IMPLEMENTASI ERROR DETECTION SYSTEM PADA KOMUNIKASI SERIAL ARDUINO MENGGUNAKAN METODE CYCLIC REDUNDANCY CHECK (CRC)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
Reynald Novaldi
NIM : 145150301111043

PROGRAM STUDI TEKNIK KOMPUTER
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2018

PENGESAHAN

IMPLEMENTASI ERROR DETECTION SYSTEM PADA KOMUNIKASI SERIAL ARDUINO MENGGUNAKAN METODE CYCLIC REDUNDANCY CHECK (CRC)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun Oleh :
Reynald Novaldi
NIM : 145150301111043

Skripsi ini telah diuji dan dinyatakan lulus pada
02 November 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Sabriansyah Rizqika Akbar, S.T, M.Eng	Rakhmadhany Primananda, S.T, M.Kom
NIP. 19820809 201212 1 004	NIK. 2016098604061001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP. 19710518 200312 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang,

Reynald Novaldi

NIM: 145150301111043

KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa, karena atas berkat dan hidayah-Nya, penulis dapat menyelesaikan dan menyusun laporan skripsi yang berjudul "IMPLEMENTASI *ERROR DETECTION SYSTEM* PADA KOMUNIKASI SERIAL ARDUINO MENGGUNAKAN METODE *CYCLIC REDUNDANCY CHECK* (CRC)*".*

Peneliti menyadari bahwa dalam penyusun dan penyelesaian laporan skripsi ini tidak akan bisa selesai tanpa mendapatkan bantuan dari pihak-pihak terkait, oleh karena itu peneliti ingin memberikan rasa hormat serta ucapan terima kasih kepada :

1. Allah SWT yang selalu memberikan rahmat, hidayah, petunjuk dan kelancaran pada skripsi ini.
2. Kedua orang tua yang sudah memberikan dukungan kepada peneliti baik dengan do'a, dukungan moral dan material.
3. Bapak Sabriansyah Rizqika Akbar, S.T, M.Eng dan Bapak Rakhmadhany Primananda, S.T, selaku dosen pembimbing yang sudah mau meluangkan waktu, pikiran dan tenaganya dalam membimbing dan mengarahkan peneliti sehingga skripsi ini dapat diselesaikan.
4. Bapak Tri Astoto Kurniawan, S.T., M.T., Ph.D selaku Ketua Juruan Teknik Informatika.
5. Cipto Bagus, Wisnu, Lutfi, Ganda, Fanani, Sandy, Ikhwan, Devi, yang telah membantu pengerjaan dan memberikan dukungan moral selama pengerjaan skripsi.
6. Seluruh civitas akademik Fakultas Ilmu Komputer Universitas Brawijaya yang sudah bersedia membantu peneliti selama menempuh studi di Universitas Brawijaya

Dalam pengerjaan skripsi ini, penulis menyadari bahwa skripsi yang dikerjakan belum sempurna , sehingga untuk segala saran dan kritik akan sangat diterima oleh peneliti. Akhir kata, penulis harap skripsi ini dapat bermanfaat bagi orang yang menggunakannya.

Malang, 1 Januari 2015

Penulis

novaldireynald@gmail.com

ABSTRAK

Masalah utama dalam sistem komunikasi data yaitu adanya kesalahan data yang menyebabkan data tersebut diterima tidak sebagaimana mestinya, adanya *error* pada data tersebut dapat disebabkan oleh *noise*, faktor cuaca, *crosstalk* dan adanya tegangan listrik yang tidak stabil. Adanya kesalahan pada data ini dapat dikurangi dan dihindari dengan menerapkan sistem pendeteksi kesalahan atau *error detection* seperti metode *Hamming Code*, *Reed Solomon Code*, dan *Cyclic Redundancy Check.* Mikrokontroller Arduino terutama Uno adalah salah satu mikrokontroller yang banyak digunakan untuk *embedded system* dan *internet of things* yang keduanya tidak bisa dijauhkan dari pertukaran data dan informasi. Metode pada penelitian ini menggunakan metode *Cyclic Redundancy Check* yang yang diimplementasikan pada sistem komunikasi serial antar arduino. Metode ini merupakan satu dari banyak metode untuk mendeteksi *error* atau kesalahan pada data yang banyak digunakan dan diimplementasikan pada perangkat penyimpanan dan sistem komunikasi. CRC sangat mudah diimplementasikan pada perangkat keras dan perangkat lunak dengan menggunakan operasi penjumlahan dan penggeseran bit yang sederhana dengan polinomialnya. Pada penelitian ini sistem *error detection* dengan metode CRC diimplementasikan pada serial komunikasi dua arduino uno yang salah satu arduino uno bertugas sebagai pengirim data dan satunya bertugas sebagai penerima data. Dari pengujian fungsionalitas sistem error detection didapatkan bahwa sistem dapat mendeteksi adanya kesalahan pada data dengan tepat dengan persentase 100% dan pada pengujian waktu didapatkan hasil bahwa sistem dengan komunikasi UART membutuhkan waktu yang lebih cepat daripada sistem dengan SPI atau I2C.

Kata kunci: CRC, *Error Detection*, Arduino, Komunikasi Serial

ABSTRACT

The main problem in the data communication system is the data error that causes the data received is not as it should, the error on the data can be caused by noise, weather factors, crosstalk and the presence of unstable voltage. An error in these data can be reduced and avoided by applying a system of detection errors or error detection methods such as Reed Solomon Code, Hamming Code, and Cyclic Redundancy Check. Arduino microcontroller especially Uno is one of the widely used mikrokontroller for embedded systems and internet of things that both of them could not be kept away from the exchange of data and information. Research on methods using Cyclic Redundancy Check methods that are implemented on the serial communication system between the arduino. This method is one of many methods to detect errors or errors in the data that are widely used and implemented in the storage device and communication system. CRC is quite easy to implement in hardware and software by using simple addition and bit shift operation with its polynomial. In this research system error detection with CRC method implemented in serial communication two arduino uno one of arduino uno served as sender of data and the only one duty as data recipient. From functionality test of the system error detection is resulted that the system can detect an error in the data correctly with a percentage of 100%, and at time test obtained the results that the system with UART communication takes a faster time than a system with SPI or I2C.

Keyword: CRC, *Error Detection*, Arduino, *Serial Communication*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Pada era perkembangan teknologi saat ini, semua bidang telah banyak mengalami perkembangan yang sangat pesat. Salah satu bidang yang mengalami perkembangan secara pesat adalah bidang komunikasi. Komunikasi pada teknologi saat ini tidak hanya dapat dilakukan secara *wired*, tetapi juga dapat dilakukan secara *wireless*. Tetapi dengan komunikasi seperti ini tidak menutup kemungkinan pengiriman informasi atau data yang diterima tidak terkirim dengan benar dan bahkan bisa belum sampai terkirim, atau bisa dikatakan data yang dikirimkan tersebut mengalami *error*.

Banyak faktor yang mempengaruhi pengiriman data tersebut menjadi gagal, misal faktor cuaca, *noise*, radiasi, tegangan listrik tidak stabil, *cross talk* (Fu, et al., 2009). Gangguan yang tidak diinginkan (*noise*) dapat terjadi di saluran komunikasi dan menyebabkan informasi yang diterima menjadi berbeda dari informasi asli dikirim (Shannon, 1948). Data biner yang rusak dapat berubah dari 1 menjadi 0 atau sebaliknya 0 menjadi 1 (Bogart Jr, 1992). Oleh karena itu keberhasilan pengiriman data dengan benar dan akurat merupakan salah satu faktor keberhasilan dalam komunikasi data. Untuk menangani masalah pengiriman data *error* terdapat beberapa metode yaitu *Error Detection* menggunakan enkode dan dekode data yang dikirim. Metode *Error Detection* ada beberapa metode yaitu Hamming Code, Cyclic Redudancy Code, Linear Feedback Shift Register, Reed Solomon, dll.

Cyclic Redundancy Check adalah salah satu teknik *error detection* dengan menggunakan teknik generator polinomial yang dapat menghasilkan bit untuk pengecekan data atau yang biasa disebut *checksum* dan mudah diaplikasikan ke dalam berbagai hal. Hal itulah yang membuat penulis menggunakan metode CRC untuk penelitian ini, karena dirasa metode ini tepat untuk data yang panjang, dan cocok untuk mendeteksi *burst error* pada perangkat keras (*hardware*) dan perangkat lunak (*software*).

Banyak penelitian terdahulu yang menggunakan teknik CRC sebagai metode untuk melakukan pengecekan *error* pada hardware seperti penelitian implementasi CRC yang didesain paralel untuk melakukan pendeteksian *error* dengan cepat yang dilakukan Gawande(2014) pada penelitian tersebut peneliti melakukan desain dan implementasi metode CRC paralel untuk pengaplikasian pada *random access memory* (RAM) dan perangkat penyimpanan atau *storage devices*. Adapun penelitian yang melakukan pendeteksian *error* pada *memory* oleh Lee(2015) yang menerapkan metode CRC untuk melakukan pengecekan data pada komunikasinya, penelitian ini menggunakan *Cyclic Redundancy Check* untuk mendeteksi *burst error* pada memori semikonduktor cepat. Penelitian untuk melakukan pendeteksian *error* pada *software* atau perangkat lunak yang dilakukan oleh Engdahl(2014) ini merancang dan melakukan implementasi metode paralel CRC untuk mendeteksi kesalahan data perangkat lunak dengan cepat.

Berdasarkan uraian penelitian diatas metode CRC banyak digunakan untuk data yang memiliki periode yang panjang, dan juga sifat CRC yang mudah diimplementasikan pada *hardware* maupun *software* dibandingkan dengan metode lain seperti *hamming code, reed solomon code*, dll. Hal tersebut yang menjadikan metode CRC memiliki kelebihan dalam segi efektifitas dan efisien dalam implementasi nya dan hal tersebut juga yang mendasari penulis menggunakan CRC sebagai metode untuk melakukan *error detection* pada komunikasi suatu sistem yang menggunakan komunikasi serial Arduino, dikarenakan arduino adalah mikrokontroller yang paling banyak digunakan untuk sistem *embedding* dan *internet of things* yang dimana terdapat pertukaran data informasi didalamnya. Penelitian ini bertujuan menerapkan salah satu dari banyaknya metode *error detection* yaitu *Cyclic Redundancy Check* atau CRC pada komunikasi antar dua mikrokontroller arduino uno sebagai solusi untuk mendeteksi *error* dan menghindari *error* pada komunikasi data antar dua mikrokontroler tersebut. Pada penelitian ini menggunakan polinomial CRC-8 sebagai pemilihan panjang polinomialnya hal ini berdasarkan dari besar data pengiriman serial Arduino Uno yang yang hanya dapat mengirimkan data sebesar 8 bit. Diharapkan penelitian ini dapat menjadi salah satu solusi dan menjadi alternatif untuk mendeteksi *error* atau kesalahan pada komunikasi data digital antar mikrokontroller.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dikemukakan, rumusan masalah pada penelitian ini antara lain:

1. Bagaimana perancangan sistem CRC untuk melakukan *error detection* pada komunikasi serial Arduino?
2. Bagaimana menerapkan metode CRC untuk melakukan *error detection* pada komunikasi serial Arduino?
3. Bagaimana hasil implementasi metode CRC untuk melakukan *error detection* pada komunikasi serial Arduino?

1.3 Tujuan

Tujuan utama dalam penelitian ini adalah untuk mengimplementasikan metode *Cyclic Redundancy Check* (CRC) dan menguji apakah sistem dapat mendeteksi adanya kesalahan pada data yang diterima.

1.4 Manfaat

Dengan dilakukan implementasi *Cyclic Redundancy Check* pada komunikasi serial arduino ini, diharapkan mampu mengamankan dan mendeteksi kesalahan pada data seperti data yang tidak utuh atau yang mengalami pergantian bit yang tidak diinginkan dengan cepat dan mudah.

1.5 Batasan masalah

1. Metode komunikasi data menggunakan komunikasi serial data Arduino Uno.
2. Data masukan melalui antarmuka serial monitor Arduino IDE.
3. Simulasi kesalahan pada data dengan cara menambah bit data sebelum codeword dikirim.

1.6 Sistematika pembahasan

Adapun uraian secara singkat mengenai metodologi penelitian ini yang ada pada masing-masing bab adalah sebagai berikut:

BAB I Pendahuluan

Menjelaskan tentang latar belakang yang mendasari penelitian ini, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika pembahasan dari penelitian ini yang berjudul "Implementasi Error Detection System Pada komunikasi Serial Arduino Menggunakan Metode Cyclic Redundancy Check (CRC)".

BAB II Landasan Kepustakaan

Pada bab ini menjelaskan tentang landasan teori yang digunakan pada penelitian dan pada bab ini juga menjelaskan tentang penelitian serupa yang pernah dilakukan sebelumnya.

BAB III Metode Penelitian

Membahas tentang langkah-langkah kerja yang dilakukan selama penelitian diantaranya studi literatur, analisis kebutuhan sistem, rancangan sistem, dan implementasi dari sistem *Error Detection* menggunakan metode CRC pada komunikasi serial Arduino.

BAB IV Rekayasa Kebutuhan

Bab ini menjelaskan secara rinci yang meliputi deskripsi umum dari sistem, kebutuhan perangkat keras, perangkat lunak, kebutuhan fungsional sistem, dan alur kerja sistem.

BAB V Perancangan dan Implementasi

Bab ini menjelaskan tentang perancangan sistem penelitian serta implementasi sistem penelitian yang berupa perangkat keras dan perangkat lunak.

BAB VI Pengujian dan Analisa

Bab ini menjelaskan tentang pengujian fungsionalitas sistem dan mendapatkan analisa dari hasil pengujian tersebut.

BAB VII Penutup

Bab ini menjelaskan tentang kesimpulan yang diperoleh dari rumusan masalah pada penelitian ini dengan melakukan pengujian dan analisis, juga memuat saran untuk penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Pada penelitian ini penulis mengkaji dari beberapa referensi jurnal dan penelitian yang ditulis Shreya Gawande dan Dr. S. A. Ladhake dari Sant Gadge Baba Amravati University, India. Adapun jurnal dari penelitian sebelumnya yang penulis jadikan referensi dari penilitian yang ditulis oleh Joong-Ho Lee dari Yongin University Korea Selatan. Dan juga jurnal internasional yang ditulis oleh Jonathan R. Engdahl dan Dukki Chung. Sedangkan untuk beberapa referensi dasar teori sebagai pengetahuan tentang komponen dan metode yang digunakan meliputi penelitian ini.

2.1 Tinjauan Pustaka

No	Nama Penulis [Tahun] Judul	Persamaan	Perbedaan	
			Penelitian terdahulu	Rencana penelitian
1	(Gawande & Ladhake, 2014) "*Design and Implementation of Parallel CRC for High Speed Application*"	Menggunakan metode CRC untuk mendeteksi *error*	Menggunakan metode paralel CRC sebagai metode untuk mendeteksi *burst error* pada hardware *Field Programming Gate Array* FPGA	Menerapkan metode CRC pada hardware arduino uno
2	(Engdahl & Chung, 2014) "*Fast Parallel CRC Implementation in Software*"	Menggunakan metode CRC untuk mendeteksi *error*	Menerapkan metode fast parallel CRC dan diimplementasikan pada perangkat lunak atau software	Menerapkan fast CRC pada komunikasi serial Arduino uno
3	(Lee, 2015) "*CRC (Cyclic Redundancy Check) Implementation in High-Speed Semiconductor Memory*"	Menggunakan metode CRC untuk mendeteksi *error*	Menerapkan metode CRC sebagai pendeteksi error pada data pada high-speed semiconductor memory	Menerapkan metode CRC pada hardware arduino uno

2.2 Dasar Teori

Pada subbab dasar teori ini akan dijelaskan referensi dasar dan teori-teori pendukung mengenai sistem komunikasi digital, konsep dasar pengkodean, deteksi *error* dan koreksi *error*, *Cyclic Redundancy Check (CRC)* serta komponen penunjang implementasi penelitian ini seperti mikrokontroller Arduino.

2.2.1 Sistem Komunikasi Digital

Sistem komunikasi digital adalah komunikasi berbasis sinyal digital. sinyal digital adalah data yang berbentuk pulsa dengan besaran 0 dan 1 atau biasa

disebut juga sinyal diskrit. Sistem komunikasi digital mempunyai beberapa elemen utama yaitu:

A. *Source* (Sumber)

Alat untuk membangkitkan data untuk bisa ditransmisikan.

B. *Transmitter* (Pengirim)

Alat ini berfungsi untuk memindahkan dan menandai informasi dengan cara menghasilkan sinyal elektromagnetik yang ditransmisikan ke beberapa sistem transmisi yang berurutan.

C. *Transmission Sistem* (Sistem Transmisi)

Berupa jaringan tunggal yang berfungsi menghubungkan antara sumber (*source*) dan tujuan (*destination*).

D. *Receiver* (Penerima)

Menerima sinyal dan menggabungkannya kedalam bentuk tertentu.

E. *Noise*

Merupakan gangguan yang muncul pada saat transmisi berlangsung. *Noise* juga mempengaruhi kualitas sinyal yang didapatkan atau diterima oleh *Receiver.*

D. *Destination* (Tujuan)

Menangkap data yang telah digabungkan oleh *Receiver.*

2.2.2 Konsep Dasar Pengkodean

Kesalahan dan keamanan merupakan masalah dalam sistem komunikasi digital, karena dapat mengurangi kinerja sistem dan mengurangi mutu dan kualitas dari komunikasi tersebut. Untuk mengatasi masalah tersebut diperlukan adanya metode untuk mengecek dan mengoreksi adanya *error*, sehingga dapat dilakukan metode penanganan *error* dengan pemeriksaan bit, metode pemeriksaan bit ada dua tipe yaitu :

a. *Backward Error Control*

Pada metode ini apabila data yang diterima terjadi *error* maka *receiver* atau penerima akan mengirimkan sinyal kepada *transmitter* atau pengirim untuk melakukan pengiriman ulang data.

b. *Forward Error Control*

Pada metode ini sebelum data dikirimkan, data akan dikodekan dengan pembangkit kode atau enkoder, untuk kemudian dikirimkan ke *receiver* atau penerima. Pada penerima tersebut akan terdapat sebuah dekoder atau penerjemah yang akan mendekodekan data tersebut, apabila terjadi *error* maka akan dilakukan pengkoreksian pada data tersebut

2.2.3 Error Detection

Pada saat transmisi data terdapat kemungkinan data tersebut mengalami *error* atau kegagalan. Transmitter melakukan proses *error detection* dengan melakukan penambahan bit (*parity check bit*) kedalam data yang akan dikirimkan tersebut. Terdapat banyak metode untuk melakukan deteksi *error* contohnya *hamming code, linear feedback shift register, reed solomon code, cyclic redudancy codes*, dll. Metode metode itu mempunyai cara kerja yang berbeda dan keefektifan yang berbeda juga, tergantung penggunaan pada panjang datanya dan fungsinya.

2.2.4 Polinomial

Dalam sistem matematika polinomial adalah suatu ekspresi yang terdiri dari variabel yang juga disebut *indeterminates* dan koefisien yang melibatkan operasi pengurangan, penjumlahan, pembagian, perkalian. Polinomial pada CRC digunakan sebagai *divisor* atau pembagi sehingga mendapatkan *remainder* atau sisa hasil perhitungan.

2.2.5 Remainder

Dalam sistem matematika *remainder* adalah jumlah yang tersisa setelah melakukan suatu perhitungan. Dalam aritmatika *remainder* adalah bilangan bulat yang tersisa setelah melakukan operasi pembagian pada satu bilangan bulat dengan bilangan bulat yang lain. Dalam aljabar *remainder* adalah sisa pembagian antara satu polinomial dengan polinomial yang lain dan dalam CRC *remainder* adalah sisa hasil pembagian antara data yang berupa biner dan polinomial yang juga berupa biner, sisa hasil pembagian ini digunakan untuk menjadi kunci atau *checksum* yang nantinya digabung dengan biner data sehingga menjadi *codeword* dan dikirim ke penerima.

2.2.6 Cyclic Redundancy Check

Cyclic Redundancy Check adalah metode *error detection* yang biasa digunakan dalam jaringan digital dan perangkat penyimpanan untuk mendeteksi perubahan tidak sangaja/ kecelakaan pada data. Blok data masuk ke dalam sistem CRC dan mendapatkan suatu *remainder* untuk blok data tersebut yang berdasarkan pada generator polinomial pada sistem CRC tersebut. Lalu *remainder* tersebut dikirimkan bersamaan dengan blok datanya pada saat penerimaan data akan dilakukan kalkulasi ulang pada data dan cheksum tersebut jika hasil kalkulasi tidak tepat maka data tersebut mengalami pergantian bit atau *error*. Metode CRC sangat populer karena sangat sederhana diimplementasikan pada *binary hardware,* mudah untuk di analisis secara matematis dan sangat bagus untuk mendeteksi kesalahan pada data yang disebabkan oleh noise pada kanal transmisi (Chakravarty, 2001).

Perhitungan matematis untuk mendapatkan *remainder* yaitu pada data yang berbentuk *binary* dilakukan pengoperasian *exclusive* OR atau XOR dan operasi geser jika *Most Significant Bit* pada data bernilai 1 maka akan di lakukan pengoperasian XOR dengan generator polinomial lalu digeser jika MSB data

bernilai 0 maka data akan digeser 1 kali begitu seterusnya sehingga mendapatkan *remainder*/ sisa hasil operasi. Lalu blok data akan dikirim ke penerima bersamaan dengan *remainder* nya.

Jika data dan *remainder* sudah diterima oleh penerima maka akan dilakukan komputasi ulang seperti pengoperasian untuk mendapatkan cheksum, jika hasil komputasi pada penerima mendapatkan hasil selain dari nilai 0 maka data tersebut terdapat pergantian data atau *error*. Jika komputasi mendapatkan nilai 0 maka data tersebut tidak mengalami pergantian bit atau tidak ada kesalahan pada data.

Metode CRC memiliki beberapa representasi polinomial yang digunakan pada pengoperasian, polinomial yang digunakan oleh CRC bermacam-macam bergantung pada penggunaan dan implementasinya. CRC memiliki polinomial yang ditunjukkan pada Tabel 2.1 .

Tabel 2.1 Polinomial CRC

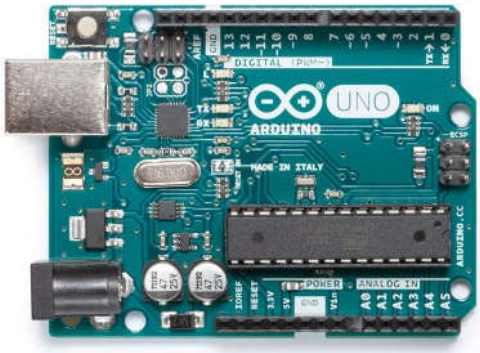
Nama	Digunakan Pada	Polinomial	
CRC-1	Berbagai macam hardware / disebut juga dengan *parity bit*	0x1	X + 1
CRC-3-GSM	Jaringan seluler	0x3	X3+X+1
CRC-5-USB	Paket token USB	0x05	X5+X2+1
CRC-6-GSM	Jaringan seluler	0x2F	X6+X5+X3+X+1
CRC-7	Sistem telekomunikasi, MMC, SD, ITU	0X09	X7+X3+1
CRC-8	*Digital Video Broadcasting Second Generation* (DVB-S2)	0XD5	X8+X5+X3+X2+X+1

CRC-8-Bluetooth	Konektivitas nirkabel	0XA7	
CRC-8-CCIT	HEC, Cell Delineation, ISDN, ATM, ITU-T	0X07	X8+X2+X+1
CRC-8-Dallas/Maxim	1-Wire Bus	0X8C	X8+X5+X4+1
CRC-10	ATM, ITU-T	0X233	X10+X9+X5+X4+X+1
CRC-16-CCIT	HDLC, XMODEM, Bluetooth, DigRF, SD	0X1021	X16+X12+X5+1
CRC-32	ISO/IEC/IEEE 802-3 (Ethernet), SATA, MPEG-2, Gzip, Bzip2, POSIX cksum, PNG,ZMODEM	0x04C11DB7	X32+X26+X23+X22+X16+X12+X11+X10+X8+X7+X5+X4+X2+X+1

Sumber : (https://en.wikipedia.org/wiki/Cyclic_redundancy_check)

2.2.7 Mikrokontroller

Mikrokontroller adalah komputer kecil yang kompleks da dirancang untuk suatu tujuan tertentu dan menjadi pengontrol pada sistem *embedded*. Penelitian ini menggunakan mikrokontroller Arduino Uno R3. Arduino adalah mikrokontroller yang menggunakan ATMEGA328 sebagai mikrokontroller dan memiliki jumlah pin I/O digital sebanyak 14 pin dan pin input analog sebanyak 6 pin. Arduino Uno adalah jenis mikrokontroller Arduino yang paling banyak digunakan dan paling disarankan untuk pemula. Dengan pemrograman cukup menggunakan USB tipe A dan tipe B.



Gambar 2.1 Mikrokontroller Arduino Uno R3

Sumber: (https://store.arduino.cc/usa/arduino-uno-rev3)

Spesifikasi yang dimiliki oleh Arduino Uno R3 ini dijelaskan pada tabel 2.1

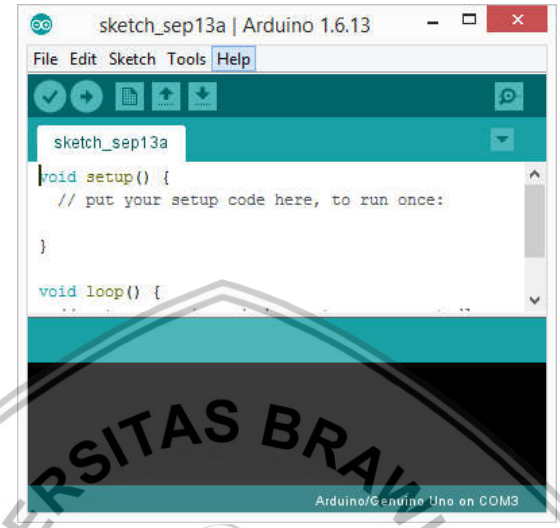
Tabel 2.2 Spesifikasi Arduino Uno R3

Microcontroller	**ATmega328P**
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Sumber: (https://store.arduino.cc/usa/arduino-uno-rev3)

2.2.8 Arduino IDE

Merupakan perangkat lunak penghubung antara user dengan mikrokontroller. Arduino IDE merupakan perangkat lunak yang mendukung bahasa C++ untuk digunakan pada board arduino. Tampilan arduino IDE ditunjukkan pada Gambar 2.3.

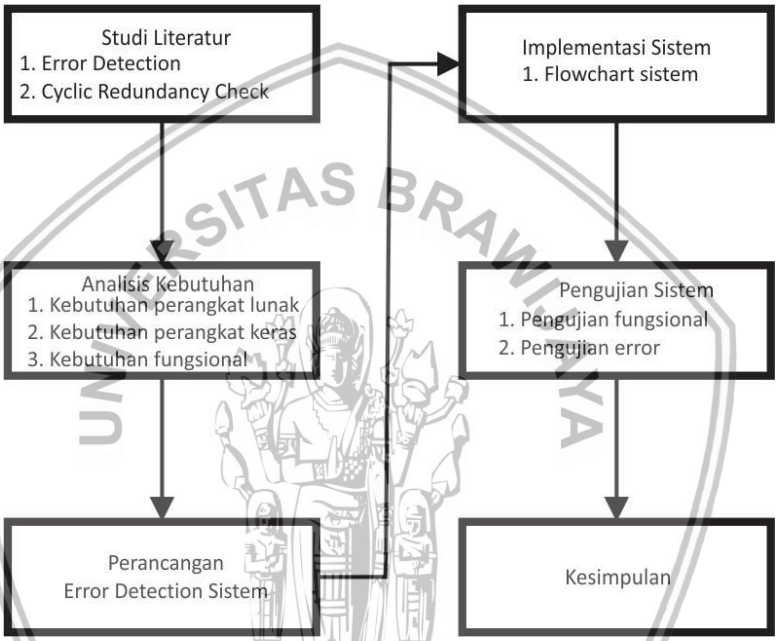


Gambar 2.2 Tampilan Arduino IDE

Terdapat 2 fungsi utama pada pemrograman arduino yaitu fungsi void setup() dan void loop(), dimana void setup() digunakan untuk fungsi yang hanya dijalankan sekali saja sedangkan void loop() digunakan untuk menjalankan proses yang memerlukan perulangan atau looping.

BAB 3 METODOLOGI

Dalam perancangan sebuah sistem penelitian terdapat proses-proses yang harus disusun dengan baik dan terorganisir agar dapat menciptakan perancangan sistem yang terstruktur dan tersusun dengan baik. Penelitian ini diawali dengan studi literatur yang terkait dengan kajian pustaka dan dasar teori. Penelitian ini bersifat implementatif karena mengintegrasikan metode Cyclic Redudancy Check pada komunikasi serial arduino untuk mengatasi kegagalan pada data atau *error*. Pada bab ini menjelaskan tentang alur metode penelitian yang dilakukan untuk pembuatan sistem ini dapat dilihat dalam bentuk diagram alir pada Gambar 3.1.



Gambar 3.1 Alur Penelitian

3.1 Studi Literatur

Studi literatur adalah tahap mencari referensi dan teori dasar yang mendukung sistem yang dirancang sebagai penunjang penelitian. Kajian studi literatur dari beberapa sumber yang dilakukan oleh penulis guna menunjang perancangan sistem pada penelitian ini, baik dari jurnal, paper, dan buku. Teori pustaka yang berkaitan dengan penelitian ini adalah:

1. *Error Detection*

   Studi literatur mengenai *error detection* dilakukan dengan mencari jurnal, paper, dan penjelasan tentang apa itu *error detection* dan apa saja metode metode yang digunakan.

2. *Cyclic Redudancy Check*

   Studi literatur tentang metode CRC dilakukan dengan mengkaji buku, penelitian terkait, jurnal dan penjelasan melalui video tentang cara kerja CRC dan cara implementasinya ke komunikasi Arduino.

3.2 Analisis Kebutuhan

Analisis kebutuhan sangat dibutuhkan dalam perancangan sebuah sistem, hal ini dilakukan dengan cara mengidentifikasi seluruh kebutuhan sistem, identifikasi sistem dilakukan dengan cara mengidentifikasi baik disisi *hardware* maupun *software*. Dengan adanya identifikasi dan analisis kebutuhan sistem ini diharapkan akan mempermudah dalam proses perancangan sistem nanti.

3.2.1 Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras (*hardware*) apa saja yang dibutuhkan sistem pada penelitian ini. Adapun perangkat keras (*hardware*) yang dibutuhkan sistem antara lain:

1. Laptop
2. Arduino Uno R3

3.2.2 Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak (*software*) apa saja yang dibutuhkan sistem pada penelitian ini. Adapun perangkat lunak (*software*) yang dibutuhkan sistem antara lain:

1. Arduino IDE 1.6

3.2.3 Kebutuhan Sistem

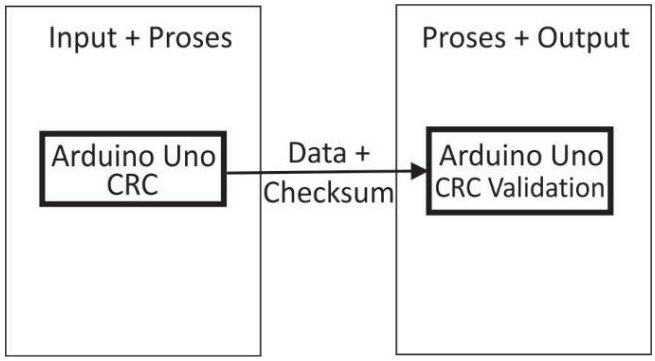
Analisis kebutuhan sistem adalah kebutuhan yang harus dipenuhi suatu sistem untuk mencapai tujuan suatu penelitian, kebutuhan sistem dari penelitian ini antara lain:

1. Arduino mampu mengirim data atau nilai ke penerima.
2. Sistem mampu untuk melakukan komputasi CRC pada sisi pengirim dan mampu melakukan validasi CRC pada sisi penerima dengan baik.
3. Metode CRC mampu mendeteksi jika terdapat *error* maupun tidak ada *error* pada data dengan baik dan tepat pada data dan *remainder* setelah diterima oleh penerima.

3.3 Perancangan Sistem

Tahap perancangan sistem dilakukan setelah tahap analisis kebutuhan selesai dilakukan dan terpenuhi. Tujuan dilakukan perancangan sistem agar implementasi sistem berjalan terstruktur dan sistematis. Adapun desain sistem penelitian ini

yang telah penulis sediakan guna mempermudah dalam perancangan sistem dengan blok diagram pada gambar 3.2.

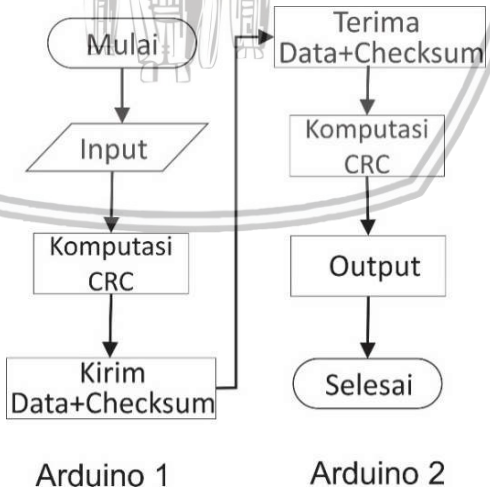


Gambar 3.2 Blok Diagram Sistem

Pada gambar 3.2 Arduino Uno 1 akan memproses data input dari user dengan komputasi menggunakan metode CRC hingga mendapatkan *remainder* setelah itu data dan *remainder* akan dikirim ke Arduino 2 dan jika sudah diterima oleh arduino 2 data dan *remainder* tersebut dilakukan komputasi CRC lagi jika data dan *remainder* tersebut menghasilkan nilai selain 0 maka data tersebut dinyatakan terdapat *error*.

3.4 Implementasi Sistem

Pada tahap implementasi ini dimulai dari merancang rangkaian komunikasi serial pada dua Arduino uno lalu mengimplementasikan *error detection* yaitu metode CRC pada komunikasi itu untuk mendeteksi *error* pada komunikasi data pada dua Arduino itu. Penerapan CRC menggunakan operasi xor operasi geser sehingga mendapatkan *remainder*. Implementasi sistem seperti digambarkan pada Gambar 3.3:



Gambar 3.3 Implementasi Sistem

Pada Gambar 3.3 dimulai dengan memasukkan data input setelah itu data akan dikomputasi sehingga mendapatkan *remainder* lalu data dan *remainder*

tersebut dikirim ke Arduino 2, setelah data dan *remainder* tersebut diterima oleh Arduino 2 akan dilakukan komputasi lagi sekaligus melakukan pengecekan kesalahan atau *error* pada data tersebut. Jika tidak ada kesalahan atau *error* maka data tersebut akan menjadi *output* data dan diterima oleh user.

3.5 Pengujian dan Analisis Sistem

Pengujian dan analisis pada penelitian ini digunakan untuk mengukur seberapa baik dan menguji apakah sistem tersebut sudah berjalan dengan baik dan sesuai dengan apa yang diinginkan. Pengujian meliputi:

1. Pengujian Fungsional

   Menguji apakah sistem komunikasi data serial pada kedua arduino berjalan dengan baik dan tidak ada kendala.

2. Pengujian *Error*

   Menguji apakah sistem dapat mendeteksi *error* dengan baik dan tepat jika diberikan *error* pada data.

3. Pengujian Waktu

   Menguji waktu yang dibutuhkan sistem untuk melakukan enkode, dekode data dan melakukan deteksi kesalahan pada data.

3.6 Kesimpulan

Kesimpulan didapatkan jika telah melakukan perancangan sistem, implementasi dan pengujian serta analisis. Kesimpulan dituliskan setelah melakukan pengujian sistem dan analisis sistem sehingga dapat menjadi acuan untuk ke depannya serta diharapkan juga menjadi acuan untuk penelitian yang lain dan memperbaiki kekurangan-kekurangan yang ada dalam penelitian ini, agar dapat lebih menyempurnakan lagi dan untuk pengembangan sistem selanjutnya.

BAB 4 REKAYASA KEBUTUHAN

Pada bab rekayasa kebutuhan ini berisikan penjelasan mengenai kebutuhan-kebutuhan dari sistem yang dirancang agar dapat memenuhi tujuan yang telah dibuat sebelumnya. Kebutuhan tersebut meliputi gambaran umum sistem, kebutuhan perangkat lunak, kebutuhan perangkat keras dan kebutuhan fungsional. Dengan adanya rekayasa kebutuhan ini diharapkan sistem yang sedang dirancang dapat bekerja dengan baik.

4.1 Gambaran Umum Sistem

Sistem yang akan dibuat yaitu sistem *error detection* menggunakan metode *Cyclic Redundancy Check* yang akan diterapkan pada komunikasi serial Arduino Uno. Arduino sebagai pengirim menerima input dari user dan melakukan komputasi untuk mencari *remainder*-nya. *Remainder* ini digunakan sebagai *checksum* yang akan dikirimkan bersamaan dengan data ke arduino penerima, pada sisi arduino penerima data dan *checksum* tadi dilakukan komputasi ulang untuk mendeteksi ada atau tidaknya kesalahan pada data tersebut.

4.2 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan yang wajib dipenuhi agar sistem dapat berjalan sesuai dengan apa yang diharapkan. Kebutuhan fungsional disini berupa kebutuhan sistem, kebutuhan perangkat keras, dan kebutuhan perangkat lunak.

4.2.1 Kebutuhan Sistem

4.2.1.1 Arduino mampu mengirim data melalui serial

Merupakan kebutuhan fungsional sistem, yaitu Arduino pengirim diharapkan mampu mengirim data ke Arduino penerima sebelum di implementasikan metode *error detection* CRC.

4.2.1.2 Sistem mampu untuk melakukan komputasi dan validasi data dengan baik.

Merupakan kebutuhan fungsional sistem, yaitu sistem diharapkan dapat melakukan komputasi dengan metode ini dengan baik dan melakukan validasi *codeword*.

4.2.1.3 Metode CRC mampu mendeteksi *error* dengan baik dan tepat pada *codeword*.

Merupakan kebutuhan fungsional sistem yaitu setelah diterapkannya metode *error detection* CRC pada komunikasi serial arduino, metode ini dapat mendeteksi *error* atau kesalahan pada data.

4.2.2 Kebutuhan Perangkat Keras

Pada bagian ini menjelaskan terkait kebutuhan perangkat keras apa saja yang digunakan pada sistem ini. Perangkat keras ini meliputi sekumpulan komponen elektronika yang membentuk suatu perangkat baru yang digunakan untuk tujuan tertentu. Berikut adalah kebutuhan perangkat keras yang digunakan untuk implementasi sistem ini, yakni sebagai berikut:

1. Mikrokontroller Arduino Uno
   Arduino Uno adalah sebuah board mikrokontroller yang banyak digunakan untuk *embedding system.* Arduino uno memiliki chip mikroprosesor berbasis ATMega328 yang memungkinkan pengguna untuk melakukan *upload program* atau *sketch* ke mikrokontroller arduino untuk melakukan embedding dan menjalankan fungsi sebagai pengirim dan penerima data melalui komunikasi serial dengan menggunakan pin Tx Rx yang sudah disediakan.

4.2.3 Kebutuhan Perangkat Lunak

Pada bagian ini menjelaskan perangkat lunak atau *software* apa saja yang digunakan untuk implementasi sistem, yakni sebagai berikut:

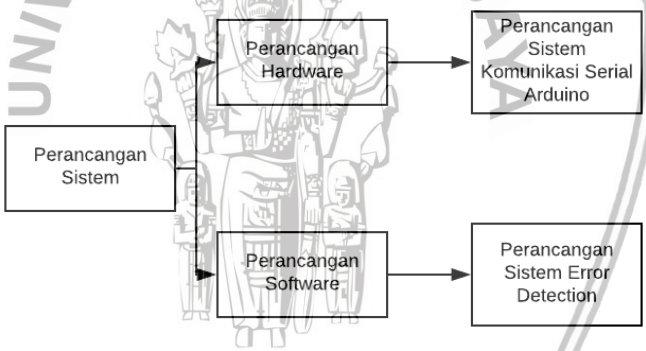
1. Arduino IDE
   IDE merupakan kependekan dari *Integrated Development Environtment* yang secara bahasa dapat diartikan sebagai lingkungan integrasi untuk pengembangan. Arduino menggunakan bahasa pemrograman yang menyerupai bahasa pemrograman C. Bahasa pemrograman Arduino atau *Sketch* dapat memudahkan pemula untuk melakukan pengembangan atau *development*. IC Arduino yaitu ATMega328 sudah diberi program *bootloader* yang menjadi perantara antara *compiler* dengan mikrokontroller. Arduino IDE sendiri dibuat dari bahasa pemrograman Java dan dilengkapi dengan *library* C maupun C++ yang biasa disebut *wiring*, hal ini memudahkan untuk melakukan operasi input dan output.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Setelah tahapan analisis kebutuhan selesai dilakukan tahapan selanjutnya yang dilakukan adalah tahap perancangan dan implementasi. Perancangan dilakukan berdasarkan hasil yang didapatkan dari analisis kebutuhan fungsional dan non fungsional, dan implementasi dilakukan berdasarkan perancangan sistem dari perangkat keras atau *hardware* dan perangkat lunak atau *software*. Pada bab ini perancangan dan implementasi dimulai dari perancangan perangkat keras sistem komunikasi serial arduino dan dilanjutkan dengan perancangan perangkat lunak sistem komunikasi serial dan sistem *error detection* lalu dilakukan implementasi komunikasi serial tersebut pada *hardware* dan dilanjutkan dengan implementasi metode CRC pada serial komunikasi tersebut untuk mendeteksi *error*.

5.1 Perancangan Sistem

Pada perancangan sistem akan dijelaskan mengenai tahapan untuk merancang dan membangun sistem ini, tahapan dimulai dari perancangan *hardware*, lalu dilanjutkan dengan perancangan *software* dimana tahapan perancangan sistem ini digambarkan pada Gambar 5.1.



Gambar 5.1 Diagram Perancangan Sistem

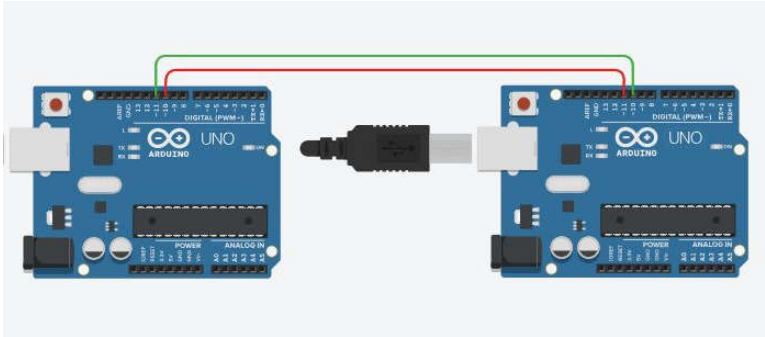
Gambar 5.1 menunjukkan perancangan sistem pada sisi *hardware* dan sisi *software.* Perancangan *hardware* yaitu perancangan sistem komunikasi serial UART antar 2 arduino dan perancangan *software* dilakukan dengan merancang sistem *error detection* pada serial komunikasi antar 2 arduino tersebut

5.1.1 Perancangan Hardware

Perancangan *hardware* merupakan tahapan alur pembuatan sistem pada bagian perangkat keras atau *hardware*. Perancangan *hardware* membutuhkan 2 (dua) buah mikrokontroller arduino yang saling berkomunikasi secara serial dengan menggunakan pin TX dan RX.

Komunikasi serial arduino menggunakan 2 Pin yang sudah *built in* atau yang sudah tersedia pada *board* arduino yaitu pin RX dan TX, pada penelitian ini penulis

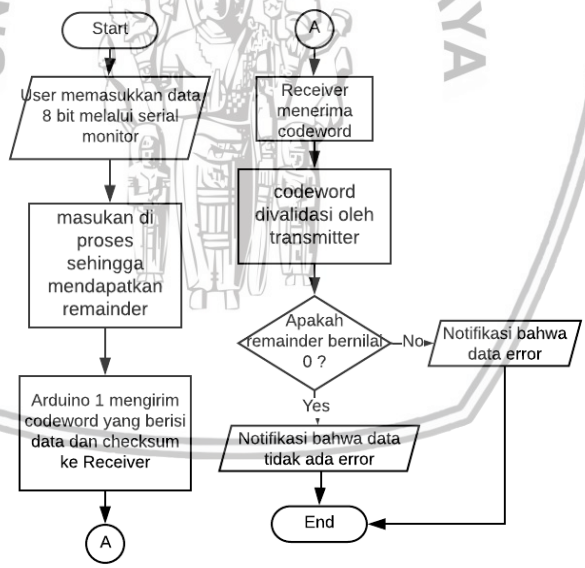
menggunakan pin 10 dan pin 11 sebagai pin RX dan TX, pin 10 arduino master dihubungkan dengan pin 11 arduino slave, dan pin 10 arduino slave disambungkan dengan pin 11 arduino master, seperti yang digambarkan dengan Gambar 5.2.



Gambar 5.2 Skematik sistem komunikasi serial Arduino

5.1.2 Perancangan Software

Perancangan software merupakan tahapan alur pembuatan sistem pada bagian perangkat lunak atau *software*. Perancangan *software* disini merupakan perancangan metode sistem CRC atau *Cyclic Redundancy Check* untuk mendeteksi *error* pada komunikasi serial antara kedua arduino tersebut. Adapun diagram alir perancangan *software* digambarkan pada Gambar 5.3.



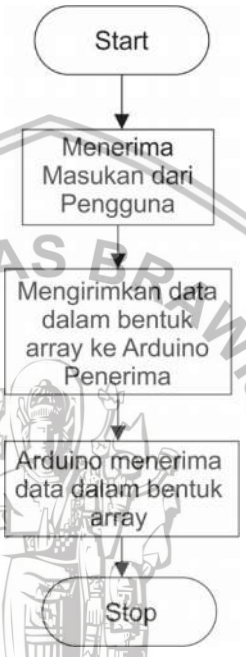
Gambar 5.3 Diagram Alir Perancangan Software Sistem

Pada perancangan *software* dimulai dari memasukkan data pada Arduino pengirim, lalu arduino melakukan komputasi pada data dan mendapatkan *remainder*, selanjutnya *remainder* dan data dikirimkan ke arduino penerima, pada penerima data akan dilakukan komputasi ulang, jika hasil komputasi tersebut

bernilai nilai 0 maka tidak ada kesalahan pada data, jika hasilnya tidak bernilai 0 maka terdapat kesalahan pada data tersebut.

5.1.2.1 Perancangan Software Sistem Komunikasi Serial Arduino

Perancangan sistem komunikasi serial arduino menggunakan library SoftwareSerial.h dengan inisialisasi pin 10 sebagai RX dan pin 11 sebagai TX. Pada perancangan ini menggunakan serial monitor sebagai tempat masukan dari pengguna dan data masukan dari pengguna ini nantinya akan dikirimkan ke arduino penerima melalui komunikasi serial UART, data ini yang nantinya akan di deteksi kesalahannya oleh arduino penerima.



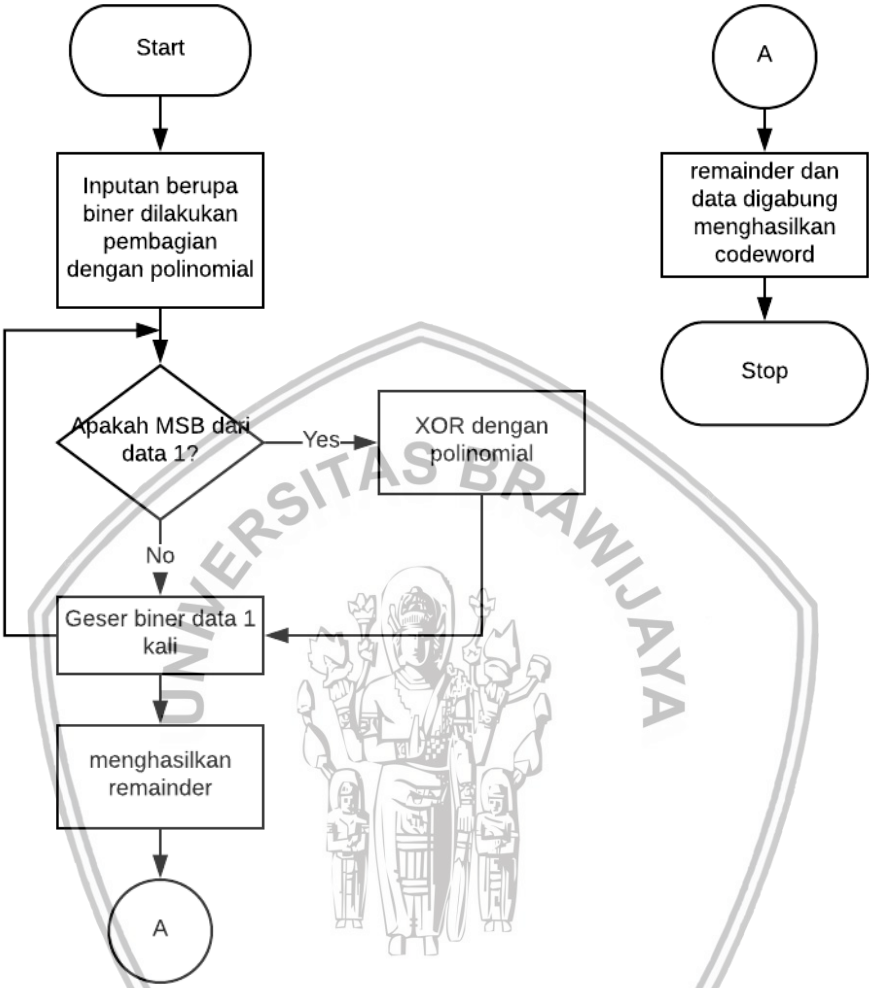
Gambar 5.4 Perancangan Software Komunikasi Serial

Pada Gambar 5.4 menampilkan urutan perancangan *software* komunikasi serial yang dimulai dari arduino pengirim menerima data masukan dari pengguna lalu arduino akan memasukkan data tersebut kedalam *array* dan dikirimkan ke arduino penerima dalam bentuk *array*, pada arduino penerima data diterima dan dibaca dalam bentuk *array* juga lalu data tersebut akan dilakukan pengecekan *error* oleh sistem.

5.1.2.2 Perancangan Software Sistem Error Detection (Cyclic Redundancy Check)

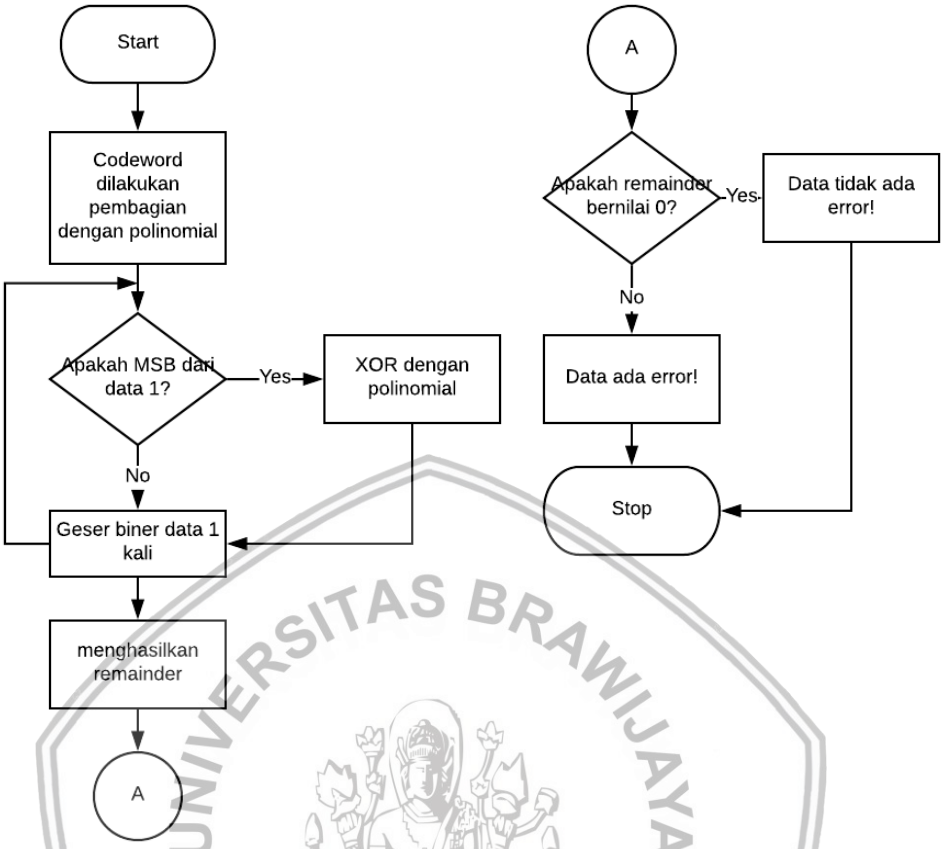
Perancangan sistem CRC berfungsi sebagai pendeteksi kesalahan yang ada pada data yang dikirimkan oleh pengirim, Arduino 1 bertugas sebagai pengirim data dan melakukan operasi pada data sehingga didapatkan *codeword*, dan Arduino 2 berfungsi sebagai penerima dan bertugas sebagai pedeteksi *error* pada data tersebut dengan melakukan operasi metode LFSR dan *polynomial division*. Data tersebut dibagi dengan *generator* polinomial yang dipilih dan sama dengan *polynomial generator* yang digunakan untuk menghasilkan *remainder* pada

Arduino 1. Jika hasil pembagian memiliki nilai selain dari nilai 0 maka data mengalami *error* baik pada saat pengiriman datanya atau pada saat penerimaan datanya. Diagram alir perancangan sistem digambarkan pada Gambar 5.4 dan 5.5.



Gambar 5.5 Perancangan sistem *Error Detection* pada sisi pengirim

Pada Gambar 5.5 merupakan perancangan sistem *error detection* yang dimulai dari arduino pengirim ketika mendapatkan masukan akan dikirimkan ke fungsi dan pada fungsi tersebut data akan diubah menjadi biner dan dilakukan pembagian dengan polinomialnya. Jika biner data tersebut memiliki *most significant bit* (MSB) bernilai 1 maka data tersebut akan dilakukan operasi XOR dengan biner polinomial dan digeser 1 step, jika MSB dari biner data tersebut bernilai 0 maka akan digeser. Proses ini dilakukan berulang ulang sampai data tersebut habis dan menghasilkan sisa hasil bagi atau *remainder*. *Remainder* ini akan dimasukkan ke dalam *array* dan dikirimkan ke arduino penerima bersamaan dengan data yang dimasukkan pengguna.



Gambar 5.6 Perancangan sistem *Error detection* pada sisi penerima

Gambar 5.6 menampilkan perancangan sistem *error detection* pada sisi penerima. Pada saat *codeword* diterima maka akan diubah ke bilangan biner dan dilakukan pembagian polinomial jika biner dari *codeword* tersebut memiliki *most significant bit / MSB* bernilai 1 maka akan dilakukan operasi XOR dengan dengan polinomial dan dilakukan operasi geser, jika MSB bernilai 0 maka hanya akan dilakukan operasi geser. Operasi pembagian ini terus dilakukan hingga biner dari codeword tersebut habis dan menghasilkan sisa hasil bagi atau *remainder*, selanjutnya melakukan operasi perbandingan pada *remainder* tersebut jika *remainder* bernilai 0 maka data tersebut tidak terdapat *error* dan sebaliknya jika *remainder* bernilai selain dari 0 maka data tersebut terjadi *error*.

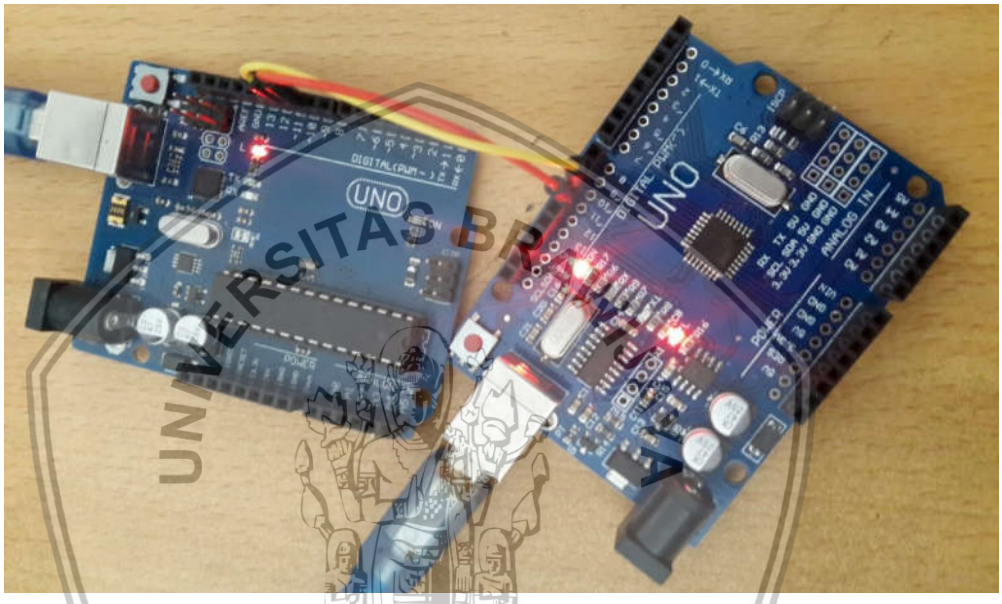
5.2 Implementasi Sistem

Implementasi sistem dilakukan setelah melakukan bab perancangan sistem baik pada sisi perangkat keras sistem maupun pada perangkat lunak sistem pada sub-bab sebelumnya. Pada sub-bab ini akan dijelaskan mengenai implementasi sistem dari sisi *hardware* dan *software* dari sistem ini.

5.2.1 Implementasi Hardware

Implementasi *hardware* mengacu pada perancangan *hardware* pada sub-bab perancangan sebelumnya. Implementasi sistem komunikasi serial dengan menggunakan pin 10 sebagai pin RX dan pin 11 sebagai pin TX pada arduino.

Pada implementasi sistem komunikasi serial arduino menggunakan pin RX dan TX yang ada pada arduino, pin RX pada arduino 1 tersambung pada pin TX arduino 2, dan pin RX arduino 2 tersambung pada arduino 1. Pada sistem yang akan dibuat Arduino 1 sebagai *transmitter* atau pengirim sedangkan arduino 2 sebagai penerima data yang dikirim oleh arduino 1. Hasil implementasi sistem komunikasi serial arduino dapat dilihat pada Gambar 5.6.



Gambar 5.7 Implementasi Serial Komunikasi Arduino

5.2.2 Implementasi Software

Implementasi *software* mengacu pada perancangan *software* pada sub-bab perancangan sebelumnya. Implementasi software mengikuti alur kerja diagram alir yang telah dijelaskan pada sub-bab rancangan dan diimplementasikan melalui kode program dengan menggunakan *software* arduino IDE. Pada sub-bab ini dijelaskan barisan-barisan kode yang digunakan untuk mengimplementasikan sistem ini.

5.2.2.1 Implementasi Software Library

Dalam implementasi kode program dibutuhkan *library* untuk mempermudah peneliti dalam implementasi sistemnya. Dalam penelitian ini penulis menggunakan *library* `SoftwareSerial.h` seperti yang ditunjukkan pada Tabel 5.1 untuk menggunakan pin 10 dan pin 11 sebagai pin rx dan tx.

Tabel 5.1 Kode Sumber Software Library

```
Software Library Pengirim dan Penerima
```

```
1  #include <SPI.h>
2  #include <Wire.h>
3  #include<stdio.h>
4  #include<string.h>
5  #include <SoftwareSerial.h>
```

5.2.2.2 Implementasi Software Pin-Pin dan Variabel

Implementasi *software* pin-pin dan variabel yang digunakan pada penelitian ini. Pin-pin yang digunakan adalah pin 10 dan pin 11 yang berfungsi sebagai pin rx dan tx pada SoftwareSerial, dan juga inisialisasi dari variabel *global* yang dapat digunakan pada fungsi-fungsi lain yang ada pada kode program. Kode inisialisasi pada sisi pengirim ditunjukkan pada Tabel 5.2 dan pada penerima ditunjukkan pada tabel 5.3.

Tabel 5.2 Kode Sumber Pin-pin dan variabel pengirim

```
Software Pin Pengirim
1  SoftwareSerial mySerial(10, 11); //RX, TX
2  byte inp[1];
3  byte arr[2];
4  byte charRead;
5  char inByte = 0;
6  int j = 0;
7  int i = 0;
8  byte poli;
9  char err;
10 unsigned long timers;
```

Pada Tabel 5.2 merupakan kode sumber pada pengirim yang terdapat inisialisasi *library* pada arduino yaitu `SPI.h` untuk melakukan pengujian sistem SPI, *library* `Wire.h` untuk melakukan pengujian sistem I2C, dan `SoftwareSerial.h` yang merupakan library untuk melakukan komunikasi secara serial UART. Pada baris 10 menunjukkan inisialisasi pin yang dipakai untuk melakukan komunikasi serial yaitu pin 10(RX) dan pin 11(TX), pada baris 7 inisilisasi variabel *array* `inp[2]` pada global variabel, 2 pada array tersebut merupakan nilai *index array* tersebut dan *array* ini digunakan untuk menerima data masukan dari pengguna. Pada baris 8 terdapat inisialisasi array untuk pengiriman data pada sistem, dan pada baris 14 adalah inisialisasi variabel polinomial yang akan digunakan oleh sistem.

Tabel 5.3 Kode sumber Software Pin-pin dan variabel penerima

```
Software Pin Penerima
1  SoftwareSerial mySerial(10, 11); //RX, TX
2  byte inp[2];
3  int j = 0;
4  int i = 0;
5  byte poli;
6  volatile byte pos;
7  volatile boolean process;
8  float buff[2];
9  unsigned long timers;
```

Pada tabel 5.3 menampilkan kode sumber inisialisasi pin, *library* dan variabel untuk arduino sebagai penerima. *Library* yang dipakai pada arduino

penerima adalah SPI.h untuk melakukan komunikasi SPI, `Wire.h` untuk melakukan komunikasi I2C, *library* `SoftwareSerial.h` untuk melakukan komunikasi UART. Untuk inisialisasi pin serial UART menggunakan variabel `mySerial` dan pin 10(RX) dan 11(TX). Untuk variabel yang diinisialisasi pada global variabel yaitu variabel array inp[2] yang memiliki *index* 2 digunakan untuk menerima array yang dikirim oleh pengirim dan digunakan untuk masukan pada fungsi *error detection,* lalu ada variabel poli yang digunakan untuk menyimpan variabel polinomial yang dipilih pengguna, selanjutnya variabel `volatile byte pos`, `process` dan `buff` yang digunakan untuk *buffer* array pada komunikasi SPI, dan variabel timer untuk menghitung timer pada pemilihan polinomial.

5.2.2.3 Implementasi Sistem Error Detection (Cyclic Redundancy Check)

Dalam mengimplementasikan kode program pada sistem *error detection* ini penulis melakukan operasi xor dan operasi geser dalam bahasa C++ dan operasi perulangan *for*, jika biner dari *input* memiliki *most significant bit* bernilai 1 maka akan dilakukan operasi xor ke polinomialnya dan jika bernilai 0 maka biner dari data tersebut hanya akan digeser, begitu seterusnya sampai biner dari data tersebut habis dan menyisakan *remainder* atau sisa yang nantinya akan digunakan sebagai *codeword*.

Tabel 5.4 Kode sumber *Sistem Error Detection*

Software Sistem Error Detection pada pengirim dan penerima

```
1  byte hash(const byte *data)
2  {
3      byte crc = *data; /* init crc directly with input byte instead
4  of 0, avoid useless 8 bitshifts until input byte is in crc register
5  */
6      for (int i = 0; i < 8; i++)
7      {
8          if ((crc & 0x80) != 0)
9          { /* most significant bit set, shift crc register and
10 perform XOR operation, taking not-saved 9th set bit into account
11 */
12             crc = (byte)((crc << 1) ^ poli);
13         }
14         else
15         { /* most significant bit not set, go to next bit */
16             crc <<= 1;
17         }
18     }
19     return crc;
20 }
```

Pada Tabel 5.4 menjelaskan tentang kode sumber yang digunakan untuk melakukan operasi LFSR dan pembagian polinomial. Kode sumber ini merupakan sebuah fungsi yang bernama `byte hash` dengan parameter `const byte data` dan `byte len`, parameter `data` adalah parameter untuk data input dan `len` adalah parameter panjang dari data tersebut. Fungsi ini berisi inisialisasi variabel `byte crc` yang bernilai 0x00 dan perulangan `while` dengan kondisi decrement variabel `len`, didalam perulangan `while` terdapat variabel `byte extract` yang bernilai sama dengan data, lalu terdapat 8 kali perulangan `for` yang berisi perulangan `if`

jika `sum` yang bernilai `extract` / data AND 0x01 bernilai 1 maka akan dilakukan operasi XOR data dengan variabel `poli` yang merupakan variabel polinomial lalu digeser 1 bit dan jika `sum` bernilai 0 maka akan dilakukan penggeseran 1 bit. Jika perulangan `for` tersebut selesai maka akan mengembalikan nilai variabel `crc` yang merupakan remainder.

5.2.2.4 *Implementasi Software Sub Program*

Implementasi *software* sub *program* adalah implementasi kode *program* pada bagian fungsi *setup* pada sistem. Pada fungsi setup ini berisikan inisialisasi *baudrate* serial yang digunakan pada sistem ini, disini penulis menggunakan 2 serial yaitu *Serial* dan *mySerial* dari *SoftwareSerial* yang inisialisasi nya pada *baudrate* yang sama yaitu 9600bps. Tabel 5.5 menunjukkan kode implementasi sub *program* pada sisi pengirim. Pada Tabel 5.6 adalah implementasi sub *program* pada sisi penerima.

Tabel 5.5 Kode sumber sub program pengirim

Software Sub Program Pengirim

```
1  void setup() {
2    Serial.begin(9600);
3    mySerial.begin(9600);
4    Serial.setTimeout(50);
5    //Wire.begin();
6    //digitalWrite(SS, HIGH);
7    //SPI.begin();//divide the clock by 8
8    //SPI.setClockDivider(SPI_CLOCK_DIV8);
9    }
```

Pada Tabel 5.5 berisi kode sumber sub program pada sisi pengirim yang berisi inisialisasi baudrate 9600bps dari variabel `serial` dan `mySerial`, selanjutnya inisialisasi serial timeout yang bernilai 50 dan inisialisasi variabel yang digunakan untuk pengujian waktu yaitu `wire`, `digitalwrite`, dan `SPI`.

Tabel 5.6 Kode sumber sub program penerima

Software Sub Program Penerima

```
1  void setup() {
2    //Wire.begin(9);
3    //Wire.onReceive(receiveEvent);
4    Serial.begin(9600);
5    mySerial.begin(9600);
6    //pinMode(MISO,OUTPUT);
7    //SPCR |= _BV(SPE);
8    //SPCR |= _BV(SPIE);
9    //pos = 0;
10   //process = false;
11   }
```

Tabel 5.6 merupakan kode sumber dari sub program penerima yang berisikan inisialisasi *baudrate* untuk komunikasi serial, inisialisasi pin I2C, dan pin SPI yang digunakan untuk pengujian waktu.

5.2.2.5 Implementasi Software Main Program

Pada implementasi *software main program* berisi inisialisasi, pengoperasian dan pemanggilan fungsi-fungsi yang telah didefinisikan terlebih dahulu. Pada sisi penerima yang ditunjukkan pada tabel 5.7 *main program* berisikan inisialisasi nilai variabel *array*, *looping* untuk mengirimkan data dari *array* dan mencetak nilai dari *array* tersebut.

Tabel 5.7 Kode sumber main program pengirim

Software Main Program Pengirim

```
1   void loop() {
2     while (Serial.available() > 0){
3       charRead = Serial.parseInt();
4     }
5     inp[0]=charRead;
6     byte hasil = hash(inp, sizeof(inp));
7     arr[0] = inp[0];
8     arr[1] = hasil;
9     for(int j = 0; j < sizeof(arr); j++)
10    {
11      mySerial.write(arr[j]);
12    }
13    delay(2000);
14  }
```

Kode sumber pada Tabel 5.7 merupakan kode sumber untuk main program pada sisi arduino pengirim yang pada baris pertama terdapat perulangan `while` jika terdapat nilai masukan pada serial monitor maka akan dibaca dan diinisialisasikan dengan variabel `charRead` dan variabel `charRead` ini dimasukkan pada array `inp[0]`, pada baris selanjutnya merupakan pemanggilan fungsi `hash` dengan parameter inp yang merupakan nilai masukan dari serial dan parameter sizeof(inp) yang merupakan fungsi untuk menghitung panjang dari array `inp[]`, lalu nilai array `inp[]` tadi dimasukkan ke dalam array `arr[]` pada index ke 0 dan hasil dari pemanggilan fungsi tersebut dimasukkan pada array `arr[]` pada index ke 1

Pada sisi penerima seperti pada Tabel 5.8 main *program* berisikan *looping* untuk membaca data yang dikirimkan oleh pengirim dan akan dikomputasi oleh *hash* setelah itu dilakukan perbandingan jika hasil komputasi bernilai 0 maka akan mencetak "tidak ada *error* pada data!" pada *serial monitor*, jika hasil tersebut bernilai selain dari 0 maka akan mencetak "terdapat *error* pada data!" pada *serial monitor*.

Tabel 5.8 Kode sumber main program penerima

Software Main Program Penerima

```
1   void loop() {
2     if (mySerial.available() >= 2) {
3     for (i=0; i<2; i++) {
4       inp[i] = mySerial.read();
5     }
6     }
7     for(j = 0; j < sizeof(inp); j++)
8     {
9       Serial.print("data diterima : ");Serial.println(inp[j]);
```

```
10    }
11    byte hasil = hash(inp,sizeof(inp));
12    if(hasil==0){
13      Serial.print("Tidak ada error pada data! ");
14    }
15    else{
16      Serial.print("Terdapat error pada data! ");
17    }
18    Serial.print("hasil validasi data = ");Serial.println(hasil);
19    delay(2000);
20  }
```

Baris pertama pada Tabel 5.8 merupakan perulangan `while` jika terdapat data pada serial sebanyak 2 data maka akan dibaca dan dimasukkan kedalam variabel array `inp[]`. Baris selanjutnya adalah perulangan `for` untuk mencetak data yang dibaca sebelumnya, setelah itu akan dilakukan pemanggilan fungsi `hash` dengan parameter array `inp[]` dan `sizeof(inp)` yang merupakan fungsi untuk menghitung panjang array `inp[]`. Baris selanjutnya merupakan perbandingan `if` jika hasil dari fungsi `hash` tersebut bernilai 0 maka akan mencetak *string* "Tidak ada *error* pada data!" pada serial monitor jika tidak bernilai 0 maka akan mencetak *string* "Terdapat error pada data!".

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab pengujian ini akan dibahas mengenai pengujian dan analisis dari fungsional sistem yang terdiri dari pengujian fungsional komunikasi serial arduino dan pengujian fungsional sistem *error detection* dan pengujian waktu yang dibutuhkan sistem *error detection* untuk melakukan pendeteksian kesalahan.

6.1 Pengujian Fungsional Komunikasi Serial Arduino

Pengujian fungsional komunikasi *serial* arduino merupakan pengujian fungsional yang dilakukan menggunakan komunikasi serial arduino dengan pin 10 dan pin 11 yang digunakan sebagai pin RX, TX.

6.1.1 Tujuan

Pengujian ini bertujuan untuk memeriksa apakah arduino penerima mampu membaca data yang dikirimkan oleh arduino pengirim, data yang dikirim oleh arduino pengirim adalah data *codeword* berupa numerik.

6.1.2 Prosedur Pengujian

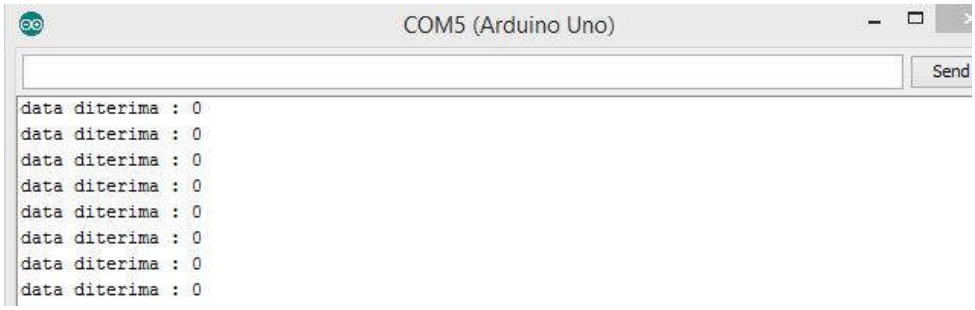
Untuk dapat menguji fungsionalitas komunikasi *serial* arduino dibutuhkan prosedur-prosedur pengujian sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu 2 buah arduino uno dan kabel penghubung atau *jumper male to male*.
2. Menyambungkan pin 10 arduino pengirim dengan pin 11 arduino penerima dengan kabel *jumper*, dan menyambungkan pin 11 arduino pengirim dengan pin 10 arduino penerima.
3. Membuka software Arduino IDE lalu ketik dan *upload source code*.
4. Buka serial monitor pada Arduino IDE lalu ketikkan masukan yang berupa numerik pada serial monitor arduino 1. Buka serial monitor pada arduino 2 dan kita akan melihat data yang telah dimasukkan pada arduino 1 telah diterima oleh arduino 2.

6.1.3 Hasil dan Analisis

Pengujian komunikasi serial arduino dilakukan dengan cara membuka *serial monitor* yang terdapat di IDE Arduino pada setiap arduino seperti pada Gambar 6.1 dan Gambar 6.2.

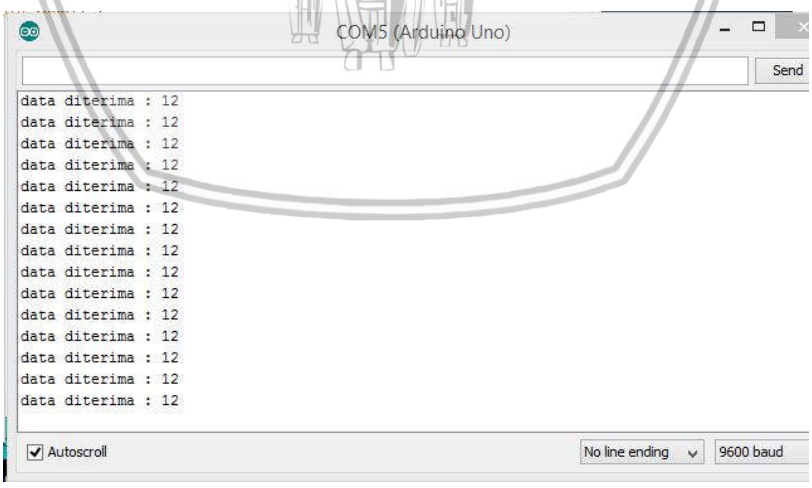
Gambar 6.1 Serial monitor pada arduino pengirim sebelum diberi masukan



Gambar 6.2 Serial monitor pada arduino penerima sebelum diberi masukan

Pada Gambar 6.3 adalah ketika pengguna memasukkan data pada *serial monitor* Arduino pengirim, pada Gambar 6.4 ketika data diterima oleh Arduino penerima.

Gambar 6.3 Serial monitor pada arduino pengirim setelah diberi masukan



Gambar 6.4 Serial monitor pada arduino pengirim sebelum diberi masukan

Pada Gambar 6.3 ditunjukkan bahwa arduino pengirim mengirim data bernilai 12 dan menampilkan data tersebut melalui serial monitor dan pada Gambar 6.4 menunjukkan bahwa arduino penerima dapat menerima data yang bernilai 12 yang mana data tersebut sama dengan data yang dikirim oleh arduino pengirim.

6.2 Pengujian Fungsional Sistem Error Detection

Pengujian fungsional sistem *error detection* menggunakan metode *cyclic redundancy code* (CRC) merupakan pengujian yang dilakukan untuk mengetahui apakah sistem dapat mendeteksi jika ada kesalahan pada data. Pada sub-bab pengujian fungsional sistem *error detection* ini peneliti melakukan 2 langkah pengujian yaitu pengujian dengan data yang tidak ada kesalahan dan pengujian dengan data yang ada kesalahan. Untuk pengujian fungsional sistem *error detection* ini menggunakan polinomial untuk data 8-bit yaitu 0x11D, 0xD5, 0x07, penggunaan dari polinomial untuk data 8-bit ini dikarenakan Arduino Uno untuk komunikasi datanya hanya bisa mengirim data maksimal 10-bit.

6.2.1 Tujuan

Pengujian ini bertujuan untuk memeriksa apakah sistem *error detection* dapat mendeteksi *error* pada data jika diberikan data yang tidak ada kesalahan maupun data yang mengalami kesalahan.

6.2.2 Prosedur Pengujian

Untuk dapat menguji fungsionalitas sistem *error detection* pada komunikasi serial arduino dibutuhkan prosedur-prosedur pengujian sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu 2 buah arduino uno dan kabel penghubung atau *jumper male to male*.
2. Menyambungkan pin 10 arduino pengirim dengan pin 11 arduino penerima dengan kabel *jumper*, dan menyambungkan pin 11 arduino pengirim dengan pin 10 arduino penerima.
3. Membuka software Arduino IDE lalu ketik dan *upload source code*.
4. Buka serial monitor pada Arduino IDE lalu ketikkan masukan yang berupa numerik pada serial monitor arduino 1. Buka serial monitor pada arduino 2 dan kita akan melihat data yang telah dimasukkan pada arduino 1 telah diterima oleh arduino 2.
5. Pada serial monitor terdapat pilihan polinomial dan pilihan untuk mencoba kesalahan pada data atau tidak. Untuk polinomial kedua Arduino harus mempunyai polinomial yang sama.

6.2.3 Hasil dan Analisis

Untuk pengujian pertama kita akan menguji apakah sistem *error detection* dapat mengetahui benar bahwa tidak ada *error* jika kita memasukkan data yang tidak ada kesalahan. Untuk hasil pengujian dapat dilihat pada Gambar 6.5 dan Gambar 6.6.

```
Coba data error ?
1. Tidak
2. Ya
Tidak mencoba data error!
========================================
data : 0
checksum didapatkan : 0
data : 12
checksum didapatkan : 3
```

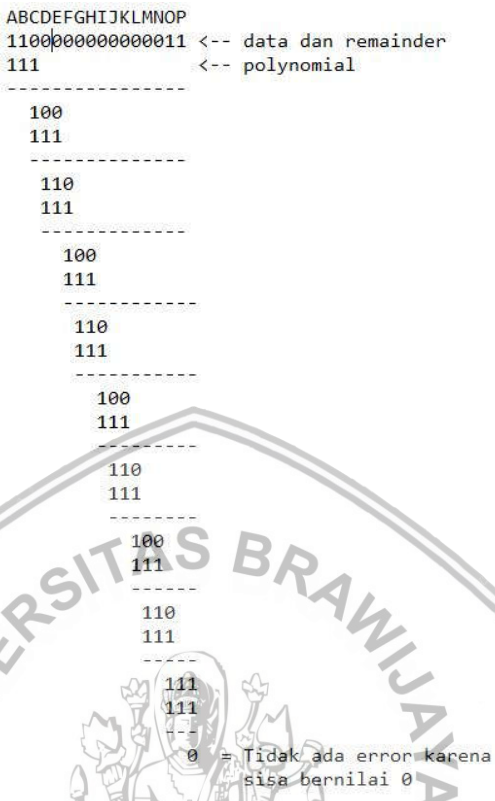
Gambar 6.5 Pengujian dengan tidak ada kesalahan pada arduino pengirim

Pada Gambar 6.5 menunjukkan bahwa ketika arduino penerima menerima masukkan bernilai 12 akan mendapatkan *remainder* atau *checksum* bernilai 3, data yang tercetak pada *serial monitor* arduino pengirim akan otomatis dikirim ke arduino penerima.

Gambar 6.6 Pengujian dengan tidak ada kesalahan pada arduino penerima

Pada Gambar 6.6 menunjukkan bahwa arduino menerima data bernilai 12 dan 3 dan langsung akan dilakukan operasi LFSR dan *polynomial division* dan mendapatkan *remainder* sebesar 0 yang didapat dari data 12 dan 3 dijadikan bentuk biner dan didapatkan biner 1100 dan 11, biner 1100 dilakukan *append* biner sehingga menjadi 8bit biner dan hasilnya 11000000, hal ini dilakukan juga pada biner 11 sehingga kedua data tersebut menjadi 1100000000000011 dan dilakukan operasi pembagian xor dengan polinomial 0x7 yang memiliki biner 111 operasi pembagian dilakukan dengan bagian paling kiri jika *most significant bit* dari data bernilai 1 maka dilakukan xor dengan polinomial jika *most significant bit* bernilai 0 maka akan dilakukan operasi geser 1 bit pada data. Operasi pembagian dikakukan berulang-ulang hingga biner data habis dan menghasilkan sisa hasil bagi atau *remainder* jika remainder bernilai 0 maka data tersebut tidak ada pergantian bit atau kesalahan tetapi jika remainder tersebut bernilai selain dari 0 maka data tersebut mengalami pergantian bit atau terdapat kesalahan. Untuk perhitungan manual dari komputasi metode ini dapat dilihat pada Gambar 6.7,sisa hasil bagi

dari data yang bernilai 0 menunjukkan bahwa pada data tersebut tidak ada kesalahan.



Gambar 6.7 Proses perhitungan deteksi kesalahan dengan tidak ada kesalahan pada data

Tabel 6.1 Tingkat Keakurasian sistem dalam mendeteksi data tidak salah

No percobaan input	Hasil deteksi sistem	Ketepatan sistem
1	Tidak ada kesalahan	Tepat
2	Tidak ada kesalahan	Tepat
3	Tidak ada kesalahan	Tepat
4	Tidak ada kesalahan	Tepat
5	Tidak ada kesalahan	Tepat
6	Tidak ada kesalahan	Tepat
7	Tidak ada kesalahan	Tepat
8	Tidak ada kesalahan	Tepat
9	Tidak ada kesalahan	Tepat
10	Tidak ada kesalahan	Tepat

Persentase keakurasian	100%

Pada Tabel 6.1 menjelaskan tentang tingkat keakurasian dan rasio kesalahan pada sistem pendeteksi yang dilakukan dengan memasukkan 10 data masukan dan menganalisa output sistem dan membandingkan dengan hasil sebenarnya, pada tabel 6.1 sistem dengan tidak ada kesalahan pada data mendapatkan hasil persentase tingkat keakurasian 100 persen, yang didapat dari hasil output – hasil sebenarnya lalu dibagi dengan hasil sebenarnya dan dikalikan 100%.

Untuk pengujian selanjutnya melakukan pengujian dengan adanya kesalahan pada data. Kesalahan yang diuji adalah jika urutan data yang diterima oleh arduino penerima tertukar antara *remainder* dan data, yang hasil dari pengujian ini dapat dilihat pada Gambar 6.8 dan pada Gambar 6.9.

```
2. CRC-8 : 0xD5
3. CRC-8-CCITT : 0x07
 Polynomial digunakan : 0x07

==========================================
Coba data error ?
1. Tidak
2. Ya
Mencoba data error!
==========================================
data : 0
checksum didapatkan : 0
data : 12
checksum didapatkan : 3
```

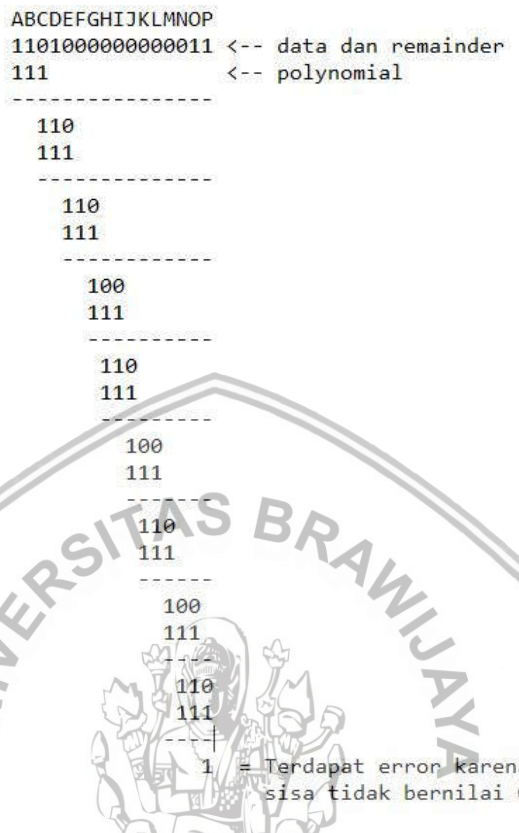
Gambar 6.8 Pengujian dengan adanya kesalahan pada arduino pengirim

```
Pilih polinomialnya :
1. CRC-8-Dallas/Maxim : 0x31
2. CRC-8 : 0xD5
3. CRC-8-CCITT : 0x07
 Polynomial digunakan : 0x07
data diterima : 13
data diterima : 3
Terdapat error pada data! hasil validasi data = 1
data diterima : 13
data diterima : 3
Terdapat error pada data! hasil validasi data = 1
```

Gambar 6.9 Pengujian dengan adanya kesalahan pada arduino penerima

Pada Gambar 6.8 menampilkan jika pengguna memasukkan nilai 12 dan mendapatkan *remainder* 3 dan otomatis dikirim ke arduino penerima, tetapi pada arduino penerima yang ditunjukkan oleh Gambar 6.8 data tersebut mengalami kesalahan perubahan bit yang seharusnya 12 menjadi 13, maka akan terbaca

adanya kesalahan pada data yang diterima tersebut. Proses perhitungan sehingga sistem dapat mendeteksi adanya error pada data tersebut ditunjukkan pada Gambar 6.10.



Gambar 6.10 Proses perhitungan deteksi kesalahan dengan adanya kesalahan pada data

Pada Gambar 6.10 adalah proses perhitungan manual sistem pendeteksi kesalahan sehingga mengetahui bahwa data yang diterima tersebut terdapat kesalahan. Sistem melakukan operasi *linear feedback shift register* dan pembagian polinomial hingga data habis dan mendapatkan sisa hasil bagi atau *remainder* bernilai 1 sehingga data tersebut dinyatakan terdapat kesalahan.

Tabel 6.2 Tingkat Keakurasian sistem dalam mendeteksi data salah

No percobaan input	Hasil deteksi sistem	Ketepatan sistem
1	Ada kesalahan	Tepat
2	Ada kesalahan	Tepat
3	Ada kesalahan	Tepat
4	Ada kesalahan	Tepat
5	Ada kesalahan	Tepat
6	Ada kesalahan	Tepat

7	Ada kesalahan	Tepat
8	Ada kesalahan	Tepat
9	Ada kesalahan	Tepat
10	Ada kesalahan	Tepat
Persentase keakurasian		100%

Pada Tabel 6.2 menjelaskan tentang tingkat keakurasian dan rasio kesalahan pada sistem pendeteksi yang dilakukan dengan memasukkan 10 data masukan dan menganalisa output sistem dan membandingkan dengan hasil sebenarnya. Sedangkan pada Tabel 6.1 sistem dengan adanya kesalahan pada data mendapatkan hasil persentase tingkat keakurasian 100 persen, yang didapat dari hasil output – hasil sebenarnya lalu dibagi dengan hasil sebenarnya dan dikalikan 100%.

6.3 Pengujian Waktu Sistem

Pengujian waktu *delay* yang dibutuhkan sistem pada *interface* protokol komunikasi seperti I2C (*Inter-grated Circuit)*, SPI (*Serial Paralel Interface)*, dan UART (*Universal Asynchrounous Receiver-Transmitter)*. Pengujian dilakukan pada sistem yang menggunakan kesalahan pada data dan tidak ada kesalahan pada data.

6.3.1 Tujuan

Melakukan pengujian pada sistem dan menghitung waktu yang dibutuhkan sistem *error detection* terhadap komunikasi *interface* yang digunakan seperti I2C, UART, dan SPI. Juga melakukan analisis apakah terdapat perbedaan waktu jika menggunakan polinomial yang berbeda terhadap komunikasi *interface* tersebut.

6.3.2 Prosedur Pengujian

Untuk dapat menguji waktu yang dibutuhkan oleh sistem *error detection* pada komunikasi serial arduino dibutuhkan prosedur-prosedur pengujian sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu 2 buah arduino uno dan kabel penghubung atau *jumper male to male*.
2. Untuk UART menyambungkan pin 10 arduino pengirim dengan pin 11 arduino penerima dengan kabel *jumper*, dan menyambungkan pin 11 arduino pengirim dengan pin 10 arduino penerima. Untuk sistem I2C menggunakan pin A4 *master* tersambung dengan pin A4 *slave* dan pin A5 *master* tersambung dengan pin A5 *slave*, lalu menyambungkan *pin ground master* dengan *pin ground slave.* Untuk sistem dengan SPI menggunakan pin 13 *master* dengan pin 13 *slave*, pin 12 *master* dengan pin 12 *slave*, pin 11 *master* dengan pin 11 *slave*, dan pin 10 *master* dengan pin 10 *slave*, juga menyambungkan antar pin *ground master* dengan pin *ground slave* dan pin vcc *master* dengan pin vcc *slave*.

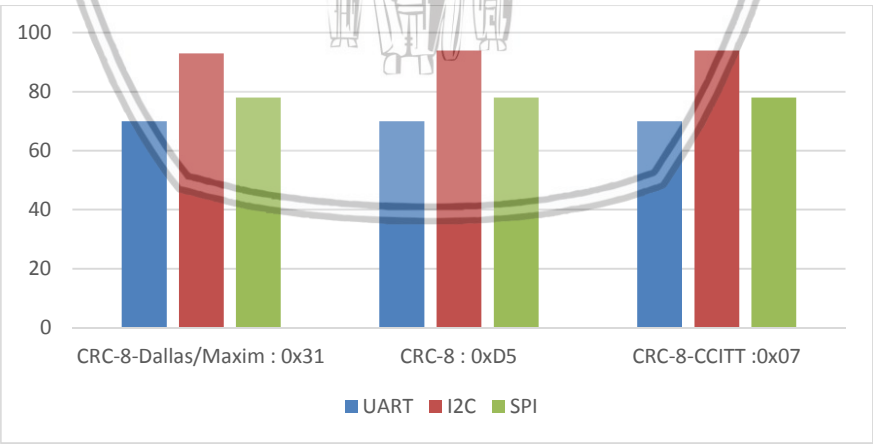
3. Membuka software Arduino IDE lalu ketik dan *upload source code*.
4. Buka serial monitor pada Arduino IDE lalu ketikkan masukan yang berupa numerik pada serial monitor arduino 1. Buka serial monitor pada arduino 2 dan kita akan melihat data yang telah dimasukkan pada arduino 1 telah diterima oleh arduino 2.
5. Pada serial monitor terdapat pilihan polinomial dan pilihan untuk mencoba kesalahan pada data atau tidak. Untuk polinomial kedua Arduino harus mempunyai polinomial yang sama.

6.3.3 Hasil dan Analisis

Pengujian waktu dilakukan dengan menghitung waktu yang dibutuhkan oleh sistem dengan fungsi millis() pada arduino, fungsi millis() pada arduino pengirim diletakkan pada saat data dimasukkan hingga data tersebut dikirim. Pada sisi penerima fungsi millis() diletakkan pada saat sebelum menerima data sampai data tersebut dideteksi. Hasil pengujian waktu *delay* antar *interface* dapat dilihat pada Tabel 6.3:

Tabel 6.3 Waktu delay interface

Polinomial	UART (ms)	I2C (ms)	SPI (ms)
CRC-8-Dallas/Maxim : 0x31	70	93	78
CRC-8 : 0xD5	70	94	78
CRC-8-CCITT :0x07	70	94	78



Gambar 6.11 Grafik Pengujian Waktu

Tabel 6.3 menunjukkan hasil pengujian waktu sistem yang diperlukan sistem ini dengan protokol komunikasi UART, I2C, dan SPI. Waktu ini didapat dari penjumlahan waktu pada sisi pengirim, waktu pengiriman, dan waktu saat data

tersebut dideteksi kesalahannya. Pada pengujian waktu sistem didapatkan hasil waktu UART dengan 3 polinomial yang digunakan memiliki waktu yang sama yaitu membutuhkan waktu 70ms. Untuk protokol I2C membutuhkan waktu untuk polinomial 0x31 yaitu selama 93ms, dan untuk polinomial yang lain membutuhkan waktu yang sama yaitu 94ms. Selanjutnya hasil pengujian waktu yang dibutuhkan protokol SPI untuk melakukan fungsi sistem yaitu selama 78ms untuk 3 polinomial yang berbeda. Dari hasil pengujian waktu tersebut didapatkan grafik seperti pada Gambar 6.11 dan dilakukan hasil analisis bahwa protol UART memiliki waktu sistem yang lebih cepat dari 2 protokol lainnya yaitu sebesar 70ms, dan protokol I2C memiliki waktu yang lebih lama dari 2 protokol lainnya.

BAB 7 PENUTUP

Pada bab 7 ini berisi kesimpulan dan saran dari penelitian ini yang berjudul "Implementasi Error Detection System Pada Komunikasi Serial Arduino Menggunakan Metode Cyclic Redundancy Check (CRC)". Kesimpulan berisi rangkuman dari keseluruhan penelitian yang berisi poin-poin penting pada penelitian ini. Lalu pada saran berisi tentang saran untuk penelitian-penelitian lain yang ingin mengembangkan penelitian ini menjadi yang lebih baik.

7.1 Kesimpulan

Kesimpulan ini diambil dari hasil perancangan, implementasi, dan pengujian sistem. Pada penelitian ini diambil kesimpulan sebagai berikut:

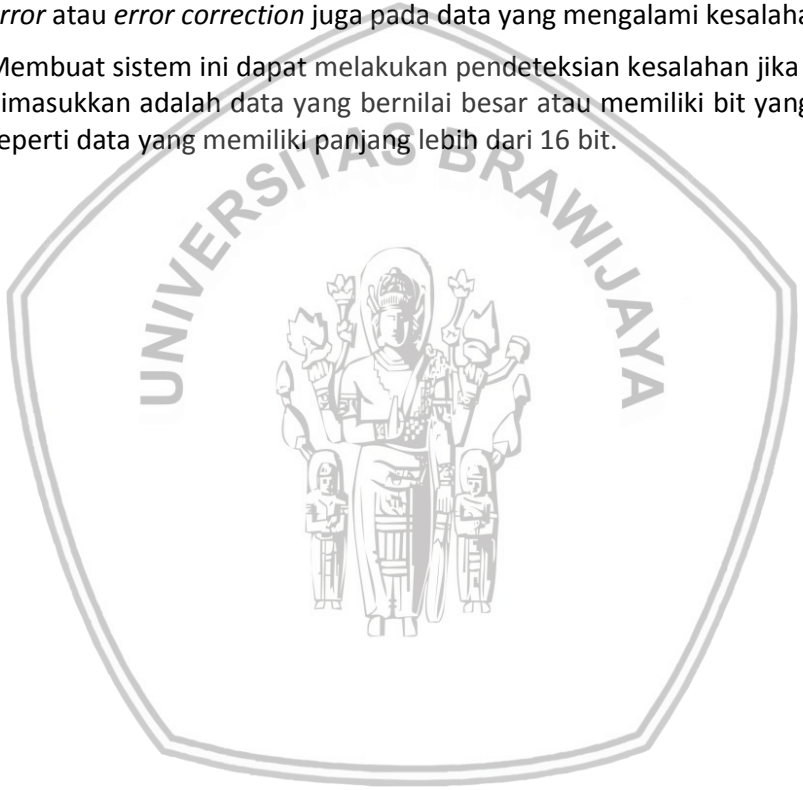
1. Sistem *error detection* dengan menggunakan metode *Cyclic Redundancy Code* ini diimplementasikan pada komunikasi serial antara 2 arduino melalui pin RX TX.
2. Kode program sistem *error detection* dibuat dengan bahasa pemrograman c++, yaitu bahasa pemrograman yang didukung oleh *board* arduino. Kode program ini berisikan fungsi yang berisi komputasi input pengguna agar menjadi *remainder*, dan main program yang berisi pembacaan masukan dari pengguna dan pengiriman *codeword*.
3. Sistem *error detection* menggunakan metode CRC memiliki cara kerja yakni melakukan operasi XOR dan geser untuk mendapatkan *remainder*nya, jika biner data memiliki *most significant bit* (MSB) bernilai 1 maka akan dilakukan operasi XOR dengan polinomial nya, jika biner data memiliki MSB bernilai 0 maka akan dilakukan operasi geser, operasi ini dilakukan terus menerus sampai biner data habis dan meninggalkan sisa atau *remainder.* Jika sudah didapatkan *remainder*, maka biner *remainder* tersebut akan digabungkan dengan biner data dan menjadi *codeword*, dan *codeword* inilah yang akan dikirimkan kepada penerima. Pada sisi penerima akan dilakukan komputasi yang sama seperti pada sisi pengirim hingga mendapatkan *remainder*, dan jika *remainder* tersebut bernilai 0 maka tidak ada *error* pada data, sebaliknya jika selain dari nilai 0 maka data tersebut mengalami kesalahan atau *error*.
4. Pada saat pengujian sistem *error detection* dengan metode *cyclic redundancy check* dapat mendeteksi adanya kesalahan pada data seperti data tersebut mengalami pergantian bit dengan cara melakukan metode LFSR dan pembagian polinomial seperti pada saat mencari remainder pada data, tingkat keakurasian sistem untuk mendeteksi mendapatkan persentase 100% untuk 10 data masukan yang diuji.
5. Pada saat pengujian waktu menggunakan variabel polinomial dan waktu enkode dan dekode, polinomial yang digunakan yaitu CRC-8-Maxim, CRC-8, dan CRC-8-CCIT. Dari hasil analisa pengujian waktu didapatkan bahwa sistem dengan menggunakan protokol UART membutuhkan waktu yang lebih cepat

daripada Sistem dengan SPI atau I2C, dan sistem dengan protokol I2C membutuhkan waktu yang lebih lama untuk melakukan deteksi kesalahan daripada protokol UART dan SPI.

7.2 Saran

Berdasarkan hasil penelitian ini, peneliti memberikan saran kepada peneliti-peneliti lain yang akan mengembangkan sistem ini agar sistem dapat berkembang lebih baik lagi, adapun saran yang diberikan sebagai berikut:

1. Sistem *error detection* menggunakan Cyclic Redundancy Code (CRC) dapat diimplementasikan pada sistem *internet of things* sebagai pendeteksi *error* dan mengurangi kesalahan yang terjadi pada data.
2. Membuat sistem *error detection* dengan CRC ini dapat melakukan koreksi *error* atau *error correction* juga pada data yang mengalami kesalahan.
3. Membuat sistem ini dapat melakukan pendeteksian kesalahan jika data yang dimasukkan adalah data yang bernilai besar atau memiliki bit yang panjang, seperti data yang memiliki panjang lebih dari 16 bit.



DAFTAR PUSTAKA

Arduino, 2018. *Arduino.* [Online] Available at: http://www.arduino.cc

Bogart Jr, T. F., 1992. *Introduction to Digital Circuits. International Edition..* s.l.:s.n.

Chakravarty, T., 2001. Performance of Cyclic Redundancy Codes for Embedded Networks.

Engdahl, J. R. & Chung, D., 2014. Fast Parallel CRC Implementation in Software.

Fu, B. & Ampadu, P., 2009. *On Hamming Product Codes With Type-II Hybrid ARQ for On-Chip.* s.l.:s.n.

Gawande, S. & Ladhake, S. A., 2014. Design and Implementation of Parallel CRC for High Speed Application.

Lee, J.-H., 2015. CRC(Cyclic Redundancy Check) Implementation in High-Speed Semiconductor Memory.

Shannon, C. E., 1948. *A Mathematical Theory of Communication. The Bell System Technical Journal.* s.l.:s.n.

Triyogo, A., 2013. *PENGEMBANGAN ALGORITMA ENKRIPSI DEKRIPSI BERBASIS LFSR MENGGUNAKAN POLINOMIAL PRIMITIF.* s.l.:s.n.

LAMPIRAN A

A.1 Kode Sumber Pada Sisi Pengirim

Kode Sumber Pengirim

```
#include <SPI.h>
#include <Wire.h>
#include<stdio.h>
#include<string.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(10, 11); //RX, TX
byte inp[1];
byte arr[2];
byte charRead;
char inByte = 0;
int j = 0;
int i = 0;
byte poli;
char err;
unsigned long timers;


void setup() {
  Serial.begin(9600);
  mySerial.begin(9600);
  //Wire.begin();
  //digitalWrite(SS, HIGH);
  //SPI.begin();//divide the clock by 8
  //SPI.setClockDivider(SPI_CLOCK_DIV8);
  Serial.setTimeout(50);
  Serial.println("=======================================");
  Serial.println("Pilih polinomialnya :");
  Serial.println("1. CRC-8-Custom : 0x11D");
  Serial.println("2. CRC-8 : 0xD5");
  Serial.println("3. CRC-8-CCITT : 0x07");
  timers = millis();
  while(millis()-timers<10000){
         if(Serial.available()){
           char temp = Serial.read();
         switch(temp){
                case '1' :
                poli = 0x11D;
         Serial.println(" Polynomial digunakan : 0x11D");
                break;
                case '2' :
                poli = 0xd5;
         Serial.println(" Polynomial digunakan : 0xd5");
                break;
                case '3' :
                poli = 0x07;
         Serial.println(" Polynomial digunakan : 0x07");
                break;
                default :
                poli = 0x8c;
                break;
                }
          break;
```

```
    }
  }
  Serial.println(" ");
  Serial.println("=======================================");
  Serial.println("Coba data error ?");
  Serial.println("1. Tidak");
  Serial.println("2. Ya");
  while(millis()-timers<10000){
         if(Serial.available()){
            char tempo = Serial.read();
         switch(tempo){
                 case '1' :
                 err = '1';
         Serial.println("Tidak mencoba data error!");

Serial.println("=======================================");
                 break;
                 case '2' :
                 err = '2';
         Serial.println("Mencoba data error!");

Serial.println("=======================================");
                 break;
                 default :
                 err = '1';
                 break;
                 }
          break;
    }
  }
}
byte hash(const byte *data) ;

void loop() {
  //uint32_t ts1 = millis();
  //Wire.beginTransmission(9);
  //digitalWrite(SS, LOW);
  while (Serial.available() > 0){
    charRead = Serial.parseInt();
  }
  inp[0]=charRead;
  byte hasil = hash(inp);
  arr[0] = inp[0];
  arr[1] = hasil;
  // print hasil
  for(int i = 0; i < sizeof(inp); i++)
  {
    Serial.print("data : ");Serial.println(inp[i]);
  }
//kirim data error dan tidak//
  switch(err){
    case '1':
    for(int j = 0; j < sizeof(arr); j++)
    {
      //SPI.transfer(arr, 2);
      //Wire.write(arr, 2);
      mySerial.write(arr[j]);
    }
    break;
```

```
    case '2':
    arr[0] += 1;
    for(int j = 0; j < sizeof(arr); j++)
    {
      //SPI.transfer(arr, 2);
      //Wire.write(arr, 2);
      mySerial.write(arr[j]);
    }
    break;
    default:
    for(int j = 0; j < sizeof(arr); j++)
    {
      //SPI.transfer(arr, 2);
      //Wire.write(arr, 2);
      mySerial.write(arr[j]);
    }
    break;
  }

  Serial.print("checksum didapatkan : ");Serial.println(hasil);
  //uint32_t ts2 = millis();
  //digitalWrite(SS, HIGH);
  delay(2000);
  //Wire.endTransmission();
  //Serial.println(" Waktu : ");Serial.print(ts2-ts1);
}


byte hash(const byte *data)
{
    byte crc = *data; /* init crc directly with input byte instead
of 0, avoid useless 8 bitshifts until input byte is in crc
register */
    for (int i = 0; i < 8; i++)
    {
        if ((crc & 0x80) != 0)
        { /* most significant bit set, shift crc register and
perform XOR operation, taking not-saved 9th set bit into account
*/
            crc = (byte)((crc << 1) ^ poli);
        }
        else
        { /* most significant bit not set, go to next bit */
            crc <<= 1;
        }
    }
    return crc;
}
```

A.2 Kode Sumber Pada Sisi Penerima

Kode Sumber Penerima

```
#include <SPI.h>
#include <Wire.h>
#include<stdio.h>
#include<string.h>
#include <SoftwareSerial.h>
```

```
#include <Utility.h>

SoftwareSerial mySerial(10, 11); //RX, TX
byte inp[2];
int j = 0;
int i = 0;
byte poli;
volatile byte pos;
volatile boolean process;
float buff[2];
uint32_t ts1;
uint32_t ts2;
unsigned long timers;

void setup() {
  //Wire.begin(9);
  //Wire.onReceive(receiveEvent);
  Serial.begin(9600);
  mySerial.begin(9600);
  //pinMode(MISO,OUTPUT);
  //SPCR |= _BV(SPE);
  //SPCR |= _BV(SPIE);
  //pos = 0;
  //process = false;
  Serial.println("Pilih polinomialnya :");
  Serial.println("1. CRC-8-Custom : 0x11D");
  Serial.println("2. CRC-8 : 0xD5");
  Serial.println("3. CRC-8-CCITT : 0x07");
  timers = millis();
  while(millis()-timers<5000){
    if(Serial.available()){
      char temp = Serial.read();
      switch(temp){
        case '1' :
        poli = 0x11D;
        Serial.println(" Polynomial digunakan : 0x11D");
        break;
        case '2' :
        poli = 0xd5;
        Serial.println(" Polynomial digunakan : 0xd5");
        break;
        case '3' :
        poli = 0x07;
        Serial.println(" Polynomial digunakan : 0x07");
        break;
        default :
        poli = 0x8c;
        break;
      }
      break;
    }
  }
}


byte hash(const byte *data, byte len);

//ISR (SPI_STC_vect) // SPI interrupt routine
//{
```

```
//byte gathered = SPDR;
// if( pos < sizeof inp)
//    {
//      inp[pos++] = gathered;
//    }
//    else
//    process = true;
//}

//void receiveEvent(int howMany) {
//  for(howMany; howMany > 0; howMany--){
//    inp[howMany - 1] = Wire.read();
// }
//}

void loop(void) {
  //uint32_t ts1 = millis();
  //if( process ){
  if (mySerial.available() >= 2) {
  for (i=0; i<2; i++) {
    inp[i] = mySerial.read();
  }
  }
  for(j = 0; j < sizeof(inp); j++)
  {
    Serial.print("data diterima : ");Serial.println(inp[j]);
  }
  byte hasil = hash(inp, sizeof(inp));
  if(hasil==0){
    Serial.print("Tidak ada error pada data!");
  }
  else{
    Serial.print("Terdapat error pada data! ");
  }
  Serial.print("hasil validasi data = ");Serial.println(hasil);
  //uint32_t ts2 = millis();
  delay(2000);
  //inp[pos] = 0;
  //pos= 0; //reset button to zero
  //process = false;
  //Serial.println(" Waktu : ");Serial.println(ts2-ts1);
  //}
}
byte hash(const byte *data, byte len) {
  byte crc = 0x00;
  while (len--) {
    crc ^= *data++;
    for (byte tempI = 8; tempI; tempI--) {
      if (( crc & 0x80) !=0) {
        crc = (byte)((crc<<1) ^ poli);
      }
      else
      crc <<= 1;
    }
  }
  return crc;
}
```

BAB 1 PENDAHULUAN

1.1 Latar belakang

Pada era perkembangan teknologi saat ini, semua bidang telah banyak mengalami perkembangan yang sangat pesat. Salah satu bidang yang mengalami perkembangan secara pesat adalah bidang komunikasi. Komunikasi pada teknologi saat ini tidak hanya dapat dilakukan secara *wired*, tetapi juga dapat dilakukan secara *wireless*. Tetapi dengan komunikasi seperti ini tidak menutup kemungkinan pengiriman informasi atau data yang diterima tidak terkirim dengan benar dan bahkan bisa belum sampai terkirim, atau bisa dikatakan data yang dikirimkan tersebut mengalami *error*.

Banyak faktor yang mempengaruhi pengiriman data tersebut menjadi gagal, misal faktor cuaca, *noise*, radiasi, tegangan listrik tidak stabil, *cross talk* (Fu, et al., 2009). Gangguan yang tidak diinginkan (*noise*) dapat terjadi di saluran komunikasi dan menyebabkan informasi yang diterima menjadi berbeda dari informasi asli dikirim (Shannon, 1948). Data biner yang rusak dapat berubah dari 1 menjadi 0 atau sebaliknya 0 menjadi 1 (Bogart Jr, 1992). Oleh karena itu keberhasilan pengiriman data dengan benar dan akurat merupakan salah satu faktor keberhasilan dalam komunikasi data. Untuk menangani masalah pengiriman data *error* terdapat beberapa metode yaitu *Error Detection* menggunakan enkode dan dekode data yang dikirim. Metode *Error Detection* ada beberapa metode yaitu Hamming Code, Cyclic Redudancy Code, Linear Feedback Shift Register, Reed Solomon, dll.

Cyclic Redundancy Check adalah salah satu teknik *error detection* dengan menggunakan teknik generator polinomial yang dapat menghasilkan bit untuk pengecekan data atau yang biasa disebut *checksum* dan mudah diaplikasikan ke dalam berbagai hal. Hal itulah yang membuat penulis menggunakan metode CRC untuk penelitian ini, karena dirasa metode ini tepat untuk data yang panjang, dan cocok untuk mendeteksi *burst error* pada perangkat keras (*hardware*) dan perangkat lunak (*software*).

Banyak penelitian terdahulu yang menggunakan teknik CRC sebagai metode untuk melakukan pengecekan *error* pada hardware seperti penelitian implementasi CRC yang didesain paralel untuk melakukan pendeteksian *error* dengan cepat yang dilakukan Gawande(2014) pada penelitian tersebut peneliti melakukan desain dan implementasi metode CRC paralel untuk pengaplikasian pada *random access memory* (RAM) dan perangkat penyimpanan atau *storage devices*. Adapun penelitian yang melakukan pendeteksian *error* pada *memory* oleh Lee(2015) yang menerapkan metode CRC untuk melakukan pengecekan data pada komunikasinya, penelitian ini menggunakan *Cyclic Redundancy Check* untuk mendeteksi *burst error* pada memori semikonduktor cepat. Penelitian untuk melakukan pendeteksian *error* pada *software* atau perangkat lunak yang dilakukan oleh Engdahl(2014) ini merancang dan melakukan implementasi metode paralel CRC untuk mendeteksi kesalahan data perangkat lunak dengan cepat.

Berdasarkan uraian penelitian diatas metode CRC banyak digunakan untuk data yang memiliki periode yang panjang, dan juga sifat CRC yang mudah diimplementasikan pada *hardware* maupun *software* dibandingkan dengan metode lain seperti *hamming code, reed solomon code*, dll. Hal tersebut yang menjadikan metode CRC memiliki kelebihan dalam segi efektifitas dan efisien dalam implementasi nya dan hal tersebut juga yang mendasari penulis menggunakan CRC sebagai metode untuk melakukan *error detection* pada komunikasi suatu sistem yang menggunakan komunikasi serial Arduino, dikarenakan arduino adalah mikrokontroller yang paling banyak digunakan untuk sistem *embedding* dan *internet of things* yang dimana terdapat pertukaran data informasi didalamnya. Penelitian ini bertujuan menerapkan salah satu dari banyaknya metode *error detection* yaitu *Cyclic Redundancy Check* atau CRC pada komunikasi antar dua mikrokontroller arduino uno sebagai solusi untuk mendeteksi *error* dan menghindari *error* pada komunikasi data antar dua mikrokontroler tersebut. Pada penelitian ini menggunakan polinomial CRC-8 sebagai pemilihan panjang polinomialnya hal ini berdasarkan dari besar data pengiriman serial Arduino Uno yang yang hanya dapat mengirimkan data sebesar 8 bit. Diharapkan penelitian ini dapat menjadi salah satu solusi dan menjadi alternatif untuk mendeteksi *error* atau kesalahan pada komunikasi data digital antar mikrokontroller.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dikemukakan, rumusan masalah pada penelitian ini antara lain:

1. Bagaimana perancangan sistem CRC untuk melakukan *error detection* pada komunikasi serial Arduino?
2. Bagaimana menerapkan metode CRC untuk melakukan *error detection* pada komunikasi serial Arduino?
3. Bagaimana hasil implementasi metode CRC untuk melakukan *error detection* pada komunikasi serial Arduino?

1.3 Tujuan

Tujuan utama dalam penelitian ini adalah untuk mengimplementasikan metode *Cyclic Redundancy Check* (CRC) dan menguji apakah sistem dapat mendeteksi adanya kesalahan pada data yang diterima.

1.4 Manfaat

Dengan dilakukan implementasi *Cyclic Redundancy Check* pada komunikasi serial arduino ini, diharapkan mampu mengamankan dan mendeteksi kesalahan pada data seperti data yang tidak utuh atau yang mengalami pergantian bit yang tidak diinginkan dengan cepat dan mudah.

1.5 Batasan masalah

1. Metode komunikasi data menggunakan komunikasi serial data Arduino Uno.
2. Data masukan melalui antarmuka serial monitor Arduino IDE.
3. Simulasi kesalahan pada data dengan cara menambah bit data sebelum codeword dikirim.

1.6 Sistematika pembahasan

Adapun uraian secara singkat mengenai metodologi penelitian ini yang ada pada masing-masing bab adalah sebagai berikut:

BAB I Pendahuluan

Menjelaskan tentang latar belakang yang mendasari penelitian ini, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika pembahasan dari penelitian ini yang berjudul "Implementasi Error Detection System Pada komunikasi Serial Arduino Menggunakan Metode Cyclic Redundancy Check (CRC)".

BAB II Landasan Kepustakaan

Pada bab ini menjelaskan tentang landasan teori yang digunakan pada penelitian dan pada bab ini juga menjelaskan tentang penelitian serupa yang pernah dilakukan sebelumnya.

BAB III Metode Penelitian

Membahas tentang langkah-langkah kerja yang dilakukan selama penelitian diantaranya studi literatur, analisis kebutuhan sistem, rancangan sistem, dan implementasi dari sistem *Error Detection* menggunakan metode CRC pada komunikasi serial Arduino.

BAB IV Rekayasa Kebutuhan

Bab ini menjelaskan secara rinci yang meliputi deskripsi umum dari sistem, kebutuhan perangkat keras, perangkat lunak, kebutuhan fungsional sistem, dan alur kerja sistem.

BAB V Perancangan dan Implementasi

Bab ini menjelaskan tentang perancangan sistem penelitian serta implementasi sistem penelitian yang berupa perangkat keras dan perangkat lunak.

BAB VI Pengujian dan Analisa

Bab ini menjelaskan tentang pengujian fungsionalitas sistem dan mendapatkan analisa dari hasil pengujian tersebut.

BAB VII Penutup

Bab ini menjelaskan tentang kesimpulan yang diperoleh dari rumusan masalah pada penelitian ini dengan melakukan pengujian dan analisis, juga memuat saran untuk penelitian selanjutnya.

ABSTRAK

Masalah utama dalam sistem komunikasi data yaitu adanya kesalahan data yang menyebabkan data tersebut diterima tidak sebagaimana mestinya, adanya *error* pada data tersebut dapat disebabkan oleh *noise*, faktor cuaca, *crosstalk* dan adanya tegangan listrik yang tidak stabil. Adanya kesalahan pada data ini dapat dikurangi dan dihindari dengan menerapkan sistem pendeteksi kesalahan atau *error detection* seperti metode *Hamming Code*, *Reed Solomon Code*, dan *Cyclic Redundancy Check.* Mikrokontroller Arduino terutama Uno adalah salah satu mikrokontroller yang banyak digunakan untuk *embedded system* dan *internet of things* yang keduanya tidak bisa dijauhkan dari pertukaran data dan informasi. Metode pada penelitian ini menggunakan metode *Cyclic Redundancy Check* yang yang diimplementasikan pada sistem komunikasi serial antar arduino. Metode ini merupakan satu dari banyak metode untuk mendeteksi *error* atau kesalahan pada data yang banyak digunakan dan diimplementasikan pada perangkat penyimpanan dan sistem komunikasi. CRC sangat mudah diimplementasikan pada perangkat keras dan perangkat lunak dengan menggunakan operasi penjumlahan dan penggeseran bit yang sederhana dengan polinomialnya. Pada penelitian ini sistem *error detection* dengan metode CRC diimplementasikan pada serial komunikasi dua arduino uno yang salah satu arduino uno bertugas sebagai pengirim data dan satunya bertugas sebagai penerima data. Dari pengujian fungsionalitas sistem error detection didapatkan bahwa sistem dapat mendeteksi adanya kesalahan pada data dengan tepat dengan persentase 100% dan pada pengujian waktu didapatkan hasil bahwa sistem dengan komunikasi UART membutuhkan waktu yang lebih cepat daripada sistem dengan SPI atau I2C.

Kata kunci: CRC, *Error Detection*, Arduino, Komunikasi Serial

ABSTRACT

The main problem in the data communication system is the data error that causes the data received is not as it should, the error on the data can be caused by noise, weather factors, crosstalk and the presence of unstable voltage. An error in these data can be reduced and avoided by applying a system of detection errors or error detection methods such as Reed Solomon Code, Hamming Code, and Cyclic Redundancy Check. Arduino microcontroller especially Uno is one of the widely used mikrokontroller for embedded systems and internet of things that both of them could not be kept away from the exchange of data and information. Research on methods using Cyclic Redundancy Check methods that are implemented on the serial communication system between the arduino. This method is one of many methods to detect errors or errors in the data that are widely used and implemented in the storage device and communication system. CRC is quite easy to implement in hardware and software by using simple addition and bit shift operation with its polynomial. In this research system error detection with CRC method implemented in serial communication two arduino uno one of arduino uno served as sender of data and the only one duty as data recipient. From functionality test of the system error detection is resulted that the system can detect an error in the data correctly with a percentage of 100%, and at time test obtained the results that the system with UART communication takes a faster time than a system with SPI or I2C.

Keyword: CRC, *Error Detection*, Arduino, *Serial Communication*

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB 1 PENDAHULUAN

1.1 Latar belakang

Pada era perkembangan teknologi saat ini, semua bidang telah banyak mengalami perkembangan yang sangat pesat. Salah satu bidang yang mengalami perkembangan secara pesat adalah bidang komunikasi. Komunikasi pada teknologi saat ini tidak hanya dapat dilakukan secara *wired*, tetapi juga dapat dilakukan secara *wireless*. Tetapi dengan komunikasi seperti ini tidak menutup kemungkinan pengiriman informasi atau data yang diterima tidak terkirim dengan benar dan bahkan bisa belum sampai terkirim, atau bisa dikatakan data yang dikirimkan tersebut mengalami *error*.

Banyak faktor yang mempengaruhi pengiriman data tersebut menjadi gagal, misal faktor cuaca, *noise*, radiasi, tegangan listrik tidak stabil, *cross talk* (Fu, et al., 2009). Gangguan yang tidak diinginkan (*noise*) dapat terjadi di saluran komunikasi dan menyebabkan informasi yang diterima menjadi berbeda dari informasi asli dikirim (Shannon, 1948). Data biner yang rusak dapat berubah dari 1 menjadi 0 atau sebaliknya 0 menjadi 1 (Bogart Jr, 1992). Oleh karena itu keberhasilan pengiriman data dengan benar dan akurat merupakan salah satu faktor keberhasilan dalam komunikasi data. Untuk menangani masalah pengiriman data *error* terdapat beberapa metode yaitu *Error Detection* menggunakan enkode dan dekode data yang dikirim. Metode *Error Detection* ada beberapa metode yaitu Hamming Code, Cyclic Redudancy Code, Linear Feedback Shift Register, Reed Solomon, dll.

Cyclic Redundancy Check adalah salah satu teknik *error detection* dengan menggunakan teknik generator polinomial yang dapat menghasilkan bit untuk pengecekan data atau yang biasa disebut *checksum* dan mudah diaplikasikan ke dalam berbagai hal. Hal itulah yang membuat penulis menggunakan metode CRC untuk penelitian ini, karena dirasa metode ini tepat untuk data yang panjang, dan cocok untuk mendeteksi *burst error* pada perangkat keras (*hardware*) dan perangkat lunak (*software*).

Banyak penelitian terdahulu yang menggunakan teknik CRC sebagai metode untuk melakukan pengecekan *error* pada hardware seperti penelitian implementasi CRC yang didesain paralel untuk melakukan pendeteksian *error* dengan cepat yang dilakukan Gawande(2014) pada penelitian tersebut peneliti melakukan desain dan implementasi metode CRC paralel untuk pengaplikasian pada *random access memory* (RAM) dan perangkat penyimpanan atau *storage devices*. Adapun penelitian yang melakukan pendeteksian *error* pada *memory* oleh Lee(2015) yang menerapkan metode CRC untuk melakukan pengecekan data pada komunikasinya, penelitian ini menggunakan *Cyclic Redundancy Check* untuk mendeteksi *burst error* pada memori semikonduktor cepat. Penelitian untuk melakukan pendeteksian *error* pada *software* atau perangkat lunak yang dilakukan oleh Engdahl(2014) ini merancang dan melakukan implementasi metode paralel CRC untuk mendeteksi kesalahan data perangkat lunak dengan cepat.

Berdasarkan uraian penelitian diatas metode CRC banyak digunakan untuk data yang memiliki periode yang panjang, dan juga sifat CRC yang mudah diimplementasikan pada *hardware* maupun *software* dibandingkan dengan metode lain seperti *hamming code, reed solomon code*, dll. Hal tersebut yang menjadikan metode CRC memiliki kelebihan dalam segi efektifitas dan efisien dalam implementasi nya dan hal tersebut juga yang mendasari penulis menggunakan CRC sebagai metode untuk melakukan *error detection* pada komunikasi suatu sistem yang menggunakan komunikasi serial Arduino, dikarenakan arduino adalah mikrokontroller yang paling banyak digunakan untuk sistem *embedding* dan *internet of things* yang dimana terdapat pertukaran data informasi didalamnya. Penelitian ini bertujuan menerapkan salah satu dari banyaknya metode *error detection* yaitu *Cyclic Redundancy Check* atau CRC pada komunikasi antar dua mikrokontroller arduino uno sebagai solusi untuk mendeteksi *error* dan menghindari *error* pada komunikasi data antar dua mikrokontroler tersebut. Pada penelitian ini menggunakan polinomial CRC-8 sebagai pemilihan panjang polinomialnya hal ini berdasarkan dari besar data pengiriman serial Arduino Uno yang yang hanya dapat mengirimkan data sebesar 8 bit. Diharapkan penelitian ini dapat menjadi salah satu solusi dan menjadi alternatif untuk mendeteksi *error* atau kesalahan pada komunikasi data digital antar mikrokontroller.

1.2 Rumusan masalah

Berdasarkan latar belakang yang telah dikemukakan, rumusan masalah pada penelitian ini antara lain:

1. Bagaimana perancangan sistem CRC untuk melakukan *error detection* pada komunikasi serial Arduino?
2. Bagaimana menerapkan metode CRC untuk melakukan *error detection* pada komunikasi serial Arduino?
3. Bagaimana hasil implementasi metode CRC untuk melakukan *error detection* pada komunikasi serial Arduino?

1.3 Tujuan

Tujuan utama dalam penelitian ini adalah untuk mengimplementasikan metode *Cyclic Redundancy Check* (CRC) dan menguji apakah sistem dapat mendeteksi adanya kesalahan pada data yang diterima.

1.4 Manfaat

Dengan dilakukan implementasi *Cyclic Redundancy Check* pada komunikasi serial arduino ini, diharapkan mampu mengamankan dan mendeteksi kesalahan pada data seperti data yang tidak utuh atau yang mengalami pergantian bit yang tidak diinginkan dengan cepat dan mudah.

1.5 Batasan masalah

1. Metode komunikasi data menggunakan komunikasi serial data Arduino Uno.
2. Data masukan melalui antarmuka serial monitor Arduino IDE.
3. Simulasi kesalahan pada data dengan cara menambah bit data sebelum codeword dikirim.

1.6 Sistematika pembahasan

Adapun uraian secara singkat mengenai metodologi penelitian ini yang ada pada masing-masing bab adalah sebagai berikut:

BAB I Pendahuluan

Menjelaskan tentang latar belakang yang mendasari penelitian ini, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika pembahasan dari penelitian ini yang berjudul "Implementasi Error Detection System Pada komunikasi Serial Arduino Menggunakan Metode Cyclic Redundancy Check (CRC)".

BAB II Landasan Kepustakaan

Pada bab ini menjelaskan tentang landasan teori yang digunakan pada penelitian dan pada bab ini juga menjelaskan tentang penelitian serupa yang pernah dilakukan sebelumnya.

BAB III Metode Penelitian

Membahas tentang langkah-langkah kerja yang dilakukan selama penelitian diantaranya studi literatur, analisis kebutuhan sistem, rancangan sistem, dan implementasi dari sistem *Error Detection* menggunakan metode CRC pada komunikasi serial Arduino.

BAB IV Rekayasa Kebutuhan

Bab ini menjelaskan secara rinci yang meliputi deskripsi umum dari sistem, kebutuhan perangkat keras, perangkat lunak, kebutuhan fungsional sistem, dan alur kerja sistem.

BAB V Perancangan dan Implementasi

Bab ini menjelaskan tentang perancangan sistem penelitian serta implementasi sistem penelitian yang berupa perangkat keras dan perangkat lunak.

BAB VI Pengujian dan Analisa

Bab ini menjelaskan tentang pengujian fungsionalitas sistem dan mendapatkan analisa dari hasil pengujian tersebut.

BAB VII Penutup

Bab ini menjelaskan tentang kesimpulan yang diperoleh dari rumusan masalah pada penelitian ini dengan melakukan pengujian dan analisis, juga memuat saran untuk penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Pada penelitian ini penulis mengkaji dari beberapa referensi jurnal dan penelitian yang ditulis Shreya Gawande dan Dr. S. A. Ladhake dari Sant Gadge Baba Amravati University, India. Adapun jurnal dari penelitian sebelumnya yang penulis jadikan referensi dari penilitian yang ditulis oleh Joong-Ho Lee dari Yongin University Korea Selatan. Dan juga jurnal internasional yang ditulis oleh Jonathan R. Engdahl dan Dukki Chung. Sedangkan untuk beberapa referensi dasar teori sebagai pengetahuan tentang komponen dan metode yang digunakan meliputi penelitian ini.

2.1 Tinjauan Pustaka

No	Nama Penulis [Tahun] Judul	Persamaan	Perbedaan	
			Penelitian terdahulu	Rencana penelitian
1	(Gawande & Ladhake, 2014) "*Design and Implementation of Parallel CRC for High Speed Application*"	Menggunakan metode CRC untuk mendeteksi *error*	Menggunakan metode paralel CRC sebagai metode untuk mendeteksi *burst error* pada hardware *Field Programming Gate Array* FPGA	Menerapkan metode CRC pada hardware arduino uno
2	(Engdahl & Chung, 2014) "*Fast Parallel CRC Implementation in Software*"	Menggunakan metode CRC untuk mendeteksi *error*	Menerapkan metode fast parallel CRC dan diimplementasikan pada perangkat lunak atau software	Menerapkan fast CRC pada komunikasi serial Arduino uno
3	(Lee, 2015) "*CRC (Cyclic Redundancy Check) Implementation in High-Speed Semiconductor Memory*"	Menggunakan metode CRC untuk mendeteksi *error*	Menerapkan metode CRC sebagai pendeteksi error pada data pada high-speed semiconductor memory	Menerapkan metode CRC pada hardware arduino uno

2.2 Dasar Teori

Pada subbab dasar teori ini akan dijelaskan referensi dasar dan teori-teori pendukung mengenai sistem komunikasi digital, konsep dasar pengkodean, deteksi *error* dan koreksi *error*, *Cyclic Redundancy Check (CRC)* serta komponen penunjang implementasi penelitian ini seperti mikrokontroller Arduino.

2.2.1 Sistem Komunikasi Digital

Sistem komunikasi digital adalah komunikasi berbasis sinyal digital. sinyal digital adalah data yang berbentuk pulsa dengan besaran 0 dan 1 atau biasa

disebut juga sinyal diskrit. Sistem komunikasi digital mempunyai beberapa elemen utama yaitu:

A. *Source* (Sumber)

Alat untuk membangkitkan data untuk bisa ditransmisikan.

B. *Transmitter* (Pengirim)

Alat ini berfungsi untuk memindahkan dan menandai informasi dengan cara menghasilkan sinyal elektromagnetik yang ditransmisikan ke beberapa sistem transmisi yang berurutan.

C. *Transmission Sistem* (Sistem Transmisi)

Berupa jaringan tunggal yang berfungsi menghubungkan antara sumber (*source*) dan tujuan (*destination*).

D. *Receiver* (Penerima)

Menerima sinyal dan menggabungkannya kedalam bentuk tertentu.

E. *Noise*

Merupakan gangguan yang muncul pada saat transmisi berlangsung. *Noise* juga mempengaruhi kualitas sinyal yang didapatkan atau diterima oleh *Receiver.*

D. *Destination* (Tujuan)

Menangkap data yang telah digabungkan oleh *Receiver.*

2.2.2 Konsep Dasar Pengkodean

Kesalahan dan keamanan merupakan masalah dalam sistem komunikasi digital, karena dapat mengurangi kinerja sistem dan mengurangi mutu dan kualitas dari komunikasi tersebut. Untuk mengatasi masalah tersebut diperlukan adanya metode untuk mengecek dan mengoreksi adanya *error*, sehingga dapat dilakukan metode penanganan *error* dengan pemeriksaan bit, metode pemeriksaan bit ada dua tipe yaitu :

a. *Backward Error Control*

Pada metode ini apabila data yang diterima terjadi *error* maka *receiver* atau penerima akan mengirimkan sinyal kepada *transmitter* atau pengirim untuk melakukan pengiriman ulang data.

b. *Forward Error Control*

Pada metode ini sebelum data dikirimkan, data akan dikodekan dengan pembangkit kode atau enkoder, untuk kemudian dikirimkan ke *receiver* atau penerima. Pada penerima tersebut akan terdapat sebuah dekoder atau penerjemah yang akan mendekodekan data tersebut, apabila terjadi *error* maka akan dilakukan pengkoreksian pada data tersebut

2.2.3 Error Detection

Pada saat transmisi data terdapat kemungkinan data tersebut mengalami *error* atau kegagalan. Transmitter melakukan proses *error detection* dengan melakukan penambahan bit (*parity check bit*) kedalam data yang akan dikirimkan tersebut. Terdapat banyak metode untuk melakukan deteksi *error* contohnya *hamming code, linear feedback shift register, reed solomon code, cyclic redudancy codes*, dll. Metode metode itu mempunyai cara kerja yang berbeda dan keefektifan yang berbeda juga, tergantung penggunaan pada panjang datanya dan fungsinya.

2.2.4 Polinomial

Dalam sistem matematika polinomial adalah suatu ekspresi yang terdiri dari variabel yang juga disebut *indeterminates* dan koefisien yang melibatkan operasi pengurangan, penjumlahan, pembagian, perkalian. Polinomial pada CRC digunakan sebagai *divisor* atau pembagi sehingga mendapatkan *remainder* atau sisa hasil perhitungan.

2.2.5 Remainder

Dalam sistem matematika *remainder* adalah jumlah yang tersisa setelah melakukan suatu perhitungan. Dalam aritmatika *remainder* adalah bilangan bulat yang tersisa setelah melakukan operasi pembagian pada satu bilangan bulat dengan bilangan bulat yang lain. Dalam aljabar *remainder* adalah sisa pembagian antara satu polinomial dengan polinomial yang lain dan dalam CRC *remainder* adalah sisa hasil pembagian antara data yang berupa biner dan polinomial yang juga berupa biner, sisa hasil pembagian ini digunakan untuk menjadi kunci atau *checksum* yang nantinya digabung dengan biner data sehingga menjadi *codeword* dan dikirim ke penerima.

2.2.6 Cyclic Redundancy Check

Cyclic Redundancy Check adalah metode *error detection* yang biasa digunakan dalam jaringan digital dan perangkat penyimpanan untuk mendeteksi perubahan tidak sangaja/ kecelakaan pada data. Blok data masuk ke dalam sistem CRC dan mendapatkan suatu *remainder* untuk blok data tersebut yang berdasarkan pada generator polinomial pada sistem CRC tersebut. Lalu *remainder* tersebut dikirimkan bersamaan dengan blok datanya pada saat penerimaan data akan dilakukan kalkulasi ulang pada data dan cheksum tersebut jika hasil kalkulasi tidak tepat maka data tersebut mengalami pergantian bit atau *error*. Metode CRC sangat populer karena sangat sederhana diimplementasikan pada *binary hardware,* mudah untuk di analisis secara matematis dan sangat bagus untuk mendeteksi kesalahan pada data yang disebabkan oleh noise pada kanal transmisi (Chakravarty, 2001).

Perhitungan matematis untuk mendapatkan *remainder* yaitu pada data yang berbentuk *binary* dilakukan pengoperasian *exclusive* OR atau XOR dan operasi geser jika *Most Significant Bit* pada data bernilai 1 maka akan di lakukan pengoperasian XOR dengan generator polinomial lalu digeser jika MSB data

bernilai 0 maka data akan digeser 1 kali begitu seterusnya sehingga mendapatkan *remainder*/ sisa hasil operasi. Lalu blok data akan dikirim ke penerima bersamaan dengan *remainder* nya.

Jika data dan *remainder* sudah diterima oleh penerima maka akan dilakukan komputasi ulang seperti pengoperasian untuk mendapatkan cheksum, jika hasil komputasi pada penerima mendapatkan hasil selain dari nilai 0 maka data tersebut terdapat pergantian data atau *error*. Jika komputasi mendapatkan nilai 0 maka data tersebut tidak mengalami pergantian bit atau tidak ada kesalahan pada data.

Metode CRC memiliki beberapa representasi polinomial yang digunakan pada pengoperasian, polinomial yang digunakan oleh CRC bermacam-macam bergantung pada penggunaan dan implementasinya. CRC memiliki polinomial yang ditunjukkan pada Tabel 2.1 .

Tabel 2.1 Polinomial CRC

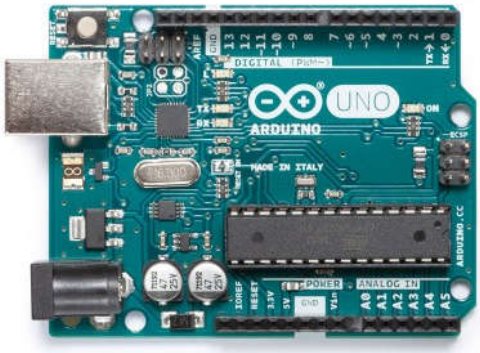
Nama	Digunakan Pada	Polinomial	
CRC-1	Berbagai macam hardware / disebut juga dengan *parity bit*	0x1	X + 1
CRC-3-GSM	Jaringan seluler	0x3	X3+X+1
CRC-5-USB	Paket token USB	0x05	X5+X2+1
CRC-6-GSM	Jaringan seluler	0x2F	X6+X5+X3+X+1
CRC-7	Sistem telekomunikasi, MMC, SD, ITU	0X09	X7+X3+1
CRC-8	*Digital Video Broadcasting Second Generation* (DVB-S2)	0XD5	X8+X5+X3+X2+X+1

CRC-8-Bluetooth	Konektivitas nirkabel	0XA7	
CRC-8-CCIT	HEC, Cell Delineation, ISDN, ATM, ITU-T	0X07	X8+X2+X+1
CRC-8-Dallas/Maxim	1-Wire Bus	0X8C	X8+X5+X4+1
CRC-10	ATM, ITU-T	0X233	X10+X9+X5+X4+X+1
CRC-16-CCIT	HDLC, XMODEM, Bluetooth, DigRF, SD	0X1021	X16+X12+X5+1
CRC-32	ISO/IEC/IEEE 802-3 (Ethernet), SATA, MPEG-2, Gzip, Bzip2, POSIX cksum, PNG,ZMODEM	0x04C11DB7	X32+X26+X23+X22+X16+X12+X11+X10+X8+X7+X5+X4+X2+X+1

Sumber : (https://en.wikipedia.org/wiki/Cyclic_redundancy_check)

2.2.7 Mikrokontroller

Mikrokontroller adalah komputer kecil yang kompleks da dirancang untuk suatu tujuan tertentu dan menjadi pengontrol pada sistem *embedded*. Penelitian ini menggunakan mikrokontroller Arduino Uno R3. Arduino adalah mikrokontroller yang menggunakan ATMEGA328 sebagai mikrokontroller dan memiliki jumlah pin I/O digital sebanyak 14 pin dan pin input analog sebanyak 6 pin. Arduino Uno adalah jenis mikrokontroller Arduino yang paling banyak digunakan dan paling disarankan untuk pemula. Dengan pemrograman cukup menggunakan USB tipe A dan tipe B.



Gambar 2.1 Mikrokontroller Arduino Uno R3

Sumber: (https://store.arduino.cc/usa/arduino-uno-rev3)

Spesifikasi yang dimiliki oleh Arduino Uno R3 ini dijelaskan pada tabel 2.1

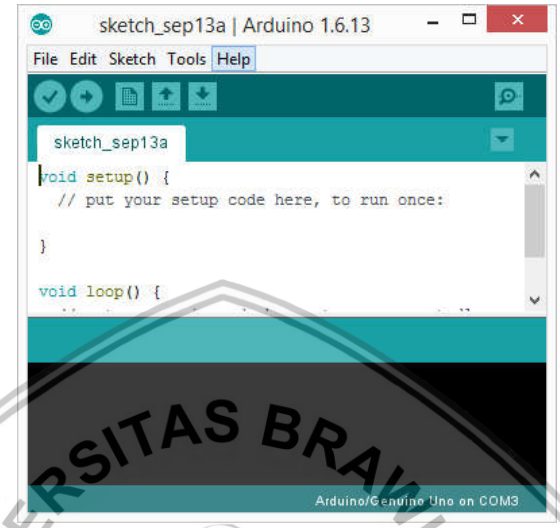
Tabel 2.2 Spesifikasi Arduino Uno R3

Microcontroller	**ATmega328P**
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Sumber: (https://store.arduino.cc/usa/arduino-uno-rev3)

2.2.8 Arduino IDE

Merupakan perangkat lunak penghubung antara user dengan mikrokontroller. Arduino IDE merupakan perangkat lunak yang mendukung bahasa C++ untuk digunakan pada board arduino. Tampilan arduino IDE ditunjukkan pada Gambar 2.3.

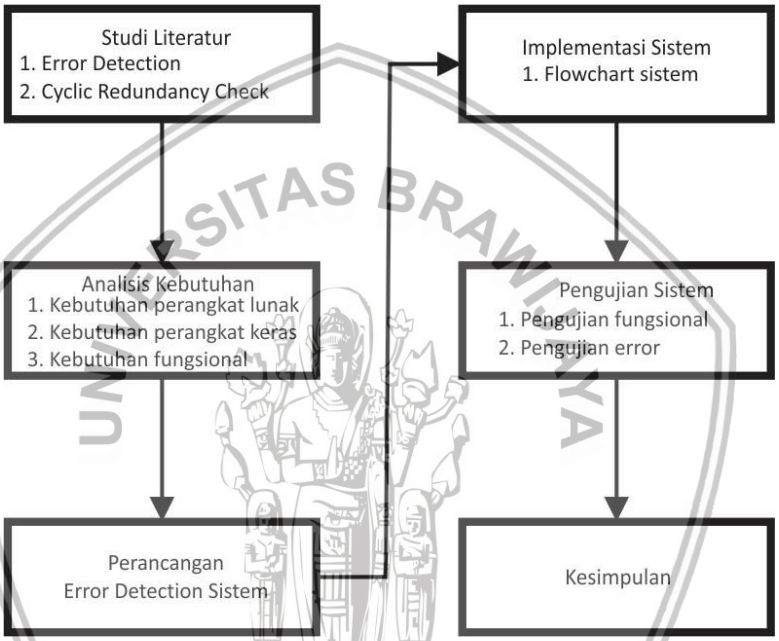


Gambar 2.2 Tampilan Arduino IDE

Terdapat 2 fungsi utama pada pemrograman arduino yaitu fungsi void setup() dan void loop(), dimana void setup() digunakan untuk fungsi yang hanya dijalankan sekali saja sedangkan void loop() digunakan untuk menjalankan proses yang memerlukan perulangan atau looping.

BAB 3 METODOLOGI

Dalam perancangan sebuah sistem penelitian terdapat proses-proses yang harus disusun dengan baik dan terorganisir agar dapat menciptakan perancangan sistem yang terstruktur dan tersusun dengan baik. Penelitian ini diawali dengan studi literatur yang terkait dengan kajian pustaka dan dasar teori. Penelitian ini bersifat implementatif karena mengintegrasikan metode Cyclic Redudancy Check pada komunikasi serial arduino untuk mengatasi kegagalan pada data atau *error*. Pada bab ini menjelaskan tentang alur metode penelitian yang dilakukan untuk pembuatan sistem ini dapat dilihat dalam bentuk diagram alir pada Gambar 3.1.



Gambar 3.1 Alur Penelitian

3.1 Studi Literatur

Studi literatur adalah tahap mencari referensi dan teori dasar yang mendukung sistem yang dirancang sebagai penunjang penelitian. Kajian studi literatur dari beberapa sumber yang dilakukan oleh penulis guna menunjang perancangan sistem pada penelitian ini, baik dari jurnal, paper, dan buku. Teori pustaka yang berkaitan dengan penelitian ini adalah:

1. *Error Detection*

   Studi literatur mengenai *error detection* dilakukan dengan mencari jurnal, paper, dan penjelasan tentang apa itu *error detection* dan apa saja metode metode yang digunakan.

2. *Cyclic Redudancy Check*

   Studi literatur tentang metode CRC dilakukan dengan mengkaji buku, penelitian terkait, jurnal dan penjelasan melalui video tentang cara kerja CRC dan cara implementasinya ke komunikasi Arduino.

3.2 Analisis Kebutuhan

Analisis kebutuhan sangat dibutuhkan dalam perancangan sebuah sistem, hal ini dilakukan dengan cara mengidentifikasi seluruh kebutuhan sistem, identifikasi sistem dilakukan dengan cara mengidentifikasi baik disisi *hardware* maupun *software*. Dengan adanya identifikasi dan analisis kebutuhan sistem ini diharapkan akan mempermudah dalam proses perancangan sistem nanti.

3.2.1 Kebutuhan Perangkat Keras

Analisis kebutuhan perangkat keras (*hardware*) apa saja yang dibutuhkan sistem pada penelitian ini. Adapun perangkat keras (*hardware*) yang dibutuhkan sistem antara lain:

1. Laptop
2. Arduino Uno R3

3.2.2 Kebutuhan Perangkat Lunak

Analisis kebutuhan perangkat lunak (*software*) apa saja yang dibutuhkan sistem pada penelitian ini. Adapun perangkat lunak (*software*) yang dibutuhkan sistem antara lain:

1. Arduino IDE 1.6

3.2.3 Kebutuhan Sistem

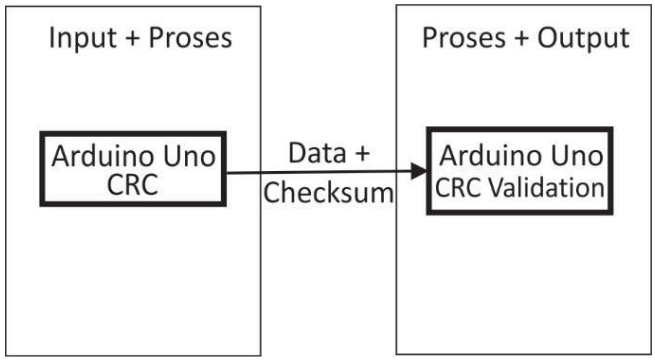
Analisis kebutuhan sistem adalah kebutuhan yang harus dipenuhi suatu sistem untuk mencapai tujuan suatu penelitian, kebutuhan sistem dari penelitian ini antara lain:

1. Arduino mampu mengirim data atau nilai ke penerima.
2. Sistem mampu untuk melakukan komputasi CRC pada sisi pengirim dan mampu melakukan validasi CRC pada sisi penerima dengan baik.
3. Metode CRC mampu mendeteksi jika terdapat *error* maupun tidak ada *error* pada data dengan baik dan tepat pada data dan *remainder* setelah diterima oleh penerima.

3.3 Perancangan Sistem

Tahap perancangan sistem dilakukan setelah tahap analisis kebutuhan selesai dilakukan dan terpenuhi. Tujuan dilakukan perancangan sistem agar implementasi sistem berjalan terstruktur dan sistematis. Adapun desain sistem penelitian ini

yang telah penulis sediakan guna mempermudah dalam perancangan sistem dengan blok diagram pada gambar 3.2.

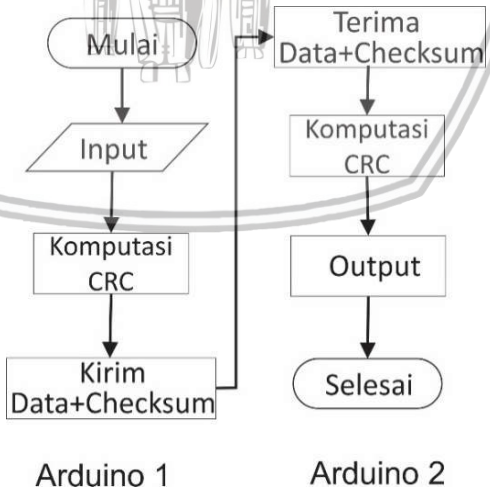


Gambar 3.2 Blok Diagram Sistem

Pada gambar 3.2 Arduino Uno 1 akan memproses data input dari user dengan komputasi menggunakan metode CRC hingga mendapatkan *remainder* setelah itu data dan *remainder* akan dikirim ke Arduino 2 dan jika sudah diterima oleh arduino 2 data dan *remainder* tersebut dilakukan komputasi CRC lagi jika data dan *remainder* tersebut menghasilkan nilai selain 0 maka data tersebut dinyatakan terdapat *error*.

3.4 Implementasi Sistem

Pada tahap implementasi ini dimulai dari merancang rangkaian komunikasi serial pada dua Arduino uno lalu mengimplementasikan *error detection* yaitu metode CRC pada komunikasi itu untuk mendeteksi *error* pada komunikasi data pada dua Arduino itu. Penerapan CRC menggunakan operasi xor operasi geser sehingga mendapatkan *remainder*. Implementasi sistem seperti digambarkan pada Gambar 3.3:



Gambar 3.3 Implementasi Sistem

Pada Gambar 3.3 dimulai dengan memasukkan data input setelah itu data akan dikomputasi sehingga mendapatkan *remainder* lalu data dan *remainder*

tersebut dikirim ke Arduino 2, setelah data dan *remainder* tersebut diterima oleh Arduino 2 akan dilakukan komputasi lagi sekaligus melakukan pengecekan kesalahan atau *error* pada data tersebut. Jika tidak ada kesalahan atau *error* maka data tersebut akan menjadi *output* data dan diterima oleh user.

3.5 Pengujian dan Analisis Sistem

Pengujian dan analisis pada penelitian ini digunakan untuk mengukur seberapa baik dan menguji apakah sistem tersebut sudah berjalan dengan baik dan sesuai dengan apa yang diinginkan. Pengujian meliputi:

1. Pengujian Fungsional

   Menguji apakah sistem komunikasi data serial pada kedua arduino berjalan dengan baik dan tidak ada kendala.

2. Pengujian *Error*

   Menguji apakah sistem dapat mendeteksi *error* dengan baik dan tepat jika diberikan *error* pada data.

3. Pengujian Waktu

   Menguji waktu yang dibutuhkan sistem untuk melakukan enkode, dekode data dan melakukan deteksi kesalahan pada data.

3.6 Kesimpulan

Kesimpulan didapatkan jika telah melakukan perancangan sistem, implementasi dan pengujian serta analisis. Kesimpulan dituliskan setelah melakukan pengujian sistem dan analisis sistem sehingga dapat menjadi acuan untuk ke depannya serta diharapkan juga menjadi acuan untuk penelitian yang lain dan memperbaiki kekurangan-kekurangan yang ada dalam penelitian ini, agar dapat lebih menyempurnakan lagi dan untuk pengembangan sistem selanjutnya.

BAB 4 REKAYASA KEBUTUHAN

Pada bab rekayasa kebutuhan ini berisikan penjelasan mengenai kebutuhan-kebutuhan dari sistem yang dirancang agar dapat memenuhi tujuan yang telah dibuat sebelumnya. Kebutuhan tersebut meliputi gambaran umum sistem, kebutuhan perangkat lunak, kebutuhan perangkat keras dan kebutuhan fungsional. Dengan adanya rekayasa kebutuhan ini diharapkan sistem yang sedang dirancang dapat bekerja dengan baik.

4.1 Gambaran Umum Sistem

Sistem yang akan dibuat yaitu sistem *error detection* menggunakan metode *Cyclic Redundancy Check* yang akan diterapkan pada komunikasi serial Arduino Uno. Arduino sebagai pengirim menerima input dari user dan melakukan komputasi untuk mencari *remainder*-nya. *Remainder* ini digunakan sebagai *checksum* yang akan dikirimkan bersamaan dengan data ke arduino penerima, pada sisi arduino penerima data dan *checksum* tadi dilakukan komputasi ulang untuk mendeteksi ada atau tidaknya kesalahan pada data tersebut.

4.2 Kebutuhan Fungsional

Kebutuhan fungsional merupakan kebutuhan yang wajib dipenuhi agar sistem dapat berjalan sesuai dengan apa yang diharapkan. Kebutuhan fungsional disini berupa kebutuhan sistem, kebutuhan perangkat keras, dan kebutuhan perangkat lunak.

4.2.1 Kebutuhan Sistem

4.2.1.1 Arduino mampu mengirim data melalui serial

Merupakan kebutuhan fungsional sistem, yaitu Arduino pengirim diharapkan mampu mengirim data ke Arduino penerima sebelum di implementasikan metode *error detection* CRC.

4.2.1.2 Sistem mampu untuk melakukan komputasi dan validasi data dengan baik.

Merupakan kebutuhan fungsional sistem, yaitu sistem diharapkan dapat melakukan komputasi dengan metode ini dengan baik dan melakukan validasi *codeword*.

4.2.1.3 Metode CRC mampu mendeteksi *error* dengan baik dan tepat pada *codeword*.

Merupakan kebutuhan fungsional sistem yaitu setelah diterapkannya metode *error detection* CRC pada komunikasi serial arduino, metode ini dapat mendeteksi *error* atau kesalahan pada data.

4.2.2 Kebutuhan Perangkat Keras

Pada bagian ini menjelaskan terkait kebutuhan perangkat keras apa saja yang digunakan pada sistem ini. Perangkat keras ini meliputi sekumpulan komponen elektronika yang membentuk suatu perangkat baru yang digunakan untuk tujuan tertentu. Berikut adalah kebutuhan perangkat keras yang digunakan untuk implementasi sistem ini, yakni sebagai berikut:

1. Mikrokontroller Arduino Uno
   Arduino Uno adalah sebuah board mikrokontroller yang banyak digunakan untuk *embedding system.* Arduino uno memiliki chip mikroprosesor berbasis ATMega328 yang memungkinkan pengguna untuk melakukan *upload program* atau *sketch* ke mikrokontroller arduino untuk melakukan embedding dan menjalankan fungsi sebagai pengirim dan penerima data melalui komunikasi serial dengan menggunakan pin Tx Rx yang sudah disediakan.

4.2.3 Kebutuhan Perangkat Lunak

Pada bagian ini menjelaskan perangkat lunak atau *software* apa saja yang digunakan untuk implementasi sistem, yakni sebagai berikut:

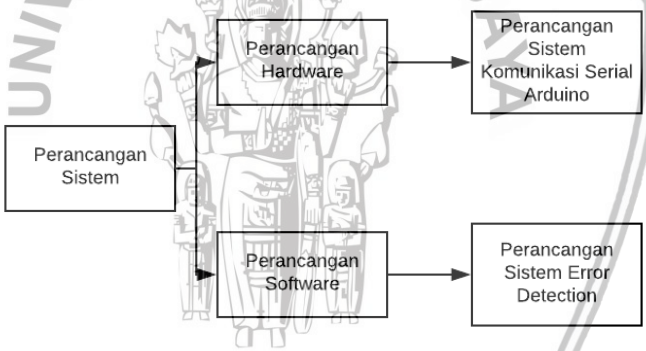
1. Arduino IDE
   IDE merupakan kependekan dari *Integrated Development Environtment* yang secara bahasa dapat diartikan sebagai lingkungan integrasi untuk pengembangan. Arduino menggunakan bahasa pemrograman yang menyerupai bahasa pemrograman C. Bahasa pemrograman Arduino atau *Sketch* dapat memudahkan pemula untuk melakukan pengembangan atau *development*. IC Arduino yaitu ATMega328 sudah diberi program *bootloader* yang menjadi perantara antara *compiler* dengan mikrokontroller. Arduino IDE sendiri dibuat dari bahasa pemrograman Java dan dilengkapi dengan *library* C maupun C++ yang biasa disebut *wiring*, hal ini memudahkan untuk melakukan operasi input dan output.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Setelah tahapan analisis kebutuhan selesai dilakukan tahapan selanjutnya yang dilakukan adalah tahap perancangan dan implementasi. Perancangan dilakukan berdasarkan hasil yang didapatkan dari analisis kebutuhan fungsional dan non fungsional, dan implementasi dilakukan berdasarkan perancangan sistem dari perangkat keras atau *hardware* dan perangkat lunak atau *software*. Pada bab ini perancangan dan implementasi dimulai dari perancangan perangkat keras sistem komunikasi serial arduino dan dilanjutkan dengan perancangan perangkat lunak sistem komunikasi serial dan sistem *error detection* lalu dilakukan implementasi komunikasi serial tersebut pada *hardware* dan dilanjutkan dengan implementasi metode CRC pada serial komunikasi tersebut untuk mendeteksi *error*.

5.1 Perancangan Sistem

Pada perancangan sistem akan dijelaskan mengenai tahapan untuk merancang dan membangun sistem ini, tahapan dimulai dari perancangan *hardware*, lalu dilanjutkan dengan perancangan *software* dimana tahapan perancangan sistem ini digambarkan pada Gambar 5.1.



Gambar 5.1 Diagram Perancangan Sistem

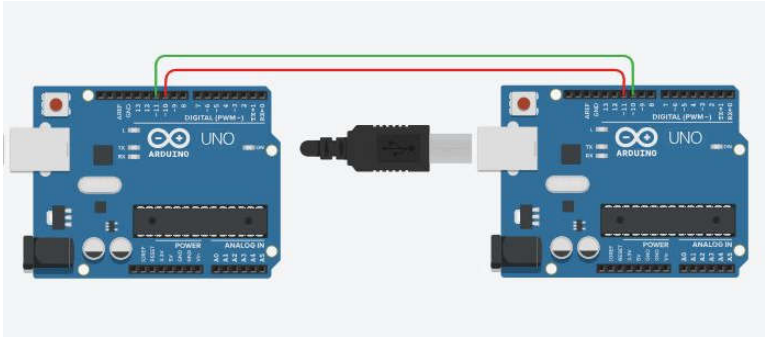
Gambar 5.1 menunjukkan perancangan sistem pada sisi *hardware* dan sisi *software.* Perancangan *hardware* yaitu perancangan sistem komunikasi serial UART antar 2 arduino dan perancangan *software* dilakukan dengan merancang sistem *error detection* pada serial komunikasi antar 2 arduino tersebut

5.1.1 Perancangan Hardware

Perancangan *hardware* merupakan tahapan alur pembuatan sistem pada bagian perangkat keras atau *hardware*. Perancangan *hardware* membutuhkan 2 (dua) buah mikrokontroller arduino yang saling berkomunikasi secara serial dengan menggunakan pin TX dan RX.

Komunikasi serial arduino menggunakan 2 Pin yang sudah *built in* atau yang sudah tersedia pada *board* arduino yaitu pin RX dan TX, pada penelitian ini penulis

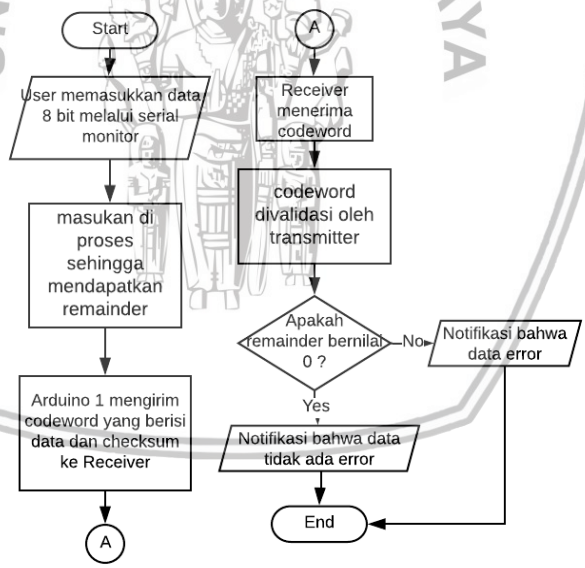
menggunakan pin 10 dan pin 11 sebagai pin RX dan TX, pin 10 arduino master dihubungkan dengan pin 11 arduino slave, dan pin 10 arduino slave disambungkan dengan pin 11 arduino master, seperti yang digambarkan dengan Gambar 5.2.



Gambar 5.2 Skematik sistem komunikasi serial Arduino

5.1.2 Perancangan Software

Perancangan software merupakan tahapan alur pembuatan sistem pada bagian perangkat lunak atau *software*. Perancangan *software* disini merupakan perancangan metode sistem CRC atau *Cyclic Redundancy Check* untuk mendeteksi *error* pada komunikasi serial antara kedua arduino tersebut. Adapun diagram alir perancangan *software* digambarkan pada Gambar 5.3.



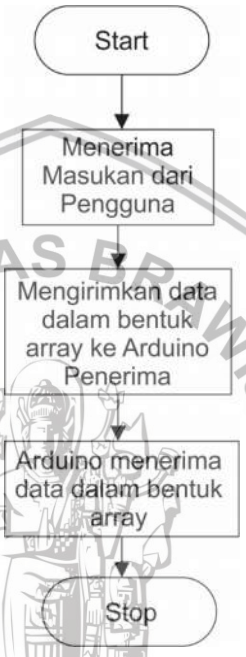
Gambar 5.3 Diagram Alir Perancangan Software Sistem

Pada perancangan *software* dimulai dari memasukkan data pada Arduino pengirim, lalu arduino melakukan komputasi pada data dan mendapatkan *remainder*, selanjutnya *remainder* dan data dikirimkan ke arduino penerima, pada penerima data akan dilakukan komputasi ulang, jika hasil komputasi tersebut

bernilai nilai 0 maka tidak ada kesalahan pada data, jika hasilnya tidak bernilai 0 maka terdapat kesalahan pada data tersebut.

5.1.2.1 Perancangan Software Sistem Komunikasi Serial Arduino

Perancangan sistem komunikasi serial arduino menggunakan library SoftwareSerial.h dengan inisialisasi pin 10 sebagai RX dan pin 11 sebagai TX. Pada perancangan ini menggunakan serial monitor sebagai tempat masukan dari pengguna dan data masukan dari pengguna ini nantinya akan dikirimkan ke arduino penerima melalui komunikasi serial UART, data ini yang nantinya akan di deteksi kesalahannya oleh arduino penerima.



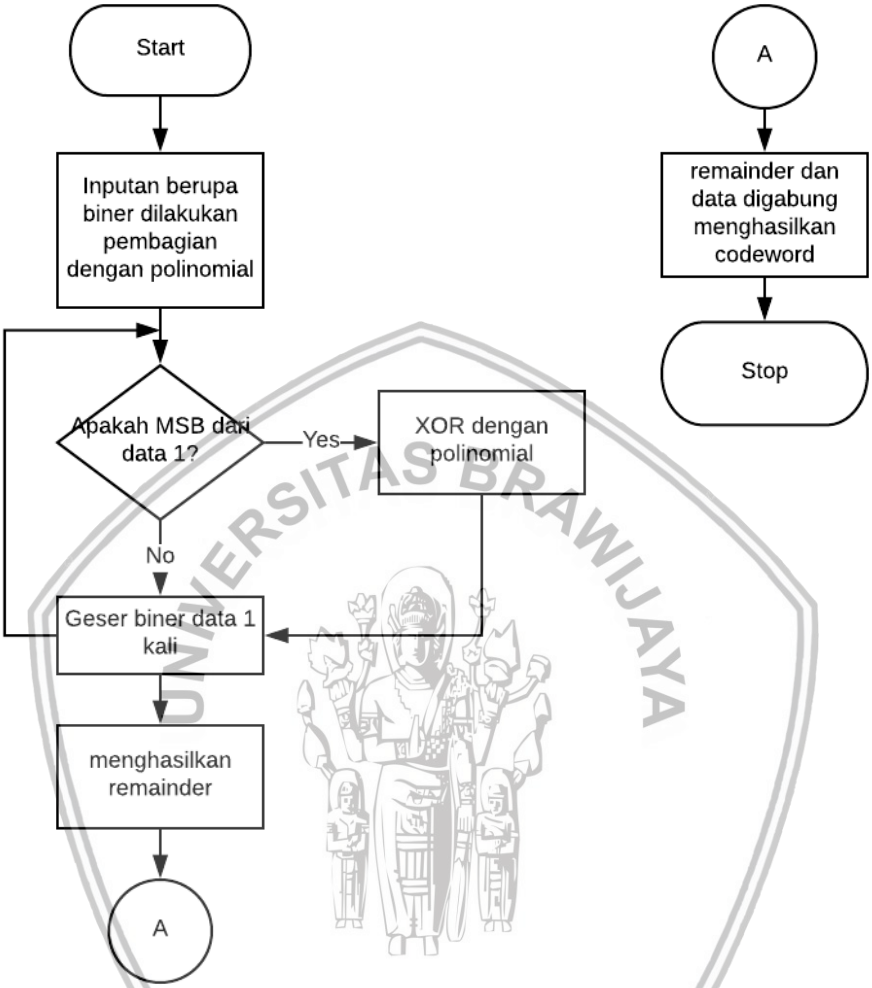
Gambar 5.4 Perancangan Software Komunikasi Serial

Pada Gambar 5.4 menampilkan urutan perancangan *software* komunikasi serial yang dimulai dari arduino pengirim menerima data masukan dari pengguna lalu arduino akan memasukkan data tersebut kedalam *array* dan dikirimkan ke arduino penerima dalam bentuk *array*, pada arduino penerima data diterima dan dibaca dalam bentuk *array* juga lalu data tersebut akan dilakukan pengecekan *error* oleh sistem.

5.1.2.2 Perancangan Software Sistem Error Detection (Cyclic Redundancy Check)

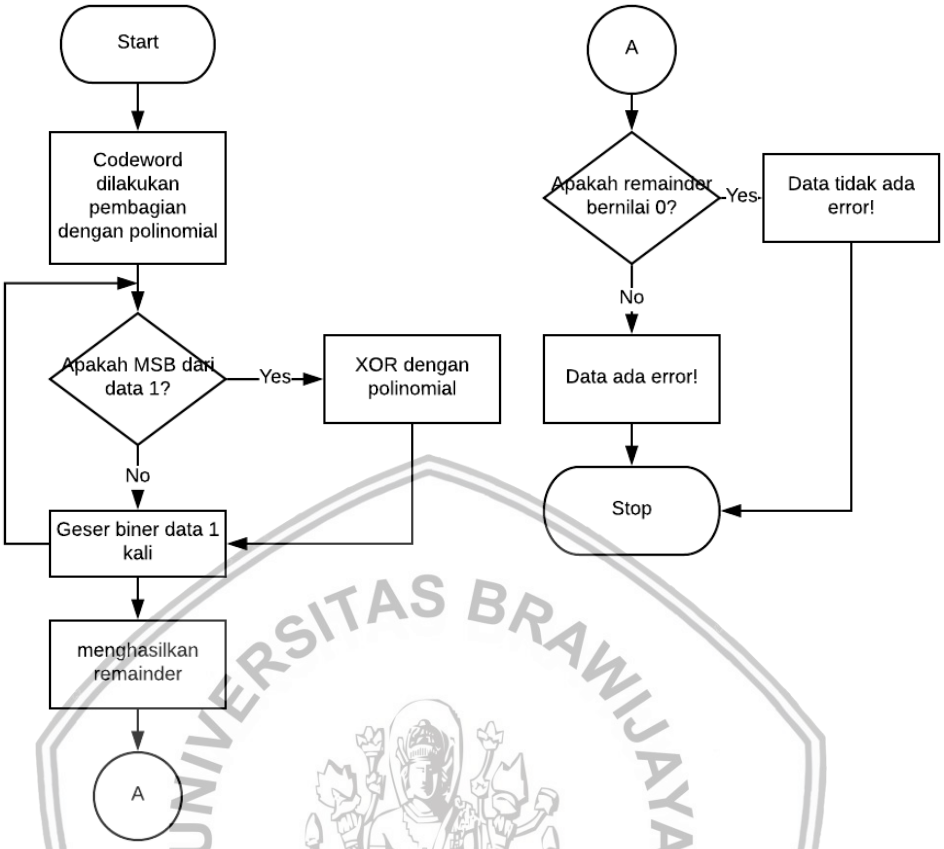
Perancangan sistem CRC berfungsi sebagai pendeteksi kesalahan yang ada pada data yang dikirimkan oleh pengirim, Arduino 1 bertugas sebagai pengirim data dan melakukan operasi pada data sehingga didapatkan *codeword*, dan Arduino 2 berfungsi sebagai penerima dan bertugas sebagai pedeteksi *error* pada data tersebut dengan melakukan operasi metode LFSR dan *polynomial division*. Data tersebut dibagi dengan *generator* polinomial yang dipilih dan sama dengan *polynomial generator* yang digunakan untuk menghasilkan *remainder* pada

Arduino 1. Jika hasil pembagian memiliki nilai selain dari nilai 0 maka data mengalami *error* baik pada saat pengiriman datanya atau pada saat penerimaan datanya. Diagram alir perancangan sistem digambarkan pada Gambar 5.4 dan 5.5.



Gambar 5.5 Perancangan sistem *Error Detection* pada sisi pengirim

Pada Gambar 5.5 merupakan perancangan sistem *error detection* yang dimulai dari arduino pengirim ketika mendapatkan masukan akan dikirimkan ke fungsi dan pada fungsi tersebut data akan diubah menjadi biner dan dilakukan pembagian dengan polinomialnya. Jika biner data tersebut memiliki *most significant bit* (MSB) bernilai 1 maka data tersebut akan dilakukan operasi XOR dengan biner polinomial dan digeser 1 step, jika MSB dari biner data tersebut bernilai 0 maka akan digeser. Proses ini dilakukan berulang ulang sampai data tersebut habis dan menghasilkan sisa hasil bagi atau *remainder*. *Remainder* ini akan dimasukkan ke dalam *array* dan dikirimkan ke arduino penerima bersamaan dengan data yang dimasukkan pengguna.



Gambar 5.6 Perancangan sistem *Error detection* pada sisi penerima

Gambar 5.6 menampilkan perancangan sistem *error detection* pada sisi penerima. Pada saat *codeword* diterima maka akan diubah ke bilangan biner dan dilakukan pembagian polinomial jika biner dari *codeword* tersebut memiliki *most significant bit / MSB* bernilai 1 maka akan dilakukan operasi XOR dengan dengan polinomial dan dilakukan operasi geser, jika MSB bernilai 0 maka hanya akan dilakukan operasi geser. Operasi pembagian ini terus dilakukan hingga biner dari codeword tersebut habis dan menghasilkan sisa hasil bagi atau *remainder*, selanjutnya melakukan operasi perbandingan pada *remainder* tersebut jika *remainder* bernilai 0 maka data tersebut tidak terdapat *error* dan sebaliknya jika *remainder* bernilai selain dari 0 maka data tersebut terjadi *error*.

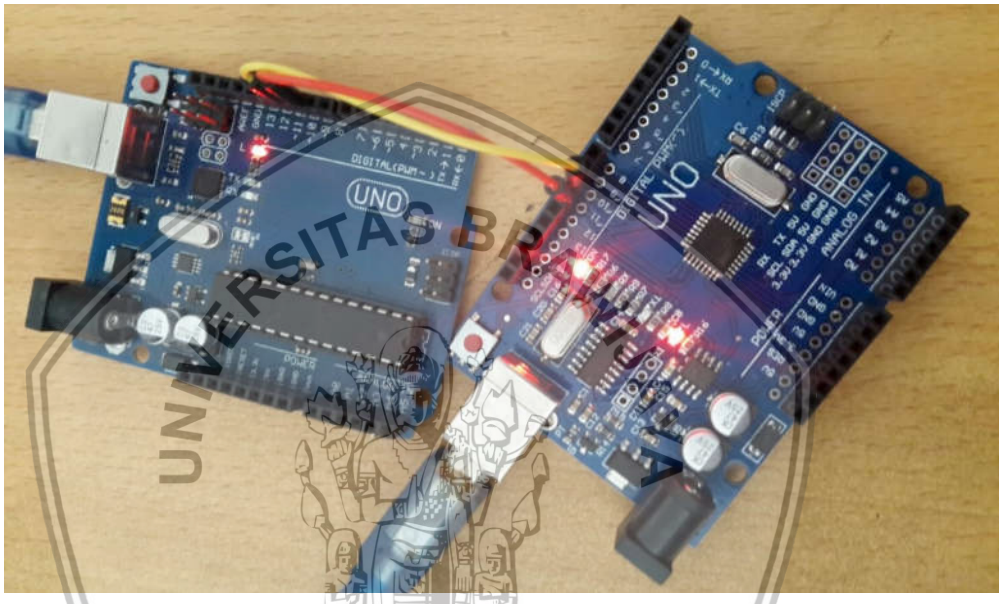
5.2 Implementasi Sistem

Implementasi sistem dilakukan setelah melakukan bab perancangan sistem baik pada sisi perangkat keras sistem maupun pada perangkat lunak sistem pada sub-bab sebelumnya. Pada sub-bab ini akan dijelaskan mengenai implementasi sistem dari sisi *hardware* dan *software* dari sistem ini.

5.2.1 Implementasi Hardware

Implementasi *hardware* mengacu pada perancangan *hardware* pada sub-bab perancangan sebelumnya. Implementasi sistem komunikasi serial dengan menggunakan pin 10 sebagai pin RX dan pin 11 sebagai pin TX pada arduino.

Pada implementasi sistem komunikasi serial arduino menggunakan pin RX dan TX yang ada pada arduino, pin RX pada arduino 1 tersambung pada pin TX arduino 2, dan pin RX arduino 2 tersambung pada arduino 1. Pada sistem yang akan dibuat Arduino 1 sebagai *transmitter* atau pengirim sedangkan arduino 2 sebagai penerima data yang dikirim oleh arduino 1. Hasil implementasi sistem komunikasi serial arduino dapat dilihat pada Gambar 5.6.



Gambar 5.7 Implementasi Serial Komunikasi Arduino

5.2.2 Implementasi Software

Implementasi *software* mengacu pada perancangan *software* pada sub-bab perancangan sebelumnya. Implementasi software mengikuti alur kerja diagram alir yang telah dijelaskan pada sub-bab rancangan dan diimplementasikan melalui kode program dengan menggunakan *software* arduino IDE. Pada sub-bab ini dijelaskan barisan-barisan kode yang digunakan untuk mengimplementasikan sistem ini.

5.2.2.1 Implementasi Software Library

Dalam implementasi kode program dibutuhkan *library* untuk mempermudah peneliti dalam implementasi sistemnya. Dalam penelitian ini penulis menggunakan *library* `SoftwareSerial.h` seperti yang ditunjukkan pada Tabel 5.1 untuk menggunakan pin 10 dan pin 11 sebagai pin rx dan tx.

Tabel 5.1 Kode Sumber Software Library

```
Software Library Pengirim dan Penerima
```

```
1  #include <SPI.h>
2  #include <Wire.h>
3  #include<stdio.h>
4  #include<string.h>
5  #include <SoftwareSerial.h>
```

5.2.2.2 Implementasi Software Pin-Pin dan Variabel

Implementasi *software* pin-pin dan variabel yang digunakan pada penelitian ini. Pin-pin yang digunakan adalah pin 10 dan pin 11 yang berfungsi sebagai pin rx dan tx pada SoftwareSerial, dan juga inisialisasi dari variabel *global* yang dapat digunakan pada fungsi-fungsi lain yang ada pada kode program. Kode inisialisasi pada sisi pengirim ditunjukkan pada Tabel 5.2 dan pada penerima ditunjukkan pada tabel 5.3.

Tabel 5.2 Kode Sumber Pin-pin dan variabel pengirim

```
Software Pin Pengirim
1  SoftwareSerial mySerial(10, 11); //RX, TX
2  byte inp[1];
3  byte arr[2];
4  byte charRead;
5  char inByte = 0;
6  int j = 0;
7  int i = 0;
8  byte poli;
9  char err;
10 unsigned long timers;
```

Pada Tabel 5.2 merupakan kode sumber pada pengirim yang terdapat inisialisasi *library* pada arduino yaitu `SPI.h` untuk melakukan pengujian sistem SPI, *library* `Wire.h` untuk melakukan pengujian sistem I2C, dan `SoftwareSerial.h` yang merupakan library untuk melakukan komunikasi secara serial UART. Pada baris 10 menunjukkan inisialisasi pin yang dipakai untuk melakukan komunikasi serial yaitu pin 10(RX) dan pin 11(TX), pada baris 7 inisilisasi variabel *array* `inp[2]` pada global variabel, 2 pada array tersebut merupakan nilai *index array* tersebut dan *array* ini digunakan untuk menerima data masukan dari pengguna. Pada baris 8 terdapat inisialisasi array untuk pengiriman data pada sistem, dan pada baris 14 adalah inisialisasi variabel polinomial yang akan digunakan oleh sistem.

Tabel 5.3 Kode sumber Software Pin-pin dan variabel penerima

```
Software Pin Penerima
1  SoftwareSerial mySerial(10, 11); //RX, TX
2  byte inp[2];
3  int j = 0;
4  int i = 0;
5  byte poli;
6  volatile byte pos;
7  volatile boolean process;
8  float buff[2];
9  unsigned long timers;
```

Pada tabel 5.3 menampilkan kode sumber inisialisasi pin, *library* dan variabel untuk arduino sebagai penerima. *Library* yang dipakai pada arduino

penerima adalah SPI.h untuk melakukan komunikasi SPI, `Wire.h` untuk melakukan komunikasi I2C, *library* `SoftwareSerial.h` untuk melakukan komunikasi UART. Untuk inisialisasi pin serial UART menggunakan variabel `mySerial` dan pin 10(RX) dan 11(TX). Untuk variabel yang diinisialisasi pada global variabel yaitu variabel array inp[2] yang memiliki *index* 2 digunakan untuk menerima array yang dikirim oleh pengirim dan digunakan untuk masukan pada fungsi *error detection,* lalu ada variabel poli yang digunakan untuk menyimpan variabel polinomial yang dipilih pengguna, selanjutnya variabel `volatile byte pos`, `process` dan `buff` yang digunakan untuk *buffer* array pada komunikasi SPI, dan variabel timer untuk menghitung timer pada pemilihan polinomial.

5.2.2.3 Implementasi Sistem Error Detection (Cyclic Redundancy Check)

Dalam mengimplementasikan kode program pada sistem *error detection* ini penulis melakukan operasi xor dan operasi geser dalam bahasa C++ dan operasi perulangan *for*, jika biner dari *input* memiliki *most significant bit* bernilai 1 maka akan dilakukan operasi xor ke polinomialnya dan jika bernilai 0 maka biner dari data tersebut hanya akan digeser, begitu seterusnya sampai biner dari data tersebut habis dan menyisakan *remainder* atau sisa yang nantinya akan digunakan sebagai *codeword*.

Tabel 5.4 Kode sumber *Sistem Error Detection*

Software Sistem Error Detection pada pengirim dan penerima

```
1  byte hash(const byte *data)
2  {
3      byte crc = *data; /* init crc directly with input byte instead
4  of 0, avoid useless 8 bitshifts until input byte is in crc register
5  */
6      for (int i = 0; i < 8; i++)
7      {
8          if ((crc & 0x80) != 0)
9          { /* most significant bit set, shift crc register and
10 perform XOR operation, taking not-saved 9th set bit into account
11 */
12             crc = (byte)((crc << 1) ^ poli);
13         }
14         else
15         { /* most significant bit not set, go to next bit */
16             crc <<= 1;
17         }
18     }
19     return crc;
20 }
```

Pada Tabel 5.4 menjelaskan tentang kode sumber yang digunakan untuk melakukan operasi LFSR dan pembagian polinomial. Kode sumber ini merupakan sebuah fungsi yang bernama `byte hash` dengan parameter `const byte data` dan `byte len`, parameter `data` adalah parameter untuk data input dan `len` adalah parameter panjang dari data tersebut. Fungsi ini berisi inisialisasi variabel `byte crc` yang bernilai 0x00 dan perulangan `while` dengan kondisi decrement variabel `len`, didalam perulangan `while` terdapat variabel `byte extract` yang bernilai sama dengan data, lalu terdapat 8 kali perulangan `for` yang berisi perulangan `if`

jika `sum` yang bernilai `extract` / data AND 0x01 bernilai 1 maka akan dilakukan operasi XOR data dengan variabel `poli` yang merupakan variabel polinomial lalu digeser 1 bit dan jika `sum` bernilai 0 maka akan dilakukan penggeseran 1 bit. Jika perulangan `for` tersebut selesai maka akan mengembalikan nilai variabel `crc` yang merupakan remainder.

5.2.2.4 *Implementasi Software Sub Program*

Implementasi *software* sub *program* adalah implementasi kode *program* pada bagian fungsi *setup* pada sistem. Pada fungsi setup ini berisikan inisialisasi *baudrate* serial yang digunakan pada sistem ini, disini penulis menggunakan 2 serial yaitu *Serial* dan *mySerial* dari *SoftwareSerial* yang inisialisasi nya pada *baudrate* yang sama yaitu 9600bps. Tabel 5.5 menunjukkan kode implementasi sub *program* pada sisi pengirim. Pada Tabel 5.6 adalah implementasi sub *program* pada sisi penerima.

Tabel 5.5 Kode sumber sub program pengirim

Software Sub Program Pengirim

```
1  void setup() {
2    Serial.begin(9600);
3    mySerial.begin(9600);
4    Serial.setTimeout(50);
5    //Wire.begin();
6    //digitalWrite(SS, HIGH);
7    //SPI.begin();//divide the clock by 8
8    //SPI.setClockDivider(SPI_CLOCK_DIV8);
9    }
```

Pada Tabel 5.5 berisi kode sumber sub program pada sisi pengirim yang berisi inisialisasi baudrate 9600bps dari variabel `serial` dan `mySerial`, selanjutnya inisialisasi serial timeout yang bernilai 50 dan inisialisasi variabel yang digunakan untuk pengujian waktu yaitu `wire`, `digitalwrite`, dan `SPI`.

Tabel 5.6 Kode sumber sub program penerima

Software Sub Program Penerima

```
1  void setup() {
2    //Wire.begin(9);
3    //Wire.onReceive(receiveEvent);
4    Serial.begin(9600);
5    mySerial.begin(9600);
6    //pinMode(MISO,OUTPUT);
7    //SPCR |= _BV(SPE);
8    //SPCR |= _BV(SPIE);
9    //pos = 0;
10   //process = false;
11   }
```

Tabel 5.6 merupakan kode sumber dari sub program penerima yang berisikan inisialisasi *baudrate* untuk komunikasi serial, inisialisasi pin I2C, dan pin SPI yang digunakan untuk pengujian waktu.

5.2.2.5 Implementasi Software Main Program

Pada implementasi *software main program* berisi inisialisasi, pengoperasian dan pemanggilan fungsi-fungsi yang telah didefinisikan terlebih dahulu. Pada sisi penerima yang ditunjukkan pada tabel 5.7 *main program* berisikan inisialisasi nilai variabel *array*, *looping* untuk mengirimkan data dari *array* dan mencetak nilai dari *array* tersebut.

Tabel 5.7 Kode sumber main program pengirim

Software Main Program Pengirim

```
1   void loop() {
2     while (Serial.available() > 0){
3       charRead = Serial.parseInt();
4     }
5     inp[0]=charRead;
6     byte hasil = hash(inp, sizeof(inp));
7     arr[0] = inp[0];
8     arr[1] = hasil;
9     for(int j = 0; j < sizeof(arr); j++)
10    {
11      mySerial.write(arr[j]);
12    }
13    delay(2000);
14  }
```

Kode sumber pada Tabel 5.7 merupakan kode sumber untuk main program pada sisi arduino pengirim yang pada baris pertama terdapat perulangan `while` jika terdapat nilai masukan pada serial monitor maka akan dibaca dan diinisialisasikan dengan variabel `charRead` dan variabel `charRead` ini dimasukkan pada array `inp[0]`, pada baris selanjutnya merupakan pemanggilan fungsi `hash` dengan parameter inp yang merupakan nilai masukan dari serial dan parameter sizeof(inp) yang merupakan fungsi untuk menghitung panjang dari array `inp[]`, lalu nilai array `inp[]` tadi dimasukkan ke dalam array `arr[]` pada index ke 0 dan hasil dari pemanggilan fungsi tersebut dimasukkan pada array `arr[]` pada index ke 1

Pada sisi penerima seperti pada Tabel 5.8 main *program* berisikan *looping* untuk membaca data yang dikirimkan oleh pengirim dan akan dikomputasi oleh *hash* setelah itu dilakukan perbandingan jika hasil komputasi bernilai 0 maka akan mencetak "tidak ada *error* pada data!" pada *serial monitor*, jika hasil tersebut bernilai selain dari 0 maka akan mencetak "terdapat *error* pada data!" pada *serial monitor*.

Tabel 5.8 Kode sumber main program penerima

Software Main Program Penerima

```
1   void loop() {
2     if (mySerial.available() >= 2) {
3     for (i=0; i<2; i++) {
4       inp[i] = mySerial.read();
5     }
6     }
7     for(j = 0; j < sizeof(inp); j++)
8     {
9       Serial.print("data diterima : ");Serial.println(inp[j]);
```

```
10    }
11    byte hasil = hash(inp,sizeof(inp));
12    if(hasil==0){
13      Serial.print("Tidak ada error pada data! ");
14    }
15    else{
16      Serial.print("Terdapat error pada data! ");
17    }
18    Serial.print("hasil validasi data = ");Serial.println(hasil);
19    delay(2000);
20  }
```

Baris pertama pada Tabel 5.8 merupakan perulangan `while` jika terdapat data pada serial sebanyak 2 data maka akan dibaca dan dimasukkan kedalam variabel array `inp[]`. Baris selanjutnya adalah perulangan `for` untuk mencetak data yang dibaca sebelumnya, setelah itu akan dilakukan pemanggilan fungsi `hash` dengan parameter array `inp[]` dan `sizeof(inp)` yang merupakan fungsi untuk menghitung panjang array `inp[]`. Baris selanjutnya merupakan perbandingan `if` jika hasil dari fungsi `hash` tersebut bernilai 0 maka akan mencetak *string* "Tidak ada *error* pada data!" pada serial monitor jika tidak bernilai 0 maka akan mencetak *string* "Terdapat error pada data!".

BAB 6 PENGUJIAN DAN ANALISIS

Pada bab pengujian ini akan dibahas mengenai pengujian dan analisis dari fungsional sistem yang terdiri dari pengujian fungsional komunikasi serial arduino dan pengujian fungsional sistem *error detection* dan pengujian waktu yang dibutuhkan sistem *error detection* untuk melakukan pendeteksian kesalahan.

6.1 Pengujian Fungsional Komunikasi Serial Arduino

Pengujian fungsional komunikasi *serial* arduino merupakan pengujian fungsional yang dilakukan menggunakan komunikasi serial arduino dengan pin 10 dan pin 11 yang digunakan sebagai pin RX, TX.

6.1.1 Tujuan

Pengujian ini bertujuan untuk memeriksa apakah arduino penerima mampu membaca data yang dikirimkan oleh arduino pengirim, data yang dikirim oleh arduino pengirim adalah data *codeword* berupa numerik.

6.1.2 Prosedur Pengujian

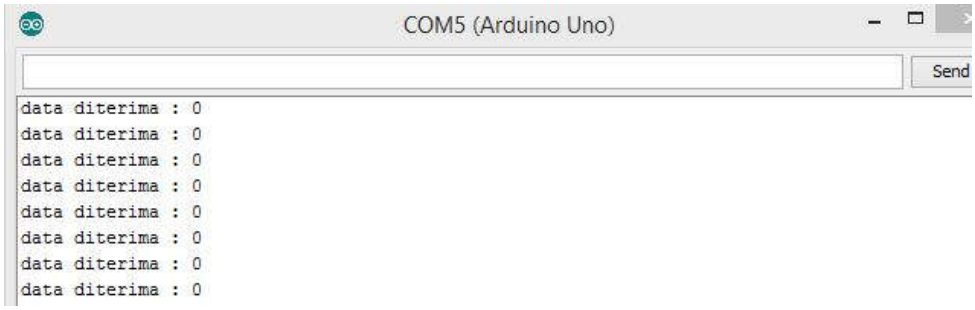
Untuk dapat menguji fungsionalitas komunikasi *serial* arduino dibutuhkan prosedur-prosedur pengujian sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu 2 buah arduino uno dan kabel penghubung atau *jumper male to male*.
2. Menyambungkan pin 10 arduino pengirim dengan pin 11 arduino penerima dengan kabel *jumper*, dan menyambungkan pin 11 arduino pengirim dengan pin 10 arduino penerima.
3. Membuka software Arduino IDE lalu ketik dan *upload source code*.
4. Buka serial monitor pada Arduino IDE lalu ketikkan masukan yang berupa numerik pada serial monitor arduino 1. Buka serial monitor pada arduino 2 dan kita akan melihat data yang telah dimasukkan pada arduino 1 telah diterima oleh arduino 2.

6.1.3 Hasil dan Analisis

Pengujian komunikasi serial arduino dilakukan dengan cara membuka *serial monitor* yang terdapat di IDE Arduino pada setiap arduino seperti pada Gambar 6.1 dan Gambar 6.2.

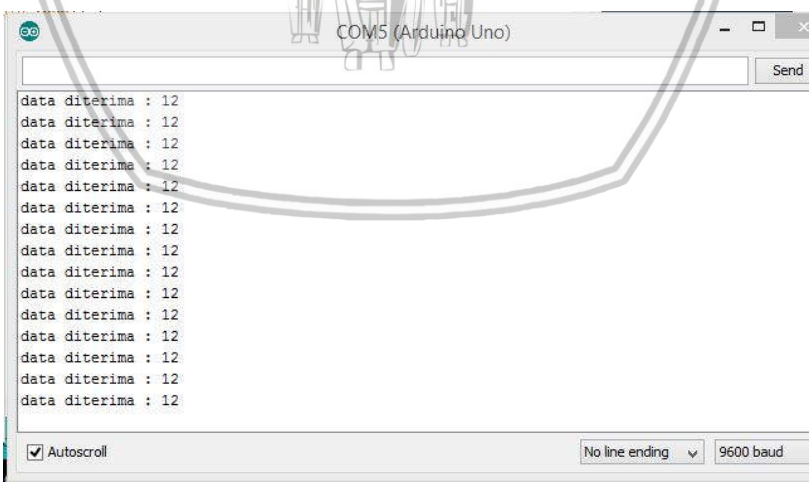
Gambar 6.1 Serial monitor pada arduino pengirim sebelum diberi masukan



Gambar 6.2 Serial monitor pada arduino penerima sebelum diberi masukan

Pada Gambar 6.3 adalah ketika pengguna memasukkan data pada *serial monitor* Arduino pengirim, pada Gambar 6.4 ketika data diterima oleh Arduino penerima.

Gambar 6.3 Serial monitor pada arduino pengirim setelah diberi masukan



Gambar 6.4 Serial monitor pada arduino pengirim sebelum diberi masukan

Pada Gambar 6.3 ditunjukkan bahwa arduino pengirim mengirim data bernilai 12 dan menampilkan data tersebut melalui serial monitor dan pada Gambar 6.4 menunjukkan bahwa arduino penerima dapat menerima data yang bernilai 12 yang mana data tersebut sama dengan data yang dikirim oleh arduino pengirim.

6.2 Pengujian Fungsional Sistem Error Detection

Pengujian fungsional sistem *error detection* menggunakan metode *cyclic redundancy code* (CRC) merupakan pengujian yang dilakukan untuk mengetahui apakah sistem dapat mendeteksi jika ada kesalahan pada data. Pada sub-bab pengujian fungsional sistem *error detection* ini peneliti melakukan 2 langkah pengujian yaitu pengujian dengan data yang tidak ada kesalahan dan pengujian dengan data yang ada kesalahan. Untuk pengujian fungsional sistem *error detection* ini menggunakan polinomial untuk data 8-bit yaitu 0x11D, 0xD5, 0x07, penggunaan dari polinomial untuk data 8-bit ini dikarenakan Arduino Uno untuk komunikasi datanya hanya bisa mengirim data maksimal 10-bit.

6.2.1 Tujuan

Pengujian ini bertujuan untuk memeriksa apakah sistem *error detection* dapat mendeteksi *error* pada data jika diberikan data yang tidak ada kesalahan maupun data yang mengalami kesalahan.

6.2.2 Prosedur Pengujian

Untuk dapat menguji fungsionalitas sistem *error detection* pada komunikasi serial arduino dibutuhkan prosedur-prosedur pengujian sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu 2 buah arduino uno dan kabel penghubung atau *jumper male to male*.
2. Menyambungkan pin 10 arduino pengirim dengan pin 11 arduino penerima dengan kabel *jumper*, dan menyambungkan pin 11 arduino pengirim dengan pin 10 arduino penerima.
3. Membuka software Arduino IDE lalu ketik dan *upload source code*.
4. Buka serial monitor pada Arduino IDE lalu ketikkan masukan yang berupa numerik pada serial monitor arduino 1. Buka serial monitor pada arduino 2 dan kita akan melihat data yang telah dimasukkan pada arduino 1 telah diterima oleh arduino 2.
5. Pada serial monitor terdapat pilihan polinomial dan pilihan untuk mencoba kesalahan pada data atau tidak. Untuk polinomial kedua Arduino harus mempunyai polinomial yang sama.

6.2.3 Hasil dan Analisis

Untuk pengujian pertama kita akan menguji apakah sistem *error detection* dapat mengetahui benar bahwa tidak ada *error* jika kita memasukkan data yang tidak ada kesalahan. Untuk hasil pengujian dapat dilihat pada Gambar 6.5 dan Gambar 6.6.

```
Coba data error ?
1. Tidak
2. Ya
Tidak mencoba data error!
========================================
data : 0
checksum didapatkan : 0
data : 12
checksum didapatkan : 3
```

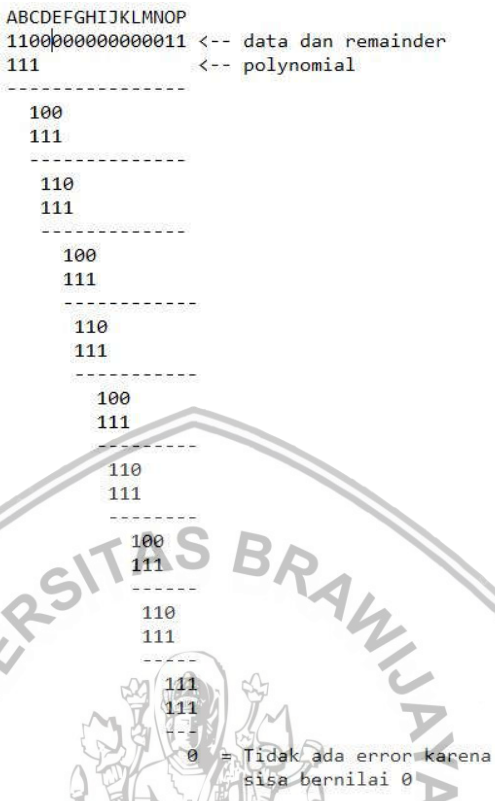
Gambar 6.5 Pengujian dengan tidak ada kesalahan pada arduino pengirim

Pada Gambar 6.5 menunjukkan bahwa ketika arduino penerima menerima masukkan bernilai 12 akan mendapatkan *remainder* atau *checksum* bernilai 3, data yang tercetak pada *serial monitor* arduino pengirim akan otomatis dikirim ke arduino penerima.

Gambar 6.6 Pengujian dengan tidak ada kesalahan pada arduino penerima

Pada Gambar 6.6 menunjukkan bahwa arduino menerima data bernilai 12 dan 3 dan langsung akan dilakukan operasi LFSR dan *polynomial division* dan mendapatkan *remainder* sebesar 0 yang didapat dari data 12 dan 3 dijadikan bentuk biner dan didapatkan biner 1100 dan 11, biner 1100 dilakukan *append* biner sehingga menjadi 8bit biner dan hasilnya 11000000, hal ini dilakukan juga pada biner 11 sehingga kedua data tersebut menjadi 1100000000000011 dan dilakukan operasi pembagian xor dengan polinomial 0x7 yang memiliki biner 111 operasi pembagian dilakukan dengan bagian paling kiri jika *most significant bit* dari data bernilai 1 maka dilakukan xor dengan polinomial jika *most significant bit* bernilai 0 maka akan dilakukan operasi geser 1 bit pada data. Operasi pembagian dikakukan berulang-ulang hingga biner data habis dan menghasilkan sisa hasil bagi atau *remainder* jika remainder bernilai 0 maka data tersebut tidak ada pergantian bit atau kesalahan tetapi jika remainder tersebut bernilai selain dari 0 maka data tersebut mengalami pergantian bit atau terdapat kesalahan. Untuk perhitungan manual dari komputasi metode ini dapat dilihat pada Gambar 6.7,sisa hasil bagi

dari data yang bernilai 0 menunjukkan bahwa pada data tersebut tidak ada kesalahan.



Gambar 6.7 Proses perhitungan deteksi kesalahan dengan tidak ada kesalahan pada data

Tabel 6.1 Tingkat Keakurasian sistem dalam mendeteksi data tidak salah

No percobaan input	Hasil deteksi sistem	Ketepatan sistem
1	Tidak ada kesalahan	Tepat
2	Tidak ada kesalahan	Tepat
3	Tidak ada kesalahan	Tepat
4	Tidak ada kesalahan	Tepat
5	Tidak ada kesalahan	Tepat
6	Tidak ada kesalahan	Tepat
7	Tidak ada kesalahan	Tepat
8	Tidak ada kesalahan	Tepat
9	Tidak ada kesalahan	Tepat
10	Tidak ada kesalahan	Tepat

Persentase keakurasian	100%

Pada Tabel 6.1 menjelaskan tentang tingkat keakurasian dan rasio kesalahan pada sistem pendeteksi yang dilakukan dengan memasukkan 10 data masukan dan menganalisa output sistem dan membandingkan dengan hasil sebenarnya, pada tabel 6.1 sistem dengan tidak ada kesalahan pada data mendapatkan hasil persentase tingkat keakurasian 100 persen, yang didapat dari hasil output – hasil sebenarnya lalu dibagi dengan hasil sebenarnya dan dikalikan 100%.

Untuk pengujian selanjutnya melakukan pengujian dengan adanya kesalahan pada data. Kesalahan yang diuji adalah jika urutan data yang diterima oleh arduino penerima tertukar antara *remainder* dan data, yang hasil dari pengujian ini dapat dilihat pada Gambar 6.8 dan pada Gambar 6.9.

```
2. CRC-8 : 0xD5
3. CRC-8-CCITT : 0x07
 Polynomial digunakan : 0x07

==========================================
Coba data error ?
1. Tidak
2. Ya
Mencoba data error!
==========================================
data : 0
checksum didapatkan : 0
data : 12
checksum didapatkan : 3
```

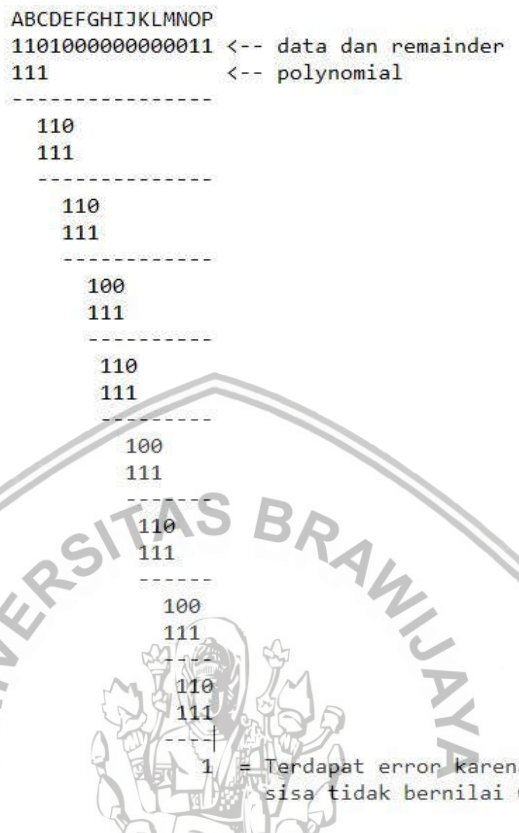
Gambar 6.8 Pengujian dengan adanya kesalahan pada arduino pengirim

```
Pilih polinomialnya :
1. CRC-8-Dallas/Maxim : 0x31
2. CRC-8 : 0xD5
3. CRC-8-CCITT : 0x07
 Polynomial digunakan : 0x07
data diterima : 13
data diterima : 3
Terdapat error pada data! hasil validasi data = 1
data diterima : 13
data diterima : 3
Terdapat error pada data! hasil validasi data = 1
```

Gambar 6.9 Pengujian dengan adanya kesalahan pada arduino penerima

Pada Gambar 6.8 menampilkan jika pengguna memasukkan nilai 12 dan mendapatkan *remainder* 3 dan otomatis dikirim ke arduino penerima, tetapi pada arduino penerima yang ditunjukkan oleh Gambar 6.8 data tersebut mengalami kesalahan perubahan bit yang seharusnya 12 menjadi 13, maka akan terbaca

adanya kesalahan pada data yang diterima tersebut. Proses perhitungan sehingga sistem dapat mendeteksi adanya error pada data tersebut ditunjukkan pada Gambar 6.10.



Gambar 6.10 Proses perhitungan deteksi kesalahan dengan adanya kesalahan pada data

Pada Gambar 6.10 adalah proses perhitungan manual sistem pendeteksi kesalahan sehingga mengetahui bahwa data yang diterima tersebut terdapat kesalahan. Sistem melakukan operasi *linear feedback shift register* dan pembagian polinomial hingga data habis dan mendapatkan sisa hasil bagi atau *remainder* bernilai 1 sehingga data tersebut dinyatakan terdapat kesalahan.

Tabel 6.2 Tingkat Keakurasian sistem dalam mendeteksi data salah

No percobaan input	Hasil deteksi sistem	Ketepatan sistem
1	Ada kesalahan	Tepat
2	Ada kesalahan	Tepat
3	Ada kesalahan	Tepat
4	Ada kesalahan	Tepat
5	Ada kesalahan	Tepat
6	Ada kesalahan	Tepat

7	Ada kesalahan	Tepat
8	Ada kesalahan	Tepat
9	Ada kesalahan	Tepat
10	Ada kesalahan	Tepat
Persentase keakurasian		100%

Pada Tabel 6.2 menjelaskan tentang tingkat keakurasian dan rasio kesalahan pada sistem pendeteksi yang dilakukan dengan memasukkan 10 data masukan dan menganalisa output sistem dan membandingkan dengan hasil sebenarnya. Sedangkan pada Tabel 6.1 sistem dengan adanya kesalahan pada data mendapatkan hasil persentase tingkat keakurasian 100 persen, yang didapat dari hasil output – hasil sebenarnya lalu dibagi dengan hasil sebenarnya dan dikalikan 100%.

6.3 Pengujian Waktu Sistem

Pengujian waktu *delay* yang dibutuhkan sistem pada *interface* protokol komunikasi seperti I2C (*Inter-grated Circuit)*, SPI (*Serial Paralel Interface)*, dan UART (*Universal Asynchrounous Receiver-Transmitter)*. Pengujian dilakukan pada sistem yang menggunakan kesalahan pada data dan tidak ada kesalahan pada data.

6.3.1 Tujuan

Melakukan pengujian pada sistem dan menghitung waktu yang dibutuhkan sistem *error detection* terhadap komunikasi *interface* yang digunakan seperti I2C, UART, dan SPI. Juga melakukan analisis apakah terdapat perbedaan waktu jika menggunakan polinomial yang berbeda terhadap komunikasi *interface* tersebut.

6.3.2 Prosedur Pengujian

Untuk dapat menguji waktu yang dibutuhkan oleh sistem *error detection* pada komunikasi serial arduino dibutuhkan prosedur-prosedur pengujian sebagai berikut:

1. Menyiapkan alat-alat yang dibutuhkan, yaitu 2 buah arduino uno dan kabel penghubung atau *jumper male to male*.
2. Untuk UART menyambungkan pin 10 arduino pengirim dengan pin 11 arduino penerima dengan kabel *jumper*, dan menyambungkan pin 11 arduino pengirim dengan pin 10 arduino penerima. Untuk sistem I2C menggunakan pin A4 *master* tersambung dengan pin A4 *slave* dan pin A5 *master* tersambung dengan pin A5 *slave*, lalu menyambungkan *pin ground master* dengan *pin ground slave.* Untuk sistem dengan SPI menggunakan pin 13 *master* dengan pin 13 *slave*, pin 12 *master* dengan pin 12 *slave*, pin 11 *master* dengan pin 11 *slave*, dan pin 10 *master* dengan pin 10 *slave*, juga menyambungkan antar pin *ground master* dengan pin *ground slave* dan pin vcc *master* dengan pin vcc *slave*.

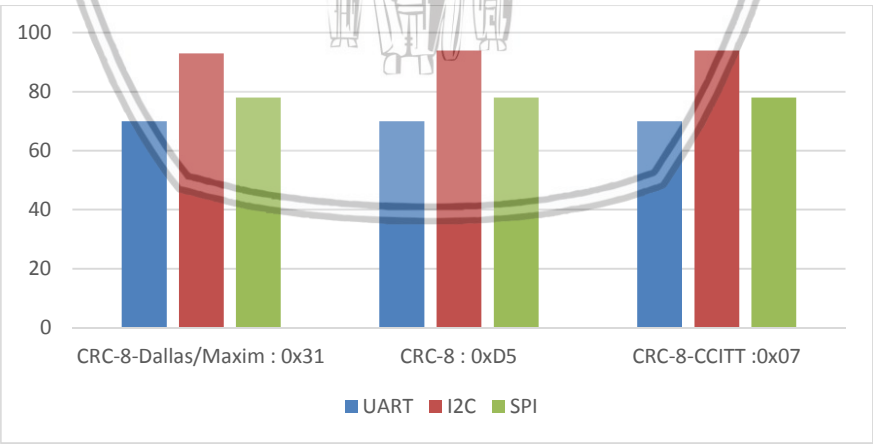
3. Membuka software Arduino IDE lalu ketik dan *upload source code*.
4. Buka serial monitor pada Arduino IDE lalu ketikkan masukan yang berupa numerik pada serial monitor arduino 1. Buka serial monitor pada arduino 2 dan kita akan melihat data yang telah dimasukkan pada arduino 1 telah diterima oleh arduino 2.
5. Pada serial monitor terdapat pilihan polinomial dan pilihan untuk mencoba kesalahan pada data atau tidak. Untuk polinomial kedua Arduino harus mempunyai polinomial yang sama.

6.3.3 Hasil dan Analisis

Pengujian waktu dilakukan dengan menghitung waktu yang dibutuhkan oleh sistem dengan fungsi millis() pada arduino, fungsi millis() pada arduino pengirim diletakkan pada saat data dimasukkan hingga data tersebut dikirim. Pada sisi penerima fungsi millis() diletakkan pada saat sebelum menerima data sampai data tersebut dideteksi. Hasil pengujian waktu *delay* antar *interface* dapat dilihat pada Tabel 6.3:

Tabel 6.3 Waktu delay interface

Polinomial	UART (ms)	I2C (ms)	SPI (ms)
CRC-8-Dallas/Maxim : 0x31	70	93	78
CRC-8 : 0xD5	70	94	78
CRC-8-CCITT :0x07	70	94	78



Gambar 6.11 Grafik Pengujian Waktu

Tabel 6.3 menunjukkan hasil pengujian waktu sistem yang diperlukan sistem ini dengan protokol komunikasi UART, I2C, dan SPI. Waktu ini didapat dari penjumlahan waktu pada sisi pengirim, waktu pengiriman, dan waktu saat data

tersebut dideteksi kesalahannya. Pada pengujian waktu sistem didapatkan hasil waktu UART dengan 3 polinomial yang digunakan memiliki waktu yang sama yaitu membutuhkan waktu 70ms. Untuk protokol I2C membutuhkan waktu untuk polinomial 0x31 yaitu selama 93ms, dan untuk polinomial yang lain membutuhkan waktu yang sama yaitu 94ms. Selanjutnya hasil pengujian waktu yang dibutuhkan protokol SPI untuk melakukan fungsi sistem yaitu selama 78ms untuk 3 polinomial yang berbeda. Dari hasil pengujian waktu tersebut didapatkan grafik seperti pada Gambar 6.11 dan dilakukan hasil analisis bahwa protol UART memiliki waktu sistem yang lebih cepat dari 2 protokol lainnya yaitu sebesar 70ms, dan protokol I2C memiliki waktu yang lebih lama dari 2 protokol lainnya.

BAB 7 PENUTUP

Pada bab 7 ini berisi kesimpulan dan saran dari penelitian ini yang berjudul "Implementasi Error Detection System Pada Komunikasi Serial Arduino Menggunakan Metode Cyclic Redundancy Check (CRC)". Kesimpulan berisi rangkuman dari keseluruhan penelitian yang berisi poin-poin penting pada penelitian ini. Lalu pada saran berisi tentang saran untuk penelitian-penelitian lain yang ingin mengembangkan penelitian ini menjadi yang lebih baik.

7.1 Kesimpulan

Kesimpulan ini diambil dari hasil perancangan, implementasi, dan pengujian sistem. Pada penelitian ini diambil kesimpulan sebagai berikut:

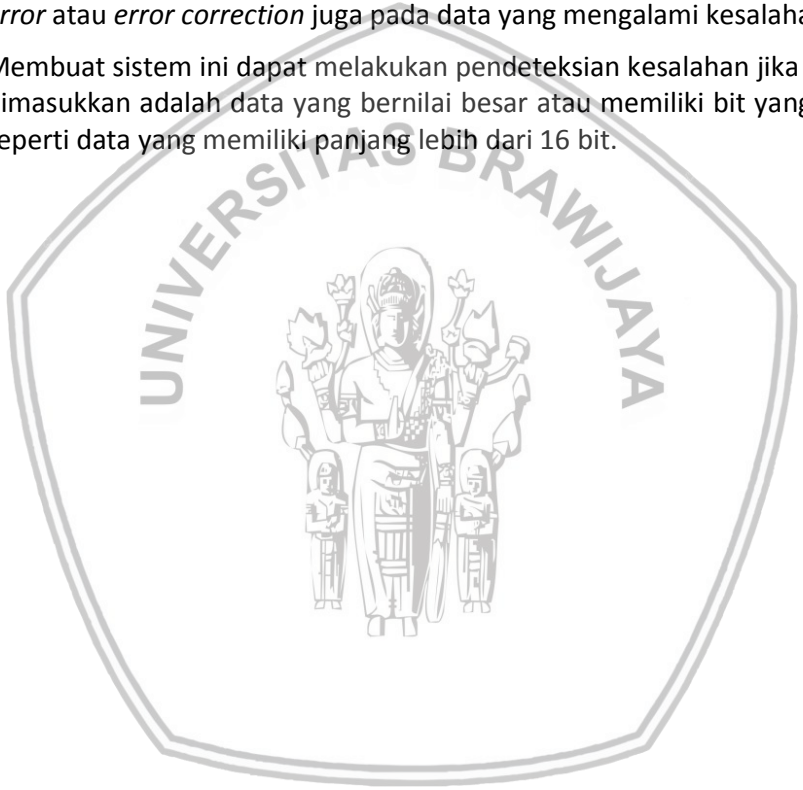
1. Sistem *error detection* dengan menggunakan metode *Cyclic Redundancy Code* ini diimplementasikan pada komunikasi serial antara 2 arduino melalui pin RX TX.
2. Kode program sistem *error detection* dibuat dengan bahasa pemrograman c++, yaitu bahasa pemrograman yang didukung oleh *board* arduino. Kode program ini berisikan fungsi yang berisi komputasi input pengguna agar menjadi *remainder*, dan main program yang berisi pembacaan masukan dari pengguna dan pengiriman *codeword*.
3. Sistem *error detection* menggunakan metode CRC memiliki cara kerja yakni melakukan operasi XOR dan geser untuk mendapatkan *remainder*nya, jika biner data memiliki *most significant bit* (MSB) bernilai 1 maka akan dilakukan operasi XOR dengan polinomial nya, jika biner data memiliki MSB bernilai 0 maka akan dilakukan operasi geser, operasi ini dilakukan terus menerus sampai biner data habis dan meninggalkan sisa atau *remainder.* Jika sudah didapatkan *remainder*, maka biner *remainder* tersebut akan digabungkan dengan biner data dan menjadi *codeword*, dan *codeword* inilah yang akan dikirimkan kepada penerima. Pada sisi penerima akan dilakukan komputasi yang sama seperti pada sisi pengirim hingga mendapatkan *remainder*, dan jika *remainder* tersebut bernilai 0 maka tidak ada *error* pada data, sebaliknya jika selain dari nilai 0 maka data tersebut mengalami kesalahan atau *error*.
4. Pada saat pengujian sistem *error detection* dengan metode *cyclic redundancy check* dapat mendeteksi adanya kesalahan pada data seperti data tersebut mengalami pergantian bit dengan cara melakukan metode LFSR dan pembagian polinomial seperti pada saat mencari remainder pada data, tingkat keakurasian sistem untuk mendeteksi mendapatkan persentase 100% untuk 10 data masukan yang diuji.
5. Pada saat pengujian waktu menggunakan variabel polinomial dan waktu enkode dan dekode, polinomial yang digunakan yaitu CRC-8-Maxim, CRC-8, dan CRC-8-CCIT. Dari hasil analisa pengujian waktu didapatkan bahwa sistem dengan menggunakan protokol UART membutuhkan waktu yang lebih cepat

daripada Sistem dengan SPI atau I2C, dan sistem dengan protokol I2C membutuhkan waktu yang lebih lama untuk melakukan deteksi kesalahan daripada protokol UART dan SPI.

7.2 Saran

Berdasarkan hasil penelitian ini, peneliti memberikan saran kepada peneliti-peneliti lain yang akan mengembangkan sistem ini agar sistem dapat berkembang lebih baik lagi, adapun saran yang diberikan sebagai berikut:

1. Sistem *error detection* menggunakan Cyclic Redundancy Code (CRC) dapat diimplementasikan pada sistem *internet of things* sebagai pendeteksi *error* dan mengurangi kesalahan yang terjadi pada data.
2. Membuat sistem *error detection* dengan CRC ini dapat melakukan koreksi *error* atau *error correction* juga pada data yang mengalami kesalahan.
3. Membuat sistem ini dapat melakukan pendeteksian kesalahan jika data yang dimasukkan adalah data yang bernilai besar atau memiliki bit yang panjang, seperti data yang memiliki panjang lebih dari 16 bit.



DAFTAR PUSTAKA

Arduino, 2018. *Arduino.* [Online] Available at: http://www.arduino.cc

Bogart Jr, T. F., 1992. *Introduction to Digital Circuits. International Edition..* s.l.:s.n.

Chakravarty, T., 2001. Performance of Cyclic Redundancy Codes for Embedded Networks.

Engdahl, J. R. & Chung, D., 2014. Fast Parallel CRC Implementation in Software.

Fu, B. & Ampadu, P., 2009. *On Hamming Product Codes With Type-II Hybrid ARQ for On-Chip.* s.l.:s.n.

Gawande, S. & Ladhake, S. A., 2014. Design and Implementation of Parallel CRC for High Speed Application.

Lee, J.-H., 2015. CRC(Cyclic Redundancy Check) Implementation in High-Speed Semiconductor Memory.

Shannon, C. E., 1948. *A Mathematical Theory of Communication. The Bell System Technical Journal.* s.l.:s.n.

Triyogo, A., 2013. *PENGEMBANGAN ALGORITMA ENKRIPSI DEKRIPSI BERBASIS LFSR MENGGUNAKAN POLINOMIAL PRIMITIF.* s.l.:s.n.