

1 Assembly Listing with NASM Syntax

Listing 1: Assembly listing with NASM syntax using the custom language `nasm/lang` and the custom style `nasm/style`.

```
1 extern printf
2 extern exit
3 global _start
4
5 [section .text]
6 _start:
7     mov     rdi, hello ; arg1: "hello world\n"
8     call   printf     ; printf("hello world\n")
9     mov     rdi, 0    ; arg1: 0
10    call   exit       ; exit(0)
11
12 [section .rodata]
13 hello:
14     db     'hello world', 10, 0
```

2 C Listing

Listing 2: C listing using the default C language and the custom style `c/style`.

```
1 #include <stdio.h>
2
3 // main prints "hello world" to standard output.
4 int main(int argc, char **argv) {
5     printf("hello world\n");
6     return 0;
7 }
```

3 Go Listing

Listing 3: Go listing using the custom language go/lang and the custom style go/style.

```
1 // fib is a command which prints the Fibonacci sequence.
2 package main
3
4 import "fmt"
5
6 func main() {
7     c := make(chan int)
8     go fib(c)
9     fmt.Println("The Fibonacci sequence:")
10    for x := range c {
11        fmt.Println(x)
12    }
13
14 }
15
16 func fib(c chan<- int) {
17     a, b := 0, 1
18     for a >= 0 {
19         c <- a
20         a, b = b, a+b
21     }
22     close(c)
23 }
```

4 REIL Instructions Listing

Listing 4: REIL instructions listing using the custom language `reil/lang` and the custom style `nasm/style`.

```
1 bisz t8, , ZF
2 str t8, , esp
3 ldm 16815620, , t0
4 str t0, , eax
5 sub esp, 4, esp
6 and esp, 4294967295, esp
7 stm esi, , esp
8 sub esp, 4, esp
9 and esp, 4294967295, esp
10 stm t2, , esp
11 add -4, ebp, t0
12 and t0, 4294967295, t1
13 stm eax, , t1
14 sub esp, 4, t0
15 and t0, 4294967295, esp
16 stm 16805479, , esp
17 jcc 1, , 16805367
```

5 LLVM IR Listing

Listing 5: LLVM IR listing using the custom language `llvm/lang` and the custom style `nasm/style`.

```
1 target triple = "x86_64-unknown-linux-gnu"
2
3 @s = private constant [13 x i8] c"hello world\0A\00"
4
5 ; main prints "hello world" to standard output.
6 define i32 @main(i32 %argc, i8** %argv) {
7     %1 = getelementptr [13 x i8]* @s, i32 0, i32 0
8     call i32 (i8*, ...)* @printf(i8* %1)
9     ret i32 0
10 }
11
12 declare i32 @printf(i8*, ...)
```

6 Diff Listing

Listing 6: Diff listing using the custom language `diff/lang` and the custom style `diff/style`.

```
1 diff --git a/lib/IR/AsmWriter.cpp b/lib/IR/AsmWriter.cpp
2 index c494d6c..1a96956 100644
3 --- a/lib/IR/AsmWriter.cpp
4 +++ b/lib/IR/AsmWriter.cpp
5 @@ -2025,13 +2025,14 @@ void AssemblyWriter::printBasicBlock(const ←
   BasicBlock *BB) {
6     Out << "\n";
7     PrintLLVMName(Out, BB->getName(), LabelPrefix);
8     Out << ':';
9 - } else if (!BB->use_empty()) {          // Don't print block # of no ←
   uses...
10 -     Out << "\n; <label>:";
11 + } else {          // Print block # for unnamed basic blocks.
12 +     Out << "\n";
13     int Slot = Machine.getLocalSlot(BB);
14     if (Slot != -1)
15         Out << Slot;
16     else
17         Out << "<badref>";
18 +     Out << ':';
19 }
20
21 if (!BB->getParent()) {
```

7 Dot Listing

Listing 7: Dot listing using the custom language `dot/lang` and the custom style `dot/style`.

```
1 digraph if {  
2   A->B [label="true"];  
3   A->C [label="false"];  
4   B->C  
5   A [label="entry"]  
6   B  
7   C [label="exit"];  
8 }
```
