Suggested Updates

- Next to avaspec.py, the globals.py should be copied as well.
- The first command in Python should be “from avaspec import *”
- The call to AVS_GetList should be a_pList = AVS_GetList(ret)
- Then ishould be “handle = AVS_Activate(a_pList[0])”
- Then the call for getting the config should be “config = AVS_GetParameter(handle, 63484)”
- Then “pixels = config.m_Detector.m_NrPixels”
- Then “wavelengths = AVS_GetLambda(handle)”
- The for loop and wavelengths.append line should be removed
- Create the MeasConfigType using “measconfig = MeasConfigType()” (Note the () !)
- Use “ret = AVS_PrepareMeasure(handle, measconfig)” and check the value of ret, it should be 0. On any other value, starting the measurement will not succeed.
- Grab the data:
  ret = AVS_GetScopeData(handle)
  timestamp = ret[0]
  spectraldata = ret[1]

And if you would like to have a complete python script included on this webpage, this script will work (although it doesn’t show the actual data in a graph):

```
from avaspec import *
import time

AVS_EnableLogging(true)
AVS_Init(0)
ret = AVS_GetNrOfDevices()
ret
req = 75 * ret
a_pList = AvsIdentityType * ret
a_pList = AVS_GetList(ret)
a_pList[0].SerialNumber, a_pList[0].Status
handle = AVS_Activate(a_pList[0])
config = DeviceConfigType
config = AVS_GetParameter(handle, 63484)
pixels = config.m_Detector_m_NrPixels
wavelengths = AVS_GetLambda(handle)
measconfig = MeasConfigType()
measconfig.m_StartPixel = 0
measconfig.m_StopPixel = pixels - 1
measconfig.m_IntegrationTime = 50
measconfig.m_IntegrationDelay = 0
measconfig.m_NrAverages = 1
```

Suggested Updates

- Next to avaspec.py, the globals.py should be copied as well.
- The first command in Python should be “from avaspec import *”
- The call to AVS_GetList should be a_pList = AVS_GetList(ret)
- Then ishould be “handle = AVS_Activate(a_pList[0])”
- Then the call for getting the config should be “config = AVS_GetParameter(handle, 63484)”
- Then “pixels = config.m_Detector.m_NrPixels”
- Then “wavelengths = AVS_GetLambda(handle)”
- The for loop and wavelengths.append line should be removed
- Create the MeasConfigType using “measconfig = MeasConfigType()” (Note the () !)
- Use “ret = AVS_PrepareMeasure(handle, measconfig)” and check the value of ret, it should be 0. On any other value, starting the measurement will not succeed.
- Grab the data:

  ret = AVS_GetScopeData(handle)
  timestamp = ret[0]
  spectraldata = ret[1]

And if you would like to have a complete python script included on this webpage, this script will work (although it doesn’t show the actual data in a graph):

```
from avaspec import *
import time

AVS_EnableLogging(true)
AVS_Init(0)
ret = AVS_GetNrOfDevices()
ret
req = 75 * ret
a_pList = AvsIdentityType * ret
a_pList = AVS_GetList(ret)
a_pList[0].SerialNumber, a_pList[0].Status
handle = AVS_Activate(a_pList[0])
config = DeviceConfigType
config = AVS_GetParameter(handle, 63484)
pixels = config.m_Detector_m_NrPixels
wavelengths = AVS_GetLambda(handle)
measconfig = MeasConfigType()
measconfig.m_StartPixel = 0
measconfig.m_StopPixel = pixels - 1
measconfig.m_IntegrationTime = 50
measconfig.m_IntegrationDelay = 0
measconfig.m_NrAverages = 1
```

```
measconfig.m_CorDynDark_m_Enable = 0
measconfig.m_CorDynDark_m_ForgetPercentage = 0
measconfig.m_Smoothing_m_SmoothPix = 0
measconfig.m_Smoothing_m_SmoothModel = 0
measconfig.m_SaturationDetection = 0
measconfig.m_Trigger_m_Mode = 0
measconfig.m_Trigger_m_Source = 0
measconfig.m_Trigger_m_SourceType = 0
measconfig.m_Control_m_StrobeControl = 0
measconfig.m_Control_m_LaserDelay = 0
measconfig.m_Control_m_LaserWidth = 0
measconfig.m_Control_m_LaserWaveLength = 0.0
measconfig.m_Control_m_StoreToRam = 0

ret = AVS_PrepareMeasure(handle, measconfig)
ret
if ret == 0:
    scans = 1
    AVS_Measure(handle, 0, scans)
    dataready = False
    while not dataready:
        dataready = AVS_PollScan(handle)
        time.sleep(measconfig.m_IntegrationTime/1000)
    ret = AVS_GetScopeData(handle)
    timestamp = ret[0]
    spectraldata = ret[1]
    for i in range(pixels):
        print(spectraldata[i])

AVS_Deactivate(handle)
AVS_Done()
```