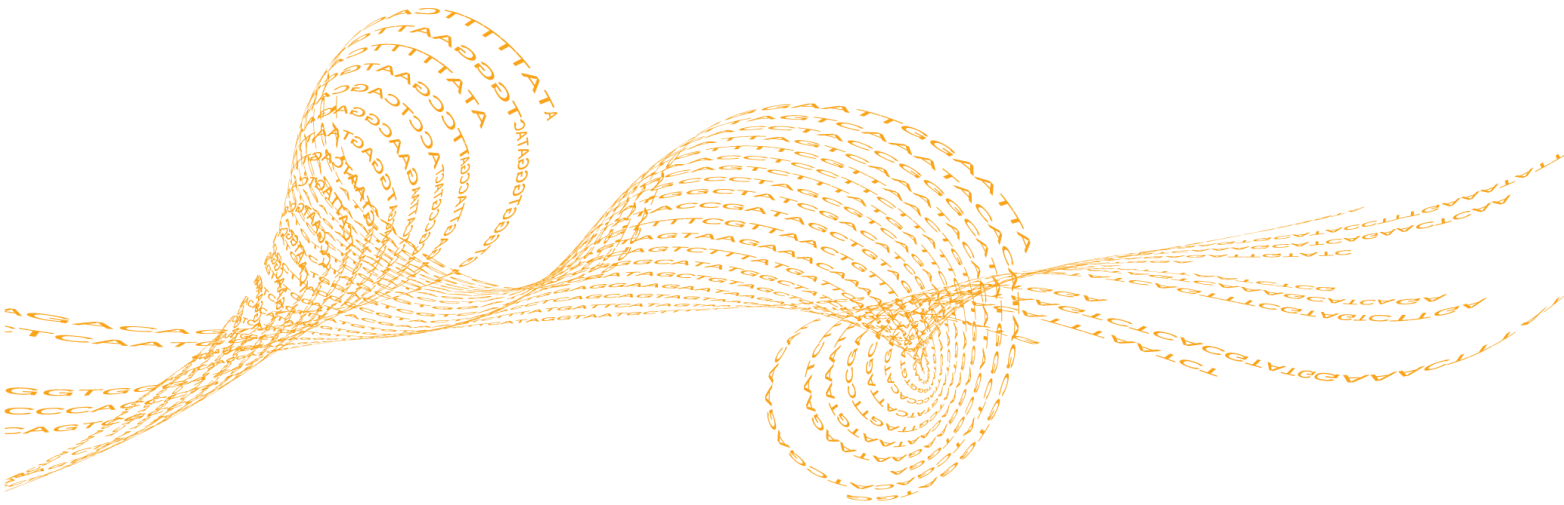
bcl2fastq2 Conversion Software Guide

Version 2.17 for MiSeq®, HiSeq®, NextSeq®, and HiSeq® X Systems

For Research Use Only. Not for use in diagnostic procedures.



ILLUMINA PROPRIETARY
15051736 Rev. G
July 2015

illumina®

Revision History

Part #	Revision	Date	Description of Change
15051736	G	July 2015	Update to software requirements, gcc version.
15051736	F	June 2015	Updates to support bcl2fastq2 v2.17.

Introduction

Illumina sequencing instruments generate per-cycle BCL base call files as primary sequencing output, but many downstream analysis applications use per-read FASTQ files as input. bcl2fastq2 Conversion Software v2.17 combines these BCL files from a run and converts them into FASTQ files.

At the same time as converting, the software separates reads from multiplexed samples (demultiplexing). Multiplexed sequencing allows you to run multiple individual samples in one lane (on a HiSeq system) or sample pool (like a single flow cell on a NextSeq, MiSeq, or HiSeq Rapid Run system). Index sequences that were attached to the template during sample prep identify the samples. The multiplexed reads are assigned to samples based on a user-generated sample sheet or on the Prep tab in BaseSpace. Then the reads are written to corresponding FASTQ files (see also *Generating the Sample Sheet* on page 22). If no sample sheet is provided, all reads are saved in Undetermined_S0_ FASTQ files.

Supported Version

This documentation supports bcl2fastq2 Conversion Software v2.17, intended for use with MiSeq, HiSeq 4000, 3000, 2500, and 2000, NextSeq, and HiSeq X Systems.

If you have an Illumina sequencing system running a version of Real-Time Analysis (RTA) software earlier than v1.18.54 and want to convert BCL to FASTQ, install bcl2fastq v1.8.4, and refer to the *bcl2fastq Conversion User Guide Version v1.8.4 (part # 15038058)* for instructions.

BCL Conversion/Demultiplexing Directory Structure

bcl2fastq2 Conversion Software v2.17 performs BCL conversion and demultiplexing in a single step, and puts the resulting demultiplexed compressed FASTQ files in <run folder>/Data/Intensities/BaseCalls by default.

The software places reads with undetermined indexes in files whose names start with Undetermined_S0_, unless the sample sheet specifies a `Sample_ID` or `Sample_Name` for reads without index.

If the `Sample_Project` column is specified for a sample in the sample sheet, FASTQ files for that sample are placed in <run folder>/Data/Intensities/BaseCalls/<Project>. Multiple samples can use the same project directory. If the `Sample_ID` and `Sample_Name` columns are both specified and do not match, the FASTQ files are placed in an additional subdirectory called <SampleId>.

BCL to FASTQ Conversion Process

The main task of bcl2fastq is to convert the base calls in the per-cycle BCL files to the per-read FASTQ format. Usually, this task involves a simple conversion for each base. Also, there are 2 other optional modifications to the reads: adapter trimming and removing the Unique Molecular Identifier (UMI) bases.

Adapter Trimming

During the conversion step, you can also perform adapter masking and trimming. With this feature, bcl2fastq2 Conversion Software v2.17 checks whether a read extends past the sample DNA insert and into adapter sequence. An approximate string matching algorithm is used to identify all or part of the adapter, treating insertions and deletions

as a single mismatch[1]. If an adapter sequence is detected, base calls matching the adapter and beyond the match are masked or removed in the resultant FASTQ file.

1 G. Myers (1999) A Fast Bit-Vector Algorithm for Approximate String Matching Based on Dynamic Programming, Journal of the ACM, 46(3): 395–415.

Unique Molecular Identifiers (UMIs)

UMIs are random k-mers attached to the genomic DNA before PCR amplification. The UMI is amplified with the amplicons, which later allows for detection of PCR duplicates and correction of amplification errors. bcl2fastq2 Conversion Software v2.17 removes these bases and places them into the read name in the FASTQ files.

Demultiplexing Method

Demultiplexing involves reorganizing the FASTQ files based on the index information, and generating the statistics and reporting files. Take care during sample preparation to avoid choosing indexes that differ by less than 3 bases.

CAUTION

By default the software tolerates a single mismatch per index when identifying per-sample reads during demultiplexing, requiring that sets of combined indexes are different from each other by > 2 bases. Where mismatch tolerance is used, and indexes used in the same lane or pool are not sufficiently different, a ***barcode collision*** can occur. In such a case a fatal error is reported and processing halted. The solution to this issue is to set the `--barcode-mismatches` parameter to a value of 0.

This section describes the demultiplexing steps.

Reorganizing FASTQ Files

The first step of demultiplexing is reorganizing the base call files, based on the index sequence. This step is done as follows for each cluster:

1 Get the raw index for each Index Read from the BCL file.

2 Identify the sample for the matching index based on the sample sheet.

3 [Optional] Detect and correct up to two errors on the barcode, and identify the appropriate sample. If there are multiple Index Reads, detect and correct up to two errors in each Index Read.

4 [Optional] Detect the presence of adapter sequence at the end of read. If adapter sequence is detected, trim or mask (with N) the corresponding base calls.

5 [Optional] Remove the UMI sequence from the beginning of the reads.

6 Append the read to the appropriate new FASTQ file for each read.

7 If the index cannot be identified, the data are written into an Undetermined sample file, unless the sample sheet specifies a sample for reads without index.

Updating Statistics and Reporting

The sample demultiplexer also generates statistics. While generating the demultiplexed FASTQ files, bcl2fastq2 Conversion Software v2.17 recalculates the base calling analysis statistics that were originally computed for the full lanes. These files are stored in the InterOp folder, and can be viewed with the Sequencing Analysis Viewer (SAV) software from Illumina.

The software also generates demultiplexing and conversion stat files, and an HTML report. For more information, see *ConversionStats.xml* on page 33, *DemultiplexingStats.xml* on page 33, and *HTML Report* on page 34.

Installing bcl2fastq2 Conversion Software v2.17

This version of bcl2fastq2 Conversion Software v2.17 is provided via download from illumina.com.

For installation requirements, see *Appendix: Requirements* on page 36.

Installing from RPM Package

Root access to the system is a prerequisite for this installation procedure. The command line for installing the RPM file is as follows:

```
yum install -y <rpm package-name>
```

The starting point for the bcl2fastq converter is the binary executable `/usr/local/bin/bcl2fastq`.

To install the rpm package in a user specified location, use the following command line:

```
rpm --install --prefix <user specified directory>
   <rpm package-name>
```

Installing from Source

The installation uses the directory locations specified by the following environment variables:

- `SOURCE`: Location of the bcl2fastq2 source code
- `BUILD`: Location of the build directory
- `INSTALL_DIR`: Location where the executable is installed

For example, these environment variables could be set as:

```
export TMP=/tmp
export SOURCE=${TMP}/bcl2fastq
export BUILD=${TMP}/bcl2fastq2-v2.17.1.x-build
export INSTALL_DIR=/usr/local/bcl2fastq2-v2.17.1.x
```

NOTE
The build directory must be different from the source directory.

The install procedure follows the usual steps: decompressing and extracting the source, configuring the build, building the package, and installing:

1 Decompress and extract the source code:

```
cd ${TMP}
tar -xvzf path-to-tarball/bcl2fastq2-v2.17.10.x.tar.gz
```

This command creates a bcl2fastq subdirectory in the ${TMP} directory.

2 Configure the build:

```
mkdir ${BUILD}
cd ${BUILD}
${SOURCE}/src/configure --prefix=${INSTALL_DIR}
```

These commands create a build directory, move to that directory, and then run configure in that directory.

The parameter `--prefix` provides the absolute path to the install directory.

The command creates a subdirectory in the ${TMP} directory.

3 Build the package:

```
make
```

4 Install:

```
make install
```

NOTE

This step can require root privilege, depending on the ${INSTALL_DIR} directory.

BCL Conversion Input Files

This section describes the input bcl2fastq2 Conversion Software v2.17 requires for each instrument type, and the BaseCalls directory required for demultiplexing.

Folder and File Naming

The top-level run folder name is generated using three fields to identify the <ExperimentName>, separated by underscores. Example:

```
YYMMDD_machinename_NNNN
```

Do not deviate from the run folder naming convention. Deviating from the convention can cause the software to stop.

1. The first field is a six-digit number specifying the date of the run. The YYMMDD ordering ensures that a numerical sorting of run folders places the names in chronological order.
2. The second field specifies the name of the sequencing machine. It can consist of any combination of upper or lower case letters, digits, or hyphens, but ***cannot*** contain any other characters (especially not an underscore). It is assumed that the sequencing platform is synonymous with the PC controlling it, and that the names assigned to the instruments are unique across the sequencing facility.
3. The third field is a four-digit counter specifying the experiment ID on that instrument. Each instrument supplies a series of consecutively numbered experiment IDs (incremental unique index) from the on-board sample tracking database or a LIMS.

NOTE
It is desirable to keep Experiment-IDs (or Sample-ID) and instrument names unique within any given enterprise. Establish a convention under which each machine is able to allocate run folder names independently of other machines to avoid naming conflicts.

A run folder named 150108_instrument1_0147 indicates experiment number 147, run on instrument 1, on the eighth of January 2015. The date and instrument name specify a unique run folder for any number of instruments. The addition of an experiment ID ensures both uniqueness and the ability to relate the contents of the run folder back to a laboratory notebook or LIMS.

More information is captured in the run folder name in fields separated by an underscore from the first three fields. For example, you can capture the flow cell number in the run folder name as follows: YYMMDD_machinename_NNNN_FCYYY.

NOTE
When publishing the data to a public database, it is desirable to extend the exclusivity globally, for instance by prefixing each machine with the identity of the sequencing center.

NextSeq Input Files

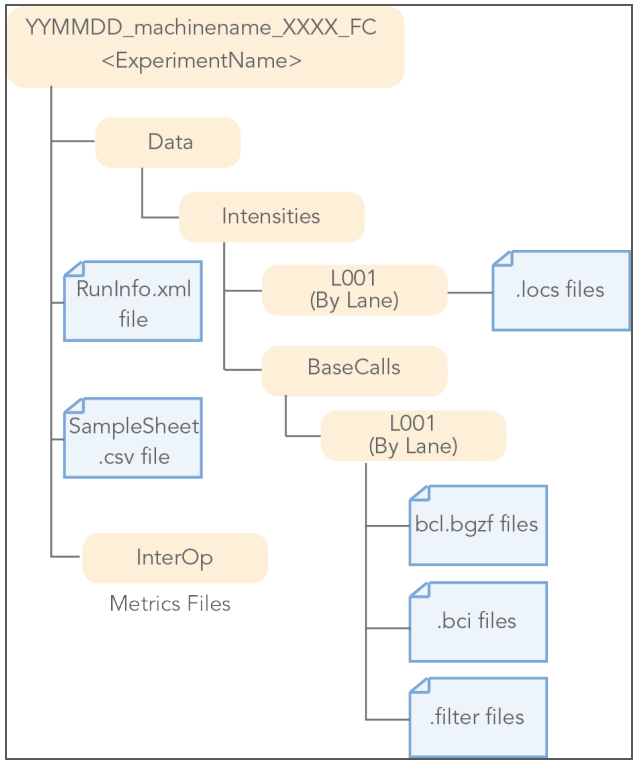
Demultiplexing requires a BaseCalls directory containing the binary base call files (BCL files) as generated by NextSeq. A sample sheet is optional, see *Sample Sheet* on page 20 for more information.

The BCL to FASTQ converter needs the following input files from the NextSeq BaseCalls directory:

- BCL files (*.bcl.bgzf)
- *.bci files
- *.filter files
- *.locs files
- RunInfo.xml file. The RunInfo.xml is at the top level of the run folder.
- [Optional] SampleSheet.csv. The sample sheet should be at the top level of the run folder and must be named SampleSheet.csv. You can specify different behavior using the `--sample-sheet` parameter (see *Options for BCL Conversion and Demultiplexing* on page 24).

The RTA software, v2 or later, is configured to copy the required files off the instrument computer to the BaseCalls directory on the analysis server specified during run setup. When in use, copy the sample sheet over manually.

Figure 1 BCL Conversion Input Files from the NextSeq System



BCL Files

The BCL files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L<lane>
```

They are named as follows:

```
<tile>.bcl.bgzf
```

The BCL files are compressed using the blocked GNU zip format (*.bgzf). The BCL files are binary base call files with the following format:

Bytes	Description	Data type
Bytes 0–3	Number N of cluster	Unsigned 32 bits little endian integer
Bytes 4–(N+3) Where N is the cluster index	Bits 0–1 are the bases, respectively [A, C, G, T] for [0, 1, 2, 3]: bits 2–7 are shifted by 2 bits and contain the quality score. All bits '0' in a byte is reserved for no-call.	Unsigned 8 bits integer

BCI Files

BCI files contain tile information for the sequencing run in binary format. The *.bci files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L<lane>
```

BCI files are named as follows:

```
s_<Lane>.bci
```

BCI files contain one record per tile, which uses the following format:

- bytes 0–3: tile number
- bytes 4–7: number of clusters in the tile.

Cluster numbers of one *.bci file sum to a cluster number listed in the beginning of each *.bcl.bgzf file of that lane.

FILTER Files

The FILTER files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L<lane>
```

They are named as follows:

```
s_<lane>.filter
```

The *.filter files are binary files containing filter results in the following format:

Bytes	Description
Bytes 0–3	Zero value (for backwards compatibility)
Bytes 4–7	Filter format version number
Bytes 8–11	Number of clusters
Bytes 12–(N+11) Where N is the cluster number	Unsigned 8 bits integer: • Bit 0 is pass or failed filter

Locs Files

The BCL to FASTQ converter uses *.locs files for position information. The Locs files can be found in the Intensities/L<lane> directories.

RunInfo.xml File

The top-level run folder contains a RunInfo.xml file. The file RunInfo.xml (normally generated by the instrument control software) identifies the boundaries of the reads (including index reads).

The XML tags in the RunInfo.xml file contain information on Run, Flow cell, and Instrument IDs, plus date and read structure. Information includes how many reads, how many cycles per read, and whether the reads are index reads.

HiSeq X and HiSeq 4000/3000 Input Files

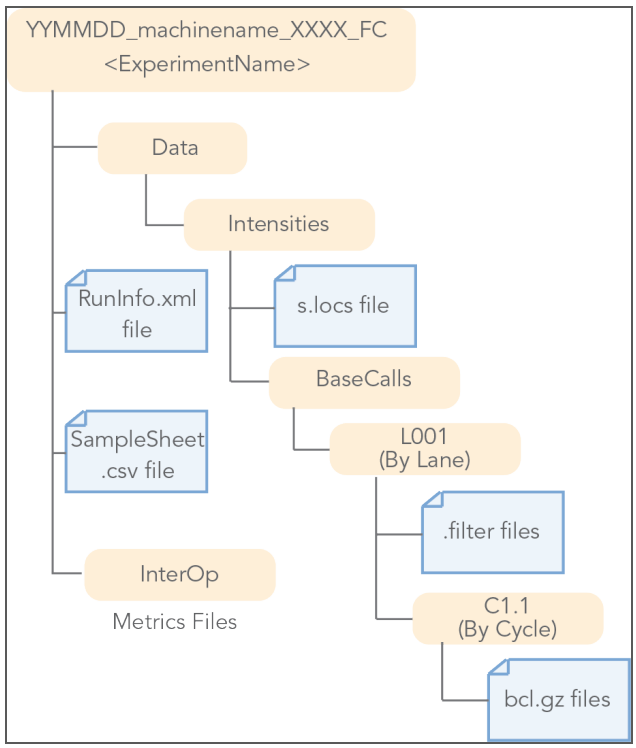
Demultiplexing requires a BaseCalls directory containing the binary base call files (BCL files) as generated by the HiSeq X, HiSeq 4000, or HiSeq 3000 system. A sample sheet is optional, see *Sample Sheet* on page 20 for more information.

The BCL to FASTQ converter needs the following input files from the BaseCalls directory:

- BCL files (*.bcl.gz)
- *.filter files
- s.locs file
- RunInfo.xml file. The RunInfo.xml is at the top level of the run folder.
- [Optional] SampleSheet.csv. The sample sheet should be at the top level of the run folder and must be named SampleSheet.csv. You can specify different behavior using the `--sample-sheet` parameter (see *Options for BCL Conversion and Demultiplexing* on page 24).

The RTA software, v2 or later, is configured to copy the required files off the instrument control computer to the BaseCalls directory on the analysis server specified during run setup. Copy the sample sheet over manually.

Figure 2 BCL Conversion Input Files from the HiSeq X System



BCL Files

The BCL files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L<lane>/C<Cycle>.1
```

They are named as follows:

```
s_<lane>_<tile>.bcl.gz
```

The BCL files are compressed using the gzip format (*.gz). The BCL files are binary base call files with the following format:

Bytes	Description	Data type
Bytes 0–3	Number N of cluster	Unsigned 32 bits little endian integer
Bytes 4–(N+3) Where N is the cluster index	Bits 0–1 are the bases, respectively [A, C, G, T] for [0, 1, 2, 3]: bits 2–7 are shifted by 2 bits and contain the quality score. All bits '0' in a byte is reserved for no-call.	Unsigned 8 bits integer

FILTER Files

The FILTER files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L<lane>
```

They are named as follows:

```
s_<lane>_<tile>.filter
```

The *.filter files are binary files containing filter results in the following format:

Bytes	Description
Bytes 0–3	Zero value (for backwards compatibility)
Bytes 4–7	Filter format version number
Bytes 8–11	Number of clusters
Bytes 12–(N+11) Where N is the cluster number	Unsigned 8 bits integer: • Bit 0 is pass or failed filter

Locs File

The BCL to FASTQ converter uses a single s.locs file for position information from the HiSeq X system. The locs file can be found in the Intensities directory.

RunInfo.xml File

The top-level run folder contains a RunInfo.xml file. The file RunInfo.xml (normally generated by the instrument control software) identifies the boundaries of the reads (including index reads).

The XML tags in the RunInfo.xml file contain information on Run, Flow cell, and Instrument IDs, plus date and read structure. Information includes how many reads, how many cycles per read, and whether the reads are index reads.

MiSeq and HiSeq 2500/2000 Input Files

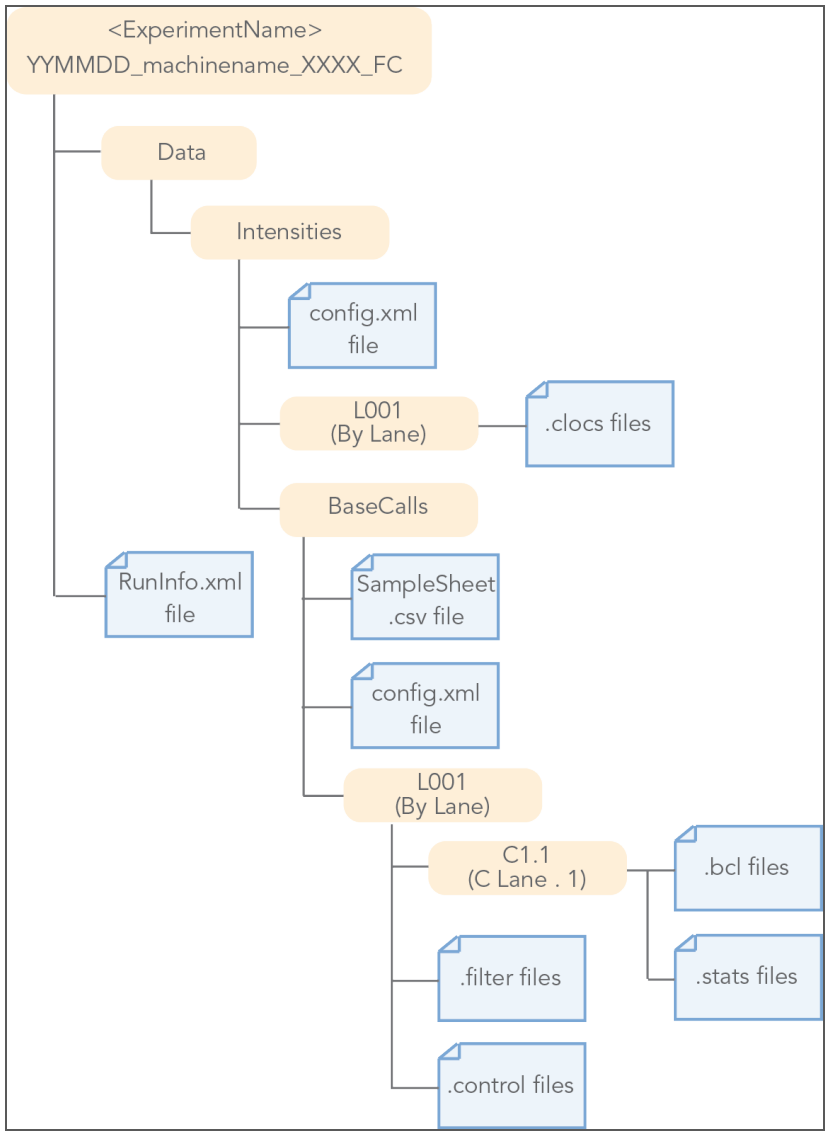
Demultiplexing requires a BaseCalls directory containing the binary base call files (BCL files) as generated by MiSeq or HiSeq 2500/2000.

The BCL to FASTQ converter needs the following input files from the MiSeq or HiSeq 2500/2000 BaseCalls directory:

- BCL files (*.bcl.gz for HiSeq v4, or where compressed BCL is selected)
- *.stats files
- *.filter files
- *.control files
- *.clocs, *.locs, or *_pos.txt files. The BCL to FASTQ converter determines which type of position file to look for based on the RTA version that was used to generate them.
- RunInfo.xml file. The RunInfo.xml is at the top level of the run folder.
- config.xml file
- [Optional] SampleSheet.csv. Place the sample sheet at the top level of the run folder and make sure that the file name is SampleSheet.csv. You can specify a different file behavior using the `--sample-sheet` parameter (see *Options for BCL Conversion and Demultiplexing* on page 24).

RTA is configured to copy the required files off the instrument computer machine to the BaseCalls directory on the analysis server specified during run setup. Copy the sample sheet over manually.

Figure 3 BCL Conversion Input Files from the MiSeq or HiSeq 2000/2500 System



BCL Files

The BCL files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L<lane>/C<Cycle>.1
```

They are named as follows:

```
s_<lane>_<tile>.bcl.gz
```

The BCL files are compressed using the gzip format (*.gz). The BCL files are binary base call files with the following format:

Bytes	Description	Data type
Bytes 0–3	Number N of cluster	Unsigned 32 bits little endian integer

Bytes	Description	Data type
Bytes 4–(N+3) Where N is the cluster index	Bits 0–1 are the bases, respectively [A, C, G, T] for [0, 1, 2, 3]: bits 2–7 are shifted by 2 bits and contain the quality score. All bits '0' in a byte is reserved for no-call.	Unsigned 8 bits integer

STATS Files

The stats files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L00<lane>/C<cycle>.1
```

They are named as follows:

```
s_<lane>_<tile>.stats
```

The Stats file is a binary file containing base calling statistics; the content is described in the following table. The data are for clusters passing filter only.

Start	Description	Data type
Byte 0	Cycle number	integer
Byte 4	Average Cycle Intensity	double
Byte 12	Average intensity for A over all clusters with intensity for A	double
Byte 20	Average intensity for C over all clusters with intensity for C	double
Byte 28	Average intensity for G over all clusters with intensity for G	double
Byte 36	Average intensity for T over all clusters with intensity for T	double
Byte 44	Average intensity for A over clusters with base call A	double
Byte 52	Average intensity for C over clusters with base call C	double
Byte 60	Average intensity for G over clusters with base call G	double
Byte 68	Average intensity for T over clusters with base call T	double
Byte 76	Number of clusters with base call A	integer
Byte 80	Number of clusters with base call C	integer
Byte 84	Number of clusters with base call G	integer
Byte 88	Number of clusters with base call T	integer
Byte 92	Number of clusters with base call X	integer
Byte 96	Number of clusters with intensity for A	integer
Byte 100	Number of clusters with intensity for C	integer
Byte 104	Number of clusters with intensity for G	integer
Byte 108	Number of clusters with intensity for T	integer

FILTER Files

The FILTER files can be found in the BaseCalls directory inside the run directory:

```
Data/Intensities/BaseCalls/L<lane>
```

They are named as follows:

```
s_<lane>_<tile>.filter
```

The *.filter files are binary files containing filter results in the following format:

Bytes	Description
Bytes 0–3	Zero value (for backwards compatibility)
Bytes 4–7	Filter format version number
Bytes 8–11	Number of clusters
Bytes 12–(N+11) Where N is the cluster number	Unsigned 8 bits integer: • Bit 0 is pass or failed filter

CONTROL Files

The control files can be found in the BaseCalls directory:

```
<run directory>/Data/Intensities/BaseCalls/L00<lane>/
```

They are named as follows:

```
s_<lane>_<tile>.control
```

The *.control files are binary files containing control results; the format is described in the following table.

Bytes	Description
Bytes 0–3	Zero value (for backwards compatibility)
Bytes 4–7	Format version number
Bytes 8–11	Number of clusters
Bytes 12–(2xN+11) Where N is the cluster index	The bits are used as follows: • Bit 0: always empty (0) • Bit 1: was the read identified as a control? • Bit 2: was the match ambiguous? • Bit 3: did the read match the PhiX tag? • Bit 4: did the read align to match the PhiX tag? • Bit 5: did the read match the control index sequence? • Bits 6,7: reserved for future use • Bits 8..15: the report key for the matched record in the controls.fasta file (specified by the REPORT_KEY metadata)

Position Files

The BCL to FASTQ converter can use different types of position files and expects a type based on the version of RTA used:

- *.locs: the locs files can be found in the Intensities/L<lane> directories.
- *.clocs: the clocs files are compressed versions of locs file and can be found in the Intensities/L<lane> directories.
- *_pos.txt: the POS files can be found in the Intensities directory.
  The *_pos.txt files are text files with 2 columns and a number of rows equal to the number of clusters. The first column is the X-coordinate and the second column is the Y-coordinate. Each line has a <cr><lf> at the end.

RunInfo.xml File

The top-level run folder contains a RunInfo.xml file. The file RunInfo.xml (normally generated by the instrument control software) identifies the boundaries of the reads (including index reads).

The XML tags in the RunInfo.xml file contain information on Run, Flow cell, and Instrument IDs, plus date and read structure. Information includes how many reads, how many cycles per read, and whether the reads are index reads.

config.xml Files

In the Intensities folder, you find the config.xml file that records any information specific to the generation of the subfolders. This file contains a tag-value list describing the cycle-image folders used to generate each folder of intensity and sequence files.

In the BaseCalls folder, there is another config.xml file containing the meta-information about the base caller runs. These files are different, though related in content.

Sample Sheet

The sample sheet (SampleSheet.csv) file details the relationship between samples and indexes specified during library creation. The sample sheet is at the top level of the run folder by default. A sample sheet is not essential; if no sample sheet is provided, all reads are assigned to the default sample Undetermined_S0, one file per lane per read. Sample sheets have four sections: Header, Reads, Settings, and Data. bcl2fastq2 Conversion Software v2.17 uses information from the [Settings] and [Data] sections.

Use the Illumina Experience Manager (IEM) version 1.9 or later to create sample sheets; see *Generating the Sample Sheet* on page 22 for more information.

Figure 4 Example of Valid Sample Sheet in Excel

[Header]								
IEMFileVersion	4							
Investigator Name	Isabelle							
Experiment Name	HiSeq X ten run							
Date	3/14/2014							
Workflow	GenerateFASTQ							
Application	HiSeq FASTQ Only							
Assay	TruSeq HT							
Description	none							
Chemistry	Default							
[Reads]								
151								
151								
[Settings]								
ReverseComplement	0							
Adapter	AGATCGGAAGAGCACACGTCTGAACTCCAGTCA							
AdapterRead2	AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT							
[Data]								
Lane	Sample_ID	Sample_Name	Sample_Plate	Sample_Well	I7_Index_ID	index	Sample_Project	Description
1	sample_ID1	test1			D701	ATTACTCG	HXten	
2	sample_ID2	test2			D712	AGCGATAG	Hxten	
3	sample_ID3	test3			D710	TCCGCGAA	Hxten	
4	sample_ID4	test4			D708	TAATGCGC	HXten	

Settings Section

bcl2fastq2 Conversion Software v2.17 uses adapter settings in the [Settings] section. If no adapter sequences are provided, the software does not complete the adapter trimming.

- Adapter Specifications

Setting	Value/Description
`Adapter` or `TrimAdapter`	The adapter sequence to be trimmed. If an AdapterRead2 is provided, this sequence is only used to trim Read 1.
`AdapterRead2` or `TrimAdapterRead2`	The adapter sequence to be trimmed in Read 2. If not provided, the same sequence specified in Adapter is used.
`MaskAdapter`	The adapter sequence to be masked rather than trimmed. If MaskAdapterRead2 is provided, this sequence is only used to mask Read 1.
`MaskAdapterRead2`	The adapter sequence to be masked in Read 2. If not provided, the same sequence specified in MaskAdapter is used.
`FindAdapterWithIndels`	1 (default) or 0. If 1 (true), an approximate string matching algorithm is used to identify the adapter, treating insertions and deletions as a single mismatch (Myers 1999, J.ACM). If 0 (false), a sliding window algorithm is used, in which insertions and deletions of bases inside the adapter sequence is not tolerated.

- Cycle and Tile Specifications

Setting	Value/Description
Read1EndWithCycle	The last cycle to use for Read 1.
Read2EndWithCycle	The last cycle to use for Read 2.
Read1StartFromCycle	The first cycle to use for Read 1.
Read2StartFromCycle	The first cycle to use for Read 2.
Read1UMILength	The length of the UMI used for Read 1.
Read2UMILength	The length of the UMI used for Read 2.
ExcludeTiles	Tiles to exclude. Separate tiles using a plus sign [+], or specified as a range with a hyphen [-]. For example: ExcludeTiles,1101+2201+1301-1306 means: skip tiles 1101, 2201, and 1301 through 1306.
ExcludeTilesLaneX	Tiles to exclude for Lane X. For example: ExcludeTilesLane6,1101-1108 means: skip tiles 1101 through 1108 for lane 6 only.

▸ FASTQ Specifications

Setting	Value/Description
CreateFastqForIndexReads	0 (default) or 1. If 1 (true), generate FASTQ files for index reads. Normally, these FASTQ files are not needed, because demultiplexing is carried out automatically based on the sample sheet. Also, the index sequence is already placed in the sequence identifiers in the FASTQ files.
ReverseComplement	0 (default) or 1. If 1 (true), all reads are reverse complemented as they are written to FASTQ files. This step is necessary in certain unusual cases (eg processing of mate-pair data using BWA, which expects paired-end data).

Data Section

bcl2fastq2 Conversion Software v2.17 uses the following sample sheet columns in the [Data] section.

Column	Description
Sample_Project	If specified, a directory with the specified name is created and FASTQ files are stored there. Multiple samples can use the same project.
Lane	If specified, FASTQ files are generated only for those samples with the specified lane number.
Sample_ID	ID of the sample
Sample_Name	Descriptive name of the sample
index	Index sequence
index2	Index sequence for index 2, if using dual indexing

NOTE
If the Sample_ID and Sample_Name columns are both specified and do not match, the FASTQ files are placed in an additional subdirectory called <SampleId>.

NOTE
Alphanumeric characters, hyphens [-], and underscores [_] are allowed in the Sample_Project, Sample_ID, and Sample_Name columns.

Reads are saved to the FASTQ files named after the samples specified in the sample sheet; see *FASTQ Files* on page 29.

NOTE
bcl2fastq2 Conversion Software v2.17 ignores any non-required columns that IEM includes.

Sample Sheet Demultiplexing Scenarios

BCL conversion and demultiplexing support four scenarios, as directed by the sample sheet:

- No sample sheet found: all reads are placed in the default Undetermined_S0 FASTQ files by lane.
- Sample sheet found, no [Data] section: all reads are placed in the default Undetermined_S0 FASTQ files by lane.
- Sample sheet found, one sample defined without indexes: all reads are placed in the sample FASTQ file that is defined in the sample sheet.
- Sample sheet found, samples defined with indexes:
  - Reads without a matching index are placed in the default Undetermined_S0 FASTQ files.
  - Reads with a valid index are placed in the sample FASTQ file as defined in the sample sheet.

For each sample, there is one file per lane per read number (if reads exist for that sample, lane, and read number).

NOTE
When the `Lane` column of the sample sheet [Data] section is populated, only those lanes are converted. When the `Lane` column is not used, all lanes are converted.

Generating the Sample Sheet

The user-generated sample sheet (SampleSheet.csv file) describes the samples and the indexes used. The sample sheet is at the top level of the run folder as default location and named SampleSheet.csv, unless explicitly specified by `--sample-sheet` option. See *Options for BCL Conversion and Demultiplexing* on page 24 for more information.

Illumina Experiment Manager

Use the Illumina Experiment Manager (IEM) version 1.9 or later to generate the sample sheet file. You can also use a text editor or Excel but IEM is recommended.

IEM is a wizard-driven application that guides you through the creation and setup of your sample sheet. IEM can be run on any Windows platform. You can download it from the Illumina website at www.illumina.com.

A sample sheet can be generated with IEM:

- For HiSeq X, 4000, 3000, 2500, and 2000 systems:
  a. Select the **HiSeq/HiScanSQ/GA** button, and then select the appropriate sequencing instrument model.
  b. Under **Select Application**, select **HiSeq FASTQ Only**.
- For MiSeq systems:
  a. Select **MiSeq**.
  b. Select **Other**, and then select **FASTQ Only**.
- For NextSeq systems:
  a. Select **NextSeq**.
  b. Under **Select Application**, select **NextSeq FASTQ Only**.

If you want to define settings that IEM does not provide, you can open and edit the sample sheet in Excel or any text editor. Examples are values for Adapter or AdapterRead2. Save as CSV file.

Illegal Characters

`Sample_ID` and `Sample_Name` field entries in the sample sheet cannot contain illegal characters not allowed by some file systems.

Allowed characters are alphanumeric characters, dashes, and underscores.

Examples of common characters not allowed are the **space character** and the following:

```
? ( ) [ ] / \ = + < > : ; " ' , * ^ | & .
```

Samples Without Index

It is possible to assign samples without index to `Sample_ID` or other identifiers by leaving the Index field empty.

Invoke bcl2fastq the following way:

```
nohup /usr/local/bin/bcl2fastq [options]
```

A commonly used option is `--runfolder-dir`, while `--output-dir` is optional. An example of a command with those options would be:

```
nohup /usr/local/bin/bcl2fastq --runfolder-dir <RunFolder>
   --output-dir <BaseCalls>
```

This command produces a set of FASTQ files in the BaseCalls directory. Reads with an unresolved or erroneous index are placed in the Undetermined_S0 FASTQ files. By default, `--runfolder-dir` is the current directory and `--output-dir` is the Data/Intensities/BaseCalls subdirectory of the run folder. With this structure, the bcl2fastq command can be issued from the run-folder.

NOTE

To generate a log file for a problematic bcl2fastq2 run, use the `-l` or `--min-log-level DEBUG` option. By default, bcl2fastq2 generates a log file with logging level `INFO`.

Options for BCL Conversion and Demultiplexing

The available command line options for bcl2fastq2 are listed in this section.

Main Options

The options used in basic bcl2fastq2 use cases are `--runfolder-dir` and `--output-dir`. For command line options that have a corresponding sample sheet setting, the value passed on the command line overwrites the value found in the sample sheet.

Option	Description
`-R, --runfolder-dir`	Path to run folder directory Default: ./
`-o, --output-dir`	Path to demultiplexed output Default: <runfolder-dir>/Data/Intensities/BaseCalls/

Advanced Options

These options are not required, and are not advised unless you require nondefault settings.

- Directory options

Option	Description
`-i, --input-dir`	Path to input directory Default: <runfolder-dir>/Data/Intensities/BaseCalls/
`--intensities-dir`	Path to intensities directory If intensities directory is specified, then the input directory must also be specified. Default: <input-dir>/../
`--interop-dir`	Path to demultiplexing statistics directory Default: <runfolder-dir>/InterOp/

Option	Description
`--stats-dir`	Path to human-readable demultiplexing statistics directory Default: <runfolder-dir>/Stats/
`--reports-dir`	Path to reporting directory Default: <runfolder-dir>/Reports/
`--sample-sheet`	Path to sample sheet, so you can specify the location and name of the sample sheet, if different from default. Default: <runfolder-dir>/SampleSheet.csv

- Processing options:

  If your computing platform supports threading, the software manages the threads by default as follows:

  - 4 threads for reading the data
  - 4 threads for writing the data
  - 20% for demultiplexing data
  - 100% for processing demultiplexed data

  The file i/o threads spend most of their time sleeping, and so take little processing time. The processing of demultiplexed data is allocated 1 thread per CPU to make sure that there are no idle CPUs, resulting in more threads than CPUs by default. You can use the following options to provide control on threading. If, for example, you share your computing resources with colleagues and wish to limit your usage, these options are useful.

Option	Description
`-r, --loading-threads`	Number of threads used for loading BCL data. Default depends on architecture.
`-d, --demultiplexing-threads`	Number of threads used for demultiplexing, Default depends on architecture.
`-p, --processing-threads`	Number of threads used for processing demultiplexed data. Default depends on architecture.
`-w, --writing-threads`	Number of threads used for writing FASTQ data. This number must not be higher than number of samples. Default depends on architecture.

If you want to use these options to assign multiple threads, consider the following:

- The most CPU demanding stage is the processing step (-p option). Assign this step the most threads.
- The second most CPU demanding stage is the demultiplexing step (-d option). Assign this step the second highest number of threads. Tests indicate 20% of processing time is used for demultiplexing a HiSeq X run.
- Reading and writing stages are lightweight and do not need many threads. This consideration is especially important for a local hard drive where too many threads mean too many parallel read write actions giving suboptimal performance.
- Use one thread per CPU core plus a little more to supply CPU with work. This method prevents CPUs being idle due to a thread being blocked while waiting for another thread.

- The number of threads depends on the data. If you specify more writing threads than samples, the extra threads do no work but can cost time due to context switching.

▸ Behavioral options

Option	Description
`--adapter-stringency`	The minimum match rate that would trigger the masking or trimming process. This value is calculated as MatchCount / (MatchCount + MismatchCount) and ranges from 0 to 1, but it is not recommended to use any value < 0.5, as this value would introduce too many false positives. The default value for this parameter is 0.9, meaning that only reads with > 90% sequence identity with the adapter are trimmed. Default: 0.9
`--aggregated-tiles`	This flag tells the converter about the structure of the input files. Accepted values: `AUTO` Automatically detects the tile setting `YES` Tiles are aggregated into single input file `NO` There are separate input files for individual tiles Default: AUTO
`--barcode-mismatches`	Number of allowed mismatches per index Multiple entries, comma delimited allowed. Each entry is applied to the corresponding index; last entry applies to all remaining indexes. Default: 1. Accepted values: 0, 1 or 2.
`--create-fastq-for-index-reads`	Create FASTQ files also for Index Reads
`--ignore-missing-bcls`	Missing or corrupt BCL files are ignored. Assumes 'N'/'#' for missing calls
`--ignore-missing-filter`	Missing or corrupt filter files are ignored. Assumes Passing Filter for all clusters in tiles where filter files are missing.
`--ignore-missing-positions`	Missing or corrupt positions files are ignored. If corresponding position files are missing, bcl2fastq writes unique coordinate positions in FASTQ header.
`--minimum-trimmed-read-length`	Minimum read length after adapter trimming. bcl2fastq trims the adapter from the read down to the value of this parameter. If there is more adapter match below this value, then those bases are masked, not trimmed (replaced by N rather than removed). Default: 35

Option	Description
`--mask-short-adapter-reads`	This option applies when a read is trimmed to below the length specified by the `--minimum-trimmed-read-length` option (default of 35). These parameters specify the following behavior: If the number of bases left after adapter trimming is less than `--minimum-trimmed-read-length`, force the read length to be equal to `--minimum-trimmed-read-length` by masking adapter bases (replace with Ns) that fall below this length. If the number of ACGT bases left after this process falls below `--mask-short-adapter-reads`, mask all bases, resulting in a read with `--minimum-trimmed-read-length` number of Ns. Default: 22

`--tiles`	The `--tiles` argument takes a regular expression to select for processing only a subset of the tiles available in the flow cell. This argument can be specified multiple times, one time for each regular expression. Examples:
	To select all the tiles ending with 5 in all lanes:
	`--tiles [0-9][0-9][0-9]5`
	To select tile 2 in lane 1 and all the tiles in the other lanes:
	`--tiles s_1_0002 --tiles s_[2-8]`

`--use-bases-mask`	The `--use-bases-mask` string specifies how to use each cycle. An `n` means ignore the cycle. A `Y` (or `y`) means use the cycle. An `I` means use the cycle for the Index Read. A number means that the previous character is repeated that many times. An asterisk [`*`] means that the previous character is repeated until the end of this read or index (length according to the RunInfo.xml). The read masks are separated with commas: `,` The format for dual indexing is as follows: `--use-bases-mask Y*,I*,I*,Y*` or variations thereof as specified. You can also specify the `--use-bases-mask` multiple times for separate lanes, like this way: `--use-bases-mask 1:y*,i*,i*,y* --use-bases-mask y*,n*,n*,y*` Where the `1:` means: Use this setting for lane 1. In this case, the second `--use-bases-mask` parameter is used for all other lanes. If this option is not specified, the mask is determined from the 'RunInfo.xml' file in the run directory. If it cannot do this determination, supply the `--use-bases-mask`.

Option	Description
`--with-failed-reads`	Include all clusters in the output, even clusters that are non-PF. These clusters would have been excluded by default.
`--write-fastq-reverse-complement`	Generate FASTQ files containing reverse complements of actual data.
`--no-bgzf-compression`	Turn off BGZF compression for FASTQ files. BGZF compression allows downstream applications to decompress in parallel. This parameter is only available in case a consumer of FASTQ data is not able to handle all standard GZIP formats.
`--fastq-compression-level`	Zlib compression level (1–9) used for FASTQ files. Default: 4
`--no-lane-splitting`	Do not split FASTQ files by lane.
`--find-adapters-with-sliding-window`	Find adapters with simple sliding window algorithm. Insertions and deletions of bases inside the adapter sequence are not handled.

NOTE

Do not use the `--no-lane-splitting` option if you wish to upload the resulting FASTQ files to BaseSpace. The FASTQ files generated from the `--no-lane-splitting` option are not compatible with the BaseSpace file uploader. Files generated without this option (the default setting) are compatible for upload to BaseSpace.

General options

Option	Description
`-h`, `--help`	Produce help message and exit
`-v`, `--version`	Print program version information
`-l`, `--min-log-level`	Minimum log level Recognized values: NONE, FATAL, ERROR, WARNING, INFO, DEBUG, TRACE To generate a log file for a problematic bcl2fastq2 run, use the `-l` or `--min-log-level DEBUG` option. Default: `INFO`

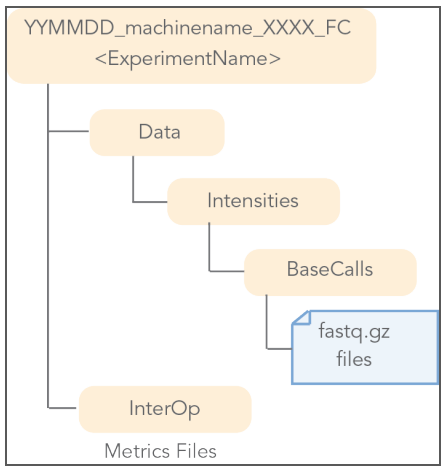
BCL Conversion Output Folder

The BCL Conversion output directory has the following characteristics:

- By default, the FASTQ output files are located in the Data/Intensities/BaseCalls directory; names of the files start with the sample name as derived from the sample sheet.
- The Undetermined files contain the reads with an unresolved or erroneous index.
- If no sample sheet exists, the software generates an Undetermined_S0 FASTQ file for each lane and read number.

For each sample, there is one FASTQ file per lane per read number (if reads exist for that sample, lane, and read number).

Figure 5 BCL Conversion Output



FASTQ Files

bcl2fastq2 Conversion Software v2.17 converts *.bcl, *.bcl.gz, and *.bcl.bgzf files into FASTQ files, which can be used as input for secondary analysis. The files are located in the Data/Intensities/BaseCalls directory. If no sample sheet is provided, the software generates one Undetermined_S0 FASTQ file for each lane and read number combination.

Naming

FASTQ files are named with the sample name and the sample number, which is a numeric assignment based on the order in the sample sheet that a sample ID first appeared in a given lane. Example:

Data/Intensities/BaseCalls/<SampleName>_S1_L001_R1_001.fastq.gz

- **<Sample_Name>**—The sample name provided in the sample sheet. If a sample name is not provided, the file name includes the sample ID, which is a required field in the sample sheet and must be unique within a lane.
- **S1**—The sample number based on the order in one lane that samples are first listed in the sample sheet starting with 1. In this example, S1 indicates that the sample ID is the first listed in the sample sheet. If this sample ID appears in another lane, the same sample number is used.

NOTE

Reads that cannot be assigned to any sample are written to a FASTQ file for sample number 0, and excluded from downstream analysis.

- **L001**—The lane number.
- **R1**—The read. In this example, R1 means Read 1. For a paired-end run, there is at least one file with R2 in the file name for Read 2. When generated, Index Reads are **I1** or **I2**.
- **001**—The last segment is always 001.

Compression

FASTQ files are saved in the compressed GNU zip format (an open source file compression program), indicated by the .gz file extension. By default, the BGZF variant of the GNU zip format is used. The BGZF variant facilitates parallel decompression of the FASTQ files by downstream applications. While BGZF is compliant with the GNU zip standard, if a downstream application cannot handle this variant, it can be turned off with the command line option --no-bgzf-compression.

Format

Each entry in a FASTQ file consists of 4 lines:

- Sequence identifier
- Sequence
- Quality score identifier line (consisting only of a +)
- Quality score

```
@<instrument>:<run number>:<flowcell ID>:<lane>:<tile>:<x-
   pos>:<y-pos>:<UMI> <read>:<is filtered>:<control
   number>:<index>
```

The following table describes the elements:

Element	Requirements	Description
@	@	Each sequence identifier line starts with @.
<instrument>	Characters allowed: a–z, A–Z, 0–9 and underscore	Instrument ID.
<run number>	Numerical	Run number on instrument.
<flowcell ID>	Characters allowed: a–z, A–Z, 0–9	
<lane>	Numerical	Lane number.
<tile>	Numerical	Tile number.
<x_pos>	Numerical	X coordinate of cluster.
<y_pos>	Numerical	Y coordinate of cluster.
<UMI>	Restricted characters: A/T/G/C/N	Optional, appears when UMI is specified in sample sheet. UMI sequences for Read 1 and Read 2, seperated by a plus [+].
<read>	Numerical	Read number. 1 can be single read or Read 2 of paired-end.
<is filtered>	Y or N	Y if the read is filtered (did not pass), N otherwise.
<control number>	Numerical	0 when none of the control bits are on, otherwise it is an even number. On HiSeq X and NextSeq systems, control specification is not performed and this number is always 0.
<index>	Restricted characters: A/T/G/C/N	Index of the read.

An example of a valid entry is as follows; note the space preceding the read number element:

```
@SIM:1:FCX:1:15:6329:1045:GATTACT+GTCTTAAC 1:N:0:ATCCGA
TCGCACTCAACGCCCTGCATATGACAAGACAGAATC
+
<>;##=><9=AAAAAAAAA9#:<#<;<<<????#=
```

Control Values

If the read is not identified as a control, then the 10th column (`<control number>`) is 0. If the read is identified as a control, the number is greater than 0, and the value specifies what type of control it is. The value is the decimal representation of a bit-wise encoding scheme. In that scheme bit 0 has a decimal value of 1, bit 1 a value of 2, bit 2 a value of 4, and so on.

Quality Scores

A quality score (or Q-score) expresses an error probability. In particular, it serves as a convenient and compact way to communicate very small error probabilities.

Given an assertion, A, the quality score, Q(A), expresses the probability that A is not true, P(~A), according to the relationship:

$$Q(A) = -10 \log_{10}(P(\sim A))$$

where *P(~A)* is the estimated probability of an assertion A being wrong.

The relationship between the quality score and error probability is demonstrated with the following table:

Quality score, Q (A)	Error probability, P (~A)
10	0.1
20	0.01
30	0.001

NOTE

On the systems we currently support, Q-scores are automatically binned. The specific binning applied depends on the current Q-table. See the white paper *Reducing Whole Genome Data Storage Footprint* for more information, available from www.illumina.com.

Quality Scores Encoding

In FASTQ files, quality scores are encoded into a compact form, which uses only 1 byte per quality value. In this encoding, the quality score is represented as the character with an ASCII code equal to its value + 33. The following table demonstrates the relationship between the encoding character, its ASCII code, and the quality score represented.

NOTE

When Q-score binning is in use, the subset of Q-scores applied by the bins is displayed.

Table 1 ASCII Characters Encoding Q-scores 0–40

Symbol	ASCII Code	Q-Score	Symbol	ASCII Code	Q-Score
!	33	0	6	54	21
"	34	1	7	55	22
#	35	2	8	56	23
$	36	3	9	57	24
%	37	4	:	58	25
&	38	5	;	59	26
'	39	6	<	60	27
(	40	7	=	61	28
)	41	8	>	62	29
*	42	9	?	63	30
+	43	10	@	64	31
,	44	11	A	65	32
-	45	12	B	66	33
.	46	13	C	67	34
/	47	14	D	68	35
0	48	15	E	69	36
1	49	16	F	70	37
2	50	17	G	71	38
3	51	18	H	72	39
4	52	19	I	73	40
5	53	20			

InterOp Files

The InterOp files can be found in the directory: <run directory>/InterOp. This directory contains binary files used by the Sequencing Analysis Viewer (SAV) software to summarize various analysis metrics such as cluster density, intensities, quality scores, and overall run quality.

The index metrics are stored in the file *IndexMetricsOut.bin*, which has the following binary format:

- Byte 0: file version (1)
- Bytes (variable length): record:
  - 2 bytes: lane number (unint16)
  - 2 bytes: tile number (unint16)
  - 2 bytes: read number (unint16)
  - 2 bytes: number of bytes Y for index name (unint16)
  - Y bytes: index name string (string in UTF8Encoding)

- 4 bytes: # clusters identified as index (uint32)
- 2 bytes: number of bytes V for sample name (unint16)
- V bytes: sample name string (string in UTF8Encoding)
- 2 bytes: number of bytes W for sample project (unint16)
- W bytes: sample project string (string in UTF8Encoding)

ConversionStats.xml

The ConversionStats.xml file can be found in <run directory>/Stats/, or in the directory specified by the `--stats-dir` option, if used. The file contains the following information:

- Per tile:
  - Raw Cluster Count
  - Read number
  - YieldQ30
  - Yield
  - QualityScore Sum
- Per lane:
  - Lane Number

DemultiplexingStats.xml

The DemultiplexingStats.xml file can be found in <run directory>/Stats/, or in the directory specified by the `--stats-dir` option. The file contains the following information per lane, barcode, sample, project, and flow cell:

- Barcode Count
- PerfectBarcode Count
- OneMismatchBarcodeCount

AdapterTrimming.txt

The AdapterTrimming.txt file is a statistic summary of adapter trimming for FASTQ file. The file is located in the <run directory>/Stats/ by default. If the `--stats-dir` option is used, the file is located in the specified directory.

The file contains the following information:

- Lane
- Read
- Project
- Sample ID
- Sample Name
- Sample Number
- TrimmedBases
- PercentageOfBased (being trimmed)

A table listing the fraction of reads with a given number of untrimmed bases exists for each sample, lane, and read number combination.

FastqSummaryF1L#.txt

The FastqSummaryF1L#.txt file (where # indicates the lane number) contains the number of raw and passed filter reads for each sample number and tile.

DemuxSummaryF1L#.txt

The DemuxSummaryF1L#.txt file (where # indicates the lane number) contains the percentage of each tile that each sample makes up. It also contains a list of the 1,000 most common unknown barcode sequences.

HTML Report

Human-readable HTML reports are generated from data in the DemultiplexingStats.xml and ConversionStats.xml files. This report is located in <run directory>/Reports/html/, or in the directory specified by the `--reports-dir` option, if used.

The Flowcell Summary in the HTML report contains the following information:

- Clusters (Raw)
- Clusters (PF)
- Yield (MBases)

NOTE

For HiSeq X, HiSeq 4000, and HiSeq 3000, the number of raw clusters is actually the number of wells on the flowcell that could potentially be seeded. The value is the same in all cases.

The Lane Summary in the HTML report provides the following information for each project, sample, and index sequence specified in the sample sheet:

- Lane #
- Clusters (Raw)
- % of the Lane
- % Perfect Barcode
- % One Mismatch
- Clusters (Filtered)
- Yield
- % PF Clusters
- %Q30 Bases
- Mean Quality Score

The Top Unknown Barcodes table in the HTML report provides the count and sequence for the 10 most common unmapped bar codes in each lane.

Troubleshooting

bcl2fastq2 Conversion Software v2.17 fails to complete.

If bcl2fastq2 Conversion Software v2.17 fails to complete processing of a run, it could be due to a missing or corrupt input file. A good first step in troubleshooting is to examine the log file, as missing or corrupt files is reported there. The exact wording of the file status reported varies depending on the nature of the file corruption. If the issue is due to a problematic BCL file, invoking the `--ignore-missing-bcls` option (described in *Options for BCL Conversion and Demultiplexing* on page 24) can often resolve this issue. This option is robust to most BCL file corruption situations that could occur.

Higher than expected percentage of reads is assigned to Undetermined.

This problem can occur if an error was made when specifying the index sequence. The Top Unknown Barcodes table in the HTML report provides the observed indexes, and helps troubleshooting this issue.

Trouble processing Small RNA samples.

For proper adapter trimming with Small RNA samples, use the following command-line settings instead of the default settings. `--minimum-trim-read-length 20` and `--mask-short-adapter-reads 20`.

Appendix: Requirements

Network Infrastructure

The large data volumes generated and moved when running bcl2fastq2 Conversion Software v2.17 mean that you must have a high-throughput ethernet connection (at least 1 Gigabit recommended) or other data transfer mechanism.

Server Configurations

You can use a single multi-processor or a multi-core computer running Linux.

Analysis Computer

Illumina supports running bcl2fastq2 Conversion Software v2.17 only on Linux operating systems. If all of the prerequisites described in this section are met, it might be possible to run the software on other 64-bit Unix variants.

Memory Requirements

bcl2fastq2 Conversion Software v2.17 requires a minimum of 32 GB RAM.

Software Requirements

bcl2fastq2 Conversion Software v2.17 has been primarily developed and tested on CentOS 6 and RedHat Enterprise Linux 5, the recommended and supported platform. If all of the prerequisites described in this section are met, it may be possible to install and run the bcl2fastq2 Conversion Software v2.17 on other 64-bit Linux distributions or on other Unix variants. Particularly a similar distribution such as Fedora.

The following software is required to run bcl2fastq2 Conversion Software v2.17:

- zlib
- librt
- libpthread

To build bcl2fastq2 Conversion Software v2.17, you need the following software. Versions listed are tested and supported; newer versions are untested.

- gcc 4.7 (with support for c++11)
- boost 1.54 (with its dependencies)
- CMake 2.8.9
- zlib
- librt
- libpthread

Technical Assistance

For technical assistance, contact Illumina Technical Support.

Table 2 Illumina General Contact Information

Website	www.illumina.com
Email	techsupport@illumina.com

Table 3 Illumina Customer Support Telephone Numbers

Region	Contact Number	Region	Contact Number
North America	1.800.809.4566	Italy	800.874909
Australia	1.800.775.688	Netherlands	0800.0223859
Austria	0800.296575	New Zealand	0800.451.650
Belgium	0800.81102	Norway	800.16836
Denmark	80882346	Spain	900.812168
Finland	0800.918363	Sweden	020790181
France	0800.911850	Switzerland	0800.563118
Germany	0800.180.8994	United Kingdom	0800.917.0041
Ireland	1.800.812949	Other countries	+44.1799.534000

Safety Data Sheets

Safety data sheets (SDSs) are available on the Illumina website at support.illumina.com/sds.html.

Product Documentation

Product documentation in PDF is available for download from the Illumina website. Go to support.illumina.com, select a product, then click **Documentation & Literature**.

15051736 Rev. G

Illumina
San Diego, California 92122 U.S.A.
+1.800.809.ILMN (4566)
+1.858.202.4566 (outside North America)
techsupport@illumina.com
www.illumina.com