Conditional and Loop Transformations for Assembly

Standard

Conditionals

The basic if-else statement:

```
if (test-expr)
    then-stmt
else
    else-stmt
```

Becomes:

```
    if( !test-expr )
        goto false
    then-stmt
    goto done
false
    else-stmt
done
```

or alternatively:

```
    if( test-expr )
    goto true
    else-stmt
    goto done
true
    then-stmt
done
```

Loops

Do-While Loops

```
do
    body-stmt
    while(test-expr)
```

Becomes:

```
loop:
    body-stmt
    if( t )
        goto loop
```

While Loops

The basic while loop:

```
while( test-expr )
    body-stmt
```

Is often translated into *jump-to-the-middle* style goto logic.

```
    goto test
loop
    body-stmt
test
    if( t )
        goto loop
```

Alternatively, while can be translated into a *guarded-do*:

```
    if( !test-expr )
        goto done
    do
        body-stmt
    while( test-expr )
done
```

which is then translated to pure goto logic as:

```
    if( !t )
        goto done
loop
    body-stmt
    if( t )
        goto loop
done
```

For Loops

For loops are easily translated into while loops:

```
for(init-expr; test-expr; update-expr)
    body-stmt
```

Is equivalent to:

```
init-expr
while( test-expr )
    body-stmt
    update-expr
```

From here we can use either of the while to goto transformations above.