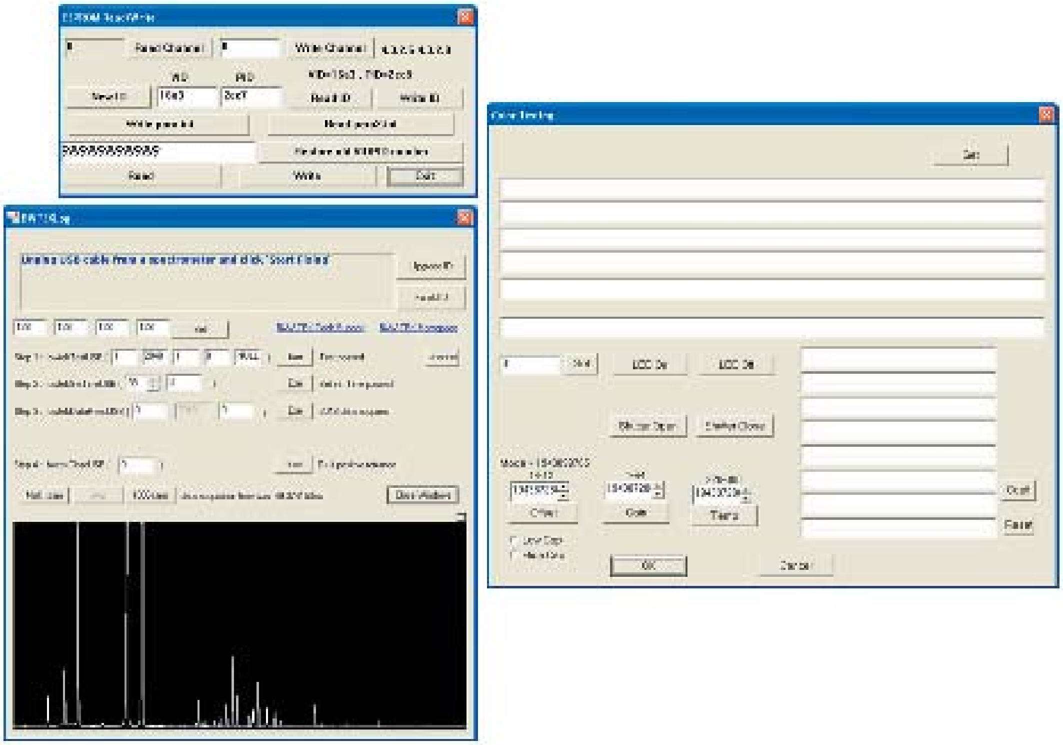
BW SDK



USER MANUAL

Software Developer's Kit for BRC, BTB, BTC, BTF, & BTR Series Array Spectrometers

Table Of Contents

Introduction

BTC and BRC series products are high performance and low cost array spectrometers. The BW Tek software developer's kit (SDK) is designed for customers who have a need to custom program the BTC and BRC spectrometers for specific applications. BTC and BRC products are available with standard RS232, or virtual USB 2.0 RS232 port, as well as true USB 2.0 interfaces. A table is provided on the following page summarizing the available models and their interfaces.

The SDK program's .DLL file is compatible with any software language supported through Microsoft Windows Operating Systems.

Getting To Know BW-Spec User Interface

BTC and BRC series products are high performance and low cost array spectrometers. The BW-Spec software is shipped together with a parameter data disk which contains your spectrometer unit specific setting information. Insert the parameter data disk into your PC's floppy disk drive and use the Import Parameter File function in the File menu from BW-Spec to update the settings.

Most BRC and BTC models have a default baudrate setting of 9600. Custom settings will be reset to the default each time on a new power up. The BW-Spec operating software will load a default baudrate of 9600 every time when the program is launched. The custom COM port setting will be saved when exiting BW-Spec.

Models

Model	Description	Digitizer Resolution (bits)	PC Interface
BRC111A	Non-Cooled 2048 CCD array spectrometer	16	USB
BRC112	Non-Cooled 2048 CCD array spectrometer	16	USB
BRC131E	Non-Cooled CCD 3680 array spectrometer	16	USB
BRC311	Non-Cooled 2048 CCD array spectrometer	16	USB
BRC711E-512	Non-Cooled 512 PDA array spectrometer	16	USB
BRC711E-1024	Non-Cooled 1024 PDA array spectrometer	16	USB
BTC111	TE Cooled 2048 CCD array spectrometer	16	USB
BTC112	TE Cooled 2048 CCD array spectrometer	16	USB
BTC211	TE Cooled 512 InGaAs array spectrometer	16	USB
BTC221E-256	TE Cooled 256 InGaAs array spectrometer	16	USB
BTC251	TE Cooled 512 InGaAs array spectrometer	16	USB
BTC251E-512	TE Cooled 512 InGaAs array spectrometer	16	USB
BTC261E-256	TE Cooled 256 InGaAs array spectrometer	16	USB
BTC261E-512	TE Cooled 512 InGaAs array spectrometer	16	USB
BTC261E-1024	TE Cooled 1024 InGaAs array spectrometer	16	USB
BTC262E-256	TE Cooled 256 InGaAs array spectrometer	16	USB
BTC262E-512	TE Cooled 512 InGaAs array spectrometer	16	USB
BTC262E-1024	TE Cooled 1024 InGaAs array spectrometer	16	USB
BTC311	TE Cooled 2048 CCD array spectrometer	16	USB
BTC611E-512	TE Cooled 512 Back Thinned CCD	16	USB
BTC611E-1024	TE Cooled 1024 Back Thinned CCD	16	USB
BTC711E-512	TE Cooled 512 PDA array spectrometer	16	USB
BTC711E-1024	TE Cooled 1024 PDA array spectrometer	16	USB
BTC811(In Development)	TE Cooled High Dynamic Range 2048 CCD array spectrometer	16	USB

Model	Description	Digitizer Resolution (bits)	PC Interface
BTF111	TE Cooled 2048 CCD array spectrometer for Fluorescence	16	USB
BTF113	TE Cooled 2048 CCD array spectrometer for Fluorescence	16	USB
BTU111	TE Cooled 2048 CCD array spectrometer for UV Applications	16	USB
BTU112	TE Cooled 2048 CCD array spectrometer for UV	16	USB
BRU111	TE Cooled 2048 CCD array spectrometer for UV Applications	16	USB
BRU112	TE Cooled 2048 CCD array spectrometer for UV Applications	16	USB
BTR111 (MiniRam)	TE Cooled 2048 CCD array spectrometer	16	USB
BTC500	TE Cooled 256 PbS array spectrometer	14	USB
BRC100	Non-cooled 2048 CCD array spectrometer	12	RS232
BRC110	Non-cooled 2048 CCD array spectrometer	12	RS232
BTC110	TE Cooled 2048 CCD array spectrometer	16	RS232
BTC200	TE Cooled 512 InGaAs array spectrometer	16	RS232
BTC251E_512_RS232	TE Cooled 512 InGaAs array spectrometer	16	RS232

Function Descriptions

USB 2.0/1.1 or RS232 Interface Spectrometers

The first six functions are the most basic functions for BRC, BTB, BTC, BTF & BTR spectrometers with USB or RS232 interfaces for writing custom software. The rest are for more advaced features. These functions are described as below.

bwtekTestUSB

```
int bwtekTestCard
(
   int nUSBTiming,        // USB Interface timing option
   int nPixelNo,          // number of pixels of a detector to be readout
  int nInputMode,         // signal conditioning stage gain value
  int nChannel,           // channel to get data from
  int pParam              // extra setting parameters only for RS232
);
```

This function is for testing the USB device status.

nUSBTiming is used to specify device USB interface timing options. It should be set to 1 for USB 2.0/1.1 based linear array spectrometers, and 2 for BTC600E 2-D array based USB products.

nPixelNo should be set to the value representing the number of pixels of the detector array used in the spectrometer device. It should be set to 2048 for Sony ILX511 CCD based spectrometers such as BTC and BRC 1xx products.
Note: See Appendix A for Pixel Number values for the spectrometer model you are using.

nInputMode is used to specify the ADC input range used in the spectrometer device being programmed for. It should be set to 0 for ADCs with unipolar input such as 0 to +5V or 0 to +10V. . It should be set to 1 for ADCs using bipolar input such as -5 to +5V or -10 to +10V (BTC and BRC111), and 2 for ADCs with unipolar and non-inverting input (BTC600).

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

pParam is a pointer for a structure as seen below for RS232 Interfaces Only.
NOTE: This pointer should be **NULL FOR ALL USB INTERFACES**

```
struct RS232Params
(
   int nPort,                // 1 for Com1, 2 for com2 ,…
  int nBautRate,             // 9600, 38400 or 115200
   int nAverage,             // number of data to be averaged
  int nTimeDelay,            // time delay for a scan
  int nReserve,              // reserved for future use
  double fCoefficient[4]     // coefficient for wavelength calibration
);
```

RETURN
If the spectrometer device is tested successfully the function will return a positive return value. Otherwise it will return negative value.
NOTE: Refer to Appendix A for default options for parameters and timing modes for each spectrometer model.

****

bwtekSetTimeUSB

```
long bwtekSetTimeUSB
(
   long  lTime,     // integration time setting
   int nChannel   // channel to get data from
);
```

This function is for setting the spectrometer integration time as specified.

lTime is the integration time value in milliseconds to be set. The value should be between 1 and 65535 for BRC111 and BTC111 spectrometers.

This function will NOT work for the BTF113 model, see *bwtekSetPulseNo*

NOTE:

For spectrometers that have 2D detectors installed, (BTC611 series), an offset integration time, *OffsetTime*, results from the extra time the sensing pixels are exposed to during the readout process of the area sensor arrays. The actual integration time the sensors are exposed is the set integration time plus this offset. The *OffsetTime* needs to be taken into account when calculating the actual integration time. Therefore the lTime value to be passed to the DLL is calculated by subtracting the *OffsetTime* from the desired integration time.

For Example: If the *OffsetTime* is 26 (ms) and desired integration time is 50 (ms).
You would NOT set lTime to 50
You MUST set *lTime* = (desired integration) – *OffsetTime*
lTime = 50 – 26 -> *lTime* = 24

This will cause the minimal integration time for the spectrometer to be affected. The minimal integration time will be *OffsetTime* + 1.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful it returns the new integration time value. Otherwise it will return negative value.

bwtekDataReadUSB

```
int bwtekDataReadUSB
(
   int  nTriggerMode,               // Base address for plug-in data acquisition board
   unsigned short *pArray,          // data value array from the read operation stored
   int nChannel                     // channel to get data from
);
```

This function is for reading out data form detector

nTriggerMode is used to set trigger mode to initiate a trigger scan process. It should be set to 0 for free running mode and 1 for external trigger mode. The external trigger signal should be supplied as a 5V TTL tpulse. It is falling edge effective . Refer to hardware user manual for the trigger pulse width definition.

pArray is a pointer to data array memory spaces dependent on the total number of pixels on the CCD for the spectrometer model that is being used. Every element in this array should be an unsigned integer with minimum 2 bytes for a 16 bit resolution.
NOTE: This Value should be equal to int *nPixelNo* in the **bwtekTestUSB** function.
See Appendix A for Pixel Number Values

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the spectrometer readout is successful as the result of this function call it returns the number of data points read. Otherwise the return value is negative.

bwtekReadResultUSB

```
int bwtekReadResultUSB
(
   int nTriggerMode,               // setting for trigger mode
   int nAverage,                   // number of scans to be averaged
   int nTypeSmoothing,             // setting for smoothing type
   int nValueSmoothing,            // setting for smoothing parameter
   unsigned short *pArray,         // an array memory space for scan data
   int nChannel=0                  // setting for channel number
);
```

This function is for reading out data form detector with data smoothing functions

nTriggerMode is used to set trigger mode to initiate a trigger scan process. It should be set to 0 for free running mode and 1 for external trigger mode. The external trigger signal should be supplied as a 5V TTL tpulse. It is falling edge effective . Refer to hardware user manual for the trigger pulse width definition.

nAverage is used to set the number of scans to be averaged.

nTypeSmoothing 0 for no smoothing function, 1 for FFT smoothing, 2 for Savitzky-Golay smoothing

nValueSmoothing When using FFT smoothing, this parameter indicates the percentage of cutoff frequency. The range should be 0 to 100.

pArray is a pointer for 2048 data array memory space . Every element in this array should be an unsigned integer with minimum 2 bytes for a 16 bits resolution.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

When using the Savitzky-Golay smoothing, this parameter indicates the number of data points. The range should be 2 to 5.
Note: If *nValueSmoothing* is set to 4, the total number of smoothing points will be 9. 1 point will be the "origin" pixel and the other 8 will be 4 pixels before the origin & 4 pixels after the origin added together with weighted factors and divided by 9.

RETURN
If the spectrometer readout is successful as the result of this function call it returns the number of data points read. Otherwise the return value is negative.

bwtekCloseUSB

```
int bwtekCloseUSB
(
  int nchannel   // channel to get data from
);
```

This function is to reset spectrometer device to a default setting. This should be used when close the programmed spectrometer application which was intiated by the bwtekTestUSB function call.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN

If closing USB connection is successful as the result of this function call it returns positive Otherwise the return value is negative.

bwtekReadEEPROMUSB

```
int bwtekReadEEPROMUSB

(
   char *OutFileName,    // The filename in which data from EEPROM would be saved
   int nChannel=0        // channel number to get data from
);
```

This function is used for retrieving data from EEPROM

OutFileName is used to specify the name of out file which will be saved in a hard drive. The out file has a text ASCII format and the default folder is a current directory unless otherwise specified. It should string type like bwtekReadEEPROMUSB(“para.ini”,0) or bwtekReadEEPROMUSB(“eeprom.txt”,1).

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the spectrometer device becomes disconnected or loses communication a -1 value will be returned.
If there is no data to be readout from EEPROM, the function will return -2.
Otherwise it will return positive value.

NOTE: The output file contains a lot of information used for internal use.
The customer may have use for some of this data, for example, the calibration section and the information on timing mode or input mode.

bwtekSetTimingsUSB

```
int bwtekSetTimingsUSB

(
  long  lTriggerExit,      // Setting external trigger timeout
  int nMultiple,           // A multifly factor for long integration time
  int nChannel=0           //channel number to get data from
);
```

This function is used for setting both an external trigger timeout and a multiple factor for long integration time.

lTriggerExit is used to specify the time period waiting for external trigger. If no trigger signal comes in within this period, a spectrometer will be released from external mode.
The real timeout period is 15ms* lTriggerExit, and the maximum is 15ms*65535=983040ms.

nMultiple is used to specify the time base for integration time. The default time base is 1ms when multiple = 1, when multiple = 2, the time base is 2ms and real integration time will be 2**lTime* (*lTime* parameter is from the **bwtekSetTimeUSB** function. The *lTime* time can be adjusted by using **bwtekSetTimeUSB** func tion. The real integration time period is *nMultiple* * lIntegration, and the maximum value for nMultiple is 16.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the spectrometer device becomes disconnected or loses communication a negative integer will be returned, indicating a failure. Otherwise the *lTriggerExit* value will be retunred.

bwtekPolyFit

```
int bwtekPolyFit
(
   double *x,              // Array of independent variables
   double *y,              // Array of dependent variables
   int const numPts,       // Number of points in independent and dependent arrays
   double *coefs,          // Pointer to array containing calculated coefficients [index from 0 to order]
   int const order         // Desired order of polynomial fit
)
```

This function is to perform a curve fit to a specified polynomial function based on supplied data values and least square algorithm. This function may be used for conversions from pixel numbers to wavelengths.

If used for wavelength conversion a wavelength calibration step needs to be performed, during which a calibration light source with known spectral features or a series of narrow band light radiation will be necessary. The known wavelength features of the calibration source along with the pixel numbers they are falling onto are the Y and X data values that need to be passed to the function for curve fit. The arrays indices should range from 0 to [Number_of_Points - 1]. They are passed to the function along with a desired polynomial fit order and an array enough to hold the coefficients (). This array may be used with ***bwtekPolyCalc*** to calculate wavelength from pixels.

x is a pointer to an array containing the independent variables. It should range from 0 to (numPts-1).

y is a pointer to an array containing the dependent variables. It should range from 0 to (numPts-1).

numPts is the total number of data pairs in the variable arrays

coefs is a pointer an array that contains the polynomial curve fit coefficients whose number of memory spaces is (*order* +1).

order is the desired polynomial curve fit order. The third order fit is recommended for use in wavelength conversions

RETURN
If the function call is successful it will return a positive interger. Otherwise it will return a negative integer.

bwtekPolyCalc

```
void bwtekPolyCalc
(
  double *coefs,
  int const order,
  int const x,
  double *y
);
```

This function performs calculations by using the following formula:
y = a0 + a1*x^1 + a2*x^2 + ... + aN*x^N, where * specified multiplication and ^ specifies “to the power of.”

coefs is a pointer to an array containing the polynomial coefficients. These can be calculated using bwtekPolyFit.

order is the polynomial order to be used and must be less than or equal to the number of available (supplied) coefs.

x is the input variable, in this case, the pixel number.

y is the value to be calculated.

RETURN
None

bwtekGetExtStatus

```
int bwtekGetExtStatus
(
  int nchannel    //channel to get data from
)
```

This function is used to retrieve the status of the input pin.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the status of input is low, a 0 will be returned. If the status of input is high, a positive integer will be returned. If the function call is unsuccessful, a negative integer will be returned.

bwtekSetExtLaser

```
int bwtekSetExtLaser
(
   int nOnOff,      //switch for On/Off
   int nchannel    //channel to get data from
)
```

This function is used for controlling the On/Off capabilites of an external Laser.

nOnOff is used to set the Laser control signal. If 0 is assigned, the output pin signal will be low.
If 1 is assigned, the output pin signal will be high.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekSetExtShutter

```
int bwtekSetExtShutter
(
   int nOnOff,      // switch for On/Off
   int nchannel    // channel to get data from
)
```

This function is used for controlling the Open/Closed capabilities of a Shutter (If Installed).

nOnOff is used to set the Shutter control signal. If 0 is assigned the output signal will be low.
If 1 is assigned the output signal will be high.

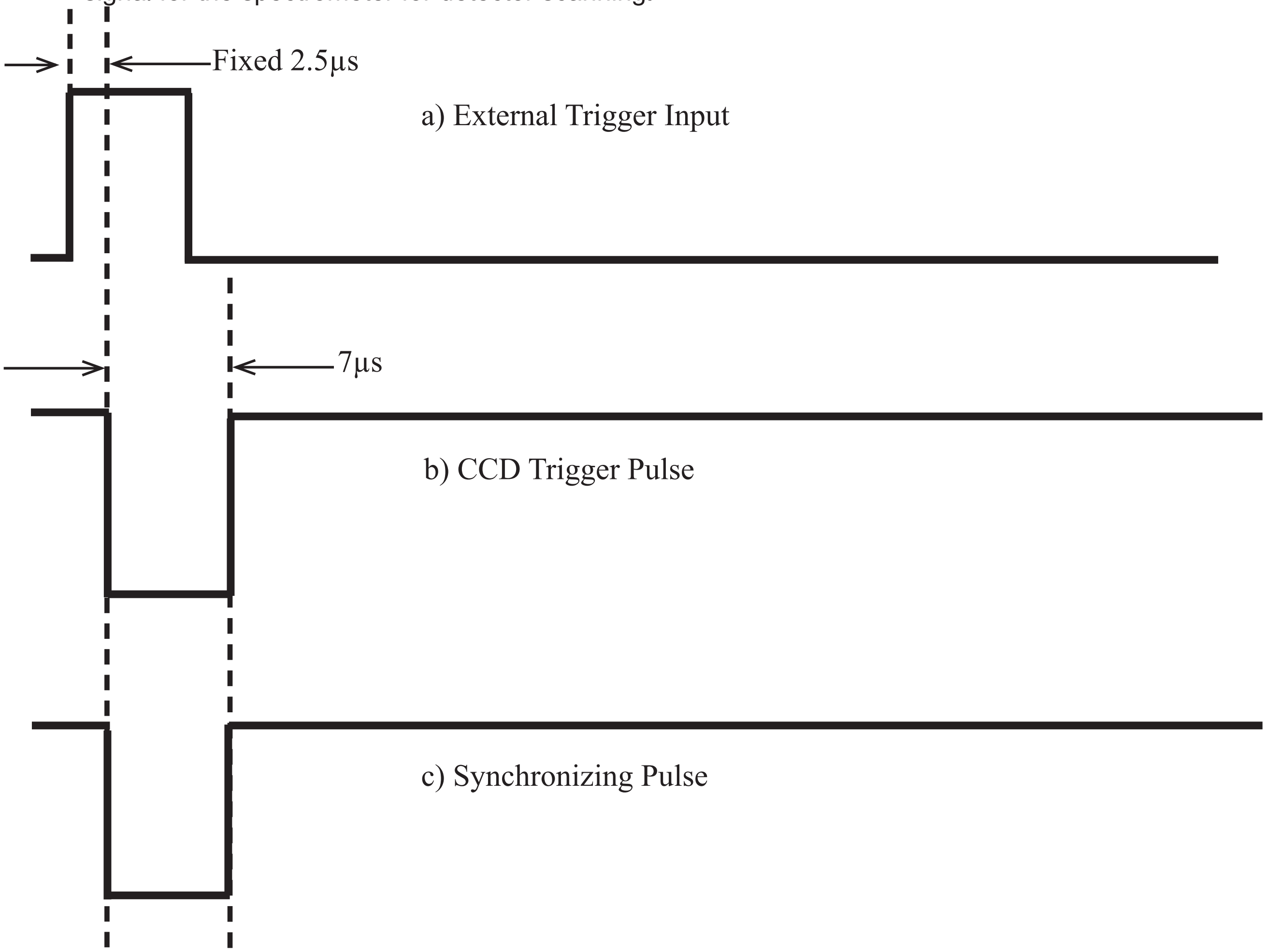
nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekSetExtSync

```
int bwtekSetExtSync
(
  int nOnOff,     //switch for On/Off
  int nchannel   //channel to get data from
)
```

This function is used to generate a pulse, which synchronizes an outside source with the external trigger signal for the spectrometer for detector scanning.



nOnOff is used to control the synchronizing pulse. If 0 assigned, the output signal will not be generated (no signal). If 1 is assigned, the output signal will be generated(pulse signal).

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

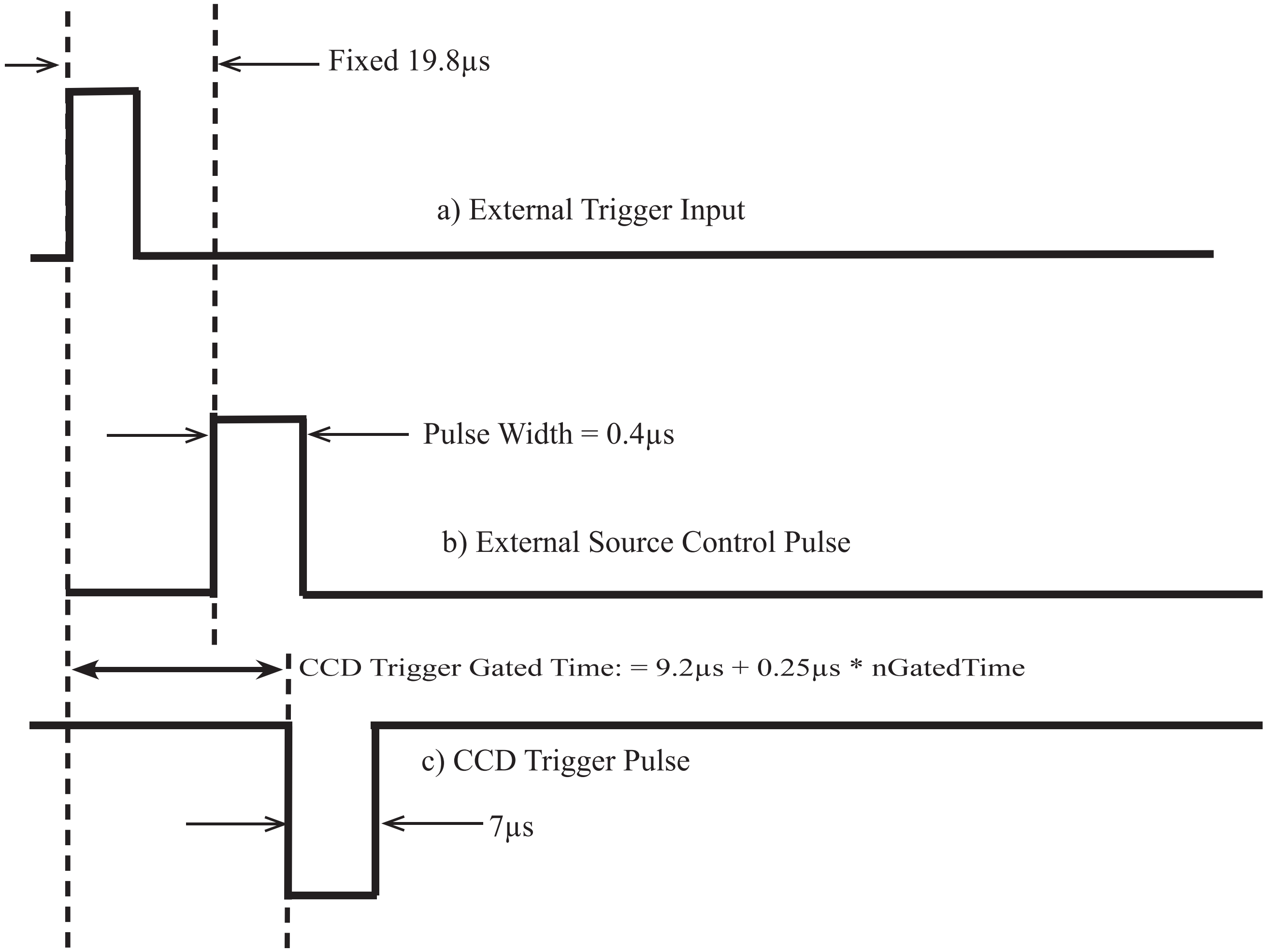
bwtekGatedMode

```
int bwtekGatedMode
(
   int nGatedTime,         // Gated Time between an External trigger and a CCD trigger
   int nchannel            // channel to get data from
)
```

This function is for controlling CCD gated delay time

nGatedTime is used to determine the time delay before the CCD starts to acquire data after an external pulse has been delivered. The parameter can be assigned an integer value of 0 to 65535. This value is in addition to the inherent delay time.
Example: When nGatedTime is 44 the gated time will be:
(9.2μs + 0.25μs*44) = 20.2μs = (9.2μs + 0.25μs*nGatedTime)
****NOTE**** When 0 is assigned to nGatedTime, the function will be disabled.

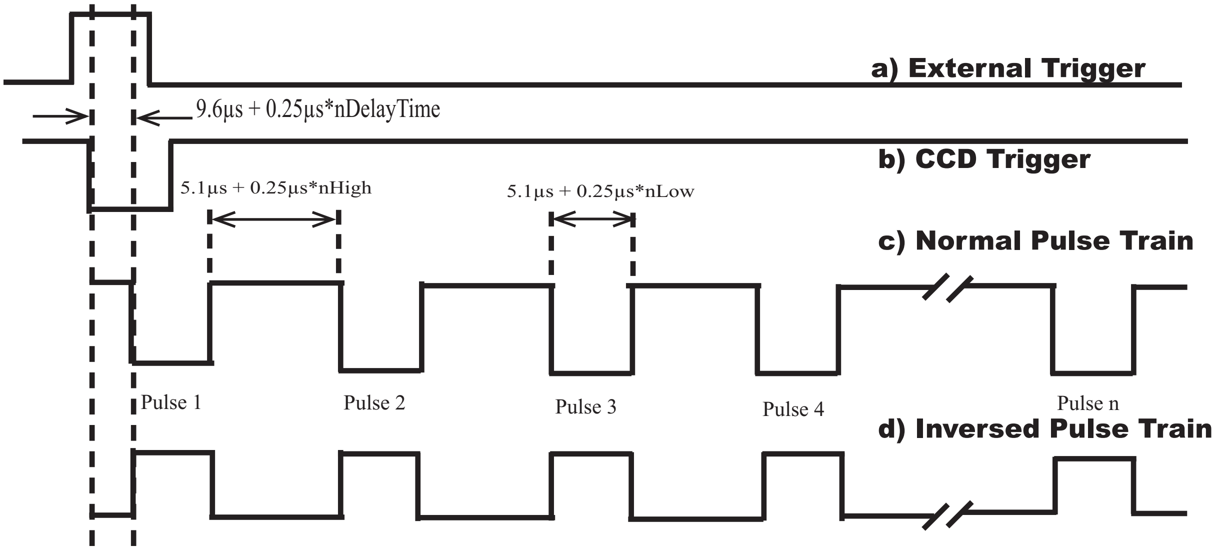
nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.for device 2,...etc.


Fixed 19.8μs
a) External Trigger Input
Pulse Width = 0.4μs
b) External Source Control Pulse
CCD Trigger Gated Time: = 9.2μs + 0.25μs * nGatedTime
c) CCD Trigger Pulse
7μs

bwtekSetExtPulse

```
int bwtekSetExtPulse
(
   int nOnOff,              // switch to turn on or off pulse train
   int nDelayTime,          // delay time between the CCD trigger and the first pulse
   int nHigh,               // time duration for high pulse width
   int nLow,                // time duration for low pulse width
   int nPulse,              // number of pulses to be generated
   int nInverse,            // pulse output will be inverted
   int nChannel             // channel to get data from
)
```

This function is used to generate a pulse train defined by nHigh, nLow and nPulse.



nOnOff will turn on or off the output pulse train. If 0 is assigned the output signal train will not be generated (no signal). If 1 is assigned, the output signal train will be synchronized with the CCD trigger pulse that is generated after the normal delay time of (9.6µs + 0.25µs*nDelayTime).

nDelayTime is used to determine the delay time between the CCD trigger and the first pulse. This parameter can be assigned an integer value of 1 to 65535. This value is in addition to the inherent delay time. Example: When nDelayTime is 50, the total delay time will be
(9.6µs+0.25µs*50) = 22.1µs = (9.6µs+0.25µs*nDelayTime)

nHigh is used to determine the high pulse width. The parameter can be assigned an integer value of 1 to 65535. This value is in addition to the inherent delay time. Example: When nHigh is 63, the high pulse width will be 5.1µs+0.25µs*63 = 20.85µs = 5.1µs+0.25µs*nHigh

nLow is used to determine the low pulse width. The parameter can be assigned an integer value of 1 to 65535. This value is in addition to the inherent delay time. Example: When nLow is 56, the low pulse width will be 5.1µs+0.25µs*56 = 19.1µs = 5.1µs+0.25µs*nLow

nPulse is used to set the number of pulses that are to be generated. The parameter can be assigned an integer value of 1 to 65535.

****NOTE**** In the function bwtekSetTime the value assigned to the ITime parameter must be longer than (10.3μs + 0.23μs*(nHigh+nLow)*nPulse + 0.23μs*nDelayTime)

nInverse is used to invert the output pulse train. 0 will output a the Normal Pluse Train, 1 will output an Inversed Pulse Train

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

LED Functions:

The following functions will work only with the BTF111 Model (MiniFluo) for turning On and Off the Excitation LED inside of this unit.

bwtekLEDOn

```
int bwtekLEDOn
(
  int nChannel   //channel number to get data from
);
```

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekLEDOff

```
int bwtekLEDOff
(
  int nChannel   //channel number to get data from
);
```

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekLEDDelay

```
int bwtekLEDDelay
(
  unsigned int nDelay,    // time delay for the LED to be turned on units = ms
  int nChannel            // channel number to get data from
);
```

nDelay is used for setting the LED delay time in ms. Once a trigger signal is received nDelay ms must pass before the aqcuistion of the data begins. If nDelay is set to 0, there will be no delay time. The LED light must then be controlled manually by using both the bwtekLEDOn and bwtekLEDOff functions.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time *nChannel* setting 0 for device 1, *nChannel* setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

Shutter Functions:

The following functions work only with units which have shutters installed.

bwtekShutterOpen

```
int bwtekShutterOpen();
```

RETURN

If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekShutterClose

```
int bwtekShutterClose();
```

RETURN

If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

BTC251E Functions:

The following functions work only with the BTC251E spectrometer.

bwtekSetGain

```
int bwtekSetGain
(
  int nGain,                  // set the gain value
  int nChannel                // channel to get data from
);
```

This function is for setting the gain value of the analog output.

nGain is used to set the gain value of an analog output signal. It can be changed from 1 to 64.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekSetOffset

```
int bwtekSetOffset
(
  int nOffset,                // set the offset value
  int nChannel                // channel to get data from
)
```

This function is for setting the offset value of the analog output.

nOffset is used to set the offset value of an analog output signal. It can be set from 1 to 512.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekGetGain

```
int bwtekGetGain
(
  int *nGain,       // return value of gain
  int nChannel    // channel to get data from
)
```

This function is for getting the gain value of the analog output.

***nGain** is a pointer to an array that will return the gain value of an analog output signal.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekGetOffset

```
int bwtekGetOffset
(
  int *nOffset,     // return value of offset
  int nChannel    // channel to get data from
)
```

This function is for getting the offset value of analog output.

***nOffset** is a pionter to an array that will return the offset value of an analog output signal.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

BTC261E & BTC262E Functions:

The following functions work only with the BTC261E & BTC262E spectrometers.

bwtekSetABGain

```
int bwtekSetABGain
(
  int nAB,         // select between the odd(A) and even(B) pixel channels
  int nGain,       // set the gain value
  int nChannel     // channel to get data from
);
```

This function is for setting the gain value of the analog output on either A or B channel.

nAB is used to choose between the **odd(A)** and **even(B)** pixel channels. Use 0 to control the odd(A) pixel channel and use 1 to control the even(B) pixel channel.
****NOTE For the BTC262E nAB must always be 0. The BTC262E uses one channel for both odd and even pixels.**

nGain is used to set the gain value of an analog output signal. It can be set from 1 to 1023.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekSetABOffset

```
int bwtekSetABOffset
(
  int nAB,         // select between the odd(A) and the even(B) pixel channels
  int nOffset,     // set the offset value
  int nChannel     // channel to get data from
)
```

This function is for setting the offset value of the analog output on either A or B channel

nAB is used to choose between the **odd(A)** and **even(B)** pixel channels. Use 0 to control the odd (A) pixel channel and use 1 to control the even (B) pixel channel.
****NOTE For the BTC262E nAB must always be 0. The BTC262E uses one channel for both odd and even pixels.**

nOffset is used to set the offset value of an analog output signal. It can be set from 1 to 1023.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekGetABGain

```
int bwtekGetABGain
(
  int nAB,          // select between the odd(A) and even(B) pixel channels
  int *nGain,       // return value of gain
  int nChannel      // channel to get data from
)
```

This function is for getting the gain value of the analog output on either A or B channel

nAB is used to choose between the **odd(A)** and **even(B)** pixel channels. Use 0 to control the odd(A) pixel channel and use 1 to control the even(B) pixel channel.
****NOTE For the BTC262E nAB must always be 0. The BTC262E uses one channel for both odd and even pixels.**

***nGain** is a pointer to an array that will return the gain value of an analog output signal.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekGetABOffset

```
int bwtekGetABOffset
(
  int nAB,          // select between the odd(A) and even(B) pixel channels
  int *nOffset,     // return value of offset
  int nChannel      // channel to get data from
)
```

This function is for getting the offset value of the analog output on either A or B channel

nAB is used to choose between the **odd(A)** and **even(B)** pixel channels. Use 0 to control the odd(A) pixel channel and use 1 to control the even(B) pixel channel.
****NOTE For the BTC262E nAB must always be 0. The BTC262E uses one channel for both odd and even pixels.**

***nOffset** is a pionter to an array that will return the offset value of an analog output signal.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

BTC251E, BTC261E, & BTC262E Functions:

The following functions work with the BTC251E, BTC261E, & BTC262E spectrometers.

bwtekSetInGaAsMode

```
int bwtekSetInGaAsMode
(
  int nMode,       // select feedback capacitor mode
  int nChannel    // channel to get data from
);
```

This function is for choosing the detector mode: high dynamic range or high sensitivity range.

nMode is used to choose the feedback capacitor mode of the detector. Use 0 to set the mode to High Sen sitivity and 1 to set the mode to High Dynamic.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

bwtekGetInGaAsMode

```
int bwtekGetInGaAsMode
(
  int *nMode,      // select detector mode
  int nChannel    // channel to get data from
)
```

This function is for retrieving the detector mode setting: high dynamic range or high sensitivity range.

***nMode** is a pointer to an array that will return the value of the feedback capacitor mode setting. If 0 is returned the detector is in High Sensitity Mode and if 1 is returned the detector is in High Dynamic Mode.

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

BRC131E Functions:

The following functions work with the BRC131E spectrometer.

bwtekSetTimeBaseUSB

```
int bwtekSetTimeBaseUSB
(
  long lTimeBase,          // time base of the integration time
  int nChannel             // channel to get data from
);
```

This function is for setting a time base for the integration time

lTimeBase is used to set the time base.
lTimeBase = 1 --> time base is 10us
lTimeBase = 10 --> timebase is 100us
lTimeBase = 100 --> timebase is 1000us (1 ms)
lTimeBase = 1000 --> timebase is 1000us (10 ms)

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If the function call is successful, a positive integer will be returned, else a negative interger will be returned.

BTF113 Functions:

The below functions work only with the BTF113 system.

bwtekSetPulseNo

```
int bwtekSetPulseNo
(
  int nPulseNo,          // number of pulses
  int nChannel           // channel to get data from
);
```

There is no need to set the integration time when calling this function for the BTF113. When *bwtekSetPulseNo* is called the integration time will automatically be set according to the trigger pulse duration.

nPulseNo is used to set the number of pulses which will trigger the Xe lamp. The integration time will be set to *nPulseNo* multipled by the Xe Pulse Duration.
The default trigger pulse duration is about 10 ms
Integration Time = 10 ms * *nPulseNo*
****Note** To turn off the Xe lamp set the *nPulseNo* = 0**

nChannel is used to address a specific spectrometer device to be operated when multiple spectrometer devices are involved. It should be set to 0 when there is only one device or for the first device when running multiple devices. A total of 7 spectrometer devices may be connected at one time nChannel setting 0 for device 1, nChannel setting 1 for device 2,...etc.

RETURN
If nPulseNo is positive, the return value will be the estimated integration time in milliseconds. The real integration time will be automatically set when this function is called. If the return value is 0, nPulseNo = 0, the Xe lamp will be turned off and the previous integration time will be kept.

Visual C++ and Visual Basic demo source code

In order to assist custom software development using the provided SDK Visual Basic and Visual C++ demo program and source codes are provided. The programs and source codes are ***vcdlldemo.exe*** and ***vcdlldemo.exe*** which can be found on the distribution CD from BW Tek.

In summary, function calls should be made following the following sequence for a BW Tek USB BTC111 or BRC111 spectrometer device:

1. bwtekTestUSB(0,2048,1.0,NULL)
2. bwtekSetTimeUSB(10,0)
3. bwtekDataReadUSB(0,pArray,0)
4. bwtekCloseUSB(0)

***The SDK Package will come with serveral source code examples: VC++, C++ Builder, C#, VB, VBA, VB.NET, and Labview

Appendix A: Default Parameters for bwtekTESTUSB

Model	Timing Mode	Pixel Number	Input Mode	Minimum Integration Time
BRC111A	1	2048	1	9 - 65535 ms
BRC112	1	2048	1	5 - 65535 ms
BRC131E	3	3680	1	9 - 65535 ms
BRC311	1	2048	1	9 - 65535 ms
BRC711E-512	3	512	2	3 - 65535 ms
BRC711E-1024	3	1024	2	3 - 65535 ms
BTC111	1	2048	1	9 - 65535 ms
BTC112	1	2048	1	5 - 65535 ms
BTC211	4	512	1	1 - 65535 ms
BTC221E-256	3	256	2	15 - 65535 ms
BTC251	4	512	1	1 - 65535 ms
BTC251E-512	3	512	11	1 - 65535 ms
BTC261E-256	3	256	13	1 - 65535 ms
BTC261E-512	3	512	13	1 - 65535 ms
BTC261E-1024	3	1024	13	1 - 65535 ms
BTC262E-256	3	256	14	1 - 65535 ms
BTC262E-512	3	512	14	1 - 65535 ms
BTC262E-1024	3	1024	14	1 - 65535 ms
BTC311	1	2048	1	9 - 65535 ms
BTC611E-512	6	512	2	27 - 65535 ms
BTC611E-1024	6	1024	2	50 - 65535 ms
BTC711E-512	3	512	2	15 - 65535 ms
BTC711E-1024	3	1024	2	15 - 65535 ms
BTC811 (In Development)	Future Product	2048		N/A

Model	Timing Mode	Pixel Number	Input Mode	Minimum Integration Time
BTF111	1	2048	1	9 - 65535 ms
BTF113	1	2048	1	9 - 65535 ms
BTU111	1	2048	1	9 - 65535 ms
BTU112	1	2048	1	5 - 65535 ms
BRU111	1	2048	1	9 - 65535 ms
BRU112	1	2048	1	5 - 65535 ms
BTR111 (MiniRam)	1	2048	1	9 - 65535 ms
BTC500	NA	256	NA	3.33 - 19980 µs
BRC100	NA	2048	NA	35 - 65535 ms
BRC110	NA	2048	NA	35 - 65535 ms
BTC110	0	2048	1	21 - 65535 ms
BTC200	NA	512	NA	6 - 65535 ms
BTC251E-512-RS232	NA	512	NA	1 - 65535 ms
BTC320	NA	2048	NA	21 - 65535 ms
BTC15x	1	2048	1	35 - 65535 ms

Warranty Terms and Conditions

B&W Tek's end user products, OEM modules, and components are warranted to be free from defects in materials and workmanship for a period of 12 months, 6 months, and 90 days, respectively, from the date of initial shipment. This warranty does not extend to incidental or consequential damages and to damage caused by negligent or improper handling in use, storage, nor for products for which the original identification markings or labels have been removed, defaced or altered.

Special contracts or contracts for nonstandard products may have modified terms of warranty and, in such cases, the terms as stated in the individual contract must be signed by the duly authorized officer of B&W Tek and will supersede the standard terms.

B&W Tek will make final determination as to cause or existence of defect and, at its option repair or replace the products, which prove to be defective during the warranty period. Products replaced under warranty will be warranted only for the balance of the warranty period from the original supplied equipment.

This warranty extends only to the original purchaser of the equipment from B&W Tek. The purchaser must notify B&W Tek within 15 days of first noticing the defect and promptly return the defective product upon receipt of RMA number(s) before expiration of the warranty period.

Products believed by purchaser to be defective shall be returned to B&W Tek transportation and insurance prepaid by purchaser. Repaired or replaced products will be returned to purchaser by B&W Tek, FOB city destination within the Continental United States, Transportation beyond these limits will be charged to purchaser.

Warranty Return Procedures

Review terms of purchase and date of shipment to determine validity of warranty claim. Warranty claim should only be made for products within terms of warranty policy.

Go to our website to apply for an RMA (Return material authorization) number:
http://bwtek.com/rma.htm

Or Contact us by E-mail at info@bwtek.com

For customers in the USA and countries where distributorship and/or representation is not available, all claims should be addressed to:

Customer Service
B & W TEK INC.
19 Shea Way, Suite 301
Delaware Industrial Park
Newark, DE 19713

Be prepared to furnish:

a. Product Model number and serial number;
b. Date of shipment/purchase;
c. Brief description of the problems encountered;
d. Name of contact person and phone number(s) at your organization for further communication

Obtain B&W Tek's instructions for transportation and packaging and ship the product (freight and insurance prepaid) with the proper documentation containing the RMA number and the information specified above.

B&W Tek will advise the purchaser of its determination of warranty at the earliest possible time. Providing complete information as requested will expedite the procedure.