



TUGAS AKHIR - KI141502

IMPLEMENTASI KOMPRESI ADAPTIVE MENGUNAKAN METODE HEATSHRINK UNTUK PENGIRIMAN DATA PADA WIRELESS SENSOR NETWORK BERBASIS ZIGBEE

MUHAMAD HENDRI FEBRIANSYAH

NRP 05111440000036

Dosen Pembimbing I

Waskitho Wibisono, S.Kom., M.Eng., Ph.D.

Dosen Pembimbing II

Ir. Muchammad Husni, M.Kom.

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI

INSTITUT TEKNOLOGI SEPULUH NOPEMBER

SURABAYA

2018



TUGAS AKHIR - KI141502

**IMPLEMENTASI KOMPRESI ADAPTIVE
MENGUNAKAN METODE HEATSHRINK
UNTUK PENGIRIMAN DATA PADA WIRELESS
SENSOR NETWORK BERBASIS ZIGBEE**

**MUHAMAD HENDRI FEBRIANSYAH
NRP 05111440000036**

**Dosen Pembimbing I
Waskitho Wibisono, S.Kom.,M.Eng.,Ph.D.**

**Dosen Pembimbing II
Ir. Muchammad Husni, M.Kom.**

**DEPARTEMEN INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN KOMUNIKASI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018**

(Halaman ini sengaja dikosongkan)



UNDERGRADUATE THESES - KI141502

**IMPLEMENTATION OF ADAPTIVE
COMPRESSION USING HEATSHRINK METHOD
FOR DATA SHIPPING ON ZIGBEE WIRELESS
SENSOR NETWORK BASED**

**MUHAMAD HENDRI FEBRIANSYAH
NRP 05111440000036**

First Advisor

Waskitho Wibisono, S.Kom.,M.Eng.,Ph.D.

Second Advisor

Ir. Muchammad Husni, M.Kom.

**INFORMATICS DEPARTEMENT
FACULTY OF INFORMATION AND COMMUNICATION
TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA
2018**

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

IMPLEMENTASI KOMPRESI ADAPTIVE MENGUNAKAN METODE HEATSHRINK UNTUK PENGIRIMAN DATA PADA WIRELESS SENSOR NETWORK BERBASIS ZIGBEE

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Berbasis Jaringan
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

MUHAMAD HENDRI FEBRIANSYAH
NRP: 051114410000036

Disetujui oleh Pembimbing Tugas Akhir:

1. Waskitho Wibisono, S.Kom., M.Eng., Ph.D.
(NIP. 197410222000031001) (Pembimbing 1)
2. Ir. Muchammad Husni M.Kom.
(NIP. 196002211984031001) (Pembimbing 2)

SURABAYA
JULI, 2018

(Halaman ini sengaja dikosongkan)

IMPLEMENTASI KOMPRESI ADAPTIVE MENGUNAKAN METODE HEATSHRINK UNTUK PENGIRIMAN DATA PADA WIRELESS SENSOR NETWORK BERBASIS ZIGBEE

**Nama Mahasiswa : MUHAMAD HENDRI
FEBRIANSYAH**
NRP : 051114410000036
Departemen : Informatika FTIK-ITS
**Dosen Pembimbing 1 : Waskitho Wibisono, S.Kom.,
M.Eng., Ph.D.**
Dosen Pembimbing 2 : Ir. Muchammad Husni M.Kom.

Abstrak

Kompresi data adalah suatu teknik untuk memampatkan data sehingga memerlukan ruang penyimpanan yang lebih kecil atau dapat mempersingkat waktu pertukaran data tersebut. Pada wireless sensor network kompresi data merupakan salah satu metode efektif untuk memperpanjang life time dan dapat digunakan pada sumber daya yang terbatas. Oleh karena itu, diperlukan adanya penerapan metode kompresi adaptive dalam pengiriman data pada wireless sensor network.

Pada tugas akhir ini, ZigBee berfungsi sebagai media pengiriman data dari node End Device ke node Coordinator. Pada node End Device dilakukan kompresi menggunakan algoritma Heatshrink dan node Coordinator akan melakukan dekompresi berdasarkan konfigurasi yang telah ditentukan.

*Pada sistem dilakukan uji coba fungsionalitas dan uji coba performa dengan menggunakan beberapa skenario yang telah ditentukan. Untuk uji coba fungsionalitas, sebagian besar sistem berjalan dengan sebagaimana mestinya. Pada uji coba performa, efektifitas kompresi paling tinggi adalah **60.5%** untuk data dengan panjang **980 karakter** menggunakan konfigurasi **HS(9,8)**. Waktu yang dibutuhkan untuk proses kompresi data paling cepat adalah*

0.02948 detik untuk data dengan panjang **584 karakter** menggunakan konfigurasi **HS(8,7)**. Sedangkan yang dibutuhkan untuk proses dekompresi data paling cepat adalah **0.011856 detik** untuk data dengan panjang **584 karakter** menggunakan konfigurasi **HS(9,8)**. Akurasi pengiriman data ZigBee berturut – turut adalah **100%** (± 10 meter), **86,8%** (± 20 meter), dan **69,5%** (± 30 meter).

Kata kunci: Kompresi data, ZigBee, Heatshrink, Wireless Sensor Network.

IMPLEMENTATION OF ADAPTIVE COMPRESSION USING HEATSHRINK METHOD FOR DATA SHIPPING ON ZIGBEE WIRELESS SENSOR NETWORK BASED

Student's Name : MUHAMAD HENDRI
FEBRIANSYAH
Student's ID : 051114410000036
Department : Informatics FTIK-ITS
First Advisor : Waskitho Wibisono, S.Kom., M.Eng.,
Ph.D.
Second Advisor : Ir. Muchammad Husni M.Kom.

Abstract

Data compression is a technique for compressing data so that it requires less storage space or can shorten the time of data exchange. In wireless sensor network data compression is one effective method to extend life time and can be used on limited resources. Therefore, it is necessary to apply adaptive compression method in data transmission on wireless sensor network.

In this final task, ZigBee serves as a data transmission medium from the Router node to the Coordinator node. At node Router compression is done using Heatshrink algorithm and Coordinator node will decompress based on predetermined configuration.

*In the system tested the functionality and performance test using some predetermined scenarios. For a test of functionality, most systems work properly. In a performance test, the highest compression effectiveness was **60.5%** for data with a length of **980 characters** using the **HS configuration (9,8)**. The time required for the fastest data compression process is **0.02948 seconds** for data with a length of **584 characters** using the **HS configuration (8,7)**. While required for the fastest data decompression process is **0.011856 seconds** for data with a length of **584 characters** using **HS configuration (9,8)**. The accuracy of ZigBee data transmission*

is **100%** (± 10 meters), **86.8%** (± 20 meters), and **69.5%** (± 30 meters) respectively.

Keywords : *Data compression, ZigBee, Heatshrink, Wireless Sensor Network.*

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Alhamdulillahirabbil'alam, segala puji bagi Allah Swt yang telah melimpahkan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul:

“IMPLEMENTASI KOMPRESI ADAPTIVE MENGUNAKAN METODE HEATSHRINK UNTUK PENGIRIMAN DATA PADA WIRELESS SENSOR NETWORK BERBASIS ZIGBEE”

yang merupakan salah satu syarat dalam menempuh ujian sidang guna memperoleh gelar Sarjana Komputer. Selesaiannya Tugas Akhir ini tidak terlepas dari bantuan dan dukungan beberapa pihak, sehingga pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Allah –‘Azz Wa Jalla– dan Nabi Muhammad –Shallallahu ‘alaihi wa sallam–.
2. Bapak Jamaludin dan Ibu Suryati selaku orang tua penulis serta keluarga penulis yang selalu memberikan dukungan doa, moral, dan material yang tak terhingga kepada penulis sehingga penulis dapat menyelesaikan Tugas Akhir ini.
3. Bapak Waskitho Wibisono, S.Kom., M.Eng., Ph.D. selaku pembimbing I yang telah membimbing dan memberikan motivasi, nasehat dan bimbingan dalam menyelesaikan Tugas Akhir ini sekaligus dosen wali penulis yang telah memberikan arahan, masukan dan motivasi kepada penulis.
4. Bapak Ir. Muchammad Husni, M.Kom. selaku II yang telah membimbing dan memberikan motivasi, nasehat dan bimbingan dalam menyelesaikan Tugas Akhir ini.
5. Bapak Darlis Herumurti, S.Kom., M.Kom. selaku kepala jurusan Teknik Informatika ITS.
6. Bapak Dr. Eng. Radityo Anggoro, S.Kom., M.Sc. selaku koordinator Tugas Akhir.

7. Bapak Ir. F.X. Arunanto, M.Sc. sebagai dosen penguji Tugas Akhir penulis.
 8. Bapak Bagus Jati Santoso, S.Kom., Ph.D.. sebagai dosen penguji Tugas Akhir penulis.
 9. Seluruh dosen dan karyawan Teknik Informatika ITS yang telah memberikan ilmu dan pengalaman kepada penulis selama menjalani masa studi di ITS.
 10. Ibu Eva Mursidah dan Ibu Sri Budiati yang selalu mempermudah penulis dalam peminjaman buku di RBTC.
 11. Para alumni Teknik Informatika ITS, terkhususnya mas Ade Ilham Fajri S.Kom yang terus memberikan bimbingan jarak jauh, mas Hanif Sudira S.Kom, mas Ghulam S.Kom, Mas Regin Iqbal S.Kom dan mas Randy Bastian S.Kom.
 12. Teman-teman Keluarga Muslim Informatika, yang sudah banyak meluruskan penulis.
 13. Teman-teman seperjuangan RMK NCC/KBJ, yang telah menemani dan menyemangati penulis.
 14. Teman-teman administrator NCC/KBJ, yang telah menemani dan menyemangati penulis selama penulis menjadi administrator, menjadi rumah kedua penulis selama penulis berkuliah.
 15. Teman-teman angkatan 2014, yang sudah mendukung saya selama perkuliahan.
 16. Sahabat penulis yang tidak dapat disebutkan satu per satu yang selalu membantu, menghibur, menjadi tempat bertukar ilmu dan berjuang bersama-sama penulis.
- Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan sehingga dengan kerendahan hati penulis mengharapakan kritik dan saran dari pembaca untuk perbaikan ke depan.

Surabaya, Juli 2018

DAFTAR ISI

LEMBAR PENGESAHAN.....	v
Abstrak.....	vii
Abstract	ix
DAFTAR ISI.....	xiii
DAFTAR GAMBAR.....	xvii
DAFTAR TABEL.....	Error! Bookmark not defined.
DAFTAR KODE SUMBER	xxi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Batasan Permasalahan	3
1.4 Tujuan.....	3
1.5 Manfaat	4
1.6 Metodologi.....	4
1.6.1 Penyusunan Proposal	4
1.6.2 Studi Literatur	5
1.6.3 Analisis dan Desain Perangkat Lunak.....	5
1.6.4 Implementasi Perangkat Lunak	5
1.6.5 Pengujian dan Evaluasi	5
1.6.6 Penyusunan Buku	6
1.7 Sistematika Penulisan Laporan	6
BAB II TINJAUAN PUSTAKA	9
2.1 Wireless Sensor Network	9
2.2 Arduino UNO	10
2.3 Arduino Mega 2560.....	12
2.4 Arduino Integrated Development Environment	13
2.5 XBee Shield.....	14
2.6 XBee Modul S2	17
2.7 Protokol ZigBee	17
2.8 Algoritma Heatshrink	19
2.9 Bahasa Pemrograman C	22
2.10 MicroSD Card Adapter.....	22

2.11 DIGI XCTU	23
BAB III PERANCANGAN PERANGKAT LUNAK.....	25
3.1 Deskripsi Umum Sistem	25
3.2 Arsitektur Umum Sistem	26
3.3 Perancangan Komunikasi Sistem.....	28
3.4 Perancangan Kompresi dan Dekompresi Data.....	32
3.4.1 Cara Kerja Algoritma Heatshrink	33
3.4.2 Konfigurasi Algoritma Heatshrink	34
3.5 Perancangan Pengiriman Data	36
3.6 Perancangan Dekompresi Data	37
3.7 Perancangan Perangkat Keras.....	38
3.8 Perancangan Perangkat ZigBee Coordinator	39
3.9 Perancangan Perangkat ZigBee Router	39
3.10 Perancangan Perangkat ZigBee End Device	40
BAB IV IMPLEMENTASI.....	43
4.1 Lingkungan Implementasi	43
4.1.1 Lingkungan Implementasi Perangkat Keras.....	43
4.1.2 Lingkungan Implementasi Perangkat Lunak	45
4.2 Implementasi Perangkat Keras.....	45
4.2.1 Perangkat Node ZigBee Coordinator	46
4.2.2 Perangkat Node ZigBee Router	47
4.2.3 Perangkat Node ZigBee End Device	48
4.3 Implementasi Inisialisasi Data Pada SD Card.....	50
4.4 Implementasi Membaca Data dari SD Card	53
4.5 Implementasi Setting Konfigurasi Encoder / Decoder ...	53
4.6 Implementasi Kompresi Data	54
4.7 Implementasi Mekanisme Pengiriman Data	55
4.8 Implementasi Dekompresi Data	56
BAB V HASIL UJI COBA DAN EVALUASI	59
5.1 Lingkungan Uji Coba	59
5.2 Data Pengujian	62
5.3 Skenario Uji Coba Fungsionalitas	62
5.3.1 Skenario Uji Coba Membaca Data dari SD Card	62

5.3.2	Skenario Uji Coba Kompresi Data	63
5.3.3	Skenario Uji Coba Komunikasi Pada Topologi Cluster Tree	65
5.3.4	Skenario Uji Coba Dekompresi Data	66
5.3.5	Skenario Uji Coba Kompresi Adaptive	67
5.4	Hasil Uji Coba Fungsionalitas.....	68
5.4.1	Hasil Uji Coba (UJ-F01) – Membaca Data dari SD Card	68
5.4.2	Hasil Uji Coba (UJ-F02) – Kompresi Data	71
5.4.3	Hasil Uji Coba (UJ-F03) – Komunikasi Pada Topologi Cluster Tree.....	71
5.4.4	Hasil Uji Coba (UJ-F04) – Dekompresi Data	72
5.4.5	Hasil Uji Coba (UJ-F05) – Kompresi Adaptive	73
5.5	Skenario Uji Coba Performa	74
5.5.1	Skenario Uji Coba Efektifitas Kompresi.....	74
5.5.2	Skenario Uji Coba Akurasi Pengiriman Data ZigBee.....	76
5.5.3	Skenario Uji Waktu Kompresi	77
5.5.4	Skenario Uji Waktu Dekompresi	79
5.6	Hasil Uji Coba Performa	80
5.6.1	Hasil Uji Coba (UJ-P01) – Efektifitas Kompresi	81
5.6.2	Hasil Uji Coba (UJ-P02) – Akurasi Pengiriman Data ZigBee.....	83
5.6.3	Hasil Uji Coba (UJ-P03) – Waktu Kompresi Data	84
5.6.4	Hasil Uji Coba (UJ-P04) – Waktu Dekompresi Data	86
5.7	Evaluasi Hasil Uji Coba.....	88
BAB VI KESIMPULAN DAN SARAN.....		91
6.1	Kesimpulan	91
6.2	Saran.....	92
DAFTAR PUSTAKA		93
LAMPIRAN.....		97
BIODATA PENULIS.....		112

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

Gambar 2.1 Ilustrasi Wireless Sensor Network [8]	10
Gambar 2.2 Arduino UNO	11
Gambar 2.3 Arduino Mega 2560.....	12
Gambar 2.4 Antarmuka Arduino IDE	14
Gambar 2.5 XBee Shield pabrikan Funduino.....	15
Gambar 2.6 XBee Shield pabrikan ITead Studio	16
Gambar 2.7 Konfigurasi Jumper mode USB.....	16
Gambar 2.8 Konfigurasi Jumper mode XBee	16
Gambar 2.9 XBee Modul S2	17
Gambar 2.10 Contoh text asli [18]	21
Gambar 2.11 Hasil teks yang ditelah dikompresi [18]	21
Gambar 2.12 MicroSD Card Adapter.....	23
Gambar 2.13 Antarmuka XCTU	24
Gambar 3.1 Deskripsi Umum Sistem.....	25
Gambar 3.2 Arsitektur detail sistem.....	27
Gambar 3.3 Konfigurasi pada node ZigBee Coordinator.....	29
Gambar 3.4 Konfigurasi pada node ZigBee Router	30
Gambar 3.5 Konfigurasi pada node ZigBee End Device	31
Gambar 3.6 Diagram alir cara kerja algoritma heatshrink	33
Gambar 3.7 Diagram alir pengiriman data	37
Gambar 3.8 Diagram alir dekompresi data.....	38
Gambar 3.9 Node ZigBee Coordinator	39
Gambar 3.10 Node ZigBee Router.....	40
Gambar 3.11 Node ZigBee End Device disertai MicroSD	41
Gambar 4.1 Perancangan Node ZigBee Coordinator	47
Gambar 4.2 Implementasi Node ZigBee Coordinator.....	47
Gambar 4.3 Perancangan Node ZigBee Router	48
Gambar 4.4 Implementasi Node ZigBee Router	48
Gambar 4.5 Perancangan Node ZigBee End Device.....	49
Gambar 4.6 Implementasi Node ZigBee End Device	49
Gambar 4.7 Data dengan panjang 584 karakter per baris.....	51
Gambar 4.8 Data dengan panjang 980 karakter per baris.....	51
Gambar 4.9 Data dengan panjang 1280 karakter per baris.....	52

Gambar 4.10 Data dengan panjang 1345 karakter per baris.....	52
Gambar 4.11 Data dengan panjang 1870 karakter per baris.....	53
Gambar 5.1 Lokasi Pertama	61
Gambar 5.2 Lokasi Kedua.....	61
Gambar 5.3 Hasil Uji Coba UJ-F01 pada panjang karakter 584.69	
Gambar 5.4 Hasil Uji Coba UJ-F01 pada panjang karakter 980.69	
Gambar 5.5 Hasil Uji Coba UJ-F01 pada panjang karakter 1280	
.....	70
Gambar 5.6 Hasil Uji Coba UJ-F01 pada panjang karakter 1345	
.....	70
Gambar 5.7 Hasil Uji Coba UJ-F01 pada panjang karakter 1870	
.....	70
Gambar 5.8 Hasil Uji Coba UJ-F05 Kompresi Adaptive.....	73
Gambar 5.9 Hasil Uji Coba UJ-F05 Kompresi Adaptive.....	74

DAFTAR TABEL

Tabel 2.1 Arduino UNO Data Sheet.....	11
Tabel 2.2 Arduino Mega 2560 Data Sheet	12
Tabel 4.1 Lingkungan Implementasi Perangkat Keras.....	43
Tabel 4.2 Lingkungan Implementasi Perangkat Lunak.....	45
Tabel 5.1 Spesifikasi Lingkungan Uji Coba.....	59
Tabel 5.2 Skenario Uji Coba Membaca Data dari SD Card.....	62
Tabel 5.3 Skenario Uji Coba Kompresi Data.....	63
Tabel 5.4 Skenario Uji Coba Komunikasi Pada Topologi Cluster Tree.....	65
Tabel 5.5 Skenario Uji Coba Dekompresi Data	66
Tabel 5.6 Skenario Uji Coba Kompresi Adaptive.....	67
Tabel 5.7 Hasil Uji Coba UJ – F02 Kompresi Data	71
Tabel 5.8 Hasil Uji Coba UJ – F03 Komunikasi Pada Topologi Cluster Tree	72
Tabel 5.9 Hasil Uji Coba UJ – F04 Dekompresi Data	72
Tabel 5.10 Uji Coba Efektifitas Kompresi	74
Tabel 5.11 Uji Coba Akurasi Pengiriman Data ZigBee	76
Tabel 5.12 Uji Coba Waktu Kompresi	77
Tabel 5.13 Uji Coba Waktu Dekompresi	79
Tabel 5.14 Hasil Uji Coba UJ – P01 Efektifitas Kompresi	81
Tabel 5.15 Hasil Uji Coba UJ – P02 Pengiriman Data ZigBee...	83
Tabel 5.16 Hasil Uji Coba UJ – P03 Waktu Kompresi Data	85
Tabel 5.17 Hasil Uji Coba UJ – P04 Waktu Dekompresi Data...	87
Tabel 5.18 Evaluasi Hasil Uji Coba Fungsionalitas	88
Tabel 5.19 Evaluasi Hasil Uji Coba Performa	89
Tabel 8.1 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 584 karakter.....	97
Tabel 8.2 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 980 karakter.....	98
Tabel 8.3 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 1280 karakter.....	98
Tabel 8.4 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 1345 karakter.....	99

Tabel 8.5 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 1870 karakter.....	100
Tabel 8.6 Rincian Hasil Uji Coba UJ-P02 pada Jarak ± 10 Meter	101
Tabel 8.7 Rincian Hasil Uji Coba UJ-P02 pada Jarak ± 20 Meter	101
Tabel 8.8 Rincian Hasil Uji Coba UJ-P02 pada Jarak ± 30 Meter	101
Tabel 8.9 Rincian Hasil Uji Coba UJ-P03 pada Panjang Data 584 Karakter.....	102
Tabel 8.10 Rincian Hasil Uji Coba UJ-P03 pada Panjang Data 980 Karakter.....	103
Tabel 8.11 Rincian Hasil Uji Coba UJ-P03 pada Panjang Data 980 Karakter.....	104
Tabel 8.12 Rincian Hasil Uji Coba UJ-P03 pada Panjang Data 1345 Karakter.....	105
Tabel 8.13 Rincian Hasil Uji Coba UJ-P03 pada Panjang Data 1870 Karakter.....	106
Tabel 8.14 Rincian Hasil Uji Coba UJ-P04 pada Panjang Data 584 Karakter.....	107
Tabel 8.15 Rincian Hasil Uji Coba UJ-P04 pada Panjang Data 980 Karakter.....	108
Tabel 8.16 Rincian Hasil Uji Coba UJ-P04 pada Panjang Data 1280 Karakter.....	109
Tabel 8.17 Rincian Hasil Uji Coba UJ-P04 pada Panjang Data 1345 Karakter.....	110
Tabel 8.18 Rincian Hasil Uji Coba UJ-P04 pada Panjang Data 1870 Karakter.....	111

DAFTAR KODE SUMBER

Kode Sumber 4.1 <i>Pseudocode</i> membaca data dari sd card	53
Kode Sumber 4.2 <i>Pseudocode</i> setting konfigurasi encoder / decoder	54
Kode Sumber 4.3 <i>Pseudocode</i> kompresi data	55
Kode Sumber 4.4 <i>Pseudocode</i> mekanisme pengiriman data.....	56
Kode Sumber 4.5 <i>Pseudocode</i> pembentukan ulang data.....	56
Kode Sumber 4.6 <i>Pseudocode</i> dekompresi data	Error! Bookmark not defined.

(Halaman ini sengaja dikosongkan)

BAB I

PENDAHULUAN

1.1 Latar Belakang

Wireless Sensor Network merupakan salah satu teknologi yang paling menjanjikan untuk kebutuhan di masa depan. Hal ini dikarenakan harganya yang murah, mudah dibangun, terdapat sensor cerdas, ukurannya kecil dan multi fungsionalitas sesuai kebutuhan. Berdasarkan research dari IDTechEx, diprediksi bahwa pasar WSN akan mengalami pertumbuhan menjadi \$ 1,8 miliar pada tahun 2024. Data ini mengacu pada WSN yang didefinisikan sebagai jaringan mesh nirkabel, yaitu self-healing dan self-organising [1].

Terdapat banyak permasalahan dan tantangan yang harus dihadapi dalam WSN untuk meningkatkan efisiensi, kelayakan dan manfaat. Tantang tersebut dapat dikategorikan kedalam empat kategori, yaitu efisiensi daya, pengumpulan data, jaringan dan strategi penyebaran [2]. WSN pada dasarnya adalah system yang berbasis event, node sensor akan mendeteksi keadaan di lingkungan sekitarnya untuk dikirim ke sink. Namun, karena node sensor sering mendeteksi fenomena umum, maka kemungkinan ada beberapa redundansi dalam data yang sumbernya beragam berkomunikasi dengan sink tertentu. Untuk itu pemfilteran dan pemrosesan dalam jaringan diperlukan agar dapat menghemat penggunaan energi yang terbatas.

Agregasi data merupakan proses pengumpulan data dari berbagai node untuk menghilangkan redundansi, meminimalkan jumlah transmisi dan memberikan informasi yang ringkas ke simpul utama [3]. Tujuan agregasi data adalah untuk memperpanjang umur jaringan dengan mengurangi transmisi waktu atau ukuran data yang dipancarkan node menggunakan algoritma cerdas. Secara umum, agregasi data dapat dikategorikan menjadi dua subsistem yang berbeda, yaitu protocol jaringan dan penggabungan data. Gagasan protocol agregasi data pada jaringan

adalah bekerjasama antar node spasial dan temporal berkorelasi [4] dalam menyebarkan data yang dikumpulkan. Pendekatan ini telah banyak diajukan dalam penelitian seperti LEECH, TEEN, HEED dan PEGASIS. Disisi lain penggabungan data bertujuan untuk mengurangi ukuran data yang ditransmisikan oleh sink atau node itu sendiri. Ukuran data dapat dikurangi menggunakan teknik operasi matematika [5] (median, average, moving average), kompresi, estimasi data dan pemodelan.

Kompresi data merupakan salah satu metode efektif untuk mengurangi penggunaan daya yang terbatas pada WSN. Diasumsikan bahwa beberapa kehilangan presisi atau kedetailan data pada saat kompresi dapat ditolerir jika hal tersebut dapat mengurangi komunikasi. Namun, perlu diperhatikan juga bahwa kualitas kedetailan data harus dipenuhi pada saat kondisi tertentu agar informasi yang didapatkan semakin banyak dan detail.

Disisi lain, menggunakan pendekatan agregasi data akan mempengaruhi perilaku komunikasi pada jaringan. Transmisi data yang harusnya kontinu diubah ke transmisi buffer berdasarkan kemampuan pemrosesan data lokal. Dengan menggabungkan agregasi data dengan komunikasi buffer maka ukuran paket data akan meningkat. Dengan demikian, slot waktu yang dibutuhkan untuk mentransmisikan paket data juga ikut meningkat. Jika terjadi suatu kondisi dimana beberapa node ingin mentransmisikan datanya secara bersamaan, maka node forwarding masing-masing lingkungan multihop akan menjadi hambatan, terutama pada jaringan yang berdaya rendah dan bandwidth yang terbatas. Maka buffer overflows dan rasio paket loss yang meningkat akan menjadi masalah besar [6].

Pada kondisi nyata, jaringan komunikasi pada wireless sensor network memiliki buffer yang sangat kecil. Di nrf24l01+ payload data yang disediakan hanya 32 byte dan di IEEE 802.15.4 payload data sebesar 133 bytes. Belum lagi jika kita menggunakan modul tambahan. Pada Xbee beban buffer yang dialokasikan untuk 802.15.4 dan ZigBee masing-masing adalah 100 dan 72 byte [7].

Oleh karena itu, dibutuhkan suatu manajemen penggunaan paket yang baik untuk menghindari terjadinya buffer overflows. Dalam tugas akhir ini metode yang diusulkan adalah implementasi kompresi adaptive menggunakan metode Heatshrink untuk pengiriman data pada wireless sensor network berbasis Zigbee. Algoritma kompresi data yang digunakan adalah Heatshrink. Algoritma ini berbasis pada algoritma Lempel-Ziv-Storer-Szymanski (LZSS) yang merupakan lossless kompresi data yang cocok untuk kompresi data pada embedded system.

1.2 Rumusan Masalah

Tugas akhir ini mengangkat beberapa rumusan masalah sebagai berikut:

1. Bagaimana cara menentukan level kompresi data?
2. Bagaimana metode yang digunakan dalam pengiriman data pada protocol ZigBee?
3. Bagaimana tingkat efisiensi dari sistem yang dibangun dapat diukur?

1.3 Batasan Permasalahan

Permasalahan yang dibahas pada tugas akhir ini memiliki batasan sebagai berikut:

1. Menggunakan mikrokontroler Arduino Mega.
2. Komunikasi nirkabel menggunakan protocol ZigBee
3. Menggunakan algoritma Heatshrink untuk melakukan kompresi dan dekompresi data.
4. Algoritma Heatshrink dibuat dalam bahasa pemrograman C.

1.4 Tujuan

Tujuan dari tugas akhir ini adalah sebagai berikut:

1. Membangun sebuah sistem kompresi data pada platform *wireless sensor network*.

2. Melakukan implementasi algoritma Heatsrink untuk kompresi adaptive pada pada platform *wireless sensor network* sehingga dapat melakukan penghematan daya.

1.5 Manfaat

Dengan dibuatnya tugas akhir ini diharapkan dapat memberikan manfaat dalam penggunaan algoritma Heatsrink untuk kompresi adaptive pada pengiriman data di *platform wireless sensor network* sehingga dapat mengemat penggunaan energi.

Sedangkan bagi penulis, tugas akhir ini bermanfaat sebagai sarana untuk mengimplementasikan ilmu yang telah dipelajari selama kuliah agar berguna bagi masyarakat.

1.6 Metodologi

Pembuatan tugas akhir ini dilakukan dengan menggunakan metodologi sebagai berikut:

1.6.1 Penyusunan Proposal

Proposal tugas akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat dari hasil pembuatan tugas akhir. Selain itu, dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan tahapan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku Tugas Akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan Tugas Akhir.

1.6.2 Studi Literatur

Tahap studi literatur merupakan tahap pembelajaran dan pengumpulan informasi yang digunakan untuk mengimplementasikan Tugas Akhir. Tahap ini diawali dengan pengumpulan literatur, diskusi, eksplorasi teknologi, dan pustaka, serta pemahaman dasar teori yang digunakan pada topik tugas akhir. Literatur-literatur yang dimaksud disebutkan yaitu mengenai Arduino, bahasa pemrograman C, algoritma Heatsrink, dan protokol ZigBee.

1.6.3 Analisis dan Desain Perangkat Lunak

Pada tahap ini akan dilakukan analisa, perancangan, dan pendefinisian kebutuhan system untuk mengetahui permasalahan yang akan dihadapi pada tahap implementasi. Kemudian akan dijabarkan kebutuhan-kebutuhan tersebut ke dalam perancangan fitur sistem. Berikut langkah yang akan dilakukan perancangan proses perangkat lunak:

1. Perancangan rangkaian node yang akan dibuat
2. Uji coba komunikasi menggunakan protocol ZigBee pada rangkaian node
3. Implementasi kompresi dan dekompresi pada node
4. Implementasi kompresi adaptive pada node

1.6.4 Implementasi Perangkat Lunak

Implementasi merupakan tahap untuk membangun metode-metode yang sudah diajukan pada proposal Tugas Akhir. Sistem dibangun menggunakan Arduino IDE (Integrated Development Environment), dengan bahasa pemrograman C dan algoritma Heatsrink.

1.6.5 Pengujian dan Evaluasi

Tahap pengujian dan evaluasi berisi pengujian aplikasi dan evaluasi berdasarkan hasil pengujian. Pada tahap ini dilakukan pengujian dari fungsionalitas dan performa system WSN yang mana nantinya akan dijalankan scenario yang sudah ditentukan. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian yang telah dilakukan. Pengujian fungsionalitas meliputi uji coba setiap bagian perangkat keras yang dirangkai pada Arduino dan juga uji coba keseluruhan sistem. Pengujian performa meliputi tingkat akurasi hasil kompresi data dan efisiensi data yang dapat di hemat.

1.6.6 Penyusunan Buku

Pada tahap ini disusun buku sebagai dokumentasi dari pelaksanaan tugas akhir yang mencakup seluruh konsep, teori, implementasi, serta hasil yang telah dikerjakan.

1.7 Sistematika Penulisan Laporan

Sistematika penulisan laporan tugas akhir adalah sebagai berikut:

1. Bab I. Pendahuluan

Bab ini berisikan penjelasan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat, metodologi, dan sistematika penulisan dari pembuatan tugas akhir.

2. Bab II. Tinjauan Pustaka

Bab ini memaparkan hasil studi literatur yang digunakan sebagai dasar untuk menyelesaikan tugas akhir ini, terdiri atas deskripsi mengenai wireless sensor network, mikrokontroler, protokol ZigBee, bahasa pemrograman C, dan algoritma Heatsrink.

3. Bab III. Perancangan Perangkat Lunak

Bab ini berisi pembahasan mengenai perancangan komunikasi pada sistem, perancangan kompresi data, pengiriman data, perancangan perangkat keras dan lunak..

4. Bab IV. Implementasi

Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan. Implementasi yang akan diterapkan berupa *Pseudocode*.

5. Bab V. Hasil Uji Coba dan Evaluasi

Bab ini menjelaskan pengujian pada sistem yang dibuat. Pengujian akan dilakukan berupa uji coba fungsionalitas dan uji coba performa.

6. Bab VI. Kesimpulan dan Saran

Bab ini merupakan bab yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan, masalah-masalah yang dialami pada proses pengerjaan Tugas Akhir, dan saran untuk pengembangan solusi ke depannya.

7. Daftar Pustaka

Bab ini berisi daftar pustaka yang dijadikan literatur dalam tugas akhir.

8. Lampiran

Lampiran yang ada berisikan kelengkapan – kelengkapan yang diperlukan dalam menyusun buku Tugas Akhir.

(Halaman ini sengaja dikosongkan)

BAB II

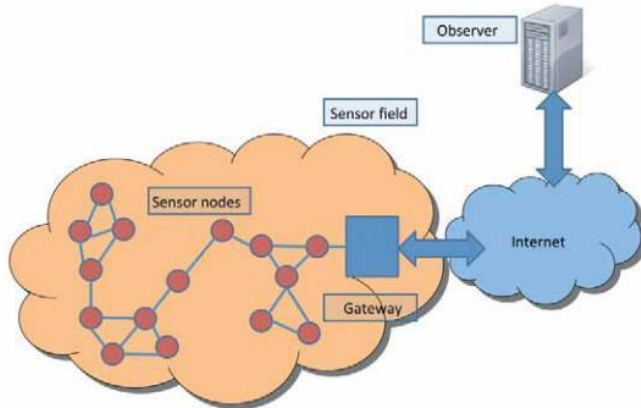
TINJAUAN PUSTAKA

Bab ini membahas teori-teori dasar yang berkaitan dengan pokok bahasan tugas akhir. Bab ini juga menjelaskan modul dan alat yang nantinya akan digunakan pada tahap implementasi program. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap alat yang digunakan dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1 Wireless Sensor Network

Wireless Network Sensor (WSN) adalah jaringan yang dibentuk oleh sekumpulan banyak node sensor dimana masing-masing node dilengkapi dengan sensor untuk mendeteksi fenomena fisik seperti cahaya, panas, tekanan, getaran dan lain-lain. WSN dianggap sebagai metode pengumpulan informasi revolusioner untuk membangun sistem informasi dan komunikasi dalam meningkatkan kehandalan dan efisiensi sistem infrastruktur. Jika dibandingkan dengan solusi kabel, WSN lebih mudah dipasang dan memiliki fleksibilitas perangkat yang lebih baik [8].

WSN pada umumnya dapat digambarkan sebagai jaringan simpul yang secara kooperatif merasakan dan mengendalikan lingkungan, memungkinkan interaksi antar orang atau computer dan lingkungan sekitar [9]. WSN saat ini biasanya mencakup node sensor, node aktuator, gateway dan klien. Sejumlah besar node sensor diletakkan secara acak didalam atau di dekat area pemantauan (sensor field), kemudian membentuk jaringan melalui self-organization. Node sensor memonitor data yang terkumpul untuk dikirim bersama ke node sensor lainnya dengan melompat. Selama proses transmisi, data yang dipantau dapat ditangani oleh beberapa node untuk sampai ke node gateway setelah multihop routing dan akhirnya mencapai node management melalui internet atau satelit. Gambar 2.1 merupakan contoh ilustrasi atau gambaran mengenai WSN.



Gambar 2.1 Ilustrasi Wireless Sensor Network [8]

2.2 Arduino UNO

Arduino adalah pengendali mikro single-board yang bersifat open-source, diturunkan dari Wiring platform, dirancang untuk memudahkan penggunaan elektronik dalam berbagai bidang. Hardwarenya memiliki prosesor Atmel AVR dan softwarenya memiliki bahasa pemrograman sendiri. Arduino juga merupakan platform hardware terbuka yang ditujukan kepada siapa saja yang ingin membuat purwarupa peralatan elektronik interaktif berdasarkan hardware dan software yang fleksibel dan mudah digunakan. Mikrokontroler diprogram menggunakan bahasa pemrograman arduino yang memiliki kemiripan syntax dengan bahasa pemrograman C. Karena sifatnya yang terbuka maka siapa saja dapat mengunduh skema hardware arduino dan membangunnya. [10]

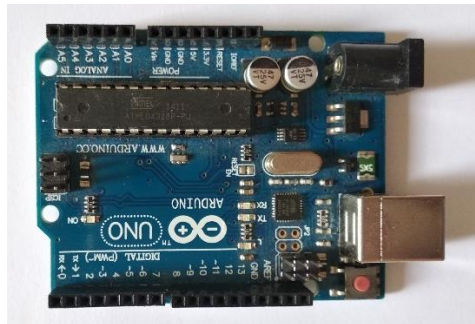
Terdapat tiga jenis memori yang ada pada mikrokontroler papan Arduino berbasis AVR [19], yaitu :

- Flash memory, berfungsi sebagai tempat untuk menyimpan sketsa Arduino.
- SRAM adalah tempat sketsa menciptakan dan memanipulasi variable yang sedang dijalankan atau digunakan.

c. EEPROM adalah memori yang dapat digunakan untuk programmer menyimpan informasi jangka panjang.

Pada Flash memory dan EEPROM informasi yang disimpan tetap ada walaupun power telah dimatikan. Sedangkan untuk SRAM, informasi yang disimpan akan hilang ketika power dimatikan.

Arduino UNO merupakan salah satu jenis mikrokontroller Arduino yang ada di pasaran. Arduino UNO menggunakan chipset ATmega 328, sebagaimana yang ditunjukkan pada Gambar 2.2



Gambar 2.2 Arduino UNO

Berikut ini adalah data sheet yang ada pada Arduino UNO Chip ATmega 238 (ditunjukkan pada Table 2.1).

Tabel 2.1 Arduino UNO Data Sheet

Mikrokontroller	ATmega238
Operating Voltage	5V
Input Voltage (recommended)	7 - 12V
Input Voltage (limits)	6 – 20 V
Digital I/O Pins	14 (6 pin merupakan pulse width modulation output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA

DC Current for 3.3V Pin	50 mA
Flash memory	32 KB (0.5KB digunakan untuk bootloader)
SRAM	2 KB
EPPROM	1 KB
Clock Speed	16 MHz

2.3 Arduino Mega 2560

Arduino Mega 2560 merupakan salah satu jenis mikrokontroler Arduino yang menggunakan chip ATmega2560. Mikrokontroler ini memiliki pin I/O yang cukup banyak yaitu sejumlah 54 buah pin digital. Berikut ini adalah gambar Arduino Mega 2560 sebagaimana yang ditunjukkan pada Gambar 2.3 beserta data sheet yang ada pada Arduino Mega2560 Chip ATmega 2560 (ditunjukkan pada Table 2.2).



Gambar 2.3 Arduino Mega 2560

Tabel 2.2 Arduino Mega 2560 Data Sheet

Mikrokontroler	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7 - 12V

Input Voltage (limits)	6 – 20 V
Digital I/O Pins	54 (14 pin merupakan pulse width modulation output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash memory	256 KB (8KB digunakan untuk bootloader)
SRAM	8 KB
EPPROM	4 KB
Clock Speed	16 MHz

2.4 Arduino Integrated Development Environment

Arduino Integrated Development Environment (IDE) merupakan editor teks untuk menulis kode, area pesan, konsol teks dan memiliki toolbar dengan tombol untuk fungsi umum dan serangkaian menu. Arduino IDE terhubung ke perangkat keras Arduino dan Genuino untuk mengunggah program dan dapat berkomunikasi dengan meraka. Program yang ditulis menggunakan Arduino Software (IDE) disebut sketsa. Sketsa ini ditulis dalam editor teks dan disimpan dengan ekstensi file .ino. Editor memiliki fitur untuk memotong / menempel dan mencari / mengganti teks. Area pesan memberi umpan balik saat menyimpan dan mengeksport dan menampilkan kesalahan. Konsol menampilkan output teks oleh Arduino Software (IDE), termasuk pesan kesalahan dan informasi lainnya yang lengkap [11].



Gambar 2.4 Antarmuka Arduino IDE

2.5 XBee Shield

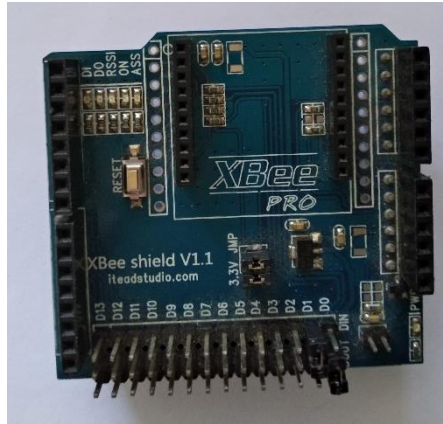
Shield adalah papan PCB atau lebih dikenal board yang dapat dihubungkan dengan papan Arduino untuk menambah fungsi dari arduino. XBee Shield ini dirancang dengan agar arduino dapat

berkomunikasi secara nirkabel dengan modul XBee dari Maxstream. Dengan adanya modul ini, sebuah arduino akan mampu berkomunikasi secara nirkabel melebihi 30 meter di dalam ruangan dan 90 meter jika di luar ruangan. Dapat digunakan sebagai serial ataupun USB [12].



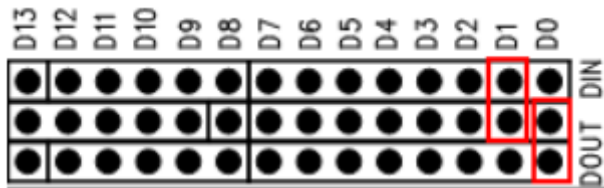
Gambar 2.5 XBee Shield pabrik Funduino

Pada penelitian ini, penulis menggunakan dua tipe XBee Shield. Pertama XBee Shield V03 keluaran dari perusahaan Funduino, dimana jumper yang digunakan bertipe switch, sehingga user lebih mudah untuk memilih mode jumper. Terdapat dua jenis jumper, yaitu mode XBee (berfungsi untuk pengiriman data) dan mode USB (berfungsi untuk programming). Perlu di perhatikan ketika memilih mode jumper, ketika ingin melakukan upload code ke arduino jumper harus berada dalam mode USB jika tidak maka code tidak akan bisa terupload. Begitu pula sebaliknya, ketika ingin mengirimkan data via wireless jumper harus berada dalam mode XBee.

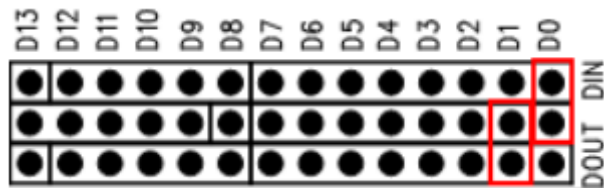


Gambar 2.6 XBee Shield pabrikan ITEAD Studio

Adapun XBee Shield jenis kedua merupakan XBee Shield V1.1 keluaran perusahaan ITEAD Studio. Terdapat dua jenis jumper yaitu mode USB dan mode XBee. Berikut posisi jumper shield tersebut



Gambar 2.7 Konfigurasi Jumper mode USB



Gambar 2.8 Konfigurasi Jumper mode XBee

2.6 XBee Modul S2

XBee modul seri 2 merupakan modul yang berfungsi untuk komunikasi antar jaringan nirkabel. Xbee modul seri 2 menggunakan protokol ZigBee untuk saling berkomunikasi. Modul ini menyediakan transfer data yang sangat handal dengan kecepatan transfer mencapai 250 kbps. Setiap perangkat output serial dapat menggunakan modul untuk transfer data, transmisi point to point dan transmisi jaringan multi-point. Modul ini dirancang untuk aplikasi high-throughput (35kbps) yang membutuhkan latency rendah dan waktu komunikasi yang dapat diprediksi.



Gambar 2.9 XBee Modul S2

2.7 Protokol ZigBee

ZigBee merupakan standar komunikasi untuk perangkat nirkabel jarak pendek berdaya rendah yang berbasis pada standar IEEE 802.15.4 untuk jaringan area pribadi (PAN). Perangkat Zigbee mampu berkomunikasi peer-to-peer, point-to-multipoint dan mesh. Teknologi ini cocok untuk transfer data rate yang rendah, konsumsi daya yang rendah, biaya rendah, protocol jaringan nirkabel yang ditujukan untuk aplikasi otomasi dan remote control. Perangkat nirkabel yang sesuai dengan ZigBee

diperkirakan dapat melakukan transmisi 10 sampai 75 meter tergantung pada lingkungan RF dan konsumsi daya yang dikeluarkan untuk aplikasi tertentu. ZigBee memiliki tiga jenis tipe, yaitu 2.4GHz global (data rate 250kbps), 915MHz Americas (data rate 40kbps) dan 868 MHz Europe (data rate 20kbps). Jaringan ZigBee terdiri dari tiga jenis perangkat, yaitu: koordinator, router, dan end devices. Setiap jaringan memiliki ID PAN 16bit. Semua perangkat dalam jaringan ZigBee diberi satu ID PAN.

Berikut ini adalah penjelasan lebih lanjut mengenai ketiga jenis perangkat ZigBee [13] :

- a. ZigBee Coordinator (ZC) : ZC bertindak sebagai coordinator yang mengatur lalu lintas jaringan komunikasi. Harus ada satu ZC dalam setiap jaringan karena perangkat ini memulai jaringan dari awal. Koordinator memulai Personal Area Network (PAN) dengan memilih saluran RF dan PAN ID. ZC memungkinkan router dan end-devices untuk bergabung dengan PAN. Selain itu ZC, mampu menyimpan informasi tentang jaringan, termasuk bertindak sebagai Trust Center dan repository untuk kunci keamanan.
- b. ZigBee Router (ZR) : ZR menjalankan fungsi aplikasi, selain itu router bertindak sebagai perantara, meneruskan data dari satu perangkat ke perangkat lain.
- c. ZigBee End Device (ZED) : ZED dapat melakukan komunikasi dengan koordinator dan router, akan tetapi tidak dapat menyampaikan data dari perangkat lain. Hubungan ini memungkinkan simpul untuk tidur dalam waktu yang cukup lama, sehingga dapat menghemat penggunaan baterai. ZED harus bergabung dengan PAN seperti router sebelum mengirimkan data sensor.

Kelebihan menggunakan ZigBee terutama terletak pada mode AT default nya, dimana lapisan PHY dan MAC frame transparan bagi pengguna. Artinya, pengguna biasa tidak akan melihat frame acknowledgment (ACK) atau transmisi ulang modul frekuensi radio (RF) termasuk semua byte aktual yang dikirim.

Pengguna hanya akan menyaksikan apakah data berhasil dikirim atau tidak, dengan semua teknis seperti Carrier Sense Multiple Access - Collision Avoidance (CSMA-CA) tersembunyi dari pandangan biasa dan karenanya menawarkan antarmuka yang lebih sederhana [7].

Beban maksimum yang dapat dialokasikan buffer XBee untuk 802.15.4 dan ZigBee masing-masing adalah 100 dan 72 byte. Manfaat buffer adalah pembacaan sensor ganda dapat dimasukkan ke dalam frame yang sama untuk satu sesi transmisi selama buffer overflow dihindari. [14] menunjukkan bahwa lonjakan arus untuk daya ZigBee menghabiskan 5 sampai 10 kali lebih besar daripada pada operasi normal. Oleh karena itu, buffer harus digunakan untuk mentransmisikan data sebanyak mungkin dalam interval yang dapat ditoleransi dengan aktivasi modul RF minimum.

2.8 Algoritma Heatshrink

Heatshrink merupakan algoritma kompresi lossless yang berbasis pada Lempel-Ziv-Storer-Szymanski (LZSS). Algoritma kompresi lossless memungkinkan untuk membentuk data asli yang tepat sama dari data yang sudah dikompresi. Algoritma ini cocok digunakan pada sistem *embedded* karena dapat berjalan dalam jumlah memori yang sangat kecil (dibawah 50 byte untuk dekompresi praktis). Selain itu, heatshrink dapat bekerja sedikit demi sedikit sambil menangani kebutuhan lain dari system yang berjalan secara *real time* [15].

Heatshrink menggunakan algoritma Lempel-Ziv-Storer-Szymanski untuk melakukan kompresi dengan beberapa detail implementasi penting yang harus diperhatikan, yaitu [15] :

1. Proses kompresi dan dekompresi telah dirancang untuk berjalan secara bertahap, pemrosesan dapat bekerja beberapa byte setiap saat. Selain itu dapat juga menangguhkan dan melanjutkan proses sebagai data tambahan atau pada buffer yang tersedia.

2. Teknik optional indexing yang digunakan dapat mempercepat proses kompresi.
3. Secara umum trade-off implementasi banyak disukai pada penggunaan memori yang rendah.

Lempel-Ziv-Storer-Szymanski (LZSS) adalah salah satu jenis algoritma kompresi yang berbasis dictionary yang bersifat lossless (data dapat di rekonstruksi ulang menjadi data asli). LZSS merupakan salah satu varian dari LZ77 (Lempel Ziv 1977) yang dikembangkan oleh Storer dan Szymansky pada tahun 1982. Perbedaan yang mendasar antara kedua algoritma ini adalah jumlah token (tanda) yang terbentuk yakni dua token pada LZSS dan tiga token pada LZ77. Dua token yang dihasilkan oleh LZSS menunjukkan indeks dan panjang karakter yang sama pada dictionary. Sedangkan pada algoritma LZ77, dua token awal mempunyai fungsi sama dengan LZSS namun ada tambahan satu token yang berisi satu karakter yang mengikuti frasa yang sama tersebut [16].

Untuk proses kompresi dan dekompresi akan dijelaskan secara detail sebagai berikut [17].

1. Proses Kompresi

Buffer dibagi menjadi dua, yakni buffer untuk pencarian dan buffer look-ahead. Setelah menginisialisasi buffer, karakter dibaca dari input data ke buffer data yang belum di kodekan. Untuk setiap karakter pada buffer yg belum di kodekan, dilakukan proses pencarian substring yang terpanjang di buffer pencarian sesuai dengan buffer look-ahead dimulai dengan karakter inputan pertama. Jika kecocokan substring sudah cukup, maka program akan mengkodekan indeks dan panjang substring ke dalam output. Jika tidak ada substring yang cocok dimulai dengan input pertama karakter masukan yang diberikan, maka karakter tersebut akan langsung ditulis ke output dengan *flag* yang menandakan tidak ada pengkodean yang dilakukan. Algoritma ini melakukan langkah-langkah ini sampai tidak ada karakter yang tertinggal. Pengkodean dua karakter yang sesuai membutuhkan jumlah byte yang sama jika kita langsung menampilkan dua karakter.

Berikut ini adalah contoh ilustrasi dari proses kompresi data [18]. Pada Gambar 5 menampilkan text asli yang masih belum di kompresi. Pada Gambar 6 akan menampilkan hasil kompresi yang telah dilakukan oleh algoritma LZSS.

```

0: I am Sam
9:
10: Sam I am
19:
20: That Sam-I-am!
35: That Sam-I-am!
50: I do not like
64: that Sam-I-am!
79:
80: Do you like green eggs and ham?
112:
113: I do not like them, Sam-I-am.
143: I do not like green eggs and ham.

```

Gambar 2.10 Contoh text asli [18]

```

0: I am Sam
9:
10: (5,3) (0,4)
16:
17: That(4,4)-I-am!(19,16)I do not like
45: t(21,14)
49: Do you(58,5) green eggs and ham?
78: (49,14) them,(24,9).(112,15)(93,18).

```

Gambar 2.11 Hasil teks yang ditelaah dikompresi [18]

Pada text asli yang belum mengalami pengompresan, jumlah byte yang dihasilkan adalah 177 byte dari 177 karakter (termasuk spasi dan enter). Setelah dilakukan kompresi, jumlah byte berkurang menjadi 94 byte. Ini tidak termasuk 12 byte pada flag yang menunjukkan apakah potongan teks berikutnya adalah pointer atau literal. Jika ditambahkan dengan jumlah flag maka

total ukurannya menjadi 106 byte, tentunya ini masih lebih pendek jika dibandingkan dengan ukuran aslinya 177 byte.

2. Proses Dekompresi

Proses dekompresi dilakukan dengan menguraikan kode secara langsung dengan melibatkan membaca dan menulis ulang hasil tanpa melakukan pencarian apapun. Flag pengkodean dibaca untuk mengetahui karakter mana yang dikodekan. Jika flag menunjukkan bahwa karakter tersebut dikodekan, jumlah karakter dan posisi awal dikumpulkan dari bagian yang dikodekan. Kemudian jumlah karakter dengan indeks yang diberikan ditulis dari jendela geser ke file output atau memori. Jika tidak dikodekan, karakternya adalah output secara langsung. Dekompresi mengkonsumsi lebih sedikit sumber daya memori dan waktu komputasi jika dibandingkan dengan proses kompresi.

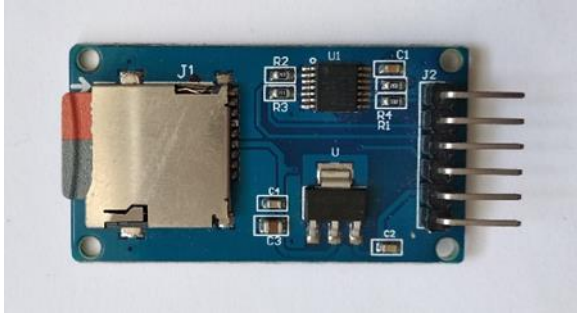
2.9 Bahasa Pemrograman C

Bahasa C merupakan salah satu bahasa pemrograman level tingkat menengah yang menjadi induk dari bahasa pemrograman modern seperti C++, C#, PHP, Javascript dan masih banyak lagi. Bahasa pemrograman C dibuat pertama kali oleh Dennis M Ritchie dengan tujuan untuk mengembangkan sistem operasi UNIX yang sebelumnya menggunakan bahasa assembly. Adapun beberapa keunggulan bahasa C dibandingkan dengan bahasa pemrograman yang lain, yaitu: bahasa C termasuk bahasa pemrograman procedural, bahasa C sangat cepat dan efisien, dan Bahasa C merupakan portable language.

2.10 MicroSD Card Adapter

MicroSD Card Adapter ini merupakan modul pembaca kartu MicroSD melalui system file dan SPI antarmuka driver, MCU untuk membaca dan menulis pada kartu microSD. Dengan

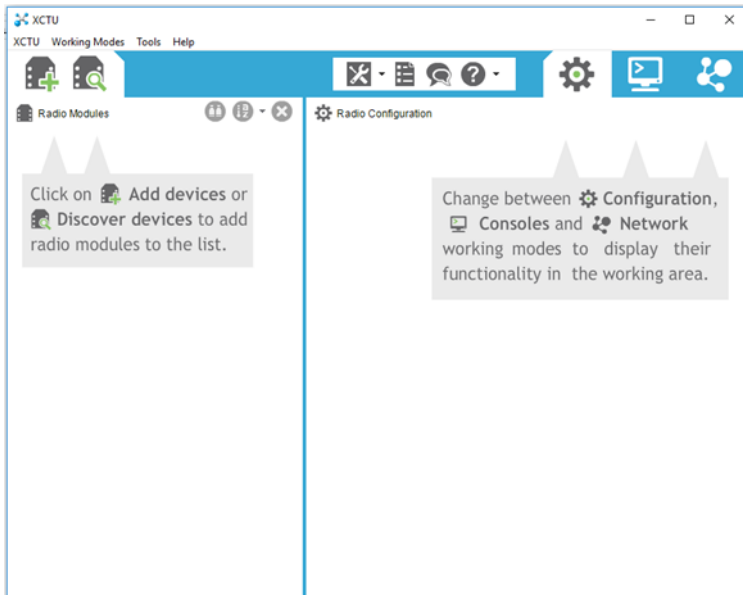
menggunakan Arduino IDE dan library SD card, pengguna dapat menginisialisasi kartu SD card, membaca dan menulisnya.



Gambar 2.12 MicroSD Card Adapter

2.11 DIGI XCTU

XCTU merupakan aplikasi multi-platform gratis yang dirancang untuk memungkinkan pengembangan modul Digi RF melalui antarmuka grafis yang mudah digunakan. Aplikasi ini dapat digunakan untuk melakukan konfigurasi dan pengujian pada modul XBee® RF. XCTU memiliki semua tools yang dibutuhkan pengembang untuk melakukan pengembangan dengan XBee. Terdapat fitur unik didalamnya, seperti tampilan jaringan grafis, yang secara grafis mewakili jaringan XBee yang ada bersama dengan kekuatan sinyal setiap sambungan. Selain itu kita dapat menggunakan XBee API yang secara intuitif membantu dalam membangun dan menafsirkan API frame untuk XBee yang menggunakan mode API. Dengan menggunakan aplikasi ini, kita lebih mudah dalam melakukan pengembangan pada platform wireless sensor network yang menggunakan XBee.



Gambar 2.13 Antarmuka XCTU

BAB III

PERANCANGAN PERANGKAT LUNAK

Bab ini membahas mengenai dasar dari perancangan sistem yang akan dibangun pada tugas akhir. Perancangan yang dibahas meliputi deskripsi umum sistem, proses perancangan, alur dan implementasinya.

3.1 Deskripsi Umum Sistem

Pada tugas akhir ini akan dibangun suatu sistem kompresi dan dekompresi data pada platform wireless sensor network dengan menggunakan algoritma heathshrink. Teknologi wireless sensor network menggunakan mikrokontroler Arduino dan protokol ZigBee sebagai jalur komunikasi.



Gambar 3.1 Deskripsi Umum Sistem

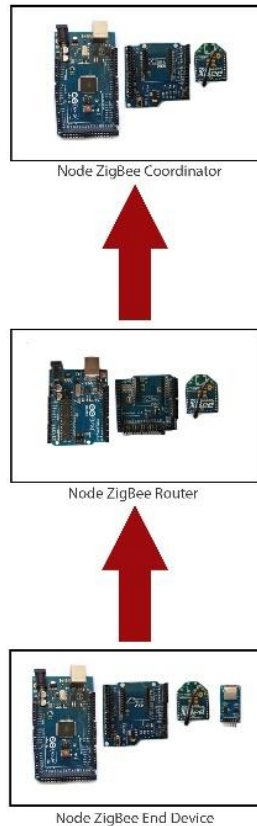
Terdapat tiga node yang akan digunakan, yaitu Coordinator, Router dan End Device. Setiap node memiliki komponen yang saling terhubung. Penjelasan komponen pendukung pada ketiga node tersebut sebagai berikut.

- **Node Coordinator**
Node ZigBee coordinator berfungsi untuk menerima data dari node Router. Data tersebut kemudian akan dikompresi sesuai konfigurasi yang ditentukan. Berikut ini adalah komponen pendukung sistem pada node Coordinator.
 1. Mikrokontroler Arduino Mega sebagai computer yang mengendalikan sistem pada node Coordinator
 2. XBee Shield
 3. XBee Modul S2 sebagai perangkat radio komunikasi

- **Node Router**
Node ZigBee Router berfungsi untuk menerima data hasil kompresi dari node End Device kemudian meneruskannya ke node Coordinator. Berikut ini adalah komponen pendukung sistem pada node Router.
 1. Mikrokontroler Arduino UNO sebagai computer yang mengendalikan sistem pada node Router
 2. XBee Shield
 3. XBee Modul S2 sebagai perangkat radio komunikasi
- **Node End Device**
Node ZigBee End Device bertugas melakukan kompresi data dan mengirimkan hasilnya beserta konfigurasi yang digunakan ke node Router. Berikut ini adalah komponen pendukung sistem pada node End Device.
 1. Mikrokontroler Arduino Mega sebagai computer yang mengendalikan sistem pada node End Device
 2. XBee Shield
 3. XBee Modul S2 sebagai perangkat radio komunikasi
 4. MicroSd card adapter yang berfungsi untuk menampung data yang akan di kompresi

3.2 Arsitektur Umum Sistem

Teknologi wireless sensor network yang dikembangkan pada tugas akhir ini, menggunakan perangkat XBee S2 yang akan berperan sebagai perangkat yang membantu komunikasi antar node dengan protokol ZigBee sebagai jalur komunikasinya. Arsitektur jaringan yang digunakan adalah *cluster tree*, dimana terdapat tiga jenis perangkat ZigBee yang dibutuhkan, yaitu node ZigBee Coordinator, node ZigBee Router dan node ZigBee End Device.



Gambar 3.2 Arsitektur detail sistem

Berdasarkan Gambar 3.2 sistem kompresi dan dekompresi data pada platform wireless sensor network memiliki alur proses yang akan dijabarkan sebagai berikut :

1. Pada ZigBee Coordinator, XBee Shield dipasang diatas Arduino Mega 2560, kemudian diatas XBee Shield tersebut dipasang modul XBee S2

2. Pada ZigBee Router, XBee Shield dipasang diatas Arduino Mega UNO, kemudian diatas XBee Shield tersebut dipasang modul XBee S2
3. Pada ZigBee End Device, XBee Shield dipasang diatas Arduino Mega 2560, kemudian diatas XBee Shield tersebut dipasang modul XBee S2. Selain itu dipasang juga modul MicroSD Card Adapter dihubungkan menggunakan kabel jumper ke pin Arduino Mega.

3.3 Perancangan Komunikasi Sistem

Perancangan komunikasi sistem merupakan salah satu proses penting yang harus dilakukan dalam membangun sistem agar node pada jaringan dapat saling berkomunikasi. Protokol jaringan komunikasi yang digunakan adalah protokol ZigBee, dimana pada jaringan tersebut minimal terdapat sebuah node yang bertindak sebagai Coordinator.

Pada penelitian tugas akhir ini, akan menggunakan satu buah node yang berfungsi sebagai ZigBee End Device, satu buah node yang berfungsi sebagai ZigBee Router dan satu node sebagai ZigBee Coordinator. Agar setiap node dapat berkomunikasi, harus dilakukan konfigurasi terlebih dahulu terhadap node coordinator dan router dengan memanfaatkan aplikasi XCTU yang disediakan oleh Digi International Inc. Beberapa hal yang harus di perhatikan pada saat melakukan konfigurasi antara lain :

- **Function Set**
- **ID PAN ID**
- **DH** Destination Address High
- **DL** Destination Address Low

Berikut ini adalah konfigurasi secara detail yang digunakan pada ZigBee Coordinator, ZigBee Router dan ZigBee End Device.

Product family: XB24-ZB

Function set: ZigBee Coordinator AT

Firmware version: 20A7

▼ Networking

Change networking settings

i ID PAN ID	121			
i SC Scan Channels	FFFF	Bitfield		
i SD Scan Duration	3	exponent		
i ZS ZigBee Stack Profile	0			
i NJ Node Join Time	FF	x 1 sec		
i OP Operating PAN ID	121			
i OI Operating 16-bit PAN ID	CED1			
i CH Operating Channel	C			
i NC Number of Remaining Children	A			

▼ Addressing

Change addressing settings

i SH Serial Number High	13A200			
i SL Serial Number Low	40ABC921			
i MY 16-bit Network Address	0			
i DH Destination Address High	0			
i DL Destination Address Low	40CC0408			
i NI Node Identifier	COORDINATOR			
i NH Maximum Hops	1E			
i BH Broadcast Radius	0			
i AR Many-to-One Route Broadcast Time	FF	x 10 sec		
i DD Device Type Identifier	30000			
i NT Node Discovery Backoff	3C	x 100 ms		
i NO Node Discovery Options	0			
i NP Maximum Number of Transmission Bytes	54			
i CR PAN Conflict Threshold	3			

Gambar 3.3 Konfigurasi pada node ZigBee Coordinator

Function Set yang digunakan pada konfigurasi node ZigBee Coordinator adalah **ZigBee Coordinator AT**. Untuk **PAN ID (ID)** nya **121**. PAN ID ini memiliki fungsi yang hampir sama dengan subnet, dimana semua node yang ada pada jaringan ZigBee nilainya harus sama agar dapat saling berkomunikasi. **Destination Address High (DH)** yang digunakan yaitu **0** dan **Destination Address Low (DL)** yaitu **40CC0408**. Konfigurasi yang digunakan

bertujuan agar node ZigBee Coordinator hanya dapat menerima pesan dari node ZigBee Router.

Product family: XB24-ZB

Function set: ZigBee Router AT

Firmware version: 22A7

▼ Networking

Change networking settings

i ID PAN ID	121		
i SC Scan Channels	FFFF	Bitfield	
i SD Scan Duration	3	exponent	
i ZS ZigBee Stack Profile	0		
i NJ Node Join Time	FF	x 1 sec	
i NW Network Watchdog Timeout	0	x 1 minute	
i JV Channel Verification	Disabled [0]		
i JN Join Notification	Disabled [0]		
i OP Operating PAN ID	121		
i OI Operating 16-bit PAN ID	CED1		
i CH Operating Channel	C		
i NC Number of Remaining Children	C		

▼ Addressing

Change addressing settings

i SH Serial Number High	13A200	
i SL Serial Number Low	40B79F40	
i MY 16-bit Network Address	A56C	
i DH Destination Address High	0	
i DL Destination Address Low	0	
i NI Node Identifier	ROUTER	
i NH Maximum Hops	1E	
i BH Broadcast Radius	0	
i AR Many-to-One Route Broadcast Time	FF	x 10 sec
i DD Device Type Identifier	30000	
i NT Node Discovery Backoff	3C	x 100 ms
i NO Node Discovery Options	0	
i NP Maximum Number of Transmission Bytes	54	
i CR PAN Conflict Threshold	3	

Gambar 3.4 Konfigurasi pada node ZigBee Router

Setelah mensetting node Coordinator, langkah selanjutnya adalah mensetting node Router. **Function Set** yang digunakan pada konfigurasi node ZigBee Router adalah **ZigBee Router AT**. Untuk **PAN ID (ID)** nilainya **121**. **Destination Address High (DH)** yang digunakan yaitu **0** dan **Destination Address Low (DL)** yaitu **0**. Konfigurasi yang digunakan bertujuan agar node Router dapat menerima pesan dari node ZigBee End Device dan dapat meneruskannya ke node ZigBee Coordinator.

Product family: XB24-ZB **Function set: ZigBee End Device AT** Firmware version: 28A7

▼ Networking
Change networking settings

i	ID PAN ID	121			
i	SC Scan Channels	FFFF	Bitfield		
i	SD Scan Duration	3	exponent		
i	ZS ZigBee Stack Profile	0			
i	NJ Rejoin Policy	FF			
i	JN Join Notification	Disabled [0]			
i	OP Operating PAN ID	121			
i	OI Operating 16-bit PAN ID	CED1			
i	CH Operating Channel	C			

▼ Addressing
Change addressing settings

i	SH Serial Number High	13A200			
i	SL Serial Number Low	40CC0408			
i	MY 16-bit Network Address	BE80			
i	MP 16-bit Parent Address	0			
i	DH Destination Address High	13A200			
i	DL Destination Address Low	40B79F40			
i	NI Node Identifier	END DEVICE			
i	NH Maximum Hops	1E			
i	BH Broadcast Radius	0			
i	DD Device Type Identifier	30000			
i	NT Node Discovery Backoff	3C	x 100 ms		
i	NO Node Discovery Options	0			
i	NP Maximum Number of Transmission Bytes	54			
i	CR PAN Conflict Threshold	3			

Gambar 3.5 Konfigurasi pada node ZigBee End Device

Function Set yang digunakan pada konfigurasi node ZigBee Router adalah **ZigBee End Device AT**. Untuk **PAN ID (ID)** nilainya **121**. **Destination Address High (DH)** yang digunakan adalah **13A200**, dimana nilai ini didapat dari **Serial Number High (SH)** pada node ZigBee Router. Selainnya itu **Destination Address Low (DL)** nilainya **40B79F40** yang merupakan nilai **Serial Number Low (SL)** pada node ZigBee Router. Konfigurasi ini bertujuan agar node ZigBee End Device dapat mengirim pesan pada node ZigBee Router.

3.4 Perancangan Kompresi dan Dekompresi Data

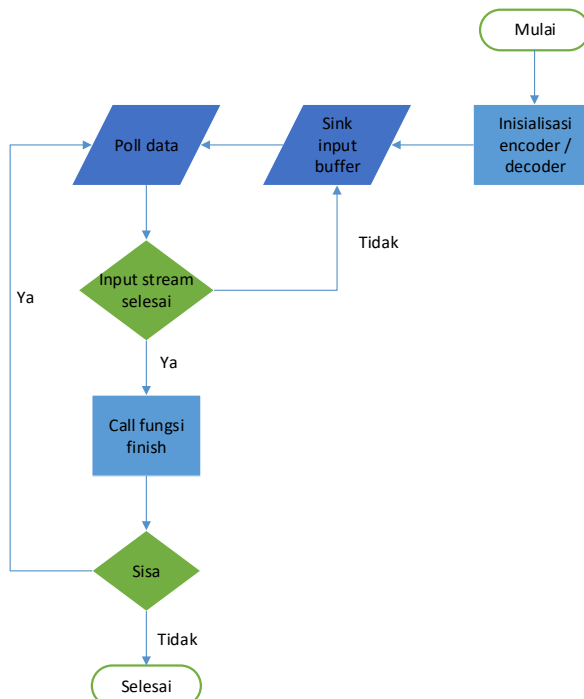
Pada bagian ini akan dijelaskan bagaimana perancangan yang dilakukan sebelum data di kompresi. Untuk melakukan proses kompresi dan dekompresi, pada penelitian ini menggunakan algoritma Heatsrink, dimana algoritma ini berbasis pada algoritma Lempel-Ziv-Storer-Szymanski (LZSS).

Kebutuhan memori merupakan suatu hal yang mendasar dalam melakukan perancangan kompresi dan dekompresi data, sebab jumlah memori yang tersedia pada Arduino sangat terbatas. Pada algoritma heatsrink ukuran buffer telah ditetapkan untuk memungkinkan terjadinya *trade-off* antara efektivitas kompresi dengan memori kerja. Persyaratan yang dibutuhkan untuk penggunaan buffer IO adalah sebagai berikut [15]:

- a. Encoding
 $16 + 2 * 2^N$ byte untuk encoding, ditambah lagi untuk indeks pencarian optional yakni $2 * 2^N$ byte untuk mempercepat pengkodean
- b. Decoding
 $16 + 2^N$ byte untuk decoding, dimana N dapat di atur pada saat pengodean (encoding)

3.4.1 Cara Kerja Algoritma Heatsrink

Lempel-Ziv-Storer-Szymanski (LZSS) membutuhkan sedikit ruang kerja dimana cache yang diperlukan adalah 2^N byte dari data terakhir (N dapat dikonfigurasi) dan proses dekompresi yang dilakukan cukup sederhana. System yang dijalankan juga cukup sederhana, dump data kedalam, terus putar engkol sampai tidak ada lagi data yang keluar, dump data lebih banyak, ulangi proses tersebut. Notifikasi encoder / decoder ketika akhir input telah tercapai, lakukan sampai selesai. Pada Gambar 3.6 menjelaskan diagram alir cara kerja algoritma Heatsrink.



Gambar 3.6 Diagram alir cara kerja algoritma heatsrink

Berikut ini adalah cara kerja pada algoritma Heatsrink :

1. Alokasikan **heatshrink_encoder** atau **heatshrink_decoder** pada state machine menggunakan fungsi **alloc** atau dapat menggunakan **static alloc** dan panggil fungsi reset untuk memulai inisialisasi
2. Gunakan **sink** untuk memasukkan input buffer kedalam state machine. Pointer pada `input_size` digunakan untuk menunjukkan seberapa banyak byte dari buffer input yang digunakan (jika nilainya 0 maka buffernya penuh)
3. Gunakan **poll** untuk memindahkan output dari state machine ke buffer output. Pointer pada `output_size` menunjukkan berapa banyak byte yang dihasilkan dan fungsi return menunjukkan apakah output selanjutnya tersedia (State machine tidak boleh mengeluarkan data sampai ia menerima input yang cukup)
4. Ulangi langkah 2 dan 3 untuk melakukan stream data melalui state machine. Pada saat kompresi data, ukuran input dan output dapat bervariasi secara signifikan. Looping diperlukan untuk buffer input dan output dalam pemrosesan data.
5. Ketika input stream selesai, panggil fungsi **finish** untuk memberitahu bahwa state machine tidak lagi bisa menerima input. Nilai kembalian dari proses yang telah selesai menunjukkan apakah ada output yang tersisa. Jika ada, panggil fungsi **poll** lagi.
6. Kemudian panggil fungsi **finish** dan flush sisa output hingga selesai, sampai sisa output habis.

3.4.2 Konfigurasi Algoritma Heatshrink

Heatshrink memiliki beberapa opsi konfigurasi yang dapat mempengaruhi penggunaan sumberdaya dan seberapa efektif ia dapat memampatkan data. Konfigurasi ini dapat diatur secara dinamis pada saat akan melakukan kompresi dan dekompresi atau dapat pula di setting statis pada file `heatshrink_config.h`. Adapun konfigurasi yang dimaksudkan adalah sebagai berikut :

- **window_sz2**

Ukuran window menentukan seberapa panjang input yang dapat dicari untuk pola yang berulang. Semakin besar ukuran window maka akan menggunakan memori semakin banyak, tetapi dapat melakukan kompresi lebih efektif dalam mendeteksi pengulangan yang lebih banyak. Sebuah window_sz2 = 8 akan menggunakan memori 256 byte (2^8), sedangkan window_sz2 = 10 akan menggunakan memori 1024 byte (2^{10}). Pengaturan window_sz yang tersedia adalah antara **4 sampai 15**

- **lookahead_sz2**

Ukuran lookahead menentukan panjang maksimal untuk pola berulang yang ditemukan. Jika lookahead_sz2 adalah 4, 'a' 50-bit dari karakter 'a' akan direpresentasikan sebagai pola 16-byte berulang (2^4). Jumlah bit yang digunakan untuk ukuran lookahead bersifat tetap, sehingga ukuran lookahead yang besar dapat mengurangi kompresi dengan menambahkan bit yang tidak digunakan ke pola-pola kecil. Pengaturan lookahead_sz2 yang ada saat ini adalah antara **3 sampai window_sz - 1**.

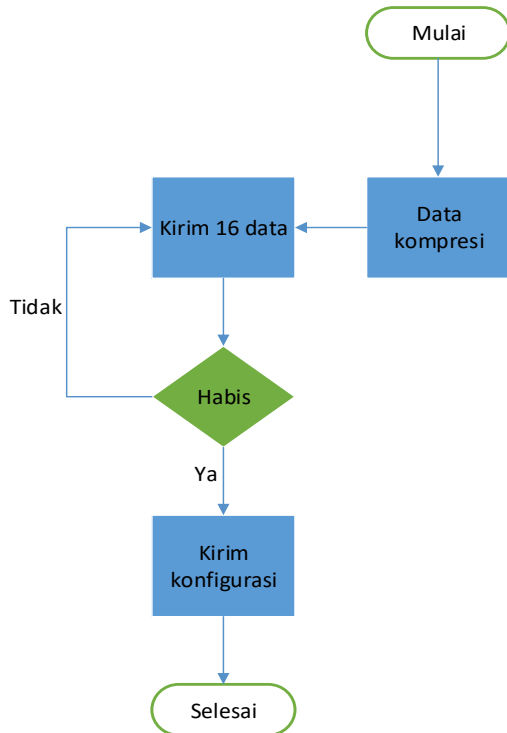
- **input_buffer_size**

Besar atau kecilnya buffer input yang digunakan untuk decoder ditentukan oleh **input_buffer_size**. Ukuran buffer input berdampak pada seberapa banyak pekerjaan yang dapat dilakukan decoder dalam satu langkah, dan semakin besar buffer maka memori yang dibutuhkan semakin banyak. Buffer yang sangat kecil (misalnya 1 byte) akan menambah overhead karena banyak melakukan pemanggilan fungsi suspend / resume, akan tetapi **input_buffer_size** tidak mempengaruhi seberapa baik dalam melakukan kompresi data.

3.5 Perancangan Pengiriman Data

Pengiriman data adalah suatu hal yang perlu diperhatikan dalam perancangan system ini, sebab keterbatasan buffer yang ada pada protokol ZigBee yaitu sebesar 72 byte menjadi landasan dasar untuk melakukan perancangan pengiriman data yang tepat.

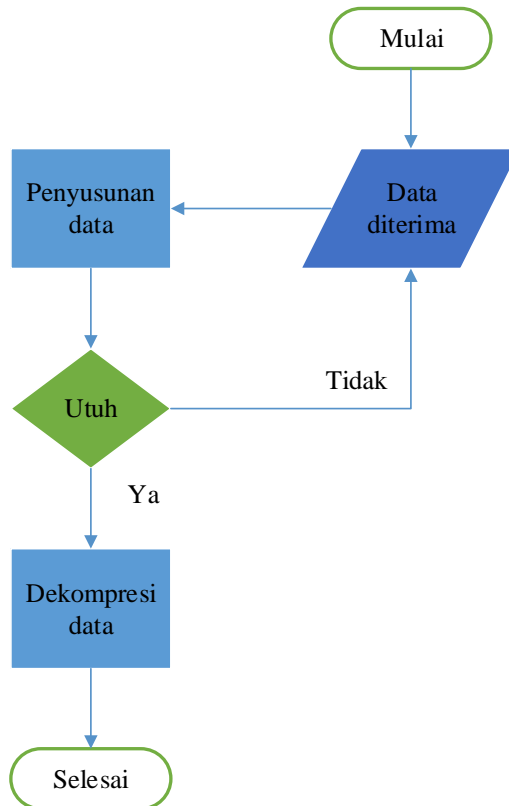
Pada penelitian ini protocol ZigBee menggunakan mode AT atau lebih dikenal mode “Transparan”. Dalam mode AT, data akan segera dikirim ke modul jarak jauh yang diidentifikasi melalui alamat tujuan yang ada pada memori modul XBee. Alamat tujuan dapat di konfigurasi oleh pengguna pada mode Command. Jika XBee mengirimkan data ke Coordinator maka akan di broadcast pada PanID. Informasi paket tidak diperlukan, tetapi prosesnya lebih sederhana, dimana Serial data dikirimkan ke Tx dari satu XBee dan akan diterima oleh Rx tujuan XBee. Mode AT cocok digunakan pada jaringan yang sangat sederhana, karena tidak perlu untuk mengubah alamat tujuan terlalu sering. Pada Gambar 3.7 menjelaskan bagaimana proses pengiriman data dari node ZigBee End Device menuju node ZigBee Router.



Gambar 3.7 Diagram alir pengiriman data

3.6 Perancangan Dekompresi Data

Data yang dikirimkan oleh node ZigBee Router akan di terima oleh node ZigBee Coordinator melalui jaringan ZigBee. Potongan – potongan data tersebut akan ditampung terlebih dahulu dan kemudian akan di satukan kembali untuk di proses kembali. Proses dekompresi akan dilakukan ketika data dekompresi yang diterima sudah utuh beserta konfigurasi encoder / decoder yang digunakan. Pada Gambar 3.8 menjelaskan bagaimana proses dekompresi data.



Gambar 3.8 Diagram alir dekompresi data

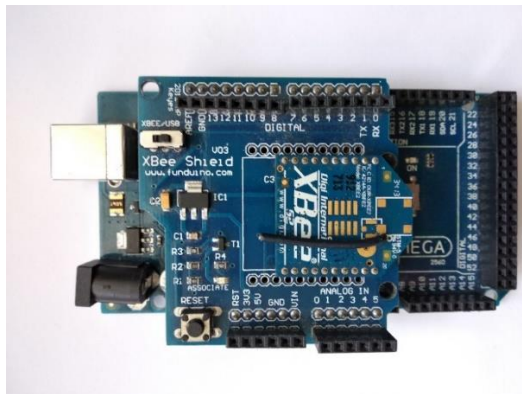
3.7 Perancangan Perangkat Keras

Perancangan perangkat keras secara umum menjelaskan mengenai penempatan perangkat keras yang digunakan dalam membangun sistem, yang mana terdiri dari rangkaian ZigBee Coordinator, ZigBee Router dan ZigBee End Device. Rangkaian perangkat keras pada sistem dapat dilihat pada Gambar 3.8 dan Gambar 3.9. Agar sistem dapat berjalan sebagai mestinya, terdapat beberapa komponen yang dibutuhkan antara lain:

1. Dua buah Arduino Mega
2. Satu buah Arduino UNO
3. Tiga buah baterai 9V
4. Tiga buah XBee Shield
5. Tiga buah XBee S2
6. Satu buah MicroSD Card Adapter
7. Satu buah SD Card

3.8 Perancangan Perangkat ZigBee Coordinator

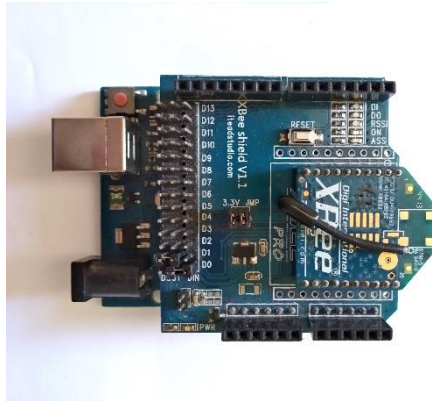
Pada rangkaian ZigBee Coordinator, XBee Shield V03 pabrikan Funduino menempati tepat di bagian atas Arduino Mega 2560 dengan posisi pin Tx dan Rx yang sama dengan pin Tx dan Rx pada XBee Shield. Kemudian modul XBee S2 Pro di letakkan pada space yang telah disediakan pada XBee Shield. Pastikan kepala modul XBee S2 menghadap arah yang berlawanan dengan konektor USB.



Gambar 3.9 Node ZigBee Coordinator

3.9 Perancangan Perangkat ZigBee Router

Pada rangkaian ZigBee Router, XBee Shield V1.1 pabrikan ITead Studio menempati tepat di bagian atas Arduino UNO dengan posisi pin Tx dan Rx yang sama dengan pin Tx dan Rx pada XBee Shield. Kemudian modul XBee S2 Pro di letakkan pada space yang telah disediakan pada XBee Shield. Pastikan kepala modul XBee S2 menghadap arah yang berlawanan dengan konektor USB.

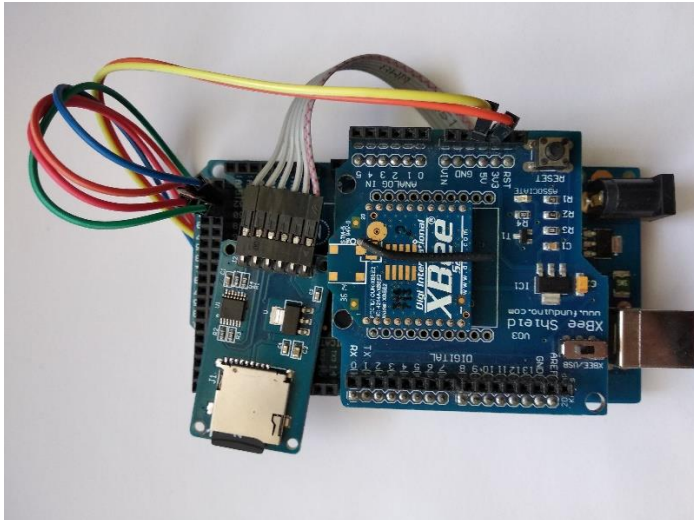


Gambar 3.10 Node ZigBee Router

3.10 Perancangan Perangkat ZigBee End Device

Pada rangkaian ZigBee End Device posisinya hampir sama dengan rangkaian ZigBee Coordinator. Xbee Shield diletakkan diatas Arduino Mega dengan posisi pin Tx dan Rx yang sama. Selain itu letakkan modul XBee S2 pada tempat yang telah disediakan. Pada perangkat ZigBee Router membutuhkan MicroSD Card Adapter yang berfungsi untuk menampung data yang akan di kompresi. Pastikan sudah terdapat SD Card pada modul tersebut, kemudian sambungkan pin yang terdapat pada MicroSD Card Adapter kepada pin yang sudah ditentukan, berikut ini rincian pinnya :

- Pin **CS** dihubungkan dengan pin **digital 53**
- Pin **SCK** dihubungkan dengan pin **digital 52**
- Pin **MOSI** dihubungkan dengan pin **digital 51**
- Pin **MISO** dihubungkan dengan pin **digital 50**
- Pin **VCC** dihubungkan dengan pin **tegangan 5V**
- Pin **GND** dihubungkan dengan pin **GND**



Gambar 3.11 Node ZigBee End Device disertai MicroSD

(Halaman ini sengaja dikosongkan)

BAB IV IMPLEMENTASI

Bab ini membahas mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijabarkan pada bab sebelumnya. Implementasi berupa pseudocode untuk membangun program. Cakupan implementasi dari perancangan sistem ini meliputi perangkat node ZigBee Router yang bertugas untuk melakukan kompresi data dan mengirimkan hasilnya ke perangkat node ZigBee Coordinator, kemudian akan di dekompresi berdasarkan konfigurasi yang telah diterima sebelumnya. Bahasa pemrograman yang digunakan adalah bahasa pemrograman C.

4.1 Lingkungan Implementasi

Lingkungan implementasi merupakan suatu lingkungan dimana sistem akan dibangun. Untuk mempermudah penjelasan, lingkungan implemetntasi akan terbagi menjadi dua bagian. Pembahasan pertama mengenai lingkungan implementasi perangkat keras dan pembahasan kedua mengenai lingkungan implementasi perangkat lunak.

4.1.1 Lingkungan Implementasi Perangkat Keras

Pada bagian ini akan dibahas mengenai perangkat keras apa saja yang dibutuhkan untuk membangun system. Linkungan implementasi perangkat keras dari system yang akan dibangun secara lebih lengkap dijelaskan pada Tabel 4.1 dibawah ini

Tabel 4.1 Lingkungan Implementasi Perangkat Keras

Perangkat	Detail Perangkat
Perangkat Komputer	Model : <ul style="list-style-type: none">• Lenovo Y410P
	Manufaktur :

	<ul style="list-style-type: none"> • Lenovo
	Processor : <ul style="list-style-type: none"> • Intel® Core™ i7-4700MQ (2.40GHz 1600MHz 6MB)
	Memori : <ul style="list-style-type: none"> • 8GB PC3-12800 DDR3L SDRAM 1600 MH
Perangkat Mikrokontroler	Mikrokontroler : <ul style="list-style-type: none"> • ATmega2560
	Model : <ul style="list-style-type: none"> • Arduino Mega 2560 (a) • Ardino UNO (b)
	Tegangan : <ul style="list-style-type: none"> • 5 V (a) • 5 V (b)
	Memori Flash : <ul style="list-style-type: none"> • 256 KB (8KB digunakan untuk bootloader) (a) • 32 KB (0.5KB digunakan untuk bootloader) (b)
	SRAM : <ul style="list-style-type: none"> • 8 KB (a) • 2 KB (b)
Perangkat XBee Shield	Model: <ul style="list-style-type: none"> • XBee Shield V0.3 (a) • XBee Shield V1.1 (b)
	Manufaktur : <ul style="list-style-type: none"> • Funduino (a) • ITead Studio (b)

	Tipe jumper : <ul style="list-style-type: none"> • Switch (a) • Pasang lepas (b)
Perangkat Modul XBee	Model : <ul style="list-style-type: none"> • XBee S2
	Manufaktur : <ul style="list-style-type: none"> • Digi International Inc.

4.1.2 Lingkungan Implementasi Perangkat Lunak

Pada bagian ini akan dibahas mengenai perangkat lunak apa saja yang dibutuhkan untuk membangun sistem. Lingkungan implementasi perangkat lunak dari sistem yang akan dibangun secara lebih detail akan dijelaskan pada Tabel 4.2 dibawah ini.

Tabel 4.2 Lingkungan Implementasi Perangkat Lunak

Perangkat	Detail Perangkat
Perangkat Lunak	Sistem Operasi : <ul style="list-style-type: none"> • Microsoft Windows 10 Pro 64-bit
	Software Arduino : <ul style="list-style-type: none"> • Arduino IDE 1.8.5
	Software XBee : <ul style="list-style-type: none"> • DIGI XCTU 6.3.13

4.2 Implementasi Perangkat Keras

Implementasi perangkat keras untuk penelitian tugas akhir ini adalah sebagai berikut :

- 2 (dua) buah Arduino Mega
- 1 (satu) buah Arduino UNO

- 3 (tiga) buah XBee Shield
- 3 (tiga) buah XBee S2
- 1 (satu) buah MicroSD Card Adapter
- 1 (satu) buah SD Card
- 3 (tiga) buah baterai 9 volt

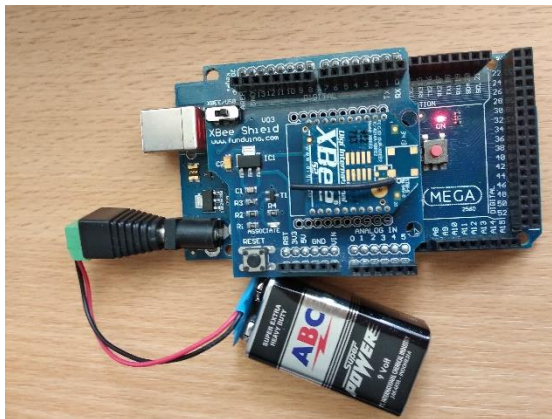
Pada node End Device, terdapat sdcard yang menampung data yang akan dikompresi. Setelah data selesai dikompresi, data kemudian dikirimkan ke node ZigBee Router beserta konfigurasi yang telah digunakan. Kemudian node ZigBee Router meneruskan data yang diterima ke node ZigBee Coordinator untuk dilakukan dekompresi berdasarkan konfigurasi yang ditetapkan. Perancangan serta implementasi perangkat node ZigBee Coordinator, node ZigBee Router dan node ZigBee End Device akan dijabarkan lebih mendetail pada Gambar 4.2, Gambar 4.4 dan Gambar 4.6

4.2.1 Perangkat Node ZigBee Coordinator

Perangkat node ZigBee Coordinator tersusun dari Arduino Mega, XBee Shield, dan modul XBee S2. Zigbee Coordinator memiliki peranan dalam menerima data dari ZigBee Router dan akan mendekompresi data yang telah diterima. Pada gambar 4.1 memperlihatkan perancangan dari node ZigBee Coordinator, sedangkan pada Gambar 4.2 merupakan implementasi dari node ZigBee Coordinator.



Gambar 4.1 Perancangan Node ZigBee Coordinator



Gambar 4.2 Implementasi Node ZigBee Coordinator

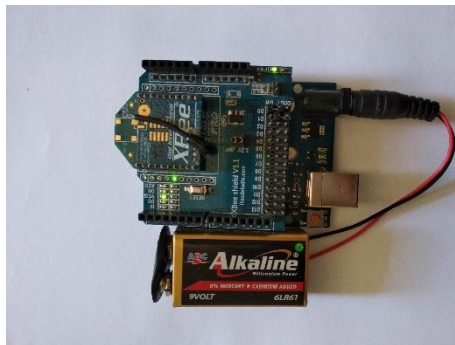
4.2.2 Perangkat Node ZigBee Router

Perangkat node ZigBee Router tersusun dari Arduino UNO, XBee Shield, dan modul XBee S2 dan MicroSD Card Adapter. ZigBee Router memiliki peranan sebagai penghubung antara node ZigBee Router dengan node ZigBee End Device. Pada Gambar 4.3 memperlihatkan perancangan dari node ZigBee Router

dan Gambar 4.4 merupakan implementasi dari node ZigBee Router.



Gambar 4.3 Perancangan Node ZigBee Router



Gambar 4.4 Implementasi Node ZigBee Router

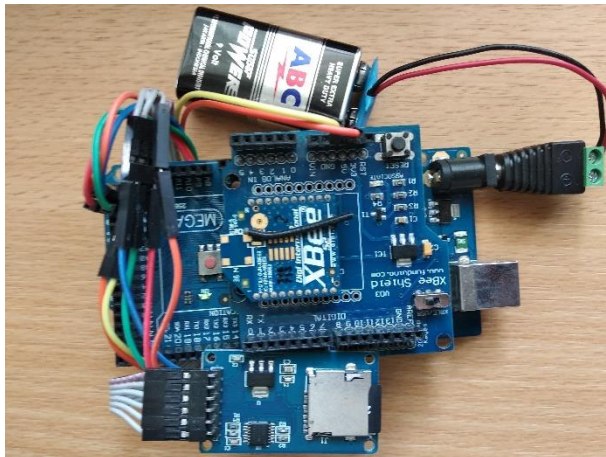
4.2.3 Perangkat Node ZigBee End Device

Perangkat node ZigBee Router tersusun dari Arduino Mega, XBee Shield, dan modul XBee S2 dan MicroSD Card Adapter. ZigBee End Device memiliki peranan dalam mengambil

data dari *sd card* melakukan kompresi data dan mengirimkan data tersebut menuju node ZigBee Router. Pada Gambar 4.5 memperlihatkan perancangan dari node ZigBee End Device dan Gambar 4.6 merupakan implementasi dari node ZigBee End Device.



Gambar 4.5 Perancangan Node ZigBee End Device



Gambar 4.6 Implementasi Node ZigBee End Device

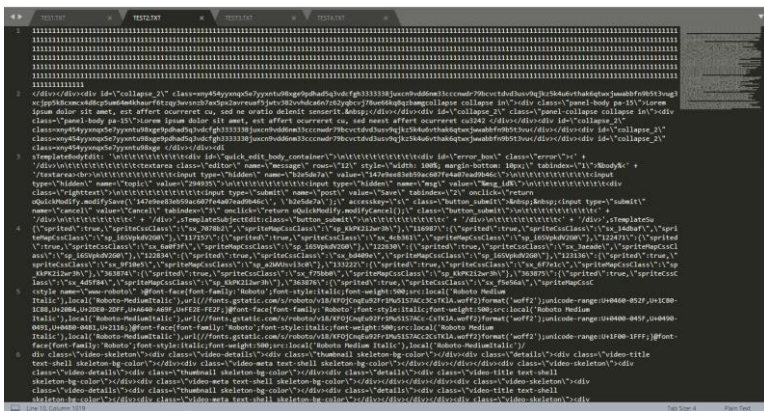
4.3 Implementasi Inisialisasi Data Pada SD Card

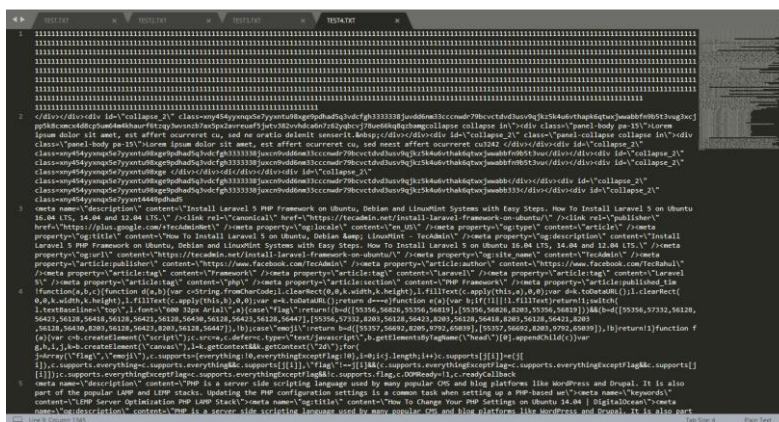
Data yang akan dikompresi berupa string diletakkan pada SD Card. Sebelum melakukan kompresi data, terlebih dahulu kita harus menginisialisasi data. Pada penelitian ini terdapat lima jenis data yang akan dikompresi, dimana masing – masing data memiliki panjang yang berbeda. Data tersebut akan disimpan dalam bentuk file txt. Berikut ini adalah kelima data tersebut :

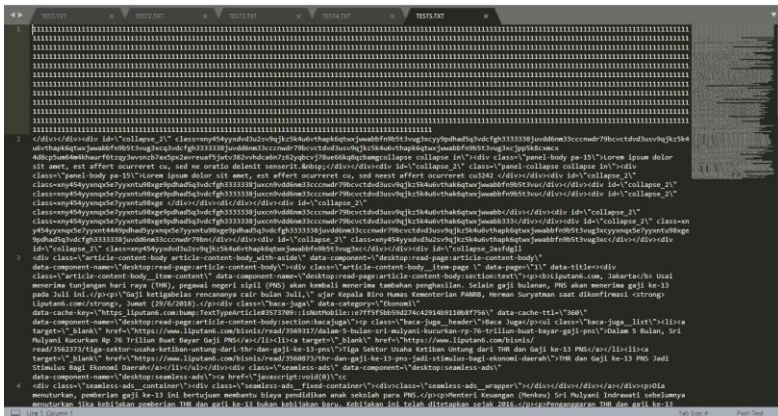
- Data string dengan panjang 584 karakter
- Data string dengan panjang 980 karakter
- Data string dengan panjang 1280 karakter
- Data string dengan panjang 1345 karakter
- Data string dengan panjang 1870 karakter

Setiap file menyimpan data dengan panjang yang sama. Sebagai contohnya file `data_584.txt` menyimpan data dengan panjang 584 karakter, dimana setiap baris data yang ada pada file tersebut panjangnya sama yaitu 584 karakter.

Pada Gambar 4.5 merupakan contoh file txt dengan panjang string 584 karakter per baris. Pada Gambar 4.6 merupakan contoh file txt dengan panjang string 980 karakter per baris. Pada Gambar 4.7 merupakan contoh file txt dengan panjang string 1280 karakter per baris. Pada Gambar 4.8 merupakan contoh file txt dengan panjang string 1345 karakter per baris. Pada Gambar 4.9 merupakan contoh file txt dengan panjang string 1870 karakter per baris.







Gambar 4.11 Data dengan panjang 1870 karakter per baris

4.4 Implementasi Membaca Data dari SD Card

Setelah data dibagi kedalam masing-masing file, maka data akan siap di kompresi. Pada node Router data akan diambil dari *sd card* per baris. Kemudian data tersebut di akan di ubah kedalam array lalu siap untuk kompresi dengan konfigurasi yang telah ditentukan.

```

1 function readData()
2   myFile <- load DataFile
3   while(myFile.available()) do
4     data <- myFile.readByLine
5     data.toArray(data_test, data_sz)
6   endwhile
7 endfunction

```

Kode Sumber 4.1 Pseudocode membaca data dari sd card

4.5 Implementasi Setting Konfigurasi Encoder / Decoder

Implementasi setting konfigurasi encoder / decoder bertujuan untuk melakukan setting konfigurasi yang akan

digunakan encoder dalam proses kompresi data. Adapun konfigurasi yang akan digunakan yaitu sebagai berikut :

- window_sz2 menggunakan konfigurasi antara 4 sampai 8
- lookahead_sz2 menggunakan konfigurasi nilai 3 sampai window_sz2 - 1
- input_buffer_size yang digunakan adalah 64

1	function setConfiguration()
2	Configuration cfg := new Configuration()
3	cfg.log_level <- value
4	cfg.window_sz <- value
5	cfg.lookahead_sz <- value
6	cfg.decoder_input_buffer_sz <- value
7	endfunction

Kode Sumber 4.2 Pseudocode setting konfigurasi encoder / decoder

4.6 Implementasi Kompresi Data

Implementasi kompresi data yang dilakukan pada penelitian ini menggunakan library Heatshrink yang merupakan salah satu algoritma kompresi data yang berbasis pada algoritma Lempel-Ziv-Storer-Szymanski (LZSS).

1	function compressData(input, input_sz, cfg)
2	heatshrink_encoder hse <-
.	heatshrink_encoder_alloc(cfg)
3	comp <- malloc
4	while(sunk < input_sz) do
5	HSE_sink_res esres <-
.	heatshrink_encoder_sink(hse,
.	input[sunk], input_sz - sunk, count)
6	sunk <- sunk + count
7	if(sunk == input_sz) then
.	heatshrink_encoder_finish(hse)
8	endif
9	HSE_poll_res pres
10	do


```

11         pres <- heatshrink_encoder_poll(hse,
.         comp[polled], comp_sz - polled, count)
.         polled <- polled + count
12         while (pres == HSER_POLL_MORE)
13             if(sunk == input_sz) then
14                 heatshrink_encoder_finish(hse)
15             endif
16         endwhile
17         free(comp)
18         heatshrink_encoder_free(hse)
19         return polled
20     endfunction

```

Kode Sumber 4.3 *Pseudocode* kompresi data

4.7 Implementasi Mekanisme Pengiriman Data

Pada penelitian ini, pengiriman data yang dilakukan oleh node ZigBee Router menggunakan mode Router AT. Proses pengiriman data memanfaatkan kelas Serial yang ada pada Arduino. Data hasil kompresi yang akan dikirimkan akan dipecah menjadi beberapa bagian kecil. Dalam satu kali pengiriman jumlah data yang dapat di kirimkan adalah sebanyak 16 buah, dimana terdapat delay 3 detik untuk setiap pengiriman data. Setelah data kompresi dikirimkan semua, konfigurasi encoder / decoder yang digunakan juga akan dikirimkan ke node ZigBee Coordinator.

```

1  function sendData(comp, cfg, polled, length_data)
2      Serial(comp[0])
3      for i=1 to polled do
4          if(i mod 16 == 0) then
5              Serial(comp[i])
6              Serial("\n")
7          else
8              Serial(comp[i])
9              Serial("\n")
10         endif
11     endfor
12     delay()

```


13	Serial(length_data)
14	Serial(cfg.window_sz)
15	Serial(cfg.lookahead_sz)
16	Serial(cfg.decoder_input_buffer_sz)
17	Serial(polled)
18	endfunction

Kode Sumber 4.4 *Pseudocode* mekanisme pengiriman data

4.8 Implementasi Dekompresi Data

Potongan – potongan data yang dikirimkan oleh node ZigBee Router akan disatukan kembali pada Node ZigBee Coordinator. Setelah data hasil kompresi utuh, maka proses dekompresi data akan siap dilakukan dengan konfigurasi encoder / decoder yang dikirimkan juga oleh node ZigBee Router.

1	function reformingData()
2	if(Serial.available() > 0) then
3	incomingByte <- Serial.read()
4	if(incomingByte != "\n") then
5	stringData <- stringData + incomingByte
6	elseif(incomingByte == "\n")) then
7	num[idx] <- stringData.toInt()
8	idx++
9	endif
10	endif
11	endfunction

Kode Sumber 4.5 *Pseudocode* pembentukan ulang data

1	function decompressData(input, input_sz, cfg,
	output, output_sz, polled)
2	heatshrink_decoder hsd <-
3	heatshrink_decoder_alloc(cfg)
4	comp_sz <- polled
5	while(sunk < comp_sz) do
6	heatshrink_decoder_sink(hsd,
	input[sunk], comp_sz - sunk, count)
	sunk <- sunk + count

```

7      if(sunk == comp_sz) then
8          heatshrink_decoder_finish(hsd)
9      endif
10     HSE_poll_res pres
11     do
12         pres <- heatshrink_decoder_poll(hsd,
13         output[polled], output_sz - polled,
14         . count)
15         . polled <- polled + count
16     while (pres == HSDR_POLL_MORE)
17     if(sunk == comp_sz) then
18         heatshrink_dencoder_finish(hsd)
19     endif
20     endwhile
21 endfunction

```

Kode Sumber 4.6 *Pseudocode* dekompresi data

(Halaman ini sengaja dikosongkan)

BAB V

HASIL UJI COBA DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada Tugas Akhir yang telah dikerjakan. Uji coba yang akan dilakukan secara garis besar terdiri dari uji coba fungsionalitas dan uji coba performa. Mekanisme uji coba dilakukan dengan menjalankan serangkaian scenario yang telah ditentukan. Pengujian fungsionalitas meliputi uji coba setiap bagian perangkat keras yang dirangkai pada Arduino dan uji coba keseluruhan sistem. Sedangkan pengujian performa meliputi efektifitas hasil kompresi, waktu yang dibutuhkan untuk kompresi dan dekompresi serta energi yang dapat dihemat oleh system. Bagian akhir dari bab ini akan membahas mengenai evaluasi dari serangkaian uji coba yang telah dilakukan.

5.1 Lingkungan Uji Coba

Lingkungan pelaksanaan uji coba meliputi perangkat keras dan perangkat lunak yang akan digunakan pada sistem ini. Spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam rangka uji coba perangkat lunak ini dicantumkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Lingkungan Uji Coba

Perangkat Keras	
Node ZigBee Coordinator	Rangkaian dari :
	- Arduino Mega 2560
	- XBee Shield
	- XBee Modul S2
	- Baterai 9 Volt
Node ZigBee Router	Rangkaian dari :
	- Arduino UNO

	- XBee Shield
	- XBee Modul S2
	- Baterai 9 Volt
Node ZigBee End Device	Rangkaian dari :
	- Arduino Mega 2560
	- XBee Shield
	- XBee Modul S2
	- MicroSD Card Adapter
	- Baterai 9 Volt
	- SD Card
Perangkat Lunak	
Sistem Operasi	Microsoft Windows 10 Pro 64-bit
Software Arduino	Arduino IDE 1.8.5
Software XBee	DIGI XCTU 6.3.13

Lokasi yang dijadikan tempat untuk uji coba ada dua. Lokasi pertama seperti Gambar 5.1 berada di Laboratorium KBJ Departemen Informatika ITS



Gambar 5.1 Lokasi Pertama

Lokasi kedua yang dijadikan tempat untuk uji coba seperti Gambar 5.2 berada di Lapangan Departemen Informatika ITS



Gambar 5.2 Lokasi Kedua

5.2 Data Pengujian

Subbab ini menjelaskan mengenai data yang digunakan pada uji coba. Seperti yang telah dijelaskan sebelumnya, data yang akan digunakan untuk melakukan pengujian harus di buat terlebih dahulu. Data yang digunakan untuk mengujian berupa data string dimana data akan terbagi menjadi lima jenis data yang berbeda berdasarkan panjangnya. Berikut ini adalah kelima data tersebut :

- Data string dengan panjang 584 karakter
- Data string dengan panjang 980 karakter
- Data string dengan panjang 1280 karakter
- Data string dengan panjang 1345 karakter
- Data string dengan panjang 1870 karakter

5.3 Skenario Uji Coba Fungsionalitas

Pengujian ini dilakukan untuk menguji apakah fungsionalitas yang diidentifikasi benar-benar diimplementasikan dan bekerja sebagaimana seharusnya. Pengujian juga dilakukan untuk mengetahui kesesuaian setiap tahapan atau langkah penggunaan fitur terhadap skenario yang dipersiapkan.

5.3.1 Skenario Uji Coba Membaca Data dari SD Card

Pada skenario uji coba ini, node ZigBee End Device akan membaca data yang ada pada sd card. Data tersebut dibaca per baris dengan jumlah total 80 baris, kemudian data akan ditampung kedalam sebuah array dan siap untuk di kompresi.

Tabel 5.2 Skenario Uji Coba Membaca Data dari SD Card

ID	UJ – F01
Nama	Uji Coba Membaca Data dari SD Card

Tujuan Uji Coba	Menguji fungsionalitas system untuk membaca data dari sd card
Kondisi Awal	Node ZigBee End Device diaktifkan
Skenario	<ol style="list-style-type: none"> 1. Node ZigBee End Device membaca data dari <i>sdcard</i> 2. Data dibaca per baris 3. Kegiatan ini akan terus berulang hingga data pada baris terakhir. 4. Mengulangi langkah 1 – 3 dengan data yang panjang karakternya 980 5. Mengulangi langkah 1 – 3 dengan data yang panjang karakternya 1280 6. Mengulangi langkah 1 – 3 dengan data yang panjang karakternya 1345 7. Mengulangi langkah 1 – 3 dengan data yang panjang karakternya 1870
Masukan	Data yang ada pada sd card
Keluaran	Menampilkan data dibaca pada serial monitor
Hasil yang Diharapkan	Data dapat terbaca dengan baik

5.3.2 Skenario Uji Coba Kompresi Data

Pada skenario uji coba ini, data yang telah di baca akan mengalami proses kompresi. Kompresi data akan dilakukan dengan berbagai jenis konfigurasi encoder / decoder.

Tabel 5.3 Skenario Uji Coba Kompresi Data

ID	UJ – F02
Nama	Uji Coba Kompresi Data

Tujuan Uji Coba	Menguji fungsionalitas system untuk melakukan kompresi dengan berbagai konfigurasi dan berbagai data yang memiliki panjang berbeda - beda
Kondisi Awal	Node ZigBee End Device diaktifkan
Skenario	<ol style="list-style-type: none"> 1. Node ZigBee End Device membaca data dengan panjang 584 karakter 2. Node ZigBee melakukan kompresi data dengan konfigurasi HS(4,3) 3. Node ZigBee melakukan kompresi data dengan konfigurasi HS(5,4) 4. Node ZigBee melakukan kompresi data dengan konfigurasi HS(6,5) 5. Node ZigBee melakukan kompresi data dengan konfigurasi HS(7,6) 6. Node ZigBee melakukan kompresi data dengan konfigurasi HS(8,7) 7. Node ZigBee melakukan kompresi data dengan konfigurasi HS(9,8) 8. Mengulangi langkah 1 – 7 dengan data uji coba yang memiliki panjang 980 karakter 9. Mengulangi langkah 1 – 7 dengan data uji coba yang memiliki panjang 1280 karakter 10. Mengulangi langkah 1 – 7 dengan data uji coba yang memiliki panjang 1345 karakter 11. Mengulangi langkah 1 – 7 dengan data uji coba yang memiliki panjang 1870 karakter
Masukan	Data string yang akan dikompresi
Keluaran	Data hasil kompresi

Hasil yang Diharapkan	Node ZigBee End Device berhasil mengompresi data
-----------------------	--

5.3.3 Skenario Uji Coba Komunikasi Pada Topologi Cluster Tree

Pada skenario uji coba ini, node ZigBee Coordinator akan terkoneksi dengan node ZigBee Router dan node ZigBee Router terkoneksi dengan node ZigBee End Device. Uji coba ini bertujuan untuk mengetahui apakah data yang dikirimkan dari node ZigBee End Device diterima oleh node ZigBee Coordinator. Data yang akan dikirimkan berupa string dan hasilnya dapat dilihat melalui Serial monitor.

Tabel 5.4 Skenario Uji Coba Komunikasi Pada Topologi Cluster Tree

ID	UJ – F03
Nama	Uji Coba Komunikasi Pada Topologi Cluster Tree.
Tujuan Uji Coba	Menguji fungsionalitas node ZigBee Coordinator untuk dapat terhubung dengan node End Device.
Kondisi Awal	Ketiga node ZigBee diaktifkan. Ketiga node berada dalam mode XBee. Data yang akan dikirimkan sudah disiapkan.
Skenario	<ol style="list-style-type: none"> 1. Node ZigBee End Device mengirimkan data hasil kompresi ke node ZigBee Router 2. Node ZigBee Router menerima data dari node End Device 3. Melakukan Pemantauan data yang diterima melalui <i>serial monitor</i>

	<ol style="list-style-type: none"> Node ZigBee Router mengirimkan data hasil kompresi ke node ZigBee Coordinator Melakukan Pemantauan data yang diterima melalui <i>serial monitor</i> node Coordinator
Masukan	Data hasil kompresi
Keluaran	Tampilan data yang diterima pada serial monitor
Hasil yang Diharapkan	Node ZigBee Coordinator dapat menerima data yang dikirimkan oleh node ZigBee End Device

5.3.4 Skenario Uji Coba Dekompresi Data

Pada skenario uji coba ini, potongan – potongan data hasil kompresi yang diterima oleh node ZigBee Coordinator akan di bentuk kembali sampai utuh. Kemudian data di dekompresi sesuai konfigurasi yang diterima.

Tabel 5.5 Skenario Uji Coba Dekompresi Data

ID	UJ – F04
Nama	Uji Coba Dekompresi Data
Tujuan Uji Coba	Menguji fungsionalitas sistem untuk melakukan dekompresi data dari potongan – potongan data yang diterima oleh node ZigBee Coordinator
Kondisi Awal	Ketiga node ZigBee diaktifkan. Ketiga node berada dalam mode XBee.
Skenario	<ol style="list-style-type: none"> Node ZigBee Coordinator menerima data hasil kompresi

	2. Node ZigBee Coordinator menyusun potongan – potongan data yang diterima 3. Node ZigBee Coordinator melakukan proses dekompresi 4. Melakukan pemantauan hasil dekompresi melalui <i>serial monitor</i>
Masukan	Data hasil kompresi
Keluaran	Tampilan data yang hasil dekompresi data pada serial monitor node ZigBee Coordinator
Hasil yang Diharapkan	Node ZigBee Coordinator dapat melakukan penyusunan potongan – potongan data kemudian di lakukan proses dekompresi

5.3.5 Skenario Uji Coba Kompresi Adaptive

Pada skenario uji coba ini, akan dilakukan kompresi data dengan settingan konfigurasi `window_sz` dan `lookahead_sz` sesuai dengan scenario yang digunakan. Konfigurasi `window_sz` dan `lookahead_sz` bersifat adaptive tergantung panjang data yang akan dikompresi. Jika panjang data **kurang dari 980** maka konfigurasi yang digunakan adalah **`window_sz = 9` dan `lookahead_sz = 8`**. Jika panjang datanya **antara 980 sampai 1345** maka konfigurasi yang digunakan adalah **`window_sz = 8` dan `lookahead_sz = 7`**. Jika panjang datanya **antara 1346 sampai 1870** maka konfigurasi yang digunakan adalah **`window_sz = 7` dan `lookahead_sz = 6`**.

Tabel 5.6 Skenario Uji Coba Kompresi Adaptive

ID	UJ – F05
Nama	Uji Coba Kompresi Adaptive
Tujuan Uji Coba	Menguji fungsionalitas sistem untuk melakukan kompresi adaptive berdasarkan panjang data

Kondisi Awal	Node ZigBee End Device diaktifkan
Skenario	<ol style="list-style-type: none"> 1. Node ZigBee End Device membaca data satu baris pada sdcard 2. Node ZigBee End Device melakukan kompresi data dengan konfigurasi window_sz dan lookahead_sz yang telah ditentukan sebelumnya 3. Mengulangi langkah 1 dan 2 sampai data terakhir pada sd card
Masukan	Data string yang akan dikompresi
Keluaran	Tampilan hasil kompresi data pada serial monitor node ZigBee End Device
Hasil yang Diharapkan	Node ZigBee End Device dapat melakukan kompresi adaptive

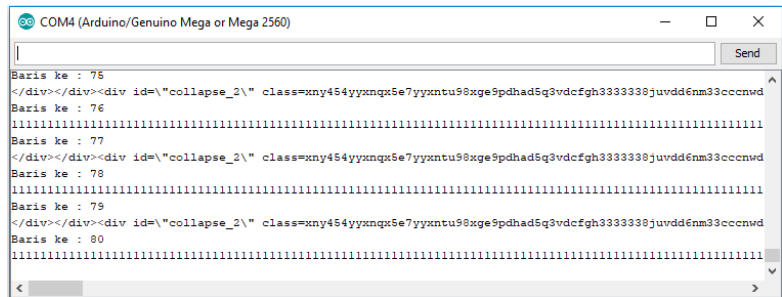
5.4 Hasil Uji Coba Fungsionalitas

Telah dijabarkan pada bagian sebelumnya mengenai skenario dari keseluruhan uji coba fungsionalitas. Skenario uji coba yang telah dijabarkan terdiri dari beberapa hal. Secara garis besar, pengujian fungsionalitas terdiri dari empat bagian, yaitu fungsionalitas membaca data, komunikasi node, kompresi data dan dekompresi data.

5.4.1 Hasil Uji Coba (UJ-F01) – Membaca Data dari SD Card

Pengujian dilakukan berdasarkan skenario uji coba pada bab 5.3.1. Langkah pertama yang dilakukan node ZigBee Router membaca data yang ada pada *sdcard*, data dibaca baris per baris dengan jumlah total baris sebanyak 80 baris. Pada pengujian yang dilakukan data yang dibaca akan di tampilkan pada serial monitor

Gambar 5.4 Hasil Uji Coba UJ-F01 pada panjang karakter 980



5.4.2 Hasil Uji Coba (UJ-F02) – Kompresi Data

Uji coba dilakukan berdasarkan skenario uji coba pada bab 5.3.2. Pada skenario dilakukan kompresi data berdasarkan konfigurasi tertentu. Berikut ini adalah hasil uji coba yang di peroleh.

Tabel 5.7 Hasil Uji Coba UJ – F02 Kompresi Data

Kompresi	Panjang Data	HS(4, 3)	HS(5, 4)	HS(6, 5)	HS(7, 6)	HS(8, 7)	HS(9, 8)
	584	berhasil	berhasil	berhasil	berhasil	berhasil	berhasil
	980	berhasil	berhasil	berhasil	berhasil	berhasil	gagal
	1280	berhasil	berhasil	berhasil	berhasil	berhasil	gagal
	1435	berhasil	berhasil	berhasil	berhasil	berhasil	gagal
	1870	berhasil	berhasil	berhasil	berhasil	gagal	gagal

Menurut hasil uji coba pada table diatas, sebagian besar proses konfigurasi menggunakan settingan konfigurasi **HS(9,8) gagal**, kecuali pada data dengan panjang 584. Kegagalan tersebut dikarenakan memori yang tidak mencukupi, sehingga tidak dapat melakukan proses kompresi data. Selain itu pada settingan HS(8,7) proses kompresi tidak dapat berjalan pada data dengan panjang 1870 karakter.

5.4.3 Hasil Uji Coba (UJ-F03) – Komunikasi Pada Topologi Cluster Tree

Uji coba yang dilakukan pada bab 5.3.3 bertujuan untuk mengetahui fungsionalitas dari komunikasi pada topologi *cluster tree* yang digunakan. Berikut ini adalah hasil uji coba yang diperoleh berdasarkan skenario yang dijalankan.

Tabel 5.8 Hasil Uji Coba UJ – F03 Komunikasi Pada Topologi Cluster Tree

Sumber	Tujuan	Hasil
Node ZigBee End Device	Node ZigBee Router	Berhasil
Node ZigBee Router	Node ZigBee Coordinator	Berhasil

Menurut hasil uji coba yang ada pada tabel di atas node ZigBee Coordinator berhasil menerima data yang dikirimkan oleh node ZigBee End Device.

5.4.4 Hasil Uji Coba (UJ-F04) – Dekompresi Data

Uji coba dilakukan berdasarkan skenario uji coba pada bab 5.3.4. Pada skenario dilakukan kompresi data berdasarkan konfigurasi tertentu. Berikut ini adalah hasil uji coba yang diperoleh.

Tabel 5.9 Hasil Uji Coba UJ – F04 Dekompresi Data

Dekompresi	Panjang Data	HS(4, 3)	HS(5, 4)	HS(6, 5)	HS(7, 6)	HS(8, 7)	HS(9, 8)
	584	berhasil	berhasil	berhasil	berhasil	berhasil	berhasil
	980	berhasil	berhasil	berhasil	berhasil	berhasil	gagal
	1280	berhasil	berhasil	berhasil	berhasil	berhasil	gagal

	1435	berha sil	berha sil	berha sil	berha sil	berha sil	gagal
	1870	berha sil	berha sil	berha sil	berha sil	gagal	gagal

Menurut hasil uji coba pada table diatas, sebagian besar proses konfigurasi menggunakan settingan konfigurasi **HS(9,8) gagal**, kecuali pada data dengan panjang 584. Kegagalan tersebut dikarenakan memori yang tidak mencukupi, sehingga tidak dapat melakukan proses kompresi data. Selain itu pada settingan HS(8,7) proses kompresi tidak dapat berjalan pada data dengan penjang 1640.

5.4.5 Hasil Uji Coba (UJ-F05) – Kompresi Adaptive

Uji coba dilakukan berdasarkan skenario uji coba pada bab 5.3.5. Pada skenario dilakukan kompresi adaptive berdasarkan konfigurasi yang telah ditetapkan. Skenario uji coba menggunakan 10 buah data string dengan panjang data yang masing-masing berbeda. Hasil pengujian akan ditampilkan pada serial monitor beserta konfigurasi encoder/decoder yang digunakan. Berikut ini adalah hasil uji coba yang diperoleh.

```

in: 590 compressed: 430
Compressed data: 189, 218, 232, 180, 153, 212, 158, 237, 56, 154, 93, 106, 243, 89, 148, 214, 141, 77, 16
Konfigurasi : HS(9,8)

in: 616 compressed: 333
Compressed data: 158, 91, 45, 54, 235, 92, 130, 233, 121, 184, 89, 103, 181, 201, 21, 210, 203, 120, 186,
Konfigurasi : HS(9,8)

in: 1107 compressed: 271
Compressed data: 189, 215, 36, 87, 59, 133, 202, 211, 116, 178, 217, 43, 146, 41, 213, 210, 229, 117, 178
Konfigurasi : HS(8,7)

in: 982 compressed: 710
Compressed data: 158, 92, 238, 151, 155, 101, 150, 65, 110, 176, 219, 108, 179, 218, 228, 138, 239, 119,
Konfigurasi : HS(8,7)

in: 1304 compressed: 772
Compressed data: 144, 217, 174, 182, 235, 29, 210, 211, 111, 183, 74, 44, 50, 203, 20, 178, 199, 41, 189,
Konfigurasi : HS(8,7)

```

Gambar 5.8 Hasil Uji Coba UJ-F05 Kompresi Adaptive

```
in: 1319 compressed: 1006
Compressed data: 158, 91, 108, 183, 75, 12, 130, 221, 97, 182, 217, 103, 181, 201, 21, 146, 203, 115, 177
Konfigurasi : HS(8,7)

in: 1345 compressed: 253
Compressed data: 187, 88, 110, 82, 11, 37, 190, 219, 97, 180, 219, 168, 118, 251, 173, 186, 233, 114, 188
Konfigurasi : HS(8,7)

in: 1345 compressed: 249
Compressed data: 178, 91, 237, 182, 27, 77, 186, 135, 111, 186, 219, 174, 151, 43, 205, 14, 223, 100, 178
Konfigurasi : HS(8,7)

in: 1702 compressed: 1310
Compressed data: 158, 92, 44, 55, 75, 68, 130, 201, 61, 174, 72, 169, 179, 25, 164, 186, 107, 48, 155, 20
Konfigurasi : HS(7,6)

in: 1391 compressed: 885
Compressed data: 158, 92, 238, 214, 121, 5, 142, 217, 97, 185, 220, 231, 181, 201, 21, 190, 199, 116, 180
Konfigurasi : HS(7,6)
```

Gambar 5.9 Hasil Uji Coba UJ-F05 Kompresi Adaptive

5.5 Skenario Uji Coba Performa

Uji coba performa sytem dilakukan untuk mengetahui performa dari system yang dibangun berdasarkan skenario yang akan dibuat. Uji coba performa sistem yang akan dilakukan meliputi efektifitas hasil kompresi dalam, akurasi pengiriman, dan waktu yang dibutuhkan untuk proses kompresi serta dekompresi.

5.5.1 Skenario Uji Coba Efektifitas Kompresi

Uji coba efektifitas kompresi bertujuan untuk mengetahui efektifitas kompresi data dari berbagai konfigurasi encoder / decoder yang digunakan. Terdapat 10 buah data string beragam dengan panjang data yang sama yang akan digunakan untuk uji coba.

Tabel 5.10 Uji Coba Efektifitas Kompresi

ID	UJ – P01
Nama	Uji Coba Efektifitas Kompresi
Tujuan Uji Coba	Menguji performa sistem untuk mengetahui efektifitas kompresi yang didapatkan dari dari berbagai konfigurasi encoder / decoder yang

	digunakan dan berbagai panjang data yang digunakan.
Kondisi Awal	Node ZigBee End Device diaktifkan
Skenario	<ol style="list-style-type: none"> 1. Node ZigBee End Device membaca data pada <i>sdc card</i> dengan panjang 584 karakter 2. Node ZigBee melakukan kompresi data dengan konfigurasi HS(4,3) 3. Node ZigBee melakukan kompresi data dengan konfigurasi HS(5,4) 4. Node ZigBee melakukan kompresi data dengan konfigurasi HS(6,5) 5. Node ZigBee melakukan kompresi data dengan konfigurasi HS(7,6) 6. Node ZigBee melakukan kompresi data dengan konfigurasi HS(8,7) 7. Node ZigBee melakukan kompresi data dengan konfigurasi HS(9,8) 8. Mengulangi pengujian dengan 10 data string yang berbeda 9. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 980 karakter 10. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 1280 karakter 11. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 1345 karakter

	12. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 1870 karakter
Masukan	Data raw yang akan dikompresi
Keluaran	Tampilan jumlah data hasil kompresi data pada serial monitor node ZigBee End Device
Hasil yang Diharapkan	Tercatatnya efektifitas kompresi pada setiap skenario uji coba

5.5.2 Skenario Uji Coba Akurasi Pengiriman Data ZigBee

Uji coba akurasi pengiriman data, bertujuan untuk mengetahui seberapa besar akurasi system dalam mengirimkan paket data melalui ZigBee. Uji coba dilakukan pada masing - masing jarak ± 10 meter, ± 20 meter dan ± 30 meter. Percobaan diulangi sebanyak sepuluh kali. Berikut ini menjelaskan skenario uji cobanya.

Tabel 5.11 Uji Coba Akurasi Pengiriman Data ZigBee

ID	UJ – P02
Nama	Uji Coba Akurasi Pengiriman Data ZigBee
Tujuan Uji Coba	Menguji performa sistem untuk mengetahui akurasi pengiriman data menggunakan ZigBee dari jarak yang berbeda – beda (± 10 meter, ± 20 meter, ± 30 meter)
Kondisi Awal	Node ZigBee Router dan node ZigBee Coordinator diaktifkan. Keduanya berada dalam mode XBee.
Skenario	1. Menyalakan Node ZigBee Router dan Node ZigBee Coordinator dalam mode XBee selama kurang lebih 2 menit pada jarak ± 10 meter

	<ol style="list-style-type: none"> 2. Mengulangi pengujian hingga 10 kali 3. Memantau hasil pada serial motor node ZigBee Coordinator 4. Mengulangi pengujian pada jarak ± 20 meter 5. Mengulangi pengujian pada jarak ± 30 meter
Masukan	Data yang akan dikirim
Keluaran	Akurasi pengiriman data pada jarak berbeda - beda
Hasil yang Diharapkan	Mendapatkan akurasi pengiriman data menggunakan ZigBee dari jarak berbeda – beda dan membandingkan hasilnya.

5.5.3 Skenario Uji Waktu Kompresi

Uji coba waktu kompresi, bertujuan untuk mengetahui waktu yang dibutuhkan untuk proses kompresi data. Terdapat 10 buah data string beragam dengan panjang data yang sama yang akan digunakan untuk uji coba.

Tabel 5.12 Uji Coba Waktu Kompresi

ID	UJ – P03
Nama	Uji Coba Waktu Kompresi
Tujuan Uji Coba	Menguji performa sistem untuk mengetahui waktu yang dibutuhkan untuk proses kompresi
Kondisi Awal	Data raw telah di siapkan pada file tersendiri. Node ZigBee End Device diaktifkan.
Skenario	<ol style="list-style-type: none"> 1. Node ZigBee End Device membaca data pada <i>sdcard</i> dengan panjang 584 karakter

	<ol style="list-style-type: none"> 2. Node ZigBee melakukan kompresi data dengan konfigurasi HS(4,3) 3. Node ZigBee melakukan kompresi data dengan konfigurasi HS(5,4) 4. Node ZigBee melakukan kompresi data dengan konfigurasi HS(6,5) 5. Node ZigBee melakukan kompresi data dengan konfigurasi HS(7,6) 6. Node ZigBee melakukan kompresi data dengan konfigurasi HS(8,7) 7. Node ZigBee melakukan kompresi data dengan konfigurasi HS(9,8) 8. Mengulangi pengujian dengan 10 data string yang berbeda 9. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 980 karakter 10. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 1280 karakter 11. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 1345 karakter 12. Mengulangi langkah 1 – 8 dengan data uji coba yang memiliki panjang 1870 karakter
Masukan	Data raw yang akan dikompresi

Keluaran	Tampilan waktu yang dibutuhkan untuk proses kompresi pada serial monitor node ZigBee End Device
Hasil yang Diharapkan	Tercatatnya waktu yang di butuhkan untuk proses kompresi data pada setiap skenario uji coba

5.5.4 Skenario Uji Waktu Dekompresi

Uji coba waktu dekompresi, bertujuan untuk mengetahui waktu yang dibutuhkan proses dekompresi data. Terdapat 10 buah data string beragam dari hasil kompresi yang telah dilakukan sebelumnya. Data tersebut akan digunakan untuk skenario uji coba.

Tabel 5.13 Uji Coba Waktu Dekompresi

ID	UJ – P04
Nama	Uji Coba Waktu Dekompresi
Tujuan Uji Coba	Menguji performa sistem untuk mengetahui waktu yang dibutuhkan untuk proses dekompresi
Kondisi Awal	Data hasil kompresi telah di siapkan. Node ZigBee End Device diaktifkan.
Skenario	<ol style="list-style-type: none"> 1. Menyiapkan data hasil kompresi dari panjang data 584 karakter sebelumnya 2. Node ZigBee melakukan dekompresi data dengan konfigurasi HS(4,3) 3. Node ZigBee melakukan dekompresi data dengan konfigurasi HS(5,4) 4. Node ZigBee melakukan dekompresi data dengan konfigurasi HS(6,5)

	<ol style="list-style-type: none"> 5. Node ZigBee melakukan dekompresi data dengan konfigurasi HS(7,6) 6. Node ZigBee melakukan dekompresi data dengan konfigurasi HS(8,7) 7. Node ZigBee melakukan kompresi data dengan konfigurasi HS(9,8) 8. Mengulangi pengujian dengan 10 data hasil kompresi sebelumnya 9. Mengulangi langkah 1 – 8 dengan hasil kompresi data dari 980 karakter 10. Mengulangi langkah 1 – 8 dengan hasil kompresi data dari 1280 karakter 11. Mengulangi langkah 1 – 8 dengan hasil kompresi data dari 1345 karakter 12. Mengulangi langkah 1 – 8 dengan hasil kompresi data dari 1870 karakter
Masukan	Data hasil kompresi
Keluaran	Tampilan waktu yang dibutuhkan untuk proses dekompresi pada serial monitor node ZigBee End Device
Hasil yang Diharapkan	Tercatatnya waktu yang di butuhkan untuk proses dekompresi data pada setiap skenario uji coba

5.6 Hasil Uji Coba Performa

Telah dijabarkan pada bagian sebelumnya mengenai skenario dari keseluruhan uji coba performa. Skenario uji coba

yang telah dijabarkan terdiri dari beberapa hal. Secara garis besar, pengujian performa terdiri dari tiga bagian, yaitu performa efektifitas, packet loss, dan waktu.

5.6.1 Hasil Uji Coba (UJ-P01) – Efektifitas Kompresi

Sesuai skenario pada bab 5.5.1, yang di ujikan adalah efektifitas kompresi data dari berbagai konfigurasi encoder / decoder yang digunakan dengan 10 buah data string yang berbeda-beda. Berikut ini adalah hasil uji coba yang diperoleh.

Tabel 5.14 Hasi Uji Coba UJ – P01 Efektifitas Kompresi

Panjang Data	Uji Coba	Konfigurasi					
		HS(4,3)	HS(5,4)	HS(6,5)	HS(7,6)	HS(8,7)	HS(9,8)
584	Efektifitas	9.3%	24.4 %	31.8 %	49.4 %	59.8 %	56.7 %
980	Efektifitas	9.6%	19.9 %	32.3 %	49.1 %	54.5 %	60.5 %
1280	Efektifitas	10.0 %	16.8 %	37.4 %	45.9 %	50.2 %	gagal
1345	Efektifitas	10.2 %	17.0 %	37.7 %	45.9 %	54.5 %	gagal
1870	Efektifitas	5.1%	11.2 %	20.4 %	27.8 %	gagal	gagal

Berdasarkan hasil uji coba efektifitas kompresi pada panjang data **584 karakter**, didapatkan persentase efektifitas kompresi paling kecil ketika menggunakan konfigurasi HS(4,3) sebesar **9.3%**. Sedangkan efektifitas kompresi paling besar adalah **56.7%** pada konfigurasi **HS(9,8)**. Tabel 5.12 menjabarkan hasil uji coba dari efektifitas kompresi pada panjang data **584 karakter** yang telah dijabarkan lebih rinci pada Table 7.1 di lampiran.

Berdasarkan hasil uji coba efektifitas kompresi pada panjang data **980 karakter**, didapatkan persentase efektifitas

kompresi paling kecil ketika menggunakan konfigurasi HS(4,3) sebesar **9.6%**. Sedangkan efektifitas kompresi paling besar adalah **60.5%** pada konfigurasi **HS(9,8)**. Tabel 5.12 menjabarkan hasil uji coba dari efektifitas kompresi pada panjang data **980 karakter** yang telah dijabarkan lebih rinci pada Table 7.2 di lampiran.

Berdasarkan hasil uji coba efektifitas kompresi pada panjang data **1280 karakter**, didapatkan persentase efektifitas kompresi paling kecil ketika menggunakan konfigurasi HS(4,3) sebesar **10.0%**. Sedangkan efektifitas kompresi paling besar adalah **50.2%** pada konfigurasi **HS(8,7)**. Tabel 5.12 menjabarkan hasil uji coba dari efektifitas kompresi pada panjang data **1280 karakter** yang telah dijabarkan lebih rinci pada Table 7.3 di lampiran.

Berdasarkan hasil uji coba efektifitas kompresi pada panjang data **1345 karakter**, didapatkan persentase efektifitas kompresi paling kecil ketika menggunakan konfigurasi HS(4,3) sebesar **10.2%**. Sedangkan efektifitas kompresi paling besar adalah **54.5%** pada konfigurasi **HS(8,7)**. Tabel 5.12 menjabarkan hasil uji coba dari efektifitas kompresi pada panjang data **1345 karakter** yang telah dijabarkan lebih rinci pada Table 7.4 di lampiran.

Berdasarkan hasil uji coba efektifitas kompresi pada panjang data **1870 karakter**, didapatkan persentase efektifitas kompresi paling kecil ketika menggunakan konfigurasi HS(4,3) sebesar **5.1%**. Sedangkan efektifitas kompresi paling besar adalah **27.8%** pada konfigurasi **HS(8,7)**. Tabel 5.12 menjabarkan hasil uji coba dari efektifitas kompresi pada panjang data **1870 karakter** yang telah dijabarkan lebih rinci pada Table 7.5 di lampiran.

Pada konfigurasi **HS (9,8)** untuk data dengan panjang lebih dari 1219 karakter tidak dapat dilakukan. Selain itu, pada konfigurasi **HS(8,7)** tidak dapat berjalan pada data dengan panjang lebih dari 1869 karakter, keadaan tersebut disebabkan karena keterbatasan kapasitas memori yang ada pada Arduino.

5.6.2 Hasil Uji Coba (UJ-P02) – Akurasi Pengiriman Data ZigBee

Mengacu pada skenario bab 5.5.2, dilakukan pengujian untuk mengetahui akurasi pengiriman data pada ZigBee. Setiap data yang diterima akan di print pada serial monitor node ZigBee Coordinator. Berikut ini table 5.13 menjelaskan hasil uji coba yang didapatkan

Tabel 5.15 Hasil Uji Coba UJ – P02 Pengiriman Data ZigBee

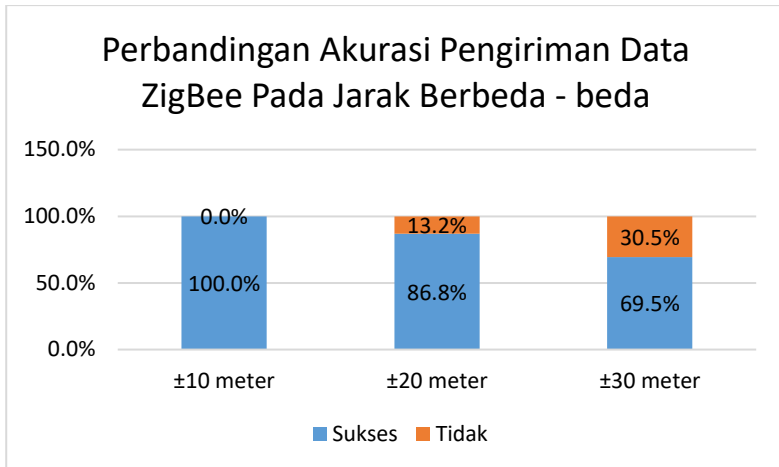
Akurasi Pengiriman ZigBee		
±10 meter	±20 meter	±30 meter
100%	86,8%	69 ,5 %

Berdasarkan hasil uji coba akurasi pada jarak **±10 meter**, didapatkan akurasi pengiriman data menggunakan ZigBee sebesar **100%**. Tabel 5.13 menjabarkan hasil uji coba dari jarak **±10 meter** yang telah dijabarkan lebih rinci pada Table 7.6 di lampiran

Berdasarkan hasil uji coba akurasi pada jarak **±20 meter**, didapatkan akurasi pengiriman data menggunakan ZigBee sebesar **86,8%**. Tabel 5.13 menjabarkan hasil uji coba dari jarak **±20 meter** yang telah dijabarkan lebih rinci pada Table 7.7 di lampiran

Berdasarkan hasil uji coba akurasi pada jarak **±30 meter**, didapatkan akurasi pengiriman data menggunakan ZigBee sebesar **69,5%**. Tabel 5.13 menjabarkan hasil uji coba dari jarak **±30 meter** yang telah dijabarkan lebih rinci pada Table 7.8 di lampiran

Untuk lebih rincinya, berikut ini adalah perbandingan data akurasi pengiriman data ZigBee pada jarak berbeda - beda.



Berdasarkan grafik diatas, dapat diketahui bahwa semakin jauh jarak komunikasi maka akurasi pengiriman data ZigBee akan semakin menurun.

5.6.3 Hasil Uji Coba (UJ-P03) – Waktu Kompresi Data

Pada skenario bab 5.5.3 dijabarkan tentang uji coba waktu yang dibutuhkan dalam melakukan proses kompresi data. Pengujian dilakukan pada 10 buah data string yang berbeda - beda untuk masing – masing panjang data pada skenario. Berikut ini adalah hasil uji coba yang didapatkan.

Tabel 5.16 Hasil Uji Coba UJ – P03 Waktu Kompresi Data

Panjang Data	Uji Coba	Konfigurasi					
		HS(4, 3)	HS(5, 4)	HS(6, 5)	HS(7, 6)	HS(8, 7)	HS(9, 8)
584	Waktu	0.05804	0.04692	0.04142	0.03231	0.02948	0.03005
980	Waktu	0.09688	0.08148	0.0688	0.05472	0.0529	0.0499
1280	Waktu	0.12733	0.11091	0.0877	0.07781	0.0784	gagal
1345	Waktu	0.13365	0.11642	0.09174	0.08168	0.08238	gagal
1870	Waktu	0.19124	0.16759	0.14966	0.14012	gagal	gagal

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **584 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.02948 detik** ketika menggunakan konfigurasi **HS (8,7)**. Tabel 5.14 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **584 karakter** yang telah dijabarkan lebih rinci pada Table 7.9 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **980 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.0499 detik** ketika menggunakan konfigurasi **HS (9,8)**. Tabel 5.14 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **980 karakter** yang telah dijabarkan lebih rinci pada Table 7.10 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **1280 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.07781 detik** ketika menggunakan konfigurasi **HS (7,6)**. Tabel 5.14 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **1280 karakter** yang telah dijabarkan lebih rinci pada Table 7.11 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **1345 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.08168 detik** ketika menggunakan konfigurasi **HS (7,6)**. Tabel 5.14 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **1345 karakter** yang telah dijabarkan lebih rinci pada Table 7.12 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **1870 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.14012 detik** ketika menggunakan konfigurasi **HS (7,6)**. Tabel 5.14 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **1345 karakter** yang telah dijabarkan lebih rinci pada Table 7.13 di lampiran.

Pada konfigurasi **HS (9,8)** untuk data dengan panjang lebih dari 1219 karakter tidak dapat dilakukan. Selain itu, pada konfigurasi **HS (8,7)** tidak dapat berjalan pada data dengan panjang lebih dari 1869 karakter, keadaan tersebut disebabkan karena keterbatasan kapasitas memori yang ada pada Arduino.

5.6.4 Hasil Uji Coba (UJ-P04) – Waktu Dekompresi Data

Pada skenario bab 5.5.4 dijabarkan tentang uji coba waktu yang dibutuhkan dalam melakukan proses dekompresi data. Pengujian dilakukan pada 10 buah data string yang berbeda - beda dari hasil kompresi data sebelumnya. Berikut ini adalah hasil uji coba yang didapatkan.

Tabel 5.17 Hasil Uji Coba UJ – P04 Waktu Dekompresi Data

Panjang Data	Uji Coba	Konfigurasi					
		HS(4, 3)	HS(5, 4)	HS(6, 5)	HS(7, 6)	HS(8, 7)	HS(9, 8)
584	Waktu	0.020436	0.018062	0.016648	0.013286	0.011669	0.011856
980	Waktu	0.033833	0.031182	0.027464	0.022094	0.01983	0.018288
1280	Waktu	0.044415	0.042452	0.034629	0.030599	0.027731	gagal
1345	Waktu	0.046591	0.044518	0.036244	0.032141	0.028978	gagal
1870	Waktu	0.066589	0.064135	0.059676	0.055481	gagal	gagal

Berdasarkan hasil uji coba waktu dekompresi data pada panjang data **584 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.011669 detik** ketika menggunakan konfigurasi **HS (8,7)**. Tabel 5.16 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **584 karakter** yang telah dijabarkan lebih rinci pada Table 7.14 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **980 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.018288 detik** ketika menggunakan konfigurasi **HS (9,8)**. Tabel 5.16 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **980 karakter** yang telah dijabarkan lebih rinci pada Table 7.15 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **1280 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.027731 detik** ketika menggunakan konfigurasi **HS (8,7)**. Tabel 5.16 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **1280 karakter** yang telah dijabarkan lebih rinci pada Table 7.16 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **1345 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.028978 detik** ketika menggunakan konfigurasi **HS (8,7)**. Tabel 5.16 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **1345 karakter** yang telah dijabarkan lebih rinci pada Table 7.17 di lampiran.

Berdasarkan hasil uji coba waktu kompresi data pada panjang data **1870 karakter**, waktu tercepat yang dibutuhkan untuk kompresi data adalah **0.055481 detik** ketika menggunakan konfigurasi **HS (8,7)**. Tabel 5.16 menjabarkan hasil uji coba dari waktu kompresi pada panjang data **1870 karakter** yang telah dijabarkan lebih rinci pada Table 7.18 di lampiran.

Pada konfigurasi **HS (9,8)** untuk data dengan panjang lebih dari 1219 karakter tidak dapat dilakukan. Selain itu, pada konfigurasi **HS (8,7)** tidak dapat berjalan pada data dengan panjang lebih dari 1869 karakter, keadaan tersebut disebabkan karena keterbatasan kapasitas memori yang ada pada Arduino.

5.7 Evaluasi Hasil Uji Coba

Pada bagian sebelumnya, telah dilakukan uji coba terhadap system yang dibuat. Uji coba yang telah dilakukan berkenaan dengan uji coba fungsionalitas dan uji coba performa. Berikut ini Tabel 5.16. merangkum evaluasi hasil uji coba fungsionalitas dan Tabel 5.17 merangkum evaluasi hasil uji coba performa yang telah dilakukan.

Tabel 5.18 Evaluasi Hasil Uji Coba Fungsionalitas

No	Kode Uji Coba	Evaluasi
1	UJ – F01	Dapat membaca data dari sd card berdasarkan skenario yang ada
2	UJ – F02	Sebagian besar data berhasil di kompresi, akan tetapi ketika menggunakan

		konfigurasi HS(9,8) pada panjang data lebih dari 1219 karakter kompresi data mengalami kegagalan dikarenakan keterbatasan memori yang ada pada Arduino.
3	UJ – F03	Komunikasi pada topologi cluster tree berjalan dengan baik, hal ini dapat dilihat dari node ZigBee Coordinator dapat menerima data yang dikirimkan oleh node ZigBee End Device
4	UJ – F04	Sebagian besar data berhasil di dekompresi, akan tetapi ketika menggunakan konfigurasi HS(9,8) pada panjang data lebih dari 1219 karakter kompresi data mengalami kegagalan dikarenakan keterbatasan memori yang ada pada Arduino.
5	UJ – F05	Kompresi adaptive berhasil dilakukan dengan konfigurasi encoder/decoder yang telah ditetapkan.

Tabel 5.19 Evaluasi Hasil Uji Coba Performa

No	Kode Uji Coba	Evaluasi
1	UJ – P01	Persentase efektifitas kompresi yang didapatkan semakin besar seiring dengan besarnya konfigurasi window_sz dan lookahead_sz yang digunakan.
2	UJ – P02	Rata – rata waktu yang dibutuhkan untuk proses kompresi data semakin cepat seiring dengan besarnya konfigurasi window_sz dan lookahead_sz yang digunakan.

3	UJ – P03	Akurasi pengiriman data ZigBee berturut – turut adalah 100% (± 10 meter), 86,8% (± 20 meter), dan 69,5% (± 30 meter). Sehingga dapat disimpulkan, bahwa akurasi ZigBee menurun seiring dengan jarak yang semakin meningkat.
4	UJ – P04	Rata – rata waktu yang dibutuhkan untuk proses dekompresi data semakin cepat seiring dengan besarnya konfigurasi window_sz dan lookahead_sz yang digunakan.

BAB VI

KESIMPULAN DAN SARAN

Bab ini berisikan kesimpulan yang dapat diambil dari hasil uji coba yang telah dilakukan. Selain kesimpulan, terdapat juga saran yang ditujukan untuk pengembangan perangkat lunak selanjutnya.

6.1 Kesimpulan

Kesimpulan yang didapatkan berdasarkan hasil uji coba kompresi adaptive menggunakan metode Heatshrink untuk pengiriman data pada *wireless sensor network* berbasis ZigBee adalah sebagai berikut:

1. Implementasi kompresi adaptive menggunakan metode Heatshrink dapat digunakan untuk mengurangi pengiriman data sehingga dapat menghemat penggunaan daya pada *wireless sensor network*.
2. Pada beberapa kondisi kompresi data tidak dapat digunakan karena keterbatasan sumberdaya yang ada pada *wireless sensor network*.
3. Performa dari system yang telah dibangun adalah sebagai berikut:
 - a. Persentasi efektifitas kompresi paling tinggi adalah **60,5%** pada panjang data 980 karakter dengan menggunakan konfigurasi HS(9,8). Persentasi efektifitas kompresi pada masing-masing data sesuai skenario uji coba rata-rata akan terus meningkat seiring dengan besar konfigurasi yang digunakan.
 - b. Akurasi pengiriman data ZigBee berturut – turut adalah **100%** (± 10 meter), **86,8%** (± 20 meter), dan **69,5%** (± 30 meter). Sehingga dapat disimpulkan, bahwa akurasi ZigBee menurun seiring dengan jarak yang semakin meningkat.

- c. Waktu yang dibutuhkan untuk kompresi data paling cepat adalah **0.02948 detik** pada panjang data 584 karakter dengan menggunakan konfigurasi HS(8,7). Waktu kompresi pada masing-masing data sesuai skenario uji coba rata-rata akan mengalami percepatan seiring dengan besar konfigurasi yang digunakan.
- d. Waktu yang dibutuhkan untuk dekompresi data paling cepat adalah **0.011856 detik** pada panjang data 584 karakter dengan menggunakan konfigurasi HS(9,8). Waktu dekompresi pada masing-masing data sesuai skenario uji coba rata-rata akan mengalami percepatan seiring dengan besar konfigurasi yang digunakan.

6.2 Saran

Saran yang diberikan terkait pengembangan pada Tugas Akhir ini adalah:

1. Menambah jumlah data untuk uji coba berdasarkan panjang datanya.
2. Menambahkan jumlah node pengirim untuk uji coba.

DAFTAR PUSTAKA

- [1] Peter Harrop dan Raghu Das , “ Wireless Sensor Networks (WSN) 2014-2024: Forecasts, Technologies, Players” IDTechEx, 2014. [Daring]. Tersedia: <https://www.idtechex.com/research/reports/wireless-sensor-networks-wsn-2014-2024-forecasts-technologies-players-000382.asp> [Diakses 25-Des-2017].
- [2] S. Rhee, D. Seetharam dan S. Liu, “Techniques for Minimizing Power Consumption in Low Data-Rate Wireless Sensor Networks”, IEEE Wireless Communications and Networking Conference, vol. 3, pp. 1727-1731, 2004.
- [3] R. Rajagopalan dan P. K. Varshney, “Data Aggregation Techniques in Sensor Networks: A Survey” in , IEEE Communication Surveys & Tutorials, vol. 8, no. 4, pp. 48-63, 2006.
- [4] C. Wang, H. Ma, Y. He dan S. Xiong, "Adaptive Approximate Data Collection for Wireless Sensor Network", IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 6, pp. 1004-1016, 2012.
- [5] P. Ghaffariyan, “An Effective Data Aggregation Mechanism for Wireless Sensor Networks”, 6th International Conference on Wireless Communications Networking and Mobile Computing, 2010.
- [6] M. Vodel dan W. Hardt, “Data Aggregation and Data Fusion Techniques In WSN/SANET Topologies - A Critical Discussion,” TENCON 2012 - 2012 IEEE Region 10 Conference, pp. 1-6, 2012
- [7] Solahuddin Y. F dan W. Ismail, “Data fusion for reducing power consumption in Arduino-Xbee wireless

- sensor network platform”, Computer and Information Sciences (ICCOINS), 2014.
- [8] Shu Yinbiao dan Kang Lee, “Internet of Things: Wireless Sensor Networks”, International Electrotechnical Commission, White Paper, 2014.
 - [9] Bröring A, “New Generation Sensor Web Enablement”, PMC, 2011. [Daring]. Tersedia: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3231615/> [Diakses 2-Jan-2018]
 - [10] “Arduino.” Wikipedia bahasa Indonesia, ensiklopedia bebas. [Daring]. Tersedia: <https://id.wikipedia.org/wiki/Arduino> [Diakses 2-Jan-2018]
 - [11] Arduino, “Arduino Software (IDE).” [Daring]. Tersedia: <https://www.arduino.cc/en/Guide/Environment> [Diakses 3-Jan-2018]
 - [12] “Shield Arduino.” [Daring]. Tersedia: <http://www.belajararduino.net/shield-arduino/> [Diakses 3-Jan-2018]
 - [13] “Zigbee Networking with XBee Series 2 and Seeed's Products.” [Daring]. Tersedia: http://wiki.seeed.cc/Zigbee_Networking_with_XBee_Series_2_and_Seed's_Products/ [Diakses 3-Jan-2018]
 - [14] J. Song dan Y. K. Tan, “Energy Consumption Analysis of ZigBee-based Energy Harvesting Wireless Sensor Network”, IEEE International Conference on Communication Systems (ICCS), 2012
 - [15] Scott Vokes, “heatshrink: An Embedded Data Compression Library.” [Daring]. Tersedia: <https://spin.atomicobject.com/2013/03/14/heatshrink-embedded-data-compression/> [Diakses 3-Jan-2018]

- [16] “ Algoritma Kompresi LZSS” [Daring]. Tersedia: <https://mahfudharun.wordpress.com/2012/06/14/algoritma-kompresi-lzss/> [Diakses 6-Jan-2018]
- [17] Adnan Ozsoy dan Martin Swamy, “ CULZSS: LZSS Lossless Data Compression on CUDA”, IEEE International Conference on Cluster Computing (CLUSTER), 2011
- [18] “Lempel–Ziv–Storer–Szymanski.” Wikipedia, the free encyclopedia . [Daring]. Tersedia: <https://en.wikipedia.org/wiki/Lempel–Ziv–Storer–Szymanski> [Diakses 6-Jan-2018]
- [19] Arduino, “Memory.” [Daring]. Tersedia: <https://www.arduino.cc/en/Tutorial/Memory> [Diakses 6-Jan-2018]
- [20] Oracle, “ Data Types and Sizes.” [Daring]. Tersedia: <https://docs.oracle.com/cd/E19253-01/817-6223/chp-typeopexpr-2/index.html> [Diakses 6-Jan-2018]

(Halaman ini sengaja dikosongkan)

LAMPIRAN

Tabel 8.1 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 584 karakter

Efektifitas Kompresi Pada Panjang Data 584		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	
Konfigurasi	HS(4,3)	87 .0 %	2. 7%	- 4. 6%	- 10 .4 %	- 2. 2%	1. 7%	- 1. 4%	2. 4%	- 0. 2%	17 .6 %	9.3%
	HS(5,4)	91 .6 %	7. 0%	9. 8%	- 1. 0%	4. 3%	23 .8 %	16 .3 %	58 .9 %	4. 3%	29 .3 %	24.4%
	HS(6,5)	94 .7 %	10 .3 %	24 .1 %	5. 1%	4. 8%	37 .5 %	15 .2 %	75 .7 %	12 .2 %	38 .9 %	31.8%
	HS(7,6)	96 .6 %	9. 4%	29 .3 %	7. 7%	12 .0 %	76 .7 %	67 .5 %	80 .0 %	72 .9 %	41 .8 %	49.4%
	HS(8,7)	97 .8 %	23 .5 %	26 .7 %	44 .2 %	38 .5 %	76 .0 %	66 .8 %	79 .5 %	71 .7 %	73 .8 %	59.8%
	HS(9,8)	98 .5 %	26 .4 %	26 .9 %	43 .3 %	42 .1 %	75 .7 %	66 .1 %	78 .8 %	71 .1 %	38 .7 %	56.7%

**Tabel 8.2 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data
980 karakter**

Efektifitas Kompresi Pada Panjang Data 980		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	
Konfigurasi	HS(4,3)	87 .1 %	3. 5%	5. 6%	0. 8%	- 3. 1%	- 1. 5%	10 .5 %	- 3. 8%	- 1. 0%	- 2. 2%	9.6%
	HS(5,4)	91 .8 %	6. 1%	10 .9 %	24 .0 %	13 .0 %	18 .3 %	10 .0 %	0. 9%	11 .7 %	12 .4 %	19.9%
	HS(6,5)	95 .0 %	8. 0%	19 .9 %	38 .6 %	17 .8 %	50 .1 %	9. 0%	26 .9 %	29 .0 %	29 .3 %	32.3%
	HS(7,6)	96 .8 %	7. 6%	31 .6 %	74 .1 %	18 .3 %	73 .0 %	67 .3 %	49 .1 %	35 .5 %	37 .7 %	49.1%
	HS(8,7)	98 .1 %	40 .9 %	39 .1 %	73 .1 %	27 .8 %	77 .9 %	68 .0 %	47 .3 %	35 .7 %	37 .2 %	54.5%
	HS(9,8)	98 .8 %	43 .1 %	45 .5 %	72 .7 %	64 .5 %	88 .2 %	68 .0 %	48 .0 %	37 .3 %	39 .1 %	60.5%

**Tabel 8.3 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data
1280 karakter**

Efektifitas Kompresi Pada Panjang Data 1280		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	

Konfigurasi	HS(4,3)	87 .3 %	3. 8%	- 5. 1%	1. 3%	- 3. 6%	- 6. 6%	13 .8 %	- 5. 1%	- 5. 5%	20 .1 %	10.0%
	HS(5,4)	92 .0 %	5. 9%	- 0. 3%	19 .3 %	3. 7%	0. 2%	20 .5 %	0. 4%	- 1. 5%	28 .1 %	16.8%
	HS(6,5)	95 .1 %	7. 0%	18 .8 %	27 .3 %	13 .0 %	24 .5 %	24 .1 %	65 .3 %	67 .6 %	31 .4 %	37.4%
	HS(7,6)	97 .0 %	10 .4 %	43 .2 %	40 .1 %	21 .0 %	24 .7 %	24 .6 %	81 .9 %	82 .1 %	34 .3 %	45.9%
	HS(8,7)	98 .2 %	53 .9 %	45 .9 %	40 .5 %	22 .5 %	26 .8 %	21 .3 %	81 .1 %	81 .2 %	30 .9 %	50.2%
	HS(9,8)	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	gagal

Tabel 8.4 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 1345 karakter

Efektifitas Kompresi Pada Panjang Data 1345		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	
Konfigurasi	HS(4,3)	87 .3 %	4. 3%	- 5. 0%	1. 1%	- 3. 3%	- 6. 1%	13 .5 %	- 5. 0%	- 5. 6%	20 .7 %	10.2%
	HS(5,4)	92 .0 %	6. 1%	- 0. 4%	19 .7 %	3. 6%	1. 2%	19 .9 %	0. 4%	- 1. 6%	29 .0 %	17.0%

[illegible]

Tabel 8.5 Rincian Hasil Uji Coba UJ-P01 pada Panjang Data 1870 karakter

[illegible]

HS(9,8)	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	ga gal	gagal
---------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------	-------

Tabel 8.6 Rincian Hasil Uji Coba UJ-P02 pada Jarak ± 10 Meter

Pengiriman Data Via ZigBee (± 10 meter)		Percobaan ke-										Akurasi
		1	2	3	4	5	6	7	8	9	10	
Jumlah Data Masuk	Sesuai	41	41	41	41	41	41	41	41	41	41	100.0%
	Tidak	0	0	0	0	0	0	0	0	0	0	

Tabel 8.7 Rincian Hasil Uji Coba UJ-P02 pada Jarak ± 20 Meter

Pengiriman Data Via ZigBee (± 20 meter)		Percobaan ke-										Akurasi
		1	2	3	4	5	6	7	8	9	10	
Jumlah Data Masuk	Sesuai	40	36	39	38	36	35	40	38	27	27	86.8%
	Tidak	1	5	2	3	5	6	1	3	14	14	

Tabel 8.8 Rincian Hasil Uji Coba UJ-P02 pada Jarak ± 30 Meter

Pengiriman Data Via ZigBee (± 30 meter)		Percobaan ke-										Akurasi
		1	2	3	4	5	6	7	8	9	10	
Jumlah Data Masuk	Sesuai	27	32	37	38	30	34	26	20	12	29	69.5%
	Tidak	14	9	4	3	11	7	15	11	29	12	

Tabel 8.9 Rincian Hasil Uji Coba UJ-P03 pada Panjang Data 584 Karakter

Waktu Kompresi Pada Panjang Data 584		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	
Konfigurasi	HS(4,3)	0.021668	0.06192	0.06382	0.066196	0.062816	0.062168	0.063084	0.060624	0.061324	0.054956	0.058035
	HS(5,4)	0.014168	0.054824	0.052968	0.058036	0.056048	0.047524	0.052024	0.03274	0.05576	0.045132	0.046922
	HS(6,5)	0.010436	0.051796	0.04422	0.05368	0.054028	0.03934	0.050396	0.02044	0.051008	0.038812	0.041416
	HS(7,6)	0.009128	0.051924	0.041576	0.052336	0.051124	0.01888	0.023332	0.017176	0.020936	0.036704	0.032312
	HS(8,7)	0.008444	0.04648	0.044192	0.035056	0.039484	0.019624	0.023364	0.018768	0.020644	0.038712	0.029477
	HS(9,8)	0.009796	0.04566	0.044852	0.036096	0.038076	0.019476	0.025196	0.019356	0.022348	0.030396	0.030046

Tabel 8.10 Rincian Hasil Uji Coba UJ-P03 pada Panjang Data 980 Karakter

Waktu Kompresi Pada Panjang Data 980		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	
Konfigurasi	HS(4,3)	0.036	0.103012	0.099108	0.104328	0.10516	0.105276	0.098072	0.107804	0.104436	0.10106	0.096876
	HS(5,4)	0.023316	0.09098	0.088132	0.079584	0.08662	0.082324	0.08842	0.097552	0.088636	0.088276	0.081484
	HS(6,5)	0.016504	0.08077724	0.07818168	0.06496464	0.078772	0.056884	0.084104	0.074172	0.073576	0.073092	0.068796
	HS(7,6)	0.013812	0.088556	0.06996	0.03318	0.07762	0.037476	0.035456	0.05668	0.06798	0.066516	0.054723
	HS(8,7)	0.01284	0.063624	0.065644	0.036612	0.072836	0.033568	0.037744	0.061048	0.072896	0.072172	0.052898
	HS(9,8)	0.013732	0.060636	0.062556	0.03756	0.040268	0.020992	0.039064	0.064876	0.078504	0.077816	0.049897

Tabel 8.14 Rincian Hasil Uji Coba UJ-P04 pada Panjang Data 584 Karakter

Waktu Dekompresi Pada Panjang Data 584		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	
Konfigurasi	HS(4,3)	0.007508	0.0218	0.022624	0.023324	0.022284	0.021784	0.022352	0.021624	0.021844	0.019212	0.020436
	HS(5,4)	0.00494	0.021284	0.020588	0.0226	0.021548	0.018476	0.020104	0.01232	0.021484	0.01728	0.018062
	HS(6,5)	0.004028	0.020812	0.018168	0.021636	0.020217	0.015712	0.02034	0.00842	0.020184	0.015476	0.016648
	HS(7,6)	0.003556	0.021136	0.017324	0.021244	0.02062	0.007972	0.009912	0.007392	0.008728	0.014976	0.013286
	HS(8,7)	0.003316	0.017804	0.017348	0.013976	0.015072	0.008064	0.00986	0.007456	0.00888	0.014912	0.011669
	HS(9,8)	0.003252	0.017548	0.017664	0.014424	0.014684	0.008332	0.010196	0.007768	0.009224	0.015464	0.011856

Tabel 8.15 Rincian Hasil Uji Coba UJ-P04 pada Panjang Data 980 Karakter

Waktu Dekompresi Pada Panjang Data 980		Percobaan ke-										Rata - rata
		1	2	3	4	5	6	7	8	9	10	
Konfigurasi	HS(4,3)	0.012284	0.035876	0.035028	0.036416	0.036992	0.036704	0.033568	0.03786	0.036616	0.036988	0.033833
	HS(5,4)	0.009132	0.03538	0.03396	0.03058	0.033016	0.031452	0.033584	0.037476	0.030337	0.03354	0.031182
	HS(6,5)	0.006336	0.035096	0.031668	0.025748	0.031816	0.02226	0.03402	0.029590	0.029090	0.029048	0.027464
	HS(7,6)	0.005524	0.030355	0.028088	0.02013848	0.0303194	0.014696	0.015896	0.02222	0.026916	0.026316	0.022094
	HS(8,7)	0.005112	0.024004	0.024576	0.024084	0.028384	0.012812	0.015588	0.022248	0.025992	0.020255	0.01983
	HS(9,8)	0.00498	0.023648	0.023448	0.024484	0.027084	0.009504	0.015828	0.022508	0.025996	0.025396	0.018288

BIODATA PENULIS



Muhamad Hendri Febriansyah merupakan anak kembar dari pasangan Bapak Jamaludin dan Ibu Suryati. Lahir di Mataram pada tanggal 28 Februari 1996. Penulis menempuh pendidikan formal dimulai SDN 29 Mataram (2002-2008), MTsN 1 Mataram (2008-2011), SMAN 1 Mataram (2011-2014) dan S1 Departemen Informatika ITS (2014-2018). Bidang studi yang diambil oleh penulis pada saat berkuliah di Teknik Informatika ITS adalah Komputasi Berbasis Jaringan (KBJ). Penulis aktif dalam organisasi seperti Himpunan Mahasiswa Teknik Computer-Informatika (2015-2016) dan KMI (2015-2017). Penulis juga aktif dalam berbagai kegiatan kepanitiaan yaitu SCHEMATICS 2015-2016 divisi Keamanan dan Perizinan. Penulis juga menyukai kegiatan pecinta alam. Penulis memiliki hobi traveling dan bermain game. Penulis dapat dihubungi melalui email: hendrifebriansyah28@gmail.com.