**OTP Review**

# OTP 简介

## OTP框架图



## 数据空间分配

为了保证OTP的正确性，我们对于一些数据program 2次，两次里面写进去一次就算写成功，冗余设计的具体空间以及目前分配的空间如表格所示。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bank | Start | End | Size | Purpose |
| Bank0 | 0x0\_0000 | 0x0\_03FF | 1Kb | access right table for 35 regions, each region has 4 access right bits, 4x35=4.375x32= 5x32, read 10 times （1-10） |
| 0x0\_0000-0x0\_0003: Bank0 access right |
| 0x0\_0004-0x0\_0007: Bank1 access right |
| 0x0\_0008-0x0\_000b: Bank2 access right |
| 0x0\_000c-0x0\_000f: Bank3 access right |
| 0x0\_0010-0x0\_0013: Bank4 access right |
| 0x0\_0014-0x0\_0017: Bank5 access right |
| 0x0\_0018-0x0\_001b: Bank6 access right |
| 0x0\_001c-0x0\_001f:: Bank7 access right |
| 0x0\_0020-0x0\_0023: Bank8 access right |
| 0x0\_0024-0x0\_0027: Bank9 access right |
| 0x0\_0028-0x0\_002b: Bank10 access right |
| 0x0\_002c-0x0\_002f : Bank11 access right |
| 0x0\_0030-0x0\_0033: Bank12 access right |
| 0x0\_0034-0x0\_0037: Bank13 access right |
| 0x0\_0038-0x0\_003b: Bank14 access right |
| 0x0\_003c-0x0\_003f: Bank15 access right |
| 0x0\_0040-0x0\_0043: Bank16 access right |
| 0x0\_0044-0x0\_0047: Bank17 access right |
| 0x0\_0048-0x0\_004b: Bank18 access right |
| 0x0\_004c-0x0\_004f:: Bank19 access right |
| 0x0\_0050-0x0\_0053: Bank20 access right |
| 0x0\_0054-0x0\_0057: Bank21 access right |
| 0x0\_0058-0x0\_005b: Bank22 access right |
| 0x0\_005c-0x0\_005f: Bank23 access right |
| 0x0\_0060-0x0\_0063: Bank24 access right |
| 0x0\_0064-0x0\_0067: Bank25 access right |
| 0x0\_0068-0x0\_006b: Bank26 access right |
| 0x0\_006c-0x0\_006f: Bank27 access right |
| 0x0\_0070-0x0\_0073: Bank28 access right |
| 0x0\_0074-0x0\_0077: Bank29 access right |
| 0x0\_0078-0x0\_007b: Bank30 access right |
| 0x0\_007c-0x0\_007f: Bank31 access right |
| 0x0\_0080-0x0\_0083: Bank32 access right |
| 0x0\_0084-0x0\_0087: Bank33 access right |
| 0x0\_0088-0x0\_008b: Bank34 access right |
| 0x0\_00a0-0x0\_012b: redundancy of the above data from Bank0 to Bank34 access right |
|  |
| 0x0\_02e0-0x0\_02FF: each checksum tag is 1 bit, 32bits checksum tag |
| 0x0\_0300-0x0\_03FF: each checksum data is 8 bits , 256 bits checksum data |
| Bank1 | 0x0\_0400 | 0x0\_07FF | 1Kb | chip HW config (128)x2, read 8 times （33-40）, abc |
| 0x0\_0400-0x0\_047F: chip HW config |
| 0x0\_0400-0x0\_04FF: redundancy of the above chip HW config |
| 0x0\_0600-0x0\_07FF: root key |
| Bank2 | 0x0\_0800 | 0x0\_0bFF | 1Kb | tRoot HW config(128+128+32)x2, read 18 times （65-82） |
| 0x0\_0800-0x0\_091F: tRoot HW config |
| 0x0\_0920-0x0\_0a3F: redundancy of the above tRoot HW config |
| Bank3 | 0x0\_0c00 | 0x0\_0FFF | 1Kb | HDMI1.4 KEY 285B+2BCRC |
| Bank4 | 0x0\_1000 | 0x0\_13FF | 1Kb |
| Bank5 | 0x0\_1400 | 0x0\_17FF | 1Kb |
| Bank6 | 0x0\_1800 | 0x0\_1bFF | 1Kb | DP1.4 KEY 285B+2BCRC |
| Bank7 | 0x0\_1c00 | 0x0\_1FFF | 1Kb |
| Bank8 | 0x0\_2000 | 0x0\_23FF | 1Kb |
| Bank9 | 0x0\_2400 | 0x0\_27FF | 1Kb | DP2.2 KEY 387B+2BCRC |
| Bank10 | 0x0\_2800 | 0x0\_2bFF | 1Kb |
| Bank11 | 0x0\_2c00 | 0x0\_2FFF | 1Kb |
| Bank12 | 0x0\_3000 | 0x0\_33FF | 1Kb |
| Bank13 | 0x0\_3400 | 0x0\_37FF | 1Kb | 0x0\_3400-0x0\_3418: TS module 5x5bits 0x0\_3419-0x0\_341d: ABB 5bits |
| Bank14 | 0x0\_3800 | 0x0\_3bFF | 1Kb | reserved |
| Bank15 | 0x0\_3c00 | 0x0\_3FFF | 1Kb | reserved |
| Bank16 | 0x0\_4000 | 0x0\_43FF | 1Kb | reserved |
| Bank17 | 0x0\_4400 | 0x0\_47FF | 1Kb | reserved |
| Bank18 | 0x0\_4800 | 0x0\_4bFF | 1Kb | reserved |
| Bank19 | 0x0\_4c00 | 0x0\_4FFF | 1Kb | reserved |
| Bank20 | 0x0\_5000 | 0x0\_53FF | 1Kb | reserved |
| Bank21 | 0x0\_5400 | 0x0\_57FF | 1Kb | reserved |
| Bank22 | 0x0\_5800 | 0x0\_5bFF | 1Kb | reserved |
| Bank23 | 0x0\_5c00 | 0x0\_5FFF | 1Kb | reserved |
| Bank24 | 0x0\_6000 | 0x0\_63FF | 1Kb | reserved |
| Bank25 | 0x0\_6400 | 0x0\_67FF | 1Kb | reserved |
| Bank26 | 0x0\_6800 | 0x0\_6bFF | 1Kb | reserved |
| Bank27 | 0x0\_6c00 | 0x0\_6FFF | 1Kb | reserved |
| Bank28 | 0x0\_7000 | 0x0\_73FF | 1Kb | reserved |
| Bank29 | 0x0\_7400 | 0x0\_77FF | 1Kb | reserved |
| Bank30 | 0x0\_7800 | 0x0\_7bFF | 1Kb | reserved |
| Bank31 | 0x0\_7c00 | 0x0\_7FFF | 1Kb | reserved |
| Bank32 | 0x0\_8000 | 0x0\_FFFF | 32Kb | reserved |
| Bank33 | 0x1\_0000 | 0x1\_7FFF | 32Kb | reserved |
| Bank34 | 0x1\_8000 | 0x1\_FFFF | 32Kb | reserved |

# 二、OTP仿真环境

在徐卓的scompile环境下可以进行OTP的相关仿真。需要配置的相关内容是：

1. myplusargs : 添加

setenv MY\_PLUSARGS\_RUN “+KP\_OTP\_LOADMAIN”

# 三、OTP相关的SOC Flow环境

OTP是包在secure\_subsys里面的，所以环境都是基于secure\_subsys的。

1. DC



1. Formal Check

