◆ 拨号界面

1)坐标调整 **wgui.h**

设置断点，跟踪

IdleScreenDigitHandler()-->ShowCategory16Screen()-->setup_dialing_keypad(&dialing_keypad)

Wgui_inputs.c 中setup_dialing_keypad()函数将wgui.h 中各个赋的值赋给结构变量 dialing_keypad 的各个成员变量

附：（调试、跟踪过程）

1. Dialing inputbox(如何画拨号数字框及显示拨号数字图片)

在 dm_redraw_category_screen 处下一个断点，跟踪，画控件 DM_DIALER_INPUT_BOX1(拨号输入框)

```
case DM_DIALER_INPUT_BOX1:
{
   dm_setup_and_draw_dialer_inputbox(&UICtrlAccessPtr_p);
   break;
}
```

跟进函数 dm_setup_and_draw_dialer_inputbox()里去，看到函数 show_dialer_inputbox()

跟进去，看到如下 代码：

```
gui_show_dialer_input_box(&MMI_dialer_inputbox);
```

再跟进去：

```
gui_show_dialer_input_box_ext(b, (-1), (-1));
```

1. Dialing inputbox(如何在拨号框画拨号数字图片)

进入 ShowCategory16Screen()

1) wgui_setup_dialer_inputbox() 画拨号界面输入框

```
#elif defined (__MMI_MAINLCD_176X220__)
   wgui_setup_dialer_inputbox(
      0,
      MMI_status_bar_height,
      UI_device_width,
      29,
      (UI_buffer_type) Buffer,
      BufferLength,
      MMI_CATEGORY16_ID,
      get_string(right_softkey),
      get_image(right_softkey_icon),
      history_buffer,
      0);
```

2) 产生触摸屏事件后，拨号界面显示(例：触摸屏按下按钮 8 后，产生一个 MMI_PEN_EVENT_DOWN 事件，显示数字键 8 的拨号图片在拨号框)

◆注册触摸屏事件(当触摸屏产生 MMI_PEN_EVENT_DOWN、MMI_PEN_EVENT_UP、MMI_PEN_EVENT_MOVE、MMI_PEN_EVENT_REPEAT、MMI_PEN_EVENT_LONG_TAP、MMI_PEN_EVENT_ABORT 等动作时，执行的函数)

```
#ifdef __MMI_TOUCH_DIAL_SCREEN__
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenDownHandler,
    MMI_PEN_EVENT_DOWN);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenUpHandler, MMI_PEN_EVENT_UP);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenMoveHandler,
    MMI_PEN_EVENT_MOVE);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenRepeatHandler,
    MMI_PEN_EVENT_REPEAT);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenLongTapHandler,
    MMI_PEN_EVENT_LONG_TAP);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenAbortHandler,
    MMI_PEN_EVENT_ABORT);
#endif /* __MMI_TOUCH_DIAL_SCREEN__ */
```

◆以 MMI_PEN_EVENT_DOWN 事件为例：

进入Cate16CategoryControlAreaPenDownHandler()函数

```
    if (ret)
    {
        gdi_lcd_repaint_all();
        ExecuteDialKeyPadKeyHandler(KEY_EVENT_DOWN);
    }
```

此处，产生了一个 KEY_EVENT_DOWN 事件（此处 key_type 常用的分为 KEY_EVENT_DOWN，KEY_EVENT_UP，KEY_EVENT_LONG_PRESS，值分别为0,1,2）

```
S32 key_id = dialing_keypad.key_type;
```

此处赋值后，dialing_keypad.key_type 为选定的按键值，此处值为 8

```
typedef struct _dialing_keypad_struct
{
   S32 keypad_x, keypad_y;
   S32 keypad_width, keypad_height;
   S32 key_width, key_height;
   S32 horizontal_gap, vertical_gap;
   S32 selected_key_x, selected_key_y;
   S32 n_rows, n_column;
#ifdef __MMI_TOUCH_DIAL_SCREEN_WITH_FUNCTION__
   S32 keypad_func_x, keypad_func_y;
   S32 func_key_width, func_key_height;
   S32 func_horizontal_gap, func_vertical_gap;
   S32 func_n_rows, func_n_column;
#endif /* __MMI_TOUCH_DIAL_SCREEN_WITH_FUNCTION__ */
   S32 key_type;  /* selected key index */
#ifndef __GDI_MEMORY_PROFILE_2__
   gdi_image_cache_bmp_struct selected_key_bitmap;
#endif
} dialing_keypad_struct;
```

跟进去ExecuteDialKeyPadKeyHandler()，此函数执行拨号界面的按键事件

```
    /* 1-9 */
    if (key_id >= MMI_DIALING_KEY_START && key_id < MMI_DIALING_KEY_STAR)
    {
        PlayDialKeyPadTone(key_type);
        ExecuteCurrKeyHandler((S16) (KEY_0 + key_id), key_type);
    }
```

判断如果按键在数字键区[1，*)之内，播放按键音，执行当前按键事件句柄ExecuteCurrKeyHandler()，开始单步跟踪。

此时currFuncPtr值如下：

currFuncPtr	0xcccccccc

初始化后，赋值

```
FuncPtr currFuncPtr = NULL;
```

currFuncPtr	0x00000000

再给把当前按键值与类型赋给currKeyCode和currKeyType，赋值前：

currKeyCode	12
currKeyType	1

```
currKeyCode = keyCode;//keyCode=Key_0+key_id
currKeyType = keyType;//key_type
```

赋值后：

currKeyCode	8
currKeyType	0

接下来一个判断语句

```
if (!gEmerencyNoDialed)
{
   if (mmi_kbd_get_concurrent_mode() ==
   {
      /* special handling for single-key mode */
      switch (keyType)
      {
         case KEY_HALF_PRESS_DOWN:
            pressKey = HALF_DOWN_STATUS;
            break;
         case KEY_EVENT_DOWN:
            /*
             * The application may call ExecuteCu
             * MMI framework use KEY_HALF_PRE
             */
            pressKey = FULL_DOWN_STATUS;
            processUpRepeatKey = MMI_TRUE;
            break;
```

pressKey与processUpRepeatKey赋值前值为

pressKey	0
processUpRepeatKe	0

赋值后值为(2- full down，所以此处赋值为2)：

pressKey	2
processUpRepeatKe	1

此时，给currFuncPtr赋值，赋值前地址：

currFuncPtr	0x00000000

```
cuncPtr = currKeyFuncPtrs[keyCode][keyType];
```

赋值后地址：

currFuncPtr	0x0098df54 MMI_key_8_down(void)

至此，得到cuncPtr按键地址，指向8

```
(*currFuncPtr)();
```

*currFuncPtr	0x0098df54 MMI_key_8_down(void)

此时currFuncPtr指针指向MMI_key_8_down(void)，按键8按下后的执行事件

```
MMI_key_input_handler(KEY_8, KEY_EVENT_DOWN);
```

跟进去，MMI_key_input_handler

```
switch (key_event)
{
    case KEY_EVENT_DOWN:
        if (MMI_key_down_handler_function != NULL)
        {
            MMI_key_down_handler_function(key_code);
        }
        break;
```

```
MMI_key_down_handler_function | 0x009845a6 dialer_inputbox_handle_key_down(int)
```

跟进去，进入 dialer_inputbox_handle_key_down()函数，此函数执行按键按下后，拨号输入框执行的函数

```
dialer_inputbox_direct_input((UI_character_type) ('0' + k));
```

跟进去，dialer_inputbox_direct_input()

```
gui_dialer_input_box_insert_character(&MMI_dialer_inputbox, c);
```

gui_dialer_input_box_insert_character()，在当前光标处插入字符

跟进去，

```
b->change_callback();
```

```
b->change_callback | 0x00984344 dialer_input_box_change_callback(void)
```

dialer_input_box_change_callback()

```
gui_show_dialer_input_box(&MMI_dialer_inputbox)
```

跟进去，显示拨号数字框

```
gui_show_dialer_input_box_ext(b, (-1), (-1));
```

在gui_show_dialer_input_box_ext()函数里设置显示的数字的坐标，颜色等属性，原来显示字符的地方，用定义的图片替换。图片显示函数gui_show_image，获取显示的字符图的函数 dialer_input_get_image()，通过获取 current_character 的值，来得到图片 ID

```
#ifdef __DIALFONT_BY_IMAGE__
  gui_show_image(text_x, text_y, dialer_input_get_image(current_character));
#else
  gui_print_character(current_character);
#endif
```

2. DIALING_SCREEN （如何画拨号界面）

进入 ShowCategory16Screen()

3) wgui_setup_dialer_inputbox() 画拨号界面输入框

```
#elif defined (__MMI_MAINLCD_176X220__)
  wgui_setup_dialer_inputbox(
    0,
    MMI_status_bar_height,
    UI_device_width,
    29,
    (UI_buffer_type) Buffer,
    BufferLength,
    MMI_CATEGORY16_ID,
    get_string(right_softkey),
    get_image(right_softkey_icon),
    history_buffer,
    0);
```

4) 产生触摸屏事件后，拨号界面显示(例：触摸屏按下按钮 8 后，产生一个 MMI_PEN_EVENT_DOWN 事件，显示数字键 8 的高亮图片)

◆注册触摸屏事件(当触摸屏产生 MMI_PEN_EVENT_DOWN、MMI_PEN_EVENT_UP、MMI_PEN_EVENT_MOVE、MMI_PEN_EVENT_REPEAT、MMI_PEN_EVENT_LONG_TAP、MMI_PEN_EVENT_ABORT 等动作时，执行的函数)

```
#ifdef __MMI_TOUCH_DIAL_SCREEN__
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenDownHandler,
    MMI_PEN_EVENT_DOWN);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenUpHandler, MMI_PEN_EVENT_UP);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenMoveHandler,
    MMI_PEN_EVENT_MOVE);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenRepeatHandler,
    MMI_PEN_EVENT_REPEAT);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenLongTapHandler,
    MMI_PEN_EVENT_LONG_TAP);
  wgui_register_category_screen_control_area_pen_handlers(
    Cate16CategoryControlAreaPenAbortHandler,
    MMI_PEN_EVENT_ABORT);
#endif /* __MMI_TOUCH_DIAL_SCREEN__ */
```

◆以 MMI_PEN_EVENT_DOWN 事件为例：

进入 Cate16CategoryControlAreaPenDownHandler()函数

```
   /* Get dialing screen  event type according to current state of dialing screen */
ret = gui_dialing_screen_translate_pen_event(
      &dialing_keypad,
      point.x,
      point.y,
      MMI_PEN_EVENT_DOWN,
      &dialing_key_menu_event,
      &dialing_key_param);
```

通过 point.x 和 point.y 获得刚刚点击的触摸屏的 X、Y 坐标(此处在点击前，先记住触摸屏点击时的坐标，此处点击坐标为 82,159)。

gui_dialing_screen_translate_pen_event() 拨号界面的五个触摸事件 MMI_PEN_EVENT_DOWN 、 MMI_PEN_EVENT_UP 、 MMI_PEN_EVENT_MOVE 、 MMI_PEN_EVENT_LONG_TAP、**MMI_PEN_EVENT_REPEAT,** 当按键按下的事件产生后，确定是哪个按钮被按下，按钮显示位置多少

gui_dialing_screen_translate_pen_event()流程如下：

1> 首先用 gui_dialing_screen_translate_pen_position()函数得到按键的序号还有按键图片 X，Y 坐标

现在以按下数字键 8，触摸坐标为(89，129)为例，理解如何将数字键 8 的高亮图画出来

```
total_height += dialing_keypad->keypad_y;
total_width += dialing_keypad->keypad_x;
*item_index = -1;  /* 053005 Calvin added for function key */
```

开始进入 for 循环，此时 total_height=62，total_width=3 为 wgui.h 里设置的拨号键起始坐标。

此时开始第一次判断

```
for (i = 0; i < dialing_keypad->n_rows; i++)
{
    total_height += dialing_keypad->key_height;
    if (total_height > y)
    {
        for (j = 0; j < dialing_keypad->n_column; j++)
        {
            total_width += dialing_keypad->key_width;
            if (total_width > x)
            {
```

total_height	62
y	129

此时，total_heigh<y，表明不是在第一行的按钮，所以给 total_height 加一个按钮的 height（25），此时 total_height 值为 87

total_height	87
y	129

接着再往下走，total_height 再加了 VERTICAL_GAP（2），此时 total_height 值为 89

total_height	89
y	129

```
else
{
    total_height += dialing_keypad->vertical_gap;
    if (total_height > y)
    {
        *item_index = -1;
        break;
    }
```

再进行判断 total_height<y，跳出，返回到顶上的 FOR 循环，再给 total_height 加上一个按钮的高度(25)，再进行判断，还是小于 y（114<129），再跳出，再加上 VERTICAL_GAP（2），再判断，再加一个图片的高度。

total_height	114
y	129

此时，total_height 值为 116+25=141，判断成立，跳入，执行下一句(此时 i 值为 2)。开始 total_width 判断是第三行的第几列的图片被选中

```
for (j = 0; j < dialing_keypad->n_column; j++)
{
    total_width += dialing_keypad->key_width;
    if (total_width > x)
    {
        *item_index = i * dialing_keypad->n_column + j + 1;
```

此时，total_widht 即为拨号按键起始坐标，值为 3，开始循环后，加了一个图片的宽度 55，total_width 值为 58

total_width	58
x	89

判断不成立，跳转出来，加一个 horizontal_gap(3)，total_width 值变为 61，再判断，不成立，再跳转出来。

```
else
{
    total_width += dialing_keypad->horizontal_gap;
    if (total_width > x)
    {
        *item_index = -1;
        break;
    }
}
```

再加一个图片的宽度(55)，此时 total_width 值为 116，判断成立(此时 j 值为 1)，执行下面语句

total_width	116
x	89

下面的语句获取按键的序号和画按键的 x,y 值

```
{
  *item_index = i * dialing_keypad->n_column + j + 1;
  dialing_keypad->selected_key_x =
    dialing_keypad->keypad_x + (dialing_keypad->key_width) * j +
    (dialing_keypad->horizontal_gap) * j;
  dialing_keypad->selected_key_y =
    dialing_keypad->keypad_y + (dialing_keypad->key_height) * i +
    (dialing_keypad->vertical_gap) * i;
  dialing_keypad->key_type = *item_index;
  break;
}
```

```
*item_index = i * dialing_keypad->n_column + j + 1;//2*3+1+1=8
dialing_keypad->selected_key_x =3+55*1+3*1=61
dialing_keypad->selected_key_y=62+25*2+2*2=116
```

2> gui_dialing_screen_translate_pen_position 的函数流程：

```
switch (pen_event)
{
  case MMI_PEN_EVENT_DOWN:
    if (PEN_CHECK_BOUND(x, y, x1, y1, dialing_keypad->keypad_width, dialing_keypad->keypad_height))
    {
      gui_dialing_screen_translate_pen_position(dialing_keypad, x, y, &item_index);
      if (item_index == -1)  /* No key is selected 没 有 按 键 被 选 中 */
      {
        ret = FALSE;
      }
  #ifdef __MMI_TOUCH_DIAL_SCREEN_WITH_FUNCTION__
      if (item_index == MMI_DIALING_KEY_PHB
        && fun_key_phb == 0)
```

……

```
    else   //当 排 除 了 上 述 情 况 后 ， 画 按 下 的 按 钮
    {
      gui_dialing_key_select(dialing_keypad, item_index);
    }
} ? end if PEN_CHECK_BOUND(x,y,x... ?
else  //如 果 按 触 摸 按 键 不 在 有 效 范 围 内 时
{
  ret = FALSE;
}
break;
```

当触摸屏产生一个 MMI_PEN_EVENT_DOWN 事件后，获取按键的序号与坐标(gui_dialing_screen_translate_pen_position)，先判断触摸范围是不是在有效的范围之内(是否在拨号按键的长宽范围之内)，然后判断按键是否被选中，电话簿、SIM1、SIM2 卡是不是在处在 Disable 状态，排除以上状态后，获取按键序号，然后，画出来(gui_dialing_key_select)

3>在 gui_dialing_key_select 中开始画高亮数字键

```
x1 = dialing_keypad->selected_key_x;
y1 = dialing_keypad->selected_key_y;
/* 053005 Calvin added for function key */
x2 = x1 + key_width - 1;
y2 = y1 + key_height - 1;
```

给 x1,y1 分给赋值为 61,116,x2,y2 赋值为 85,170

```
gdi_layer_push_clip();
gdi_layer_set_clip(x1, y1, x2, y2);

if (0<= (item_index-1) && (item_index-1) < sizeof(dialing_key_image) / sizeof(dialing_key_image[0]))
{
   image_id = dialing_key_image[item_index - 1];
   gui_measure_image(get_image(image_id), &width, &height);
   /* 053005 Calvin added for function key */
   x1 = x1 + ((key_width - width) >> 1);
   y1 = y1 + ((key_height - height) >> 1);
   /* Calvin end */
   gui_show_image(x1, y1, get_image(image_id));
}
```

再判断按键序号在范围之内，然后取其对应的图片赋给 image_id,然后将图片 show 出来，至此，拨号键 8 显示完成

补记：

将 wgui.h 中的设置的各个坐标的值赋值给结构变量 dialing_keypad 的各个成员变量

结构申明如下：（结构_dialing_keypad_struct 别名为 dialing_keypad_struct）

```
typedef struct _dialing_keypad_struct
{
   S32 keypad_x, keypad_y;
   S32 keypad_width, keypad_height;
   S32 key_width, key_height;
   S32 horizontal_gap, vertical_gap;
   S32 selected_key_x, selected_key_y;
   S32 n_rows, n_column;
#ifdef __MMI_TOUCH_DIAL_SCREEN_WITH_FUNCTION__
   S32 keypad_func_x, keypad_func_y;
   S32 func_key_width, func_key_height;
   S32 func_horizontal_gap, func_vertical_gap;
   S32 func_n_rows, func_n_column;
#endif /* __MMI_TOUCH_DIAL_SCREEN_WITH_FUNCTION__ */
   S32 key_type;   /* selected key index */
#ifndef __GDI_MEMORY_PROFILE_2__
   gdi_image_cache_bmp_struct selected_key_bitmap;
#endif
} dialing_keypad_struct;
```

定义结构变量 dialing_keypad

```
#ifdef __MMI_TOUCH_DIAL_SCREEN__
dialing_keypad_struct dialing_keypad;

#endif /* __MMI_TOUCH_DIAL_SCREEN__ */
```

show_dialing_screen()画拨号界面背景图 IMG_DIALING_SCREEN