- HIDL_FETCH_ICameraProvider（）
  - get_number_of_cameras /* 先调到camxhal3entry.cpp里的这个函数，然后在实例化Dispatch对象的时候static Dispatch g_dispatchHAL3(&g_jumpTableHAL3);将g_jumpTableHAL3结构体指针注册上去，从而调到camxhal3.cpp里对应的函数 */
    - pHAL3->get_number_of_cameras()
      - HwEnvironment::GetInstance() /* 获取Instance的过程中触发HwEnvironment的构造函数操作*/
        - HwEnvironment::Initialize()
          - SettingsManager::Create(NULL)
            - pSettingsManager->Initialize(pStaticSettings)
              - OverrideSettingsFile::Create()
                - pOverrideSettingsFile->Initialize() /* 打开vendor/etc/camera/camxoverridesettings.txt文件并逐行解析，将每一个配置项及其value值做相应处理后put到哈希表里暂存起来待后续使用*/
              - InitializeDefaultSettings() /* 给m_pStaticSettings各个参数初始化一个默认值*/
              - InitializeDefaultDebugSettings() /* 给m_pStaticSettings里调试相关的参数初始化一个默认值，例如统计开关、ImageDump开关等*/
              - LoadOverrideSettings(m_pOverrideSettingsStore) /* 将之前从camxoverridesettings.txt里解析出来的配置值载入到m_pStaticSettings并替换默认值*/
              - UpdateLogSettings() /* 将m_pStaticSettings里log打印级别相关的配置更新给g_logInfo结构体，其中也包含了离线log的相关配置*/
          - GetExternalComponent() /* 拿到m_externalComponent指针待后续使用*/
          - pStaticSettingsManager->GetStaticSettings() /* 拿到m_pStaticSettings指针 */
          - CSLInitialize(&params) /* 触发CSLModeManager的构造函数*/
            - CSLModeManager::CSLModeManager(const CSLInitializeParams* pInitParams) /* 这里注册了g_CSLjumpTableHW结构体函数指针方便跳转到对应的camxcslhw.cpp流程中*/
            - pJumpTable->CSLInitialize() /* 即CSLInitializeHW()函数 */
              - CSLHwEnumerateAndAddCSLHwDevice(CSLInternalHwVideodevice, CAM_VNODE_DEVICE_TYPE) /* 查找并获取dev/videoX设备，该节点对应kernel部分的Request Manager */
              - CSLHwEnumerateAndAddCSLHwDevice(CSLInternalHwVideoSubdevice, CAM_CPAS_DEVICE_TYPE) /* 查找并获取dev/v4l-subdevX caps设备*/
              - CSLHwEnumerateAndAddCSLHwDevice(CSLInternalHwVideoSubdeviceAll, 0) /* 查找并获取dev/v4l-subdevX其它设备，例如Sensor/IFE/IPE/Flash等*/
              - CSLHwInstanceSetState(CSLHwValidState);
          - 查找vendor/lib64/hw路径下的com.qti.chi.override.so文件并将文件名存入到chiOverrideSoFileName[0]里
          - 查找vendor/lib64/camera/components路径下名称中含有"node"的so文件并将文件名存入到soFilesName二维数组里
          - 查找vendor/lib64/camera/components路径下名称中含有"stats"的so文件并将文件名存入到soFilesName二维数组里
          - 查找vendor/lib64/camera/components路径下名称中含有"hvx"的so文件并将文件名存入到soFilesName二维数组里
          - ProbeChiComponents(pExternalComponent, &m_numExternalComponent)
            - 遍历"node"命名的so并将"ChiNodeEntry"命名的函数指针注册到pExternalComponentInfo里，如果有自己定义的vendortag也一并记录下来
            - 遍历"stats"命名的so并将各自的入口函数指针注册到pExternalComponentInfo里，如果有自己定义的vendortag也一并记录下来
            - 遍历"hvx"命名的so并将"ChiISPHVXAlgorithmEntry"命名的函数指针注册到pExternalComponentInfo里，如果有自己定义的vendortag也一并记录下来
            - GetComponentTag(pQueryVendorTag) /* 拿到"chi_hal_query_vendertag"命名的函数指针并将chi里高通及用户自定义的vendortag记录下来 */
          - HwEnvironment::GetInstance()->GetStaticSettings() /* 拿到m_pStaticSettings配置值*/
          - 查找vendor/lib64/hw路径下的com.qti.chi.override.so文件并将文件名存入到moduleFileName[0]里
          - 打开com.qti.chi.override.so链接库并找到"chi_hal_override_entry"命名的入口函数指针，传入m_ChiAppCallbacks回调结构体
            - ExtensionModule::GetInstance()
              - 触发ExtensionModule::ExtensionModule()构造函数
                - 打开vendor/lib64/hw/camera.qcom.so链接库并找到"ChiEntry"命名的入口函数，将结构体g_chiContextOps里的指针函数一一注册
                - 将高通及用户自定义的vendortag记录到VendortagManager里
            - 将chi里的函数接口一一注册到m_ChiAppCallbacks结构体里
          - m_ChiAppCallbacks.chi_get_num_cameras(&m_numFwCameras, &m_numLogicalCameras)
            - pExtensionModule->GetNumCameras(numFwCameras, numLogicalCameras)
              - SortCameras() /* 对LogicalCameraConfiguration结构体数组进行排序*/
              - EnumerateCameras()
              - FillLogicalCameraCaps()
        - HAL3Module::GetInstance() /* 获取Instance的过程中触发HAL3Module的构造函数操作*/
          - HAL3Module::HAL3Module()
      - HAL3Module::GetInstance()->GetNumCameras() /* 返回m_numFwCameras变量值，该值最初是在调用m_ChiAppCallbacks.chi_get_num_cameras函数时传入的，意义是展示给framework底层有多少个逻辑cameraID*/