Man Machine Interface

Agenda

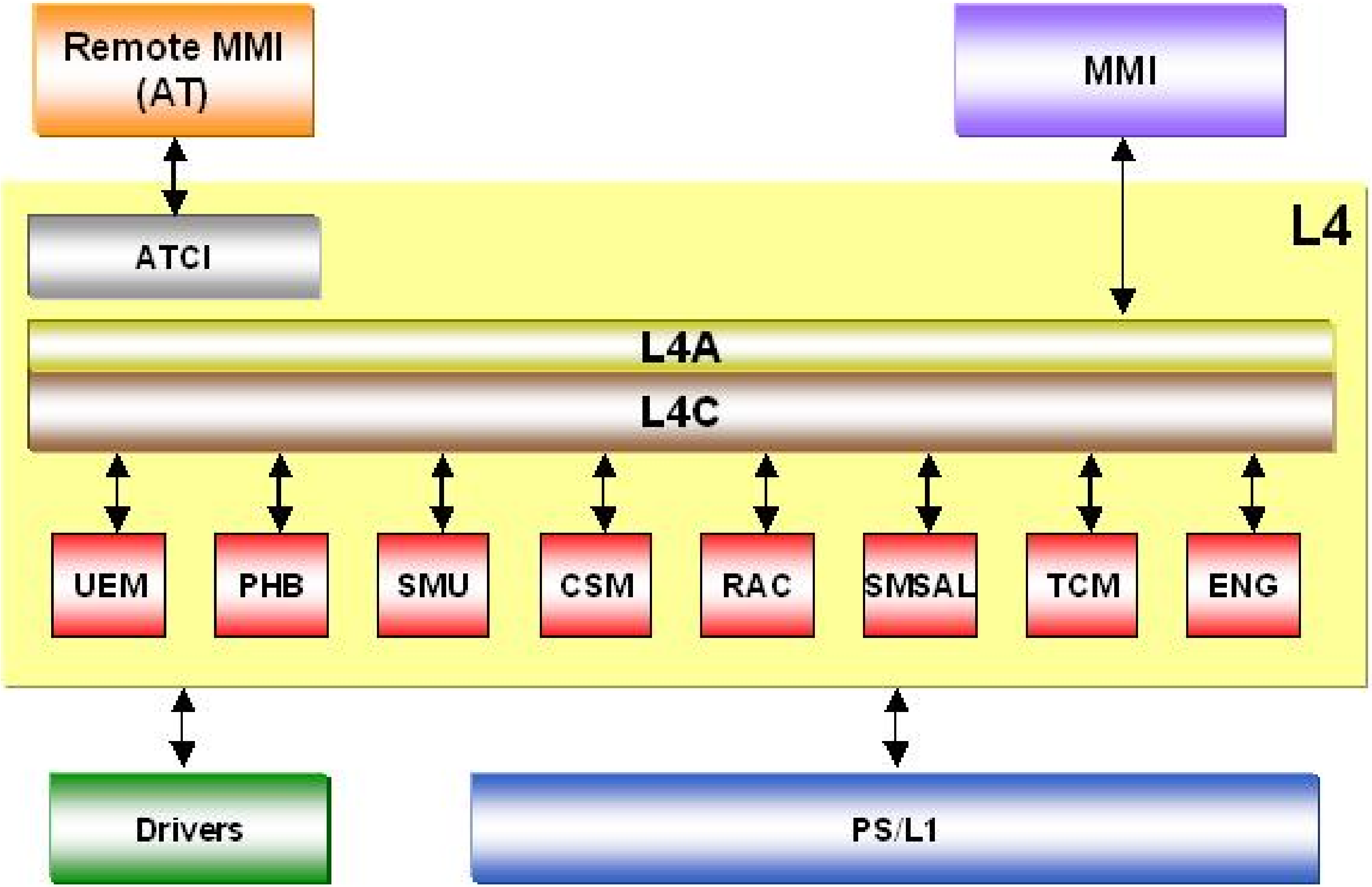
- MTK Software Architecture
- MTK MMI Architecture
- Example to Write an Application
- Third Party Software
- Tool
- Q&A

MTK Software Architecture

MTK Software Architecture

- Software Architecture
- KAL and OSL
- Date Type
- Task Management

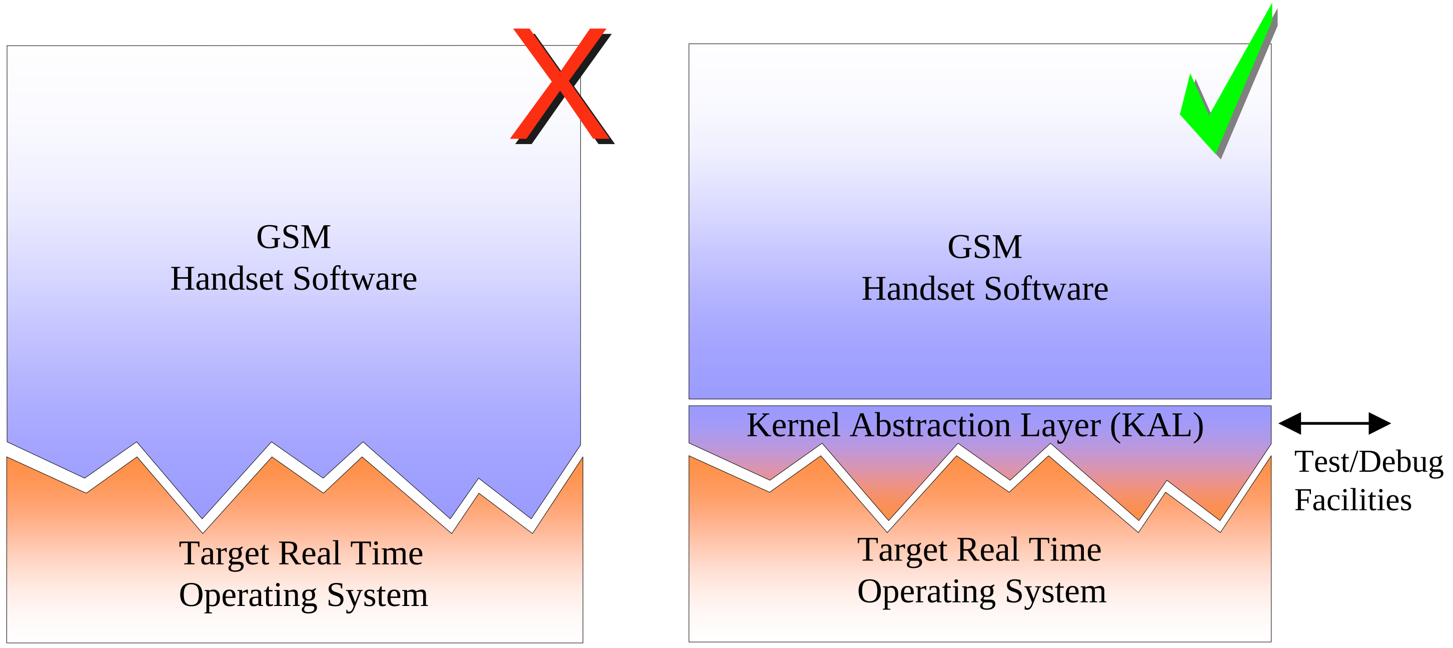
Software Architecture



Software Architecture – abbreviations

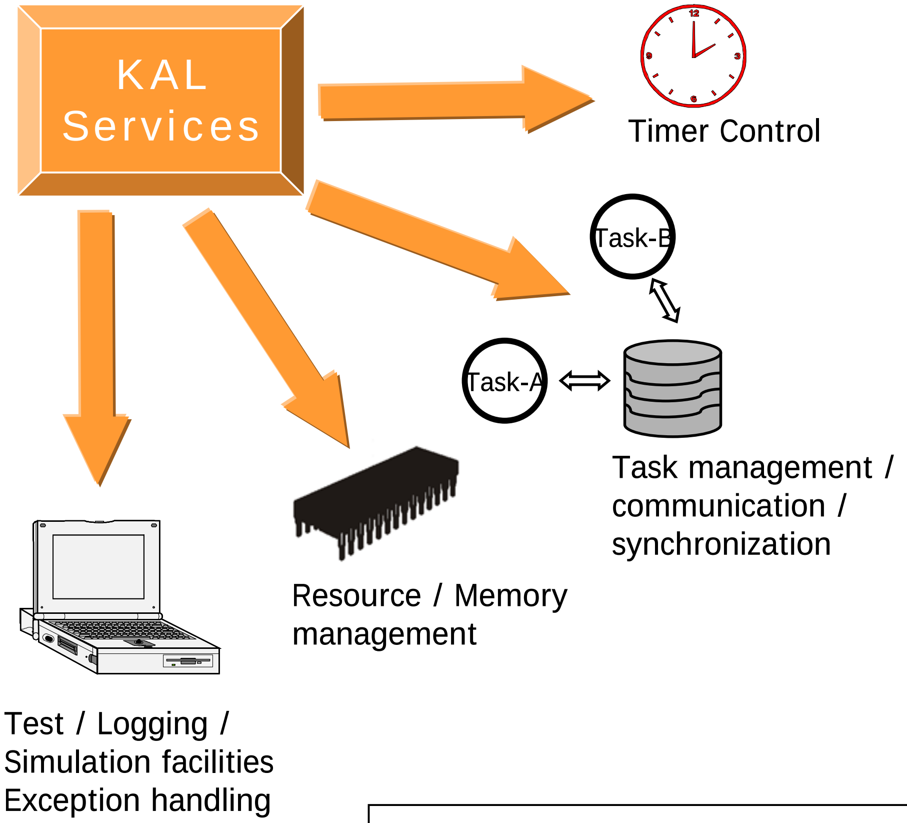
- RMI: Remote MMI, i.e., PC side, which uses AT commands to communicate with Protocol stack.
- L4: The adaptation layer between MMI/AT and protocol stack.
- L4A: Layer 4 Adaptation to translate primitives sent from upper layers to function calls.
- L4C: Layer 4 Controller, coordinates all L4 modules to serve upper layers.
- ATCI: AT Command Interpreter.
- UEM: User Equipments module used to abstract basic device drivers like keypad, LED, GPIO.
- PHB: Phone Book management.
- SMU: Security Management (SIM, STK).
- CSM: Call Service Management (bearer capability handling, CSD/FAX service, CC, SS).
- RAC: Registration Access Control (GSM/GPRS registration management, PLMN list/selection, RSSI report)
- SMSAL: SMS Application Layer (message storage, MO/MT messages, CB).
- TCM: Terminal Context Management (PDP context profiles, context activate/deactivate, relay of packet data), interface to PPP/TCPIP/SNDCP.
- ENG: Engineer Mode to log information.

KAL (Kernel Abstraction Layer)



- Portability
- Common design philosophy
- Test/Debug facilities
- Easier code integration

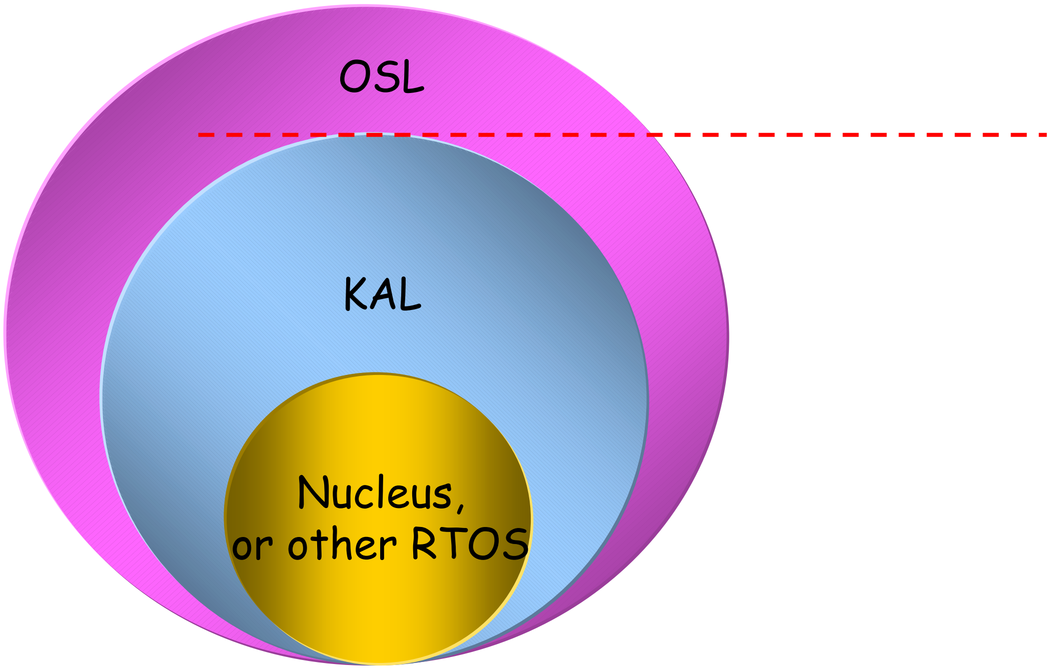
KAL Services


KAL Services
Timer Control
Task-B
Task-A
Task management / communication / synchronization
Resource / Memory management
Test / Logging / Simulation facilities
Exception handling

- Reference:
  - KAL_ProgrammingGuide_20041005.pdf
  - SystemServiceUserManual_20050527.pdf

OSL

N.B. to ensure all programs within MMI task can run correctly on PC, use of OSL APIs is MUST.



PC simulator – simulate the OSL services on Win32 to facilitate development and debugging MMI task on PC.

Data Types

General data types:

- Kal_non_specific_general_types.h

KAL specific data types and functions' prototypes:

- Kal_release.h

OSL specific data types used within MMI Tasks:

- PixtelDataTypes.h

Task Management & Identification

module_type and task_indx_type

- Defined in "Stack_config.h"
- Used to define index of all modules and all tasks

mod_task_g [RPS_TOTAL_STACK_MODULES]

- Defined in "syscomp_config.c"
- Used to map MODULE-ID to TASK-ID

sys_comp_config_tbl [RPS_TOTAL_STACK_TASKS + 1]

- Defined in "syscomp_config.c"
- Used to define information of all tasks, e.g., task's name, task queue's name, priority, size of external/internal queue, task creation function, whether to use internal ram.

custom_comp_config_tbl [MAX_CUSTOM_TASKS]

- Used for customer defined modules or tasks.

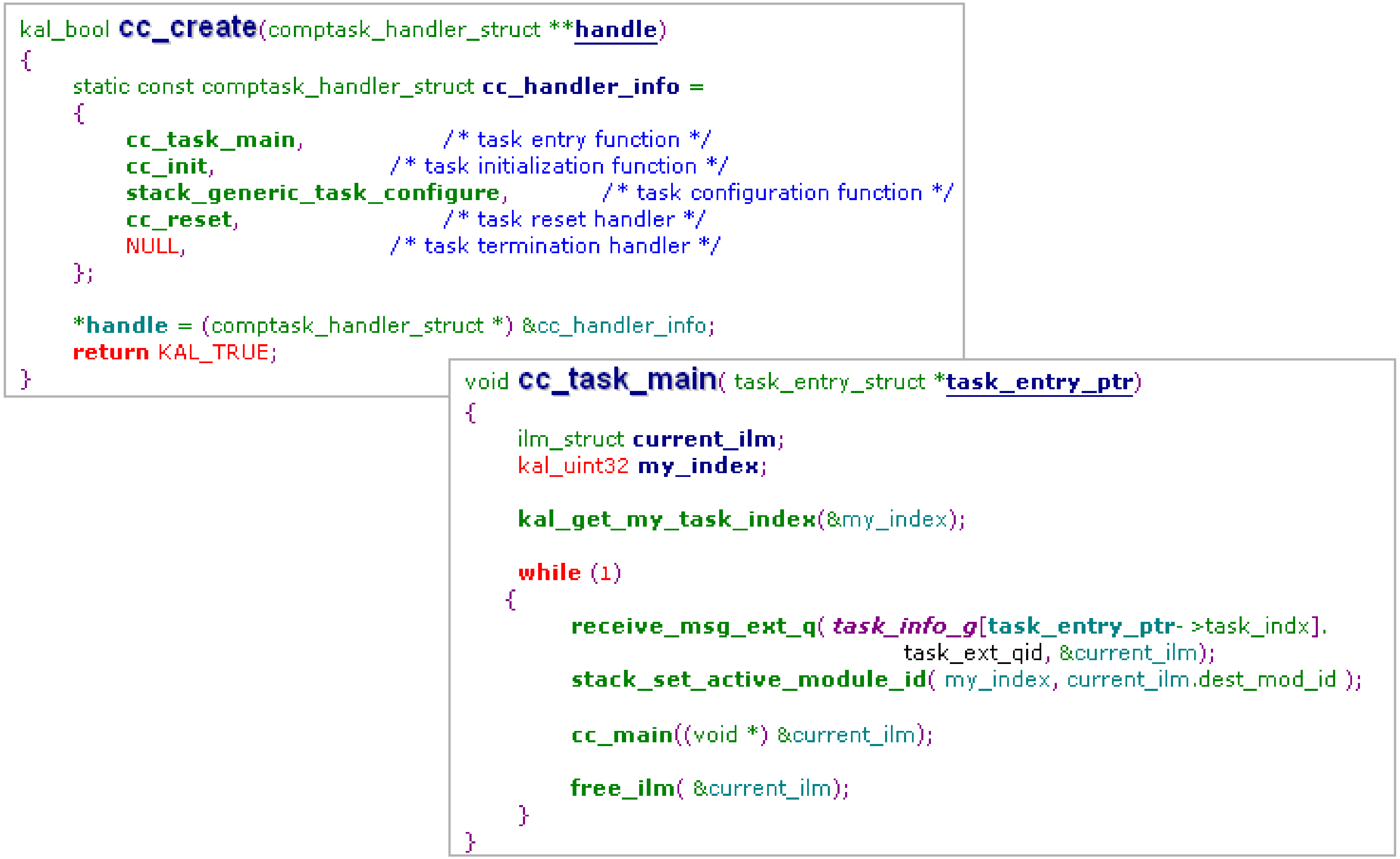
task_info_g [RPS_TOTAL_STACK_TASKS + 1]

- Global array containing component task information, which will be filled in while calling stack_init_comp_info().

module_info_g [MAX_MULTIMOD_TASK_NUM]

- Global array containing component task information, which will be filled in while calling stack_init_module_info().

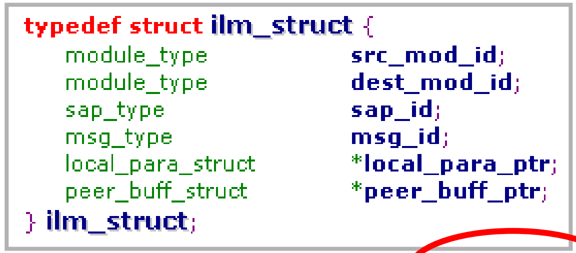
Task Routines



```
kal_bool cc_create(comptask_handler_struct **handle)
{
    static const comptask_handler_struct cc_handler_info =
    {
        cc_task_main,                   /* task entry function */
        cc_init,                    /* task initialization function */
        stack_generic_task_configure,           /* task configuration function */
        cc_reset,                       /* task reset handler */
        NULL,                       /* task termination handler */
    };

    *handle = (comptask_handler_struct *) &cc_handler_info;
    return KAL_TRUE;
}
```

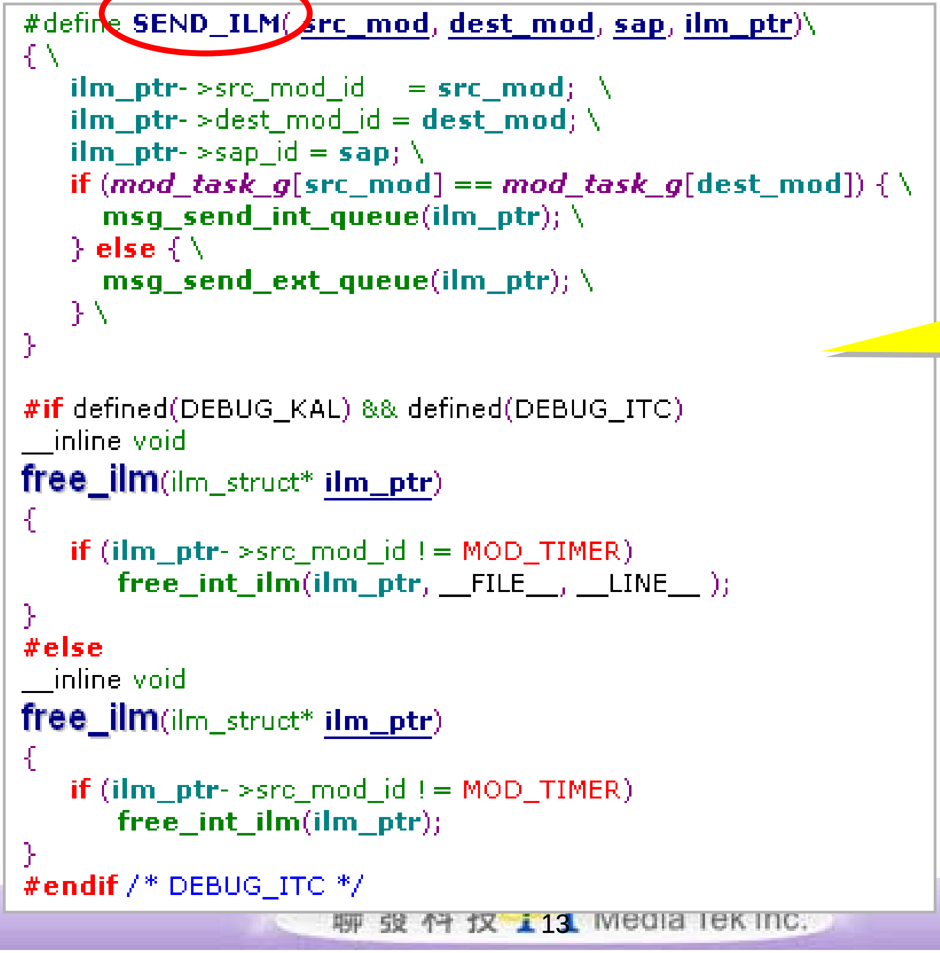
```
void cc_task_main( task_entry_struct *task_entry_ptr)
{
    ilm_struct current_ilm;
    kal_uint32 my_index;

    kal_get_my_task_index(&my_index);

    while (1)
   {
        receive_msg_ext_q( task_info_g[task_entry_ptr- >task_indx].
                                  task_ext_qid, &current_ilm);
        stack_set_active_module_id( my_index, current_ilm.dest_mod_id );

        cc_main((void *) &current_ilm);

        free_ilm( &current_ilm);
    }
}
```

Task Communication



```
typedef struct ilm_struct {
    module_type          src_mod_id;
    module_type          dest_mod_id;
    sap_type             sap_id;
    msg_type             msg_id;
    local_para_struct    *local_para_ptr;
    peer_buff_struct     *peer_buff_ptr;
} ilm_struct;
```

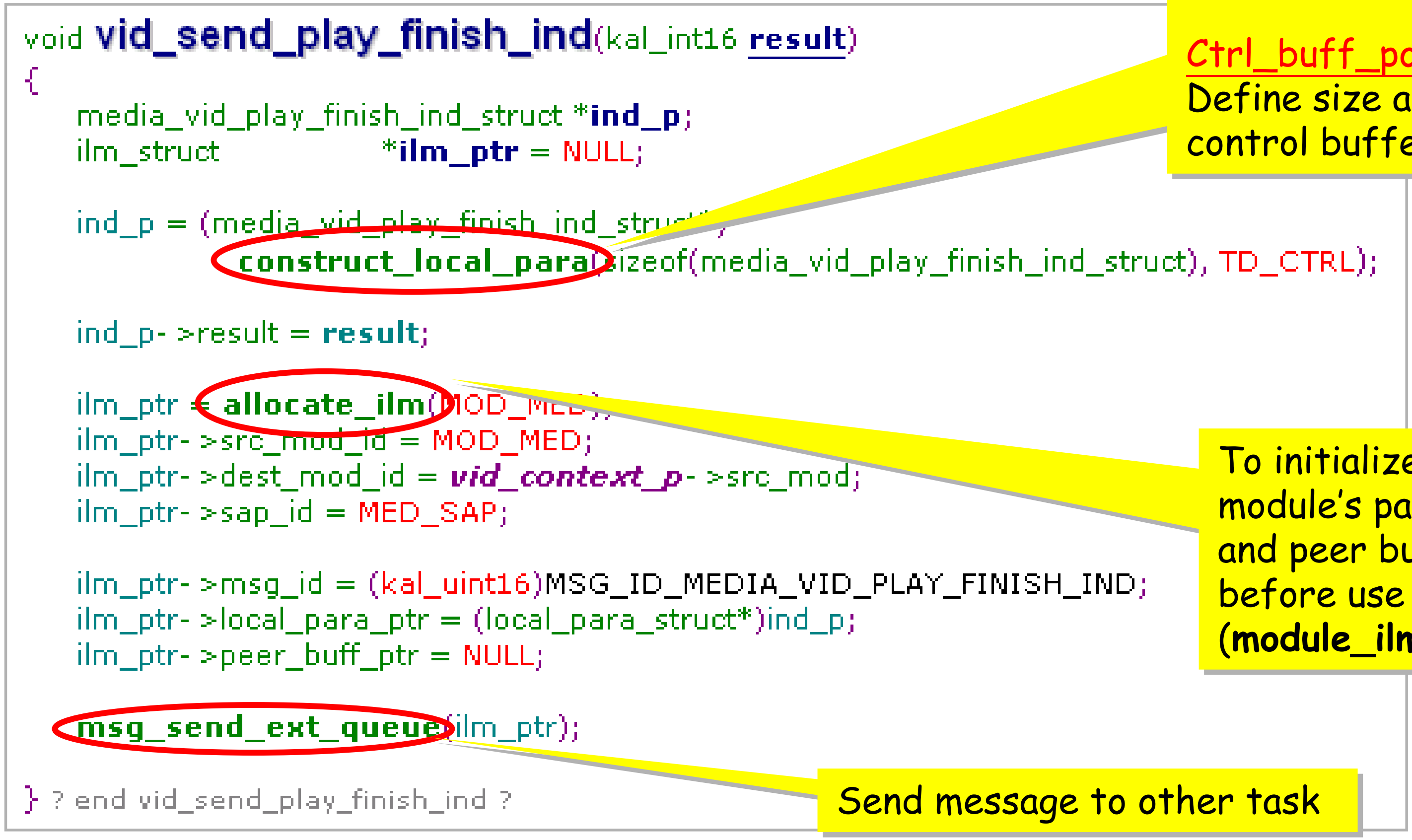
App_ltlcom.h
Data structure of massage used for inter-layer communication



```
#define SEND_ILM( src_mod, dest_mod, sap, ilm_ptr)\
{ \
    ilm_ptr- >src_mod_id    = src_mod;  \
    ilm_ptr- >dest_mod_id = dest_mod; \
    ilm_ptr- >sap_id = sap; \
    if (mod_task_g[src_mod] == mod_task_g[dest_mod]) { \
      msg_send_int_queue(ilm_ptr); \
    } else { \
      msg_send_ext_queue(ilm_ptr); \
    } \
}

#if defined(DEBUG_KAL) && defined(DEBUG_ITC)
__inline void
free_ilm(ilm_struct* ilm_ptr)
{
    if (ilm_ptr- >src_mod_id != MOD_TIMER)
        free_int_ilm(ilm_ptr, __FILE__, __LINE__ );
}
#else
__inline void
free_ilm(ilm_struct* ilm_ptr)
{
    if (ilm_ptr- >src_mod_id != MOD_TIMER)
        free_int_ilm(ilm_ptr);
}
#endif /* DEBUG_ITC */
```

Stack_ltlcom.h
Macro and API used to send/free messages

Task Communication – example 1



```
void vid_send_play_finish_ind(kal_int16 result)
{
    media_vid_play_finish_ind_struct *ind_p;
    ilm_struct                 *ilm_ptr = NULL;

    ind_p = (media_vid_play_finish_ind_struct*)
             construct_local_para(sizeof(media_vid_play_finish_ind_struct), TD_CTRL);

    ind_p- >result = result;

    ilm_ptr = allocate_ilm(MOD_MED);
    ilm_ptr- >src_mod_id = MOD_MED;
    ilm_ptr- >dest_mod_id = vid_context_p- >src_mod;
    ilm_ptr- >sap_id = MED_SAP;

    ilm_ptr- >msg_id = (kal_uint16)MSG_ID_MEDIA_VID_PLAY_FINISH_IND;
    ilm_ptr- >local_para_ptr = (local_para_struct*)ind_p;
    ilm_ptr- >peer_buff_ptr = NULL;

    msg_send_ext_queue(ilm_ptr);

} ? end vid_send_play_finish_ind ?
```

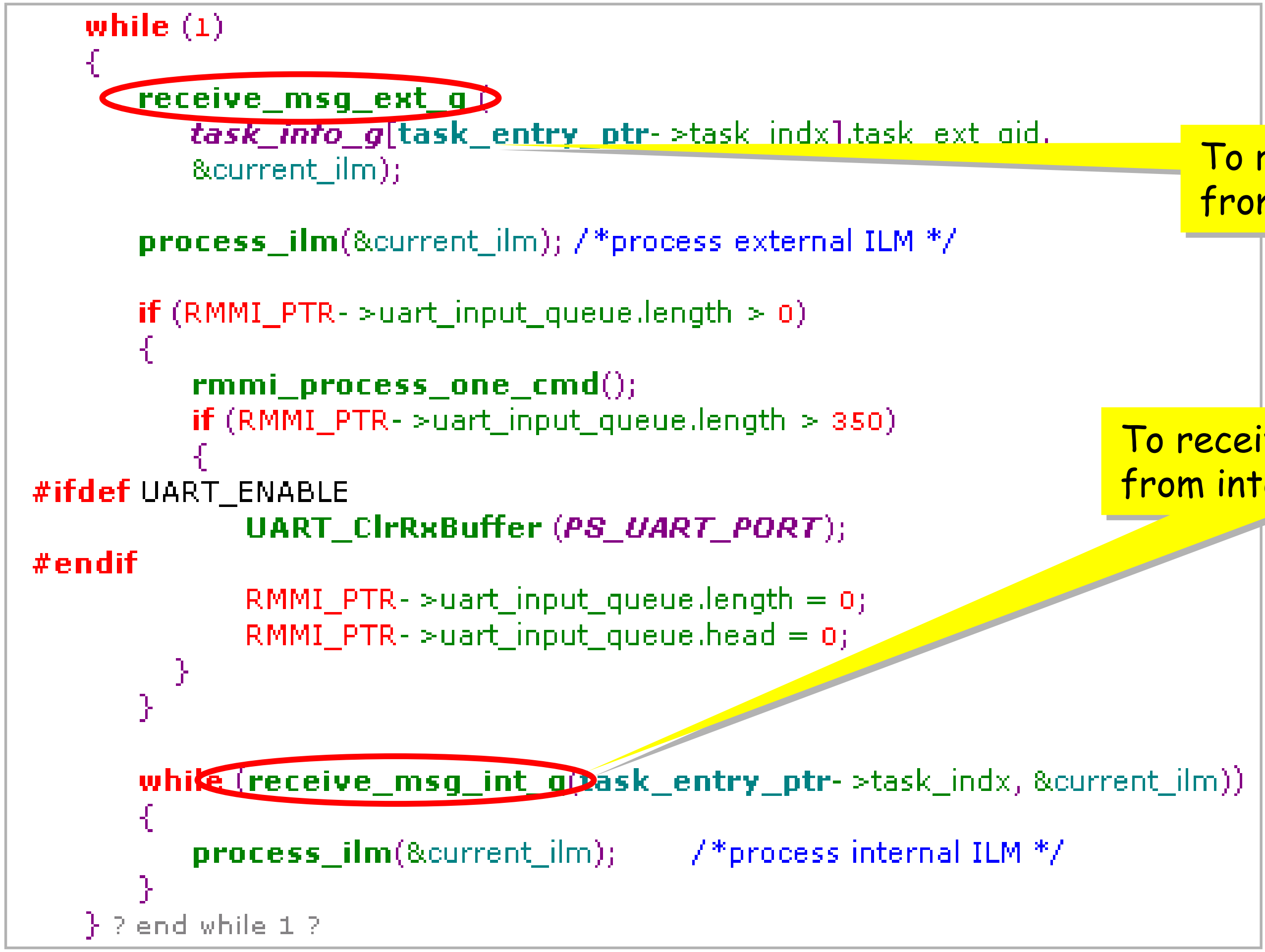
To allocate memory from shared memory pool.

Ctrl_buff_pool.h
Define size and number of control buffer (memory pool)

To initialize specific module's parameter pointer and peer buffer pointer before use it. (**module_ilm_g**[module_id])

Send message to other task

Task Communication – example 2



```
    while (1)
    {
      receive_msg_ext_q (
         task_info_g[task_entry_ptr- >task_indx].task_ext_qid,
         &current_ilm);

      process_ilm(&current_ilm); /*process external ILM */

      if (RMMI_PTR- >uart_input_queue.length > 0)
      {
         rmmi_process_one_cmd();
         if (RMMI_PTR- >uart_input_queue.length > 350)
         {
#ifdef UART_ENABLE
              UART_ClrRxBuffer (PS_UART_PORT);
#endif
              RMMI_PTR- >uart_input_queue.length = 0;
              RMMI_PTR- >uart_input_queue.head = 0;
        }
      }

      while (receive_msg_int_q(task_entry_ptr- >task_indx, &current_ilm))
      {
         process_ilm(&current_ilm);      /*process internal ILM */
      }
    } ? end while 1 ?
```

To receive message from external queue

To receive message from internal queue

QueueGprot.h
Usage of OSL send/receive internal/external msg.

MTK MMI Architecture

MTK MMI Architecture

- MMI Task structure
- MMI and L4 Communication
- MMI Architecture
  - Framework
    - Provides OS abstraction
    - Event Handlers
    - History Manager
    - NVRAM Access
    - File System Management
  - UI, Resource
- MMI Directories

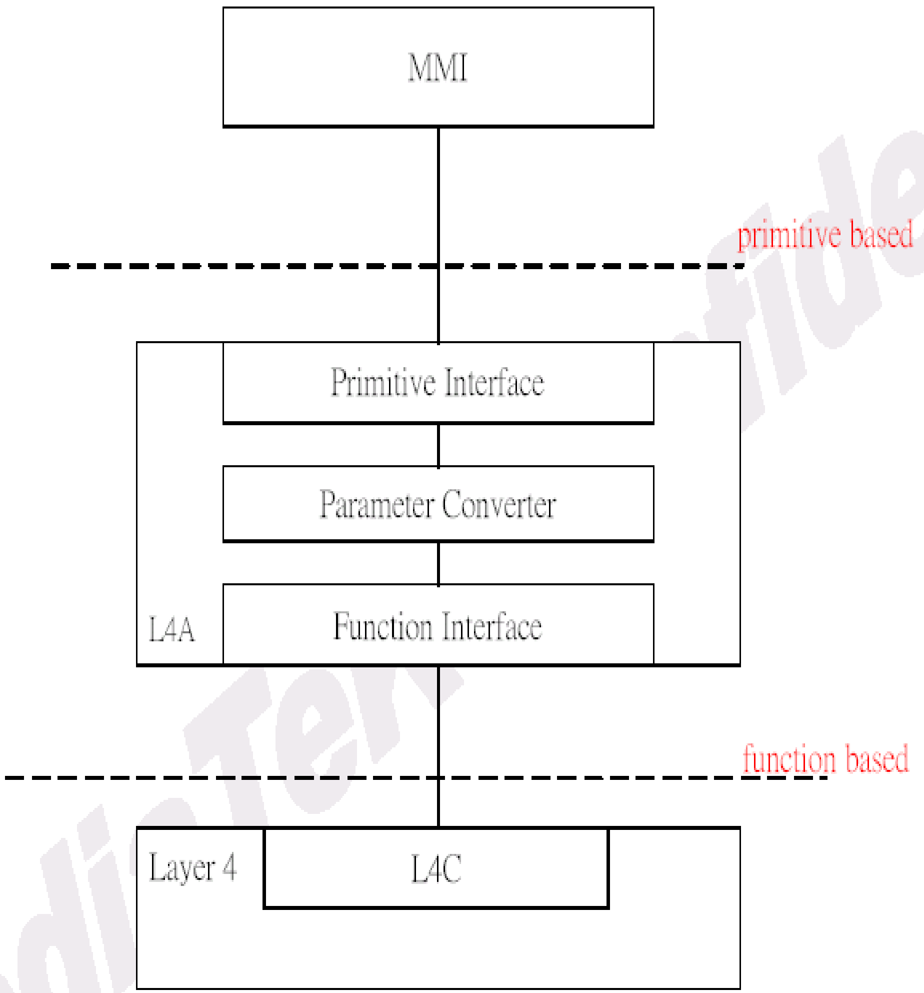
Task structure

- Task struct(sys_comp_config_tbl):

```
typedef struct {
   kal_char                    *comp_name_ptr;
   kal_char                    *comp_qname_ptr;
   kal_uint32                  comp_priority; //3-255
   kal_uint16                  comp_stack_size;
   kal_uint8                   comp_ext_qsize;
   kal_uint8                   comp_int_qsize;
   kal_create_func_ptr         comp_create_func;
   kal_bool                    comp_internal_ram_stack;
} comptask_info_struct;
```

MMI Task
"MMI"
“MMI Q”
TASK_PRIORITY_MMI
4096
30
100
mmi_create
KAL_FALSE

Layer 4 Adapter



Example:

MOD_MMI--->MOD_L4C

mmi_frm_sms_send_message( )

PRT_MSG_ID_MMI_SMS_SEND_MSG_REQ

L4a_callback.c

l4a_recv_msg_ft[MSG_ID_MMI_MESSAGE_SUM]

_call_MSG_ID_MMI_SMS_SEND_MSG_REQ_( )

MOD_L4C--->MOD_SMSAL

l4c_sms_exe_post_msg_req( )

MSG_ID_L4CSMSAL_SEND_REQ

MMI and L4 Communication(1/3)

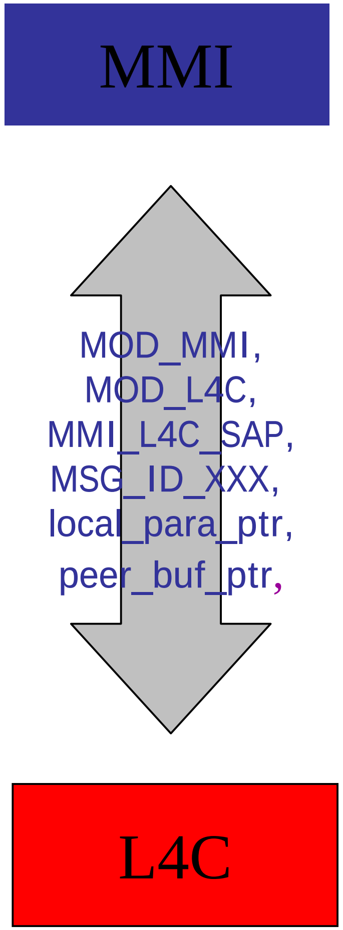
- How To Communicate
  - Send/Receive messages thru the message Queue.

```
#define OslMsgSendExtQueue        msg_send_ext_queue
#define OslReceiveMsgExtQ         receive_msg_ext_q
SetProtocolEventHandler(FuncCB, msg_id);
```

Queue

- Communication Data

```
typedef struct ilm_struct {
    oslModuleType    oslSrcId; // Source module ID.
    oslModuleType    oslDestId; // Destination module ID.
    oslMsgType        oslSapId; // service access point.
    oslMsgType        oslMsgId; // message name ID.
    oslParaType        *oslDataPtr; //local parameter buffer
    oslPeerParaPtr      *oslPeerBuffPtr; //peer buffer pointer
} ilm_struct;
```



MMI and L4 Communication(2/3)

- How to listen a message from MMI Queue:
    - From task create and entry a message loop.
        - *OslReadCircularQ*(&Message);
        - *OslReceiveMsgExtQ*(mmi_qid, &mmi_message);
- How to write a message to MMI Circular Queue:
    - When NVRAM receive other messages.
        - *OslWriteCircularQ*(&ilm_ptr);

MMI and L4 Communication(3/3)

- How to receive a message from L4C:
    - Register a response message callback.
        - *SetProtocolEventHandler*(FuncCB, msg_id);

- How to send a message to L4C:
    - Step1: Construct a local parameter buffer.
    - Step2: Assign required values into local parameter buffer.
    - Step3: Send out the message to the L4C module.
        - *OslMsgSendExtQueue*(&Message);

Message Information(1/3)

- Message Info = Header info + Data info
  - Local parameter Header info:

    -
      ```
      #define LOCAL_PARA_HDR \
          kal_uint8       ref_count; \
          kal_uint16      msg_len;
      ```
  - peer buffer parameter Header info :

    -
      ```
      #define PEER_BUFF_HDR \
          kal_uint16      pdu_len; \
          kal_uint8       ref_count; \
          kal_uint8       pb_resvered; \
          kal_uint16      free_header_space; \
          kal_uint16      free_tail_space;
      ```

Message Information (2/3)

- Local parameter:
  - Header info + Data info:

    Ex: typedef struct {

    ```
      LOCAL_PARA_HDR
      kal_uint8  volume_type;
      kal_uint8  volume_level;
    } mmi_eq_set_volume_req_struct;
    ```
- How To Create Local Parameter:
  - Dynamic to allocate memory buffer:
    - *OslConstructDataPtr*(sizeof(mmi_at_alarm_query_res_req_struct);
- When to Free Local Parameter:
  - While L4 receive the information, after finishing to process the message, L4 task will automatically free this buffer.
    - *OslFreeDataPtr*(sizeof(mmi_at_alarm_query_res_req_struct);

Message Information (3/3)

- Peer buffer parameter:
  - Header info + Data info

    Ex: typedef struct {

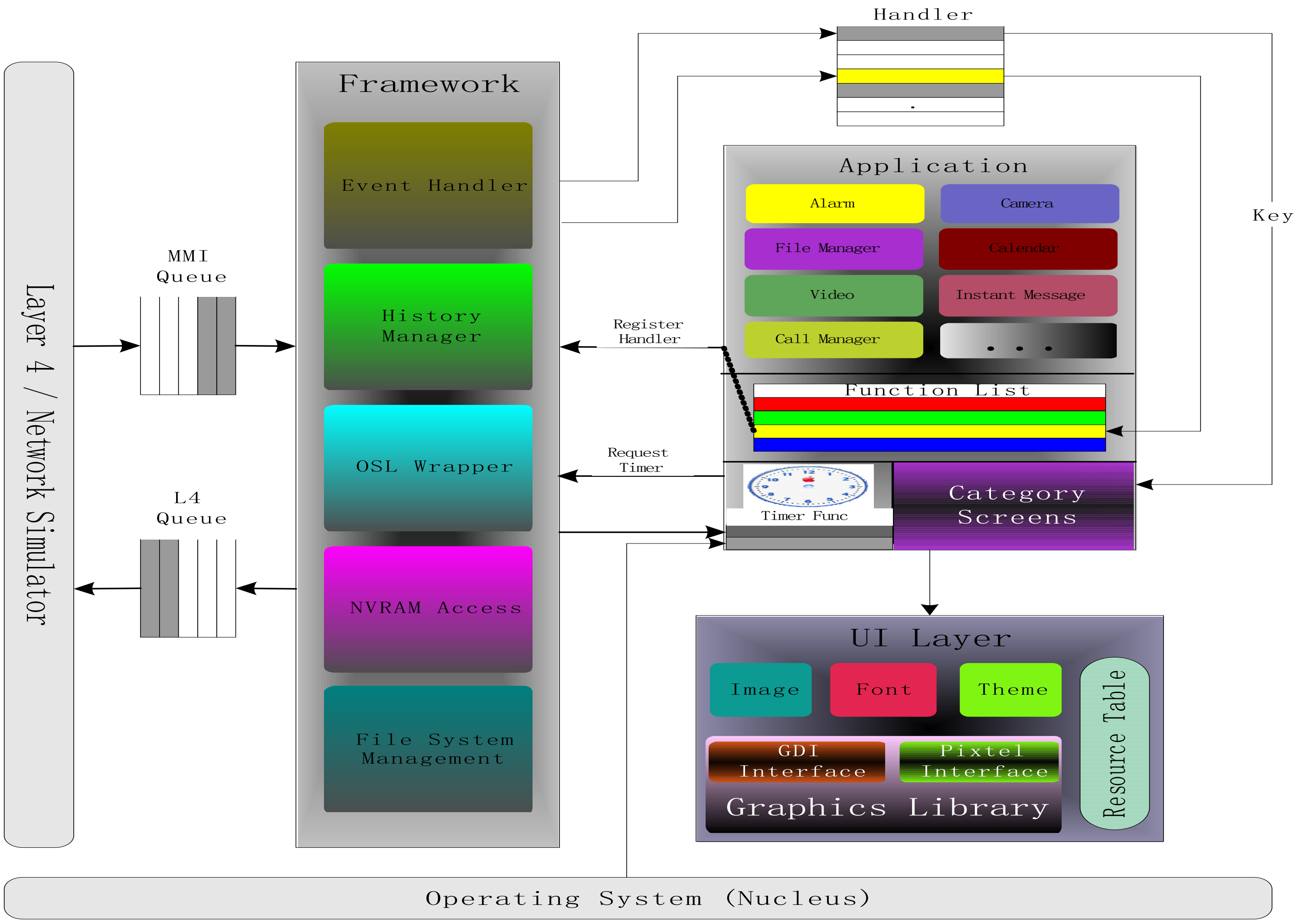
    ```
     PEER_BUFF_HDR
    void *ptr;
    } mmi_example;
    ```
- How To Create Peer Buffer Parameter:
  - Dynamic to allocate memory buffer:
    - Ps: The MMI did not use this buffer to communicate with L4.
    - *construct_peer_buff*(pdu_len, header_len, tail_len, direction);
- When will Free Peer Buffer:
  - While receive the information, after finishing to process the message, L4 task will automatically free this buffer.
    - *free_peer_buff*(peer_buff);

Example – Set Volume

- Set a volume request:

```
void SetVolumeLevelReq(volume_type_enum volume_type,U8 volume_level)
{
    MYQUEUE Message;
    mmi_eq_set_volume_req_struct *setVolumeLevelReq;
    Message.oslMsgId = MSG_ID_MMI_EQ_SET_VOLUME_REQ;
    //Message ID, reference the l4a.h file
    setVolumeLevelReq = OslConstructDataPtr(sizeof(mmi_eq_set_volume_req_struct));
    //Create local parameter buffer
    setVolumeLevelReq->volume_type = volume_type;
    setVolumeLevelReq->volume_level = volume_level;
    Message.oslDataPtr = (oslParaType *)setVolumeLevelReq; //Local parameter buffer
    Message.oslPeerBuffPtr= NULL; //Peer parameter buffer
    Message.oslSrcId=MOD_MMI; //Send from Source module
    Message.oslDestId=MOD_L4C; //Send to destination  module
    OslMsgSendExtQueue(&Message); //Send to L4 task
}
```

MMI Architecture



Core Functionality Provided by Framework

- OSL wrapper：make MMI code adaptive
  - Queue
  - Timer
- Management of event handler
- Screen management – History mechanism
- NVRAM access
- File system management

Provides OS abstraction

- Provides OS abstraction

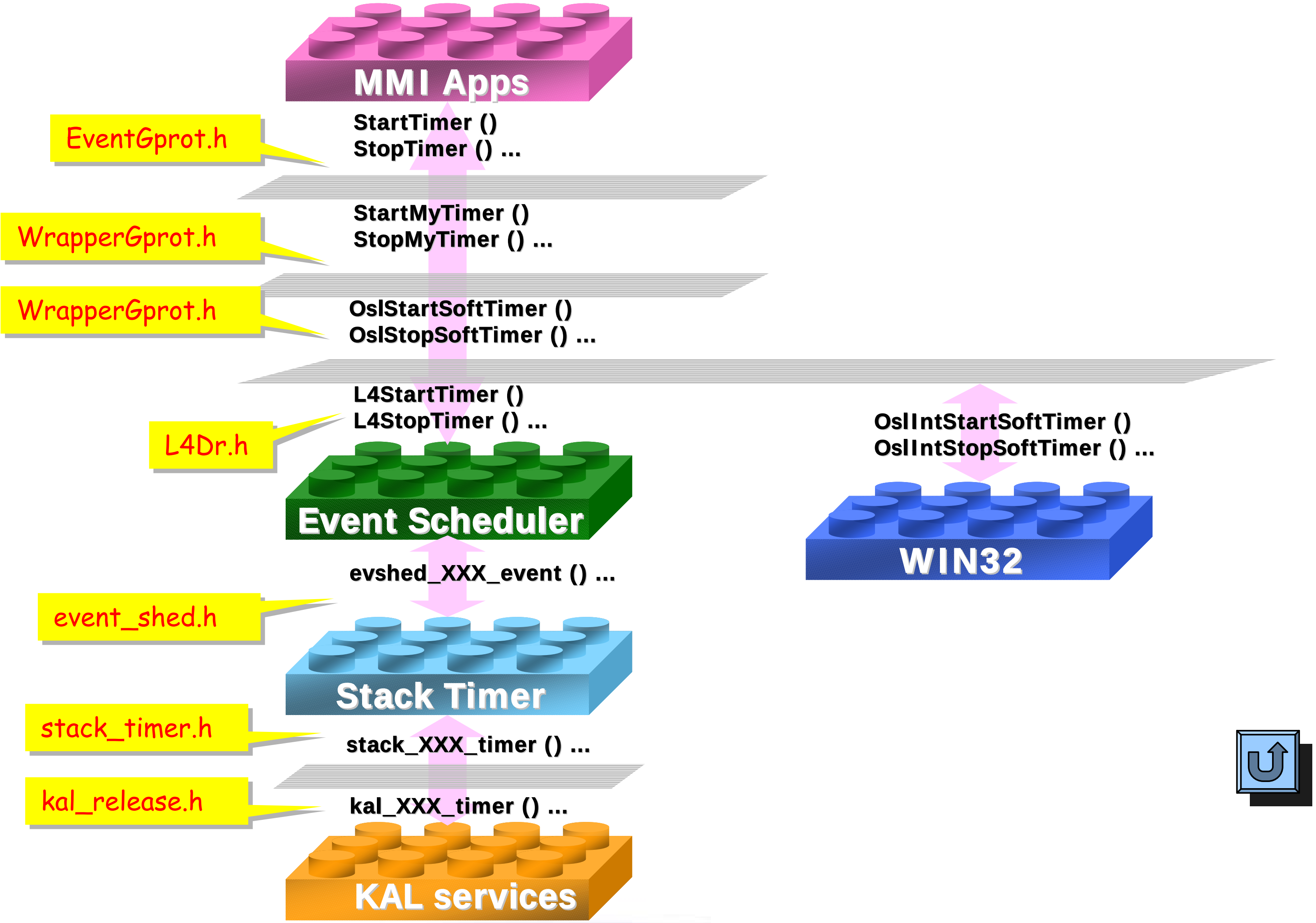
  Provides wrappers to all operating system dependent calls to be made by the application.

  - Queue
    - QueueGprot.h
  - Timer
    - WrapperGprot.h
    - MMIFrameworkComponents.pdf

Queue

- External queue(Inter-task queue, mod to mod(In the diff task))
  - Reseive message: OslReceiveMsgExtQ(receive_msg_ext_q)
  - Send message: OslMsgSendExtQueue(msg_send_ext_queue)
  - Implement: mcu\adaptation\src\stack_ltlcom.c
- Internal queue(Intra-task queue, mod to mod(In the same task))
  - Reseive message: receive_msg_int_q
  - Send message: msg_send_int_queue
  - Implement: mcu\adaptation\src\stack_ltlcom.c
- Circular queue(MMI only, default size 30)
  - Reseive message (From MMI Task): OslReadCircularQ
  - Send message (For NVRAM Access): OslWriteCircularQ
  - Implement: mcu\plutommi\MMI\Framework\Osl\OslSrc\Queue.c

Timer Usage for MMI Apps



Event Handlers

- Event Handlers

  Registers and executes application call backs for various events

  - Protocol events
    - the basic event
    - Indicate by unique protocol event ID
  - Key events
    - One kind of protocol event
  - Highlight events
    - Man-made event, base on key event
    - Associated with hint info

Protocol Events(1/2)

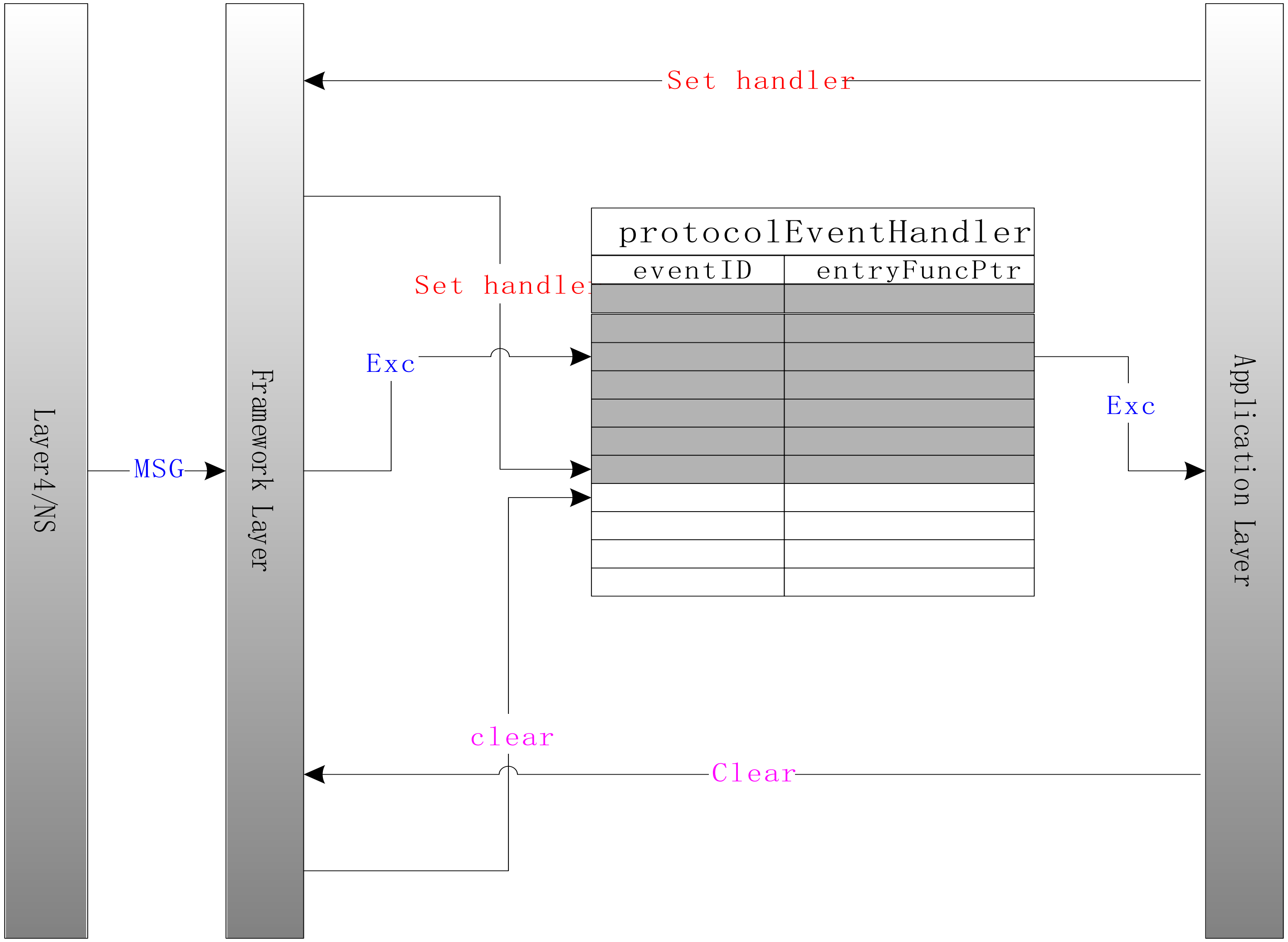


Figure. Protocol Event Handler

Protocol Events (2/2)

- Set Event Handler:

```
void SetProtocolEventHandler(PsFuncPtr funcPtr, U16 eventID)
{
   protocolEventHandler[countOfProtocolEvent].eventID = eventID;
   protocolEventHandler[countOfProtocolEvent].entryFuncPtr = funcPtr;
}
```

- Execute Event Handler:

```
void ExecuteCurrProtocolHandler(U16 eventID,void* MsgStruct,int
   mod_src, void* peerBuf)
{
   PsExtPeerFuncPtr currFuncPtr =
   (PsExtPeerFuncPtr)protocolEventHandler[count].entryFuncPtr;
   (*currFuncPtr)(MsgStruct, mod_src, peerBuf);
}
```

- Event ID: See plutommi\mmi\AsyncEvents\AsyncEventsInc\ProtocolEvents.h

Key Events (1/2)

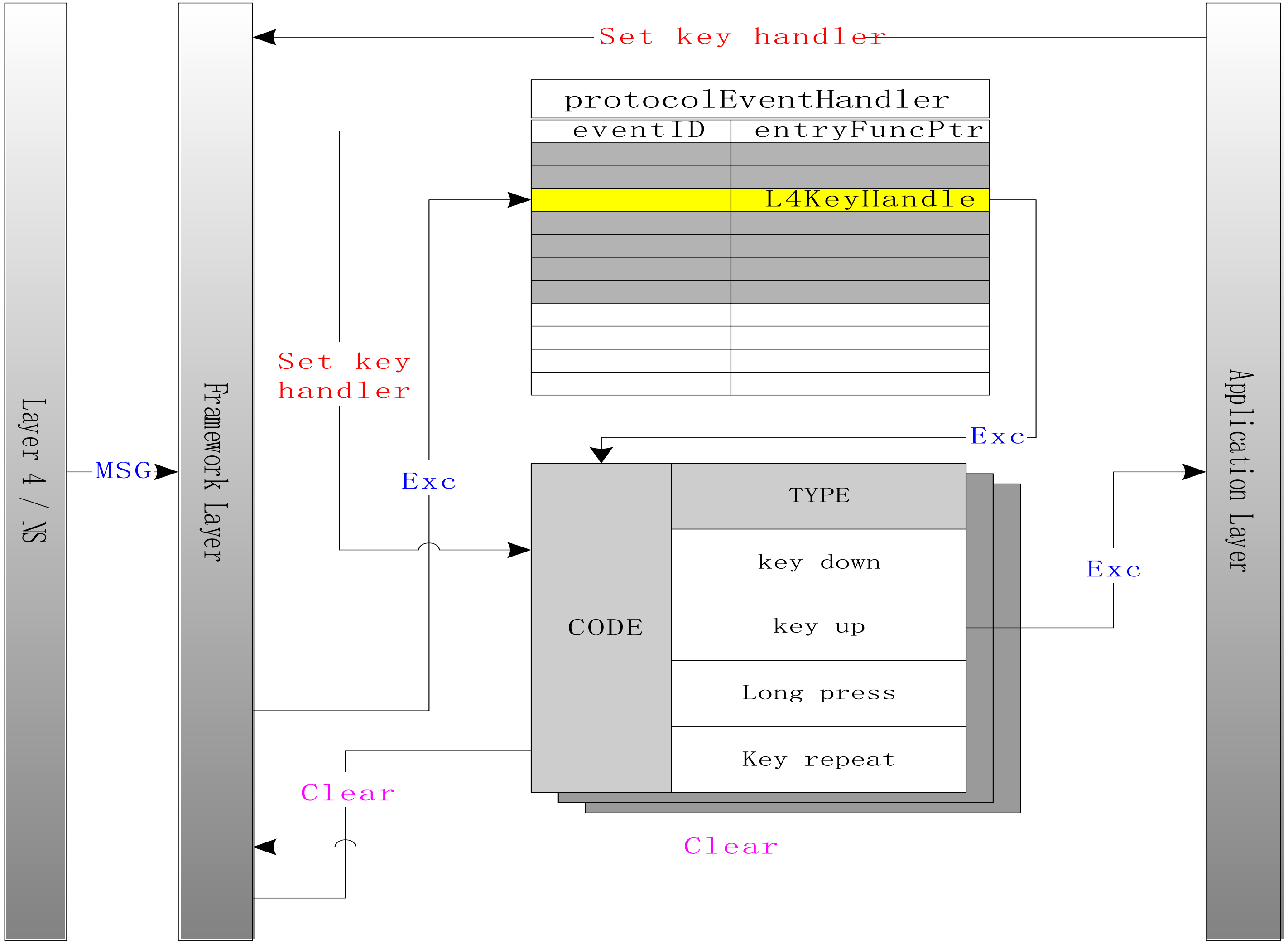


Figure. Key Event Handler

Key Events (2/2)

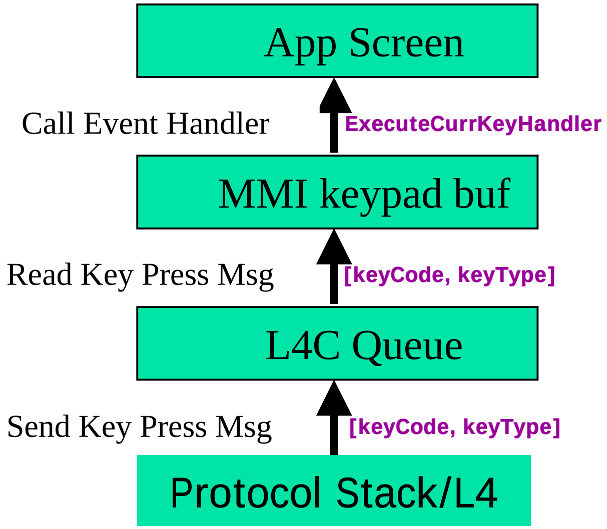
- Key Press Event:
  - Set Key Event Handler:

```
void SetKeyHandler(FuncPtr funcPtr, U16 keyCode, U16 keyType)
{
    currKeyFuncPtrs[keyCode][keyType] = funcPtr;
}
```

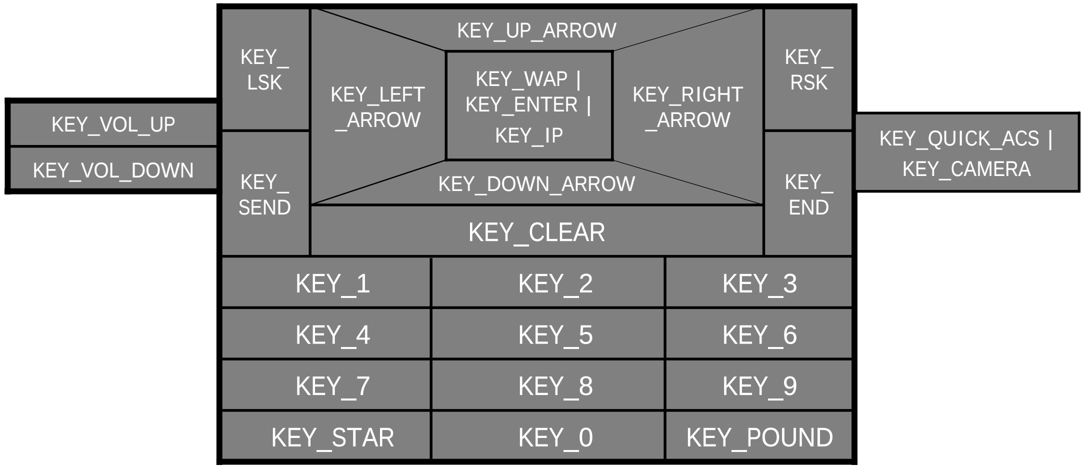
  - Execute Key Event Handler:

```
void ExecuteCurrKeyHandler(S16 keyCode, S16 keyType)
{
        (*currKeyFuncPtrs[keyCode][keyType])
}
```

Key press events flow



KEY CODE


KEY_VOL_UP
KEY_VOL_DOWN
KEY_LSK
KEY_SEND
KEY_UP_ARROW
KEY_LEFT_ARROW
KEY_WAP | KEY_ENTER | KEY_IP
KEY_RIGHT_ARROW
KEY_DOWN_ARROW
KEY_CLEAR
KEY_RSK
KEY_END
KEY_QUICK_ACS | KEY_CAMERA

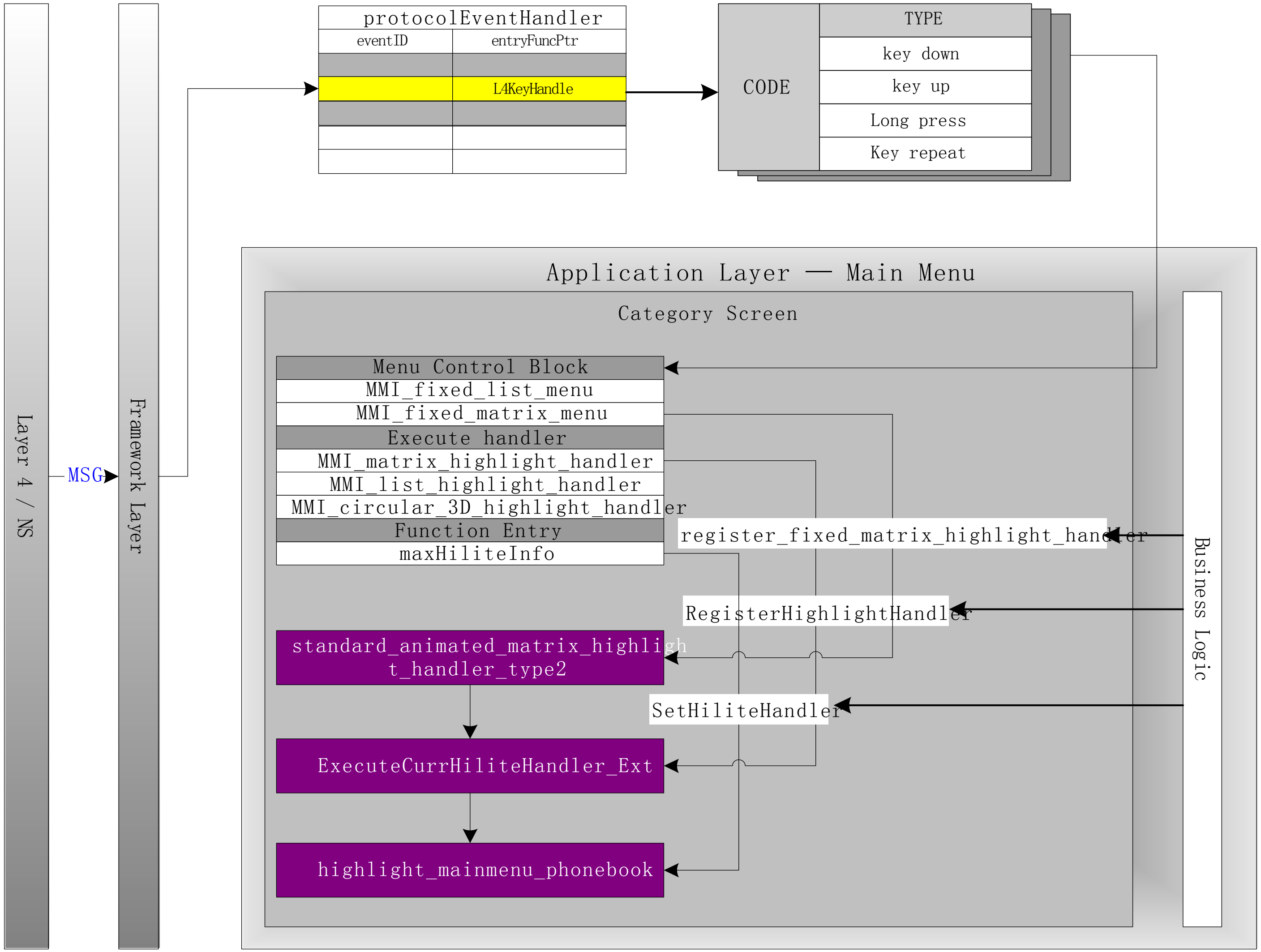
KEY_1	KEY_2	KEY_3
KEY_4	KEY_5	KEY_6
KEY_7	KEY_8	KEY_9
KEY_STAR	KEY_0	KEY_POUND

KEY TYPE

KEY_EVENT_DOWN
KEY_EVENT_UP
KEY_LONG_PRESS
KEY_REPEAT

Refer files:Keypad_def.c, Kbd_table.h

Highlight Events



Layer 4 / NS

MSG

Framework Layer

protocolEventHandler

eventID	entryFuncPtr
	L4KeyHandle

CODE	TYPE
	key down
	key up
	Long press
	Key repeat

Application Layer — Main Menu

Category Screen

Menu Control Block
MMI_fixed_list_menu
MMI_fixed_matrix_menu
Execute handler
MMI_matrix_highlight_handler
MMI_list_highlight_handler
MMI_circular_3D_highlight_handler
Function Entry
maxHiliteInfo

register_fixed_matrix_highlight_handler

RegisterHighlightHandler

SetHiliteHandler

standard_animated_matrix_highlight_handler_type2

ExecuteCurrHiliteHandler_Ext

highlight_mainmenu_phonebook

Business Logic

Figure. Highlight Handler

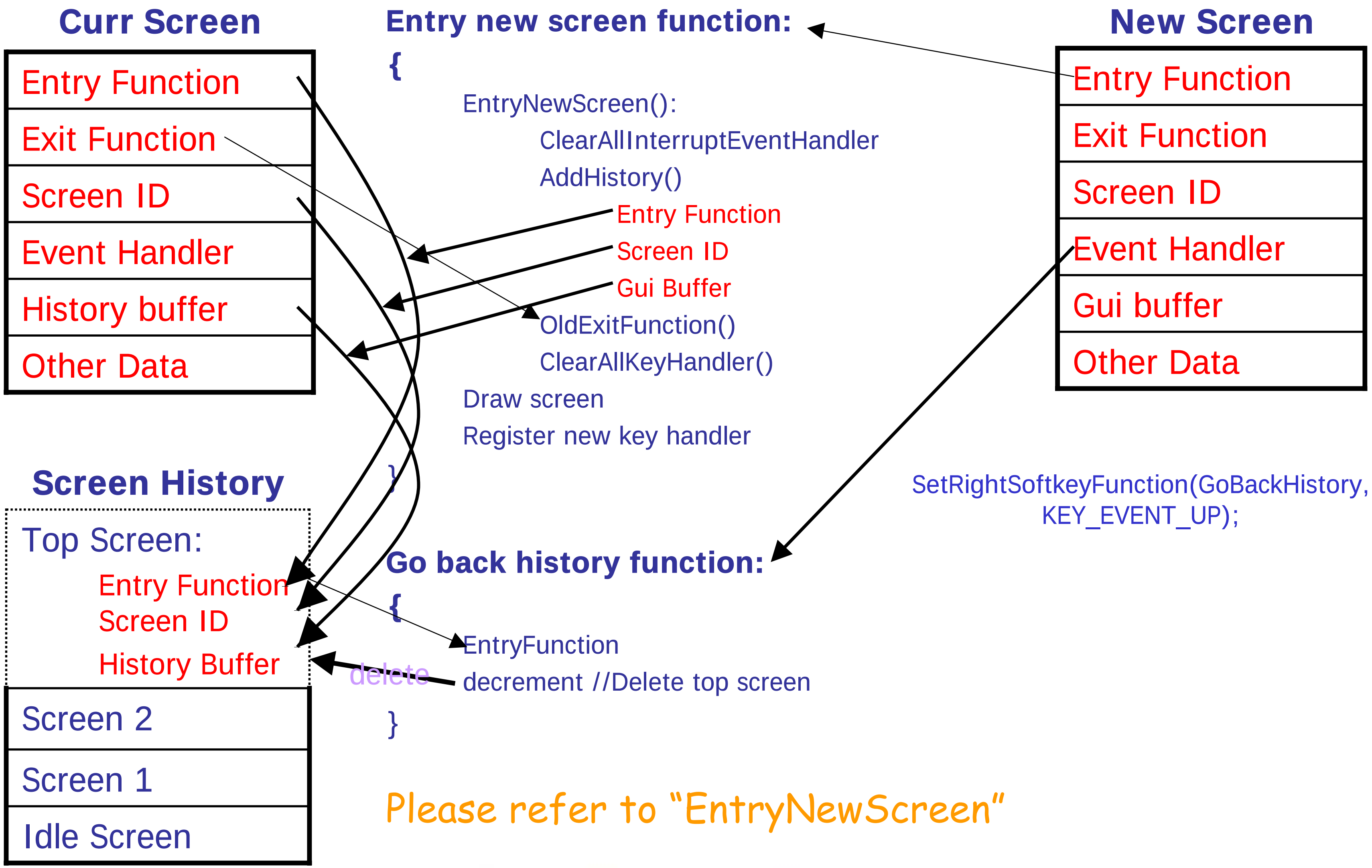
History Manager

- History Manager →
  Helps application maintain screen flow and store intermediate data.

```
typedef struct _historyNode
{
   U16                scrnID;
   FuncPtr            entryFuncPtr;
   U8                 *inputBuffer;
   U8                 *guiBuffer;
  } historyNode;
```

- Structure of history node
  - Screen ID - of screen to be saved
  - Entry Function Pointer – to redraw the screen
  - Input Buffer – to save running text data for this screen
  - GUI Buffer – to save UI related information for this screen

History mechanism



Curr Screen

- Entry Function
- Exit Function
- Screen ID
- Event Handler
- History buffer
- Other Data

Entry new screen function:

```
{
        EntryNewScreen():
                ClearAllInterruptEventHandler
                AddHistory()
                        Entry Function
                        Screen ID
                        Gui Buffer
                OldExitFunction()
                ClearAllKeyHandler()
        Draw screen
        Register new key handler
}
```

New Screen

- Entry Function
- Exit Function
- Screen ID
- Event Handler
- Gui buffer
- Other Data

SetRightSoftkeyFunction(GoBackHistory,
KEY_EVENT_UP);

Screen History

Top Screen:

- Entry Function
- Screen ID
- History Buffer

- Screen 2
- Screen 1
- Idle Screen

Go back history function:

```
{
        EntryFunction
        decrement //Delete top screen
}
```

Please refer to “EntryNewScreen”

History API List

- EntryNewScreen
  - U16 newscrnID
  - FuncPtr newExitHandler
  - FuncPtr newEntryHandler: NULL, if do not want add the new screen to history later
  - void *peerBuf
- AddHistory
  - Max capacity of history stack is 50
- Other API
  - Delete nodes from history
  - Delete ‘N’ nodes from history
  - Go back ‘N’ nodes in history
  - Retrieve history for a screen
  - Retrieve input buffer for screen
  - Retrieve UI buffer for screen
- Detail please refer to : \plutommi\MMI\Framework\History\HistoryInc\HistoryDef.h

NVRAM Access

- NVRAM Access
  Provides wrappers for data storage and retrieval of data from NVRAM.
  - Value
    - *ReadValue*(nId,pBuffer,nDataType,pError);
    - *WriteValue*(nId,pBuffer,nDataType,pError);
  - Record
    - *WriteRecord*(nFileId,nRecordId,pBuffer,nBufferSize,pError);
    - *ReadRecord*(nFileId,nRecordId,pBuffer,nBufferSize,pError);

NVRAM_Configuration_Guide_User.pdf

File System Management

- File System Management
  Provides wrappers for data storage and retrieval of data from File System
  - API
    - Int *FS_Open*(const WCHAR * FileName, UINT Flag);
    - int *FS_Close*(FS_HANDLE FileHandle);
    - int *FS_Read*(FS_HANDLE FileHandle, void * DataPtr, UINT Length, UINT * Read);
    - int *FS_Write*(FS_HANDLE FileHandle, void * DataPtr, UINT Length, UINT * Written);
    - int *FS_Seek*(FS_HANDLE FileHandle, int Offset, int Whence);
    - int *FS_Delete*(const WCHAR * FileName);
    - int *FS_GetFileSize*(FS_HANDLE FileHandle, UINT * Size);

FileSystem_Document_20050216_W05.09.pdf

UI

- Category Screen
  - Category Functions
    - The category layer consists of a set of functions that an application can use to define its User Interface.
    - Each Category screen contains the following functions:
      - Function to enter (display)
      - Function to exit
      - Function to get the size of History
      - Function to get the History
- MMI Resource
  - Image, Audio, Strings, Fonts, Themes, Menu Tree.

Category screen mechanism

Every category screen has a set of functions :

- ShowCategoryXXXScreen
  - Register event handler
  - Pre-process UI element
  - Call redraw function
- RedrawCategoryXXXScreen
  - Draw screen using GDI functions
- ExitCategoryXXXScreen
  - Reset function pointer
  - Other operation depend on vary screens
- GetCategoryXXXHistorySize
  - Be used to return the size of gui buffer & input buffer
- GetCategoryXXXHistory
  - Be used to return the data of gui buffer & input buffer
- GetCategoryXXXData
  - Be used to return input buffer

Example: GetCategory157Data, GetCategory200History

set_list_menu_category_history, get_list_menu_category_history

Example:

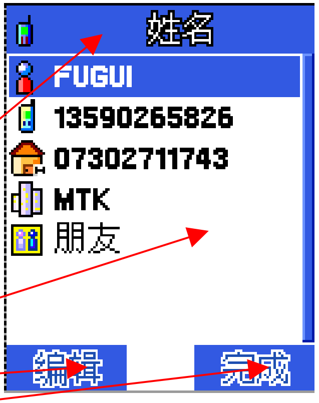
```
void ShowCategory1Screen (
STRING_ID          Title,
IMAGE_ID           TitleIcon,
STRING_ID          LSKLabel,
IMAGE_ID           LSKIcon,
STRING_ID          RSKLabel,
IMAGE_ID           RSKIcon,
INT               NumberOfItems,
STRING_ID*         ListOfItems,
BYTE*              HistoryBuffer );
```

Screen example

Category Screen

- category1 screen
- category2 screen
- category3 screen
- ……
- CategoryXXX screen
  - ShowCategoryXXXScreen
  - RedrawCategoryXXXScreen
  - ExitCategoryXXXScreen
  - GetCategoryXXXHistory
  - GetCategoryXXX1HistorySize
- ……

```
ShowCategoryXXXScreen:
{
        Init XXXScreen data;
        RedrawCategoryXXXScreen;
}

RedrawCategoryXXXScreen:
{
        draw_title();
        show_fixed_list();
        show_left_softkey();
        show_right_softkey();
}
```


姓名

FUGUI

13590265826

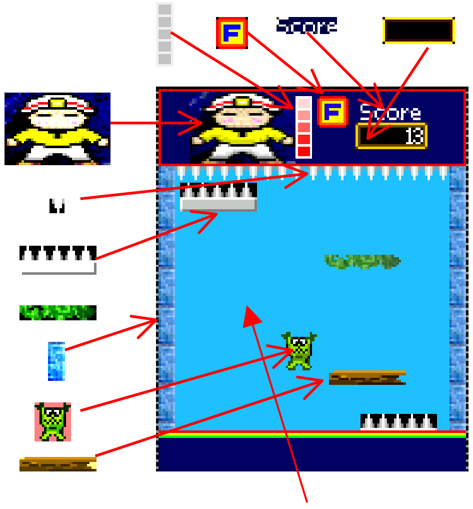
07302711743

MTK

朋友

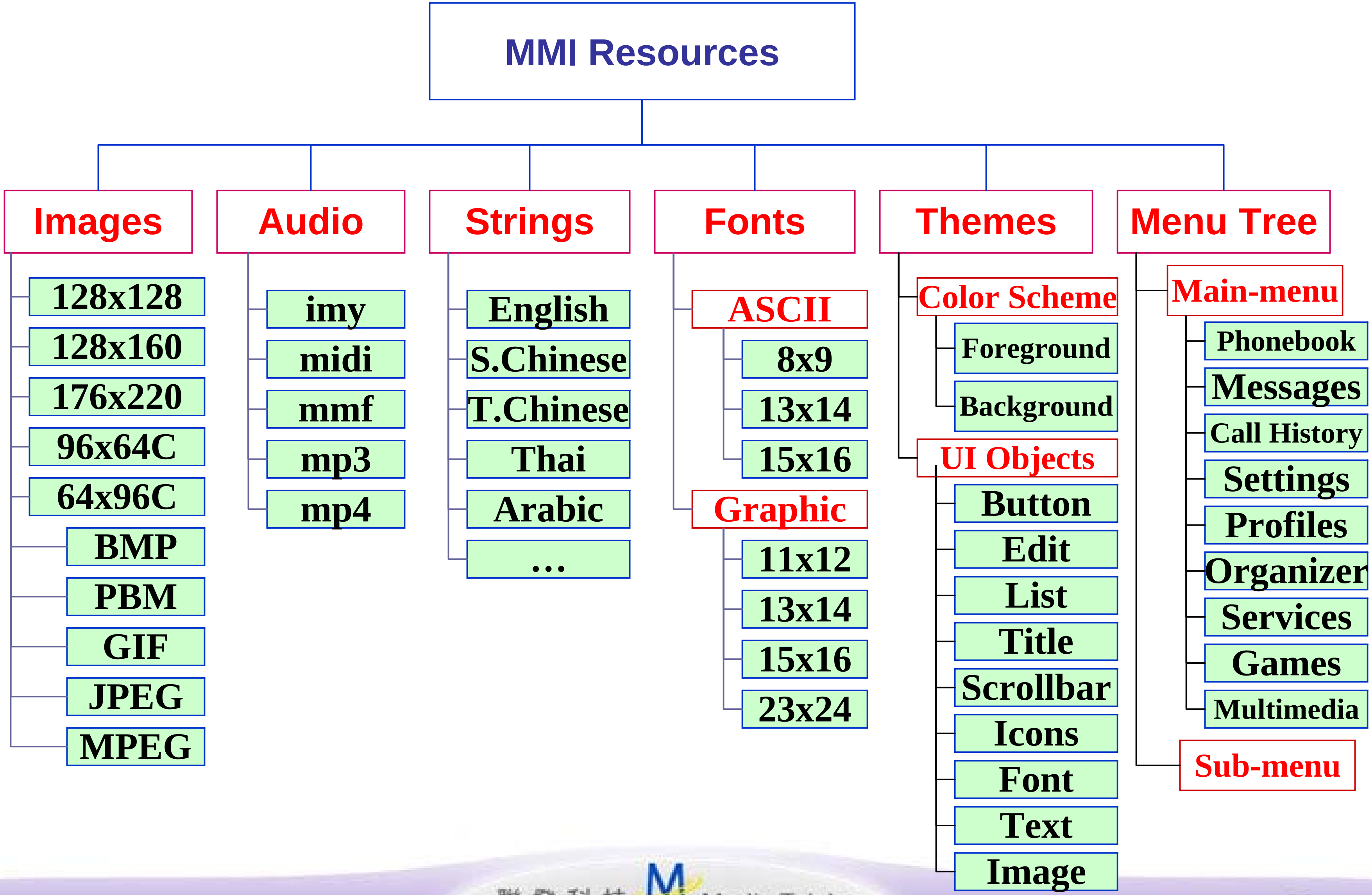
Common Screen:

pixtel_UI_show_image



pixtel_UI_fill_rectangle

MMI Resources (1/5 :classification)



MMI Resources

- Images
  - 128x128
  - 128x160
  - 176x220
  - 96x64C
  - 64x96C
  - BMP
  - PBM
  - GIF
  - JPEG
  - MPEG
- Audio
  - imy
  - midi
  - mmf
  - mp3
  - mp4
- Strings
  - English
  - S.Chinese
  - T.Chinese
  - Thai
  - Arabic
  - …
- Fonts
  - ASCII
    - 8x9
    - 13x14
    - 15x16
  - Graphic
    - 11x12
    - 13x14
    - 15x16
    - 23x24
- Themes
  - Color Scheme
    - Foreground
    - Background
  - UI Objects
    - Button
    - Edit
    - List
    - Title
    - Scrollbar
    - Icons
    - Font
    - Text
    - Image
- Menu Tree
  - Main-menu
    - Phonebook
    - Messages
    - Call History
    - Settings
    - Profiles
    - Organizer
    - Services
    - Games
    - Multimedia
  - Sub-menu

MMI Resource (2/5)

- String
  - Step 1: add string to ref_list.txt
  - Step 2: add string ID to ENUM associated with app
  - Step 3: using macro ADD_APPLICATION_STRING2
  - Step 4: S8* my_string = GetString(MY_STR_ID);
  - Using APP_BASE to guarantee the uniqueness of string ID
- Image
  - Step 1: put images in the folder assigned to app
  - Step 2: add image ID to ENUM associated with app
  - Step 3: using macro ADD_APPLICATION_STRING2
  - Step 4: using image ID directly as parameter
  - Using APP_BASE to guarantee the uniqueness of image ID

MMI Resource(3/5)

- Menu
  - Parent menu
  - Unique menu item ID
  - Hilite function and LSK handler
  - Associated with screen
- Audio
- Skin Layout
  - Audio player
  - Calculator
  - FMRadio
- Theme
- Fonts

TOOLS:
MCT, MCU\tools\AudioResGen

MMI Resource (4/5 : Macro)

- ADD_APPLICATION_STRING2(STR_CAL_MONTH,"M","Chinese month");
  - String ID, Value, Description
- ADD_APPLICATION_IMAGE2(IMG_CAL_ON,CUST_IMG_BASE_PATH"\\\\EmptyImage.bmp","Icon for On Button.");
  - Image ID, Path, Description
- ADD_APPLICATION_MENUITEM

```
((MENU_CAL_TYPE,                  /* Menu ID */
ORGANIZER_CALENDER_MENU,          /* Parent ID*/
1,                                /* Child number*/
MENU_ID_CHILD_1,                  /* Child ID */
SHOW,                             /* Hide or show*/
NONMOVEABLE,                      /* Move attribute*/
DISP_LIST,                        /* Display attribute*/
CAL_STRING_LUNAR,                 /* String ID*/
0));                              /* ICON ID*/
```

MMI Resource (5/5)

CLASS	COMPONENT	SOURCE FILE	TEMPORARY FILE	PRIMAL FILE	MCT TOOL
Images	CustImgRes.obj	CustImgRes.c, custimgdatahw.h		**Image files**: Mcu\plutommi\Customer\Images\ **IDs:** mcu\plutommi\mmi\AppXXX_dir\inc\AppXXXDef.h **Populate:** mcu\plutommi\Customer\Res_MMI\Res_AppXXX.c	Verify Image
	CustImgMap.obj	CustImgMap.c			
	resource_image_jtbl.obj	resource_image_jtbl.c			
Strings	CustStrRes.obj	CustStrRes.c	enum_list.h CustResList_out.txt	**String files**: Mcu\\plutommi\Customer\CustResource\ref_list.txt **IDs:** mcu\plutommi\mmi\AppXXX_dir\inc\AppXXXDef.h **Populate:** mcu\plutommi\Customer\Res_MMI\Res_AppXXX.c	
	CustStrMap.obj	CustStrMap.c			
	resource_str_jtbl.obj	resource_str_jtbl.c			
Menus	CustMenuRes.obj	CustMenuRes.c	CustMenuTree_Out.c CustMenuTreeID_Out.c	**IDs:** mcu\plutommi\mmi\AppXXX_dir\inc\AppXXXDef.h **Populate:** mcu\plutommi\Customer\Res_MMI\Res_AppXXX.c	Preview
Fonts	FontRes.obj	FontRes.c,L_1_Large.h, L_1_Medium.h......		pluto_large.bdf pluto_medium.bdf Pluto_small.bdf …….	Font Merger, Font Splitter, Font Viewer, Font Customizer
	FontType.obj	FontType.c			
	resource_font_jtbl.obj	resource_font_jtbl.c			
Audio	resource_audio.obj	resource_audio.c resource_audio.h		Mcu\tools\AudioResGen\*.*	Audio Generator
Themes	ThemeRes.obj	themecomponents.h, ThemeRes.c		New or old XXX.thm file	Theme Generator
App Resource	resource_audply_skins.obj	resource_audply_skins.c		plutommi\Customer\Images\ProjectName\MainLCD\AudioPlayer	Skin Layouter
	resource_camera_skins.obj	resource_camera_skins.c		plutommi\Customer\Images\PLUTO128X160\MainLCD\Camera	
	resource_fmradio_skins.obj	resource_fmradio_skins.c		plutommi\Customer\Images\PLUTO176X220\MainLCD\FMRadio	
	resource_video_skins.obj	resource_video_skins.c		plutommi\Customer\Images\PLUTO176X220\MainLCD\Video	
	resource_world_clock_city.obj	resource_world_clock_city.c		City_Database.txt, City_Database_Coord.txt,Map, ref_list.txt	World Clock Map
Other	CustMiscData.obj	CustMiscData.c			
	gui_wrapper.obj	gui_wrapper.c			
	StandaloneRes.obj	StandaloneRes.c			

MMI Resource Customization (1/2)

- String : modify ref_list.txt by hand
- Image
  - Replace the original picture with same dimension
  - Verify using MCT
  - Refer to 176X220GPRS.pdf
- Font
  - Font_And_Input_Method_Spec_for_Different_Languages.pdf
  - Using MCT

MMI Resource Customization (2/2)

- Menu
  - Customization by hand
- Ring
  - Mcu\tools\AudioResGen\AudioResGen.exe
  - User Manual For Audio Resource Generator tool.doc

	Input Files	Format	Menu	Resource_audio.h	Resource_audio.c
来电铃声/闹铃	imy.txt	*.imy	Ring 1- Ring 10	#define MIN_RING_TONE_ID 101 #define RING_TONE_1 101 #define RING_TONE_2 102 …… #define RING_TONE_10 110 #define MAX_RING_TONE_ID 110	mtk_resource_imelodys[ ]
	midi.txt	*.mid(*.mp3)	MIDI 1- MIDI 15	#define MIN_MIDI_ID 151 #define MIDI_1 151 #define MIDI_2 152 …… #define MIDI_15 165 #define MAX_MIDI_ID 165	mtk_resource_midis[ ]
信息/ 开关机/开关盖	message.txt sound.txt	*.mid	Tone 1- Tone 10	#define MIN_SND_ID 201 #define SOUND_1 201 #define SOUND_2 202 …… #define SOUND_10 210 #define MAX_SND_ID 210	mtk_resource_message_sounds[ ] mtk_resource_sounds[ ]
EMS旋律	ems_imy.txt	*.imy	Melody	#define MIN_EMS_IMY_ID 141 #define MAX_EMS_IMY_ID 145	mtk_resource_ems_imelodys[ ]
EMS预设声音	ems.txt	*.mid	Predefined Sound	#define MIN_MSG_SND_ID 221 #define MAX_MSG_SND_ID 230	mtk_resource_ems_sounds[ ]
MSS	mms_snd.txt	*.mid		#define MIN_MMS_SND_ID 241 #define MAX_MMS_SND_ID 250	mtk_resource_mms_sounds[ ]

MMI directories

- Application:

Idle Screen:	plutommi\mmi\IdleScreen
Main Menu:	plutommi\mmi\MainMenu
Phone Book:	plutommi\mmi\PhoneBook
Messages:	plutommi\mmi\Messages
Call History:	plutommi\mmi\Calls
Call Management:	plutommi\mmi\CallManagement
Setting:	plutommi\mmi\Setting
File Manager:	plutommi\mtkapp\FileMgr
Fun &Games:	plutommi\mmi\FunAndGames
User Profiles:	plutommi\mmi\PROFILES
Organizer:	plutommi\mmi\Organizer
Services:	plutommi\mmi\SAT
Shortcuts:	plutommi\mmi\Shortcuts
Audio Player:	plutommi\mtkapp\AudioPlayer
Camera:	plutommi\mtkapp\Camera
FMRadio:	plutommi\mtkapp\FMRadio
Photo Editor:	plutommi\mtkapp\PhotoEditor
Sound Recorder:	plutommi\mtkapp\SoundRecorder

MMI directories (cont)

- Common MMI features (Hardware and Win32):
  - mcu\plutommi\Customer\CustResource\MMI_features[PROJ].h

- MMI framework:

Osl:	plutommi\mmi\Framework\Osl
Task:	plutommi\mmi\Framework\Tasks
History:	plutommi\mmi\Framework\History
Event:	plutommi\mmi\Framework\EventHandling
NVRAM:	plutommi\mmi\Framework\NVRAMManager

MMI directories (cont)

- Category resource(mcu\plutommi\Customer\):
  - CustResource:

    All data settings and resources for each specific customer and these will be copy to \CustomerInc and \Res_MMI for building software.
  - Image: Graphics resources in PBM (portable bitmap format), BMP, GIF formats.
  - Res_MMI: Populator resources
  - ResGenerator: Resource generation tool
  - ResourceDLL: DLL for resource generation tool

- Category multimedia:
  - GUI: plutommi\mmi\GUI
  - GDI: plutommi\mtkapp\GDI
  - MDI: plutommi\mtkapp\MDI

Write an application – Resource

- Define APP_BASE in PixtelDataTypes.h
- Declaration unique ids for
  - Screens
  - Strings
  - Images
  - Menu Items (GlobalMenuItems.h)
- Write function to populate resources
  - Invoked by Resource Generator (PopulateRes.c)
  - Use macro ADD_APPLICATION_XXX
- Modify “Makefile” of ResGenerator and “readexcel.c”

Write an application – make file

- Add key macro in make file
- Add feature macro in MMI_features$Proj.h
- Add library file
  - COMPOBJS
- Add compile list
  - Create directory in mcu\make
  - Add directory name to CUS_REL_SRC_COMP

Write an application - Initialization

- Initialization Function
  - Invoked from bootup time from Initialization of MMITask function InitializeAll (Not all)
  - Initializes various event handlers.

Sample Code

```
void InitIncomingCall (void) {
                        …
SetProtocolEventHandler (psCBackCallIncoming, PRT_INCOMINGCALL_EVENT);
SetHiliteHandler(MITEM_INC_OPT_ANSWER, HiliteMenuIncomingAnswer);
                        …
}
```

Write an application – Entry and Exit(1/2)

- Entry and Exit Function
  - Flow of screens controlled by Entry Function and Exit Functions
  - Typical Execution of Entry Functions
    - Call To Execute Current Exit Handler
    - Get GUI Buffer for current screen
    - Get elements to show on the screens
    - Register highlight handler
    - Call Category function to draw screen
    - Set Exit Handler

Write an application – Entry and Exit(2/2)

- Entry Functions should be re-entrant
- Typical Flow of Exit Functions
  - Create History Node
  - Save Entry Function in history node
  - Fill input buffer and GUI buffer in history node
  - Save history
- Highlight Handlers
  - Written to execute user defined code on highlight of a menu item
  - Typical Flow of highlight handlers
    - Change handlers for left and right soft keys

Example

- Add MenuItem [My Setting] to [Settings], See Image 1.
- Add MenuItem [My Setting1] and [My Setting2] to [My Setting], See Image 2.
- On Screen [My Setting], display popup if press left soft key, See Image 3.

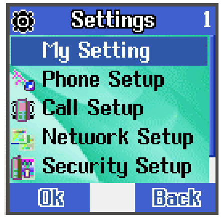


Image 1

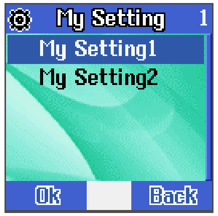


Image 2

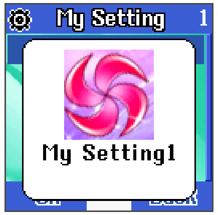


Image 3

Example (cont.)

- Step 1: Add Resource
  - Add material:

    Add image MY_SETTING.GIF to
    plutommi\Customer\Images\BULL600\MAINLCD\SubMenu\Settings

    Add 3 strings to plutommi\Customer\CustResource\BULL600_MMI\ref_list.txt:

```
STR_MY_SETTING      Undefined  10         My Setting  My Setting  我的设定    我的设定
                    My Setting  My Setting  My Setting  My Setting  My Setting  My Setting  My Setting
                    My Setting
STR_MY_SETTING1     Undefined  11         My Setting1             My Setting1
                    我的设定1  我的设定1  My Setting1             My Setting1             My Setting1
                    My Setting1             My Setting1             My Setting1             My Setting1
                    My Setting1
STR_MY_SETTING2     Undefined  11         My Setting2             My Setting2
                    我的设定2  我的设定2  My Setting2             My Setting2             My Setting2
                    My Setting2             My Setting2             My Setting2             My Setting2
                    My Setting2
```

Example –step 1 cont

- Add resource ID:

  Add menu item ID to GlobalMenuItems.h:

  Add MENU_MY_SETTING, MENU_MY_SETTING1, MENU_MY_SETTING2 to enum GLOBALMENUITEMSID

  Add image ID,string ID,screen ID to SettingDefs.h

  Add SCR_MY_SETTING, SCR_MY_SETTING1, SCR_MY_SETTING2 to enum SCR_SETTING_LIST

  Add STR_MY_SETTING, STR_MY_SETTING1, STR_MY_SETTING2 to enum STR_SETTING_LIST

  Add IMG_MY_SETTING to enum IMG_SETTING_LIST

- Add populate code:

  Add MENU_MY_SETTING to MAIN_MENU_SETTINGS_MENUID, See:

```
ADD_APPLICATION_MENUITEM((MAIN_MENU_SETTINGS_MENUID,IDLE_SCREEN_MENU_ID,6,
        MENU_MY_SETTING,MENU9102_INITIAL_SETUP,MENU8237_SCR8093_MNGCALL_MENU_MAIN,
        MENU9185_NETWORK_SETUP,         MENU9101_SECURITY,MENU_SETTING_RESTORE,
        0,MOVEABLEACROSSPARENT,1,MAIN_MENU_SETTINGS_TEXT,
        MAIN_MENU_SETTINGS_ICON));
```

Example (cont.)

Add the flow to function populateSettingMenu(in Res_Setting.c):

```
ADD_APPLICATION_MENUITEM((MENU_MY_SETTING,MAIN_MENU_SETTINGS_MENUID,2,
        MENU_MY_SETTING1,
        MENU_MY_SETTING2,
        SHOW, MOVEABLEWITHINPARENT, DISP_LIST,STR_MY_SETTING,0));
ADD_APPLICATION_MENUITEM((MENU_MY_SETTING1,MENU_MY_SETTING,0,
        SHOW, MOVEABLEWITHINPARENT, DISP_LIST,STR_MY_SETTING1,0));
ADD_APPLICATION_MENUITEM((MENU_MY_SETTING2,MENU_MY_SETTING,0,
        SHOW, MOVEABLEWITHINPARENT, DISP_LIST,STR_MY_SETTING2,0));
ADD_APPLICATION_IMAGE2(IMG_MY_SETTING,
        CUST_IMG_PATH"\\\\MainLCD\\\\SubMenu\\\\Settings\\\\MY_SETTING.GIF","My
Setting.");
ADD_APPLICATION_STRING2(STR_MY_SETTING,      "My Setting","My Setting");
ADD_APPLICATION_STRING2(STR_MY_SETTING1,     "My Setting1","My Setting1");
ADD_APPLICATION_STRING2(STR_MY_SETTING2,     "My Setting2","My Setting2");
```

Example (cont.)

- Step 2: Implement (modify SettingSrc.c)

1. Set HiliteHandler (Add to function InitSettingApp):

```
SetHiliteHandler(MENU_MY_SETTING,HighlightMySetting);
SetHiliteHandler(MENU_MY_SETTING1,HighlightMySetting1);
SetHiliteHandler(MENU_MY_SETTING2,HighlightMySetting2);
```

2. Implement 3 HiliteHandler:

```
void HighlightMySetting(void)
{
    SetKeyHandler(GoBackHistory, KEY_LEFT_ARROW, KEY_EVENT_DOWN);
    SetRightSoftkeyFunction(GoBackHistory,KEY_EVENT_UP);
    SetKeyHandler(EntryMySetting, KEY_RIGHT_ARROW,KEY_EVENT_DOWN);
    SetLeftSoftkeyFunction(EntryMySetting,KEY_EVENT_UP);
}
void HighlightMySetting1(void)
{
    SetKeyHandler(GoBackHistory, KEY_LEFT_ARROW, KEY_EVENT_DOWN);
    SetRightSoftkeyFunction(GoBackHistory,KEY_EVENT_UP);
    SetKeyHandler(EntryMySetting1, KEY_RIGHT_ARROW,KEY_EVENT_DOWN);
    SetLeftSoftkeyFunction(EntryMySetting1,KEY_EVENT_UP);
}
void HighlightMySetting2(void)
{
    SetKeyHandler(GoBackHistory, KEY_LEFT_ARROW, KEY_EVENT_DOWN);
    SetRightSoftkeyFunction(GoBackHistory,KEY_EVENT_UP);
    SetKeyHandler(EntryMySetting2, KEY_RIGHT_ARROW,KEY_EVENT_DOWN);
    SetLeftSoftkeyFunction(EntryMySetting2,KEY_EVENT_UP);
}
```

Example (cont.)

3. ## Implement 3 entry function

```
void EntryMySetting(void)
{
        U16 nStrItemList[MAX_SUB_MENUS];              /* Stores the strings id of submenus returned */
        U16 nNumofItem;                                /* Stores no of children in the submenu*/
        U8* guiBuffer;                                /* Buffer holding history data */
        U16 ImageList[MAX_SUB_MENUS];
        EntryNewScreen(SCR_MY_SETTING, NULL, EntryMySetting, NULL);
        /* 2 Get current screen to gui buffer  for history purposes*/
        guiBuffer = GetCurrGuiBuffer(SCR_MY_SETTING);
        /* 3. Retrieve no of child of menu item to be displayed */
        nNumofItem = GetNumOfChild(MENU_MY_SETTING);
        /* 4. Retrieve string ids in sequence of given menu item to be displayed */
        GetSequenceStringIds(MENU_MY_SETTING,nStrItemList);
        GetSequenceImageIds(MENU_MY_SETTING, ImageList);
        /* 5 Set current parent id*/
        SetParentHandler(MENU_MY_SETTING);
        /* 6 Register highlight handler to be called in menu screen */
        RegisterHighlightHandler(ExecuteCurrHiliteHandler);
        /* 7 Display Category1 Screen */
        ShowCategory15Screen(STR_MY_SETTING, IMG_SCR_SETTING_CAPTION,  STR_GLOBAL_OK, IMG_GLOBAL_OK,
                STR_GLOBAL_BACK, IMG_GLOBAL_BACK,  nNumofItem, nStrItemList, ImageList,  LIST_MENU, 0, guiBuffer);
        /* 8.Register function with right softkey */
        SetRightSoftkeyFunction(GoBackHistory,KEY_EVENT_UP);
}
void EntryMySetting1(void)
{
        S8 * string = GetString(STR_MY_SETTING1);
        U16 imageId = IMG_MY_SETTING;
        EntryNewScreen(SCR_MY_SETTING1, NULL,  EntryMySetting1,NULL);
        ShowCategory65Screen((U8*)string,imageId,NULL);
        SetRightSoftkeyFunction(GoBackHistory,KEY_EVENT_UP);
}
void EntryMySetting2(void)
{
        S8 * string = GetString(STR_MY_SETTING2);
        U16 imageId = IMG_MY_SETTING;
        EntryNewScreen(SCR_MY_SETTING2, NULL, EntryMySetting2, NULL);
        ShowCategory65Screen((U8*)string,imageId,NULL);
        SetRightSoftkeyFunction(GoBackHistory,KEY_EVENT_UP);
}
```

Third party software - MTK already support

- License is the first important key
- General step
  - Turn on feature in $custom_$project.mak
  - Turn on feature in MMI_features$project.h
  - Copy file to the dedicated position
- Nucleus and File system
- Font and Input method(Zi/T9)
  - SOP for MTK Font and IME Configuration
- Obigo WAP/MMS
- Handwriting(汉王/盟田)
- III JAVA

3rd Party License Contact 20051128.doc

Third party software - MTK not support yet

- Input method
    - InputMethodPortingGuide.doc
- Others refer to “Write Application”

Tool

- Catcher
  - Catcher_USER_MANUAL.pdf

    *void **kal_prompt_trace**(module_type, kal_char *fmt,…)*
  - Catcher_Filter_Settings_for_MMI_and_Protocol_Issues.pdf
- Flash_tool
  - FlashTool_V2.6_Application_Note.pdf
- Phone Suite
  - PhoneSuite_User_Manual.pdf
- Meta
  - META_MAUI_APP_note 0.14.pdf
- MCT
  - User Manual for MCT 4.2.pdf

General intro of Simulator

- MMI Simulator
  - Guide to Pixtel Network Simulator.pdf

Q&A

Thank you!