NIRSPEC

UCLA Astrophysics Program **U.C. Berkeley** **W.M.Keck Observatory**

Fred Lacayanga

September 14, 1998

NIRSPEC Software Programming Note 27.00
Server code file locations and descriptions

The server code resides in: **`/kroot/kss/nirspec/keyword`**

`Authorize` : Nirspec authorization file
`Nirspec.cfg` `-> /tmp/Nirspec.cfg` : Backup configuration file
`NirspecConfig.csh*` : Configuration file that is run after server startup. Modifies keywords
`NirspecInstr.log` `-> /tmp/NirspecInstr.log` : Instrument Log
`NirspecServerInit` : Nirspec server environment variables
`error.h`

Server error codes and messages
Client error codes and messages
TSP error codes and messages
Motor error codes and messages

`fileio.c` : File I/O routines

`Fits_init( void )` : initialize FITS header using header environment variables
`Fits_headerOpen( int chan, char *file )` : Open FITS header file
`Fits_headerInit( int chan, int nf )` : Initialize FITS header keyword table
`Fits_headerUpdate( int chan, int nf )` : Update FITS keyword values
`Fits_write( int chan, int naxis, long naxis1, long naxis2, int bitpix )` : Write FITS file from frame buffer
`Fits_copy( int chan )` : copy a tmpfs FITS file from memory to disk
`FileIO_open( void )` : file open I/O
`FileIO_write( int chan, int frame_num, int abort_flag )` : write image data
`FileIO_copy( char *from_file, char *to_path, char *to_file )` : File copy
`FileIO_saveTest( int chan )` : save test frame (in FITS) from tmpfs to disk
`FrameBuffer_readSection( long pix_off, int num_pix )` : read a section of pixel data from frame buffer
`Filename_check( int chan, char *filename )` : Check whether a file exists

Broadcast duplicate filename and current file number

`Filename_make( int chan, char *filename )` : Make an image file name by incrementing file running number. Current limit is 10,000 files per night
`interval( struct timeval *start, struct timeval *end )` : calculate a time interval
`write_dcs_keywords( int channel, int num_keywords )` : write DCS keywords to FITS header.

Open DCS keyword library
Send KTL_HEADERS request to the DCS
Format the output
Write to FITS header

`header_info.nirspec_scam` : FITS keyword file
`header_info.nirspec_spec` : FITS keyword file
`libnirspec_keyword.so.0.0*` : Nirspec sharable keyword library
`libnirspecsim_keyword.so.0.0*` : Nirspec sharable keyword library for the simulated server
`makefile*`

makefile_purify*

makefile_sim*

makefile_sim_purify*

motor.c : Step motor routines

STP_initAllMotors(void) : commented out

STP_init(void) : send `init` command to transputers

STP_movePos(void) : send `pos` command to transputers

STP_filterPos(char *pos_name) : not used anymore

STP_getFilterPosIndex(char *pos_name) : get dual filter wheel position index

STP_moveSteps(int motor_index, long steps) : send step command to transputers

STP_moveAnglePos(int motor_index, double angle) : move image rotator, echelle, or disperser to specified angle.

STP_track(int velocity) : send `track` command to transputers

STP_abort(int motor_index) : send `abort` command to transputers

STP_readPos(int motor_index) : send `location` command to transputers

STP_readSwitch(int motor_index) : send `switch` command to transputers

STP_setPos(char *motor_name) : not utilized

STP_getZeroPos(int motor_index, int param) : not utilized

STP_initZeroPos(void) : not utilized

STP_checkLock(int motor_index) : commented out

STP_getMotorIndex(char *motor_name) : determine motor index using `motor_table` in `nirspec.h`

STP_getPosIndex(int motor_index, char *pos_name) : determine position index for a specific motor using the motor tables in `nirspec.h`

STP_checkRange(int motor_index) : not utilized

STP_checkRange2(int motor_index, long *steps) : not utilized

STP_motorIndex2cid(int motor_index) : not utilized

nirspec.h

Min/max itime for spec and scam
Transputer cids
Nirspec keyword table
Sampling mode table
Lamp table
Motor indexes
Motor initialized positions
Motor list
Filter list
Slit list
RPC service handle structure
Observing parameter structure
FITS table structure

nirspec.log -> `/var/log/nirspec.log` : syslog of all nirspec events

nirspec_keyword.c

nirspec_server_bin*

nirspecsim_server_bin*

nrpc.h : Generated using rpcgen

nrpc_clnt.c : Generated using rpcgen

nrpc_face.c : Routines that act as an interface between KICS and RPC procedures

nrpc_server.c : RPC server routines

main(int argc, char *argv[])

Check arguments
Set up error logging
Initialize keywords on recovery
Initialize transputer values

- Initialize all motors
- Open file I/O
- Make connection to DCS keyword library
- Open log file
- Create timeout pipe
    - Poll TSP link
- Asynchronous I/O loop
    - Read TSP link pipe
        - Process message from tsplink
- Write engineering log
- Process DCS events
- Process client requests

rpc_prog_1(rqstp, transp) : Server dispatch routine

config_default(void) : Initialize keywords from keyword table

config_backup(void) : Initialize keywords from configuration auto-backup file.

config_update(int i) : Update configuration auto-backup file (called on keyword writes)

cid_init(void) : Initialize all keywords with transputer cids. Sends default cids and params to tranputers.

instrlog_open(void) : Open instrument log file.

instrlog_update(void) : Update log file

tsplink_msg_process(int cid, long param) : Process messages from TSP link

- Frame Ready signals
- Temp sensor response
- Motor replies
    - Initialize : Initialize motor
    - InitLoc : Tells transputers that the motor is at a specific step number
    - Tracking : Image rotator tracking velocity
    - Step Move : Move motor a relative number of step positions
    - Positional Move : Move to a numbered position
    - Location : Read motor step location
    - Switch status : Read switch positions
    - Abort : Abort motor

STP_resetCounter(int motor_index) : reset step counter (step counter not really used)

STP_initErrorInfo(int motor_index, int code)

- Interpret initialization reply from transputers
    - `STP_ERR_INIT_FAIL;`
    - `STP_ERR_INIT_FAIL_NO_SWITCH;`
    - `STP_ERR_INIT_FAIL_STUCK_SWITCH;`
    - `STP_ERR_INIT_FAIL_INTERM_SWITCH;`
    - `STP_ERR_INIT_ABORT;`
    - `STP_ERR_INIT_SUCCESS_PRIMARY;`
    - `STP_ERR_INIT_SUCCESS_SECONDARY;`
- Broadcast filter and slit positions
    - (Motor)`stat`
    - (Motor)`initstat`
    - (Motor)`name`
- Broadcast status messages to `tspstat` keyword

STP_moveErrorInfo(int motor_index, int code)

- Interpret motor move reply from transputers
    - `STP_ERR_MOVE_VELOCITY;`
    - `STP_ERR_MOVE_POS_SWITCH;`
    - `STP_ERR_MOVE_ABORT;`
    - `STP_ERR_MOVE_LIMIT_SWITCH;`
    - `STP_ERR_MOVE_POS_UNKNOWN;`
    - `STP_ERR_MOVE_BUSY;`

    - STP_ERR_MOVE_INVALID_PARAMETER;
    - STP_ERR_MOVE_INVALID_COMMAND;
    - STP_ERR_MOVE_SUCCESS;
  - Broadcast filter and slit positions
      - (Motor)name
      - (Motor)stat
  - Broadcast status messages to tspstat keyword
- ***STP_abortInfo(int motor_index)*** : Broadcast status messages to tspstat keyword
- ***STP_locationInfo(int motor_index, long steps)*** : Print out motor step location information
- ***STP_switchInfo(int motor_index, int code)*** : Print out positional switch information
- ***lookup(char *keyword)*** : get keyword index in KeywordTable[ ] array.
- ***lookup2(int cid)*** : get keyword index in KeywordTable[ ] array using the cid
- ***broadcast_tsp_msg(char* keyword, char* message, int value, int type)*** : Send an RPC keyword broadcast to all clients. Broadcasts keywords with either an integer or string value.
  - Place keyword value in keyword value array kvalue[ ]
  - Broadcast the keyword using rpc_broadcast_1( ) function.
- ***convert_temp_sensor_data(long data)*** : Interpret temperature sensor data sent back from the transputers.
  The transputer sends back:
  (temperature in millikelvins) + (1000000 * Sensor Channel number)
- ***create_dcs_interest(KTL_HANDLE *ktl_handle)*** : No need to create interest. watch_imrot does this. This function is commented out.

nrpc_svc.c : Generated using rpcgen

nrpc_svc_proc.c : RPC remote procedures

- ***rpc_access_1(rpc_arg *arg)*** : check access permission of the client using the authorization file
- ***rpc_setbroadcast_1(rpc_arg *arg)*** : set up RPC broadcast
- ***rpc_closebroadcast_1(rpc_arg *arg)*** : close RPC broadcast
- ***rpc_read_1(rpc_arg *arg)*** : reads keyword value from keyword value array kvalue[ ]
- ***rpc_write_1(rpc_arg *arg)*** : remote procedure for writing a keyword to TSP link. Most keywords just go straight through, although there are a bunch of special cases (mostly motor commands) that require some processing first.
  - tpupdate
  - tpreset
  - dsheaders
  - tptrace
  - tpcid
  - tpparam
  - savetest
  - fileovwr
  - fileovwr2
  - go
  - go2
  - Motor commands
    - init
    - track
    - step
    - pos
    - loc
    - sw
    - abort
- ***rpc_broadcast_1(rpc_arg *arg)*** : make an RPC broadcast
- ***client_put(char *host, int prognum, CLIENT *client)*** : put an RPC broadcast client
- ***client_list(void)*** : list all RPC broadcast clients

- `client_delete( char *host, int prognum )` : delete an RPC broadcast client
- `broadcast_status( char* keyword, char* message )` : Broadcast a status message string.
- `check_integration_limits( int cid )` : Determine if integration time is allowable for selected sampling mode.
- `check_keyword_range(int i, rpc_arg *arg)` : Range checking for keyword values. Uses min/max from `KeywordTable[ ]` array in `nirspec.h`
- `get_DCS_Headers( void )` : test subroutine to grab DCS headers

`nrpc_xdr.c` : Generated using rpcgen

`run_dcs*`

`run_nirspec_server*`

`stop_nirspec*`

`stop_nirspecsim*`

`tspcom.c` : Routines that handle TSP link I/O

- `TSPCom_init( char *boot_file )`
- `TSPCom_boot( char *filename )`
- `TSPCom_close( void )`
- `TSPCom_reset( char *boot_file )`
- `TSPCom_sendCommand( int cid, long param )`
- `TSPCom_sendMessage( int *cid, long *param )`
- `TSPCom_getMessage( int *cid, long *param )`
- `TSPCom_getFrame( int chan, int frame_num )`
- `TSPCom_getPacket( void )`
- `TSPCom_sendPacket( void )`
- `TSPCom_trace( int mode )`

`tspcom.h` : Link error/status codes