arm中对kgdb breakpoint的支持是通过“undefined instruction” trap来实现的。

kgdb breakpoint在arm中分为2种

```
#define KGDB_BREAKINST      0xe7ffdefe
#define KGDB_COMPILED_BREAK 0xe7ffdeff
```

这两条在arm中都是"undefined instruction".前者是用户动态设置breakpoint时使用的，而后者是静态编译在code中的。

in arch/arm/include/asm/kgdb.h

```
static inline void arch_kgdb_breakpoint(void)
{
    asm(__inst_arm(0xe7ffdeff));
}
```

arch_kgdb_breakpoint()即产生一条KGDB_COMPILED_BREAK instruction.

所以在kernel code(driver code)中可以调用该函数来嵌入一个breakpoint.

Note: 需要include如下文件

```
#include <linux/kgdb.h>
```

in kernel/debug/debug_core.c

```
/**
 * kgdb_breakpoint - generate breakpoint exception
 *
 * This function will generate a breakpoint exception.  It is used at the
 * beginning of a program to sync up with a debugger and can be used
 * otherwise as a quick means to stop program execution and "break" into
 * the debugger.
 */
noinline void kgdb_breakpoint(void)
{
    atomic_inc(&kgdb_setting_breakpoint);
    wmb(); /* Sync point before breakpoint */
    arch_kgdb_breakpoint();
    wmb(); /* Sync point after breakpoint */
    atomic_dec(&kgdb_setting_breakpoint);
}
EXPORT_SYMBOL_GPL(kgdb_breakpoint);
```

使用该API可能更合理。只需要

```
#include <linux/kgdb.h>
```