UNIVERSITY OF CALIFORNIA
College of Engineering
Electrical Engineering and Computer Sciences Department

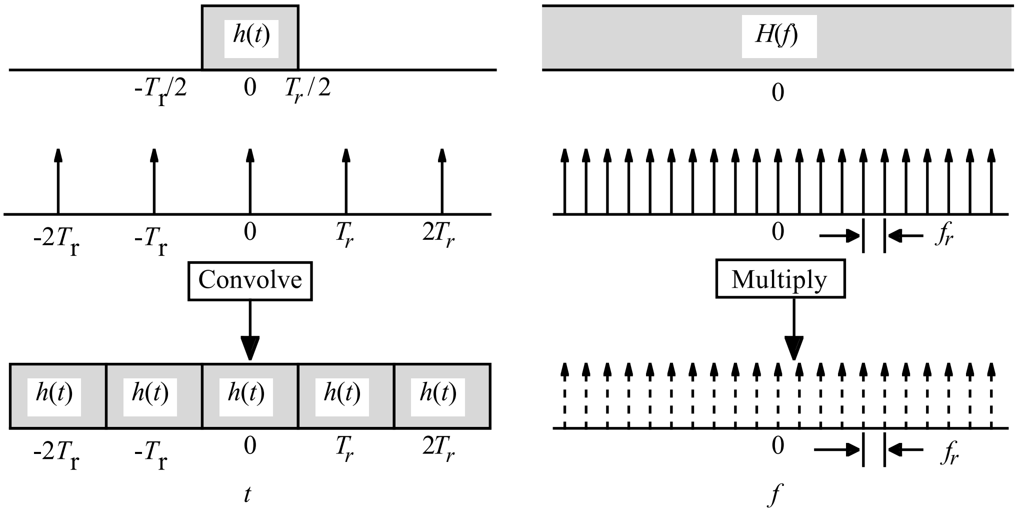
145M Microcomputer Interfacing Lab

Final Exam Solutions May 15, 2009

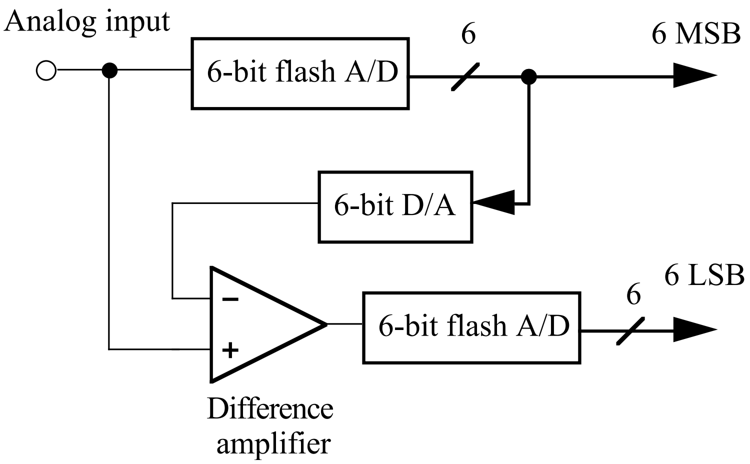
1.1 Fourier Convolution Theorem: The Fourier transform of the convolution of two functions is the product of their Fourier transforms (Textbook p 391, lecture slide #218).

[3 points off for stating the Fourier Frequency Convolution Theorem]

1.2 A periodic waveform is the convolution of a waveform with an infinite train of delta functions. The Fourier transform of the latter is an infinite train of delta functions in frequency. By the Fourier convolution theorem, the Fourier transform of a periodic waveform is the product of its Fourier transform with an infinite train of delta functions in frequency. The resulting Fourier transform has non-zero values only at discrete frequencies (Textbook Figure 5.26, lecture slide #224).



2.1



S/H amplifier or holding capacitors were not required for full credit

2.2

1 Use one 6-bit flash A/D converter to determine the 6 most significant bits

2 Use the 6-bit D/A converter to convert the 6 most significant bits into an analog voltage

3 Use a difference amplifier to subtract the 6-bit D/A output from the analog input

4 Use the second 6-bit flash A/D converter to convert the analog difference from step 3 into the 6 least significant bits

2.3 The 6-bit A/D converter that determines the 6 most significant bits and the 6-bit D/A converter that converts these into an analog voltage must be so accurate that the output of the difference amplifier is accurate to 1/8 step size out of 2^{12} = 4096 steps or one part in 32k. Since their accuracy is determined by the accuracy of their resistors, a resistor accuracy of 1 part in 32k would be safe.

Here is an example: Imagine that the first stage has 64 steps each 64 mV wide for a total of 4096 mV and the second stage has 64 steps each 1 mV wide. Since the entire 12-bit converter has 1 mV steps and an absolute accuracy of 1/8 LSB, the transition voltages of the first stage must be accurate to 1/8 mV. As a worst case, imagine that the first 32 resistors of the first stage have resistance R + e and the second 32 resistors have resistance R – e, where e is a resistance error. The voltage at the center point of the resistor string of the first stage would then be 4096 mV [32 (R + e) / 64R], which would be 2048 mV plus a error of 2048 mV (e/R) < 1/8 mV. This means that e/R < 1/16,000 and the resistors must be accurate to 1 part in 16k. Note that statistical analysis does not apply if the errors are systematic.

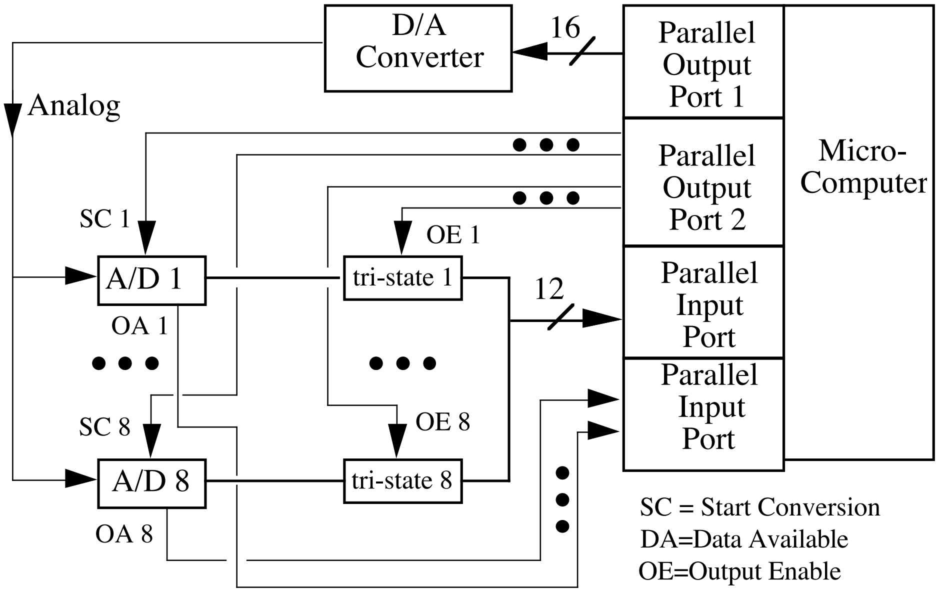
The 6-bit A/D converter that determines the 6 least significant bits only needs to have the accuracy of a 6-bit converter, so the resistors need an accuracy of one part in 2^9, or one part in 512.

[Full credit for resistor accuracy 1 part in 32k or 1 part in 16k]

[3 points off for 1 part in 1,000] [4 points off for 1 part in 100]

[6 points off for expressing resistor accuracy in units of LSB- this will not be meaningful to a resistor manufacturer]

3.1



The following are essential [3 points off for each omitted]:

- Connect the analog output of the D/A to the analog inputs of all A/Ds

Note: S/H amplifiers are not needed since the D/A can provide the hold

- Connect the 12 bit A/Ds to separate tri-state drivers
- Connect the outputs of the tri-state drivers together to form a data bus (can't input 8 x 12 bits in parallel)
- Connect the data bus to 12 bits of the parallel input port
- Provide 8 separate output port lines for initiating conversion of the 8 A/Ds (could be combined with next signal)
- Provide 8 separate output port lines for enabling the 8 separate tri-state drivers (this was essential)
- Provide input for 8 separate input port lines to indicate when individual A/Ds have completed conversion

3.2 The steps needed to measure the first transition voltage V(0,1) for the first A/D converter.

1 set SC1 low, disable all tri-state drivers, set N = 0 (16 bits)
2 Put N on D/A
3 wait 10μs until D/A has settled [using wait(10)]
4 Put low-high edge on SC1 output line to start conversion
5 Wait until output data available
6 enable tri-state 1 (disable all others)
7 read input port to get value M
8 Put high-low edge on SC1 output port to end conversion cycle

9 if M=0, increase N by one and loop back to step 2

10 If M=1, save (D/A voltage step)(N-1/2) as the transition voltage

[1 points off for not starting the D/A input at zero]

[2 points off for not incrementing the D/A input until a A/D transition occurs]

[2 points off for not waiting for D/A to settle]

[2 points off for not waiting for A/D output data available signal; you do not know the conversion time]

[2 points off for not enabling tri-state #1 while putting all others in high-impedance mode]

3.3 Send successive 16-bit numbers 0 to 2^{16}–1 to the D/A converter and convert the analog output with the A/D. Whenever the A/D output value changes, store the corresponding D/A value in a transition voltage table

Determine the D/A values corresponding to first and last A/D transition voltages, and the equation of the line that passes through them. **Linearity** is a measure of how closely the other measured transition values pass through the line.

[2 points off for determining maximum differential linearity or maximum absolute accuracy]

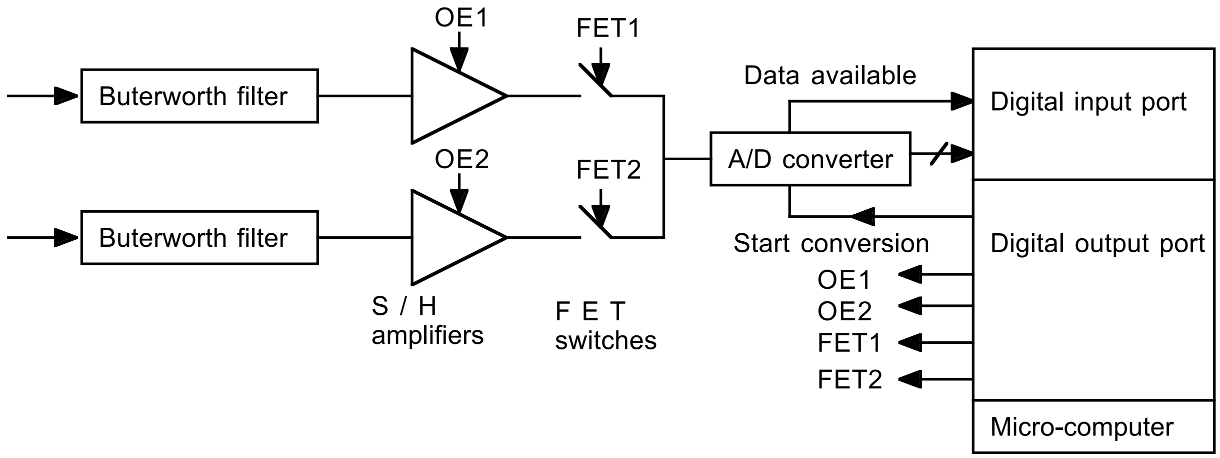
[3 points off for using the maximum deviation between the measured and ideal A/D output, which can never be better than ½ LSB. The linearity error for the A/D is the difference between the measured and ideal transition voltages, which can be much better than ½ LSB]

3.4 The method can determine the A/D accuracies to 1/16 LSB (±1/32 LSB was OK).

Note that 1 A/D LSB = 16 D/A LSBs.

[5 points off for an answer of 1/2 or 1 A/D LSB]

4.1



[3 points off if Butterworth is after the S/H- the Nyquist theorem requires LPF before sampling]

[2 points off for using tri-states (which has digital input and output) rather than an FET switches (which cannot be called a tri-state because it does not have three discrete output states)]

4.2 Gain = 0.99 at f1/fc = 0.784 f1 = 0.784 * 25 kHz = 19.6 kHz

4.3 Gain = 0.001 at f2/fc = 2.371 f2 = 25 kHz * 2.371 = 59.3 kHz

4.4 minimum fs = f1 + f2 = 78.9 kHz

[2 points off for minimum fs = 2 f2]

4.5

```
1   Read the system timer to get the current tick count $n$ in ms and set data storage index $i = 0$.
2   open both FET switches and set both S/H to sample mode
3   wait until tick count = n + 10i
4   bring both S/H to hold simultaneously
5   close FET switch 1
6   start conversion high
7   when DA high then read input port
8   start conversion low
9   open switch 1, close switch 2
10  start conversion high
11  when DA high then read input port
12  start conversion low
13  open switch 2
14  i = i + 1
15  loop back to step 3
```

[3 points off for not waiting for every 10th tick count to sample at an accurate 100 kHz]

[3 points off for not setting both S/Hs to hold mode simultaneously]

5.1

Standard	Wi-Fi™ 802.11b	Bluetooth™ 802.15.1	ZigBee® 802.15.4
Battery Life (days)	0.5 – 5.0 (short)	1 – 7 (medium)	100 – 1,000 (long)

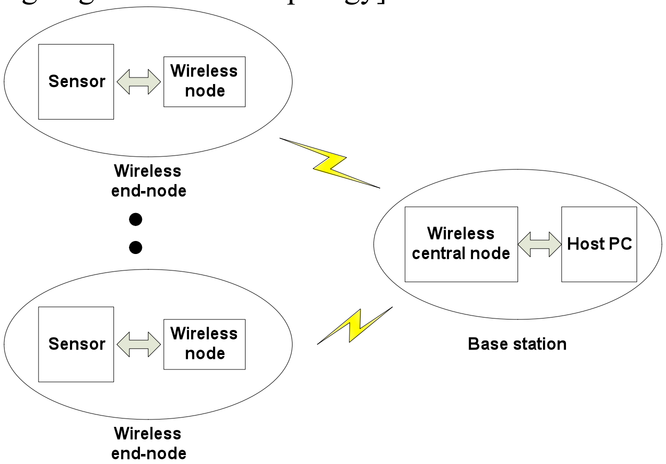
Network Size (# of nodes)	32 (medium)	7 (small)	> 64,000 (big)
Data rate (kb/s)	11,000 (high)	720 (medium)	20 – 250 (low)

5.2

Standard	Wi-Fi™ 802.11b	Bluetooth™ 802.15.1	ZigBee® 802.15.4
Application	Web, Email, Video	Cable Replacement	Monitoring & Control

5.3

We can adapt the simplest star topology. Mesh or tree topology is also ok.
[4 points off for not designing the network topology]



(a) Data rate between the sensor and the wireless end node: 16 bits/second (or 2bytes/second);
The wireless end node should be able to control/monitor the ADC sampling of the sensor circuits and fetch the 16 bits data from the sensor circuits.
One possible design is to use 18 digital IOs to interface the sensor and the wireless end node:
(i)16 bit digital input to wireless end node (data lines)
(ii)1 bit digital output from wireless end node (start of conversion)
(iii) 1 bit digital input to wireless end node (end of conversion)

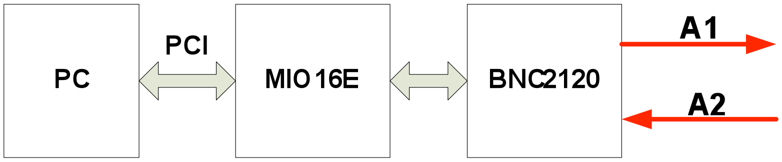
[3 points off for not designing the appropriate interface between the sensor and the wireless end node]

(b) Data rate between the central node and end node: 16 bits/second (or 2bytes/second);
Both Wi-Fi and Bluetooth are not suitable for this application. None of them can support 256 nodes. ZigBee is the most suitable technology because it supports more than 256 nodes and has very low power consumption.
[4 points off for selecting Wi-Fi or Bluetooth]

(c) Data rate between the central node and host PC: 16*256 bit/second (or 4kbps).
Any interface protocol with a data rate higher than 4kps (e.g. USB, RS232 interface with a baud rate higher than 9600bps, and etc.) is suitable. If we choose a parallel digital bus to transfer data

from the central node to the host PC, we will need three types of wires for data, status and control signal respectively.
[4 points off for not designing the appropriate interface between the central node and host PC]

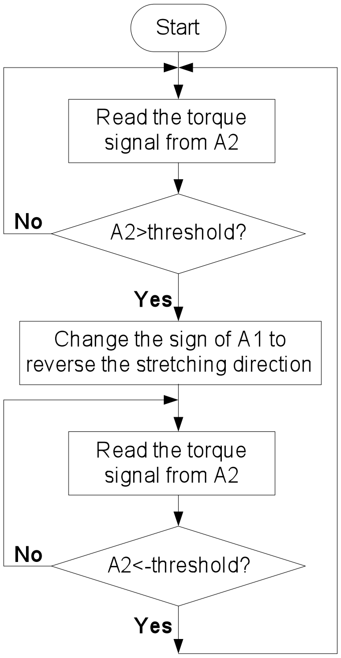
6.1



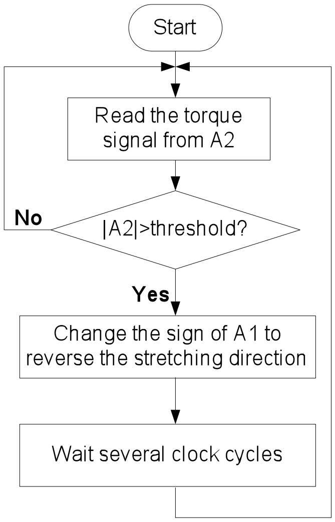
[3 points off for using a system other than the experimental platform]

6.2

Solution 1:


Start
Read the torque signal from A2
A2>threshold?
No
Yes
Change the sign of A1 to reverse the stretching direction
Read the torque signal from A2
A2<-threshold?
No
Yes

Solution 2:


Start
Read the torque signal from A2
|A2|>threshold?
No
Yes
Change the sign of A1 to reverse the stretching direction
Wait several clock cycles

Note: The robotic arm has inertia and therefore won't reverse the direction of stretching immediately after you reverse the sign of A1. The absolute value of A2 will keep increasing until the robotic arm starts rotating reversely. So, we need to wait several clock cycles before we check absolute value of A2 again.
[3 points off if "several clock cycles waiting" was not included in solution2]

6.3
Our current experimental platform is not a real time system. If glitches happen when the torque reaches the preset limit, the controller's response (to reverse the direction of stretching) will be delayed. The robotic device which is running in the velocity control mode will keep the original angular velocity. That could lead to over-stretching or even hurt the patient.
[3 points off if you didn't know the major potential risk is caused by the non-real-time controller]

Any real time system (DOS, RT Linux, Target PC, stand-alone micro-controller and etc.) will reduce the risk.
[1 point off if giving solutions like "decreasing the maximum angular velocity", or "adding mechanical/electronic safety switches/stops".]

145M Final Exam Grades:

Problem	1	2	3	4	5	6	Total
Average	12.6	33.1	38.5	41.8	24.6	17.1	167.8
rms	4.7	5.4	4.5	2.9	4.8	2.9	16.1
Maximum	15	40	45	45	30	25	200

145M Numerical Grades:

	Short labs	Long labs	Lab Partic.	Midterm #1	Midterm #2	Final	**Total**
Average	88.9	373.4	100	83.9	77.1	167.8	**891.1**
rms	14.2	50.3	0	9.2	10.5	16.1	**81.5**

Maximum	100	400	100	100	100	200	**1000**

145M Letter Grade Distribution

Letter Grade	Course Totals (1000 max)
A+	958
A	934, 931, 926*, 924, 922, 922*
A–	916*, 914*, 913, 904
B+	
B	854
B-	824
C+	
C	
C–	
D+	
D	
D–	633

* Graduate student