```
! ------------------------------------------------------------------------
! CHRONOAMPEROMETRY (CA)
! This is a test program for UWED to make basic CA recordings
! Written by Alar Ainla, 2016-2018. Whitesides Group, Harvard.
! ------------------------------------------------------------------------

! ------------------------------------------------------------------------
! OPERATION PARAMETERS
! ------------------------------------------------------------------------

InitV=-0.1  ! [V] This is measurement potential
timeStep=25 ! [ms] timestep between measurements divided by 2.
! E.g. this is 50ms timestep between measurements
mTime=30    ! [s] Total measurement time

! ------------------------------------------------------------------------
! PROGRAM CODE
! ------------------------------------------------------------------------

DIM nrDataPoints AS LONG
nrDataPoints=mTime*1000/(2*timeStep) ! Total number of points
DIM nrScans AS INTEGER
nrScans=CINT(nrDataPoints/1000)+1 ! 1000 per scan (1000 points is maximum per scan)
DIM time0 AS LONG

! Here we lock working electrode potential W[d]=22500 W[V]=1.5031
! And only varied is Reference electrode potential
! Constants then according to calibration
! ADC[d]=(ADC[V])*2.01513E4+2.38491E2
! ADC[V]=(ADC[d]-2.38491E2)/2.01513E4
! Curr[uA]=0.00608*(dADC-30694)-0.04 ! Working electrode current
! (W-R)[V]=R[d]*(-6.64898E-5)+1.50310 ! Working electrode potential vs ref
Rslope=1.5040E4 ! This is number of digits per volt, when Ref is changed

! Calculate required data structure
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
initR=22606-Rslope*InitV ! This is the potential value
initR0=22606

nrSteps1=1000
nrSteps2=nrDataPoints MOD 1000

DIM nrSteps(nrScans) AS INTEGER
FOR i=1 TO nrScans-1
 nrSteps(i)=nrSteps1
NEXT i
nrSteps(nrScans)=nrSteps2
! Output these to console for feedback
PRINT "InitW"
PRINT initW
PRINT "InitR"
PRINT initR
PRINT "stepR"
PRINT stepR
PRINT "nrSteps"
```

```
PRINT nrSteps
PRINT "nrScans"
PRINT nrScans
PRINT "nrDataPoints"
PRINT nrDataPoints

! Resulting data storage arrays
DIM Vw(nrDataPoints) AS DOUBLE ! Potential in volts
DIM IuA(nrDataPoint) AS DOUBLE ! Electrode current in uA
DIM dWe(nrDataPoints) AS LONG ! Working electrode as set digital value
DIM dRe(nrDataPoints) AS LONG ! Reference electrode as set digital value
DIM dADC(nrDataPoints) AS LONG ! ADC value
DIM Ts(nrDataPoints) AS LONG ! Time step
DIM Ni(nrDataPoints) 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 ! Last receiveing Index, to avoid double reading
lastIndex=0

! -- GUI --
! - Initialize basic GUI -
DIM startButton AS Button ! This button starts the measurement
DIM myMess AS Label ! This is text message area
DIM p1 AS Plot ! This is the result plot
DIM pCA AS PlotPoint ! This is one dataset
DIM myCA(nrDataPoints,2)
FOR i=1 TO nrDataPoints
    myCA(i,1)=0
    myCA(i,2)=0
NEXT i
p1=Graphics.newPlot
pCA=p1.newPlot(myCA)
p1.setView(0,-100,mTime,100,0)
p1.setXaxisLabel("Time (s)")
p1.setYaxisLabel("Current working electrode (µA)")
p1.setTitle("Electrochemical CA with UWED")
startButton=Graphics.newButton(10,10)
startButton.setTitle("Start")
myMess=Graphics.newLabel(110,10,200,20)
myMess.setText("....")
System.ShowGraphics(1)

! -- BLE --
print "Hello 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
```

```
! ------------------------------------------------------------------------

SUB InitializeUWED()
    DPi=0 ! No data points yet
    SCi=0 ! Scan counter
    ! Make graph empty
    FOR i=1 TO nrDataPoints
        myCA(i,1)=0
        myCA(i,2)=0
    NEXT i
    ! Start setting up experiment in UWED
    sendToUWED("A("&STR(initR)&")")
    sendToUWED("B("&STR(initW)&")")
    sendToUWED("E("&STR(timeStep)&")")
    sendToUWED("D(3)")
    sendToUWED("C(1)")
END SUB

SUB InitializeScan()
    SCi=SCi+1 ! Increase scan counter
    IF SCi>nrScans THEN ! If scan counter is larger than number of scans, then DONE
        sendToUWED("A("&STR(0)&")")
        sendToUWED("B("&STR(0)&")")
        system.wait(0.5) ! Wait a bit
        sendToUWED("C(0)") ! Put UWED to potentiometric (off state)
        ! Save data
        OPEN "CAData.txt" FOR OUTPUT AS #1
        FOR i=1 TO DPi
            PRINT #1, STR(dRe(i))&" "&STR(DWe(i))&" "&STR(dADC(i))&_
                " "&STR(Ts(i))&" "&STR(Ni(i))
        NEXT i
        CLOSE #1
        ! Send now data over email to server
        DIM em AS Email
        DIM em_mess AS STRING
        em_mess="This is data from CA measurement from iPhone"
        em=System.newEmail("email@emailserver.com","CA Data", em_mess)
        em.addAttachment("CAData.txt", "text/plain")
        em.send
        myMess.setText("DONE")
        ScanRunning=0
    ELSE ! Initialize new scan
        ! Increase same amount both steps
        sendToUWED("G("&STR(0)&")") ! Do not change potential
        sendToUWED("I("&STR(0)&")") ! Do not change potential
        sendToUWED("L("&STR(nrSteps(SCi))&")") ! Number of steps
        sendToUWED("M()") ! Run the sequence
        ScanRunning=1
    END IF
END SUB

! - Process incoming data
SUB procData(input_array() AS INTEGER)
    PRINT "Index: "&CHR(input_array(1)) ! Feedback to console
    ! 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)
    ! Now look if new data came M character on second position (This is data then)
    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)-timeStep
            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 potential vs ref
            Curr=0.00608*(dADC(DPi-1)-30694)-0.04 ! Current in uA
            IF DPi=2 THEN
                time0=Ts(1)
            END IF
            myCA(DPi-1,1)=(Ts(DPi-1)-time0)/1000
            myCA(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
            myCA(DPi,1)=(Ts(DPi)-time0)/1000
            myCA(DPi,2)=Curr
            ! Show graph
            pCA.setPoints(myCA)
            p1.setView(0,minY(myCA),mTime,maxY(myCA),0)
        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 nothing
OutSub:
END SUB

! - 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

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

! When button is clicked
SUB touchUpInside(crtl AS Button, when AS DOUBLE)
    IF crtl=startButton THEN ! Send now
        myMess.setText("Start the scan")
        InitializeUWED
        InitializeScan
    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 connect to the device
        bm=peripheral
        ble.connect(bm)
        ble.stopScan ! Do not look any further
        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
                myMess.setText("BLE CONNECTED!")
            END IF
        NEXT
    END IF
END SUB

! - If BLE Characteristic info comes
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
```

```
! ------------------------------------------------------------------------

! - Find minimum value in the array (for auto scaling of plots)
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

! - Find maximum value in the array (for auto scaling of plots)
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

! - END OF THE CODE -
```