

ECE 391 Exam 1, Spring 2012  
Thursday February 23rd

Name:

- Be sure that your exam booklet has 16 pages.
- Write your name at the top of each page.
- This is a closed book exam.
- You are allowed one  $8.5 \times 11$ " sheet of notes.
- Absolutely no interaction between students is allowed.
- Show all of your work.
- Don't panic, and good luck!

*Name:* \_\_\_\_\_

2

Problem 1    23 points    \_\_\_\_\_

Problem 2    12 points    \_\_\_\_\_

Problem 3    20 points    \_\_\_\_\_

Problem 4    15 points    \_\_\_\_\_

Problem 5    20 points    \_\_\_\_\_

Problem 6    10 points    \_\_\_\_\_

Total        100 points    \_\_\_\_\_

Name: \_\_\_\_\_

3

**Problem 1 (23 points): Short Answers**

Please answer concisely. If you find yourself writing more than a sentence or two, your answer is probably wrong.

**Part A (4 points):** Recall the user-level test harness provided for your use with MP1. Describe one advantage and one disadvantage of developing and using such a testing strategy when writing new kernel code, relative to doing all testing of the new code directly in the kernel.

**Part B (5 points):** You have a device attached to IRQ0 on the PIC. Everytime that device generates an interrupt, the `divide_by_zero` exception handler is invoked instead of your device handler. What have you set up incorrectly and how do you fix it?

Name: \_\_\_\_\_

4

**Problem 1, continued:**

**Part C** (4 points): Why is it necessary to save the state of the caller saved registers immediately after receiving an interrupt?

**Part D** (5 points): Why does Linux make use of tasklets (i.e., software interrupts) instead of executing all interrupt-related activity in the (hardware) interrupt handler?

Name: \_\_\_\_\_

5

**Problem 1, continued:**

**Part E** (5 points): Explain why it is not enough to do CLI/STI when synchronizing accesses to resources shared by your code and interrupt handlers in a multiprocessor setting.

Name: \_\_\_\_\_

6

**Problem 2** (12 points): PIC Design Rationale and Issues

You may find it helpful to consult the 8259A diagram on the back of the exam for this problem.

**Part A** (4 points): Explain the role of the CAS (cascade) bus in Intel's 8259A PIC. Specifically, why it is necessary, and how is it used?

**Part B** (4 points): Three 8259A PICs are cascaded together, with slave X occupying IR0 on the master PIC and slave Y occupying IR4. Assuming that the standard priority scheme is used on each PIC (IR0 is high, IR7 is low), show the overall priority scheme for the lines on the master M (call them M0 through M7) and slaves X (X0 through X7) and Y (Y0 through Y7).

**Part C** (4 points): Draw the glue logic necessary to connect the A (address, 1 bit) and  $\overline{\text{CS}}$  (chip select, 1 bit, active low) ports of an 8259A PIC to the 16-bit address bus of a processor such that the PIC occupies ports 0x100 and 0x101. Your diagram should not be gate-level, but be sure that any component meanings are clear.

Name: \_\_\_\_\_

7

### Problem 3 (20 points): Calling Convention

You have been asked to write a recursive function to emulate an iteration of a sum-reduction collective operation (multiprocessor function to get the sum of an extremely large set of data). The C code implementation is already written for you below (and at the end of the exam). The x86 code is generated from this C code. You are to fill in the missing x86 GNU Assembly instructions related to the C calling convention you learned in class.

```
typedef struct elem{
    int value;
    elem_t* next
} elem_t;

// Returns the length of the new "sum linked-list" made during traversal
int recursive_reduce_iter(int blocksize, elem_t* start)
{
    int new_list_len, sum;
    elem_t* end = start;
    int temp = blocksize;
    if (start == NULL)          // Base Case
        return 0;
    while(temp != 0){           // Compute next block's address
        end = end->next;
        temp--;
    }
    end = end->next;

    new_list_len = recursive_reduce_iter(blocksize, end);    // Recursive Call
    sum = sum_nodes(start, end);        // sum_nodes call

    start->value = sum;    // Update node's value and next ptr
    start->next = end;
    return new_list_len+1;    // Return the length of the new list
}

int sum_nodes(elem_t* start_node, elem_t* end_node);    // Helper func, returns sum
```

The x86 Assembly is below and continues to the next page. Remember to **follow the C calling convention** and **only add code where the x86 comments say to add code.**

```
recursive_reduce_iter:
    pushl %ebp
    movl %esp, %ebp

                                # ADD YOUR CODE HERE TO...
                                # blocksize => ecx
                                # start => eax
                                # new_list_len => local var on stack

    movl %eax, %edx
    cmpl $0, %eax
    je base_case
base_loop:
    cmpl $0, %ecx
    je next_block
    movl NEXT(%edx), %edx
    decl %ecx
    jmp block_loop
```

Name: \_\_\_\_\_

8

**Problem 3, continued:**

```
next_block:
    movl NEXT(%edx), %edx
```

```
# ADD YOUR CODE HERE TO...
# Do recursive call
# followed by sum_nodes call
```

```
    movl %ecx, VALUE(%eax)
    movl %edx, NEXT(%eax)
```

```
# ADD YOUR CODE HERE TO....
# Set return value to new_list_len+1
```

```
done:
    leave
    ret
base_case:
    movl $0, %eax
    jmp done
```



Name: \_\_\_\_\_

9

#### Problem 4 (15 points): Synchronization

There is another synchronization method other than the ones taught in class called a barrier. Barriers make sure that all threads stop at a certain point before continuing. An example would be a parallel read/sort function. Several threads would read some data in parallel and stop at a barrier before moving on to do the actual sort.

```
extern static int NUM_THREADS;    // assume the number of threads does not change

// you may add additional members to this struct,
// but NO NEW synchronization primitives
typedef struct {
    spinlock_t lock;

} barrier_t;
```

**Part A (5 points):** Implement the initialization function below. Remember to initialize any members you added to the `barrier_t` struct.

```
void barrier_init(barrier_t *b)
{

}

}
```

**Part B (10 points):** Implement a `barrier_wait` function below.

```
void barrier_wait(barrier_t *b)
{

}

}
```

Name: \_\_\_\_\_

10

### Problem 5 (20 points): x86 Assembly and C

You've decided to implement a new function to supplement those you created in MP1. The purpose of the function is to make the fish appear dead, and you intend on calling it at the end of the user level test harness. To achieve this goal, you will replace the eye of the fish with an 'x' during the "off frame". In Part A, you will implement the function as a new ioctl using x86 assembly.

```

/\ /\ /\ /\ /\ /\ /\ /\ /\ /\
      o
    o   o
  o
    o   o
  o   o
    \
| \/. \ | \ / / /
|= _> \ | \ /
| /\_ / | / | /
-----M-----M-----

```

```

\ \ \ \ \ \ \ \ \ \ \ \ \ \ \ \
      o   o
    o
      o
    o   o
      /
| \ / x \ \ \ \ \ /
|= _> \ \ \ \ |
| /\_> | / | /
-----M-----M-----

```

Traverse the `mp1_list_head` list, looking for an element whose `ON_CHAR` field matches the ascii value for '.' (0x2E). If there is such an element, replace the `OFF_CHAR` field with 'x' (0x78). You are guaranteed that there is at most one '.' in the list and that this always corresponds to the "eye" you should replace. You are **NOT** guaranteed that the '.' is at a fixed location, so you must search the list based on the `ON_CHAR` field rather than the `LOCATION` field. The argument is only present for the sake of consistency, and contains only garbage. Return 0 on success, and -1 on failure. Insert the code to implement `mp1_ioctl_kill` in `mp1.S` shown on the next page.

**Use x86 GNU Assembly for this part!**

```

.data
# Useful offset constants for accessing members of a mp1_blink_struct structure
LOCATION    = 0
ON_CHAR    = 2
OFF_CHAR   = 3
ON_LENGTH  = 4
OFF_LENGTH = 6
COUNTDOWN = 8
STATUS     = 10
NEXT       = 12

```

Name: \_\_\_\_\_

11

**Problem 5, continued:**

```
.global mpl_ioctl_kill

# Pointer to head of list (initialized to NULL)
mpl_list_head: .long 0

# int mpl_ioctl_kill(unsigned long arg)
# follows C calling convention
# %ecx MUST maintain list pointer

mpl_ioctl_kill:
```

LOOP:

FOUND:

NOT\_FOUND:

Name: \_\_\_\_\_

12

### Problem 6 (10 points): Debugging(\*\*\*)

There is one bug in the following code where unexpected behavior may happen. Explain **what conditions must occur for the bug to happen, what occurs as a result of this bug, and how you would fix it.**

Assume the operations done on `arg1` and `arg2` do not overflow the `int` return value from that operation. The (\*\*\*) means this is a challenge problem. Proportion your time appropriately.

```
# int dispatcher(unsigned int arg1, unsigned int arg2, unsigned int operation)
#     Dispatcher function that uses a function pointer jump table
#     to execute the appropriate operation function.
#
#     Inputs: operation - index into function pointer jump table
#             arg1, arg2 - arguments that the functions operate on
#
#     Outputs: Returns -1 if operation is out of array bounds, otherwise
#             the function that is jumped to sets the return value
#
#     Note: The function calling dispatcher as well as each of the functions
#           in the jump table follow the C calling convention. Recall that the
#           dispatcher is a special function (MP1's mpl_ioctl that you wrote
#           was a dispatcher)

dispatcher:
    movl 12(%esp), %ecx
    cmpl $0, %ecx
    jl bad_op
    cmpl $2, %ecx
    jg bad_op
    jmp *jump table(, %ecx, 4)
bad_op:
    movl $-1, %eax
    leave
    ret

# int op_<function> (unsigned int arg1, unsigned int arg2)
#     Note: Assume that the op_<function> does not overflow
#           the return value

jump table:
    .long op_add, op_mult, op_abs
```

*Name:* \_\_\_\_\_

13

(scratch paper)

Name: \_\_\_\_\_

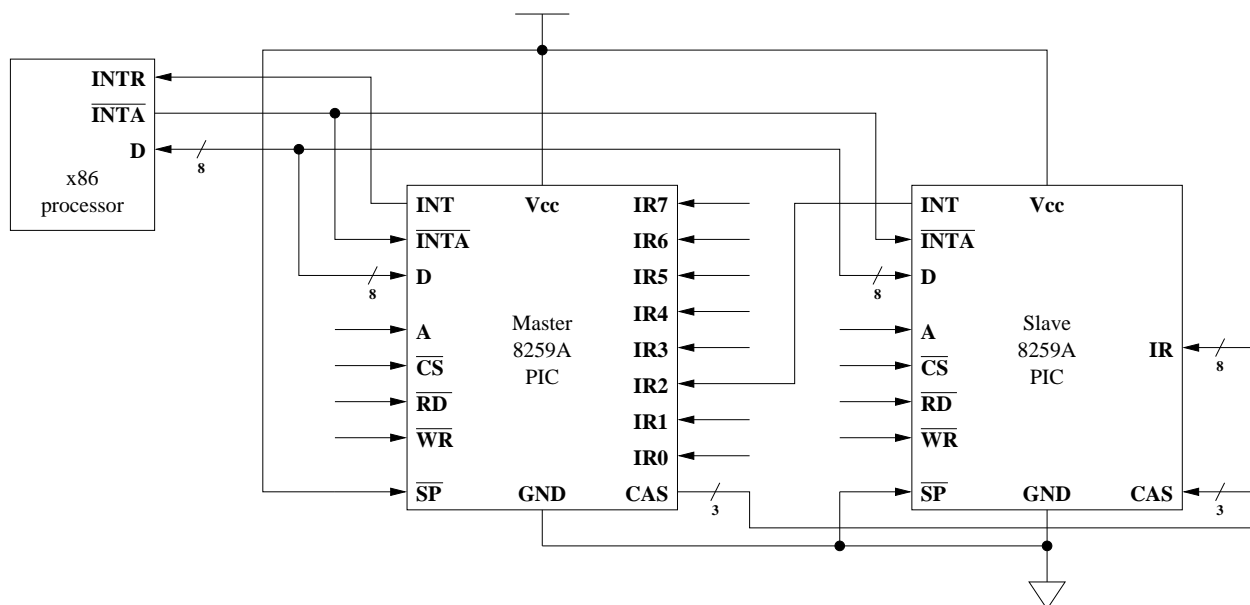
14

You may tear off this page to use as a reference

## Synchronization API reference

|  |   |
|--|---|
| <code>spinlock_t lock;</code>  | Declare an uninitialized spinlock   |
| <code>spinlock_t lock1 = SPIN_LOCK_UNLOCKED;</code><br><code>spinlock_t lock2 = SPIN_LOCK_LOCKED;</code> | Declare a spinlock and initialize it  |
| <code>void spin_lock_init(spinlock_t* lock);</code>  | Initialize a dynamically-allocated spin lock (set to unlocked)  |
| <code>void spin_lock(spinlock_t *lock);</code>   | Obtain a spin lock; waits until available   |
| <code>void spin_unlock(spinlock_t *lock);</code>   | Release a spin lock   |
| <code>void spin_lock_irqsave(spinlock_t *lock, unsigned long&amp; flags);</code>                         | Save processor status in flags, mask interrupts and obtain spin lock (note: flags passed by name (macro)) |
| <code>void spin_lock_irqrestore(spinlock_t *lock, unsigned long flags);</code>                           | Release a spin lock, then set processor status to flags   |
| <code>struct semaphore sem;</code>   | Declare an uninitialized semaphore  |
| <code>static DECLARE_SEMAPHORE_GENERIC (sem, val);</code>  | Allocate statically and initialize to val   |
| <code>DECLARE_MUTEX (mutex);</code>  | Allocate on stack and initialize to one   |
| <code>DECLARE_MUTEX_LOCKED (mutex);</code>   | Allocate on stack and initialize to zero  |
| <code>void sema_init(struct semaphore *sem, int val);</code>   | Initialize a dynamically allocated semaphore to val   |
| <code>void init_MUTEX(struct semaphore *sem);</code>   | Initialize a dynamically allocated semaphore to one.  |
| <code>void init_MUTEX_LOCKED(struct semaphore *sem);</code>  | Initialize a dynamically allocated semaphore to zero.   |
| <code>void down(struct semaphore *sem);</code>   | Wait until semaphore is available and decrement (P)   |
| <code>void up(struct semaphore *sem);</code>   | Increment the semaphore   |

## a PICTure of the 8259A



Name: \_\_\_\_\_

15

**You may tear off this page to use as a reference**

## **Recursive\_reduce\_iter() C code**

```
typedef struct elem{
    int value;
    elem_t* next
} elem_t;

// Returns the length of the new "sum linked-list" made during traversal
int recursive_reduce_iter(int blocksize, elem_t* start)
{
    int new_list_len, sum;
    elem_t* end = start;
    int temp = blocksize;
    if (start == NULL)        // Base Case
        return 0;
    while(temp != 0){        // Compute next block's address
        end = end->next;
        temp--;
    }
    end = end->next;

    new_list_len = recursive_reduce_iter(blocksize, end);    // Recursive Call
    sum = sum_nodes(start, end);    // sum_nodes call

    start->value = sum;    // Update node's value and next ptr
    start->next = end;
    return new_list_len+1;    // Return the length of the new list
}

int sum_nodes(elem_t* start_node, elem_t* end_node);    // Helper func, returns sum
```

You may tear off this page to use as a reference

## x86 reference

|  |                     |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
|--|---------------------|------------------------------|-----------|--------|----------------|-----------------|-----|-------|-----------|-----|------|---------|-----|---------------|------------------------|-----|----------|-----------|-----|--------|-------------------------|-----|------|----------------------|-----|----|--|-----|----|--|---|------|------------|---------------|------|--------------|-------------------|------|-----------------|---------------------|------|-------------------|--------------------------|------|---------------|-------------------|------|-----------------|-----------------------|------|-------------------|-------------------------|------|---------------------|------------------------------|------|---------|---------------|------|-----------|-----------------|------|--------------|--------------------|------|---------------|--------------|--|--|--|------|-----------------|-----------------------|------|---------------------|----------------------------|--|--|--|------|--------------|---------------|------|-----------------|------------------|------|--------------------|--------------|--|--|--|------|--------|----------------------------|------|-------|-------------------------|------|---------|----------------------------|------|-------|-----------------------------|------|------------------|---------------------|--|--|-----------------------|
| <table><tr><td></td><td></td><td>8-bit</td></tr><tr><td>32-bit</td><td>16-bit</td><td>high low</td></tr><tr><td>EAX</td><td>AX</td><td>AH AL</td></tr><tr><td>EBX</td><td>BX</td><td>BH BL</td></tr><tr><td>ECX</td><td>CX</td><td>CH CL</td></tr><tr><td>EDX</td><td>DX</td><td>DH DL</td></tr><tr><td>ESI</td><td>SI</td><td></td></tr><tr><td>EDI</td><td>DI</td><td></td></tr><tr><td>EBP</td><td>BP</td><td></td></tr><tr><td>ESP</td><td>SP</td><td></td></tr></table> <div><div>3116 158 70</div><div>AX</div><div>AHAL</div><div>EAX</div></div> |                     |                              | 8-bit     | 32-bit | 16-bit         | high low        | EAX | AX    | AH AL     | EBX | BX   | BH BL   | ECX | CX            | CH CL                  | EDX | DX       | DH DL     | ESI | SI     |                         | EDI | DI   |                      | EBP | BP |  | ESP | SP |  | <table><tr><td>movb</td><td>(%ebp),%al</td><td># AL ← M[EBP]</td></tr><tr><td>movb</td><td>-4(%esp),%al</td><td># AL ← M[ESP - 4]</td></tr><tr><td>movb</td><td>(%ebx,%edx),%al</td><td># AL ← M[EBX + EDX]</td></tr><tr><td>movb</td><td>13(%ecx,%ebp),%al</td><td># AL ← M[ECX + EBP + 13]</td></tr><tr><td>movb</td><td>(,%ecx,4),%al</td><td># AL ← M[ECX * 4]</td></tr><tr><td>movb</td><td>-6(,%edx,2),%al</td><td># AL ← M[EDX * 2 - 6]</td></tr><tr><td>movb</td><td>(%esi,%eax,2),%al</td><td># AL ← M[ESI + EAX * 2]</td></tr><tr><td>movb</td><td>24(%eax,%esi,8),%al</td><td># AL ← M[EAX + ESI * 8 + 24]</td></tr><tr><td>movb</td><td>100,%al</td><td># AL ← M[100]</td></tr><tr><td>movb</td><td>label,%al</td><td># AL ← M[label]</td></tr><tr><td>movb</td><td>label+10,%al</td><td># AL ← M[label+10]</td></tr><tr><td>movb</td><td>10(label),%al</td><td># NOT LEGAL!</td></tr><tr><td colspan="3"> </td></tr><tr><td>movb</td><td>label(%eax),%al</td><td># AL ← M[EAX + label]</td></tr><tr><td>movb</td><td>7*6+label(%edx),%al</td><td># AL ← M[EDX + label + 42]</td></tr><tr><td colspan="3"> </td></tr><tr><td>movw</td><td>\$label,%eax</td><td># EAX ← label</td></tr><tr><td>movw</td><td>\$label+10,%eax</td><td># EAX ← label+10</td></tr><tr><td>movw</td><td>\$label(%eax),%eax</td><td># NOT LEGAL!</td></tr><tr><td colspan="3"> </td></tr><tr><td>call</td><td>printf</td><td># (push EIP), EIP ← printf</td></tr><tr><td>call</td><td>*%eax</td><td># (push EIP), EIP ← EAX</td></tr><tr><td>call</td><td>*(%eax)</td><td># (push EIP), EIP ← M[EAX]</td></tr><tr><td>call</td><td>*fptr</td><td># (push EIP), EIP ← M[fptr]</td></tr><tr><td>call</td><td>*10(%eax,%edx,2)</td><td># (push EIP), EIP ←</td></tr><tr><td></td><td></td><td># M[EAX + EDX*2 + 10]</td></tr></table> | movb | (%ebp),%al | # AL ← M[EBP] | movb | -4(%esp),%al | # AL ← M[ESP - 4] | movb | (%ebx,%edx),%al | # AL ← M[EBX + EDX] | movb | 13(%ecx,%ebp),%al | # AL ← M[ECX + EBP + 13] | movb | (,%ecx,4),%al | # AL ← M[ECX * 4] | movb | -6(,%edx,2),%al | # AL ← M[EDX * 2 - 6] | movb | (%esi,%eax,2),%al | # AL ← M[ESI + EAX * 2] | movb | 24(%eax,%esi,8),%al | # AL ← M[EAX + ESI * 8 + 24] | movb | 100,%al | # AL ← M[100] | movb | label,%al | # AL ← M[label] | movb | label+10,%al | # AL ← M[label+10] | movb | 10(label),%al | # NOT LEGAL! |  |  |  | movb | label(%eax),%al | # AL ← M[EAX + label] | movb | 7*6+label(%edx),%al | # AL ← M[EDX + label + 42] |  |  |  | movw | \$label,%eax | # EAX ← label | movw | \$label+10,%eax | # EAX ← label+10 | movw | \$label(%eax),%eax | # NOT LEGAL! |  |  |  | call | printf | # (push EIP), EIP ← printf | call | *%eax | # (push EIP), EIP ← EAX | call | *(%eax) | # (push EIP), EIP ← M[EAX] | call | *fptr | # (push EIP), EIP ← M[fptr] | call | *10(%eax,%edx,2) | # (push EIP), EIP ← |  |  | # M[EAX + EDX*2 + 10] |
|  |                     | 8-bit                        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| 32-bit   | 16-bit              | high low                     |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| EAX  | AX                  | AH AL                        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| EBX  | BX                  | BH BL                        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| ECX  | CX                  | CH CL                        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| EDX  | DX                  | DH DL                        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| ESI  | SI                  |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| EDI  | DI                  |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| EBP  | BP                  |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| ESP  | SP                  |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | (%ebp),%al          | # AL ← M[EBP]                |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | -4(%esp),%al        | # AL ← M[ESP - 4]            |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | (%ebx,%edx),%al     | # AL ← M[EBX + EDX]          |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | 13(%ecx,%ebp),%al   | # AL ← M[ECX + EBP + 13]     |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | (,%ecx,4),%al       | # AL ← M[ECX * 4]            |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | -6(,%edx,2),%al     | # AL ← M[EDX * 2 - 6]        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | (%esi,%eax,2),%al   | # AL ← M[ESI + EAX * 2]      |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | 24(%eax,%esi,8),%al | # AL ← M[EAX + ESI * 8 + 24] |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | 100,%al             | # AL ← M[100]                |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | label,%al           | # AL ← M[label]              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | label+10,%al        | # AL ← M[label+10]           |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | 10(label),%al       | # NOT LEGAL!                 |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
|  |                     |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | label(%eax),%al     | # AL ← M[EAX + label]        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movb   | 7*6+label(%edx),%al | # AL ← M[EDX + label + 42]   |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
|  |                     |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movw   | \$label,%eax        | # EAX ← label                |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movw   | \$label+10,%eax     | # EAX ← label+10             |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| movw   | \$label(%eax),%eax  | # NOT LEGAL!                 |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
|  |                     |                              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| call   | printf              | # (push EIP), EIP ← printf   |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| call   | *%eax               | # (push EIP), EIP ← EAX      |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| call   | *(%eax)             | # (push EIP), EIP ← M[EAX]   |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| call   | *fptr               | # (push EIP), EIP ← M[fptr]  |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| call   | *10(%eax,%edx,2)    | # (push EIP), EIP ←          |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
|  |                     | # M[EAX + EDX*2 + 10]        |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| <table><tr><td>jb</td><td>below</td><td>CF is set</td></tr><tr><td>jbe</td><td>below or equal</td><td>CF or ZF is set</td></tr><tr><td>je</td><td>equal</td><td>ZF is set</td></tr><tr><td>jle</td><td>less</td><td>SF ≠ OF</td></tr><tr><td>jle</td><td>less or equal</td><td>(SF ≠ OF) or ZF is set</td></tr><tr><td>jo</td><td>overflow</td><td>OF is set</td></tr><tr><td>jp</td><td>parity</td><td>PF is set (even parity)</td></tr><tr><td>js</td><td>sign</td><td>SF is set (negative)</td></tr></table>  | jb                  | below                        | CF is set | jbe    | below or equal | CF or ZF is set | je  | equal | ZF is set | jle | less | SF ≠ OF | jle | less or equal | (SF ≠ OF) or ZF is set | jo  | overflow | OF is set | jp  | parity | PF is set (even parity) | js  | sign | SF is set (negative) |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| jb   | below               | CF is set                    |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| jbe  | below or equal      | CF or ZF is set              |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| je   | equal               | ZF is set                    |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| jle  | less                | SF ≠ OF                      |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| jle  | less or equal       | (SF ≠ OF) or ZF is set       |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| jo   | overflow            | OF is set                    |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| jp   | parity              | PF is set (even parity)      |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |
| js   | sign                | SF is set (negative)         |           |        |                |                 |     |       |           |     |      |         |     |               |                        |     |          |           |     |        |                         |     |      |                      |     |    |  |     |    |  |   |      |            |               |      |              |                   |      |                 |                     |      |                   |                          |      |               |                   |      |                 |                       |      |                   |                         |      |                     |                              |      |         |               |      |           |                 |      |              |                    |      |               |              |  |  |  |      |                 |                       |      |                     |                            |  |  |  |      |              |               |      |                 |                  |      |                    |              |  |  |  |      |        |                            |      |       |                         |      |         |                            |      |       |                             |      |                  |                     |  |  |                       |

Conditional branch sense is inverted by inserting an “N” after initial “J,” e.g., JNB. Preferred forms in table below are those used by debugger in disassembly. Table use: after a comparison such as  
cmp %ebx,%esi # set flags based on (ESI - EBX)  
choose the operator to place between ESI and EBX, based on the data type. For example, if ESI and EBX hold unsigned values, and the branch should be taken if ESI ≤ EBX, use either JBE or JNA. For branches other than JE/JNE based on instructions other than CMP, check the branch conditions above instead.

|                |     |      |     |    |     |      |                      |
|----------------|-----|------|-----|----|-----|------|----------------------|
| preferred form | jnz | jnae | jna | jz | jnb | jnbe | unsigned comparisons |
|                | jne | jb   | jbe | je | jae | ja   |                      |
|                | ≠   | <    | ≤   | =  | ≥   | >    | signed comparisons   |
| preferred form | jne | jl   | jle | je | jge | jg   |                      |
|                | jnz | jnge | jng | jz | jnl | jnle |                      |