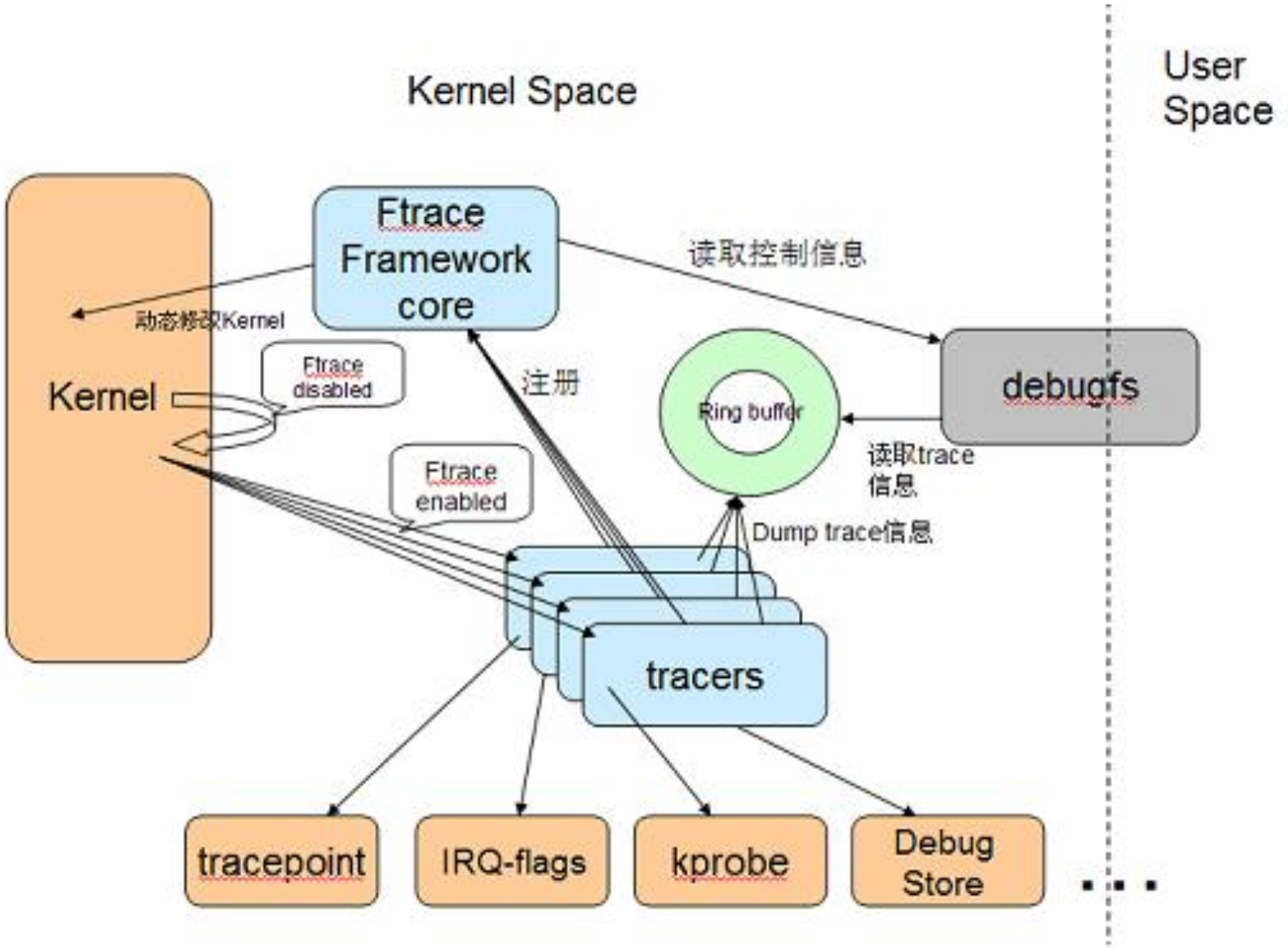
What is Ftrace?

- Provides a generic framework for tracing
  - Infrastructure for defining tracepoints
  - Ability to register different kinds of tracers
  - Specialized data structure (ring buffer) for trace data storage

Ftrace Components



FTrace Operation

- Instrumentation
  - Explicit
    - Tracepoints defined by declaration
    - Calls to trace handler written in source code
  - Implicit
    - Automatically inserted by compiler
      - Uses gcc '-pg' option
    - Inserts call to 'mcount' in each function prologue
    - Easy to maintain – no source code modifications
    - Only practical way to maintain 20,000+ tracepoints

mcount Routine

- 'mcount' is called by every kernel function
  - Except inlines and a few special functions
- Must be a low-overhead routine
- Incompatible with some compiler optimizations
  - E.g. cannot omit frame-pointers on ARM
  - Compiler disables some optimizations automatically
  - Analysis of assembly indicates that mcount callers have well-defined frames
- Misc note:
  - New mcount routine (_gnu_mcount) is coming

Code to Call mcount

```
00000570 <sys_sync>:
570: e1a0c00d mov ip, sp
574: e92dd800 stmdb sp!, {fp, ip, lr,
   pc}
578: e24cb004 sub fp, ip, #4 ; 0x4




57c: e3a00001 mov r0, #1 ; 0x1
580: ebffffa0 bl 408 <do_sync>
584: e3a00000 mov r0, #0 ; 0x0
588: e89da800 ldmia sp, {fp, sp, pc}
```

```
00000570 <sys_sync>:
570: e1a0c00d mov ip, sp
574: e92dd800 stmdb sp!, {fp, ip, lr,
   pc}
578: e24cb004 sub fp, ip, #4 ; 0x4
57c: e1a0c00e mov ip, lr
580: ebffffe bl 0 <mcount>
584: 00000028 andeq r0, r0, r8, lsr #32
588: e3a00001 mov r0, #1 ; 0x1
58c: ebffff9d bl 408 <do_sync>
590: e3a00000 mov r0, #0 ; 0x0
594: e89da800 ldmia sp, {fp, sp, pc}
```

Function tracer的实现