VATT Thermal System

The VATT Thermal System is controlled by an Arduino Mega board located in the mirror cell. It is connected to 7 analog 8D590K thermal sensors. Four of these sensors are located on the telescope struts and 3 are in the mirror cell. The Arduino board communicates with vatttel through an RS-232 serial port. vatttel (TCS) requests the thermal information from each thermal sensor over the serial port by sending a 't' followed a carriage return '\r'. The arduino board then sends a string of white space separated temperatures.

Thermal Daughter Board

The thermal daughter board is a circuit board that sits on top of the arduino board. It has 1 serial port 2 mux chips and 3 Phoenix Connectors. Each Mux chip has four channels to handle the analog signal coming from the 7 thermal sensors.

Mux Channel	GUI Name	Wire Label	Phoenix Connector	Signal Wire Color
00	Mirror Temp	T1	P9	White
01	Mirror Air Temp	T0	P9	Red
02	Mirror Temp	T2	P9	Red
03	Air Temp	T3	P10	Red
10	Strut Temp	T4	P10	White
11	Strut Temp	T5	P10	Red
11	Air Temp	T6	P11	White

Firmware

Firmware as of 9-3-2014

```
#include <Wire.h>
#include <Adafruit_ADS1015.h>

#define NCHAN 8//Number of channels.
#define AVGCNTS 100.0//Number of values to average over
#define MV_PER_KEL 10.0 //Resistor value
#define KEL2CEL -272.1
#define MV_PER_CNT 0.188
#define KELS_PER_CNT MV_PER_CNT/MV_PER_KEL

/*This is the firmware that resides on the Arduino mMega
VATT Thermal system board. It reads in temperatures from
seven analog sensors located on the VATT Telescope. When
Querried with a 't\r' via its serial port  it returns a string
of 7 white space seperated temperatures followed by a '\n'

The order of the sensor data located in the return string is as follows:

MIRROR_TEMP MIRROR_AIR_TEMP MIRROR_TEMP AIR_TEMP STRUT_TEMP STRUT_TEMP
```

```
AIR_TEMP

For more information on the VATT thermal system please go to this website:
https://soweb.as.arizona.edu/~tscopewiki/doku.php?id=vatt:mirror_cell_therma
l_reader


Author Scott Swindell
Date 9-3-2014
*/



Adafruit_ADS1115 ads1( 0x48 );  //first mux chip
Adafruit_ADS1115 ads2( 0x49 );  //Second mux chip

char inChar;//The charactre read in from the serial port.
char lastInChar;

long adc_sum[NCHAN];//keeps a sum of the temperature readings for averaging
int16_t adc[NCHAN];//Stores averaged temperature readings
int inVal;

int val13;//For toggling pin 13
short sum_iter = 1;//keeps count of how many we have summed



//Offsets. Most of this is counter-acting the hard coded offsets in
//vatttel
float temp_offsets[NCHAN] = {0.4, 2.7, 0.4, -1.5, 3.3, 3.3, -1.5, 0.0};



void setup(void)
{
  val13 = HIGH;
  pinMode(13, OUTPUT);
  digitalWrite( 13, val13 );
  //Serial.begin(9600);
  Serial3.begin(9600);
 /* Serial.println("Hello!");
  Serial.println("Getting single-ended readings from AIN0..3");
  Serial.println("ADC Range: +/- 6.144V (1 bit = 3mV/ADS1015,
0.1875mV/ADS1115)");*/
  // The ADC input range (or gain) can be changed via the following
  // functions, but be careful never to exceed VDD +0.3V max, or to
  // exceed the upper and lower limits if you adjust the input range!
  // Setting these values incorrectly may destroy your ADC!
```

```
  //                                                                ADS1015
ADS1115
  //                                                                -------
-------
   ads1.setGain(GAIN_TWOTHIRDS);  // 2/3x gain +/- 6.144V  1 bit = 3mV
0.1875mV (default)
   ads2.setGain(GAIN_TWOTHIRDS);
  // ads.setGain(GAIN_ONE);        // 1x gain   +/- 4.096V  1 bit = 2mV
0.125mV
  // ads.setGain(GAIN_TWO);        // 2x gain   +/- 2.048V  1 bit = 1mV
0.0625mV
  // ads.setGain(GAIN_FOUR);       // 4x gain   +/- 1.024V  1 bit = 0.5mV
0.03125mV
  // ads.setGain(GAIN_EIGHT);      // 8x gain   +/- 0.512V  1 bit = 0.25mV
0.015625mV
  // ads.setGain(GAIN_SIXTEEN);    // 16x gain  +/- 0.256V  1 bit = 0.125mV
0.0078125mV
  //ads2.setGain(GAIN_TWOTHIRDS); //
  ads1.begin();
  ads2.begin();
}

void loop(void)
{
  /*In the arduino loop function we do the reading of the
   *analogue temp sensors and keep a sum of each channel
   *After ACGCNTS number of loops we average the last previous data and
   *store it in the adc array.
   */
  //read first mux chip and sum it with last previous readings
  for (short channel_num=0; channel_num<( 4 ); channel_num++)
  {
    inVal = ads1.readADC_SingleEnded( channel_num );
    adc_sum[channel_num] = inVal  + adc_sum[channel_num];
  }
  //read the second mux chip and average it with last reading
  for (short channel_num=0; channel_num<( 4 ); channel_num++)
  {
     inVal = ads2.readADC_SingleEnded( channel_num );
     adc_sum[channel_num + 4] = inVal + adc_sum[channel_num+4];
     //delay(10);
     //if( channel_num+4 == 6 ){ Serial.println(inVal); }
  }
  if (sum_iter == AVGCNTS)
  {
    for( short channel_num=0; channel_num<NCHAN; channel_num++ )
    {
      //Now lets do the averaging
      adc[channel_num] = adc_sum[channel_num]/AVGCNTS;
      adc_sum[channel_num] = 0;
      sum_iter = 1;
```

```
    }
  }
  else
  {
    sum_iter++;
  }
}
/*commented out for now but may be used later.
 *If it works it is a bit more sophisticated
 *than the serialEvent3 below it.
void serialEvent3()
{
  while ( Serial3.available() )
  {
    while(inChar != '\n' && inChar != '\r')
    {//read until a carriage return or new line.
      lastInChar = inChar;
      inChar = (char) Serial3.read();
    }
    if(lastInChar == 't')
    {//Did a  't' come before the carriage return?
      delay(500);//Do we need a delay?
      printTemps( adc );
      lastInChar = '\0';
      toggle13();
    }
    inChar = '\0';
  }
}*/

void serialEvent3()
{
  /*this function is called anytime serial port 3
  gets data.  */
  while ( Serial3.available() )
  {
    inChar = Serial3.read();
    //As soon as we get a carriage
    //return a 't\r' assuming because
    //vatttel is expecting an echo
    //Then proc
    //serial port with printTemps
    //function.
    if( inChar == '\r' )
    {
      toggle13();//toggle the light for debug
      //Serial3.print('t\r');
      //vatttel expects an echo.
      printTemps( adc );
```

```
      printTemps( adc );
    }

  }
}

/*
*Name: printTemps
*args: tempArr (list of average temperatures from the adc )
*Decsr: Prints white space seperated temperatures from the
*  VATT mirror cell sensors to the Serial port which is connected
*  vatttel
*Author: Scott Swindell
*/
void printTemps( int16_t tempArr[NCHAN] )
{
  char charBuff[10];
  String strBuff;
  float temp_celsius;
  //There are NCHAN channels but we are only using 7 of them.
  for( short i=0; i<NCHAN-1; i++ )
  {//add 7 used channels to the string (NCHAN-1)
    temp_celsius = ( tempArr[i]*MV_PER_CNT + temp_offsets[i] )/MV_PER_KEL +
KEL2CEL;
    dtostrf( temp_celsius, 3, 1,  charBuff );//convert float to string
    strBuff = strBuff +  charBuff + ' ';//Build string of temperatures as
String type
  }
  strBuff.trim();//Remove trailing whitespace
  strBuff = strBuff + '\n';//add new line
  Serial3.print( strBuff );//Send off the data!
  //Serial3.print("1.0 2.0 3.0 4.0 5.0 6.0 7.0\n");

}



/*Name: Toggle13
 *args: N/A
 *Descr:  toggles pin 13 between high and low
 *  (and therefore an LED) on the arduino board
 *  this is very usefull for debugging
 *Author: Scott Swindell
 *
 */
void toggle13()
{
  if( val13 == HIGH )
  {
    val13 = LOW;
    digitalWrite( 13, val13 );
  }
```

```
  else
  {
    val13 = HIGH;
    digitalWrite( 13, val13 );
  }
}
```