- **cre_init(int driver, int hdid, int sources, int mode)**
  - ***int driver: Specifies the driver type. We use it to determine sample rate, and to open live audio by default.***
    - ⇒ The following are 48kHz drivers:
      - **Atrn_ARS1: Acoustic Room Simulator release 1: outdated.**
      - **Atrn_DPL1: Doppler Processor 1: 1 source per Spectrum.**
      - **Atrn_CMP1: Convolvotron Minimum Phase release 1.**
      - **Atrn_ARS2: Acoustic Room Simulator release 2.**
      - **Atrn_AS48: for AuSIM Engineering server 48kHz driver**
    - ⇒ The following are 44.1kHz drivers:
      - **Atrn_BMP1: Alpha/Beachtron Minimum Phase release 1.**
      - **Atrn_BMP2: Alpha/Beachtron Minimum Phase 22kHz.**
      - **Atrn_BMP3: Alpha/Beachtron Minimum Phase Doppler.**
      - **Atrn_BMP4: Alpha/Beachtron Transmission Loss Filter.**
      - **Atrn_SAB1: for internal use**
      - **Atrn_SAD1: for internal use**
      - **Atrn_ARS3: Acoustic Room Simulator release 3 (July 1995)**
      - **Atrn_ASM1: for AuSIM Engineering server**
    - ⇒ These two drivers automatically open live audio on all sources:
      - **Atrn_CMP1:**
      - **Atrn_BMP1:**
  - ***int hdid: Specifies which listener to initialize. Specify different listener numbers for multiple listener support.***
    - ⇒ hdid | _ORATOR_
      - **sets that listener to have the source with the same id as the listener to be linked to the position of the listener, and is heard by all listeners except the one that is linked**
  - ***int sources: Number of sources to be available***
  - ***int mode:***
    - ⇒ Bits are:
      - **0:3 == units**
        - 0: AtrnINCHES
        - 1: Atrn_FEET_
        - 2: AtrnMMETER
        - 3: AtrnCMETER
        - 4: Atrn_METER
      - **4:5 == resolution**
        - 2: AtrnFIX0: no decimal
        - 3: AtrnFIX1: one decimal
        - 0: AtrnFIX2: two decimals (default)
        - 1: AtrnFIX3: three decimals
      - **6 == shift bit (ignored)**
      - **7:10 == AHM selector**
        - 0: Default (TWS, or TRONAHM environment variable)
        - 1: Unity

    * 2: SLV
    * 3: TWS
    * 4: SDO
    * 5: WLC
    * 14: Test

◊ **11 == head is linked to orator source (ignored)**
◊ **13 == _NOREPLY_, uni-directional client**
◊ **14 == _SHOWSET_ (ignored)**
◊ **15 == _VERBOSE_**
◊ **16 == _CONSOLE_**

- **cre_close(int driver, int head)**
  - ***int driver:***
    - ⇒ Atrn_RESET:
      - ◊ **Same as a call to cre_reset_atron(0);**
    - ⇒ Atrn_CLOS:
      - ◊ **Close a driver**
  - ***int head: Which head to close or ALL_HEADS***
- **cre_end()**
  - ***Same as cre_close(Atron_CLOS, ALL_HEADS)***
- **cre_update_audio ()**
  - ***Causes position calls to cre_locate_source and cre_locate_head to take effect.***
  - ***Also, the engine will be held until the first cre_update_audio call happens***
- **cre_hold_audio(int state)**
  - ***If state is non-zero, then the engine stops, allowing for wavefile synchronization***
  - ***If the state is zero the engine resumes.***
- **cre_define_head(int id, int prm, int pts, const void *data)**
  - ***int id: head id, as given to a previous cre_init call***
  - ***int prm:***
    - ⇒ AtrnAURALocular:
      - ◊ **pts = 1**
      - ◊ **data = float[1] = aural offset in x dir**
    - ⇒ AtrnAURALpinnae:
      - ◊ **pts = 1**
      - ◊ **data = float[1] = aural offset in y dir**
    - ⇒ AtrnAURALcrown:
      - ◊ **pts = 1**
      - ◊ **data = float[1] = aural offset in z dir**
    - ⇒ AtrnAURALoffsets:
      - ◊ **pts = 3**
      - ◊ **data = float[3] = aural offset x, y, & z**
    - ⇒ AtrnINTERAURAL:
      - ◊ **pts = 1**
      - ◊ **data = float[1] = interaural dist**
    - ⇒ AtrnHRTFmodel:
      - ◊ **pts = HRTF Model type one of:**
        - ∗ 0: AtrnMODELfar
        - ∗ 1: AtrnMODELnear
        - ∗ 2: AtrnMODELmixed
        - ∗ 3: AtrnMODELsphere (not supported)
        - ∗ 4: AtrnMODELsynth (not supported)

- ◊ **data = NULL**
- ⇒ AtrnNEARDISTscale:
  - ◊ **pts = 1**
  - ◊ **data = float[1] = near-field exaggeration (1.0 = no exaggeration)**
- ⇒ AtrnHEADgain:
  - ◊ **pts = 1**
  - ◊ **data = float[1] = output gain in dB**
- ⇒ AtrnHRTFresolve:
  - ◊ **pts = filter trim**
  - ◊ **data = NULL**
- ⇒ AtrnAHMname:
  - ◊ **pts = 3**
  - ◊ **data = char[3] = abbreviation for preset**
    - “DFT”
    - “UNO”
    - “SLV”
    - “TWS”
    - “SDO”
    - “WLC”
- ⇒ AtrnHRTFfile:
  - ◊ **pts = strlen(hrtf_file + 1)**
  - ◊ **data = char[pts] = null terminated string of ahm file to load**
- ⇒ AtrnLoadHeadModel: Installs a new model for the source (not supported yet)
  - ◊ **pts = which model in DLL to install**
  - ◊ **data = char* = string of DLL name to load from**
- ⇒ AtrnSetHeadModelAttr:
  - ◊ **pts = attribute in the model to set**
  - ◊ **data = float[1] = data for the attribute**
  - ◊ **Valid Attributes:**

    0) Filter Trim

    1) Model Type

       0) MODEL_FAR_FIELD

       1) MODEL_NEAR_FILED

       2) MODEL_MIXED_FIELD

    2) Near-Field Distance Scale

    3) Head Shadow Scale

    4) ITD Scale

- ⇒ AtrnHeadModelEnable:
  - ◊ **pts = one of the following:**
    - ∗ AtrnEnableGain
    - ∗ AtrnDisableGain
    - ∗ AtrnEnableFilter
    - ∗ AtrnDisableFilter
    - ∗ AtrnEnableDelay
    - ∗ AtrnDisableDelay
  - ◊ **data = NULL**
- ⇒ AtrnPATHgain:
  - ◊ **pts = sizeof(IdAndGain);**
  - ◊ **data = IdAndGain**
    - ∗ id: source id for the path to modify
    - ∗ gain: gain in db for the path or PATH_GAIN_DISABLE_PATH to disable the path

- **cre_locate_head(int id, const float *hloc)**
  - ♦ ***int id: head id as passed to a previous cre_init***
  - ♦ ***const float *hloc = float[6] = { x, y, z, yaw, pitch, roll}***
- **cre_define_source(int id, int prm, int pts, const void *data)**
  - ♦ ***int id: source id, or ALL_SOURCES***
  - ♦ ***int prm:***
    - ⇒ AtrnCHNLinput
      - ◊ **same as cre_select_source(id,pts);**
    - ⇒ AtrnSPATIALoff
      - ◊ **int pts:**
        - ∗ -1: enable all spatialization
          - *data = NULL*
        - ∗ 0: Spatialization turned off, rendered in stereo, source automatically paned
          - *data = NULL*
        - ∗ 1: Spatialization turned off, rendered in mono
          - *data = float[1] = gain in dB or SPATIAL_OFF_AUTO_GAIN to have automatic source level*
        - ∗ 2: Spatialization turned off, rendered in stereo
          - *data = float[2] = { left_gain, right_gain}, where left_gain and right_gain are either the gain in dB or SPATIAL_OFF_AUTO_GAIN to have automatic panning.*

- ⇒ AtrnSPATIALon: Turns on all spatialization
  - ◊ **int pts = 0**
  - ◊ **data = NULL**
- ⇒ AtrnSPRDrolloff:
  - ◊ **int pts = 1**
  - ◊ **data = float[1] = roll off exponent factor, default is 1.0**
- ⇒ AtrnGAINdist:
  - ◊ **int pts = 1**
  - ◊ **data = float[1] = 0 dB gain distance**
- ⇒ AtrnTIMEdelay:
  - ◊ **int pts: time delay type one of: (Note, corrected, absolute and relative are the same right now)**
    - ∗ AtrnTDNone: no IATD for this source, relative reflections
    - ∗ AtrnTDCorrected: absolute corrected for estimated system latency
    - ∗ AtrnTDAbsolute: absolute time delay for all paths
    - ∗ AtrnTDRelative: IATD for direct paths, reflections relative to direct
- ⇒ AtrnRADfields: implemented the same as AtrnRADprofile, but not documented the same in GoldServ.pdf
- ⇒ AtrnRADprofile:
  - ◊ **int pts = number of points in radiation profile**
  - ◊ **data = float[pts] = profile points in dB**
- ⇒ AtrnPROFILEpts:
  - ◊ **int pts = number of radiation points to use**
  - ◊ **data = NULL**
- ⇒ AtrnHEADlink:
  - ◊ **int pts = head id to link the source to**
  - ◊ **data = NULL**
- ⇒ AtrnHEADunlink: Unlinks source from all heads
  - ◊ **int pts = 0**
  - ◊ **data = NULL**
- ⇒ AtrnLoadSrcModel: Installs a new model for the source
  - ◊ **pts = which model in DLL to install**
  - ◊ **data = char* = string of DLL name to load from**
- ⇒ AtrnSetSrcModelAttr:
  - ◊ **pts = attribute in the model to set**
  - ◊ **data = float[1] = data for the attribute**
  - ◊ **Valid Attributes**

    0) Zero dB Distance

    1) Roll Off Exp Factor

    2) Radiation Profile Points

    3) Radiation Profile (can’t be set with AtrnSetSrcModelAttr, use AtrnRADprofile)

4) ExtraTimeDelaySamples

⇒ AtrnSrcModelEnable:

◊ **pts = one of the following:**

∗ AtrnEnableGain

∗ AtrnDisableGain

∗ AtrnEnableFilter

∗ AtrnDisableFilter

∗ AtrnEnableDelay

∗ AtrnDisableDelay

◊ **data = NULL**

⇒ AtrnPATHgain:

◊ **pts = sizeof(IdAndGain);**

◊ **data = IdAndGain**

∗ id: head id for the path to modify

∗ gain: gain in db for the path or PATH_GAIN_DISABLE_PATH to disable the path

- **cre_locate_source(int id, const float *sloc)**
  - ***int id: source_id***
  - ***const float *sloc = float[6] = { x, y, z, yaw, pitch, roll}***
- **cre_amplfy_source(int id, float dB)**
  - ***int id: source id or ALL_SOURCES***
  - ***float dB: input gain in dB***
- **cre_select_source(int id, int input)**
  - ***int id: source id or ALL_SOURCES***
  - ***int input:***
    - ⇒ 0: No source
    - ⇒ Lower 8 bits: Channel to select
    - ⇒ Channel | AtrnROUTEparent: request the channel from the Circuit Manager
- **cre_fetch_value(int prm)**
  - ***int prm:***
    - ⇒ AtrnLSTNRinit: number of listeners initialized
    - ⇒ AtrnASRCinnit: number of sources initialized
    - ⇒ AtrnSYSunits: current units
- **cre_define_medium (int volm, int prm, int pts, void *data)**
  - ***int volm: medium id***
  - ***int prm:***
    - ⇒ AtrnROLLOFF:
      - ◊ **pts = 1**
      - ◊ **data = float[1] = Roll off exponential, default is 0.8**
    - ⇒ AtrnLoadEnvModel:
      - ◊ **pts = which model in DLL to install**
      - ◊ **data = char* = string of DLL name to load from**
    - ⇒ AtrnSetEnvModelAttr:
      - ◊ **pts = attribute in the model to set**
      - ◊ **data = float[1] = data for the attribute**
      - ◊ **Valid Attributes**

        0) Roll Of Exponential

        1) Absorption Distance (in current Units)

        2) Speed Of Sound (in current Units/second)

    - ⇒ AtrnEnvModelEnable:

      - ◊ **pts = one of the following:**

        - ∗ AtrnEnableGain

        - ∗ AtrnDisableGain

* AtrnEnableFilter
* AtrnDisableFilter
* AtrnEnableDelay
* AtrnDisableDelay

        - ◊ **data = NULL**

- **cre_test_atron(int verbose)**
    - ♦ ***int verbose: 1 to print message to the screen***
- **cre_reset_atron(int status): resets the server**
    - ♦ ***int status: ignored***
- **cre_client_nap(unsigned long msecs)**
    - ♦ ***Sleep(msecs);***
- **wavFt* cre_open_wave (const char *wavefile, int mode)**
    - ♦ ***char *wavefile: name of wavefile to load, either absolute or relative to the WAVEFORM environment variable***
    - ♦ ***mode: ignored***
    - ♦ ***Note: wavefiles opened BEFORE a cre_init will persist after a cre_close. Wavfiles opened AFTER a cre_init, will be cleaned up on a cre_close***
- **cre_ctrl_wave (int src_id, wavFt *wave, int cmd, void *data)**
    - ♦ ***int src_id: id of the source to link a wavefile to***
    - ♦ ***wavFt *wave: pointer to wave structure returned by a previous cre_open_wave***
    - ♦ ***int cmd:***
        - ⇒ WaveCTRL_PSET: Set wavefile position
            - ◊ **data = unsigned int = wavefile position**
        - ⇒ WaveCTRL_RWND: Set wavefile position to 0
        - ⇒ WaveCTRL_STRT: Rewind and play
        - ⇒ WaveCTRL_PLAY: Play non-looped at current position
        - ⇒ WaveCTRL_LOOP: Play looped at current position
        - ⇒ WaveCTRL_STOP: Stop at current position
        - ⇒ WaveCTRL_NOLP: Clear the loop flag
        - ⇒ WaveCTRL_WDAT: Fill in values in wavFt struct if Client/Server
        - ⇒ WaveCTRL_STAT: Get stats on the current wavefile
            - ◊ **bit0: wavFt is busy**
            - ◊ **bit1: SRC is busy**
            - ◊ **bit2: SRC is playing this wavFt**
            - ◊ **bit3: 0, unused**
            - ◊ **bits4-31: number of samples to play till the end**
- **cre_close_wave (wavFt *wave)**
    - ♦ ***wavFt *wave: pointer to wave structure returned by a previous cre_open_wave***
- **cre_define_router (int id, int prm, int pts, const void *data)**
    - ♦ ***int id: head_id of the listener whose voice will be patched***
    - ♦ ***int prm***
        - ⇒ AtrnDELpatch: Remove one patch from the listener's voice to the give output circuit
            - ◊ **pts = output circuit**
            - ◊ **data = NULL**

- ⇒ AtrnADDpatch: Adds one patch from the listener's voice to the given output circuit
  - ◊ **pts = output circuit**
  - ◊ **data = NULL**
- ⇒ AtrnDELpatches: Deletes multiple patches from the listener's voice to a list of output circuits
  - ◊ **pts = number of circuits**
  - ◊ **data = int [pts] = output circuits**
- ⇒ AtrnADDpatches: Adds multiple patches from the listener's voice to a list of output circuits
  - ◊ **pts = number of circuits**
  - ◊ **data = int [pts] = output circuits**
- ⇒ AtrnSETrouting: Deletes any previous patches from the given listener, and then adds patches from the listener's voice to the list of output circuits
  - ◊ **pts = number of circuits**
  - ◊ **data = int [pts] = output circuits**
- ⇒ AtrnCHNLactive: Controls Push To Talk for the listener's voice. Tries to acquire exclusive control of the circuits that the listener is patched to.
  - ◊ **pts = BOOL = push to talk**
  - ◊ **data = NULL**
  - ◊ **return = BOOL = 1 if circuit was available and acquired, 0 if the circuit is busy**