Quiz 2 v2: Review Warnings,Errors, Functional Equivalence

Compiler Warnings

If the compiler gives you a warning, then your code definitely has an error

- ◯ True
- ◯ False

Compiler Warnings

```
int foo(int condition) {
  int x;
  if (condition) {
    x = 5;
  }
  int y = x;
  return y;
}
```

Clang gives a warning

Compiler Warnings

```
int foo(int condition) {
  int x;
  if (condition) {
    x = 5;
  }
  int y = x;
  return y;
}
```

What if its only called like this?

```
int main() {
  foo(1);
  return 0;
}
```

Uninitialized variables

An uninitialized variable can give you any value, however, the value that it gives you will be the same each time you run the program

- ◯ True
- ◯ False

Uninitialized variables

An uninitialized variable can give you any value, however, the value that it gives you will be the same each time you run the program

- ◯ True
- ◯ False

This behavior can depend both on the operating system, and on the level of optimizations you use.

Compilers modifying code

Compilers are allowed to modify a function in any way just so long as it returns the same value as the original function

- ◯ True
- ◯ False

Compilers modifying code

- Consider this:

```
int write_data_to_file(char * data) {
  f = fopen("data.txt");
  f.write(data);
  f.close();
  return 0;
}
```

Can the compiler transform it to this?

```
int write_data_to_file(char * data) {
  return 0;
}
```

Compilers modifying code

- Consider this:

Anything that a function does that has an effect outside of itself is called a “side effect”

```
int write_data_to_file(char * data) {
  f = fopen("data.txt");
  f.write(data);
  f.close();
  return 0;
}
```

Can the compiler transform it to this? NO

```
int write_data_to_file(char * data) {
  return 0;
}
```

Compilers modifying code

- Consider another one:

```
int signal(int * flag) {
  *flag = 1;
  return 0;
}
```

Memory writes cannot be optimized!

Can the compiler transform it to this? NO

```
int signal(int * flag) {
  return 0;
}
```

Compilers modifying code

- Consider another one:

```
int signal(int * flag) {
  *flag = 1;
  return 0;
}
```

Are memory reads side effects?

Can the compiler transform it to this? NO

```
int signal(int * flag) {
  return 0;
}
```

Compilers modifying code

- Consider another one:

```
int signal(int * flag) {
  *flag = 1;
  return 0;
}
```

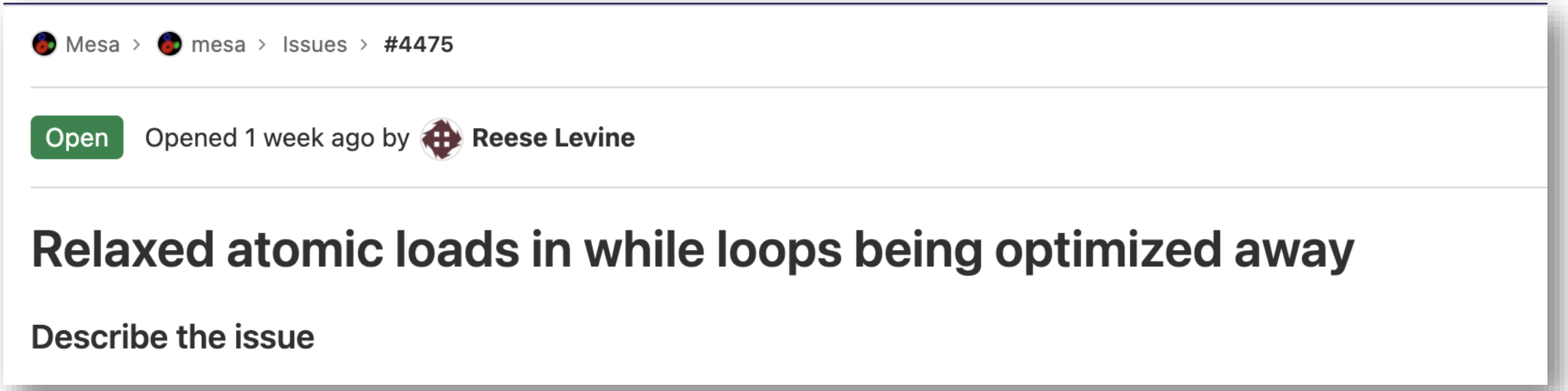
```
int wait(int * flag) {
   while (*flag != 0);
   return 0;
}
```

Can the compiler transform it to this?

```
int signal(int * flag) {
  return 0;
}
```

Can the compiler transform it to this?

```
int wait(int * flag) {
  return 0;
}
```



issues discovered by UCSC grad students!

https://gitlab.freedesktop.org/mesa/mesa/-/issues/4475