6 BCF specification

VCF is very expressive, accommodates multiple samples, and is widely used in the community. Its biggest drawback is that it is big and slow. Files are text and therefore require a lot of space on disk. A normal batch of a hundred exomes is a few ~~GB~~gigabytes, but large-scale VCFs with thousands of exome samples quickly become hundreds of ~~GBs~~gigabytes. Because the file is text, it is extremely slow to parse.

Overall, the idea behind is BCF2 is simple. BCF2 is a binary, compressed equivalent of VCF that can be indexed with tabix and can be efficiently decoded from disk or streams. For efficiency reasons BCF2 only supports a subset of VCF, in that all info and genotype fields must have their full types specified. That is, BCF2 requires that if e.g. an info field `AC` is present then it must contain an equivalent VCF header line noting that `AC` is an allele indexed array of type integer.

6.1 Overall file organization

A BCF2 file is composed of a mandatory header, followed by a series of BGZF compressed blocks of binary BCF2 records. The BGZF blocks allow BCF2 files to be indexed with tabix.

BGZF blocks are composed of a VCF header with a few additional records and a block of records. Following the last BGZF BCF2 record block is an empty BGZF block (a block containing zero type of data), indicating that the records are done.

A BCF2 header follows exactly the specification as VCF, with a few extensions/restrictions:

- All BCF2 files must have fully specified contigs definitions. No record may refer to a contig not present in the header itself.
- All INFO and GENOTYPE fields must be fully typed in the BCF2 header to enable type-specific encoding of the fields in records. An error must be thrown when converting a VCF to BCF2 when an unknown or not fully specified field is encountered in the records.

6.2 Header

The BCF2 header contains the following items:

Field	Type	Notes
magic	char[3]	The characters "`BCF`"
major_version	uint8_t	2
minor_version	uint8_t	2
l_text	uint32_t	Length of the "text" field, including the terminating NUL character
text	char[l_text]	VCF format header text, NUL-terminated

The "magic" field and version numbers can be used to quickly examine the file to determine that it's a BCF2.2 file. The "text" field contains the standard VCF header lines in text format, from `##fileformat=VCFv4.4` to `#CHROM ...` inclusive, terminated by a NUL character.

Because the type is encoded directly in the header, the recommended extension for BCF2 formatted files is *.bcf*. BCF2 supports encoding values in a dictionary of strings. The string map is provided by the keyword `##dictionary=S0,S1,...,SN` as a comma-separate ordered list of strings. See the "Dictionary of strings" section for more details.

6.2.1 Dictionary of strings

Throughout the BCF file most string values are be specified by integer reference to their dictionary values. For example, the following VCF record:

```
##INFO=<ID=ASP,Number=0,Type=Flag,Description="X">
##INFO=<ID=RSPOS,Number=1,Type=Integer,Description="Y">
##INFO=<ID=dbSNPBuildID,Number=1,Type=Integer,Description="Z">
##contig=<ID=20,length=62435964,assembly=B36,md5=f126cdf8a6e0c7f379d618ff66beb2da,species="Homo sapiens">
#CHROM POS ID REF ALT QUAL FILTER INFO
20 10144 rs144773400 TA T . PASS ASP;RSPOS=10145,dbSNPBuildID=134
20 10228 rs143255646 TA T . PASS ASP;RSPOS=10229;dbSNPBuildID=134
```