**Chapter 5 – Solutions to Instructor Reserve Problems**

**I-1 Instructor**

Push G1(s) to the left past the pickoff point.



Thus, 

**I-2 Instructor**

**a.** Split G3 and combine with G2 and G4. Also use feedback formula on G6 loop.



Push G2 +G3 to the left past the pickoff point.



Using the feedback formula and combining parallel blocks,



Multiplying the blocks of the forward path and applying the feedback formula,



**I-3 Instructor**

**a.** Push G7 to the left past the pickoff point. Add the parallel blocks, G3+G4.



Push G3+G4 to the right past the summing junction.



Collapse the minor loop feedback.



Push to the left past the pickoff point.



Push G1 to the right past the summing junction.



Add the parallel feedback paths to get the single negative feedback,

H(s) = + - . Thus,

T(s) *=* 

**b.**

**Program:**

G1=tf([0 1],[1 7]); %G1=1/s+7 input transducer

G2=tf([0 0 1],[1 2 3]); %G2=1/s^2+2s+3

G3=tf([0 1],[1 4]); %G3=1/s+4

G4=tf([0 1],[1 0]); %G4=1/s

G5=tf([0 5],[1 7]); %G5=5/s+7

G6=tf([0 0 1],[1 5 10]); %G6=1/s^2+5s+10

G7=tf([0 3],[1 2]); %G7=3/s+2

G8=tf([0 1],[1 6]); %G8=1/s+6

G9=tf([1],[1]); %Add G9=1 transducer at the input

T1=append(G1,G2,G3,G4,G5,G6,G7,G8,G9);

Q=[1 -2 -5 9

2 1 8 0

3 1 8 0

4 1 8 0

5 3 4 -6

6 7 0 0

7 3 4 -6

8 7 0 0];

inputs=9;

outputs=7;

Ts=connect(T1,Q,inputs,outputs);

T=tf(Ts)

**Computer response:**

Transfer function:

6 s^7 + 132 s^6 + 1176 s^5 + 5640 s^4 + 1.624e004 s^3

+ 2.857e004 s^2 + 2.988e004 s + 1.512e004

-----------------------------------------------------------

s^10 + 33 s^9 + 466 s^8 + 3720 s^7 + 1.867e004 s^6

+ 6.182e004 s^5 + 1.369e005 s^4 + 1.981e005 s^3

+ 1.729e005 s^2 + 6.737e004 s - 1.044e004

**I-4 Instructor**

Push G3 to the left past the pickoff point. Push G6 to the left past the pickoff point.



Hence,

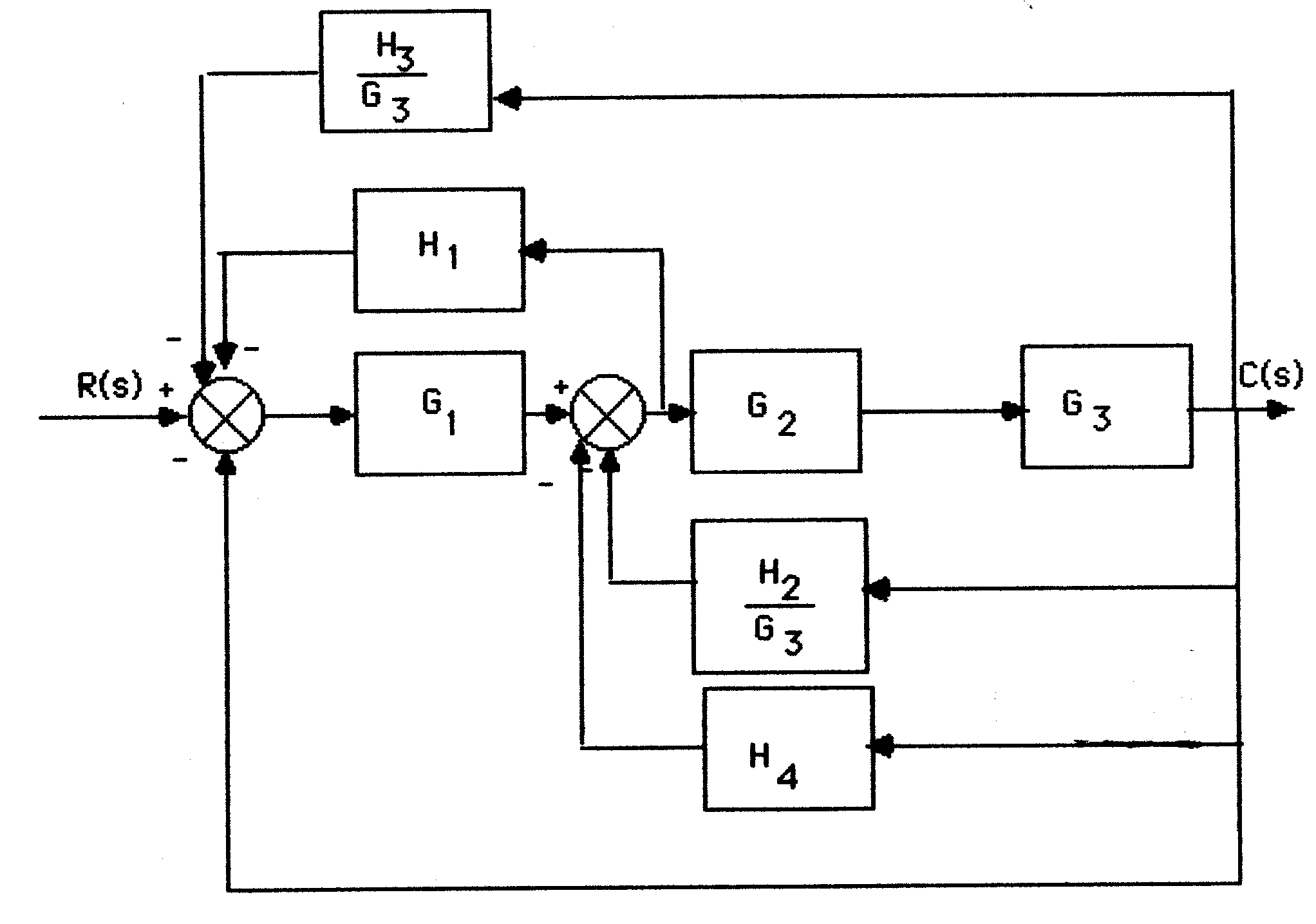


Thus the transfer function is the product of the functions, or

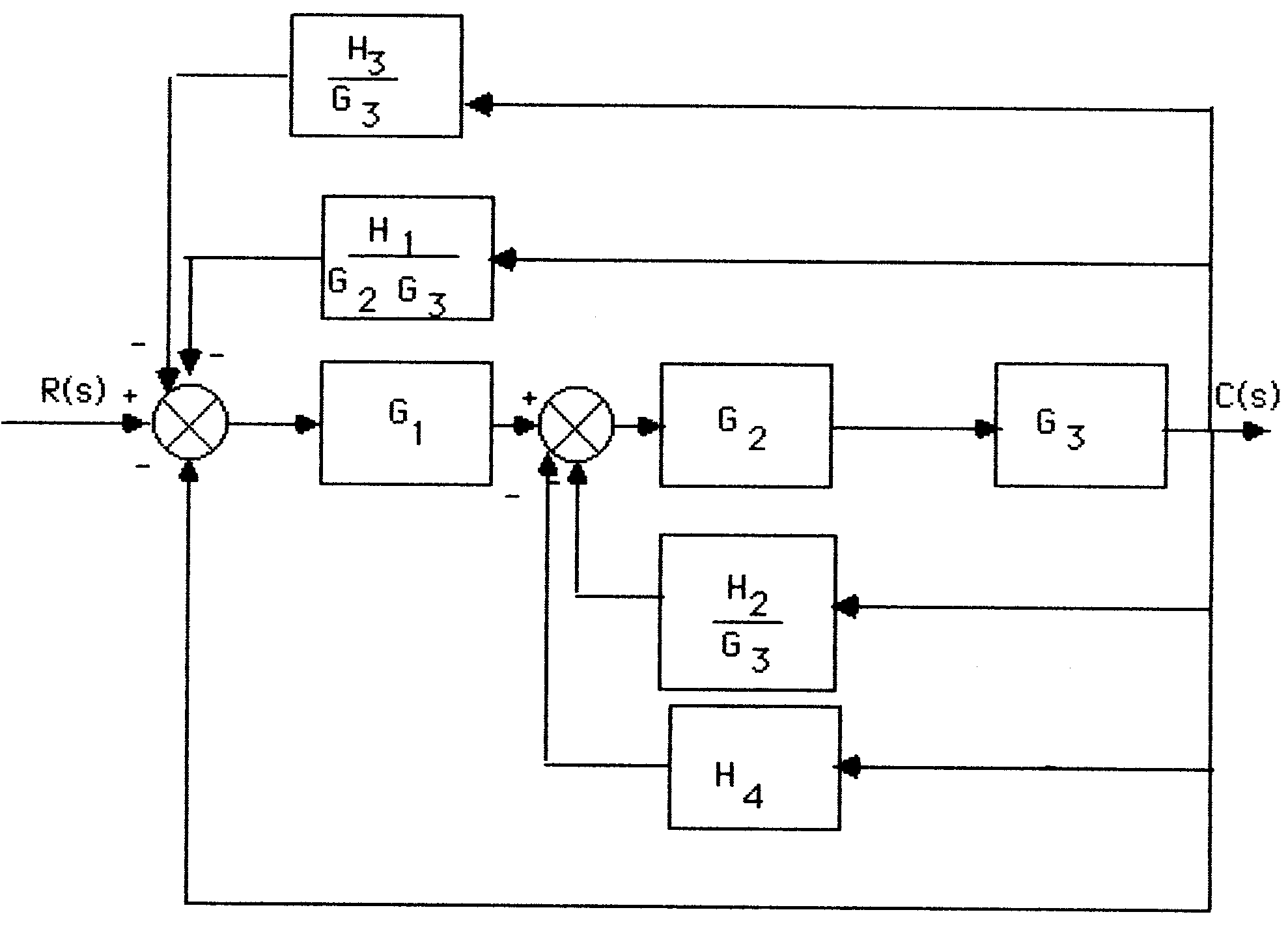
=

**I-5 Instructor**

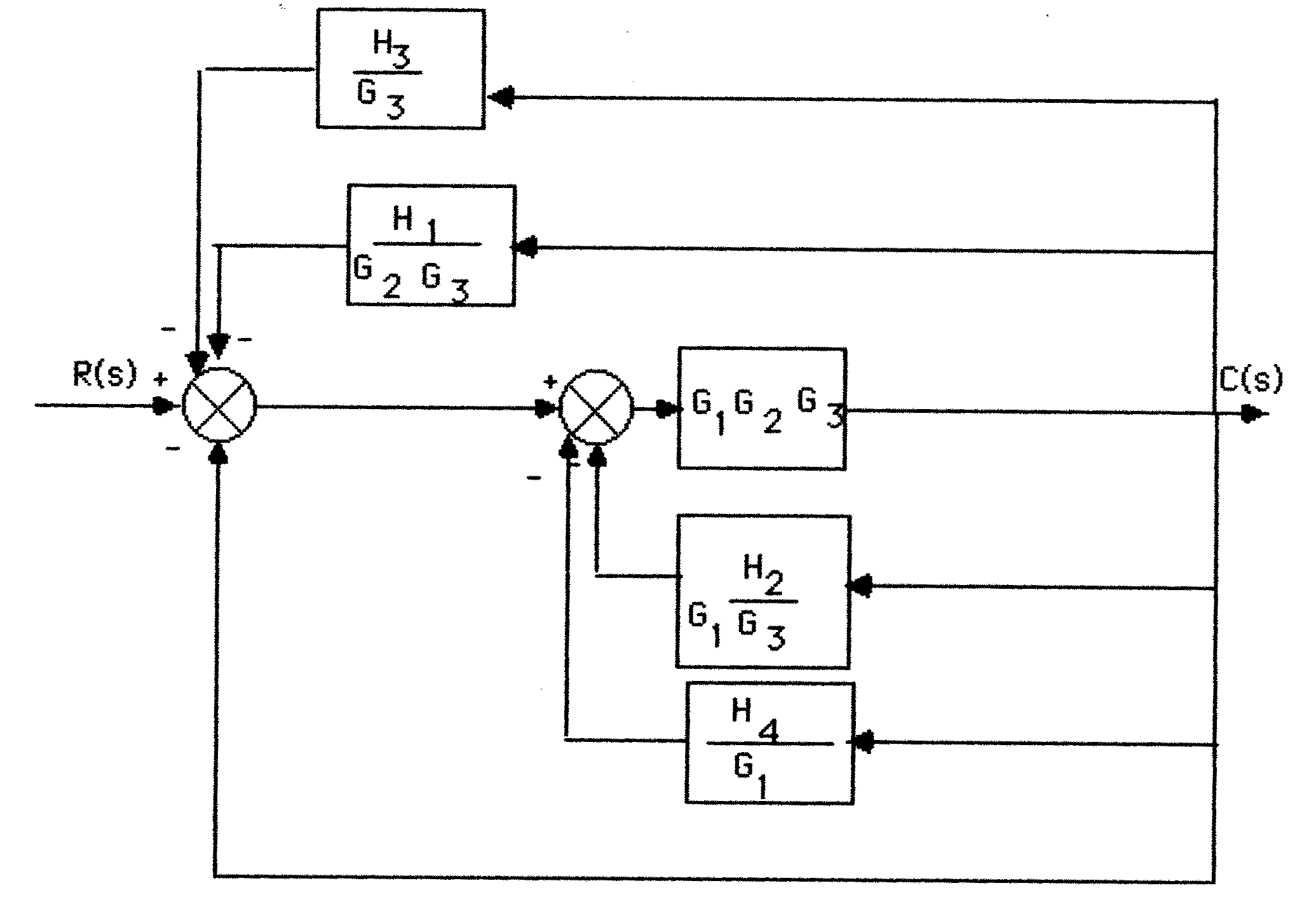
Push G3(s) to the left past the pickoff point.



Push G2(s)G3(s) to the left past the pickoff point.



Push G1(s) to the right past the summing junction.



Collapsing the summing junctions and adding the feedback transfer functions,



where



**I-6 Instructor**

The equivalent mechanical system is found by reflecting all mechanical impedances to the spring.



Writing the equations of motion:



Solving for **2(s),



The angular rotation of the pot is 0.25 that of **2, or



For the pot:



For the electrical network: Using voltage division,



Substituting the previously obtained values,



**I-7 Instructor**

**a.**

****



**b.**





**c.**

****



**I-8 Instructor**

Closed-loop gains: G2G4G6G7H3; G2G5G6G7H3; G3G4G6G7H3; G3G5G6G7H3; G6H1; G7H2

Forward-path gains: T­1 = G1G2G4G6G7; T2 = G1G2G5G6G7; T3 = G1G3G4G6G7; T4 = G1G3G5G6G7

Nontouching loops 2 at a time: G6H1G7H2

 = 1 - [H3G6G7(G2G4 + G2G5 + G3G4 + G3G5) + G6H1 + G7H2] + [G6H1G7H2]

1 = 2 = 3 = 4 = 1

T(s) =

=

**I-9 Instructor**

T(s) = =

­­­

**I-10 Instructor**

**Program:**

numg=8;

deng=poly([0 -8 -10]);

G=tf(numg,deng);

T=feedback(G,1);

[numt,dent]=tfdata(T,'v');

[A,B,C,D]=tf2ss(numt,dent); %Obtain controller canonical form

'(a)'; %Display label

A=rot90(A,2); %Convert to phase-variable form

A=fliplr(A); %Convert to phase-variable form

B=rot90(B,2); %Convert to phase-variable form

C=fliplr(C); %Convert to phase-variable form

'(b)'; %Display label

[a,b,c,d]=canon(A,B,C,D); %Convert to parallel form

**Computer response:**

T =

8

-----------------------

s^3 + 18 s^2 + 80 s + 8

Continuous-time transfer function.

numt =

0 0 0 8

dent =

1 18 80 8

ans =

(a)

A =

0 1 0

1 0 0

-18 -80 -8

B =

0

0

1

C =

8 0 0

ans =

(b)

a =

-8 0 0

0 1 0

0 0 -1

b =

0.0625

0

0

c =

0 0.0442 0.0442

d =

0

**I-11 Instructor**

1 = **A1x1** + **B1**r (1)

y1 = **C1x1** (2)

2 = **A2x2** + **B2**r (3)

y2 = **C2x2** (4)

In vector-matrix form,





**I-12 Instructor**



**I-13 Instructor**

Eigenvalues are -1, -2, and -3 since,

|**** - A | = (+ 3) (+ 2) (+ 1)

Solving for the eigenvectors, **Ax** = **x**

or,







For  = -1, x2 = 0, x1 = x3 . For  = -2, x1 = x2 = . For  = -3, x1 = - , x2 = x3 . Thus,

**= P**-1**APz** + **P**-1**B**u; y = **CPz**, where



**I-14 Instructor**

**Program:**

numg1=-0.125\*[1 0.435]

deng1=conv([1 1.23],[1 0.226 0.0169])

'G1'

G1=tf(numg1,deng1)

'G2'

G2=tf(2,[1 2])

G3=-1

'H1'

H1=tf([-1 0],1)

'Inner Loop'

Ge=feedback(G1\*G2,H1)

'Closed-Loop'

T=feedback(G3\*Ge,1)

**Computer response:**

numg1 =

-0.1250 -0.0544

deng1 =

1.0000 1.4560 0.2949 0.0208

ans =

G1

Transfer function:

-0.125 s - 0.05438

------------------------------------

s^3 + 1.456 s^2 + 0.2949 s + 0.02079

ans =

G2

Transfer function:

2

-----

s + 2

G3 =

-1

ans =

H1

Transfer function:

-s

ans =

Inner Loop

Transfer function:

-0.25 s - 0.1088

------------------------------------------------

s^4 + 3.456 s^3 + 3.457 s^2 + 0.7193 s + 0.04157

ans =

Closed-Loop

Transfer function:

0.25 s + 0.1088

-----------------------------------------------

s^4 + 3.456 s^3 + 3.457 s^2 + 0.9693 s + 0.1503

**I-15 Instructor**

**a.** Since *VL(s)* = *Vg(s)* – *VR(s)*, the summing junction has *Vg(s)* as the positive input and *VR(s)* as the negative input, and *VL(s)* as the error. Since *I(s)* = *VL(s)* (1*/(Ls)*), *G(s)* = 1*/(Ls).* Also, since *VR(s)* = *I(s)R*, the feedback is *H(s)* = *R*. Summarizing, the circuit can be modeled as a negative feedback system, where *G(s)* = 1*/(Ls*), *H(s*) = *R*, input = *Vg(s*), output = *I(s)*, and error = *VL(s),* where the negative input to the summing junction is *VR(s).*

**b. .** Hence, **.**

**c.** Using circuit analysis, .

**I-16 Instructor**

1. Adding currents at the op-amp’s inverting terminal, under ideal condition we get

 which after some algebraic manipulations gives 

Also from the circuits diagram 

1. These equations can be represented by the following block diagram

+

-

v1

+

- A





vo

vi

+

+

vi

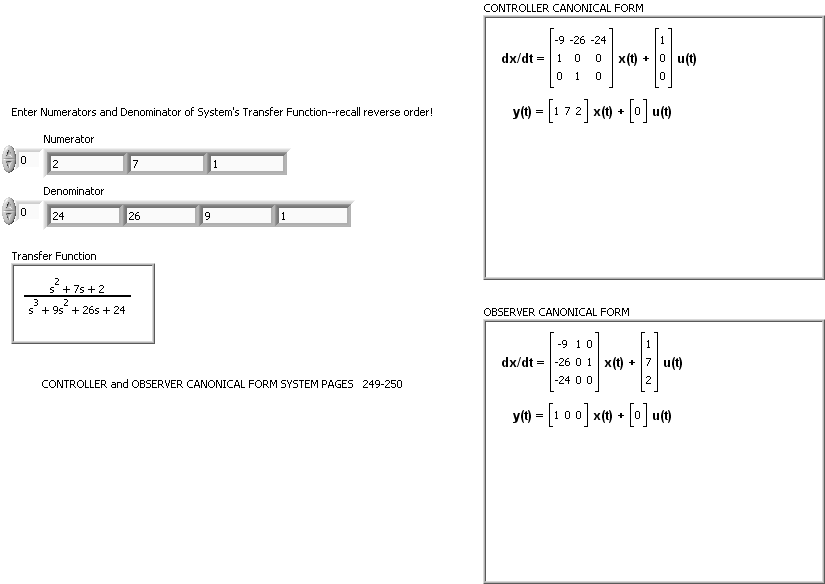
Vo

We have that ; ; ; 

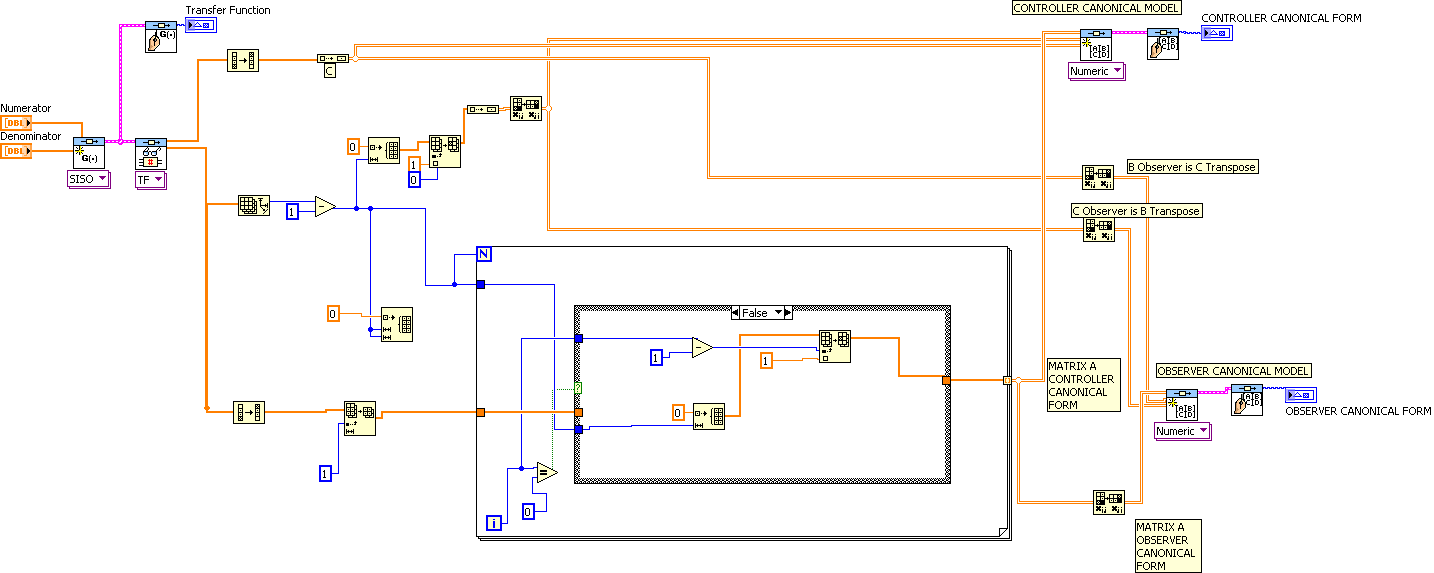


**c.**  

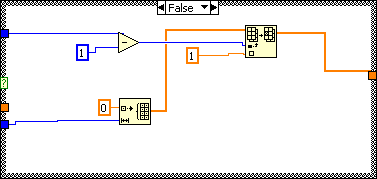
**I-17 Instructor**

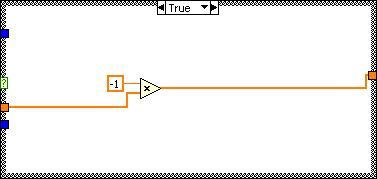


Block Diagram:

Case

Structure Details:

****

****

**I-18 Instructor**

1. There are three forward paths:

The loops are:

There are no non-touching loops.

and

1. There are three forward paths:

The loops are:

There are no non-touching loops.

and

The loops and are the same as in part a. There is only one forward path   
 and



=-

**I-19 Instructor**

1. Following the procedure described in Chapter 3 we define  and . In time domain  . and we also define . These equations give





1. The eigenvalues can be obtained directly from the transfer function poles. Thus 

Consider , the first eigenvector is found from the solution of  or . This results in . Arbitrarily let  so the first eigenvector is .

Similarly for ;  or  resulting in . Letting arbitrarily  the second eigenvector is .

1. The similarity transformation matrix is 



The matrices for the diagonalized are calculated as follows:







The diagonalized representation is:





**SOLUTIONS TO DESIGN PROBLEMS**

**I-20 Instructor**

**a.**

T(s) = ; from which, 2n = 1 and n = 5. Hence,  = 0.1. Therefore,

; Ts = = 8.

**b.**

T(s) = ; from which, 2n = 1+25K2 and n = 5.

Hence, 

Also, Ts = = 0.25, Thus, n = 16; from which K2 =  and

n = 35.09. Hence, K1 = 49.25.

**I-21 Instructor**

1. The transfer function derived for that system was:

**, where:

; ; .

or

**

 and . From Ts = 10 sec = 4/n, we find n = 2.

1. In Figure P5.58, *KG(s)* = ; and *T(s)* = ,

where 2n = 0.8; 2n = 1.6

This results in: n = 1.265 rad/sec &  = 0.32; % O.S. = 33% and Ts = 10 sec.

**I-22 Instructor**

1. The block diagram of this system is shown below.



1. The transfer function of *GP(s)* of the circuit representing the process may be derived as shown in example 2.6. That yields (see equation 2.66):



Hence:  and for a critically damped process (*ζ* =1), we have:



1. In the circuit representing the final control element:

; 



The overall system transfer function is given by:



Given that  and , the dominant poles should be:

, which are the roots of .

Thus, the third closed-loop pole may be found by dividing the characteristic polynomial by , which gives .

This third pole is non-dominant for *KP* > 2.6.

Given that its tolerance is ± 10%, I would set the controller potentiometer, *RF*, to:

