TIETO- JA SÄHKÖTEKNIIKAN TIEDEKUNTA

ELEKTRONIIKAN JA TIETOLIIKENNETEKNIIKAN TUTKINTO-OHJELMA

KANDIDAATINTYÖ

PAKKAUSMENETELMIEN VERTAILU SULAUTETUSSA JÄRJESTELMÄSSÄ

Tekijä Samu Lähdesmäki

Ohjaaja Antti Mäntyniemi

Heinäkuu 2021

Lähdesmäki S. (2021) Pakkausmenetelmien vertailu sulautetussa järjestelmässä. Oulun yliopisto, tieto- ja sähkötekniikan tiedekunta, elektroniikan ja tietoliikennetekniikan tutkinto-ohjelma. Kandidaatintyö, 34 s.

TIIVISTELMÄ

Työssä vertailtiin eri häviöttömien pakkausmenetelmien soveltuvuutta datan pakkaukseen sulautetussa järjestelmässä. Tavoitteena oli löytää rajoitetulla muistilla ja laskentateholla varustettuun laitteeseen parhaiten soveltuva pakkausmenetelmä. Työssä etsittiin ensin taustatutkimuksen avulla kolme käyttötarkoitukseen todennäköisimmin soveltuvinta menetelmää, jotka olivat FastLZ, MiniZ ja Heatshrink.

Työssä havaittiin selkeitä eroja eri pakkausmenetelmien vahvuuksissa: FastLZ oli nopea, MiniZ tehokas ja Heatshrink pärjäisi todella vähällä muistilla. Kuitenkin tutkimuksessa käytetyssä ympäristössä MiniZ osoittautui selkeästi monikäyttöisimmäksi ja tehokkaimmaksi.

Avainsanat: pakkausalgoritmi, pakkausmenetelmä, sulautettu järjestelmä, MiniZ, FastLZ, Heatshrink.

Lähdesmäki S. (2021) Comparison of compression methods in an embedded system. University of Oulu, Degree Programme in Electronics and Communications Engineering. Bachelor's thesis, 34 pages

ABSTRACT

In this thesis different compression methods were assessed for suitability for data compression in an embedded system. The goal was to find the best algorithm for use in a device with limited memory and processing power. First a study was conducted to find three methods that were most likely to be suitable for the intended use: FastLZ, MiniZ and Heatshrink.

Clear differences were observed between the strengths of the methods: FastLZ was fast, MiniZ was efficient and Heatshrink could function in a very low-memory system. In conclusion however, MiniZ was clearly the most efficient and best all-around solution in the environment of this study.

Key words: compression algorithm, compression method, embedded system, MiniZ, FastLZ, Heatshrink.

SISÄLLYSLUETTELO

ALKULAUSE

Aloitin tämän työn teknisen osan syystalvella 2020 ja se valmistui kesällä 2021. Työn aikana opin paljon paitsi pakkausmenetelmistä myös ohjelmoinnista ja sulautetuista järjestelmistä yleisesti. Haluan kiittää Bittium Oyj:tä mielenkiintoisesta aiheesta ja mahdollisuudesta käyttää projektiin aikaa enemmän kuin muuten olisi ollut mahdollista. Lisäksi haluan kiittää kaikkia minua auttaneita henkilöitä saamastani tuesta.

Oulussa 6.7.2021

Samu Lähdesmäki

LYHENTEIDEN JA MERKKIEN SELITYKSET

EKG	elektrokardiografia, sydänfilmi
EDF	European Data Format -tiedostomuoto
BLE	Bluetooth Low Energy
IoT	Internet of Things, esineiden internet
RAM	Random Access Memory
Mt	megatavu
kt, kB	kilotavu, kilobyte

1 JOHDANTO

Esineiden internetin aikakaudella mitattavan datan määrä ja sen siirtämisen tarve on kasvanut räjähdysmäisesti. Kun tämä yhdistetään jatkuvaan tarpeeseen pienentää käytettäviä komponentteja, kysyntä datan resurssitehokkaaseen pakkaamiseen soveltuville menetelmille on suurta.

Tämän työn tarkoitus on tutkia ja vertailla eri häviöttömien pakkausmenetelmien käyttöä sulautetussa järjestelmässä. Työ koostuu teoria-, tutkimus- ja vertailuosasta. Teoriaosassa käsitellään pakkausalgoritmien toimintaa yleisesti ja niiden erilaisia tyyppejä sekä arviointimenetelmiä. Tutkimusosiossa käydään tarkemmin läpi käyttötarkoitukseen soveltuvia pakkausmenetelmiä sekä mitkä niistä lopulta valittiin testaukseen ja miksi. Testausvaiheessa valitut menetelmät implementoidaan STM32-järjestelmään, jossa niitä ajetaan eri parametreilla, otetaan mittaustulokset ylös ja vertaillaan. Menetelmiä vertaillaan niiden pakkaustehossa suhteutettuna pakkaamiseen kuluvaan aikaan.

2 TEORIA

2.1 Pakkausalgoritmeista yleisesti

Tiedon pakkaaminen tarkoittaa menetelmiä, joiden avulla datan sisältämä informaatio korvataan lyhyemmällä kuvauksella. Kaikkia pakkausalgoritmeja voidaan kuvata seuraavasti: etsitään toistuvuutta ja korvataan toistuvat kohdat lyhyemmällä esitystavalla. Esimerkiksi suomenkielisessä tekstidatassa kirjain A voidaan korvata lyhyemmällä kuvauksella kuin kirjain Z, sillä se toistuu useammin. Tavat, joilla toistuvuutta etsitään ja joilla datan kuvaustapaa lyhennetään, vaihtelevat menetelmästä toiseen. Tietyn pakkausmenetelmän soveltuvuus tiettyyn tehtävään riippuu vahvasti pakattavan datan tyypistä. Yleisesti voidaan sanoa, että mitä erikoistuneempi algoritmi on, sitä tehokkaammin se kykenee pakkaamaan annettua dataa. EKG-käyrien pakkaamiseen erikoistunut algoritmi pakkaa niitä hyvin todennäköisesti tehokkaammin, kuin yksinkertainen yleiskäyttöinen algoritmi. [1]

Pakkausmenetelmiä on siis monenlaisia, jokaisella omat vahvuutensa ja heikkoutensa. Näitä menetelmiä on jaoteltu eri ominaisuuksien mukaan.

2.2 Häviötön ja häviöllinen pakkaaminen

Pakkausalgoritmit on jaoteltu ylemmällä tasolla häviöllisiin ja häviöttömiin. Häviölliset pakkausmenetelmät approksimoivat käytettyä dataa, mikä johtaa siihen, että osa informaatiosta menetetään. Tällaiset menetelmät soveltuvat esimerkiksi äänen ja kuvan pakkaamiseen, sillä käyttötavasta riippuen ihmissilmä tai -korva ei välttämättä erota informaation menetystä. Joissakin käyttötarkoituksissa, esimerkiksi äänen tapauksessa, lopputulos voi kuulostaa jopa paremmalta kuin alkuperäinen pakkaamaton data, kun erilaiset häiritsevät taustaäänet häviävät pakkaamisen yhteydessä. Häviölliset pakkausalgoritmit saavuttavat monissa käyttötarkoituksissa paremman pakkaustehon, mutta pakatusta tiedostosta ei voida enää palauttaa alkuperäistä dataa. [3]

Valokuvaajille tuttu häviöllinen pakkausmenetelmä on diskreetillä kosinimuunnoksella toimiva JPEG, jota käyttämällä menetetään raakakuvaformaattiin verrattuna kuvan sisältämää informaatiota, mutta lopputulos sellaisenaan voi näyttää ihmissilmään paremmalta. Raakakuvien säilyttämä suurempi informaatiomäärä tarjoaa kuitenkin laajemmat mahdollisuudet kuvien jälkikäsittelyyn. [4]

Häviöttömiä pakkausmenetelmiä käytetään, kun datan täytyy purettuna olla täsmälleen samanlaista kuin alkuperäinen. Tällaisia datamuotoja ovat esimerkiksi tekstimuotoinen data sekä lääketieteellinen mittausdata. Lopputulos ei ole hyväksyttävä, jos esimerkiksi kirjasta puuttuu purkamisen jälkeen sanoja, tai kirjaimia on vaihtunut toisiin. Tuttuja häviöttömiä pakkausalgoritmeja käyttäviä sovelluksia ovat esimerkiksi ZIP- ja RAR -tiedostomuodot, jotka käyttävät useita eri pakkausmenetelmiä. [2]

Tämä työ keskittyy häviöttömään pakkaukseen. Seuraavaksi esitellään erilaisia häviöttömissä pakkausalgoritmeissa käytettyjä tekniikoita.

2.3 Häviöttömiä pakkausalgoritmitekniikoita

2.3.1 Tilastollinen lähestymistapa

Tilastollisessa lähestymistavassa tiedosto analysoidaan merkki merkiltä ja etsitään yleisimpiä kirjaimia, sanoja tai muita kohteita. Pakkaus tapahtuu kahdessa osassa: mallinnuksessa ja koodauksessa. Mallinnuksessa jokaiselle kohteelle lasketaan todennäköisyys. Koodauksessa kohteille annetaan koodisana, tyypillisesti binääri, joka on sitä lyhyempi, mitä suurempi sille laskettu todennäköisyys on [2]. Purkaminen tapahtuu näiden todennäköisyyksien avulla. Lähestymistapa soveltuu monenlaiseen dataan, ja sitä käytetään osana useita häviöttömiä ja häviöllisiä pakkausalgoritmeja muiden lähestymistapojen rinnalla. [1]

2.3.2 Sanastollinen lähestymistapa

Sanastollisessa lähestymistavassa tekstin tai datajoukon yleisimmät osat, esimerkiksi sanat tai lauseet, korvataan binääriluvuilla. Nämä osat ja niitä vastaavat luvut tallennetaan sanakirjaan, joka tallennetaan pakatun tiedoston yhteyteen. Lähestymistapa toimii erityisen hyvin tekstidatan kanssa, sillä jopa kokonaisia lauseita saadaan merkittyä vain parilla tavulla. Koska sanakirjan tulee olla pakatun tiedoston mukana purkamista varten, kasvaa tiedoston koko jonkin verran. Oikein käytettynä lopputulos on kuitenkin alkuperäistä tiedostoa pienempi. [2]

2.3.3 Toistuvien kokonaisuuksien etsintä

Joskus datassa voi olla sellaisia toistuvia jaksoja, jotka voidaan helposti esittää pakatummassa muodossa. Lähestymistavasta paras esimerkki on Run-Length Encoding, eli jakson pituuden

koodaus, jossa tutkitaan, onko datassa useita samanlaisia merkkejä peräkkäin. Tällainen jakso merkitään merkkijonolla, jossa on ensin toistuvien merkkien lukumäärä kokonaislukuna ja perässä toistuva merkki. Näin esimerkiksi BBBBBBWWWW voidaan merkitä 6B4W. Lähestymistapaa voidaan hyödyntää esimerkiksi yksinkertaisten mustavalkokuvien kanssa, jolloin kuva voidaan ilmaista mustina (B) ja valkoisina (W) pikseleinä. [5]

2.3.4 Mukautuvat ratkaisut

Mukautuvassa pakkauksessa kohteiden todennäköisyyksiä ei lasketa ainoastaan globaalisti koko tiedostossa vaan myös sen eri osissa. Esimerkiksi sanastollisesti pakattavassa kirjassa, jossa henkilö X esitellään vasta luvussa seitsemän, X:n nimi esiintyy todennäköisemmin luvusta seitsemän eteenpäin kuin luvuissa 1-6. Täten lukukohtainen sanakirja voi saavuttaa paremman pakkaustehon. [1]

2.4 Pakkausalgoritmien vertailutavat

Työn tarkoitus on vertailla eri pakkausalgoritmien sopivuutta käyttöön sulautetussa laitteessa. Tätä varten täytyy löytää ominaisuudet, joita vertailemalla voidaan valita testattavat algoritmit satojen joukosta, sekä testeissä mitattavat suureet. Seuraavaksi käydään läpi joitain tapoja, joiden avulla algoritmeja voidaan laittaa paremmuusjärjestykseen.

2.4.1 Pakkaussuhde

Pakkaussuhteella tarkoitetaan sitä, kuinka paljon pakattu tiedosto pienenee verrattuna alkuperäiseen. Suhdetta kuvataan yleensä joko suoraan pakkaamattoman ja pakatun koon suhteena tai prosentuaalisesti säästettynä tilana. Esimerkiksi jos 50 Mt tiedostosta tulee pakkaamisen seurauksena 20 Mt, on suhde 5:2 ja tilansäästö prosentteina 60 %. [1]

$$Pakkaussuhde = \frac{Pakkaamaton\ koko}{Pakattu\ koko}$$

$$Säästetty\ tila\ prosentteina = 1 - \frac{Pakattu\ koko}{Pakkaamaton\ koko} * 100\ \%$$

2.4.2 Overhead

Overhead eli ”kiinteät kustannukset” (kauppatieteen termi) tarkoittavat sitä osaa pakatun tiedoston koosta, joka ei sisällä varsinaista alkuperäisen tiedoston informaatiota pakattuna. Esimerkiksi sanastollisissa pakkausmenetelmissä täytyy pakkaamiseen käytetyn sanakirjan olla purkuvaiheessa purkuohjelman saatavilla, jotta se osaa yhdistää oikeat indeksit oikeisiin merkkijonoihin. Tämä ylimääräinen data kasvattaa pakatun tiedoston kokoa, joten overhead pyritään pitämään mahdollisimman pienenä. Joissain tapauksissa voi pakatun tiedoston mukana olla kokonainen pieni purkuohjelma, joka muuttaa tiedoston ”itsepurkavaksi arkistoksi” (self-extracting archive). [1]

Tämän työn testeissä overhead sisältyy pakkaussuhteeseen, sillä pakattujen tiedostojen kokoja vertaillaan erottelematta niistä overheadia.

2.4.3 Pakkausnopeus ja purkunopeus

Pakkausnopeus ja purkunopeus tarkoittavat yksinkertaisesti aikaa, joka ohjelmalla kuluu tietynkokoisen tiedoston pakkaamiseen tai purkamiseen. Kuten kaikissa muissakin tietokoneella suoritettavissa operaatioissa, nämä nopeudet riippuvat käytettävissä olevasta laskentatehosta ja muistista.

2.4.4 Vaihtelu

Vaihtelulla tarkoitetaan sitä, kuinka tasaisesti pakkausmenetelmä suoriutuu erityyppisten tiedostojen pakkaamisesta. Suunniteltu käyttötarkoitus määrittää, kuinka paljon painoarvoa tälle annetaan: yleiskäyttöisen pakkausohjelman tulee kyetä pakkaamaan tehokkaasti monenlaisia tiedostoja, kun taas kuvien pakkaamiseen tarkoitetun algoritmin ei tarvitse pakata musiikkia tehokkaasti. [1]

2.4.5 Patentin status

Monet pakkausalgoritmit ovat kokonaan tai osittain patentoituja. Patenttistatuksen vaikutus sopivan algoritmin valintaan riippuu algoritmia käyttävän ohjelman käyttötarkoituksesta. Kaupalliseen tarkoitukseen menevissä sovelluksissa tulee tähän kiinnittää erityistä huomiota. [6]

3 TUTKIMUS

3.1 Työn tavoitteiden määrittelystä

Tämän työn tavoitteena on vertailla sulautetussa järjestelmässä toimivia pakkausmenetelmiä. Koska pakkausalgoritmien resurssivaatimuksilla ja suorituskyvyillä eri tiedostotyyppien kanssa on eroja, täytyy ensin määritellä käytettävä laitteisto ja pakattavat tiedostotyypit. Sen jälkeen selvitetään, mitä ominaisuuksia testeihin valittavilta menetelmiltä vaaditaan ja minkä mukaan niitä vertaillaan.

3.2 Laitteisto

Sulautetussa järjestelmässä suoritettavaan työhön täytyi valita laite. Jotta tutkimustulokset olisivat mahdollisimman yleishyödyllisiä, oli kehitysalustan oltava mahdollisimman yleinen ja sekä muistiltaan että laskentateholtaan keskiverto. Lisäksi toivottavaa oli hyvät liitännät, erityisesti SD-korttipaikka ja helppo USB-debuggaus.

Näillä kriteereillä päädyttiin ARM Cortex-M4F -pohjaiseen STM32F412G-evaluointialustaan. ARM Cortex -arkkitehtuuriin perustuvat laitteet ovat todella yleisiä ja M4F on näistä tehoiltaan ylempää keskiluokkaa. ST-electronicsin STM32-kehitysalustat ovat hyvin dokumentoituja ja äärimmäisen suosittuja, ja F412G vastasi liitännöiltään työn tarpeita parhaiten.

Työn kannalta oleelliset tiedot STM32F412G-kehitysalustasta:

- Arm Cortex-M4F arkkitehtuuri
- Flash-muistia 1 Mt
- Ram 256 kt
- ST-LINK/V2-1 -integroitu debuggeri
- Micro SD -korttipaikka [7]

3.3 Kohdetiedostot

Koska työssä tutkitaan sulautettuja järjestelmiä, tuli testauksessa pakattavien tiedostotyyppien olla niiden käyttötarkoitukselle tyypillisiä. Tyypillinen teollisen internetin laite kerää sensoridataa lähetettäväksi erilliselle päätelaitteelle. Sitä edustaa tässä työssä monenlaista

lääketieteellistä sensoridataa sisältävät EDF-tiedostot. Lisäksi vertailun vuoksi työssä pakattiin laajasti käytössä olevia erityyppisiä testitiedostoja, jotka ovat:

- asyoulik.txt, englanninkielinen tekstitiedosto (122 kt)
- fields.c, yksinkertainen C-kielinen lähdekoodi (10,8 kt)
- kennedy.xls, Excel-taulukko (0,98 kt)
- landing.wav, äänitiedosto ensilaskeutumisesta Kuuhun (1,19 Mt)
- lenna.tif, yleisesti pakkaustestauksessa käytettävä valokuvatiedosto (768 kt)
- lenna_hires.jpg, sama kuva eri muodossa (557 kt)
- Mona_Lisa.png, valokuva Da Vincin La Gioconda -maalauksesta png-muodossa (12,4 Mt)
- T_complex.EDF, monenlaisilta sensoreilta saatuja biosignaaleja sisältävä tiedosto. Sisältää enemmän sensoridataa kuin saman tiedoston simple-versiot (3,30 Mt)
- T_simple.EDF (8,55 Mt)
- T_simple_L.EDF (42,0 Mt)
- video.mjpeg, yksinkertainen kuvasarjasta koottu videotiedosto, vertailukelpoinen esimerkiksi valvontakamerastriimin kanssa (4,57 Mt)

3.4 Pakkausmenetelmät

Työhön valittiin vertailtavaksi kolme pakkausmenetelmää. Valinta suoritettiin listaamalla kaikki potentiaaliset häviöttömät pakkausmenetelmät ja karsimalla niistä seuraavin kriteerein:

1. Laaja ilmainen lisenssi myös kaupallisessa käyttötarkoituksessa
2. Alhaiset resurssivaatimukset
3. Tiedostokooltaan pieni ja suhteellisen helppokäyttöinen C-kirjasto
4. Tunnetusti käytetty tai käyttöön harkittu menetelmä sulautetussa järjestelmässä

Näillä kriteereillä saatiin lopulta valittua työhön sopivat pakkausmenetelmät. Erityisesti lisenssivaatimus osoittautui hyvin karsivaksi, sillä vaikka monet algoritmit ovat avointa lähdekoodia, niihin kuuluu usein vaatimuksia esimerkiksi käyttökohteen lähdekoodin avoimuudesta, mikä on usein kaupallisessa käytössä mahdotonta. Seuraavaksi esitellään lyhyesti valitut menetelmät.

3.4.1 FastLZ

FastLZ on ANSI C/C90 -implementaatio Lempel-Ziv 77 (LZ77) -algoritmista. Se soveltuu tekstien, raa'an pikselidatan tai muiden paljon toistoa sisältävien datablokkien pakkaamiseen. FastLZ on kehitetty erityisesti saavuttamaan hyvin korkea pakkaus- ja purkunopeus.

FastLZ on laajasti käytetty ja toimii monenlaisissa ympäristöissä. Se koostuu vain kahdesta tiedostosta, fastlz.c ja fastlz.h. MIT-lisenssi. [8]

3.4.2 Heatshrink

Heatshrink on varta vasten vähämuistisia sulautettuja järjestelmiä varten kehitetty pakkausmenetelmä, joka pohjautuu Lempel-Ziv-Storer-Szymanski (LZSS) -algoritmiin. Se vaatii äärimmäisen vähän muistia, jopa alle 50 tavua, ja toimii myös hyvin pienellä pakkauspuskurilla.

Heatshrink koostuu kahdesta konfigurointitiedostosta ja pakkaus- tai purkutiedostosta sekä sen otsikkotiedostosta. ISC-lisenssi. [9]

3.4.3 MiniZ

MiniZ on pakkauskirjasto, joka implementoi zlib ja Deflate -pakkausmenetelmiä. Zlib ja Deflate taas käyttävät yhdistelmää LZ77-algoritmista ja Huffman -koodauksesta.

MiniZ on käyttökelpoinen monipuolisesti erilaisiin tiedostotyyppeihin ja se sisältää korkean tason apufunktioita käytön helpottamiseksi. MiniZ koostuu vain kahdesta tiedostosta, miniz.c ja miniz.h. MIT-lisenssi. [10]

3.5 Tutkimuksessa käytetty ohjelmakoodi

Työkaluna ohjelmakoodin luomisessa toimi laitteen valmistajan oma Eclipse-pohjainen STM32Cube-kehitysympäristö. Sen avulla kortin rajapintojen ja kellojen ohjelmakoodin generointi sujui vaivattomasti ja koodin pystyi kirjoittamaan kortille suoraan kehitysympäristöstä.

Koska pakkauskirjastot eivät toimi sellaisenaan tiedostojen pakkaamiseen, täytyi jokaiselle menetelmälle implementoida omat pakkaus- ja purkufunktiot. Yksinkertaistettuna

pakkausfunktiot toimivat samalla peruskaavalla. Funktiolle syötettiin parametreina pakattavan tiedoston nimi, pakkauspuskurin koko sekä mahdollisia muita menetelmäkohtaisia parametreja. Funktio avasi pakattavan tiedoston ja loi uuden tiedoston pakattua dataa varten. Pakattava tiedosto luettiin silmukkarakenteella puskuriin pala kerrallaan, ja nämä palat pakattiin ja kirjoitettiin uuteen tiedostoon. Purkufunktiot toimivat samalla kaavalla, mutta koska tässä tutkimuksessa tutkittiin vain pakkaamista, niitä käytettiin ainoastaan pakkausfunktioiden toimivuuden verifioimiseen.

Testaaminen toteutettiin kutsumalla pakkausfunktioita main()-funktion kautta eri argumenteilla. Jokainen mahdollinen yhdistelmä toimivia parametreja käytiin läpi jokaiselle pakkausmenetelmälle ja tiedostolle. Kokonaisuudessaan ainutlaatuisia testejä tehtiin 349, ja niiden yhteenlaskettu pakkausaika oli yli 23 tuntia.

Testien tulokset saatiin talteen tallentamalla testilaitteen tietokoneen sarjaporttiin lähettämät tulostukset lokitiedostoksi TeraTerm-terminaaliemulaattorilla. Selkokielisestä lokitiedostosta muotoiltiin tarkoitusta varten tehdyllä Python-ohjelmalla tarvittava testidata .csv-tiedostoksi.

3.5.1 Tulostaulukon lukeminen

Testitulokset sisältävä .csv-taulukkotiedosto sisälsi jo kaiken tarvittavan testidatan, mutta jotta tuloksista saataisiin käyttökelpoisempaa dataa, lisättiin siihen Excelissä erinäisiä laskutoimituksia sisältäviä sarakkeita (Kuva 1). Seuraavaksi luetellaan sarakkeiden selitykset ja niiden taustalla olevat laskukaavat.

Algorithm	File	Size(MB)	Buffer(	Level/window	Time(s)	Ratio	Speed(kB/s)	Comp/Time(kB/s)	Saved(s) BLE4.1	Saved/MB(s) BLE4.1	Saved(s) BLE4.2	Saved/MB(s) BLE4.2
FastLZ	asyoulik.txt	0,13 MB	8192 B	2	0,64 s	40,56 %	203,13 kB/s	82,39 kB/s	14,43 s	110,96 s	3,29 s	25,35 s

Kuva 1: Tulostaulukon sarakkeet

1. **Algorithm:** Käytetty pakkausmenetelmä.
2. **File:** Pakatun tiedoston nimi ja tiedostopääte.
3. **Size:** Pakatun tiedoston koko megatavuina.
4. **Buffer:** Pakkauspuskurin koko.
5. **Level/Window:** Pakkausmenetelmäkohtainen parametri, eivät vertailukelpoisia keskenään. Parametrien selitykset:

- **FastLZ - Level:** Tasolla 1 nopeampi pakkaaminen, tasolla 2 parempi pakkaussuhde.
- **Heatshrink - Window size:** Ikkunan koko, josta toistuvuuksia etsitään. Kahden potenssi, eli esimerkiksi arvolla 10 ikkunan koko on $2^{10} = 1024$ tavua.
- **Miniz - Level:** Tasot 1, 9 ja 10. Mitä suurempi taso, sitä tehokkaampi mutta hitaampi pakkaaminen.

6. **Time:** Pakkaamiseen kulunut aika sekunteina.
7. **Ratio:** Säästetty tila prosentteina.
8. **Speed:** Pakkausnopeus, eli tiedoston koko jaettuna pakkaamiseen kuluneella ajalla.
9. **Comp/Time:** Tehollinen pakkausnopeus, eli **Ratio*Speed.**

Sarakkeet 10-13 esittävät säästettyä aikaa skenaariossa, jossa laite esimerkiksi mittausdatan kerättyään pakkaa sen ja tämän jälkeen lähettää sen Bluetooth-yhteydellä päätelaitteelle. Pakkaamiseen ja pakatun tiedoston lähettämiseen yhteensä kulunutta aikaa siis verrataan siihen, kauanko olisi mennyt, jos data olisi lähetetty pakkaamattomana heti päätelaitteelle. Joissain tapauksissa näiden sarakkeiden arvo voi olla negatiivinen, mikä tarkoittaa sitä, että olisi ollut nopeampaa lähettää tiedosto ilman pakkaamista.

$$Säästetty\ aika(s) = \frac{Tiedoston\ koko(kB)}{BLE\ lähetysnopeus\left(\frac{kB}{s}\right)} - \frac{Tiedoston\ koko(kB) - Tiedoston\ koko(kB) * Säästetty\ tila(\%)}{BLE\ lähetysnopeus\left(\frac{kB}{s}\right)} - Pakkausaika(\text{s})$$

Sarakkeet 10 ja 12 kertovat absoluuttiset arvot säästetylle ajalle ja sarakkeet 11 ja 13 kuvaavat laskennallista säästettyä aikaa yhtä megatavua kohden. Viitteellisenä tiedonsiirtonopeutena käytettiin BLE4.1 yhteydellä 3,5 kt/s, ja BLE4.2 yhteydellä 13,4 kt/s. Arvot perustuvat tämän työn ulkopuolella toteutettuihin testeihin. Niitä tulee kuitenkin pitää vain suuntaa antavina, johtuen muun muassa mahdollisista eroista laitteistojen suorituskyvyissä. Tämän vuoksi niitä ei käytetty tulosten arvioinnissa.

3.6 Tulokset

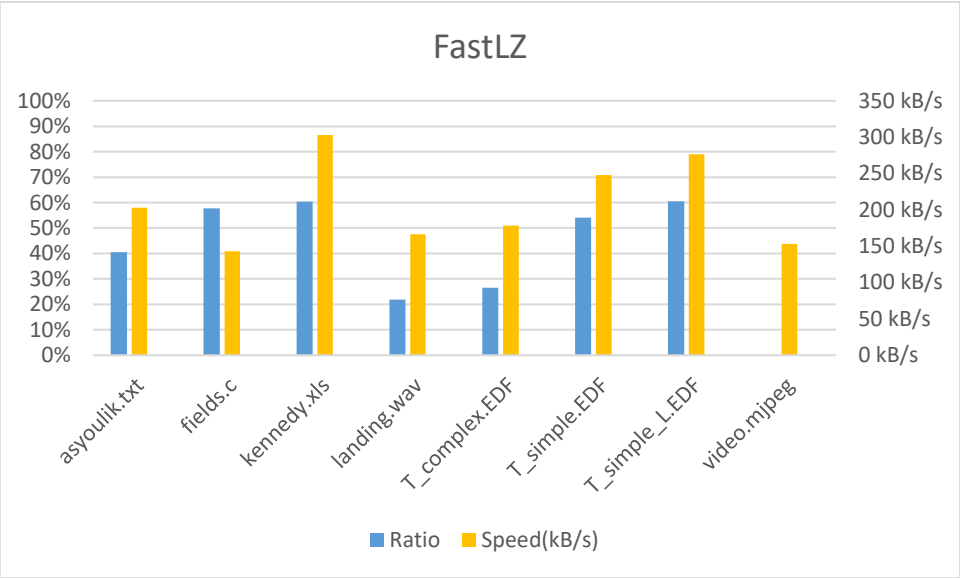
Koska testitapauksia oli jokaiselle tiedostolle ja pakkausmenetelmälle ajettu lukuisilla eri parametreilla, täytyi näistä valita mahdollisimman edustavat arvot. Keskiarvojen tai mediaanien käyttö ei tullut kysymykseen, sillä parametrien äärirajoilla tulokset saattoivat olla niin huonoja, että tulos ei olisi realistinen. Vertailtaviksi arvoiksi päätettiin ottaa jokaisen

tiedoston kohdalla kultakin pakkausmenetelmältä se testitapaus, jolla oli paras tehollinen pakkausnopeus. Tämän ollessa säästetyn tilan kertoimen ja pakkausnopeuden tulo saatiin vertailtavaksi hyvin molemmilla mittareilla pärjänneitä testitapauksia. Tarkastelemalla taulukkoa havaittiin tällä perusteella valitun datan edustavan hyvin pakkausmenetelmien suorituskykyä.

3.6.1 FastLZ

FastLZ-pakkausmenetelmän pakkausnopeus oli nimensä mukaisesti joukon korkein lähes kaikissa tapauksissa. Parhaiten se suoriutui numeerisen datan pakkaamisesta: yksinkertaisen sensoridatan ja excel-taulukon pakkaamisessa saavutettiin yli 50 % pakkausteho ja 250–300 kt/s pakkausnopeus.

Kuvien pakkaamiseen FastLZ ei kyennyt lainkaan, ja mjpeg -videodatassa tilansäästö jäi alle promillen (Kuva 2).

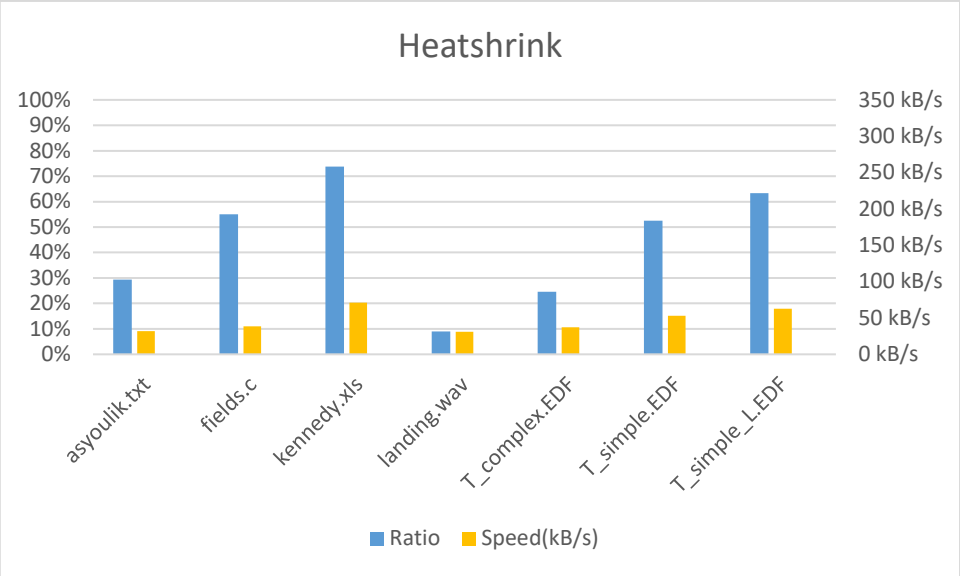


Kuva 2: FastLZ-pakkausmenetelmän tulokset tiedostoittain

3.6.2 Heatshrink

Heatshrink oli testatuista menetelmistä kaikilla tiedostotyypeillä hitain ja lähes kaikissa tapauksissa myös tehottomin. Excel-taulukon ja suuren EDF-tiedoston pakkaussuhteessa se voitti kuitenkin FastLZ-menetelmän, mutta oli näissäkin tapauksissa sitä yli neljä kertaa hitaampi. Erityisen heikosti se pärjäsi suhteessa muihin helpoksi oletetun tekstidatan

pakkaamisessa: tilansäästö jäi alle 30 prosentin (Kuva 3). Kuvien ja videon pakkaamiseen Heatshrink ei kyennyt lainkaan.

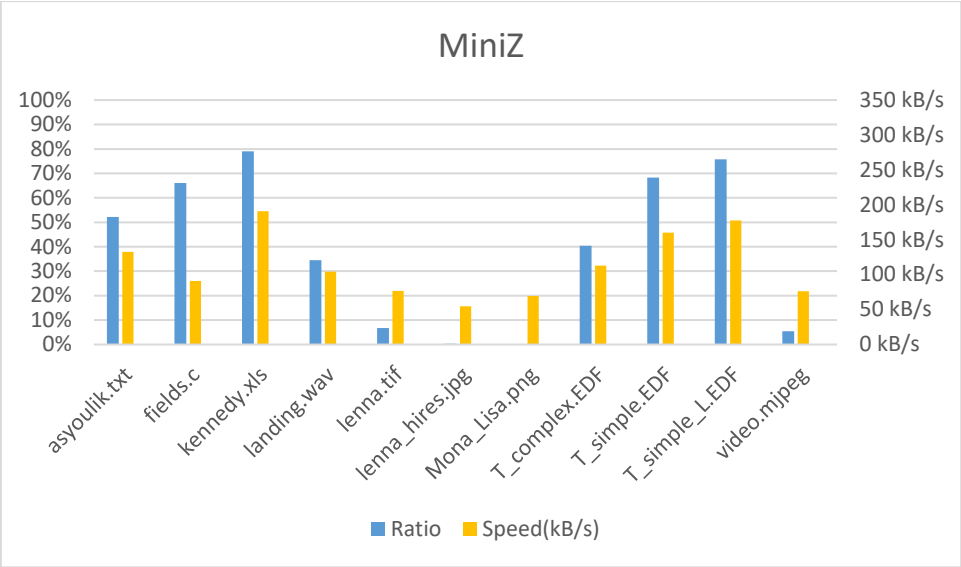


Kuva 3: Heatshrink-pakkausmenetelmän tulokset tiedostoittain

3.6.3 MiniZ

MiniZ osoittautui menetelmistä monipuolisimmaksi ja saavutti kaikissa tiedostoissa parhaan pakkaustehon. Jopa kuvan ja videon pakkaaminen onnistui, mutta .jpg ja .png -tiedostojen tapauksessa tilansäästö jäi alle puolen prosentin (Kuva 4).

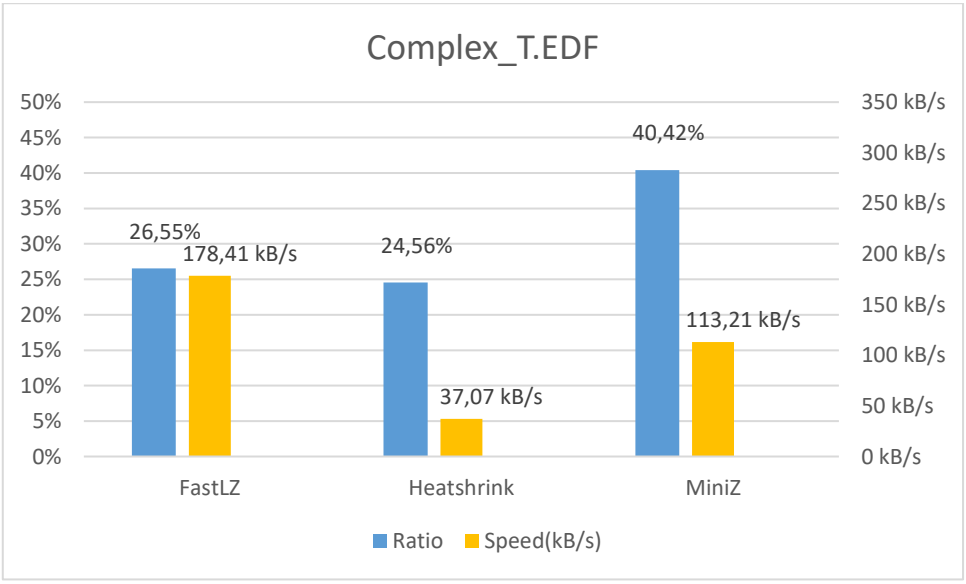
Pakkausnopeudessa MiniZ sijoittui kolmikon keskiväliin ollen kuitenkin selvästi nopeampi kuin Heatshrink.



Kuva 4: MiniZ-pakkausmenetelmän tulokset tiedostoittain.

3.6.4 Tulosten vertailu

Seuraavaksi vertaillaan pakkausmenetelmien tuloksia yhden tiedoston pakkaamisessa. Tähän valittiin monimutkaista biosignaalidataa sisältävä Complex_T.EDF, sillä se on datatyyppinä varsinkin teollisen internetin sovelluksia mietittäessä kiinnostava ja siinä eri menetelmien suhteelliset suorituskyvyt vastaavat hyvin koko tutkimuksessa todettuja tuloksia. Jokaisen menetelmän pakkaustulos pakkaustehona ja pakkausnopeutena kuvassa 5.



Kuva 5: Monimutkaisen biosignaalidatatiedoston pakkaustulokset kaikilla pakkausmenetelmillä

Pakkausteholtaan MiniZ oli kokeessa ylivoimainen saavuttaen 40,42 % pakkaustehon. FastLZ taas oli selkeästi MiniZ:aa nopeampi, mutta pakkaustehon jäädessä tämänkaltaisella monimutkaisella tiedostolla alhaiseksi, voidaan MiniZ:aa pitää onnistuneempana pakkauksessa. Heatshrink puolestaan pakkasi lähes yhtä tehokkaasti kuin FastLZ, mutta se jäi nopeudessa reilusti kahta muuta jälkeen.

4 POHDINTA

Työn tarkoituksena oli etsiä sulautetulle laitteelle parhaiten sopiva pakkausmenetelmä. Työssä selvitettiin kolmen parhaiten sopivan menetelmän nopeutta, pakkaustehoa ja yleistä soveltuvuutta testaamalla jokaisen menetelmän suoriutumista 11 erityyppisen tiedoston pakkaamisessa. Tämän jälkeen tuloksia vertailtiin ja huomiota kiinnitettiin erityisesti suoriutumiseen sensoridatan pakkaamisessa, sillä se on tyypillinen pakkauksen kohde IoT-laitteessa.

Työssä algoritmien implementointi sulautettuun järjestelmään oli välillä yllättävänkin haastavaa ja vaati monenlaisia toimenpiteitä ja paljon uuden opettelua. Itse testaus kuitenkin sujui varsin hyvin ja siinä korostuivat selkeästi eri algoritmien ominaisuudet ja vahvuudet.

MiniZ oli odotetusti yleispätevä ja suoriutui ainoana kaikkien tiedostojen pakkaamisesta, joskin kuvien ja videoiden kanssa silläkin oli vaikeuksia. Yleisesti se saavutti korkean pakkaustehon ja suoriutuikin siitä järjellisillä nopeuksilla. Menetelmä kuitenkin vaatii yli 100 kt RAM-muistia, mikä asettaa vaatimuksia laitteelle.

FastLZ oli todella nopea, mutta pakkaustehossa se usein jäi MiniZ:lle. Erityisesti monimutkaisten EDF-tiedostojen kanssa teho jäi reilusti alhaisemmaksi. Kuitenkin joissakin käyttötarkoituksissa loistava pakkausnopeus voi kompensoida hieman heikomman pakkaustehon.

Pienimuistisille järjestelmille suunniteltu Heatshrink jäi kolmikosta heikoimmaksi erityisesti pakkausnopeuden ja usein tehonkin suhteen. Työssä käytetyssä laitteessa oli varsin paljon muistia, mutta Heatshrink ei kyennyt hyödyntämään sitä.

Työn perusteella voidaan sanoa, että mikäli yleiskäyttöisyys ja pakkausteho ovat tärkein prioriteetti, kannattaa valita MiniZ. Jos taas halutaan pakkauksen tapahtuvan mahdollisimman nopeasti, on FastLZ selkeästi paras. Heatshrinkiin valinta kannattaa suunnata, jos käytetyn laitteiston kapasiteetti ei yksinkertaisesti riitä tehokkaampien algoritmien käyttöön.

5 LÄHDELUETTELO

[1] Wayne P (1999) Compression algorithms for real programmers, 1st ed., Morgan Kaufman, Chestnut Hill, MA

[2] Gupta A, Bansal A & Khanduja V (2017), Modern Lossless Compression Techniques: Review, Comparison and Analysis, 2017 Second International Conference on Electrical, Computer and Communication Technologies (ICECCT)

[3] Lossy compression. Saatavissa, URI: https://cs.stanford.edu/people/eroberts/courses/soco/projects/data-compression/lossy/index.htm. Viitattu 2021/2/13.

[4] Overview of JPEG (2014). Saatavissa, URI: https://jpeg.org/jpeg/. Viitattu 2021/2/13.

[5] Murray JD & Van Ryper W (1996), Encyclopedia of graphics file formats, 2nd ed, O'Reilly & Associates, Sebastopol, CA

[6] What is open source? (2021). Saatavissa, URI: https://opensource.com/resources/what-open-source. Viitattu 2021/2/13.

[7] 32F412GDISCOVERY Data brief (2016). STMicroelectronics

[8] FastLZ documentation (2020). Saatavissa, URI: https://github.com/ariya/FastLZ. Viitattu 2020/12/10.

[9] Heatshrink documentation (2020). Saatavissa, URI: https://github.com/atomicobject/heatshrink. Viitattu 2020/12/10.

[10] MiniZ documentation (2020). Saatavissa, URI: https://github.com/richgel999/miniz. Viitattu 2020/12/10.

6 LIITELUETTELO

Liite 1 Pakkaustestien tulostaulukko

Algorithm	File	Size	Buffer	Level/ Window	Time	Ratio	Speed	Comp/ Time	Saved BLE4.1	Saved/MB BLE4.1	Saved BLE4.2	Saved/MB BLE4.2
FastLZ	kennedy.xls	1,03 MB	8192 B	1	3,40 s	60,32 %	302,94 kB/s	182,73 kB/s	174,11 s	169,04 s	42,97 s	41,71 s
FastLZ	kennedy.xls	1,03 MB	8192 B	2	3,45 s	59,98 %	298,55 kB/s	179,07 kB/s	173,06 s	168,02 s	42,65 s	41,41 s
FastLZ	T_simple_L.E	44,14 MB	8192 B	1	159,70 s	60,51 %	276,39 kB/s	167,25 kB/s	7471,48 s	169,27 s	1833,52 s	41,54 s
FastLZ	T_simple_L.E	44,14 MB	8192 B	2	161,43 s	60,54 %	273,43 kB/s	165,54 kB/s	7473,53 s	169,31 s	1832,78 s	41,52 s
FastLZ	kennedy.xls	1,03 MB	4096 B	1	3,89 s	59,99 %	264,78 kB/s	158,84 kB/s	172,65 s	167,62 s	42,22 s	40,99 s
FastLZ	kennedy.xls	1,03 MB	4096 B	2	3,99 s	59,81 %	258,15 kB/s	154,40 kB/s	172,02 s	167,01 s	41,98 s	40,76 s
MiniZ	kennedy.xls	1,03 MB	16384 B	1	5,39 s	79,09 %	191,09 kB/s	151,14 kB/s	227,36 s	220,74 s	55,40 s	53,79 s
FastLZ	T_simple_L.E	44,14 MB	4096 B	1	179,90 s	59,93 %	245,36 kB/s	147,04 kB/s	7378,13 s	167,15 s	1794,21 s	40,65 s
MiniZ	kennedy.xls	1,03 MB	8192 B	1	5,57 s	79,09 %	184,92 kB/s	146,25 kB/s	227,18 s	220,56 s	55,22 s	53,61 s
FastLZ	T_simple_L.E	44,14 MB	4096 B	2	181,33 s	59,94 %	243,42 kB/s	145,91 kB/s	7377,96 s	167,15 s	1793,11 s	40,62 s
MiniZ	T_simple_L.E	44,14 MB	16384 B	1	248,65 s	75,72 %	177,52 kB/s	134,42 kB/s	9300,72 s	210,71 s	2245,59 s	50,87 s
FastLZ	T_simple.EDF	8,97 MB	8192 B	2	36,18 s	54,14 %	247,93 kB/s	134,23 kB/s	1351,35 s	150,65 s	326,23 s	36,37 s
MiniZ	T_simple_L.E	44,14 MB	8192 B	1	254,41 s	75,72 %	173,50 kB/s	131,37 kB/s	9294,96 s	210,58 s	2239,83 s	50,74 s
FastLZ	T_simple.EDF	8,97 MB	8192 B	1	38,79 s	54,09 %	231,25 kB/s	125,08 kB/s	1347,46 s	150,22 s	323,29 s	36,04 s
FastLZ	T_simple.EDF	8,97 MB	4096 B	2	40,35 s	53,15 %	222,30 kB/s	118,16 kB/s	1321,81 s	147,36 s	315,44 s	35,17 s
FastLZ	T_simple.EDF	8,97 MB	4096 B	1	41,89 s	53,13 %	214,13 kB/s	113,77 kB/s	1319,76 s	147,13 s	313,76 s	34,98 s
MiniZ	T_simple.EDF	8,97 MB	16384 B	1	56,01 s	68,26 %	160,15 kB/s	109,32 kB/s	1693,40 s	188,78 s	400,92 s	44,70 s
MiniZ	T_simple.EDF	8,97 MB	8192 B	1	57,21 s	68,26 %	156,79 kB/s	107,03 kB/s	1692,20 s	188,65 s	399,72 s	44,56 s
FastLZ	fields.c	0,01 MB	4096 B	1	0,07 s	57,74 %	142,86 kB/s	82,49 kB/s	1,58 s	157,97 s	0,36 s	36,09 s
FastLZ	asyoulik.txt	0,13 MB	8192 B	2	0,64 s	40,56 %	203,13 kB/s	82,39 kB/s	14,43 s	110,96 s	3,29 s	25,35 s
FastLZ	asyoulik.txt	0,13 MB	8192 B	1	0,64 s	40,21 %	203,13 kB/s	81,68 kB/s	14,30 s	109,96 s	3,26 s	25,08 s
FastLZ	fields.c	0,01 MB	4096 B	2	0,08 s	57,79 %	125,00 kB/s	72,24 kB/s	1,57 s	157,11 s	0,35 s	35,13 s
FastLZ	fields.c	0,01 MB	8192 B	2	0,08 s	57,79 %	125,00 kB/s	72,24 kB/s	1,57 s	157,11 s	0,35 s	35,13 s
FastLZ	fields.c	0,01 MB	8192 B	1	0,08 s	57,74 %	125,00 kB/s	72,18 kB/s	1,57 s	156,97 s	0,35 s	35,09 s
MiniZ	asyoulik.txt	0,13 MB	16384 B	1	0,98 s	52,18 %	132,65 kB/s	69,22 kB/s	18,40 s	141,55 s	4,08 s	31,40 s
MiniZ	asyoulik.txt	0,13 MB	8192 B	1	1,01 s	52,18 %	128,71 kB/s	67,16 kB/s	18,37 s	141,32 s	4,05 s	31,17 s
FastLZ	asyoulik.txt	0,13 MB	4096 B	2	0,79 s	38,68 %	164,56 kB/s	63,65 kB/s	13,58 s	104,44 s	2,96 s	22,79 s
FastLZ	asyoulik.txt	0,13 MB	4096 B	1	0,80 s	38,53 %	162,50 kB/s	62,61 kB/s	13,51 s	103,93 s	2,94 s	22,60 s
MiniZ	fields.c	0,01 MB	8192 B	1	0,11 s	66,05 %	90,91 kB/s	60,05 kB/s	1,78 s	177,71 s	0,38 s	38,29 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	6	14,52 s	73,72 %	70,94 kB/s	52,29 kB/s	202,43 s	196,53 s	42,15 s	40,92 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	5	14,89 s	75,24 %	69,17 kB/s	52,05 kB/s	206,53 s	200,52 s	42,94 s	41,69 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	5	14,93 s	75,24 %	68,99 kB/s	51,91 kB/s	206,49 s	200,48 s	42,90 s	41,65 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	6	14,65 s	73,72 %	70,31 kB/s	51,83 kB/s	202,30 s	196,41 s	42,02 s	40,79 s
MiniZ	fields.c	0,01 MB	16384 B	9	0,15 s	72,04 %	66,67 kB/s	48,03 kB/s	1,91 s	190,83 s	0,39 s	38,76 s
MiniZ	fields.c	0,01 MB	8192 B	10	0,15 s	72,04 %	66,67 kB/s	48,03 kB/s	1,91 s	190,83 s	0,39 s	38,76 s
MiniZ	fields.c	0,01 MB	16384 B	10	0,15 s	72,04 %	66,67 kB/s	48,03 kB/s	1,91 s	190,83 s	0,39 s	38,76 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	7	15,56 s	72,41 %	66,20 kB/s	47,93 kB/s	197,53 s	191,78 s	40,10 s	38,93 s
FastLZ	T_complex.El	3,47 MB	8192 B	1	19,45 s	26,55 %	178,41 kB/s	47,37 kB/s	243,77 s	70,25 s	49,30 s	14,21 s
FastLZ	T_complex.El	3,47 MB	8192 B	2	19,59 s	26,66 %	177,13 kB/s	47,22 kB/s	244,72 s	70,53 s	49,45 s	14,25 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	7	15,87 s	72,41 %	64,90 kB/s	47,00 kB/s	197,22 s	191,48 s	39,79 s	38,63 s
MiniZ	T_complex.El	3,47 MB	16384 B	1	30,65 s	40,42 %	113,21 kB/s	45,76 kB/s	370,09 s	106,65 s	74,02 s	21,33 s
MiniZ	fields.c	0,01 MB	8192 B	9	0,16 s	72,04 %	62,50 kB/s	45,03 kB/s	1,90 s	189,83 s	0,38 s	37,76 s
MiniZ	T_complex.El	3,47 MB	8192 B	1	31,44 s	40,42 %	110,37 kB/s	44,61 kB/s	369,30 s	106,43 s	73,23 s	21,10 s
FastLZ	T_complex.El	3,47 MB	4096 B	1	20,93 s	25,25 %	165,79 kB/s	41,86 kB/s	229,41 s	66,11 s	44,46 s	12,81 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	8	17,61 s	71,17 %	58,49 kB/s	41,63 kB/s	191,83 s	186,25 s	37,10 s	36,01 s
FastLZ	T_complex.El	3,47 MB	4096 B	2	21,26 s	25,29 %	163,22 kB/s	41,28 kB/s	229,47 s	66,13 s	44,23 s	12,75 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	8	18,07 s	71,17 %	57,00 kB/s	40,57 kB/s	191,37 s	185,80 s	36,64 s	35,57 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	7	704,35 s	63,32 %	62,67 kB/s	39,68 kB/s	7281,21 s	164,96 s	1381,43 s	31,30 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	8	711,01 s	63,86 %	62,08 kB/s	39,64 kB/s	7342,65 s	166,35 s	1392,56 s	31,55 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	7	709,64 s	63,32 %	62,20 kB/s	39,39 kB/s	7275,92 s	164,84 s	1376,14 s	31,18 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	8	716,24 s	63,86 %	61,63 kB/s	39,36 kB/s	7337,42 s	166,23 s	1387,33 s	31,43 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	9	795,24 s	68,19 %	55,51 kB/s	37,85 kB/s	7804,49 s	176,81 s	1450,96 s	32,87 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	6	741,99 s	63,54 %	59,49 kB/s	37,80 kB/s	7271,31 s	164,73 s	1351,04 s	30,61 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	9	800,14 s	68,19 %	55,17 kB/s	37,62 kB/s	7799,59 s	176,70 s	1446,06 s	32,76 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	6	747,51 s	63,54 %	59,05 kB/s	37,52 kB/s	7265,79 s	164,61 s	1345,52 s	30,48 s
FastLZ	landing.wav	1,26 MB	8192 B	1	7,57 s	21,91 %	166,45 kB/s	36,47 kB/s	71,31 s	56,59 s	13,03 s	10,34 s
MiniZ	landing.wav	1,26 MB	16384 B	1	12,07 s	34,45 %	104,39 kB/s	35,96 kB/s	111,95 s	88,85 s	20,32 s	16,13 s
FastLZ	landing.wav	1,26 MB	8192 B	2	7,72 s	21,93 %	163,21 kB/s	35,79 kB/s	71,23 s	56,53 s	12,90 s	10,24 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	5	816,88 s	64,42 %	54,03 kB/s	34,81 kB/s	7307,40 s	165,55 s	1305,13 s	29,57 s
MiniZ	landing.wav	1,26 MB	8192 B	1	12,48 s	34,45 %	100,96 kB/s	34,78 kB/s	111,54 s	88,52 s	19,91 s	15,80 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	10	975,38 s	69,95 %	45,25 kB/s	31,66 kB/s	7846,31 s	177,76 s	1328,79 s	30,10 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	10	981,43 s	69,95 %	44,98 kB/s	31,46 kB/s	7840,26 s	177,62 s	1322,74 s	29,97 s
FastLZ	landing.wav	1,26 MB	4096 B	2	8,36 s	20,76 %	150,72 kB/s	31,29 kB/s	66,38 s	52,68 s	11,16 s	8,86 s
FastLZ	landing.wav	1,26 MB	4096 B	1	8,41 s	20,76 %	149,82 kB/s	31,10 kB/s	66,33 s	52,64 s	11,11 s	8,82 s

MiniZ	asyoulik.txt	0,13 MB	16384 B	9	2,60 s	61,01 %	50,00 kB/s	30,51 kB/s	20,06 s	154,31 s	3,32 s	25,53 s
MiniZ	asyoulik.txt	0,13 MB	16384 B	10	2,60 s	61,01 %	50,00 kB/s	30,51 kB/s	20,06 s	154,31 s	3,32 s	25,53 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	9	23,61 s	69,28 %	43,63 kB/s	30,22 kB/s	180,27 s	175,02 s	29,64 s	28,78 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	9	24,07 s	69,28 %	42,79 kB/s	29,65 kB/s	179,81 s	174,57 s	29,18 s	28,33 s
MiniZ	fields.c	0,01 MB	16384 B	1	0,23 s	66,05 %	43,48 kB/s	28,72 kB/s	1,66 s	165,71 s	0,26 s	26,29 s
MiniZ	asyoulik.txt	0,13 MB	8192 B	9	2,80 s	61,01 %	46,43 kB/s	28,33 kB/s	19,86 s	152,78 s	3,12 s	23,99 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	6	169,49 s	52,51 %	52,92 kB/s	27,79 kB/s	1176,27 s	131,13 s	182,01 s	20,29 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	6	170,13 s	52,51 %	52,72 kB/s	27,69 kB/s	1175,63 s	131,06 s	181,37 s	20,22 s
MiniZ	asyoulik.txt	0,13 MB	8192 B	10	2,88 s	61,01 %	45,14 kB/s	27,54 kB/s	19,78 s	152,16 s	3,04 s	23,38 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	7	163,44 s	49,97 %	54,88 kB/s	27,42 kB/s	1117,22 s	124,55 s	171,06 s	19,07 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	7	164,39 s	49,97 %	54,57 kB/s	27,27 kB/s	1116,27 s	124,44 s	170,11 s	18,96 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	5	182,80 s	55,39 %	49,07 kB/s	27,18 kB/s	1236,77 s	137,88 s	187,98 s	20,96 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	5	183,71 s	55,39 %	48,83 kB/s	27,05 kB/s	1235,86 s	137,78 s	187,07 s	20,86 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	9	176,21 s	50,87 %	50,91 kB/s	25,90 kB/s	1127,52 s	125,70 s	164,32 s	18,32 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	9	176,63 s	50,87 %	50,78 kB/s	25,83 kB/s	1127,10 s	125,65 s	163,90 s	18,27 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	8	168,18 s	47,99 %	53,34 kB/s	25,60 kB/s	1061,74 s	118,37 s	153,07 s	17,06 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	8	170,23 s	47,99 %	52,69 kB/s	25,29 kB/s	1059,69 s	118,14 s	151,02 s	16,84 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	11	199,71 s	55,01 %	44,92 kB/s	24,71 kB/s	1210,12 s	134,91 s	168,53 s	18,79 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	11	199,79 s	55,01 %	44,90 kB/s	24,70 kB/s	1210,04 s	134,90 s	168,45 s	18,78 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	11	1267,75 s	70,27 %	34,82 kB/s	24,47 kB/s	7594,30 s	172,05 s	1046,96 s	23,72 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	11	1273,63 s	70,27 %	34,66 kB/s	24,35 kB/s	7588,42 s	171,92 s	1041,08 s	23,59 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	10	195,72 s	50,79 %	45,83 kB/s	23,28 kB/s	1105,96 s	123,29 s	144,27 s	16,08 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	10	196,56 s	50,79 %	45,63 kB/s	23,18 kB/s	1105,12 s	123,20 s	143,43 s	15,99 s
MiniZ	landing.wav	1,26 MB	16384 B	9	22,99 s	39,46 %	54,81 kB/s	21,63 kB/s	119,07 s	94,50 s	14,11 s	11,20 s
MiniZ	landing.wav	1,26 MB	16384 B	10	23,15 s	39,46 %	54,43 kB/s	21,48 kB/s	118,91 s	94,37 s	13,95 s	11,07 s
MiniZ	landing.wav	1,26 MB	8192 B	9	23,27 s	39,46 %	54,15 kB/s	21,37 kB/s	118,79 s	94,27 s	13,83 s	10,98 s
MiniZ	landing.wav	1,26 MB	8192 B	10	23,29 s	39,46 %	54,10 kB/s	21,35 kB/s	118,77 s	94,26 s	13,81 s	10,96 s
Heatshrink	fields.c	0,01 MB	8192 B	10	0,26 s	55,05 %	38,46 kB/s	21,17 kB/s	1,31 s	131,29 s	0,15 s	15,08 s
Heatshrink	fields.c	0,01 MB	8192 B	11	0,28 s	57,90 %	35,71 kB/s	20,68 kB/s	1,37 s	137,43 s	0,15 s	15,21 s
Heatshrink	fields.c	0,01 MB	16384 B	11	0,28 s	57,90 %	35,71 kB/s	20,68 kB/s	1,37 s	137,43 s	0,15 s	15,21 s
Heatshrink	fields.c	0,01 MB	8192 B	9	0,25 s	50,68 %	40,00 kB/s	20,27 kB/s	1,20 s	119,80 s	0,13 s	12,82 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	10	35,27 s	68,11 %	29,20 kB/s	19,89 kB/s	165,17 s	160,36 s	17,08 s	16,59 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	10	35,47 s	68,11 %	29,04 kB/s	19,78 kB/s	164,97 s	160,16 s	16,88 s	16,39 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	12	255,97 s	56,26 %	35,04 kB/s	19,72 kB/s	1185,89 s	132,21 s	120,64 s	13,45 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	12	257,15 s	56,26 %	34,88 kB/s	19,62 kB/s	1184,71 s	132,08 s	119,46 s	13,32 s
Heatshrink	fields.c	0,01 MB	16384 B	9	0,26 s	50,68 %	38,46 kB/s	19,49 kB/s	1,19 s	118,80 s	0,12 s	11,82 s
Heatshrink	fields.c	0,01 MB	16384 B	8	0,24 s	46,59 %	41,67 kB/s	19,41 kB/s	1,09 s	109,11 s	0,11 s	10,77 s
Heatshrink	fields.c	0,01 MB	8192 B	8	0,26 s	46,59 %	38,46 kB/s	17,92 kB/s	1,07 s	107,11 s	0,09 s	8,77 s
Heatshrink	fields.c	0,01 MB	16384 B	12	0,33 s	58,93 %	30,30 kB/s	17,86 kB/s	1,35 s	135,37 s	0,11 s	10,98 s
Heatshrink	fields.c	0,01 MB	8192 B	12	0,34 s	58,93 %	29,41 kB/s	17,33 kB/s	1,34 s	134,37 s	0,10 s	9,98 s
Heatshrink	fields.c	0,01 MB	16384 B	10	0,32 s	55,05 %	31,25 kB/s	17,20 kB/s	1,25 s	125,29 s	0,09 s	9,08 s
Heatshrink	fields.c	0,01 MB	8192 B	7	0,25 s	41,77 %	40,00 kB/s	16,71 kB/s	0,94 s	94,34 s	0,06 s	6,17 s
Heatshrink	fields.c	0,01 MB	16384 B	7	0,25 s	41,77 %	40,00 kB/s	16,71 kB/s	0,94 s	94,34 s	0,06 s	6,17 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	13	318,49 s	57,54 %	28,16 kB/s	16,21 kB/s	1156,18 s	128,89 s	66,68 s	7,43 s
Heatshrink	fields.c	0,01 MB	8192 B	13	0,36 s	58,22 %	27,78 kB/s	16,17 kB/s	1,30 s	130,34 s	0,07 s	7,45 s
Heatshrink	fields.c	0,01 MB	16384 B	13	0,36 s	58,22 %	27,78 kB/s	16,17 kB/s	1,30 s	130,34 s	0,07 s	7,45 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	13	319,69 s	57,54 %	28,06 kB/s	16,14 kB/s	1154,98 s	128,76 s	65,48 s	7,30 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	12	1975,87 s	70,84 %	22,34 kB/s	15,83 kB/s	6958,07 s	157,64 s	357,62 s	8,10 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	12	1980,80 s	70,84 %	22,28 kB/s	15,79 kB/s	6953,14 s	157,52 s	352,69 s	7,99 s
Heatshrink	fields.c	0,01 MB	8192 B	14	0,36 s	56,19 %	27,78 kB/s	15,61 kB/s	1,25 s	124,54 s	0,06 s	5,93 s
Heatshrink	fields.c	0,01 MB	16384 B	14	0,37 s	56,19 %	27,03 kB/s	15,19 kB/s	1,24 s	123,54 s	0,05 s	4,93 s
MiniZ	T_simple.EDF	8,97 MB	16384 B	9	449,95 s	72,65 %	19,94 kB/s	14,48 kB/s	1411,97 s	157,41 s	36,37 s	4,05 s
MiniZ	T_simple.EDF	8,97 MB	8192 B	9	451,07 s	72,65 %	19,89 kB/s	14,45 kB/s	1410,85 s	157,28 s	35,25 s	3,93 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	11	56,81 s	66,79 %	18,13 kB/s	12,11 kB/s	139,74 s	135,67 s	-5,47 s	-5,31 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	11	57,07 s	66,79 %	18,05 kB/s	12,05 kB/s	139,48 s	135,42 s	-5,73 s	-5,56 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	14	440,06 s	58,26 %	20,38 kB/s	11,88 kB/s	1053,06 s	117,40 s	-50,07 s	-5,58 s
MiniZ	T_complex.EI	3,47 MB	16384 B	9	127,59 s	43,50 %	27,20 kB/s	11,83 kB/s	303,68 s	87,52 s	-14,94 s	-4,31 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	14	441,78 s	58,26 %	20,30 kB/s	11,83 kB/s	1051,34 s	117,21 s	-51,79 s	-5,77 s
MiniZ	kennedy.xls	1,03 MB	16384 B	9	69,89 s	80,11 %	14,74 kB/s	11,81 kB/s	165,86 s	161,03 s	-8,31 s	-8,07 s
MiniZ	kennedy.xls	1,03 MB	8192 B	9	69,97 s	80,11 %	14,72 kB/s	11,79 kB/s	165,78 s	160,95 s	-8,39 s	-8,15 s
MiniZ	T_complex.EI	3,47 MB	8192 B	9	128,77 s	43,50 %	26,95 kB/s	11,72 kB/s	302,50 s	87,18 s	-16,12 s	-4,65 s
MiniZ	T_simple.EDF	8,97 MB	16384 B	10	559,43 s	72,69 %	16,03 kB/s	11,66 kB/s	1303,51 s	145,32 s	-72,84 s	-8,12 s
MiniZ	T_simple.EDF	8,97 MB	8192 B	10	560,50 s	72,69 %	16,00 kB/s	11,63 kB/s	1302,44 s	145,20 s	-73,91 s	-8,24 s
MiniZ	T_complex.EI	3,47 MB	16384 B	10	138,54 s	43,54 %	25,05 kB/s	10,91 kB/s	293,13 s	84,47 s	-25,79 s	-7,43 s
MiniZ	T_complex.EI	3,47 MB	8192 B	10	139,72 s	43,54 %	24,84 kB/s	10,81 kB/s	291,95 s	84,13 s	-26,97 s	-7,77 s

Heatshrink	T_simple_L.E	44,14 MB	16384 B	13	2908,17 s	71,18 %	15,18 kB/s	10,80 kB/s	6068,64 s	137,49 s	-563,48 s	-12,77 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	13	2912,95 s	71,18 %	15,15 kB/s	10,79 kB/s	6063,86 s	137,38 s	-568,26 s	-12,87 s
Heatshrink	fields.c	0,01 MB	8192 B	6	0,29 s	30,41 %	34,48 kB/s	10,49 kB/s	0,58 s	57,89 s	-0,06 s	-6,31 s
MiniZ	T_simple_L.E	44,14 MB	16384 B	9	3531,90 s	80,53 %	12,50 kB/s	10,06 kB/s	6624,08 s	150,07 s	-879,22 s	-19,92 s
MiniZ	T_simple_L.E	44,14 MB	8192 B	9	3536,82 s	80,53 %	12,48 kB/s	10,05 kB/s	6619,16 s	149,96 s	-884,14 s	-20,03 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	10	4,06 s	29,32 %	32,02 kB/s	9,39 kB/s	6,83 s	52,54 s	-1,22 s	-9,35 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	9	3,63 s	26,21 %	35,81 kB/s	9,39 kB/s	6,11 s	46,96 s	-1,09 s	-8,36 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	10	4,08 s	29,32 %	31,86 kB/s	9,34 kB/s	6,81 s	52,39 s	-1,24 s	-9,50 s
Heatshrink	T_complex.El	3,47 MB	8192 B	6	93,61 s	24,56 %	37,07 kB/s	9,10 kB/s	149,88 s	43,19 s	-30,01 s	-8,65 s
Heatshrink	T_complex.El	3,47 MB	16384 B	6	93,61 s	24,56 %	37,07 kB/s	9,10 kB/s	149,88 s	43,19 s	-30,01 s	-8,65 s
Heatshrink	T_complex.El	3,47 MB	8192 B	5	100,32 s	25,05 %	34,59 kB/s	8,66 kB/s	148,03 s	42,66 s	-35,45 s	-10,22 s
Heatshrink	T_complex.El	3,47 MB	16384 B	5	100,37 s	25,05 %	34,57 kB/s	8,66 kB/s	147,98 s	42,65 s	-35,50 s	-10,23 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	9	3,96 s	26,21 %	32,83 kB/s	8,60 kB/s	5,78 s	44,42 s	-1,42 s	-10,90 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	12	77,78 s	63,52 %	13,24 kB/s	8,41 kB/s	109,15 s	105,97 s	-28,95 s	-28,11 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	12	77,88 s	63,52 %	13,23 kB/s	8,40 kB/s	109,05 s	105,87 s	-29,05 s	-28,21 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	11	4,83 s	30,63 %	26,92 kB/s	8,24 kB/s	6,55 s	50,36 s	-1,86 s	-14,30 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	11	4,93 s	30,63 %	26,37 kB/s	8,08 kB/s	6,45 s	49,59 s	-1,96 s	-15,06 s
Heatshrink	T_complex.El	3,47 MB	8192 B	7	93,07 s	21,24 %	37,28 kB/s	7,92 kB/s	117,51 s	33,86 s	-38,07 s	-10,97 s
Heatshrink	T_complex.El	3,47 MB	16384 B	7	93,37 s	21,24 %	37,16 kB/s	7,89 kB/s	117,21 s	33,78 s	-38,37 s	-11,06 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	8	3,54 s	20,63 %	36,72 kB/s	7,58 kB/s	4,12 s	31,71 s	-1,54 s	-11,84 s
Heatshrink	T_complex.El	3,47 MB	8192 B	8	110,35 s	23,78 %	31,45 kB/s	7,48 kB/s	125,41 s	36,14 s	-48,77 s	-14,05 s
Heatshrink	T_complex.El	3,47 MB	16384 B	8	110,47 s	23,78 %	31,41 kB/s	7,47 kB/s	125,29 s	36,11 s	-48,89 s	-14,09 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	8	3,60 s	20,63 %	36,11 kB/s	7,45 kB/s	4,06 s	31,25 s	-1,60 s	-12,30 s
MiniZ	kennedy.xls	1,03 MB	16384 B	10	113,10 s	80,10 %	9,11 kB/s	7,29 kB/s	122,62 s	119,05 s	-51,53 s	-50,03 s
MiniZ	kennedy.xls	1,03 MB	8192 B	10	113,18 s	80,10 %	9,10 kB/s	7,29 kB/s	122,54 s	118,97 s	-51,61 s	-50,11 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	7	3,38 s	18,15 %	38,46 kB/s	6,98 kB/s	3,36 s	25,86 s	-1,62 s	-12,46 s
Heatshrink	T_simple_L.E	44,14 MB	16384 B	14	4546,98 s	71,32 %	9,71 kB/s	6,92 kB/s	4447,49 s	100,76 s	-2197,68 s	-49,79 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	7	3,41 s	18,15 %	38,12 kB/s	6,92 kB/s	3,33 s	25,63 s	-1,65 s	-12,69 s
Heatshrink	T_simple_L.E	44,14 MB	8192 B	14	4551,71 s	71,32 %	9,70 kB/s	6,92 kB/s	4442,76 s	100,65 s	-2202,41 s	-49,90 s
MiniZ	T_simple_L.E	44,14 MB	16384 B	10	5862,67 s	80,85 %	7,53 kB/s	6,09 kB/s	4333,67 s	98,18 s	-3199,45 s	-72,48 s
MiniZ	T_simple_L.E	44,14 MB	8192 B	10	5867,43 s	80,85 %	7,52 kB/s	6,08 kB/s	4328,91 s	98,07 s	-3204,21 s	-72,59 s
Heatshrink	T_complex.El	3,47 MB	8192 B	9	127,67 s	22,19 %	27,18 kB/s	6,03 kB/s	92,33 s	26,61 s	-70,21 s	-20,23 s
Heatshrink	T_complex.El	3,47 MB	16384 B	9	128,37 s	22,19 %	27,03 kB/s	6,00 kB/s	91,63 s	26,41 s	-70,91 s	-20,43 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	12	7,66 s	32,66 %	16,97 kB/s	5,54 kB/s	4,47 s	34,39 s	-4,49 s	-34,55 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	12	7,67 s	32,66 %	16,95 kB/s	5,54 kB/s	4,46 s	34,31 s	-4,50 s	-34,63 s
MiniZ	lenna.tif	0,79 MB	16384 B	1	10,27 s	6,81 %	76,92 kB/s	5,24 kB/s	5,10 s	6,46 s	-6,26 s	-7,92 s
MiniZ	lenna.tif	0,79 MB	8192 B	1	10,50 s	6,81 %	75,24 kB/s	5,12 kB/s	4,87 s	6,17 s	-6,49 s	-8,21 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	6	3,49 s	13,01 %	37,25 kB/s	4,85 kB/s	1,34 s	10,33 s	-2,23 s	-17,14 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	6	3,66 s	13,01 %	35,52 kB/s	4,62 kB/s	1,17 s	9,02 s	-2,40 s	-18,44 s
Heatshrink	fields.c	0,01 MB	8192 B	5	0,34 s	15,70 %	29,41 kB/s	4,62 kB/s	0,11 s	10,86 s	-0,22 s	-22,28 s
Heatshrink	T_complex.El	3,47 MB	16384 B	10	149,14 s	19,32 %	23,27 kB/s	4,50 kB/s	42,40 s	12,22 s	-99,11 s	-28,56 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	13	143,86 s	62,60 %	7,16 kB/s	4,48 kB/s	40,36 s	39,19 s	-95,74 s	-92,95 s
Heatshrink	T_complex.El	3,47 MB	8192 B	10	149,59 s	19,32 %	23,20 kB/s	4,48 kB/s	41,95 s	12,09 s	-99,56 s	-28,69 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	13	144,01 s	62,60 %	7,15 kB/s	4,48 kB/s	40,21 s	39,04 s	-95,89 s	-93,10 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	13	10,10 s	34,76 %	12,87 kB/s	4,47 kB/s	2,81 s	21,62 s	-6,73 s	-51,75 s
Heatshrink	fields.c	0,01 MB	16384 B	5	0,36 s	15,70 %	27,78 kB/s	4,36 kB/s	0,09 s	8,86 s	-0,24 s	-24,28 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	13	10,57 s	34,76 %	12,30 kB/s	4,28 kB/s	2,34 s	18,01 s	-7,20 s	-55,37 s
MiniZ	lenna.tif	0,79 MB	16384 B	10	13,58 s	7,30 %	58,17 kB/s	4,25 kB/s	2,90 s	3,67 s	-9,28 s	-11,74 s
MiniZ	lenna.tif	0,79 MB	16384 B	9	13,59 s	7,30 %	58,13 kB/s	4,24 kB/s	2,89 s	3,65 s	-9,29 s	-11,75 s
MiniZ	lenna.tif	0,79 MB	8192 B	10	13,76 s	7,30 %	57,41 kB/s	4,19 kB/s	2,72 s	3,44 s	-9,46 s	-11,97 s
MiniZ	video.mjpeg	4,80 MB	16384 B	1	63,02 s	5,47 %	76,17 kB/s	4,17 kB/s	12,00 s	2,50 s	-43,43 s	-9,05 s
MiniZ	lenna.tif	0,79 MB	8192 B	9	13,87 s	7,30 %	56,96 kB/s	4,16 kB/s	2,61 s	3,30 s	-9,57 s	-12,11 s
MiniZ	video.mjpeg	4,80 MB	8192 B	1	64,60 s	5,47 %	74,30 kB/s	4,06 kB/s	10,42 s	2,17 s	-45,01 s	-9,38 s
Heatshrink	fields.c	0,01 MB	16384 B	6	0,75 s	30,41 %	13,33 kB/s	4,05 kB/s	0,12 s	11,89 s	-0,52 s	-52,31 s
MiniZ	video.mjpeg	4,80 MB	16384 B	10	84,33 s	7,01 %	56,92 kB/s	3,99 kB/s	11,81 s	2,46 s	-59,22 s	-12,34 s
MiniZ	video.mjpeg	4,80 MB	16384 B	9	84,45 s	7,01 %	56,84 kB/s	3,98 kB/s	11,69 s	2,43 s	-59,34 s	-12,36 s
MiniZ	video.mjpeg	4,80 MB	8192 B	9	86,06 s	7,01 %	55,78 kB/s	3,91 kB/s	10,08 s	2,10 s	-60,95 s	-12,70 s
MiniZ	video.mjpeg	4,80 MB	8192 B	10	86,06 s	7,01 %	55,78 kB/s	3,91 kB/s	10,08 s	2,10 s	-60,95 s	-12,70 s
Heatshrink	T_complex.El	3,47 MB	16384 B	11	180,82 s	17,31 %	19,19 kB/s	3,32 kB/s	-9,20 s	-2,65 s	-135,99 s	-39,19 s
Heatshrink	T_complex.El	3,47 MB	8192 B	11	182,24 s	17,31 %	19,04 kB/s	3,30 kB/s	-10,62 s	-3,06 s	-137,41 s	-39,60 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	14	14,30 s	36,06 %	9,09 kB/s	3,28 kB/s	-0,91 s	-6,97 s	-10,80 s	-83,09 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	14	14,33 s	36,06 %	9,07 kB/s	3,27 kB/s	-0,94 s	-7,20 s	-10,83 s	-83,32 s
Heatshrink	landing.wav	1,26 MB	8192 B	10	40,41 s	8,96 %	31,18 kB/s	2,79 kB/s	-8,15 s	-6,47 s	-31,98 s	-25,38 s
Heatshrink	landing.wav	1,26 MB	16384 B	10	41,07 s	8,96 %	30,68 kB/s	2,75 kB/s	-8,81 s	-7,00 s	-32,64 s	-25,91 s
Heatshrink	landing.wav	1,26 MB	8192 B	11	46,18 s	9,76 %	27,28 kB/s	2,66 kB/s	-11,04 s	-8,77 s	-37,00 s	-29,37 s

Heatshrink	landing.wav	1,26 MB	16384 B	11	47,30 s	9,76 %	26,64 kB/s	2,60 kB/s	-12,16 s	-9,65 s	-38,12 s	-30,26 s
Heatshrink	landing.wav	1,26 MB	8192 B	9	36,87 s	7,12 %	34,17 kB/s	2,43 kB/s	-11,24 s	-8,92 s	-30,18 s	-23,95 s
Heatshrink	landing.wav	1,26 MB	16384 B	9	38,16 s	7,12 %	33,02 kB/s	2,35 kB/s	-12,53 s	-9,94 s	-31,47 s	-24,97 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	14	279,81 s	61,67 %	3,68 kB/s	2,27 kB/s	-98,32 s	-95,46 s	-232,41 s	-225,64 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	14	280,10 s	61,67 %	3,68 kB/s	2,27 kB/s	-98,61 s	-95,74 s	-232,70 s	-225,92 s
Heatshrink	T_complex.El	3,47 MB	16384 B	12	317,00 s	17,43 %	10,95 kB/s	1,91 kB/s	-144,19 s	-41,55 s	-271,86 s	-78,35 s
Heatshrink	T_complex.El	3,47 MB	8192 B	12	317,20 s	17,43 %	10,94 kB/s	1,91 kB/s	-144,39 s	-41,61 s	-272,06 s	-78,40 s
Heatshrink	landing.wav	1,26 MB	8192 B	12	64,76 s	9,80 %	19,46 kB/s	1,91 kB/s	-29,48 s	-23,40 s	-55,55 s	-44,08 s
Heatshrink	landing.wav	1,26 MB	16384 B	12	65,47 s	9,80 %	19,25 kB/s	1,89 kB/s	-30,19 s	-23,96 s	-56,26 s	-44,65 s
Heatshrink	landing.wav	1,26 MB	8192 B	7	34,79 s	4,74 %	36,22 kB/s	1,72 kB/s	-17,73 s	-14,07 s	-30,33 s	-24,07 s
Heatshrink	landing.wav	1,26 MB	16384 B	7	35,89 s	4,74 %	35,11 kB/s	1,66 kB/s	-18,83 s	-14,94 s	-31,43 s	-24,95 s
Heatshrink	landing.wav	1,26 MB	8192 B	13	88,11 s	11,56 %	14,30 kB/s	1,65 kB/s	-46,49 s	-36,90 s	-77,24 s	-61,30 s
Heatshrink	landing.wav	1,26 MB	16384 B	13	88,29 s	11,56 %	14,27 kB/s	1,65 kB/s	-46,67 s	-37,04 s	-77,42 s	-61,44 s
Heatshrink	landing.wav	1,26 MB	8192 B	8	35,62 s	4,21 %	35,37 kB/s	1,49 kB/s	-20,46 s	-16,24 s	-31,66 s	-25,13 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	5	3,96 s	4,44 %	32,83 kB/s	1,46 kB/s	-2,31 s	-17,78 s	-3,53 s	-27,15 s
Heatshrink	T_complex.El	3,47 MB	16384 B	13	428,76 s	17,99 %	8,09 kB/s	1,46 kB/s	-250,40 s	-72,16 s	-382,17 s	-110,14 s
Heatshrink	landing.wav	1,26 MB	16384 B	8	36,52 s	4,21 %	34,50 kB/s	1,45 kB/s	-21,36 s	-16,96 s	-32,56 s	-25,84 s
Heatshrink	T_complex.El	3,47 MB	8192 B	13	429,79 s	17,99 %	8,07 kB/s	1,45 kB/s	-251,43 s	-72,46 s	-383,20 s	-110,43 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	5	4,17 s	4,44 %	31,18 kB/s	1,38 kB/s	-2,52 s	-19,39 s	-3,74 s	-28,76 s
Heatshrink	landing.wav	1,26 MB	8192 B	14	130,71 s	12,43 %	9,64 kB/s	1,20 kB/s	-85,96 s	-68,22 s	-119,02 s	-94,46 s
Heatshrink	landing.wav	1,26 MB	16384 B	14	131,08 s	12,43 %	9,61 kB/s	1,19 kB/s	-86,33 s	-68,52 s	-119,39 s	-94,76 s
Heatshrink	T_complex.El	3,47 MB	16384 B	14	590,94 s	18,06 %	5,87 kB/s	1,06 kB/s	-411,89 s	-118,70 s	-544,17 s	-156,82 s
Heatshrink	T_complex.El	3,47 MB	8192 B	14	591,25 s	18,06 %	5,87 kB/s	1,06 kB/s	-412,20 s	-118,79 s	-544,48 s	-156,91 s
Heatshrink	landing.wav	1,26 MB	8192 B	6	36,75 s	2,33 %	34,29 kB/s	0,80 kB/s	-28,36 s	-22,51 s	-34,56 s	-27,43 s
Heatshrink	landing.wav	1,26 MB	16384 B	6	37,10 s	2,33 %	33,96 kB/s	0,79 kB/s	-28,71 s	-22,79 s	-34,91 s	-27,71 s
MiniZ	lenna_hires.j	0,57 MB	16384 B	10	10,41 s	0,28 %	54,76 kB/s	0,15 kB/s	-9,95 s	-17,46 s	-10,29 s	-18,05 s
MiniZ	lenna_hires.j	0,57 MB	16384 B	9	10,42 s	0,28 %	54,70 kB/s	0,15 kB/s	-9,96 s	-17,48 s	-10,30 s	-18,07 s
MiniZ	lenna_hires.j	0,57 MB	8192 B	9	10,64 s	0,28 %	53,57 kB/s	0,15 kB/s	-10,18 s	-17,87 s	-10,52 s	-18,46 s
MiniZ	lenna_hires.j	0,57 MB	8192 B	10	10,67 s	0,28 %	53,42 kB/s	0,15 kB/s	-10,21 s	-17,92 s	-10,55 s	-18,51 s
MiniZ	lenna_hires.j	0,57 MB	16384 B	1	8,15 s	0,20 %	69,94 kB/s	0,14 kB/s	-7,82 s	-13,73 s	-8,06 s	-14,15 s
MiniZ	lenna_hires.j	0,57 MB	8192 B	1	8,31 s	0,20 %	68,59 kB/s	0,14 kB/s	-7,98 s	-14,01 s	-8,22 s	-14,43 s
FastLZ	video.mjpeg	4,80 MB	8192 B	2	31,29 s	0,07 %	153,40 kB/s	0,11 kB/s	-30,33 s	-6,32 s	-31,04 s	-6,47 s
MiniZ	Mona_Lisa.pr	13,16 MB	16384 B	1	189,18 s	0,05 %	69,56 kB/s	0,03 kB/s	-187,30 s	-14,23 s	-188,69 s	-14,34 s
MiniZ	Mona_Lisa.pr	13,16 MB	8192 B	1	195,38 s	0,05 %	67,36 kB/s	0,03 kB/s	-193,50 s	-14,70 s	-194,89 s	-14,81 s
MiniZ	Mona_Lisa.pr	13,16 MB	16384 B	10	244,34 s	0,05 %	53,86 kB/s	0,03 kB/s	-242,46 s	-18,42 s	-243,85 s	-18,53 s
MiniZ	Mona_Lisa.pr	13,16 MB	16384 B	9	244,52 s	0,05 %	53,82 kB/s	0,03 kB/s	-242,64 s	-18,44 s	-244,03 s	-18,54 s
MiniZ	Mona_Lisa.pr	13,16 MB	8192 B	10	249,43 s	0,05 %	52,76 kB/s	0,03 kB/s	-247,55 s	-18,81 s	-248,94 s	-18,92 s
MiniZ	Mona_Lisa.pr	13,16 MB	8192 B	9	249,48 s	0,05 %	52,75 kB/s	0,03 kB/s	-247,60 s	-18,81 s	-248,99 s	-18,92 s
Heatshrink	landing.wav	1,26 MB	16384 B	5	41,23 s	-0,64 %	30,56 kB/s	-0,20 kB/s	-43,53 s	-34,55 s	-41,83 s	-33,20 s
Heatshrink	landing.wav	1,26 MB	8192 B	5	40,40 s	-0,64 %	31,19 kB/s	-0,20 kB/s	-42,70 s	-33,89 s	-41,00 s	-32,54 s
FastLZ	lenna.tif	0,79 MB	8192 B	2	5,58 s	-0,26 %	141,58 kB/s	-0,37 kB/s	-6,17 s	-7,81 s	-5,73 s	-7,26 s
FastLZ	lenna.tif	0,79 MB	8192 B	1	5,29 s	-0,27 %	149,34 kB/s	-0,40 kB/s	-5,90 s	-7,47 s	-5,45 s	-6,90 s
FastLZ	lenna.tif	0,79 MB	4096 B	2	5,88 s	-0,48 %	134,35 kB/s	-0,64 kB/s	-6,96 s	-8,81 s	-6,16 s	-7,80 s
FastLZ	lenna.tif	0,79 MB	4096 B	1	5,90 s	-0,49 %	133,90 kB/s	-0,66 kB/s	-7,01 s	-8,87 s	-6,19 s	-7,83 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	14	300,77 s	-7,25 %	15,96 kB/s	-1,16 kB/s	-400,20 s	-83,37 s	-326,74 s	-68,07 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	14	297,61 s	-7,25 %	16,13 kB/s	-1,17 kB/s	-397,04 s	-82,72 s	-323,58 s	-67,41 s
Heatshrink	lenna.tif	0,79 MB	8192 B	14	60,07 s	-10,57 %	13,15 kB/s	-1,39 kB/s	-83,93 s	-106,24 s	-66,30 s	-83,93 s
Heatshrink	lenna.tif	0,79 MB	16384 B	14	59,00 s	-10,57 %	13,39 kB/s	-1,42 kB/s	-82,86 s	-104,88 s	-65,23 s	-82,57 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	14	36,21 s	-12,23 %	15,74 kB/s	-1,93 kB/s	-56,13 s	-98,47 s	-41,41 s	-72,65 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	14	35,78 s	-12,23 %	15,93 kB/s	-1,95 kB/s	-55,70 s	-97,71 s	-40,98 s	-71,90 s
FastLZ	video.mjpeg	4,80 MB	4096 B	2	33,49 s	-1,40 %	143,33 kB/s	-2,01 kB/s	-52,69 s	-10,98 s	-38,50 s	-8,02 s
Heatshrink	Mona_Lisa.pr	13,16 MB	8192 B	14	808,04 s	-12,43 %	16,29 kB/s	-2,02 kB/s	-1275,41 s	-96,92 s	-930,11 s	-70,68 s
Heatshrink	Mona_Lisa.pr	13,16 MB	16384 B	14	797,20 s	-12,43 %	16,51 kB/s	-2,05 kB/s	-1264,57 s	-96,09 s	-919,27 s	-69,85 s
Heatshrink	lenna.tif	0,79 MB	8192 B	13	40,28 s	-10,78 %	19,61 kB/s	-2,11 kB/s	-64,61 s	-81,79 s	-46,64 s	-59,03 s
Heatshrink	lenna.tif	0,79 MB	16384 B	13	39,55 s	-10,78 %	19,97 kB/s	-2,15 kB/s	-63,88 s	-80,86 s	-45,91 s	-58,11 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	13	212,89 s	-12,04 %	22,55 kB/s	-2,71 kB/s	-378,01 s	-78,75 s	-256,02 s	-53,34 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	13	212,07 s	-12,04 %	22,63 kB/s	-2,73 kB/s	-377,19 s	-78,58 s	-255,20 s	-53,17 s
Heatshrink	lenna.tif	0,79 MB	8192 B	5	27,21 s	-9,40 %	29,03 kB/s	-2,73 kB/s	-48,43 s	-61,30 s	-32,75 s	-41,46 s
Heatshrink	lenna.tif	0,79 MB	16384 B	5	26,50 s	-9,40 %	29,81 kB/s	-2,80 kB/s	-47,72 s	-60,40 s	-32,04 s	-40,56 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	13	24,72 s	-12,26 %	23,06 kB/s	-2,83 kB/s	-44,69 s	-78,40 s	-29,94 s	-52,52 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	13	24,60 s	-12,26 %	23,17 kB/s	-2,84 kB/s	-44,57 s	-78,19 s	-29,82 s	-52,31 s
Heatshrink	lenna.tif	0,79 MB	8192 B	12	30,52 s	-11,02 %	25,88 kB/s	-2,85 kB/s	-55,39 s	-70,12 s	-37,02 s	-46,86 s
Heatshrink	lenna.tif	0,79 MB	16384 B	12	30,11 s	-11,02 %	26,24 kB/s	-2,89 kB/s	-54,98 s	-69,60 s	-36,61 s	-46,34 s
Heatshrink	Mona_Lisa.pr	13,16 MB	8192 B	13	560,59 s	-12,49 %	23,48 kB/s	-2,93 kB/s	-1030,21 s	-78,28 s	-683,25 s	-51,92 s
Heatshrink	Mona_Lisa.pr	13,16 MB	16384 B	13	559,36 s	-12,49 %	23,53 kB/s	-2,94 kB/s	-1028,98 s	-78,19 s	-682,02 s	-51,83 s

Heatshrink	lenna.tif	0,79 MB	8192 B	6	24,90 s	-9,29 %	31,73 kB/s	-2,95 kB/s	-45,87 s	-58,06 s	-30,38 s	-38,45 s
Heatshrink	lenna.tif	0,79 MB	8192 B	11	27,31 s	-10,31 %	28,93 kB/s	-2,98 kB/s	-50,58 s	-64,03 s	-33,39 s	-42,26 s
Heatshrink	lenna.tif	0,79 MB	16384 B	6	24,44 s	-9,29 %	32,32 kB/s	-3,00 kB/s	-45,41 s	-57,48 s	-29,92 s	-37,87 s
Heatshrink	lenna.tif	0,79 MB	16384 B	11	26,70 s	-10,31 %	29,59 kB/s	-3,05 kB/s	-49,97 s	-63,25 s	-32,78 s	-41,49 s
Heatshrink	lenna.tif	0,79 MB	8192 B	7	24,13 s	-9,75 %	32,74 kB/s	-3,19 kB/s	-46,14 s	-58,40 s	-29,88 s	-37,82 s
Heatshrink	lenna.tif	0,79 MB	16384 B	7	23,65 s	-9,75 %	33,40 kB/s	-3,26 kB/s	-45,66 s	-57,79 s	-29,40 s	-37,21 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	5	162,14 s	-11,16 %	29,60 kB/s	-3,30 kB/s	-315,19 s	-65,66 s	-202,12 s	-42,11 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	5	160,90 s	-11,16 %	29,83 kB/s	-3,33 kB/s	-313,95 s	-65,41 s	-200,88 s	-41,85 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	12	173,05 s	-12,01 %	27,74 kB/s	-3,33 kB/s	-337,76 s	-70,37 s	-216,07 s	-45,01 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	12	172,35 s	-12,01 %	27,85 kB/s	-3,34 kB/s	-337,06 s	-70,22 s	-215,37 s	-44,87 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	11	160,35 s	-11,76 %	29,93 kB/s	-3,52 kB/s	-321,63 s	-67,01 s	-202,48 s	-42,18 s
FastLZ	lenna_hires.j	0,57 MB	4096 B	1	4,35 s	-2,69 %	131,03 kB/s	-3,52 kB/s	-8,73 s	-15,32 s	-5,49 s	-9,64 s
FastLZ	video.mjpeg	4,80 MB	4096 B	1	33,20 s	-2,44 %	144,58 kB/s	-3,53 kB/s	-66,66 s	-13,89 s	-41,94 s	-8,74 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	12	19,85 s	-12,29 %	28,72 kB/s	-3,53 kB/s	-39,87 s	-69,94 s	-25,08 s	-44,00 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	11	159,42 s	-11,76 %	30,11 kB/s	-3,54 kB/s	-320,70 s	-66,81 s	-201,55 s	-41,99 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	12	19,74 s	-12,29 %	28,88 kB/s	-3,55 kB/s	-39,76 s	-69,75 s	-24,97 s	-43,80 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	6	149,24 s	-11,09 %	32,16 kB/s	-3,57 kB/s	-301,33 s	-62,78 s	-188,97 s	-39,37 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	6	148,69 s	-11,09 %	32,28 kB/s	-3,58 kB/s	-300,78 s	-62,66 s	-188,42 s	-39,25 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	5	19,30 s	-12,16 %	29,53 kB/s	-3,59 kB/s	-39,10 s	-68,60 s	-24,47 s	-42,93 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	5	19,25 s	-12,16 %	29,61 kB/s	-3,60 kB/s	-39,05 s	-68,51 s	-24,42 s	-42,85 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	12	454,22 s	-12,50 %	28,97 kB/s	-3,62 kB/s	-924,22 s	-70,23 s	-576,98 s	-43,84 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	12	453,68 s	-12,50 %	29,01 kB/s	-3,63 kB/s	-923,68 s	-70,19 s	-576,44 s	-43,80 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	5	447,46 s	-12,36 %	29,41 kB/s	-3,64 kB/s	-912,20 s	-69,32 s	-568,85 s	-43,23 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	5	444,18 s	-12,36 %	29,63 kB/s	-3,66 kB/s	-908,92 s	-69,07 s	-565,57 s	-42,98 s
Heatshrink	lenna.tif	0,79 MB	8192 B	10	25,55 s	-11,90 %	30,92 kB/s	-3,68 kB/s	-52,41 s	-66,34 s	-32,57 s	-41,22 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	7	143,53 s	-11,07 %	33,44 kB/s	-3,70 kB/s	-295,35 s	-61,53 s	-183,18 s	-38,16 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	10	151,28 s	-11,69 %	31,73 kB/s	-3,71 kB/s	-311,60 s	-64,92 s	-193,15 s	-40,24 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	7	142,76 s	-11,07 %	33,62 kB/s	-3,72 kB/s	-294,58 s	-61,37 s	-182,41 s	-38,00 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	10	150,51 s	-11,69 %	31,89 kB/s	-3,73 kB/s	-310,83 s	-64,76 s	-192,38 s	-40,08 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	11	18,51 s	-12,19 %	30,79 kB/s	-3,75 kB/s	-38,36 s	-67,30 s	-23,70 s	-41,57 s
FastLZ	lenna_hires.j	0,57 MB	4096 B	2	4,08 s	-2,69 %	139,71 kB/s	-3,76 kB/s	-8,46 s	-14,84 s	-5,22 s	-9,17 s
Heatshrink	lenna.tif	0,79 MB	16384 B	10	24,93 s	-11,90 %	31,69 kB/s	-3,77 kB/s	-51,79 s	-65,56 s	-31,95 s	-40,44 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	11	18,42 s	-12,19 %	30,94 kB/s	-3,77 kB/s	-38,27 s	-67,14 s	-23,61 s	-41,41 s
Heatshrink	lenna.tif	0,79 MB	8192 B	9	24,61 s	-11,83 %	32,10 kB/s	-3,80 kB/s	-51,31 s	-64,95 s	-31,58 s	-39,98 s
Heatshrink	lenna.tif	0,79 MB	8192 B	8	24,26 s	-11,77 %	32,56 kB/s	-3,83 kB/s	-50,83 s	-64,34 s	-31,20 s	-39,49 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	11	426,32 s	-12,46 %	30,87 kB/s	-3,85 kB/s	-894,82 s	-68,00 s	-548,69 s	-41,69 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	11	425,72 s	-12,46 %	30,91 kB/s	-3,85 kB/s	-894,22 s	-67,95 s	-548,09 s	-41,65 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	9	145,08 s	-11,67 %	33,09 kB/s	-3,86 kB/s	-305,13 s	-63,57 s	-186,88 s	-38,93 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	9	144,61 s	-11,67 %	33,19 kB/s	-3,87 kB/s	-304,66 s	-63,47 s	-186,41 s	-38,84 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	6	17,74 s	-12,07 %	32,13 kB/s	-3,88 kB/s	-37,40 s	-65,61 s	-22,87 s	-40,13 s
Heatshrink	lenna.tif	0,79 MB	16384 B	9	24,02 s	-11,83 %	32,89 kB/s	-3,89 kB/s	-50,72 s	-64,21 s	-30,99 s	-39,23 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	6	17,64 s	-12,07 %	32,31 kB/s	-3,90 kB/s	-37,30 s	-65,43 s	-22,77 s	-39,95 s
FastLZ	video.mjpeg	4,80 MB	8192 B	1	29,78 s	-2,42 %	161,18 kB/s	-3,90 kB/s	-62,97 s	-13,12 s	-38,45 s	-8,01 s
FastLZ	lenna_hires.j	0,57 MB	8192 B	2	3,86 s	-2,65 %	147,67 kB/s	-3,91 kB/s	-8,18 s	-14,34 s	-4,99 s	-8,75 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	8	142,81 s	-11,65 %	33,61 kB/s	-3,92 kB/s	-302,58 s	-63,04 s	-184,54 s	-38,45 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	8	142,61 s	-11,65 %	33,66 kB/s	-3,92 kB/s	-302,38 s	-63,00 s	-184,34 s	-38,40 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	6	412,07 s	-12,32 %	31,94 kB/s	-3,93 kB/s	-875,30 s	-66,51 s	-533,06 s	-40,51 s
Heatshrink	lenna.tif	0,79 MB	16384 B	8	23,55 s	-11,77 %	33,55 kB/s	-3,95 kB/s	-50,12 s	-63,44 s	-30,49 s	-38,59 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	10	17,65 s	-12,23 %	32,29 kB/s	-3,95 kB/s	-37,57 s	-65,91 s	-22,85 s	-40,09 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	6	410,04 s	-12,32 %	32,09 kB/s	-3,95 kB/s	-873,27 s	-66,36 s	-531,03 s	-40,35 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	10	17,60 s	-12,23 %	32,39 kB/s	-3,96 kB/s	-37,52 s	-65,82 s	-22,80 s	-40,00 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	10	408,41 s	-12,46 %	32,22 kB/s	-4,01 kB/s	-876,91 s	-66,63 s	-530,78 s	-40,33 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	10	408,00 s	-12,46 %	32,25 kB/s	-4,02 kB/s	-876,50 s	-66,60 s	-530,37 s	-40,30 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	7	17,03 s	-12,01 %	33,47 kB/s	-4,02 kB/s	-36,59 s	-64,19 s	-22,14 s	-38,84 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	7	16,93 s	-12,01 %	33,67 kB/s	-4,04 kB/s	-36,49 s	-64,02 s	-22,04 s	-38,66 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	7	395,65 s	-12,27 %	33,26 kB/s	-4,08 kB/s	-857,00 s	-65,12 s	-516,15 s	-39,22 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	7	393,46 s	-12,27 %	33,45 kB/s	-4,10 kB/s	-854,81 s	-64,96 s	-513,96 s	-39,05 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	9	17,02 s	-12,27 %	33,49 kB/s	-4,11 kB/s	-37,00 s	-64,92 s	-22,24 s	-39,02 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	9	17,00 s	-12,27 %	33,53 kB/s	-4,11 kB/s	-36,98 s	-64,88 s	-22,22 s	-38,98 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	8	16,95 s	-12,29 %	33,63 kB/s	-4,13 kB/s	-36,97 s	-64,85 s	-22,18 s	-38,91 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	9	396,49 s	-12,47 %	33,19 kB/s	-4,14 kB/s	-865,36 s	-65,76 s	-518,96 s	-39,43 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	8	16,92 s	-12,29 %	33,69 kB/s	-4,14 kB/s	-36,94 s	-64,80 s	-22,15 s	-38,86 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	9	394,64 s	-12,47 %	33,35 kB/s	-4,16 kB/s	-863,51 s	-65,62 s	-517,11 s	-39,29 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	8	392,30 s	-12,48 %	33,55 kB/s	-4,19 kB/s	-861,55 s	-65,47 s	-514,86 s	-39,12 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	8	390,25 s	-12,48 %	33,72 kB/s	-4,21 kB/s	-859,50 s	-65,31 s	-512,81 s	-38,97 s

FastLZ	lenna_hires.j	0,57 MB	8192 B	1	3,57 s	-2,65 %	159,66 kB/s	-4,23 kB/s	-7,89 s	-13,83 s	-4,70 s	-8,24 s
FastLZ	Mona_Lisa.pı	13,16 MB	4096 B	2	90,74 s	-3,02 %	145,03 kB/s	-4,38 kB/s	-204,29 s	-15,52 s	-120,40 s	-9,15 s
FastLZ	Mona_Lisa.pı	13,16 MB	4096 B	1	89,00 s	-3,04 %	147,87 kB/s	-4,50 kB/s	-203,30 s	-15,45 s	-118,86 s	-9,03 s
FastLZ	Mona_Lisa.pı	13,16 MB	8192 B	2	85,22 s	-3,00 %	154,42 kB/s	-4,63 kB/s	-198,02 s	-15,05 s	-114,68 s	-8,71 s
Heatshrink	asyoulik.txt	0,13 MB	16384 B	15	3,36 s	-12,28 %	38,69 kB/s	-4,75 kB/s	-7,92 s	-60,93 s	-4,55 s	-35,01 s
Heatshrink	fields.c	0,01 MB	8192 B	15	0,26 s	-12,50 %	38,46 kB/s	-4,81 kB/s	-0,62 s	-61,71 s	-0,35 s	-35,33 s
Heatshrink	fields.c	0,01 MB	16384 B	15	0,26 s	-12,50 %	38,46 kB/s	-4,81 kB/s	-0,62 s	-61,71 s	-0,35 s	-35,33 s
FastLZ	Mona_Lisa.pı	13,16 MB	8192 B	1	80,27 s	-3,03 %	163,95 kB/s	-4,97 kB/s	-194,20 s	-14,76 s	-110,03 s	-8,36 s
Heatshrink	asyoulik.txt	0,13 MB	8192 B	15	3,13 s	-12,41 %	41,53 kB/s	-5,15 kB/s	-7,74 s	-59,53 s	-4,33 s	-33,34 s
Heatshrink	T_simple.EDF	8,97 MB	8192 B	15	184,71 s	-11,92 %	48,56 kB/s	-5,79 kB/s	-490,20 s	-54,65 s	-264,50 s	-29,49 s
Heatshrink	landing.wav	1,26 MB	16384 B	15	26,72 s	-12,42 %	47,16 kB/s	-5,86 kB/s	-71,43 s	-56,69 s	-38,40 s	-30,48 s
Heatshrink	landing.wav	1,26 MB	8192 B	15	26,58 s	-12,42 %	47,40 kB/s	-5,89 kB/s	-71,29 s	-56,58 s	-38,26 s	-30,36 s
Heatshrink	Mona_Lisa.pı	13,16 MB	8192 B	15	279,35 s	-12,50 %	47,11 kB/s	-5,89 kB/s	-749,35 s	-56,94 s	-402,11 s	-30,56 s
Heatshrink	lenna.tif	0,79 MB	8192 B	15	16,76 s	-12,50 %	47,14 kB/s	-5,89 kB/s	-44,97 s	-56,93 s	-24,13 s	-30,54 s
Heatshrink	T_complex.EI	3,47 MB	8192 B	15	71,69 s	-12,19 %	48,40 kB/s	-5,90 kB/s	-192,55 s	-55,49 s	-103,26 s	-29,76 s
Heatshrink	T_simple.EDF	8,97 MB	16384 B	15	181,06 s	-11,92 %	49,54 kB/s	-5,91 kB/s	-486,55 s	-54,24 s	-260,85 s	-29,08 s
Heatshrink	T_complex.EI	3,47 MB	16384 B	15	71,58 s	-12,19 %	48,48 kB/s	-5,91 kB/s	-192,44 s	-55,46 s	-103,15 s	-29,73 s
Heatshrink	lenna_hires.j	0,57 MB	8192 B	15	12,03 s	-12,49 %	47,38 kB/s	-5,92 kB/s	-32,37 s	-56,79 s	-17,34 s	-30,43 s
Heatshrink	video.mjpeg	4,80 MB	8192 B	15	101,26 s	-12,49 %	47,40 kB/s	-5,92 kB/s	-272,55 s	-56,78 s	-146,00 s	-30,42 s
Heatshrink	lenna_hires.j	0,57 MB	16384 B	15	11,87 s	-12,49 %	48,02 kB/s	-6,00 kB/s	-32,21 s	-56,51 s	-17,18 s	-30,15 s
Heatshrink	Mona_Lisa.pı	13,16 MB	16384 B	15	274,01 s	-12,50 %	48,03 kB/s	-6,00 kB/s	-744,01 s	-56,54 s	-396,77 s	-30,15 s
Heatshrink	video.mjpeg	4,80 MB	16384 B	15	99,60 s	-12,49 %	48,19 kB/s	-6,02 kB/s	-270,89 s	-56,44 s	-144,34 s	-30,07 s
Heatshrink	lenna.tif	0,79 MB	16384 B	15	16,34 s	-12,50 %	48,35 kB/s	-6,04 kB/s	-44,55 s	-56,40 s	-23,71 s	-30,01 s
Heatshrink	kennedy.xls	1,03 MB	8192 B	15	20,75 s	-12,33 %	49,64 kB/s	-6,12 kB/s	-57,04 s	-55,37 s	-30,23 s	-29,35 s
Heatshrink	kennedy.xls	1,03 MB	16384 B	15	20,26 s	-12,33 %	50,84 kB/s	-6,27 kB/s	-56,55 s	-54,90 s	-29,74 s	-28,87 s

Liite 2 Ohjelmakoodi

Liitteessä on kaksi lyhyttä pätkää laitteen varsin laajasta ohjelmakoodista, jotka havainnollistavat koodin toimintaa. A-kohta esittelee sen pätkän main.c -tiedostosta, joka ajoi kaikki testit peräjälkeen. B-kohdassa on valittu havainnollistavaksi esimerkiksi pakkaussilmukoiden rakenteesta fastlz_comp.c-tiedoston fastlz_comp_loop-funktio. Selvennykseksi lukijalle: alwaysPrint() ja debugPrint() -funktiot ovat sarjaporttiin tulostavia funktioita, joista debugPrint tulostaa vain kun DEBUG-lippu on käytössä. Muut esittelemättömät funktiot ovat STM32-järjestelmän omia funktioita.

A. main.c

```
alwaysPrint(id, "Starting tests.");

int iterator;
uint8_t window_size = 5;

int iter_min_heatshrink = 8;
int iter_min_miniz = 8;
int iter_min_fastlz = 8;
int iter_max_heatshrink = 8;
int iter_max_miniz = 16;
int iter_max_fastlz = 16;

static const char *fileList[] = { "asyoulik.txt", "fields.c", "kennedy.xls",
        "T_simple.EDF", "T_complex.EDF", "Mona_Lisa.png",
        "lenna.tif", "lenna_hires.jpg", "landing.wav", "video.mjpeg", 0 };
char **ptr = fileList;
char **start_ptr = ptr;
int level;

alwaysPrint(id, "--Heatshrink--");

iterator = iter_min_heatshrink;

while (iterator <= iter_max_heatshrink) {
        while (window_size < 15) {
                while (*ptr != 0) {
                        if (heatshrink_comp_loop(iterator, window_size, *ptr) == 0) {
                                break;
                        }
                        ++ptr;
                        HAL_Delay(500);
                }
                ptr = start_ptr;
                window_size++;
        }
        ptr = start_ptr;
        iterator *= 2;
        window_size = 5;
}

alwaysPrint(id, "--MiniZ--");

level = 1;
ptr = start_ptr;
iterator = iter_min_miniz;
while (iterator <= iter_max_miniz) {
        while (*ptr != 0) {
                if (miniz_comp_loop(iterator, level, *ptr) == 0) {
                        break;
                }
                ++ptr;
                HAL_Delay(500);
        }
        ptr = start_ptr;
        iterator *= 2;
}

level = 9;
ptr = start_ptr;
iterator = iter_min_miniz;

while (iterator <= iter_max_miniz) {
        while (*ptr != 0) {
                if (miniz_comp_loop(iterator, level, *ptr) == 0) {
                        break;
                }
                ++ptr;
                HAL_Delay(500);
        }
```

```
        ptr = start_ptr;
        iterator *= 2;
}

level = 10;
ptr = start_ptr;
iterator = iter_min_miniz;

while (iterator <= iter_max_miniz) {
        while (*ptr != 0) {
                if (miniz_comp_loop(iterator, level, *ptr) == 0) {
                        break;
                }
                ++ptr;
                HAL_Delay(500);
        }
        ptr = start_ptr;
        iterator *= 2;
}

alwaysPrint(id, "--FastLZ--");

level = 1;
ptr = start_ptr;
iterator = iter_min_fastlz;

while (iterator <= iter_max_fastlz) {
        while (*ptr != 0) {
                fastlz_comp_loop(iterator, level, *ptr);
                ++ptr;
                HAL_Delay(500);
        }
        ptr = start_ptr;
        iterator *= 2;
}

level = 2;
ptr = start_ptr;
iterator = iter_min_fastlz;

while (iterator <= iter_max_fastlz) {
        while (*ptr != 0) {
                fastlz_comp_loop(iterator, level, *ptr);
                ++ptr;
                HAL_Delay(500);
        }
        ptr = start_ptr;
        iterator *= 2;
}

alwaysPrint(id, "Run done.");
```

B. fastlz_comp.c

```
void fastlz_comp_loop(int multiplier, int level, char input_filename[]) {

	char id[] = "FastLZ_compress";

	FATFS fileSystem;
	FIL outputFile;
	FIL inputFile;
	unsigned char *in_buffer[512 * multiplier];
	unsigned char *out_buffer[1024 * multiplier];
	UINT br, bw;
	unsigned int iterator = 0;
	int compressed_size;
	int infile_size;
	int outfile_size;
	char output_filename[12];
	sprintf(output_filename, "F%dC%d_%s", level, multiplier, input_filename);
	char strbuff[40];

	if (f_mount(&fileSystem, SDPath, 1) == FR_OK) { // Mount SD

		HAL_GPIO_TogglePin(GPIOE, LED2_Pin);
		debugPrint(id, "FS mounted");

		if (f_open(&outputFile, output_filename, FA_WRITE | FA_CREATE_ALWAYS)
			== FR_OK) { //Create output file

			HAL_GPIO_TogglePin(GPIOE, LED3_Pin);
			debugPrint(id, "Output file created.");

			if (f_open(&inputFile, input_filename, FA_READ)
				== FR_OK) { //Open input file

				HAL_GPIO_TogglePin(GPIOE, LED4_Pin);
				debugPrint(id, "Input file opened");

				infile_size = (int) f_size(&inputFile);
				float in_size_mb = (float) infile_size / 1000000;
				sprintf(strbuff, "Compress of %s %.2fMB starting:",
					input_filename, in_size_mb);
				alwaysPrint(id, strbuff);
				sprintf(strbuff, "Buffer size %d level %d.",
					512 * multiplier, level);
				alwaysPrint(id, strbuff);

				/*THE COMPRESSION LOOP*/

				for (;;) {
					iterator++;
					sprintf(strbuff, "Loop %d", iterator);
					debugPrint(id, strbuff);
					f_read(&inputFile, in_buffer, sizeof in_buffer, &br);
					if (br == 0)
						break;
					compressed_size = fastlz_compress_level(level, in_buffer,
						br, out_buffer);
					f_write(&outputFile, out_buffer, compressed_size, &bw);
					sprintf(strbuff, "Compressed size: %d", compressed_size);
					debugPrint(id, strbuff);
					HAL_GPIO_TogglePin(GPIOE, LED2_Pin);
				}

				/* END OF LOOP*/

				outfile_size = (int) f_size(&outputFile);
				debugPrint(id, "Compress loop done.");
			}

		}
		if (f_close(&inputFile) == FR_OK) { // Close files
```

```
                HAL_GPIO_TogglePin(GPIOE, LED3_Pin);
                debugPrint(id, "Input file closed.");
        }
        if (f_close(&outputFile) == FR_OK) {
                HAL_GPIO_TogglePin(GPIOE, LED4_Pin);
                debugPrint(id, "Output file closed.");
        }
        float floated_in = (float) infile_size / 10000;
        float floated_out = (float) outfile_size / 10000;
        sprintf(strbuff, "Compress of %s done.", output_filename);
        alwaysPrint(id, strbuff);
        compressRatio(floated_in, floated_out);

    }
}
```