Function	**Info**	**Return Value**
IV_MaxDevices	Number of channels that can be controlled by one instance	Number of Channels
IV_getdevicestatus	This returns the status of the device	-1 = IviumSoft not running 0 = Not connected / No Device 1 = Available Idle 2 = Available Running
IV_selectdevice(int)	Select the device by channel number	No return value
IV_SelectSn(string)	Select the device by serial number	Returns the channel number of the SN
IV_HostHandle	Returns the handle of the controlled IviumSoft	Handle of the controlled IviumSoft instance
IV_VersionDll	Return the version of the dll	Version of DLL
IV_VersionCheck	Checks if the version of the DLL matches the IviumSoft version	0 = not matched 1= matched
IV_VersionDllFile	Return the version of the dll	Build version of the DLL file
IV_open	Opens the driver	0 = OK 1 = No Client
IV_close	Closes the driver	0 = OK
IV_readSN(*char)	Returns the serial number of the selected device	0 = OK 1 = No Device
IV_connect(int)	Connect to the selected device (0 = disconnect, 1 = connect)	0 = OK 1 = Could not connect 3 = Invalid State (device is already connected / disconnected / no device)
IV_VersionHost(int)		
IV_getcellstatus(int)	Returns cell status: bit 2=I_ovl bit 4 =Anin1_ovl bit 5 = E_ovl bit 7 = CellOff_button pressed	0 = OK 3 = Invalid State (device is not idle)

IV_setconnectionmode(int)	Select configuration 0 = off 1 = EStat4EL (default) 2 = EStat2EL 3 = Estat Dummy1 4 = Estat Dummy2 5 = Estat Dummy3 6 = EstatDummy4 7 = Istat4EL 8 = Istat2EL 9 = IstatDummy 10 = BiStat4EL 11 = BiStat2EL	0 = OK 3 = Invalid State (device is not idle)
IV_setcellon(int)	Set cell on off to close cell relay 0 = Off, 1 = On	0 = OK 3 = Invalid State (device is not idle)
IV_setpotential(double)	Set cell potential	0 = OK 3 = Invalid State (device is not idle)
IV_setpotentialWE2(double)	Set BiStat offset potential	0 = OK 3 = Invalid State (device is not idle)
IV_setcurrent(double)	Set cell current (Galvanostatic mode)	0 = OK 3 = Invalid State (device is not idle)
IV_getpotential(double)	Get measured potential	0 = OK 3 = Invalid State (device is not idle)
IV_setcurrentrange(int)	Set current range, 0 = 10A, 1 = 1A, etc,	0 = OK 3 = Invalid State (device is not idle)
IV_setcurrentrangeWE2(int)	Set current range for BiStat, 0 = 10mA, 1 = 1mA, etc,	0 = OK 3 = Invalid State (device is not idle)
IV_getcurrent(double)	Get measured current	0 = OK 3 = Invalid State (device is not idle)
IV_getcurrentWE2(double)	Get measured current from WE2 (BiPotentiostat)	0 = OK 3 = Invalid State

		(device is not idle)
IV_setfilter(int)	Set filter, 0 = 1MHz, 1 = 100kHz, 2 = 10kHz, 3 = 1kHz, 4 = 10Hz	0 = OK 3 = Invalid State (device is not idle)
IV_setstability(int)	Set stability, 0 = HighSpeed, 1 = Standard, 2 = HighStability	0 = OK 3 = Invalid State (device is not idle)
IV_setbistatmode(int)	Select mode for BiStat, for int 0=standard, 1=scanning *This bistat_mode function also controls the Automatic E-ranging function of the instrument; 0 = AutoEranging off; 1 = AutoEranging on*	0 = OK 3 = Invalid State (device is not idle)
IV_setdac(int, double)	Set analog output on external port, int=0 for AnOut1, int=1 for AnOut2	0 = OK 3 = Invalid State (device is not idle)
IV_getadc(int, double)	Get measured voltage on analog input on external port, int = channelnr. (0-7)	0 = OK 3 = Invalid State (device is not idle)
IV_setmuxchannel(int)	Set channel of multiplexer, int = channelnr. (starting from 0)	0 = OK 3 = Invalid State (device is not idle)
IV_setdigout(int)	Set digital lines on external port, int = bitmask	0 = OK 3 = Invalid State (device is not idle)
IV_getdigin(int)	Get status of digital inputs on external port, int = bitmask	0 = OK 3 = Invalid State (device is not idle)
IV_setfrequency(double)	Set ac frequency, double = frequency in Hz	0 = OK 3 = Invalid State (device is not idle)
IV_setamplitude(double)	Set ac amplitude, double = amplitude in Volt	0 = OK 3 = Invalid State (device is not idle)
IV_getcurrenttrace(int, double, *double)	Get a sequence of measured currents at defined samplingrate (npoints, interval,	0 = OK 3 = Invalid State

	array of double): npoints<=256, interval: 10us to 20ms	(device is not idle)
IV_getcurrentWE2trace(int, double, *double)	Get a sequence of measured WE2 currents at defined samplingrate (npoints, interval, array of double): npoints<=256, interval: 10us to 20ms	0 = OK 3 = Invalid State (device is not idle)
IV_getpotentialtrace(int, double, *double)	Get a sequence of measured potentials at defined samplingrate (npoints, interval, array of double): npoints<=256, interval: 10us to 20ms	0 = OK 3 = Invalid State (device is not idle)
IV_we32setchannel(int)	Select active WE32 channel int = channelnr. (starting from 0)	0 = OK 3 = Invalid State (device is not idle)
IV_we32setoffset(int, double)	Set WE32 offset int = channelnr.,double = value, value -2 to +2V. Use chan=0 to apply the same offset to all channels.	0 = OK 3 = Invalid State (device is not idle)
IV_we32setoffsets(int, *double)	Set WE32 offset int = number of channels, array of double = value, value -2 to +2V.	0 = OK 3 = Invalid State (device is not idle)
IV_we32getoffsets(int, *double)	Get WE32 offset values, int = number of channels, array of double = offset values	0 = OK 3 = Invalid State (device is not idle)
IV_we32readcurrents(*double)	Get all WE32 channels current values	0 = OK 3 = Invalid State (device is not idle)
IV_readmethod(*char)	Loads method procedure from disk, char = filename	0 = OK 2 = File Not Found 3 = Invalid State (device is not idle)
IV_savemethod(*char)	Saves method procedure to disk, char = filename	0 = OK 3 = Invalid State (device is not idle)
IV_startmethod(*char)	Start method procedure. If char is empty then presently loaded procedure is used, else the procedure is loaded from disk.	0 = OK 2 = File Not Found 3 = Invalid State (device is not idle)

IV_abort	Abort the ongoing method procedure	0 = OK 3 = Invalid State (device is not measuring)
IV_savedata(*char)	Saves actual result data to disk, with char as filename	0 = OK 3 = Invalid State (device is not idle)
IV_setmethodparameter(*char1, *char2)	0 = OK 3 = Invalid State (device is not idle), 4 = Invalid Parameter	
IV_Ndatapoints	Get actual available number of datapoints: indicates progress during a run	
IV_getdata(int, double1, double2, double3)	Read datapoint with index int, returns 3 doubles (double1/double2/double3) that represent measured values depending on the used technique, for example LSV/CV methods return (E/I/0) Transient methods return (time/I,E/0), Impedance methods return (Z1,Z2,freq) etc.	-1 = Requested number of points exceeds available number of points 0 = OK
IV_getdatafromline(int1, int2, double1, double2, double3)	Same as IV_readdata, but with the additional int2 parameter which specifies the scannr. This function will allow reading data from non-selected (previous) scans	-1 = Requested number of points exceeds available number of points or scannr. Exceeds number of scans. 0 = OK,
IV_getDbFileName(*char)	Get filename of the database	
IV_StatusParGet(int)	Get the status of the global status variable. Can be used for synchronization between channels.	0 = OK 3 = Invalid State (device is not available)
IV_StatusParSet(int)	Set the status of the global status variable. Can be used for synchronization between channels.	0 = OK 3 = Invalid State (device is not available)
IV_UpdateTemperature(double)	When temperature sharing is active a temperature can be shared (beta)	0 = OK 3 = Invalid State (device is not available)