Department of Electrical and Computer Engineering  
University of Wisconsin – Madison  
ECE 734 VLSI Array Structures for Digital Signal Processing  

Homework #2 Solution  
Due Date: Friday, March 24, 2004

This homework consists of questions taken from the notes and open-ended questions. You must do the homework by yourself. **No collaborations are allowed**. There are total 100 points. This homework is worth 10% of your overall grades.

1. (5 points)  
(a) (3 points) Consider the following Fortran-like program  
```
DO I=1,N  
  DO J=2,N  
    A(I,J)=B(I,J)+C(I,J)  
    C(I,J)=D(I,J)/2  
    E(I,J)=A(I,J-1)**2+E(I,J-1)  
  END  
END  
```
Assume that each array has already been initialized before executing this program. Rewrite the program into a vectorized format.

**Answer:** Note the anti-dependence of 1st and 2nd statement. There is a true dependence relation with iteration dependence vector (0, 1) in the third statement. That is the only dependence vector in each iteration. Thus, all operations with the same I index can be vectorized. This leads to

```
Do J=2,N  
  Doall I=1,N  
    A(:,J)=B(:,J)+C(:,J)  
    C(:,J)=D(:,J)/2  
    E(:,J)=A(:,J-1)**2+E(:,J-1)  
  End Doall  
```

(b) (2 points) Consider the following Fortran-like program  
```
DO I=1,5  
  A(I)=B(I)+I  
  D(I)=A(I)+A(I+1)  
END  
```
Assume that arrays \{A(I)\} and \{B(I)\} have already been initialized before executing this program. Rewrite the program into a vectorized format.

**Answer:**

\[
\begin{align*}
A1(1:5) &= B(1:5) + [1 2 3 4 5] \\
D(1:5) &= A1(1:5) + A(2:6)
\end{align*}
\]

2. (10 points) Consider the following Fortran-like program  
```
C listing 1  
DO 10 I=1,3  
  DO 10 J=a*I+b, c*I+d  
  X(I,J)=X(I+2,J-1)+X(I-1,J)  
10  CONTINUE
```
(a) (4 points) let $a$, $b$, $c$, and $d$ be four constant integers. Find the condition(s) such that this nested loop is a regular nested loop.

**Answer:** Rewrite the index constraints in the formats of $\mathbf{p}_i \leq \mathbf{p}_0$, $\mathbf{q}_i \leq \mathbf{q}_0$:

$1 \leq i \leq 3; \quad ai + b \leq j \leq ci + d \Rightarrow$

$$
\mathbf{p}_i = \begin{bmatrix} 1 & 0 \\ a & -1 \end{bmatrix} \begin{bmatrix} i \end{bmatrix} \leq \begin{bmatrix} 3 \\ -b \end{bmatrix} = \mathbf{p}_0 \quad \text{and} \quad \mathbf{q}_i = \begin{bmatrix} 1 & 0 \\ c & -1 \end{bmatrix} \begin{bmatrix} i \end{bmatrix} \geq \begin{bmatrix} 1 \\ -d \end{bmatrix} = \mathbf{q}_0
$$

For a regular nested loop, it is required that $\mathbf{P} = \mathbf{Q}$. Hence, one must have $a = c$.

(b) (1 points) Derive the dependence matrix $\mathbf{D}$ of this algorithm.

**Answer:**

$$
\mathbf{D} = \begin{bmatrix} -2 & 1 \\ 1 & 0 \end{bmatrix}
$$

(c) (2 points) If $a = c = 0$, $b = 1$, and $d = 2$. In order to execute this algorithm, which elements of the $X$ array must be given as initial conditions?

**Answer:** The states to be executed are:

C listing 1

\[
\text{DO 10 } I=1,3 \\
\quad \text{DO 10 } J=1,2 \\
\quad \text{X(I,J)} = \text{X(I+2,J-1)} + \text{X(I-1,J)}
\]

10 CONTINUE

\[
\begin{align*}
\text{X(1,1)} &= \text{X(3,0)} + \text{X(0,1)} \\
\text{X(1,2)} &= \text{X(3,1)} + \text{X(0,2)} \\
\text{X(2,1)} &= \text{X(4,0)} + \text{X(1,1)} \\
\text{X(2,2)} &= \text{X(4,1)} + \text{X(1,2)} \\
\text{X(3,1)} &= \text{X(5,0)} + \text{X(2,1)} \\
\text{X(3,2)} &= \text{X(5,1)} + \text{X(2,2)}
\end{align*}
\]

The initial conditions are array elements that are marked with yellow shades.

(d) (3 points) This algorithm as specified in part (c) may have a problem executing properly. Discuss what may be the problem and provide a reformulated program that can be executed correctly. Assuming all the initial conditions of the array $X$ are available in the memory.

**Answer:** From the answer of part (c), the dependence dictates that the order of execution of this program is as follows:

$11 \rightarrow 21 \rightarrow 31 \rightarrow 12 \rightarrow 22 \rightarrow 32$

Clearly, the indices $I$ and $J$ must be interchanged. So the modified algorithm looks like this:

C listing 1

\[
\text{DO 10 } J=1,2 \\
\quad \text{DO 10 } I=1,3 \\
\quad \text{X(I,J)} = \text{X(I+2,J-1)} + \text{X(I-1,J)}
\]

10 CONTINUE


9. (4 points) Text book [Parhi], Chapter 3, problem 3. Hint: y(n) is the sum of two FIR filters.
11. (4 points) Text book [Parhi], Chapter 4, problem 1 (a), (b)
12. (6 points) Text book [Parhi], Chapter 4, problem 2.
13. (4 points) Text book [Parhi], Chapter 4, problem 3.
14. (6 points) Text book [Parhi], Chapter 4, problem 5.
15. (4 points) Text book [Parhi], Chapter 4, problem 7.
16. (6 points) Text book [Parhi], Chapter 4, problem 8.
17. (4 points) Text book [Parhi], Chapter 4, problem 10.
18. (8 points) Text book [Parhi], Chapter 4, problem 11.
19. (7 points) Refer to figure 4.20, page 115 of the text book [Parhi]. Assume the addition takes 2 t.u. and multiplication takes 5 t.u.
   (a) (2 points) Find the iteration bound $T_\infty$.
   Answer: $T_\infty = 5 + 2 + 2 = 9$ t.u.
   (b) (2 points) Identify the critical path $P_{cr}$. Give the answer in the form of $n_1 \rightarrow n_2 \rightarrow \ldots$ where $n_1, n_2$ are node numbers.
   Answer: $4 \rightarrow 2 \rightarrow 1 \rightarrow 7$ or $3 \rightarrow 2 \rightarrow 1 \rightarrow 7$. The length is 11 t.u.
   (c) (3 points) In order to reduce the clock cycle time to the iteration bound, retiming are performed. Among many possible solutions, some of them will NOT add delays to the computation of the output (i.e. $y(n)$ will be computed during the same clock cycle $x(n)$ is sampled), and will NOT increase (decrease is OK) the total number of registers. Find two such solutions by giving their corresponding retimed DFGs.
   Answer: $r(3) = r(4) = 1$ or $r(3) = r(4) = r(5) = r(6) = 1$

Solution for the problem in the text book

Chapter 2

These result are generated from file lpm.m and mcm.m on the course webpage
**Answer of Proc. 1 (b)**

```
>> ib_cyclemean(inv_Gd)
inv_Gd=
    Inf   0   Inf
    -7   Inf   -3
    -3   Inf   Inf

f0=
   [0   Inf   Inf]

f1=
   [Inf   0   Inf]

f2=
   [-7   Inf   -3]

f3=
   [-6   -7   Inf]

ans =
    3.5000
```

2(a)

```
L1=
   -1   0   -1   -1   -1   -1   -1   -1
   -1   -1   0   -1   -1   -1   -1   -1
   -1   -1   -1   0   -1   -1   -1   -1
   3   -1   -1   -1   6   -1   -1   -1
   -1   -1   -1   -1   -1   0   -1   -1
   -1   -1   -1   -1   -1   -1   0   -1
   -1   -1   -1   -1   -1   -1   -1   0
   4   -1   -1   -1   -1   -1   -1   -1

L2=
   -1   -1   0   -1   -1   -1   -1   -1
   -1   -1   -1   0   -1   -1   -1   -1
   3   -1   -1   -1   6   -1   -1   -1
   -1   3   -1   -1   -1   6   -1   -1
   -1   -1   -1   -1   -1   0   -1   -1
   -1   -1   -1   -1   -1   -1   0   -1
   4   -1   -1   -1   -1   7   -1   -1   -1
   -1   4   -1   -1   -1   7   -1   -1   -1

L3=
   -1   -1   -1   0   -1   -1   -1   -1
   3   -1   -1   -1   6   -1   -1   -1
   -1   3   -1   -1   -1   6   -1   -1
   -1   -1   3   -1   -1   -1   6   -1
   -1   -1   -1   -1   -1   -1   0   -1
   4   -1   -1   -1   -1   7   -1   -1   -1
   -1   4   -1   -1   -1   7   -1   -1   -1
   -1   -1   4   -1   -1   -1   7   -1   -1
```
\[
L_4 = 
\begin{array}{cccccccc}
3 & -1 & -1 & -1 & 6 & -1 & -1 & -1 \\
-1 & .3 & -1 & -1 & -1 & 6 & -1 & -1 \\
-1 & -1 & 3 & -1 & -1 & 6 & -1 & -1 \\
-1 & -1 & -1 & 3 & -1 & -1 & 6 & -1 \\
4 & -1 & -1 & -1 & 7 & -1 & -1 & -1 \\
-1 & 4 & -1 & -1 & -1 & 7 & -1 & -1 \\
-1 & -1 & 4 & -1 & -1 & 7 & -1 & -1 \\
-1 & -1 & -1 & 4 & -1 & -1 & 7 & -1 \\
\end{array}
\]

\[
L_5 = 
\begin{array}{cccccccc}
-1 & 3 & -1 & -1 & -1 & 6 & -1 & -1 \\
-1 & -1 & 3 & -1 & -1 & -1 & 6 & -1 \\
-1 & -1 & -1 & 3 & -1 & -1 & -1 & 6 \\
10 & -1 & -1 & -1 & 13 & -1 & -1 & -1 \\
-1 & 4 & -1 & -1 & -1 & 7 & -1 & -1 \\
-1 & -1 & 4 & -1 & -1 & 7 & -1 & -1 \\
-1 & -1 & -1 & 4 & -1 & -1 & 7 & -1 \\
11 & -1 & -1 & -1 & 14 & -1 & -1 & -1 \\
\end{array}
\]

\[
L_6 = 
\begin{array}{cccccccc}
-1 & -1 & 3 & -1 & -1 & -1 & 6 & -1 \\
-1 & -1 & -1 & 3 & -1 & -1 & -1 & 6 \\
10 & -1 & -1 & -1 & 13 & -1 & -1 & -1 \\
-1 & 10 & -1 & -1 & -1 & 13 & -1 & -1 \\
-1 & -1 & 4 & -1 & -1 & 7 & -1 & -1 \\
-1 & -1 & -1 & 4 & -1 & -1 & 7 & -1 \\
11 & -1 & -1 & -1 & 14 & -1 & -1 & -1 \\
-1 & 11 & -1 & -1 & -1 & 14 & -1 & -1 \\
\end{array}
\]

\[
L_7 = 
\begin{array}{cccccccc}
-1 & -1 & -1 & 3 & -1 & -1 & -1 & 6 \\
-1 & -1 & -1 & -1 & 13 & -1 & -1 & -1 \\
10 & -1 & -1 & -1 & 13 & -1 & -1 & -1 \\
-1 & 10 & -1 & -1 & -1 & 13 & -1 & -1 \\
-1 & -1 & 4 & -1 & -1 & 7 & -1 & -1 \\
-1 & -1 & -1 & 4 & -1 & -1 & 7 & -1 \\
11 & -1 & -1 & -1 & 14 & -1 & -1 & -1 \\
-1 & 11 & -1 & -1 & -1 & 14 & -1 & -1 \\
\end{array}
\]

\[
L_8 = 
\begin{array}{cccccccc}
10 & -1 & -1 & -1 & 13 & -1 & -1 & -1 \\
-1 & 10 & -1 & -1 & -1 & 13 & -1 & -1 \\
-1 & -1 & 10 & -1 & -1 & 13 & -1 & -1 \\
-1 & -1 & -1 & 10 & -1 & -1 & 13 & -1 \\
11 & -1 & -1 & -1 & 14 & -1 & -1 & -1 \\
-1 & 11 & -1 & -1 & -1 & 14 & -1 & -1 \\
-1 & -1 & 11 & -1 & -1 & 14 & -1 & -1 \\
-1 & -1 & -1 & 11 & -1 & -1 & 14 & -1 \\
\end{array}
\]

\[
\text{ans} = 1.7500
\]

2(b)

\[
\text{inv}_Gd = 
\begin{array}{cccccccc}
\text{Inf} & 0 & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & 0 & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & 0 & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \\
-3 & \text{Inf} & \text{Inf} & \text{Inf} & -6 & \text{Inf} & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & 0 & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & 0 & \text{Inf} \\
-4 & \text{Inf} & \text{Inf} & \text{Inf} & -7 & \text{Inf} & \text{Inf} & \text{Inf} \\
\end{array}
\]
\[ \begin{align*}
f_0 &= [0 \ Inf \ Inf \ Inf \ Inf \ Inf \ Inf \ Inf]^T, \\
f_1 &= [Inf \ 0 \ Inf \ Inf \ Inf \ Inf \ Inf \ Inf]^T, \\
f_2 &= [Inf \ Inf \ 0 \ Inf \ Inf \ Inf \ Inf \ Inf]^T, \\
f_3 &= [Inf \ Inf \ Inf \ 0 \ Inf \ Inf \ Inf \ Inf]^T, \\
f_4 &= [-3 \ Inf \ Inf \ Inf \ -6 \ Inf \ Inf \ Inf]^T, \\
f_5 &= [Inf \ -3 \ Inf \ Inf \ Inf \ -6 \ Inf \ Inf]^T, \\
f_6 &= [Inf \ Inf \ -3 \ Inf \ Inf \ Inf \ -6 \ Inf]^T, \\
f_7 &= [Inf \ Inf \ Inf \ -3 \ Inf \ Inf \ Inf \ -6]^T, \\
f_8 &= [-10 \ Inf \ Inf \ Inf \ -13 \ Inf \ Inf \ Inf]^T, \\
\text{ans} &= 1.7500
\end{align*} \]

3(a)

\[ \begin{align*}
L_1 &= \begin{bmatrix}
4 & 4 & -1 \\
-1 & -1 & 0 \\
4 & 4 & -1
\end{bmatrix}, \\
L_2 &= \begin{bmatrix}
8 & 8 & 4 \\
4 & 4 & -1 \\
8 & 8 & 4
\end{bmatrix}, \\
L_3 &= \begin{bmatrix}
12 & 12 & 8 \\
8 & 8 & 4 \\
12 & 12 & 8
\end{bmatrix}, \\
\text{ans} &= 4
\end{align*} \]
3(b)

\[ \text{inv}_{\text{Gd}} = \begin{bmatrix}
-4 & -4 & \text{Inf} \\
\text{Inf} & \text{Inf} & 0 \\
-4 & -4 & \text{Inf}
\end{bmatrix} \]

\[ f_0 = \begin{bmatrix}
0 & \text{Inf} & \text{Inf}
\end{bmatrix} \]

\[ f_1 = \begin{bmatrix}
-4 & -4 & \text{Inf}
\end{bmatrix} \]

\[ f_2 = \begin{bmatrix}
-5 & -5 & -4
\end{bmatrix} \]

\[ f_3 = \begin{bmatrix}
-12 & -12 & -8
\end{bmatrix} \]

\[ \text{ans } = 4 \]

4(a)

\[ L_1 = \begin{bmatrix}
4 & 0 & -1 \\
5 & -1 & 0 \\
5 & -1 & -1
\end{bmatrix} \]

\[ L_2 = \begin{bmatrix}
8 & 4 & 0 \\
9 & 5 & -1 \\
9 & 5 & -1
\end{bmatrix} \]

\[ L_3 = \begin{bmatrix}
12 & 8 & 4 \\
13 & 9 & 5 \\
13 & 9 & 5
\end{bmatrix} \]

\[ \text{ans } = 4 \]
4(b)

\[
\text{inv}_Gd = \\
\begin{bmatrix}
-4 & 0 & \text{Inf} \\
-5 & \text{Inf} & 0 \\
-5 & \text{Inf} & \text{Inf}
\end{bmatrix}
\]

\[f_0 = \\
\begin{bmatrix}
0 & \text{Inf} & \text{Inf}
\end{bmatrix}
\]

\[f_1 = \\
\begin{bmatrix}
-4 & 0 & \text{Inf}
\end{bmatrix}
\]

\[f_2 = \\
\begin{bmatrix}
-8 & -4 & 0
\end{bmatrix}
\]

\[f_3 = \\
\begin{bmatrix}
-12 & -8 & -4
\end{bmatrix}
\]

\[\text{ans} = 4\]

Chapter 3

1

(a) \(T_{\text{sample}} = 4T\)
\(f_{\text{sample}} = 1/4T\)

(b) The pipelining levels are shown by the dashed lines in Figure 3.1. 9 registers are required.

2.

(a). The critical path is \(M_1 - A_2 - M_2 - A_1 - M_3 - A_9 - A_4\) as shown by the dashed line in Figure 3.2(a).

(b). Pipelining latches are placed on the dotted lines in Figure 3.2(b). As can be seen, the critical path is one multiply and one add, which is 3 time units. ■
Let:

\[ y_1(n) = a_1x_1(n) + a_2x_1(n-1) + a_3x_1(n-2) + a_4x_1(n-3) + a_5x_1(n-4) \]  
(3.3)

\[ y_2(n) = b_1x_2(n) + b_2x_2(n-1) + b_3x_2(n-2) + b_4x_2(n-3) + b_5x_2(n-4) \]  
(3.4)

\[ y(n) = y_1(n) + y_2(n) \]  
(3.5)

Then transpose operation can be applied to \( y_1(n) \) and \( y_2(n) \) separately. The equivalent data-broadcast implementation is shown in Figure 3.3.
(a). The pipelining level is shown by dashed line in Figure 3.5(a).

(b). The block filter architecture is illustrated in Figure 3.5(b). \( f_{sample} = 3 \times T \).

\[
\begin{align*}
y(3k) &= ax(3k) + bx(3k - 1) + cx(3k - 2) \\
y(3k + 1) &= ax(3k + 1) - bx(3k - 1) + cx(3k - 2) \\
y(3k + 2) &= ax(3k + 2) - bx(3k) + cx(3k - 1)
\end{align*}
\]
Chapter 4

1. 
(a) 
\[ T_{\text{bound}} = \frac{T_m + 2T_a}{2} = 18\text{ns} \]

(b) 
\[ T_{\text{critical}} = 2(T_m + 3T_a) = 88\text{ns} \]

2. 
(a) The maximum sample rate is limited by the critical path:
\[ \text{SampleRate}_{\text{max}} = \frac{1}{T_{\text{critical}}} = \frac{1}{30} \quad (4.3) \]

(b) The fundamental limit on the sample period is determined by the iteration bound:
\[ \text{SamplePeriod}_{\text{limited}} = T_{\text{bound}} = \max\left[\frac{30}{2}, \frac{25}{1}\right] = 25 \quad (4.4) \]

(c) The answer is shown in Figure 4.2

![Figure 4.2](image)

Fig. 4.2 Answer of Exercise 2(c)

3. 
(a) By inspection, the iteration bound is 7/4 u.t.
(b) The critical path time of the circuit is 7 u.t. \( M_2 \to A_1 \to A_3 \to M_3 \to A_4 \).
(c) The retimed circuit is shown in Figure 4.3.

![Figure 4.3](image)

Fig. 4.3 Retimed data flow graph for Exercise 3.
5.

(a) $T_{\infty} = 4$  \hspace{1cm} (4.7)

$T_{\text{critical}} = 7$ \hspace{1cm} (4.8)

(b) The minimum achievable clock period obtained with pipelining and retiming is the iteration bound of the DFG, which equals to 4 u.t. in this problem.

![Retimed data flow graph for Exercise 5.](image)

7.

According to $w_r(u-v) = w(u-v) + r(v) - r(u)$, we get the retimed DFG as in Figure 4.8.

![Retimed DFG for Exercise 7.](image)

8. Refer to spbf.m and spfw.m on the course webpage.

(a) The constrained graph is shown in Figure 4.9. Using Bellman-Ford algorithm, one solution is given as $r_1 = r_2 = 0, r_3 = -1, r_4 = -2, r_5 = 0$.

(b) Using Floyd-Warshall algorithm, the matrix $R(6)$ is as

\[
\begin{array}{cccccc}
\text{Inf} & \text{Inf} & 1 & 0 & 2 & \text{Inf} \\
1 & \text{Inf} & 1 & 0 & 3 & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & -1 & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} & \text{Inf} \\
\text{Inf} & \text{Inf} & -1 & -2 & \text{Inf} & \text{Inf} \\
0 & 0 & -1 & -2 & 0 & 0
\end{array}
\]

We get the same answer as using Bellman algorithm: $r_1 = r_2 = 0, r_3 =$
10. The method to reduce the critical path by pipeline and retiming is shown in Figure 4.11.

![Diagram of retiming/pipeline](image)

**Fig. 4.11** Retiming/pipeline of Exercise 10.
11.

(a) Use cutset retiming, we have retimed DFG as in figure 4.12.

(b) First use 2-D slow down, and then apply cutset retiming. The hardware utilization efficiency of this system is 50%.