ASZip压缩测试

Exported from Confluence on 2024 January 24

BnR Generic pdf Template Version 2

Table of Contents

1.目录

2.测试原因

测试ASzip压缩功能，方便项目使用

3.测试目的

使用ASZip库对项目文件进行压缩测试，分别从以下角度考虑，得出测试结果：

- 能压缩成什么格式
- 同一种压缩格式下，相同文件时，不同循环周期压缩结果
- 同一种压缩格式下，相同文件时，不同压缩等级压缩结果
- 同一种压缩格式下，不同大小文件时，压缩结果对比

4.测试报告

测试结果

4.1 可压缩格式

- .tar　(.tar为打包文件，仅做文件的合并，例如多个文件打包为一个文件，没有压缩作用)
- .tar.gz
- .gz　(.gz为压缩文件)

4.2 不同格式压缩测试

4.2.1测试环境

- AS: AS4.7.6.114
- AR: F4.73
- CPU: X20CP3687X

4.2.2 测试程序

- 新建空项目

4.3 不同压缩格式&不同大小文件&不同循环周期的测试结果

4MB小文件：b2_2000_01_01_08_14_0.csv

110MB大文件：2022_11_21_15_08.7z

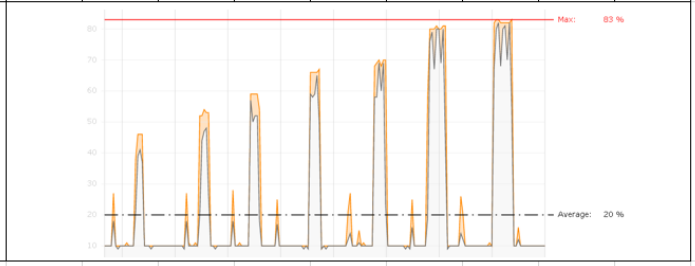
4.3.1 压缩测试结果

新建项目&ASZip通用性测试报告

4.3.1.1 大文件测试结果

现场大文件

- .gz格式测试结果
  - 20ms循环周期

.gz格式20ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25217KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→44%	10%→46%	10%→54%	10%→59%	10%→67%	10%→70%	10%→81%	10%→83%
iCycleCount	826	818	847	929	1054	1209	1308	1947	2139
压缩完成时间	16520ms	16360ms	16940ms	18580ms	21080ms	24180ms	26160ms	38940ms	42780ms
负载波动图									

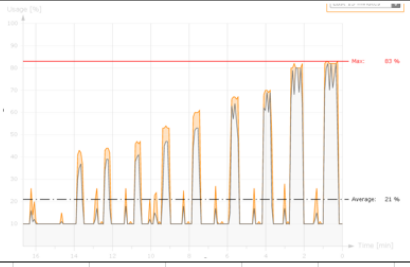
- 50ms循环周期

.gz格式50ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25217KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	15%→42%	15%→44%	15%→48%	15%→54%	15%→60%	15%→68%	15%→74%	15%→82%	15%→83%
iCycleCount	336	328	338	378	422	489	528	776	855
压缩完成时间	16800ms	16400ms	16900ms	18900ms	21100ms	24450ms	26400ms	38800ms	42750ms
负载波动图									

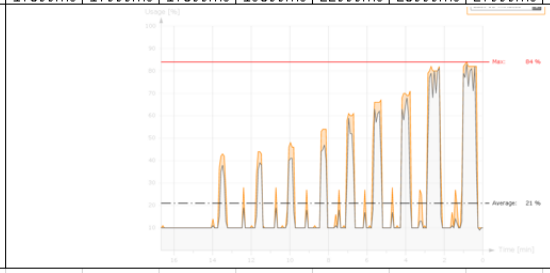
- 100ms循环周期

.gz格式100ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25217KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→44%	10%→47%	10%→54%	10%→63%	10%→68%	10%→70%	10%→82%	10%→83%
iCycleCount	169	167	170	188	211	246	262	389	428
压缩完成时间	16900ms	16700ms	17000ms	18800ms	21100ms	24600ms	26200ms	38900ms	42800ms
负载波动图									

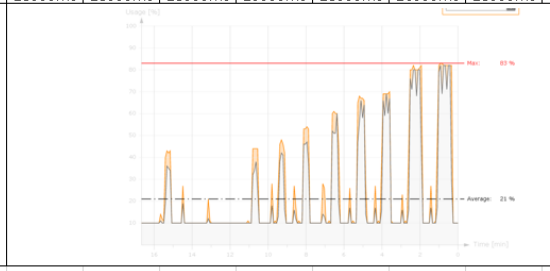
- 200ms循环周期

.gz格式200ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25217KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→44%	10%→47%	10%→54%	10%→61%	10%→67%	10%→70%	10%→82%	10%→83%
iCycleCount	85	83	87	94	107	123	133	196	215
压缩完成时间	17000ms	16600ms	17400ms	18800ms	21400ms	24600ms	26600ms	39200ms	43000ms
负载波动图									

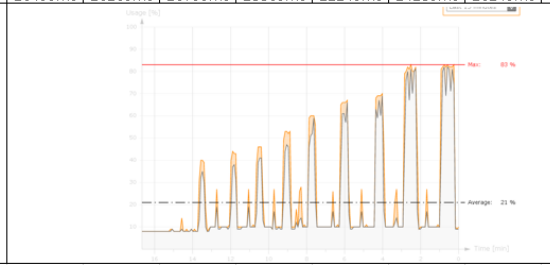
- 500ms循环周期

.gz格式500ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25217KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→44%	10%→48%	10%→54%	10%→61%	10%→67%	10%→71%	10%→82%	10%→84%
iCycleCount	35	34	35	39	44	50	54	79	87
压缩完成时间	17500ms	17000ms	17500ms	19500ms	22000ms	25000ms	27000ms	39500ms	43500ms
负载波动图									

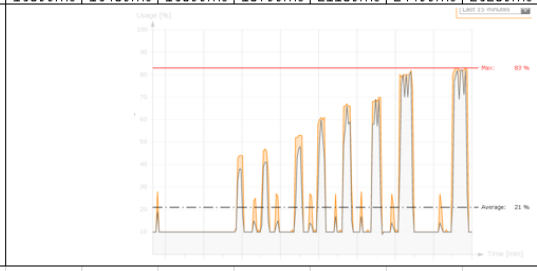
- 1000ms循环周期

.gz格式1000ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25217KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→44%	10%→48%	10%→54%	10%→61%	10%→68%	10%→70%	10%→82%	10%→84%
iCycleCount	18	18	18	20	23	26	28	40	44
压缩完成时间	18000ms	18000ms	18000ms	20000ms	23000ms	26000ms	28000ms	40000ms	44000ms
负载波动图									

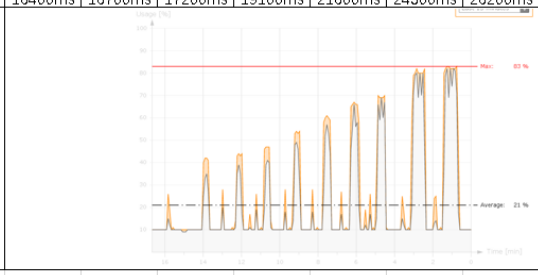
- .tar.gz格式测试结果
  - 20ms循环周期

.tar.gz格式20ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25218KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→40%	10%→44%	10%→46%	10%→53%	10%→60%	10%→67%	10%→70%	10%→83%	10%→83%
iCycleCount	820	813	835	928	1057	1209	1307	1946	2134
压缩完成时间	16400ms	16260ms	16700ms	18560ms	21140ms	24180ms	26140ms	38920ms	42680ms
负载波动图									

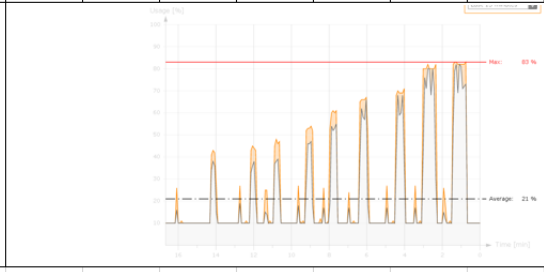
  - 50ms循环周期

.tar.gz格式50ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25218KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→42%	15%→44%	15%→47%	15%→53%	15%→61%	15%→67%	15%→70%	15%→82%	15%→83%
iCycleCount	326	329	336	374	423	488	525	775	851
压缩完成时间	16300ms	16450ms	16800ms	18700ms	21150ms	24400ms	26250ms	38750ms	42500ms
负载波动图									

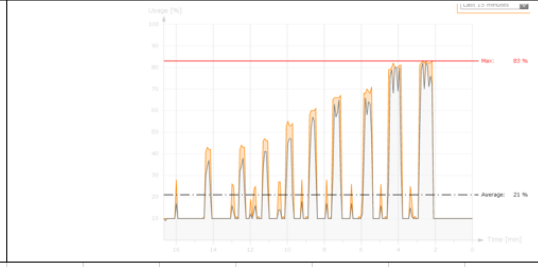
- 100ms循环周期

.tar.gz格式100ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25218KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→42%	10%→44%	10%→47%	10%→54%	10%→61%	10%→67%	10%→70%	10%→82%	10%→83%
iCycleCount	164	167	172	191	216	245	262	389	427
压缩完成时间	16400ms	16700ms	17200ms	19100ms	21600ms	24500ms	26200ms	38900ms	42700ms
负载波动图									

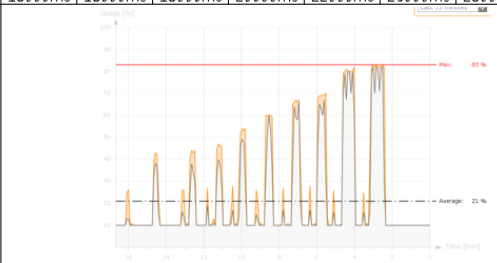
- 200ms循环周期

.tar.gz格式200ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25218KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→45%	10%→48%	10%→54%	10%→61%	10%→67%	10%→71%	10%→82%	10%→83%
iCycleCount	83	84	85	94	107	123	131	196	216
压缩完成时间	16600ms	16800ms	17000ms	18800ms	21400ms	24600ms	26200ms	39200ms	43200ms
负载波动图									

- 500ms循环周期

.tar.gz格式500ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25218KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→44%	10%→47%	10%→55%	10%→61%	10%→67%	10%→71%	10%→82%	10%→83%
iCycleCount	35	34	35	39	43	50	54	79	86
压缩完成时间	17500ms	17000ms	17500ms	19500ms	21500ms	25000ms	27000ms	39500ms	43000ms
负载波动图									

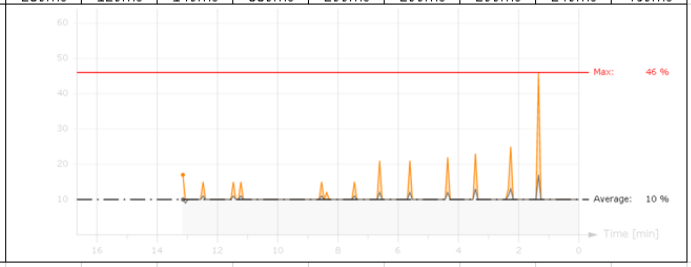
- 1000ms循环周期

.tar.gz格式1000ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	112051KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	27231KB	25218KB	23755KB	22084KB	20611KB	19679KB	19190KB	18850KB	18838KB
不同压缩等级等于负载波动情况	10%→43%	10%→44%	10%→47%	10%→54%	10%→60%	10%→67%	10%→70%	10%→82%	10%→83%
iCycleCount	18	18	18	20	22	26	28	40	44
压缩完成时间	18000ms	18000ms	18000ms	20000ms	22000ms	26000ms	28000ms	40000ms	44000ms
负载波动图									

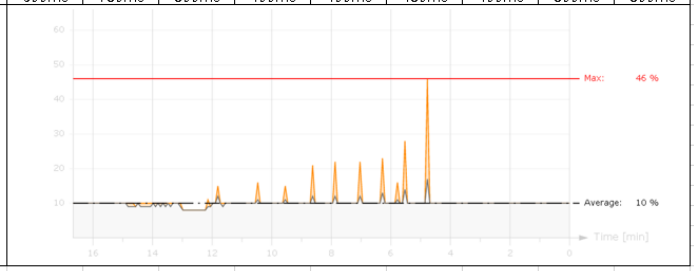
4.3.1.2 小文件测试结果

小文件测试结果

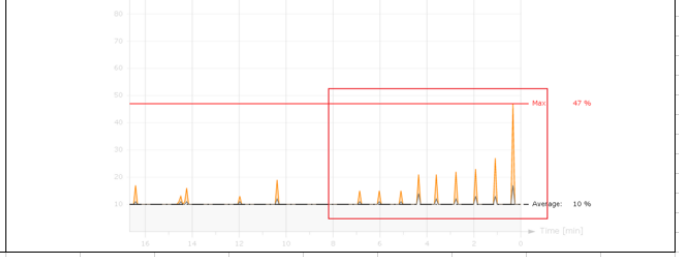
- .gz格式测试结果
  - 20ms循环周期

.gz格式20ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	67KB	60KB	51KB	50KB
不同压缩等级等于负载波动情况	10%→15%	10%→15%	10%→15%	10%→21%	10%→21%	10%→22%	10%→23%	10%→25%	10%→46%
iCycleCount	14	6	7	19	10	10	10	12	23
压缩完成时间	280ms	120ms	140ms	380ms	200ms	200ms	200ms	240ms	460ms
负载波动图									

  - 50ms循环周期

.gz格式50ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	67KB	60KB	51KB	50KB
不同压缩等级等于负载波动情况	10%→15%	10%→16%	10%→15%	10%→21%	10%→22%	10%→22%	10%→23%	10%→28%	10%→46%
iCycleCount	12	15	6	8	8	9	8	10	10
压缩完成时间	600ms	750ms	300ms	400ms	400ms	450ms	400ms	500ms	500ms
负载波动图									

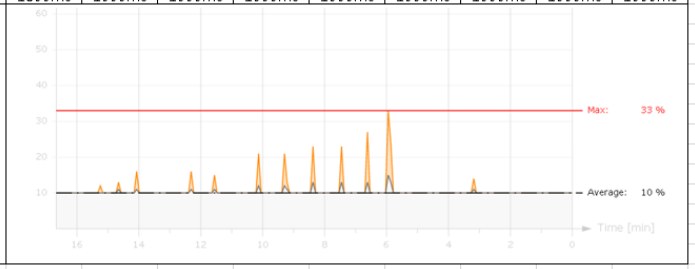
  - 100ms循环周期

.gz格式100ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	67KB	60KB	51KB	50KB
不同压缩等级等于负载波动情况	10%→15%	10%→15%	10%→15%	10%→21%	10%→21%	10%→22%	10%→23%	10%→27%	10%→47%
iCycleCount	3	3	2	5	3	3	3	4	6
压缩完成时间	300ms	300ms	200ms	500ms	300ms	300ms	300ms	400ms	600ms
负载波动图									

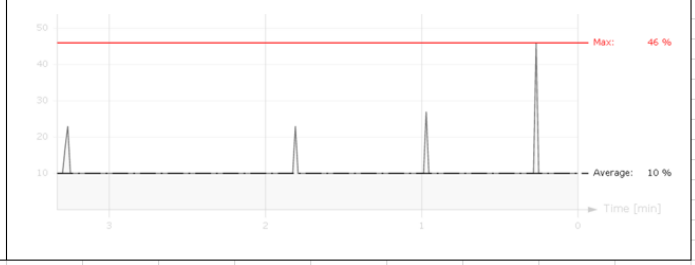
- 200ms循环周期

.gz格式200ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	67KB	60KB	51KB	50KB
不同压缩等级等于负载波动情况	10%→15%	10%→15%	10%→15%	10%→21%	10%→21%	10%→22%	10%→23%	10%→22%	10%→47%
iCycleCount	4	2	2	2	2	2	2	3	4
压缩完成时间	800ms	400ms	400ms	400ms	400ms	400ms	400ms	600ms	800ms
负载波动图									

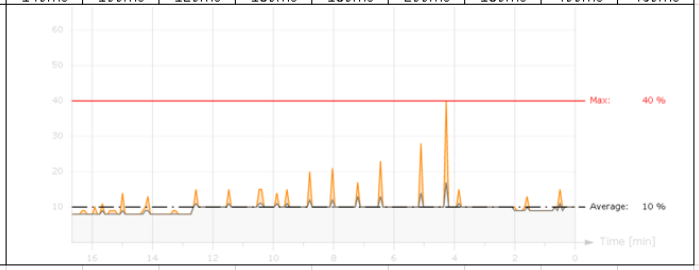
- 500ms循环周期

.gz格式500ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	67KB	60KB	51KB	50KB
不同压缩等级等于负载波动情况	10%→16%	10%→16%	10%→16%	10%→21%	10%→21%	10%→23%	10%→23%	10%→27%	10%→33%
iCycleCount	3	2	2	2	2	2	2	2	2
压缩完成时间	1500ms	1000ms	1000ms	1000ms	1000ms	1000ms	1000ms	1000ms	1000ms
负载波动图									

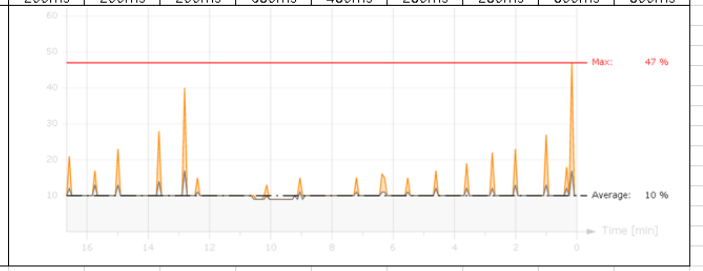
- 1000ms循环周期

.gz格式1000ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	67KB	60KB	51KB	50KB
不同压缩等级等于负载波动情况	10%→16%	10%→16%	10%→16%	10%→21%	10%→22%	10%→23%	10%→23%	10%→27%	10%→46%
iCycleCount	2	2	2	2	2	2	2	2	2
压缩完成时间	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms
负载波动图									

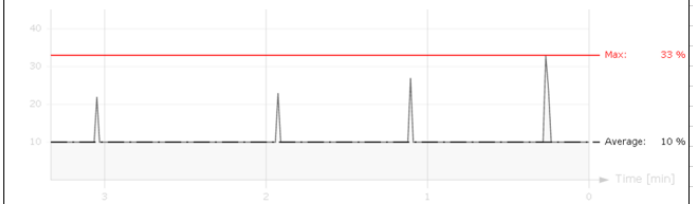
- .tar.gz格式测试结果
  - 20ms循环周期

.tar.gz格式20ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	68KB	60KB	51KB	51KB
不同压缩等级等于负载波动情况	10%→15%	10%→15%	10%→15%	10%→20%	10%→21%	10%→21%	10%→23%	10%→28%	10%→40%
iCycleCount	7	5	6	9	9	10	9	20	23
压缩完成时间	140ms	100ms	120ms	180ms	180ms	200ms	180ms	400ms	460ms
负载波动图									

  - 50ms循环周期

.tar.gz格式50ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	68KB	60KB	51KB	51KB
不同压缩等级等于负载波动情况	10%→15%	15%→16%	15%→16%	15%→17%	15%→19%	15%→22%	15%→23%	15%→27%	15%→47%
iCycleCount	4	4	4	13	9	5	5	6	10
压缩完成时间	200ms	200ms	200ms	650ms	450ms	250ms	250ms	300ms	500ms
负载波动图									

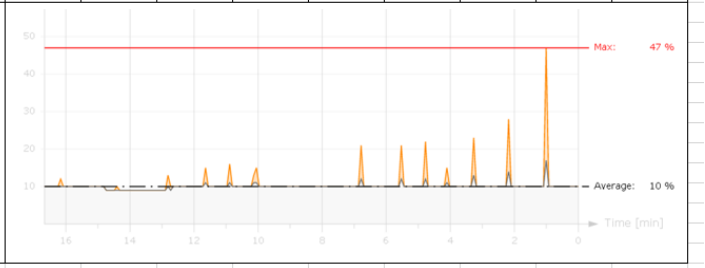
  - 100ms循环周期

.tar.gz格式100ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	68KB	60KB	51KB	51KB
不同压缩等级等于负载波动情况	10%→15%	10%→15%	10%→15%	10%→21%	10%→21%	10%→22%	10%→23%	10%→27%	10%→33%
iCycleCount	2	3	2	3	3	3	5	4	6
压缩完成时间	200ms	300ms	200ms	300ms	300ms	300ms	500ms	400ms	600ms
负载波动图									

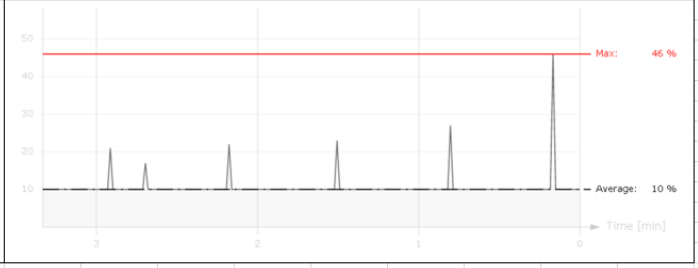
  - 200ms循环周期

.tar.gz格式200ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	68KB	60KB	51KB	51KB
不同压缩等级等于负载波动情况	10%→13%	10%→16%	10%→16%	10%→21%	10%→21%	10%→22%	10%→23%	10%→27%	10%→37%
iCycleCount	4	4	2	2	2	3	2	3	10
压缩完成时间	800ms	800ms	400ms	400ms	400ms	600ms	26200ms	600ms	2000ms
负载波动图									

- 500ms循环周期

.tar.gz格式500ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	68KB	60KB	51KB	51KB
不同压缩等级等于负载波动情况	10%→15%	10%→16%	10%→16%	10%→21%	10%→21%	10%→22%	10%→23%	10%→28%	10%→47%
iCycleCount	3	2	2	2	2	2	2	2	2
压缩完成时间	1500ms	1000ms	1000ms	1000ms	1000ms	1000ms	1000ms	1000ms	1000ms
负载波动图									

- 1000ms循环周期

.tar.gz格式1000ms循环周期下不同压缩等级情况统计									
原始未压缩文件大小	4170KB								
压缩等级	1	2	3	4	5	6	7	8	9
不同压缩等级对应压缩结果	110KB	85KB	76KB	91KB	75KB	68KB	60KB	51KB	51KB
不同压缩等级等于负载波动情况	10%→16%	10%→16%	10%→16%	10%→21%	10%→21%	10%→22%	10%→23%	10%→27%	10%→46%
iCycleCount	2	2	2	2	2	2	2	2	2
压缩完成时间	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms	2000ms
负载波动图									

4.3.2 总结

总结

在进行压缩时：

- **不同的循环周期下，文件大小相同时：相同压缩等级时，压缩时间和负载波动大致相同；**
- **相同的循环周期下，文件大小相同时：不同的压缩等级，负载的波动不同；压缩等级越大，负载波动越大；**
- **相同的循环周期下，文件大小不同时：文件越大，负载波动越大，压缩时间越长。**

5 附件

相关文档

5.1 测试程序

测试相关文件

测试程序：ASZipTestProgram.7z

5.2 ASHelp中的例子

Ansi C:

```
VAR
	zipArchive_0 : zipArchive;
	zipExtract_0 : zipExtract;
	archiveDevice1 : STRING[80];
	archiveDevice : STRING[80];
	archiveFile : STRING[80];
	srcDevice : STRING[80];
	srcFile : STRING[80];
	options : STRING[80];
	step : DINT;
END_VAR
```

```
#include <bur/plctypes.h>
#ifdef _DEFAULT_INCLUDES
#include <AsDefault.h>
#endif
void _INIT SourceAsZipInit(void)
{
	/* Initializes the parameters for zipArchive and zipExtract
	file devices: UPDATE, UPDATE1, EXTRACT */

	strcpy(archiveDevice, "UPDATE");
	strcpy(archiveDevice1, "UPDATE1");
	strcpy(archiveFile, "myFile.tar.gz");
	strcpy(srcDevice, "EXTRACT");
	strcpy(srcFile, "V100");
	strcpy(options, "/COMPRESSION_LEVEL=7");	/*Default = "/COMPRESSION_LEVEL=6" */
}
void _CYCLIC SourceAsZipCyclic(void)
{
	switch (step)
	{
	case 0: break;
/*****************************************************************************************/
```

```
/* zipArchive, sets step to 10 and starts archiving files */
/*****************************************************************************************/
	case 10:
		/*Specifies the parameters for the function block*/
		zipArchive_0.enable = 1;
		zipArchive_0.pArchiveDevice = (UDINT) archiveDevice;
		zipArchive_0.pArchiveFile = (UDINT) archiveFile;
		zipArchive_0.pSrcDevice = (UDINT) srcDevice;
		zipArchive_0.pSrcFile = (UDINT) srcFile;
		zipArchive_0.pOptions = (UDINT) options;
		step = 20;
		break;
	case 20:
		/*Calls function block until status = 0 or an error occurs*/
		zipArchive(&zipArchive_0);
		if (zipArchive_0.status != ERR_FUB_BUSY)
		{
			if (zipArchive_0.status == ERR_OK)
				step = 1000;
			else
				step = 999;
		}
		break;
/*****************************************************************************************/
/* zipExtract, sets step to 30 and starts extracting files */
/*****************************************************************************************/
	case 30:
		/*Specifies the parameters for the function block*/
		zipExtract_0.enable = 1;
		zipExtract_0.pArchiveDevice = (UDINT) archiveDevice1;
		zipExtract_0.pArchiveFile = (UDINT) archiveFile;
		zipExtract_0.pOutDevice = (UDINT) srcDevice;
		zipExtract_0.pOutFolder = (UDINT) srcFile;
		zipExtract_0.pOptions = (UDINT) options;
		step = 40;
		break;
	case 40:
		/*Calls function block until status = 0 or an error occurs*/
		zipExtract(&zipExtract_0);
		if (zipExtract_0.status != ERR_FUB_BUSY)
		{
			if (zipExtract_0.status == ERR_OK)
				step = 1000;
			else
				step = 999;
		}
		break;
	}
}
```