```
// FILE: alias1_0.nxc - Works!
// DATE: 09/15/16 09:23
// AUTH: P.Oh
// DESC: Capture sine wave into file (called alias.csv.  Demonstrate aliasing
//       User specifies sampling time and desired maximum elapsed time;
// REFS: timer1_0.nxc, celsius1_2.nxc
// USES: Uses Brick's Port 1's WHITE (AN) and BLACK (GND) lines
//       Connect function generator into WHITE (AN) line
//       Treats WHITE and BLACK lines as input into Brick's internal 10-bit ADC
//       Measure a 1 Hz signal; DC offset of 2.5V; peak-to-peak voltage 4.5V
//       Capture 5 seconds worth of data

#include "fileSavingFunctionsForAlias.h"

task main() {

  // Boolean related variables - Brick buttons to start/stop execution
  bool orangeButtonPushed, greyButtonPushed;

  // Timing related variables
  long prevTick, curTick, deltaTick; // previous, current and difference in ticks
  string strDeltaTick; // string form of deltaTick for file writing
  float deltaTickInSeconds;  // NB: deltaTick in [msec]
  float elapsedTimeInSeconds; // elapsed time in [sec]
  string strElapsedTimeInSeconds; // string form of elapsed time for file writing
  float maxElapsedTimeInSeconds; // total time in [sec] for data acquisition
  float samplingTimeInSeconds;
  float waitingTimeInSeconds; // for delay loop
  long  waitingTimeInMilliSeconds;
  float epsilonTime = 0.0001; // maxElapsedTimeInSeconds - elapsedTimeInSeconds
will be almost zero

  // function generator and ADC related variables
  int touchSensorRawValue; // a number between 0 and 1023 (10-bit ADC)
  float voltageValue; // in [V]
  string strVoltageValue;

  // Create a new file to capture values... call the write-to-file function
  InitWriteToFile();

  // Prompt user to begin
  TextOut (0, LCD_LINE1, "Orange Btn starts");
  do {
     orangeButtonPushed = ButtonPressed(BTNCENTER, FALSE);
  } while(!orangeButtonPushed);

  ClearScreen();
  TextOut(0, LCD_LINE1, "Grey Btn Stops");

  // Initialize timing- and calculation-related variables
  // References for a 1 Hz frequency input:
  //   10 samples/sec (10 Hz)  = 0.10 sec = 100 ms  - sampling 1 Hz should be fine
  //    5 samples/sec (5 Hz)   = 0.20 sec = 200 msec - sampling 1 Hz should be fine
  //    2 samples/sec (2 Hz)   = 0.50 sec = 500 msec - sampling 1 Hz should be
borderline
  //    1 samples/sec (0.5 Hz) = 1 sec = 1000 msec - sampling 1 Hz should be poor

  samplingTimeInSeconds = 1.0; // sampling time in [sec]
<------------
  maxElapsedTimeInSeconds = 5.0; // total time of data acquisition
<------------
  elapsedTimeInSeconds = 0.0; // set elapsed time to zero
  prevTick = CurrentTick();
```

```
  SetSensorTouch(IN_1);  // homemade touch sensor on Brick Port 1

  do {
     greyButtonPushed = ButtonPressed(BTNEXIT, FALSE);
     TextOut(0, LCD_LINE6, FormatNum("Time = %5.3f s" , elapsedTimeInSeconds));

     // perform desired work i.e. capture function generator voltage
     touchSensorRawValue = SensorRaw(IN_1); // read raw value at port
     TextOut(0, LCD_LINE3, FormatNum("Raw Value: %d", touchSensorRawValue));
     voltageValue = (touchSensorRawValue * 5)/1023; // in [V]
     TextOut(0, LCD_LINE4, FormatNum("Voltage = %3.3f [V]" , voltageValue));
     strVoltageValue = FormatNum("%5.2f" , voltageValue);

     // check how much time to wait for sampling interval to elapse
     curTick = CurrentTick(); // get current tick count
     deltaTick = curTick - prevTick; // measure elapsed ticks [msec]
     deltaTickInSeconds = deltaTick/1000.0; // in [sec]
     waitingTimeInSeconds = samplingTimeInSeconds - deltaTickInSeconds;
     waitingTimeInMilliSeconds = waitingTimeInSeconds * 1000;
     Wait(waitingTimeInMilliSeconds);

     // sampling time interval has past, so capture time and write to file
     curTick = CurrentTick();
     deltaTick = curTick - prevTick; // measure elapsed ticks [msec]
     deltaTickInSeconds = deltaTick/1000.0; // in [sec]
     elapsedTimeInSeconds = elapsedTimeInSeconds + deltaTickInSeconds; // in [sec]
     strElapsedTimeInSeconds = FormatNum("%5.3f" , elapsedTimeInSeconds);
     text=StrCat(strElapsedTimeInSeconds, "," , strVoltageValue);

     WriteToFile(text);

     // Update current tic value
     prevTick = curTick;

  } while( ((maxElapsedTimeInSeconds - elapsedTimeInSeconds) >= epsilonTime) &&
!greyButtonPushed);
  PlayTone(400,200);
  TextOut(0, LCD_LINE4, "Done!");

  StopAllTasks();
  StopWriteToFile();

} // end main
```