Week6 WAL / Manifest

Made by Suhwan Shin, Isu Kim, Seyeon Park

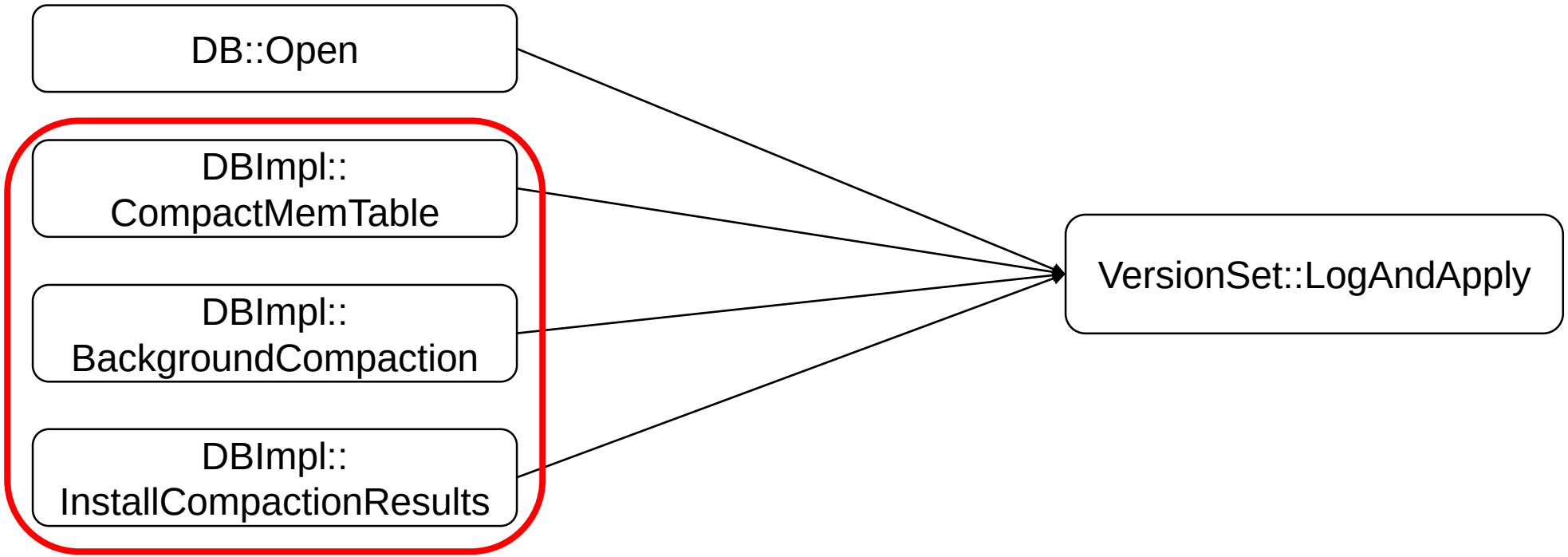
E-Mail: tlstnghks77@dankook.ac.kr

Content

- Code flow
  - VersionSet
- Code flows & Example
  - log::Writer::AddRecord
  - log::Writter::EmitPhysicalRecord
  - PosixWritableFile::Append
  - Put example
- Log format
- Code flow
  - log::reader

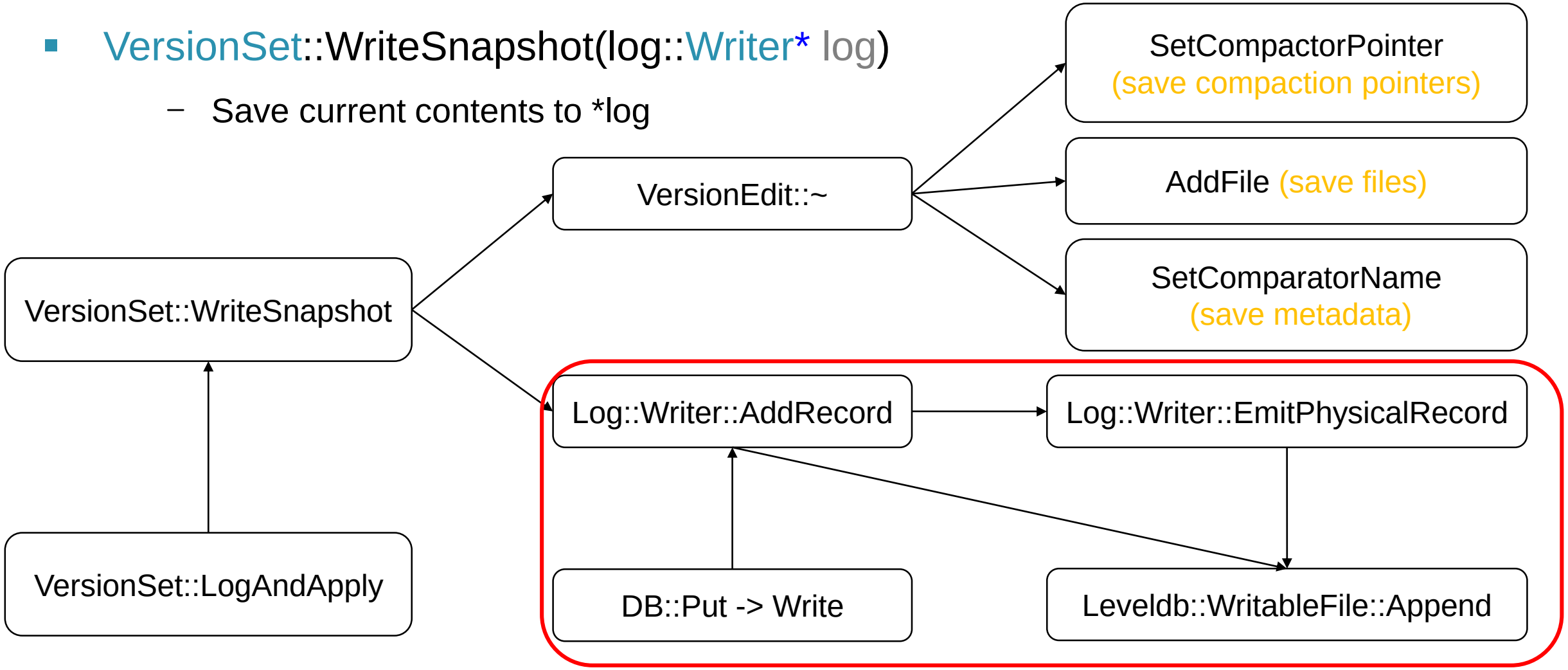
VersionSet::LogAndApply

- Status VersionSet::LogAndApply(VersionEdit* edit, port::Mutex* mu)
  - Apply *edit to the current version to form a new descriptor that is both saved to persistent state and installed as the new current version.



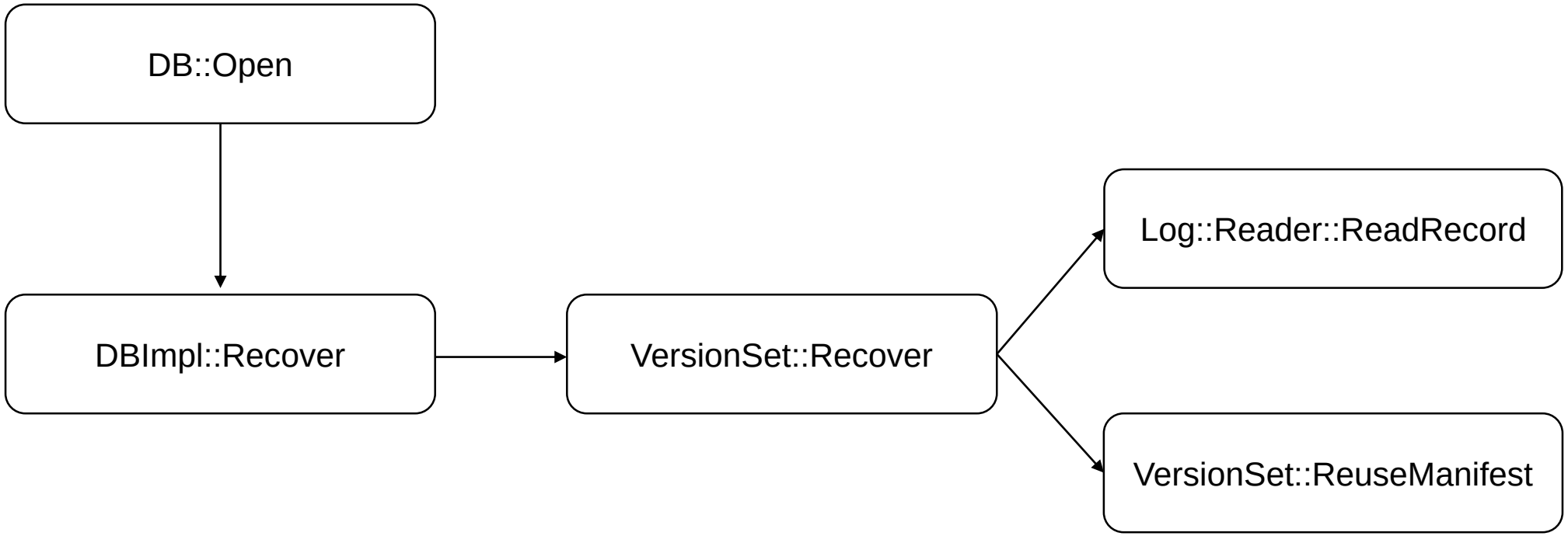
VersionSet::WriteSnapshot

- VersionSet::WriteSnapshot(log::Writer* log)
  - Save current contents to *log



VersionSet::Recover

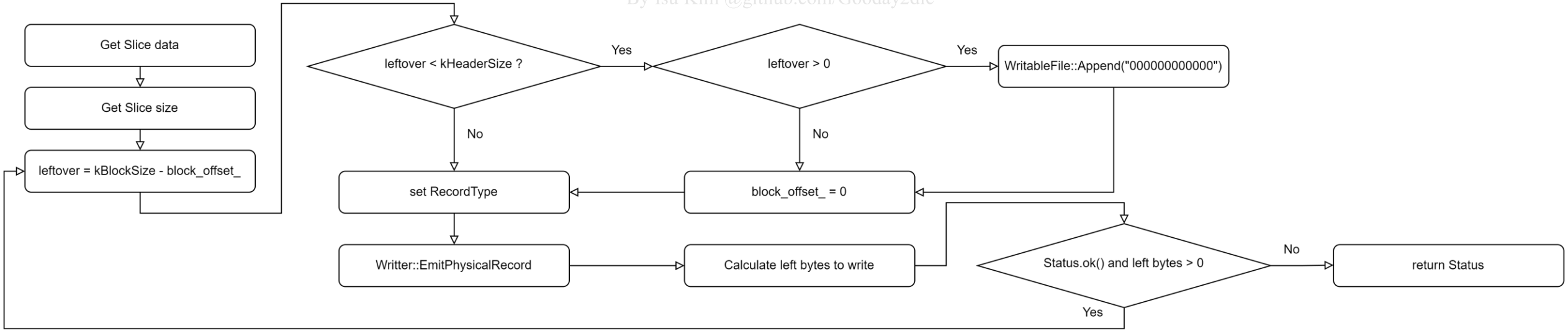
- VersionSet::Recover(bool* save_manifest)
  - Recover the last saved descriptor from persistent storage.



log::Writer::AddRecord

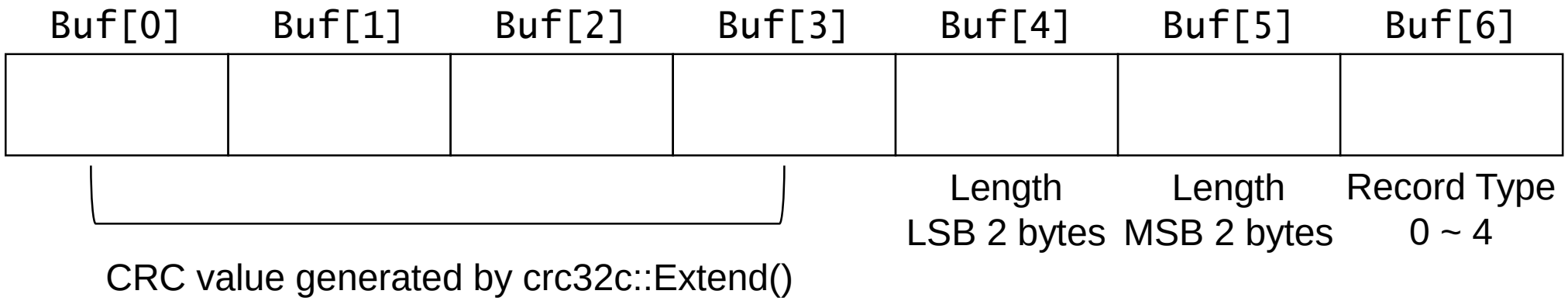
- Iterate and write Slice data till Slice is empty
- Defined in log_writer.cc @ line 34
- Checks Record Type and writes data using EmitPhysicalRecord.

https://github.com/DKU-StarLab/leveldb-study/



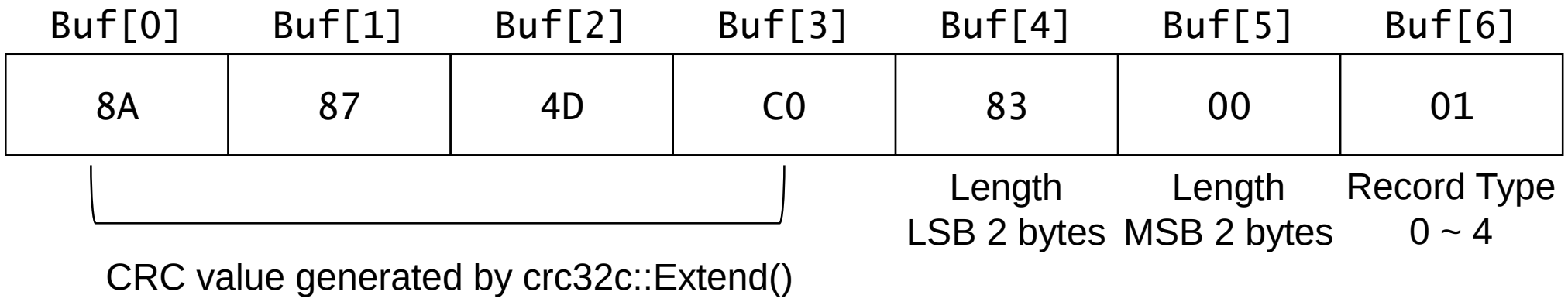
log::Writer::EmitPhysicalRecord

- Generates Header in following format
- Defined in log_writer.cc @ line 86
- `kHeaderSize` defaults to 7
- Will be written using `PosixWritableFile::Append`



log::Writer::EmitPhysicalRecord - Example

- .log file generated by db_bench
- **CRC Value** : 0xC04D878A
- **Length** : 0x0083 (131 bytes)
- **Record Type** : 0x01 (kFullType)



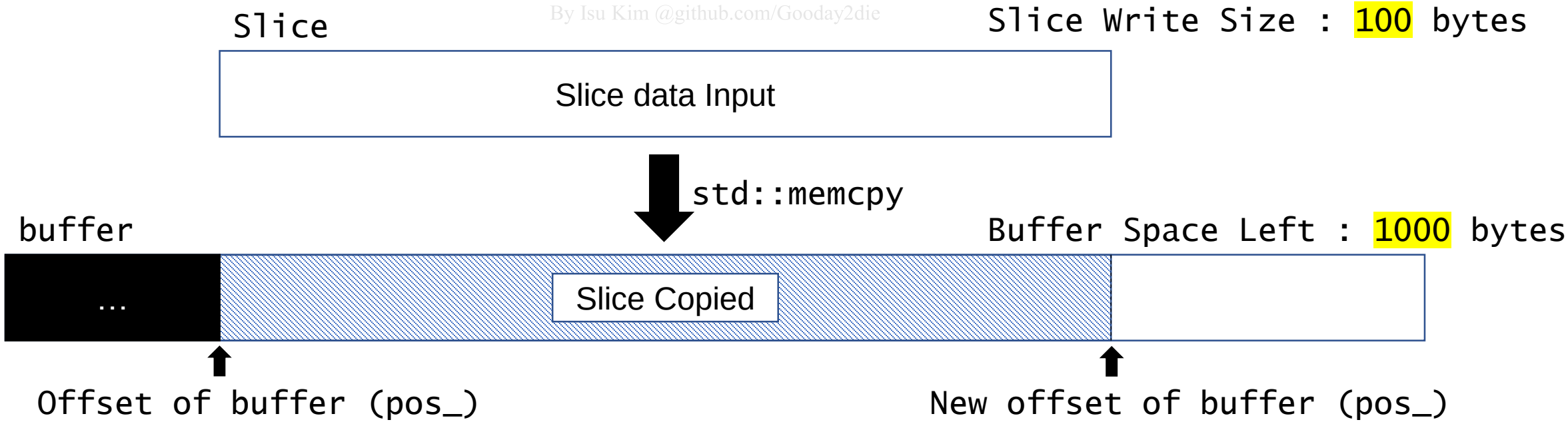
PosixWritableFile::Append – Case 1

- When buffer size of WritableFile was **bigger** than write size.

1. Copy all Slice data.
2. Calculate buffer offset(pos_) and return Status::OK().

https://github.com/DKU-StarLab/leveldb-study/

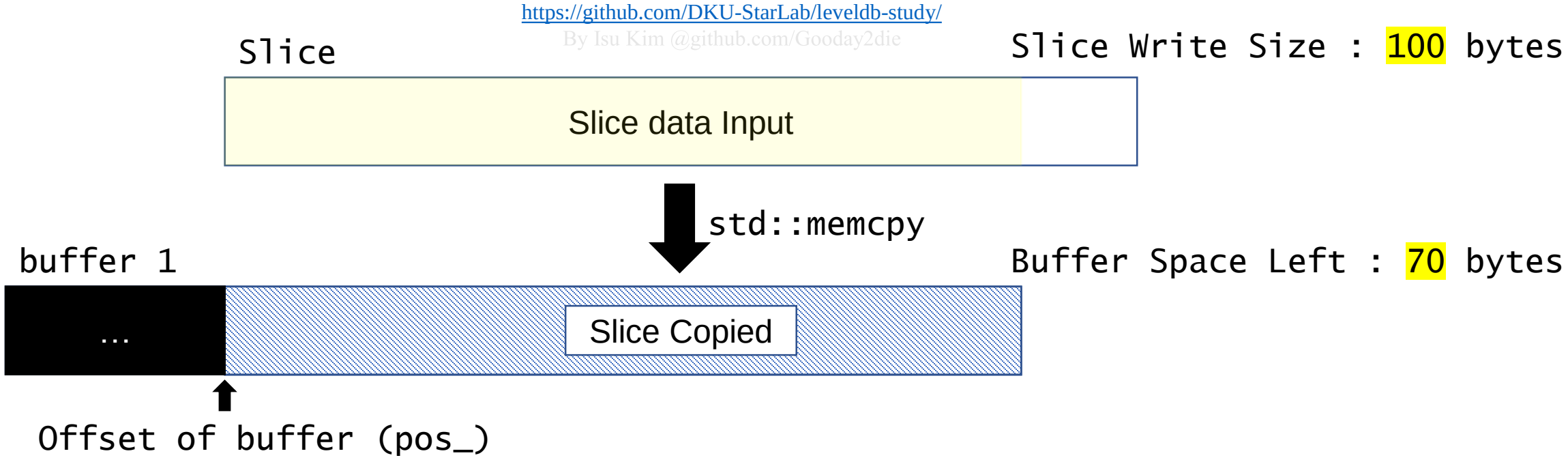
By Isu Kim @github.com/Gooday2die



PosixWritableFile::Append – Case 2.1

- When buffer size of WritableFile was **smaller** than write size.

1. Copy as much data as possible from Slice.
2. Perform FlushBuffer()

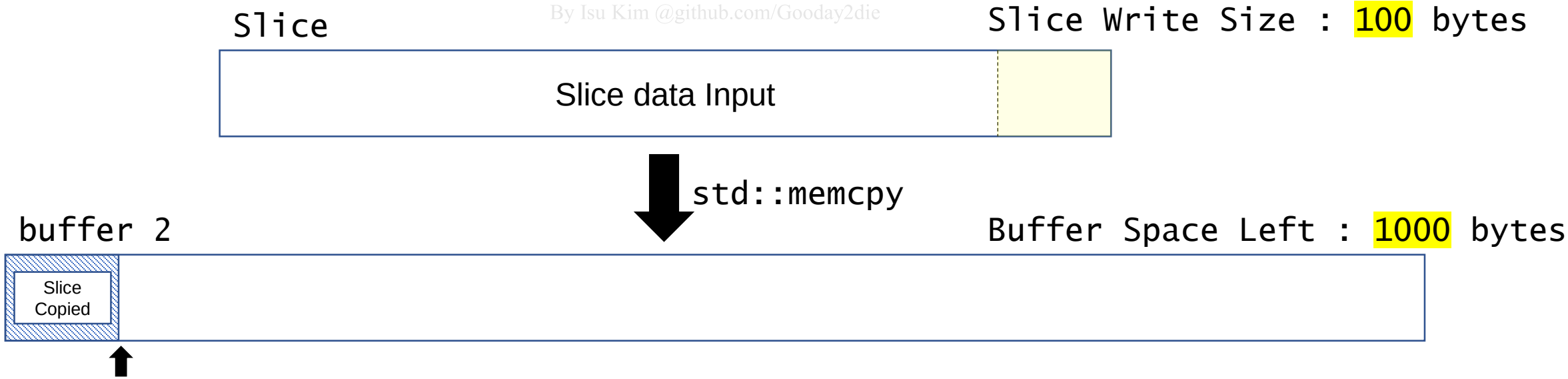


PosixWritableFile::Append – Case 2.1

- When buffer size of WritableFile was **bigger** than write size.

1. Copy all Slice data.
2. Calculate buffer offset(pos_) and return Status::OK().

https://github.com/DKU-StarLab/leveldb-study/



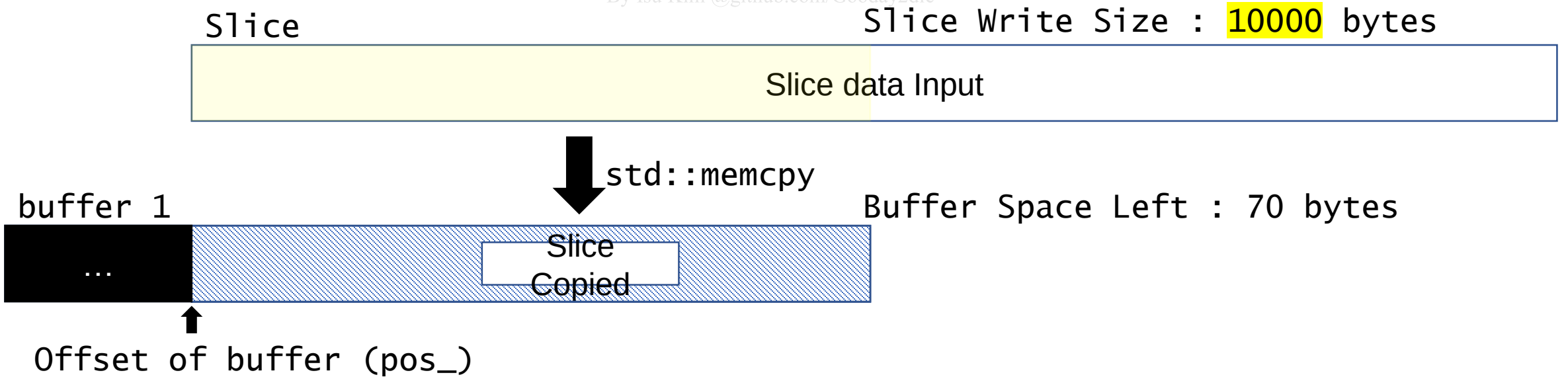
PosixWritableFile::Append – Case 2.2

- When buffer size of WritableFile was **smaller** than write size.

1. Copy as much data as possible from Slice.
2. Perform FlushBuffer()

https://github.com/DKU-StarLab/leveldb-study/

By Isu Kim @github.com/Gooday2die

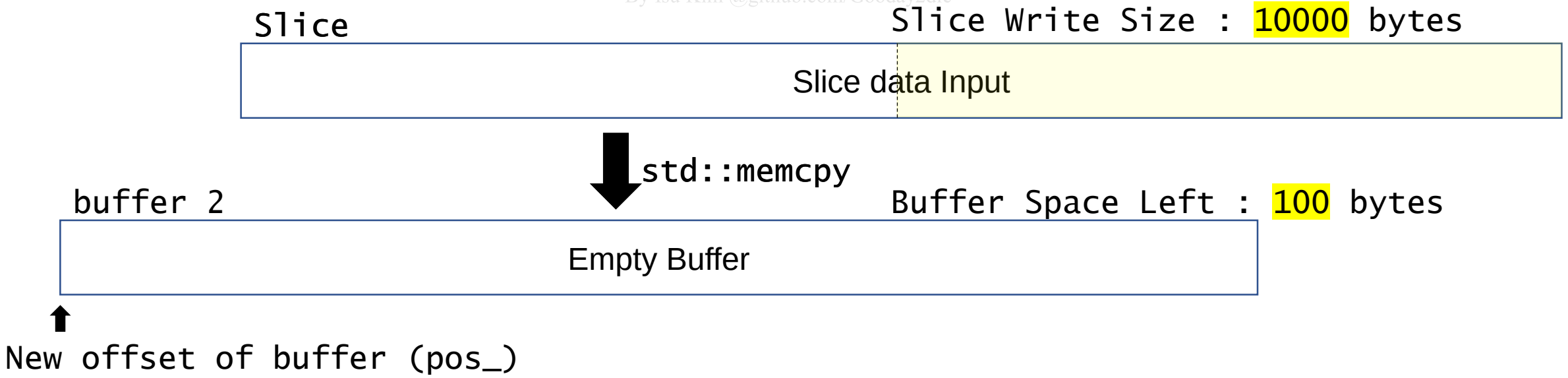


PosixWritableFile::Append – Case 2.2

- When buffer size of WritableFile was **bigger** than write size.

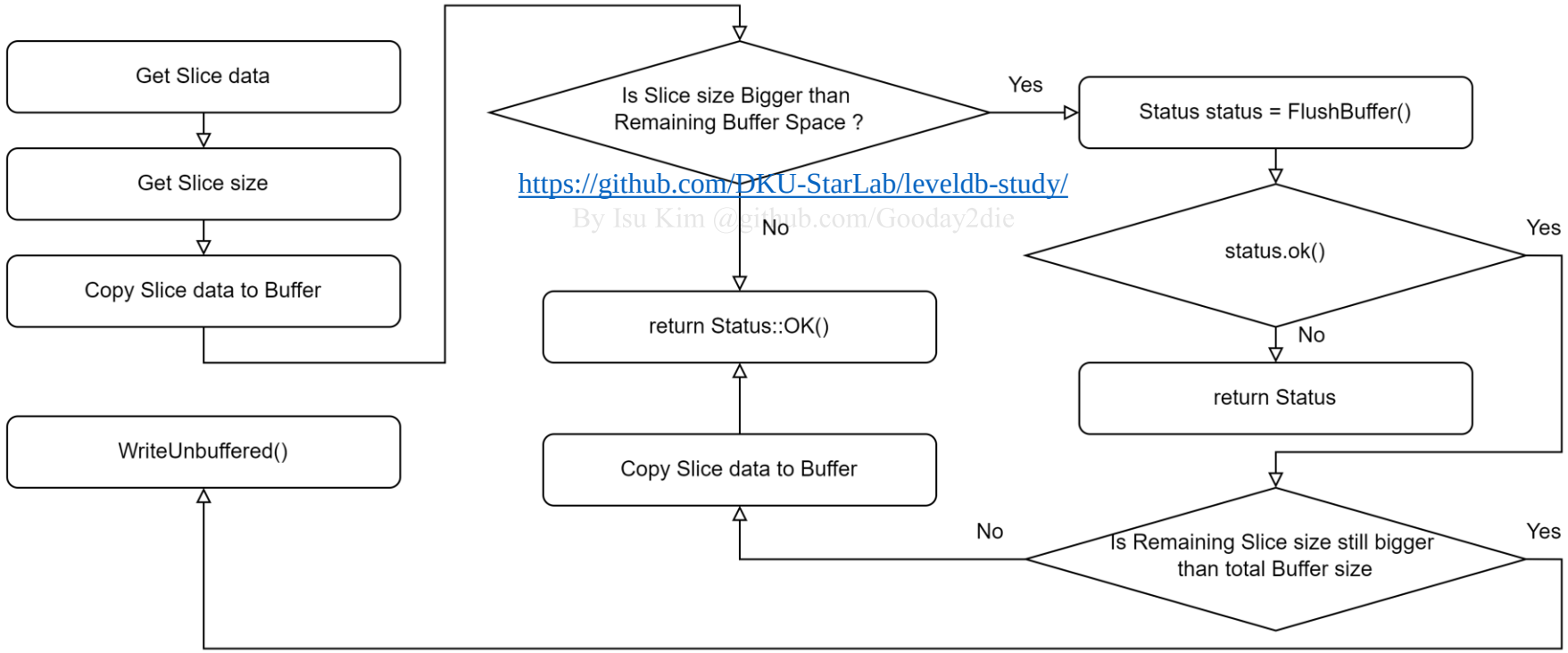
1. Write left data using WriteUnbuffered();
2. Return Status::OK().

https://github.com/DKU-StarLab/leveldb-study/



PosixWritableFile::Append - Flowchart

- Flowchart of PosixWritableFile::Append



WAL Example for Put - Header

- Tested ‘put’ using {"A": "Hello world!", "B“: "Good bye world!“, "C“: "I am hungry"}

```
0000h: C0 55 16 4E 1C 00 01 01 00 00 00 00 00 00 00 01  ÀU.N............
0010h: 00 00 00 01 01 41 0C 48 65 6C 6C 6F 20 77 6F 72  .....A.Hello wor
0020h: 6C 64 21 8E C7 D9 9D 1F 00 01 02 00 00 00 00 00  ld!ŽÇÙ..........
0030h: 00 00 01 00 00 00 01 01 42 0F 47 6F 6F 64 20 62  ........B.Good b
0040h: 79 65 20 77 6F 72 6C 64 21 26 5D 13 F3 1B 00 01  ye world!&].ó...
0050h: 03 00 00 00 00 00 00 00 01 00 00 00 81 01 01 43  ...............C.
0060h: 49 20 61 6D 20 68 75 6E 67 72 79                 I am hungry
```

.log file

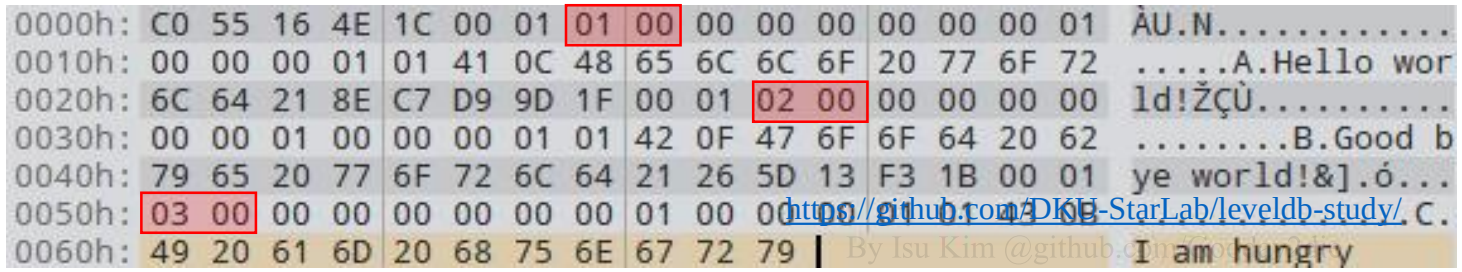
```
int main (void) {
    dbTest* db = new dbTest();

    db->putValue("A", "Hello world!");
    db->putValue("B", "Good bye world!");
    db->putValue("C", "I am hungry");

    delete(db);
    return 0;
```


- Example with {“A”: “Hello World!”}

Buf[0]	Buf[1]	Buf[2]	Buf[3]	Buf[4]	Buf[5]	Buf[6]
C0	55	16	4E	1C	00	01
CRC value generated by crc32c::Extend()				Length LSB 2 bytes	Length MSB 2 bytes	Record Type 0 ~ 4

➔ **Checksum : 0x4E1655C0 / Length: 0x1C(28) bytes / Record Type : kFullType**

WAL Example for Put – Entry Number

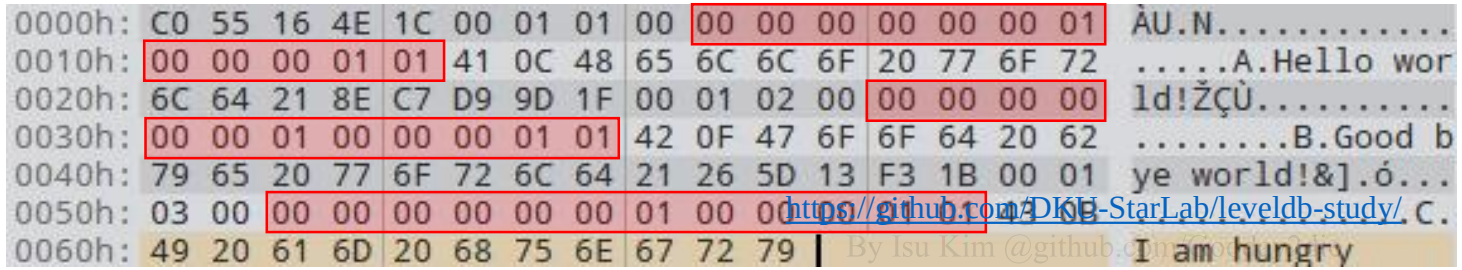


.log file

```int main (void) {
    dbTest* db = new dbTest();

    db->putValue("A", "Hello world!");
    db->putValue("B", "Good bye world!");
    db->putValue("C", "I am hungry");

    delete(db);
    return 0;
```


- For “Hello World!” the Entry number was 0x0001
- For “Good bye world!” the Entry number was 0x0002
- For “I am hungry” the Entry number was 0x0003

WAL Example for Put – Mystery



.log file


```
int main (void) {
    dbTest* db = new dbTest();

    db->putValue("A", "Hello world!");
    db->putValue("B", "Good bye world!");
    db->putValue("C", "I am hungry");

    delete(db);
    return 0;
```

- Seems like batch information.
- But needs confirmation.

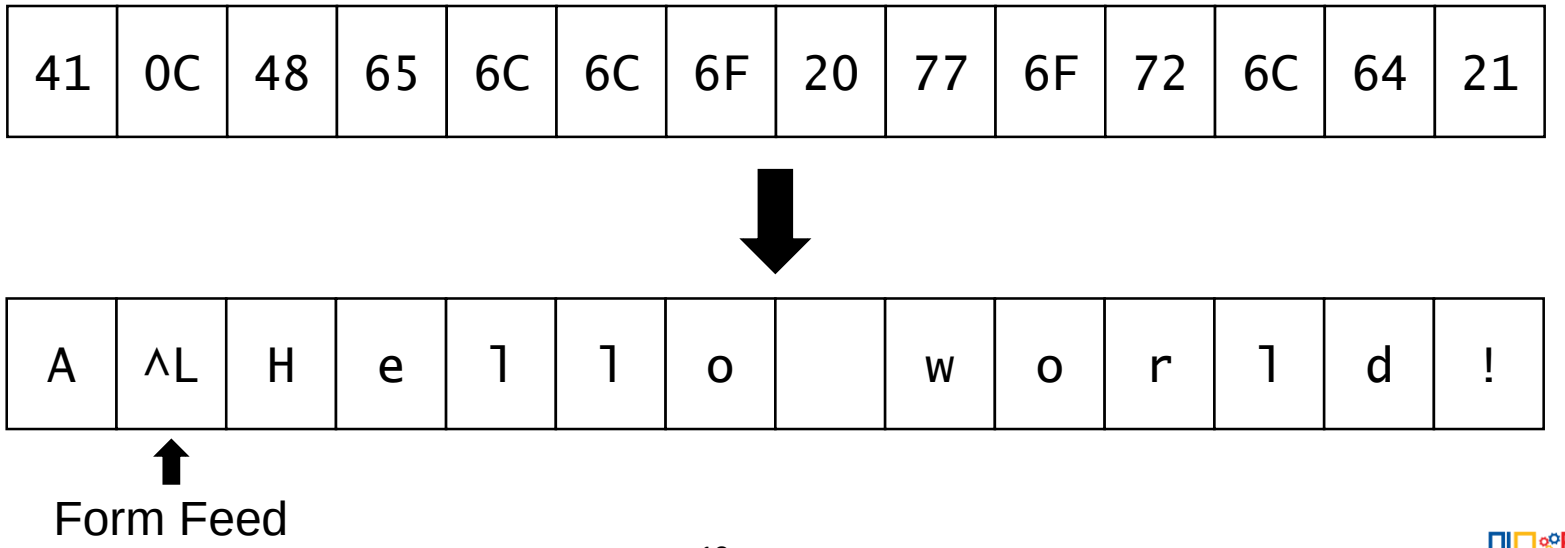
WAL Example file for Put - Data

- Tested ‘put’ for {"A”: "Hello world!”}

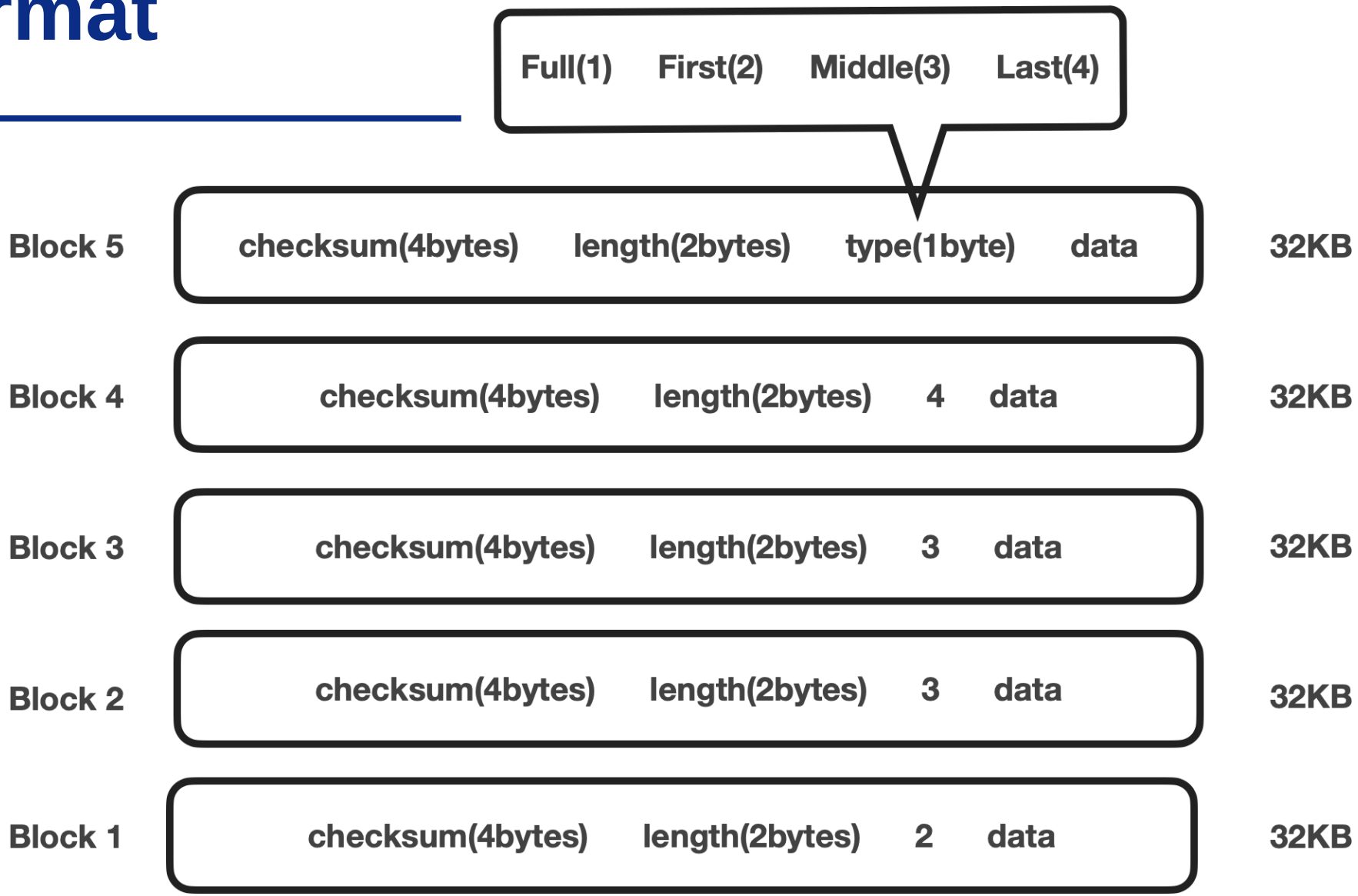
```
0000h: C0 55 16 4E 1C 00 01 01 00 00 00 00 00 00 00 01  ÀU.N............
0010h: 00 00 00 01 01 41 0C 48 65 6C 6C 6F 20 77 6F 72  .....A.Hello wor
0020h: 6C 64 21 8E C7 D9 9D 1F 00 01 02 00 00 00 00 00  ld!ŽÇÙ..........
0030h: 00 00 01 00 00 00 01 01 42 0F 47 6F 6F 64 20 62  ........B.Good b
0040h: 79 65 20 77 6F 72 6C 64 21 26 5D 13 F3 1B 00 01  ye world!&].ó...
0050h: 03 00 00 00 00 00 00 00 01 00 00 00 81 01 43 0B  ..............C.
0060h: 49 20 61 6D 20 68 75 6E 67 72 79                  I am hungry
```


.log file

```
int main (void) {
    dbTest* db = new dbTest();

    db->putValue("A", "Hello world!");
    db->putValue("B", "Good bye world!");
    db->putValue("C", "I am hungry");

    delete(db);
    return 0;
```



Log format


Full(1) First(2) Middle(3) Last(4)
Block 5
checksum(4bytes) length(2bytes) type(1byte) data
32KB
Block 4
checksum(4bytes) length(2bytes) 4 data
32KB
Block 3
checksum(4bytes) length(2bytes) 3 data
32KB
Block 2
checksum(4bytes) length(2bytes) 3 data
32KB
Block 1
checksum(4bytes) length(2bytes) 2 data
32KB

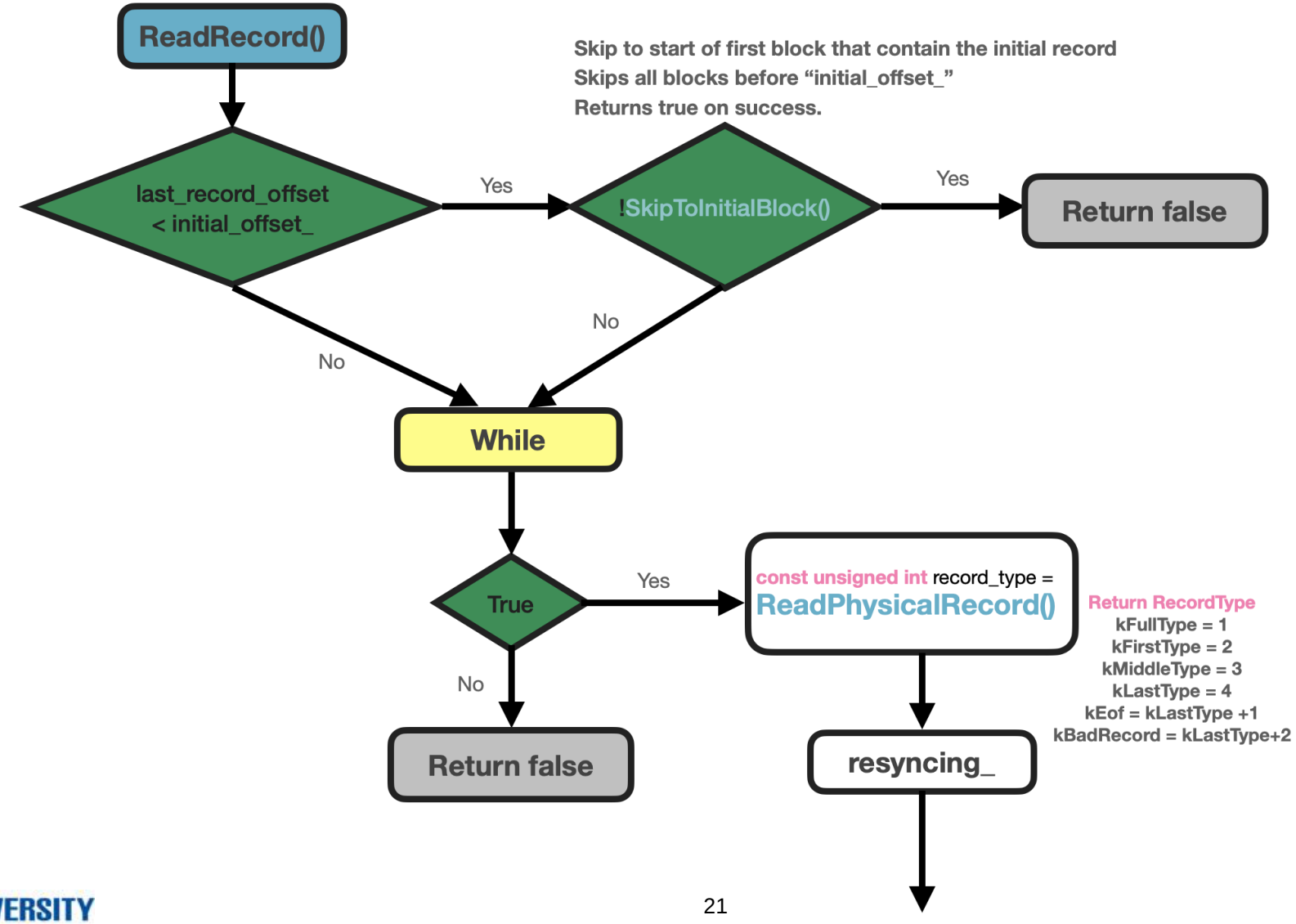
Code flow - log::reader

```
Class Reader
      Class Reporter
      Reader(SequentialFile* file, Reporter* reporter, bool checksum,
             uint64_t initial_offset)
   : file_(file),
     reporter_(reporter),
     checksum_(checksum),
     backing_store_(new char[kBlockSize]),
     buffer_(),
     eof_(false),
     last_record_offset_(0),
     end_of_buffer_offset_(0),
     initial_offset_(initial_offset),
     resyncing_(initial_offset > 0) {}
```

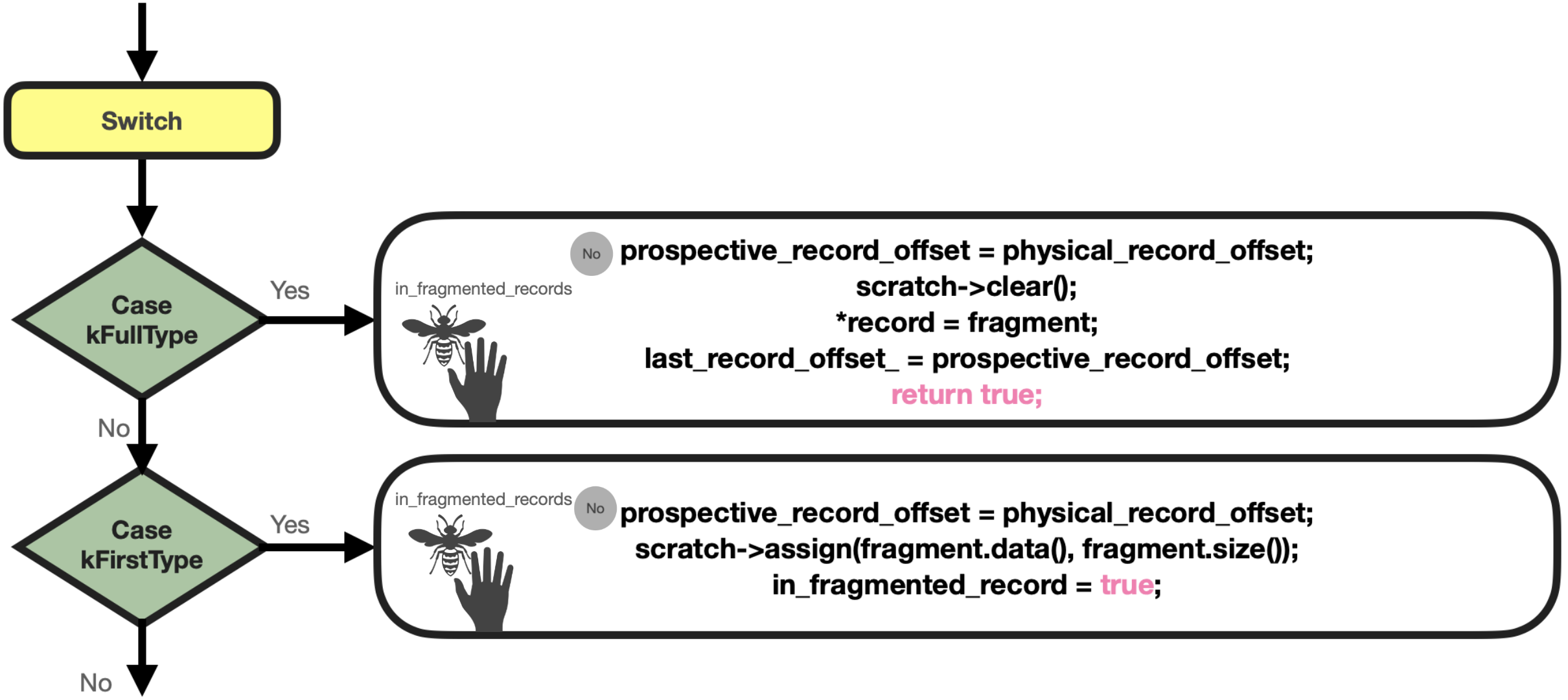
functions:

```
bool Reader::ReadRecord(Slice* record, std::string* scratch)
bool Reader::SkipToInitialBlock()
unsigned int Reader::ReadPhysicalRecord(Slice* result)
uint64_t Reader::LastRecordOffset() { return last_record_offset_; }
void Reader::ReportDrop(uint64_t bytes, const Status& reason)
void Reader::ReportCorruption(uint64_t bytes, const char* reason)
```

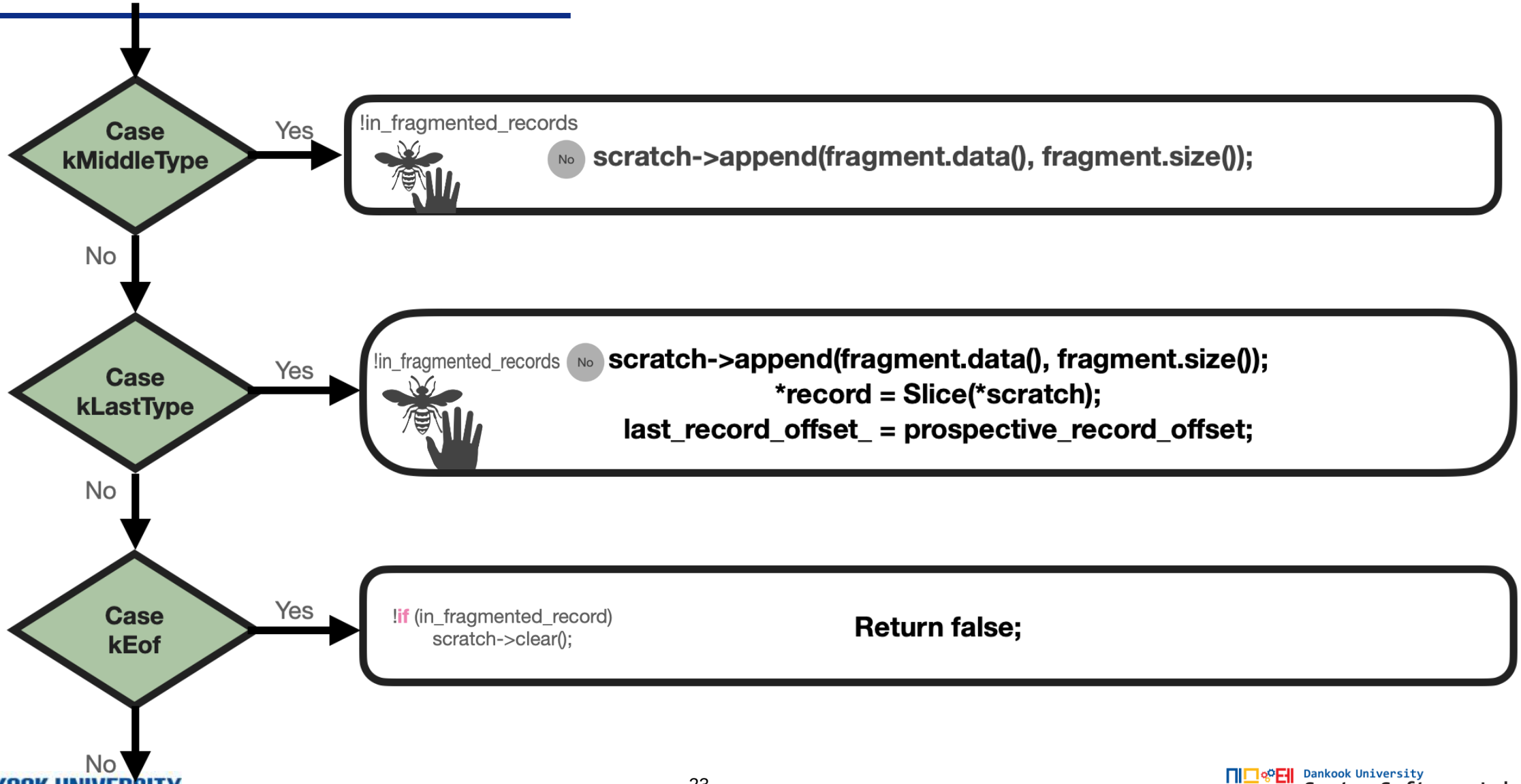
Code flow - log::reader



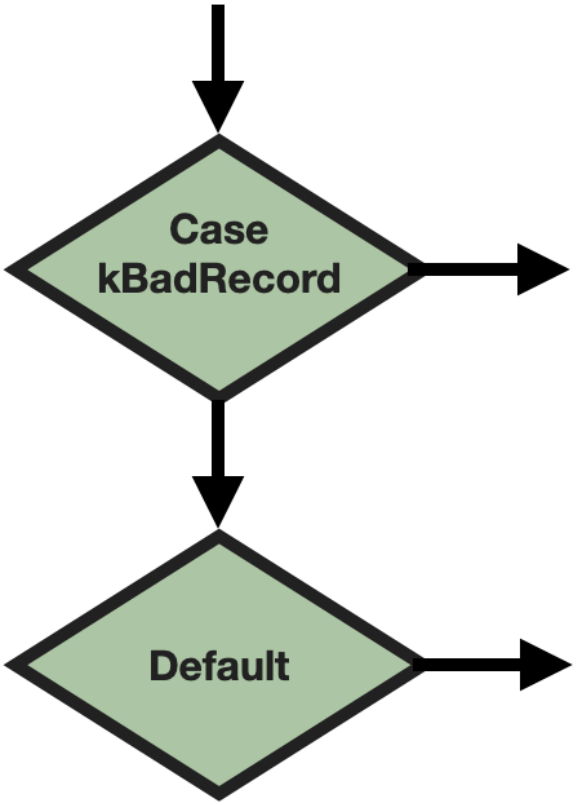
Code flow - log::reader



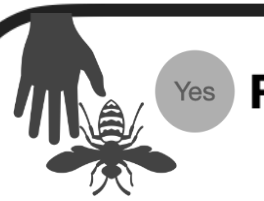
Code flow - log::reader



Code flow - log::reader



* The record has an invalid CRC (ReadPhysicalRecord reports a drop)
* The record is a 0-length record (No drop is reported)
* The record is below constructor's initial_offset (No drop is reported)




```
ReportCorruption(scratch->size(), "error in middle of record");
in_fragmented_record = false;
scratch->clear();
```

```
char buf[40];
std::snprintf(buf, sizeof(buf), "unknown record type %u", record_type);
ReportCorruption((fragment.size() + (in_fragmented_record ? scratch->size() : 0)), buf);
in_fragmented_record = false;
scratch->clear();
```

Question