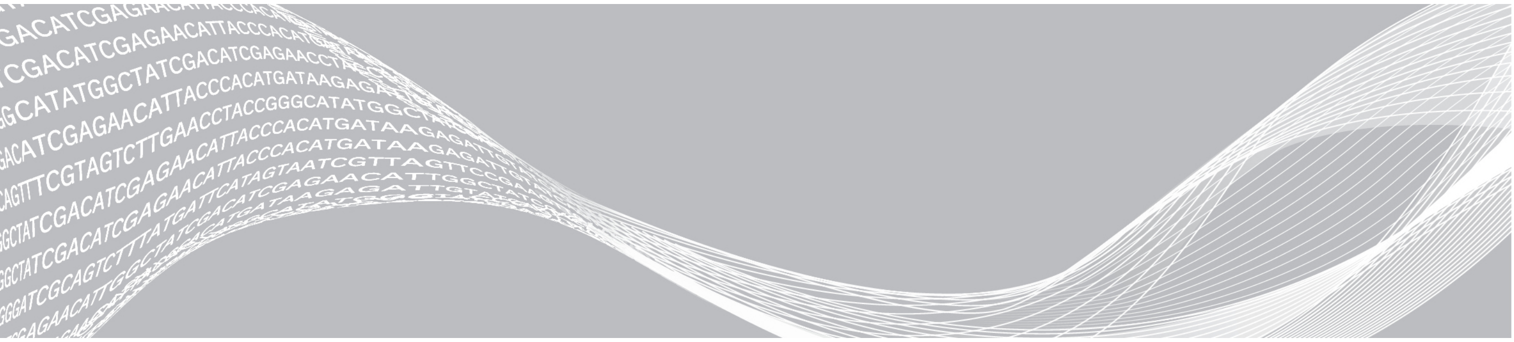
illumina®

bcl2fastq2 Conversion Software v2.20

Software Guide



Document # 15051736 v03
February 2019

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

This document and its contents are proprietary to Illumina, Inc. and its affiliates ("Illumina"), and are intended solely for the contractual use of its customer in connection with the use of the product(s) described herein and for no other purpose. This document and its contents shall not be used or distributed for any other purpose and/or otherwise communicated, disclosed, or reproduced in any way whatsoever without the prior written consent of Illumina. Illumina does not convey any license under its patent, trademark, copyright, or common-law rights nor similar rights of any third parties by this document.

The instructions in this document must be strictly and explicitly followed by qualified and properly trained personnel in order to ensure the proper and safe use of the product(s) described herein. All of the contents of this document must be fully read and understood prior to using such product(s).

FAILURE TO COMPLETELY READ AND EXPLICITLY FOLLOW ALL OF THE INSTRUCTIONS CONTAINED HEREIN MAY RESULT IN DAMAGE TO THE PRODUCT(S), INJURY TO PERSONS, INCLUDING TO USERS OR OTHERS, AND DAMAGE TO OTHER PROPERTY, AND WILL VOID ANY WARRANTY APPLICABLE TO THE PRODUCT(S).

ILLUMINA DOES NOT ASSUME ANY LIABILITY ARISING OUT OF THE IMPROPER USE OF THE PRODUCT(S) DESCRIBED HEREIN (INCLUDING PARTS THEREOF OR SOFTWARE).

Introduction

The Illumina bcl2fastq2 Conversion Software v2.20 demultiplexes sequencing data and converts base call (BCL) files into FASTQ files. For every cycle of a sequencing run, the Real-Time Analysis (RTA) software generates a BCL file containing base calls and associated quality scores (Q-scores). Most data analysis applications require FASTQ files as input.

Local Run Manager and MiSeq Reporter automatically convert BCL files into FASTQ files as a first step in an analysis. When a run is streamed to BaseSpace Sequence Hub for analysis, BaseSpace Sequence Hub also automatically converts BCL files. The resulting FASTQ files are then used as input for the analysis app.

For other data analysis applications, use the bcl2fastq2 Conversion Software to convert BCL files from any Illumina sequencing system running RTA v1.18.54, or later. For earlier versions of RTA, use bcl2fastq v1.8.4.

BCL to FASTQ Conversion Process

The software uses input files, which are the output of a sequencing run, to convert BCL files into FASTQ files. For each cluster that passes filter (PF), the software writes one entry to one FASTQ file for each sample in each read.

- For a single-read run, the software creates one Read 1 FASTQ file per sample.
- For a paired-end run, the software creates one Read 1 and one Read 2 FASTQ file per sample.

The sample FASTQ files are compressed and appended with the *fastq.gz extension. Thus, per-cycle BCL files are converted into per-read FASTQ files that can be used as input for data analysis.

Demultiplexing Process

Multiplexing adds a unique index adapter sequence to each sample during library prep, generating uniquely tagged libraries that can be identified and sorted for analysis. Demultiplexing then assigns clusters to a sample based on the index adapter sequence of the cluster.

NOTE

To optimize demultiplexing results, choose index adapters that optimize color balance when performing library prep. For more information, see the *Index Adapters Pooling Guide (document # 1000000041074)*.

The bcl2fastq2 Conversion Software demultiplexes multiplexed samples as part of the conversion process. If samples are not multiplexed, the software skips demultiplexing and assigns all clusters in a flow cell lane to one sample.

Adapter Trimming and UMI Removal

Depending on settings, the bcl2fastq2 Conversion Software trims adapter sequences and removes Unique Molecular Identifier (UMI) bases from reads:

- Adapter trimming—The software determines whether a read extends past the DNA insert and into the sequencing adapter. An approximate string matching algorithm identifies all or part of the adapter sequence and treats inserts and deletions (indels) as one mismatch. Base calls matching the adapter sequence and beyond are masked or removed from the FASTQ file.
- UMI removal—UMIs are random k-mers attached to the genomic DNA (gDNA) before polymerase chain reaction (PCR) amplification. After the UMI is amplified with amplicons, the software can retrieve the bases and include them in the read name in the FASTQ files. When the TrimUMI sample sheet setting is active, the software can also remove the bases from the reads.

Requirements and Installation

Download the bcl2fastq2 Conversion Software v2.20 from the bcl2fastq Conversion Software support pages on the Illumina website, and then install it on a computer that meets the following requirements.

Component	Requirements
Network infrastructure	1 Gb minimum
Server infrastructure	Single multiprocessor or multicore computer running Linux
Memory	32 GB RAM
Operating system	Red Hat Enterprise Linux 6 or CentOS 6*
Software	The following software is always required: • zlib • librt • libpthread Installing from source requires the following additional software: • gcc 4.8.2 or later (with support for C++11) • boost 1.54 • CMake 2.8.9

* Other Linux distributions might function but are not supported.

The bcl2fastq2 Conversion Software has a command-line interface. Installation requires assistance from an IT representative or system administrator with the appropriate privileges. You can install from an RPM package (recommended) or from source (advanced).

Install From RPM Package

Installing from the RPM package is the typical, recommended installation option. The starting point is the binary executable `/usr/local/bin/bcl2fastq`.

1 Make sure that you have access to the root system.

2 Install the RPM package using one of the following commands:

- To install the software in the default location, enter:

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

- To specify a custom install location, enter:

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

Install From Source

Installing from source is intended for advanced users who are not using the recommended operating systems.

Directory Locations

The following environment variables specify directory locations for installation. The build directory and source directory must be different.

Variables	Description
SOURCE	Location of the bcl2fastq2 Conversion Software source code.
BUILD	Location of the build directory.
INSTALL_DIR	Location where the executable is installed.

For example, you can set the environment variables as:

```
export TMP=/tmp
export SOURCE=${TMP}/bcl2fastq
export BUILD=${TMP}/bcl2fastq2-v2.19.x-build
export INSTALL_DIR=/usr/local/bcl2fastq2-v2.19.x
```

Build and Install the Software

1 Make sure that you have access to the ${INSTALL_DIR} directory:
  - In step 3, the directory requires write permission.
  - In step 4, the directory might require root privilege.

2 Decompress and extract the source code using the following command, which populates the directory ${TMP}/bcl2fastq:

```
cd ${TMP}
tar -xvzf bcl2fastq2-v2.19.x.tar.gz
```

3 Configure the build using the following commands, which create and populate the build directory:

```
mkdir ${BUILD}
cd ${BUILD}
chmod ugo+x ${SOURCE}/src/configure
chmod ugo+x ${SOURCE}/src/cmake/bootstrap/installCmake.sh
${SOURCE}/src/configure --prefix=${INSTALL_DIR}
```

In the final command, --prefix provides the absolute path to the installation directory.

4 Build and install the package using the following commands:

```
cd ${BUILD}
make
make install
```

Input Files

For each run, the control software generates an output folder to hold the BCL files and other sequencing data. The bcl2fastq2 Conversion Software uses this output as input. The output is recorded in various file formats, which are described in the following sections. If a sample sheet is uploaded to the control software during run setup, it is included among the output.

NOTE

This guide uses the terms output folder and run folder interchangeably. The output folder is a copy of the run folder, so either folder is acceptable input for bcl2fastq2 Conversion Software. When configuring the instrument or setting up a run, you have the option of setting the output folder location. The run folder location is system-defined.

If your instrument is configured to save the output folder locally on the control computer, you must transfer the folder to the computer with bcl2fastq2 Conversion Software installed. Otherwise, you can access the output folder from a network location.

Sequencing Data

The following table lists the output files that comprise sequencing data. The bcl2fastq2 Conversion Software uses these output files as input.

System	Input for bcl2fastq2 Conversion Software
HiSeq X, HiSeq 4000, and HiSeq 3000	• Base call files (*.bcl.gz) • Filter files (*.filter) • Cluster location files (s.locs) • RunInfo.xml • [Optional] Sample sheet (*.csv)
iSeq 100	• Base call files (*.bcl.bgzf) • Filter files (*.filter) • Cluster location files (s.locs) • RunInfo.xml • [Optional] Sample sheet (*.csv)
MiniSeq, NextSeq 550, and NextSeq 500	• Base call files (*bcl.bgzf) • Base call index files (*.bci) • Filter files (*.filter) • Cluster location files (*.locs) • RunInfo.xml • [Optional] Sample sheet (*.csv)
MiSeq, HiSeq 2500, and HiSeq 2000	• Base call files (*.bcl.gz) • Statistics files (*.stats) • Filter files (*.filter) • Cluster location files (*.locs) • RunInfo.xml • Configuration files • [Optional] Sample sheet (*.csv)
NovaSeq 6000	• Concatenated base call files (*.cbcl) • Filter files (*.filter) • Cluster location files (s.locs) • RunInfo.xml • [Optional] Sample sheet (*.csv)

All output files reside in the output folder. The output folder naming convention varies by system and can include the following variables separated by underscores:

- The six- or eight-digit date of the run in YYMMDD or YYYYMMDD format.
- The instrument or control computer ID consisting of any combination of alphanumeric characters and hyphens.
- A consecutively numbered experiment or run ID consisting of at least one digit.
- The flow cell ID.

For example, the iSeq 100 System uses the naming format <YYYYMMDD>_<Instrument ID>_<Run Number>_<Flow Cell ID>, resulting in an output folder named 20180331_FFSP247_4_BNS417-05-25-12. For more information on output folder directories and names, base calling, and tiles, see the system guide for your instrument.

As a best practice, give experiments and samples unique names to prevent naming conflicts. When publishing data to a public database, use a prefix for each instrument with the identity of the lab.

Base Call Files

Base call (BCL) files are compressed with the gzip (*.gz) or blocked GNU zip (*.bgzf) format.

Table 1 BCL File Format

Bytes	Description	Data Type
Bytes 0–3	Number of N cluster	Unsigned 32 bits integer
Bytes 4–(N+3) N–Cluster index	Bits 0–1 are the bases, [A, C, G, T] for [0, 1, 2, 3]. Bits 2–7 are shifted by 2 bits and contain the quality score. All bits with 0 in a byte are reserved for no call.	Unsigned 8 bits integer

Concatenated Base Call Files

Concatenated base call (CBCL) files contain aggregated BCL data. Tiles from the same lane and surface are aggregated into one CBCL file for each lane and surface.

CBCL File Header		
Bytes/Field	**Description**	**Data Type**
Bytes 0–1	Version number, current version is 1	unsigned 16 bits little endian integer
Bytes 2–5	Header size	unsigned 32 bits little endian integer
Byte 6	Number of bits per base call	unsigned
Byte 7	Number of bits per q-score	unsigned
q-val mapping info		
Bytes 0–3	Number of bins (B), zero indicates no mapping	
B pairs of 4 byte values (if B > 0)	{from, to}, {from, to}, {from, to} … from: quality score bin to: quality score	
Number of tile records		unsigned 32 bits little endian integer
gzip virtual file offsets, one record per tile		
Bytes 0–3: tile number		
Bytes 4–7	Number of clusters written into the current block (required due to bit-packed q-scores)	unsigned 32 bit integer
Bytes 8–11	Uncompressed block size of the tile data (useful for sanity check when excluding non-PF clusters)	unsigned 32 bit integer
Bytes 12–15	Compressed block size of the tile data	unsigned 32 bit integer
non-PF clusters excluded flag	1: non-PF clusters are excluded 0: non-PF clusters are included	

CBCL File Content
N blocks of gzip files, where N is the number of tiles. Each block consists of C number of base calls and quality score pairs where C is the number of clusters for the given tile. Each base call and quality score pair has the following format (assuming base calls use 2 bits): • Bits 0–1: Base calls (respectively [A, C, G, T] for [00, 01, 10, 11]) • Bits 2 and up: Quality score (unsigned Q bit little endian integer where Q is the number of bits per q-score). For a 2-bit quality score, each byte has two clusters where the bottom 4 bits are the first cluster and the higher 4 bits are the second cluster.

Base Call Index Files

Base call index files (BCI) files contain one record per tile in binary format.

Bytes	Description
Bytes 0–3	The tile number.
Bytes 4–7	The number of clusters in the tile.

Statistics Files

Statistics (STATS) files are binary files that contain base calling statistics.

Table 2 STATS File Format

Start	Description	Data Type
Byte 0	Cycle number.	integer
Byte 4	Average cycle intensity.	double
Byte 12	Average intensity for A for all clusters with intensity for A.	double
Byte 20	Average intensity for C for all clusters with intensity for C.	double
Byte 28	Average intensity for G for all clusters with intensity for G.	double
Byte 36	Average intensity for T for all clusters with intensity for T.	double
Byte 44	Average intensity for A for clusters with base call A.	double
Byte 52	Average intensity for C for clusters with base call C.	double
Byte 60	Average intensity for G for clusters with base call G.	double
Byte 68	Average intensity for T for 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

* X indicates an unknown base.

Filter Files

Filter files are binary files that specify whether clusters passed filter.

Table 3 Filter File Format

Bytes	Description
Bytes 0–3	Zero value (for backwards compatibility).
Bytes 4–7	The filter format version number.
Bytes 8–11	The number of clusters.
Bytes 12–(N+11) N—cluster number	Unsigned 8 bits integer. Bit 0 is pass or failed filter.

Configuration Files

One configuration file resides in the Intensities folder and records information on the generation of subfolders in the output folder directory. It contains a tag-value list describing the cycle-image folders used to generate each folder of intensity and sequence files.

The other configuration file resides in the BaseCalls folder and contains metadata on the sequencing run. Both files are in XML format.

Location Files

Location files (LOCS) are binary files that contain the cluster positions on the flow cell. CLOCS files are compressed versions of LOCS files.

Files appended with _pos.txt are text-based files containing two columns and a number of rows equal to the number of clusters. The first column records the X-coordinate and the second column records the Y-coordinate. Each row ends with <cr><lf>.

Run Information File

The run information file (RunInfo.xml) resides at the root level of the output folder. The file contains the run name, number of cycles, whether a read is an Index Read, and the number of swaths and tiles.

Sample Sheets

A sample sheet (SampleSheet.csv) records information about samples and the corresponding index adapters. The bcl2fastq2 Conversion Software uses this information to demultiplex and convert BCL files.

For most runs, a sample sheet is optional. The default location is the root output folder, but you can use the command --sample-sheet to specify any CSV file in any location. When a sample sheet is not provided, the software assigns all reads to the default sample Undertermined_S0.

Settings Section

The software uses the Settings section of the sample sheet to specify adapter trimming, cycle, UMI, and index options.

Table 4 Adapter Trimming Specifications

Setting	Description
`Adapter` or `TrimAdapter`	The sequence of the adapter to be trimmed. If a sequence for AdapterRead2 is specified, the setting applies to Read 1 only. To trim multiple adapters, separate the sequences with a plus sign (+) to indicate independent adapters that must be independently assessed for trimming for each read.
`AdapterRead2` or `TrimAdapterRead2`	The adapter sequence to be trimmed in Read 2. If not provided, the sequence specified in Adapter or TrimAdapter is applied. To trim multiple adapters, separate the sequences with a plus sign (+) to indicate independent adapters that must be assessed for trimming independently for each read.
`MaskAdapter`	The adapter sequence to be masked rather than trimmed. If MaskAdapterRead2 is provided, this setting masks Read 1 only.
`MaskAdapterRead2`	The adapter sequence to be masked in Read 2. If not provided, the same sequence specified in MaskAdapter is applied.
`FindAdaptersWithIndels`	0—False. A sliding window algorithm is used and indels of adapter sequence bases are not allowed. 1—True (default). An approximate string matching algorithm identifies the adapter, treating indels as one mismatch.

Table 5 Cycle, UMI, and Tile Specifications

Setting	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.
`Read1UMIStartFromCycle`	The first cycle to use for UMI in Read 1. The cycle index is absolute and not affected by the Read1StartFromCycle setting. The software supports UMIs only at the beginning or end of reads. Unless paired with Read1UMILength, the software ignores this setting.
`Read2UMIStartFromCycle`	The first cycle to use for UMI in Read 2. The cycle index is absolute and not affected by the Read2StartFromCycle setting. The software supports UMIs only at the beginning or end of reads. Unless paired with Read2UMILength, the software ignores this setting.
`TrimUMI`	0—False (default). 1—True. The software trims the UMI bases from Read 1 and Read 2.
`ExcludeTiles`	Tiles to exclude. Separate tiles with a plus sign (+) or specify a range with a hyphen (-). For example: `ExcludeTiles,1101+2201+1301-1306` skips tiles 1101, 2201, and 1301–1306.
`ExcludeTilesLaneX`	Tiles to exclude for a given lane. For example: `ExcludeTilesLane6,1101-1108` skips tiles 1101–1108 for lane 6.

Table 6 FASTQ Specifications

Setting	Description
`CreateFastqForIndexReads`	0—False (default). 1—True. The software generates FASTQ files for index reads. Normally, FASTQ files for index reads are not needed because the index adapter sequences are included in the FASTQ files. Demultiplexing is automatic and based on the sample sheet.*
`ReverseComplement`	0—False (default). 1—True. All reads are are written to FASTQ files in the reverse complement. The reverse complements are necessary when processing mate-pair data using BWA, which requires paired-end data, and other nonstandard cases.

* FASTQ file generation is based on Index Read masks specified in the --use-bases-mask option or RunInfo.xml (when --use-bases-mask is not used).

Data Section

The software uses columns in the Data section to sort samples and index adapters.

Column	Description
Lane	When specified, the software generates FASTQ files only for the samples with the specified lane number.
Sample_ID	The sample ID.
Sample_Name	The sample name.
Sample_Project	The sample project name. The software creates a directory with the specified sample project name and puts the FASTQ files there. You can assign multiple samples to the same project.
index	The Index 1 (i7) index adapter sequence.
index2	The Index 2 (i5) Index adapter sequence.

The Sample_Project, Sample_ID, and Sample_Name columns accept alphanumeric characters, hyphens (-), and underscores (_). Many file systems do not support other symbols or spaces.

Do not use all or unknown as a sample ID, all or undetermined as a sample name, or all or default as the sample project name. Samples with these terms are omitted from the report. If the Sample_ID and Sample_Name columns do not match, the software writes the FASTQ files to the SampleID subdirectory.

Demultiplexing Scenarios

For each sample listed in a sample sheet, the software produces one FASTQ file for each sample for each read.

- When a sample sheet contains multiplexed samples, the software:
  - Places reads without a matching index adapter sequence in the Undetermined_S0 FASTQ file.
  - Places reads with valid index adapter sequences in the sample FASTQ file.
- When a sample sheet contains one unindexed sample, all reads are placed in the sample FASTQ files (one each for Read 1 and Read 2).
- When a sample sheet does not exist, or exists but has no Data section, all reads are placed in one FASTQ file named Undetermined_S0.
- When the Lane column in the Data section is not used, all lanes are converted. Otherwise, only populated lanes are converted.

Sample Sheet Creation

The Illumina Experiment Manager (IEM) software is compatible with most Illumina sequencing systems and analysis software. Use IEM to create and edit sample sheets before starting library prep. For more information, visit the Illumina Experiment Manager support pages on the Illumina website.

When sequencing in Manual mode on the iSeq 100 System, create a sample sheet by editing the *iSeq 100 System Sample Sheet Template for Manual Mode*. Download the template from the iSeq 100 Sequencing System support pages.

Local Run Manager and the BaseSpace Sequence Hub Prep tab create sample sheets for you and save them in the appropriate location. When using either of these applications, IEM and the sample sheet template are not necessary.

Convert and Demultiplex BCL Files

Use the following instructions to demultiplex and convert BCL files. Add command options to modify the software operation as needed. If you add options that have a corresponding sample sheet setting, the command-line value overwrites the sample sheet value.

1 Open a command-line window.

2 Type the following command and add options as needed.

```
nohup /usr/local/bin/bcl2fastq
```

For example, the following command line populates BaseCalls with FASTQ files. By default, --runfolder-dir is the run folder and --output-dir is the BaseCalls subfolder (<run folder>\BaseCalls).

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

Directory Options

Directory options determine the paths of various directories. The first two entries in the following table are main options. The remaining entries are advanced options that provide more control of the conversion process, but are not necessary for standard use.

Option	Description	Default
`-R, --runfolder-dir`	A main command-line option that indicates the path to the run folder directory.	./
`-o, --output-dir`	A main command-line option that indicates the path to demultiplexed output.	<runfolder-dir>/Data/Intensities/BaseCalls/
`-i, --input-dir`	Indicates the path to the input directory.	<runfolder-dir>/Data/Intensities/BaseCalls/
`--sample-sheet`	Indicates the path to the sample sheet so you can specify the sample sheet location and name, if different from the default.	<runfolder-dir>/SampleSheet.csv
`--intensities-dir`	Indicates the path to the intensities directory. When the intensities directory is specified, the input directory must also be specified.	<input-dir>/../
`--interop-dir`	Indicates the path to the demultiplexing statistics directory.	<runfolder-dir>/InterOp/
`--stats-dir`	Indicates the path to the demultiplexing statistics directory (human-readable).	<output-dir>/Stats/
`--reports-dir`	Indicates the path to the reporting directory.	<output-dir>/Reports/

Processing Options

Processing options control threading. For example, you want to limit your usage because you share computing resources.

Option	Description	Default
`-r, --loading-threads`	Number of threads to load BCL data.	Depends on system architecture.

`-p,`
`--processing-threads`
	Number of threads to process demultiplexing data.	Depends on system architecture.

`-w,`
`--writing-threads`
	Number of threads to write FASTQ data. This number must be lower than number of samples.	Depends on system architecture.

When threading is supported, the software uses the follow defaults to manage threads for processing:

- Four threads for reading the data.
- Four threads for writing the data.
- Twenty percent of threads for demultiplexing data.
- One hundred percent of threads for processing demultiplexed data.

The file i/o threads are typically inactive and consume minimal processing time. Processing demultiplexed data is allocated one thread per central processing unit (CPU) to prevent idle CPUs, resulting in more threads than CPUs by default.

Considerations for Multiple Threads

When using processing options to assign multiple threads, consider the following information:

- The most demanding step is the processing step (-p option). Assign this step the most threads.
- The reading and writing stages are simple and do not need many threads. This consideration is important for a local hard drive. Too many threads cause too many parallel read-write actions and suboptimal performance.
- Use one thread per CPU core plus some extra. This method prevents CPUs from 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 cost time due to context switching.

Behavioral Options

Behavioral options determine how the software responds to file compression, tile and cycle processing, missing or corrupt files, masking, and trimming. Masking replaces values with N instead of removing them as trimming does.

Option	Description	Default
`--adapter-stringency`	The minimum match rate that triggers masking or trimming. This value is calculated as MatchCount / (MatchCount + MismatchCount). Accepted values are 0–1. However, using any value < 0.5 introduces too many false positives and is not recommended. The default value of 0.9 indicates that only reads with > 90% sequence identity with the adapter are trimmed.	0.9
`--barcode-mismatches`	The number of mismatches allowed per index adapter. Accepted values are 0, 1, or 2. Multiple, comma-delimited entries are allowed. Each entry is applied to the corresponding index adapter. The last entry applies to all remaining index adapters.	1
`--create-fastq-for-index-reads`	Create FASTQ files for index reads based on the following guidelines: • The --use-bases-mask option specifies index read masks. • When --use-bases-mask is not used, use RunInfo.xml.	N/A*
`--ignore-missing-bcls`	The software ignores missing or corrupt BCL files and assumes 'N'/'#' for missing calls.	N/A*
`--ignore-missing-filter`	The software ignores missing or corrupt filter files and assumes that all clusters in tiles with missing filter files passed filter.	N/A*
`--ignore-missing-positions`	The software ignores missing or corrupt cluster location files. When cluster location files are missing, the software writes unique coordinate positions into the FASTQ header.	N/A*
`--minimum-trimmed-read-length`	The minimum read length after adapter trimming. The software trims adapter sequences from reads to the value of this parameter. Bases below the specified value are masked.	35
`--mask-short-adapter-reads`	Specifies the following behavior: • If the number of bases remaining after adapter trimming is less than --minimum-trimmed-read-length, force the read length to be equal to --minimum-trimmed-read-length. Mask adapter bases below this length. • If the remaining number of bases is below --mask-short-adapter-reads, mask all bases to result in a read masked per --minimum-trimmed-read-length. • Reads shorter than the setting for --mask-short-adapter-reads are also masked. Specify a value that is less than or equal to --minimum-trimmed-read-length, otherwise this option does not apply. A greater value automatically defaults to the same value as --minimum-trimmed-read-length. Applying this option does not require trimming the read. Rather, it must be below the --minimum-trimmed-read-length.	22

`--tiles`	Selects a subset of available tiles for processing. To make multiple selections, separate the regular expressions with commas.
	For example:
	• To select all 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,s_[2-8]`
		N/A*

Option	Description	Default

`--use-bases-mask`	Specifies how to process each cycle:
	• n—Ignore the cycle.
	• Y (or y)—Use the cycle.
	• I—Use the cycle for an Index Read.
	• A number—Repeat the previous character the indicated number of times.
	• *—Repeat the previous character until the end of the read or index (length per RunInfo.xml).
	Commas separate read masks. The format for dual indexing is the following syntax or specified variations:
	`--use-bases-mask Y*,I*,I*,Y*`
	You can also specify --use-bases-mask multiple times for separate lanes. In the following example,`1:` indicates that the setting applies to lane 1. The second --use-bases-mask parameter applies to all other lanes.
	`--use-bases-mask 1:y*,i*,i*,y* --use-bases-mask y*,n*,n*,y*`
	If this option is not specified, RunInfo.xml determines the mask. If it cannot determine the mask, specify the --use-bases-mask option. When specified, the number of index cycles and the index length in the sample sheet must match.
		N/A*

`--with-failed-reads`	Include all clusters in the output, including those that did not pass filter. By default, clusters that did not pass filter are excluded. FASTQ files containing failed reads cannot be uploaded to BaseSpace Sequence Hub. After cycle 25, RTA2 stops reading clusters that do not pass filter. Systems other than MiSeq and HiSeq 2500 produce 25 bases, then all Ns. This option cannot be applied to CBCL files.	N/A*
`--write-fastq-reverse-complement`	Generate FASTQ files with the reverse complements of actual data.	N/A*
`--no-bgzf-compression`	Turn off BGZF and use GZIP to compress FASTQ files. BGZF compression allows downstream applications to decompress in parallel. This option is available for FASTQ data consumers that cannot handle standard GZIP formats.	N/A*
`--fastq-compression-level`	The Zlib compression level (1–9) to apply to FASTQ files.	4
`--no-lane-splitting`	Do not split FASTQ files by lane. If you plan to upload the FASTQ files to BaseSpace Sequence Hub, *do not use* this option. It generates FASTQ files that are not compatible with the file uploader.	N/A*
`--find-adapters-with-sliding-window`	Finds adapters using a simple sliding window algorithm. Ignores adapter sequence indels.	N/A*

* Not applicable

General Options

General options determine miscellaneous settings for help, version information, and the minimum log level.

Option	Description	Default

`-h,`
`--help`
	Produce a help message and exit the application.	Not applicable

`-v,`
`--version`
	Print version information.	Not applicable

Option	Description	Default

`-l,`
`--min-log-level`
	The minimum log level, prioritizes messages. Acceptable values are NONE, FATAL, ERROR, WARNING, INFO, DEBUG, and TRACE.	`INFO`

Output Files and Directory

The bcl2fastq2 Conversion Software v2.20 generates the following files as output:

- FASTQ files
- InterOp files
- ConversionStats file
- DemultiplexingStats file
- Adapter Trimming file
- FastqSummary and DemuxSummary
- HTML reports
- JavaScript Object Notation (JSON) file

FASTQ Files

As converted versions of BCL files, FASTQ files are the primary output of the bcl2fastq2 Conversion Software. Like BCL files, FASTQ files contain base calls with associated Q-scores. Unlike BCL files, which contain per-cycle data, FASTQ files contain the per-read data that most analysis applications require.

The software generates one FASTQ file for every sample and every read. For example, for each sample in a paired-end run, the software generates two FASTQ files: one for Read 1 and one for Read 2. In addition to these sample FASTQ files, the software generates one FASTQ file containing all unknown samples. FASTQ files for Index Read 1 and Index Read 2 are typically not necessary, but are generated when the option --create-fastq-for-index-reads is applied.

FASTQ Files Directory

The software writes compressed, demultiplexed FASTQ files to the directory <run folder>\Data\Intensities\BaseCalls.

- If a sample sheet specifies the Sample_Project column for a sample, the software places the FASTQ files for that sample in the directory <run folder>\Data\Intensities\BaseCalls\<Project>. The same project directory contains the files for multiple samples.
- If the Sample_ID and Sample_Name columns are specified but do not match, the FASTQ files reside in a <SampleID> subdirectory where files use the Sample_Name value.

Reads with unidentified index adapters are recorded in one file named Undetermined_S0_. If a sample sheet includes multiple samples without specified index adapters, the software displays a missing barcode error and ends the analysis.

NOTE

The software allows one unindexed sample because identification is not necessary to sequence one sample. However, sequencing multiple samples requires multiplexing so the samples can be identified for analysis.

File Names

FASTQ files are named with the sample name and number, the flow cell lane, and read. The file extension is *.fastq.gz. For example: samplename_S1_L001_R1_001.fastq.gz.

- samplename—The name of the sample provided in the sample sheet. If a sample name is not available, the file name uses the sample ID instead.
- S1—The number of the sample based on the order that samples are listed in the sample sheet, starting with 1. In the example, S1 indicates that the sample is the first sample listed for the run.

  NOTE

  Reads that cannot be assigned to any sample are written to a FASTQ file as sample number 0 and excluded from downstream analysis.
- L001—The lane number of the flow cell, starting with lane 1, to the number of lanes supported.
- R1—The read. In the example, R1 indicates Read 1. R2 indicates Read 2 of a paired-end run.
- 001—The last portion of the file name is always 001.

File Format

FASTQ files are text-based files that contain base calls with corresponding Q-scores for each read. Each file has one 4-line entry:

- A sequence identifier with information about the run and cluster, formatted as:

```
@Instrument:RunID:FlowCellID:Lane:Tile:X:Y:UMI Read:Filter:0:IndexSequence
    or SampleNumber
```

- The sequence (base calls A, G, C, T, and N, for unknown bases).
- A plus sign (+) that functions as a separator.
- The Q-score using ASCII 33 encoding (see *Quality Score Encoding*).

Table 7 Sequence Identifier Fields

Field	Description
`@`	Each sequence identifier line starts with @.
`instrument`	The instrument ID.
`run ID`	The run number on the system.
`flow cell ID`	The flow cell ID.
`lane`	The flow cell lane number.
`tile`	The flow cell tile number.
`x_pos`	The X coordinate of the cluster.
`y_pos`	The Y coordinate of the cluster.
`UMI`	[Optional] The UMI sequence (A, G, C, T, and N). When the sample sheet specifies UMIs, a plus sign separates the Read 1 and Read 2 sequences.
`read`	1—Read 1, which is the first read of a paired-end run or the only read of a single-read run. 2—Read 2, which is the second read of a paired-end run.
`is filtered`	Y—The read is filtered (shows when the --with-failed-reads option is applied). N—The read is not filtered.

Field	Description
`control number`	0—Control bits are not turned on.*
`index sequence or sample number`	The Index Read sequence (A, G, C, T, and N). • If the sample sheet indicates indexing, the index adapter sequence is appended to the end of the read identifier. • If indexing is not indicated (one sample per lane), the sample number is appended to the read identifier.

* Since the deprecation of control files starting with bcl2fastq2 Conversion Software v2.19, the control number value is always 0.

A complete FASTQ file entry resembles the following example:

```
@SIM:1:FCX:1:2106:15337:1063:GATCTGTACGTC 1:N:0:ATCACG

GATCTGTACGTCTCTGCNTCACCTCCACCGTGCAACTCATCACGCAGCTCATGCCCTTCGGCTGCCTCCTGGACTA

+

CCCCCGGGGGGGGGGGG#:CFFGFGFGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGEGGFGGG
```

File Compression

FASTQ files are compressed in the GNU zip format and appended with *.gz to create the extension *.fastq.gz. You can use tools such as gzip (command line) or 7-zip (GUI) to decompress FASTQ files.

NOTE

FASTQ files are too large to open in a standard text editor, and viewing them is not necessary. If you must view FASTQ files for troubleshooting purposes: use a text editor that can handle large files or a Unix or Linux operating system that can view large files via the command line.

The BGZF compression variant facilitates parallel decompression of FASTQ files by downstream applications. If a downstream application cannot handle the BGZF variant, use the --no-bgzf-compression option to turn it off. See *Behavioral Options* on page 13.

Quality Scores

A quality score, or Q-score, is a prediction of the probability of an incorrect base call. A higher Q-score implies that a base call is higher quality and more likely to be correct.

Based on the Phred scale, the Q-score is a compact way to communicate small error probabilities. Given a base call, X, the probability that X is not true, P(~X), results in a quality score, Q(X). The following calculation shows this relationship, where P(~X) is the estimated error probability:

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

The following table shows the relationship between the quality score and error probability.

Q-score Q(X)	Error Probability P(~X)
Q40	0.0001 (1 in 10,000)
Q30	0.001 (1 in 1,000)
Q20	0.01 (1 in 100)
Q10	0.1 (1 in 10)

During a run, base call quality scores are calculated after cycle 25. The results are recorded in BCL files with the base call for the cycle.

Quality Score Encoding

In FASTQ files, Q-scores are encoded into a compact form that uses only 1 byte per quality value. This encoding represents the quality score as the character with an ASCII code equal to the value + 33.

The following table demonstrates the relationship between the encoding character, ASCII code, and represented Q-score. When Q-score binning is used, the subset of Q-scores applied by the bins is displayed.

Table 8 ASCII Characters Encoding Q-Scores 0–40

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

Symbol	ASCII Code	Q-score
A	65	32
B	66	33
C	67	34
D	68	35
E	69	36
F	70	37
G	71	38
H	72	39
I	73	40

InterOp Files

InterOp files reside in the InterOp folder of the run directory. The Sequencing Analysis Viewer (SAV) software uses InterOp files as input to summarize run metrics, such as cluster density, intensities, and Q-scores.

The IndexMetricsOut.bin file generated by bcl2fastq2 Conversion Software stores index metrics and 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 File

The ConversionStats.xml file resides in the Stats folder of the output directory or in the directory specified by the --stats-dir option. The file contains the lane number for each lane and the following information for each tile:

- Raw Cluster Count
- Read Number
- YieldQ30
- Yield
- QualityScore Sum

DemultiplexingStats File

The DemultiplexingStats.xml file resides in Stats folder of the output directory or in the directory specified by the --stats-dir option.

The file contains the flow cell ID and project name. For each sample, index, and lane, the file lists the BarcodeCount, PerfectBarcodeCount, and OneMismatchBarcodeCount (if applicable).

Adapter Trimming File

The adapter trimming file is a text-based file that contains a statistics summary of adapter trimming for a FASTQ file. The file resides in the Stats folder of the output directory or in the directory specified by the --stats-dir option.

The file contains the fraction of reads with untrimmed bases for each sample, lane, and read number plus the following information:

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

FastqSummaryF1L# File

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

The files reside in the Stats folder of the output directory or in the directory specified by the --stats-dir option.

DemuxSummaryF1L# File

DemuxSummaryF1L#.txt files, where # indicates the lane number, are generated when the sample sheet contains at least one indexed sample. A file contains the percentage of each tile that each sample occupies. It also lists the 1000 most common unknown index adapter sequences and the total number of reads with each index adapter identified.

NOTE

To improve processing speed, the total for each index adapter is based on an estimate from a sampling algorithm.

These files are located in the Stats folder of the output directory or in the directory specified by the --stats-dir option.

HTML Reports

HTML reports are generated from data in DemultiplexingStats.xml and ConversionStats.xml. The reports reside in Reports\html in the output directory or in the directory specified by the --reports-dir option.

The flow cell summary contains the following information:

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

NOTE
For patterned flow cells, the number of raw clusters is equal to the number of wells on the flow cell.

The lane summary 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 index adapters in each lane.

Java Script Object Notation File

The JavaScript Object Notation (JSON) file facilitates parsing the output data. The data in the JSON file are a combination of the following files:

- InterOP
- ConversionStats
- DemultiplexingStats
- Adapter trimming
- FastqSummary and DemuxSummary
- HTML report
- The JSON file format is similar to the following example:

```
{
    Flowcell: string //matches Flowcell from RunInfo.xml
    RunNumber: int, //matches Run Number from RunInfo.xml
    RunId: string, //matches Run Id from RunInfo.xml
    ReadInfosForLanes: [ //details per-lane read information
        {
            LaneNumber: int,
            ReadInfos: [
            Number: int, //indicates read 1 or read 2 (possible values: 1
               and 2)
            NumCycles: int, //indicates number of cycles for this read
            IsIndexedRead, bool // indicates whether or not this read is an
               index read
            ]
        }
    ],
    ConversionResults:[ //details the conversion/demultiplexing results
        {
            LaneNumber: int,
            TotalClustersRaw: int, //number of raw clusters in this lane
               (null for HiSeq X)
            TotalClustersPf: int //number of clusters passing filter in this
               lane
            Yield: int, //total yield in this lane
            DemuxResults: [ //do not include undetermined reads in this
               array
               {
                  SampleId: string,
                  SampleName: string,
                  IndexMetrics: [ //empty array if no indices were used for
                     demultiplexing this sample
                     {
                        IndexSequence: string, //if there are two indices,
                           then concatenate with '+' character (e.g.
                           "ATCGTCG+TGATCTA")
                        MismatchCounts: {
                           0: int, //count of perfectly matching barcodes
                           1: int //count of barcodes with one mismatch
                        }
                     }
                  ],
                  NumberReads: int, //number of read pairs identified as
                     index/index-pair
                  Yield: int, //number of bases after trimming
                  ReadMetrics: [
                     {
                        ReadNumber: int,
                 Yield: int,
                 YieldQ30: int,
                 QualityScoreSum: int,
                 TrimmedBases: int
               }
             ]
           }
        ]
      }
    ],
    UnknownBarcodes: [ //details all the unknown barcodes for a given lane
      and number of times it was encountered
      {
        Lane: int,
        Barcodes: {
           string: int //example: "ATGAAGAT": 5888
        }
      }
    ]
}
```

Troubleshooting

- If the software fails to complete an analysis, review the log file for missing input files or corrupt files. The reported file status varies depending on the type of file corruption. If a BCL file is the problem, apply the --ignore-missing-bcls command. See *Behavioral Options* on page 13.
- If the software cannot process TruSeq Small RNA samples, apply the --minimum-trim-read-length 20 and --mask-short-adapter-reads 20 options to overwrite the default values. See *Behavioral Options* on page 13.
- If the software assigns a high percentage of reads as undetermined, review the Top Unknown Barcodes table in the HTML report.

Revision History

Document	Date	Description of Change
Document # 15051736 v03	February 2019	Updated software descriptions to bcl2fastq2 Conversion Software v2.20, which supports the iSeq 100 System. Updated the name of BaseSpace to BaseSpace Sequence Hub. Updated file format descriptions for consistency. Updated installation information: • Added descriptions of both installation options. • Indicated that installing the software requires assistance from a system administrator or IT representative. Added information on: • Viewing FASTQ files. • Sample sheets for Local Run Manager and the BaseSpace Sequence Hub Prep tab. • The difference between the term output folder and run folder. Renamed this guide to *bcl2fastq2 Conversion Software v2.20 Software Guide*. Renamed and combined some sections to improve continuity. Referenced the *Index Adapters Pooling Guide (document # 1000000041074)* for information on multiplexing. Removed diagrams of input file directories, which are available in the system guides. Removed the even number option from the description of the control number field. Clarified the following points: • Illumina Experiment Manager (IEM) produces sample sheets, but does not convert or demultiplex files. • The bcl2fastq2 Conversion Software generates one FASTQ file for each sample for each read, plus one file for all unassigned samples in a read. • Only one unindexed sample is permitted in a sample sheet. • N and X each indicate an unknown base. • Position files are cluster location files. Redirected the software downloads link to the bcl2fastq Conversion Software support pages. Corrected the definition of JSON to JavaScript Object Notation. Corrected descriptions of read field entries and DemultiplexingStats.xml contents. Corrected information on the run folder naming convention.
Document # 15051736 v02	March 2018	Updated software descriptions to bcl2fastq2 Conversion Software v2.19. Updated the BCL2FASTQ options. Added the NovaSeq 6000 System file structure. Added information on the CBCL file format.
Document # 15051736 v01	April 2016	Updated to support bcl2fastq2 v2.18. Updated BCL2FASTQ options and sample sheet settings. Added JSON file and input files list for the MiniSeq System.
Document # 15051736 Rev. G	July 2015	Updated to software requirements, gcc version.
Document # 15051736 Rev. F*	June 2015	Updated to support bcl2fastq2 v2.17.

* Dates and change descriptions for revisions A–E are not available.

Technical Assistance

For technical assistance, contact Illumina Technical Support.

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

Illumina Customer Support Telephone Numbers

Region	Toll Free	Regional
North America	+1.800.809.4566	
Australia	+1.800.775.688	
Austria	+43 800006249	+43 19286540
Belgium	+32 80077160	+32 34002973
China	400.066.5835	
Denmark	+45 80820183	+45 89871156
Finland	+358 800918363	+358 974790110
France	+33 805102193	+33 170770446
Germany	+49 8001014940	+49 8938035677
Hong Kong	800960230	
Ireland	+353 1800936608	+353 016950506
Italy	+39 800985513	+39 236003759
Japan	0800.111.5011	
Korea	+82 80 234 5300	
Netherlands	+31 8000222493	+31 207132960
New Zealand	0800.451.650	
Norway	+47 800 16836	+47 21939693
Singapore	+1.800.579.2745	
Spain	+34 911899417	+34 800300143
Sweden	+46 850619671	+46 200883979
Switzerland	+41 565800000	+41 800200442
Taiwan	00806651752	
United Kingdom	+44 8000126019	+44 2073057197
Other countries	+44.1799.534000	

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

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

Illumina
5200 Illumina Way
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

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

illumina®