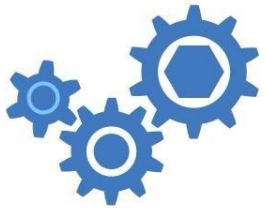
High Performance GC

--IBM Z14 Guarded-Storage Facility Implementation

郝庆丰　IBM KVM Development on System Z



IBM Runtime Technologies

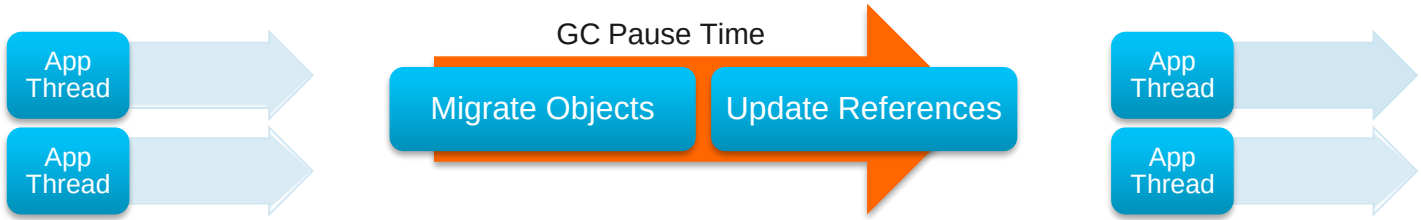
Sep 1, 2017

Agenda

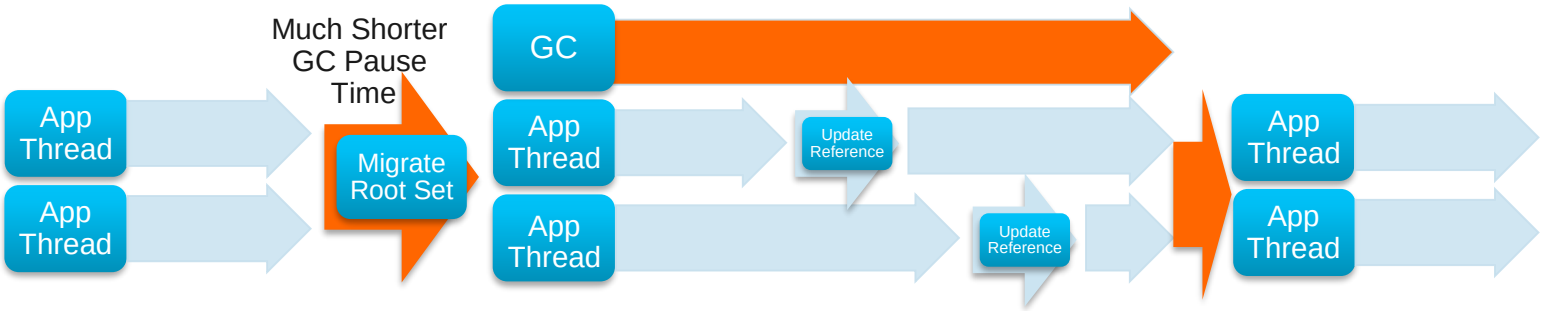
- **Background**
- Guarded-Storage Facility
- Handling Flow
- Implementation
- A sample of pause-less GC

Background

- **Challenge**: *Object References* need to be updated as objects are migrated in the heap
- **Currently**: Stop app threads, Migrate objects, Update references, Resume app threads



- **Pause-Less GC**: Individual app thread *traps* on reference to guarded region, Migrate object / Update referenced value, App thread resumes



Agenda

- Background
- **Guarded-Storage Facility**
- Handling Flow
- Implementation
- A sample of pause-less GC

Hardware Concepts

- Heap divided to 64 regions
  - Region size: 512KB~1PB
  - 64-bit mask register controlling each region
  - Per-thread
  - New fetch instructions（LGG/LLGFSG）
  - New trap
- Lightweight Trap

Z14
• Instruction address of Load • Memory operand address • Memory operand value • Return address (would be TBEGIN when within TX)

Guarded-Storage Facility

- For memory-reclaim program: GC
- Support multiple languages
- Each storage region has a separate mask bit
- Enable/Disable by control register bit 59
- Four special instructions: LGG，LLGFSG，LGSC，STGSC
- User-defined event handler
- Per-thread

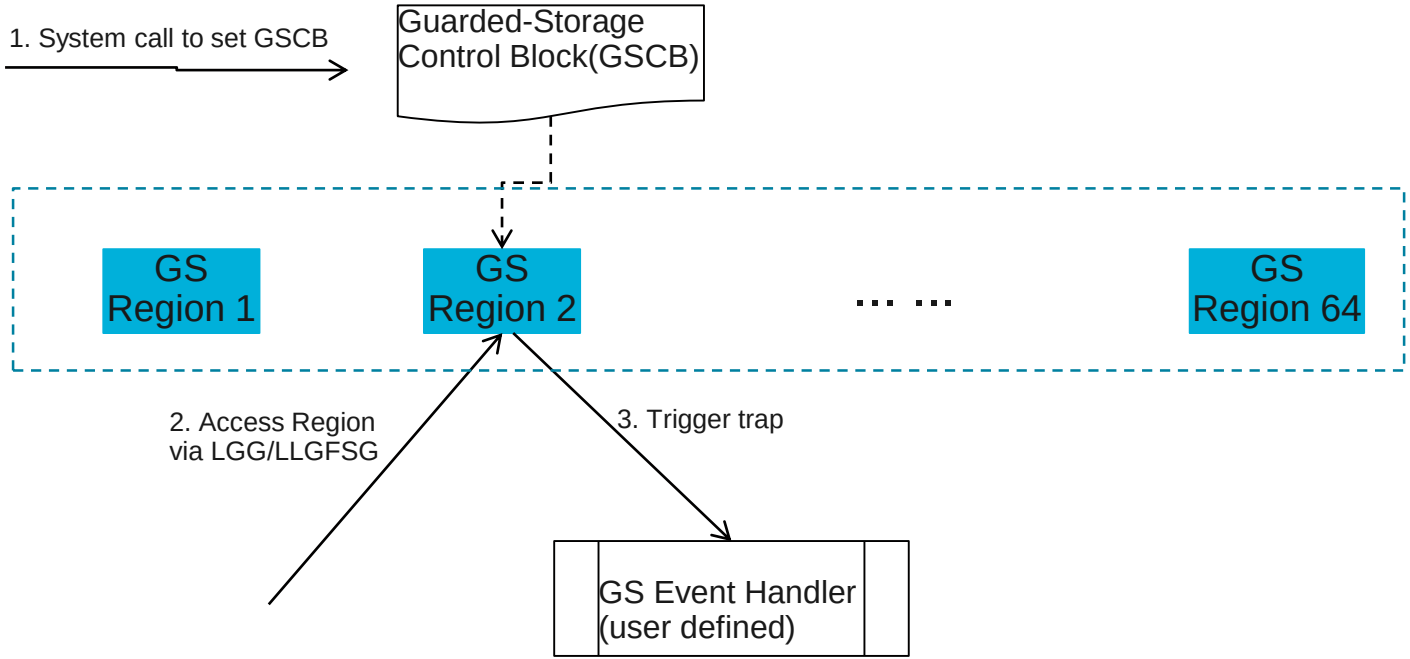
Special instructions

- LOAD GUARDED (LGG)
- LOAD LOGICAL AND SHIFT GUARDED (LLGFSG)
- LOAD GUARDED STORAGE CONTROLS (LGSC)
- STORE GUARDED STORAGE CONTROLS (STGSC)

Agenda

- Background
- Guarded-Storage Facility
- **Handling Flow**
- Implementation
- A sample of pause-less GC

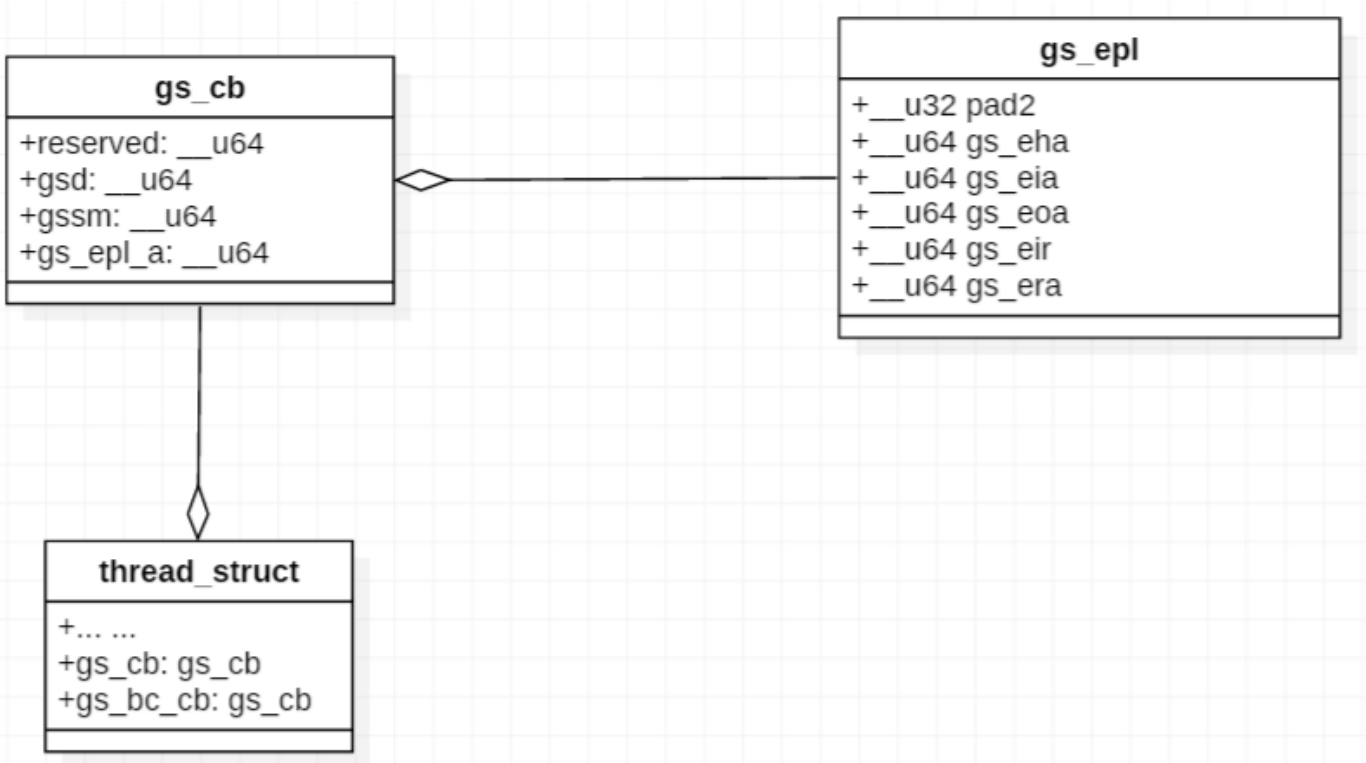
Handling Flow



Agenda

- Background
- Guarded-Storage Facility
- Handling Flow
- **Implementation**
- A sample of pause-less GC

Implementation



Implementation

```
#define GS_ENABLE         0
#define GS_DISABLE        1
#define GS_SET_BC_CB      2
#define GS_CLEAR_BC_CB    3
#define GS_BROADCAST      4

int s390_guarded_storage(int command, struct gs_cb __user gs_cb);
syscall(378, command, gs_cb);
```

Upstream commits:

e525f8a s390/gs: add regset for the guarded storage broadcast control block
4e0b1ab KVM: s390: gs support for kvm guests
916cda1 s390: add a system call for guarded storage

Example

This example uses the system call of guarded-storage facility to protect 64M memory, which is [0x1234000000, 0x1234400000].

When application accesses any address in this region, a trap occurs.

```
/* Exec LGG instruction with address p.*/
static inline unsigned long load_guarded(unsigned long *p)
{
    unsigned long v;
    asm(".insn rxy,0xe3000000004c, %0,%1" : "=d" (v) : "m" (*p) : "14");
    return v;
}

/* Guarded-storage event handler */
void gs_handler(struct gs_cb *this_cb)
{
    struct gs_epl *gs_epl = (struct gs_epl *) this_cb->gs_epl_a;
    printf("gs_handler called for %016lx at %016lx\n", gs_epl->gs_eir, gs_epl->gs_eia);
    /* Disable guarded-storage factiliy */
    gs_cb.gssm = 0UL;
    load_gs_cb(&gs_cb);
}
```

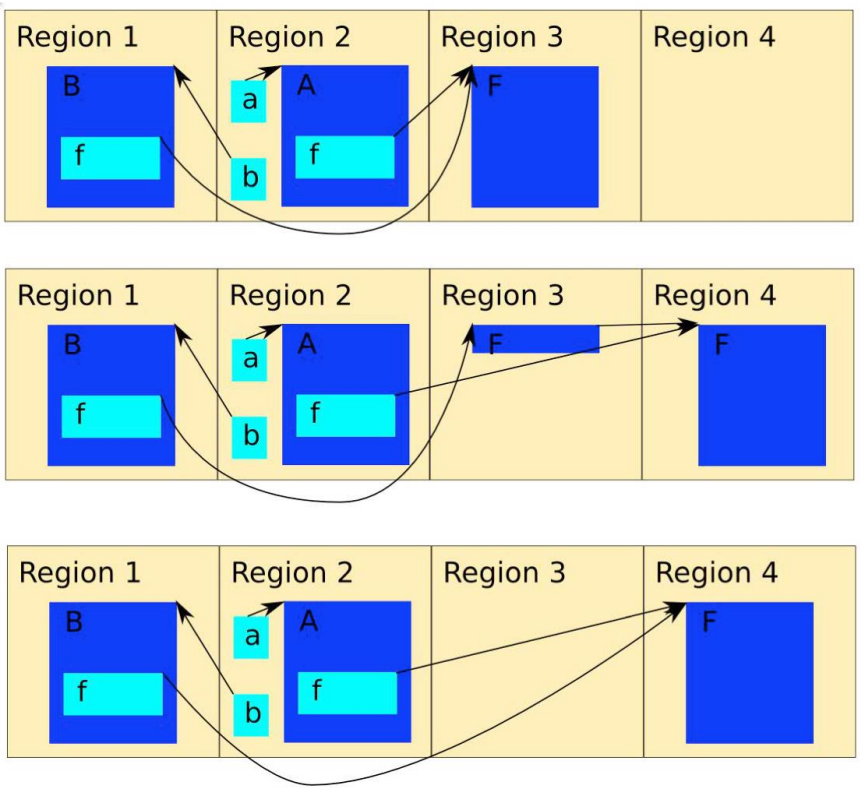
```c
unsigned long test_addr = 0x1234000000UL;
int main(int argc, char *argv[])
{
    struct gs_cb gs_cb;
    struct gs_epl gs_epl;

    /* Enable guarged_storage facility */
    if (syscall(378, GS_ENABLE) != 0)
        exit(1);
    gs_cb.gsd = test_addr | 26;
    gs_cb.gssm = 0x5555555555555555UL;
    gs_cb.gs_epl_a = (unsigned long) &gs_epl;
    gs_epl.gs_eha = (unsigned long) gs_handler_asm;
    syscall(378, GS_SET_BC_CB, &gs_cb);
    /* This will trigger guarded-storage event */
    printf("Access to test = %p: %016lx\n", test, load_guarded(&test));
    return 0;
}
```

Agenda

- Background
- Guarded-Storage Facility
- Handling Flow
- Implementation
- **A sample of pause-less GC**

A sample of pause-less GC



- **Region 3 identified to be guarded-storage**
- Object F in Region 3 needs to be migrated to Region 4
- **LGG ptr, a.f**
  - PauseLessGCMoveObjectHelper(F)
  - Compare-Swap a.f = new_F
  - Re-execute LGG ptr, a.f
- **LGG ptr, b.f**
  - Compare-Swap b.f = new_F
  - Re-execute LGG ptr, b.f

Performance data of pause-less GC

Java GC-tuning made easier

High scavenge pause times made this application a candidate for Pause-less GC

- Up to **3.4x** better throughput for **response-time** constrained Service Level Agreements (SLAs)
- Up to **10x** better average GC pause-times

Standard:

https://www.spec.org/jbb2015/docs/userguide.pdf

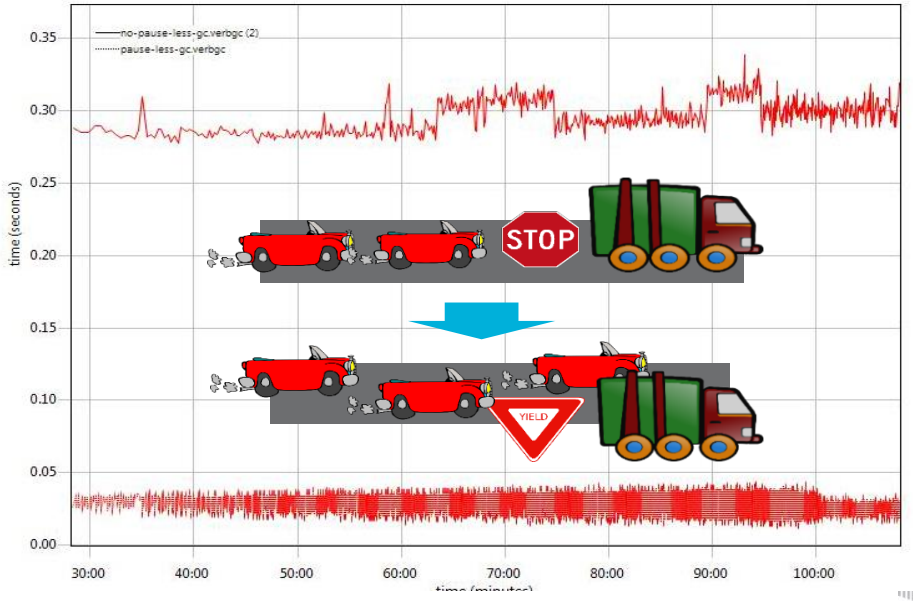
Enable Pause-less GC with:

- IBM Java 8 SR5 or newer
- IBM z14's Guarded Storage Facility
- JVM option:

  -Xgc:concurrentScavenge

Pause time

Variant	Mean	Minimum	Maximum	Total
	time (seconds)	time (seconds)	time (seconds)	time (seconds)
no-pause-less-gc.verbgc (2)	0.3	0.28	0.34	199
pause-less-gc.verbgc	0.03	0.01	0.04	54.1



谢 谢！ Thanks!

Q&A