```
! ------------------------------------------------------------------------
! GRAPHICAL USER INTERFACE (GUI) for CV and SWV measurements
! This is a test program for UWED to make CV and SWV recordings
! and enter parameters on graphical user interface
! Written by Julia Redston, 2016. Whitesides Group, Harvard.
! ------------------------------------------------------------------------

! ------------------------------------------------------------------------
! VARIABLES AND OBJECTS
! ------------------------------------------------------------------------

! -- Labels and text boxes for inputs
dim textBox1 AS TextField , textBox2 AS TextField, textBox3 AS TextField
dim textBox4 AS TextField, textBox5 AS TextField, textBox6 AS TextField
dim label1 AS Label, label2 AS Label, label3 AS Label, label4 AS Label
dim label5 AS Label, label6 AS Label
dim label8 AS Label
dim hzLabel AS Label
dim fileName as TextField
dim startButton AS Button
dim stopButton AS Button
dim settingsButton AS Button
dim firstUni AS INTEGER
firstUni = 0
dim textNote as Label
dim textCycles AS TextField
dim labelCycles AS Label
dim segControl as SegmentedControl
mdone=0
dim dirControl as SegmentedControl
dim cycleControl as SegmentedControl
dim pulseAmp AS DOUBLE
DIM labelPulse AS Label
DIM pulseText AS TextField
DIM startLabel AS Label
DIM startText AS TextField
dim startVol as DOUBLE
Dim stringFile as STRING

! -- Program parameters --
DIM upperVoltage AS Double ! [V] This is upper voltage value
DIM lowerVoltage AS Double ! [V] This is lower voltage value
numberOfCycles = 2! This is number of full cycles
DIM scanRate AS Double ! [mV/s]
DIM timeStep AS Double ! [ms] timestep between measuruments divided by 2.

DIM initW AS LONG
initW=22500 ! This is now constant and not calculated
DIM initR AS LONG
initR=22606 ! This should start as close to zero as possible
Rslope=1.5040E4 ! This is number of digits per volt when Ref is changed

DIM numElec !number of electrons

DIM StepHeightAbs AS INTEGER !absolute value of step height
DIM stepR(50) AS INTEGER
DIM nrSteps(50) AS INTEGER
DIM nrDataPoints AS LONG
```

```
DIM nrScans AS INTEGER
dim pulseAmpD AS  INTEGER

! Resulting data storage
DIM Vw(10000) AS DOUBLE ! Potential in volts
DIM IuA(10000) AS DOUBLE ! Electrode current in uA
DIM dWe(10000) AS LONG ! Working electrode as set
DIM dRe(10000) AS LONG ! Reference electrode as set
DIM dADC(10000) AS LONG ! ADC value
DIM Ce(10000) AS INTEGER ! Counter electrode potential (as character)
DIM Ni(10000) AS INTEGER ! Index of point in scan
DIM DPi AS INTEGER ! Data point counter
DIM SCi AS INTEGER ! Scan counter
DIM ScanRunning AS INTEGER ! if 0 does not run, if 1 running
ScanRunning=0
DIM lastIndex AS INTEGER
lastIndex=0

DIM TS(10000) AS LONG

! - Graphs visualization
DIM p1 AS Plot          ! This is the result plot
DIM pCV AS PlotPoint    ! This is one dataset
DIM myCV(10000,2)       !holds all cv points
DIM myCV2(10000,2)      ! holds all square wave points
DIM graphArray(2000,2)  !holds the most recent 2000 points,
! this is what is displayed on graphed

! ------------------------------------------------------------------------
! SETUP GUI AND APPLICATION
! ------------------------------------------------------------------------

p1=Graphics.newPlot         ! create new plot
pCV=p1.newPlot(graphArray)  !set to the Graphing Array
!p1.setTitle("Electrochemical Measurements")
p1.setXaxisLabel("Ref - Work")
p1.setYaxisLabel("Current")

! this allows you to zoom and move around graph, but not click points
p1.setAllowedGestures($F0FF)

p1.setRect( Graphics.width/2 ,Graphics.height/2 - 250 , _
            Graphics.width/2 + 200 ,Graphics.width/2 + 200)
p1.setXaxisLabel("Potential Working vs Ref (V)")
p1.setYaxisLabel("Current working electrode (µA)")
p1.setTitle("Electrochemical Measurements with RFUWED")
textNote = Graphics.newLabel(10, Graphics.height/30,160)

! This forces the app to be locked in the horizontal orientation
! Application shall be launched horizontal
System.setAllowedOrientations(2)

! - initialize and properties of all the buttons and text boxes -

startButton = Graphics.newButton(10,Graphics.height/15)
startButton.setTitle("Start")
startButton.setBackgroundColor(.85, .85, .85)
```

```
stopButton = Graphics.newButton(120,Graphics.height/15)
stopButton.setTitle("Stop")
stopButton.setBackgroundColor(.85, .85, .85)

settingsButton = Graphics.newButton(230,Graphics.height/15)
settingsButton.setTitle("Save Settings")
settingsButton.setBackgroundColor(.85, .85, .85)

segControl = Graphics.newSegmentedControl(10, Graphics.height/15 *2, 200, 30)
segControl.insertSegment("CV",1,0)
segControl.insertSegment("Square Wave",2,0)
segControl.setSelected(1)

label2 = Graphics.newLabel(10, Graphics.height/15  *3,160)
label2.setText("Voltage Upper(V)")

textBox2 =  Graphics.newTextField( 150, Graphics.height/15 *3)
textBox2.setBackgroundColor(.85,.85,.85)

label3 = Graphics.newLabel(10, Graphics.height/15  *4,160)
label3.setText("Voltage Lower(V)")

textBox3 =  Graphics.newTextField( 150, Graphics.height/15 *4)
textBox3.setBackgroundColor(.85,.85,.85)

label5 = Graphics.newLabel(10, Graphics.height/15  *5,160)
label5.setText("Scan Rate(mV/s)")

textBox5 =  Graphics.newTextField( 150, Graphics.height/15 *5)
textBox5.setBackgroundColor(.85,.85,.85)

label6 = Graphics.newLabel(10, Graphics.height/15  *6,160)
label6.setText("Delta Time(ms)")

textBox6 =  Graphics.newTextField( 150, Graphics.height/15 *6)
textBox6.setBackgroundColor(.85,.85,.85)

hzLabel = Graphics.newLabel(250, Graphics.height/15  *6,120)
hzLabel.setText("Hz ")

labelCycles = Graphics.newLabel(10, Graphics.height/15  *7,160)
labelCycles.setText("Num Scans")

textCycles =  Graphics.newTextField( 150, Graphics.height/15 *7)
textCycles.setBackgroundColor(.85,.85,.85)

label8 = Graphics.newLabel(10, Graphics.height/15  *8,160)
label8.setText("File Name")

fileName =  Graphics.newTextField( 150, Graphics.height/15 *8, 180)
fileName.setBackgroundColor(.85,.85,.85)

labelPulse = Graphics.newLabel(10, Graphics.height/15  *9,160)
labelPulse.setText("Pulse Amp(mV)")

pulseText =  Graphics.newTextField( 150, Graphics.height/15 *9)
pulseText.setBackgroundColor(.85,.85,.85)
labelPulse.setHidden(1)
```

```
pulseText.setHidden(1)

startLabel = Graphics.newLabel(10, Graphics.height/15  *9,160)
startLabel.setText("Start Value(V)")

startText =  Graphics.newTextField( 150, Graphics.height/15 *9)
startText.setBackgroundColor(.85,.85,.85)

dirControl = Graphics.newSegmentedControl(10, Graphics.height/15 *10, 200, 30)
dirControl.insertSegment("Negative ->",2,0)
dirControl.insertSegment("Positive <-",1,0)
dirControl.setSelected(1)

dirControl.setHidden(1)

cycleControl = Graphics.newSegmentedControl(10, Graphics.height/15 *11, 200, 30)
cycleControl.insertSegment("Unidirectional",1,0)
cycleControl.insertSegment("Bidirectional",2,0)
cycleControl.setSelected(1)

cycleControl.setHidden(1)

System.ShowGraphics(1) ! SHOW GRAPHICS

! creating the file settings
! if the settings file exists load the saved settings
if( EXISTS("UWEDSettings.txt")) then
    OPEN "UWEDSettings.txt" FOR INPUT AS #2
    LINE INPUT #2, a$
    if a$ = "Square Wave" then
        segControl.setSelected(2)
        LINE INPUT #2, a$
        textBox2.setText(a$)
        LINE INPUT #2, a$
        textBox3.setText(a$)
        LINE INPUT #2, a$
        textBox5.setText(a$)
        LINE INPUT #2, a$
        textBox6.setText(a$)
        IF NOT (textBox6.getText = "") then
            hzLabel.setText(STR(250/VAL(textBox6.getText))&" Hz")
        end if
        LINE INPUT #2, a$
        textCycles.setText(a$)
        LINE INPUT #2, a$

        pulseText.setText(a$)
        labelPulse.setHidden(0)
        pulseText.setHidden(0)
        dirControl.setHidden(0)
        cycleControl.setHidden(0)
        startLabel.setHidden(1)
        startText.setHidden(1)
    else
        segControl.setSelected(1)
        LINE INPUT #2, a$
        textBox2.setText(a$)
        LINE INPUT #2, a$
```

```
        textBox3.setText(a$)
        LINE INPUT #2, a$
        textBox5.setText(a$)
        LINE INPUT #2, a$
        textBox6.setText(a$)
        IF NOT (textBox6.getText = "") then
            hzLabel.setText(STR(250/VAL(textBox6.getText))&" Hz")
        end if
        LINE INPUT #2, a$
        textCycles.setText(a$)
        LINE INPUT #2, a$
        startText.setText(a$)

        labelPulse.setHidden(1)
        pulseText.setHidden(1)
        dirControl.setHidden(1)
        cycleControl.setHidden(1)
        startLabel.setHidden(0)
        startText.setHidden(0)
    end if
    CLOSE #2
end if

! -- BLE --
!starts scanning for ble
DIM bm AS BLEPeripheral
ble.startble
DIM uuid(0) AS STRING
DIM services(1) AS STRING
DIM bleWriteChar AS BLECharacteristic
ble.startscan(uuid)

END

! ------------------------------------------------------------------------
! FUNCTIONS
! ------------------------------------------------------------------------

!Initialize UWED
! Gets the UWED ready to start, sends constants
SUB InitializeUWED()
    DPi=0 ! No data points yet
    SCi=0 ! Scan counter
    ! Make graph empty
    FOR i=1 TO nrDataPoints
        myCV(i,1)=0
        myCV(i,2)=0
        myCV2(i,1)=0
        myCV2(i,2)=0
    NEXT i
    ! Start setting up
    sendToUWED("A("&STR(initR - Rslope*startVol)&")")
    sendToUWED("E("&STR(timeStep)&")")
    sendToUWED("B("&STR(initW)&")")
    sendToUWED("D(3)")
    sendToUWED("C(1)")
END SUB
```

```
! Initialize a scan
! called by either touchupinside or procdata
! performs 1 scan if scans havent completed
! saves data to file if scans have been completed
SUB InitializeScan()
    SCi=SCi+1 ! Increase scan counter
    system.wait(.1) ! waiting 1/10 a second here prevents skipping data points
    print "on scan ";SCI
    textNote.setText("on scan "&STR(SCI))
    IF SCi>nrScans THEN ! If increase scan counter is larger, DONE
        !save to file
        print "SCI larger than nrScans "
        print SCi; " > "; nrScans
        textNote.setText("writing to file")
        print "writing to file"
        sendToUWED("C(0)") ! Put UWED to potentiometric (off state)
        ! Save data
        if NOT( fileName.getText = "" ) then    ! if a file name has been given
            stringFile = fileName.getText & "-raw.txt"
            dim stringFile2 AS String
            stringFile2 = fileName.getText & "-proc.txt"
            OPEN stringFile FOR OUTPUT AS #1 !create raw text file
            OPEN stringFile2 FOR OUTPUT AS #3 !create processed text file
            !first deal with processed data, saves the actual data points graphed
            !aka the data in myCV and myCV2
            IF segControl.title(segControl.selected) = "Square Wave" then
                PRINT #1, "Square Wave "
                PRINT #3, "Square Wave "
                for p=1 to (DPi/2)
                    PRINT #3, STR(myCV2(p,1))&", "STR(myCV2(p,2))!
                next p
            else
                PRINT #1, "CV "
                PRINT #3, "CV "
                print "dpi " &STR(DPi)
                for p=1 to DPi-1
                    PRINT #3, STR(myCV(p,1))&", "STR(myCV(p,2))!
                next p
            end if
            ! raw data
            ! saves reference electrode (dRe), working electrode (DWe), ADC value (dADC)
            ! counter electrode potential (Ce) and index of point (Ni)
            FOR i=1 TO DPi
                PRINT #1, STR(dRe(i))&" "&STR(DWe(i))&" "&STR(dADC(i))&_
                " "&STR(Ce(i))&" "&STR(Ni(i))
            NEXT i
            CLOSE #1
            CLOSE #3
            print "written to file"
        end if
        ! Send email now
        ! if you dont want to send an email comment out all of this
        ! sends from the built in "mail" app and whatever email is saved there
        if NOT( fileName.getText = "" ) then
            ! only sends email with file if there is a file
            DIM em AS Email
            DIM em_mess AS STRING !this is the message of the data
            DIM em_sub AS STRING !this is the subject of the data
```

```
            DIM em_add AS STRING !this is who the email is sent to
            ! IF DATA IS ALWAYS BEING SENT TO SAME ADDRESS
            ! YOU SHOULD EDIT THIS LINE TO ADDRESS BEING SENT TO
            ! example em_add = "johnsmith@gmail.com"
            em_add=""
            em_mess="This is data from UWED measurement"
            em_sub= "UWED Data"
            em=System.newEmail(em_add, em_sub, em_mess)
            ! attaches the processed and raw data files
            em.addAttachment(stringFile, "text/plain")
            em.addAttachment(stringFile2, "text/plain")
            em.send
        end if
        print "done"
        textNote.setText("DONE")
        ScanRunning=0
    ELSE ! this is not the last scan, send data to UWED
        ! Increase same amount both steps
        print "step height "; stepR(SCi)
        print "numsteps ";nrSteps(SCi)
        if segControl.title(segControl.selected) = "CV"  then
            sendToUWED("G("&STR(stepR(SCi))&")")
            sendToUWED("I("&STR(stepR(SCi))&")")
        else
            if cycleControl.title(cycleControl.selected) = "Unidirectional"  then
                sendToUWED("A("&STR(initR - Rslope*startVol)&")")
                firstUni =1
            end if
            sendToUWED("G("&STR(0-2*pulseAmpD)&")")
            sendToUWED("I("&STR(2*stepR(SCi)+2*pulseAmpD)&")")
        end if
        sendToUWED("L("&STR(nrSteps(SCi))&")")
        sendToUWED("M()") ! Run the sequence
        ScanRunning=1
    END IF
END SUB

! - Process incoming data
! called by BLECharacteristicinfo
! takes in the integer array sent by the UWED
! converts that array into a data point and graphs it
SUB procData(input_array() AS INTEGER)
    PRINT "Index: "&CHR(input_array(1))&CHR(input_array(2))
        ! Feedback, so we can see continuity
    ! First check that there is no dublication
    IF(input_array(1)=lastIndex) THEN
        GOTO OutSub ! If same as last time do nothing
    END IF
    lastIndex=input_array(1) ! Otherwise put it same
    ! Now look if new data came M character on second position
    IF(input_array(2)=ASC("M"))THEN ! New data came
        IF ScanRunning=1 THEN ! Only if scan is running there is something to come
            DPi=DPi+1
            dRe(DPi)=ConvertToNumberBin(input_array,3,4)
            dADC(DPi)=ConvertToNumberBin(input_array,5,6)
            DPi=DPi+1
            dRe(DPi)=ConvertToNumberBin(input_array,7,8)
            dADC(DPi)=ConvertToNumberBin(input_array,9,10)
```

```
        Ts(DPi)=ConvertToNumberBin(input_array,11,14)
        Ts(DPi-1)=Ts(DPi)
        dWe(DPi)=ConvertToNumberBin(input_array,15,16)
        dWe(DPi-1)=dWe(DPi)
        Ni(DPi)=ConvertToNumberBin(input_array,17,18)
        Ni(DPi-1)=Ni(DPi)
        ! Do the math for visualization
        ! ODD
        VoltW=dRe(DPi-1)*(-6.64898E-5)+1.50310  ! Working electrode pot vs ref
        Curr=0.00608*(dADC(DPi-1)-30694)-0.04 ! Current in uA
        myCV(DPi-1,1)=VoltW
        myCV(DPi-1,2)=Curr
        ! EVEN
        VoltW=dRe(DPi)*(-6.64898E-5)+1.50310  ! Working electrode pot vs ref
        Curr=0.00608*(dADC(DPi)-30694)-0.04 ! Current in uA
        myCV(DPi,1)=VoltW
        myCV(DPi,2)=Curr
        if segControl.title(segControl.selected) = "CV" then ! CV calculations
            graphArray((DPi MOD 2000 )+ 1 , 1) = myCV(DPi,1)
            graphArray((DPi MOD 2000 )+ 1 , 2) = myCV(DPi,2)
            graphArray(((DPi -1) MOD 2000 )+ 1 , 1) = myCV(DPi-1,1)
            graphArray(((DPi -1)MOD 2000 )+ 1 , 2) = myCV(DPi-1,2)
            ! Show graph
            if( DPi MOD 3 = 0 ) then ! only update graph every 3 points
                pCV.setPoints(graphArray)
            end if
        else ! Square wave calculations
            ! if its the first data point in a unidirectional sweep
            ! then set points to 0
            ! prevents noise
            if firstUni = 1  then
                myCV2(DPi/2,1) = (myCV(DPi,1)+myCV(DPi-1,1))/2
                myCV2(DPi/2,2) = 0
                firstUni = 0
            else ! if its not the first point of unidirectional
                ! then perform calculations for square wave
                myCV2(DPi/2,1)=(myCV(DPi,1)+myCV(DPi-1,1))/2 ! average two
                myCV2(DPi/2,2)=myCV(DPi,2)-myCV(DPi-1,2) ! different between two
                graphArray((DPi/2 MOD 2000 )+ 1 , 1) = myCV2(DPi/2,1)
                    ! put into the graphing array
                graphArray((DPi/2 MOD 2000 )+ 1 , 2) = myCV2(DPi/2,2)
                print myCV2(DPi/2,1);", "; myCV2(DPi/2,2)
                !  print "dpi "; dpi
                print "dpi mod ";( (DPi/2 MOD 2000) +1)
            end if
            ! Show graph
            if( DPi MOD 3 = 0) then ! only update graph every 3 points
                pCV.setPoints(graphArray)
            end if
        end if
    END IF
END IF
! If end of the scan start new
IF (input_array(2)=ASC("m")) THEN ! End of scan
    InitializeScan ! Start next scan
END IF
! In other cases do currently nothing
OutSub:
```

```
END SUB

! - Convert to Number
! called by procdata converts integer array into characters then those chars into a long
FUNCTION ConvertToNumber(input_array() AS INTEGER, _
begi AS INTEGER, _
endi AS INTEGER) AS LONG
    DIM nets AS STRING
    nets=""
    FOR i=begi TO endi
    nets=nets & chr(input_array(i))
    NEXT i
    ConvertToNumber=VAL(nets)
END FUNCTION

! ------------------------------------------------------------------------
! EVENTS
! ------------------------------------------------------------------------

! Called, when button is clicked
! Takes in the button that was clicked and time
SUB touchUpInside(ctrl AS Button, when AS DOUBLE)
    mdone=0
    DIM writeValue(1) AS INTEGER
    if NOT (bm = NULL) AND NOT (bleWriteChar = NULL) then ! if ble is connected
        if ctrl = stopButton then ! if stop button send "STOP" command to UWED
            sendToUWED("N()")
        else if ctrl = settingsButton then
            ! if its the settings button, save current settings
            ! save stuff to file
            print "writing settings file"
            OPEN "UWEDSettings.txt" FOR OUTPUT AS #2
            if segControl.title(segControl.selected) = "Square Wave" then
                PRINT #2, "Square Wave"
                PRINT #2, textBox2.getText ! upper voltage
                PRINT #2, textBox3.getText ! lower voltage
                PRINT #2, textBox5.getText !scan rate
                PRINT #2, textBox6.getText !time Step
                PRINT #2, textCycles.getText ! num cycles
                PRINT #2, pulseText.getText ! pulse amp
            else
                PRINT #2, "CV"
                PRINT #2, textBox2.getText ! upper voltage
                PRINT #2, textBox3.getText ! lower voltage
                PRINT #2, textBox5.getText !scan rate
                PRINT #2, textBox6.getText !time Step
                PRINT #2, textCycles.getText ! num cycles
                PRINT #2, startText.getText ! start vol
                CLOSE #2
            end if
        else if ctrl = startButton then
            ! if start button, make calculations start first scan
            DPi =0
            ! clear graphing array
            for p= 1 to 2000
                graphArray(p,1) =0
                graphArray(p,2)=0
            NEXT P
```

```
upperVoltage= VAL(textBox2.getText) ! [V] This is upper voltage value
lowerVoltage= VAL(textBox3.getText) ! [V] This is lower voltage value
scanRate=VAL(textBox5.getText )     ! [mV/s] change in voltage per second
timeStep=VAL(textBox6.getText)
    ! [ms] time between each step (2x for cv 4x for square)
nrScans = VAL(textCycles.getText)   ! number of scans
numberOfCycles = (nrScans -1 )/2    ! number of cycles
print "numscans total ";nrScans
nrDataPoints=2*numberOfCycles*(nrSteps2+1)
p1.setView(lowerVoltage, -50, upperVoltage, 50, 0)

IF segControl.title(segControl.selected) = "Square Wave" then ! if square wave
    ! calculate number of steps
    nrSteps1=1000000*ABS(upperVoltage - lowerVoltage)/(scanRate*timeStep*2)
    ! calculate stepheight
    stepHeightAbs=ABS((upperVoltage-lowerVoltage)*Rslope)/nrSteps1
    ! calculate number data points
    nrDataPoints=nrSteps1+2

    IF(dirControl.title(dirControl.selected) = "Negative ->")THEN ! negative
        stepR(1)=(-1)*stepHeightAbs !starts in negative direction
        startVol = lowerVoltage !starts at lower voltage
    ELSE ! positive
        stepR(1)=stepHeightAbs ! starts positive direction
        startVol = upperVoltage !starts at the upper voltage
    END IF
    nrSteps(1)=nrSteps1
    nrDataPoints=nrSteps1+2
    pulseAmp = VAL(pulseText.getText)
    pulseAmpD = (pulseAmp/1000)*Rslope  !Pulse amplitutude
    DIM myCV2(nrDataPoints)

    dim directionalVec as INTEGER
        ! controls if switch directions every scan or not
    directionalVec = -1
    if cycleControl.title(cycleControl.selected) = "Unidirectional" then
        directionalVec = 1
    end if
    print "dirVec ";directionalVec
    for i = 1 TO nrScans
        nrSteps(i) = nrSteps1
        if i MOD 2 = 0 then
            stepR(i) = (directionalVec * stepR(1))
            print "stepR at ";i;" ";stepR(i)
        else
            stepR(i) = stepR(1)
        end if
    NEXT i
else ! CV calculations
    DIM myCV(nrDataPoints)
    startVol = 0
    if(NOT (startText.getText = "")) then !if start text is not empty
        startVol = VAL(startText.getText)
    end if
    ! calculate num steps from 0 to upper voltage
    nrSteps1=1000000*(upperVoltage)/(scanRate*timeStep*2)
    dim nrSteps1Start as double
    ! calculate num steps from start voltage to upper voltage
```

```
                nrSteps1Start=1000000*(upperVoltage-startVol)/(scanRate*timeStep*2)
                ! calculate num steps from lower voltage to uper voltage
                nrSteps2=1000000*(upperVoltage-lowerVoltage)/(scanRate*timeStep*2)
                ! calculate num steps from lower voltage to 0
                nrSteps3=nrSteps2-nrSteps1
                print "nrstep1 "; nrSteps1
                print "nrstep2 "; nrSteps2
                print "nrstep3 "; nrSteps3
                ! calculate step height
                stepHeightAbs=ABS(((upperVoltage-lowerVoltage)* Rslope)/nrSteps2)
                print "step height abs ";stepHeightAbs
                print "upper Vol "; upperVoltage
                print"lower Vol "; lowerVoltage
                ! start going negative direction
                stepR(1)=(-1)*stepHeightAbs
                print "step height at 1 ";stepR(1)
                ! number of steps for first scan is
                ! num steps from start voltage to upper voltage
                nrSteps(1)=nrSteps1Start
                print "numsteps at 1 ";nrSteps(1)
                ! sets stepheight for odd and even scans
                FOR i=1 TO ((nrScans - 1 )/2 )
                    stepR(i*2)= stepHeightAbs
                    print "step height at "; (i*2);" ";stepR(i*2)
                    stepR(i*2+1)=(-1)*stepHeightAbs
                    print "step height at "; (i*2+1);" ";stepR(i*2+1)
                    nrSteps(i*2)=nrSteps2
                    print "numsteps at "; (i*2);" ";nrSteps(i*2)
                    nrSteps(i*2+1)=nrSteps2
                    print "numsteps at "; (i*2+1);" ";nrSteps(i*2+1)
                NEXT i
                ! sets step height for the last scan (from lower to 0)
                nrSteps(nrScans)=nrSteps3
                print "numsteps at ";nrScans;" ";nrSteps3
            END IF
            InitializeUWED
            InitializeScan
        END IF
    END IF
END SUB

! BLE EVENTS and FUNCTIONS
! - Send command to UWED
SUB sendToUWED(inp AS STRING)
    DIM ax(LEN(inp)) AS INTEGER
    FOR i=1 TO LEN(INP)
        ax(i)=ASC(MID(inp,i,1))
    NEXT
    bm.writeCharacteristic(bleWriteChar,ax)
END SUB

! - First when device is discovered
SUB BLEDiscoveredPeripheral (   time AS DOUBLE, _
                                peripheral AS BLEPeripheral, _
                                services() AS STRING, _
                                advertisements(,) AS STRING, _
                                rssi)
    PRINT "Device Found";
```

```
    IF peripheral.bleName="UWED" THEN ! Right device found
        bm=peripheral
        ble.connect(bm)
        ble.stopScan
        print "Device is UWED"
    END IF
END SUB

! - When information about the device is fetched
SUB BLEPeripheralInfo(  time AS DOUBLE, _
                        peripheral AS BLEPeripheral, _
                        kind AS INTEGER, _
                        message AS STRING, _
                        err AS LONG)
    IF kind=1 THEN ! Connection completed
        peripheral.discoverServices(uuid)
    ELSE IF kind=2 OR kind=3 THEN ! Connection lost
        ble.connect(bm)
    ELSE IF kind=4 THEN ! Service dound
        DIM avServ(1) AS bleservice
        avServ=peripheral.services
        FOR a=1 TO UBOUND(services,1)
            FOR a=1 TO UBOUND(services,1)
                IF avServ(a).uuid="2220" THEN ! If right service id has been found
                    peripheral.discoverCharacteristics(uuid, avServ(a))
                    print "Service discovered"
                END IF
            NEXT
        NEXT
    END IF
END SUB

! - When information about service is obtained
SUB BLEServiceInfo(     time AS DOUBLE, _
                        peripheral AS BLEPeripheral, _
                        service AS BLEService, _
                        kind AS INTEGER, _
                        message AS STRING, _
                        err AS LONG)
    IF kind=1 THEN
        DIM chx(1) AS blecharacteristic
        chx=service.characteristics
        FOR i=1 TO UBOUND(chx,1)
            IF service.uuid="2220" THEN
                IF chx(i).uuid="2221" THEN
                    peripheral.setNotify(chx(i),1) ! Notify if changes
                    print "Read characteristic found"
                ELSE IF chx(i).uuid="2222" THEN
                    bleWriteChar=chx(i)
                    print "Write characteristic found"
                END IF
                textNote.setText("BLE CONNECTED!")
            END IF
        NEXT
    END IF
END SUB

! - If BLE Characteristic info comes
```

```
! when UWED sends back data
SUB BLECharacteristicInfo(      time AS DOUBLE, _
                                peripheral AS BLEPeripheral, _
                                characteristic AS BLECharacteristic, _
                                kind AS INTEGER, _
                                message AS STRING, _
                                err AS LONG)
    IF kind=2 THEN
        IF characteristic.uuid="2221" THEN
            DIM value(1) AS INTEGER
            value=characteristic.value
            procData(value)
        END IF
    END IF
END SUB

! ------------------------------------------------------------------------
! HELPING FUNCTIONS
! ------------------------------------------------------------------------

! currently not used
! could be used to autoscale the graph, but it makes the program run much slower
FUNCTION minY(inarray() as DOUBLE) AS DOUBLE
    a=inarray(1,2)
    for i=1 to DPi
        if a>inarray(i,2) then
            a=inarray(i,2)
        end if
    next
    minY=a
end function

! currently not used
! could be used to autoscale the graph, but it makes the program run much slower
function maxY(inarray() as DOUBLE) AS DOUBLE
    a=inarray(1,2)
    for i=1 to DPi
        if a<inarray(i,2) then
            a=inarray(i,2)
        end if
    next
    maxY=a
end function

! - Convert to Number from String source - currently not used
FUNCTION ConvertToNumberStr(input_array() AS INTEGER, _
                            begi AS INTEGER, _
                            endi AS INTEGER) AS LONG
    DIM nets AS STRING
    nets=""
    FOR i=begi TO endi
        nets=nets & chr(input_array(i))
    NEXT i
    ConvertToNumberStr=VAL(nets)
END FUNCTION

! - Convert to Number from Binary source - used in data receiving
FUNCTION ConvertToNumberBin(input_array() AS INTEGER, _
```

```
                          begi AS INTEGER, _
                          endi AS INTEGER) AS LONG
    DIM sum AS LONG
    sum=input_array(begi)
    FOR i=begi+1 TO endi
        sum=sum*256
        sum=sum+input_array(i)
    NEXT i
    ConvertToNumberBin=sum
END FUNCTION

! - when the segcontrol's value is changed (when a tab is clicked)
SUB valueChanged (  ctrl AS Control, _
                    time AS DOUBLE)
    IF ctrl = segControl then ! change segControl value
        ! if set to CV, make appropriate textboxes etc visible
        IF segControl.title(segControl.selected) = "CV" then
            labelPulse.setHidden(1)
            pulseText.setHidden(1)
            dirControl.setHidden(1)
            cycleControl.setHidden(1)
            startLabel.setHidden(0)
            startText.setHidden(0)
        Else
            ! if set to Square Wave, make appropriate textboxes etc visible
            labelPulse.setHidden(0)
            pulseText.setHidden(0)
            dirControl.setHidden(0)
            cycleControl.setHidden(0)
            startLabel.setHidden(1)
            startText.setHidden(1)
        END IF
    ELSE IF ctrl = textBox6 then
        hzLabel.setText((250/VAL(textBox6.getText))&" Hz")
    END IF
END SUB

! - END OF THE CODE -
```