# KProbes Support for MIPS

Lubna, Vikas, Madhvesh

Sony India Software Centre

# Contents

- Kprobes Overview
- Kprobes Internals
- MIPS Kprobes Design
- Kprobes Processing Flow For MIPS
- Limitations

# Kprobes Overview

- Kprobes is simple and lightweight mechanism to collect debugging information dynamically.
- What is a Probe?
  - An explanatory action designed to investigate and obtain information on unknown region.
- Can be used to collect information in the interrupt handler with minimal disruption.
- Linux main line kernel 2.6.16 (and later) includes support for
  - x86, ppc64, Sparc, IA64, x86_64
- We have implemented Kprobes support for MIPS Arch for vanilla kernel 2.6.16-24.

# Kprobes Overview (Contd)

- Kprobes introduces kernel probes and their corresponding probe handlers from the user to the kernel code.
- A kernel probe is a set of handlers placed on a certain instruction address.
- Kprobe mainly contains three handlers
  - Pre-Handler
  - Post-Handler
  - Fault Handler

# Kprobes Overview (Contd)

- Pre-Handler
  - Executed before the probed instruction
  - It can be used to dump the register contents before executing the probed instruction
- Post-Handler
  - Executed after the probed instruction
  - It can be used to dump the register contents after executing the probed instructions
- Fault Handler
  - Executed When some fault occur in pre handler or post handler or in the instruction being debugged

# Kprobes Internals

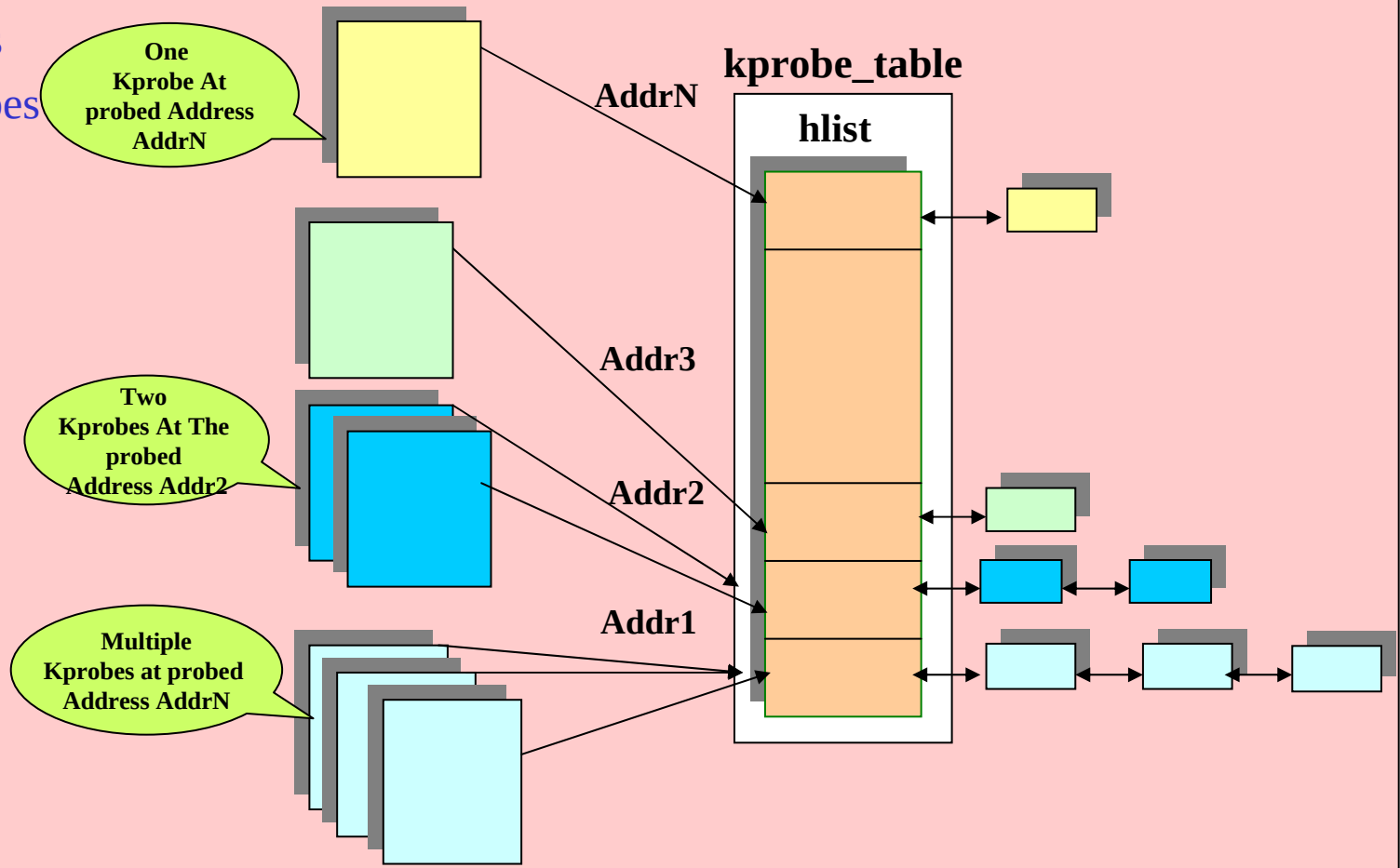
## Kprobes Management (Arch Independent)

Includes:

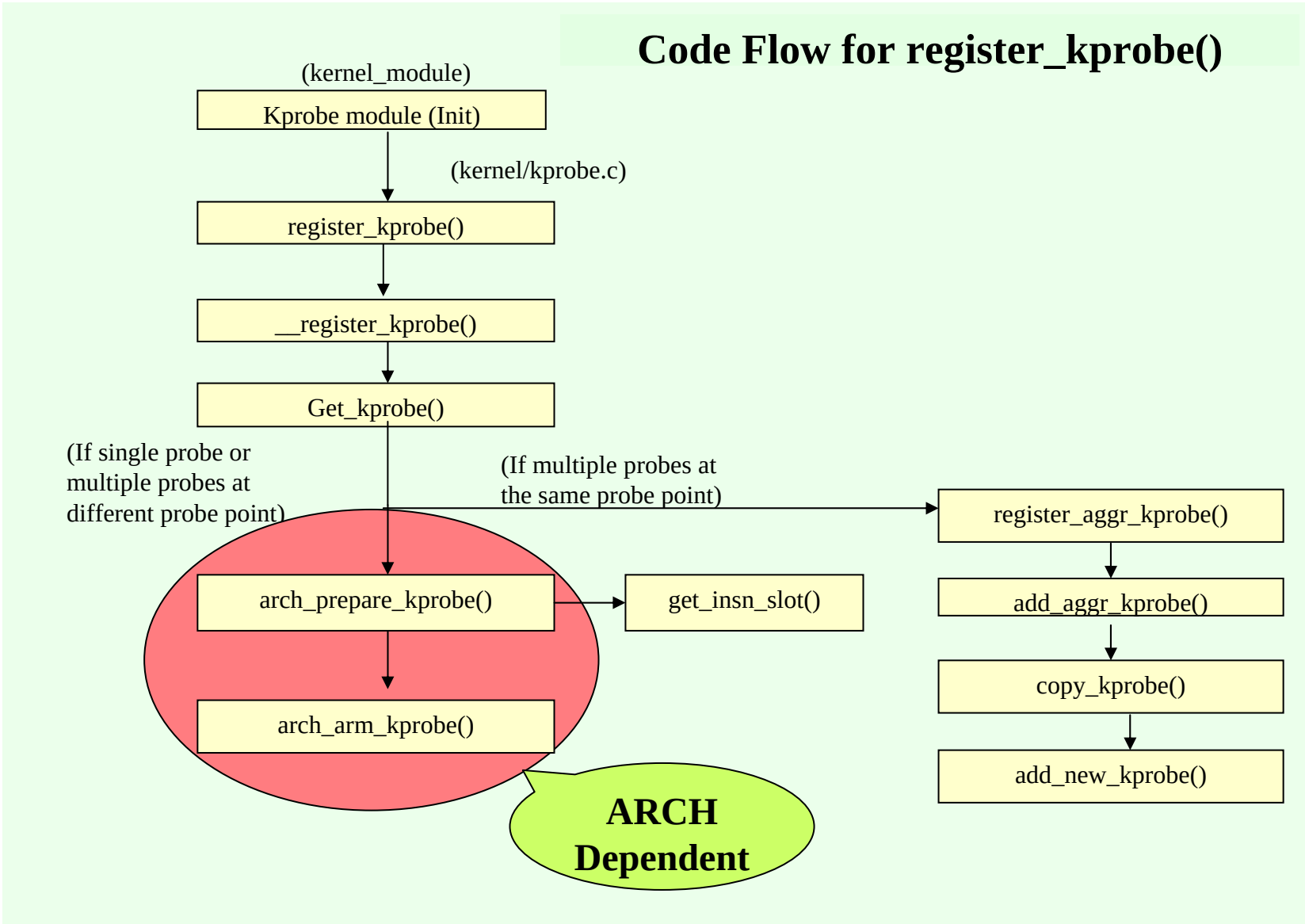
-Initializing the kprobes

-Register kprobes

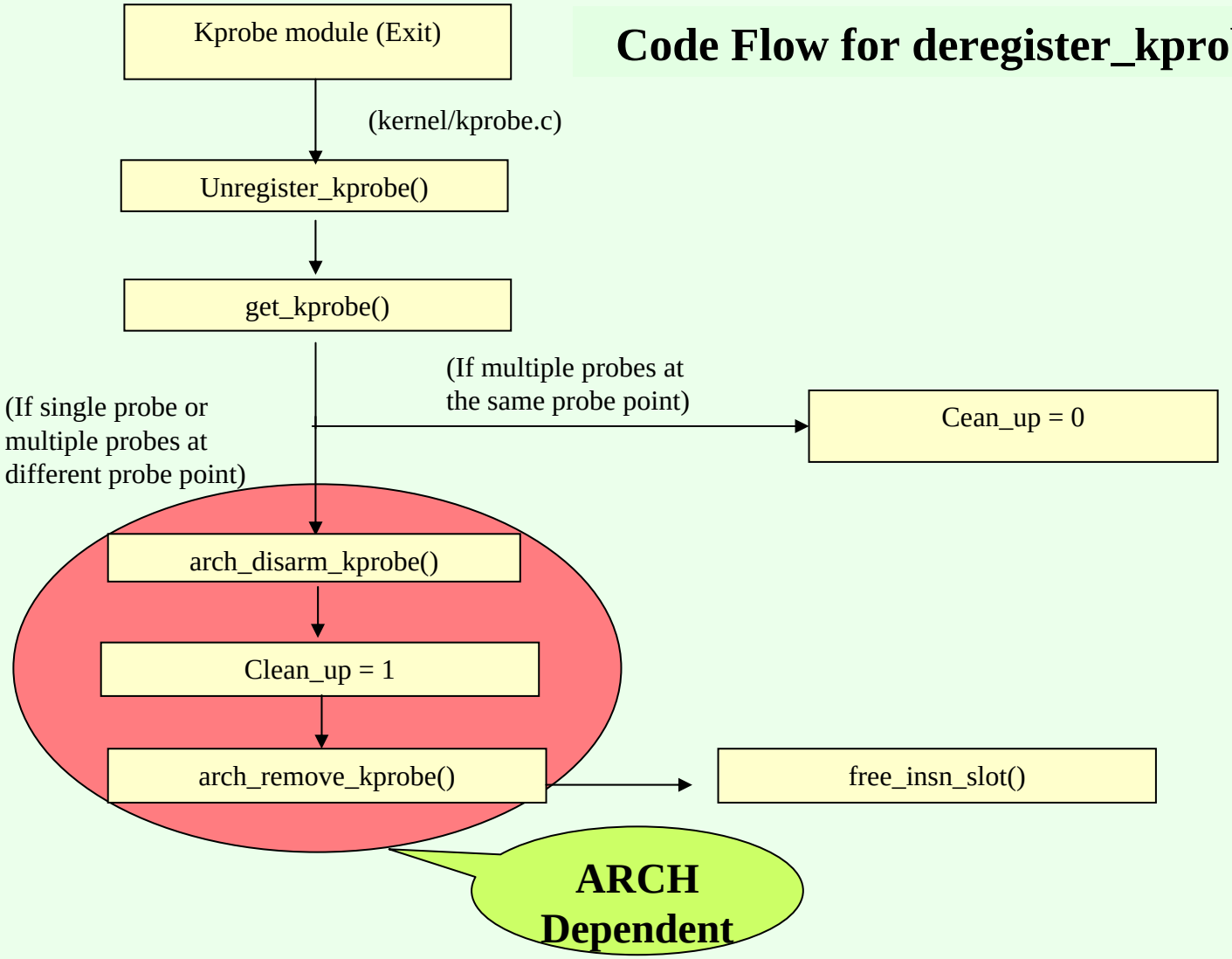
-Deregister kprobes



# Kprobes Internals (Contd)

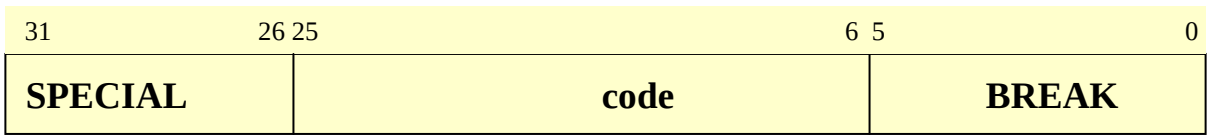


# Kprobes Internals (Contd)



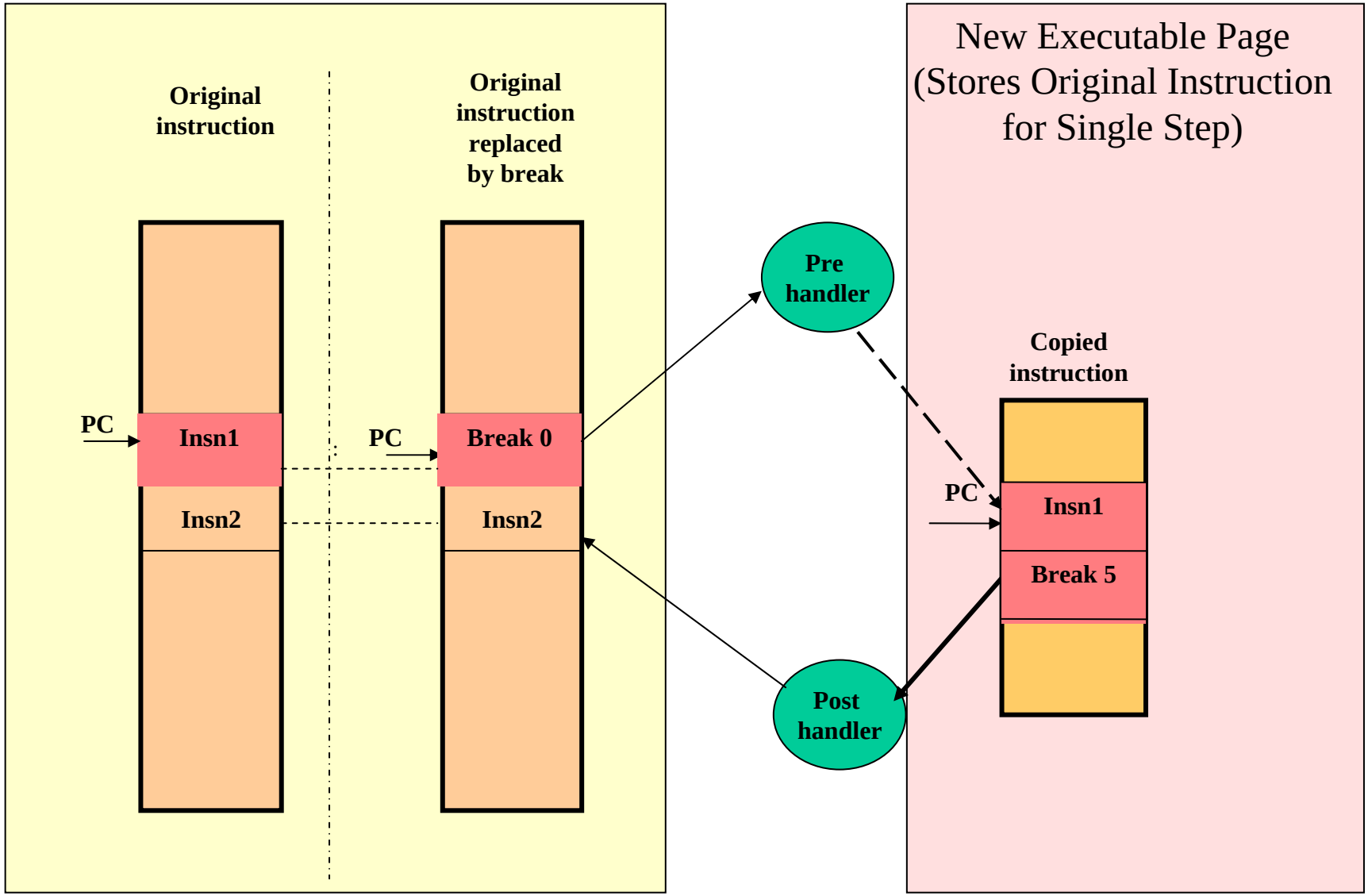
# MIPS Kprobes Design

- MIPS Arch does not have single step instruction.
- MIPS “break” instruction is used to implement kprobes breakpoint and single step operations
- Break instruction format is




- The “code” field is utilized in implementing kprobes as below
- Code Field = 0 (BRK_USERBP): Indicates break point put by kprobes to probe the instruction to be debugged. (break0=0x0000000d)
- Code Field = 5 (BRK_SSTEPBP): Indicates software single step to debug the probed instruction. There is no hardware single-stepping on MIPS. Hence software single stepping is implemented with breakpoint trap break 5. (break5=0x0000014d)

# MIPS Kprobes Design (Contd)



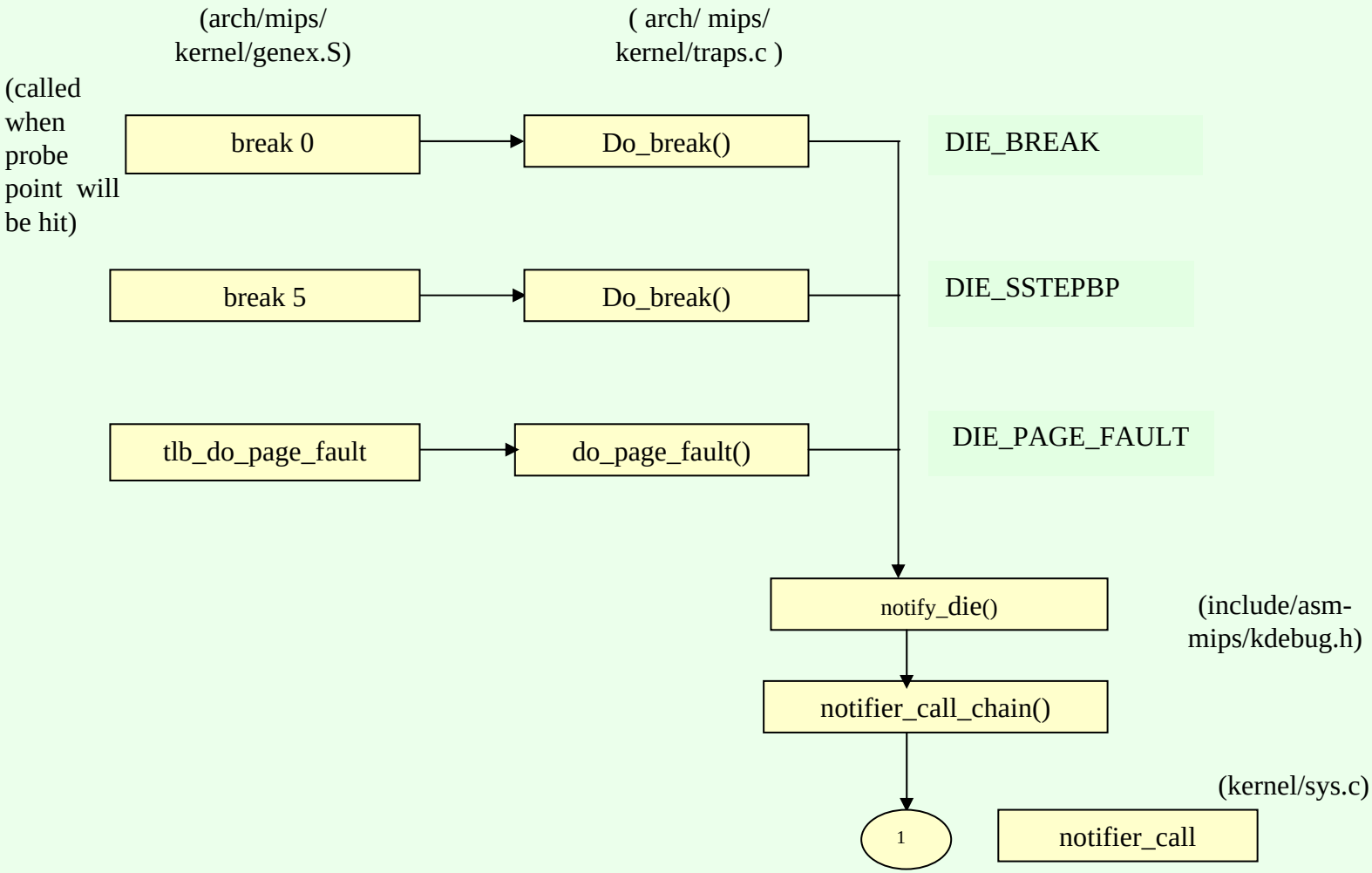
# MIPS Kprobes Design (Contd)

- When a particular instruction “Insn1” is probed, Kprobes will copy the original instruction on the executable page and replaces the probed instruction by ‘break 0’ i.e 0x0000000d
- ‘break 5’ (0x0000014d) will be copied on the executable page after the original probed instruction
- A breakpoint exception occurs, immediately and unconditionally transferring control to the exception handler.
- When break 0 is hit, kprobes exception notifier will be called by the exception handler do_break() with the die value DIE_BREAK. This will in turn call kprobe_handler() which performs executing pre_handler() associated with kprobes.

# MIPS Kprobes Design (Contd)

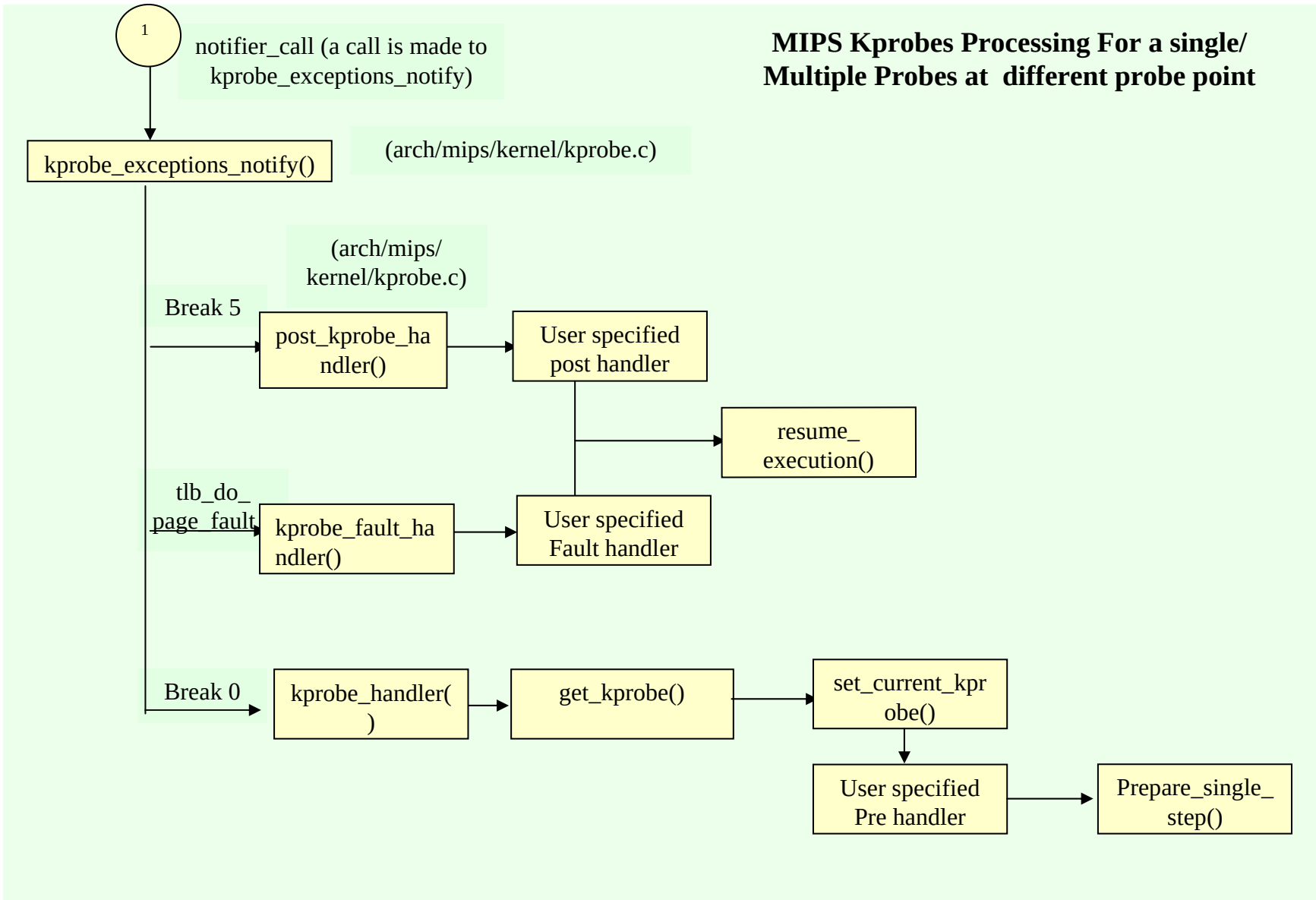
- kprobe_handler() takes care of moving program counter to the copied instruction by calling prepare_single_step().
- After executing the probed instruction, the next instruction on the copied page ‘break 5’ will be executed
- When break 5 is hit, again kprobes exception notifier will be called by the exception handler do_break() with the die value DIE_SSTEPBP. This will in turn call user defined post_handler().
- After the instruction debugging, the execution resumes to the next instruction “Insn2”

# Kprobes Processing Flow For MIPS

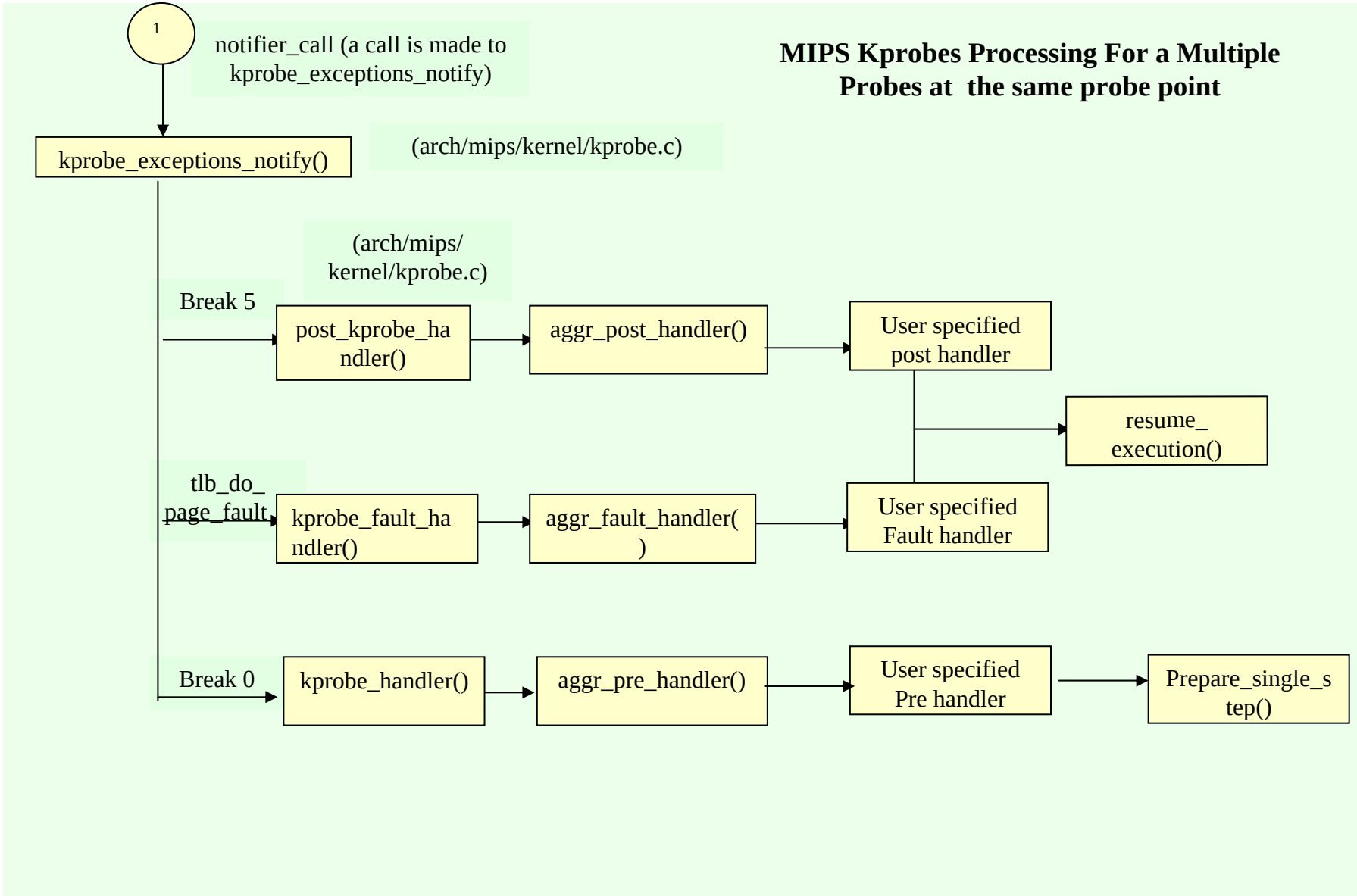



**MIPS Generic Processing**

# Kprobes Processing Flow For MIPS (Contd)



# Kprobes Processing Flow For MIPS (Contd)



# Limitations

- Kprobe cannot be set for branch and jump instructions
- Not tested in SMP configuration