Teleop:

```
#pragma config(Hubs,  S1, HTMotor,  HTMotor,  HTMotor,  HTMotor)
#pragma config(Hubs,  S2, HTServo,  none,     none,     none)
#pragma config(Sensor, S1,     ,               sensorI2CMuxController)
#pragma config(Sensor, S2,     ,               sensorI2CMuxController)
#pragma config(Motor,  mtr_S1_C1_1,     left,          tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C1_2,     llauncher,     tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C2_1,     left2,         tmotorTetrix, openLoop,
#pragma config(Motor,  mtr_S1_C2_2,     lifter,        tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C3_1,     rlauncher,     tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C3_2,     right,         tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C4_1,     right2,        tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C4_2,     intake,        tmotorTetrix, openLoop)
#pragma config(Servo,  srvo_S2_C1_1,    grabber,
tServoStandard)
#pragma config(Servo,  srvo_S2_C1_2,    opener,
tServoStandard)
#pragma config(Servo,  srvo_S2_C1_3,    servo3,
tServoNone)
#pragma config(Servo,  srvo_S2_C1_4,    servo4,
tServoNone)
#pragma config(Servo,  srvo_S2_C1_5,    servo5,
tServoNone)
#pragma config(Servo,  srvo_S2_C1_6,    servo6,
tServoNone)
//*!!Code automatically generated by 'ROBOTC' configuration wizard
!!*//

//Team 7347, Nick Vosseteig, Alex Iverson

#include "JoystickDriver.c"

const int driveType = 1;
int LaunchEncoderValue = 0;
//int PrevLaunchEncoderValue = 0;
//int PrevEncoderReadTime = 0;
int EncoderReadTime = 0;
int EncoderChange = 0;
float LauncherAngularVelocity = 0;
int LoopTime = 0;
void LauncherReverse(){
  motor[llauncher] = 127;
  motor[rlauncher] = 127;
}
void LauncherForward(){
  motor[llauncher] = -127;
  motor[rlauncher] = -127;
}
void LauncherStop(){
```

```
  motor[llauncher] = 0;
  motor[rlauncher] = 0;
}
void UpdateEncoders(){
  EncoderReadTime = time1[T1];
  LaunchEncoderValue = nMotorEncoder[llauncher];
  //PrevLaunchEncoderValue = LauncherEncoderValue;
  //PrevEncoderReadTime = EncoderReadTime;
  //LauncherEncoderValue = nMotorEncoder[llauncher];
  //EncoderReadTime = time1[T1];
  //EncoderChange = LauncherEncoderValue-PrevLaunchEncoderValue;
  //LoopTime = EncoderReadTime-PrevEncoderReadTime;
  LauncherAngularVelocity = ((float)LaunchEncoderValue/(float)EncoderReadTime);

  //nxtDisplayCenteredTextLine(7, "Change: %d",EncoderChange);
  //nxtDisplayCenteredTextLine(3, "lenc: %d", LaunchEncoderValue);
  nxtDisplayCenteredTextLine(2, "AngVel:␣%f", LauncherAngularVelocity);
  //nxtDisplayCenteredTextLine(7, "Time: %d", EncoderReadTime);
  //nxtDisplayCenteredTextLine(6, "Loop: %d", LoopTime);
  //nxtDisplayCenteredTextLine(5, "PrevEnc: %d",PrevLaunchEncoderValue);
  nMotorEncoder[llauncher] = 0;
  ClearTimer(T1);
}
task main()
{
  nMotorEncoder[llauncher] = 0;
  nMotorEncoder[rlauncher] = 0;

  while(true)
  {

    //each stick controls a motor
    getJoystickSettings(joystick);
    if(time1[T1]>200){
      UpdateEncoders();
    }


    if (driveType == 0) {

      if(joystick.joy1_y1<20 && joystick.joy1_y1>-20){
        motor[left] = 0;
        motor[left2] = 0;
        }else{
        motor[left] = -joystick.joy1_y1;
        motor[left2] = joystick.joy1_y1;
      }
      if(joystick.joy1_y2<20 && joystick.joy1_y2>-20){
        motor[right] = 0;
        motor[right2] = 0;
```

```
    }else{
    motor[right] = joystick.joy1_y2;
    motor[right2] = joystick.joy1_y2;
  }
  } else if (driveType == 1) {
  int leftIn = joystick.joy1_y1;
  int leftPwr = leftIn * leftIn / 127;
  if (leftIn < 0) {
    leftPwr = leftPwr * -1;
  }
  motor[left] = -leftPwr;
  motor[left2] = leftPwr;
  int rightIn = joystick.joy1_y2;
  int rightPwr = rightIn * rightIn / 127;
  if (rightIn < 0) {
    rightPwr = rightPwr * -1;
  }
  motor[right] = rightPwr;
  motor[right2] = rightPwr;
}

if(joy1Btn(6)){
  //right top button (RB)
  //  motor[intake] = -127;
  LauncherForward();
  }else if(joy1Btn(8)){
  //right bottom button (RT)
  //motor[intake] = 127;
  LauncherReverse();
  }else{
  if(joy2Btn(6)){
    //right top button (RB)
    //  motor[intake] = -127;
    if((time1[T2] > 1000)&&(LauncherAngularVelocity>-0.0001)){
      LauncherStop();
    }
    LauncherForward();
    }else if(joy2Btn(8)){
    //right bottom button (RT)
    //motor[intake] = 127;
    LauncherReverse();
    }else{
    //motor[intake] = 0;
    LauncherStop();
    ClearTimer(T2);
  }

  //motor[intake] = 0;

}
```

```
    if(joy1Btn(2)){
      servo[grabber] = 100;
      }else if(joy1Btn(1)){
      servo[grabber] = 160;
    }
    if(joy1Btn(3)){
      servo[opener] = 0;
      }else if(joy1Btn(4)){
      servo[opener] = 255;
    }
    //added jan 7
    if(joy1Btn(5)){
      motor[lifter] = 100;
      }else if(joy1Btn(7)){
      motor[lifter] = -100;
    }
    else{
      motor[lifter] = 0;
    }
  }
}
```

Autonomous:

```
#pragma config(Hubs,  S1, HTMotor,  HTMotor,  HTMotor,  HTMotor)
#pragma config(Hubs,  S2, HTServo,  none,     none,     none)
#pragma config(Sensor, S1,     ,               sensorI2CMuxController)
#pragma config(Sensor, S2,     ,               sensorI2CMuxController)
#pragma config(Motor,  mtr_S1_C1_1,     left2,         tmotorTetrix, PIDControl
#pragma config(Motor,  mtr_S1_C1_2,     lifter,        tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C2_1,     left,          tmotorTetrix, PIDControl
#pragma config(Motor,  mtr_S1_C2_2,     intake,        tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C3_1,     launcher,      tmotorTetrix, openLoop)
#pragma config(Motor,  mtr_S1_C3_2,     right,         tmotorTetrix, PIDControl
#pragma config(Motor,  mtr_S1_C4_1,     right2,        tmotorTetrix, PIDControl
#pragma config(Motor,  mtr_S1_C4_2,     motorK,        tmotorTetrix, openLoop)
#pragma config(Servo,  srvo_S2_C1_1,    grabber,
tServoStandard)
#pragma config(Servo,  srvo_S2_C1_2,    opener,
tServoStandard)
#pragma config(Servo,  srvo_S2_C1_3,    servo3,
tServoNone)
#pragma config(Servo,  srvo_S2_C1_4,    servo4,
tServoNone)
#pragma config(Servo,  srvo_S2_C1_5,    servo5,
tServoNone)
#pragma config(Servo,  srvo_S2_C1_6,    servo6,
tServoNone)
//Team 7347 Nick Vosseteig

#include "JoystickDriver.c"

void initializeRobot() {
  servo[grabber] = 0;
  servo[opener] = 188;
  return;
}

void drive(int speed) {
  motor[left] = speed;
  motor[right] = speed;
  motor[left2] = speed;
  motor[right2] = speed;
}

task main() {
  initializeRobot();
  waitForStart(); // Wait for the beginning of autonomous phase.
  drive(-50);
  wait10Msec(200);
  drive(0);
}
```