University of California, Berkeley - College of Engineering
Department of Electrical Engineering and Computer Sciences
Instructor: Scott Beamer
Summer 2007

CS 61C MIDTERM

7/23/2007

Last Name	Key
First Name	Grading
SID	
Login	cs61c-
First Letter of Login	a b c d
Second letter of Login	a b c d e f g h i j k l m n o p q r s t u v w x y z
Lab Time	101 102 103
Name of the person to your left	
Name of the person to your right	
All the work is my own. I had no prior knowledge of the exam contents nor will I share the contents with others in CS61C who have not taken it yet. (please sign)	

Instruction

- This booklet contains 7 numbered pages including the cover page. Put all answers on these pages (feel free to use the back of any page for scratch work); don't hand in any stray pieces of paper.
- Please turn off all pagers, cell phones & beepers. Remove all hats & headphones. Place your backpacks, laptops and jackets at the front. Sit in every other seat. Nothing may be placed in the "no fly zone" spare seat/desk between students.
- Fill in the front of this page and put your name & login on every sheet of paper for Question 0.
- You have 180 minutes to complete this exam. The exam is closed book, no computers, PDAs or calculators. You may use one pages (US Letter, front and back) of notes, plus the green reference sheet from COD 3/e.
- There may be partial credit for incomplete answers; write as much of the solution as you can. We will deduct points if your solution is far more complicated than necessary. When we provide a blank, please fit your answer within the space provided. You have 3 hours...relax.

Question	0	1	2	3	4	5	Total
Points	1	13	16	16	14	15	75
Score	1	13	16	16	14	15	75

Name:_____ Login:_____

Question 1: The CS61C Grab Bag (13pts, 20min)

a) How many bits does it take to address N things?
(hint: you may use the ceiling or floor function)

ceil(log(N))

b) Let: `x = 0xfde7` `z = 0x80d5`

i. What is `x` in binary?

1111 1101 1110 0111

ii. What is the value of `x` if it was interpreted as a 2's Complement? (answer in decimal)

-537

iii. What is the encoding of the result of part ii in 1's Complement? (answer in hex)

fde6

iv. Let `x + y = z`. Interpreting `z` as a sign-magnitude number, what 1's Complement encoding of `y` will make this true? (answer in hex)

0144

c) Match all the choices on the right with the most appropriate items on the left. Some choices will not be used, and others may be used more than once. If more than one choice on the right matches an item on the left, list all of those choices.

K	Next-fit
I	Copying garbage collection
D, L	Linker
F, J	Assembler
H	Buddy Memory Allocation
B	Loader
E	Compiler
A	256 Gibi

A. 2^{38}
B. Makes request for memory
C. Uses static predefined chunks
D. Resolves jump statements
E. Converts C to MAL
F. Resolves branch statements
G. 2^{27}
H. Can vary the size of chunks
I. Wastes half the space
J. Makes the relocation table
K. Attempt to improve First-fit
L. Makes executable

d) Suppose you want to add a new pseudoinstruction to your assembler to operate on memory directly. The syntax of the instruction is:

`maddi register immediate` (ie `maddi $t2, 14`)

It will add the given immediate to the value in memory that the register holds the address for. You may assume that the absolute value of the immediate is less than 2^{15}. Write out what the instruction to the right translates to in TAL.

```
maddi $s0, 0xbeef
```

```
lw $at, 0($s0)
addi $at, $at, 0xbeef
sw $at, 0($s0)
```

Name:_____ Login:_____

Question 1 Standard

Part a: 1pt all or nothing

Part b (i-iv): 1pt each, all or nothing (4pts total)

Part c: 0.5pts for each correct letter (5pts total)

-0.5 pts for each wrong answer (but cap at 0, no negative)

Part d: 3pts possible

3/3 for perfect

2/3 for slight error

1/3 for some knowledge shown (using memory)

0/3 for no clue

Question 1 Common Mistakes

Part a: Thinking it is asking how many things N bits can represent (2^N)

Part b.iv: Done below

```
x = -537 (part ii)

z = -213 (decoding sign magnitude)

x + y = z   =>    z - x = y

(-213) - (-537) = 324 = y

y = 0000 0001 0100 0100 (unsigned)

y = 0000 0001 0100 0100 (1's Complement, positive so no change)

y = 0    1    4    4 = 0x144
```

Part d: Not using `$at` (it must be used when breaking up pseudoinstructions)

Not using `lw` or `sw` (must be used to get to memory)

Name:_____ Login:_____

Question 2: The Matrix (16pts, 25min)

In this problem you will help write some functions to deal with matrices in the mathematical sense. Our ultimate goal is to generate an `add` function for two matrices. You can use `strlen` and `strcpy` from `string.h`, but no other library functions (reference info on next page). For all of the following parts you may not need to use of all the variables declared, but you may **not** declare any additional variables.

```
struct matrixStruct {
      int *data;
      char *name;
      int numRows;
      int numCols;
};

typedef struct matrixStruct *matrix;
```

a) In one line of C complete the `getElement` function. You are guaranteed valid input, and you need to return the value in the matrix `A` at row `r` and column `c`. The matrix numbering starts with `0`, so if `r=0` and `c=0` it is the upper left entry.

```
int getElement(matrix A, int r, int c) {        2pts possible
      return (A->data)[c + r*(A->numCols)];           1/2 for using r*c + c
      (many possible ways)                            1/2 if missing A->data
                                                      0 pts for A[r][c]
      -0.5 pts for order of op error (1-array ref, 2-struct ref, 3-deref, 4-mult)
}
```

b) We want to generate a string that represents the name of the sum matrix. The string should be the name of `A`, concatenated with `" + "` (note the spaces before and after the `'+'`), and concatenated with the name of `B`. You are guaranteed valid input. Complete `getSumName`.

Example: `A->name = "PB"` `B->name="J"` result of `getSumName(A,B) = "PB + J"`

```
char* getSumName(matrix A, matrix B) {          5pts possible
      int newLength, i;
      char *newName, *temp;
1pt   newLength = strlen(A->name) + strlen(B->name) + 4;
1pt   newName = (char*) malloc(sizeof(char) * newLength);
1pt_/ temp = newName;
    \ temp = strcpy(temp, A->name);       -2 if didn't move pointer
1pt_/ temp += strlen(A->name);            -0.5 for each simple syntax error
    \ temp = strcpy(temp, " + ");         -1.5 major pointer error, but rest good
1pt_/ temp+=3;                            -2 set newName = A->name
    \ temp = strcpy(temp, B->name);       -1 for no null character

      return newName;
}
```

Name:_____ Login:_____

Question 2 (continued): The Matrix (16pts, 25min)

c) Now that we have helping functions, lets write out the `add` function. It will be given two matrices `A` and `B`, and should return their sum. If the input matrices' dimensions do not agree, they can't be added so return `NULL`. You are guaranteed non-NULL input matrices. The result matrix should have its name set as described in Part B. You may not declare any new variables or functions.

```
matrix add(matrix A, matrix B) {          6pts possible
      int i, j;
      matrix result;
0.5pt if((A->numRows != B->numRows)||(A->numCols != B-> numCols))
            return NULL;            -0.25 if logic off


1pt   result = (matrix) malloc(sizeof(struct matrixStruct));
1pt_/ result->numRows = A->numRows;
    \ result->numCols = A->numCols;
0.5pt result->name = getSumName(A, B);
1pt_/ result->data =
    \ (int* ) malloc(sizeof(int) * result->numRows * result->numCols);
     /for(i = 0; i < result->numRows; i++)
2pt_/       for(j = 0; j < result->numCols; j++)
    \             (result->data)[j + i*(A->numCols)] =
     \                  getElement(A, i, j) + getElement(B,i, j);
                  -2 for (result->data)[i][j]
                  -0.5 for syntax
                  -1 close, adds wrong elements or indexes
                  -2 wrong

      return result;
}
```

d) Fill in the function below to free a `matrix`. You have no guarantee of the integrity of `A`.

```
void freeMatrix(matrix A) {               3pts possible
      if(A==NULL)                               -2 if no check for null
            return;
                                                -1 for each missing line (cap at 3)
      free(A->data);
      free(A->name);
      free(A);
}
```

String Reference (copied from K & R)

`char *strcpy(s, ct)`	copy string `ct` to string `s`, including `'\0'` return `s`
`size_t strlen(cs)`	return length of `cs`

Name:_____ Login:_____

Question 3: Floating Away (16pts, 35min)

A new hybrid floating point standard has been developed that uses 24 bits. It uses the same tricks and style as the IEEE 754 standard, but does it with different amounts of bits. This standard uses 24 bits, with the first bit being the sign, the next 6 bits being the exponent, and the last 17 bits being the significand (mantissa). When expressing values that aren't part of an encoded float, you may give your answer as a linear combination of powers of 2.

sign	exponent	significand (also called mantissa or fraction)
1 bit	6 bits	17 bits

a) Like the IEEE 754 standard, it uses a bias. What should it be such that it has the same effect as the bias for standard floats? 31

What is the largest non-infinite number possible? 2^32 - 2^14

What is the smallest positive integer that is not precisely representable with the new hybrid float? 2^18 + 1

b) This new hybrid standard uses unbiased rounding with one guard bit.
Let: `x =0x360000 y = 0x320004 z = 0x368001`. With the hybrid standard, `x + y = z`. For what range of values of `y` (give hex of encoded floats) will produce the same result?

0x320004 ≤ y ≤ 0x350005

0x320003 also correct to interpretation

How many valid float encodings are in that range? 2

c) Now you are designing a new floating point standard to meet range and precision requirements. We will do it in the the same spirit as the IEEE 754 standard, but with a different bit allocation. Bits are expensive so you want to use the minimum:

- Range from +/- 10^{150} to 10^{-150}
- Precise enough to distinguish (1 + 1/(512 Mebi)) from (1)

What is the minimum number of bits for the exponent field? 10

What is the minimum number of bits for the significand field? 29

What is the minimum number of total bits? 40

Question 3 Standard

Part a: 2pts per box (6pts total)

2nd Box: 1pt partial credit if close

Close is 2^{31} or 2^{33} - 215, or off by one bit

3rd Box: 1pt partial credit if close

Close is 2^{18}

Part b: 4 points possible

4/4 for perfect

3/4 for very close (upper or lower bound off by a couple, and thus total off by 1)

2/4 for close (upper and lower bound off by a couple, and thus total off)

1/4 for one bound being almost reasonable

Part c: 2pts per box (6pts total)

1st Box: 1pt partial credit if off by 1 (9 or 11)

2nd Box: 1pt partial credit if off by 1 (28 or 30)

3rd Box: 1pt partial credit if added S + E + 1 (and S or E correct and other close)

Question 3 Common Mistakes

Part a: Third Box: 2^{18}, although the encoding can't hold all the bits, what it can't are taken to be 0, so it still works - so $2^{18} + 1$ isn't representable

Part b: Not getting rounding, ask TA or Scott for detailed explanation of problem

Part c: First Box: Giving some number in the thousands, correct way...

$10^{150} = (10^3)^{50} \sim (2^{10})^{50} = 2^{500}$, since also 2^{-500} needed,

roughly 1000 needed in range, 10 bits will do, 9 is too few

Third Box: Forgetting to add the sign bit (problem says +/-)

Name:_____ Login:_____

Question 4: I'm Bringing Hexy Back (14pts, 20min)

We want to write a MIPS procedure called `xtoi`. It takes in a C string of a number written in hexadecimal in ASCII characters and it returns the number as an integer. You are guaranteed valid input that is only characters `'0' - '9'` and `'a' - 'f'`. The input string will be less than or equal to 8 characters. The header for an equivalent C function would look like:

```
int xtoi(char* inpStr)
```

If `*inpStr = "0a4"`, running `xtoi(inpStr)` would return `164`. Fill in the following MIPS function to complete it. Be sure to obey MIPS register conventions. You are allowed to use pseudoinstructions, and you may need less lines than provided, but you can **not** add any extra lines in.
(Hint: Your green sheet has an ASCII chart on the back)

```
## a0 is pointer to start of string

xtoi:       addi  $sp, $sp, -12          2pts possible for this block
            sw    $s0, 0($sp)                  -1 for not saving $s0, $s1
            sw    $s1, 4($sp)                  -2 for completely wrong
            sw    $ra, 8($sp)
            add   $s0, $a0, $0
            add   $s1, $0, $0
loop:       lb    $a0, 0($s0)            5ts possible for this block
            beq   $a0, $0, allDone             2pts for lb line
            addi  Ss0, $s0, 1                  2pts for multiply (or shift)
            jal   digit                        1pt for increment $s1
            sll   $s1, $s1, 4
            add   $s1, $s1, $v0
            j     loop
allDone:    add   $v0, $s1, $0           2pts possible for this block
            lw    $s0, 0($sp)                  -1 for not restoring $s0, $s1
            lw    $s1, 4($sp)                  -1 for not restoring $sp
            lw    $ra, 8($sp)
            addi  $sp, $sp, 12
            jr    $ra

                                         -1pt for wrong order or inst's
## subroutine that takes a hex digit in $a0 and returns value as a nibble
digit:      addi  $v0, $a0, -48          5pts possible for this block
            slti  $t0, $a0, 10                 2 points for converting ASCII
            bne   $t0, $0, dDone                     to integer
            addi  $v0, $a0, -87                3 for determining proper
digitDone:  jr    $ra                              conversion
```

Question 4 Common Mistakes

Forgetting to save and restore registers. Since $s0, $s1 are used by xtoi, it as the callee must save and restore them. You must also save and restore $ra, since xtoi acts as a caller when it calls digit.

Using lw when lb should be used. Since we are dealing with characters, they are only 1 byte so we need to access a byte at a time.

Many people tried to multiply. It can be done, but since it is by 16, it is much better to just shift left by 4.

Name:_____ Login:_____

Question 5: Play that `funky` music (15pts, 20min)

Use the following MIPS code to answer the questions after it. You may assume the first instruction is at address `0x40000000`.

```
            j           funk
tasty:      addiu       $t1, $t1, 1
            sw          $t1, 0($t0)
funk:       and         $s0, $s0, $s1
            la          $t0, funk
            lw          $t1, 0($t0)
            addi        $t2, $0, 0x3f
            and         $t2, $t2, $t1
            addi        $t3, $0, 0x27
            bne         $t2, $t3, tasty
```

a) Convert the 4th instruction, (`funk: and $s0, $s0, $s1`) into machine code. Express your answer in binary, as clearly as possible.

```
000000 10000 10001 10000 00000 100100
      2pts possible (1pt partial credit for off by 1)
```

b) Given that `$s0` ← `x` and `$s1` ← `y`, with regard to the contents of `$s0` and `$s1`, what four lines of C have the same result as the above MIPS?

```
x = x&y;                    6pts possible
x = x | y;                     5/6 for 3 of the lines correct
x = x ^ y;                     3 or 4 for couple lines correct
x = ~(x | y);                  2/6 for 1 line correct
                               extra incorrect lines take points off
```

c) Come up with a single MIPS instruction which has the same result as the provided code.

```
nor $s0, $s1, $0            4pts possible
                               3/4 for nor (but wrong registers)
```

d) Translate the final instruction (`bne $t2, $t3, tasty`) into machine code. Express your answer in hexadecimal.

```
0x154BFFF7                  3pts possible
                               2/3 if branch offset wrong by few
                               1/3 if branch, but offset totally wrong
```