? Sigma
instruments
1518 Duff Drive ? Fort Collins, Colorado 80524 ? (970) 416-9660 ? Fax (970) 416-9330

Communications with the SQC-122

There are three possibilities for communicating with the SQC-122. The first is a DOS-based terminal program. The next is a 32 bit Windows DLL. The third is a Windows program that can configure and operate the SQC-122.

SQM-TERM.EXE

This simple DOS-based terminal program allows you to send a command and receive a response via the computer's RS-232 port. The computer's COM1 port baud rate must be set to match the SQC-122 before the program is run. Source code (SQM-TERM.C) is provided, so it can be customized easily. The command and response strings are detailed in the attached Communications Protocol.

SIGMACOM.DLL

This 32 bit DLL should be placed in your Windows System directory. No Windows registration is needed. The first call should be to InitCom to establish the comm port and baud rate. Functions and their calling convention are listed in the attached SIGMACOM.DLL Functions document.

SQC-122.EXE

This program uses SIGMACOM.DLL to communicate with the SQC-122. It must be installed in Windows using the Setup SQC-122 installation program. It allows you to set film parameters and names, download them to the SQC-122, and perform front panel functions. This program is written in Visual Basic, contact Sigma Instruments if you need the source code for this program.

? Sigma
instruments
1918 Duff Drive ? Fort Collins, Colorado 80524 ? (970) 416-9660 ? Fax (970) 416-9330

SQC-122 Communications Protocol

The SQC-122 communicates with a host computer via an ASCII based protocol. The instrument defaults to 19200 baud, 8 data bits, and no parity. The baud rate can be changed in the System Menu of the SQC-122, but is always 8 data bits with no parity.

The basic protocol is:

<sync character> <length character> <1 to n data characters> <CRC1><CRC2>

Once a valid command has been transmitted to the SQC-122, a response is returned. The structure of the packet is identical in both the command and response. In the response, the first character is a Response Status. These are summarized in the following table.

Response Letter	Meaning
A	Command understood, normal response
B	Command understood, but instrument reset
C	Invalid command
D	Problem with data in command
E	Instrument in wrong mode for this command

The sync character is an exclamation point '!'. Anytime this character is received, the communications for that packet is reset. Following the sync character is the length character. This is the number of characters in the packet starting with the length and counting the 2 CRC characters. This character has a decimal 34 added to it so there cannot accidentally be a sync character embedded in the packet. The two character CRC is computed using the following algorithm:

1. The CRC is initialized to 3FFF hex.

2. Each character in the message is examined, bit by bit, and added to the CRC in the following manner:

    a) The character is exclusive or'd with the CRC.
    b) The CRC is shifted right one bit position.
    c) If the character's least significant bit is a 0 then the CRC is exclusive or'd with 2001 hex.
    d) Steps b and c are repeated for each of the 8 bits in the character.

The CRC contains 14 significant bits. This is split into two characters of 7 bits each, and then a decimal 34 is added to offset the character outside the range of the Sync

Character. See the code example in the SQM-TERM.C file for an example of managing the CRC.

These examples represent the data in unpacked format to better illustrate the function of the specific commands.

Command: @

Parameters: None

Description: Returns the model number and software version number.

Example: @ SQC122 Ver 1.2

Command: L

Parameters: [1..2]

Description: Read the current Rate for a channel.

Example: L1 A9.32 Channel one's rate is 9.32 Angstroms/S

Command: M

Parameters: None.

Description: Read the current Average Rate.

Example: M A10.42 Average Rate is 10.42 Angstroms/S

Command: N

Parameters: [1..2]

Description: Read the current thickness for a channel.

Example: N2 A1.187 Channel two's Thickness is 1.187 Kilo Angstroms.

Command: O

Parameters: None.

Description: Read the current Average Thickness

Example: O A2.376 The current Average Thickness is 2.376 kilo Angstroms.

Command: P

Parameters: [1..2

Description: Read the current Frequency for a channel.

Example: P2 A5701563.2 Channel two's current Frequency 5701563.2Hz

---

Command: R

Parameters: [1..2]

Description: Read the Crystal Life for a channel.

Example: R2 A57.82 Channel two's remaining life is 57.82%.

Command: S

Parameters: None.

Description: Zero Average Thickness and Rate.

Example: S A

Command: T

Parameters: None.

Description: Zero Time

Example: T A Zeroes time display on unit.

Command: U

Parameters: 0 to 33

Description: Controls operation of the SQC-122.

0 = Start Process	6 = Start Process 1
1 = Stop Process	7 = Start Process 2
2 = Start Layer	8-30 = Start Process 3-25
3 = Stop Layer	31 = Soak Hold
4 = Start Next Layer	32 = Zero Thickness
5 = Force Final Thickness	33 = Zero Time

Example: U1 A Starts the first layer of the active process.

Command: V

Parameters: None

Description: Controls operation of the SQC-122.

0 = Stopped	8 = Soak 2	16 = Start Next Layer
1 = Crystal Verify	9 = Soak Hold	17 = Crystal Fail
2 = Initialize layer	10 = Shutter Delay	18 = Stop Layer
3 = Manual Start Layer	11 = Deposit	19 = Manual Power
4 = Pocket Rotate	12 = Rate Ramp	
5 = Ramp 1	13 = Rate Ramp Deposit	
6 = Soak 1	14 = Timed Power	
7 = Ramp 2	15 = Idle Ramp	

Example: U1 A9 Units is in Soak Hold phase.

Command: Y

Parameters: None.

Description: Read the Power-Up Reset flag. The Power-Up Reset flag is set during boot-up of the unit and stays set until read through the RS-232 interface.

After the flag is read, it is reset and will not be set again until the unit is power cycled.

Example:	Y	A1	Power-Up Reset flag is set.
	Y	A0	Power-Up Reset flag is reset.

Command: Z

Parameters: None.

Description: Set all Film and System parameters to defaults.
Note that this command can take over 1 second to complete

Example:	Z	A	All Film and System parameters are set to defaults.

? Sigma
instruments
1918 Duff Drive ? Fort Collins, Colorado 80524 ? (970) 416-9660 ? Fax (970) 416-9330

SIGMACOM.DLL Function Descriptions

This dll acts as an interpreter between an application and the SQC122. The dll transforms function calls to specific command sequences that the unit understands.

Transfer of data to the unit, in general, requires two function calls. The first function call is to transfer the data to the unit. The data to be sent is usually contained in the function's parameter(s). The second function call is to *ChkCommDone*. This function call ensures that the data was sent properly to the unit.

Data retrieval requires three function calls. The first function call is used to tell the unit what data is being requested. The second function call is to *ChkCommDone*. This function call is used to determine when all of the data has been transferred from the unit to the dll or if an error occurred in the communications. The third function call is used to retrieve the data from the dll.

InitComm

Parameters: 16 Bit Integer, 32 Bit Integer
Return : 16 Bit Integer.

InitComm is used to initialize the dll com port. The function's first parameter is the com port number to initialize (1 - 99 are valid). The second parameter is the baud rate for the port. The function returns zero if initialization was successful or a bit flag to indicate the failure of the initialization :

bit 0 : Communications Port handle is invalid.
bit 1 : Communications Port Set parameters invalid (Baud Rate)
bit 2 : Communications Port Set timeouts invalid.
bit 3 : Communications Port Set mask invalid.
bit 4 : Communications Port Error – Already exists.
bit 5 : Communications Port Set Read Thread fail.
bit 6 : Communications Port Set Read Thread priority fail.

Example:

```
ReturnVal =InitComm(1,19200)     initialize Com1 to 19200 baud
if (ReturnVal != 0)              if port did not initialize correctly
    CloseComm()                  close the port
```

CloseComm

Parameters: None.
Return : 16 Bit Integer, always returns a 1.

CloseComm is used to close the currently opened communications port. *CloseComm* should always be used before attempting to open another port or before exiting the dll's calling application. The dll can have only one port open at a time.

Example:

ReturnVal =CloseComm() Close the currently open comm port

ChkCommDone

Parameters: None.
Return : 16 Bit Integer.

ChkCommDone is used to check the status of a single communications iteration. The function returns one of five different types of values:

-1:	communications not complete
Positive integer :	communications complete, value is byte count of returned message.
-99 :	communications complete, but return message incomplete due to timeout with unit.
-98 :	communications complete, but return message not valid due to a CRC error.
-97 :	communications complete, but message not understood by unit.

Example:

ReturnVal =ChkCommDone() check communications status

ClearComm

Parameters: None.
Return : 16 Bit Integer, always returns a 1.

ClearComm is used to clear the communications buffers in the dll.

Example:

```
ReturnVal =ClearComm()          Clear the comm buffers in the dll
```

SendGetVers

Parameters: None.
Return : 16 Bit Integer, always returns a 1.

SendGetVers is used to retrieve the software version of the unit from the unit. This function must precede the use of the *GetVers* function

GetVers

Parameters: Pointer to Null-Terminated string.
Return : 16 Bit Integer, always returns a 1.

GetVers is used to retrieve the software version of the unit from the dll. This function must be preceded by the *SendGetVers.* The Null-terminated string is used to return the version from the dll.

Example:

```
ReturnVal = SendGetVers()                tell unit to transfer version to dll
do while(ChkCommDone == -1)              wait for comm to finish
ReturnVal = GetVers(&VersionString[0])   VersionString contains version info
```

SetActiveProcess

Parameters: Process # (16 Bit Integer).
Return : 16 Bit Integer, always returns a 1.

SetActiveProcess is used to select the active Process in the unit.

Example:

```
ReturnVal = SetActiveProcess(1t)         set to Process 1
do while(ChkCommDone == -1)              wait for comm to finish
```

SetProcess

Parameters: Process # (16 Bit Integer), Pointer to a Process Structure.
Return : 16 Bit Integer, always returns a 1.

SetProcess is used to set a Process' Parameters in the unit. All of the parameters are passed to the function through the Process Structure.

Example:

```
ReturnVal = SetProcess(1, &ProcessStruct)    set Process 1 parameters
do while(ChkCommDone == -1)                  wait for comm to finish
```

SendGetProcess

Parameters: 16 Bit Integer.
Return : 16 Bit Integer, always returns a 1.

SendGetProcess is used to get a Process' parameters from the unit. The Processr number (1-25) is passed to the function. This function must precede the use of *GetProcess*.

GetProcess

Parameters: Pointer to a Process Structure.
Return : 16 Bit Integer, always returns a 1.

GetProcess is used to retrieve a Process' parameters (requested by *SendGetProcess)*. The parameters are passed through the Process Structure.

Example:

```
ReturnVal = SendGetProcess(ProcessNum)    tell to transfer Process#
do while(ChkCommDone == -1)               wait for comm to finish
ReturnVal = GetProcess(&ProcessStruct)    contains Process info
```

SetLayer

Parameters: Layer # (16 Bit Integer), Pointer to a Layer Structure.
Return : 16 Bit Integer, always returns a 1.

SetLayer is used to set a Layer's parameters in the unit. All of the parameters are passed to the function through the Layer Structure.

Example:

```
ReturnVal = SetLayer(1, &LayerStruct)     set Layer 1 parameters
do while(ChkCommDone == -1)               wait for comm to finish
```

SendGetLayer

Parameters: 16 Bit Integer.
Return : 16 Bit Integer, always returns a 1.

SendGetLayer is used to get a Layer's parameters from the unit. The Layer number (1-99) is passed to the function. This function must precede the use of *GetLayer*.

GetLayer

Parameters: Pointer to a Layer Structure.
Return : 16 Bit Integer, always returns a 1.

GetLayer is used to retrieve a Layer's parameters (requested by *SendGetLayer)*. The parameters are passed through the Film Structure.

Example:

```
ReturnVal = SendGetFilm(FilmNum)      tell unit to transfer Film #
do while(ChkCommDone == -1)           wait for comm to finish
ReturnVal = GetLayer(&LyrStruct)      LyrStruct contains Film info
```

DeleteLayers

Parameters: None
Return : 16 Bit Integer, always returns a 1.

Delete Layers removes all layers from the active process.

Example:

```
ReturnVal = DeleteLayers              set film parameters
do while(ChkCommDone == -1)           wait for comm to finish
```

Set122Film

Parameters: Film # (16 Bit Integer), Pointer to a Film Structure.
Return : 16 Bit Integer, always returns a 1.

Set122Film is used to set a Film's parameters in the unit. All of the parameters are passed to the function through the Film Structure.

Example:

```
ReturnVal = SetFilm(&FilmStruct)      set film parameters
do while(ChkCommDone == -1)           wait for comm to finish
```

SendGetFilm

Parameters: 16 Bit Integer.
Return : 16 Bit Integer, always returns a 1.

SendGetFilm is used to get a Film's parameters from the unit. The Film number (1-25) is passed to the function. This function must precede the use of *Get122Film*.

Get122Film

Parameters: Pointer to a Film Structure.
Return : 16 Bit Integer, always returns a 1.

Get122Film is used to retrieve a Film's parameters (requested by *SendGetFilm)*. The parameters are passed through the Film Structure.

Example:

```
ReturnVal = SendGetFilm(FilmNum)      tell unit to transfer Film #
do while(ChkCommDone == -1)           wait for comm to finish
ReturnVal = Get122Film(&FilmStruct)   FilmStruct contains Film info
```

Set122Material

Parameters: Material # (16 Bit Integer), Pointer to a Material Structure.
Return : 16 Bit Integer, always returns a 1.

Set122Material is used to set a Material's parameters in the unit. All of the parameters are passed to the function through the Material Structure.

Example:

```
ReturnVal = SetMaterial(&MaterialStruct)      set Material parameters
do while(ChkCommDone == -1)                   wait for comm to finish
```

SendGetMaterial

Parameters: 16 Bit Integer.
Return : 16 Bit Integer, always returns a 1.

SendGetMaterial is used to get a material's parameters from the unit. The Material number (1 - 99) is passed to the function. This function must precede the use of *Get122Material*.

Get122Material

Parameters: Pointer to a Material Structure.
Return : 16 Bit Integer, always returns a 1.

Get122Material is used to retrieve a Material's parameters (requested by *SendGetMaterial)*. The parameters are passed through the Material Structure.

Example:

```
ReturnVal = SendGetMaterial(MaterialNum)     request transfer Material #
do while(ChkCommDone == -1)                  wait for comm to finish
ReturnVal = Get122Material(&MaterialStruct)  MaterialSruct contains info
```

SendGetRunState

Parameters: None
Return : 16 Bit Integer, always returns a 1.

SendGetRunState is used to retrieve the operating phase of the deposition. This function must precede the use of *GetRunState*.

GetRunState

Parameters: None
Return : 16 Bit Integer

GetRunState returns an integer whose value represents the current operating phase of the instrument, as shown below:

0 = Stopped	8 = Soak 2	16 = Start Next Layer
1 = Crystal Verify	9 = Soak Hold	17 = Crystal Fail
2 = Initialize layer	10 = Shutter Delay	18 = Stop Layer
3 = Manual Start Layer	11 = Deposit	19 = Manual Power
4 = Pocket Rotate	12 = Rate Ramp	
5 = Ramp 1	13 = Rate Ramp Deposit	
6 = Soak 1	14 = Timed Power	
7 = Ramp 2	15 = Idle Ramp	

UnitControl

Parameters: 16 Bit Integer.
Return : 16 Bit Integer, always returns a 1.

UnitControl is used to set the operating state of the SQC-122.

0 = Start Process	6 = Start Process 1
1 = Stop Process	7 = Start Process 2
2 = Start Layer	8-30 = Start Process 3-25
3 = Stop Layer	31 = Soak Hold
4 = Start Next Layer	32 = Zero Thickness
5 = Force Final Thickness	33 = Zero Time

Sigma
instruments
1318 Duff Drive ? Fort Collins, Colorado 80524 ? (970) 416-9660 ? Fax (970) 416-9330

SendGetInputs

Parameters: None
Return : 16 Bit Integer, always returns a 1.

SendGetInputs is used to retrieve the mapping of the 8 physical digital inputs to the 35 Input events. This function must precede the use of *GetInputs*.

GetInputs

Parameters: Pointer to an IO structure (integer array(7).
Return : 16 Bit Integer, always returns a 1.

GetInputs is used to retrieve the mapping of the 8 physical digital inputs to the 35 Input events. Values in the array(0-7) are the events (0-34) assigned to each input. Input events are:

0 = Start Process	6 = Start Process 1	31 = Start Soak Hold
1 = Stop Process	7 = Start Process 2	32 = Zero Thickness
2 = Start Layer	8 to 30 =	33 = Zero Time
3 = Stop Layer	Start Process n-5	34 = Pocket Ready
4 = Start next layer		
5 = Force Final Thickness		

Example:

```
ReturnVal = SendGetInputs          request transfer Inputs
do while(ChkCommDone == -1)        wait for comm to finish
ReturnVal = GetInputs(&IoP)        I/O Sruct contains info
```

SetInputs

Parameters: Pointer to a I/O Structure.
Return : 16 Bit Integer, always returns a 1.

SetInputs is used to set the event definitions of the eight digital inputs as described above. All of the parameters are passed to the function through the I/O Structure.

Example:

```
ReturnVal = SetInputs(&IoP)        set Input events
do while(ChkCommDone == -1)        wait for comm to finish
```

SendGetRelays

Parameters: None
Return : 16 Bit Integer, always returns a 1.

SendGetRelays is used to retrieve the mapping of the 8 physical relays to the 21 Relay events. This function must precede the use of *GetRelays*.

GetRelays

Parameters: Pointer to an IO structure (integer array(7).
Return : 16 Bit Integer, always returns a 1.

GetRelays is used to retrieve the mapping of the 8 physical relays to the 21 relay events. Values in the array(0-7) are the events (0-20) assigned to each input. Relay events are:

0 = Source Shutter 1	7 = Deposit Phase	14 = Time Setpoint
1 = Source Shutter 2	8 = PreCond Phase	15 = Thickness Setpoint
2 = Sensor Shutter 1	9 = SoakHold Phase	16 = Final Thickness
3 = Sensor Shutter 2	10 = Process Active	17 = Pocket BCD Bit 0
4 = All Sensor Fail	11 = Manual Mode	18 = Pocket BCD Bit 1
5 = Sensor 1 Fail	12 = Max. Power	19 = Pocket BCD Bit 2
6 = Sensor 2 Fail	13 = Stopped	20 = Secondary Sensor

Example:

```
ReturnVal = SendGetRelays          request transfer Relays
do while(ChkCommDone == -1)        wait for comm to finish
ReturnVal = GetRelays(&IoP)        I/O Sruct contains info
```

SetRelays

Parameters: Pointer to a I/O Structure.
Return : 16 Bit Integer, always returns a 1.

SetRelays is used to set the event definitions of the eight realys as described above. All of the parameters are passed to the function through the I/O Structure.

Example:

```
ReturnVal = SetRelays(&IoP)        set Relay events
do while(ChkCommDone == -1)        wait for comm to finish
```

SendGetSys1

Parameters: None.
Return : 16 Bit Integer, always returns a 1.

SendGetSys1 is used to get the System1 Parameters from the unit. This function must precede the use of the *GetSys1* function.

Get122Sys

Parameters: Pointer to a System1 Structure.
Return : 16 Bit Integer, always returns a 1.

GetSys1 is used to retrieve the System1 Parameters from the dll. The parameters are passed through the System1 Structure.

Example:

```
ReturnVal = SendGetSys1()          tell unit to transfer System1 parameters
do while(ChkCommDone == -1)        wait for comm sequence to finish
ReturnVal = GetSys1(&Sys1Struct)   Sys1Struct contains System1 info
```

Set122Sys

Parameters: Pointer to a System2 Structure.
Return : 16 Bit Integer, always returns a 1.

SetSys2 is used to set the System2 Parameters. The parameters are passed to the function through the System2 Structure.

Example:

```
ReturnVal = SetSys2(&Sys2Struct)   set System2 parameters to
                                   Sys2Struct values
do while(ChkCommDone == -1)        wait for comm to finish
```

SendGetAllData

Parameters: None.
Return : 16 Bit Integer, always returns a 1.

SendGetAllData is used to get the data from the unit. This function must precede the use of the *Get122Data* function.

Get122Data

Parameters: Pointer to an Measurement Data Structure.
Return : 16 Bit Integer, always returns a 1.

Get122Data is used to retrieve the data from the dll. The parameters are passed through the Measurement Data Structure. If the TimeStamp parameter of the AllData structure returned is equal to -1 then the unit does not have new data available.

Example:

```
ReturnVal = SendGetAllData()              tell unit to transfer AllData
do while(ChkCommDone == -1)               wait for comm to finish
ReturnVal = Get122Data(&MeasurementDataStruct) Measurement
                                          DataStruct contains run info
if (AllData.TimeStamp != -1) then         if new data available
    ProcessData()                          then graph or save data
while(Running)
```

Data Structures:

The size of each data type in the structures is :

double : 8 bytes, the LSB is thrown out before transmission to the unit.
single : 4 bytes
int : 2 bytes.
char : 1 byte.

Process Data

int	NumLayers	number of layers in the process
int	FirstLayer	film number of first layer
char	ProcName[16]	process name

Layer Data

single	Rate	layer rate
single	FnlThick	final thickness
integer	TimeSet	time setpoint (sec.)
single	ThickLimit	thickness limit setpoint
int	FilmNum	layer's film number
int	NextLayer	next layer number
int	StartMode	auto(0) or manual (1)
int	Default	l ayer is in the active process (1)

Film Data

int	Pterm	proportional term
single	Iterm	integral term
single	Dterm	differential term
int	Sensor1	sensor 1 on/off
int	Sensor2	sensor 2 on/off
int	Output	active output (0/1)
single	MaxPower	0 to 100%
single	SlewRate	0 to 100%
int	MatIndex	film material number
single	R1Power	ramp 1 power (%)
single	R1Time	ramp 1 time (sec.)
single	S1Time	soak 1 time (sec.)
single	R2Power	ramp 2 power (%)
single	R2Time	ramp 2 time (sec.)
single	S2Time	soak 2 time (sec.)
single	IdlePwr	idle power (%)
single	IdleTime	idle time (sec.)
int	Pocket	source pocket

int	Tooling	film tooling factor
char	Name[16]	film name

Material Data

single	Density	material density (gm/cc)
single	ZFactor	material z factor
char	MatName[16]	material name

InOut Data

Int	Io(8)	

122Sys Data

double	Period	measurement period (sec.)
int	SysTool	system tooling
int	Xtool(1)	individual sensor tooling
int	SimMode	simulation mode (1 = on)
double	FMin	minimum frequency
double	FMax	maximum frequency
double	PwrScale(1)	
double	RateDev	

Measurement Data

double	TimeStamp	time relative to start time data was acquired
double	AvgRate	average rate
double	AvgThick	average thickness
double	Power(1)	average power
double	ChRate(5)	up to six individual channels of rate
double	ChThick(5)	up to six individual channels of thickness
double	ChFreq(5)	up to six individual channels of frequency