

1 main — MIR Walkthrough

Purpose: TODO: Describe why this walkthrough exists

1.1 Source Context

```
fn main() {  
    assert!(-128_i8 << 1 == 0);  
    assert!(-32768_i16 << 1 == 0);  
    assert!(-2147483648_i32 << 1 == 0);  
    assert!(-9223372036854775808_i64 << 1 == 0);  
    assert!(-170141183460469231731687303715884105728_i128 << 1 == 0);  
}
```

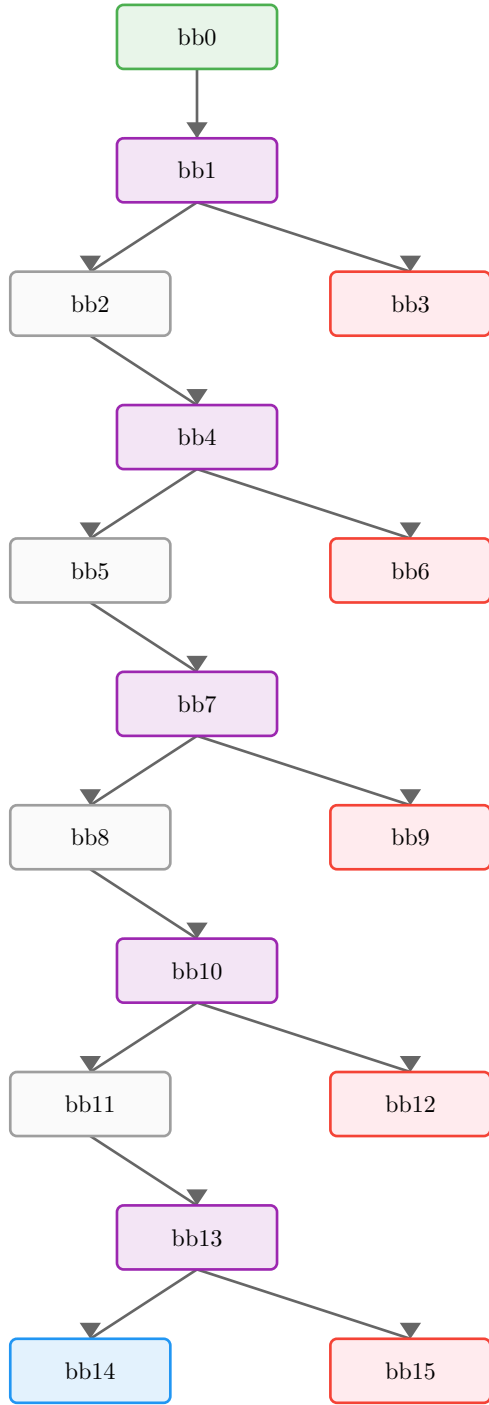
1.2 Function Overview

- **Function:** main
- **Basic blocks:** 16
- **Return type:** ()
- **Notable properties:**
 - Contains panic path
 - Contains assertions
 - Has conditional branches

1.3 Locals

Local	Type	Notes
0	()	Return place
1	i8	
2	u32	
3	bool	
4	!	
5	i16	
6	u32	
7	bool	
8	!	
9	i32	
10	u32	
11	bool	
12	!	
13	i64	
14	u32	
15	bool	
16	!	
17	i128	
18	u32	
19	bool	
20	!	

1.4 Control-Flow Overview



1.5 Basic Blocks

1.5.1 bb0 — entry

Entry point of the function.

MIR	Annotation
<code>_2 = 1 as RigidTy(Uint(U32))</code>	Integer conversion
<code>_3 = move _2 < 8</code>	Less than operation
<code>→ assert(move _3 == true) → bb1</code>	Panic if move <code>_3</code> is false

1.5.2 bb1 — branch point

MIR	Annotation
_1 = 128 \<\< 1	Shift left operation
→ switch(move _1) \[0→bb2; else→bb3\]	Branch on move _1

1.5.3 bb2

MIR	Annotation
_6 = 1 as RigidTy(Uint(U32))	Integer conversion
_7 = move _6 \< 16	Less than operation
→ assert(move _7 == true) → bb4	Panic if move _7 is false

1.5.4 bb3 — panic path

Panic/diverging path.

MIR	Annotation
→ _4 = panic(\[16 bytes\])	Call panic

1.5.5 bb4 — branch point

MIR	Annotation
_5 = -32768 \<\< 1	Shift left operation
→ switch(move _5) \[0→bb5; else→bb6\]	Branch on move _5

1.5.6 bb5

MIR	Annotation
_10 = 1 as RigidTy(Uint(U32))	Integer conversion
_11 = move _10 \< 32	Less than operation
→ assert(move _11 == true) → bb7	Panic if move _11 is false

1.5.7 bb6 — panic path

Panic/diverging path.

MIR	Annotation
→ _8 = panic(\[16 bytes\])	Call panic

1.5.8 bb7 — branch point

MIR	Annotation
_9 = -2147483648 \<\< 1	Shift left operation
→ switch(move _9) \[0→bb8; else→bb9\]	Branch on move _9

1.5.9 bb8

MIR	Annotation
_14 = 1 as RigidTy(Uint(U32))	Integer conversion
_15 = move _14 \< 64	Less than operation
→ assert(move _15 == true) → bb10	Panic if move _15 is false

1.5.10 bb9 — panic path

Panic/diverging path.

MIR	Annotation
→ <code>_12 = panic([16 bytes])</code>	Call panic

1.5.11 bb10 — branch point

MIR	Annotation
<code>_13 = -9223372036854775808 << 1</code>	Shift left operation
→ <code>switch(move _13) {0→bb11; else→bb12}</code>	Branch on move __13

1.5.12 bb11

MIR	Annotation
<code>_18 = 1 as RigidTy(Uint(U32))</code>	Integer conversion
<code>_19 = move _18 < 128</code>	Less than operation
→ <code>assert(move _19 == true) → bb13</code>	Panic if move __19 is false

1.5.13 bb12 — panic path

Panic/diverging path.

MIR	Annotation
→ <code>_16 = panic([16 bytes])</code>	Call panic

1.5.14 bb13 — branch point

MIR	Annotation
<code>_17 = [16 bytes] << 1</code>	Shift left operation
→ <code>switch(move _17) {[0→bb14; else→bb15]}</code>	Branch on move __17

1.5.15 bb14 — return / success

Normal return path.

MIR	Annotation
→ <code>return</code>	Return from function

1.5.16 bb15 — panic path

Panic/diverging path.

MIR	Annotation
→ <code>_20 = panic([16 bytes])</code>	Call panic

1.6 Key Observations

TODO: Add bullet points summarizing what this MIR teaches

-
-

1.7 Takeaways

TODO: One or two sentences to generalize this example

