ECE4703 Final Exam

Your Name: SOLUTION Your box #: _____

December 16, 2010

Tips:

- Look over all of the questions before starting.
- Budget your time to allow yourself enough time to work on each question.
- Write neatly and show your work!
- This exam is worth a total of 200 points.
- Attach your "cheat sheet" to the exam when you hand it in.

problem 1	problem 2	problem 3	problem 4	total final exam score
40 points	40 points	60 points	60 points	200 points

1. 40 points. Suppose you write a sorting function that takes an array x of N unsorted floating point values and copies the elements to another array y sorted in ascending order. Your sorting algorithm works as follows:

   (a) Set $n = 0$.

   (b) Search through the whole array $x[0], \ldots, x[N-1]$ to find the minimum element. Denote the index of this minimum element as m where $m \in \{0, \ldots, N-1\}$.

   (c) Copy $x[m]$ to $y[n]$

   (d) Set $x[m] = \infty$.

   (e) Increment n.

   (f) If $n < N$, branch to step (b), otherwise end.

   Note that there are no multiplies or explicit additions/subtractions in this algorithm. What is a reasonable definition for an *operation* in this case? Using this definition, determine the asymptotic complexity of this sorting algorithm.

A reasonable definition for "operation" in this case is comparisons.

In step (b), we need to perform N-1 comparisons to find the minimum element of x.

In step (f), we need to perform one more comparison.

Steps (b) - (f) are repeated N times.

Hence, there are a total of N^2 comparisons required by this sort algorithm and the asymptotic complexity is $\Theta(N^2)$.

2. 40 points. Suppose you have an array x that contains four complex numbers and you wish to perform a 8-point FFT on this array by padding it with four zeros and then performing the 8-point FFT. In other words, you form a new 8-element complex array *xpad* such that

$$xpad[k] = \begin{cases} x[k] & k \in \{0,1,2,3\} \\ 0 & k \in \{4,5,6,7\} \end{cases}$$

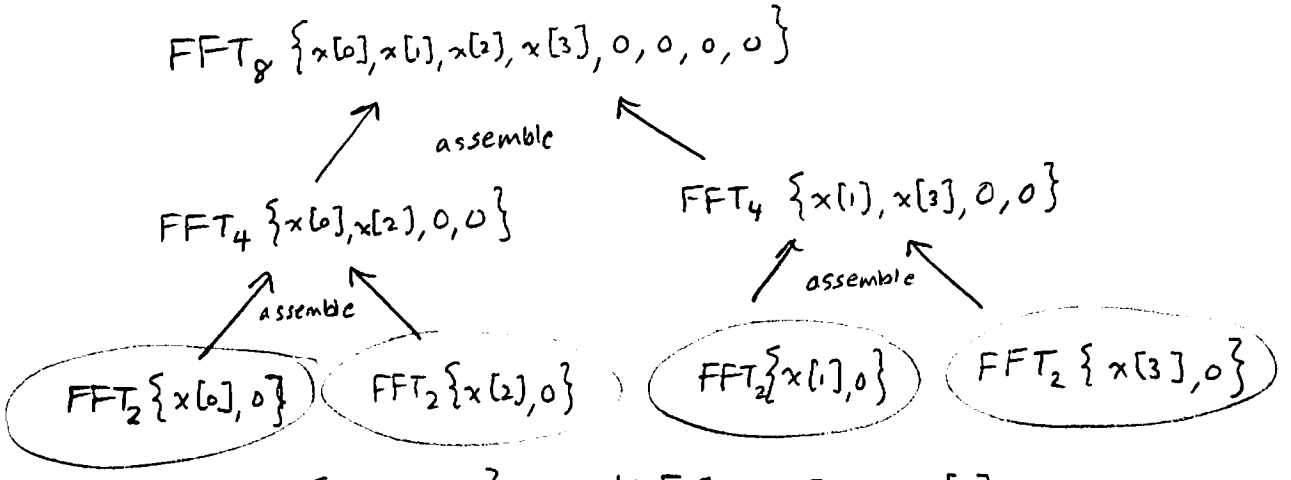
and perform the 8-point FFT on the *xpad* array. Recall that the number of multiply+accumulate operations in a normal 8-point FFT is $8\log_2(8) = 24$. But, in this case, four of the inputs to the FFT are zero. **If we do not count "multiply by zero" or "add zero" as operations, how many operations does the 8-point FFT require in this case?**

Hint: Recall that the N-point FFT is computed by dividing the FFT into an $N/2$-point FFT of the even part and an $N/2$-point FFT of the odd part. This "divide-and-conquer" approach is repeated until $N = 2$. Determine the number of operations in the 2-point FFT and then work your way back to the 8-point FFT.

$xpad = \{x[0], x[1], x[2], x[3], 0, 0, 0, 0\}$

$XPAD = FFT_8\{xpad\}$ and $XPAD[k] = XPAD_{even}[k] + e^{-j2\pi k/8} XPAD_{odd}[k]$

to perfom the FFT_8, we split it as follows:



Look at $FFT_2\{z[0], z[1]\}$: $Z[0] = z[0] + z[1]$

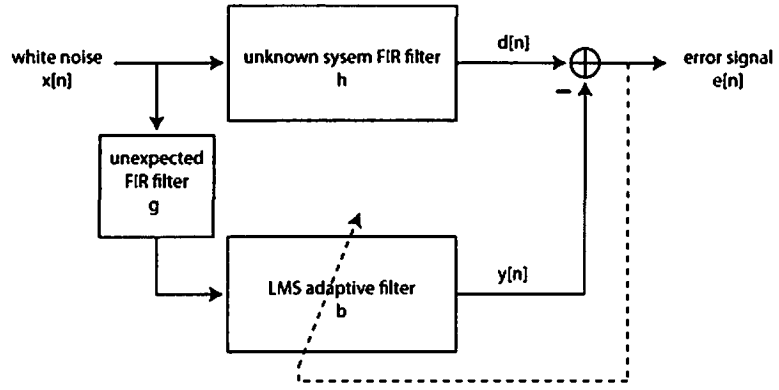
$Z[1] = z[0] - z[1]$

In every FFT_2 circled above, the second element ($z[1]$) is a zero.

Hence $FFT_2\{z[0], 0\}$: $Z[0] = Z[1] = z[0]$ ← no operations to compute FFT_2

- Assembling the 4-point FFT from the 2-point FFTs: 4 operations
- Assembling the 8-point FFT from the 4-point FFTs: 8 operations
- Hence, the total number of operations required is **12 operations**, which is more than the 4-point FFT, but less than the 8-point FFT.

3. 60 points total. Consider the system identification adaptive filtering system shown below.



For the following questions, assume that

- $\mathrm{E}\{x^2[n]\} = \sigma_x^2$,
- the unknown system filter coefficients are $\boldsymbol{h} = [1, 2, 1]$,
- the LMS adaptive filter $\boldsymbol{b}$ coefficients are initialized to zero prior to adaptation, and
- the LMS step-size is small enough to allow for convergence of the algorithm to the minimum mean squared error (MMSE) solution.

(a) 20 points. Suppose that the "unexpected" FIR filter $g = 1$ and the LMS adaptive FIR filter $\boldsymbol{b}$ has **three** coefficients. What will $\boldsymbol{b}$ be after convergence of the LMS algorithm? What will the MSE be after convergence?

In this case, the "unexpected" FIR filter g has no effect. The LMS adapted filter will seek to minimize the MSE by converging to $b = [1,2,1]$, exactly matching $h = [1,2,1]$ and causing the MSE to become zero after convergence.

(b) 20 points. Now suppose that the "unexpected" FIR filter $g = 1$ and the LMS adaptive FIR filter $\boldsymbol{b}$ has **two** coefficients. What will $\boldsymbol{b}$ be after convergence of the LMS algorithm? What will the MSE be after convergence?

Again, the "unexpected" FIR filter has no effect. In this case, the best the LMS adaptive filter can do is match the first two coefficients of h, i.e. $b = [1,2]$. Then $e[n] = d[n] - y[n] = (x[n] + 2x[n-1] + x[n-2]) - (x[n] + 2x[n-1]) = x[n-2]$ and $E\{e^2[n]\} = E\{x^2[n-2]\} = \sigma_x^2$. Hence the MSE after convergence will be σ_x^2.

(c) 20 points. Now suppose that the "unexpected" FIR filter $g = [1, 1]$ and the LMS adaptive FIR filter $\boldsymbol{b}$ has **two** coefficients. What will $\boldsymbol{b}$ be after convergence of the LMS algorithm? What will the MSE be after convergence?

In this case, $y = x * g * b$. Note that if $b = [1,1]$, $g * b = [1,2,1] = h$. In other words, it is possible for the LMS adaptive filter to drive the MSE to zero by converging to $b = [1,1]$. The "overall" filter $f = g * b = [1,2,1] = h$, hence $y[n] = d[n]$ and $e[n] = 0$ $\forall n$. Hence the MSE will be zero after convergence.

4. 60 points total. Suppose your DSP is running the assembly code given on the last page of this exam (this code is adapted from the TI optimized FFT code that you used in Lab 5).

✓(a) 10 pts. Draw a box around the instruction(s) in the first fetch packet. Label it FP1.

✓(b) 10 pts. Draw a box around the instruction(s) in the fifth execute packet. Label it EP5.

✓(c) 10 points. Suppose the LDDW instructions on lines 22-23 are currently in pipeline stage E4. Put a pound sign (#) next to the instruction(s) currently in pipeline stage E1.

✓(d) 10 points. Put stars (⋆) next to the instruction(s) in the execute packet that runs immediately after the results from the LDDW commands on lines 22-23 become valid.

(e) 10 points. What is the minimum number of NOPs you should have in line 51 if you want the results of the MPYSPs on lines 48-49 to be valid before the ADDSP on line 52?

3 NOPs are necessary because MPYSP has 3 delay slots

using your result from part (e).

(f) 10 points. ✓ How many total cycles does it take for this function to execute? Count all cycles up to, but not including, the branch on line 53. Hint: Conditional instructions take a cycle to execute even if the condition evaluates as false.

20 cycles.

```
-1    _ myfunc:
 0            ADD     .D1     A4,A6,A3        ; f xx2 = x + n*4
 1    ||      MV      .L2     B4,B12          ; f wsave = w
 2    ||      SHRU    .S2X    A6,1,B4         ; f nsave = n>>1
 3    ||      MV      .L1X    B4,A5           ; f w1 = w
 4    ||      MVK     .S1     1,A14           ; f onea = 1
 5            MV      .S2X    A3,B8           ; f xx1 = xx2
 6    ||      MV      .L2     B4,B0           ; f i = nsave
 7    ||      SHL     .S1     A6,2,A10        ; f k1 = n<<2
 8            LDDW    .D2     *B8++,B7:B6     ; p @ t2_1:t2_0 = *xx1++
 9    ||      LDDW    .D1     *A5++,A7:A6     ; p @ s:c = *w1
10    ||      MPY     .M2     B0,1,B13        ; f ireset = i*1
11    ||      ADD     .L2X    A6,1,B9         ; f bk = n+1
12    ||      MV      .S2     B8,B5           ; f xx3 = xx1
13            MV      .L1     A4,A11          ; f xx4 = x
14    ||      SHL     .S2X    A6,2,B1         ; f k = n * 4
15            MV      .L1X    B9,A0           ; f bk1 = bk
16    ||      SUB     .L2     B0,1,B0         ; p if (i) i = i-1
17    ||      MV      .S1     A4,A3           ; f xx2 = x
18    ||      MVK     .S2     1,B14           ; f oneb = 1
19            ADD     .L2     B8,B1,B8        ; p if (!i) xx1 = xx1 + k
20    ||      MV      .S2     B13,B2          ; f t3_ctr = ireset
21    ||      MV      .L1X    B13,A1          ; f st_ctr = ireset
22            LDDW    .D2     *B8++,B7:B6     ; p @@ t2_1:t2_0 = *xx1++
23    ||      LDDW    .D1     *A5++,A7:A6     ; p @@ if (!i) s:c = *w1
24    ||      MPY     .M2     B14,B13,B0      ; p if (!i) i = ireset*1
25            MPYSP   .M1X    A6,B6,A13       ; p rtemp2 = c*t2_0
26    ||      MPYSP   .M2X    A6,B7,B11       ; p itemp2 = c*t2_1
27      [B0]  SUB     .S2     B0,1,B0         ; p if (i) i = i-1
28    ||      SUB     .L1X    B4,1,A2         ; f l = nsave - 1
29            MPYSP   .M1X    A7,B7,A15       ; p rtemp3 = s*t2_1
30    ||      MPYSP   .M2X    A7,B6,B3        ; p itemp3 = s*t2_0
31    ||      ADD     .L2     B8,B1,B8        ; p if (!i) xx1 = xx1 + k
32            LDDW    .D2     *B8++,B7:B6     ; p @@@ t2_1:t2_0 = *xx1++
33    ||      LDDW    .D1     *A5++,A7:A6     ; p @@@ if (!i) s:c = *w1
34    ||      MPY     .M2     B14,B13,B0      ; p if (!i) i = ireset*1
35            MPYSP   .M1X    A6,B6,A13       ; p rtemp2 = c*t2_0
36    ||      MPYSP   .M2X    A6,B7,B11       ; p itemp2 = c*t2_1
37      [B0]  SUB     .S2     B0,1,B0         ; p if (i) i = i-1
38            LDDW    .D1     *A3++,A9:A8     ; p @ t3_1:t3_0 = *xx2++
39    ||      MPYSP   .M1X    A7,B7,A15       ; p rtemp3 = s*t2_1
40    ||      MPYSP   .M2X    A7,B6,B3        ; p itemp3 = s*t2_0
41    ||      ADD     .D2     B8,B1,B8        ; p if (!i) xx1 = xx1 + k
42    ||      ADDSP   .L1     A13,A15,A12     ; p rtemp1 = rtemp2 + rtemp3
43    ||      SUBSP   .L2     B11,B3,B10      ; p itemp1 = itemp2 - itemp3
44            LDDW    .D2     *B8++,B7:B6     ; p @@@@ t2_1:t2_0 = *xx1++
45    ||      LDDW    .D1     *A5++,A7:A6     ; p @@@@ if (!i) s:c = *w1
46    ||      MPY     .M2     B14,B13,B0      ; p if (!i) i = ireset*1
47    ||[B2]  SUB     .S2     B2,1,B2         ; p if (t3_ctr) t3_ctr -= 1
48            MPYSP   .M1X    A6,B6,A13       ; p rtemp2 = c*t2_0
49    ||      MPYSP   .M2X    A6,B7,B11       ; p itemp2 = c*t2_1
50    ||      ADD     .S1     A3,A10,A3       ; p if (!t3_ctr) xx2 = xx2 + k1
51            NOP     3
52            ADDSP   .M1X    A13,B11,A4
53            B       .S2     B3
54            NOP             5
```

EP1
EP2
EP3
EP4
EP5
EP6
EP7
EP8
EP9
EP10
EP11
EP12
EP13
EP14
EP15
EP16
EP17-19
EP20

← FP1 (8 instructions)

← EP5

pipeline E4

pipeline E3

pipeline E2

pipeline E1