

下面是对 `enum class AS_stmkind` 中各个指令类型的详细介绍，包括它们的参数作用和打印出来的样式：

AS_stmkind 枚举类

1. BINOP

- **描述:** 二元操作指令，用于执行加法、减法、乘法、除法等操作。乘法和除法两个操作数都必须是寄存器。
- **参数:**
 - `op`: 操作类型（如加法、减法等）
 - `dst`: 目标寄存器
 - `left`: 左操作数寄存器
 - `right`: 右操作数寄存器
- **示例:**

```
1 // add x1, x2, x3
2 add    x1, x2, x3
```

2. MOV

- **描述:** 数据移动指令，将数据从源寄存器移动到目标寄存器。
- **参数:**
 - `dst`: 目标寄存器
 - `src`: 源寄存器
- **示例:**

```
1 // mov x1, x2
2 mov    x1, x2
```

3. MOVZ

- **描述:** 将立即数加载到目标寄存器高位。
- **参数:**
 - `dst`: 目标寄存器
 - `src`: 立即数
- **示例:**

```
1 // movz x1, #1234, LSL #16
2 movz    x1, #1234, LSL #16
```

4. MOVK

- **描述:** 将立即数加载到目标寄存器的低位，不改变高位。
- **参数:**
 - `dst`: 目标寄存器
 - `src`: 立即数
- **示例:**

```
1 // movk x1, #5678, LSL #0
2 movk     x1, #5678, LSL #0
```

5. LDR

- **描述:** 从内存加载数据到目标寄存器。
- **参数:**
 - `dst`: 目标寄存器
 - `ptr`: 内存地址指针
 - `post_index`: 后置索引。为0时不设置
- **示例:**

```
1 // ldr x1, [sp], #8
2 ldr     x1, [sp], #8 //后索引 (Post-index) 寻址模式
3 ldr     x1, [sp, #8] //偏移量寻址模式
4 ldr     x1, [x29]    //基址寄存器寻址模式
5
```

6. LDP

- **描述:** 从内存加载一对寄存器。
- **参数:**
 - `dst1`: 第一个目标寄存器
 - `dst2`: 第二个目标寄存器
 - `ptr`: 内存地址指针
 - `post_index`: 后置索引。
- **示例:**

```
1 // ldp x1, x2, [sp], #16
2 ldp     x1, x2, [sp], #16
```

7. STR

- **描述:** 将数据从源寄存器存储到内存。
- **参数:**
 - `src`: 源寄存器
 - `ptr`: 内存地址指针
 - `pre_index`: 前置索引。为0时不设置
- **示例:**

```
1 // str x1, [sp, #8]!  
2 str    x1, [sp, #8]!
```

8. STP

- **描述:** 将一对寄存器的数据存储在内存。
- **参数:**
 - `src1`: 第一个源寄存器
 - `src2`: 第二个源寄存器
 - `ptr`: 内存地址指针
 - `pre_index`: 前置索引
- **示例:**

```
1 // stp x1, x2, [sp, #16]!  
2 stp    x1, x2, [sp, #16]!
```

9. LABEL

- **描述:** 定义一个标签，用于代码的跳转。
- **参数:**
 - `name`: 标签名称
- **示例:**

```
1 // label_name:  
2 label_name:
```

10. B

- **描述:** 无条件跳转到指定的标签。
- **参数:**
 - `jump`: 跳转的标签
- **示例:**

```
1 // b label_name
2 b      label_name
```

11. BCOND

- 描述: 条件跳转到指定的标签。
- 参数:
 - `op`: 条件操作码 (如等于、不等于等)
 - `jump`: 跳转的标签
- 示例:

```
1 // b.eq label_name
2 b.eq      label_name
```

12. BL

- 描述: 调用子程序。
- 参数:
 - `jump`: 跳转的标签
- 示例:

```
1 // bl label_name
2 bl      label_name
```

13. CMP

- 描述: 比较两个寄存器的值。
- 参数:
 - `left`: 左操作数寄存器
 - `right`: 右操作数寄存器
- 示例:

```
1 // cmp x1, x2
2 cmp    x1, x2
```

14. RET

- 描述: 返回到调用者。
- 参数: 无
- 示例:

```
1 // ret
2 ret
```

15. ADR

- **描述:** 地址计算指令，计算标签（如全局变量）的地址并存储在目标寄存器中。
- **参数:**
 - `reg`: 目标寄存器
 - `label`: 标签
- **示例:**

```
1 // adrp x1, label_name
2 adrp    x1, label_name
3 // add x1, x1, #:lo12:label_name
4 add     x1, x1, #:lo12:label_name
```

16. LLVMIR

- **描述:** 内嵌LLVM中间表示（IR）代码。
- **参数:**
 - `llvmir`: LLVM IR代码
- **示例:**

```
1 // ; <LLVM IR code>
2 // llvmir code here
```