

1 main — MIR Walkthrough

Purpose: TODO: Describe why this walkthrough exists

1.1 Source Context

```
fn main() {  
    assert!(42 % 10 == 2);  
}
```

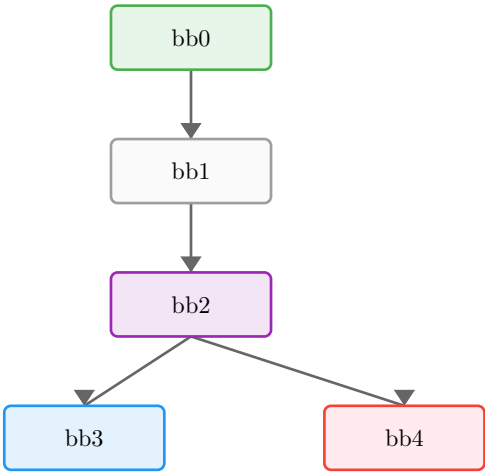
1.2 Function Overview

- **Function:** main
- **Basic blocks:** 5
- **Return type:** () (0 bytes, align 1)
- **Notable properties:**
 - Contains panic path
 - Contains assertions
 - Has conditional branches

1.3 Locals

Local	Type	Notes
0	() (0 bytes, align 1)	Return place
1	Int(I32)	
2	Bool	
3	Bool	
4	Bool	
5	Bool	
6	()	

1.4 Control-Flow Overview



1.5 Basic Blocks

1.5.1 bb0 — entry

Entry point of the function.

MIR	Annotation
-----	------------

<code>_2 = 10 == 0</code>	Equal operation
<code>→ assert(move _2 == false) → bb1</code>	Panic if move _2 is true

1.5.2 bb1

MIR	Annotation
<code>_3 = 10 == -1</code>	Equal operation
<code>_4 = 42 == -2147483648</code>	Equal operation
<code>_5 = move _3 & move _4</code>	AND operation
<code>→ assert(move _5 == false) → bb2</code>	Panic if move _5 is true

1.5.3 bb2 — branch point

MIR	Annotation
<code>_1 = 42 % 10</code>	Remainder operation
<code>→ switch(move _1) [2→bb3; else→bb4]</code>	Branch on move _1

1.5.4 bb3 — return / success

Normal return path.

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

1.5.5 bb4 — panic path

Panic/diverging path.

MIR	Annotation
<code>→ _6 = 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

