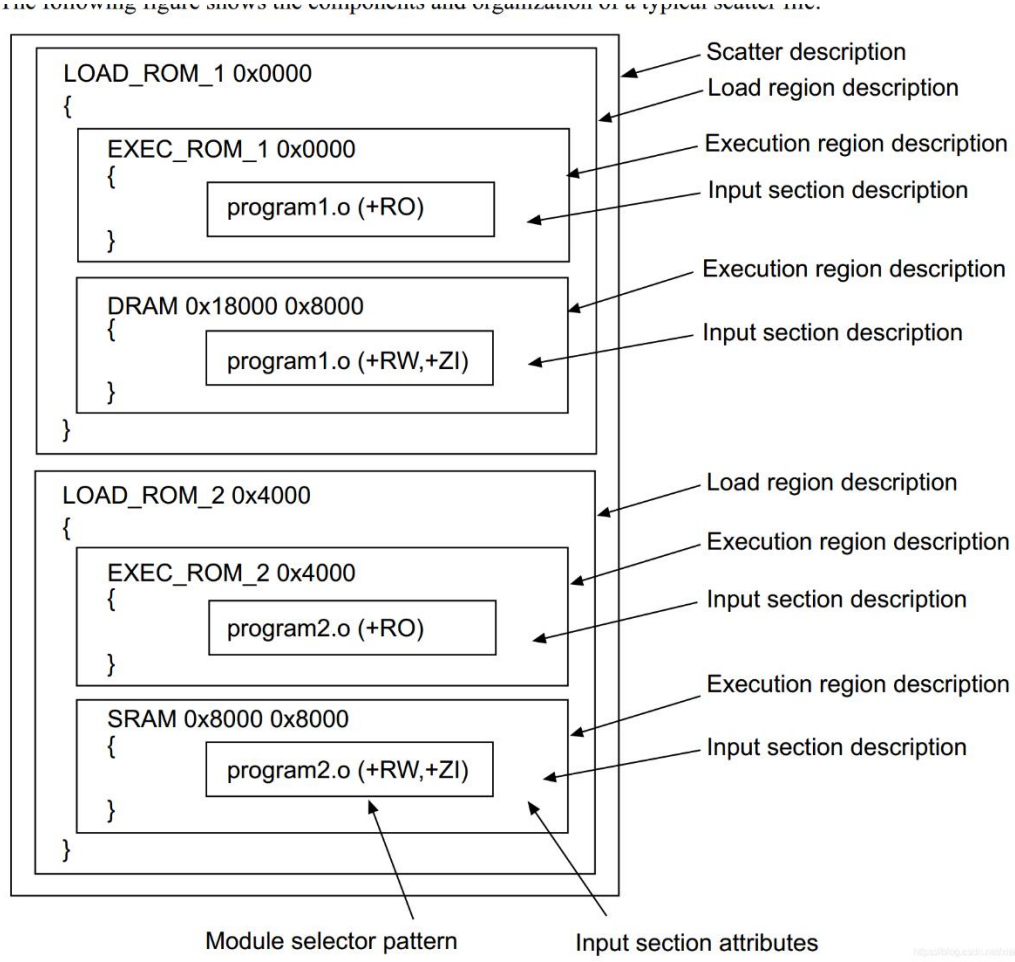
armlink 第二章 scatter 語法（一）

原創 安仔都有人用 2020-06-18 21:36

第二章 scatter 語法（一）基本結構

先來看一下，一個 scatter 文件的整體結構，如下圖：



接下來分別對其進行詳細說明。

2.1 加載 region 的描述信息

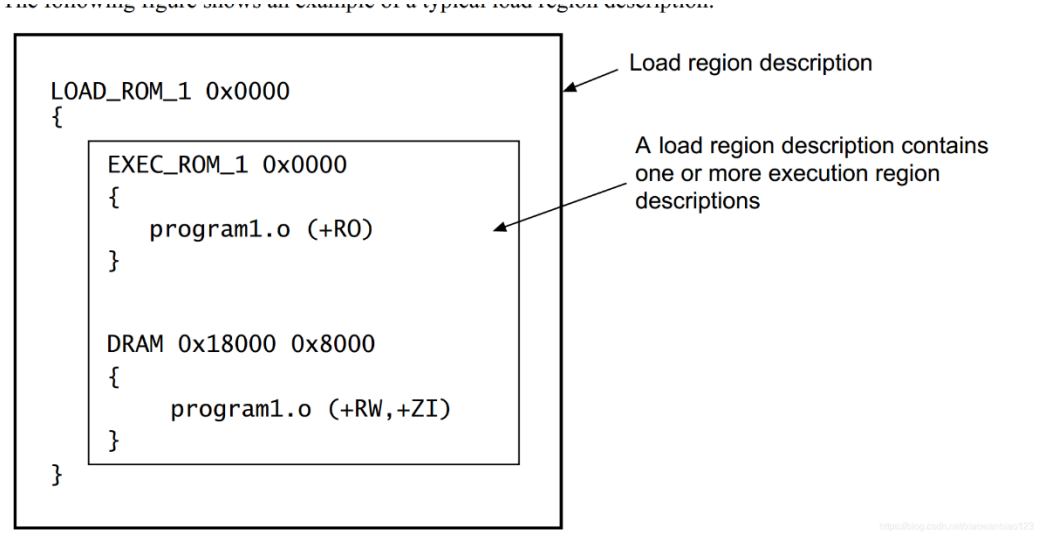
加載 region 的描述信息指出了它的孩子——執行 region——的放置情況

2.1.1 加載 region 的描述信息組成元素

加載 region 描述信息由下面組成：

1. 名字——被鏈接器用於標記唯一的加載 region
2. 基址——在這個 region 下的數據和代碼開始的地址
3. 屬性
4. 可選的最大長度
5. 一個或多個執行 regoin

如下圖



2.1.2 加載 region 的屬性

ABSOLUTE:

鏈接之後，內容被放置在一個固定地址。除非你使用 PI 或者 RELOC 屬性，否則這個屬性是默認值

ALIGN aligment:

將 4 字節對齊，增加到 aligment 字節對齊。aligment 必須是 2 的正數冪。如果加載 region 直接指定基址，那麼基址必須對齊；如果加載 region 使用+offset 的方式指定基址，那麼鏈接器自動計算以保證對齊。

NOCOMPRESS:

默認情況下 RW 數據被壓縮，該屬性使得 RW 數據內容不被壓縮。

OVERLAY：

該屬性使得在同一個地址，具有多個加載 region。ARM 工具集不會提供 overlay 機制。為了在同一個位置放置多個加載 region，你必須提供自己的 overlay 管理器。

PI:

標記該 region 是地址無關的。內容不會依賴於任何固定的地址。

PROTECTED：

該屬性阻止：

1. 加載 region 的重疊
2. venner 的共享
3. 使用–merge 選項共享字符串

注意：venner，指的是鏈接器生成的一小段代碼，這段代碼可能用於長距離的跳轉等。

RELOC:

標記該 region 是可重定位的。重定位信息會被保存，使其能夠被其他的工具移動到其他位置，已達到可重定位的功效

2.1.3 加載 region 的地址屬性繼承規則

加載 region 可以繼承上一個加載 region 的屬性。為了達到此目的，只需要使用+offset 的方式設置基址即可。如下情況下無法繼承屬性：

1. 顯示的設置了加載 region 的屬性
2. 前一個加載 region 具有 OVERLAY 屬性

你可以顯式的使用 ABSOLUTE,PI,RELOC,OVERLAY 指定地址屬性,當地址屬性沒有被指定時，下面的繼承規則將會被應用：

1. OVERLAY 屬性不會被繼承
2. 加載和執行 region 的基址默認為 ABSOLUTE

3. +offset 會繼承上一個 region 的地址屬性，如果沒有上一個 region，則為 ABSOLUTE 屬性。

例如：

```
LR1 0x8000 PI
{
    …
}
LR2 +0 ; LR2 從 LR1 繼承 PI 屬性
{
    …
}
LR3 0x1000 ; LR3 不會繼承因為它沒有相對基址，所以為默認屬性 ABSOLUTE
{
    …
}
LR4 +0 ; LR4 從 LR3 繼承 ABSOLUTE
{
    …
}
LR5 +0 RELOC ; LR5 不會繼承，因為它顯式設置了 RELOC
{
    …
}
LR6 +0 OVERLAY ; LR6 不會繼承，因為有 OVERLAY
{
    …
}
LR7 +0 ; LR7 不會繼承 OVERLAY，因此是默認屬性 ABSOLUTE
{
    …
}
```

2.1.4 RELOC 地址屬性的繼承規則

你可以給加載 region 設置一個 RELOC 屬性。但是，執行 region 僅可以從父加載 region 中繼承 RELOC。

注意：對於 BP 鏈接模型，如果加載 region 有 RELOC 屬性，那麼所有的在這個加載 region 內的執行 region 必須是+offset 方式來指定基址。這個保證了執行 region 繼承父加載 region 的重定位信息。

例如：

```
LR1 0x8000 RELOC
{
    ER1 +0 ; 從 LR1 中繼承 RELOC 屬性
    {
        …
    }
    ER2 +0 ; 從 ER1 中繼承 RELOC 屬性
    {
        …
    }
    ER3 +0 RELOC ; 錯誤,不能在執行 region 中顯式指定 RELOC 屬性
    {
        …
    }
}
```

2.2 執行 region 的描述信息

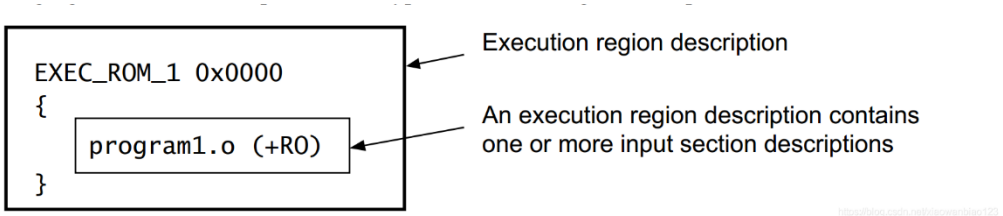
這部分指定了 region 在運行時處在內存的什麼位置。

2.2.1 執行 region 的描述信息組成元素

由下面的元素組成：

1. 名字——被鏈接器標識，用於區分不同的執行 region
2. 基址——不是絕對值就是相對值
3. 屬性
4. 可選的最大長度
5. 一個或多個輸入 section 的模式字符串

如下圖：



2.2.2 執行 region 的屬性

ABSOLUTE:

鏈接之後，內容被放置於一個固定的地址。

ALIGN aligment:

將 4 字節對齊提升到 aligment 字節對齊。aligment 必須是 2 的正數冪。如果執行 region 被設置了基址，那麼他必須按照 aligment 字節對齊。如果執行 region 被設置了偏移，那麼鏈接器自動計算地址，以達到 aligment 字節對齊。

ALIGNALL value:

增加執行 region 中各 section 的對齊為 value 字節對齊。value 的值必須是 2 的正數冪，並且必須大於等於 4

ANY_SIZE max_size:

指定，該 region 可以放置的未分配 section（即.ANY 匹配的 section）的最大大小。

例如：如果一個執行 region，將僅被.ANY 匹配的 section 填充，那麼 2%將用於 venner。剩下的 98%纔會用於.ANY 匹配的 section

使用這個關鍵字時，注意下面的限制：

1. max_size 必須小於或者等於 region 大小
2. 可以在沒有.ANY 選擇器的 region 上使用 ANY_SIZE.但是，他會被 armlink 忽略。

EMPYT [-]length:

在內存中保留一個給定大小的空的區域。通常被用作堆或棧區域。在帶有 EMPTY 屬性的 region 中，不能放置任何 section。

length 為正數，則基址為起始地址。length 為負數，則基址為結束地址。

FILL value:

鏈接器創建一個 region，該 region 填充 value。如果你指定了 FILL，那麼就必須跟一個 value。例如 FILL 0xFFFFFFFF.他相當於 EMPTY ZEROPAD PADVALUE

FIXED：

固定地址，鏈接器嘗試使執行地址和加載地址相同。如果無法相同將報錯。

NOCOMPRESS：

armlink 默認是打開 RW 數據壓縮的。該屬性使得 RW 數據不被壓縮。

OVERLAY:

指示該 region 可能會有重疊的 section。

PADVALUE:

定義填充的值。PADVALUE 必須跟一個值，例如：

```
EXEC 0x10000 PADVALUE 0xffffffff EMPTY ZEROPAD 0x2000
```

這將創建一個大小 0x2000,填滿 0xffffffff 的 region

PADVALUE 屬性的值必須是字大小，且被加載 region 所忽略。

PI：

該 region 僅包含位置無關的 section。

注意：該屬性在包含 XO section 時，不支持

SORTTYPE:

指定執行 region 的排序算法，例如：

```
ER1 +0 SORTTYPE CallTree
```

UNINIT:

用於創建包含未初始化數據或者內存映射 IO 的 region。

ZEROPAD:

零初始化的 section 被寫入 ELF 中，因此不用在運行時填充零

該屬性設置 ZI 輸出 section 的長度，並保存在
Image$$region_name$$ZI$$Length 中。

注意：Image$$region_name$$ZI$$Length 是鏈接器使用的一些符號。因為本系列文章的主旨是 scatter 文件。因此不再對鏈接器的符號做進一步說明。

只有 root region 可以使用這個屬性。否則將產生一個警告，並忽略。

2.2.3 執行 region 的地址屬性的繼承規則

執行 region 可以繼承上一個執行 region 的屬性。

為了繼承，直接使用+offset 的形式來指定基址。第一個執行 region 繼承其父加載 region 的屬性。

執行 region 在下面情況下無法繼承屬性：

1. 顯式設置了執行 region 的屬性
2. 上一個執行 region 具有 OVERLAY 屬性

你可以使用 ABSOLUTE,PI,OVERLAY 屬性，顯式的設置執行 region。但是，執行 region 僅繼承父加載 region 的 RELOC 屬性。

當沒有地址屬性被設置時，下面的繼承規則會發生：

1. OVERLAY 屬性不會被繼承，含有 OVERLAY 屬性的 region 也不會繼承
2. 加載或者執行 region 的基址默認是 ABSOLUTE
3. 通過+offset 設置的地址，將會繼承上一個執行 region 的地址屬性，如果沒有上一個，則繼承父加載 region 的地址屬性

例如：

```
LR1 0x8000 PI
```

```
{
    ER1 +0 ; ERI 從 LR1 中繼承 PI
    {
        …
    }
    ER2 +0 ; ER2 從 ER1 中繼承 PI
    {
        …
    }
    ER3 0x10000 ; ER3 沒有繼承，因為他沒有使用相對基址，因此它使用了默認屬性 ABSOLUTE
    {
        …
    }
    ER4 +0 ; ER4 從 ER3 中繼承 ABSOLUTE
    {
        …
    }
    ER5 +0 PI ; ER5 不會繼承，因為他顯式設置了 PI
    {
        …
    }
    ER6 +0 OVERLAY ; ER6 不會繼承，因為他設置了 OVERLAY
    {
        …
    }
    ER7 +0 ; ER7 不會繼承 OVERLAY，因此使用默認屬性 ABSOLUTE
    {
        …
    }
}
```

2.2.4 相對地址用在加載 region 上的情況

當相對地址用在加載 region 上時，需要注意如下情況：

1. 如果使用+offset 的加載 region（LR2）跟在一個含有 ZI 數據的加載 region(LR1)之後。那麼 LR2 將覆蓋 ZI 數據。為了修復這個，使用 ImageLimit()函數來指定 LR2 的基址。

注意：這在下一章中有例子

2. LR2 繼承 LR1 的屬性，除非如下情況：

- LR1 有 OVERLAY 屬性
- LR2 顯式設置了屬性

如果一個加載 region 無法繼承屬性，那麼它將使用默認值 ABSOLUTE。

3. 在 ROM 鏡像中，如果一個 region 含有 RW 數據壓縮，且下一個 region 使用+offset 設置基址，則在這兩個 region 之間，可能存在一個空隙。這是因為鏈接器計算相對基址時，是相對於未壓縮數據的。但是這個空隙會在運行時消失。因為運行時會解壓出來。

2.2.5 相對地址用在執行 region 上的情況

當相對地址用在執行 region 上時，需要注意一下情況：

1. 除非執行 region 被顯式的設置屬性，否則，第一個執行 region 繼承父加載 region 的屬性。
2. 使用相對地址的執行 region（ER2）繼承執行 region（ER1）的屬性，除非如下情況：

- ER1 有 OVERLAY 屬性
- ER2 有一個顯式的屬性設置

如果一個執行 region 無法繼承屬性，那麼將使用默認值 ABSOLUTE

3. 如果父加載 region 有 RELOC 屬性，那麼在其內的所有的執行 region 必須使用相對地址。

2.3 輸入 section 的描述信息

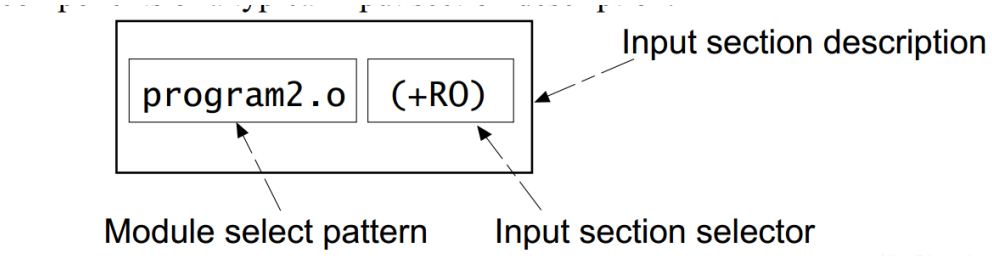
輸入 section 的描述信息是一個模式字符串，用於選擇 section

2.3.1 輸入 section 的描述信息組成元素

由如下部分組成：

1. 名字，可以使用通配符——中間文件名，庫成員名，或者庫名字
2. 輸入 sectoin 名，或者輸入 section 屬性，例如 READ-ONLY,或者 CODE。輸入 section 名也可以使用通配符
3. 符號名

如下圖：



2.3.2 模塊和輸入 section 模式字符串舉例

模塊匹配舉例：

- * 匹配任何模塊和庫
- *.o 匹配任何以 o 結尾的模塊
- math.o 匹配 math.o 模塊
- *armlib* 匹配所有 ARM 支持的 C 庫
- “file1.o” 匹配 file1.o
- *math.lib 匹配所有以 math.lib 結尾的庫。例如：c:\apps\lib\math\satmath.lib

輸入 section 匹配舉例：

- +RO 匹配所有 RO 代碼和 RO 數據
- +RW,+ZI 匹配所有的 RW 代碼，所有的 RW 數據，所有 ZI 數據
- BLOCK_42 匹配名字叫做 BLOCK_42 的 section。這個名字可能有多個 section，但是他們的屬性可能不同。

本章完，下章 scatter 文件中使用的表達式和函數