M851 WristApp API Reference Guide

Timex Corporation
January 26, 2004
## DOCUMENT REVISION HISTORY

<table>
<thead>
<tr>
<th>REVISION</th>
<th>DATE</th>
<th>Niño Aldrin L. Sarmiento</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>07/18/2002</td>
<td>Created document.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFFECTED PAGES</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>Created document.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REVISION</th>
<th>DATE</th>
<th>Niño Aldrin L. Sarmiento</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>04/09/2003</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFFECTED PAGES</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>114</td>
<td>Updated example document for API LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REVISION</th>
<th>DATE</th>
<th>Niño Aldrin L. Sarmiento</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>05/27/2003</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFFECTED PAGES</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>229</td>
<td>Updated example document for API KSTP_COPY_RESOURCE_TO_BUFFER</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REVISION</th>
<th>DATE</th>
<th>Niño Aldrin L. Sarmiento</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.3</td>
<td>07/10/2003</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFFECTED PAGES</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>117</td>
<td>Remove a line in sample code for the API LCD_DISP_BIG_2DIGIT_DM_DATA_NO_ZERO_SUP</td>
</tr>
<tr>
<td>61-73</td>
<td>Updated examples for database APIs</td>
</tr>
<tr>
<td>127</td>
<td>Update example for LCD_SET_BLINK_FLAG_STATUS</td>
</tr>
<tr>
<td>196-197</td>
<td>Updated example for KTOD_SUBTRACT_DAYS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REVISION</th>
<th>DATE</th>
<th>Niño Aldrin L. Sarmiento</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
<td>01/26/2004</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AFFECTED PAGES</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>57</td>
<td>Change IY to HL as the base address of new melody. Updated sample code.</td>
</tr>
</tbody>
</table>
# TABLE OF CONTENTS

Introduction ........................................................................................................................................... 4  
Document Convention ............................................................................................................................ 4  
API Summary ....................................................................................................................................... 4  
API Reference ...................................................................................................................................... 17  
   CORE API ...................................................................................................................................... 17  
      Changing Foreground Application ............................................................................................... 17  
      Changing Application Foreground State Handler ......................................................................... 18  
      Requesting Timeouts ..................................................................................................................... 19  
      Peeking At Background Application Data .................................................................................. 22  
      Modifying System Operation ....................................................................................................... 23  
      Blinking and Scrolling Operations ............................................................................................... 24  
      Primary Mode Icon Operations .................................................................................................. 27  
      Switch, Ring and Crown Operations ............................................................................................ 28  
      Background Task Operations ....................................................................................................... 37  
      Popup Operations ......................................................................................................................... 43  
      Utilities ...................................................................................................................................... 50  
   AUDIO API .................................................................................................................................... 56  
   DATABASE API ............................................................................................................................... 59  
      Open and Close Database Operations ......................................................................................... 59  
      Database Information Header Operations ................................................................................... 60  
      Record Read and Write Operations ............................................................................................. 63  
      Linked-List Operations .................................................................................................................. 73  
   DISPLAY API ................................................................................................................................ 76  
      Character Bitmap Patterns ............................................................................................................. 84  
      Segment Display Patterns .............................................................................................................. 84  
      Small Font Dot-Matrix Patterns ...................................................................................................... 85  
      Large Font Dot-Matrix Patterns ..................................................................................................... 89  
      Flag Display ................................................................................................................................ 90  
      Clearing Display ........................................................................................................................... 98  
      Clearing Large-Font Digits ............................................................................................................. 100  
      Clearing Display Regions .............................................................................................................. 101  
      Data Formatting ............................................................................................................................ 104  
      Character Display .......................................................................................................................... 107  
      Large-Font Character Display ....................................................................................................... 109  
      Numeric Display ............................................................................................................................ 109  
      Large-Font Numeric Display ......................................................................................................... 118  
      Message Display ............................................................................................................................ 121  
      Banner Message Display .............................................................................................................. 124  
      Large-Font Message Display ......................................................................................................... 125  
      Icon/Flag Blinking ........................................................................................................................... 127  
      Blinking ...................................................................................................................................... 129  
      Scrolling ..................................................................................................................................... 133  
      Primary Mode Icon Resource ....................................................................................................... 136  
      Pixel Operations ............................................................................................................................ 137
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display Canned Messages</td>
<td>139</td>
</tr>
<tr>
<td>Utilities</td>
<td>185</td>
</tr>
<tr>
<td>TIME ZONE RESOURCE API</td>
<td>189</td>
</tr>
<tr>
<td>Flag Manipulation</td>
<td>190</td>
</tr>
<tr>
<td>Data Manipulation</td>
<td>194</td>
</tr>
<tr>
<td>Data Transfer</td>
<td>200</td>
</tr>
<tr>
<td>Resource Utilities</td>
<td>205</td>
</tr>
<tr>
<td>TIME ZONE CHECK RESOURCE API</td>
<td>211</td>
</tr>
<tr>
<td>Flag Manipulation</td>
<td>211</td>
</tr>
<tr>
<td>Data Manipulation</td>
<td>214</td>
</tr>
<tr>
<td>BACKUP RESOURCE API</td>
<td>220</td>
</tr>
<tr>
<td>Flag Manipulation</td>
<td>220</td>
</tr>
<tr>
<td>Data Manipulation</td>
<td>223</td>
</tr>
<tr>
<td>STOPWATCH RESOURCE API</td>
<td>225</td>
</tr>
<tr>
<td>Data Manipulation</td>
<td>226</td>
</tr>
<tr>
<td>Data Transfer</td>
<td>232</td>
</tr>
<tr>
<td>Resource Utilities</td>
<td>232</td>
</tr>
<tr>
<td>TIMER RESOURCE API</td>
<td>235</td>
</tr>
<tr>
<td>Data Manipulation</td>
<td>236</td>
</tr>
<tr>
<td>Data Transfer</td>
<td>246</td>
</tr>
<tr>
<td>Resource Utilities</td>
<td>247</td>
</tr>
<tr>
<td>SYNCHRO RESOURCE API</td>
<td>249</td>
</tr>
<tr>
<td>Data Manipulation</td>
<td>250</td>
</tr>
<tr>
<td>Data Transfer</td>
<td>256</td>
</tr>
<tr>
<td>UTILITIES API</td>
<td>258</td>
</tr>
<tr>
<td>Conversion</td>
<td>261</td>
</tr>
<tr>
<td>Math Functions</td>
<td>265</td>
</tr>
<tr>
<td>Copy</td>
<td>269</td>
</tr>
<tr>
<td>Comparison</td>
<td>270</td>
</tr>
<tr>
<td>Acceleration</td>
<td>273</td>
</tr>
<tr>
<td>Message Editing</td>
<td>276</td>
</tr>
<tr>
<td>Time Structure Math</td>
<td>284</td>
</tr>
<tr>
<td>Trademarks</td>
<td>291</td>
</tr>
</tbody>
</table>
Introduction
This reference guide is divided into two parts:

- “API Summary” provides a quick reference for looking up the API required to handle specific functions.
- “Detailed API Reference” provides the usage of an API along with its required input parameters, assumptions, expected output and microcontroller registers affected by the operation.

NOTE: The information in this reference guide is the best available at the time of release. This will be updated from time to time.

Document Convention
This manual uses the following typographic conventions.

<table>
<thead>
<tr>
<th>Example of Convention</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>{ ON</td>
<td>OFF }</td>
</tr>
<tr>
<td>{ expression_list }</td>
<td>In syntax, items in braces and in italics should be replaced with a constant.</td>
</tr>
</tbody>
</table>

API Summary

CORE API

Changing Foreground Application
- CORE_REQ_MODE_CHANGE
- CORE_REQ_MODE_CHANGE_NEXT
- CORE_REQ_MODE_CHANGE_NEXT_NO_PEEK
- CORE_FORCE_PRIMARY_AS_NEXT_MODE

Changing Application Foreground State Handler
- CORE_REQ_STATE_CHANGE
- CORE_REQ_STATE_CHANGE_NO_CLEAR_DISPLAY

Requesting Timeouts
- CORE_REQ_TIMEOUT_HIRES
- CORE_REQ_TIMEOUT_LORES
- CORE_REQ_TIMEOUT_STICKY
- CORE_CANCEL_TIMEOUTS

Peeking At Background Application Data
- CORE_REQ.Peek_APP_TYPE
Modifying System Operation
CORE_SET_AUTORETURN
CORE_CLEAR_AUTORETURN

Blinking and Scrolling Operations
CORE_REQ_BLINK_2HZ
CORE_CANCEL_BLINK_2HZ
CORE_REQ_BLINK_4HZ
CORE_CANCEL_BLINK_4HZ
CORE_CANCEL_SCROLLING

Primary Mode Icon Operations
CORE_BACKGROUND_ICON_REFRESH_ENABLE
CORE_BACKGROUND_ICON_REFRESH_DISABLE

Switch, Ring and Crown Operations
CORE_ALLOW_KEYS
CORE_MASK_KEYS
CORE_ALLOW_ALL_KEYMASK
CORE_SUSPEND_RING_EVENTS
CORE_ENABLE_RING_EVENTS
CORE_SUSPEND_RING_LEADING_EVENTS
CORE_ENABLE_RING_LEADING_EVENTS
CORE_ENABLE_SWITCH_RELEASE
CORE_SUSPEND_SWITCH_RELEASE
CORE_ENABLE_GENERIC_SWITCH_EVENT
CORE_DISABLE_GENERIC_SWITCH_EVENT
CORE_REQUEST_MELODY_POPUP_CANCEL
CORE_ENABLE_PULSE_MODE
CORE_DISABLE_PULSE_MODE
HW_KBDCANCEL_CURRENT_SWITCH_RELEASE

Background Task Operations
CORE_REQ_BACKGROUND_TASK
CORE_REQ_BACKGROUND_TASK_WITH_PRIORITYCHECK
CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE
CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE_WITH_PRIORITYCHECK
CORE_REQ_BACKGROUND_TASK_FOR_PTZBASEDAPPS
CORE_CLEAR_BACKGROUND_APPLICATION_TASK
CORE_CLEAR_ALL_BACKGROUND_APPLICATION_TASK

Popup Operations
CORE_REQ_POPDOWN
CORE_SET_POPDOWN_STATE
CORE_ENABLE_POPUPS
CORE_SUSPEND_POPUPS
CORE_CANCEL_ALL_POPUPS
CORE_CANCEL_ANY_PTZ_POPUPS
CORETASK_POPUP_REQUEST_BCK
CORETASK_POPUPCANCEL_BCK
CORETASK_POPUP_REQUEST_TZC
CORETASK_POPUPCANCEL_TZC
CORETASK_POPUP_REQUEST_TMR
Utilities
CORE_ENABLE_SET_MODE
CORE_DISABLE_SET_MODE
CORE_SET_RANDOM_SEED
CORE_GENERATE_RANDOM_NUMBER
CORE_DATABASE_MODIFIED_BY_USER
CORE_APP_ID_PRESENT
CORE_GET_APP_INDEX
CORE_GET_APP_INDEX_FROM_APP_TYPE

AUDIO API
AUDSTART_SYSTEM_MELODY
AUDSETUP_MELODYADDRESS
AUDSTOP_MELODY

DATABASE API
Open and Close Database Operations
DB_OPEN_FILE
DB_OPEN_FILE_LINK_LIST
DB_CLOSE_FILE

Database Information Header Operations
DB_READ_APPLICATION_INFOHEADER
DB_WRITE_APPLICATION_INFOHEADER
DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMVAR
DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMFIX

Record Read and Write Operations
DB_WRITE_RECORD
DB_WRITE_RECORD_WITHOFFSET
DB_WRITE_RECORD_RANDOMFIX
DB_WRITE_RECORD_WITHOFFSET_RANDOMFIX
DB_WRITE_RECORD_RANDOMVAR
DB_WRITE_RECORD_WITHOFFSET_RANDOMVAR
DB_READ_RECORD
DB_READ_RECORD_WITHOFFSET
DB_READ_RECORD_RANDOMFIX
DB_READ_RECORD_WITHOFFSET_RANDOMFIX
DB_READ_RECORD_RANDOMVAR
DB_READ_RECORD_WITHOFFSET_RANDOMVAR

Linked-List Operations
DB_REMOVE_RECORD_LINKLIST
DB_INSERT_RECORD_LINKLIST
DB_LOCATE_INSERTION_BYSIZE_LINKLIST
DISPLAY API

Flag Display
- LCD_CLR_ALL_FLAGS
- LCD_UPD_L_FLAG
- LCD_UPD_A_FLAG
- LCD_UPD_P_FLAG
- LCD_UPD_LAP_FLAG
- LCD_UPD_AP_FLAG
- LCD_UPD_NOTE_FLAG
- LCD_UPD_ACLK_FLAG
- LCD_UPD_TAIL_FLAG
- LCD_UPD_ARROW_FLAG
- LCD_UPD_TMR_FLAG
- LCD_UPD_CDC_FLAG
- LCD_UPD_CDR_FLAG
- LCD_UPD CHR_FLAG
- LCD_UPD MOON_FLAG
- LCD_UPD DASH1_FLAG
- LCD_UPD PER1_FLAG
- LCD_UPD COLON1_FLAG
- LCD_UPD DASH2_FLAG
- LCD_UPD PER2_FLAG
- LCD_UPD COLON2_FLAG

Clearing Display
- LCD_CLR_2DIGIT_SEG
- LCD_CLR_3DIGIT_SEG
- LCD_CLR_SMALL_PROP_WIDTH_2DIG_DM_DATA
- LCD_CLR_SMALL_FIXED_WIDTH_2DIG_DM_DATA
- LCD_CLR_SMALL_PROP_WIDTH_3DIG_DM_DATA
- LCD_CLR_SMALL_FIXED_WIDTH_3DIG_DM_DATA

Clearing Large-Font Digits
- LCD_CLR_BIG_2DIGIT_DM_DATA
- LCD_CLR_BIG_3DIGIT_DM_DATA

Clearing Display Regions
- LCD_CLR_DISPLAY
- LCD_FILL_DISPLAY
- LCD_CLR_SEG_LINE
- LCD_CLEAR_UPPER_DM
- LCD_CLEAR_MAIN_DM
- LCD_CLR_MAIN_DM_LINE1
- LCD_CLR_MAIN_DM_LINE2

Data Formatting
- LCD_FORMAT_3DIGIT_DATA_SUP_ZERO
- LCD_FORMAT_2DIGIT_DATA_SUP_ZERO
- LCD_FORMAT_3DIGIT_DATA_NO_LSD_SUP
- LCD_FORMAT_2DIGIT_DATA_SUP_ZERO_MSD
- LCD_FORMAT_3DIGIT_DATA_NO_ZERO_SUP
- LCD_FORMAT_2DIGIT_DATA_NO_ZERO_SUP
Character Display
LCD_DISP_SEG_CHAR
LCD_DISP_SMALL_PROP_WIDTH_DM_CHAR
LCD_DISP_SMALL_FIXED_WIDTH_DM_CHAR

Large-Font Character Display
LCD_DISP_BIG_DM_CHAR

Numeric Display
LCD_DISP_2DIG_SEG_DATA_WITHZERO_SUP
LCD_DISP_3DIG_SEG_DATA_WITHZERO_SUP
LCD_DISP_2DIG_SEG_DATA_SUP_ZERO_MSD
LCD_DISP_3DIG_SEG_DATA_NO_LSD_SUP
LCD_DISP_2DIG_SEG_DATA_NO_ZERO_SUP
LCD_DISP_3DIG_SEG_DATA_NO_ZERO_SUP
LCD_DISP_4DIG_SEG_DATA_WITHZERO_SUP
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO
LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_SUP_ZERO
LCD_DISP_SMALL_PROP_WIDTH_3DIG_DM_DATA_SUP_ZERO
LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_SUP_ZERO
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO_MSD
LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_SUP_ZERO_MSD
LCD_DISP_SMALL_PROP_WIDTH_3DIG_DM_DATA_NO_LSD_SUP
LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_LSD_SUP
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP
LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP

Large-Font Numeric Display
LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO
LCD_DISP_BIG_3DIGIT_DM_DATA_SUP_ZERO
LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO_MSD
LCD_DISP_BIG_3DIGIT_DM_DATA_NO_LSD_SUP
LCD_DISP_BIG_2DIGIT_DM_DATA_NO_ZERO_SUP
LCD_DISP_BIG_3DIGIT_DM_DATA_NO_ZERO_SUP

Message Display
LCD_DISP_SEG_LINE_MSG
LCD_DISP_FORMATTED_SMALL_PROP_WIDTH_DM_MSG
LCD_DISP_FORMATTED_SMALL_FIXED_WIDTH_DM_MSG
LCD_DISP_UNFORMATTED_SMALL_PROP_WIDTH_DM_MSG
LCD_DISP_UNFORMATTED_SMALL_FIXED_WIDTH_DM_MSG

Banner Message Display
LCD_DISP_BANNER_MSG

Large-Font Message Display
LCD_DISP_FORMATTED_BIG_FONT_DM_MSG
LCD_DISP_UNFORMATTED_BIG_FONT_DM_MSG

Icon/Flag Blinking
LCD_SET_BLINK_FLAG_STATUS
LCD_CLR_BLINK_FLAG_STATUS
LCD_CANCEL_ALL_BLINK_FLAGS

Blinking
LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_ADDR
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_ADDR
LCD_WRITE_4HZ_GEN_BLINK_POSITION
LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_PRELOADED
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_PRELOADED
LCD_WRITE_4HZ_GEN_BLINK_POSITION_PRELOADED
LCD_DISP_4HZ_DATA_FIRST

Scrolling
LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE1
LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE2
LCD_PAUSE_SCROLLING
LCD_RESUME_SCROLLING
LCD_SCROLL_MSG_LEFT
LCD_SCROLL_MSG_RIGHT

Primary Mode Icon Resource
LCD_UPDATE_TOD_FLAG_RESOURCE_STATE
LCD_UPDATE_TIMELINE_RESOURCE

Pixel Operations
LCD_DISPLAY_MAIN_DM_PIXEL
LCD_CLEAR_MAIN_DM_PIXEL
LCD_GET_STATE_OF_PIXEL

Display Canned Messages
LCD_DISP_UPPER_DM_MSG_ONE_HALF
LCD_DISP_UPPER_DM_MSG_TO
LCD_DISP_UPPER_DM_MSG_LO
LCD_DISP_SMALL_DM_MSG_HOLD_TO_SWITCH
LCD_DISP_MSG_NO_APPT_UPCOMING
LCD_DISP_SMALL_DM_MSG_NO_OCCASION_UPCOMING
LCD_DISP_SMALL_DM_MSG_SET_TIME_SLASH_DATE
LCD_DISP_SMALL_DM_MSG_SHOW_DAY_OF_WEEK
LCD_DISP_SMALL_DM_MSG_SHOW_WEEK_NUMBER
LCD_DISP_SMALL_DM_MSG_FREE_MEMORY_LAPS
LCD_DISP_SMALL_DM_MSG_TO_STORE_TURN_CROWN
LCD_DISP_SMALL_DM_MSG_MEMORY_FULL
LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLR_ALL
LCD_DISP_SMALL_DM_MSG_HOLD_TO_RESET
LCD_DISP_SMALL_DM_MSG_WORKOUT_STORED
LCD_DISP_SMALL_DM_MSG_RECALL_SLASH_FORMAT
LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR_WORKOUT
LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR
LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR_WORKOUT
LCD_DISP_SMALL_DM_MSG_RECALL_SLASH_FORMAT
LCD_DISP_SMALL_DM_MSG_HOLD_TO_RESET
LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLR_ALL
LCD_DISP_SMALL_DM_MSG_MEMORY_FULL
LCD_DISP_SMALL_DM_MSG_TO_STORE_TURN_CROWN
LCD_DISP_SMALL_DM_MSG_FREE_MEMORY_LAPS
LCD_DISP_SMALL_DM_MSG_SHOW_WEEK_NUMBER
LCD_DISP_SMALL_DM_MSG_SHOW_DAY_OF_WEEK
LCD_DISP_SMALL_DM_MSG_NO_OCCASION_UPCOMING
LCD_DISP_MSG_NO_APPT_UPCOMING
LCD_DISP_MSG_HOLD_TO_SWITCH
LCD_DISP_MAIN_DM_PIXEL
LCD_CLEAR_MAIN_DM_PIXEL
LCD_GET_STATE_OF_PIXEL

LCD_DISP_LARGE_DM_STOP
LCD_DISP_LARGE_DM_TOTAL
LCD_DISP_MSG_TIME_LAP
LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR
LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR_WORKOUT
LCD_DISP_SMALL_DM_MSG_MEMORY_FULL
LCD_DISP_SMALL_DM_MSG_TO_STORE_TURN_CROWN
LCD_DISP_SMALL_DM_MSG_FREE_MEMORY_LAPS
LCD_DISP_SMALL_DM_MSG_SHOW_WEEK_NUMBER
LCD_DISP_SMALL_DM_MSG_SHOW_DAY_OF_WEEK
LCD_DISP_SMALL_DM_MSG_NO_OCCASION_UPCOMING
LCD_DISP_MSG_NO_APPT_UPCOMING
LCD_DISP_MSG_HOLD_TO_SWITCH
LCD_DISP_MAIN_DM_PIXEL
LCD_CLEAR_MAIN_DM_PIXEL
LCD_GET_STATE_OF_PIXEL

LCD_DISP_LARGE_DM_STOP
LCD_DISP_LARGE_DM_TOTAL
LCD_DISP_MSG_TIME_LAP
LCD_DISP_SMALL_DM_MSG_SET_TIMER
LCD_DISP_SMALL_DM_MSG_UNUSED_ENTRY
LCD_DISP_SMALL_DM_MSG_UNUSED_ENTRIES
LCD_DISP_SMALL_DM_MSG_STOP_AT_END
LCD_DISP_SMALL_DM_MSG_REPEAT_AT_END
LCD_DISP_SMALL_DM_MSG_CHRONO_AT_END
LCD_DISP_SMALL_DM_MSG_SET_FORMAT
LCD_DISP_SMALL_DM_MSG_SET_COUNTER
LCD_DISP_SMALL_DM_MSG_COUNT_UP
LCD_DISP_SMALL_DM_MSG_COUNT_DOWN
LCD_DISP_SMALL_DM_MSG_HOLD_TO_DELETE
LCD_DISP_SMALL_DM_MSG_SET_ALARM
LCD_DISP_SMALL_DM_MSG_ENTRY_DELETED
LCD_DISP_SMALL_DM_MSG_SET_APPT
LCD_DISP_SMALL_DM_MSG_APPT_DATE
LCD_DISP_SMALL_DM_MSG_1ST_APPT_DATE
LCD_DISP_SMALL_DM_MSG_EDIT_NOTE
LCD_DISP_SMALL_DM_MSG_BUTTON_BEEP_ON
LCD_DISP_SMALL_DM_MSG_BUTTON_BEEP_OFF
LCD_DISP_SMALL_DM_MSG_NIGHTMODE_OFF
LCD_DISP_SMALL_DM_MSG_NIGHTMODE_ON
LCD_DISP_SMALL_DM_MSG_NIGHTMODE_AUTO
LCD_DISP_SMALL_DM_MSG_CHIME_OFF
LCD_DISP_SMALL_DM_MSG_CHIME_ON
LCD_DISP_SMALL_DM_MSG_CHIME_AUTO
LCD_DISP_SMALL_DM_MSG_ON_OFF_TIME
LCD_DISP_SMALL_DM_MSG_SORTING_TRY_LATER
LCD_DISP_SMALL_DM_MSG_ENTER_PASSWORD
LCD_DISP_SMALL_DM_MSG_PASSWORD_INVALID
LCD_DISP_SMALL_DM_MSG_PASSWORD
LCD_DISP_SMALL_DM_MSG_NO_ENTRY_SELECTED
LCD_DISP_SMALL_DM_MSG_SNOOZE
LCD_DISP_SMALL_DM_MSG_END_OF_LIST
LCD_DISP_SMALL_DM_MSG_EDIT
LCD_DISP_SMALL_DM_MSG_ON_TIME
LCD_DISP_SMALL_DM_MSG_OFF_TIME
LCD_DISP_SMALL_DM_MSG_COMM_READY
LCD_DISP_SMALL_DM_MSG_COMM_ERROR
LCD_DISP_SMALL_DM_MSG_PUSH_CROWN_IN
LCD_DISP_SEG_MSG_12HR
LCD_DISP_SEG_MSG_24HR
LCD_DISP_SEG_MSG_FREE
LCD_DISP_SEG_MSG_CHRONO
LCD_DISP_SEG_MSG_INT
LCD_DISP_SEG_MSG_HR_MIN
LCD_DISP_SEG_MSG_SECOND
LCD_DISP_SEG_MSG_TOTAL
LCD_DISP_SEG_MSG_DAILY
LCD_DISP_SEG_MSG_PUSHPULL
LCD_DISP_SEG_MSG_WKENDS
LCD_DISP_SEG_MSG_WEEKLY
LCD_DISP_SEG_MSG_MNTHLY
LCD_DISP_SEG_MSG_YEARLY
LCD_DISP_SEG_MSG_ALARM
LCD_DISP_SEG_MSG_1_DAY
LCD_DISP_SEG_MSG_MMDYY
LCD_DISP_SEG_MSG_DDMMYY
LCD_DISP_SEG_MSG_YYMMDD
LCD_DISP_SEG_MSG_BATT
LCD_DISP_SEG_MSG_HOLD
LCD_DISP_SEG_MSG_STORE
LCD_DISP_SEG_MSG_TIME
LCD_DISP_SEG_MSG_1_MIN
LCD_DISP_SEG_MSG_ALERT
LCD DISP_SEG_MSG_SHOW
LCD_DISP_SEGMSG_COUNT
LCD_DISP_SEG_MSG_SET
LCD_DISP_SEG MSG STOP
LCD_DISP_MAIN_DM_LINE1_SELECT
LCD_DISP_MAIN_DM_LINE1_ALARM
LCD_DISP_MAIN_DM_LINE2_ALARM
LCD_DISP_MAIN_DM_LINE1_ALARM_AT
LCD_DISP_MAIN_DM_LINE2_TZ
LCD_DISP_MAIN_DM_LINE2_LAP
LCD_DISP_MAIN_DM_LINE2_LAPS
LCD_DISP_MAIN_DM_LINE1_MEMORY
LCD DISP MAIN DM_LINE2_BEST_LAP
LCD DISP MAIN DM LINE2_LAP_AVG
LCD DISP MAIN DM_LINE1_APPT_AT
LCD DISP MAIN DM_LINE1_HOURS
LCD DISP MAIN DM_LINE1_MINS
LCD DISP MAIN DM_LINE2_PRIOR
LCD DISP MAIN DM_LINE2_MINS
LCD DISP MAIN DM_LINE1_BDAY
LCD DISP MAIN DM_LINE1_ANNV
LCD DISP MAIN DM LINE1_HOLIDAY
LCD DISP MAIN DM_LINE1_VACATION
LCD DISP MAIN DM_LINE1_4_DASHES
LCD DISP MAIN DM_LINE1_CHIME
LCD DISP MAIN DM_LINE2_ON
LCD DISP MAIN DM_LINE2_OFF
LCD DISP MAIN DM_LINE2_AUTO
LCD DISP MAIN DM_LINE1_YEAR
LCD DISP MAIN DM_LINE1_AM
LCD DISP MAIN DM_LINE1_PM
LCD DISP MAIN DM_LINE2_AM
LCD DISP MAIN DM LINE2_PM

Utilities
UTL_DISPLAY_DAY_OF_WEEK
UTL_DISPLAY_DATE_COMPLETE_AND_DOW
UTL_DISPLAY_DATE_COMPLETE
UTL_DISPLAY_HOUR_MIN_DATA_L1
UTL_DISPLAY_HOUR_MIN_DATA_L2

TIME ZONE RESOURCE API

Flag Manipulation
KTOD_ACTIVATERESOURCE
KTOD_DEACTIVATE_RESOURCE
KTOD_ENABLE_DISP_UPD_SEC_EVENT
KTOD_DISABLE_DISP_UPD_SEC_EVENT
KTOD_ENABLE_DISP_UPD_MIN_EVENT
KTOD_DISABLE_DISP_UPD_MIN_EVENT
KTOD_DEACTIVATE_ALL_DISPLAY_UPDATES
KTOD_MAKE_AS_PRIMARY_TZ
KTOD_SET_FOR_EURO_FORMAT
KTOD_SET_FOR_US_FORMAT

Data Manipulation
KTOD_CALC_DOW
KTOD_CALC_WEEK_NUMBER
KTOD_ADD_DAYS
KTOD_SUBTRACT_DAYS
KTOD_ADD_YEARS
KTOD_SUBTRACT_YEARS
KTOD_CORRECT_MDY_DATA
KTOD_ADJUST_DATE_AND_YEAR

Data Transfer
KTOD_COPY_RESOURCE_TO_RESOURCE
KTOD_COPY_TIME_MIN_TO_DOW_FROM_RESOURCE
KTOD_COPY_TIME_FROM_RESOURCE
KTOD_COPY_MDY_FROM_RESOURCE
KTOD_COPY_HMS_FROM_RESOURCE
KTOD_WRITE_TIME_TO_RESOURCE
KTOD_WRITE_MDY_TO_RESOURCE
KTOD_WRITE_HMS_TO_RESOURCE

Resource Utilities
KTOD_LOAD_RESOURCE_START_ADDRESS
KTOD_GET_UPDATE_FLAGS
KTOD_CHECK_IF_LEAP_YEAR
KTOD_GET_MAX_DAYS_OF_A_MONTH
KTOD_DIVIDE_BY_2
KTOD_CONVERT_BCD_TO_HEX
KTOD_CHECK_IF_DIVISIBLE_BY_4
KTOD_GET_PTZ_ADDRESS
KTOD_GET_TZ_ADDRESS

TIME ZONE CHECK RESOURCE API

Flag Manipulation
KTZC_DEACTIVATE_ALL_RESOURCES
KTZC_ACTIVATE_RESOURCE
KTZC_DISABLE_RESOURCE
KTZC_SETUP_POPUP_GENERATION
KTZC_SETUP_EVENT_GENERATION
KTZC_CANCEL_POPUPEVENT_GENERATION

Data Manipulation
KTZC_RETURN_RESOURCE_OWNER
KTZC_SETUP_MDY_CHECK
KTZC_SETUP_HOUR_MINUTE_MDY
KTZC_SETUP_HOUR_MINUTE
KTZC_SETUP_MONTH
KTZC_SETUP_DATE
KTZC_SETUP_YEAR

BACKUP RESOURCE API

Flag Manipulation
- KBCK_DEACTIVATE_ALL_RESOURCES
- KBCK_ACTIVATE_RESOURCE
- KBCK_DEACTIVATE_RESOURCE
- KBCK_SETUP_POPUP_GENERATION
- KBCK_SETUP_EVENT_GENERATION
- KBCK_CANCEL_POPUP_EVENT_GENERATION

Data Manipulation
- KBCK_RETURN_RESOURCE_OWNER
- KBCK_SETUP_SNOOZE_TIME

STOPWATCH RESOURCE API

Data Manipulation
- KSTP_LINK_NEXT_RESOURCE
- KSTP_ENABLE_DISP_UPD_EVENT
- KSTP_DISABLE_DISP_UPD_EVENT
- KSTP_DEACTIVATE_ALL_DISPLAY_UPDATES
- KSTP_SETUP_POPUP_GENERATION
- KSTP_SETUP_EVENT_GENERATION
- KSTP_CANCEL_POPUP_EVENT_GENERATION
- KSTP_STOP_RESOURCE
- KSTP_STOP_ALL_RESOURCES
- KSTP_TAKE_SPLIT
- KSTP_RESET_RESOURCE
- KSTP_RESET_ALL_RESOURCE
- KSTP_CLEAR_RESOURCE_DATA

Data Transfer
- KSTP_COPY_RESOURCE_TO_BUFFER

Resource Utilities
- KSTP_COMPARE_BUFFER_FROM_RESOURCE
- KSTP_GET_RESOURCE_STATUS

TIMER RESOURCE API

Data Manipulation
- KTMR_SETUP_USER_AND_WORK_HMS
- KTMR_SETUP_WORKING_HMS
- KTMR_SETUP_PREWARNING_HMS
- KTMR_DISABLE_PREWARNING_HMS
- KTMR_DISABLE_PREWARNING_TRACKING
- KTMR_ENABLE_DISP_UPD_EVENT
KTMR_DISABLE_DISP_UPD_EVENT
KTMR_DEACTIVATE_ALL_DISPLAY_UPDATES
KTMR_SETUP_COUNTDOWN
KTMR_SET_RUNNING_REPEAT_COUNTER
KTMR_GET_RUNNING_REPEAT_COUNTER
KTMR_SETUP_POPUP_GENERATION
KTMR_SETUP_EVENT_GENERATION
KTMR_CANCEL_POPUP_AND_EVENT_GENERATION
KTMR_SETUP_REPEAT_AT_END
KTMR_CANCEL_REPEAT_AT_END
KTMR_SETUP_TMR_LINK
KTMR_SETUP_STP_LINK
KTMR_CANCEL_ALL_LINKS
KTMR_RESET_RESOURCE
KTMR_START_RESOURCE
KTMR_STOP_RESOURCE

Data Transfer
KTMR_COPY_RESOURCE_TO_BUFFER

Resource Utilities
KTMR_GET_STATUS
KTMR_GET_UPDATE_BITS

SYNCHRO RESOURCE API

Data Manipulation
KSYN_GET_CAUSED_TO_START
KSYN_GET_TOTAL_TIME_UPD_BITS
KSYN_GET_STOPPAGE_TIME_UPD_BITS
KSYN_GET_TOTAL_STATUS
KSYN_GET_STOPPAGE_STATUS
KSYN_ENABLE_DISP_UPD_EVENT
KSYN_DISABLE_DISP_UPD_EVENT
KSYN_SETUP_POPUP_GENERATION
KSYN_SETUP_EVENT_GENERATION
KSYNCANCEL_POPUP_EVENT_GENERATION
KSYN_RESET_AND_START
KSYN_STOP_TOTAL_AND_STOPPAGE_TIME
KSYN_RESET_RESOURCE

Data Transfer
KSYN_COPY_TOTAL_TO_BUFFER
KSYN_COPY_STOPPAGE_TO_BUFFER

UTILITIES API

Generic
KRES_CLEAR_N_BYTES
HW_RESET_MCU
HW_RESET_WATCHDOG
HW_LAMP_POPUP_REQUEST
HW_LAMP_POPUP_REQUEST_OFF
UTL_BINARY_MATH_MODE
UTL_DECIMAL_MATH_MODE

Conversion
UTL_CONVERT_HEX_TO_3DIGIT_BCD
UTL_CONVERT_HEX_TO_2DIGIT_BCD
UTL_CONVERT_2BYTE_HEX_TO_2BYTE_BCD
UTL_CONVERT_1BYTE_BCD_TO_1BYTE_HEX
UTL_CONVERT_4BYTE_FROM_BCD_TO_HEX
UTL_CONVERT_4BYTE_FROM_HEX_TO_BCD
UTL_CONVERT_TO_12HR_FORMAT

Math Functions
UTL_DIVIDE_HMSH_BY_N_IN_HEX
UTL_DIVIDE_HMSH_BY_N_IN_BCD
UTL_ADD_HMSH
UTL_SUBTRACT_HMSH

Copy
UTL_COPY_BUFFER
UTL_COPY_IYREG_TO_IXREG

Comparison
UTL_COMPARE_4BYTE_BUFFER
UTL_COMPARE_HLREG_WITH_IXREG
UTL_COMPARE_HLREG_WITH_IYREG

Acceleration
UTLACCELERATION_1SEC
UTL_ACCELERATION_1INCREMENT
UTLACCELERATION_1MIN
UTLACCELERATION_5MIN
UTL_ACCELERATION_5INCREMENT
UTLACCELERATION_DATE
UTLACCELERATION_DATE

Message Editing
UTL_SETUP_VARS_FOR_EDITING
UTL_FILL_SCROLL_BUFFER_WITH_SPACE
UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK
UTL_MOVE_CURSOR_BACKWARD_AND_REQ_BLINK
UTL_DISPLAY_MSG_AND_REQ_BLINK
UTL_POINT_TO_PREV_CHAR_AND_REQ_BLINK
UTL_POINT_TO_NEXT_CHAR_AND_REQ_BLINK
UTL_POINT_TO_PREV_CHAR
UTL_POINT_TO_NEXT_CHAR
UTL_CLEANUP_EDIT_BUFFER
UTL_TRIM_SENTINEL_CHAR
UTL_CHANGE_CHAR_TO_SPACE

Time Structure Math
KRES_CLEARDBUFFERS
KRES_HUNDSECMINHR_UPD
KRES_SECMINHR_UPD
KRES_MINHR_UPD
KRES_HUNDSECMINHR_SUB
KRES_SECMINHR_SUB
KRES_MINHR_SUB
API Reference

CORE API

Changing Foreground Application

CORE_REQ_MODE_CHANGE

Description
Requests a mode change to a specified mode. If the requested mode is not active, then the system will find the next active mode.

Usage
CORE_REQ_MODE_CHANGE

Assumptions
None

Input
A - Application Index to proceed

Output
None

Destroys
B

Example
; proceed to application 3rd application in the mode list index
ld A, #03H
CORE_REQ_MODE_CHANGE

CORE_REQ_MODE_CHANGE_NEXT

Description
Requests a mode change to the next mode with primary mode peek operation.

If application is active for more than a specified time, then it will initiate a mode change to the PRIMARY mode. This will enable a system peek to primary mode. If the user, releases the mode switch depression within a specified time period, the system will cancel the mode change request.

Usage
CORE_REQ_MODE_CHANGE_NEXT

Assumptions
None

Input
None

Output
None

Destroys
B

Example
; request a mode change to the next application or TOD
CORE_REQ_MODE_CHANGE_NEXT
**CORE_REQ_MODE_CHANGE_NEXT_NO_PEEK**

**Description**
Requests a mode change to the next mode without a peek to the primary mode.

If application is active for more than a specified time, then it will initiate a mode change to the PRIMARY mode.

**Usage**
CORE_REQ_MODE_CHANGE_NEXT_NO_PEEK

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
B

**Example**
`;request a mode change to the next application or TOD with no TOD peek
CORE_REQ_MODE_CHANGE_NEXT_NO_PEEK`

---

**CORE_FORCE_PRIMARY_AS_NEXT_MODE**

**Description**
Forces the primary mode to be the next mode on future mode change next requests. This API does not initiate a mode change to the next mode.

**Usage**
CORE_FORCE_PRIMARY_AS_NEXT_MODE

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
HL

**Example**
`;make TOD the next mode on the next mode change request
CORE_FORCE_PRIMARY_AS_NEXT_MODE
CORE_REQ_MODE_CHANGE_NEXT
CORE_REQ_MODE_CHANGE_NEXT_NO_PEEK`

---

**Changing Application Foreground State Handler**

**CORE_REQ_STATE_CHANGE**
**Description**  
Request for a state change. This will clear the entire display area.

**Usage**  
`CORE_REQ_STATE_CHANGE`

**Assumptions**  
Required parameters are already setup prior to calling this macro. No checking is done to verify a valid macro input.

**Input**  
B - State within the current mode to proceed.

**Output**  
None

**Destroys**  
B

**Example**  
`;change state to the set banner state`  
`ld B, #CORESET1BANNERSTATE`  
`CORE_REQ_STATE_CHANGE`

---

**CORE_REQ_STATE_CHANGE_NO_CLEAR_DISPLAY**

**Description**  
Request for a state change without clearing the display area.

**Usage**  
`CORE_REQ_STATE_CHANGE_NO_CLEAR_DISPLAY`

**Assumptions**  
Required parameters are already setup prior to calling this macro. No checking is done to verify a valid macro input.

**Input**  
B - State within the current mode to proceed.

**Output**  
None

**Destroys**  
B

**Example**  
`;change state to the set banner state`  
`ld B, #CORESET1BANNERSTATE`  
`CORE_REQ_STATE_CHANGE_NO_CLEAR_DISPLAY`

---

**Requesting Timeouts**

**CORE_REQ_TIMEOUT_HIRES**

**Description**  
Request for a high resolution timeout.

**Usage**  
`CORE_REQ_TIMEOUT_HIRES {Timeout_Value}`

**Assumptions**  
None
Input

Timeout_Value - Duration before the system event COREEVENT_TIMEOUTDONE_HIGHRES is passed to the foreground application. This value is specified in 125ms increments.

The kernel provides some built-in constants for Timeout_Value:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMEOUTHIRES_1SEC</td>
<td>1 second timeout</td>
</tr>
<tr>
<td>TIMEOUTHIRES_1P5SEC</td>
<td>1.5 second timeout</td>
</tr>
<tr>
<td>TIMEOUTHIRES_2SEC</td>
<td>2 second timeout</td>
</tr>
<tr>
<td>TIMEOUTHIRES_3SEC</td>
<td>3 second timeout</td>
</tr>
<tr>
<td>TIMEOUTHIRES_4SEC</td>
<td>4 second timeout</td>
</tr>
<tr>
<td>TIMEOUTHIRES_5SEC</td>
<td>5 second timeout</td>
</tr>
</tbody>
</table>

Output

None

Destroys

B, A

Example

;request a 1.5 second timeout for banner display
CORE_REQ_TIMEOUT_HIRES TIMEOUTHIRES_1P5SEC

---

CORE_REQ_TIMEOUT_LORES

Description

Request for a Low resolution timeout.

Usage

CORE_REQ_TIMEOUT_LORES {Timeout_Value}

Assumptions

None

Input

Timeout_Value - Duration before the system event COREEVENT_TIMEOUTDONE_LOWRES is passed to the foreground application. This value is specified in 1 second increments.

The kernel provides some built-in constants for Timeout_Value:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMEOUTLORES_2SEC</td>
<td>2 second timeout</td>
</tr>
<tr>
<td>TIMEOUTLORES_3SEC</td>
<td>3 second timeout</td>
</tr>
<tr>
<td>TIMEOUTLORES_4SEC</td>
<td>4 second timeout</td>
</tr>
<tr>
<td>TIMEOUTLORES_10SEC</td>
<td>10 second timeout</td>
</tr>
<tr>
<td>TIMEOUTLORES_20SEC</td>
<td>20 second timeout</td>
</tr>
</tbody>
</table>

Output

None

Destroys

B, A

Example

;request a 10 second timeout
CORE_REQ_TIMEOUT_LORES TIMEOUTLORES_10SEC

---
**CORE_REQ_TIMEOUT_STICKY**

**Description**  
Request for a sticky timeout.

**Usage**  
CORE_REQ_TIMEOUT_STICKY \{Timeout\_Value\}

**Assumptions**  
None

**Input**  
 Timeout\_Value - Duration before the system event COREEVENT\_STICKY\_TIMEOUTDONE is passed to the foreground application. This value is specified in 125ms increments.

The kernel provides some built in constants for Timeout\_Value:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TIMOUTHIRESE_1SEC</td>
<td>1 second timeout</td>
</tr>
<tr>
<td>TIMOUTHIRESE_1P5SEC</td>
<td>1.5 second timeout</td>
</tr>
<tr>
<td>TIMOUTHIRESE_2SEC</td>
<td>2 second timeout</td>
</tr>
<tr>
<td>TIMOUTHIRESE_3SEC</td>
<td>3 second timeout</td>
</tr>
<tr>
<td>TIMOUTHIRESE_4SEC</td>
<td>4 second timeout</td>
</tr>
<tr>
<td>TIMOUTHIRESE_5SEC</td>
<td>5 second timeout</td>
</tr>
</tbody>
</table>

**Output**  
None

**Destroys**  
B, A

**Example**

```assembly
ld  A, [CORECurrentEvent]
cp  A, #COREEVENT\_STARTSTOPDEPRESS
jr  NZ, check\_other\_events

;request a 4 second sticky timeout if STARTSTOP depress detected
CORE_REQ\_TIMEOUT\_STICKY TIMOUTHIRESE\_4SEC
ret
```

check\_other\_events:

```assembly
ld  A, [CORECurrentEvent]
cp  A, #COREEVENT\_STARTSTOPRELEASE
jr  Z, ProcessReleaseEventIfSwitchReleaseAfterTimeout

cp  A, #COREEVENT\_STICKY\_TIMEOUTDONE
jr  Z, ProcessTimeoutDoneIfSwitchReleaseWithinTimeout
```

**CORE_CANCEL_TIMEOUTS**

**Description**  
Cancel all active timeouts in the system.

**Usage**  
CORECANCEL\_TIMEOUTS

**Assumptions**  
This is the default condition whenever a mode or state change occurs.
Function

Input

Output

Destroys

Example

;cancel all active timeout in the system
CORE_CANCEL_TIMEOUTS

Peeking At Background Application Data

**CORE_REQ_PEEK_APP_TYPE**

Description

Request the system to search for a specified application type and execute that application's background handler with the background event COREEVENT_PEEK.

Usage

CORE_REQ_PEEK_APP_TYPE

Assumptions

The application called to be “peek” must support the system event COREEVENT_PEEK in its background handler. If not supported by the application, the display will just be cleared.

If multiple active applications have the same type, then the criteria will be based on mode list order.

Input

None

B - Application Type

The kernel provides some built-in constants for application type:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREAPPTYPE_SYSTEM</td>
</tr>
<tr>
<td>COREAPPTYPE_COMM</td>
</tr>
<tr>
<td>COREAPPTYPE_OPTION</td>
</tr>
<tr>
<td>COREAPPTYPE_TOD</td>
</tr>
<tr>
<td>COREAPPTYPE_DATE</td>
</tr>
<tr>
<td>COREAPPTYPE_CHRONO</td>
</tr>
<tr>
<td>COREAPPTYPE_TIMER</td>
</tr>
<tr>
<td>COREAPPTYPE_SYNCHRO_TIMER</td>
</tr>
<tr>
<td>COREAPPTYPE_COUNTER</td>
</tr>
<tr>
<td>COREAPPTYPE_CONTACT</td>
</tr>
<tr>
<td>COREAPPTYPE_TASK</td>
</tr>
<tr>
<td>COREAPPTYPE_NOTES</td>
</tr>
<tr>
<td>COREAPPTYPE_SCHEDULE</td>
</tr>
<tr>
<td>COREAPPTYPE_TIDE</td>
</tr>
<tr>
<td>COREAPPTYPE_ALARM</td>
</tr>
<tr>
<td>COREAPPTYPE_GAME</td>
</tr>
<tr>
<td>COREAPPTYPE_APPPOINTMENT</td>
</tr>
<tr>
<td>COREAPPTYPE_OCCASION</td>
</tr>
</tbody>
</table>
Output
None

Destroys
B

Example
; peek at upcoming appointments
; this should be called only when there is an appointment app loaded
ld B, #COREAPPTYPEAPPOINTMENT
CORE_REQ_PEEK_APP_TYPE

Modifying System Operation

CORE_SET_AUTORETURN

Description
Request for an Auto-Return to Primary mode after 3 minutes of idle activity.

Usage
CORE_SET_AUTORETURN

Assumptions
None

Input
None

Output
None

Destroys
B

Example
; request autoreturn
CORE_SET_AUTORETURN

CORE_CLEAR_AUTORETURN

Description
Cancel any pending request for Auto-Return to primary mode on idle activity.

Usage
CORE_CLEAR_AUTORETURN

Assumptions
This is the default condition whenever a mode or state change occurs.

Input
None

Output
None

Destroys
B

Example
; do not initiate an autoreturn sequence in this mode
CORE_CLEAR_AUTORETURN
Blinking and Scrolling Operations

**CORE_REQ_BLINK_2HZ**

**Description**
Request blinking of an LCD icon. Blink rate is 2Hz.

**Usage**
CORE_REQ_BLINK_2HZ \{lcd_icon\}

**Assumptions**
None

**Input**

\texttt{lcd_icon} - LCD icon to blink

The following constants are available for \texttt{lcd_icon}:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR_FLAG</td>
<td>Stopwatch icon</td>
</tr>
<tr>
<td>MOON_FLAG</td>
<td>Moon icon</td>
</tr>
<tr>
<td>ACLK_FLAG</td>
<td>Alarm Clock icon</td>
</tr>
<tr>
<td>ARROW_FLAG</td>
<td>Arrow icon</td>
</tr>
<tr>
<td>TAIL_FLAG</td>
<td>Tail icon</td>
</tr>
<tr>
<td>TMR_FLAG</td>
<td>Hourglass icon</td>
</tr>
<tr>
<td>NOTE_FLAG</td>
<td>Note icon</td>
</tr>
<tr>
<td>P_FLAG</td>
<td>P icon</td>
</tr>
<tr>
<td>A_FLAG</td>
<td>A icon</td>
</tr>
<tr>
<td>L_FLAG</td>
<td>L icon</td>
</tr>
</tbody>
</table>

**Output**
None

**Destroys**
A, B

**Example**
\texttt{;blink the stopwatch icon}
CORE_REQ_BLINK_2HZ \{CHR_FLAG\}

---

**CORE_CANCEL_BLINK_2HZ**

**Description**
Cancel blinking of an LCD icon. Blink rate is 2Hz.

**Usage**
CORE_CANCEL_BLINK_2HZ \{lcd_icon\}

**Assumptions**
None

**Input**

\texttt{lcd_icon} - LCD icon to blink
The following constants are available for `lcd_icon`:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR_FLAG</td>
<td>Stopwatch icon</td>
</tr>
<tr>
<td>MOON_FLAG</td>
<td>Moon icon</td>
</tr>
<tr>
<td>ACLK_FLAG</td>
<td>Alarm Clock icon</td>
</tr>
<tr>
<td>ARROW_FLAG</td>
<td>Arrow icon</td>
</tr>
<tr>
<td>TAIL_FLAG</td>
<td>Tail icon</td>
</tr>
<tr>
<td>TMR_FLAG</td>
<td>Hourglass icon</td>
</tr>
<tr>
<td>NOTE_FLAG</td>
<td>Note icon</td>
</tr>
<tr>
<td>P_FLAG</td>
<td>P icon</td>
</tr>
<tr>
<td>A_FLAG</td>
<td>A icon</td>
</tr>
<tr>
<td>L_FLAG</td>
<td>L icon</td>
</tr>
</tbody>
</table>

NOTE: Multiple LCD icons can be ORed together into a single API call.

**Output**  
None

**Destroys**  
A, B

**Example**  
; cancel blinking of the stopwatch icon  
CORECancelar_BLINK_2HZ CHR_FLAG

---

**CORE_REQ_BLINK_4HZ**

**Description**  
Request Blinking at 4Hz. The LCD generic 4Hz blinking arguments have been set before calling this macro.

**Usage**  
CORE_REQ_BLINK_4HZ

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
A, B

**Example**  
; setup the display and clear routines for the generic blink engine  
LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_ADDR apptDisplaySetPositionOnly  
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_ADDR apptClearSetPositionOnly  
CORE_REQ_BLINK_4HZ

; force the data to be displayed first  
LCD_DISP_4HZ_DATA_FIRST

. . .

apptDisplaySetPositionOnly:  
// put code here to display the data for the generic blink engine  
. . .
apptClearSetPositionOnly:
    // put code here to clear the display area used in the display
    // display routine

CORE_CANCEL_BLINK_4HZ

Description  Cancel generic 4Hz blinking
Usage        CORE_CANCEL_BLINK_4HZ
Assumptions  None
Input        None
Output       None
Destroys     A, B
Example      ; cancel active generic blinking
             CORE_CANCEL_BLINK_4HZ

CORE_CANCEL_SCROLLING

Description  Cancel the active scrolling operation.
Usage        CORE_CANCEL_SCROLLING
Assumptions  This is the default condition whenever a mode or state change occurs.
Input        None
Output       None
Destroys     A, B
Example      ; cancel scrolling
             CORE_CANCEL_SCROLLING
Primary Mode Icon Operations

**CORE_BACKGROUND_ICON_REFRESH_ENABLE**

**Description**
Requests the system to inform the foreground application of any updates to the LCD Icon resource.

When the system detects a background operation changing the state of any of the primary mode LCD icon resource, it will send the event `COREEVENT_ICON_REFRESH` to the foreground application for processing.

**Usage**
`CORE_BACKGROUND_ICON_REFRESH_ENABLE`

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
`HL`

**Example**
```
;request an event to be passed to the foreground application
;if any of the primary mode lcd resource was updated (foreground or
;background)
CORE_BACKGROUND_ICON_REFRESH_ENABLE
```

---

**CORE_BACKGROUND_ICON_REFRESH_DISABLE**

**Description**
Cancels requests informing the foreground application of any updates to the LCD Icon resource.

**Usage**
`CORE_BACKGROUND_ICON_REFRESH_ENABLE`

**Assumptions**
This is the default condition whenever a mode or state change occurs.

**Input**
None

**Output**
None

**Destroys**
`HL`

**Example**
```
;do not pass an event to be passed to the foreground application
;if any of the primary mode lcd resource was updated (foreground or
;background)
CORE_BACKGROUND_ICON_REFRESH_DISABLE
```
Switch, Ring and Crown Operations

**CORE_ALLOW_KEYS**

**Description**
Allow specified switch events to be passed to the foreground application.

**Usage**
CORE_ALLOW_KEYS { switch_mask }

**Assumptions**
None

**Input**

*switch_mask* - Switch bit mask. Switches specified in this mask will be processed by the system.

The following constants are available for *switch_mask*:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bCOREModeSwitch</td>
<td>MODE switch</td>
</tr>
<tr>
<td>bCOREStopResetSwitch</td>
<td>STOP/RESET switch</td>
</tr>
<tr>
<td>bCOREStartSplitSwitch</td>
<td>START/SPLIT switch</td>
</tr>
<tr>
<td>bCORECWSwitch</td>
<td>RING CW switch</td>
</tr>
<tr>
<td>bCORECCWSwitch</td>
<td>RING CCW switch</td>
</tr>
<tr>
<td>bCOREELSwitch</td>
<td>EL switch</td>
</tr>
</tbody>
</table>

**NOTE:**
- EL switch should normally be used only for activating the INDIGLO Night-Light.
- The mask bCOREModeSwitch must be specified for default state handlers to allow mode changes.

**Output**
None

**Destroyed**
B

**Example**

; allow only the mode switch to be passed to the state handler
CORE_ALLOW_KEYS bCOREModeSwitch

---

**CORE_MASK_KEYS**

**Description**
Prevent specified switch events from being passed to the foreground application.

**Usage**
CORE_MASK_KEYS { switch_mask }

**Assumptions**
None

**Input**

*switch_mask* - Switch bit mask. Switches specified in this mask will be discarded by the system.

The following constants are available for *switch_mask*:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bCOREModeSwitch</td>
<td>MODE switch</td>
</tr>
</tbody>
</table>

**NOTE:**

- EL switch should normally be used only for activating the INDIGLO Night-Light.
- The mask bCOREModeSwitch must be specified for default state handlers to allow mode changes.
NOTE: EL switch should normally be used only for activating the INDIGLO Night-Light.

**Output**
None

**Destroys**
B

**Example**
```
; do not pass the ring events to the state handler
CORE_MASK_KEYS (bCORECWSwitch|bCORECCWSwitch)
```
Example

;suspend ring events
CORE_SUSPEND_RING_EVENTS

CORE_ENABLE_RING_EVENTS

Description
Enable any ring events to be passed to the application.

Usage
CORE_ENABLE_RING_EVENTS

Assumptions
Any of the following switches – bCORECWSwitch, bCORECCWSwitch must be allowed in the system. Refer to the CORE_ALLOW_KEYS for activating these switches.

Input
None

Output
None

Destroys
HL

Example

;enable ring events to be passed to the foreground application
CORE_ENABLE_RING_EVENTS

CORE_SUSPEND_RING_LEADING_EVENTS

Description
Suspends any ring events matching CW/CCW leading transitions to be passed to the application.

Usage
CORE_SUSPEND_RING_LEADING_EVENTS

Assumptions
Any of the following switches – bCORECWSwitch, bCORECCWSwitch must be allowed in the system. Refer to the CORE_ALLOW_KEYS for activating these switches.

This is the default condition whenever a mode or state change occurs.

Input
None

Output
None

Destroys
HL

Example

;do not allow ring leading events to be passed to the foreground application
CORE_SUSPEND_RING_LEADING_EVENTS
**CORE_ENABLE_RING_LEADING_EVENTS**

**Description** Enables any ring events matching CW/CCW leading transitions to be passed to the application.

**Usage** CORE_ENABLE_RING_LEADING_EVENTS

**Assumptions** Any of the following switches – bCORECWSwitch, bCORECCWSwitch must be allowed in the system. Refer to the CORE_ALLOW_KEYS for activating these switches.

**Input** None

**Output** None

**Destroys** HL

**Example**

;allow ring leading events to be passed to the foreground application

CORE_ENABLE_RING_LEADING_EVENTS

---

**CORE_ENABLE_SWITCH_RELEASE**

**Description** Enables any switch releases to be passed to the application.

**Usage** CORE_ENABLE_SWITCH_RELEASE

**Assumptions** Any of the following switches -- bCOREModeSwitch, bCOREStopResetSwitch, bCOREStartSplitSwitch -- must be allowed in the system. Refer to the CORE_ALLOW_KEYS for activating these switches.

**Input** None

**Output** None

**Destroys** HL

**Example**

;allow switch release events to be passed to the foreground application

CORE_ENABLE_SWITCH_RELEASE

---

**CORE_SUSPEND_SWITCH_RELEASE**

**Description** Suspends any switch releases to be passed to the application.

**Usage** CORE_SUSPEND_SWITCH_RELEASE

**Assumptions** Any of the following switches -- bCOREModeSwitch, bCOREStopResetSwitch, bCOREStartSplitSwitch -- must be allowed in the system. Refer to the
CORE_ALLOW_KEYS for activating these switches.

This is the default condition whenever a mode or state change occurs.

**Input** None

**Output** None

**Destroys** HL

**Example**

; do not pass the switch release events to the foreground application
CORE_SUSPEND_SWITCH_RELEASE

---

**CORE_ENABLE_GENERIC_SWITCH_EVENT**

**Description** Request the system to generate a generic switch depress or release event for application processing instead of sending specific switch event.

This will convert the following system events into the generic events
COREEVENT_ANYSWITCHDEPRESS and COREEVENT_ANYSWITCHRELEASE

COREEVENT_SWITCH1DEPRESS } COREEVENT_ANYSWITCHDEPRESS
COREEVENT_SWITCH2DEPRESS } --> COREEVENT_ANYSWITCHDEPRESS
COREEVENT_SWITCH3DEPRESS }

COREEVENT_SWITCH1RELEASE }
COREEVENT_SWITCH2RELEASE } --> COREEVENT_ANYSWITCHRELEASE
COREEVENT_SWITCH3RELEASE }

**Usage** CORE_ENABLE GENERIC SWITCH EVENT

**Assumptions** Any of the following switches -- bCOREModeSwitch, bCOREStopResetSwitch, bCOREStartSplitSwitch -- must be allowed in the system. Refer to the CORE_ALLOW KEYS for activating these switches.

**Input** None

**Output** None

**Destroys** HL

**Example**

; process state entry
ld A, [CORECurrentEvent]
cp A, #COREEVENT_STATEENTRY
jr NZ, check_next_event1

; request for a generic switch event. We do this since by default, the system will always pass specific switch events
CORE_ENABLE GENERIC SWITCH EVENT

; we would like to handle only the mode and stop/reset switch
CORE_ALLOW KEYS {bCOREModeSwitch|bCOREStopResetSwitch}
```assembly
ret

check_next_event1:
    cp A, #COREEVENT_ANYSWITCHDEPRESS
    jr NZ, check_next_event2
    ; TODO: write code to handle any switch depression
    ret

check_next_event2:
    --
```

**CORE_DISABLE_GENERIC_SWITCH_EVENT**

**Description**  
Request the system to generate a specific switch depress or release events for application processing.

**Usage**  
CORE_DISABLE_GENERIC_SWITCH_EVENT

**Assumptions**  
Any of the following switches -- bCOREModeSwitch, bCOREStopResetSwitch, bCOREStartSplitSwitch -- must be allowed in the system. Refer to the CORE_ALLOW_KEYS for activating these switches.

This is the default condition whenever a mode or state change occurs.

**Input**  
None

**Output**  
None

**Destroys**  
HL

**Example**  
; process state entry
ld A, [CORECurrentEvent]
cp A, #COREEVENT_STATEENTRY
jr NZ, check_next_event1

; request for a generic switch event. We do not need to do this
; since by default, the system will always pass specific switch events,
; but is shown here for reference to differentiate between ENABLE and
; DISABLE GENERIC SWITCH EVENT APIs
CORE_DISABLE_GENERIC_SWITCH_EVENT

; we would like to handle only the mode and stop/reset switch
CORE_ALLOW_KEYS (bCOREModeSwitch|bCOREStopResetSwitch)
ret

check_next_event1:
    cp A, #COREEVENT_MODESWITCHDEPRESS
    jr NZ, check_next_event2
    CORE_REQ_MODE_CHANGE_NEXT
    ret
check_next_event2:
   cp A, #COREEVENT_STOPRESETSWITCHDEPRESS
   jr NZ, check_next_event3
   CORE_REQ_MODE_CHANGE_NEXT
   ret

check_next_event3:
   ...

CORE_REQUEST_MELODY_POPUP_CANCEL

Description Request the system to send the system event COREEVENT_MELODYPOPUPCANCEL when any active melody during a popup is to be cancelled by any switch depression.

This facilitates popup melody cancellation using a single system event rather than handling all the different switch depression events. The switch depression event that was used to cancel the popup melody can be accessed from the variable COREEventArgument.

This is a one time request. Once the COREEVENT_MELODYPOPUPCANCEL event is passed to the foreground application, the system will automatically cancel the request.

Usage CORE_REQUEST_MELODY_POPUP_CANCEL

Assumptions None

Input None

Output None

Destroys None

Example ;request that the popup melody be cancelled by any switch (that was ;previously allowed by the key mask) and pass the event MELODYPOPUPCANCEL to ;the foreground application with the actual switch in the COREEventArgument ;variable
   CORE_REQUEST_MELODY_POPUP_CANCEL
   ...
   ld A, [CORECurrentEvent]
   cp A, #COREEVENT_MELODYPOPUPCANCEL
   jr NZ, CheckNextEvent1
   ld A, [COREEventArgument]
   cp A, #COREEVENT_MODESWITCHDEPRESS
   jr Z, ModeSwitchWasUsedToCancelPopupMelody
   CheckNextEvent1:
CORE_ENABLE_PULSE_MODE

Description  Sets the system to send ring pulse events to the foreground application instead of ring edge events on crown rotation.

Usage  CORE_ENABLE_PULSE_MODE

Assumptions  Any of the following switches – bCORECWSwitch, bCORECCWSwitch must be allowed in the system. Refer to the CORE_ALLOW_KEYS for activating these switches.

Input  None

Output  None

Destroys  HL, A, B

Example  ;enable pulses to be sent to the foreground application rather than edges
  CORE_ENABLE_PULSE_MODE

  ...

  ld  A,  [CORECurrentEvent]
  cp  A,  #COREEVENT_CW_PULSES
  jr  NZ, CheckNextEvent1

  ;process CW pulses
  ld  B,  [COREEventArgument]
  ;B = number of CW pulses detected

  CheckNextEvent1:

  cp  A,  #COREEVENT_CCW_PULSES
  jr  NZ, CheckNextEvent2

  ;process CCW pulses
  ld  B,  [COREEventArgument]
  ;B = number of CCW pulses detected

  CheckNextEvent2:

CORE_DISABLE_PULSE_MODE

Description  Sets the system to send ring edge events to the foreground application instead of ring pulse events on crown rotation.

Usage  CORE_DISABLE_PULSE_MODE

Assumptions  Any of the following switches – bCORECWSwitch, bCORECCWSwitch must be allowed in the system. Refer to the CORE_ALLOW_KEYS for activating these switches.
This is the default condition whenever a mode or state change occurs.

**Input**  
None

**Output**  
None

**Destroys**  
HL, A, B

**Example**  
;enable edge to be sent to the foreground application rather than edges
CORE_DISABLE_PULSE_MODE
CORE_SUSPEND_RING.LEADING_EVENTS

...  
ld A, [CORECurrentEvent]
cp A, #COREEVENT_CW.EDGE.TRAILING
jr NZ, CheckNextEvent1

;process CW trailing edge

CheckNextEvent1:

cp A, #COREEVENT_CCW.EDGE.TRAILING
jr NZ, CheckNextEvent2

;process CCW trailing edge

CheckNextEvent2:

---

**HW_KBD_CANCEL_CURRENT_SWITCH_RELEASE**

**Description**  
Indicates to the keyboard hardware driver not to pass currently depressed switch's release event to the core.

**Usage**  
HW_KBD_CANCEL_CURRENT_SWITCH_RELEASE

**Assumptions**

**Input**  
None

**Output**  
None

**Destroys**  
HL

**Example**  
ld A, [CORECurrentEvent]
cp A, #COREEVENT_START.SPLITDEPRESS
jr NZ, CheckNextEvent1

;process START/SPLIT depress

;we do not want to handle the release of this switch depression
Background Task Operations

**CORE_REQ_BACKGROUND_TASK**

**Description**  
Request a background task to be initiated by the system.

**Usage**  
CORE_REQ_BACKGROUND_TASK

**Assumptions**  
None

**Input**  
L - Application Index  
A - System Event to be passed to the background handler routine for processing

The kernel provides the following constants for system events to be processed in the background handler:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREEVENT_PRIMARY_TIME_CHANGE</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_START</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_START</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_NULL</td>
</tr>
</tbody>
</table>

**Output**  
None

**Destroys**  
None

**Example**  
; background handler processing  
ld A, [COREBackgroundEvent]  
cp A, #COREEVENT_REFRESH_START  
jr NZ, checknextevent1  
; todo: write code here to process refresh event initialization  
; we have determined that the handler needs more time to process  
; request for a callback whenever the system is not busy. Use the  
; CONTINUE event...  
ld L, [COREBackgroundAppIndex]  
ld A, #COREEVENT_REFRESH_CONTINUE
**CORE_REQ_BACKGROUND_TASK_WITH_PRIORITYCHECK**

**Description**
Request a background task to be initiated. A pending background task will not be overwritten if it has a higher priority setting compared to the new requested task.

**Usage**
CORE_REQ_BACKGROUND_TASK_WITH_PRIORITYCHECK

**Assumptions**
None

**Input**
- L - Application Index
- A - System Event to be passed to the background handler routine for processing

The kernel provides the following constants for system events to be processed in the background handler sorted according to descending priority:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREEVENT_PRIMARY_TIME_CHANGE</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_START</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_START</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_NULL</td>
</tr>
</tbody>
</table>

**Output**
None

**Destroys**
None

**Example**
DayUpdateProcessing:

```
ld  B, [UpdateFlag]
bit B, #bDayUpdate
jr  NZ, HourUpdateProcessing

ld  L, [COREBackgroundAppIndex]
ld  A, #COREEVENT_DAYUPDATE_START
CORE_REQ_BACKGROUND_TASK_WITH_PRIORITYCHECK
```

HourUpdateProcessing:

```
bit B, #bHourUpdate
jr  NZ, MinuteUpdateProcessing

; do not request a new background task if the DayUpdate code has been
```
MinuteUpdateProcessing:

bit B, #bMinuteUpdate
jr NZ, UpdateProcessingEnd

; do not request a new background task if the DayUpdate and HourUpdate code has been executed
ld L, [COREBackgroundAppIndex]
ld A, #COREEVENT_PEEKSEARCH_START
CORE_REQ_BACKGROUND_TASK_WITH_PRIORITYCHECK

UpdateProcessingEnd:

---

**CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE**

**Description**
Request a background task to be initiated for all active application that matches the specified application type.

**Usage**
CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE

**Assumptions**
None

**Input**
- A - Application Type
- L - System Event to be passed to the background handler routine for processing

The kernel provides the following constants for system events to be processed in the background handler:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREEVENT_PRIMARY_TIME_CHANGE</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_START</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_START</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_NULL</td>
</tr>
</tbody>
</table>

The kernel provides some built-in constants for application type:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREAPPTYPESYSTEM</td>
</tr>
<tr>
<td>COREAPPTYPECOMM</td>
</tr>
<tr>
<td>COREAPPTYPEOPTION</td>
</tr>
<tr>
<td>COREAPPTYPETOD</td>
</tr>
</tbody>
</table>
COREAPPTYPEDATE
COREAPPTYPECHRONO
COREAPPTYPESYNCHROTIMER
COREAPPTYPECOUNTER
COREAPPTYPECONTACT
COREAPPTYPETASK
COREAPPTYPENOTES
COREAPPTYPESCHEDULE
COREAPPTYPEEVENT
COREAPPTYPEDEMO
COREAPPTYPEGAME
COREAPPTYPEALARM
COREAPPTYPEAPPOINTMENT
COREAPPTYPEOCCASION

Output  None
Destroys None
Example  todSettingComplete:

; since we have updated the primary time zone, we
; have to alert alarm and appointment active in the system
; with this change

; tell any alarm application in system about the change
ld A, #COREAPPTYPEALARM
ld L, #COREEVENT_PRIMARY_TIME_CHANGE
CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE

; tell any appointment application in system about the change
ld A, #COREAPPTYPEAPPOINTMENT
ld L, #COREEVENT_PRIMARY_TIME_CHANGE
CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE

CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE_WITH_PRIORITYCHECK

Description  Request a background task to be initiated for the specified application type but checks first before overwriting a higher priority task.

Usage  CORE_REQ_BACKGROUND_TASK_FOR_APPTYPE_WITH_PRIORITYCHECK

Assumptions None

Input  
A - Application Type
L - System Event to be passed to the background handler routine for processing

The kernel provides the following constants for system events to be processed in the background handler sorted according to descending priority:

Constant
The kernel provides some built-in constants for application type:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREAPPTYPESYSTEM</td>
</tr>
<tr>
<td>COREAPPTYPECOMM</td>
</tr>
<tr>
<td>COREAPPTYPEOPTION</td>
</tr>
<tr>
<td>COREAPPTYPEOD</td>
</tr>
<tr>
<td>COREAPPTYPEDATE</td>
</tr>
<tr>
<td>COREAPPTYPECHRONO</td>
</tr>
<tr>
<td>COREAPPTYPEHISTORY</td>
</tr>
<tr>
<td>COREAPPTYPESYNCCHRONO</td>
</tr>
<tr>
<td>COREAPPTYPECOUNTER</td>
</tr>
<tr>
<td>COREAPPTYPECONTACT</td>
</tr>
<tr>
<td>COREAPPTYPETASK</td>
</tr>
<tr>
<td>COREAPPTYPENOTES</td>
</tr>
<tr>
<td>COREAPPTYPESCHEDULE</td>
</tr>
<tr>
<td>COREAPPTYPEEVENT</td>
</tr>
<tr>
<td>COREAPPTYPEDATE</td>
</tr>
<tr>
<td>COREAPPTYPEGAME</td>
</tr>
<tr>
<td>COREAPPTYPEALARM</td>
</tr>
<tr>
<td>COREAPPTYPEAPPOINTMENT</td>
</tr>
<tr>
<td>COREAPPTYPEEVENT</td>
</tr>
</tbody>
</table>

Output: None

Destroys: None

Example:

; request a background task for any alarm type application
ld  A, #COREAPPTYPEALARM
ld  L, # COREEVENT_REFRESH_START
CORE_REQ_BACKGROUND_TASK_FOR_PTZBASEDAPPS

**CORE_REQ_BACKGROUND_TASK_FOR_PTZBASEDAPPS**

**Description**
Request a background task to be initiated for application types that are dependent upon the primary time zone data.

These application types are usually dependent on the primary time zone:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREEVENT_PRIMARY_TIME_CHANGE</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_START</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_START</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_NULL</td>
</tr>
</tbody>
</table>
By definition, application type with indexes from 0xE0 to 0xFF are considered dependent on the primary time zone.

**Usage**

CORE_REQ_BACKGROUND_TASK_FOR_PTZBASEDAPPS

**Assumptions**

None

**Input**

L - System Event to be passed to the background handler routine for processing

The kernel provides the following constants for system events to be processed in the background handler sorted according to descending priority:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREEVENT_PRIMARY_TIME_CHANGE</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_START</td>
</tr>
<tr>
<td>COREEVENT_REFRESH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_START</td>
</tr>
<tr>
<td>COREEVENT_PEEK_SEARCH_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_DAY_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_START</td>
</tr>
<tr>
<td>COREEVENT_HOUR_UPDATE_CONTINUE</td>
</tr>
<tr>
<td>COREEVENT_NULL</td>
</tr>
</tbody>
</table>

**Output**

None

**Destroys**

None

**Example**

todSettingComplete:

; since we have updated the primary time zone, we
; have to alert all applications dependent on the primary time zone
ld L, #COREEVENT_PRIMARY_TIME_CHANGE
CORE_REQ_BACKGROUND_TASK_FOR_PTZBASEDAPPS
**CORE_CLEAR_ALL_BACKGROUND_APPLICATION_TASK**

Description
Clears all pending background application task.

Usage
CORE_CLEAR_ALL_BACKGROUND_APPLICATION_TASK

Assumptions
None

Input
None

Output
None

Destroys
None

Example
```
; clear all background task
CORE_CLEAR_ALL_BACKGROUND_APPLICATION_TASK
```

---

**Popup Operations**

**CORE_REQ_POPDOWN**

Description
Request a popdown operation. Once processed, the system will restore the interrupted foreground application. The state return is usually the interrupt state index unless the CORE_SET_POPDOWN_STATE specified a different return state.

Usage
CORE_REQ_POPDOWN

Assumptions
This must be called only in a popup state handler. Using this API on a non-popup state will produce unpredictable results or even a reset.

Input
None

Output
None

Destroys
B

Example
```
; completed processing the popup, request for a popdown
CORE_REQ_POPDOWN
ret
```
**CORE_SET_POPDOWN_STATE**

**Description**
Sets the return state index when the system returns from a popup.

This is usually called in the state entry event handler to indicate to the core to popdown to a different state if a popup occurs.

**Usage**
CORE_SET_POPDOWN_STATE {popdown_state}

**Assumptions**
None

**Input**
- popdown_state - State index of the current application to proceed on a popdown.

**Output**
None

**Destroys**
B

**Example**
; application banner state handler
; if a popup occurs in this state, it should proceed to the default
; state on a popup
CORE_SET_POPDOWN_STATE COREDEFAULTSTATE

---

**CORE_ENABLE_POPUP**

**Description**
Allow popups to occur in the current state. On exit from the popup, it will return to the interrupted state unless CORE_SET_POPDOWN_STATE was executed prior to the popup.

**Usage**
CORE_ENABLE_POPUPS

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
None

**Example**
; allow popups to occur in the current state handler
CORE_ENABLE_POPUPS

---

**CORE_SUSPEND_POPUP**
### CORE_SUSPEND_POPUPS

**Description**  
Suspend any popups from occurring in the current state.

**Usage**  
CORE_SUSPEND_POPUPS

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
None

**Example**  

```
;do not allow any popups to occur in the current state handler
CORE_SUSPEND_POPUPS
```

---

### CORE_CANCEL_ALL_POPUPS

**Description**  
Cancel all pending popups.

**Usage**  
CORE_CANCEL_ALL_POPUPS

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
A, HL

**Example**  

```
;cancel all popups currently being queued in the system
CORE_CANCEL_ALL_POPUPS
```

---

### CORE_CANCEL_ANY_PTZ_POPUPS

**Description**  
Cancels all popups queued that is dependent on the primary time zone. These popups are queued under the Time Zone Check resource and the Backup resource.

**Usage**  
CORE_CANCEL_ANY_PTZ_POPUPS

**Assumptions**  
None

**Input**  
None

**Output**  
None

---
Destroys

A, B, HL

Example

; cancel pending popups of application dependent on the primary time zone
CORE_CANCEL_ANY_PTZ_POPUPS

---

**CORETASK.Popup_Request_BCK**

Description

Request for a popup to be queued under the Backup Resource.

Usage

CORETASK.Popup_Request_BCK

Assumptions

None

Input

A - Application Index
L - Backup Resource Index

Output

None

Destroys

A, HL, IY

Example

; **called in the background handler**
; request a popup to the owner of the backup resource
ld A, [COREBackgroundAppIndex]
lb L, #0 ; resource 0
CORE_TASK.Popup_Request_BCK

---

**CORETASK.Popup_Cancel_BCK**

Description

Cancel popup request queued under a Backup Resource

Usage

CORETASK.Popup_Cancel_BCK

Assumptions

None

Input

L - Backup Resource Index

Output

None

Destroys

A, HL, IY

Example

; **called in the background handler**
; cancel pending backup resource popup
ld L, #0 ; resource 0
CORE_TASK.Popup_Cancel_BCK
**CORETASK_POPUP_REQUEST_TZC**

**Description**  
Request for a popup to be queued under the Time Zone Check Resource.

**Usage**  
CORETASK_POPUP_REQUEST_TZC

**Assumptions**  
None

**Input**  
A - Application Index
L - Time Zone Check Resource Index

**Output**  
None

**Destroys**  
A, HL, IY

**Example**  
;** called in the background handler**
;request a popup to the owner of the time zone check resource
ld A, [COREBackgroundAppIndex]
ld L, #0 ;resource 0
CORE_TASK_POPUP_REQUEST_TZC

---

**CORETASK_POPUP_CANCEL_TZC**

**Description**  
Cancel popup request queued under a Time Zone Check Resource

**Usage**  
CORETASK_POPUP_CANCEL_TZC

**Assumptions**  
None

**Input**  
L - Time Zone Check Resource Index

**Output**  
None

**Destroys**  
A, HL, IY

**Example**  
;** called in the background handler**
;cancel pending time zone resource popup
ld L, #0 ;resource 0
CORE_TASK_POPUP_CANCEL_TZC

---

**CORETASK_POPUP_REQUEST_TMR**

**Description**  
Request for a popup to be queued under the Timer Resource.
**Usage**  
CORETASK_POPUP_REQUEST_TMR

**Assumptions**  
None

**Input**  
A - Application Index  
L - Timer Resource Index

**Output**  
None

**Destroys**  
A, HL, IY

**Example**  
;;** called in the background handler**  
;;request a popup to the owner of the timer resource  
ld A, [COREBackgroundAppIndex]  
ld L, #0 ;;resource 0  
CORE_TASK_POPUP_REQUEST_TMR

---

**CORETASK_POPUP_CANCEL_TMR**

**Description**  
Cancel popup request queued under a Timer Resource

**Usage**  
CORETASK_POPUP_CANCEL_TMR

**Assumptions**  
None

**Input**  
L - Timer Resource Index

**Output**  
None

**Destroys**  
A, HL, IY

**Example**  
;;** called in the background handler**  
;;cancel pending timer resource popup  
ld L, #0 ;;resource 0  
CORE_TASK_POPUP_CANCEL_TMR

---

**CORETASK_POPUP_REQUEST_STP**

**Description**  
Request for a popup to be queued under the Stopwatch Resource.

**Usage**  
CORETASK_POPUP_REQUEST_STP

**Assumptions**  
None

**Input**  
A - Application Index  
L - Stopwatch Resource Index

---
**CORETASK_POPUP_CANCEL_STP**

**Description**
Cancel popup request queued under a Stopwatch Resource

**Usage**
CORETASK_POPUP_CANCEL_STP

**Assumptions**
None

**Input**
L - Stopwatch Resource Index

**Output**
None

**Destroys**
A, HL, IY

**Example**
;** called in the background handler**
;cancel pending stopwatch resource popup
ld L, #0 ;resource 0
CORE_TASK.Popup_Request_STP

**CORETASK_POPUP_REQUEST_SYN**

**Description**
Request for a popup to be queued under the Synchro Resource.

**Usage**
CORETASK_POPUP_REQUEST_SYN

**Assumptions**
None

**Input**
A - Application Index
L - Synchro Resource Index

**Output**
None

**Destroys**
A, HL, IY

**Example**
;** called in the background handler**
;request a popup to the owner of the synchro resource
ld A, [COREBackgroundAppIndex]
CORETASK_POPUP_CANCEL_SYN

Description: Cancel popup request queued under a Synchro Resource

Usage: CORETASK_POPUP_CANCEL_SYN

Assumptions: None

Input: L - Synchro Resource Index

Output: None

DESTroys: A, HL, IY

Example:

;** called in the background handler**
;cancel pending synchro resource popup
ld L, #0 ;resource 0
CORE_TASK_POPUP_CANCEL_SYN

Utilities

CORE_ENABLE_SET_MODE

Description: Sets a general purpose flag usually used to indicate that the application is currently in a set mode.

Usage: CORE_ENABLE_SET_MODE

Assumptions: None

Input: None

Output: None

DESTroys: HL

Example:

;set flag
CORE_ENABLE_SET_MODE
**CORE_DISABLE_SET_MODE**

Description: Clears a general purpose flag usually used to indicate that the application is not in a set mode.

Usage: CORE_DISABLE_SET_MODE

Assumptions: None

Input: None

Output: None

Destroys: HL

Example: ;clear flag

```
CORE_DISABLE_SET_MODE
```

---

**CORE_SET_RANDOM_SEED**

Description: Sets the base number for the pseudo-random number generator.

Usage: CORE_SET_RANDOM_SEED

Assumptions: None

Input: None

Output: None

Destroys: None

Example: ;get a new seed for the random number generator

```
CORE_SET_RANDOM_SEED
```

---

**CORE_GENERATE_RANDOM_NUMBER**

Description: Generates a pseudo-random number from 0 – 255.

Usage: CORE_GENERATE_RANDOM_NUMBER

Assumptions: None

Input: None

Output: A - Psuedo-Random number from 0-255. The new number is based on the old random number generated.
**CORE_DATABASE_MODIFIED_BY_USER**

**Description**
Updates the flag that the application database was modified by the user. This flag is checked by the PC to determine if it should include it in its list to upload the database.

**Usage**
CORE_DATABASE_MODIFIED_BY_USER

**Assumptions**
None

**Input**
A - Application Index

**Output**
None

**Destroys**
None

**Example**
;database has been modified. Set the flag to alert the PC during upload
ld A, [CORECurrentMode]
CORE_DATABASE_MODIFIED_BY_USER

---

**CORE_APP_ID_PRESENT**

**Description**
Checks if a given an application ID (Type and Instance) is presently active in the system.

**Usage**
CORE_APP_ID_PRESENT

**Assumptions**
The application during design time must know beforehand what the instance number of an application. Though there are other ways (documented and undocumented) to get the instance number to be used for this function. This might end up to be a least used utility.

**Input**
IX - Application ID
    LOWBYTE( IX ) - Application Type
    HIBYTE( IX ) - Application Instance

The kernel provides some built-in constants for application type:

**Constant**
**COREAPPTYPESYSTEM**
**COREAP>TypeCOMM**
**COREAP>TypeOPTION**
**COREAP>TypeTOD**
**COREAP>TypeDATE**
**COREAP>TypeCHRONO**
**COREAP>TypeTIMER**
**COREAP>TypeSYNCHROTIMER**
**COREAP>TypeCOUNTER**
**COREAP>TypeCONTACT**
**COREAP>TypeTASK**
**COREAP>TypeNOTES**
**COREAP>TypeSCHEDULE**
**COREAP>TypeTIDE**
**COREAP>TypeDEMO**
**COREAP>TypeGAME**
**COREAP>TypeALARM**
**COREAP>TypeAPPOINTMENT**
**COREAP>TypeOCCASION**

### Output
None

**Success:** A = Application Index
**Failure:** A = 0xFF

### Destroys
None

### Example
; check if a specific ID is present in the system
ld IX, #((COREAPResponseTypeCHRONO << 8) | 00H)
CORE_APP_ID_PRESENT
cp A, #0FFH
jr Z, RequestedAppIDIsNotPresent

---

**CORE_GET_APP_INDEX**

**Description**
Checks if a given application ID (Type and Instance) is presently active in the system. The same functionality as CORE_APP_ID_PRESENT.

**Usage**
CORE_GET_APP_INDEX

**Assumptions**
The application during design time must know beforehand what the instance number of an application. Though there are other ways (documented and undocumented) to get the instance number to be used for this function. This might end up to be a least used utility.

**Input**
<table>
<thead>
<tr>
<th>IX</th>
<th>Application ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOWBYTE( IX )</td>
<td>Application Type</td>
</tr>
<tr>
<td>HIBYTE( IX )</td>
<td>Application Instance</td>
</tr>
</tbody>
</table>

The kernel provides some built-in constants for application type:

**Constant**

COREAP_TYPESYSTEM
COREAPPTYPECOMM
COREAPPTYPEOPTION
COREAPPTYPETOD
COREAPPTYPEDATE
COREAPPTYPECHRONO
COREAPPTYPETIMER
COREAPPTYPESYNCHROTIMER
COREAPPTYPECOUNTER
COREAPPTYPECONTACT
COREAPPTYPETASK
COREAPPTYPENOTES
COREAPPTYPESCHEDULE
COREAPPTYPETIDE
COREAPPTYPEDEMO
COREAPPTYPEGAME
COREAPPTYPEALARM
COREAPPTYPEAPPOINTMENT
COREAPPTYPEOCCASION

**Output**

None

<table>
<thead>
<tr>
<th>Success:</th>
<th>A = Application Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>Failure:</td>
<td>A = 0xFF</td>
</tr>
</tbody>
</table>

**Destroys**

None

**Example**

; check if a specific ID is present in the system
ld IX, #((COREAPPTYPECHRONO << 8) | 00H )
CALL CORE_GET_APP_INDEX
cp A, #0FFH
jr Z, RequestedAppIDIsNotPresent

---

**CORE_GET_APP_INDEX_FROM_APP_TYPE**

**Description**

Checks if a given Application Type is currently active in the system. If more than one application matches the same type, the first application found based on the mode list table will be selected.

**Usage**

CORE_GET_APP_INDEX_FROM_APP_TYPE

**Assumptions**

None

**Input**

A - Application Type

The kernel provides some built-in constants for application type:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>COREAPPTYPESYSTEM</td>
</tr>
<tr>
<td>COREAPPTYPECOMM</td>
</tr>
<tr>
<td>COREAPPTYPEOPTION</td>
</tr>
<tr>
<td>COREAPPTYPETOD</td>
</tr>
<tr>
<td>COREAPPTYPEDEMO</td>
</tr>
<tr>
<td>COREAPPTYPEGAME</td>
</tr>
<tr>
<td>COREAPPTYPESCHEDULE</td>
</tr>
<tr>
<td>COREAPPTYPECHRONO</td>
</tr>
</tbody>
</table>
COREAPPTYPETIMER
COREAPPTYPESYNCHROTIMER
COREAPPTYPECOUNTER
COREAPPTYPECONTACT
COREAPPTYPETASK
COREAPPTYPENOTES
COREAPPTYPESCHEDULE
COREAPPTYPETIDE
COREAPPTYPEDEMO
COREAPPTYPEGAME
COREAPPTYPEALARM
COREAPPTYPEAPPOINTMENT
COREAPPTYPEOCCASION

Output
None
Success:     A = Application Index
Failure:     A = 0xFF

Destroys
None

Example
; get the first app index of an appointment type application
ld   A, #COREAPPTYPEAPPOINTMENT
CORE_GET_APP_INDEX_FROM_APP_TYPE
cp   A, #0FFH
jr   Z, ThereIsNoAppointmentApplicationPresentInSystem
**AUDIO API**
These are used to handle melody generation.

**AUDSTART_SYSTEM_MELODY**

**Description**
Starts generation of a system melody.

**Usage**
AUDSTART_SYSTEM_MELODY {melody_index}, {melody_done_action}

**Assumptions**
None

**Input**
- melody_index - system melody to generate
- melody_done_action - Action to be done when melody generation is complete.

The following constants are available for melody_index:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDSWBEEPMELODY</td>
<td>Switch beep melody</td>
</tr>
<tr>
<td>AUDHOURCHIMEMELODY</td>
<td>Hour Chime melody</td>
</tr>
<tr>
<td>AUDALARMEMELODY</td>
<td>Alarm melody</td>
</tr>
<tr>
<td>AUDAPPPOINTMENTMELODY</td>
<td>Appointment melody</td>
</tr>
<tr>
<td>AUDTIMERMELODY</td>
<td>Timer Melody</td>
</tr>
<tr>
<td>AUDINTERVALTIMERMELODY</td>
<td>Interval Timer Melody</td>
</tr>
<tr>
<td>AUDHALFTIMERMELODY</td>
<td>Half timer melody</td>
</tr>
<tr>
<td>AUDCOMMERRORMELODY</td>
<td>Communication Error melody</td>
</tr>
<tr>
<td>AUDCUSTOMMELODY</td>
<td>Custom melody</td>
</tr>
</tbody>
</table>

The following constants are available for melody_done_action:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDSENDMELODYDONEEVENT</td>
<td>Send COREEVENT_MELODY_DONE at the end of melody to application.</td>
</tr>
<tr>
<td>AUDNOMELODYDONEEVENT</td>
<td>Do not send COREEVENT_MELODY_DONE at the end of melody to application.</td>
</tr>
</tbody>
</table>

**Output**
None

**Destroys**
BA

**Example**
; request for the alarm melody to be generated and send an event when the ; entire melody pattern has been completed
AUDSTART_SYSTEM_MELODY AUDALARMEMELODY, AUDSENDMELODYDONEEVENT

**AUDSETUP_MELODYADDRESS**

**Description**
Setup the base address of a melody into the melody table. Usually used to setup a new melody pattern for the custom melody offset.
Usage
AUDSETUP_MELODYADDRESS

Assumptions
None

Input
HL - base memory address of where the melody pattern is located
BA - melody table index to setup

The following constants are available for BA:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AUDSWBEEPMELODY</td>
<td>Switch beep melody</td>
</tr>
<tr>
<td>AUDHOURCHIMELELODY</td>
<td>Hour Chime melody</td>
</tr>
<tr>
<td>AUDALARMMELODY</td>
<td>Alarm melody</td>
</tr>
<tr>
<td>AUDAPPOINTMENTMELODY</td>
<td>Appointment melody</td>
</tr>
<tr>
<td>AUDTIMERMELODY</td>
<td>Timer Melody</td>
</tr>
<tr>
<td>AUDINTERVALTIMERMELODY</td>
<td>Interval Timer Melody</td>
</tr>
<tr>
<td>AUDHALFTIMERMELODY</td>
<td>Half timer melody</td>
</tr>
<tr>
<td>AUDCOMMERORMELODY</td>
<td>Communication Error melody</td>
</tr>
<tr>
<td>AUDCUSTOMMELODY</td>
<td>Custom melody</td>
</tr>
</tbody>
</table>

Output
None

Destroys
IX

Example
;Generate a custom melody
; setup the address of the user melody
ld HL, #MyNewMelodyPatternAddress
ld BA, #AUDCUSTOMMELODY
AUDSETUP_MELODYADDRESS

; start generating the user melody
AUDSTART_SYSTEM_MELODY AUDCUSTOMMELODY, AUDNOMELODYDONEEVENT

---

**AUDSTOP_MELODY**

Description
Stops any melody currently being generated.

Usage
AUDSTOP_MELODY

Assumptions
None

Input
None

Output
None

Destroys
HL

Example
; stop any current melody being generated
AUDSTOP_MELODY
DATABASE API
These are used to handle accesses to data or code stored in external memory.

Open and Close Database Operations

DB_OPEN_FILE

Description
Open a database file in external memory for read and write access. The EEPROM is powered up before the file is opened.

Usage
DB_OPEN_FILE

Assumptions
The database file opened must follow the database header structure requirements. This macro should not be used to open a link-list type of database. Use DB_OPEN_FILE_LINK_LIST instead.

Input
DBExternalMemoryAddress - 16-bit absolute address of the database file in external memory

Output
DBExternalBaseAddress - Base address of database use to convert offsets to absolute address (16-bit). Used only by the database drivers.
DBNumberOfRecords - Number of records stored for the current database (16-bit).
DBSizePerRecord - Size of record of database. Significant only for fixed-size random access database.

Destroys
None

Example
; Open the associated database of the current application
ld      HL, [CORECurrentADDAddress]
ld      [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; get the number of records of the opened database
ld      HL, [DBNumberOfRecords]
ld      [CONTTotalNumberOfRecords], HL
DB_CLOSE_FILE

DB_OPEN_FILE_LINK_LIST

Description
Open a linked-list database file in external memory for read and write access. The EEPROM is powered up before the file is opened.

Usage
DB_OPEN_FILE_LINK_LIST

Assumptions
The database file opened must follow the database header structure requirements. This macro should not be used to open a non-link-list type of database. Use DB_OPEN_FILE instead.

Input
DBExternalMemoryAddress - 16-bit absolute address of the database file in external memory

Output
DBExternalBaseAddress - Base address of database use to convert offsets to absolute
address (16-bit). Used only by the database drivers.

DBNumberOfRecords - Number of records stored for the current database (16-bit).

Destroy
None

Example
; Open the associated linked list database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE_LINK_LIST

---

**DB_CLOSE_FILE**

Description
Close the currently opened database file. The EEPROM is powered down.

Usage
DB_CLOSE_FILE

Assumptions
None

Input
None

Output
None

Destroy
None

Example
; close any database currently opened. This will remove power
; from the eeprom
DB_CLOSE_FILE

---

Database Information Header Operations

**DB_READ_APPLICATION_INFOHEADER**

Description
Reads the application specific data from a database header. This macro will power up and power down the EEPROM.

This is usually used during COREEVENT_INIT processing in the background handler. When the PC downloads a database to the watch, it passes information about the database in the header section of the database. For example, the PC will specify in this section the number of records stored in the database. This will be used by the WristApp for its internal processing to prevent it from writing records beyond the specified number of records.

Usage
DB_READ_APPLICATION_INFOHEADER

Assumptions
The database structure must conform to the M851 Database Structure requirements.

Input
DBExternalMemoryAddress - absolute address of database in external memory (16-bit)
### DBInternalMemoryAddress
- destination address in internal memory (16-bit)

### DBLengthLo
- number of bytes to read (8-bit)

### DBRecordOffset
- from start of app specific header to begin reading (8-bit)

#### Output
None

#### Destroys
None

#### Example

```assembly
; Open the associated database of the current application
ld HL, [CORECurrentADDRAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; setup the absolute address of the current application’s database
ld IY, [COREBackgroundADDRAddress]
ld [DBExternalMemoryAddress], IY

; setup the internal memory to store the database header information
ld HL, #MyApplicationHeaderBaseAddress
ld [DBInternalMemoryAddress], HL

; setup the number of bytes to read from the header information
ld A, #MyApplicationHeaderSize
ld [DBLengthLo], A

; setup the offset from the start of the header information to begin reading
ld A, #0
ld [DBRecordOffset], A

; read the header information
DB_READ_APPLICATION_INFOHEADER
```

---

#### DB_WRITE_APPLICATION_INFOHEADER

**Description**
Updates the data in the application specific header of the database. This macro will powerup and powerdown the EEPROM.

This macro is usually used during **COREEVENT_UPDATEDATABASEHEADER** in the background handler for WristApps that modifies database contents in the watch. This will update the header section so that a WristApp configuration application on the PC can decipher the changes made to the database. For example, changes made to a linked list database head pointer (deletion of the first record or insertion to the first record) needs to be communicated back to the PC configuration application so it can follow the new link list.

**Usage**
DB_WRITE_APPLICATION_INFOHEADER

**Assumptions**
The database structure must conform to the M851 Database Structure requirements.

**Input**
- **DBExternalMemoryAddress**: absolute address of database in external memory (16-bit)
- **DBInternalMemoryAddress**: source address in internal memory (16-bit)
- **DBLengthLo**: number of bytes to write (8-bit)
- **DBRecordOffset**: from start of app specific header to begin writing (8-bit)
**DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMVAR**

**Description**
Returns the absolute address of a record in a variable-sized random access database.

**Usage**
DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMVAR

**Assumptions**
The database file has been previously opened.
The database must be a variable-sized random access database structure.
Record size is limited to 256 bytes.

**Input**
DBRecordNumber - record number to compute (16-bit)

**Output**
BA - absolute address of specified record in EEPROM (16-bit)
DBExternalMemoryAddress - absolute address of specified record in EEPROM (16-bit)

**Destroys**
None

**Example**

```
; Open the associated database of the current application
ld  HL, [CORECurrentAddAddress]
ld  [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; get the absolute eeprom address of the specified record
ld  HL, [MyCurrentRecord]
ld  [DBRecordNumber], HL
DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMVAR
```
**DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMFIX**

**Description**
Returns the absolute address of a record in a fixed-sized random access database.

**Usage**
DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMFIX

**Assumptions**
The database file has been previously opened.
The database must be a fixed-sized random access database structure.
Record size is limited to 256 bytes.

**Input**
- DBRecordNumber - record number to compute (16-bit)

**Output**
- BA - absolute address of specified record in EEPROM (16-bit)
- DBExternalMemoryAddress - absolute address of specified record in EEPROM (16-bit)

**Destroys**
None

**Example**
```assembly
; Open the associated database of the current application
ld  HL, [CORECurrentADDAddress]
ld  [DBExternalMemoryAddress], HL
DB_OPEN_FILE

;get the absolute eeprom address of the specified record
ld  HL, [MyCurrentRecord]
ld  [DBRecordNumber], HL
DB_GET_ABSOLUTE_ADDRESS_OF_RECORD_RANDOMFIX
```

---

**Record Read and Write Operations**

**DB_WRITE_RECORD**

**Description**
Writes a specified record data to database in external memory starting at the start address of record.
This is used for writing data to a sequential access and linked list database structure.

**Usage**
DB_WRITE_RECORD

**Assumptions**
The database file has been previously opened.
Record size is limited to 256 bytes.

**Input**
- DBInternalMemoryAddress - source base address in internal memory (16-bit)
- DBExternalMemoryAddress - destination record offset in external memory (16-bit)
- DBLengthLo - number of bytes to write (8-bit)

**Output**
None
Destroys None

Example

; Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE_LINK_LIST

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; write data from internal memory to eeprom
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; specify the base offset of a record in the database
ld HL, [MyLinkListHeadPtr]
ld [DBExternalMemoryAddress], HL

; specify the source data array to write to the database
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL

; specify the number of bytes to transfer
ld A, #MyRecordDataSize
ld [DBLengthLo], A

; write the data array to eeprom database
DB_WRITE_RECORD

---

DB_WRITE_RECORD_WITHOFFSET

Description Writes a specified record data to database in external memory starting at an offset from start address of record.

Usage DB_WRITE_RECORD_WITHOFFSET

Assumptions The database file has been previously opened. Record size is limited to 256 bytes.

Input

- DBInternalMemoryAddress - source base address in internal memory (16-bit)
- DBExternalMemoryAddress - destination record offset in external memory (16-bit)
- DBLengthLo - number of bytes to write (8-bit)
- DBRecordOffset - offset from start address of record to begin writing (8-bit)

Output None

Destroys None

Example

; Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE_LINK_LIST

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; write data from internal memory to eeprom
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

---
; specify the base address of a record in the database
ld HL, [MyLinkListHeadPtr]
ld [DBExternalMemoryAddress], HL

; specify the source data array to write to the database
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL

; specify the number of bytes to transfer
ld A, #MyRecordDataSize
ld [DBLengthLo], A

; specify the byte offset from base address of a record
ld A, #ActualStartOfRecordData
ld [DBRecordOffset], A

; write the data array to eeprom database
DB_WRITE_RECORD_WITHOFFSET

---

**DB_WRITE_RECORD_RANDOMFIX**

**Description**  
Writes to a specific record in a fixed-sized random access database in external memory starting at the start address of record.

**Usage**  
DB_WRITE_RECORD_RANDOMFIX

**Assumptions**  
The database file has been previously opened.  
The database structure must adhere to the header requirements of a random fixed database structure.  
Record size is limited to 256 bytes.

**Input**  
- **DBInternalMemoryAddress** - source base address in internal memory (16-bit)
- **DBRecordNumber** - Record number (16-bit)
- **DBLengthLo** - number of bytes to write (8-bit)

**Output**  
None

**Destroys**  
None

**Example**  
; Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; specify the parameters to write a data array to a random fix record
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL
ld HL, [MyCurrentRecordNumber]
ld [DBRecordNumber], HL
ld A, #MyRecordDataSize
ld [DBLengthLo], A

; write the data to eeprom
DB_WRITE_RECORD_RANDOMFIX
**DB_WRITE_RECORD_WITHOFFSETRANDOMFIX**

**Description**
Writes to a specific record in a fixed-sized random access database in external memory starting at a specified offset from the start address of record.

**Usage**
DB_WRITE_RECORD_WITHOFFSETRANDOMFIX

**Assumptions**
The database file has been previously opened.
The database structure must adhere to the header requirements of a random fixed database structure.
Record size is limited to 256 bytes.

**Input**
- DBInternalMemoryAddress - source base address in internal memory (16-bit)
- DBRecordNumber - Record number (16-bit)
- DBLengthLo - number of bytes to write (8-bit)
- DBRecordOffset - offset from start address of record to begin writing (8-bit)

**Output**
None

**Destroys**
None

**Example**
```
; Open the associated database of the current application
 ld HL, [CORECurrentADDAddress]
 ld [DBExternalMemoryAddress], HL
 DB_OPEN_FILE

; specify the parameters to write a data array to a random fix record
 ld HL, #MyRecordDataBaseAddress
 ld [DBInternalMemoryAddress], HL
 ld HL, [MyCurrentRecordNumber]
 ld HL, [DBRecordNumber], HL
 ld A, #MyRecordDataSize
 ld [DBLengthLo], A

; specify the offset into the record
 ld A, #MYRECORDDATAFIELD5OFFSET
 ld [DBRecordOffset], A

; write the data to eeprom
 DB_WRITE_RECORD_WITHOFFSETRANDOMFIX
```

**DB_WRITE_RECORD_RANDOMVAR**

**Description**
Writes to a specific record in a variable-sized random access database in external memory starting at the start address of record.

**Usage**
DB_WRITE_RECORD_RANDOMVAR

**Assumptions**
The database file has been previously opened.
The database structure must adhere to the header requirements of a random variable size database structure. Record size is limited to 256 bytes.

**Input**
- DBInternalMemoryAddress - source base address in internal memory (16-bit)
- DBRecordNumber - Record number (16-bit)
- DBLengthLo - number of bytes to write (8-bit)

**Output**
None

**Destroys**
None

**Example**
```
; Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; specify the parameters to write a data array to a random variable record
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL
ld HL, [MyCurrentRecordNumber]
ld A, #MyRecordDataSize
ld [DBLengthLo], A
DB_WRITE_RECORD_RANDOMVAR
```

---

### DB_WRITE_RECORD_WITHOFFSET_RANDOMVAR

**Description**
Writes to a specific record in a variable-sized random access database in external memory starting at a specified offset from the start address of the record.

**Usage**
DB_WRITE_RECORD_WITHOFFSET_RANDOMVAR

**Assumptions**
The database file has been previously opened.
The database structure must adhere to the header requirements of a random variable size database structure. Record size is limited to 256 bytes.

**Input**
- DBInternalMemoryAddress - source base address in internal memory (16-bit)
- DBRecordNumber - Record number (16-bit)
- DBLengthLo - number of bytes to write (8-bit)
- DBRecordOffset - offset from start address of record to begin writing (8-bit)

**Output**
None

**Destroys**
None

**Example**
```
; Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; specify the parameters to write a data array to a random variable record
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL
ld HL, [MyCurrentRecordNumber]
ld A, #MyRecordDataSize
ld [DBLengthLo], A
DB_WRITE_RECORD_RANDOMVAR
```
DB_READ_RECORD

Description
Reads a specified record data to database in external memory starting at the start address of record.
This is used for reading data from a sequential access and linked list database structure.

Usage
DB_READ_RECORD

Assumptions
The database file has been previously opened.
Record size is limited to 256 bytes.

Input
- DBInternalMemoryAddress - destination base address in internal memory (16-bit)
- DBExternalMemoryAddress - source record offset in external memory (16-bit)
- DBLengthLo - number of bytes to read (8-bit)

Output
None

Destroys
None

Example
;Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; //////////////////////////////////////////////////////////////////////////////////
; read in data from eeprom to internal memory
; this example is accessing a sequential type structure
; //////////////////////////////////////////////////////////////////////////////////

; specify the base offset of a record in the database
; always 0 for sequential access memory
ld HL, #0000H
ld [DBExternalMemoryAddress], HL

; specify the memory buffer to store the data read from eeprom
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL

; specify the number of bytes to transfer
ld A, #MyRecordDataSize
ld [DBLengthLo], A

; read the data array from eeprom database
**DB_READ_RECORD**

**Description**
Reads a specified record data to database in external memory starting at an offset from the start address of record. This is used for reading data from a sequential access and linked list database structure.

**Usage**
DB_READ_RECORD_WITHOFFSET

**Assumptions**
The database file has been previously opened.
Record size is limited to 256 bytes.

**Input**
- DBInternalMemoryAddress - destination base address in internal memory (16-bit)
- DBExternalMemoryAddress - source record offset in external memory (16-bit)
- DBLengthLo - number of bytes to read (8-bit)
- DBRecordOffset - offset from start address of record to begin reading (8-bit)

**Output**
None

**Example**
```
;Open the associated database of the current application
ld  HL, [CORECurrentADDAddress]
ld  [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; read in data from eeprom to internal memory
; this example is accessing a sequential type structure

; specify the base offset of a record in the database
; always 0 for sequential access memory
ld  HL, #0000H
ld  [DBExternalMemoryAddress], HL

; specify the memory buffer to store the data read from eeprom
ld  HL, #MyRecordDataBaseAddress
ld  [DBInternalMemoryAddress], HL

; specify the number of bytes to transfer
ld  A, #MyRecordDataSize
ld  [DBLengthLo], A

; specify the offset from start of record
ld  A, #MyRecordDataOffsetForField3
ld  [DBRecordOffset], A

; read the data array from eeprom database
DB_READ_RECORD_WITHOFFSET
```
**DB_READ_RECORD_RANDOMFIX**

**Description**
Read from a specific record in a fixed-sized random access database in external memory starting at the start address of record.

**Usage**
DB_READ_RECORD_RANDOMFIX

**Assumptions**
The database file has been previously opened.
The database structure must adhere to the header requirements of a random fixed size database structure.
Record size is limited to 256 bytes.

**Input**
- DBInternalMemoryAddress - destination base address in internal memory (16-bit)
- DBRecordNumber - Record number (16-bit)
- DBLengthLo - number of bytes to read (8-bit)

**Output**
None

**Destroys**
None

**Example**

```assembly
; Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; get the absolute eeprom address of the specified record
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL
ld HL, [MyCurrentRecordNumber]
ld [DBRecordNumber], HL
ld A, #MyRecordDataSize
ld [DBLengthLo], A
DB_READ_RECORD_RANDOMFIX

; Open the associated database of the current application
ld HL, [CORECurrentADDAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; specify the parameters to read from a random fix record to a data array
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL
ld HL, [MyCurrentRecordNumber]
ld [DBRecordNumber], HL
ld A, #MyRecordDataSize
ld [DBLengthLo], A

; read the data from eeprom
DB_READ_RECORD_RANDOMFIX
```
**DB_READ_RECORD_WITHOFFSET_RANDOMFIX**

**Description**
Reads a specific record in a fixed-sized random access database in external memory starting at a specified offset from the start address of record.

**Usage**
DB_READ_RECORD_WITHOFFSET_RANDOMFIX

**Assumptions**
The database file has been previously opened.
The database structure must adhere to the header requirements of a random fixed size database structure.
Record size is limited to 256 bytes.

**Input**
- DBInternalMemoryAddress - destination base address in internal memory (16-bit)
- DBRecordNumber           - Record number (16-bit)
- DBLengthLo               - number of bytes to read (8-bit)
- DBRecordOffset           - offset from start address of record to begin reading (8-bit)

**Output**
None

**Destroys**
None

**Example**

```assembly
; Open the associated database of the current application
ld  HL, [CORECurrentADDAddress]
ld  [DBExternalMemoryAddress], HL
DB_OPEN_FILE

;get the absolute eeprom address of the specified record
ld  HL, #MyRecordDataBaseAddress
ld  [DBInternalMemoryAddress], HL
ld  HL, [MyCurrentRecordNumber]
ld  [DBRecordNumber], HL
ld  A, #MyRecordDataSize
ld  [DBLengthLo], A
DB_READ_RECORD_RANDOMFIX

; Open the associated database of the current application
ld  HL, [CORECurrentADDAddress]
ld  [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; specify the parameters to read from a random fix record to a data array
ld  HL, #MyRecordDataBaseAddress
ld  [DBInternalMemoryAddress], HL
ld  HL, [MyCurrentRecordNumber]
ld  [DBRecordNumber], HL
ld  A, #MyRecordDataSize
ld  [DBLengthLo], A

; specify the offset into the record
ld  A, #MYRECORDDATAFIELD5OFFSET
ld  [DBRecordOffset], A

; read the data from eeprom
DB_READ_RECORD_WITHOFFSET_RANDOMFIX
```
### DB_READ_RECORD_RANDOMVAR

**Description**  
Read from a specific record in a fixed-sized random access database in external memory starting at the start address of record.

**Usage**  
DB_READ_RECORD_RANDOMVAR

**Assumptions**  
The database file has been previously opened.  
The database structure must adhere to the header requirements of a random variable size database structure.  
Record size is limited to 256 bytes.

**Input**  
- DBInternalMemoryAddress - destination base address in internal memory (16-bit)  
- DBRecordNumber - Record number (16-bit)  
- DBLengthLo - number of bytes to read (8-bit)

**Output**  
None

**Destroys**  
None

**Example**  
; Open the associated database of the current application  
ld HL, [CORECurrentADDAddress]  
ld [DBExternalMemoryAddress], HL  
DB_OPEN_FILE

; specify the parameters to read from random variable size record  
ld HL, #MyRecordDataBaseAddress  
ld [DBInternalMemoryAddress], HL  
ld HL, [MyCurrentRecordNumber]  
ld [DBRecordNumber], HL  
ld A, #MyRecordDataSize  
ld [DBLengthLo], A

; read the record into internal memory  
DB_READ_RECORD_WITHOFFSET_RANDOMVAR

### DB_READ_RECORD_WITHOFFSET_RANDOMVAR

**Description**  
Reads a specific record in a variable-sized random access database in external memory starting at a specified offset from the start address of record.

**Usage**  
DB_READ_RECORD_WITHOFFSET_RANDOMVAR

**Assumptions**  
The database file has been previously opened.  
The database structure must adhere to the header requirements of a random variable size database structure.  
Record size is limited to 256 bytes.

**Input**  
- DBInternalMemoryAddress - destination base address in internal memory (16-bit)  
- DBRecordNumber - Record number (16-bit)
DBLengthLo - number of bytes to read (8-bit)
DBRecordOffset - offset from start address of record to begin reading (8-bit)

Output
None

Destroys
None

Example
; Open the associated database of the current application
ld HL, [CORECurrentAddrAddress]
ld [DBExternalMemoryAddress], HL
DB_OPEN_FILE

; specify the parameters to read from random variable size record
ld HL, #MyRecordDataBaseAddress
ld [DBInternalMemoryAddress], HL
ld HL, [MyCurrentRecordNumber]
ld [DBRecordNumber], HL
ld A, #MyRecordDataSize
ld [DBLengthLo], A

; specify the offset into the record
ld A, #MYRECORDDATAFIELD5OFFSET
ld [DBRecordOffset], A

;read the record into internal memory
DB_READ_RECORD_WITHOFFSET_RANDOMVAR

Linked-List Operations

**DB_REMOVE_RECORD_LINKLIST**

Description
Removes a record from a link list database structure.

Usage
DB_REMOVE_RECORD_LINKLIST

Assumptions
Double linked list database has been previously opened.

Input
HL - Absolute Address (internal memory) Link List Head Pointer
DBRecordAddress - Address offset of record to remove from the link list structure

Output
*HL - Updated with the new record address offset if removing first record in list.

Destroys
None

Example
ld HL, #MyLinkListHeadPointer
ld BA, [MyCurrentRecordPointer]
DB_REMOVE_RECORD_LINKLIST
**DB_INSERT_RECORD_LINKLIST**

**Description**
Inserts a record into a link list after a specified record.

**Usage**
DB_INSERT_RECORD_LINKLIST

**Assumptions**
Double Linked list database has been previously opened.

**Input**
- HL - Absolute Address (internal memory) Link List Head Pointer
- DBRecordAddress - Address offset of record to remove from the link list structure
- DBInsertRecordAddress - Address offset of record where the new record is to be appended. If DBNULLRECORDOFFSET, then new record will be inserted as the first record in list.

**Output**
- *HL - Updated with the new record address offset if the new record is the first record in list.

**Destroys**
None

**Example**
```
ld HL, #MyLinkListHeadPointer
ld BA, [MyCurrentRecordPointer]
ld [DBRecordAddress], BA
ld BA, [MyNewRecordPointer]
ld [DBInsertRecordAddress], BA
DB_INSERT_RECORD_LINKLIST
```

---

**DB_LOCATE_INSERTION_BYSIZE_LINKLIST**

**Description**
Locates the record where a new record can be inserted. The new record will be inserted to sort the list database according to size of the record. The new record is inserted after the specified record.

**Usage**
DB_LOCATE_INSERTION_BYSIZE_LINKLIST

**Assumptions**
Double Linked list database has been previously opened.

**Input**
- HL - record offset to begin searching for a record
- DBRecordSize - search criteria to locate where to insert a record in the list

**Output**
- BA - Results of the search function:
  - BAReg != DBNULLRECORDOFFSET
    - Record address offset where a new record can be inserted after.
  - BAReg == DBNULLRECORDOFFSET
    - Record should be located as the first record in the list.

**Destroys**
None

**Example**
```
; locate the address where we can insert the new record
```
ld HL, #MyUnusedLinkListHeadPointer
ld A, #SizeOfRecordToSearch
ld [DBRecordSize], A
DB_LOCATE_INDERTION_BYSIZE_LINKLIST

; insert the new record at the address return by the previous operation
ld [DBInsertRecordAddress], BA
ld HL, #MyLinkListHeadPointer
ld BA, [MyCurrentRecordPointer]
ld [DBRecordAddress], BA
DB_REMOVE_RECORD_LINKLIST
**DISPLAY API**

These are used to handle writing characters, numbers, and messages to the display device.

### Starting Segment Digit Positions

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCDSEGDIGIT1</td>
</tr>
<tr>
<td>LCDSEGDIGIT2</td>
</tr>
<tr>
<td>LCDSEGDIGIT3</td>
</tr>
<tr>
<td>LCDSEGDIGIT4</td>
</tr>
<tr>
<td>LCDSEGDIGIT5</td>
</tr>
<tr>
<td>LCDSEGDIGIT6</td>
</tr>
</tbody>
</table>

### Starting Main Dot-Matrix Line 1 Digit Positions:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCDMAINDMLINE1COL1</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL2</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL3</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL4</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL5</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL6</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL7</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL8</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL9</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL10</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL11</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL12</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL13</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL14</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL15</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL16</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL17</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL18</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL19</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL20</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL21</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL22</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL23</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL24</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL25</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL26</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL27</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL28</td>
</tr>
<tr>
<td>LCDMAINDMLINE1COL29</td>
</tr>
</tbody>
</table>
Starting Main Dot-Matrix Line 2 Digit Positions:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCDMAINDMLINE2COL1</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL2</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL3</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL4</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL5</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL6</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL7</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL8</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL9</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL10</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL11</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL12</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL13</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL14</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL15</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL16</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL17</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL18</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL19</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL20</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL21</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL22</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL23</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL24</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL25</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL26</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL27</td>
</tr>
<tr>
<td>LCDMAINDMLINE2COL28</td>
</tr>
</tbody>
</table>
Starting Upper Dot-Matrix Digit Positions:

<table>
<thead>
<tr>
<th>Constant</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LCDUPPERDMCOL1</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL2</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL3</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL4</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL5</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL6</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL7</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL8</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL9</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL10</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL11</td>
<td></td>
</tr>
<tr>
<td>LCDUPPERDMCOL12</td>
<td></td>
</tr>
</tbody>
</table>

Large Font Digit Positions

<table>
<thead>
<tr>
<th>Constant</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>LCDBIGCHARDMCOL1</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL2</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL3</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL4</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL5</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL6</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL7</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL8</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL9</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL10</td>
<td></td>
</tr>
<tr>
<td>LCDBIGCHARDMCOL11</td>
<td></td>
</tr>
</tbody>
</table>
Flag Address and Bit Definitions

<table>
<thead>
<tr>
<th>Address</th>
<th>Bit Pattern</th>
</tr>
</thead>
<tbody>
<tr>
<td>L_ADDR</td>
<td>bLCDBITPATTERN_L</td>
</tr>
<tr>
<td>A_ADDR</td>
<td>bLCDBITPATTERN_A</td>
</tr>
<tr>
<td>P_ADDR</td>
<td>bLCDBITPATTERN_P</td>
</tr>
<tr>
<td>TAIL_ADDR</td>
<td>bLCDBITPATTERN_TAIL</td>
</tr>
<tr>
<td>TMR_ADDR</td>
<td>bLCDBITPATTERN_TMR</td>
</tr>
<tr>
<td>ARROW_ADDR</td>
<td>bLCDBITPATTERN_ARROW</td>
</tr>
<tr>
<td>CHR_ADDR</td>
<td>bLCDBITPATTERN_CHR</td>
</tr>
<tr>
<td>MOON_ADDR</td>
<td>bLCDBITPATTERN_MOON</td>
</tr>
<tr>
<td>ACLK_ADDR</td>
<td>bLCDBITPATTERN_ACLK</td>
</tr>
<tr>
<td>NOTE_ADDR</td>
<td>bLCDBITPATTERN_NOTE</td>
</tr>
<tr>
<td>PER1_ADDR</td>
<td>bLCDBITPATTERN_PER1</td>
</tr>
</tbody>
</table>
DASH1_ADDR  bLCDBITPATTERN_DASH1
PER2_ADDR   bLCDBITPATTERN_PER2
DASH2_ADDR  bLCDBITPATTERN_DASH2

### 9-Segment Character Set

<table>
<thead>
<tr>
<th>Segment</th>
<th>Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEG_0</td>
<td>equ 0</td>
</tr>
<tr>
<td>SEG_1</td>
<td>equ 1</td>
</tr>
<tr>
<td>SEG_2</td>
<td>equ 2</td>
</tr>
<tr>
<td>SEG_3</td>
<td>equ 3</td>
</tr>
<tr>
<td>SEG_4</td>
<td>equ 4</td>
</tr>
<tr>
<td>SEG_5</td>
<td>equ 5</td>
</tr>
<tr>
<td>SEG_6</td>
<td>equ 6</td>
</tr>
<tr>
<td>SEG_7</td>
<td>equ 7</td>
</tr>
<tr>
<td>SEG_8</td>
<td>equ 8</td>
</tr>
<tr>
<td>SEG_9</td>
<td>equ 9</td>
</tr>
<tr>
<td>SEG_SPACE</td>
<td>equ 10</td>
</tr>
<tr>
<td>SEG_A</td>
<td>equ 11</td>
</tr>
<tr>
<td>SEG_B</td>
<td>equ 12</td>
</tr>
<tr>
<td>SEG_C</td>
<td>equ 13</td>
</tr>
<tr>
<td>SEG_D</td>
<td>equ 14</td>
</tr>
<tr>
<td>SEG_E</td>
<td>equ 15</td>
</tr>
<tr>
<td>SEG_F</td>
<td>equ 16</td>
</tr>
<tr>
<td>SEG_G</td>
<td>equ 17</td>
</tr>
<tr>
<td>SEG_H</td>
<td>equ 18</td>
</tr>
<tr>
<td>SEG_I</td>
<td>equ 19</td>
</tr>
<tr>
<td>SEG_J</td>
<td>equ 20</td>
</tr>
<tr>
<td>SEG_K</td>
<td>equ 21</td>
</tr>
<tr>
<td>SEG_L</td>
<td>equ 22</td>
</tr>
<tr>
<td>SEG_M</td>
<td>equ 23</td>
</tr>
<tr>
<td>SEG_N</td>
<td>equ 24</td>
</tr>
<tr>
<td>SEG_O</td>
<td>equ SEG_0</td>
</tr>
<tr>
<td>SEG_P</td>
<td>equ 25</td>
</tr>
<tr>
<td>SEG_Q</td>
<td>equ 26</td>
</tr>
<tr>
<td>SEG_R</td>
<td>equ 27</td>
</tr>
<tr>
<td>SEG_S</td>
<td>equ SEG_5</td>
</tr>
<tr>
<td>SEG_T</td>
<td>equ 28</td>
</tr>
<tr>
<td>SEG_U</td>
<td>equ 29</td>
</tr>
<tr>
<td>SEG_V</td>
<td>equ 30</td>
</tr>
<tr>
<td>SEG_W</td>
<td>equ 31</td>
</tr>
<tr>
<td>SEG_Y</td>
<td>equ 32</td>
</tr>
<tr>
<td>SEG_Z</td>
<td>equ SEG_2</td>
</tr>
<tr>
<td>SEG_MINUS</td>
<td>equ 33</td>
</tr>
<tr>
<td>SEG_DASH</td>
<td>equ SEG_MINUS</td>
</tr>
<tr>
<td>SEG_PLUS</td>
<td>equ 34</td>
</tr>
</tbody>
</table>
SEG_COLON                    equ     SEG_I
SEG_OPENPAR                  equ     SEG_C
SEG_CLOSEPAR                 equ     35
SEG_DOLLAR                   equ     36

5x5 Dot Matrix Character Set

DM5_0                        equ     0
DM5_1                        equ     1
DM5_2                        equ     2
DM5_3                        equ     3
DM5_4                        equ     4
DM5_5                        equ     5
DM5_6                        equ     6
DM5_7                        equ     7
DM5_8                        equ     8
DM5_9                        equ     9
DM5_SPACE                    equ     10
DM5_A                        equ     11
DM5_B                        equ     12
DM5_C                        equ     13
DM5_D                        equ     14
DM5_E                        equ     15
DM5_F                        equ     16
DM5_G                        equ     17
DM5_H                        equ     18
DM5_I                        equ     19
DM5_J                        equ     20
DM5_K                        equ     21
DM5_L                        equ     22
DM5_M                        equ     23
DM5_N                        equ     24
DM5_O                        equ     25
DM5_P                        equ     26
DM5_Q                        equ     27
DM5_R                        equ     28
DM5_S                        equ     29
DM5_T                        equ     30
DM5_U                        equ     31
DM5_V                        equ     32
DM5_W                        equ     33
DM5_X                        equ     34
DM5_Y                        equ     35
DM5_Z                        equ     36
DM5_EXCLAMATION              equ     37
DM5_DBLQUOTE                 equ     38
DM5NUMBER equ 39
DM5DOLLAREqu 40
DM5PERCENT equ 41
DM5ANDERSANDEqu 42
DM5SGQUOTE equ 43
DM5OPENPAREnequ 44
DM5CLOSEPAREnequ 45
DM5ASTERISK equ 46
DM5PLUSEqu 47
DM5COMMAequ 48
DM5DASHequ 49
DM5MINUSEqu DM5DASH
DM5PERIOD equ 50
DM5SLASHEqu 51
DM5COLONEqu 52
DM5SEMICOLON equ 53
DM5LESSTHANEqu 54
DM5EQUALEqu 55
DM5GREATERTHANEqu 56
DM5QUESTIONEqu 57
DM5ATREVERSEEqu 58
DM5OPENSQBRACKET equ 59
DM5BACKSLASHequ 60
DM5CLOSESBQACKETequ 61
DM5CIRCUMFLEXequ 62
DM5UNDERSCOREequ 63
DM5BACKPOSTROPHEequ 64
DM5OPENBRACEequ 65
DM5VERTBARequ 66
DM5CLOSEBRACEequ 67
DM5TILEDequ 68
DM5SECTIONequ 69

DM5EUROequ 70
DM5POUNDEqu 71
DM5YENequ 72
DM5AUMLAUTEqu 73
DM5ARINGequ 74
DM5AELIGATUREequ 75
DM5CCEDILLAequ 76
DM5NTILDEequ 77
DM5OUMLAUTequ 78
DM5OSLASHequ 79
DM5UUMLAUTequ 80
DM5SZLIGATUREequ 81
DM5INVECMATIONequ 82
DM5INVQUESTionequ 83
DM5_FEMORDINAL    equ     84
DM5_DEGREE        equ     85
DM5_MACRON        equ     86

DM5_SPADE         equ     87
DM5_CLUB          equ     88
DM5_HEART         equ     89
DM5_DIAMOND       equ     90
DM5_TEN           equ     91
DM5_NEWMOON       equ     92
DM5_FIRSTQUARTER  equ     93
DM5_LASTQUARTER   equ     94
DM5_DOWNARROW     equ     95
DM5_UPARROW       equ     96
DM5_AM            equ     97
DM5_PM            equ     98
DM5_MFORAMPM      equ     99
DM5_COMPRESS_1    equ    100
DM5_LEFTARROW     equ    101
DM5_RIGHTARROW    equ    102
DM5_CURSOR        equ    103
DM5_SENTINEL      equ    104
DM5_BLANK         equ    105

**8x8 Dot Matrix Character Set**

DM8_0            equ      0
DM8_1            equ      1
DM8_2            equ      2
DM8_3            equ      3
DM8_4            equ      4
DM8_5            equ      5
DM8_6            equ      6
DM8_7            equ      7
DM8_8            equ      8
DM8_9            equ      9
DM8_SPACE        equ     10
DM8_A            equ     11
DM8_L            equ     12
DM8_P            equ     13
DM8_S            equ     14
DM8_T            equ     15

**8x4 Dot Matrix Character Set**

DM8_I            equ     16
### 8x1 Dot Matrix Character Set

<table>
<thead>
<tr>
<th>Character</th>
<th>No.</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>DM8_PERIOD</td>
<td>equ</td>
<td>17</td>
</tr>
<tr>
<td>DM8_DOT</td>
<td>equ</td>
<td>17</td>
</tr>
<tr>
<td>DM8_COLON</td>
<td>equ</td>
<td>18</td>
</tr>
<tr>
<td>DM8_DASH</td>
<td>equ</td>
<td>19</td>
</tr>
</tbody>
</table>

#### Character Bitmap Patterns

The table below shows the font for the nine-segment character positions of the LCD. Some letters cannot be reasonably represented with nine segments, and they are shown as blank characters in the table.

#### Segment Display Patterns

<table>
<thead>
<tr>
<th>No.</th>
<th>Character</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>J</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>K</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>Q</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>Z</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>space</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>dash</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>:</td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>(</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>)</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>$</td>
<td></td>
</tr>
</tbody>
</table>
## Small Font Dot-Matrix Patterns

<table>
<thead>
<tr>
<th>No.</th>
<th>Character</th>
<th>Fixed-Width Display</th>
<th>Proportional Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td><img src="image1" alt="Fixed-Width Display" /></td>
<td><img src="image2" alt="Proportional Display" /></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td><img src="image3" alt="Fixed-Width Display" /></td>
<td><img src="image4" alt="Proportional Display" /></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td><img src="image5" alt="Fixed-Width Display" /></td>
<td><img src="image6" alt="Proportional Display" /></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td><img src="image7" alt="Fixed-Width Display" /></td>
<td><img src="image8" alt="Proportional Display" /></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td><img src="image9" alt="Fixed-Width Display" /></td>
<td><img src="image10" alt="Proportional Display" /></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td><img src="image11" alt="Fixed-Width Display" /></td>
<td><img src="image12" alt="Proportional Display" /></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td><img src="image13" alt="Fixed-Width Display" /></td>
<td><img src="image14" alt="Proportional Display" /></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td><img src="image15" alt="Fixed-Width Display" /></td>
<td><img src="image16" alt="Proportional Display" /></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td><img src="image17" alt="Fixed-Width Display" /></td>
<td><img src="image18" alt="Proportional Display" /></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td><img src="image19" alt="Fixed-Width Display" /></td>
<td><img src="image20" alt="Proportional Display" /></td>
</tr>
<tr>
<td>10</td>
<td>blank</td>
<td><img src="image21" alt="Fixed-Width Display" /></td>
<td><img src="image22" alt="Proportional Display" /></td>
</tr>
<tr>
<td>11</td>
<td>A</td>
<td><img src="image23" alt="Fixed-Width Display" /></td>
<td><img src="image24" alt="Proportional Display" /></td>
</tr>
<tr>
<td>12</td>
<td>B</td>
<td><img src="image25" alt="Fixed-Width Display" /></td>
<td><img src="image26" alt="Proportional Display" /></td>
</tr>
<tr>
<td>13</td>
<td>C</td>
<td><img src="image27" alt="Fixed-Width Display" /></td>
<td><img src="image28" alt="Proportional Display" /></td>
</tr>
<tr>
<td>14</td>
<td>D</td>
<td><img src="image29" alt="Fixed-Width Display" /></td>
<td><img src="image30" alt="Proportional Display" /></td>
</tr>
<tr>
<td>15</td>
<td>E</td>
<td><img src="image31" alt="Fixed-Width Display" /></td>
<td><img src="image32" alt="Proportional Display" /></td>
</tr>
<tr>
<td>16</td>
<td>F</td>
<td><img src="image33" alt="Fixed-Width Display" /></td>
<td><img src="image34" alt="Proportional Display" /></td>
</tr>
<tr>
<td>17</td>
<td>G</td>
<td><img src="image35" alt="Fixed-Width Display" /></td>
<td><img src="image36" alt="Proportional Display" /></td>
</tr>
<tr>
<td>18</td>
<td>H</td>
<td><img src="image37" alt="Fixed-Width Display" /></td>
<td><img src="image38" alt="Proportional Display" /></td>
</tr>
<tr>
<td>19</td>
<td>I</td>
<td><img src="image39" alt="Fixed-Width Display" /></td>
<td><img src="image40" alt="Proportional Display" /></td>
</tr>
<tr>
<td>20</td>
<td>J</td>
<td><img src="image41" alt="Fixed-Width Display" /></td>
<td><img src="image42" alt="Proportional Display" /></td>
</tr>
<tr>
<td>21</td>
<td>K</td>
<td><img src="image43" alt="Fixed-Width Display" /></td>
<td><img src="image44" alt="Proportional Display" /></td>
</tr>
<tr>
<td>22</td>
<td>L</td>
<td><img src="image45" alt="Fixed-Width Display" /></td>
<td><img src="image46" alt="Proportional Display" /></td>
</tr>
<tr>
<td>23</td>
<td>M</td>
<td><img src="image47" alt="Fixed-Width Display" /></td>
<td><img src="image48" alt="Proportional Display" /></td>
</tr>
<tr>
<td>24</td>
<td>N</td>
<td><img src="image49" alt="Fixed-Width Display" /></td>
<td><img src="image50" alt="Proportional Display" /></td>
</tr>
<tr>
<td>25</td>
<td>O</td>
<td><img src="image51" alt="Fixed-Width Display" /></td>
<td><img src="image52" alt="Proportional Display" /></td>
</tr>
<tr>
<td>26</td>
<td>P</td>
<td><img src="image53" alt="Fixed-Width Display" /></td>
<td><img src="image54" alt="Proportional Display" /></td>
</tr>
<tr>
<td>27</td>
<td>Q</td>
<td><img src="image55" alt="Fixed-Width Display" /></td>
<td><img src="image56" alt="Proportional Display" /></td>
</tr>
<tr>
<td>28</td>
<td>R</td>
<td><img src="image57" alt="Fixed-Width Display" /></td>
<td><img src="image58" alt="Proportional Display" /></td>
</tr>
<tr>
<td>29</td>
<td>S</td>
<td><img src="image59" alt="Fixed-Width Display" /></td>
<td><img src="image60" alt="Proportional Display" /></td>
</tr>
<tr>
<td>30</td>
<td>T</td>
<td><img src="image61" alt="Fixed-Width Display" /></td>
<td><img src="image62" alt="Proportional Display" /></td>
</tr>
<tr>
<td>31</td>
<td>U</td>
<td><img src="image63" alt="Fixed-Width Display" /></td>
<td><img src="image64" alt="Proportional Display" /></td>
</tr>
<tr>
<td>No.</td>
<td>Character</td>
<td>Fixed-Width Display</td>
<td>Proportional Display</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>---------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>32</td>
<td>V</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>Z</td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>&amp;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>’</td>
<td></td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>(</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>47</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>48</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>53</td>
<td>&lt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>54</td>
<td>&gt;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>55</td>
<td>?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>56</td>
<td>@</td>
<td></td>
<td></td>
</tr>
<tr>
<td>57</td>
<td>[</td>
<td></td>
<td></td>
</tr>
<tr>
<td>58</td>
<td>\</td>
<td></td>
<td></td>
</tr>
<tr>
<td>59</td>
<td>]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>^</td>
<td></td>
<td></td>
</tr>
<tr>
<td>61</td>
<td>underscore</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Character</td>
<td>Fixed-Width Display</td>
<td>Proportional Display</td>
</tr>
<tr>
<td>-----</td>
<td>-----------</td>
<td>---------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>64</td>
<td>`</td>
<td></td>
<td></td>
</tr>
<tr>
<td>65</td>
<td>(</td>
<td></td>
<td></td>
</tr>
<tr>
<td>66</td>
<td>]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>67</td>
<td>)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>68</td>
<td>~</td>
<td></td>
<td></td>
</tr>
<tr>
<td>69</td>
<td>§</td>
<td></td>
<td></td>
</tr>
<tr>
<td>70</td>
<td>euro</td>
<td></td>
<td></td>
</tr>
<tr>
<td>71</td>
<td>£</td>
<td></td>
<td></td>
</tr>
<tr>
<td>72</td>
<td>¥</td>
<td></td>
<td></td>
</tr>
<tr>
<td>73</td>
<td>Å</td>
<td></td>
<td></td>
</tr>
<tr>
<td>74</td>
<td>Á</td>
<td></td>
<td></td>
</tr>
<tr>
<td>75</td>
<td>Æ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>76</td>
<td>Ç</td>
<td></td>
<td></td>
</tr>
<tr>
<td>77</td>
<td>ń</td>
<td></td>
<td></td>
</tr>
<tr>
<td>78</td>
<td>Ö</td>
<td></td>
<td></td>
</tr>
<tr>
<td>79</td>
<td>Ø</td>
<td></td>
<td></td>
</tr>
<tr>
<td>80</td>
<td>U</td>
<td></td>
<td></td>
</tr>
<tr>
<td>81</td>
<td>B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>82</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>83</td>
<td>Δ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>84</td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>85</td>
<td>”</td>
<td></td>
<td></td>
</tr>
<tr>
<td>86</td>
<td>&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>87</td>
<td>?</td>
<td>spade</td>
<td></td>
</tr>
<tr>
<td>88</td>
<td>?</td>
<td>club</td>
<td></td>
</tr>
<tr>
<td>89</td>
<td>?</td>
<td>heart</td>
<td></td>
</tr>
<tr>
<td>90</td>
<td>?</td>
<td>diamond</td>
<td></td>
</tr>
<tr>
<td>91</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>92</td>
<td>new moon</td>
<td></td>
<td></td>
</tr>
<tr>
<td>93</td>
<td>first quarter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>last quarter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>↓</td>
<td>down arrow</td>
<td></td>
</tr>
<tr>
<td>No.</td>
<td>Character</td>
<td>Fixed-Width Display</td>
<td>Proportional Display</td>
</tr>
<tr>
<td>-----</td>
<td>------------</td>
<td>---------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>96</td>
<td>↑</td>
<td><img src="image1" alt="Fixed-Width Display" /></td>
<td><img src="image2" alt="Proportional Display" /></td>
</tr>
<tr>
<td>97</td>
<td>^</td>
<td><img src="image3" alt="Fixed-Width Display" /></td>
<td><img src="image4" alt="Proportional Display" /></td>
</tr>
<tr>
<td>98</td>
<td>´</td>
<td><img src="image5" alt="Fixed-Width Display" /></td>
<td><img src="image6" alt="Proportional Display" /></td>
</tr>
<tr>
<td>99</td>
<td>M</td>
<td><img src="image7" alt="Fixed-Width Display" /></td>
<td><img src="image8" alt="Proportional Display" /></td>
</tr>
<tr>
<td>100</td>
<td>1</td>
<td><img src="image9" alt="Fixed-Width Display" /></td>
<td><img src="image10" alt="Proportional Display" /></td>
</tr>
<tr>
<td>101</td>
<td>?</td>
<td><img src="image11" alt="Fixed-Width Display" /></td>
<td><img src="image12" alt="Proportional Display" /></td>
</tr>
<tr>
<td>102</td>
<td>?</td>
<td><img src="image13" alt="Fixed-Width Display" /></td>
<td><img src="image14" alt="Proportional Display" /></td>
</tr>
<tr>
<td>103</td>
<td></td>
<td><img src="image15" alt="Fixed-Width Display" /></td>
<td><img src="image16" alt="Proportional Display" /></td>
</tr>
<tr>
<td>104</td>
<td>↓</td>
<td><img src="image17" alt="Fixed-Width Display" /></td>
<td><img src="image18" alt="Proportional Display" /></td>
</tr>
</tbody>
</table>
### Large Font Dot-Matrix Patterns

<table>
<thead>
<tr>
<th>No.</th>
<th>Character</th>
<th>Proportional Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td><img src="image1" alt="Proportional Display" /></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td><img src="image2" alt="Proportional Display" /></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td><img src="image3" alt="Proportional Display" /></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td><img src="image4" alt="Proportional Display" /></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td><img src="image5" alt="Proportional Display" /></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td><img src="image6" alt="Proportional Display" /></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td><img src="image7" alt="Proportional Display" /></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td><img src="image8" alt="Proportional Display" /></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td><img src="image9" alt="Proportional Display" /></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td><img src="image10" alt="Proportional Display" /></td>
</tr>
<tr>
<td>10</td>
<td>blank</td>
<td><img src="image11" alt="Proportional Display" /></td>
</tr>
<tr>
<td>11</td>
<td>A</td>
<td><img src="image12" alt="Proportional Display" /></td>
</tr>
<tr>
<td>12</td>
<td>I</td>
<td><img src="image13" alt="Proportional Display" /></td>
</tr>
<tr>
<td>13</td>
<td>L</td>
<td><img src="image14" alt="Proportional Display" /></td>
</tr>
<tr>
<td>14</td>
<td>P</td>
<td><img src="image15" alt="Proportional Display" /></td>
</tr>
<tr>
<td>15</td>
<td>S</td>
<td><img src="image16" alt="Proportional Display" /></td>
</tr>
<tr>
<td>16</td>
<td>T</td>
<td><img src="image17" alt="Proportional Display" /></td>
</tr>
<tr>
<td>17</td>
<td>.</td>
<td><img src="image18" alt="Proportional Display" /></td>
</tr>
<tr>
<td>18</td>
<td>:</td>
<td><img src="image19" alt="Proportional Display" /></td>
</tr>
<tr>
<td>19</td>
<td>dash</td>
<td><img src="image20" alt="Proportional Display" /></td>
</tr>
</tbody>
</table>
## Flag Display

**LCD_CLR_ALL_FLAGS**

**Description**  
Clear all the LCD flags.

**Usage**  
LCD_CLR_ALL_FLAGS

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA, IX, HL

**Example**  
; Turn off all flag Icons on LCD display  
LCD_CLR_ALL_FLAGS

---

## LCD_UPD_L_FLAG

**Description**  
Clears or displays the “L” icon.

**Usage**  
LCD_UPD_L_FLAG { ON | OFF }

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
HL

**Example**  
; Clear L flag, but display A flag  
LCD_UPD_L_FLAG OFF  
LCD_UPD_A_FLAG ON

---

## LCD_UPD_A_FLAG

**Description**  
Clears or displays the “A” icon.

**Usage**  
LCD_UPD_A_FLAG { ON | OFF }

**Assumptions**  
None

**Input**  
None
**Output**  
None

**Destroys**  
HL.

**Example**  
; Clear L flag, but display A flag
LCD_UPD_L_FLAG OFF
LCD_UPD_A_FLAG ON

---

**LCD_UPD_P_FLAG**

**Description**  
Clears or displays the “P” icon.

**Usage**  
LCD_UPD_P_FLAG{ ON | OFF }

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
HL.

**Example**  
; Clear L flag and P flags, but display A flag
LCD_UPD_L_FLAG OFF
LCD_UPD_A_FLAG ON
LCD_UPD_P_FLAG OFF

---

**LCD_UPD_LAP_FLAG**

**Description**  
Clears or displays the “LAP” an icon (a combination of the L, A, and P flags)

**Usage**  
LCD_UPD_LAP_FLAG{ ON | OFF }

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
HL.

**Example**  
; display the LAP icon
LCD_UPD_LAP_FLAG ON
**LCD_UPD_AP_FLAG**

Description  Clears or displays the “AP” icon combination of A/P flags.

Usage  LCD_UPD_AP_FLAG{ ON | OFF }

Assumptions  None

Input  None

Output  None

Destroys  HL

Example  ; Clear the AP flag, and display the L flag
          LCD_UPD_AP_FLAG OFF
          LCD_UPD_L_FLAG ON

**LCD_UPD_NOTE_FLAG**

Description  Clears or displays the “NOTE” icon.

Usage  LCD_UPD_NOTE_FLAG{ ON | OFF }

Assumptions  None

Input  None

Output  None

Destroys  HL

Example  ; Display the NOTE flag
          LCD_UPD_NOTE_FLAG ON

**LCD_UPD_ACLK_FLAG**

Description  Clears or displays the “ALARM CLOCK” icon.

Usage  LCD_UPD_ACLK_FLAG{ ON | OFF }

Assumptions  None

Input  None

Output  None
**Destroys**

| HL |

**Example**

; Clear the ALARM CLOCK flag

LCD_UPD_ACLK_FLAG OFF

---

**LCD_UPD_TAIL_FLAG**

**Description**

Clears or displays the “TAIL” icon.

**Usage**

LCD_UPD_TAIL_FLAG{ ON | OFF }

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

HL

**Example**

; Display the TAIL and the TIMER to indicate CDR function

LCD_UPD_TAIL_FLAG ON

LCD_UPD_TMR_FLAG ON

---

**LCD_UPD_ARROW_FLAG**

**Description**

Clears or displays the “ARROW” icon.

**Usage**

LCD_UPD_ARROW_FLAG{ ON | OFF }

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

HL

**Example**

; Clear the ARROW and TAIL, and display the TIMER to show CDS function

LCD_UPD_ARROW_FLAG OFF

LCD_UPD_TAIL_FLAG OFF

LCD_UPD_TMR_FLAG ON

---

**LCD_UPD_TMR_FLAG**

**Description**

Clears or displays the “TIMER” icon.

**Usage**

LCD_UPD_TMR_FLAG{ ON | OFF }

---

Timex Corporation
Assumptions  None
Input  None
Output  None
Destroys  HL
Example  ; Clear the TIMER flag to indicate the TIMER has stopped
         LCD_UPD_TMR_FLAG OFF

LCD_UPD_CDC_FLAG
Description  Clears or displays the “COUNT DOWN TO CHRONO” icon combination, which consists of the TIMER/ARROW/STOPWATCH flags.
Usage  LCD_UPD_CDC_FLAG{ ON | OFF }
Assumptions  None
Input  None
Output  None
Destroys  HL
Example  ; Display CDC flag to indicate CDC function is active.
         LCD_UPD_CDC_FLAG ON

LCD_UPD_CDR_FLAG
Description  Clears or displays the “COUNT DOWN AND REPEAT” icon combination of TIMER/TAIL flags.
Usage  LCD_UPD_CDR_FLAG{ ON | OFF }
Assumptions  None
Input  None
Output  None
Destroys  HL
Example  ; Display CDR flag to indicate CDR function is active.
         LCD_UPD_CDR_FLAG ON
**LCD_UPD_CHR_FLAG**

Description: Clears or displays the “STOPWATCH” icon.

Usage: LCD_UPD_CHR_FLAG{ ON | OFF }

Assumptions: None

Input: None

Output: None

Destroys: HL

Example: ; Display CHR flag to indicate CHRONO is active.
LCD_UPD_CHR_FLAG ON

**LCD_UPD_MOON_FLAG**

Description: Clears or displays the “MOON” icon.

Usage: LCD_UPD_MOON_FLAG{ ON | OFF }

Assumptions: None

Input: None

Output: None

Destroys: HL

Example: ; Clear the MOON flag to indicate NightMode is disabled.
LCD_UPD_MOON_FLAG OFF

**LCD_UPD_DASH1_FLAG**

Description: Clears or displays the “DASH 1” icon.

Usage: LCD_UPD_DASH1_FLAG{ ON | OFF }

Assumptions: None

Input: None
**Output**
None

**Destroys**
HL

**Example**
; Turn on the DASH between Segmented Digits 3 and 4

```
LCD_UPD_DASH1_FLAG ON
```

---

**LCD_UPD_PER1_FLAG**

**Description**
Clears or displays the “PERIOD 1” icon.

**Usage**
```
LCD_UPD_PER1_FLAG{ ON | OFF }
```

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
HL

**Example**
; Turn on the PERIOD between Segmented Digits 3 and 4

```
LCD_UPD_PER1_FLAG ON
```

---

**LCD_UPD_COLON1_FLAG**

**Description**
Clears or displays the “COLON 1” icon.

**Usage**
```
LCD_UPD_COLON1_FLAG{ ON | OFF }
```

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
HL

**Example**
; Turn on the COLON between Segmented Digits 3 and 4

```
LCD_UPD_COLON1_FLAG ON
```

---

**LCD_UPD_DASH2_FLAG**

**Description**
Clears or displays the “DASH 2” icon.
Usage
LCD_UPD_DASH2_FLAG{ ON | OFF }

Assumptions
None

Input
None

Output
None

Destroys
HL

Example
; Turn on the DASH between Segmented Digits 4 and 5
LCD_UPD_DASH2_FLAG ON

---

**LCD_UPD_PER2_FLAG**

Description
Clears or displays the “PERIOD 2” icon.

Usage
LCD_UPD_PER2_FLAG{ ON | OFF }

Assumptions
None

Input
None

Output
None

Destroys
HL

Example
; Turn on the PERIOD between Segmented Digits 4 and 5
LCD_UPD_PER2_FLAG ON

---

**LCD_UPD_COLON2_FLAG**

Description
Clears or displays the “COLON 2” icon.

Usage
LCD_UPD_COLON2_FLAG{ ON | OFF }

Assumptions
None

Input
None

Output
None

Destroys
HL

Example
; Turn on the COLON between Segmented Digits 4 and 5
LCD_UPD_COLON2_FLAG ON
Clearing Display

**LCD_CLR_2DIGIT_SEG**

**Description**: Clears two 9-segment character positions.

**Usage**: LCD_CLR_2DIGIT_SEG

**Assumptions**: None

**Input**: IX - Leftmost starting digit position we wish to clear.

**Output**: None

**Destroys**: A, B, L, IX

**Example**

```assembly
; Clear digits 3 and 4 in the segmented display area.
ld   IX, #LCDSEGDIGIT3
LCD_CLR_2DIGIT_SEG
```


**LCD_CLR_3DIGIT_SEG**

**Description**: Clears three consecutive 9-segment character positions

**Usage**: LCD_CLR_3DIGIT_SEG

**Assumptions**: None

**Input**: IX - Leftmost starting digit position we wish to clear.

**Output**: None

**Destroys**: A, B, L, IX

**Example**

```assembly
; Clear digits 1,2 and 3 in the segmented display area.
ld   IX, #LCDSEGDIGIT1
LCD_CLR_3DIGIT_SEG
```


**LCD_CLR_SMALL_PROP_WIDTH_2DIG_DM_DATA**
### Description
Clear a small-font, proportional-width 2-digit data.

### Usage
LCD_CLR_SMALL_PROP_WIDTH_2DIG_DM_DATA

### Assumptions
None

### Input
IX - Starting dot matrix column and line position.

### Output
None

### Destroys
A, B, L, IX

### Example
; Clear 2 digit space starting at column 20, in line 2 of the big dot matrix
ld IX, #LCDMAINDMLINE2COL20
LCD_CLR_SMALL_PROP_WIDTH_2DIG_DM_DATA

---

### Description
Clear a small-font, fixed-width 2-digit data.

### Usage
LCD_CLR_SMALL_FIXED_WIDTH_2DIG_DM_DATA

### Assumptions
None

### Input
IX - Starting dot matrix column and line position

### Output
None

### Destroys
A, B, L, IX

### Example
; Clear 2 digit space starting at column 20, in line 2 of the big dot matrix
ld IX, #LCDMAINDMLINE2COL20
LCD_CLR_SMALL_FIXED_WIDTH_2DIG_DM_DATA

---

### Description
Clear a small-font, proportional-width 3-digit data.

### Usage
LCD_CLR_SMALL_PROP_WIDTH_3DIG_DM_DATA

### Assumptions
None

### Input
IX - Starting dot matrix column and line position
**Output**
None

**Destroys**
A, B, L, IX

**Example**

```
; Clear 3 digit space starting at column 22, in line 1 of the big dot matrix
ld   IX, #LCDMAINDMLINE1COL22
LCD_CLR_SMALL_PROP_WIDTH_3DIG_DM_DATA
```

---

**LCD_CLR_SMALL_FIXED_WIDTH_3DIG_DM_DATA**

**Description**
Clear a small-font, fixed-width 3-digit data.

**Usage**
LCD_CLR_SMALL_FIXED_WIDTH_3DIG_DM_DATA

**Assumptions**
None

**Input**
IX - Starting dot matrix column and line position

**Output**
None

**Destroys**
A, B, L, IX

**Example**

```
; Clear 3 digit space starting at column 22, in line 1 of the big dot matrix
ld   IX, #LCDMAINDMLINE1COL22
LCD_CLR_SMALL_FIXED_WIDTH_3DIG_DM_DATA
```

---

**Clearing Large-Font Digits**

**LCD_CLR_BIG_2DIGIT_DM_DATA**

**Description**
Clears a large-font, 2-digit DM data.

**Usage**
LCD_CLR_BIG_2DIGIT_DM_DATA

**Assumptions**
None

**Input**
IX - Starting dot matrix column position

**Output**
None
Destroys  A, B, L, IX

Example  ; Clear 2 digit space starting at column 21, of the big dot matrix
          ld   IX, #LCDBIGCHARDMCOL21
          LCD_CLR_BIG_2DIGIT_DM_DATA

LCD_CLR_BIG_3DIGIT_DM_DATA

Description  Clears a large-font, 3-digit DM data.
Usage  LCD_CLR_BIG_3DIGIT_DM_DATA
Assumptions  None
Input  IX - Starting dot matrix column position
Output  None
Destroys  A, B, L, IX
Example  ; Clear 3 digit space starting at column 21, of the big dot matrix
          ld   IX, #LCDBIGCHARDMCOL21
          LCD_CLR_BIG_3DIGIT_DM_DATA

Clearing Display Regions

LCD_CLR_DISPLAY

Description  Clears the entire LCD.
Usage  LCD_CLR_DISPLAY
Assumptions  None
Input  None
Output  None
Destroys  BA, HL
Example  ; Clear the entire display
          LCD_CLR_DISPLAY
**LCD_FILL_DISPLAY**

Description: Display all segments in the LCD.

Usage: LCD_FILL_DISPLAY

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: ; Turn on all segments of the LCD display
         LCD_FILL_DISPLAY

**LCD_CLR_SEG_LINE**

Description: Clears the entire 9-segment line. This does NOT include the punctuation.

Usage: LCD_CLR_SEG_LINE

Assumptions: None

Input: None

Output: None

Destroys: A, B, L, IX

Example: ; Clear all 6 9-segment digits, not the punctuation
         LCD_UPD_PER2_FLAG ON
         LCD_CLR_SEG_LINE

**LCD_CLEAR_UPPER_DM**

Description: Clear upper-dot matrix area.

Usage: LCD_CLEAR_UPPER_DM

Assumptions: None
Input: None
Output: None
Destroys: A, IX

Example:
; Clear the upper dot matrix display
LCD_UPD_PER2_FLAG ON

**LCD_CLEAR_MAIN_DM**

Description: Clear main dot matrix area.
Usage: LCD_CLEAR_MAIN_DM
Assumptions: None
Input: None
Output: None
Destroys: A, IX

Example:
; Clear the main dot matrix display
LCD_CLEAR_MAIN_DM

**LCD_CLR_MAIN_DM_LINE1**

Description: Clear line 1 of the main dot matrix area.
Usage: LCD_CLR_MAIN_DM_LINE1
Assumptions: None
Input: None
Output: None
Destroys: A, IX

Example:
; Clear line 1 ONLY of the main dot matrix display. Line 2 is unaffected.
LCD_CLR_MAIN_DM_LINE1
**LCD_CLR_MAIN_DM_LINE2**

**Description**
Clear line 2 of the main dot matrix area.

**Usage**
LCD_CLR_MAIN_DM_LINE2

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
A, IX

**Example**

```
; Clear line 2 ONLY of the main dot matrix display. Line 1 is unaffected.
LCD_CLR_MAIN_DM_LINE2
```

---

**Data Formatting**

**LCD_FORMAT_3DIGIT_DATA_SUP_ZERO**

**Description**
Format a 3-digit BCD data by deleting all leading zeros.

**Usage**
LCD_FORMAT_3DIGIT_DATA_SUP_ZERO

**Assumptions**
None

**Input**
- B - 100's digit BCD data
- A - 10's digit BCD data (high nibble)
  1's digit BCD data (low nibble)

**Output**
- LCDFormatData+2
- LCDFormatData+1
- LCDFormatData+0
  - Contain the formatted 3 digit data, from most significant digit to least significant digit.

**Destroys**
BA

**Example**

```
; Format the BCD value 0x10 for display. This will result in a space followed by "10"
; Set BA = 0x010
ld   B, #0
ld   A, #10H
LCD_FORMAT_3DIGIT_DATA_SUP_ZERO
```

---

Timex Corporation 104
**LCD_FORMAT_2DIGIT_DATA_SUP_ZERO**

**Description**
Format a 2-digit BCD data by deleting all leading zeros.

**Usage**
LCD_FORMAT_2DIGIT_DATA_SUP_ZERO

**Assumptions**
None

**Input**
- A - 10's digit BCD data (high nibble)
- B - 1's digit BCD data (low nibble)

**Output**
- LCDFormatData+1 - Contain the formatted 2 digit data, from most significant digit to least significant digit.
- LCDFormatData+0

**Destroys**
BA

**Example**
; Format the BCD #0x89 for display
ld   A, #89H
LCD_FORMAT_2DIGIT_DATA_SUP_ZERO

---

**LCD_FORMAT_3DIGIT_DATA_NO_LSD_SUP**

**Description**
Format a 3-digit BCD data by deleting all leading zeros except on the LSD.

**Usage**
LCD_FORMAT_3DIGIT_DATA_NO_LSD_SUP

**Assumptions**
None

**Input**
- A - 10's digit BCD data (high nibble)
- B - 100's digit BCD data

**Output**
- LCDFormatData+2 - Formatted 3 digit data
- LCDFormatData+1
- LCDFormatData+0

**Destroys**
BA

**Example**
; Format the BCD value 0x000 for display. After the macro call, this will be displayed as 2 spaces followed by a "0"
ld   B, #0
ld   A, #0

---

**LCD_FORMAT_2DIGIT_DATA_SUP_ZERO_MSD**
**Description**  Format a 2-digit BCD data by deleting the MSD if equal to zero.

**Usage**  
LCD_FORMAT_2DIGIT_DATA_SUP_ZERO_MSD

**Assumptions**  None

**Input**  
A - 10's digit BCD data (high nibble)  
B - 1's digit BCD data (low nibble)

**Output**  
LCDFormatData+1  Contain the formatted 2 digit data, from most significant digit to least significant digit.

LCDFormatData+0

**Destroys**  BA

**Example**

;format 2-digit number with suppression of MSB if zero
ld  A, #05H
LCD_FORMAT_2DIGIT_DATA_SUP_ZERO_MSD

;expected result:
;LCDFormatData[1] = SPACE
;LCDFormatData[0] = 5

;format 2-digit number with suppression of MSB if zero
ld  A, #25H
LCD_FORMAT_2DIGIT_DATA_SUP_ZERO_MSD

;expected result:
;LCDFormatData[1] = 2
;LCDFormatData[0] = 5

---

**Description**  Format a 3-digit BCD data with no zero suppression on all digits.

**Usage**  LCD_FORMAT_3DIGIT_DATA_NO_ZERO_SUP

**Assumptions**  None

**Input**  
B - 100's digit BCD data  
A - 10's digit BCD data (high nibble)  
1's digit BCD data (low nibble)

**Output**  
LCDFormatData+2  Contain the formatted 3 digit data, from most significant digit to least significant digit.

LCDFormatData+1

LCDFormatData+0

**Destroys**  BA

**Example**

;format 2-digit number with suppression of MSB if zero
ld  BA, #0123H
LCD_FORMAT_3DIGIT_DATA_NO_ZERO_SUP

;expected result:
;LCDFormatData[2] = SPACE
;LCDFormatData[1] = 1
;LCDFormatData[0] = 2
;LCDFormatData[3] = SPACE
LCD_FORMAT_2DIGIT_DATA_NO_ZERO_SUP

Description: Format a 2-digit BCD data with no zero suppression on all digits.

Usage: LCD_FORMAT_2DIGIT_DATA_NO_ZERO_SUP

Assumptions: None

Input:
A - 10’s digit BCD data (high nibble)
B - 1’s digit BCD data (low nibble)

Output:
LCDFormatData+1 - Contain the formatted 2 digit data, from most significant digit to least significant digit.

LCDFormatData+0

Destroys: BA

Example:
; Format the BCD value 0x00 for display. After the macro call, this will be displayed as “00”
ld A, #0
LCD_FORMAT_2DIGIT_DATA_NO_ZERO_SUP

Character Display

LCD_DISP_SEG_CHAR

Description: Display a segmented character.

Usage: LCD_DISP_SEG_CHAR

Assumptions: None

Input:
L - character to be displayed from 9-segment character set
IX - starting digit position

Output: None

Destroys: A, B, L
Example

; Display the Character “C” in digit 4 of the segmented display.
ld   L, #SEG_C
ld   IX, #LCDSEGDIGIT4
LCD_DISP_SEG_CHAR

---

**LCD_DISP_SMALL_PROP_WIDTH_DM_CHAR**

**Description**
Display small-font, proportional-width dot matrix character.

**Usage**
LCD_DISP_SMALL_PROP_WIDTH_DM_CHAR

**Assumptions**
None

**Input**
L - character to be displayed from the small font dot-matrix character set.
IX - starting DM column

**Output**
bOverflowFlag - 1 = DM display address is out of range
0 = DM address is in range (IXReg points to the next column)

**Destroys**
BA, H, IX

Example

; Display the Character “C” in column 4, line 2 of the main dot matrix.
ld   L, #DM5_C
ld   IX, #LCDMAINDMLINE2COL4
LCD_DISP_SMALL_PROP_WIDTH_DM_CHAR

---

**LCD_DISP_SMALL_FIXED_WIDTH_DM_CHAR**

**Description**
Display small-font, fixed-width dot matrix character.

**Usage**
LCD_DISP_SMALL_FIXED_WIDTH_DM_CHAR

**Assumptions**
None

**Input**
L - character to be displayed from the small font dot matrix character set.
IX - starting DM column

**Output**
bOverflowFlag - 1 = DM display address is out of range
0 = DM address is in range (IXReg points to the next column)

**Destroys**
BA, H, IX

Example

; Display the Character “C” in column 5, line 1 of the main dot matrix.
ld   L, #DM5_C
ld   IX, #LCDMAINDMLINE1COL5
LCD_DISP_SMALL_FIXED_WIDTH_DM_CHAR
Large-Font Character Display

**LCD_DISP_BIG_DM_CHAR**

**Description**
Display a large-font, dot matrix character.

**Usage**
LCD_DISP_BIG_DM_CHAR

**Assumptions**
None

**Input**
- L - character to be displayed from the large font dot matrix character set.
- IX - starting DM column

**Output**
- bOverflowFlag - 1 = DM display address is out of range
  0 = DM address is in range (IXReg points to the next column)

**Destroys**
L, IX

**Example**
; Display the large Character “S” in column 25 of the main dot matrix.
ld   L, #DM8_S
ld   IX, #LCDBIGCHARDMCOL25
LCD_DISP_BIG_DM_CHAR

---

Numeric Display

**LCD_DISP_2DIG_SEG_DATA_WITH_ZERO_SUP**

**Description**
Display a 2-digit BCD data with zero suppression on both digits in the 9-segment area.

**Usage**
LCD_DISP_2DIG_SEG_DATA_WITH_ZERO_SUP

**Assumptions**
None

**Input**
- A - packed BCD data to be displayed.
- IX - Leftmost 9-segment digit position where we wish to display the data

**Output**
None

**Destroys**
A, B, L, IX

**Example**
; Display the BCD value 0x00. After the macro call, this will be
; displayed as 2 spaces
ld   A, #0
ld   IX, #LCDSEGDIGIT1
LCD_DISP_2DIG_SEG_DATA_WITH_ZERO_SUP
**LCD_DISP_3DIG_SEG_DATA_WITH_ZERO_SUP**

**Description**
Display a 3-digit segmented BCD data with zero suppression on leading digits.

**Usage**
LCD_DISP_3DIG_SEG_DATA_WITH_ZERO_SUP

**Assumptions**
None

**Input**
- B - 100's digit BCD data
- A - packed 10's and 1's digit data
- IX - Leftmost 9-segment digit position where we wish to display the data

**Output**
None

**Destroys**
A, B, L, IX

**Example**
; Display the BCD value 0x000 After the macro call, this will be
; displayed as 3 spaces
ld B, #0
ld A, #0
ld IX, #LCDSEGDIGIT1
LCD_DISP_3DIG_SEG_DATA_WITH_ZERO_SUP

---

**LCD_DISP_2DIG_SEG_DATA_SUP_ZERO_MSD**

**Description**
Display a 2-digit BCD data with zero suppression on the MSD.

**Usage**
LCD_DISP_2DIG_SEG_DATA_SUP_ZERO_MSD

**Assumptions**
None

**Input**
- A - packed 10's and 1's digit data
- IX - Leftmost 9-segment digit position where we wish to display the data

**Output**
None

**Destroys**
A, B, L, IX

**Example**
; Display the BCD value 0x00. After the macro call, this will be
; displayed as 1 space followed by a "0"
ld A, #0
ld IX, #LCDSEGDIGIT1
LCD_DISP_2DIG_SEG_DATA_SUP_ZERO_MSD
# LCD_DISP_3DIG_SEG_DATA_NO_LSD_SUP

**Description**
Display a 3-digit segmented BCD data with zero suppression on all digits except the LSD.

**Usage**
LCD_DISP_3DIG_SEG_DATA_NO_LSD_SUP

**Assumptions**
None

**Input**
- B - 100's digit BCD data
- A - packed 10's and 1's digit data
- IX - Leftmost 9-segment digit position where we wish to display the data.

**Output**
None

**Destroys**
A, B, L, IX

**Example**
```asm
; Display the BCD value 0x000. After the macro call, this will be displayed as 2 spaces followed by a "0"
ld   B, #0
ld   A, #0
ld   IX, #LCDSEGDIGIT1
LCD_DISP_3DIG_SEG_DATA_NO_LSD_SUP
```

---

# LCD_DISP_2DIG_SEG_DATA_NO_ZERO_SUP

**Description**
Display a 2-digit BCD data with no zero suppression on both digits.

**Usage**
LCD_DISP_2DIG_SEG_DATA_NO_ZERO_SUP

**Assumptions**
None

**Input**
- A - packed 10's and 1's digit data
- IX - Leftmost 9-segment digit position where we wish to display the data.

**Output**
None

**Destroys**
A, B, L, IX

**Example**
```asm
; Display the BCD value 0x00. After the macro call, this will be displayed as "00"
ld   A, #0
ld   IX, #LCDSEGDIGIT1
LCD_DISP_2DIG_SEG_DATA_NO_ZERO_SUP
```
**LCD_DISP_3DIG_SEG_DATA_NO_ZERO_SUP**

**Description**
Display a 3-digit segmented BCD data with no zero suppression on all digits.

**Usage**
LCD_DISP_3DIG_SEG_DATA_NO_ZERO_SUP

**Assumptions**
None

**Input**
- B - 100's digit BCD data
- A - packed 10's and 1's digit data
- IX - Leftmost 9-segment digit position where we wish to display the data.

**Output**
None

**Destroys**
A, B, L, IX

**Example**

```asm
; Display the BCD value 0x000. After the macro call, this will be
; displayed as "000"
ld   B, #0
ld   A, #0
ld   IX, #LCDSEGDIGIT1
LCD_DISP_3DIG_SEG_DATA_NO_ZERO_SUP
```

---

**LCD_DISP_4DIG_SEG_DATA_WITH_ZERO_SUP**

**Description**
Display a 4-digit segmented data with zero suppression in the leftmost 4 9-segment display

**Usage**
LCD_DISP_4DIG_SEG_DATA_WITH_ZERO_SUP

**Assumptions**
None

**Input**
- BA - BCD 4 digit packed data to be displayed. B contains the 1000s and 100s digits, and A contains the 10s and 1s digits.

**Output**
None

**Destroys**
A, B, L, IX

**Example**

```asm
; Display the BCD value 0x0001. After the macro call, this will be
; displayed as 3 spaces followed by a "1" in digits 1 through 4
ld   B, #0
ld   A, #01H
ld   IX, #LCDSEGDIGIT1
LCD_DISP_4DIG_SEG_DATA_WITH_ZERO_SUP
```

---

**LCD_DISP_SMALL_PROP_WIDTH_2DI_G_DM_DATA_SUP_ZERO**
### LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO

**Description** Display small-font, proportional-width 2-digit data with zero suppression on both digits.

**Usage**

`LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO`

**Assumptions** None

**Input**

- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column and line where we wish to display the data.

**Output** None

**Destroys** BA, HL, IX

**Example**

```assembly
; Display the BCD value 0x01 in Line 2, column 20. After the macro call,
; this will be displayed as 1 space followed by "1"
ld A, #01H
ld IX, #LCDMAINDMLINE2COL20
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO
```

### LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_SUP_ZERO

**Description** Display small-font, fixed-width 2-digit data with zero suppression on both digits.

**Usage**

`LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_SUP_ZERO`

**Assumptions** None

**Input**

- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column and line where we wish to display the data.

**Output** None

**Destroys** BA, HL, IX

**Example**

```assembly
; Display the BCD value 0x00 in Line 1, column 23. After the macro call,
; this will be displayed as 2 spaces
ld A, #00H
ld IX, #LCDMAINDMLINE1COL23
LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_SUP_ZERO
```

### LCD_DISP_SMALL_PROP_WIDTH_3DIG_DM_DATA_SUP_ZERO

**Description** Display small-font, proportional-width 3-digit data with zero suppression on all leading digits.

**Usage**

`LCD_DISP_SMALL_PROP_WIDTH_3DIG_DM_DATA_SUP_ZERO`

**Assumptions** None

**Input** None

**Output** None

**Destroys** BA, HL, IX

**Example**

```assembly
; Display the BCD value 0x001 in Line 1, column 20. After the macro call,
; this will be displayed as 1 space followed by "01"
ld A, #001H
ld IX, #LCDMAINDMLINE1COL20
LCD_DISP_SMALL_PROP_WIDTH_3DIG_DM_DATA_SUP_ZERO
```
### Assumptions
None

### Input
- **B**: 100's digit BCD data
- **A**: Packed 10's and 1's digit BCD data
- **IX**: Leftmost dot matrix column and line where we wish to display the data.

### Output
None

### Destroys
BA, HL, IX

### Example
; Display the BCD value 0x007 in Line 1, column 23. After the macro call,
; this will be displayed as 2 spaces followed by a "7".
lde A, #07H
dl B, #00H
dl IX, #LCDMAINDMLINE1COL23
LCD_DISP_SMALL_PROP_WIDTH_3DIG_DM_DATA_SUP_ZERO

---

### LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_SUP_ZERO

**Description**
Display small-font, fixed-width 3-digit data with zero suppression on all leading digits.

**Usage**
LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_SUP_ZERO

**Assumptions**
None

**Input**
- **B**: 100's digit BCD data
- **A**: Packed 10's and 1's digit BCD data
- **IX**: Leftmost dot matrix column and line where we wish to display the data.

**Output**
None

**Destroys**
BA, HL, IX

**Example**
; Display the BCD value 0x007 in Line 1, column 23. After the macro call,
; this will be displayed as 2 spaces followed by a "7".
lde A, #07H
dl B, #00H
dl IX, #LCDMAINDMLINE1COL23
LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_SUP_ZERO

---

### LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO_MSD

**Description**
Display small-font, proportional-width 2-digit data with zero suppression on the MSD.

**Usage**
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO_MSD
**Assumptions**  None

**Input**  
- A  - Packed 10's and 1's digit BCD data
- IX  - Leftmost dot matrix column and line where we wish to display the data.

**Output**  None

**Destroys**  BA, HL, IX

**Example**  
; Display the BCD value 0x00 in Line 1, column 23. After the macro call, 
; this will be displayed as 1 space followed by “0”
ld A, #00H
ld IX, #LCDMAINDMLINE1COL23
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO_MSD

**Description**  
Display small-font, fixed-width 2-digit data with zero suppression on the MSD.

**Usage**  
LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_SUP_ZERO_MSD

**Assumptions**  None

**Input**  
- A  - Packed 10's and 1's digit BCD data
- IX  - Leftmost dot matrix column and line where we wish to display the data

**Output**  None

**Destroys**  BA, HL, IX

**Example**  
; Display the BCD value 0x00 in Line 1, column 23. After the macro call, 
; this will be displayed as 1 space followed by “0”
ld A, #00H
ld IX, #LCDMAINDMLINE1COL23
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_SUP_ZERO_MSD

**Description**  
Display small-font, proportional-width 3-digit data with suppression on leading zeros except the LSD.

**Usage**  
LCD_DISP_SMALL_PROP_WIDTH_3DIG_DM_DATA_NO_LSD_SUP

**Assumptions**  None

**Input**  
- B  - 100's digit BCD data
- A  - Packed 10's and 1's digit BCD data
- IX  - Leftmost dot matrix column and line where we wish to display the data
**Output**

None

**Destroys**

BA, HL, IX

**Example**

; Display the BCD value 0x000 in Line 1, column 23. After the macro call, this will be displayed as 2 spaces followed by a “0”.

ld A, #00H
ld B, #00H
ld IX, #LCDMAINDLINE1COL23

LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_LSD_SUP

---

**LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_LSD_SUP**

**Description**

Display small-font, fixed-width 3-digit data with suppression on leading zeros except LSD.

**Usage**

LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_LSD_SUP

**Assumptions**

None

**Input**

B - 100's digit BCD data
A - Packed 10's and 1's digit BCD data
IX - Leftmost dot matrix column and line where we wish to display the data

**Output**

None

**Destroys**

BA, HL, IX

**Example**

; Display the BCD value 0x000 in Line 1, column 23. After the macro call, this will be displayed as 2 spaces followed by a “0”.

ld A, #00H
ld B, #00H
ld IX, #LCDMAINDLINE1COL23

LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_LSD_SUP

---

**LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP**

**Description**

Display small-font, fixed-width 2-digit data with no suppression on both zeros.

**Usage**

LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP

**Assumptions**

None

**Input**

A - Packed 10's and 1's digit BCD data
IX - Leftmost dot matrix column and line where we wish to display the data
### Output
None

### Destroys
BA, HL, IX

### Example
```
; Display the BCD value 0x00 in Line 1, column23. After the macro call,
; this will be displayed as "00".
ld  A, #00H
ld  IX, #LCDMAINDMIDLINE1COL23
LCD_DISP_SMALL_PROP_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP
```

### LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP

**Description**
Display small-font, fixed-width 2-digit data with no suppression on both zeros.

**Usage**
`LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP`

**Assumptions**
None

**Input**
- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column and line where we wish to display the data

**Output**
None

**Destroys**
BA, HL, IX

**Example**
```
; Display the BCD value 0x00 in Line 1, column23. After the macro call,
; this will be displayed as "00".
ld  A, #00H
ld  IX, #LCDMAINDMIDLINE1COL23
LCD_DISP_SMALL_FIXED_WIDTH_2DIG_DM_DATA_NO_ZERO_SUP
```

### LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_ZERO_SUP

**Description**
Display small-font, fixed-width 3-digit data with no suppression on all zeros.

**Usage**
`LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_ZERO_SUP`

**Assumptions**
None

**Input**
- B - 100's digit BCD data
- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column and line where we wish to display the data

**Output**
None

**Destroys**
BA, HL, IX
Example

; Display the BCD value 0x000 in line 1, column23. After the macro call, this will be displayed as "000".
ld B, #00H
ld A, #00H
ld IX, #LCDMAINDM1COL23
LCD_DISP_SMALL_FIXED_WIDTH_3DIG_DM_DATA_NO_ZERO_SUP

Large-Font Numeric Display

**LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO**

**Description**
Display a large-font, 2-digit DM data with zero suppression on leading digit positions.

**Usage**
LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO

**Assumptions**
None

**Input**
- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column where we wish to display the data.

**Output**
None

**Destroys**
BA, HL, IX

**Example**

; Display the BCD value 0x08 in column23. After the macro call, this will be displayed as "_8" (space followed by 8)
ld A, #08H
ld IX, #LCDBIGCHARDMCOL23
LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO

**LCD_DISP_BIG_3DIGIT_DM_DATA_SUP_ZERO**

**Description**
Display a large-font, 3-digit DM data with zero suppression on leading digit positions.

**Usage**
LCD_DISP_BIG_3DIGIT_DM_DATA_SUP_ZERO

**Assumptions**
None

**Input**
- B - 100's digit BCD data
- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column where we wish to display the data.

**Output**
None
**LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO_MSD**

**Description**
Display a large-font, 2-digit DM data with zero suppression on the MSD.

**Usage**
LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO_MSD

**Assumptions**
None

**Input**
- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column where we wish to display the data.

**Output**
None

**Destroys**
BA, HL, IX

**Example**
; Display the BCD value 0x00 in column 23. After the macro call, this will be displayed as 1 space followed by a “0”.
ld   A, #00H
ld   IX, #LCDBIGCHARDMCOL23
LCD_DISP_BIG_2DIGIT_DM_DATA_SUP_ZERO_MSD

**LCD_DISP_BIG_3DIGIT_DM_DATA_NO_LSD_SUP**

**Description**
Display a large-font, 3-digit DM data with zero suppression on all digits except the LSD.

**Usage**
LCD_DISP_BIG_3DIGIT_DM_DATA_NO_LSD_SUP

**Assumptions**
None

**Input**
- B - 100's digit BCD data
- A - Packed 10's and 1's digit BCD data
- IX - Leftmost dot matrix column where we wish to display the data.

**Output**
None

**Destroys**
BA, HL, IX

**Example**
; Display the BCD value 0x000 in Line 1, column 23
; After the macro call, this will be displayed as 3 spaces.
ld   B, #00H
ld   A, #00H
ld   IX, #LCDBIGCHARDMCOL23
LCD_DISP_BIG_3DIGIT_DM_DATA_SUP_ZERO

Timex Corporation
Example

; Display the BCD value 0x000 in column23. After the macro call, this will
; be displayed as 2 spaces followed by “0”.
ld B, #00H
ld A, #00H
ld IX, #LCDBIGCHARDMCOL23
LCD_DISP_BIG_3DIGIT_DM_DATA_NO_LSD_SUP

LCD_DISP_BIG_2DIGIT_DM_DATA_NO_ZERO_SUP

Description
Display a large-font, 2-digit DM data with no zero suppression on all digits.

Usage
LCD_DISP_BIG_2DIGIT_DM_DATA_NO_ZERO_SUP

Assumptions
None

Input
A - Packed 10's and 1's digit BCD data
IX - Leftmost dot matrix column where we wish to display the data.

Output
None

Destroys
BA, HL, IX

Example
; Display the BCD value 0x00 in column23. After the macro call, this will
; be displayed as “00”.
ld A, #00H
ld IX, #LCDBIGCHARDMCOL23
LCD_DISP_BIG_2DIGIT_DM_DATA_NO_ZERO_SUP

LCD_DISP_BIG_3DIGIT_DM_DATA_NO_ZERO_SUP

Description
Display a large-font, 3-digit DM data with no zero suppression on all digits.

Usage
LCD_DISP_BIG_3DIGIT_DM_DATA_NO_ZERO_SUP

Assumptions
None

Input
B - 100's digit BCD data
A - Packed 10's and 1's digit BCD data
IX - Leftmost dot matrix column where we wish to display the data.

Output
None

Destroys
BA, HL, IX

Example
; Display the BCD value 0x000 in column23. After the macro call, this will
; be displayed as “000”.
ld B, #00H
ld   A, #00H
ld   IX, #LCDBIGCHARDMCOL23
LCD_DISP_BIG_3DIGIT_DM_DATA_NO_ZERO_SUP

Message Display

**LCD_DISP_SEG_LINE_MSG**

**Description**
Display a 6-character segmented message.

**Usage**
LCD_DISP_SEG_LINE_MSG

**Assumptions**
None

**Input**
IY - starting address of the message

The message structure is formatted as follows:

db  char_1, char_2, … , char_6

For example:

Seg_Message:
    db SEG_S, SEG_P, SEG_L, SEG_I, SEG_T, SEG_SPACE_
Here, IY would be set to whatever address the label “Seg_message” assembled to.

**Output**
None

**Destroys**
A, B, L, IX

**Example**

; Display “SPLIT” message
ld   IY, #lcdSegMsg_SPLIT
LCD_DISP_SEG_LINE_MSG

**LCD_DISP_FORMATTED_SMALL_PROP_WIDTH_DM MSG**

**Description**
Display a formatted, small-font, proportional-width dot-matrix message.

**Usage**
LCD_DISP_FORMATTED_SMALL_PROP_WIDTH_DM_MSG

**Assumptions**
None

**Input**
IY - starting address of the message

The message structure is formatted as follows:
```
dw \{display_address,ie columnn and line to start message at\}
db \{number_of_characters\}, char_1, char_2, ... , char_N

For example:

dw LCDMAINDMLINE2COL4
db 5, DM5_S, DM5_P, DM5_L, DM5_I, DM5_T

Output None
Destroys BA, HL, IX
Example ; Display the stored message “ENTER” in line 1, column 8
        ld IY, #lcdSmDMMsg_ENTER
        LCD_DISP_FORMATTED_SMALL_FIXED_WIDTH_DM_MSG
```

---

```
LCD_DISP_FORMATTED_SMALL_FIXED_WIDTH_DM_MSG

Description Display a formatted, small-font, fixed-width dot-matrix message.
Usage LCD_DISP_FORMATTED_SMALL_FIXED_WIDTH_DM_MSG
Assumptions None
Input IY - starting address of the message

The message structure is formatted as follows:

dw \{display_address ie, columnn and line to start message at\}
db \{number_of_characters\}, char_1, char_2, ... , char_N

For example:

DM_Message:

dw LCDMAINDMLINE2COL4
db 5, DM5_S, DM5_P, DM5_L, DM5_I, DM5_T

Here, IY would be set to whatever address the label “DM_Message” assembled to.

Output None
Destroys BA, HL, IX
Example ; Display the stored message “ENTER” in line 1, column 8
        ld IY, #lcdSmDMMsg_ENTER
        LCD_DISP_FORMATTED_SMALL_FIXED_WIDTH_DM_MSG
```
**LCD_DISP_UNFORMATTED_SMALL_PROP_WIDTH_DM_MSG**

**Description**
Display an unformatted, small-font, proportional-width dot-matrix message.

**Usage**
LCD_DISP_UNFORMATTED_SMALL_PROP_WIDTH_DM_MSG

**Assumptions**
None

**Input**
- IY - starting address of the message
- IX - starting display address
- B - Number of characters to be displayed

The message structure is formatted as follows:

```
db  char_1, char_2, ... , char_N
```

For example:

**DM_Message:**
```
db  DM5_S, DM5_P, DM5_L, DM5_I, DM5_T
```

Here, IY would be set to whatever address the label “DM_Message” assembled to.

**Output**
None

**Destroys**
BA, HL, IX

**Example**

```
; Display the stored message “ENTER” in line 1, column 8
ld   IY, #(lcdSmDMMsg_ENTER +3)
ld   IX, #LCDMAINDMLINE1COL8
ld   B, #5
LCD_DISP_UNFORMATTED_SMALL_PROP_WIDTH_DM_MSG
```

---

**LCD_DISP_UNFORMATTED_SMALL_FIXED_WIDTH_DM_MSG**

**Description**
Display a unformatted, small-font, fixed-width dot-matrix message.

**Usage**
LCD_DISP_UNFORMATTED_SMALL_FIXED_WIDTH_DM_MSG

**Assumptions**
None

**Input**
- IY - starting address of the message
- IX - Leftmost address of the LCD, column and line, where we wish to display the message
- B - Number of characters to be displayed

The message structure is formatted as follows:

```
db  char_1, char_2, ... , char_N
```

**Example**

```
; Display the stored message “ENTER” in line 1, column 8
ld   IY, #(lcdSmDMMsg_ENTER +3)
ld   IX, #LCDMAINDMLINE1COL8
ld   B, #5
LCD_DISP_UNFORMATTED_SMALL_FIXED_WIDTH_DM_MSG
```
For example:
DM_Message:
  db  DM5_S, DM5_P, DM5_L, DM5_I, DM5_T

Here, IY would be set to whatever address the label “DM_Message” assembled to.

Output  None
Destroys  BA, HL, IX

Example  ; Display the stored message “ENTER” in line 1, column 8
          ld  IY, #(lcdSmDMMsg_ENTER +3)
          ld  IX, #LCDMAINDMLINE1COL8
          ld  B, #5
          LCD_DISP_UNFORMATTED_SMALL_FIXED_WIDTH_DM_MSG

Banner Message Display

**LCD_DISP_BANNER_MSG**

Description  Display a banner message.

Usage  LCD_DISP_BANNER_MSG

Assumptions  None

Input  IY - starting address of the banner message structure

The banner message has control codes embedded in the message that either indicates the column position and the end of the message. The table below shows the control codes:

<table>
<thead>
<tr>
<th>Control Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LCDBANNER_COLn</strong></td>
<td>Signals the starting column to display the message where ( n ) indicates the column position. This is always the first byte in the message. If the control code is present in the middle of the message array, this will signify the starting column in the second line.</td>
</tr>
<tr>
<td><strong>LCD_END_BANNER</strong></td>
<td>Signals the end of the banner message.</td>
</tr>
</tbody>
</table>

Sample code for a two-line mode banner message:

```assembly
lcdBannerMsg_TIME_OF_DAY:
  db  LCDBANNER_COL12
  db  DM5_T, DM5_I, DM5_M, DM5_E
  db  LCDBANNER_COL6
```
Sample code for a one-line mode banner message:

```assembly
lcdBannerMsg_CHRONO:
  db   LCBANNER_COL5
  db   DM5_C, DM5_H, DM5_R, DM5_O, DM5_N, DM5_O
  db   LCD_END_BANNER

IY would be set to the address lcdBannerMsg_CHRONO assembles to.
```

Sample code for a one-line mode banner message in the second line:

```assembly
lcdBannerMsg_CHRONO:
  db   LCBANNER_COL1
  db   LCBANNER_COL5
  db   DM5_C, DM5_H, DM5_R, DM5_O, DM5_N, DM5_O
  db   LCD_END_BANNER

IY would be set to the address lcdBannerMsg_CHRONO assembles to.
```

Output
None

Destroys
BA, HL, IX

Example
See explanations above under "INPUT".

---

**Large-Font Message Display**

**LCD_DISP_FORMATTED_BIG_FONT_DM_MSG**

**Description**
Display a formatted, large-font, dot matrix message.

**Usage**
`LCD_DISP_FORMATTED_BIG_FONT_DM_MSG`

**Assumptions**
None

**Input**
`IY` - starting address of the message

The message structure is formatted as follows:

```assembly
dw  (display_address, ie, leftmost column message starts at )
db  (number_of_characters), char_1, char_2, ... , char_N
```

For example:
DM_BIG_Message:

dw  LCDBIGCHARDMCOL3
    db  5, DM8_S, DM8_P, DM8_L, DM8_I, DM8_T

IY would be set to the address DM_BIG_Message assembles to.

Output     None
Destroys   BA, HL, IX
Example    ; Display the stored message “SPLIT” in column 1
            ld   IY, #lcdBigDDMmsg_SPLIT
            LCD_DISP_FORMATTED_BIG_FONT_DM_MSG

LCD_DISP_UNFORMATTED_BIG_FONT_DM_MSG

Description Display an unformatted, large-font, dot matrix message.
Usage       LCD_DISP_UNFORMATTED_BIG_FONT_DM_MSG
Assumptions None
Input

| IY     | - starting address of the message |
| IX     | - Leftmost column on LCD that the message is displayed to. |
| B      | - Number of characters to be displayed |

The message structure is formatted as follows:

db  char_1, char_2, ... , char_N

For example:

DM_BIG_Message:

db  DM8_S, DM8_L, DM8_I, DM8_T

IY would be set to the address DM_BIG_Message assembles to.

Output     None
Destroys   BA, HL, IX
Example    ; Display the stored message “SPLIT” in column 1
            ld   IY, #lcdBigDDMmsg_SPLIT +3
            ld   IX, #LCDBIGCHARDMCOL1
            ld   B, #5
            LCD_DISP_UNFORMATTED_BIG_FONT_DM_MSG
Icon/Flag Blinking

**LCD_SET_BLINK_FLAG_STATUS**

**Description**
Sets a blink flag bit when an application requests flag to be blinked in foreground at a 2Hz rate.

**Usage**

```
LCD_SET_BLINK_FLAG_STATUS {flag_name}
```

**Assumptions**
None

**Input**

```
flag_name
```
- Resource icon to be blinked

The following definitions are available for the `flag_name`:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR_FLAG</td>
</tr>
<tr>
<td>MOON_FLAG</td>
</tr>
<tr>
<td>ACLK_FLAG</td>
</tr>
<tr>
<td>ARROW_FLAG</td>
</tr>
<tr>
<td>TAIL_FLAG</td>
</tr>
<tr>
<td>TMR_FLAG</td>
</tr>
<tr>
<td>NOTE_FLAG</td>
</tr>
<tr>
<td>P_FLAG</td>
</tr>
<tr>
<td>A_FLAG</td>
</tr>
<tr>
<td>L_FLAG</td>
</tr>
</tbody>
</table>

**Output**
None

**Destroys**
BA

**Example**

```
; To blink only one icon:
; Blink the CHRONO flag and then Set the core request flags to
; enable 2Hz blinking
LCD_SET_BLINK_FLAG_STATUS   CHR_FLAG
CORE_ENABLE_2HZ_BLINKING

; Alternative API. . .
; Use the API below that replaces the two lines in the example above
CORE_REQ_BLINK_2HZ CHR_FLAG

; To blink more than one icon:
LCD_SET_BLINK_FLAG_STATUS   CHR_FLAG
LCD_SET_BLINK_FLAG_STATUS   TMR_FLAG
CORE_ENABLE_2HZ_BLINKING
```
**LCD_CLR_BLINK_FLAG_STATUS**

**Description**  
Clears a blink flag bit to stop blinking the specified flag.

**Usage**  
`LCD_CLR_BLINK_FLAG_STATUS [flag_name]`

**Assumptions**  
None

**Input**  
`flag_name` - flag whose blinking is to be halted

The following definitions are available for the `flag_name`:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHR_FLAG</td>
</tr>
<tr>
<td>MOON_FLAG</td>
</tr>
<tr>
<td>ACLK_FLAG</td>
</tr>
<tr>
<td>ARROW_FLAG</td>
</tr>
<tr>
<td>TAIL_FLAG</td>
</tr>
<tr>
<td>TMR_FLAG</td>
</tr>
<tr>
<td>NOTE_FLAG</td>
</tr>
<tr>
<td>P_FLAG</td>
</tr>
<tr>
<td>A_FLAG</td>
</tr>
<tr>
<td>L_FLAG</td>
</tr>
</tbody>
</table>

**Output**  
None

**Destroys**  
BA

**Example**  
`; Stop blinking the CHRONO and TIMER flags
LCD_CLR_BLINK_FLAG_STATUS CHR_FLAG
LCD_CLR_BLINK_FLAG_STATUS TMR_FLAG`

---

**LCD_CANCEL_ALL_BLINK_FLAGS**

**Description**  
Clears all the blink status flags.

**Usage**  
`LCD_CANCEL_ALL_BLINK_FLAGS`

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA

**Example**  
`; Stop all flag blinking
LCD_CANCEL_ALL_BLINK_FLAGS`
Blinking

**LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_ADDR**

**Description** Specify the display routine for the 4Hz generic blinking.

**Usage** LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_ADDR \{display\_routine\}

**Assumptions** None

**Input**

- \textit{display\_routine} - address of the function that displays the required data

For example, if we wish to blink the ALARM flag at 4Hz, we could set up two routines, one to turn the flag ON, and one to turn the flag OFF, and call them alternately at a 4Hz rate.

Below is the routine to turn it on:

BlinkAlarm\_ON:

\begin{verbatim}
LCD\_UPD\_ACLIK\_FLAG ON
ret
\end{verbatim}

Similarly, to turn the flag OFF:

Below is the routine to turn it on:

BlinkAlarm\_Off:

\begin{verbatim}
LCD\_UPD\_ACLIK\_FLAG OFF
ret
\end{verbatim}

To execute these routines, we need to load their addresses into the buffer used to tell the generic 4Hz blink routine what routine should be called to do the blinking. We call the macro described here to load the ON routine, with the address the routine assembles at as the parameter.

\begin{verbatim}
LCD\_WRITE\_4HZ\_GEN\_BLINK\_DISP\_ROUTINE\_ADDR BlinkAlarm\_ON
LCD\_WRITE\_4HZ\_GEN\_BLINK\_CLR\_ROUTINE\_ADDR BlinkAlarm\_OFF
\end{verbatim}

**Output** None

**Destroys** BA

**Example**

; Load address of alarm routine into BLINK ON area.

\begin{verbatim}
LCD\_WRITE\_4HZ\_GEN\_BLINK\_DISP\_ROUTINE\_ADDR BlinkAlarm\_ON
\end{verbatim}

**LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_ADDR**
Description
Specify the clear routine for the 4Hz generic blinking.

Usage
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_ADDR {clear_routine}

Assumptions
None

Input
clear_routine - address of the function that clears the required data

For example, if we wish to blink the ALARM flag at 4Hz, we could set up two routines, one to turn the flag ON, and one to turn the flag OFF, and call them alternately at a 4Hz rate.

Below is the routine to turn it on;

BlinkAlarm_ON:
  LCD_UPD_ACLK_FLAG ON
  ret

Similarly, to turn the flag OFF;

BlinkAlarm_Off:
  LCD_UPD_ACLK_FLAG OFF
  ret

To execute these routines, we need to load their addresses into the buffer used to tell the generic 4Hz blink routine what routine should be called to do the blinking. We call the macro described here to load the ON routine, with the address the routine assembles at as the parameter.

LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_ADDR BlinkAlarm_ON
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_ADDR BlinkAlarm_OFF

Output
None

Destroys
BA

Example
; Load the CLEAR routine address for the ALARM flag
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_ADDR BlinkAlarm_OFF

---

LCD_WRITE_4HZ_GEN_BLINK_POSITION

Description
Specify the position for blinking.

Usage
LCD_WRITE_4HZ_GEN_BLINK_POSITION {blink_position}

Assumptions
None

Input
 blink_position - Position for blinking data

This routine loads an address on the LCD into a buffer in RAM. This location can be used by the
generic 4Hz blink routine to blink a specific address or group of addresses. The blink routines the user writes are NOT restricted to using this address only. The routine may specify any address within itself.

Output
None

Destroys
BA

Example
; Tell the 4Hz routine that it will blink the ALARM flag.
LCD_WRITE_4HZ_GEN_BLINK_POSITION ACLK_ADDR

---

**LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_PRELOADED**

**Description**
Specifies in register BA the display routine used for the generic blinking.

**Usage**
LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_PRELOADED

**Assumptions**
None

**Input**
BA - 16-bit address of the display routine

**Output**
None

**Destroys**
BA

**Example**
; Set BLINK ON routine address into BA
ld BA, # BlinkAlarm_ON
; Call MACRO to load BA into address area
LCD_WRITE_4HZ_GEN_BLINK_DISP_ROUTINE_PRELOADED

---

**LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_PRELOADED**

**Description**
Specifies in register BA the clear routine used for the generic blinking.

**Usage**
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_PRELOADED

**Assumptions**
None

**Input**
BA - 16-bit address of the clear routine

**Output**
None

**Destroys**
BA

**Example**
; Set BLINK OFF routine address into BA
ld BA, # BlinkAlarm_OFF
; Call MACRO to load BA into address area
LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_PRELOADED
### LCD_WRITE_4HZ_GEN_BLINK_CLR_ROUTINE_PRELOADED

**Description**
Specifies in register BA the blink position on the LCD used for the generic blinking.

**Usage**
LCD_WRITE_4HZ_GEN_BLINK_POSITION_PRELOADED

**Assumptions**
None

**Input**
BA - 16-bit blink positions

**Output**
None

**Destroys**
BA

**Example**
```assembly
; Load BLINK address into BA
ld BA, # ACLK_ADDR
; Call MACRO to load BA into position area
LCD_WRITE_4HZ_GEN_BLINK_POSITION_PRELOADED
```

### LCD_DISP_4HZ_DATA_FIRST

**Description**
Display data first when blinking generic data using the 4hz update.

**Usage**
LCD_DISP_4HZ_DATA_FIRST

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
HL

**Example**
```assembly
; Setup 4Hz blinking so we display data on the first blink, NOT the cleared position.
LCD_DISP_4HZ_DATA_FIRST
```
Scrolling
The message to be scrolled should have the DM5_SENTINEL as the last byte. This will indicate the end of the scrolling message. The code section below shows an example of a message that can be scrolled.

```
; Message "FOUR SCORES AND SEVEN YEARS AGO" + SENTINEL

GettysburgMessage:
        db   DM5_F, DM5_O, DM5_U, DM5_R, DM5_SPACE
        db   DM5_S, DM5_C, DM5_O, DM5_R, DM5_E, DM5_S, DM5_SPACE
        db   DM5_A, DM5_N, DM5_D, DM5_SPACE
        db   DM5_S, DM5_E, DM5_V, DM5_E, DM5_N, DM5_SPACE
        db   DM5_Y, DM5_E, DM5_A, DM5_G, DM5_O, DM5_SENTINEL
```

**LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE1**

**Description** Scroll a message from the ROM or RAM in the 1st line of the main DM area.

**Usage** LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE1 {EVENT_ON | EVENT_OFF}

**Assumptions** None

**Input**

- IY - starting address of the message to be scrolled

The macro parameter:

- EVENT_ON - generate a 'CORE_EVENT_END_OF_SCROLLING_MESS' event when the entire message has been scrolled
- EVENT_OFF - Doesn’t generate an event

**Output** None

**Destroys** BA, HL, IX

**Example**

```
; Scroll the message given in the example, FOUR SCORE AND SEVEN YEARS AGO.
; Get an event after the scrolling is completed.
    ld   IY, #GettysburgMessage
    LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE1     EVENT_ON
```

**LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE2**

**Description** Scroll a message from the ROM or RAM in the 2nd line of the main DM area.

**Usage** LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE2 {EVENT_ON | EVENT_OFF}

**Assumptions** None
### Input

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>IY</strong></td>
<td>starting address of the message to be scrolled</td>
<td></td>
</tr>
</tbody>
</table>

The macro parameter:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EVENT_ON</strong></td>
<td>generate a 'COREEVENT_END_OF_SCROLLING_MESS' event when the entire message has been scrolled</td>
</tr>
<tr>
<td><strong>EVENT_OFF</strong></td>
<td>Doesn’t generate an event</td>
</tr>
</tbody>
</table>

### Output

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td></td>
</tr>
</tbody>
</table>

### Destroys

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>BA, HL, IX</td>
<td></td>
</tr>
</tbody>
</table>

### Example

```asm
; Scroll the message given in the example, FOUR SCORE AND SEVEN YEARS AGO.
; Non’t get an event after the scrolling is completed.
ld IY, #GettysburgMessage
LCD_SCROLL_RAM_OR_ROM_MSG_MAIN_DM_LINE2 EVENT_OFF
```

---

### LCD_PAUSE_SCROLLING

**Description**

Pause scrolling.

**Usage**

```
LCD_PAUSE_SCROLLING
```

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

HL

**Example**

```asm
; Temporarily halt the scrolling operation
LCD_PAUSE_SCROLLING
```

---

### LCD_RESUME_SCROLLING

**Description**

Resume scrolling operation.

**Usage**

```
LCD_RESUME_SCROLLING
```

**Assumptions**

None

**Input**

None

**Output**

None
**DESTROY**

```
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_SCROLL_MSG_LEFT</td>
<td>Scroll a message to the left.</td>
</tr>
<tr>
<td>LCD_SCROLL_MSG_RIGHT</td>
<td>Scroll a message to the right.</td>
</tr>
</tbody>
</table>

**Assumptions**

Automatic scrolling is not active and all scroll parameters are set.

**Input**

- A - number of pixel columns to be scrolled \(1 - 14\)

**Output**

None

**Example**

```
; Restart the scrolling operation
LCD_RESUME_SCROLLING

; Pause the message
LCD_PAUSE_SCROLLING

; Now scroll the message left by 5 pixel columns
ld   A, #5
LCD_SCROLL_MSG_LEFT

; Pause the message
LCD_PAUSE_SCROLLING

; Now scroll the message right by 5 pixel columns
ld   A, #5
LCD_SCROLL_MSG_RIGHT
```
Primary Mode Icon Resource

**LCD_UPDATE_TOD_FLAG_RESOURCE_STATE**

**Description**  
Update the state of a TOD flag resource.

**Usage**  
`LCD_UPDATE_TOD_FLAG_RESOURCE_STATE [flag_name], {flag_state}`

**Assumptions**  
None

**Input**  
- `A` - The application index of calling function. This is used to check if the specified application index is the owner of the lcd icon resource.
- `flag_name` - Resource icon to be updated
- `flag_state` - Specifies how the icon is to be displayed.

The following definitions are available for the `flag_name`:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOON_RSRC_FLAG</td>
</tr>
<tr>
<td>BATT_RSRC_FLAG</td>
</tr>
<tr>
<td>NOTE_RSRC_FLAG</td>
</tr>
<tr>
<td>HAND_RSRC_FLAG</td>
</tr>
<tr>
<td>ACLK_RSRC_FLAG</td>
</tr>
<tr>
<td>ARROW_RSRC_FLAG</td>
</tr>
<tr>
<td>TAIL_RSRC_FLAG</td>
</tr>
<tr>
<td>TMR_RSRC_FLAG</td>
</tr>
<tr>
<td>CHR_RSRC_FLAG</td>
</tr>
<tr>
<td>P_RSRC_FLAG</td>
</tr>
<tr>
<td>A_RSRC_FLAG</td>
</tr>
<tr>
<td>L_RSRC_FLAG</td>
</tr>
</tbody>
</table>

The following definitions are available for the `flag_state`:

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLINK_ON</td>
</tr>
<tr>
<td>BLINKOFF_FLAGOFF</td>
</tr>
<tr>
<td>BLINKOFF_FLAGON</td>
</tr>
<tr>
<td>FLAG_ON</td>
</tr>
<tr>
<td>FLAG_OFF</td>
</tr>
</tbody>
</table>

**Output**  
None

**Destroys**  
BA

**Example**  
; Update the status of the MOON flag to BLINKING
; Get the current APP Index.
; We assume that the application wishing to modify the status is the
define application.
ld A, [CORECurrentMode]
; Update the flag status
LCD_UPDATE_TOD_FLAG_RESOURCE_STATE MOON_RSRC_FLAG, BLINK_ON
**LCD_UPDATE_TIMELINERESOURCE**

**Description**  
Update the Timeline resource data owned by the application.

**Usage**  
LCD_UPDATE_TIMELINERESOURCE

**Assumptions**  
None

**Input**  
- A: Application Index of the application which wants to modify the resource
- IY: Address of data to be copied into the resource status buffer. This data consists of one byte

**Output**  
None

**Destroys**  
BA, IX

**Example**  
; Get the current APP Index.  
; We assume that the application wishing to modify the status is the foreground application.  
ld A, [CORECurrentMode]  
ld IY, #APPTimelineData  
; Get the application’s TIMELINE data  
ld IY, #APPTimelineData  
; Call the macro to alter the timeline status  
LCD_UPDATE_TIMELINERESOURCE

---

**Pixel Operations**

**LCD_DISPLAY_MAIN_DM_PIXEL**

**Description**  
Display a pixel in the main DM area.

**Usage**  
LCD_DISPLAY_MAIN_DM_PIXEL

**Assumptions**  
None

**Input**  
- A: x-coordinate in the main DM area \{ 1 – 42 \}  
- B: y-coordinate in the main DM area \{ 1 – 11 \}

**Output**  
None

**Destroys**  
BA, HL

**Example**  
; Turn on Pixel (x,y) = (22, 8)  
ld A, #22  
ld B, #8  
LCD_DISPLAY_MAIN_DM_PIXEL
**LCD_CLEAR_MAIN_DM PIXEL**

**Description**  
Clears a pixel in the main DM area.

**Usage**  
LCD_CLEAR_MAIN_DM_PIXEL

**Assumptions**  
None

**Input**  
A - x-coordinate in the main DM area { 1 – 42 }
B - y-coordinate in the main DM area { 1 – 11 }

**Output**  
None

**Destroys**  
BA, HL

**Example**  
; Turn off Pixel (x,y) = (22, 8)
  ld A, #22
  ld B, #8
  LCD_CLEAR_MAIN_DM_PIXEL

---

**LCD_GET_STATE_OF_PIXEL**

**Description**  
Returns the status of a pixel column (displayed or cleared).

**Usage**  
LCD_GET_STATE_OF_PIXEL

**Assumptions**  
None

**Input**  
A - x-coordinate in the main DM area { 1 – 42 }
B - y-coordinate in the main DM area { 1 – 11 }

**Output**  
Zero Flag = 1 - if pixel is cleared
Zero Flag = 0 - if pixel is displayed

**Destroys**  
BA, HL

**Example**  
; Get status of pixel
  ld A, #22
  ld B, #8
  LCD_GET_STATE_OF_PIXEL
  ; Test Z flag to see if pixel is on
  jr Z, Exit
  ; If we get here, the pixel is OFF
Display Canned Messages

**LCD_DISP_UPPER_DM_MSG_ONE_HALF**

**Description**
Display '1/2' on the upper DM area.

**Usage**
LCD_DISP_UPPER_DM_MSG_ONE_HALF

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**

```
;display "1/2"
LCD_DISP_UPPER_DM_MSG_ONE_HALF
```

---

**LCD_DISP_UPPER_DM_MSG_TO**

**Description**
Display 'TO' on the upper DM area.

**Usage**
LCD_DISP_UPPER_DM_MSG_TO

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**

```
No Example
```

---

**LCD_DISP_UPPER_DM_MSG_LO**

**Description**
Display 'LO' on the upper DM area.

**Usage**
LCD_DISP_UPPER_DM_MSG_LO

**Assumptions**
None
**Description**
Display 'HOLD TO SWITCH' on the LCD.

**Usage**

```plaintext
LCD_DISP_SMALL_DM_MSG_HOLD_TO_SWITCH
```

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**Description**
Display 'NO APPT UPCOMING' on the LCD.

**Usage**

```plaintext
LCD_DISP_MSG_NO_APPT_UPCOMING
```

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example
### LCD_DISP_SMALL_DM_MSG_NO_OCCASION_UPCOMING

**Description**
Display 'NO OCCASION UPCOMING' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_NO_OCCASION_UPCOMING

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

### LCD_DISP_SMALL_DM_MSG_SET_TIME_SLASH_DATE

**Description**
Display 'SET TIME/DATE' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_SET_TIME_SLASH_DATE

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

### LCD_DISP_SMALL_DM_MSG_SHOW_DAY_OF_WEEK

**Description**
Display 'SHOW DAY OF WEEK' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_SHOW_DAY_OF_WEEK

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL
**LCD_DISP_SMALL_DM_MSG_SHOW_WEEK_NUMBER**

**Description**
Display 'SHOW WEEK NUMBER' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_SHOW_WEEK_NUMBER

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SMALL_DM_MSG_FREE_MEMORY_LAPS**

**Description**
Display 'FREE MEMORY LAPS' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_FREE_MEMORY_LAPS

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SMALL_DM_MSG_TO_STORE_TURN_CROWN**

**Description**
Display 'TO STORE TURN CROWN' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_TO_STORE_TURN_CROWN
Assumptions  None
Input          None
Output         None
Destroys       BA, HL
Example        No Example

LCD_DISP_SMALL_DM_MSG_MEMORY_FULL

Description    Display 'MEMORY FULL' on the LCD.
Usage          LCD_DISP_SMALL_DM(MSG_MEMORY_FULL)
Assumptions    None
Input          None
Output         None
Destroys       BA, HL
Example        No Example

LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLR_ALL

Description    Display 'HOLD TO CLR ALL' on the LCD.
Usage          LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLR_ALL
Assumptions    None
Input          None
Output         None
Destroys       BA, HL
Example        No Example
### LCD_DISP_SMALL_DM_MSG_HOLD_TO_RESET

**Description**
Display 'HOLD TO RESET' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_HOLD_TO_RESET

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

### LCD_DISP_SMALL_DM_MSG_WORKOUT_STORED

**Description**
Display 'WORKOUT STORED' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_WORKOUT_STORED

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

### LCD_DISP_SMALL_DM_MSG_RECALL_SLASH_FORMAT

**Description**
Display 'RECALL/FORMAT' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_RECALL_SLASH_FORMAT

**Assumptions**
None

**Input**
None

**Output**
None
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Usage</th>
<th>Assumptions</th>
<th>Input</th>
<th>Output</th>
<th>Destroys</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_DISP_SMALL_DM_MSG_FORMAT</td>
<td>Display 'FORMAT' on the LCD.</td>
<td>LCD_DISP_SMALL_DM_MSG_FORMAT</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
<tr>
<td>LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR_WORKOUT</td>
<td>Display 'HOLD TO CLEAR WORKOUT' on the LCD.</td>
<td>LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR_WORKOUT</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
<tr>
<td>LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR</td>
<td>Display 'HOLD TO CLEAR' on the LCD.</td>
<td>LCD_DISP_SMALL_DM_MSG_HOLD_TO_CLEAR</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
</tbody>
</table>
Assumptions: None
Input: None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP_SMALL_DM.MSG.YES.ROCK**

Description: Display 'YOU ROCK' on the LCD.
Usage: LCD_DISP_SMALL_DM.MSG.YES.ROCK
Assumptions: None
Input: None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP.MSG.LAP_SPLIT**

Description: Display 'LAP' on the 9-segment area and 'SPLIT' on the main dot-matrix area.
Usage: LCD_DISP.MSG.LAP_SPLIT
Assumptions: None
Input: None
Output: None
Destroys: BA, HL
Example: No Example
**LCD_DISP_MSG_SPLIT_LAP**

**Description**
Display 'SPLIT' on the 9-segment area and 'LAP' on the main dot-matrix area.

**Usage**
LCD_DISP_MSG_SPLIT_LAP

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_MSG_TIME_SPLIT**

**Description**
Display 'TIME' on the 9-segment area and 'SPLIT' on the main dot-matrix area.

**Usage**
LCD_DISP_MSG_TIME_SPLIT

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_LARGE_DM_STOP**

**Description**
Display 'STOP' in large characters on the main DM area.

**Usage**
LCD_DISP_LARGE_DM_STOP

**Assumptions**
None

**Input**
None

**Output**
None

Timex Corporation 147
LCD_DISP_LARGE_DM_TOTAL

Description: Display 'TOTAL' in large characters on the main DM area.

Usage: LCD_DISP_LARGE_DM_TOTAL

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

LCD_DISP_MSG_TIME_LAP

Description: Display 'TIME' on the 9-segment area and 'LAP' on the main dot-matrix area.

Usage: LCD_DISP_MSG_TIME_LAP

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

LCD_DISP_SMALL_DM_MSG_SET_TIMER

Description: Display 'SET TIMER' on the main dot-matrix area.
Usage       LCD_DISP_SMALL_DM_MSG_SET_TIMER
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example

**LCD_DISP_SMALL_DM_MSG_UNUSED_ENTRY**

Description  Display 'UNUSED ENTRY' on the LCD.
Usage         LCD_DISP_SMALL_DM_MSG_UNUSED_ENTRY
Assumptions   None
Input         None
Output        None
Destroys      BA, HL
Example       No Example

**LCD_DISP_SMALL_DM_MSG_UNUSED_ENTRIES**

Description  Display 'UNUSED ENTRIES' on the LCD.
Usage         LCD_DISP_SMALL_DM_MSG_UNUSED_ENTRIES
Assumptions   None
Input         None
Output        None
Destroys      BA, HL
Example       No Example
**LCD_DISP_SMALL_DM_MSG_STOP_AT_END**

Description: Display 'STOP AT END' on the LCD.

Usage: LCD_DISP_SMALL_DM_MSG_STOP_AT_END

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_SMALL_DM_MSG_REPEAT_AT_END**

Description: Display 'REPEAT AT END' on the LCD.

Usage: LCD_DISP_SMALL_DM_MSG_REPEAT_AT_END

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_SMALL_DM_MSG_CHRONO_AT_END**

Description: Display 'CHRONO AT END' on the LCD.

Usage: LCD_DISP_SMALL_DM_MSG_CHRONO_AT_END

Assumptions: None

Input: None

---
<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Usage</th>
<th>Assumptions</th>
<th>Input</th>
<th>Output</th>
<th>Destroys</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>LCD_DISP_SMALL_DM_MSG_SET_FORMAT</code></td>
<td>Display 'SET FORMAT' on the LCD.</td>
<td><code>LCD_DISP_SMALL_DM_MSG_SET_FORMAT</code></td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
<tr>
<td><code>LCD_DISP_SMALL_DM_MSG_SET_COUNTER</code></td>
<td>Display 'SET COUNTER' on the LCD.</td>
<td><code>LCD_DISP_SMALL_DM_MSG_SET_COUNTER</code></td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
<tr>
<td><code>LCD_DISP_SMALL_DM_MSG_COUNT_UP</code></td>
<td>Display 'COUNT UP' on the LCD.</td>
<td></td>
<td></td>
<td>None</td>
<td>None</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Usage        LCD_DISP_SMALL_DM_MSG_COUNT_UP
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example

LCD_DISP_SMALL_DM_MSG_COUNT_DOWN

Description  Display 'COUNT DOWN' on the LCD.
Usage        LCD_DISP_SMALL_DM_MSG_COUNT_DOWN
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example

LCD_DISP_SMALL_DM_MSG_HOLD_TO_DELETE

Description  Display 'HOLD TO DELETE' on the LCD.
Usage        LCD_DISP_SMALL_DM_MSG_HOLD_TO_DELETE
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example
**LCD_DISP_SMALL_DM_MSG_SET_ALARM**

Description: Display 'SET ALARM' on the LCD.

Usage: LCD_DISP_SMALL_DM_MSG_SET_ALARM

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_SMALL_DM_MSG_ENTRY_DELETED**

Description: Display 'ENTRY DELETED' on the LCD.

Usage: LCD_DISP_SMALL_DM_MSG_ENTRY_DELETED

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_SMALL_DM_MSG_SET_APPT**

Description: Display 'SET APPT' on the LCD.

Usage: LCD_DISP_SMALL_DM_MSG_SET_APPT

Assumptions: None

Input: None
Output  None
Destroys  BA, HL
Example  No Example

**LCD_DISP_SMALL_DM_MSG_APPT_DATE**

Description  Display 'APPT DATE' on the LCD.
Usage  LCD_DISP_SMALL_DM_MSG_APPT_DATE
Assumptions  None
Input  None
Output  None
Destroys  BA, HL
Example  No Example

**LCD_DISP_SMALL_DM_MSG_1ST_APPT_DATE**

Description  Display '1ST APPT DATE' on the LCD.
Usage  LCD_DISP_SMALL_DM_MSG_1ST_APPT_DATE
Assumptions  None
Input  None
Output  None
Destroys  BA, HL
Example  No Example

**LCD_DISP_SMALL_DM_MSG_EDIT_NOTE**

Description  Display 'EDIT_NOTE' on the LCD.
<table>
<thead>
<tr>
<th>Usage</th>
<th>LCD_DISP_SMALL_DM_MSG_EDIT_NOTE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assumptions</td>
<td>None</td>
</tr>
<tr>
<td>Input</td>
<td>None</td>
</tr>
<tr>
<td>Output</td>
<td>None</td>
</tr>
<tr>
<td>Destroys</td>
<td>BA, HL</td>
</tr>
<tr>
<td>Example</td>
<td>No Example</td>
</tr>
</tbody>
</table>

**LCD_DISP_SMALL_DM_MSG_BUTTON_BEEP_ON**

Description: Display 'BUTTON BEEP ON' on the LCD.

<table>
<thead>
<tr>
<th>Usage</th>
<th>LCD_DISP_SMALL_DM_MSG_BUTTON_BEEP_ON</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assumptions</td>
<td>None</td>
</tr>
<tr>
<td>Input</td>
<td>None</td>
</tr>
<tr>
<td>Output</td>
<td>None</td>
</tr>
<tr>
<td>Destroys</td>
<td>BA, HL</td>
</tr>
<tr>
<td>Example</td>
<td>No Example</td>
</tr>
</tbody>
</table>

**LCD_DISP_SMALL_DM_MSG_BUTTON_BEEP_OFF**

Description: Display 'BUTTON BEEP OFF' on the LCD.

<table>
<thead>
<tr>
<th>Usage</th>
<th>LCD_DISP_SMALL_DM_MSG_BUTTON_BEEP_OFF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assumptions</td>
<td>None</td>
</tr>
<tr>
<td>Input</td>
<td>None</td>
</tr>
<tr>
<td>Output</td>
<td>None</td>
</tr>
<tr>
<td>Destroys</td>
<td>BA, HL</td>
</tr>
<tr>
<td>Example</td>
<td>No Example</td>
</tr>
</tbody>
</table>
**LCD_DISP_SMALL_DM_MSG_NIGHTMODE_OFF**

**Description**  
Display 'NIGHTMODE OFF' on the LCD.

**Usage**  
LCD_DISP_SMALL_DM_MSG_NIGHTMODE_OFF

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA, HL

**Example**  
No Example

---

**LCD_DISP_SMALL_DM_MSG_NIGHTMODE_ON**

**Description**  
Display 'NIGHTMODE ON' on the LCD.

**Usage**  
LCD_DISP_SMALL_DM_MSG_NIGHTMODE_ON

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA, HL

**Example**  
No Example

---

**LCD_DISP_SMALL_DM_MSG_NIGHTMODE_AUTO**

**Description**  
Display 'NIGHTMODE AUTO' on the LCD.

**Usage**  
LCD_DISP_SMALL_DM_MSG_NIGHTMODE_AUTO

**Assumptions**  
None

**Input**  
None
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Usage</th>
<th>Assumptions</th>
<th>Input</th>
<th>Output</th>
<th>Destroys</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCD_DISP_SMALL_DM_MSG_CHIME_OFF</td>
<td>Display 'CHIME OFF' on the LCD.</td>
<td>LCD_DISP_SMALL_DM_MSG_CHIME_OFF</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
<tr>
<td>LCD_DISP_SMALL_DM_MSG_CHIME_ON</td>
<td>Display 'CHIME ON' on the LCD.</td>
<td>LCD_DISP_SMALL_DM_MSG_CHIME_ON</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
<tr>
<td>LCD_DISP_SMALL_DM_MSG_CHIME_AUTO</td>
<td>Display 'CHIME AUTO' on the LCD.</td>
<td></td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>BA, HL</td>
<td>No Example</td>
</tr>
</tbody>
</table>
Usage       LCD_DISP_SMALL_DM_MSG_CHIME_AUTO
Assumptions None
Input       None
Output      None
Destroys   BA, HL
Example    No Example

LCD_DISP_SMALL_DM_MSG_ON_OFF_TIME

Description  Display 'ON/OFF TIME' on the LCD.
Usage       LCD_DISP_SMALL_DM_MSG_ON_OFF_TIME
Assumptions None
Input       None
Output      None
Destroys   BA, HL
Example    No Example

LCD_DISP_SMALL_DM_MSG_PASSWORD_NEEDED

Description  Display 'PASSWORD NEEDED' on the LCD.
Usage       LCD_DISP_SMALL_DM_MSG_PASSWORD_NEEDED
Assumptions None
Input       None
Output      None
Destroys   BA, HL
Example    No Example
**LCD_DISP_SMALL_DM_MSG_SORTING_TRY_LATER**

**Description**
Display 'SORTING TRY LATER' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_SORTING_TRY_LATER

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SMALL_DM_MSG_ENTER_PASSWORD**

**Description**
Display 'ENTER PASSWORD' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_ENTER_PASSWORD

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SMALL_DM_MSG_PASSWORD_INVALID**

**Description**
Display 'PASSWORD INVALID' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_PASSWORD_INVALID

**Assumptions**
None

**Input**
None
**Output**  None

**Destroys**  BA, HL

**Example**  No Example

---

**LCD_DISP_SMALL_DM_MSG_PASSWORD**

**Description**  Display 'PASSWORD' on the LCD.

**Usage**  LCD_DISP_SMALL_DM_MSG_PASSWORD

**Assumptions**  None

**Input**  None

**Output**  None

**Destroys**  BA, HL

**Example**  No Example

---

**LCD_DISP_SMALL_DM_MSG_NO_ENTRY_SELECTED**

**Description**  Display 'NO ENTRY SELECTED' on the LCD.

**Usage**  LCD_DISP_SMALL_DM_MSG_NO_ENTRY_SELECTED

**Assumptions**  None

**Input**  None

**Output**  None

**Destroys**  BA, HL

**Example**  No Example

---

**LCD_DISP_SMALL_DM_MSG_SNOOZE**

**Description**  Display 'SNOOZE' on the LCD.
Usage    LCD_DISP_SMALL_DM_MSG_SNOOZE
Assumptions    None
Input    None
Output    None
Destroys    BA, HL
Example    No Example

LCD_DISP_SMALL_DM_MSG_END_OF_LIST
Description    Display 'END OF LIST' on the LCD.
Usage    LCD_DISP_SMALL_DM_MSG_END_OF_LIST
Assumptions    None
Input    None
Output    None
Destroys    BA, HL
Example

LCD_DISP_SMALL_DM_MSG_EDIT
Description    Display 'EDIT' on the LCD.
Usage    LCD_DISP_SMALL_DM_MSG_EDIT
Assumptions    None
Input    None
Output    None
Destroys    BA, HL
Example    No Example
LCD_DISP_SMALL_DM_MSG_ON_TIME

Description
Display 'ON TIME' on the LCD.

Usage
LCD_DISP_SMALL_DM_MSG_ON_TIME

Assumptions
None

Input
None

Output
None

Destroys
BA, HL

Example
No Example

---

LCD_DISP_SMALL_DM_MSG_OFF_TIME

Description
Display 'OFF TIME' on the LCD.

Usage
LCD_DISP_SMALL_DM_MSG_OFF_TIME

Assumptions
None

Input
None

Output
None

Destroys
BA, HL

Example
No Example

---

LCD_DISP_SMALL_DM_MSG_COMM_READY

Description
Display 'COMM READY' on the LCD.

Usage
LCD_DISP_SMALL_DM_MSG_COMM_READY

Assumptions
None

Input
None
### LCD_DISP_SMALL_DM_MSG_COMM_ERROR

**Description**
Display 'COMM ERROR' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_COMM_ERROR

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

### LCD_DISP_SMALL_DM_MSG_PUSH_CROWN_IN

**Description**
Display 'PUSH CROWN IN' on the LCD.

**Usage**
LCD_DISP_SMALL_DM_MSG_PUSH_CROWN_IN

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

### LCD_DISP_SEG_MSG_12HR

**Description**
Display '12-HR' on the 9-segment area.
## Usage

### LCD_DISP_SEG_MSG_12HR

- **Assumptions**: None
- **Input**: None
- **Output**: None
- **Destroys**: BA, HL
- **Example**: No Example

---

## Description

Display '24-HR' on the 9-segment area.

### LCD_DISP_SEG_MSG_24HR

- **Usage**: LCD_DISP_SEG_MSG_24HR
- **Assumptions**: None
- **Input**: None
- **Output**: None
- **Destroys**: BA, HL
- **Example**: No Example

---

## Description

Display 'FREE' on the 9-segment area.

### LCD_DISP_SEG_MSG_FREE

- **Usage**: LCD_DISP_SEG_MSG_FREE
- **Assumptions**: None
- **Input**: None
- **Output**: None
- **Destroys**: BA, HL
- **Example**: No Example
**LCD_DISP_SEG_MSG_CHRONO**

**Description**
Display 'CHRONO' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_CHRONO

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SEG_MSG_INT**

**Description**
Display 'INT' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_INT

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SEG_MSG_HR_MIN**

**Description**
Display 'HR-MIN' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_HR_MIN

**Assumptions**
None

**Input**
None
### LCD_DISP_SEG_MSG_SECOND

**Description**: Display 'SECOND' on the 9-segment area.

**Usage**: LCD_DISP_SEG_MSG_SECOND

**Assumptions**: None

**Input**: None

**Output**: None

**Destroys**: BA, HL

**Example**: No Example

### LCD_DISP_SEG_MSG_TOTAL

**Description**: Display 'TOTAL' on the 9-segment area.

**Usage**: LCD_DISP_SEG_MSG_TOTAL

**Assumptions**: None

**Input**: None

**Output**: None

**Destroys**: BA, HL

**Example**: No Example

### LCD_DISP_SEG_MSG_DAILY

**Description**: Display 'DAILY' on the 9-segment area.
Usage    LCD_DISP_SEG_MSG_DAILY
Assumptions None
Input None
Output None
Destroys BA, HL
Example No Example

LCD_DISP_SEG_MSG_WKDAYS
Description    Display 'WKDAYS' on the 9-segment area.
Usage    LCD_DISP_SEG_MSG_WKDAYS
Assumptions None
Input None
Output None
Destroys BA, HL
Example No Example

LCD_DISP_SEG_MSG_WKENDS
Description    Display 'WKENDS' on the 9-segment area.
Usage    LCD_DISP_SEG_MSG_WKENDS
Assumptions None
Input None
Output None
Destroys BA, HL
Example No Example
**LCD_DISP_SEG_MSG_WEEKLY**

Description: Display 'WEEKLY' on the 9-segment area.

Usage: LCD_DISP_SEG_MSG_WEEKLY

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_SEG_MSG_MNTHLY**

Description: Display 'MNTHLY' on the 9-segment area.

Usage: LCD_DISP_SEG_MSG_MNTHLY

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_SEG_MSG_YEARLY**

Description: Display 'YEARLY' on the 9-segment area.

Usage: LCD_DISP_SEG_MSG_YEARLY

Assumptions: None

Input: None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP_SEG_MSG_ALARM**

Description: Display 'ALARM' on the 9-segment area.
Usage: LCD_DISP_SEG_MSG_ALARM
Assumptions: None
Input: None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP_SEG_MSG_1_DAY**

Description: Display '1-DAY' on the 9-segment area.
Usage: LCD_DISP_SEG_MSG_1_DAY
Assumptions: None
Input: None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP_SEG_MSG_MMDDYY**

Description: Display 'MM-DD-YY' on the 9-segment area.
Usage     LCD_DISP_SEG_MSG_MMDDYY
Assumptions   None
Input       None
Output      None
Destroys   BA, HL
Example    No Example

LCD_DISP_SEG_MSG_DDMMYY
Description      Display 'DD.MM.YY' on the 9-segment area.
Usage            LCD_DISP_SEG_MSG_DDMMYY
Assumptions      None
Input            None
Output           None
Destroys         BA, HL
Example          No Example

LCD_DISP_SEG_MSG_YYMMDD
Description      Display 'YY-MM-DD' on the 9-segment area.
Usage            LCD_DISP_SEG_MSG_YYMMDD
Assumptions      None
Input            None
Output           None
Destroys         BA, HL
Example          No Example
**LCD_DISP_SEG_MSG_BATT**

**Description**
Display 'BATT' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_BATT

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SEG_MSG_HOLD**

**Description**
Display 'HOLD' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_HOLD

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SEG_MSG_STORE**

**Description**
Display 'STORE' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_STORE

**Assumptions**
None

**Input**
None
Output: None
Destroy: BA, HL
Example: No Example

**LCD_DISP_SEG_MSG_TIME**

Description: Display 'TIME' on the 9-segment area.
Usage: LCD_DISP_SEG_MSG_TIME
Assumptions: None
Input: None
Output: None
Destroy: BA, HL
Example: No Example

**LCD_DISP_SEG_MSG_1_MIN**

Description: Display '1 MIN' on the 9-segment area.
Usage: LCD_DISP_SEG_MSG_1_MIN
Assumptions: None
Input: None
Output: None
Destroy: BA, HL
Example: No Example

**LCD_DISP_SEG_MSG_ALERT**

Description: Display 'ALERT' on the 9-segment area.
Usage        LCD_DISP_SEG_MSG_ALERT
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example

**LCD_DISP_SEG_MSG_SHOW**

Description  Display 'SHOW' on the 9-segment area.
Usage        LCD_DISP_SEG_MSG_SHOW
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example

**LCD_DISP_SEG_MSG_COUNT**

Description  Display 'COUNT' on the 9-segment area.
Usage        LCD_DISP_SEG_MSG_COUNT
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example
**LCD_DISP_SEG_MSG_SET**

**Description**
Display 'SET' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_SET

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_SEG_MSG_STOP**

**Description**
Display 'STOP' on the 9-segment area.

**Usage**
LCD_DISP_SEG_MSG_STOP

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_MAIN_DM_LINE1_SELECT**

**Description**
Display 'SELECT' on line 1 of the main DM area.

**Usage**
LCD_DISP_MAIN_DM_LINE1_SELECT

**Assumptions**
None

**Input**
None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP_MAIN_DM_LINE1_ALARM**

Description: Display 'ALARM' on line 1 of the main DM area.
Usage: LCD_DISP_MAIN_DM_LINE1_ALARM
Assumptions: None
Input: None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP_MAIN_DM_LINE2_ALARM**

Description: Display 'ALARM' on line 2 of the main DM area.
Usage: LCD_DISP_MAIN_DM_LINE2_ALARM
Assumptions: None
Input: None
Output: None
Destroys: BA, HL
Example: No Example

**LCD_DISP_MAIN_DM_LINE1_ALARM_AT**

Description: Display 'ALARM AT' on line 1 of the main DM area.
### Usage

**Usage**

LCD_DISP_MAIN_DM_LINE1_ALARM_AT

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

BA, HL

**Example**

No Example

---

### LCD_DISP_MAIN_DM_LINE2_TZ

**Description**

Display 'TZ' on line 2 of the main DM area.

**Usage**

LCD_DISP_MAIN_DM_LINE2_TZ

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

BA, HL

**Example**

No Example

---

### LCD_DISP_MAIN_DM_LINE2_LAP

**Description**

Display 'LAP' on line 2 of the main DM area.

**Usage**

LCD_DISP_MAIN_DM_LINE2_LAP

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

BA, HL

**Example**

No Example
**LCD_DISP_MAIN_DM_LINE2_LAPS**

**Description**  
Display 'LAPS' on line 2 of the main DM area.

**Usage**  
LCD_DISP_MAIN_DM_LINE2_LAPS

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA, HL

**Example**  
No Example

---

**LCD_DISP_MAIN_DM_LINE1_MEMORY**

**Description**  
Display 'MEMORY' in line 1 of the main DM area.

**Usage**  
LCD_DISP_MAIN_DM_LINE1_MEMORY

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA, HL

**Example**  
No Example

---

**LCD_DISP_MAIN_DM_LINE2_BEST_LAP**

**Description**  
Display 'BEST LAP' in line 2 of the main DM area.

**Usage**  
LCD_DISP_MAIN_DM_LINE2_BEST_LAP

**Assumptions**  
None

**Input**  
None
**LCD_DISP_MAIN_DM_LINE2_LAP_AVG**

**Description**
Display 'LAP AVG' in line 2 of the main DM area.

**Usage**
LCD_DISP_MAIN_DM_LINE2_LAP_AVG

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_MAIN_DM_LINE1_APPT_AT**

**Description**
Display 'APPT AT' in line 1 of the main DM area.

**Usage**
LCD_DISP_MAIN_DM_LINE1_APPT_AT

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
BA, HL

**Example**
No Example

---

**LCD_DISP_MAIN_DM_LINE1_HOURS**

**Description**
Display 'HOURS' in line 1 of the main DM area.
**Usage**

`LCD_DISP_MAIN_DM_LINE1_HOURS`

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

BA, HL

**Example**

No Example

---

**LCD_DISP_MAIN_DM_LINE1_MINS**

**Description**

Display 'MINS' in line 1 of the main DM area.

**Usage**

`LCD_DISP_MAIN_DM_LINE1_MINS`

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

BA, HL

**Example**

No Example

---

**LCD_DISP_MAIN_DM_LINE2_PRIOR**

**Description**

Display 'PRIOR' in line 2 of the main DM area.

**Usage**

`LCD_DISP_MAIN_DM_LINE2_PRIOR`

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

BA, HL

**Example**

No Example
**LCD_DISP_MAIN_DM_LINE2_MINS**

**Description**  
Display 'MINS' in line 2 of the main DM area.

**Usage**  
LCD_DISP_MAIN_DM_LINE2_MINS

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA, HL

**Example**  
No Example

---

**LCD_DISP_MAIN_DM_LINE1_BDAY**

**Description**  
Display 'BDAY' in line 1 of the main DM area.

**Usage**  
LCD_DISP_MAIN_DM_LINE1_BDAY

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
BA, HL

**Example**  
No Example

---

**LCD_DISP_MAIN_DM_LINE1_ANNV**

**Description**  
Display 'ANNV' in line 1 of the main DM area.

**Usage**  
LCD_DISP_MAIN_DM_LINE1_ANNV

**Assumptions**  
None

**Input**  
None
### LCD_DISP_MAIN_DM_LINE1_HOLIDAY

**Description**: Display 'HOLIDAY' in line 1 of the main DM area.

**Usage**: LCD_DISP_MAIN_DM_LINE1_HOLIDAY

**Assumptions**: None

**Input**: None

**Output**: None

**Destroys**: BA, HL

**Example**: No Example

---

### LCD_DISP_MAIN_DM_LINE1_VACATION

**Description**: Display 'VACATION' in line 1 of the main DM area.

**Usage**: LCD_DISP_MAIN_DM_LINE1_VACATION

**Assumptions**: None

**Input**: None

**Output**: None

**Destroys**: BA, HL

**Example**: No Example

---

### LCD_DISP_MAIN_DM_LINE1_4_DASHES

**Description**: Display '----' in line 1 of the main DM area.
Usage        LCD_DISP_MAIN_DM_LINE1_4_DASHES
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example

LCD_DISP_MAIN_DM_LINE1_CHIME
Description  Display 'CHIME' in line 1 of the main DM area.
Usage        LCD_DISP_MAIN_DM_LINE1_CHIME
Assumptions  None
Input        None
Output       None
 Destroys     BA, HL
Example      No Example

LCD_DISP_MAIN_DM_LINE2_ON
Description  Display 'ON' in line 2 of the main DM area.
Usage        LCD_DISP_MAIN_DM_LINE2_ON
Assumptions  None
Input        None
Output       None
Destroys     BA, HL
Example      No Example
**LCD_DISP_MAIN_DM_LINE2_OFF**

Description: Display 'OFF' in line 2 of the main DM area.

Usage: LCD_DISP_MAIN_DM_LINE2_OFF

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_MAIN_DM_LINE2_AUTO**

Description: Display 'AUTO' in line 2 of the main DM area.

Usage: LCD_DISP_MAIN_DM_LINE2_AUTO

Assumptions: None

Input: None

Output: None

Destroys: BA, HL

Example: No Example

---

**LCD_DISP_MAIN_DM_LINE1_YEAR**

Description: Display 'YEAR' in line 1 of the main DM area.

Usage: LCD_DISP_MAIN_DM_LINE1_YEAR

Assumptions: None

Input: None
**Output** None
**Destroys** BA, HL
**Example** No Example

---

**LCD_DISP_MAIN_DM_LINE1_AM**

**Description** Display 'AM' in line 1 of the main DM area.
**Usage** LCD_DISP_MAIN_DM_LINE1_AM
**Assumptions** None
**Input** None
**Output** None
**Destroys** BA, HL
**Example** No Example

---

**LCD_DISP_MAIN_DM_LINE1_PM**

**Description** Display 'PM' in line 1 of the main DM area.
**Usage** LCD_DISP_MAIN_DM_LINE1_PM
**Assumptions** None
**Input** None
**Output** None
**Destroys** BA, HL
**Example** No Example

---

**LCD_DISP_MAIN_DM_LINE2_AM**

**Description** Display 'AM' in line 2 of the main DM area.
Usage  
LCD_DISP_MAIN_DM_LINE2_AM

Assumptions  
None

Input  
None

Output  
None

Destroys  
BA, HL

Example  
No Example

---

**LCD_DISP_MAIN_DM_LINE2_PM**

Description  
Display 'PM' in line 2 of the main DM area.

Usage  
LCD_DISP_MAIN_DM_LINE2_PM

Assumptions  
None

Input  
None

Output  
None

Destroys  
BA, HL

Example  
No Example

---

**Utilities**

**UTL_DISPLAY_DAY_OF_WEEK**

Description  
Displays the Day Of Week on the upper dot matrix region.

Usage  
UTL_DISPLAY_DAY_OF_WEEK

Assumptions  
None

Input  
A - Day of week to display

<table>
<thead>
<tr>
<th>Index</th>
<th>Display Characters</th>
<th>Day of Week</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SU</td>
<td>Sunday</td>
</tr>
<tr>
<td>1</td>
<td>MO</td>
<td>Monday</td>
</tr>
<tr>
<td>2</td>
<td>TU</td>
<td>Tuesday</td>
</tr>
</tbody>
</table>

Timex Corporation 185
3 WE Wednesday
4 TH Thursday
5 FR Friday
6 SA Saturday

Output None
Destroys None

Example
; Display SATURDAY (SA) in upper dot matrix
ld A, #6
UTL_DISPLAY_DAY_OF_WEEK

---

**UTL_DISPLAY_DATE_COMPLETE_AND_DOW**

**Description**
Displays the complete Date of an application based from the primary timezone configuration in the 6-digit segmented region, and displays the DOW in the upper dot matrix.

**Usage**
UTL_DISPLAY_DATE_COMPLETE_AND_DOW

**Assumptions**
Date Data Structure should be in the format: Day, Month, Year Lo, Year High, DOW. All data is in BCD 2 digit per byte format. This means that:
- “Day” is expressed as 0x01 to 0x31,
- Month is 0x01 to 0x12,
- etc.

**Input**
IY - IYReg should point to the applications Day field.

**Output**
None

**Destroys**
BA, IX, HL

**Example**
; Display date and day of week information
ld IY, #TODTimeZone1Day
ld IY, #TODTimeZone1Day
UTL_DISPLAY_DATE_COMPLETE_AND_DOW

---

**UTL_DISPLAY_DATE_COMPLETE**

**Description**
Displays the complete Date of an application based from the primary timezone configuration in the 6-digit segmented display

**Usage**
UTL_DISPLAY_DATE_COMPLETE

**Assumptions**
Date Data Structure should be in the format: Day, Month, Year Lo, Year High, DOW. All data is in BCD 2 digit per byte format. This means that:
- “Day” is expressed as 0x01 to 0x31,
Month is 0x01 to 0x12, etc.

**Input**
- IY - IYReg should point to the applications Day field.

**Output**
- None

**Destroys**
- BA, IX, HL

**Example**
```
; Display date information in 6 digit 9 segment display.
ld   IY, #TODTimeZone1Day
UTL_DISPLAY_DATE_COMPLETE
```

---

### UTL_DISPLAY_HR_MIN_DATA_L1

**Description**
Displays a fixed-width hour and minute data on line 1.
Displays the colon between the hour and minute.
Displays either in 12 or 24 hour format. This will depend on the PTZ format.
Displays 'AM' or 'PM'. This will depend on the PTZ format.
Displays a fixed-width hour and minute data on line 1.

**Usage**
- UTL_DISPLAY_HR_MIN_DATA_L1

**Assumptions**
Data structure: Minute, Hour, where each number is stored as a single byte two digit BCD value.
Minutes is expressed as 0x00 to 0x59,
and Hours is expressed as 0x00 to 0x23

**Input**
- HL - address of the minute data.

**Output**
- None

**Destroys**
- IY

**Example**
```
; Display time information in Line 1 of the dot matrix
ld   HL, #TODTimeZone1Minutes
UTL_DISPLAY_HR_MIN_DATA_L1
```

---

### UTL_DISPLAY_HR_MIN_DATA_L2

**Description**
Displays a fixed-width hour and minute data on line 2.
Displays the colon between the hour and minute.
Displays either in 12 or 24 hour format. This will depend on the PTZ format.
Displays 'AM' or 'PM'. This will depend on the PTZ format.
Displays a fixed-width hour and minute data on line 2

**Usage**
- UTL_DISPLAY_HR_MIN_DATA_L2
### Assumptions
Data structure: Minute, Hour, where each number is stored as a single byte two digit BCD value. Minutes is expressed as 0x00 to 0x59, and Hours is expressed as 0x00 to 0x23

### Input
HL - address of the minute data.

### Output
None

### Destroys
IY

### Example
; Display time information in Line 2 of the dot matrix
ld HL, #TODTimeZone1Minutes
UTLgetDisplayHR_MIN_DATA_L2
**TIME ZONE RESOURCE API**

The available resource index for the TOD resource are: 0x00, 0x01, 0x02, 0x03. Resource index 0x00 to 0x02 is reserved for the TOD application. Resource index 0x03 is reserved for the M851 OS.

This is the required time zone structure that is assumed for most Time Zone Resource APIs.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KTODAPPINDEXOFFSET</td>
</tr>
<tr>
<td>1</td>
<td>KTODRESFLAGOFFSET</td>
</tr>
<tr>
<td>2</td>
<td>KTODUPDFLAGOFFSET</td>
</tr>
<tr>
<td>3</td>
<td>KTODSECONDOFFSET</td>
</tr>
<tr>
<td>4</td>
<td>KTODMINUTEOFFSET</td>
</tr>
<tr>
<td>5</td>
<td>KTODHOUROFFSET</td>
</tr>
<tr>
<td>6</td>
<td>KTOODATEOFFSET</td>
</tr>
<tr>
<td>7</td>
<td>KTODMONTHOFFSET</td>
</tr>
<tr>
<td>8</td>
<td>KTODYEARLOOFFSET</td>
</tr>
<tr>
<td>9</td>
<td>KTODYEARHIOFFSET</td>
</tr>
<tr>
<td>10</td>
<td>KTODDOWOFFSET</td>
</tr>
<tr>
<td>11</td>
<td>KTODWEEKOFFSET</td>
</tr>
<tr>
<td>12</td>
<td>KTODPREVIOUSSAMPLEOFFSET</td>
</tr>
</tbody>
</table>

**KTODRESFLAGOFFSET** Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKReserved</td>
</tr>
<tr>
<td>1</td>
<td>bKActive</td>
</tr>
<tr>
<td>2</td>
<td>bKDispUpdRequestSec</td>
</tr>
<tr>
<td>3</td>
<td>bKDispUpdRequestMin</td>
</tr>
<tr>
<td>4</td>
<td>bKTODPrimaryTZ</td>
</tr>
<tr>
<td>5</td>
<td>bKTODEuroFormat</td>
</tr>
<tr>
<td>6</td>
<td>bKGeneratePopup</td>
</tr>
<tr>
<td>7</td>
<td>bKGenerateEvent</td>
</tr>
</tbody>
</table>

**KTODUPDFLAGOFFSET** Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKTODSecondUpd</td>
</tr>
<tr>
<td>1</td>
<td>bKTODMinuteUpd</td>
</tr>
<tr>
<td>2</td>
<td>bKTODHourUpd</td>
</tr>
<tr>
<td>3</td>
<td>bKTODDateUpd</td>
</tr>
<tr>
<td>4</td>
<td>bKTODMonthUpd</td>
</tr>
<tr>
<td>5</td>
<td>bKTODYearUpd</td>
</tr>
<tr>
<td>6</td>
<td>bKTODWeekUpd</td>
</tr>
</tbody>
</table>
Flag Manipulation

**KTOD_ACTIVATE_RESOURCE**

Description: Enables a resource to be updated every second.

Usage: KTOD_ACTIVATE_RESOURCE

Assumptions: TOD resource must previously be owned by an application.

Input: A - TOD Resource Index

Output: None

Destroys: None

Example:

```
; enable the first TOD resource
ld A, #0
KTOD_ACTIVATE_RESOURCE
```

**KTOD_DEACTIVATE_RESOURCE**

Description: Disables updates to a specified resource.

Usage: KTOD_DEACTIVATE_RESOURCE

Assumptions: TOD resource must previously be owned by an application.

Input: A - TOD Resource Index

Output: None

Destroys: None

Example:

```
; prevent any updates to the TOD resource used as the primary time zone
ld A, [COREPTZIndex]
KTOD_DEACTIVATE_RESOURCE
```

**KTOD_ENABLE_DISP_UPD_SEC_EVENT**

Description: Send the event COREEVENT_DISPLAY_UPDATE_TODRES to the foreground application whenever seconds are updated in the specified resource.
### KTOD_ENABLE_DISP_UPD_SEC_EVENT

#### Description
Send the event COREEVENT_DISPLAY_UPDATE_TODRES to the foreground application whenever minutes are updated in the specified resource.

#### Usage
KTOD_ENABLE_DISP_UPD_SEC_EVENT

#### Assumptions
TOD resource must previously be owned by an application.

#### Input
A - TOD Resource Index

#### Output
None

#### Example
;enable seconds updates from the primary time zone resource
ld  A, [COREPTZIndex]
KTOD_ENABLE_DISP_UPD_SEC_EVENT

### KTOD_DISABLE_DISP_UPD_SEC_EVENT

#### Description
Disable one second display updates to the foreground application.

#### Usage
KTOD_DISABLE_DISP_UPD_SEC_EVENT

#### Assumptions
TOD resource must previously be owned by an application.

#### Input
A - TOD Resource Index

#### Output
None

#### Example
;disable seconds updates from the primary time zone resource
ld  A, [COREPTZIndex]
KTOD_DISABLE_DISP_UPD_SEC_EVENT
**KTOD_DISABLE_DISP_UPD_MIN_EVENT**

**Description**: Disable one minute display updates to the foreground application.

**Usage**: KTOD_DISABLE_DISP_UPD_MIN_EVENT

**Assumptions**: TOD resource must previously be owned by an application.

**Input**: A - TOD Resource Index

**Output**: None

**Destroys**: None

**Example**

```assembly
; disable minute update from the primary time zone resource
; this will cancel seconds update request
ld A, [COREPTZIndex]
KTOD_DISABLE_DISP_UPD_MIN_EVENT
```

**KTOD_DEACTIVATE_ALL_DISPLAY_UPDATES**

**Description**: Disables all display updates of all active TOD resource from being passed to the application.

**Usage**: KTOD_DEACTIVATE_ALL_DISPLAY_UPDATES

**Assumptions**: TOD resource must previously be owned by an application.

**Input**: None

**Output**: None

**Destroys**: None

**Example**

```assembly
; disable both second and minute updates from the primary time zone resource
ld A, [COREPTZIndex]
KTOD_DEACTIVATE_ALL_DISPLAY_UPDATES
```
**KTOD_MAKE_AS_PRIMARY_TZ**

**Description** Set the specified TOD resource as the primary time zone. The other active TOD resources are set as secondary time zone.

**Usage** KTOD_MAKE_AS_PRIMARY_TZ

**Assumptions** TOD resource must previously be owned by an application.

**Input** A - TOD Resource Index

**Output** None

**Destroys** None

**Example**

```
;make tod resource 2 as the primary time zone
ld A, #2
KTOD_MAKE_AS_PRIMARY_TZ
```

---

**KTOD_SET_FOR_EURO_FORMAT**

**Description** Sets MONDAY as the first day of week used during week number calculation.

**Usage** KTOD_SET_FOR_EURO_FORMAT

**Assumptions** TOD resource must previously be owned by an application.

**Input** A - TOD Resource Index

**Output** None

**Destroys** None

**Example**

```
;set tod resource 1 to euro format
ld A, #1
KTOD_SET_FOR_EURO_FORMAT
```

---

**KTOD_SET_FOR_US_FORMAT**

**Description** Sets SUNDAY as the first day of week used during week number calculation.

**Usage** KTOD_SET_FOR_US_FORMAT

**Assumptions** TOD resource must previously be owned by an application.
**Input**  A  -  TOD Resource Index

**Output**  None

**Destroys**  None

**Example**  
;set tod resource 1 to US format  
ld  A, #1  
KTOD_SET_FOR_US_FORMAT

---

**Data Manipulation**

**KTOD_CALC_DOW**

**Description**  Calculates the day of the week from the given TOD structure.

**Usage**  KTOD_CALC_DOW

**Assumptions**  None

**Input**  IY  -  Base address of the Day data in the structure

**Output**  A  -  Day of the week

The system defines the following equates for Day of Week:

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUNDAY</td>
<td>0</td>
</tr>
<tr>
<td>MONDAY</td>
<td>1</td>
</tr>
<tr>
<td>TUESDAY</td>
<td>2</td>
</tr>
<tr>
<td>WEDNESDAY</td>
<td>3</td>
</tr>
<tr>
<td>THURSDAY</td>
<td>4</td>
</tr>
<tr>
<td>FRIDAY</td>
<td>5</td>
</tr>
<tr>
<td>SATURDAY</td>
<td>6</td>
</tr>
</tbody>
</table>

**Destroys**  B

**Example**

TODDateStructure:

;October 03, 2002  
db  03H  ;date in BCD  
db  10H  ;month in BCD  
db  02H  ;year lo in BCD  
db  20H  ;year hi in BCD  

; compute the DOW from a given tod structure
**KTOD_CALC_WEEK_NUMBER**

**Description**
Calculates the week number.

**Usage**
KTOD_CALC_WEEK_NUMBER

**Assumptions**
None

**Input**
- IY - Base address of the Date data in the structure
- ZeroFlag - 1 = US Format, SUNDAY as first day.
  0 = EURO Format, MONDAY as first day

**Output**
- A - Week Number in BCD

**Destroys**
B

**Example**
TODDateStructure:

```assembly
; October 03, 2002
db 03H ; date in BCD
db 10H ; month in BCD
db 02H ; year lo in BCD
db 20H ; year hi in BCD
```

```assembly
ld A, [COREPTZFormat]  
b A, #bTODEuroFormat   ; sets ZeroFlag to correct value for EURO/US format  
ld IY, #TODDateStructure  
KTOD_CALC_WEEK_NUMBER
```

**KTOD_ADD_DAYS**

**Description**
Adds a number of days to the structure. This updates the month and day data only.

**Usage**
KTOD_ADD_DAYS

**Assumptions**
None

**Input**
- A - The number of days to add in BCD format.
- IY - Base address of DAY in the TOD structure.
**Output**

<table>
<thead>
<tr>
<th>B.bKTODMonthUpd</th>
<th>0 - if month data didn't change</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.bKTODYearUpd</td>
<td>0 - if year data didn't change</td>
</tr>
</tbody>
</table>

**Destroys**

A

**Example**

TODDateStructure:

;October 28, 2002
db 28H ;date in BCD
db 10H ;month in BCD
db 02H ;year lo in BCD
db 20H ;year hi in BCD

NumberOfDaysToAdd:

db 07H ;7 days

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; compute the week number from a given tod structure
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ld A, [NumberOfDaysToAdd]
ld IY, #TODDateStructure
KTOD_ADD_DAYS

;results to TODDataStructure to contain: November 5, 2002
;results to B.kTODMonthUpd = 1 (month update detected)
;results to B.kTODYearUpd = 0 (no year update required)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;check if we need to increase the year data

---

**KTOD_SUBTRACT_DAYS**

**Description**

Subtracts a number of days from the structure. This will only update the month and day data in the data structure.

**Usage**

KTOD_SUBTRACT_DAYS

**Assumptions**

None

**Input**

| A         | - The number of days to subtract in BCD format. |
| IY        | - Base address of DAY in the TOD structure. |

**Output**

<table>
<thead>
<tr>
<th>B.bKTODMonthUpd</th>
<th>0 - if month data didn't change</th>
</tr>
</thead>
<tbody>
<tr>
<td>B.bKTODYearUpd</td>
<td>0 - if year data didn't change</td>
</tr>
</tbody>
</table>

1 – month/data was updated

1 – year data was updated
Destroys A

Example TODDateStructure:

;October 03, 2002
db 28H ;date in BCD
db 10H ;month in BCD
db 02H ;year lo in BCD
db 20H ;year hi in BCD
db 04H ;DOW (Thursday)

; subtract 7 days from the specified TOD date structure
ld IY, #TODDateStructure
ld A, #7
KTOD_SUBTRACT_DAYS

;results to TODDataStructure to contain: September 27, 2002
;results to B.KTODMonthUpd = 1 (month update detected)
;results to B.KTODYearUpd = 0 (no year update required)

; If it is required to completely adjust the year and DOW information to
; compensate for year rollovers (DEC TO JAN) and leap year,
; then following code sections are required.

; adjust the month and year to compensate for leap year and year rollover
; the API below requires the B register to contain the month and year update
; flags set correctly. In this example, the B register is setup by the
; KTOD_SUBTRACT_DAYS API
ld IY, #TODDateStructure
KTOD_ADJUST_DATE_AND_YEAR

; compute the new DOW for the specified TOD structure and store it back
; into the date structure
ld IY, #TODDateStructure
KTOD_CALC_DOW
ld [TODDateStructure+4], A

---

**KTOD_ADD_YEARS**

Description Adds a number of years to the structure.

Usage KTOD_ADD_YEARS

Assumptions None

Input A - The number of year to add in BCD format.
IY - Base address of YearLo data in TOD structure

Output None

Destroys None
Example

TODDateStructure:

;October 28, 2002
db 28H ;date in BCD
db 10H ;month in BCD
db 02H ;year lo in BCD
db 20H ;year hi in BCD

NumberOfDaysToAdd:

db 07H ;7 days

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

; compute the week number from a given tod structure
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ld A, [NumberOfDaysToAdd]
ld IY, #TODDateStructure
KTOD_ADD_DAYS

;check if we need to increase the year data
bit B, # KTODYearUpd
jr Z, DoNotIncrementTheYear

; add one year since we have detected a december to January rollover
ld A, #1                        ; add 1 year
ld IY, #(TODDateStructure+2)   ; point to the year lo address
KTOD_ADD_YEARS

DoNotIncrementTheYear:

KTOD_SUBTRACT_YEARS

Description
Subtracts a number of years from the structure

Usage
KTOD_SUBTRACT_YEARS

Assumptions
None

Input
A - The number of year to subtract in BCD format.
IY - Base address of YearLo data in TOD structure

Output
None

Destroys
None

Example
TODDateStructure:

;October 28, 2002
db 28H ;date in BCD
db 10H ;month in BCD
db 02H ;year lo in BCD
db 20H ;year hi in BCD
; subtract one year
ld A, #1
ld IY, #(TODDateStructure+2) ; point to the year lo address
KTOD_SUBTRACT_YEARS

---

**KTOD_CORRECT_MDY_DATA**

**Description** Checks the date, month and year data of the structure if it is valid. If it is invalid, then it is set back to a known correct value which is its maximum possible date for a month.

Adjust the year data to '6999' if it is not within the boundary conditions: 0001 - 6999

Adjust the month data to 'DECEMBER' if it is not within the boundary conditions: JANUARY – DECEMBER.

Adjust the day data based on the month and year information with compensation for leap year.

**Usage** KTOD_CORRECT_MDY_DATA

**Assumptions** None

**Input** IY - Base address of Date data in TOD structure

Date:Month:YearLo:YearHi

**Output** None

**Destroys** None

**Example**

TODDateStructure:

; October 28, 2002
db 28H ; date in BCD
db 10H ; month in BCD
db 02H ; year lo in BCD
db 20H ; year hi in BCD

... ; adjust to correct MDY
ld IY, #TODDateStructure ; point to the date
KTOD_CORRECT_MDY_DATA

---

**KTOD_ADJUST_DATE_AND_YEAR**

**Description** If year is updated, subtract 1 from the year data. If month is updated, subtract 1 from day data if the month is FEBRUARY and not a leap year.

This can only be called if there is a date subtraction.
Usage

**KTOD_ADJUST_DATE_AND_YEAR**

Assumptions

None

Input

IY - Base address of Day data in TOD structure

B - Contains the year and month update flags

Output

None

Destroys

None

Example

```assembly
TODDateStructure:

;October 03, 2002
  db 28H ;date in BCD
  db 10H ;month in BCD
  db 02H ;year lo in BCD
  db 20H ;year hi in BCD

NumberOfDaysToSubtract:
  db 07H ;7 days

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
; compute the week number from a given tod structure
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

ld A, [NumberOfDaysToSubtract]
ld IY, #TODDateStructure
KTOD_SUBTRACT_DAYS

;results to TODDataStructure to contain: September 27, 2002
;results to B.kTODMonthUpd = 1 (month update detected)
;results to B.kTODYearUpd = 0 (no year update required)

; adjust the year and month and date due to the subtract operation
ld IY, # TODDateStructure
KTOD_ADJUST_DATE_AND_YEAR
```

Data Transfer

**KTOD_COPY_RESOURCE_TO_RESOURCE**

Description

Copies the tod resource data from into another tod resource structure. App index, resource flag, update flag are not copied.

Usage

**KTOD_COPY_RESOURCE_TO_RESOURCE**

Assumptions

None
**KTOD_COPY_TIME_MIN_TO_DOW_FROM_RESOURCE**

**Description**
Copies day of week, year, month, date, hour and minute data of a specified resource into a memory buffer.

**Usage**
KTOD_COPY_TIME_MIN_TO_DOW_FROM_RESOURCE

**Assumptions**
The destination buffer is allocated with the number of bytes that will be copied.

**Input**
- A - Resource Index of source structure
- IY - Start address of the destination buffer.

**Output**
None

**Destroys**
None

**Example**
```
MyWorkBuffer:
    db 00   ;minute
    db 00   ;hour
    db 00   ;date
    db 00   ;month
    db 00   ;year lo
    db 00   ;year hi
    db 00   ;dow
    ...

;copy the primary time zone data to work buffer
ld  A, [COREPTZIndex]
ld  IY, #MyWorkBuffer
KTOD_COPY_TIME_MIN_TO_DOW_FROM_RESOURCE
```

**KTOD_COPY_TIME_FROM_RESOURCE**

**Description**
Copies week number, day of week, year, month, date, hour, minute and second data of a specified

Timex Corporation

Rev 1.4
resource into a buffer.

Usage

**KTOD_COPY_TIME_FROM_RESOURCE**

Assumptions
The destination buffer is allocated with the number of bytes that will be copied.

Input
- **A** - Resource Index of source structure
- **IY** - Start address of the destination buffer.

Output
None

Destroys
None

Example

```asm
MyWorkBuffer:
    db 00 ; second
    db 00 ; minute
    db 00 ; hour
    db 00 ; date
    db 00 ; month
    db 00 ; year lo
    db 00 ; year hi
    db 00 ; dow
    db 00 ; week number
```

;copy the primary time zone data to work buffer
ld  A, [COREPTZIndex]
ld  IY, #MyWorkBuffer
KTOD_COPY_TIME_FROM_RESOURCE

---

**KTOD_COPY_MDY_FROM_RESOURCE**

Description
Copies year, month and date data of a specified resource into a buffer.

Usage

**KTOD_COPY_MDY_FROM_RESOURCE**

Assumptions
The destination buffer is allocated with the number of bytes that will be copied.

Input
- **A** - Resource Index of source structure
- **IY** - Start address of the destination buffer.

Output
None

Destroys
None

Example

```asm
MyWorkBuffer:
    db 00 ; date
    db 00 ; month
    db 00 ; year lo
    db 00 ; year hi
```
; copy the primary time zone data to work buffer
ld  A, [COREPTZIndex]
ld  IY, #MyWorkBuffer
KTOD_COPY_MDY_FROM_RESOURCE

### KTOD_COPY_HMS_FROM_RESOURCE

**Description**
Copies hour, minute and second data of a specified resource into a buffer.

**Usage**
KTOD_COPY_HMS_FROM_RESOURCE

**Assumptions**
The destination buffer is allocated with the number of bytes that will be copied.

**Input**
- A - Resource Index of source structure
- IY - Start address of the destination buffer.

**Output**
None

**Destroys**
None

**Example**

```
MyWorkBuffer:
db 00 ; second
    ; copy the primary time zone data to work buffer
    ld  A, [COREPTZIndex]
    ld  IY, #MyWorkBuffer
    KTOD_COPY_HMS_FROM_RESOURCE
```

### KTOD_WRITE_TIME_TO_RESOURCE

**Description**
Copies year, month, date, hour, minute and second data from a buffer into a specified resource.

Copies the data from the KTODPreviousRTCSample and use it as the previous sample.
Corrects the validity of date data stored, calculates the day of the week and week number based on the Euro Format bit. If the Euro Format bit is going to be changed, then changed it first before writing the data to the resource.

**Usage**
KTOD_WRITE_TIME_TORESOURCE

**Assumptions**
None
Input       A                   - Resource Index of destination structure
           IY                   - Start address of the source buffer.

Output     None

Destroys  None

Example    MyWorkBuffer:

; 13:30:00 May 14, 2002
db 00H   ;second
db 30H   ;minute
db 13H   ;hour
db 14H   ;date
db 05H   ;month
db 02H   ;year lo
db 20H   ;year hi

; load new time data to resource
ld  A, [COREPTZIndex]
ld  IY, #MyWorkBuffer
KTOD_WRITE_TIME_TO_RESOURCE
; load new month date year data to resource
ld  A, [COREPTZIndex]
ld  IY, #(MyWorkBuffer+3) ; point to date
KTOD_WRITE_MDY_TO_RESOURCE

KTOD_WRITE_HMS_TO_RESOURCE

Description  Copies hour, minute and second data from a buffer into a specified resource

Usage  KTOD_WRITE_HMS_TORESOURCE

Assumptions  None

Input  
  A  - Resource Index of destination structure
  IY  - Start address of the source buffer.

Output  None

Destroys  None

Example  
  MyWorkBuffer:

  ; 13:30:00 May 14, 2002
  db 00H ; second
  db 30H ; minute
  db 13H ; hour
  db 14H ; date
  db 05H ; month
  db 02H ; year lo
  db 20H ; year hi

  ; load new hour minute second data to resource
  ld  A, [COREPTZIndex]
  ld  IY, #(MyWorkBuffer+0) ; point to second
  KTOD_WRITE_HMS_TO_RESOURCE

Resource Utilities

KTOD_LOAD_RESOURCE_START_ADDRESS

Description  Loads the start address of a specified resource into HL

Usage  KTOD_LOAD_RESOURCE_START_ADDRESS
Assumptions  None

Input  A  - Resource index of resource

Output  HL  - Base address of the resource

Destroys  None

Example  ; get the base address of the primary time zone resource
         ld  A, [COREPTZIndex]
         KTOD_LOAD_RESOURCE_START_ADDRESS

KTOD_GET_UPDATE_FLAGS

Description  Gets the update flags of a specified tod resource.

Usage  KTOD_GET_UPDATE_FLAGS

Assumptions  None

Input  A  - Resource index of resource

Output  A  - Update Flags

Update Flag bit structure is shown below:

<table>
<thead>
<tr>
<th>Bit Position</th>
<th>Flag Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X – don’t care</td>
</tr>
<tr>
<td>1</td>
<td>bKTODMinuteUpd</td>
</tr>
<tr>
<td>2</td>
<td>bKTODHourUpd</td>
</tr>
<tr>
<td>3</td>
<td>bKTODDateUpd</td>
</tr>
<tr>
<td>4</td>
<td>bKTODMonthUpd</td>
</tr>
<tr>
<td>5</td>
<td>bKTODYearUpd</td>
</tr>
<tr>
<td>6</td>
<td>bKTODWeekUpd</td>
</tr>
<tr>
<td>7</td>
<td>X – don’t care</td>
</tr>
</tbody>
</table>

Destroys  None

Example  ; get the current update status flag of the primary time zone resource
         ld  A, [COREPTZIndex]
         KTOD_GET_UPDATE_FLAGS

KTOD_CHECK_IF_LEAP_YEAR
**KTOD_CHECK_IF_LEAP_YEAR**

**Description**
Checks if a specified year is a leap year

**Usage**
KTOD_CHECK_IF_LEAP_YEAR

**Assumptions**
None

**Input**
BA - YearHi:YearLo in BCD.

**Output**
bZeroFlag - TRUE -- leap year
FALSE -- not a leap year

**Destroys**
A

**Example**

```
MyWorkBuffer:
; 13:30:00 May 14, 2002
db 00H ;second
db 30H ;minute
db 13H ;hour
db 14H ;date
db 05H ;month
db 02H ;year lo
db 20H ;year hi

;check if current year is a leap year
ld BA, [MyWorkBuffer+5] ;load the year into BA register
KTOD_CHECK_IF_LEAP_YEAR
jr Z, ItIsALeapYear
jr NZ, ItIsNotALeapYear
```
\begin{verbatim}
    db 02H ;year lo
    db 20H ;year hi

    ;check if current year is a leap year
    ld L, [MyWorkBuffer+4] ;load the month into L register
    KTOD_GET_MAX_DAYS_OF_A_MONTH

    ;B register will contain the max days for the queried month
\end{verbatim}

**KTOD_DIVIDE_BY_2**

*Description*  
Divides a 2-byte BCD structure by two.

*Usage*  
KTOD_DIVIDE_BY_2

*Assumptions*  
None

*Input*  
BA - BCD Structure (Hi:Lo)

*Output*  
BA - Quotient

*Destroys*  
None

*Example*  
;divide BCD number by 2
ld BA,#1250H
KTOD_DIVIDE_BY_2

;Results: BA = 0625H

**KTOD_CONVERT_BCD_TO_HEX**

*Description*  
Converts a 2-byte BCD structure into a two-byte HEX structure.

*Usage*  
KTOD_CONVERT_BCD_TO_HEX

*Assumptions*  
None

*Input*  
BA - BCD Structure (Hi:Lo)

*Output*  
BA - HEX Structure (Hi:Lo)

*Destroys*  
IY, HL

*Example*  
;convert BCD number to hex
ld BA,#1250H
KTOD_CONVERT_BCD_TO_HEX
KTOD_CHECK_IF_DIVISIBLE_BY_4

Description
Checks a BCD number if it is divisible by 4. This is only used to check if year is a leap year since the
watch only support the year 2000 to 2099.

Usage
KTOD_CHECK_IF_DIVISIBLE_BY_4

Assumptions
None

Input
A - BCD Number

Output
ZeroFlag - 1  -- divisible by 4
0  -- not divisible by 4

Destroys
A

Example
MyWorkBuffer:

; 13:30:00 May 14, 2002
db 00H ;second
db 30H ;minute
db 13H ;hour
db 14H ;date
db 05H ;month
db 02H ;year lo
db 20H ;year hi

;check if year lo is divisible by 4
ld  A,[MyWorkBuffer+5]
KTOD_CHECK_IF_DIVISIBLE_BY_4
jr  NZ, ValueNotDivisibleBy4
jr  Z, ValueDivisibleBy4

KTOD_GET_PTZ_ADDRESS

Description
Returns the primary time zone address for the given input offset.

Usage
KTOD_GET_PTZ_ADDRESS

Assumptions
None

Input
B - Offset into the TOD resource

Output
IY - Address in the Primary TOD resource data structure for the
given input offset
Destroys None

Example

;Get the absolute address of the seconds data of primary time zone
ld B, # KTODSECONDOFFSET
KTOD_GET_PTZ_ADDRESS

;store 0 to primary time zone seconds
ld A, #00H
ld [IY], A

---

**KTOD_GET_TZ_ADDRESS**

Description Returns the time zone resource address for the given resource index and input offset.

Usage KTOD_GET_PTZ_ADDRESS

Assumptions None

Input
- A - Resource index
- B - Offset into the TOD resource

Output
- IY - Address in the Primary TOD resource data structure for the given input offset

Destroys None

Example

;Get the absolute address of the seconds data of TOD resource
ld B, # KTODSECONDOFFSET
KTOD_GET_TZ_ADDRESS

;store 0 to tod resource seconds
ld A, #00H
ld [IY], A
**TIME ZONE CHECK RESOURCE API**

The available resource index for the TZC resource are: 0x00, 0x01, 0x02, 0x03, 0x04, 0x05.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KTZCAPINDEXOFFSET</td>
</tr>
<tr>
<td>1</td>
<td>KTZCRESFLAGOFFSET</td>
</tr>
<tr>
<td>2</td>
<td>KTZCUPDFLAGOFFSET</td>
</tr>
<tr>
<td>3</td>
<td>KTZCMINUTEOFFSET</td>
</tr>
<tr>
<td>4</td>
<td>KTZCHOUROFFSET</td>
</tr>
<tr>
<td>5</td>
<td>KTZCDATEOFFSET</td>
</tr>
<tr>
<td>6</td>
<td>KTZCMONTHOFFSET</td>
</tr>
<tr>
<td>7</td>
<td>KTZCYEARLLOFFSET</td>
</tr>
<tr>
<td>8</td>
<td>KTZCYEARHIOFFSET</td>
</tr>
</tbody>
</table>

**KTZCRESFLAGOFFSET** Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKReserved</td>
</tr>
<tr>
<td>1</td>
<td>bKActive</td>
</tr>
<tr>
<td>2</td>
<td>Unused</td>
</tr>
<tr>
<td>3</td>
<td>Unused</td>
</tr>
<tr>
<td>4</td>
<td>Unused</td>
</tr>
<tr>
<td>5</td>
<td>Unused</td>
</tr>
<tr>
<td>6</td>
<td>bKGeneratePopup</td>
</tr>
<tr>
<td>7</td>
<td>bKGenerateEvent</td>
</tr>
</tbody>
</table>

**KTZCUPDFLAGOFFSET** Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unused</td>
</tr>
<tr>
<td>1</td>
<td>Unused</td>
</tr>
<tr>
<td>2</td>
<td>Unused</td>
</tr>
<tr>
<td>3</td>
<td>bKTZCDateUpd</td>
</tr>
<tr>
<td>4</td>
<td>bKTZCMonthUpd</td>
</tr>
<tr>
<td>5</td>
<td>bKTZCYearUpd</td>
</tr>
<tr>
<td>6</td>
<td>Unused</td>
</tr>
<tr>
<td>7</td>
<td>Unused</td>
</tr>
</tbody>
</table>

**Flag Manipulation**

**KTZC_DEACTIVATE_ALL_RESOURCES**

**Description**  
Disables all periodic checking on all the time zone check resource.
**Usage**

KTZC_DEACTIVATE_ALL_RESOURCES

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

None

**Example**

; user just entered tod setting
; do not allow any checking to the time zone check while changing
; the primary time zone
KTZC_DEACTIVATE_ALL_RESOURCES

---

**KTZC_ACTIVATE_RESOURCE**

**Description**

Activates the specified time zone check resource.

**Usage**

KTZC_ACTIVATE_RESOURCE

**Assumptions**

No check is being done if the specified resource is already allocated

**Input**

A - Resource index

**Output**

None

**Destroys**

None

**Example**

; appointment application just found the next occurrence of appointment
; ...
; loaded the occurrence data into resource
; ...
; activate the resource. First get the resource index owned by application
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTZCRESOURCEOFFSET0]
KTZC_ACTIVATE_RESOURCE

---

**KTZC_DISABLE_RESOURCE**

**Description**

Deactivates the specified time zone check resource.

**Usage**

KTZC_DISABLE_RESOURCE

**Assumptions**

No check is being done if the specified resource is already allocated
**KTZC_DEACTIVATE_RESOURCE**

**Description**
Setup resource to deactivate the resource owned by the current application.

**Usage**
KTZC_DEACTIVATE_RESOURCE

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
A - Resource index

**Output**
None

**Destroys**
None

**Example**
;deactivate the resource owned by the current application
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTZCRESOURCEOFFSET0]
KTZC_DEACTIVATE_RESOURCE

---

**KTZC_SETUP_POPUP_GENERATION**

**Description**
Setup resource to generate popup on matching resource with time zone.

**Usage**
KTZC_SETUP_POPUP_GENERATION

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
A - Resource index

**Output**
None

**Destroys**
None

**Example**
;specify that a popup be generated
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTZCRESOURCEOFFSET0]
KTZC_SETUP_POPUP_GENERATION

---

**KTZC_SETUP_EVENT_GENERATION**

**Description**
Setup resource to generate event on matching resource with time zone.

**Usage**
KTZC_SETUP_EVENT_GENERATION

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
A - Resource index

**Output**
None

**Destroys**
None

**Example**
;specify that An event be generated and passed to the foreground application
ld IY, [CORECurrentASDAddress]
KTZC_CANCEL_POPUPEVENT_GENERATION

**Description**
Cancel popup or event generation of specified resource

**Usage**
KTZC_CANCEL_POPUPEVENT_GENERATION

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
- A - Resource index

**Output**
None

**Destroys**
None

**Example**
; do not generate any event or popup
ld  IY, [CORECurrentASDAddress]
ld  A, [IY + APPTZCRESOURCESOFFSET0]
KTZC_CANCEL_POPUPEVENT_GENERATION

Data Manipulation

KTZC_RETURNRESOURCEOWNER

**Description**
Returns the application index that owns the specified resource index

**Usage**
KTZC_RETURNRESOURCEOWNER

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
- A - Resource index

**Output**
- B - Application Index

**Destroys**
None

**Example**
; return the owner of the time zone check resource 0
ld  A, #0
KTZC_RETURNRESOURCEOWNER
**KTZC_SETUP_MDY_CHECK**

**Description**
Specify if the resource should check the month, day and year data stored in the resource.

**Usage**
KTZC_SETUP_MDY_CHECK

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
- A - Resource index
- B - MDY check data pattern

The following MDY bit patterns can be ORed together to form the criteria for checking month, day and year:

<table>
<thead>
<tr>
<th>Bit Position</th>
<th>Flag Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>unused</td>
<td>1 - Check date data</td>
</tr>
<tr>
<td>1</td>
<td>unused</td>
<td>1 - Check month data</td>
</tr>
<tr>
<td>2</td>
<td>unused</td>
<td>1 - Check year data</td>
</tr>
<tr>
<td>3</td>
<td>bKTZCDateUpd</td>
<td>1 - Check date data</td>
</tr>
<tr>
<td>4</td>
<td>bKTZCMonthUpd</td>
<td>1 - Check month data</td>
</tr>
<tr>
<td>5</td>
<td>bKTZCYearUpd</td>
<td>1 - Check year data</td>
</tr>
<tr>
<td>6</td>
<td>unused</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>unused</td>
<td></td>
</tr>
</tbody>
</table>

**Output**
None

**Destroys**
None

**Example**
```
; we want to check only the Date data in the TZC resource for an
; automatic monthly appointment popup
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTTZCRESOURCEOFFSET0]
ld B, #( bKTZCDateUpd )
KTZC_SETUP_MDY_CHECK
```

---

**KTZC_SETUP_HOUR_MINUTE_MDY**

**Description**
Update the hour, minute and MDY data structure with user specified data.

**Usage**
KTZC_SETUP_HOUR_MINUTE_MDY

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
- A - Resource index
- IY - Source Address of Hour, Minute, MDY data structure.

The following is the required TZC structure:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Minute data</td>
</tr>
</tbody>
</table>
Hour data
Date date
Month data
Year Lo data
Year Hi data

Output None
Destroys None

Example

NextOccuringAppointment:

    db 00H ;minute
    db 00H ;hour
    db 00H ;date
    db 00H ;month
    db 00H ;yearlo
    db 00H ;yearhi

    ; setup the time and date to generate the appointment popup
    ld IY, [CORECurrentASDAddress]
    ld A, [IY + APPTTZCRESOURCEOFFSET0]
    ld IY, # NextOccuringAppointment
    KTZC_SETUP_HOUR_MINUTE_MDY

KTZC_SETUP_HOUR_MINUTE

Description Update the hour and minute data structure with user specified data.

Usage KTZC_SETUP_HOUR_MINUTE

Assumptions No check is being done if the specified resource is already allocated

Input A - Resource index
       IY - Source Address of Hour, Minute.

The following is the required TZC structure:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Minute data</td>
</tr>
<tr>
<td>1</td>
<td>Hour data</td>
</tr>
</tbody>
</table>

Output None
Destroys None

Example

NextOccuringAppointment:

    db 00H ;minute
    db 00H ;hour
    db 00H ;date
db 00H ;month
db 00H ;yearlo
db 00H ;yearhi

; setup the hour and minute
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTTZCRESOURCESOFTSET0]
ld IY, #(NextOccuringAppointment+0) ;point to minute
KTZC_SETUP_HOUR_MINUTE

KTZC_SETUP_MONTH

Description        Update the month data structure with user specified data.
Usage               KTZC_SETUP_MONTH
Assumptions        No check is being done if the specified resource is already allocated
Input               A - Resource index
                    IY - Source Address of Month data

The following is the required TZC structure:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Month data</td>
</tr>
</tbody>
</table>

Output        None
Destroys      None

Example       NextOccuringAppointment:

    db 00H ;minute
    db 00H ;hour
    db 00H ;date
    db 00H ;month
    db 00H ;yearlo
    db 00H ;yearhi

; setup the month
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTTZCRESOURCESOFFSET0]
ld IY, #(NextOccuringAppointment+3) ;point to month
KTZC_SETUP_MONTH

KTZC_SETUP_DATE

Description        Update the date data structure with user specified data.
Usage  

KTZC_SETUP_DATE

Assumptions  
No check is being done if the specified resource is already allocated

Input  

A - Resource index
IY - Source Address of Date data

The following is the required TZC structure:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Date data</td>
</tr>
</tbody>
</table>

Output  
None

Destroys  
None

Example  

NextOccuringAppointment:

```asm
    db 00H ;minute
    db 00H ;hour
    db 00H ;date
    db 00H ;month
    db 00H ;yearlo
    db 00H ;yearhi

    ; setup the month
    ld IY, [CORECurrentASDAddress]
    ld A, [IY + APPTTZCRESOURCESOURSEOFSET0]
    ld IY, #(NextOccuringAppointment+2) ;point to date
    KTZC_SETUP_MONTH
```

---

**KTZC_SETUP_YEAR**

Description  
Update the year data structure with user specified data.

Usage  

KTZC_SETUP_YEAR

Assumptions  
No check is being done if the specified resource is already allocated

Input  

A - Resource index
IY - Source Address of Year data structure

The following is the required TZC structure:

<table>
<thead>
<tr>
<th>Offset</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Year Lo data</td>
</tr>
<tr>
<td>1</td>
<td>Year Hi data</td>
</tr>
</tbody>
</table>

Output  
None

Destroys  
None
Example

NextOccuringAppointment:

db 00H ;minute
db 00H ;hour
db 00H ;date
db 00H ;month
db 00H ;yearlo
db 00H ;yearhi

; setup the month
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTZCRESOURCESOURCEOFFSET0]
ld IY, #(NextOccuringAppointment+4) ;point to year lo
KTZC_SETUP_YEAR
**BACKUP RESOURCE API**

The available resource index for the BCK resource are: 0x00, 0x01, 0x02.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KBCKAPPINDEXOFFSET</td>
</tr>
<tr>
<td>1</td>
<td>KBCKRESFLAGOFFSET</td>
</tr>
<tr>
<td>2</td>
<td>KBCKSNOOZEOFFSET</td>
</tr>
</tbody>
</table>

**KBCKRESFLAGOFFSET** Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKReserved</td>
</tr>
<tr>
<td>1</td>
<td>bKActive</td>
</tr>
<tr>
<td>2</td>
<td>Unused</td>
</tr>
<tr>
<td>3</td>
<td>Unused</td>
</tr>
<tr>
<td>4</td>
<td>Unused</td>
</tr>
<tr>
<td>5</td>
<td>Unused</td>
</tr>
<tr>
<td>6</td>
<td>bKGeneratePopup</td>
</tr>
<tr>
<td>7</td>
<td>bKGenerateEvent</td>
</tr>
</tbody>
</table>

**Flag Manipulation**

**KBCK_DEACTIVATE_ALL_RESOURCES**

- **Description**: Enables all periodic checking on all the backup resource.
- **Usage**: KBCK_DEACTIVATE_ALL_RESOURCES
- **Assumptions**: None
- **Input**: None
- **Output**: None
- **Destroys**: None
- **Example**: None

**KBCK_ACTIVATE_RESOURCE**

- **Description**: Allows update every minute to the snooze data in the resource
KBCK_ACTIVATE_RESOURCE

Usage
KBCK_ACTIVATE_RESOURCE

Assumptions
No check is being done if the specified resource is already allocated

Input
A - Resource index

Output
None

Destroys
None

Example
; activate resource
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTBCKRESOURCEOFFSET0]
KBCK_ACTIVATE_RESOURCE

KBCK_DEACTIVATE_RESOURCE

Description
Disallows updates every minute to the snooze data in the resource.

Usage
KBCK_DEACTIVATE_RESOURCE

Assumptions
No check is being done if the specified resource is already allocated

Input
A - Resource index

Output
None

Destroys
None

Example
; deactivate resource
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTBCKRESOURCEOFFSET0]
KBCK_DEACTIVATE_RESOURCE

KBCK_SETUP_POPUP_GENERATION

Description
Setup resource to generate popup when snooze time expires.

Usage
KBCK_SETUP_POPUP_GENERATION

Assumptions
No check is being done if the specified resource is already allocated

Input
A - Resource index

Output
None

Destroys
None
Example:

`; setup for popup
ld  IY, [CORECurrentASDAddress]
ld  A, [IY + APPTBCKRESOURCEOFFSET0]
KBCK_SETUP_POPUP_GENERATION`

---

**KBCK_SETUP_EVENT_GENERATION**

**Description**: Setup resource to generate event when snooze time expires.

**Usage**: KBCK_SETUP_EVENT_GENERATION

**Assumptions**: No check is being done if the specified resource is already allocated

**Input**:

- `A` - Resource index

**Output**: None

**Destroys**: None

**Example**:

`; setup for event to be passed to foreground application
ld  IY, [CORECurrentASDAddress]
ld  A, [IY + APPTBCKRESOURCEOFFSET0]
KBCK_SETUP_EVENT_GENERATION`

---

**KBCK_CANCEL_POPUPEVENT_GENERATION**

**Description**: Cancel popup or event generation of specified resource

**Usage**: KBCK_CANCEL_POPUPEVENT_GENERATION

**Assumptions**: No check is being done if the specified resource is already allocated

**Input**:

- `A` - Resource index

**Output**: None

**Destroys**: None

**Example**:

`; do not generate an event or popup
ld  IY, [CORECurrentASDAddress]
ld  A, [IY + APPTBCKRESOURCEOFFSET0]
KBCK_CANCEL_POPUPEVENT_GENERATION`
Data Manipulation

KBCK_RETURN_RESOURCE_OWNER

Description
Returns the application index that owns the specified resource index.

Usage
KBCK_RETURN_RESOURCE_OWNER

Assumptions
No check is being done if the specified resource is already allocated

Input
A - Resource index

Output
B - Application Index

Destroys
None

Example
; get owner of the backup resource #1
ld A, #1
KBCK_RETURN_RESOURCE_OWNER

KBCK_SETUP_SNOOZE_TIME

Description
Sets up the snooze time of the specified resource index.
Cancels any queued backup check resource of the same index.

Usage
KBCK_SETUP_SNOOZE_TIME

Assumptions
No check is being done if the specified resource is already allocated

Input
A - Resource index
B - Snooze time specified in HEX.

Output
None

Destroys
None

Example
; setup the snooze time for 10 minutes
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTBCKRESOURCEOFFSET0]
ld B, #10
KBCK_SETUP_SNOOZE_TIME
STOPWATCH RESOURCE API

The available resource index for the STP resource are: 0x00, 0x01.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KSTPAPPINDEXOFFSET</td>
</tr>
<tr>
<td>1</td>
<td>KSTPRESFLAGOFFSET</td>
</tr>
<tr>
<td>2</td>
<td>KSTPUPDFLAGOFFSET</td>
</tr>
<tr>
<td>3</td>
<td>KSTPPREVHUNDTHSSAMPLEOFFSET</td>
</tr>
<tr>
<td>4</td>
<td>KSTPPREVSECONDSSAMPLEOFFSET</td>
</tr>
<tr>
<td>5</td>
<td>KSTPHUNDREDTHSOFFSET</td>
</tr>
<tr>
<td>6</td>
<td>KSTPSECONDOFFSET</td>
</tr>
<tr>
<td>7</td>
<td>KSTPMINUTEOFFSET</td>
</tr>
<tr>
<td>8</td>
<td>KSTPHouroffset</td>
</tr>
<tr>
<td>9</td>
<td>KSTPTMRIDOFFSET</td>
</tr>
<tr>
<td>10</td>
<td>KSTPPPOPUPSOURCEAPPIDOFFSET</td>
</tr>
<tr>
<td>11</td>
<td>KSTPSTPIDOFFSET</td>
</tr>
</tbody>
</table>

KSTPRESFLAGOFFSET Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKReserved</td>
</tr>
<tr>
<td>1</td>
<td>bKActive</td>
</tr>
<tr>
<td>2</td>
<td>bKDispUpdRequest</td>
</tr>
<tr>
<td>3</td>
<td>bKTMRLink</td>
</tr>
<tr>
<td>4</td>
<td>bKSTPLink</td>
</tr>
<tr>
<td>5</td>
<td>bKNotReset</td>
</tr>
<tr>
<td>6</td>
<td>bKGeneratePopup</td>
</tr>
<tr>
<td>7</td>
<td>bKGenerateEvent</td>
</tr>
</tbody>
</table>

KSTPUPDFLAGOFFSET Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKSTPSecondUpd</td>
</tr>
<tr>
<td>1</td>
<td>bKSTPMinuteUpd</td>
</tr>
<tr>
<td>2</td>
<td>bKSTPHourUpd</td>
</tr>
<tr>
<td>3</td>
<td>Unused</td>
</tr>
<tr>
<td>4</td>
<td>bKSTPSecondNonZero</td>
</tr>
<tr>
<td>5</td>
<td>bKSTPMinuteNonZero</td>
</tr>
<tr>
<td>6</td>
<td>bKSTPHoursNonZero</td>
</tr>
<tr>
<td>7</td>
<td>bKSTPRunout</td>
</tr>
</tbody>
</table>
Data Manipulation

**KSTP_LINK_NEXTRESOURCE**

**Description**
Indicates that the next resource is a link to this specified resource.

**Usage**
KSTP_LINK_NEXTRESOURCE

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
A - Resource index

**Output**
None

**Destroys**
HL

**Example**
; request that when the current resource is started, it will also
; start the next resource (by index)
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET0]
KSTP_LINK_NEXTRESOURCE

**KSTP_ENABLE_DISP_UPD_EVENT**

**Description**
Allows the foreground application to receive display update events from the specified resource.

**Usage**
KSTP_ENABLE_DISP_UPD_EVENT

**Assumptions**
No check is being done if the specified resource is already allocated

**Input**
A - Resource index

**Output**
None

**Destroys**
HL

**Example**
; request that any updates done to the stopwatch at 16HZ interval that
; a event be passed to the foreground application for display updates
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET0]
KSTP_ENABLE_DISP_UPD_EVENT

**KSTP_DISABLE_DISP_UPD_EVENT**

**Description**
Cancels any requests to send the foreground application to receive display update events from the specified resource.
Usage: KSTP_DISABLE_DISP_UPD_EVENT

Assumptions: No check is being done if the specified resource is already allocated

Input: A - Resource index

Output: None

Destroys: HL

Example:
```
;do not sent any display update events when stopwatch resource is
;updated
ld  IY, [CORECurrentASDAddress]
ld  A, [IY + APPTSTPRESOURCEROFFSET0]
KSTP_DISABLE_DISP_UPD_EVENT
```

---

**KSTP_DEACTIVATE_ALL_DISPLAY_UPDATES**

Description: Cancels all requests of all stopwatch resource to send the foreground application to receive display update events.

Usage: KSTP_DEACTIVATE_ALL_DISPLAY_UPDATES

Assumptions: No check is being done if the specified resource is already allocated

Input: None

Output: None

Destroys: HL, B

Example:
```
;disable all stopwatch resource display update event generation
;to the foreground application
KSTP_DEACTIVATE_ALL_DISPLAY_UPDATES
```

---

**KSTP_SETUP_POPUP_GENERATION**

Description: Setup the specified resource to generate popup on stopwatch runout.

Usage: KSTP_SETUP_POPUP_GENERATION

Assumptions: No check is being done if the specified resource is already allocated

Input: A - Resource index

Output: None
**KSTP_SETUP_EVENT_GENERATION**

- **Description**: Setup resource to generate event on stopwatch runout.
- **Usage**: KSTP_SETUP_EVENT_GENERATION
- **Assumptions**: No check is being done if the specified resource is already allocated
- **Input**: A - Resource index
- **Output**: None
- **Destroys**: HL
- **Example**: ;request an event to be passed to the foreground application when the stopwatch resource has reached 100 hours
  
  ld  IY, [CORECurrentASDAddress]
  ld  A, [IY + APPTSTPRESOURCESOURCEOFFSET0]
  KSTP_SETUP_EVENT_GENERATION

---

**KSTP_CANCEL_POPUPEVENT_GENERATION**

- **Description**: Cancel popup or event generation of specified resource.
- **Usage**: KSTP_CANCEL_POPUPEVENT_GENERATION
- **Assumptions**: No check is being done if the specified resource is already allocated
- **Input**: A - Resource index
- **Output**: None
- **Destroys**: HL
- **Example**: ;no events when the stopwatch has reached 100 hours.

  ld  IY, [CORECurrentASDAddress]
  ld  A, [IY + APPTSTPRESOURCESOURCEOFFSET0]
  KSTP_CANCEL_POPUPEVENT_GENERATION
**KSTP_START_RESOURCE**

Description: Starts a specific stopwatch resource.

Usage: KSTP_START_RESOURCE

Assumptions: KRESStopwatchHundthsDataBuffer and KRESStopwatchSecondsDataBuffer holds the value of the free running counter snapshot or the values where this resource should use as reference to start.

Input: A - Resource index

Output: None

Destroys: None

Example:
```
; start stopped stopwatch resource
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET0]
KSTP_START_RESOURCE
```

**KSTP_STOP_RESOURCE**

Description: Stops a specific stopwatch resource.

Usage: KSTP_STOP_RESOURCE

Assumptions: KRESStopwatchHundthsDataBuffer and KRESStopwatchSecondsDataBuffer holds the value of the free running counter snapshot or the values where this resource should use as reference to stop.

Input: A - Resource index

Output: None

Destroys: None

Example:
```
; stop a running stopwatch resource
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET0]
KSTP_STOP_RESOURCE
```

**KSTP_STOP_ALLResources**

Description: Stops all active stopwatch resource.
### KSTP_STOP_ALL_RESOURCES

**Usage**

KSTP_STOP_ALL_RESOURCES

**Assumptions**

KRESSStopwatchHundthsDataBuffer and KRESSStopwatchSecondsDataBuffer holds the value of the free running counter snapshot or the values where this resource should use as reference to stop.

**Input**

None

**Output**

None

**Destroys**

None

**Example**

; stop all stopwatch resource
KSTP_STOP_ALL_RESOURCES

### KSTP_TAKE_SPLIT

**Description**

Updates a specific stopwatch resource so a split time can be stored.

**Usage**

KSTP_TAKE_SPLIT

**Assumptions**

KRESSStopwatchHundthsDataBuffer and KRESSStopwatchSecondsDataBuffer holds the value of the free running counter snapshot or the values where this resource should use as reference to stop.

**Input**

A - Resource index

**Output**

None

**Destroys**

None

**Example**

; take a split and freeze updates to the display (resource is still running)
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET0]
KSTP_TAKE_SPLIT
KSTP_DISABLE_DISP_UPD_EVENT
; display the stopwatch data for user to see

### KSTP_RESETRESOURCE

**Description**

Resets a specific stopwatch resource to 00:00:00.00.

**Usage**

KSTP_RESET_RESOURCE

**Assumptions**

None

**Input**

A - Resource index
Output: None

Destroys: None

Example:
```
; check if STOP switch was depress
cp A, #COREEVENT_STOPRESETDEPRESS
jr NZ, check_next_event1

; reset the stopwatch
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET0]
KSTP_RESET_RESOURCE
ret

check_next_event1:
```

---

**KSTP_RESET_ALL_RESOURCE**

Description: Resets all stopwatch resource to 00:00:00.00.

Usage: KSTP_RESET_ALL_RESOURCE

Assumptions: None

Input: None

Output: None

Destroys: A

Example:
```
; reset all stopwatch resource
KSTP_RESET_ALL_RESOURCE
```

---

**KSTP_CLEAR_RESOURCE_DATA**

Description: Resets a specific stopwatch resource to 00:00:00.00.

Usage: KSTP_CLEAR_RESOURCE_DATA

Assumptions: None

Input: A - Resource index

Output: None

Destroys: HL
Example

; clear the lap stopwatch resource since we just took a split time
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET1]
KSTP_CLEAR_RESOURCE_DATA

Data Transfer

**KSTP_COPY_RESOURCE_TO_BUFFER**

Description
Copies stopwatch resource data H:M:S.h data from a specified resource index into a buffer.

Usage
KSTP_COPY_RESOURCE_TO_BUFFER

Assumptions
None

Input
- A - Resource index
- IY - Destination address to store hundredths, second, minute and hour in BCD format.

Output
None

Destroys
HL

Example

MyStopwatchBuffer:

db 00H ; hundredths
db 00H ; seconds
db 00H ; minute
db 00H ; hour

...;
copy stopwatch data to buffer
ld IY, [CORECurrentASDAddress]
ld A, [IY + APPTSTPRESOURCEOFFSET0]
ld IY, #MyStopwatchBuffer
KSTP_COPY_RESOURCE_TO_BUFFER

Resource Utilities

**KSTP_COMPARE_BUFFER_FROM_RESOURCE**

Description
Compares a buffer to a resource

Usage
KSTP_COMPARE_BUFFER_FROM_RESOURCE

Assumptions
None
M851 WristApp API Reference Guide  Rev 1.4

Input

A - Resource index
IY - Hours Information of the Buffer using a stopwatch data structure

Output

Set/Clears the ZeroFlag and Carry Flag based on the following conditions:

If Resource == Buffer
   ZeroFlag == TRUE, CarryFlag == FALSE

If Resource < Buffer
   ZeroFlag == FALSE, CarryFlag == TRUE

If Resource > Buffer
   ZeroFlag == FALSE, CarryFlag == FALSE

Destroys

A, B, IY, HL

Example

MyStopwatchBuffer:

db 00H ;app index
db 00H ;resource flag
db 00H ;update flag
db 00H ;previous hundredth sample
db 00H ;previous seconds sample
db 00H ;hundredths
db 00H ;seconds
db 00H ;minute
db 00H ;hour
db 00H ;timer id
db 00H ;popupsource app id
db 00H ;stopwatch id

;compare stopwatch date with buffer data
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTSTPRESOURCEOFFSET0]
ld IY, #[MyStopwatchBuffer+8]; point to hour data
KSTP_COMPARE_BUFFER_FROM_RESOURCE

jr Z, ResourceIsEqualBufferData
jr C, ResourceIsLessThanBufferData
jr NC, ResourceIsgreaterThanBufferData

KSTP_GET_RESOURCE_STATUS

Description

Gets the status of a specified resource index.

Usage

KSTP_GET_RESOURCE_STATUS

Assumptions

None

Input

A - Resource index

Output

AReg holds the status of the resource. The following values are returned in A to indicate the status of
the resource.

<table>
<thead>
<tr>
<th>Value</th>
<th>Resource Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x00</td>
<td>Reset</td>
</tr>
<tr>
<td>0x20</td>
<td>Not reset and stopped.</td>
</tr>
<tr>
<td>0x22</td>
<td>Not reset and running</td>
</tr>
</tbody>
</table>

Destroys HL

Example

```
;get resource status
ld   IX, [CORECurrentASDAddress]
ld   A, [IX + APPTSTPRESOURCEOFFSET0]
KSTP_GET_RESOURCE_STATUS
cp   A, #00H
jr   Z, StopwatchResourceIsReset
cp   A, #20H
jr   Z, StopwatchNotResetAndStopped
cp   A, #22H
jr   Z, StopwatchNotResetAndRunning
```
**TIMER RESOURCE API**

The available resource index for the TMR resource are: 0x00, 0x01, 0x02.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KTMRAPPINDEXOFFSET</td>
</tr>
<tr>
<td>1</td>
<td>KTMRRESFLAGOFFSET</td>
</tr>
<tr>
<td>2</td>
<td>KTMRUPDATEFLAGOFFSET</td>
</tr>
<tr>
<td>3</td>
<td>KTMRPREVHUNDREDTHSSAMPLEOFFSET</td>
</tr>
<tr>
<td>4</td>
<td>KTMRPREVSECONDSSAMPLEOFFSET</td>
</tr>
<tr>
<td>5</td>
<td>KTMRWORKHUNDREDTHSOFFSET</td>
</tr>
<tr>
<td>6</td>
<td>KTMRWORKSECONDOFFSET</td>
</tr>
<tr>
<td>7</td>
<td>KTMRWORKMINUTEOFFSET</td>
</tr>
<tr>
<td>8</td>
<td>KTMRWORKHOUROFFSET</td>
</tr>
<tr>
<td>9</td>
<td>KTMRTMRIDOFFSET</td>
</tr>
<tr>
<td>10</td>
<td>KTMRSTPIDOFFSET</td>
</tr>
<tr>
<td>12</td>
<td>KTMRUSERSECONDOFFSET</td>
</tr>
<tr>
<td>13</td>
<td>KTMRUSERMINUTEOFFSET</td>
</tr>
<tr>
<td>14</td>
<td>KTMRUSERHOUROFFSET</td>
</tr>
<tr>
<td>15</td>
<td>KTMRPREWARNSECONDOFFSET</td>
</tr>
<tr>
<td>16</td>
<td>KTMRPREWARNMINUTEOFFSET</td>
</tr>
<tr>
<td>17</td>
<td>KTMRPREWARNHOUROFFSET</td>
</tr>
<tr>
<td>18</td>
<td>KTMRCOUNTEROFFSET</td>
</tr>
<tr>
<td>19</td>
<td>KTMRNUMOFREPSOFFSET</td>
</tr>
</tbody>
</table>

**KTMRRESFLAGOFFSET** Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKReserved</td>
</tr>
<tr>
<td>1</td>
<td>bKActive</td>
</tr>
<tr>
<td>2</td>
<td>bKDispUpdRequest</td>
</tr>
<tr>
<td>3</td>
<td>bKTMRLink</td>
</tr>
<tr>
<td>4</td>
<td>bKSTPIDOFFSET</td>
</tr>
<tr>
<td>5</td>
<td>bKNotReset</td>
</tr>
<tr>
<td>6</td>
<td>bKGeneratePopup</td>
</tr>
<tr>
<td>7</td>
<td>bKGenerateEvent</td>
</tr>
</tbody>
</table>

**KTMRUPDATEFLAGOFFSET** Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKTMRSecondUpd</td>
</tr>
<tr>
<td>1</td>
<td>bKTMRMinuteUpd</td>
</tr>
<tr>
<td>2</td>
<td>bKTMRHourUpd</td>
</tr>
<tr>
<td>3</td>
<td>bKTMRPreWarnEnTracking</td>
</tr>
<tr>
<td>4</td>
<td>bKTMRPreWarnReqTracking</td>
</tr>
</tbody>
</table>
KTMR_SETUP_USER_AND_WORK_HMS

**Description**
Sets up the user set and working HMS data.

When setup for a count-down operation, user data is the initial data for countdown and will be loaded into the working time structure when reset.

When setup for a count-up operation, the working data usually starts at 00:00.00.00 (using the reset API) and counts up until the value stored in the user time structure. At runout, it will stop the timer and generated the required event for background task processing or popup processing.

The working time variables can be loaded with other data through the SETUP_WORKING_HMS API.

**Usage**

```
KTMR_SETUP_USER_AND_WORK_HMS
```

**Assumptions**
The counting direction has already been set.

**Input**

- **A** - Resource index
- **IY** - Points to the beginning of the Seconds, Minutes and Hour structure from memory buffer

**Output**
None

**Destroys**
None

**Example**

```
NextUserSetSecond:

;user time: 01:30:00
db 00H ;second
db 30H ;minute
db 01H ;hour
...

; setup both user and working time structure in a countdown operation
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
ld IY, #NextUserSetSecond
KTMR_SETUP_USER_AND_WORK_HMS
```

KTMR_SETUP_WORKING_HMS

**Description**
Sets up the working HMS data from the user data.

**Usage**

```
KTMR_SETUP_WORKING_HMS
```
**Assumptions**  
The counting direction has already been set.

**Input**  
A - Resource index

**Output**  
None

**Destroys**  
None

**Example**  
;setup working time structure from user time structure
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_SETUP_WORKING_HMS

---

**KTMR_SETUP_PREWARNING_HMS**

**Description**  
Sets up the pre-warning data. If pre-warning data is equal to the user set data then the pre-warning tracking will not be enabled. If it is successful then pre-warning tracking will be enabled.

**Usage**  
None

**Assumptions**  
The counting direction has already been set.

**Input**  
A - Resource index  
IY - Points to the beginning of the Seconds, Minutes and Hour structure from memory buffer

**Output**  
None

**Destroys**  
None

**Example**  
NextUserSetSecond:

;user time: 01:30:00
db 00H ;second
db 30H ;minute
db 01H ;hour

PreNotificationTime:

;based on half of user time: 01:30:00/2 = 00:45:00
db 00H ;second
db 45H ;minute
db 00H ;hour

;setup prewarning time structure
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
ld IY, # PreNotificationTime
KTMR_SETUP_PREWARNING_HMS
**KTMR_DISABLE_PREWARNING_TRACKING**

**Description**  
Disables the pre-warning processing by the specified resource.

**Usage**  
KTMR_DISABLE_PREWARNING_TRACKING

**Assumptions**  
None

**Input**  
A - Resource index

**Output**  
None

**Destroys**  
None

**Example**  
; disable prewarning tracking  
ld IX, [CORECurrentASDAddress]  
ld A, [IX + APPTMRRESOURCESOFFSET0]  
KTMR_DISABLE_PREWARNING_TRACKING

---

**KTMR_ENABLE_DISP_UPD_EVENT**

**Description**  
Enable display update event to be passed to the foreground application.

**Usage**  
None

**Assumptions**  
The counting direction has already been set.

**Input**  
A - Resource index

**Output**  
None

**Destroys**  
B

**Example**  
; enable display update events to be passed to foreground application  
ld IX, [CORECurrentASDAddress]  
ld A, [IX + APPTMRRESOURCESOFFSET0]  
KTMR_ENABLE_DISP_UPD_EVENT

---

**KTMR_DISABLE_DISP_UPD_EVENT**

**Description**  
Disable display update event to be passed to the foreground application.
**Usage**

KTMR_DISABLE_DISP_UPD_EVENT

**Assumptions**

None

**Input**

A - Resource index

**Output**

None

**Destroys**

B

**Example**

; disable display update events to be passed to foreground application
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESouceOFFSET0]
KTMR_DISABLE_DISP_UPD_EVENT

---

**KTMR_DEACTIVATE_ALL_DISPLAY_UPDATES**

**Description**

Deactivates display update for all timer resources.

**Usage**

KTMR_DEACTIVATE_ALL_DISPLAY_UPDATES

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

B

**Example**

; cancel all display update events on all timer resource
KTMR_DEACTIVATE_ALL_DISPLAY_UPDATES

---

**KTMR_SETUP_COUNTUP**

**Description**

Set the timer resource as a count up resource.

**Usage**

KTMR_SETUP_COUNTUP

**Assumptions**

None

**Input**

A - Resource index

**Output**

None

**Destroys**

B

**Example**

; setup the timer resource for count UP operation
ld IX, [CORECurrentASDAddress]
KTMR_SETUP_COUNTDOWN

Description  Make the timer resource as a count down resource.

Usage  KTMR_SETUP_COUNTDOWN

Assumptions  None

Input  A - Resource index

Output  None

Destroys  B

Example  ;setup the timer resource for count DOWN operation
ld  IX, [CORECurrentASDAddress]
ld  A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_SETUP_COUNTDOWN

KTMR_SET_RUNNING_REPEAT_COUNTER

Description  Forcibly change the value of the repeat counter.

Usage  KTMR_SET_RUNNING_REPEAT_COUNTER

Assumptions  None

Input  A - Resource index
       B - Starting repetition counter formatted as HEX data.

Output  None

Destroys  None

Example  ;change the current value of the repeat counter
ld  IX, [CORECurrentASDAddress]
ld  A, [IX + APPTMRRESOURCEOFFSET0]
ld  B, #0                                ;set repeat counter to 0
KTMR_SET_RUNNING_REPEAT_COUNTER
### KTMR_GET_RUNNING_REPEAT_COUNTER

**Description**  Get the number of repetitions of a specified resource.

**Usage**  KTMR_GET_RUNNING_REPEAT_COUNTER

**Assumptions**  None

**Input**  
- A - Resource index

**Output**  
- B - Running repeat counter stored as a HEX value.

**Destroys**  HL

**Example**  
; get current running counter
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_GET_RUNNING_REPEAT_COUNTER

---

### KTMR_SETUP_POPUP_GENERATION

**Description**  Set the bit that tells the resource that it has to generate a popup when the resource expires.

**Usage**  KTMR_SETUP_POPUP_GENERATION

**Assumptions**  None

**Input**  
- A - Resource index

**Output**  None

**Destroys**  B

**Example**  
; request a popup when counter has reached 0 (if countdown) or if reach specified max time (if countup)
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_SETUP_POPUP_GENERATION

---

### KTMR_SETUP_EVENT_GENERATION

**Description**  Set the bit that tells the resource that it has to generate an event when the timer expires.

**Usage**  KTMR_SETUP_EVENT_GENERATION
Assumptions: None
Input: A - Resource index
Output: None
Destroys: B

Example:
; request an event to be passed to the foreground application when
; counter has reached 0 (if countdown) or if reach specified max
; time (if countup)
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_SETUP_EVENT_GENERATION

KTMR_CANCEL_POPUP_AND_EVENT_GENERATION

Description: Cancel popup event and event generation.
Usage: KTMR_CANCEL_POPUP_AND_EVENT_GENERATION
Assumptions: None
Input: A - Resource index
Output: None
Destroys: None

Example:
; do not generate a popup or event when timer expires
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_CANCEL_POPUP_AND_EVENT_GENERATION

KTMR_SETUP_REPEAT_AT_END

Description: Enable the repeat at end action and sets up the number of times we need to repeat the timer.
Usage: KTMR_SETUP_REPEAT_AT_END
Assumptions: None
Input: A - Resource index
B - Number of Repetitions formatted as HEX data.

If number of repetitions is 0 then it is a continuous repeat.
Output  None
Destroys  None
Example  ;setup timer to repeat at end
         ld  IX, [CORECurrentASDAddress]
         ld  A, [IX + APPTMRRESOURCEOFFSET0]
         KTMR_SETUP_REPEAT_AT_END

**KTMRCancelar_REPEAT_AT_END**

Description  Disables the repeat at end action.
Usage  KTMRCancelar_REPEAT_AT_END
Assumptions  None
Input  A  -  Resource index
Output  None
Destroys  B
Example  ;cancel timer to repeat at end
         ld  IX, [CORECurrentASDAddress]
         ld  A, [IX + APPTMRRESOURCEOFFSET0]
         KTMRCancelar_REPEAT_AT_END

**KTMR_SETUP_TMR_LINK**

Description  Set a timer link on timer ending.
Usage  KTMR_SETUP_TMR_LINK
Assumptions  None
Input  A  -  Timer Resource index to establish the link from
       B  -  Timer Resource index to link to when the Timer ends
Output  None
Destroys  None
Example  ;setup the timer resource to start when the specified timer expires
         ld  IX, [CORECurrentASDAddress]
         ld  A, [IX + APPTMRRESOURCEOFFSET0]
ld B, [IX + APPTMRRESOURCEOFFSET1]
KTMR_SETUP_TMR_LINK

---

**KTMR_SETUP_STP_LINK**

**Description**
Set the bit that indicates to link a stopwatch when the timer expires. The stopwatch resource to start will be started when the timer expires.

**Usage**
KTMR_SETUP_STP_LINK

**Assumptions**
None

**Input**
A - Timer resource to link Stopwatch Resource to

**Output**
None

**Destroys**
None

**Example**
;setup the timer resource to start a stopwatch resource (that is reset) when the specified timer expires
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_SETUP_TMR_LINK

---

**KTMRCANCEL_ALL_LINKS**

**Description**
Cancels all links to TMR and STP of a specified resource.

**Usage**
KTMR_CANCEL_ALL_LINKS

**Assumptions**
None

**Input**
A - Resource index

**Output**
None

**Destroys**
None

**Example**
;cancel all timer and stopwatch resource links when timer expires
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_CANCEL_ALL_LINKS
**KTMR_RESET_RESOURCE**

**Description**
Sets up the working HMS data depending on the direction of the timer. Clears the running repeat counter. This also stops the total and stoppage time if timer caused a synchro resource to start.

**Usage**
KTMR_RESET_RESOURCE

**Assumptions**
None

**Input**
A - Resource index

**Output**
None

**Destroys**
None

**Example**

```assembly
; reset the timer resource
ld IX, [CORECurrentASDAddress]
ld A, IX + APPTMRRESOURCEOFFSET0
KTMR_RESET_RESOURCE
```

---

**KTMR_START_RESOURCE**

**Description**
Start background update for a timer resource with the stopwatch data buffers already loaded with the value to be used.

A synchro resource will be started from reset if the specified timer resource is started from reset. The application owner of this timer will also be the one to control the synchro resource. The synchro stoppage time will be stopped if this resource caused it to start.

**Usage**
KTMR_START_RESOURCE

**Assumptions**
None

**Input**
A - Resource index

**Output**
None

**Destroys**
None

**Example**

```assembly
; start the timer resource
ld IX, [CORECurrentASDAddress]
ld A, IX + APPTMRRESOURCEOFFSET0
KTMR_START_RESOURCE
```

---

**KTMR_STOP_RESOURCE**
**Description**
Stop background update for a timer resource with the stopwatch data buffers already loaded with the value to be used.

The synchro resource stoppage time will be started if the specified timer resource caused it to start from reset. The synchro running time will be stopped.

**Usage**
KTMR_STOP_RESOURCE

**Assumptions**
None

**Input**
A - Resource index

**Output**
None

**Destroy**
None

**Example**

; stop a running timer resource
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
KTMR_STOP_RESOURCE

---

**Data Transfer**

**KTMR_COPYRESOURCE_TO_BUFFER**

**Description**
Copy the working HMS data to a buffer of a specified resource using the time resource structure for hour, minute and seconds.

**Usage**
KTMR_COPYRESOURCE_TO_BUFFER

**Assumptions**
None

**Input**
A - Resource index
IY - Start address of buffer to store the timer resource data.

**Output**
None

**Destroy**
HL, IY

**Example**

TimerTimeBuffer:

db 00H ;second
db 00H ;minute
db 00H ;hour
...

; copy working timer hour, minute and seconds to buffer
ld IX, [CORECurrentASDAddress]
ld A, [IX + APPTMRRESOURCEOFFSET0]
ld IX, #TimerTimeBuffer
KTMR_COPYRESOURCE_TO_BUFFER
Resource Utilities

**KTMR_GET_STATUS**

**Description**
Gets the status of a specified resource index.

**Usage**
KTMR_GET_STATUS

**Assumptions**
None

**Input**
A - Resource index

**Output**
B - Resource status

The following constants indicate the status of the timer resource.

<table>
<thead>
<tr>
<th>Constant</th>
</tr>
</thead>
<tbody>
<tr>
<td>KTMRSTATUSRESET</td>
</tr>
<tr>
<td>KTMRSTATUSSTOPPED</td>
</tr>
<tr>
<td>KTMRSTATUSRUNNING</td>
</tr>
</tbody>
</table>

**Destroys**
HL

**Example**
```
ld  IX, [CORECurrentASDAAddress]
ld  A, [IX + APFTMRRESOURCEOFFSET0]
KTMR_GET_STATUS

cp  B, #KTMRSTATUSRESET
jr  Z, TimerIsReset

cp  B, #KTMRSTATUSSTOPPED
jr  Z, TimerIsNotResetAndStopped

cp  B, #KTMRSTATUSRUNNING
jr  Z, TimerIsNotResetAndRunning
```

---

**KTMR_GET_UPDATE_BITS**

**Description**
Get the update bits of a specified resource.

**Usage**
KTMR_GET_UPDATE_BITS

**Assumptions**
None

**Input**
A - Resource index
Output

B - Update flag

The following bit flags indicate which timer data has been updated.

<table>
<thead>
<tr>
<th>Bit Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bKTMRSecondUpd</td>
<td>Set to 1 if seconds data has been updated</td>
</tr>
<tr>
<td>bKTMRMinuteUpd</td>
<td>Set to 1 if minute data has been updated</td>
</tr>
<tr>
<td>bKTMRHourUpd</td>
<td>Set to 1 if hour data has been updated</td>
</tr>
</tbody>
</table>

Destroys HL

Example

```assembly
ld IX, [CORECurrentASDAddress]
lA, [IX + APPTMRRSOURCEOFFSET0]

KTMR_GET_UPDATE_BITS

bit B, #bKTMRHourUpd
jr Z, HourMinuteSecondDisplay

bit B, #bKTMRMinuteUpd
jr Z, MinuteSecondDisplay

bit B, #bKTMRSecondUpd
jr Z, SecondDisplay

HourMinuteSecondDisplay:
; display hour data

MinuteSecondDisplay:
; display minute data

SecondDisplay:
; display second data
```
SYNCHRO RESOURCE API

The available resource index for the STP resource are: 0x00.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KSYNAPPINDEXOFFSET</td>
</tr>
<tr>
<td>1</td>
<td>KSYNRESFLAGOFFSET</td>
</tr>
<tr>
<td>2</td>
<td>KSYNAPPOWNEROFFSET</td>
</tr>
<tr>
<td>3</td>
<td>KSYNTOTALPREVHUNDSAMPLEOFFSET</td>
</tr>
<tr>
<td>4</td>
<td>KSYNTOTALPREVSECONDSSAMPLEOFFSET</td>
</tr>
<tr>
<td>5</td>
<td>KSYNTOTALHUNDREDTHSOFFSET</td>
</tr>
<tr>
<td>6</td>
<td>KSYNTOTALSECONDOFFSET</td>
</tr>
<tr>
<td>7</td>
<td>KSYNTOTALMINUTEOFFSET</td>
</tr>
<tr>
<td>8</td>
<td>KSYNTOTALHOUROFFSET</td>
</tr>
<tr>
<td>9</td>
<td>KSYNUPDFLAGTOTALOFFSET</td>
</tr>
<tr>
<td>10</td>
<td>KSYNSTOPPREVHUNDSAMPLEOFFSET</td>
</tr>
<tr>
<td>11</td>
<td>KSYNSTOPPREVSECONDSSAMPLEOFFSET</td>
</tr>
<tr>
<td>12</td>
<td>KSYNSTOPHUNDREDTHSOFFSET</td>
</tr>
<tr>
<td>13</td>
<td>KSYNSTOPSECONDOFFSET</td>
</tr>
<tr>
<td>14</td>
<td>KSYNSTOPMINUTEOFFSET</td>
</tr>
<tr>
<td>15</td>
<td>KSYNSTOPHOUROFFSET</td>
</tr>
<tr>
<td>16</td>
<td>KSYNUPDFLAGSTOPPAGEOFFSET</td>
</tr>
</tbody>
</table>

KSYNRESFLAGOFFSET Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKReserved</td>
</tr>
<tr>
<td>1</td>
<td>bKActive</td>
</tr>
<tr>
<td>2</td>
<td>bKDispUpdRequest</td>
</tr>
<tr>
<td>3</td>
<td>bKSYNStartStoppage</td>
</tr>
<tr>
<td>4</td>
<td>Unused</td>
</tr>
<tr>
<td>5</td>
<td>bKNotReset</td>
</tr>
<tr>
<td>6</td>
<td>bKGeneratePopup</td>
</tr>
<tr>
<td>7</td>
<td>bKGenerateEvent</td>
</tr>
</tbody>
</table>

KSYNUPDFLAGTOTALOFFSET Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKSYNTotalSecondUpd</td>
</tr>
<tr>
<td>1</td>
<td>bKSYNTotalMinuteUpd</td>
</tr>
<tr>
<td>2</td>
<td>bKSYNTotalHourUpd</td>
</tr>
<tr>
<td>3</td>
<td>Unused</td>
</tr>
<tr>
<td>4</td>
<td>bKSYNTotalSecondNonZero</td>
</tr>
<tr>
<td>5</td>
<td>bKSYNTotalMinuteNonZero</td>
</tr>
<tr>
<td>6</td>
<td>bKSYNTotalHourNonZero</td>
</tr>
</tbody>
</table>
7 bKSYNTotalRunout

KSYNUPDFLAGSTOPPAGEOFFSET Bit Definitions.

<table>
<thead>
<tr>
<th>Bit</th>
<th>Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>bKSYNStoppageSecondUpd</td>
</tr>
<tr>
<td>1</td>
<td>bKSYNStoppageMinuteUpd</td>
</tr>
<tr>
<td>2</td>
<td>bKSYNStoppageHourUpd</td>
</tr>
<tr>
<td>3</td>
<td>Unused</td>
</tr>
<tr>
<td>4</td>
<td>bKSYNStoppageSecondNonZero</td>
</tr>
<tr>
<td>5</td>
<td>bKSYNStoppageMinuteNonZero</td>
</tr>
<tr>
<td>6</td>
<td>bKSYNStoppageHourNonZero</td>
</tr>
<tr>
<td>7</td>
<td>bKSYNStoppageRunout</td>
</tr>
</tbody>
</table>

Data Manipulation

**KSYN_GET_CAUSED_TO_START**

Description
Returns the application index that caused the specified synchro resource to start.

Usage
KSYN_GET_CAUSED_TO_START

Assumptions
None

Input
None

Output
A - Application Index

Destroys
HL

Example
;get the application index that caused the synchro resource to start
KSYN_GET_CAUSED_TO_START

**KSYN_GET_TOTAL_TIME_UPD_BITS**

Description
Gets the TOTAL TIME update status.

Usage
KSYN_GET_TOTAL_TIME_UPD_BITS

Assumptions
None

Input
None

Output
A - Update status for the total time structure

The following bit flags indicate which data field has been updated.
### Bit Flag

<table>
<thead>
<tr>
<th>Bit Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bKSYNTotalSecondUpd</td>
<td>TRUE, if second data is updated</td>
</tr>
<tr>
<td>bKSYNTotalMinuteUpd</td>
<td>TRUE, if minute data is updated</td>
</tr>
<tr>
<td>bKSYNTotalHourUpd</td>
<td>TRUE, if hour data is updated</td>
</tr>
<tr>
<td>bKSYNTotalSecondNonZero</td>
<td>TRUE, if seconds is not zero</td>
</tr>
<tr>
<td>bKSYNTotalMinuteNonZero</td>
<td>TRUE, if minutes is not zero</td>
</tr>
<tr>
<td>bKSYNTotalHourNonZero</td>
<td>TRUE, if hours is not zero</td>
</tr>
<tr>
<td>bKSYNTotalRunout</td>
<td>TRUE, if synchro total time has ran out</td>
</tr>
</tbody>
</table>

### Example

; get the total time update flags
KSYNC_GET_TOTAL_TIME_UPD_BITS
bit A, #bKSYNCTotalSecondUpd
jr NZ, SynchroTotalTimeSecondUpdated
bit A, #bKSYNCTotalMinuteUpd
jr NZ, SynchroTotalTimeMinuteUpdated
bit A, #bKSYNCTotalRunout
jr NZ, SynchroTotalHasRunout

### KSYNC_GET_STOPPAGE_TIME_UPD_BITS

**Description**
Gets the STOPPAGE TIME update status

**Usage**
KSYNC_GET_STOPPAGE_TIME_UPD_BITS

**Assumptions**
None

**Input**
None

**Output**
A - Update status for the stoppage time structure

The following bit flags indicate which data field has been updated.

<table>
<thead>
<tr>
<th>Bit Flag</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bKSYNStoppageSecondUpd</td>
<td>TRUE, if second data is updated</td>
</tr>
<tr>
<td>bKSYNStoppageMinuteUpd</td>
<td>TRUE, if minute data is updated</td>
</tr>
<tr>
<td>bKSYNStoppageHourUpd</td>
<td>TRUE, if hour data is updated</td>
</tr>
<tr>
<td>bKSYNStoppageSecondNonZero</td>
<td>TRUE, if seconds is not zero</td>
</tr>
<tr>
<td>bKSYNStoppageMinuteNonZero</td>
<td>TRUE, if minutes is not zero</td>
</tr>
<tr>
<td>bKSYNStoppageHourNonZero</td>
<td>TRUE, if hours is not zero</td>
</tr>
<tr>
<td>bKSYNStoppageRunout</td>
<td>TRUE, if synchro stoppage time has ran out</td>
</tr>
</tbody>
</table>

**Destroys**
HL

**Example**

; get the stoppage time update flags
KSYNC_GET_STOPPAGE_TIME_UPD_BITS
bit A, #bKSYNCSStoppageSecondUpd
jr NZ, SynchroStoppageTimeSecondUpdated
bit A, #bKSYNCSStoppageMinuteUpd
jr NZ, SynchroStoppageTimeMinuteUpdated
bit A, #bKSYNCSStoppageRunout
**KSYN_GET_TOTAL_STATUS**

Description: Gets the status of the total time. Either reset, stop or running.

Usage: KSYN_GET_TOTAL_STATUS

Assumptions: None

Input: None

Output: B - Status of total time operations.

The following constants indicate the status of total time operations.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSYNTOTALSTATUSRESET</td>
<td>Reset</td>
</tr>
<tr>
<td>KSYNTOTALSTATUSSTOPPED</td>
<td>Stop</td>
</tr>
<tr>
<td>KSYNTOTALSTATUSRUNNING</td>
<td>Running</td>
</tr>
</tbody>
</table>

Destroys: HL

Example:
```
;j get current status of the total time
KSYN_GET_TOTAL_STATUS

cp B, #KSYNTOTALSTATUSRESET
jr Z, SynchroTotalTimeIsReset

cmp B, #KSYNTOTALSTATUSSTOPPED
jr Z, SynchroTotalTimeIsNotResetAndStopped

cp B, #KSYNTOTALSTATUSRUNNING
jr Z, SynchroTotalTimeIsNotResetAndRunning
```

**KSYN_GET_STOPPAGE_STATUS**

Description: Gets the status of the stoppage time. Either reset, stop or running.

Usage: KSYN_GET_STOPPAGE_STATUS

Assumptions: None

Input: None

Output: B - Status of stoppage time operations.
The following constants indicate the status of stoppage time operations.

<table>
<thead>
<tr>
<th>Constant</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSYNSTOPSTATUSRESET</td>
<td>Reset</td>
</tr>
<tr>
<td>KSYNSTOPSTATUSSTOPPED</td>
<td>Stop</td>
</tr>
<tr>
<td>KSYNSTOPSTATUSRUNNING</td>
<td>Running</td>
</tr>
</tbody>
</table>

**Destroys**

HL

**Example**

```asm
; get current status of the stoppage time
KSYN_GET_STOPPAGE_STATUS

cp  B, #KSYNSTOPSTATUSRESET
jr  Z, SynchroStoppageTimeIsReset

cp  B, #KSYNSTOPSTATUSSTOPPED
jr  Z, SynchroStoppageTimeIsNotResetAndStopped

cp  B, #KSYNSTOPSTATUSRUNNING
jr  Z, SynchroStoppageTimeIsNotResetAndRunning
```

---

**KSYN_ENABLE_DISP_UPD_EVENT**

**Description**

Allows the resource to pass a display update event to the foreground application.

**Usage**

KSYN_ENABLE_DISP_UPD_EVENT

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

HL

**Example**

```asm
; enable display events
KSYN_ENABLE_DISP_UPD_EVENT
```

---

**KSYN_DISABLE_DISP_UPD_EVENT**

**Description**

Disallows the resource to pass a display update event to the foreground application.

**Usage**

KSYN_DISABLE_DISP_UPD_EVENT

**Assumptions**

None

**Input**

None
<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
<th>Usage</th>
<th>Assumptions</th>
<th>Input</th>
<th>Output</th>
<th>Destroys</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>KSYN_DISABLE_DISP_UPD_EVENT</td>
<td>Setup resource to generate popup when synchro time has ran-out.</td>
<td>KSYN_DISABLE_DISP_UPD_EVENT</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>HL</td>
<td>;disable display events KSYN_DISABLE_DISP_UPD_EVENT</td>
</tr>
<tr>
<td>KSYN.Setup_Popup_Generation</td>
<td>Setup resource to generate event when synchro time has ran-out</td>
<td>KSYN.Setup_Popup_Generation</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>HL</td>
<td>;request popup when synchro time has reached 100 hours KSYN.Setup_Popup_Generation</td>
</tr>
<tr>
<td>KSYN.Setup_Event_Generation</td>
<td>Setup resource to generate event when synchro time has ran-out</td>
<td>KSYN.Setup_Event_Generation</td>
<td>None</td>
<td>None</td>
<td>None</td>
<td>HL</td>
<td>;request event to be passed to foreground application when ;synchro time has reached 100 hours KSYN.Setup_Event_Generation</td>
</tr>
</tbody>
</table>
### KSYNCANCEL_POPUPEVENT_GENERATION

**Description**  
Cancel resource to generate popup and event when synchro time has ran-out.

**Usage**  
KSYN_CANCEL_POPUPEVENT_GENERATION

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
HL

**Example**  
;no event or popup when synchro time has reached 100 hours  
KSYN_CANCEL_POPUPEVENT_GENERATION

---

### KSYN_RESET_AND_START

**Description**  
Resets and starts the Synchro timer.

**Usage**  
KSYN_RESET_AND_START

**Assumptions**  
KRESStopwatchHundthsDataBuffer and KRESStopwatchSecondsDataBuffer contains the snap shot of the free running hundredths and seconds counter.

**Input**  
None

**Output**  
None

**Destroys**  
None

**Example**  
;reset and start synchro timer  
KSYN_RESET_AND_START

---

### KSYN_STOP_TOTAL_AND_STOPPAGE_TIME

**Description**  
Stops synchro total and stoppage time and update the values with the stored RTC.

**Usage**  
KSYN_STOP_TOTAL_AND_STOPPAGE_TIME

**Assumptions**  
KRESStopwatchHundthsDataBuffer and KRESStopwatchSecondsDataBuffer contains the snap shot of the free running hundredths and seconds counter.
This will only be called when the resource is forcibly stopped due to switch depression (in synchro mode).

Input
None

Output
None

Destroys
None

Example
;stop both total time and stoppage time
KSYN_STOP_TOTAL_AND_STOPPAGE_TIME

---

**KSYN_RESET_RESOURCE**

Description
Resets a synchro resource to 00:00:00.00.

Usage
KSYN_RESET_RESOURCE

Assumptions
None

Input
None

Output
None

Destroys
None

Example
;reset synchro resource to 0
KSYN_RESET_RESOURCE

---

**Data Transfer**

**KSYN_COPY_TOTAL_TO_BUFFER**

Description
Copies the total time H:M:S.h into a buffer

Usage
KSYN_COPY_TOTAL_TO_BUFFER

Assumptions
None

Input
HL - Start address of buffer to write into

Output
HL - Points to the next byte after the destinations structure

Destroys
B, IY
Example

MyTimeBuffer:

db 00H ; hundredth
db 00H ; second
db 00H ; minute
db 00H ; hour

ld HL, #MyTimeBuffer
KSYN_COPY_STOPPAGE_TO_BUFFER

KSYN_COPY_STOPPAGE_TO_BUFFER

Description
Copies the stoppage time H:M:S.h into a buffer.

Usage
KSYN_COPY_STOPPAGE_TO_BUFFER

Assumptions
None

Input
HL - Start address of buffer to write into

Output
HL - Points to the next byte after the destinations structure

Destroys
B, IY

Example
MyTimeBuffer:

db 00H ; hundredth
db 00H ; second
db 00H ; minute
db 00H ; hour

ld HL, #MyTimeBuffer
KSYN_COPY_STOPPAGE_TO_BUFFER
**UTILITIES API**

**KRES_CLEAR_N_BYTES**

**Description**  
Clears n consecutive bytes

**Usage**  
KRES_CLEAR_N_BYTES

**Assumptions**  
None

**Input**  
B - number of bytes to clear  
HL - start address of the buffer to be cleared

**Output**  
None

**Destroys**  
B, HL

**Example**  
MyBuffer:
```
db  0FFH
db  0ECH
db  012H
db  034H
```

MyBufferEnd:
```
...
```

;setup the number of bytes to clear  
ld  B, #(MyBufferEnd - MyBuffer)

;setup the start address to clear  
ld  HL, #MyBuffer  
KRES_CLEAR_N_BYTES

---

**HW_RESET_MCU**

**Description**  
Reset the MCU.

**Usage**  
HW_RESET_MCU

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
Doesn’t matter.

**Example**  
;reset the microcontroller  
HW_RESET_MCU
**HW_RESET_WATCHDOG**

**Description**
Reset the WATCHDOG timer to prevent a reset.

Normal operation of the M851 OS will reset the watchdog timer. If a certain operation within the wristapp takes longer than 3 seconds, then it should use the API within the procedure to prevent the watch reset.

**Usage**
HW_RESET_WATCHDOG

**Assumptions**
BR is point to the hardware memory page

**Input**
None

**Output**
None

**Destroys**
None

**Example**
; reset the watchdog timer
HW_RESET_WATCHDOG

---

**HW_LAMP_POPUP_REQUEST**

**Description**
Sets lamp during popup request bit to enable lamp flashing during a popup.

This allows the audio generator to turn on the lamp during the OFF phases (read: no frequency generated or a rest tone) of the melody pattern.

**Usage**
HW_LAMP_POPUP_REQUEST

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
HL

**Example**
; request to alternate between lamp and buzzer during a popup melody
HW_LAMP_POPUP_REQUEST
AUDSTART_SYSTEM_MELODY AUDALARMMELODY, AUDSENDMELODYDONEEVENT
**HW_LAMP_POPUP_REQUEST_OFF**

**Description**  
Clears lamp during popup request bit to disable lamp flashing during a popup.

**Usage**  
HW_LAMP_POPUP_REQUEST_OFF

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
HL

**Example**  
;popup is complete request for a popdown  
HW_LAMP_POPUP_REQUEST_OFF  
ORE_REQ_POPDOWN

---

**UTL_BINARY_MATH_MODE**

**Description**  
Setup micro-processor to perform add and sub instructions in binary mode.

By default, the MCU is in binary mode during WristApp execution. The MCU will be in binary mode until the macro UTL_DECIMAL_MATH_MODE is called. On return to the M851 OS, the system will set it back to binary mode.

**Usage**  
UTL_BINARY_MATH_MODE

**Assumptions**  
None

**Input**  
None

**Output**  
None

**Destroys**  
None

**Example**  
ld A, #25H  
add A, #09H  
;result A = 2EH

---

**UTL_DECIMAL_MATH_MODE**

**Description**  
Setup micro-processor to perform add and sub instructions in binary mode. NOTE: Not all add and sub
instructions support decimal operations.

By default, the MCU is in binary mode during WristApp execution. The MCU will be in decimal mode until the macro UTL_BINARY_MATH_MODE is called. On return to the M851 OS, the system will set it back to binary mode.

**Usage**

**UTL_DECIMAL_MATH_MODE**

**Assumptions**

None

**Input**

None

**Output**

None

**Destroys**

None

**Example**

```
ld A, #25H
add A, #09H
;result A = 34H
```

---

**Conversion**

**UTL_CONVERT_HEX_TO_3DIGIT_BCD**

**Description**

Converts a 1-byte HEX number to 2-byte BCD number.

**Usage**

**UTL_CONVERT_HEX_TO_3DIGIT_BCD**

**Assumptions**

None

**Input**

L - hex data to be converted (0x00 – 0xFF)

**Output**

B - 100's digit BCD data
A - 10's digit BCD data (high nibble)
1's digit BCD data (low nibble)

**Destroys**

None

**Example**

```
;convert hex 0FFH to BCD
ld L, #FFH
UTL_CONVERT_HEX_TO_3DIGIT_BCD
;result: B = 02H
;result: A = 55H
```
**UTL_CONVERT_HEX_TO_2DIGIT_BCD**

**Description**
Converts a 1-byte HEX number to 1-byte BCD number.

**Usage**
UTL_CONVERT_HEX_TO_2DIGIT_BCD

**Assumptions**
None

**Input**

L - hex data to be converted (0x00 – 0x63)

Maximum HEX number that can be converted is '0x63' since the 1-byte BCD can only hold up to '99'.

**Output**

A - 10's digit BCD data (high nibble)

1's digit BCD data (low nibble)

**Destroys**
None

**Example**

;convert HEX to BCD
ld  L, #63H
UTL_CONVERT_HEX_TO_2DIGIT_BCD

;result: A = 99H

---

**UTL_CONVERT_2BYTE_HEX_TO_2BYTE_BCD**

**Description**
Converts a 2-byte HEX number to 2-byte BCD number.

**Usage**
UTL_CONVERT_2BYTE_HEX_TO_2BYTE_BCD

**Assumptions**
None

**Input**

HL - HEX data to be converted

**Output**

BA - BCD equivalent

**Destroys**
None

**Example**

;convert hex to BCD
ld  HL, #1234H
UTL_CONVERT_2BYTE_HEX_TO_2BYTE_BCD

;result: BA = 4460H

---

**UTL_CONVERT_1BYTE_BCD_TO_1BYTE_HEX**
**UTL_CONVERT_1BYTE_BCD_TO_1BYTE_HEX**

**Description**
Converts a 1-byte BCD number to 1-byte HEX number.

**Usage**
`UTL_CONVERT_1BYTE_BCD_TO_1BYTE_HEX`

**Assumptions**
None

**Input**
A - BCD data to be converted

**Output**
A - hexadecimal byte

**Destroys**
B, HL

**Example**

```
; convert BCD to HEX
ld A, #99H
UTL_CONVERT_1BYTE_BCD_TO_1BYTE_HEX

; result A = 63H
```

**UTL_CONVERT_4BYTE_FROM_BCD_TO_HEX**

**Description**
Converts a 4 byte buffer from BCD to HEX.

**Usage**
`UTL_CONVERT_4BYTE_FROM_BCD_TO_HEX`

**Assumptions**
Dataset is stored in a consecutive 4 byte buffer, order least significant to most significant.

**Input**
IX - Points to the least significant byte of the data buffer.

**Output**
IX - Points to the most significant byte of the data buffer. The buffer contains the data in HEX format.

**Destroys**
B

**Example**

```
MyBCDNumber:
    db  99H
    db  35H
    db  15H
    db  09H

; convert BCD to HEX
ld IX, #MyBCDNumber
UTL_CONVERT_4BYTE_FROM_BCD_TO_HEX

; results stored in MyBCDNumber
; MyBCDNumber:
; db 63H
; db 23H
; db 0FH
; db 09H
```
**UTL_CONVERT_4BYTE_FROM_HEX_TO_BCD**

**Description**
Converts a 4 byte buffer from HEX to BCD.

**Usage**
UTL_CONVERT_4BYTE_FROM_HEX_TO_BCD

**Assumptions**
Dataset is stored in a consecutive 4 byte buffer, order least significant to most significant.

**Input**
IX - Points to the least significant byte of the data buffer.

**Output**
IX - Points to the most significant byte of the data buffer. The buffer contains the data in BCD format.

**Destroys**
B

**Example**
MyBCDNumber:

```
  db 63H
  db 23H
  db 0FH
  db 09H
```

; convert HEX to BCD
ld IX, #MyBCDNumber
UTL_CONVERT_4BYTE_FROM_HEX_TO_BCD

; results stored in MyBCDNumber
; MyBCDNumber:
;
;   db 99H
;   db 35H
;   db 15H
;   db 09H

---

**UTL_CONVERT_TO_12HR_FORMAT**

**Description**
Converts the 24 hour data into 12 hour data and determine if it is AM or PM.

**Usage**
UTL_CONVERT_TO_12HR_FORMAT

**Assumptions**
Data is in BCD format.

**Input**
A - Hour data to be formatted.

**Output**
A - 12 hour format data.
CarryFlag - TRUE if AM
FALSE if PM

**Destroys**
None
Example

;convert the hour into 12hour format
ld A, #22H
UTL_CONVERT_TO_12HR_FORMAT
;result A = 10H

jr C, DisplayAMText
jr NC, DisplayPMText

Math Functions

**UTL_DIVIDE_HMSH_BY_N_IN_HEX**

**Description**
Divides a data buffer of the format Hr:Min:Sec.Hun by a divisor N and places the result in the buffer.

**Usage**
UTL_DIVIDE_HMSH_BY_N_IN_HEX

**Assumptions**
Dataset is stored in a consecutive 4 byte buffer, order least significant to most significant.
Data is unsigned, in Hexadecimal format

**Input**
- IX - Points to the most significant byte of the data buffer. (HEX data)
- A  Divisor. (HEX data)

**Output**
- IX - Points to the least significant byte of the data buffer. The buffer contains the data (result) in hexadecimal format.

**Destroys**
HL

**Example**

MyTimeBuffer:

;Time: 01:30:16.50
db 50H ;hundredths
db 16H ;second
db 30H ;minute
db 01H ;hour

NumberOfLaps:

db 0AH ;10 laps

;convert time buffer from BCD to HEX
ld IX, #MyBCDNumber
UTL_CONVERT_4BYTE_FROM_BCD_TO_HEX

ld IX, #(MyTimeBuffer+3) ;point to hour data (most significant byte)
ld A, [NumberOfLaps]
UTL_DIVIDE_HMSH_BY_N_IN_HEX

;convert back result to BCD
ld IX, #MyBCDNumber
UTL_CONVERT_4BYTE_FROM_HEX_TO_BCD

;result:
; Time: 00:09.01.65
; db 65H ;hundredths
UTL_DIVIDE_HMSH_BY_N_IN_BCD

Description
Divides a data buffer of the format Hr:Min:Sec.Hun by a divisor N and places the result in the buffer.

Usage
UTL_DIVIDE_HMSH_BY_N_IN_BCD

Assumptions
Dataset is stored in a consecutive 4 byte buffer, order least significant to most significant.
Data is unsigned, in Hexadecimal format

Input
IX - Points to the LSB of the data buffer (BCD Format).
A Divisor. (HEX data)

Output
IX - Points to the most significant byte of the data buffer.

Destroys
B, HL

Example
MyTimeBuffer:
; Time: 01:30:16.50
; db 50H ; hundredths
; db 16H ; second
; db 30H ; minute
; db 01H ; hour

NumberOfLaps:
; db 0AH ; 10 laps
...

ld IX, #(MyTimeBuffer+3) ; point to hour data (most significant byte)
lA, [NumberOfLaps]
UTL_DIVIDE_HMSH_BY_N_IN_BCD

; result:
; Time: 00:09.01.65
; db 65H ; hundredths
; db 01H ; second
; db 09H ; minute
; db 00H ; hour

UTL_ADD_HMSH

Description
Adds data in the Util Hr:Min:Sec:Hun buffer to user data.

Usage
UTL_ADD_HMSH
Assumptions

Data is already loaded into the UTL Hr:Min:Sec:Hun (KRESxxxBuffer). The structure of the KRESxxxBuffer is shown below:

- KRESHundBuffer equ (KRESBuffer+0)
- KRESSecBuffer equ (KRESBuffer+1)
- KRESMinBuffer equ (KRESBuffer+2)
- KRESHrBuffer equ (KRESBuffer+3)
- KRESHrRollOverValue equ (KRESBuffer+4)
- KRESHourRolloverExcess equ (KRESBuffer+5)

Input

HL - Points to the hundredths data of the user buffer.

Output

HL - Points to last data location in buffer. (Hour data)
A - Set to indicate data that was modified after addition.

The register A will be have the following bits set:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Seconds Changed</td>
</tr>
<tr>
<td>1</td>
<td>Minutes Changed</td>
</tr>
<tr>
<td>2</td>
<td>Hours Changed</td>
</tr>
</tbody>
</table>

Destroys

May destroy the contents of the following variables: KRESHrBuffer, KRESMinBuffer, KRESSecBuffer and KRESHundBuffer.

Example

MyUserData:

```assembly
;user time: 01:05.10.50
db  50H ;hundredths
db  10H ;seconds
db  05H ;minute
db  01H ;hour
```

```assembly
;clear all entries in the KRESBuffer
ld  B, #6
ld  HL, #KRESBuffer
KRES_CLEAR_N_BYTES

;setup the data to add and store in KRESBuffer
;add 30 minutes to the user data
ld  A, #30H
ld  [KRESMinBuffer], A

;add 30 minutes to user data
ld  HL, #MyUserData
UTL_ADD_HMSH

;result: new user time = 01:35.10.50
```

**UTL_SUBTRACT_HMSH**

Description

Subtracts data in the UTL Hr:Min:Sec:Hun buffer from user data.
Usage

UTL_SUBTRACT_HMSH

Assumptions

Data is already loaded into the UTL Hr:Min:Sec:Hun (KRESxxxBuffer). The structure of the KRESxxxBuffer is shown below:

- KRESHundBuffer          equ (KRESBuffer+0)
- KRESSecBuffer            equ (KRESBuffer+1)
- KRESMinBuffer            equ (KRESBuffer+2)
- KRESHrBuffer             equ (KRESBuffer+3)
- KRESHrRollOverValue      equ (KRESBuffer+4)
- KRESHourRolloverExcess   equ (KRESBuffer+5)

Input

HL - Points to the hundredths data of the user buffer.

Output

HL - Points to last data location in buffer. (Hour data)
A - Set to indicate data that was modified after addition.

The register A will be have the following bits set:

<table>
<thead>
<tr>
<th>Bit</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Seconds Changed</td>
</tr>
<tr>
<td>1</td>
<td>Minutes Changed</td>
</tr>
<tr>
<td>2</td>
<td>Hours Changed</td>
</tr>
</tbody>
</table>

Destroys

May destroy the contents of the following variables: KRESHrBuffer, KRESMinBuffer, KRESSecBuffer and KRESHundBuffer.

Example

MyUserData:

```
;user time: 01:05.10.50
db 50H ;hundredths
db 10H ;seconds
db 05H ;minute
db 01H ;hour
```

;clear all entries in the KRESBuffer
ld B, #6
ld HL, #KRESBuffer
KRES_CLEAR_N_BYTES

;set up the data to add and store in KRESBuffer
;subtract 30 minutes to the user data
ld A, #30H
ld [KRESMinBuffer], A

;subtract 30 minutes to user data
ld HL, #MyUserData
UTL_SUBTRACT_HMSH

;result: new user time = 00:35.10.50
Copy

UTL_COPY_BUFFER

Description
Copies the data pointed by IYReg to the location pointed by HLReg.

Usage
UTL_COPY_BUFFER

Assumptions
None

Input
B - Number of bytes to copy
IY - Base address of the source data
HL - Base address of the destination

Output
None

Destroys
None

Example
MyUserData:

;user time: 01:05.10.50
db 50H ;hundredths
db 10H ;seconds
db 05H ;minute
db 01H ;hour
;
;copy user data into KRESBuffer
ld B, #4
ld HL, #KRESBuffer ;destination buffer
ld IY, #MyUserData ;source buffer
UTL_COPY_BUFFER

UTL_COPY_IYREG_TO_IXREG

Description
Copies the data pointed by IYReg to the location pointed by IXReg.

Usage
UTL_COPY_IYREG_TO_IXREG

Assumptions
None

Input
B - Number of bytes to copy
IY - Base address of the source data
IX - Base address of the destination

Output
None

Destroys
None

Example
MyUserData:
db 50H ;hundredths
db 10H ;seconds
db 05H ;minute
db 01H ;hour

MyWorkData:
    db 00H
    db 00H
    db 00H

;copy 4 bytes from MyUserData structure to the MyWorkData structure
ld B, #4
ld IY, #MyUserData
ld IX, #MyWorkData
UTL_COPY_IYREG_TO_IXREG

---

Comparison

**UTL_COMPARE_4BYTE_BUFFER**

**Description**

Compares the data in a 4 byte buffer, and sets the MCU condition flags to indicate if one data set is greater than, less than or equal.

**Usage**

UTL_COMPARE_4BYTE_BUFFER

**Assumptions**

None

**Input**

<table>
<thead>
<tr>
<th>HL</th>
<th>points to most significant byte of data to be compared to</th>
</tr>
</thead>
<tbody>
<tr>
<td>IX</td>
<td>points to most significant byte of data we are comparing</td>
</tr>
</tbody>
</table>

**Output**

- bZeroFlag = TRUE , bCarryFlag = FALSE  - HLReg = IXReg
- bZeroFlag = FALSE , bCarryFlag = TRUE   - HLReg < IXReg
- bZeroFlag = FALSE , bCarryFlag = FALSE   - HLReg > IXReg

**Destroys**

BA, HL, IX

**Example**

MyUserData:
    db 50H ;hundredths
    db 10H ;seconds
    db 05H ;minute
    db 01H ;hour

MyWorkData:
    db 00H
    db 00H
    db 00H
db 00H

...;
compare MyUserData with MyWorkData structures
ld B, #4
ld HL, # (MyUserData + 3) ; point to hour data (most significant byte)
ld IX, # (MyWorkData + 3) ; point to hour data (most significant byte)
UTL_COMPARE_4BYTE_BUFFER

jr Z, TwoStructuresAreEqual
jr C, UserDataIsLessThanWorkData
jr NC, UserDataIsGreaterThanWorkData

**UTL_COMPARE_HLREG_WITH_IXREG**

**Description**
Compares two structures as pointed by HLReg and IXReg.

**Usage**
UTL_COMPARE_HLREG_WITH_IXREG

**Assumptions**
None

**Input**

<table>
<thead>
<tr>
<th>Input</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HL</td>
<td>- points to most significant byte of data to be compared to</td>
</tr>
<tr>
<td>IX</td>
<td>- points to most significant byte of data we are comparing</td>
</tr>
<tr>
<td>B</td>
<td>- Number of byte fields are to be compared</td>
</tr>
</tbody>
</table>

**Output**

- bZeroFlag = TRUE, bCarryFlag = FALSE - HLReg = IXReg
- bZeroFlag = FALSE, bCarryFlag = TRUE - HLReg < IXReg
- bZeroFlag = FALSE, bCarryFlag = FALSE - HLReg > IXReg

**Destroys**
BA, HL, IX

**Example**

MyUserData:

```
db 50H
db 10H
db 05H
db 01H
db 01H
db 01H
db 01H
db 01H
db 01H
```

MyWorkData:

```
db 00H
db 00H
db 00H
db 00H
db 00H
db 00H
db 00H
db 00H
db 00H
```

...;
compare MyUserData with MyWorkData structures
ld B, #7
ld HL, #((MyUserData+6) ;point to most significant byte
ld IX, #((MyWorkData+6) ;point to most significant byte
UTL_COMPARE_HLREG_WITH_IXREG
jr Z, TwoStructuresAreEqual
jr C, UserDataIsLessThanWorkData
jr NC, UserDataIsGreaterThanWorkData

UTL_COMPARE_HLREG_WITH_IYREG

**Description**
Compared two structures as pointed by HLReg and IYReg.

**Usage**
UTL_COMPARE_HLREG_WITH_IYREG

**Assumptions**
None

**Input**
- HL - points to most significant byte of data to be compared to
- IY - points to most significant byte of data we are comparing
- B - Number of bytes fields are to be compared

**Output**
- bZeroFlag = TRUE, bCarryFlag = FALSE - HLReg = IXReg
- bZeroFlag = FALSE, bCarryFlag = TRUE - HLReg < IXReg
- bZeroFlag = FALSE, bCarryFlag = FALSE - HLReg > IXReg

**Destroys**
BA, HL, IY

**Example**
MyUserData:
```assembly
db 50H
db 10H
db 05H
db 01H
db 01H
db 01H
db 01H
```
MyWorkData:
```assembly
db 00H
db 00H
db 00H
db 00H
db 00H
db 00H
db 00H
```

`;compare MyUserData with MyWorkData structures
ld B, #7
ld HL, #((MyUserData+6) ;point to most significant byte
ld IY, #((MyWorkData+6) ;point to most significant byte
UTL_COMPARE_HLREG_WITH_IYREG
jr Z, TwoStructuresAreEqual  
jr C, UserDataIsLessThanWorkData  
jr NC, UserDataIsGreaterThanWorkData

---

**Acceleration**

**UTLACCELERATION_1SEC**

**Description**
1 Sec acceleration table.

**Usage**
UTLACCELERATION_1SEC

**Assumptions**
None

**Input**
COREEventArgument - Index into the acceleration table contains the number of pulses detected within 16hz window.

**Output**
KRESSecBuffer - the Acceleration value of data

**Destroys**
HL, IY, IX

**Example**
;clear all entries in the KRESBuffer
ld B, #6
ld HL, #KRESBuffer
KRES_CLEAR_N_BYTES

;data to convert is already loaded in the COREEVENTArgument variable
UTLACCELERATION_1SEC

;result in KRESSecBuffer

---

**UTL_ACCELERATION_1INCREMENT**

**Description**
Generic 1 unit acceleration table.

**Usage**
UTL_ACCELERATION_1INCREMENT

**Assumptions**
None

**Input**
COREEventArgument - Index into the acceleration table contains the number of pulses detected within 16hz window.

**Output**
KRESSecBuffer - the Acceleration value of data

**Destroys**
HL, IY, IX
Example

; clear all entries in the KRESBuffer
ld B, #6
ld HL, #KRESBuffer
KRES_CLEAR_N_BYTES

; data to convert is already loaded in the COREEVENTArgument variable
UTLACCELERATION_1INCREMENT

; result in KRESSecBuffer

UTLACCELERATION_1MIN

Description
1 Min acceleration table.

Usage
UTLACCELERATION_1MIN

Assumptions
Clears the KRESxxxBuffer first before loading the acceleration value.

Input
COREEventArgument - Index into the acceleration table contains the number of pulses detected within 16hz window.

Output
KRESMinBuffer the Acceleration value of data

Destroys
HL, IY, IX

Example

; clear all entries in the KRESBuffer
ld B, #6
ld HL, #KRESBuffer
KRES_CLEAR_N_BYTES

; data to convert is already loaded in the COREEVENTArgument variable
UTLACCELERATION_1MIN

; result in KRESMinBuffer

UTLACCELERATION_5MIN

Description
5 Min acceleration table

Usage
UTLACCELERATION_5MIN

Assumptions
Clears the KRESxxxBuffer first before loading the acceleration value.

Input
COREEventArgument - Index into the acceleration table contains the number of pulses detected within 16hz window.

Output
KRESMinBuffer the Acceleration value of data
**UTL_ACCELERATION_5INCREMENT**

**Description**
Generic 5 units acceleration

**Usage**
UTL_ACCELERATION_5INCREMENT

**Assumptions**
Clears the KRESxxxBuffer first before loading the acceleration value.

**Input**
COREEventArgument - Index into the acceleration table contains the number of pulses detected within 16hz window.

**Output**
KRESMinBuffer - the Acceleration value of data

**Destroys**
HL, IY, IX

**Example**
;clear all entries in the KRESBuffer
ld B, #6
ld HL, #KRESBuffer
KRES_CLEAR_NgetBytes

;data to convert is already loaded in the COREEVENTArgument variable
UTLACCELERATION_5INCREMENT

;result in KRESMinBuffer

---

**UTLACCELERATION_DATE**

**Description**
Get number of days from the utlAccelerationTable1Min acceleration table and return the number in A.

**Usage**
UTLACCELERATION_DATE

**Assumptions**
Clears the KRESxxxBuffer first before loading the acceleration value.

**Input**
COREEventArgument - Index into the acceleration table contains the number of pulses detected within 16hz window.

**Example**
;clear all entries in the KRESBuffer
ld B, #6
ld HL, #KRESBuffer
KRES_CLEAR_NBYTES

;data to convert is already loaded in the COREEVENTArgument variable
UTLACCELERATION_5INCREMENT

;result in KRESMinBuffer
UTLACCELERATION_DATE

Description
Get number of days from the utlAccelerationTable1Min acceleration table and return the number in A.

Usage
UTLACCELERATION_DATE

Assumptions
Clears the KRESxxxBuffer first before loading the acceleration value.

Input
COREEventArgument - Index into the acceleration table contains the number of pulses detected within 16hz window.

Output
A
Number of days from the acceleration table for the given number of pulses.

Destroys
None

Example
;clear all entries in the KRESBuffer
ld B, #6
ld HL, #KRESBuffer
KRES_CLEAR_N_BYTES

;data to convert is already loaded in the COREEVENTArgument variable
UTLACCELERATION_DATE

;result in A register

Message Editing

UTL_SETUP_VARS_FOR_EDITING

Description
Sets up the variables used for editing a message. This includes the sentinel character.

Usage
UTL_SETUP_VARS_FOR_EDITING

Assumptions
LCDScrollBuffer should contain the message to be edited.

Input
B
Number of characters allocated for the message.

Output
None
**DESTROYS**

None

**Example**

LCDScrollBuffer:

```
db   DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
    db   DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
    db   DM5_SENTINEL
```

```
ld  B, #17
```

**UTL_FILL_SCROLL_BUFFER_WITH_SPACE**

**Description**

Fills up the scroll buffer with space character.

**Usage**

UTL_FILL_SCROLL_BUFFER_WITH_SPACE

**Assumptions**

LCDScrollBuffer should contain the message to be edited.

**Input**

None

**Output**

None

**Destroys**

None

**Example**

LCDScrollBuffer:

```
db   DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
    db   DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
    db   DM5_SENTINEL
```

```
;fill up scroll buffer (101 bytes total) with space
UTL_FILL_SCROLL_BUFFER_WITH_SPACE
```

```
;RESULT: all 101 bytes in LCDScrollBuffer will be set to DM5_SPACE
```

**UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK**

**Description**

Move to the next character to be edited if it is not yet the last character. Display cursor on the position of the character to be edited. Request blinking on the character to be edited.

**Usage**

UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK

**Assumptions**

LCDScrollBuffer should contain the message to be edited.

**Input**

None
Output  

bCarryFlag  

-  0 -- if moving the cursor forward is valid.

  1 -- if moving the cursor forward is invalid

Destroys  

BA, IX, IY, HL

Example  

LCDScrollBuffer:

    db  DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
    db  DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
    db  DM5_SENTINEL

;setup for message editing
ld  B, #17
UTL_SETUP_VARS_FOR EDITING

;display the section of message where character is to be edited
UTL_DISPLAY_MSG_AND_REQ_BLINK

;move cursor forward
UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK

UTL_MOVE_CURSOR_BACKWARD_AND_REQ_BLINK

Description  

Move to the previous character to be edited if its not yet the first character.  
Display cursor on the position of the character to be edited.  
Request blinking on the character to be edited.

Usage  

UTL_MOVE_CURSOR_BACKWARD_AND_REQ_BLINK

Assumptions  

LCDScrollBuffer should contain the message to be edited.

Input  

None

Output  

bCarryFlag  

-  0 -- if moving the cursor backward is valid.

  1 -- if moving the cursor backward is invalid

Destroys  

BA, IX, IY, HL

Example  

LCDScrollBuffer:

    db  DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
    db  DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
    db  DM5_SENTINEL

;setup for message editing
ld  B, #17
UTL_SETUP_VARS_FOR EDITING

;display the section of message where character is to be edited
UTL_DISPLAY_MSG_AND_REQ_BLINK

;move cursor forward
UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK
;move cursor forward
UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK

;move cursor forward
UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK

;move cursor backward
UTL_MOVE_CURSOR_BACKWARD_AND_REQ_BLINK

;move cursor backward
UTL_MOVE_CURSOR_BACKWARD_AND_REQ_BLINK

--

**UTL_DISPLAY_MSG_AND_REQ_BLINK**

**Description**
Displays a part of the message that has the character to be edited. The cursor character will be displayed on the position of the character to be edited.

**Usage**
UTL_DISPLAY_MSG_AND_REQ_BLINK

**Assumptions**
LCDScrollBuffer should contain the message to be edited.

**Input**
None

**Output**
None

**Destroys**
BA, IX, IY, HL

**Example**
LCDScrollBuffer:
```assembly
db   DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
    db   DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
    db   DM5_SENTINEL
...```

;display the section of message where character is to be edited
UTL_DISPLAY_MSG_AND_REQ_BLINK

--

**UTL_POINT_TO_PREV_CHAR_AND_REQ_BLINK**

**Description**
Point to the previous character (refer to lcd character table on what the previous character with respect to the character to be changed). Displays the new character and request blinking.

**Usage**
UTL_POINT_TO_PREV_CHAR_AND_REQ_BLINK

**Assumptions**
LCDScrollBuffer should contain the message to be edited.

**Input**
B - Number to be subtracted to the current character definition to point to the next
desired character. This would look like it is accelerated

Output
None

Destroys
BA, IX, IY, HL

Example
LCDScrollBuffer:
   db  DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
   db  DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
   db  DM5_SENTINEL

;setup for message editing
ld  B, #17
UTL_SETUP_VARS_FOR_EDITING

;display the section of message where character is to be edited
UTL_DISPLAY_MSG_AND_REQ_BLINK

;change character under cursor to the next available character index
;with wraparound with the offset position defined in the
;COREEventArgument variable
ld  B, [COREEventArgument]
UTL_POINT_TO_PREV_CHAR_AND_REQ_BLINK

UTL_POINT_TO_NEXT_CHAR_AND_REQ_BLINK

Description
Point to the next character (refer to LCD character table on what is the next character with respect to the character to be changed).
Displays the new character and request blinking.

Usage
UTL_POINT_TO_NEXT_CHAR_AND_REQ_BLINK

Assumptions
LCDScrollBuffer should contain the message to be edited.

Input
B - Number to be subtracted to the current character definition to point to the next desired character. This would look like it is accelerated

Output
None

Destroys
BA, IX, IY, HL

Example
LCDScrollBuffer:
   db  DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
   db  DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
   db  DM5_SENTINEL

;setup for message editing
ld  B, #17
UTL_SETUP_VARS_FOR_EDITING

;display the section of message where character is to be edited
UTL_DISPLAY_MSG_AND_REQ_BLINK

; change character under cursor to the previous available character index
; with wraparound with the offset position defined in the
; COREEventArgument variable
ld  B, [COREEventArgument]
UTL_POINT_TO_PREV_CHAR_AND_REQ_BLINK

**UTL_POINT_TO_PREV_CHAR**

**Description**
Point to the previous character (refer to lcd character table on what is the previous character with respect to the character to be changed).

**Usage**
UTL_POINT_TO_PREV_CHAR

**Assumptions**
None

**Input**
IY  Base address of the message where the character will be edited
L  Offset from the base address of the character to be edited
B  Number to be subtracted to the current character definition to point to the next desired character. This would look like it is accelerated.

**Output**
A  New character. The new character is stored on the position pointed by (IYReg + LReg).

**Destroys**
None

**Example**
MyCityCode:
   db   DM5_N, DM5_Y DM5_C
   ...

   ; change character specified to the previous available character index
   ; with wraparound with the offset position defined in the
   ; COREEventArgument variable
   ld  IY, #MyCityCode
   ld  L,  #1                   ; edit the 2ND city code
   ld  B,  #1                   ; get previous character in set
   UTL_POINT_TO_PREV_CHAR

   ; result
   ; MyCityCode:
   ;   db   DM5_N, DM5_X, DM5_C

**UTL_POINT_TO_NEXT_CHAR**

**Description**
Point to the next character (refer to lcd character table on what is the previous character with respect to the character to be changed).
Usage

**UTL_POINT_TO_NEXT_CHAR**

Assumptions

None

Input

- **IY**
  - Base address of the message where the character will be edited
- **L**
  - Offset from the base address of the character to be edited
- **B**
  - Number to be subtracted to the current character definition to point to the next desired character. This would look like it is accelerated.

Output

- **A**
  - New character. The new character is stored on the position pointed by (IYReg + LReg).

Destroys

None

Example

```assembly
MyCityCode:
    db   DM5_N, DM5_Y DM5_C
    ...

; change character specified to the next available character index
; with wraparound with the offset position defined in the
; COREEventArgument variable
ld  IY, #MyCityCode
ld  L,  #1                   ; edit the 2ND city code
ld  B,  #1                   ; get next character in set
UTL_POINT_TO_NEXT_CHAR

; result
; MyCityCode:
;   db   DM5_N, DM5_Z, DM5_C
```

---

**UTL_CLEANUP_EDIT_BUFFER**

Description

Trims the trailing spaces.

Usage

**UTL_CLEANUP_EDIT_BUFFER**

Assumptions

LCDScrollBuffer should contain the message to be edited.

Input

None

Output

None

Destroys

None

Example

```assembly
LCDScrollBuffer:
    db   DM5_A, DM5_B DM5_SPACE, DM5_D, DM5_E, DM5_F, DM5_G, DM5_SPACE
    db   DM5_I, DM5_J, DM5_SPACE, DM5_SPACE, DM5_SPACE, DM5_SPACE, DM5_SPACE,
    db   DM5_SENTINEL
    ...

; clean up edit buffer of trailing spaces
```
UTL_CLEANUP_EDIT_BUFFER

;result
;MyCityCode:
;    ;message "AB DEFG IJ" + SENTINEL
;    db   DM5_A, DM5_B DM5_SPACE, DM5_D, DM5_E, DM5_F, DM5_G, DM5_SPACE
;    db   DM5_I, DM5_J, DM5_SENTINEL, DM5_SENTINEL, DM5_SENTINEL
;    db   DM5_SENTINEL, DM5_SENTINEL, DM5_SENTINEL
;    db   DM5_SENTINEL

UTL_TRIM_SENTINEL_CHAR

Description  Trims the trailing sentinel character. The last sentinel character is not included.

Usage  UTL_TRIM_SENTINEL_CHAR

Assumptions  LCDScrollBuffer should contain the message to be edited.

Input  None

Output  None

Destroys  None

Example  LCDScrollBuffer:
        ;message "AB DEFG IJ" + SENTINEL
        db   DM5_A, DM5_B DM5_SPACE, DM5_D, DM5_E, DM5_F, DM5_G, DM5_SPACE
        db   DM5_I, DM5_J, DM5_SENTINEL, DM5_SENTINEL, DM5_SENTINEL
        db   DM5_SENTINEL, DM5_SENTINEL, DM5_SENTINEL
        db   DM5_SENTINEL
        ...
        ;setup for message editing
        ld  B, #17
        UTL_SETUP_VARS_FOR_EDITING
        ...
        ;setup the buffer with spaces until the end of the valid edit buffer size
        UTL_TRIM_SENTINEL_CHAR
        ...
        ;result
        ;MyCityCode:
        ;    ;message "AB DEFG IJ" + SENTINEL
        ;    db   DM5_A, DM5_B DM5_SPACE, DM5_D, DM5_E, DM5_F, DM5_G, DM5_SPACE
        ;    db   DM5_I, DM5_J, DM5_SPACE, DM5_SPACE, DM5_SPACE, DM5_SPACE,
        ;    db   DM5_SPACE, DM5_SPACE

UTL_CHANGE_CHAR_TO_SPACE
Description
Change the current character into a space character.

Usage
UTL_CHANGE_CHAR_TO_SPACE

Assumptions
LCDScrollBuffer should contain the message to be edited.

Input
None

Output
None

Destroys
None

Example
LCDScrollBuffer:
```
db   DM5_A, DM5_B DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
    db   DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
    db   DM5_SENTINEL
```

;setup for message editing
ld  B, #17
UTL_SETUP_VARS_FOR_EDITING

;display the section of message where character is to be edited
UTL_DISPLAY_MSG_AND_REQ_BLINK

;move cursor forward
UTL_MOVE_CURSOR_FORWARD_AND_REQ_BLINK

;change character under cursor to a space
UTL_CHANGE_CHAR_TO_SPACE

; result
; LCDScrollBuffer:
;    db   DM5_A, DM5_SPACE, DM5_C, DM5_D, DM5_E, DM5_F, DM5_G, DM5_H
;    db   DM5_I, DM5_J, DM5_K, DM5_L, DM5_M, DM5_N, DM5_O, DM5_P
;    db   DM5_SENTINEL

Time Structure Math
The following API uses the KRESBuffer as the default buffer where data to be added or subtracted from the user data structure of the same format.

<table>
<thead>
<tr>
<th>Offset</th>
<th>Variable Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>KRESHundBuffer</td>
</tr>
<tr>
<td>1</td>
<td>KRESSecBuffer</td>
</tr>
<tr>
<td>2</td>
<td>KRESMinBuffer</td>
</tr>
<tr>
<td>3</td>
<td>KRESHrBuffer</td>
</tr>
<tr>
<td>4</td>
<td>KRESHrRollOverValue</td>
</tr>
<tr>
<td>5</td>
<td>KRESHourRolloverExcess</td>
</tr>
</tbody>
</table>

The following are the bit definitions that indicate what data has been updated:
### Bit Position Variable Name

<table>
<thead>
<tr>
<th>Bit Position</th>
<th>Variable Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>00000001B</td>
<td>bKRESSecUpd</td>
</tr>
<tr>
<td>00000010B</td>
<td>bKRESMinUpd</td>
</tr>
<tr>
<td>00000100B</td>
<td>bKRESHrUpd</td>
</tr>
<tr>
<td>00001000B</td>
<td>bKRESDayUpd</td>
</tr>
</tbody>
</table>

### KRES_CLEARDATABUFFERS

**Description**
Clears the KRESHundBuffer, KRESSecBuffer, KRESMinBuffer, KRESHrBuffer and KRESHrRollOverValue.

**Usage**
KRES_CLEARDATABUFFERS

**Assumptions**
None

**Input**
None

**Output**
None

**Destroys**
HL

**Example**

`; clear all variables in the buffer prior to loading the required data for addition
KRES_CLEARDATABUFFERS

`; load new hour and minute
ld A, [NumberOfHoursToAdd]
ld [KRESHrBuffer], A
ld A, [NumberOfMinutesToAdd]
ld [KRESMinBuffer], A

`; setup hour rollover value
ld A, #24H
ld [KRESHrRollOverValue], A

`; add hour and minute
ld HL, #MyTimeStructure
KRES_MINHR_UPD

---

MyTimeStructure:

<table>
<thead>
<tr>
<th></th>
<th>ds</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>MyMinute</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MyHour</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### KRES_HUNDSECM NHR_UPD
**Description**
Updates the data structure HH:MM:SS.hh by adding the buffer HH:MM:SS.hh to the data structure.

**Usage**
KRES_HUNDSECMINHR_UPD

**Assumptions**
KRESBuffer is already loaded with the required data to add into the user time structure.

**Input**
- HL - Points to the hundredths data
- KRES_hrRollOverValue - Maximum hour rollover value

**Output**
- A - Update flags

**Destroys**
May destroy the contents of the following variables: KRES_hrBuffer, KRES_minBuffer, KRES_secBuffer, and KRES_hundBuffer.

**Example**
; clear all variables in the buffer prior to loading the required data for addition
KRES_CLEARBUFFERS

; load number of hundredths to add
ld A, [NumberOfHundredthsToAdd]
ld [KRES_hundBuffer], A

; setup hour rollover value
ld A, #24H
ld [KRES_hrRolloverValue], A

; add in the accumulated hundredths into the structure
ld HL, #MyHundred
KRES_HUNDSECMINHR_UPD

... 

MyTimeStructure:

MyHundred: ds 1
MySecond: ds 1
MyMinute: ds 1
MyHour: ds 1

---

**KRES_SECMINHR_UPD**

**Description**
Updates the HH:MM:SS structure by adding the buffer HH:MM:SS to the data structure.

**Usage**
KRES_SEC_MINHR_UPD

**Assumptions**
KRESBuffer is already loaded with the required data to add into the user time structure.

**Input**
- HL - Points to the seconds data
- KRES_hrRollOverValue - Maximum hour rollover value

**Output**
- A - Update flags

**Destroys**
May destroy the contents of the following variables: KRES_hrBuffer, KRES_minBuffer.
KRESSecBuffer and KRESHundBuffer.

**Example**

```assembly
; clear all variables in the buffer prior to loading the required data for addition
KRES_CLEARDATABUFFERS

; load number of hundredths to add
ld A, [NumberOfSecondsToAdd]
ed [KRESSecBuffer], A

; setup hour rollover value
ld A, #24H
ld [KRESHrRollOverValue], A

; add in the accumulated seconds into the structure
ld HL, #MySecond
KRES_SECMINHR_UPD

...

MyTimeStructure:

MyHundred: ds 1
MySecond: ds 1
MyMinute: ds 1
MyHour: ds 1
```

**KRES_MINHR_UPD**

**Description**

Updates the HH:MM structure by adding the buffer HH:MM to the data structure.

**Usage**

KRES_MINHR_UPD

**Assumptions**

KRESBuffer is already loaded with the required data to add into the user time structure.

**Input**

- HL             - Points to the minutes data
- KRESHrRollOverValue             - Maximum hour rollover value

**Output**

- A             - update flags
- KRESHourRolloverExcess             - Number of days to subtract resulting from API execution

**Destroys**

May destroy the contents of the following variables: KRESHrBuffer, KRESMinBufferm KRESSecBuffer and KRESHundBuffer.

**Example**

```assembly
; clear all variables in the buffer prior to loading the required data for addition
KRES_CLEARDATABUFFERS

; load number of minutes and hours to add
ld A, [NumberOfMinutesToAdd]
ed [KRESMinBuffer], A
ld A, [NumberOfHoursToAdd]
ed [KRESHrBuffer], A

; setup hour rollover value
```
ld A, #24H
ld [KRES_HrRollOverValue], A

; add in the accumulated minutes into the structure
ld HL, #MyMinute
KRES_MINHR_UPD

MyTimeStructure:
    MyHundred: ds 1
    MySecond: ds 1
    MyMinute: ds 1
    MyHour: ds 1

**KRES_HUNDSEC_MINHR_SUB**

**Description**
subtracts the HH:MM:SS.hh buffer from HH:MM:SS.hh data structure and store the result at data structure.

**Usage**
KRES_HUNDSEC_MINHR_SUB

**Assumptions**
KRESBuffer is already loaded with the required data to add into the user time structure.

**Input**
- HL - Points to the hundredths data
- KRES_HrRollOverValue - Maximum hour rollover value

**Output**
A - update flags

**Destroys**
May destroy the contents of the following variables: KRES_HrBuffer, KRES_MinBuffer, KRES_SecBuffer and KRES_HundBuffer.

**Example**
; clear all variables in the buffer prior to loading the required data for addition
KRES_CLEARDATABUFFERS

; load number of hundredths to subtract
ld A, [NumberOfHundredths]
ld [KRES_HundBuffer], A

; setup hour rollover value
ld A, #24H
ld [KRES_HrRollOverValue], A

; subtract the accumulated hundredths into the structure
ld HL, #MyHundred
KRES_HUNDSEC_MINHR_SUB

MyTimeStructure:
    MyHundred: ds 1
    MySecond: ds 1
**KRES_SEC_MH_SUB**

**Description**
subtracts the buffer HH:MM:SS from HH:MM:SS structure and store the result at data structure.

**Usage**
KRES_SEC&MH_SUB

**Assumptions**
KRESBuffer is already loaded with the required data to add into the user time structure.

**Input**
- HL - Points to the seconds data
- KRESHrRollOverValue - Maximum hour rollover value

**Output**
- A - update flags

**Destroys**
May destroy the contents of the following variables: KRESHrBuffer, KRESMinBufferm KRESSecBuffer and KRESHundBuffer.

**Example**
```
;clear all variables in the buffer prior to loading the required data for addition
KRES_CLEARDATABUFFERS

;load number of hundredths to subtract
ld  A, [NumberOfSeconds]
ld  [KRESSecBuffer], A

;setup hour rollover value
ld  A, #24H
ld  [KRESHrRollOverValue], A

;subtract the accumulated seconds into the structure
ld   HL, #MySecond
KRES_SEC&MH_SUB
```

```
MyTimeStructure:
    MyHundred:  ds  1
    MySecond:  ds  1
    MyMinute:  ds  1
    MyHour:  ds  1
```
**Usage**

KRES_MINHR_SUB

**Assumptions**

KRESBuffer is already loaded with the required data to add into the user time structure.

**Input**

- HL - Points to the minute data
- KRESHrRollOverValue - Maximum hour rollover value

**Output**

- A - Update flags
- KRESHourRolloverExcess - Number of days to subtract resulting from API execution

**Destroys**

May destroy the contents of the following variables: KRESHrBuffer, KRESMinBuffer, KRESSecBuffer and KRESHundBuffer.

**Example**

; clear all variables in the buffer prior to loading the required data for addition
KRES_CLEARDATABUFFERS

; load number of minutes and hours to subtract
ld A, [NumberOfMinutes]
ld [KRESMinBuffer], A
ld A, [NumberOfHours]
ld [KRESHrBuffer], A

; setup hour rollover value
ld A, #24H
ld [KRESHrRollOverValue], A

; subtract accumulated minute and hour data into the structure
ld HL, #MyMinute
KRES_MINHR_SUB

...

MyTimeStructure:

- MyHundred: ds 1
- MySecond: ds 1
- MyMinute: ds 1
- MyHour: ds 1
Trademarks
TIMEX is a registered trademark and service mark of Timex Corporation.
TIMEX DATA LINK and WristApp are trademarks of Timex Corporation in the U.S. and other countries.
Night-Mode is a registered trademark of Timex Corporation.
INDIGLO is a registered trademark of Indiglo Corporation.