HFD-6 SDK User Manual

/**********Standard SDK**********/

int hfdGetDeviceCount()

Function description: Return the number of compatible haptic devices connected to the system. This encompasses all devices connected locally, including devices already locked by other applications. Devices are given a unique identifier.

Parameters: void

Returns: the number of devices connected on success, -1 otherwise

int hfdGetAvailableCount()

Function description: Return the number of available haptic devices connected to the system. This encompasses all devices connected locally, but excludes devices already locked by other applications. Devices are given a unique identifier.

Parameters: void

Returns: the number of devices available on success, -1 otherwise

int hfdSetDevice(char ID)

Function description: Select the default device that will receive the SDK commands. The SDK supports multiple devices. Any subsequent SDK call that does not specifically mention the device ID in its parameter list will be sent to that device.

Parameters: @ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetDeviceID()

Function description: Return the ID of the current default device.

Parameters: void

Returns: device ID on success, 0 otherwise

int hfdOpen ()

Function description: Open a connection to the first available device connected to the system. The order in which devices are opened persists until devices are added or removed.

Parameters: void

Returns: The device ID on success, 0 otherwise

int hfdOpenID(char index)

Function description: Open a connection to one particular device connected to the system. The order in which devices are opened persists until devices are added or removed.

Parameters: @ID[in]: the device ID (must be between 0 and the number of devices connected to the system)

@ index[in]:

Returns: The device ID on success, 0 otherwise

int hfdClose (char ID= -1)

Function description: Close the connection to a particular device.

Parameters: @ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdInit (char ID= -1)

Function description: Initialize the specified device.

Parameters: @ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdCalibrateDevice (char ID= -1)

Function description: Get calibration value from device memory and then calibrate the device.

Parameters: @ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdEnableForce (HFDuchar val, char ID= -1)

Function description: Enable force mode for the device controller.

Parameters:

@ val[in]: HFD_ON to enable force mode, HFD_OFF to disable it

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetOperationMode (HFDuchar* a_mode, char ID= -1)

Function description: HFD_ON to enable force, HFD_OFF to disable it

Parameters:

@ a_mode[out]: pointer to the operation mode

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetButton (int index, char ID= -1)

Function description: Return the status of the button located on the end-effector.

Parameters:

@ index[in]: button index, default=0, 0 indicates button below, 1 indicates button above.

@ ID[in]: device ID, default= -1

Returns: HFD_ON if the button is pressed, HFD_OFF otherwise, or -1 on error.

int hfdSetBrakes (int val=HFD_ON, char ID= -1)

Function description: Enable/disable the device electromagnetic brakes.

Parameters:

@ val[in]: default= HFD_ON, desired state of the brakes (HFD_ON or HFD_OFF)

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetPosition (double*px, double*py, double*pz, char ID= -1)

Function description: Retrieve the position of the end-effector in Cartesian coordinates. Please refer to your device user manual for more information on your device coordinate system.

Parameters:

@ px[out]: device position on the X axis in[mm]

@ py[out]: device position on the Y axis in[mm]

@ pz[out]: device position on the Z axis in[mm]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetOrientationRad (double*oa, double*ob, double*og, char ID= -1)

Function description: Retrieve individual angle of each joint of the WRIST, starting with the one nearest to the hand.

Parameters:

@ oa[out]: device orientation around the Y axis in [rad]

@ ob[out]: device orientation around the X axis in [rad]

@ og[out]: device orientation around the Z axis in [rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetOrientationDeg (double*oa, double*ob, double*og, char ID= -1)

Function description: Retrieve individual angle of each joint of the WRIST, starting with the one nearest to the hand.

Parameters:

@ oa[out]: device orientation around the Y axis in [deg]

@ ob[out]: device orientation around the X axis in [deg]

@ og[out]: device orientation around the Z axis in [deg]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetOrientationFrame (double matrix[3][3], char ID= -1)

Function description: Retrieve the rotation matrix of the WRIST structure.

Parameters:

@ matrix[out]: orientation matrix frame

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetPositionAndOrientationRad (double*px, double*py, double*pz, double*oa, double*ob, double*og, char ID= -1)

Function description: Retrieve the position and orientation of the end-effector in Cartesian coordinates.

Parameters:

@ px[out]: device position on the X axis in [mm]

@ py[out]: device position on the Y axis in [mm]

@ pz[out]: device position on the Z axis in [mm]

@ oa[out]: device orientation around the Y axis in[rad]

@ ob[out]: device orientation around the X axis in[rad]

@ og[out]: device orientation around the Z axis in[rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetPositionAndOrientationDeg (double*px, double*py, double*pz, double*oa, double*ob, double*og, char ID= -1)

Function description: Retrieve the position and orientation of the end-effector in Cartesian coordinates.

Parameters:

@ px[out]: device position on the X axis in [mm]

@ py[out]: device position on the Y axis in [mm]

@ pz[out]: device position on the Z axis in [mm]

@ oa[out]: device orientation around the Y axis in [deg]

@ ob[out]: device orientation around the X axis in [deg]

@ og[out]: device orientation around the Z axis in [deg]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetPositionAndOrientationFrame (double*px, double*py, double*pz, double*oa, double*ob, double*og, char ID= -1)

Function description: Retrieve the position and orientation matrix of the end-effector in Cartesian coordinates.

Parameters:

@ px[out]: device position on the X axis in [mm]

@ py[out]: device position on the X axis in [mm]

@ pz[out]: device position on the X axis in [mm]

@ matrix[out]: orientation matrix frame

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetForce (double*fx, double*fy, double*fz, char ID= -1)

Function description: Retrieve the force vector applied to the end-effector.

Parameters:

@ fx[out]: force on the X axis in [N]

@ fy[out]: force on the Y axis in [N]

@ fz[out]: force on the Z axis in [N]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSetForce (double fx, double fy, double fz, char ID= -1)

Function description: Set the desired force vector in Cartesian coordinates to be applied to the end-effector of the device.

Parameters:

@ fx[in]: force on the X axis in [N]

@ fy[in]: force on the Y axis in [N]

@ fz[in]: force on the Z axis in [N]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetForceAndTorque (double*fx, double*fy, double*fz, double*tx, double*ty, double* tz, char ID= -1)

Function description: Retrieve the force and torque vectors applied to the device end-effector.

Parameters:

@ fx[out]: force on the X axis in [N]

@ fy[out]: force on the Y axis in [N]

@ fz[out]: force on the Z axis in [N]

@ tx[out]: torque around the X axis in[mNm]

@ ty[out]: torque around the X axis in[mNm]

@ tz[out]: torque around the X axis in[mNm]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSetForceAndTorque (double fx, double fy, double fz, double tx, double ty, double tz, char ID= -1)

Function description:

Parameters:

@ fx[out]: force on the X axis in [N]

@ fy[out]: force on the Y axis in [N]

@ fz[out]: force on the Z axis in [N]

@ tx[out]: torque around the X axis in[mNm]

@ ty[out]: torque around the X axis in[mNm]

@ tz[out]: torque around the X axis in[mNm]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

double hfdGetComFreq (char ID= -1)

Function description: Return the communication refresh rate between the computer and the device.

Parameters: @ ID[in]: device ID, default= -1

Returns: the refresh rate in [Hz], 0.0 otherwise

int hfdGetLinearVelocity (double*vx, double*vy, double*vz, char ID= -1)

Function description: Retrieve the estimated instantaneous translational velocity.

Parameters:

@ vx[out]: velocity along the X axis[mm/s]

@ vy[out]: velocity along the Y axis[mm/s]

@ vz[out]: velocity along the Z axis[mm/s]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetAngularVelocityRad (double*wx, double*wy, double*wz, char ID= -1)

Function description: Retrieve the estimated instantaneous angular velocity in [rad/s].

Parameters:

@ wx[out]: angular velocity around the Y axis [rad/s]

@ wy[out]: angular velocity around the X axis [rad/s]

@ wz[out]: angular velocity around the Z axis [rad/s]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetAngularVelocityDeg (double*wx, double*wy, double*wz, char ID= -1)

Function description:

Parameters:

@ wx[out]: angular velocity around the Y axis [deg/s]

@ wy[out]: angular velocity around the X axis [deg/s]

@ wz[out]: angular velocity around the Z axis [deg/s]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdErrorGetLast (unsigned short* errcode)

Function description: Retrieve the last error code encountered in the running thread

Parameters: @ errcode[out]: the last error code encountered in the running thread

Returns: 1 on success, 0 otherwise

const char* hfdErrorGetLastStr ()

Function description: Returns a brief string describing the last error encountered in the running thread

Parameters: void

Returns: A pointer to a character array which describe the error

const char*hfdErrorGetStr (int error)

Function description: Returns a brief string describing a given error code

Parameters:

@ error[in]: error code

Returns: A pointer to a character array which describe the error

void hfdClearError ()

Function description: Refresh error information in HFD_OPEN hierarchy.

Parameters: void

Returns: void

/**********Expert SDK**********/

int hfdEnableExpertMode ()

Function description: Enable the expert mode.

Parameters: void

Returns: 1 on success, 0 otherwise

int hfdDisableExpertMode ()

Function description: Disable the expert mode.

Parameters: void

Returns: 1 on success, 0 otherwise

int hfdSaveCalibration (int* minValue, int*maxValue, int*homeValue, unsigned char mask, char ID= -1)

Function description: Save calibration value to device memory

Parameters:

@ minValue[in]: min value of the selected joint

@ maxValue[in]: max value of the selected joint

@ homeValue[in]: home value of selected joint

@ mask[in]: bitwise mask of which joints should be calibrated

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdUpdateEncoders (char ID= -1)

Function description: Force an update of the internal encoder values in the state vector. This call retrieves the encoder readings from the device and places them into the state vector. No kinematic model is called.

Parameters:

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetDeltaEncoders (int*enc0, int*enc1, int*enc2, char ID= -1)

Function description: Read all encoders values of the Delta structure.

Parameters:

@ enc0[out]: Delta axis 0 encoder reading

@ enc1[out]: Delta axis 1 encoder reading

@ enc2[out]: Delta axis 2 encoder reading

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetWristEncoders (int*enc0, int*enc1, int*enc2, char ID= -1)

Function description: Read the encoders values of the WRIST structure, starting with the one nearest to the hand.

Parameters:

@ enc0[out]: wrist axis 0 encoder reading

@ enc1[out]: wrist axis 1 encoder reading

@ enc2[out]: wrist axis 2 encoder reading

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetEncoder (int index, char ID= -1)

Function description: Read a single encoder value from the haptic device.

Parameters:

@ index[in]: the encoder index number as defined by HFD_ENCODER_COUNT

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetEnc (int enc[HFD_ENCODER_COUNT], HFDushort mask=0x03FF, char ID= -1)

Function description: Get selective encoder values into encoder array. Particularly useful when using the generic controller directly, without a device model attached.

Parameters:

@ enc[out]: out encoder values array

@ mask[in]: [default = 0x3FF] bitwise mask of which encoders should be read in

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdEnableDevice (bool a_on, char ID= -1)

Function description: Enable or Disable the device

Parameters:

@ a_on[in]: HFD_ON to enable device, HFD_OFF to disable it

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSetMotor (int index, HFDushort val, char ID= -1)

Function description: Program a command to a single motor channel.

Parameters:

@ index[in]: the motor index number as defined by HFD_MAX_DOF

@ val[in]: the motor DAC value

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSetDeltaMotor (HFDushort mot0, HFDushort mot1, HFDushort mot2, char ID= -1)

Function description: Set desired motor commands to the amplifier channels commanding the Delta motors.

Parameters:

@ mot0[in]: Delta axis 0 motor command

@ mot1[in]: Delta axis 1 motor command

@ mot2[in]: Delta axis 2 motor command

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSetWristMotor (HFDushort mot0, HFDushort mot1, HFDushort mot2, char ID= -1)

Function description: Set desired motor commands to the amplifier channels commanding the Rotational motors.

Parameters:

@ mot0[in]: WRIST axis 0 motor command

@ mot1[in]: WRIST axis 1 motor command

@ mot2[in]: WRIST axis 2 motor command

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdDeltaEncoderToPosition (int enc0, int enc1, int enc2, double*px, double*py, double*pz, char ID= -1)

Function description: Computes and returns the position of the end-effector for a given set of encoder readings.

Parameters:

@ enc0[in]: Delta encoder reading on axis 0

@ enc1[in]: Delta encoder reading on axis 1

@ enc2[in]: Delta encoder reading on axis 2

@ px[out]: Delta end - effector position on the X axis[mm]

@ py[out]: Delta end - effector position on the Y axis[mm]

@ pz[out]: Delta end - effector position on the Z axis[mm]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdDeltaPositionToEncoder (double px, double py, double pz, int*enc0, int*enc1, int*enc2, char ID= -1)

Function description: Computes and returns encoder values for a given end-effector position.

Parameters:

@ px[in]: Delta end - effector position on the X axis[mm]

@ py[in]: Delta end - effector position on the Y axis[mm]

@ pz[in]: Delta end - effector position on the Z axis[mm]

@ enc0[out]: Delta encoder reading on axis 0

@ enc1[out]: Delta encoder reading on axis 1

@ enc2[out]: Delta encoder reading on axis 2

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdDeltaEncoderToJointAngles (int enc0, int enc1, int enc2, double*j0, double* j1, double* j2, char ID= -1)

Function description: This routine computes and returns the Delta joint angles for a given set of encoder readings.

Parameters:

@ enc0[in]: Delta encoder reading on axis 0

@ enc1[in]: Delta encoder reading on axis 1

@ enc2[in]: Delta encoder reading on axis 2

@ j0[out]: joint angle for axis 0 in[rad]

@ j1[out]: joint angle for axis 1 in[rad]

@ j2[out]: joint angle for axis 2 in[rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdDeltaJointAnglesToEncoders (double j0, double j1, double j2, int*enc0, int*enc1, int*enc2, char ID= -1)

Function description: This routine computes and returns the Delta encoder readings for a given set of joint angles.

Parameters:

@ j0[in]: joint angle for axis 0 in[rad]

@ j1[in]: joint angle for axis 1 in[rad]

@ j2[in]: joint angle for axis 2 in[rad]

@ enc0[out]: Delta encoder reading on axis 0

@ enc1[out]: Delta encoder reading on axis 1

@ enc2[out]: Delta encoder reading on axis 2

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdDeltaMotorToForce (HFDushort mot0, HFDushort mot1, HFDushort mot2, double*fx, double*fy, double*fz, char ID= -1)

Function description: Computes and returns the force applied to the end-effector for a given set of motor commands at a given position (defined by encoder readings).

Parameters:

@ mot0[in]: motor command on Delta axis 0

@ mot1[in]: motor command on Delta axis 1

@ mot2[in]: motor command on Delta axis 2

@ fx[out]: force on the Delta end - effector on the X axis[N]

@ fy[out]: force on the Delta end - effector on the Y axis[N]

@ fz[out]: force on the Delta end - effector on the Z axis[N]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdDeltaForceToMotor (double fx, double fy, double fz, HFDushort*mot0, HFDushort*mot1, HFDushort*mot2, char ID= -1)

Function description: Computes and returns the motor commands necessary to obtain a given force on the end-effector at a given position (defined by encoder readings).

Parameters:

@ fx[in]: force on the Delta end - effector on the X axis[N]

@ fy[in]: force on the Delta end - effector on the Y axis[N]

@ fz[in]: force on the Delta end - effector on the Z axis[N]

@ mot0[out]: motor command on Delta axis 0

@ mot1[out]: motor command on Delta axis 1

@ mot2[out]: motor command on Delta axis 2

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdWristEncoderToOrientation (int enc0, int enc1, int enc2, double*oa, double*ob, double*og, char ID= -1)

Function description: Compute individual angle of each joint, starting with the one located nearest to the hand.

Parameters:

@ enc0[in]: WRIST encoder reading on axis 0

@ enc1[in]: WRIST encoder reading on axis 1

@ enc2[in]: WRIST encoder reading on axis 2

@ oa[out]: WRIST end-effector orientation around the Y axis [rad]

@ ob[out]: WRIST end-effector orientation around the X axis [rad]

@ og[out]: WRIST end-effector orientation around the Z axis [rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdWristOrientationToEncoder (double oa, double ob, double og, int*enc0, int*enc1, int*enc2, char ID= -1)

Function description: Compute the encoder values from individual angle of each joint, starting with the one located nearest to the hand.

Parameters:

@ oa[in]: WRIST end-effector orientation around the Y axis [rad]

@ ob[in]: WRIST end-effector orientation around the X axis [rad]

@ og[in]: WRIST end-effector orientation around the Z axis [rad]

@ enc0[out]: WRIST encoder reading on axis 0

@ enc1[out]: WRIST encoder reading on axis 1

@ enc2[out]: WRIST encoder reading on axis 2

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdWristEncodersToJointAngles (int enc0, int enc1, int enc2, double*j0, double*j1, double*j2, char ID= -1)

Function description: Computes and returns the wrist joint angles for a given set of encoder readings, starting with the one located nearest to the hand.

Parameters:

@ enc0[out]: WRIST encoder reading on axis 0

@ enc1[out]: WRIST encoder reading on axis 1

@ enc2[out]: WRIST encoder reading on axis 2

@ j0[out]: joint angle for axis 0 in[rad]

@ j1[out]: joint angle for axis 1 in[rad]

@ j2[out]: joint angle for axis 2 in[rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdWristJointAnglesToEncoders (double j0, double j1, double j2, int*enc0, int*enc1, int*enc2, char ID= -1)

Function description: Computes and returns the wrist encoder readings for a given set of joint angles.

Parameters:

@ j0[out]: joint angle for axis 0 in [rad]

@ j1[out]: joint angle for axis 1 in [rad]

@ j2[out]: joint angle for axis 2 in [rad]

@ enc0[out]: WRIST encoder reading on axis 0

@ enc1[out]: WRIST encoder reading on axis 1

@ enc2[out]: WRIST encoder reading on axis 2

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetDeltaJointAngles (double*j0, double*j1, double*j2, char ID= -1)

Function description: Retrieve the joint angles in [rad] for the Delta structure.

Parameters:

@ j0[out]: joint angle for axis 0 in [rad]

@ j1[out]: joint angle for axis 1 in [rad]

@ j2[out]: joint angle for axis 2 in [rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetWristJointAngles (double*j0, double*j1, double*j2, char ID= -1)

Function description: Retrieve the joint angles in [rad] for the WRIST structure.

Parameters:

@ j0[out]: joint angle for axis 0 in [rad]

@ j1[out]: joint angle for axis 1 in [rad]

@ j2[out]: joint angle for axis 2 in [rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetJointTorques (double*t0, double*t1, double*t2, char ID= -1)

Function description: Get joint torques in Delta joint.

Parameters:

@ t0[out]: Delta joint axis 0 torque command[mNm]

@ t1[out]: Delta joint axis 1 torque command[mNm]

@ t2[out]: Delta joint axis 2 torque command[mNm]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetJointAngles(double j[HFD_MAX_DOF], char ID= -1)

Function description: Retrieve the joint angles in [rad] for all sensed degrees-of-freedom of the current device.

Parameters:

@ j[out]: out array of joint angles in [rad]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetJointVelocities(double v[HFD_MAX_DOF], char ID= -1)

Function description: Retrieve the joint angle velocities in [rad/s] for all sensed degrees-of-freedom of the current device.

Parameters:

@ v[out]: out array of joint angle velocities in[rad/s]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetEncVelocities(double v[HFD_MAX_DOF], char ID= -1)

Function description: Retrieve the encoder angle velocities in [increments/s] for all sensed degrees-of-freedom of the current device.

Parameters:

@ v[out]: out array of encoder angle velocities in[rad/s]

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSetParaInt (unsigned short param, int value, char ID= -1)

Function description: Set the specified parameter with the given value, the parameter is int type.

Parameters:

@ param[in]: parameter name, expressed in macro

@ value[in]: the given value to be set

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSetParaShort (unsigned short param, short value, char ID= -1)

Function description: Set the specified parameter with the given value, the parameter is short type.

Parameters:

@ param[in]: parameter name, expressed in macro

@ value[in]: the given value to be set

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetParaInt (unsigned short param, int*value, char ID= -1)

Function description: Retrieve the specified parameter value, the parameter is int type.

Parameters:

@ param[in]: parameter name, expressed in macro

@ value[out]: value of the retrieved parameter

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdGetParaShort (unsigned short param, short*value, char ID= -1)

Function description: Retrieve the specified parameter value, the parameter is short type.

Parameters:

@ param[in]: parameter name, expressed in macro

@ value[out]: value of the retrieved parameter

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

int hfdSaveConfiguration (char ID= -1)

Function description: Save all configuration to EEPROM so that the configuration won't be lost even if power off.

Parameters:

@ ID[in]: device ID, default= -1

Returns: 1 on success, 0 otherwise

/*********************************OS independent utilities*********************************/

int hfdGetTime ()

Function description: Returns the current value from the high-resolution system counter in [s]. The resolution of the system counter may be machine-dependent, as it is usually derived from one of the CPU clocks signals. The time returned is guaranteed to be monotonic.

Parameters: void

Returns: The current time in [s]

int hfdStartThread (void*func(void*), void*arg, int priority)

Function description: Create a thread on any operating systems supported by the SDK.

Parameters:

@ func[in]: function to run in the thread

@ arg[in]: optional pointer to an argument passed to the thread function

@ priority[in]: priority given to the thread. The SDK will try to set the priority level, but will continue without error if the OS does not accept the request.

Returns: 1 on success, 0 otherwise