READ AND FOLLOW THESE INSTRUCTIONS.

- Do not begin until you are told to do so.
- You have 50 minutes; budget your time. The questions are not of equal weight; do not spend too much time on a question that is not worth many points.
- Read through all of the questions before starting to work.
- This exam is **closed notes**. You may use the MPC823 data book, the "PowerPC Programming Pocket Book", and/or a printout of Appendix F from the green book as reference material. You may *not* share reference materials with other students.

Name: _____

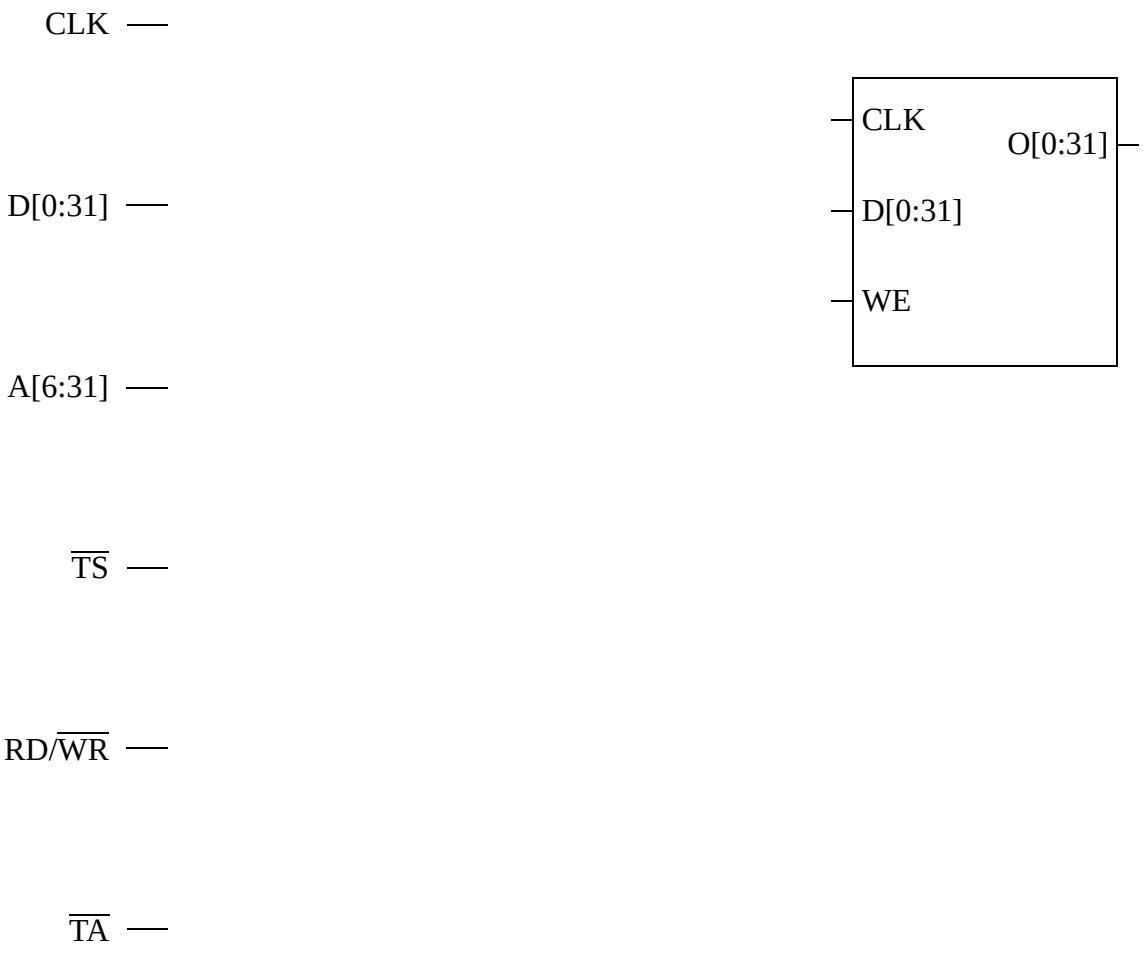
Uniqname: _____

Honor Code statement: I have neither given nor received aid on this exam.

Signature: _____

Question	Points	Score
1	35	
2	35	
3	30	
Total	100	

1. (35 pts) You are given a 32-bit hardware register built from 32 D flip-flops. The operation of the individual flip-flops and their interconnection are provided on the following pages. Using this register, construct a 32-bit word-addressable (not byte-addressable) memory location for the MPC823 at address 0x02600000. To minimize the address decoding logic, decode only A[6:10]. Use the space below to draw your circuit. All of the MPC823 bus signals you need to worry about are indicated on the left, and the register to interface is shown on the right. (If you need them, the basic MPC823 bus timing diagrams are found on pages 13-10 and 13-13 of the MPC823 data book.)



Internal schematic for 32-bit register

2. (35 pts) Below is the edited listing file from an assembly-language program. Use it to answer the following questions. *Read all of the questions before you begin.*

```
                                .data
00003000                        .equ   stack, 0x3000
00002000 45 45 43 53    strptr: .asciz "EECS373 RULES!"
         33 37 33 20
         52 55 4C 45
         53 21 00
                                .text
00001000 3C 20 00 00    _start: lis    r1, stack@h
00001004 60 21 30 00            ori    r1, r1, stack@l
00001008 3C 60 00 00            lis    r3, strptr@h
0000100C 60 63 20 00            ori    r3, r3, strptr@l
00001010 48 00 00 09            bl     func1
00001014 48 00 00 00    done:   b      done

00001018 94 21 FF F0    func1:  stwu   r1, -16(r1)
0000101C 7C 08 02 A6            mflr   r0
00001020 90 01 00 04            stw    r0, 4(r1)
00001024 93 C1 00 08            stw    r30, 8(r1)
00001028 93 E1 00 0C            stw    r31, 12(r1)

0000102C 3B E3 FF FF            addi   r31, r3, -1
00001030 8F DF 00 01    f1loop: lbzu   r30, 1(r31)
00001034 2C 1E 00 00            cmpwi  r30, 0
00001038 41 82 00 20            beq    f1done
0000103C 7F C3 F3 78            mr     r3, r30
00001040 48 00 00 31            bl     func2
00001044 2C 03 00 00            cmpwi  r3, 0
00001048 41 82 FF E8            beq    f1loop
0000104C 3B DE 00 20            addi   r30, r30, 0x20
00001050 9B DF 00 00            stb    r30, 0(r31)
00001054 4B FF FF DC            b      f1loop

00001058 83 E1 00 0C    f1done: lwz    r31, 12(r1)
0000105C 83 C1 00 08            lwz    r30, 8(r1)
00001060 80 01 00 04            lwz    r0, 4(r1)
00001064 7C 08 03 A6            mtlr   r0
00001068 38 21 00 10            addi   r1, r1, 16
0000106C 4E 80 00 20            blr

00001070 7C 64 1B 78    func2:  mr     r10, r3
00001074 38 60 00 00            li     r3, 0
00001078 2C 04 00 41            cmpwi  r10, 0x41
0000107C 41 80 00 10            blt    f2done
00001080 2C 04 00 5A            cmpwi  r10, 0x5a
00001084 41 81 00 08            bgt    f2done
00001088 38 60 00 01            li     r3, 1
0000108C 4E 80 00 20    f2done: blr
```

a. (4 pts) Step through the first five instructions (starting at `_start`). After executing exactly five instructions, what are the values of the following registers?

PC	R1	R3	LR

b. (3 pts) Continue stepping through the instructions through the `stw r31, 12(r1)` instruction at 0x1028. After executing this instruction, what are the contents of the stack pointer and the first two locations in the stack frame?

Address	Contents
R1+4	
R1 = 0x_____	

c. (2 pts) What is the name for the sequence of instructions from 0x1018 to 0x1028?

d. (5 pts) Continue stepping until you reach the `bl func2` at address 0x1040. *After* executing this instruction, what are the values of the following registers?

PC	R3	R30	R31	LR

e. (3 pts) Func2 takes a single argument and returns either 0 or 1. For what range of argument values will func2 return 1? Intuitively, what does the return value of this function indicate? (Hint: use the ASCII table on the next page.)

f. (3 pts) What does func1 do if func2 returns 1?

g. (5 pts) What string will be stored at strptr after func1 returns?

(parts h and i are on the next page)

h. (5 pts) Internally, func1 uses r30 and r31, and func2 uses r10. Func1 saves and restores the old values of the registers it uses, but func2 doesn't. Why?

i. (5 pts) Imagine that you rewrite func1 to use r10 instead of r30. In addition to simply changing the register names in the instructions, what else would you have to do to make func1 work correctly again? You don't need to write any code, just describe any necessary changes.

ASCII table: Codes 0x00 through 0x1F (NUL to US) and code 0x7F (DEL) are non-printable control characters. Codes 0x20 through 0x7E are printable characters; 0x20 is a space (' ').

3. (30 pts) Here is an excerpt from the Unix man page for the C library function *strpbrk*:

NAME

strpbrk - search a string for any of a set of characters

SYNOPSIS

char *strpbrk(char *s, char *accept);

DESCRIPTION

The strpbrk() function locates the first occurrence in the string s of any of the characters in the string accept.

RETURN VALUE

The strpbrk() function returns a pointer to the character in s that matches one of the characters in accept.

For example, if s is the address of the string “Hello world”, and accept is the address of the string “rxoq”, then strpbrk(s, accept) should return the address of the ‘o’ at the end of “Hello”. Recall that the end of a string in C is marked by a null (0) byte.

Write a simple PowerPC assembly-language version of the strpbrk function. Use the ABI conventions discussed in class. Focus on correctness rather than performance. If you run out of space, continue on the back of this sheet.