|  |  |  |
| --- | --- | --- |
|  | You don't often get email from [quincy.j.crane@gmail.com](mailto:quincy.j.crane@gmail.com). [Learn why this is important](https://aka.ms/LearnAboutSenderIdentification) |  |

|  |
| --- |
|  |

//Code functions  
#include <kipr/wombat.h>  
#include <stdio.h>  
  
const int small\_turn = 20;  
const int medium\_turn = 60;  
const int large\_turn = 80;  
const int pause\_time = 1000;  
const int sensor = 0;  
int looking;  
  
  
// Declare functions  
void forward(int distance);  
void backward(int distance);  
void left(int distance, int radius);  
void right(int distance, int radius);  
void rotate\_left(int turn);  
void rotate\_right(int turn);  
void arm\_up();  
void arm\_down();  
void claw\_open();  
void claw\_close();  
void pause();  
void find\_line();  
  
// Lookup table for short (1) turns  
// left (500); //45B0  
// left turn: 1000 // 90B0  
// 180B0 turn: 2450 // 180B0  
// 270B0 turn: 3900 // 270B0  
// 360B0 turn: 5200 // 360B0  
  
// Lookup table for medium (2) turns  
// left(1450,2); // 45B0  
// left(2900,2); // 90B0  
// left(5800); // 180B0  
// left(7300); //270B0  
// left(1160);11600 // 360B0  
  
// Lookup table for large (3) turns  
// left(); //  
// left(); //  
// left(); //  
// left(); //  
// left(); //  
  
  
// Main loop  
int main()  
{  
 printf("Starting main\n");  
   
   
 return 0;  
}  
  
  
  
  
  
// Move forward  
void forward(int distance)  
{  
 printf("Moving robot forward ");  
 printf("%d", distance);  
 printf("\n");  
 mav(0,980);  
 mav(3,990);  
  
 msleep(distance);  
 ao();  
}  
  
// Move backwards  
void backward(int distance)  
{  
 printf("Moving robot backward ");  
 printf("%d", distance);  
 printf("\n");  
 msleep(1000); // Pause for a moment  
 mav(0,-980);  
 mav(3,-990);  
 msleep(distance);  
 ao();  
 msleep(1000); // Pause for a moment  
}  
  
// Turn left  
void left(int distance, int radius)  
{  
 printf("Moving robot left ");  
 printf("%d", distance);  
 printf("\n");  
  
 if (radius == 1)  
 {  
 mav(0,1000);  
 mav(3,small\_turn);  
 }  
  
 if (radius == 2)  
 {  
 mav(0,1000);  
 mav(3,medium\_turn);  
 }  
  
 if (radius == 3)  
 {  
 motor(0,1000);  
 mav(3,large\_turn);  
 }  
  
 msleep(distance);  
 ao();  
}  
  
// Turn right  
void right(int distance, int radius)  
{  
 printf("Moving robot right ");  
 printf("%d", distance);  
 printf("\n");  
  
 if (radius == 1)  
 {  
 mav(0,small\_turn);  
 mav(3,1000);  
 }  
  
 if (radius == 2)  
 {  
 mav(0,medium\_turn);  
 mav(3,1000);  
 }  
  
 if (radius == 3)  
 {  
 mav(0,large\_turn);  
 mav(3,1000);  
 }  
  
 msleep(distance);  
 ao();  
}  
// Move servo arm up  
void arm\_up()  
{  
 printf("Moving robot arm up\n");  
 enable\_servos();  
 msleep(250);  
 set\_servo\_position(0,280); // Arm up  
 msleep(250);  
 disable\_servos();  
  
}  
  
// Move arm down  
void arm\_down()  
{  
 printf("Moving robot arm down\n");  
 enable\_servos();  
 msleep(250);  
 set\_servo\_position(0,1735); // Arm down  
 msleep(250);  
 disable\_servos();  
}  
  
// Open claw  
void claw\_open()  
{  
 printf("Opening the claw\n");  
 enable\_servos();  
 msleep(500);  
 set\_servo\_position(2,1160); // Claw open  
 msleep(500);  
 disable\_servos();  
}  
  
// Close claw  
void claw\_close()  
{  
 printf("Closing the claw\n");  
 enable\_servos();  
 msleep(500);  
 set\_servo\_position(2,300); // Claw closed  
 msleep(500);  
 disable\_servos();  
}  
  
// Pause  
void pause()  
{  
 printf("Pause\n");  
 ao();  
 msleep(pause\_time);  
}  
  
// Rotate  
void rotate\_left(int turn)  
{  
 mav(0,1000);  
 mav(3,-1000);  
 msleep(turn);  
 ao();  
}  
  
// Rotate  
void rotate\_right(int turn)  
{  
 mav(0,-1000);  
 mav(3,1000);  
 msleep(turn);  
 ao();  
}  
  
//Find Line  
void find\_line()  
{  
 while (looking == 1)  
 {  
 mav(0,1000);  
 mav(3,1000);  
 if (analog(sensor) <= 3800)  
 {  
 ao();  
 looking = 0;  
 }  
 }  
}  
const int Crane\_Motor = 2;  
//Crane  
void Crane\_Up(int distance)  
{  
  
 move\_relative\_position ( Crane\_Motor,30,distance);  
  
}