How To Thread

Why Thread

➔ Threading is doing more than one thing at once. It allows you to do things like opening a claw and driving at the same time. Whereas if you didn't use threading you'd have to stop, open the claw, stop, and then drive.

First you need to include threading.h

- The lines of code you'll need to thread are in a different library and to access them you need to include the library that stores it.

It should look something like this …

```
#include <kipr/wombat.h>
#include <threading.h>
```

There are four main lines of code that you will need in order to thread. These lines of code are:

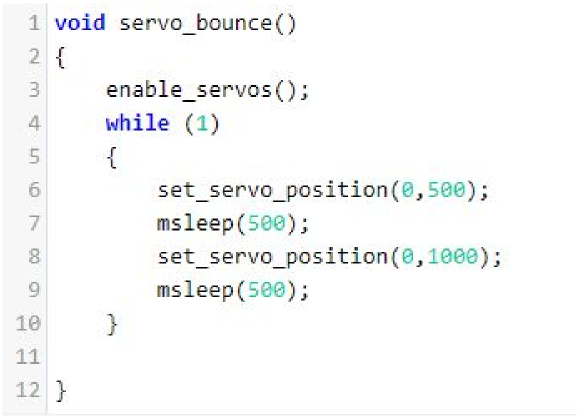
- thread example;
- example = thread_create(function);
- thread_ destroy(example);

Where there is the word "example" in the lines of code above, is the name of the thread you will create. For instance, if you created a thread that moves a servo and drives at the same time, the servo moving up and down looks like a wave. So, let's name this thread "wave". It will look something like this:

- thread wave;
- wave = thread_create(function);
- thread_destroy(wave);

Right now we are coding in main.c but we need to create a function so let's add a function in include files. First you'll need to add a file in the "include files" section. Let's name this file the same name we are going to call our function. For this example we're going to call the file 'servo_bounce'. Once you name the file it will look like this:

- servo_bounce.h



```
void servo_bounce()
{
    enable_servos();
    while (1)
    {
        set_servo_position(0,500);
        msleep(500);
        set_servo_position(0,1000);
        msleep(500);
    }

}
```

Once you have your file named go ahead and create your function. "Servo_bounce" is the name of our function just like "wave" is the name of our thread. Remember this can be named anything you'd like. Compile and return back to your main.c file.

How To Thread

Now let’s put it all together. We have a name for our thread, a name for our function, and we can now combine everything we have learned. When we put it all together it will look something like this:

```
#include <kipr/wombat.h>
#include <threading.h>

thread wave;
Int main();
{
     printf(“Hello World\n”);

     wave = thread_create(servo_bounce);
     thread_start(wave);

     mav(0,500);
     mav(3,500);
     msleep(5000);

    thread_destroy(wave);
    return 0;
}
```