Bug Analysis by Problem (6-87)

Problem	Category	Pattern / Specific Error	Freq.
P6	3. Omission of Critical Instructions	Fails to calculate the SIMD loop's iteration count.	1
P7	4. Incorrect Instruction-Level Semantics	Pattern 4.3: Incorrect Instruction Ordering (resets variable before storing it).	1
P8	1. Incorrect Register State Management	Pattern 1.1: Premature Update (increments index before using it for a store).	1
P9	3. Omission of Critical Instructions	1. Fails to initialize SIMD product vectors. 2. Omits the final SIMD reduction step.	2
	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering (overwrites sum vectors during multiplication).	1
P10	2. Flawed Control Flow and Logic	Pattern 2.2: Premature/Incorrect Return (returns an integer instead of a pointer via a bad branch).	1
P15	4. Incorrect Instruction-Level Semantics	Pattern 4.1: Invalid Addressing Mode (uses [x0, x0]).	1
	5. Literal Translation Artifacts (x86-isms)	Mimics a complex shl/sar trick for multiplication instead of using a single lsl/sbfiz.	1
P18	1. Incorrect Register State Management	Pattern 1.3: Failure to Propagate Return Values (omits mov x21, x0 after realloc).	1
	2. Flawed Control Flow and Logic	Pattern 2.1: Incorrect Branch Target (branches to a loop instead of the error handler).	1
	4. Incorrect Instruction-Level Semantics	Pattern 4.2: Incorrect Immediate Value (1. Wrong stack offset. 2. Wrong constant for a note).	2
P19	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering (overwrites the main loop counter with an input value).	1
P20	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering (overwrites original character with lowercase version before store).	1
	4. Incorrect Instruction-Level Semantics	Pattern 4.2: Incorrect Immediate Value (uses wrong ASCII values for comparisons).	1
P21	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering (min_diff register s2 is overwritten inside the loop).	1
	4. Incorrect Instruction-Level Semantics	Pattern 4.2: Incorrect Immediate Value (fails to move FLT_MAX into a float register).	1
P22	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering (Calculates min/max into s4/s3 but later code incorrectly reads from s0/s1).	1
P23	3. Omission of Critical Instructions	Fails to advance the main string pointer (mov x20, x9) after parsing a number, causing an infinite loop.	1

Problem	Category	Pattern / Specific Error	Freq.
	1. Incorrect Register State Management	Fails to use the return value from strtol (in w0), storing a garbage character from w8 instead.	1
P27	1. Incorrect Register State Management	1. Pattern 1.2: Register Clobbering (inner loop counter corrupts outer loop's index). 2. Stores a loop counter instead of an array element.	2
	3. Omission of Critical Instructions	Fails to initialize a separate counter for an inner loop.	1
P28	7. Misinterpretation of Algorithm's Goal	Grossly incorrect SIMD implementation (unrolls to 64 bytes and botches logic).	1
	3. Omission of Critical Instructions	Completely omits the logic for swapping uppercase to lowercase in the scalar path.	1
P29	1. Incorrect Register State Management	Fails to preserve the array pointer for a second pass, consuming it in the first loop.	1
P33	1. Incorrect Register State Management	1. Fails to update the main state variable (d0). 2. Calculates powers of the wrong variable.	2
	2. Flawed Control Flow and Logic	Incorrect branching; falls through into hallucinated code instead of looping correctly.	1
	9. Code Hallucination	Generates several blocks of nonsensical, spurious code.	1
P35	1. Incorrect Register State Management	Initializes an inner loop counter with the array element's value instead of the count of items to check.	1
	3. Omission of Critical Instructions	Completely omits the function epilogue (updating out_count, restoring registers, ret).	1
P36	1. Incorrect Register State Management	Resets the max_val register back to its initial small value in every loop iteration.	1
P37	7. Misinterpretation of Algorithm's Goal	Fails to translate compound conditional logic, incorrectly merging two independent checks with ccmp.	1
	3. Omission of Critical Instructions	Omits a key mul instruction needed for a conditional check.	1
P38	6. Incorrect Loop Pointer/Index Management	Fails to advance the main SIMD source pointer, causing an infinite loop on the same data.	1
	3. Omission of Critical Instructions	Completely omits the final merging loop that constructs the output array.	1
	2. Flawed Control Flow and Logic	Pattern 2.1: Incorrect Branch Target (branches out of an inner sort loop prematurely).	1
P40	1. Incorrect Register State Management	Operates on a copy of the main state variable (n), which is never updated inside the inner loop.	1

Problem	Category	Pattern / Specific Error	Freq.
	2. Flawed Control Flow and Logic	Unconditional branch in a nested loop creates an infinite loop.	1
P41	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering (Completely corrupts all loop iterators and flags at setup).	1
	7. Misinterpretation of Algorithm's Goal	Fails to understand the j+k < size loop condition, checking k < size instead.	1
	2. Flawed Control Flow and Logic	Pattern 2.2: Premature/Incorrect Return (exits immediately on first match instead of continuing search).	1
P44	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering (Completely corrupts loop iterators and flags at setup, similar to P41).	1
	2. Flawed Control Flow and Logic	Pattern 2.2: Premature/Incorrect Return (exits immediately on first match).	1
P45	6. Incorrect Loop Pointer/Index Management	Uses a stagnant (never incremented) write pointer, overwriting out_str[0] in every loop iteration.	1
	8. Failure to Generate Idiomatic/Optimized Code	Fails to generate the optimized SIMD path (rev64.8b) for string reversal.	1
P47	6. Incorrect Loop Pointer/Index Management	Instruction Reordering: The relative order of ldr and pointer updates is wrong, causing the recurrence relation to use stale data.	1
P50	1. Incorrect Register State Management	Pattern 1.4: Incorrect Source/Destination in Update: Fails to use the running result as input for the next iteration.	1
P51	9. Code Hallucination	Generates a nonsensical, 500+ instruction block instead of a simple SIMD loop.	1
	3. Omission of Critical Instructions	Completely omits the scalar fallback path for short strings.	1
P60	1. Incorrect Register State Management	Operates on a temporary copy of n inside the "divide out" loop, so n is never updated.	1
	2. Flawed Control Flow and Logic	Unconditional branch in the "divide out" loop creates an infinite loop.	1
P63	4. Incorrect Instruction-Level Semantics	Pattern 4.1: Invalid Addressing Mode: Provides a pre-scaled offset where a raw index is expected, causing double scaling.	1
P64	4. Incorrect Memory Addressing and Management	Uses incorrect stack offsets for its temporary array.	1
	6. Incorrect Loop Pointer/Index Management	str instruction corrupts the base pointer used for ldur in the next iteration.	1
	1. Incorrect Register State Management	Fails to preserve the input n, using an uninitialized register for the final read index.	1

Problem	Category	Pattern / Specific Error	Freq.
P65	1. Incorrect Register State Management	Consumes the main string pointer in the first loop, so it's invalid for the second operation (reading the last character).	1
	7. Misinterpretation of Algorithm's Goal	Incorrectly translates a simple if...then into a flawed csinc instruction.	1
P66	4. Incorrect Memory Addressing and Management	Uses an incorrect and dangerously small stack offset for a temporary buffer, risking a buffer overflow.	1
	2. Flawed Control Flow and Logic	Pattern 2.1/2.2: Incorrect branching after main logic fails to set the return value correctly for one code path.	1
P68	7. Misinterpretation of Algorithm's Goal	Fails to translate a compound conditional for state switching, producing jumbled ccmp/csel logic.	1
	1. Incorrect Register State Management	Uses the wrong length register (len1) as the index into the second number's buffer.	1
P71	6. Incorrect Loop Pointer/Index Management	Instruction Reordering: Decrements a "from-end" index before the ldr, causing an off-by-one read.	1
	1. Incorrect Register State Management	Pattern 1.1: Premature Update: Increments the output write index before all writes for the current iteration are complete.	1
P72	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering: Destroys input registers (a, b, c) during the Triangle Inequality check.	1
	7. Misinterpretation of Algorithm's Goal	Fails to understand that the inequality check requires all original values, leading to the state destruction.	1
P76	1. Incorrect Register State Management	Operates on a temporary copy of n inside the "divide out" loop, so n is never updated. (Same as P60).	1
	2. Flawed Control Flow and Logic	Unconditional branch in the "divide out" loop creates an infinite loop. (Same as P60).	1
P77	7. Misinterpretation of Algorithm's Goal	Incorrectly merges two independent loop conditions (power > n and count < 100) into a single faulty ccmp.	1
	1. Incorrect Register State Management	Unconditionally increments the loop counter, even on the final iteration that exits the loop.	1
P81	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering: The fixed pattern character register is overwritten inside the loop.	1
	7. Misinterpretation of Algorithm's Goal	Completely misinterprets the loop's comparison logic.	1
P82	4. Incorrect Instruction-Level Semantics	Pattern 4.2: Loads incorrect floating-point constants for grade thresholds. Pattern 4.1: Uses an invalid addressing mode (str x0, [x0]).	2
	6. Incorrect Loop Pointer/Index Management	Uses a stagnant output pointer, writing every result to out_array[0].	1

Problem	Category	Pattern / Specific Error	Freq.
P85	6. Incorrect Loop Pointer/Index Management	Uses incorrect pointer arithmetic to advance through a string.	1
	2. Flawed Control Flow and Logic	Uses an incorrect termination condition for a string reversal loop.	1
P86	1. Incorrect Register State Management	Pattern 1.2: Register Clobbering: A register holding a loop index is overwritten by a SIMD result before it's used.	1
P87	1. Incorrect Register State Management	1. Fails to initialize a pointer for the inner sort loop. 2. Pattern 1.1: Premature Update (resets word length before use).	2

Summary Statistics

Total Problems Analyzed: 45 problems (P6-P87, excluding gaps)

Category Distribution:

- **Category 1** (Incorrect Register State Management): Most frequent category
- **Category 2** (Flawed Control Flow and Logic): Second most common
- **Category 3** (Omission of Critical Instructions): Significant occurrence
- **Category 4** (Incorrect Instruction-Level Semantics): Multiple patterns identified
- **Category 6** (Incorrect Loop Pointer/Index Management): Common in loop-heavy algorithms
- **Category 7** (Misinterpretation of Algorithm's Goal): Complex logic translation issues
- **Category 8** (Failure to Generate Idiomatic/Optimized Code): Optimization failures
- **Category 9** (Code Hallucination): Rare but severe errors

Key Patterns:

- **Pattern 1.1:** Premature Update
- **Pattern 1.2:** Register Clobbering (most common)
- **Pattern 1.3:** Failure to Propagate Return Values
- **Pattern 1.4:** Incorrect Source/Destination in Update
- **Pattern 2.1:** Incorrect Branch Target
- **Pattern 2.2:** Premature/Incorrect Return
- **Pattern 4.1:** Invalid Addressing Mode
- **Pattern 4.2:** Incorrect Immediate Value
- **Pattern 4.3:** Incorrect Instruction Ordering