**Functions sensor\_commands.h**

|  |  |
| --- | --- |
| **Syntax** | void setSensorValue(int value, sensor& target) |
| **Purpose** | set a sensor to a desired value |
| **Entry Conditions** | * *value* - the desired sensor value * *target* -the target sensor |
| **Code**  **Example** | //create solenoid  sensor s1;  //assign s1  assignSensor(sensorDigitalOut, dgtl1);  //fire the solenoid  setSensorValue(1, s1); |

|  |  |
| --- | --- |
| **Syntax** | void setSensorValue(int value, sensorArray& target) |
| **Purpose** | set an array of sensors to a desired value |
| **Entry Conditions** | * *value* - the desired sensor array value * *target* - the target sensor array |
| **Code**  **Example** | //create solenoid  sensor s1;  sensor s2;  //create solenoid array  sensorArray catapult;  //assign s1  assignSensor(sensorDigitalOut, dgtl1);  //assign s2  assignSensor(sensorDigitalOut, dgtl2);  //assign sensor array  assignSensorArray(catapult, s1, s2);  //fire the solenoid  setSensorValue(1, catapult); |

|  |  |
| --- | --- |
| **Syntax** | void zeroSensor(sensor& target) |
| **Purpose** | set the value of a sensor to zero |
| **Entry Conditions** | * target - the target sensor |
| **Code**  **Example** | //reset the right drive encoder  zeroSensor(rightDriveSensor); |

|  |  |
| --- | --- |
| **Syntax** | void zeroSensor(sensorArray& target) |
| **Purpose** | set the value of the sensors in a sensor array to zero |
| **Entry Conditions** | * target - the target sensor array |
| **Code**  **Example** | //create a sensor array  sensorArray driveEncoders;  //assign the sensor array  assignSensorArray(driveEncoders, rightDriveSensor,  leftDriveSensor);  //reset the drive encoders  zeroSensor(driveEncoders); |

|  |  |
| --- | --- |
| **Syntax** | int getSensorValue(sensor& target) |
| **Purpose** | return the value of the target sensor |
| **Entry Conditions** | * *target* - the target sensor |
| **Code**  **Example** | //display the value of the right drive sensor on the lcd  displayLCDNumber(0,0, getSensorValue(rightDrive)); |

|  |  |
| --- | --- |
| **Syntax** | float pid(float constant, int target, int currentValue) |
| **Purpose** | return a pid motor value |
| **Entry Conditions** | * *constant* - controls the motor speed ratio * *target* - the target value trying to be reached * *currentValue* - the value which it is currently at |
| **Code**  **Example** | //create potentiometer sensor  sensor pot;  //initialize the potentiometer  assignSensor(pot, sensorPotentiometer, in1);  //control motor speed  while(true)  motor[port1] = pid(0.7, 2000, getSensorValue(pot)) |

|  |  |
| --- | --- |
| **Syntax** | float xAccelerometerValue(sensor& accelerometer) |
| **Purpose** | return the raw x accelerometer value |
| **Entry Conditions** | * *accelerometer* - the target accelerometer |
| **Code**  **Example** | //create accelerometer sensor  sensor accel;  //assign accel sensor  assignSensor(accel, sensorAccelerometer, in1, in2, in3);  //display the value of the x accelerometer on the lcd  displayLCDNumber(0,0, xAccelerometerValue(accel)); |

|  |  |
| --- | --- |
| **Syntax** | float yAccelerometerValue(sensor& accelerometer) |
| **Purpose** | return the raw y accelerometer value |
| **Entry Conditions** | * *accelerometer* - the target accelerometer |
| **Code**  **Example** | //create accelerometer sensor  sensor accel;  //assign accel sensor  assignSensor(accel, sensorAccelerometer, in1, in2, in3);  //display the value of the y accelerometer on the lcd  displayLCDNumber(0,0, yAccelerometerValue(accel)); |

|  |  |
| --- | --- |
| **Syntax** | float zAccelerometerValue(sensor& accelerometer) |
| **Purpose** | return the raw z accelerometer value |
| **Entry Conditions** | * *accelerometer* - the target accelerometer |
| **Code**  **Example** | //create accelerometer sensor  sensor accel;  //assign accel sensor  assignSensor(accel, sensorAccelerometer, in1, in2, in3);  //display the value of the z accelerometer on the lcd  displayLCDNumber(0,0, zAccelerometerValue(accel)); |

|  |  |
| --- | --- |
| **Syntax** | float xAccelerometerVelocity(sensor& accelerometer) |
| **Purpose** | return the instantaneous velocity of the x accelerometer |
| **Entry Conditions** | * *accelerometer* - the target accelerometer |
| **Code**  **Example** | //create accelerometer sensor  sensor accel;  //assign accel sensor  assignSensor(accel, sensorAccelerometer, in1, in2, in3);  //display the velocity of the x accelerometer on the lcd  displayLCDNumber(0,0, xAccelerometerVelocity(accel)); |

|  |  |
| --- | --- |
| **Syntax** | float yAccelerometerVelocity(sensor& accelerometer) |
| **Purpose** | return the instantaneous velocity of the y accelerometer |
| **Entry Conditions** | * *accelerometer* - the target accelerometer |
| **Code**  **Example** | //create accelerometer sensor  sensor accel;  //assign accel sensor  assignSensor(accel, sensorAccelerometer, in1, in2, in3);  //display the velocity of the y accelerometer on the lcd  displayLCDNumber(0,0, yAccelerometerVelocity(accel)); |

|  |  |
| --- | --- |
| **Syntax** | float zAccelerometerVelocity(sensor& accelerometer) |
| **Purpose** | return the instantaneous velocity of the z accelerometer |
| **Entry Conditions** | * *accelerometer* - the target accelerometer |
| **Code**  **Example** | //create accelerometer sensor  sensor accel;  //assign accel sensor  assignSensor(accel, sensorAccelerometer, in1, in2, in3);  //display the velocity of the z accelerometer on the lcd  displayLCDNumber(0,0, zAccelerometerVelocity(accel)); |