} = \begin{pmatrix} 2 \\ -1 \end{pmatrix}$, $d = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $e = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$

$\langle \vec{s}, d\rangle = 2 \neq 0$ ✓

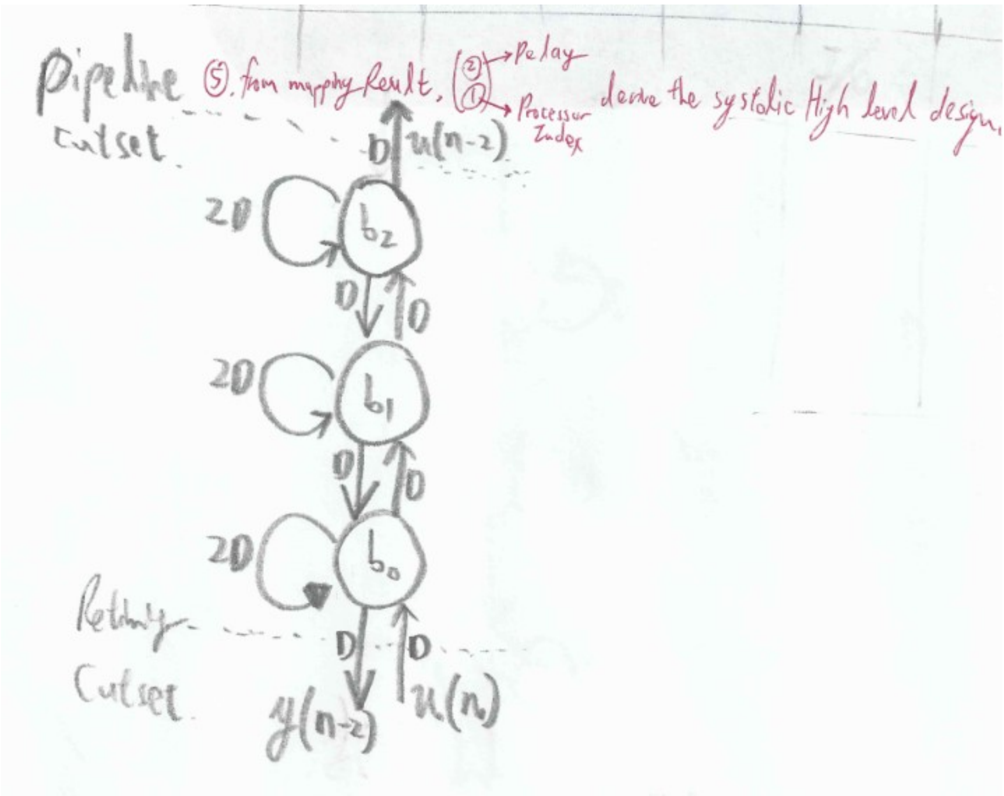
$\langle \vec{s}, y\rangle = 1 \geq 0$ ✓

$\langle \vec{s}, u\rangle = 1 \geq 0$ ✓ , i.e. $\langle \vec{s}, e\rangle \geq 0, \forall e$

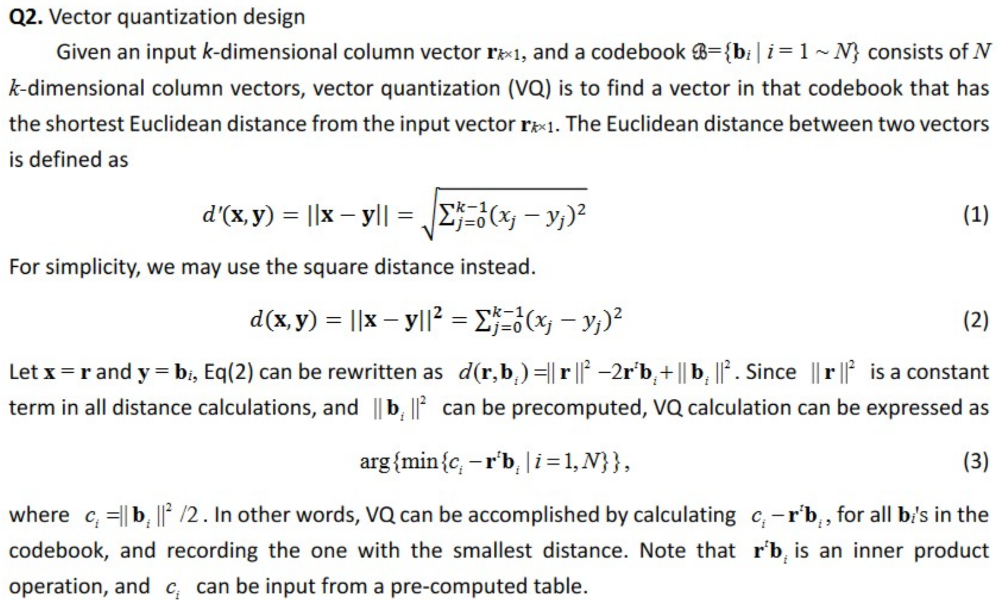
$\langle \vec{s}, b\rangle = 2 \geq 0$ ✓

④. Find M, then do the mapping.

$\vec{s}$	d	P	M	$y = \begin{pmatrix} 0 \\ -1 \end{pmatrix}$	$u = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$	$b = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$
$\begin{pmatrix} 2 \\ -1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 0 \end{pmatrix}$	$\begin{pmatrix} 0 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 2 & -1 \\ 0 & 1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ -1 \end{pmatrix}$	$\begin{pmatrix} 1 \\ 1 \end{pmatrix}$	$\begin{pmatrix} 2 \\ 0 \end{pmatrix}$


Pipeline Cutset
⑤. from mapping Result, ② → Delay ① → Processor Index derive the systolic High level design,
D
u(n-2)
2D
b2
D
D
2D
b1
D
D
2D
b0
Retiming Cutset
D
D
y(n-2)
u(n)

Q2



Q2. Vector quantization design

Given an input k-dimensional column vector $\mathbf{r}_{k\times 1}$, and a codebook $\mathfrak{B}=\{\mathbf{b}_i \mid i = 1 \sim N\}$ consists of N k-dimensional column vectors, vector quantization (VQ) is to find a vector in that codebook that has the shortest Euclidean distance from the input vector $\mathbf{r}_{k\times 1}$. The Euclidean distance between two vectors is defined as

$$d'(\mathbf{x},\mathbf{y}) = ||\mathbf{x}-\mathbf{y}|| = \sqrt{\sum_{j=0}^{k-1}(x_j - y_j)^2} \tag{1}$$

For simplicity, we may use the square distance instead.

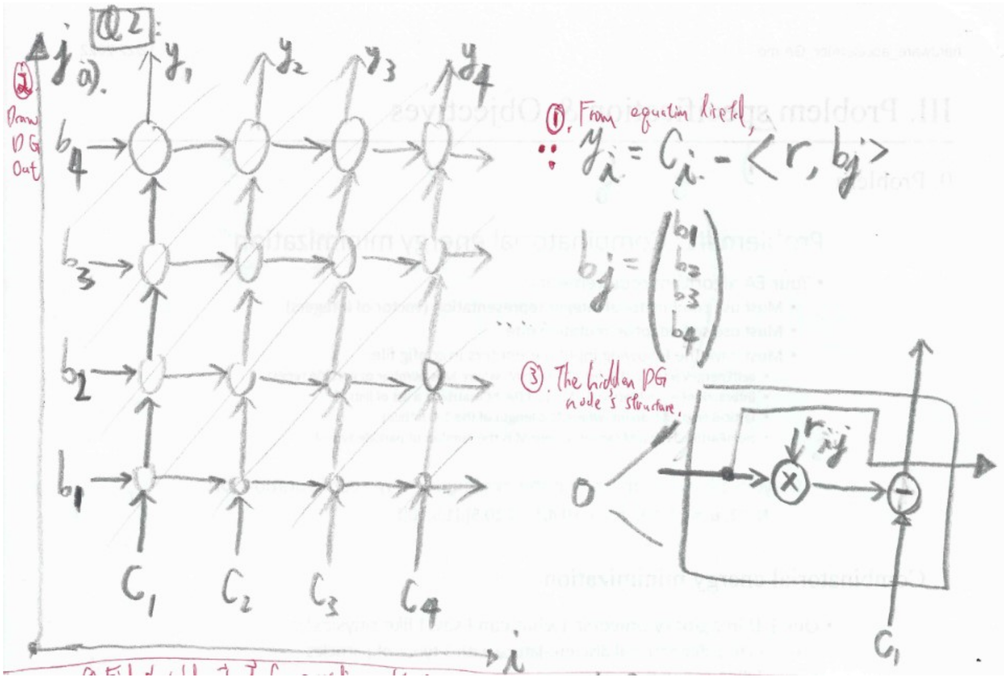
$$d(\mathbf{x},\mathbf{y}) = ||\mathbf{x}-\mathbf{y}||^2 = \sum_{j=0}^{k-1}(x_j - y_j)^2 \tag{2}$$

Let $\mathbf{x} = \mathbf{r}$ and $\mathbf{y} = \mathbf{b}_i$, Eq(2) can be rewritten as $d(\mathbf{r},\mathbf{b}_i) = ||\mathbf{r}||^2 - 2\mathbf{r}^t\mathbf{b}_i + ||\mathbf{b}_i||^2$. Since $||\mathbf{r}||^2$ is a constant term in all distance calculations, and $||\mathbf{b}_i||^2$ can be precomputed, VQ calculation can be expressed as

$$\arg\{\min\{c_i - \mathbf{r}^t\mathbf{b}_i \mid i = 1, N\}\}, \tag{3}$$

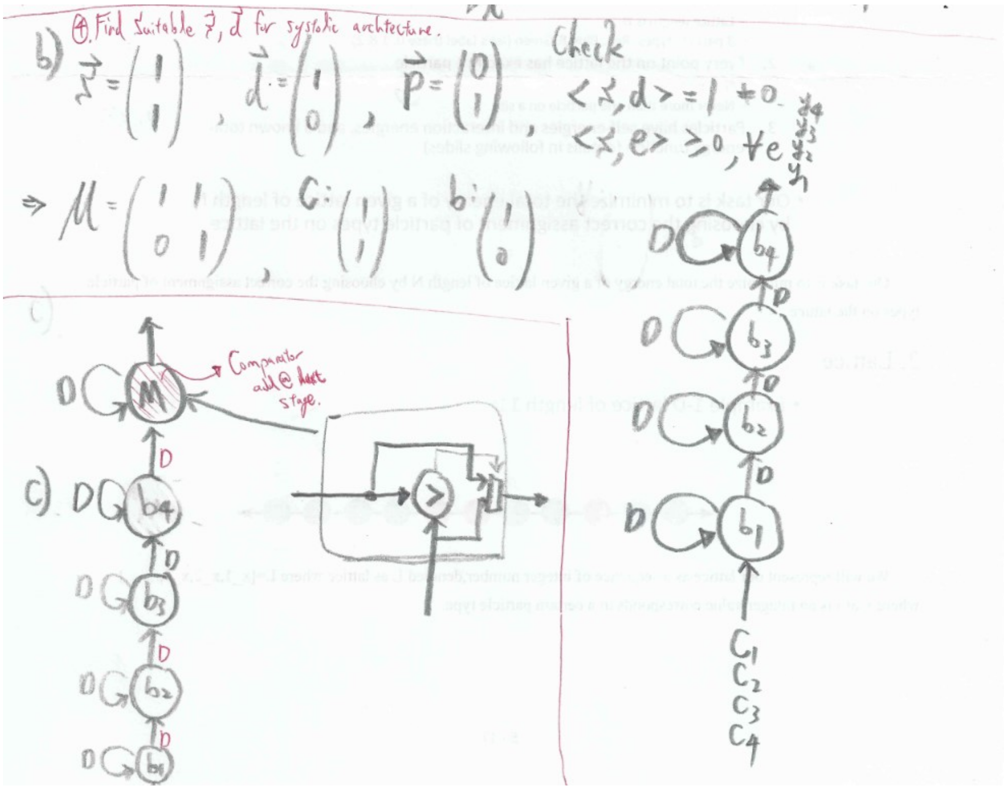
where $c_i = ||\mathbf{b}_i||^2/2$. In other words, VQ can be accomplished by calculating $c_i - \mathbf{r}^t\mathbf{b}_i$, for all $\mathbf{b}_i$'s in the codebook, and recording the one with the smallest distance. Note that $\mathbf{r}^t\mathbf{b}_i$ is an inner product operation, and c_i can be input from a pre-computed table.

(a) Please draw the DG of the $y(i) = c_i - \mathbf{r}^t\mathbf{b}_i$ for i = 1~N. For simplicity, assume the vector dimension k is 4. In each iteration, c_i and $\mathbf{b}_i$ are regarded as input and $y(i)$ is the output.



(b) Select a scheduling and projection scheme to obtain its systolic array design (at least one delay element in every inter-processor data link).

(c) Add a comparator module so that the vector index i corresponding to the minimum Euclidean distance can be obtained at the end of N iterations.


b) ④ Find suitable s, d for systolic architecture.
s = (1 1), d = (1 0), P = (0 1)
Check
<s, d> = 1 ≠ 0
<s, e> ≥ 0, ∀e
⇒ M = (1 1; 0 1), C: (1 1), b: (1 0)
y4 y3 y2 y1
D b4 D b3 D b2 D b1
C1 C2 C3 C4
c)
D M
Comparator add @ last stage.
D b4 D D b3 D D b2 D D b1
>
D

Note

1. Though when deriving the high level architecture for systolic array, only the edges needed to be used, however; the schedule of inputs and outputs are also important during implementation.
2. While transforming from algorithm to DG, beware of the indices, also try to check whether the indices you assigned make sense by going through some examples.

III. References

[1] VLSI DIGITAL SIGNAL PROCESSING SYSTEMS DESIGN AND IMPLEMENTATION, CH7 Systolic Archtiecture design, Prof. Keshab Parhi, p189~p212

[2] VLSI DIGITAL SIGNAL PROCESSING SYSTEMS DESIGN AND IMPLEMENTATION, CH7 Systolic Archtiecture design, exercises 11, Prof. Keshab Parhi, p215

[3] UMN EE-5329 VLSI Signal Processing Lecture-15 (Spring 2019),Systolic Architecture Design , Prof. Keshab Parhi

[4] UMN EE-5329 VLSI Signal Processing Lecture-16 (Spring 2019),Systolic Architecture Design, Space-Time Mapping , Prof. Keshab Parhi

[5] Computer Architecture - Lecture 27: Systolic Arrays (ETH Zürich, Fall 2020), Prof. Onur Mutlu