For Loops

Lecture 10
Section 2.7

Robb T. Koether

Hampden-Sydney College

Mon, Sep 16, 2019

1 For Loops in MIPS

Outline

For Loops in MIPS

Equivalent `while` Loop

```
i = 0;
while (i < limit)
{
     (action)
     i++;
}
```

- Recall the **`while`** loop equivalent of a **`for`** loop.

For Loops in MIPS

Equivalent Loop in Unstructured C

```
    i = 0;
loop_beg:
    if (i >= limit) goto loop_end;
    {
        (action)
        i++;
        goto loop_beg;
    }
loop_end:
```

- Recall the **`while`** loop equivalent of a **`for`** loop.

For Loops in MIPS

Equivalent Loop in Unstructured C

```
    i = 0;
    goto loop_end;
loop_beg:
    {
        (action)
        i++;
    }
loop_end:
    if (i < limit) goto loop_beg;
```

- The improved **`while`** loop.

For Loops in MIPS

For Loops in MIPS

```
        li      $t0, 0                  # i = 0
        li      $s0, 10                 # limit = 10
        j       loop_end                # Jump to test
loop_beg:
        (action)                        # Perform action
        addi    $t0, $t0, 1             # i = i + 1
loop_end:
        blt     $t0, $s0, loop_beg # Branch if i < 10
```

- Assume that the counter is in `$t0` and the limit is in `$s0`.

For Loops in MIPS

For Loops in MIPS (Count Down)

```
        li      $t0, 10          # i = 10
        j       loop_end         # Jump to test
loop_beg:
        (action)                 # Perform action
        addi    $t0, $t0, -1     # i = i - 1
loop_end:
        bgtz    $t0, loop_beg    # Continue if i > 0
```

- Sometimes it is more convenient to count down instead of up in a **`for`** loop.

For Loops in MIPS

- Examples
  - Run `count_to_10.asm` example.