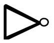
The University of Alabama in Huntsville
Electrical and Computer Engineering Department
CPE 221 01
Test 1 Solution
Fall 2018

This test is closed book, closed notes. You may not use a calculator. You should have the 6 page ARM Instruction Reference. *You must show your work to receive full credit.*

1. (1 point) A **byte** consists of 8 bits.

2. (1 point) A **combinational** logic element is a circuit whose output depends only on its current inputs.

3. (1 point) An AND gate has a controlling value of **0**.

4. (1 point) The symbol for an inverter is .

5. (1 point) The ARM processor has **16** registers.

6. (10 points) Convert decimal +327 and -456 to binary, using the signed-2's complement representation and enough digits to accommodate the numbers.

```
16        0   r 1                    16        0   r  1
16        1   r 4                    16        1   r 12
16       20   r 7                    16       28   r  8
16      327                          16      456

+327 = 0001 0100 0111  = 0x147
+456 = 0001 1100 1000
-456 = 1110 0011 1000  = 0XE38
```

7. (3 points) What is the decimal equivalent of 11001100_5 (assume positional notation and unsigned integer formats)?

$\mathbf{11001100_5 = 1 \times 5^7 + 1 \times 5^6 + 0 \times 5^5 + 0 \times 5^4 + 1 \times 5^3 + 1 \times 5^2 + 0 \times 5^1 + 0 \times 5^0 = 78125 + 15625 + 125 + 25 = 93{,}900}$

8. (12 points) If `r1 = 0x000F 0FFF` and `r2` = 4, what is the value of `r0` after each of the following instructions has been executed? Assume that each instruction uses the same data.

(a) ADD r0, r1, r1, LSL #9

```
r1 =            0000 0000 0000 1111 0000 1111 1111 1111
r1 LSL #9 =     0001 1110 0001 1111 1111 1110 0000 0000
Sum =           0001 1110 0010 1111 0000 1101 1111 1111
Sum = 0x1E2F 0DFF
```

(b) ADD r0, r1, r1, ROR #5

```
r1 =            0000 0000 0000 1111 0000 1111 1111 1111
r1 ROR #5 =     1111 1000 0000 0000 0111 1000 0111 1111
Sum =           1111 1000 0000 1111 1000 1000 0111 1110
Sum = 0xF80F 887E
```

(c) ADD r0, r1, r1, LSR r2

```
r1 =            0000 0000 0000 1111 0000 1111 1111 1111
r1 LSR r2 =     0000 0000 0000 0000 1111 0000 1111 1111
Sum =           0000 0000 0001 0000 0000 0000 1111 1110
Sum = 0x0010 00FE
```

9. (10 points) For each of the following operations on 6 bit signed numbers, calculate the values of the C, Z, V, and N flags

(a) 101111 + 001101

```
 001111   C
  101111
  001101
  111100   S
```

CZVN = 0001

(b) 110011 + 001001

```
 000011
  110011
  001001
  111100   S
```

CZVN = 0001

10. (15 points) For each of the following cases,

1. Explain the effect of each of the following instructions using register transfer notation.
2. Give the value in `r2` after each instruction executes.
3. Give the value of the effective address.

Assume that `r2` contains the initial value `0xFF00 1110` and that `r0` contains `0xFFFF 8700`. Use these initial values for each instruction individually.

(a) LDR r1, [r2]

Register Transfer _r1 ← M[r2]_

r2 _r2 = 0xFF00 1110_

Effective Address _EA = 0xFF00 1110_

(b) STR r1, [r2, #2_1101]

Register Transfer _M[r2 + 13] ← r1_

r2 _EA = 0xFF00 1110_

Effective Address _EA = 0xFF00 111D_

(c) LDR r1, [r2, #0x2C]!

Register Transfer _r2 ← r2 + 44, r1 ← M[r2]_

r2 _EA = 0xFF00 113C_

Effective Address _EA = 0xFF00 113C_

(d) STR r1, [r2], #-4

Register Transfer _M[r2] ← r1, r2 ← r2 -4_

r2 _EA = 0xFF00 110C_

Effective Address _EA = 0xFF00 1110_

```
(e) LDR  r1, [r2, r0, ASR #3]
```

Register Transfer _r1 ← M[r2 + r0 >> 3]_

r2 _EA = 0xFF00 1110_

Effective Address _EA = 0xFF00 1110 + 0xFFFF F0E0 = 0xFF00 01F0_

11. (25 points) Consider the following ARM program. Trace the values of the registers shown as they change during program execution. Also, trace the writes to memory by the STR instructions. There may be unused columns or rows in the tables. If you need to add columns or rows, you may do so. DCD 1 reserves one word of storage and sets it equal to 1. SPACE 3 reserves 3 bytes of memory but does not give those bytes a value.

```
                AREA    PROB_11, CODE, READONLY
                ENTRY
0               ADR     r0, x
4               ADR     r1, y
8               ADR     r2, z
12              LDR     r3, size
16              LDR     r4, i
20    loop      CMP     r4, r3
24              BGE     done
28              LDR     r5, [r0], #4
32              LDR     r6, [r1], #4
36              CMP     r5, r6
40              STRGT   r5, [r2], #4
44              STRLE   r6, [r2], #4
48              ADD     r4, r4, #1
52              B       loop
56    done      B       done
60    x         DCD     100, 3, -1, 2, 4, 4
84    y         DCD     -53, 247, 95, -7, 481, 91
108   z         SPACE    24
132   i         DCD      0
136   size      DCD      6
                END
```

r0	60	64	68	72	76	80	84									
r1	84	88	92	96	100	104	108									
r2	108	112	116	120	124	128	132									
r3	6															
r4	0	1	2	3	4	5	6									
r5	100	3	-1	2	4	4										
r6	-53	247	95	-7	481	91										

Results of the STR instructions.

Address	Contents
108	100
112	247
116	95
120	2
124	481
128	91

12. (20 points) Complete the ARM assembly language program below so that it implements the following C++ statements.

```
int size = 10;
for i = 0; i < size; i++)
  ndigit[i] = i + 1;
```

```
        AREA     PROB_12, CODE, READONLY
        ENTRY
        LDR      r0, i
        LDR      r1, size
        ADR      r2, ndigit
loop    CMP      r0, r1
        BGE      done
        ADD      r0, r0, #1
        STR      r0, [r2], #4
        B        loop
done    B        done
        ndigit   SPACE  40
        i        DCD    0
        size     DCD    10
                 END
```