IMPLEMENTASI *HARDWARE REDUNDANCY* PADA SISTEM AKUISISI DATA SENSOR DENGAN MENGGUNAKAN METODE *HOT STANDBY SPARING*

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:
Arie Prayogo Pangestu
NIM: 145150301111032

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

PENGESAHAN

IMPLEMENTASI *HARDWARE REDUNDANCY* PADA SISTEM AKUISISI DATA SENSOR DENGAN MENGGUNAKAN METODE *HOT STANDBY SPARING*

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Teknik

Disusun Oleh :
Arie Prayogo Pangestu
NIM: 145150301111032

Skripsi ini telah diuji dan dinyatakan lulus pada
25 Juli 2018
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Sabriansyah Rizqika Akbar, S.T, M.Eng.	Ir. Primantara Hari Trisnawan, M. Sc.
NIP: 19820809 201212 1 004	NIP: 19680912 199403 1 002

Mengetahui
Ketua Jurusan 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, Juni 2018

Arie Prayogo Pangestu

NIM: 145150301111032

KATA PENGANTAR

Puji syukur kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya sehingga laporan skripsi yang berjudul "Implementasi *Hardware Redundancy* Pada Sistem Akuisisi Data Sensor Dengan Menggunakan Metode *Hot Standby Sparing*" ini dapat terselesaikan. Penulis menyadari bahwa penyusunan laporan skripsi ini tidak akan berhasil tanpa bantuan dari beberapa pihak.Oleh karena itu, penulis ingin menyampaikan rasa hormat dan terimakasih kepada:

1. Allah yang Maha Esa yang selalu memberikan petunjuk dan hikmah dalam penulisan ini.
2. Orang Tua dan Keluarga atas nasehat, kasih sayang serta dukungan materil dan moril.
3. Sabriansyah Rizqika Akbar, S.T., M.Eng. dan Ir. Primantara Hari Trisnawan, M. Sc. selaku dosen pembimbing yang telah memberikan banyak dukungan dalam proses penyusunan skripsi ini baik teknis maupun non teknis.
4. Tri Astoto Kurniawan, S.T., M.T., Ph.D. selaku ketua jurusan Teknik Informatika.
5. Sahabat penulis Fajar, Adam, Istiqlal, Rizteg, Yudhapep, Fathia, Khurin, Kawan Fake, Teknik Komputer khususnya angkatan 2014 yang sudah memberikan semangat.
6. Eksekutif Mahasiswa Teknik Komputer khususnya divisi Infokom 2017 yang telah menjadi keluarga baru dan memberikan banyak dukungan kepada penulis.
7. Aldina Selvia selaku teman penulis yang selalu memotivasi lebih dan membantu dukungan moril serta materil selama pengerjaan skripsi ini.
8. Dan orang-orang yang selalu mendukung serta mendoakan kelancaran proses skripsi ini yang tidak bisa disebutkan satu per satu atas semua doa dan dukungannya.

Penulis menyadari dalam penyusunan skripsi ini masih banyak kekurangan, sehingga saran dan kritik yang membangun sangat penulis harapkan. Akhir kata penulis berharap skripsi ini bermanfaat bagi semua pihak yang menggunakannya.

Malang, Juni 2018
Penulis

ariepangestu.app@gmail.com

ABSTRAK

Saat ini banyak muncul produk yang digunakan untuk memonitoring suatu objek dari berbagai vendor penyedia sistem layanan. Produk yang dimaksud yaitu sensor-sensor dengan mikrokontroler yang saling terhubung untuk memberikan informasi dari apa yang sensor tangkap pada suatu kondisi objek,dalam hal ini digunakan untuk memonitoring kondisi air yang digunakan sebagai media tambak. Sensor sendiri merupakan perangkat yang berfungsi untuk men-*sensing* suatu kondisi dan dapat memberikan nilai dari kondisi tersebut. Sedangkan tambak merupakan media budidaya berupa air, baik air tawar maupun air asin, sehingga hal yang harus diperhatikan agar kualitas ikan yang dihasilkan sangat baik adalah dengan memonitoring kondisi air. Ketika air di tambak mulai keruh, kotor, suhu air meningkat dan kadar garam tidak teratur maka akan mempengaruhi kualitas dari hasil tambak tersebut. Terkadang beberapa sensor pada sistem monitoring dapat mengalami kerusakan, bisa disebabkan oleh faktor dari dalam maupun dari luar, sedangkan sistem monitoring tersebut merupakan *critical system* yang ketersediaannya sangat dibutuhkan walaupun masih terdapat kesalahan pasa sistem. Oleh karena itu dibutuhkan sebuah sistem yang mampu meredundansi sistem utama apabila terjadi kesalahan. Pada penelitian ini terdapat tiga sensor yaitu sensor suhu, kekeruhan dan sensor kadar garam pada masing-masing modul *master* dan *slave* serta menggunakan metode *Hot Standdby Sparing.* Metode ini merupakan cara penyelesaian agar modul yang akan dijadikan sebagai *slave modul* tetap hidup atau dalam kondisi *idle* walaupun masuk kedalam kondisi *sleep* dan siap menggantikan modul master ketika terdapat kesalahan sistem. Modul *master* dan *slave* dapat saling berkomunikasi dengan menggunakan media *wire*. Dari hasil pengujian, modul *master* dan modul *slave* dapat berfungsi dengan baik. Modul *slave* mampu meredundansi modul *master* ketika terdapat kesalahan pada salah satu atau beberapa sensor yang ada serta adanya gangguan daya pada modul *master*, dan sistem dapat menghindari kondisi *race* antara modul *master* dan *slave* ketika pertama kali dihidupkan.

Kata kunci: Redundancy, Fault Tolerance, Standby Redundancy, Hot Standby Sparing.

ABSTRACT

Currently there are many products that are used to monitor an object from various vendor service system providers. The intended product is group of sensors with interconnected microcontrollers to provide information of what the sensors capture in an object condition, in this case is used to monitor the condition of water used as a pond media. The sensor itself is a device that serves to sensing a condition and can provide value from the condition. While the pond is a cultivation media in the form of water, both fresh water and salt water, so things that must be considered for the quality of fish produced very well is to monitor the condition of water. When the water in the is turbid, dirty, the water temperature increases and the salt content is irregular it will affect the quality of the fishpond. Sometimes some sensors on the monitoring system can be damaged, due to by factors from within and from outside, while the monitoring system is a critical system whose availability is needed even if there is still a system error.Therefore, a system that is able to redundancy the main system in case of error. In this research there are three sensors namely temperature sensor, turbidity and salinity sensors in each master module and slave and using Hot Standdby Sparing method. This method is a way of completion so that the module that will be used as slave module remain alive or in idle condition even though it goes into sleep condition and ready to replace master module when there is system error. Master and slave modules can communicate with each other using media wire.From the test results, the master module and slave module can work properly. The slave module can redundant master modules when there is an error on one or more existing sensors as well as a power failure on the master module, and the system can avoid race conditions between master and slave modules when first turned on.

Keywords: Redundancy, Fault Tolerance, Standby Redundancy, Hot Standby Sparing.

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Monitoring saat ini menjadi salah satu kebutuhan bagi kalangan tertentu yang membutuhkan informasi mengenai data suatu objek. Salah satu objek yang sering dimonitoring ialah kondisi air. Kondisi air dapat dimonitoring dengan menggunakan bantuan sensor seperti sensor suhu, kekeruhan dan kadar garam. Terkadang terjadi kesalahan pada sensor baik karena faktor kesalahan dari luar maupun dari dalam sistem. Dampak dari kesalahan sistem sensor tersebut dapat mengakibatkan kerugian material. Hal ini diperkuat dengan adanya data tentang kerugian tambak udang dengan luas 4 hektar dapat mengakibatkan kerugian hingga puluhan juta rupiah. (Atiek, 2017)

Dalam hal ini dapat diimplementasikan beberapa metode yang mampu mentolerir adanya kesalahan pada sistem sensor. Diantaranya *software* redundansi, *hardware* redundansi, *information* redundansi, dan *time* redundansi. Perdedaan dari empat metode tersebut yaitu terletak pada inti permasalahan yang mengakibatkan adanya kesalahan pada sistem tersebut. Kesalahan dapat terjadi pada sisi *software*, *hardware*, information, maupun *time*.

Berdasarkan permasalahan yang telah dipaparkan sebelumnya, maka penulis memilih menggunakan *hardware* redundansi sebagai acuan dalam menyelesaikan kesalahan pada sistem sensor. Dalam *hardware* redundansi terdapat beberapa metode penyelesaian antara lain *hot standby spare*, dan *cold standby spare*. Perdedaan dari kedua metode tersebut yaitu terletak pada kondisi sistem cadangan. Pada metode *hot standby spare*, sistem cadangan berada pada kondisi sleep. Sedangkan pada metode *cold standby spare*, sistem cadangan berada pada kondisi mati.

Oleh karena itu, penulis memilih menggunakan *hardware* redundansi dengan metode *hot standby spare* dalam menyelesaikan permasalahan pada sistem akusisi data sensor. Dengan menerapkan mekanisme *hardware* redundansi pada sistem akuisisi data sensor ini, diharapkan mampu meningkatkan *availability* sistem saat mengalami kegagalan dari sisi sensor. Karena nilai *availability* sangat berguna bagi sensor yang ada pada sistem monitoring.

1.2 Rumusan masalah

Berdasarkan latar belakang tersebut dapat dirumuskan beberapa rumusan masalah yang dapat diangkat,antara lain sebagai berikut:

1. Bagaimana sistem dapat mendeteksi ketika adanya kegagalan atau *fault* pada sistem sensor saat salah satu atau beberapa sensor dalam keadaan mati yang terdapat pada sistem monitoring kondisi air?
2. Bagaimana sistem monitoring air dapat tetap bekerja,sedangkan terdapat kegagalan atau *fault* pada salah satu atau beberapa sensor?

3. Bagaimana mekanisme komunikasi antara modul *master* dan modul *slave* pada sistem monitoring kondisi air sehingga modul *slave* dapat meredundansi modul *master* ketika terjadi *fault* pada modul *master*?

1.3 Tujuan

Adapun tujuan penulis dalam melakukan penelitian skripsi ini adalah sebagai berikut :

1. Mengimplementasikan sistem yang dapat mendeteksi ketika adanya kegagalan pada sistem sensor saat salah satu atau beberapa sensor tiba-tiba mati.
2. Mengimplementasikan sistem yang dapat tetap bekerja saat terdapat kesalahan karena menerapkan metode *hot standby sparing* yaitu dengan menyediakan perangkat cadangan yang memiliki fungsi sama.
3. Mengimplementasikan sistem yang dapat saling berkomunikasi sehingga ketika terdapat kesalahan pada modul *master*, modul *slave* mampu meredundansi modul *master* yang mengalami gangguan.

1.4 Manfaat

Manfaat penelitian yang diharapkan adalah sebagai berikut :

Diharapkan sistem dapat melakukan redundasi dari sisi *hardware* apabila terdapat kesalahan pada sistem sensor. Serta implementasi dari metode *Hot Standby Sparing* dapat diterapkan pada semua sistem yang menggunakan komunikasi serial berbasis Arduino.

1.5 Batasan masalah

Agar penelitian lebih fokus dan tidak meluas lebih jauh dari pembahasan yang dimaksud, dalam proposal skripsi ini penulis membatasinya pada ruang lingkup penelitian sebagai berikut :

a. Penelitian hanya melakukan deteksi kegagalan dan merancang sistem yang dapat menjadi peragkat *slave* atau cadangan bagi sistem akuisisi data sensor yang sudah diteliti sebelumnya .
b. Penelitian menggunakan dua mikrokontorler,satu sebagai modul *master* dan satu mikrokontroler yang digunakan sebagai perangkat modul *slave*.
c. Penelitian hanya meneliti ketika terjadi *fault* pada salah satu atau beberapa sensor yang ada di modul *master* atau *fault* pada modul *master* itu sendiri karena kondisi mati mendadak saat sumber daya mati.
d. Pada modul *slave*,kondisi sensor-sensor dianggap selalu aktif dan tidak mengalami masalah apapun.
e. Penelitian hanya fokus pada penelitian tentang redundansi sistem dan tidak terfokus pada hasil sensing sensor-sensor yang ada pada sistem.

1.6 Sistematika pembahasan

Penelitian ini disusun dalam tujuh bab pembahasan sebagai acuan dalam berfikir secara sistematis, adapun rancangan sistematika pembahasan laporan skripsi ini sebagai berikut :

BAB I PENDAHULUAN

Pada bab ini merupakan gambaran umum isi penelitian yang terdiri dari latar belakang, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian, dan sistematika pembahasan.

BAB II LANDASAN KEPUSTAKAAN

Pada bab ini berisi tentang tinjauan pustaka yang membahas tentang penelitian-penelitian sebelumnya sebagai rujukan dalam penelitian ini serta dasar teori yang berhubungan dengan penelitian seperti teori tentang *Hardware Redundancy* dan *Standby Redundancy*.

BAB III METODOLOGI PENELITIAN

Pada bab ini berisi gambaran umum tentang studi literatur penelitian,rekayasa kebutuhan sistem,perancangan dan implementasi sistem,pengujian dan analisis hasil penelitian serta pengambilan kesimpulan.

BAB IV REKAYASA KEBUTUHAN

Pada bab ini berisi penjelasan khusus terkait analisis kebutuhan sistem yang dibutuhkan dalam perancangan sistem meliputi kebutuhan baik kebutuhan perangkat keras dan perangkat lunak,dan kebutuhan fungsional merupakan kebutuhan penting yang digunakan agar tujuan dari pembuatan sistem tercapai. .

BAB V PERANCANGAN DAN IMPLEMENTASI

Bab ini menjelaskan secara detail bagaimana perancangan sistem dilakukan sesuai dengan metode yang digunakan serta studi literatur yang ada dan kemudian setelah itu mengimplementasikan sistem.

BAB VI PENGUJIAN DAN ANALISIS

Bab ini berisi tentang bagaimana sekenario pengujian dari suatu sistem dan melakukan analisis sehingga dapat diketahui apakah kebutuhan fungsional sudah tercapai atau belum tercapai.

BAB VII PENUTUP

Bab ini berisi tentang uraian kesimpulan yang diperoleh dari perancangan sistem.Kemudian memberikan saran dan masukan dari hasil pengujian sehingga dapat dijadikan sebagai rujukan untuk penelitian serupa selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Pada landasan kepustakaan terdapat 2 macam pembahasan yaitu tinjauan pustaka yang berisi paparan beberapa penelitian terdahulu yang dijadikan sebagai refrensi, objek dan dasar teori dari berbagai sumber pustaka terkait dengan teori dan metode yang digunakan dalam penelitian "Implementasi *Hardware Redundancy* Pada Sistem Akuisisi Data Sensor Dengan Menggunakan Metode *Hot Standby Sparing*" .

2.1 Tinjauan Pustaka

Berikut merupakan beberapa penelitian terkait sistem kontrol prosesor tambak udang yang menjadi landasan dari sistem yang akan dibuat. Penelitian yang terkait yaitu, pertama penelitian yang dilakukan oleh (Agustiningsih, 2016) yang berjudul "Perancangan Perangkat Monitoring Kualitas Air Pada Kolam Budidaya Berbasis Web *Localhost*".Penelitian ini membahas tentang monitoring kualitas air dilihat dari segi kadar garam, suhu air dan kesadahan air. Sistem tersebut dibuat agar dapat memonitoring kualitas air yang dapat dipantau melalui web *localhost*.

Penelitian berikutnya yaitu penelitian yang dilakukan oleh (Sambora, 2016) yang berjudul "Monitoring Kualitas Air Pada Budidaya Udang Berbasis ATMEGA328 yang Terkonfigurasi *Bluetooth* HC-05". Penelitian ini membahas tentang kualitas air dari segi kadar garam dan suhu air. Sistem tersebut kemudian dapat mengirimkan data yang diperoleh sensor melalui konektivitas *bluetooth* sehingga dapat dimonitoring menggunakan *smartphone*.

Dari dua penelitian yang pernah dilakukan oleh Agustiningsih dan Sambora belum ada dari keduanya yang menerapkan konsep *redundancy* sebagai antisipasi adanya *fault* pada sistem,sehingga penulis memiliki ide untuk membuat sistem redundasi untuk monitoring kondisi air dengan menggunakan metode *hot stadby sparing* sebagai redundansi dari sistem monitoring kondisi air tambak.

2.2 Dasar Teori

Dalam bab dasar teori akan terdapat penjelasan yang mendukung pembuatan sistem berupa pengertian, cara kerja, spesifikasi, karakteristik dan lainnya untuk memudahkan pembaca memahami isi dari penelitian ini.

2.2.1 *Hardware Redundancy*

Hardware Redundancy dicapai dengan menyediakan dua atau lebih salinan fisik komponen perangkat keras. Misalnya, sistem mungkin berisi beberapa prosesor, beberapa memori, beberapa bus, atau beberapa sumber daya. Pada teknik lainnya, seperti penggunaan lebih banyak komponen handal, kualitas kontrol manufaktur, pengujian, penyederhanaan desain, dan lain-lain sudah tidak dapat dipergunakan, *Hardware Redundancy* mungkin satu-satunya cara untuk memperbaiki *reliability* suatu sistem. Misalnya, dalam situasi di mana peralatan

tidak bisa dirawat, seperti satelit komunikasi, komponen yang berlebih memungkinkan untuk tidak terganggu waktu operasi untuk waktu yang berkepanjangan (Dubrova, 2013).

Ada tiga jenis *Hardware Redundancy* yaitu pasif, aktif, dan *hybrid*. Pasif redundansi mencapai toleransi kesalahan dengan menutupi kesalahan yang terjadi tanpa membutuhkan tindakan dari sistem atau operator. Redundansi aktif membutuhkan deteksi kesalahan sebelum ditolerir. Setelah dideteksi kesalahannya, diketahui lokasinya, pemulihan dilakukan untuk menghilangkan komponen yang salah dari sistem. Teknik aktif mengharuskan sistem dihentikan dan dikonfigurasi ulang untuk mentoleransi kesalahan. Redundansi *hybrid* menggabungkan pendekatan pasif dan aktif. Penyisipan kesalahan digunakan untuk mencegah hasil yang salah. Deteksi kesalahan, lokasi, dan pemulihan digunakan untuk mengganti komponen yang salah dengan cadangan. *Hybrid* redundansi memungkinkan rekonfigurasi tanpa *downtime* sistem.

2.2.2 Metode *Redundancy*

Redundansi dalam implementasinya memiliki beberapa metode, teknik, dan terminologi. Terdapat tiga metode yang umum digunakan dalam dunia industri. Diantaranya adalah *Standby Redundancy*, *N Modular Redundancy*, dan *1:N Redundancy*. Untuk tugas akhir skripsi ini penulis menggunakan metode *hot standby redundancy* atau *hot standby sparing* .

2.2.2.1 Standby Redundancy

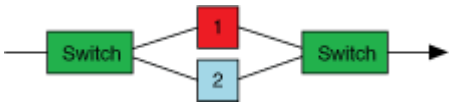
Standby Redundancy atau bisa disebut *Backup Redundancy* adalah ketika sebuah sistem memiliki unit sekunder yang identik dengan unit primer. Unit Sekunder secara tipikal tidak memonitor sistem, namun hanya bersifat cadangan. Unit sekunder atau unit *standby* umumnya tidak tersinkronisasi dengan unit primer, maka harus dicocokkan input dan output sinyalnya ketika mengambil alih dari *Device Under Control* (*DUC*). Pendekatan ini memberi kesempatan untuk memberikan "bump" pada transfer, yang berarti sinyal sekunder dapat mengirimkan sinyal kontrol ke *DUC* yang tidak selaras dengan sinyal kontrol terakhir yang berasal dari unit utama.

Mekanisme ini memerlukan pihak ketiga untuk menjadi pengawas, yang memonitor sistem untuk memutuskan kapan kondisi peralihan terpenuhi dan memerintahkan sistem untuk mengalihkan kontrol ke unit *stanby* dan voter, yang merupakan komponen yang memutuskan kapan harus beralih dan mana unit diberi kontrol dari *DUC*. Kenaikan biaya sistem untuk jenis redundansi ini biasanya sekitar 2X atau kurang tergantung pada biaya pengembangan perangkat lunak Anda. Dalam redundansi *standby* ada dua tipe dasar, *Cold Standby* dan *Hot Standby*.

a. *Cold Standby*

   Dalam *cold standby*, unit sekunder dimatikan, sehingga menjaga *reliability* unit. Kelemahan dari metode ini adalah *downtime* lebih besar daripada metode *hot standby*, karena sistem harus menyalakan unit *standby* dan

membawanya *online* ke keadaan yang diketahui. Hal ini lebih memberikan *challenge* untuk masalah sinkronisasi, namun dilakukan dengan waktu yang panjang dibutuhkan untuk membawa unit *standby* secara *online*, biasanya akan terjadi masalah yang cukup besar ketika peralihan.



Gambar 2.1 Blok diagram cold standby

Sumber: Dubrova (2012)

b. Hot Standby

Dalam metode *hot standby*, unit sekunder dinyalakan dan secara opsional dapat memantau *DUC*. Jika menggunakan unit sekunder sebagai pengawas dan / atau voter untuk memutuskan kapan harus beralih, Anda dapat menghilangkan kebutuhan pihak ketiga untuk pekerjaan ini. Metode ini tidak menjaga *reliability* unit *standby* seperti metode *cold standby*. Namun, ini memperpendek *downtime*, yang pada gilirannya meningkatkan *availability* sistem.

Gambar 2.2 Blok diagram hot standby

Sumber: Dubrova (2012)

2.2.3 Arduino UNO

Arduino UNO adalah platform elektronik yang *open-source* berbasis *hardware* dan *software* yang mudah digunakan. *Board* Arduino dapat membaca *input* dari berbagai macam sensor seperti cahaya, sidik jari, bahkan pesan twitter lalu memprosesnya menjadi *output* untuk mengaktifkan aktuasi sepeti motor, lampu *LED*, dan lain – lain. *Board* Arduino dapat di beri perintah dengan memberikan satu set perintah pada *microcontroller* yang ada pada *board* Arduino. Untuk melakukannya dapat menggunakan Arduino programming language dan Arduino *Software (IDE*) (*arduino.cc,* 2017*).*

Arduino lahir di *Ivrea Interaction Design Institute* sebagai alat yang mudah digunakan untuk pembuatan prototipe dengan cepat. Arduino ditujukan untuk siswa yang tidak memiliki latar belakang dalam bidang elektronika dan pemrograman. Begitu sampai pada komunitas yang lebih luas, *board* Arduino mulai dikembangkan untuk menyesuaikan diri dengan kebutuhan dan tantangan baru, dari papan 8-bit sederhana hingga produk untuk aplikasi IoT, 3D *painting*,

dan *embedded environment*. Semua *board* Arduino benar-benar *open-source*, memberdayakan pengguna untuk membangunnya secara mandiri dan menyesuaikannya dengan kebutuhan khusus mereka. Perangkat lunak juga *open source*, dan berkembang melalui kontribusi pengguna di seluruh dunia. Berikut spesifikasi dari *board* Arduino UNO:

Chip mikrokontroller	ATmega328P
Tegangan operasi	5V
Tegangan input (rekomendasi, via jack DC)	7V - 12V
Tegangan input (limit, via jack DC)	6V - 20V
Digital I/O pin	14 buah, 6 diantaranya PWM
Analog Input pin	6 buah
Arus DC per pin I/O	40 mA
Arus DC pin 3.3V	50 mA
Memori Flash	32 KB, 0.5 KB digunakan bootloader
SRAM	2 KB
EEPROM	1 KB
Clock speed	16 Mhz
Dimensi	68.6 mm x 53.4 mm
Berat	25 g

Gambar 2.3 Spesifikasi Arduino UNO

Sumber: Ecadio (2017)

Pada Gambar 2.3 dapat dilihat bahwa pin yang terdapat di dalam *board* Arduino UNO yang terbagi menjadi *pin power*,*pin input* dan *Output* berupa pin analog dan digital,serta pin lain yang mendukung kerja *Microcontroller* Arduino UNO. Pada Gambar 2.4 dapat dilihat *board* Arduino UNO dengan keterangan pinnya.

Gambar 2.4 Arduino UNO

Sumber*: (www.Arduino.cc, 2015)*

Untuk penjelasan Gambar 2.4 *board* Arduino UNO dapat dilihat pada Tabel 2.1 untuk keterangan *pin power* pada Arduino UNO:

Tabel 2.1 Table Keterangan Pin Power

Pin	Keterangan
GND	*Ground* atau negatif
Vin	Pin yang digunakan jika ingin memberikan power langsung ke *board* Arduino dengan rentang tegangan yang disarankan 7V – 12V
5V	*Output* dimana pada pin tersebut mengalir tegangan 5V yang telah melalui regulator
3.3V	*Output* dimana pada pin tersebut disediakan tegangan 3.3V yang telah melalui regulator
IOREF	Pin yang menyediakan referensi tegangan *Microcontroller*. Biasanya digunakan pada *board shield* untuk memperoleh tegangan yang sesuai,apakah 5V atau 3.3V

Berikut merupakan pin *input* dan *output* pada Arduino UNO yang memiliki fungsi khusus yang dapat dilihat pada Tabel 2.2 berikut:

Tabel 2.2 Table Keterangan Pin Khusus

Kebutuhan	Keterangan Pin Khusus
Serial	Terdiri dari 2 pin : pin 0 (RX) dan pin 1 (TX) yang digunakan untuk menerima (RX) dan mengirim (TX) data serial.
External Interrupts	Yaitu pin 2 dan pin 3.Kedua pin tersebut dapat digunakan untuk mengaktifkan *interrupts*.
PWM	Pin 3,5,6,9,10, dan 11 menyediakan output PWM 8-bit dengan menggunakan fungsi analogWrite().
SPI	Pin 10(SS), 11(MOSI), 12(MISO), dan 13(SCK) mendukung komunikasi SPI dengan menggunakan SPI *Library.*
LED	Pin 13. Pada pin 13 terhubung *built-in* LED yang dikendalikan oleh digital pin 13.
TWI	Pin A4(SDA) dan pin A5(SCL) yang mendukung komunikasi TWI dengan menggunakan *Wire Library*.

2.2.4 Arduino IDE

Arduino *development* berisi editor teks untuk kode, area pesan, konsol teks, *toolbar* dengan tombol untuk fungsi-fungsi umum menulis, dan serangkaian menu. Ini menghubungkan ke perangkat keras Arduino untuk meng-*upload* program dan untuk berkomunikasi. *Software* yang ditulis menggunakan Arduino disebut sketsa. Sketsa ini ditulis dalam editor teks. Sketsa disimpan dengan ekstensi file .ino. Ini memiliki fitur untuk meng-*cut/paste* dan untuk mencari/ mengganti teks. Daerah pesan memberikan umpan balik sambil menyimpan dan

mengekspor serta menampilkan kesalahan. Konsol menampilkan teks *output* dengan lingkungan Arduino termasuk pesan *error* lengkap dan informasi lainnya. Bagian bawah sebelah kanan sudut jendela menampilkan papan saat ini dan *port serial*. Tombol-tombol toolbar memungkinkan Anda untuk memverifikasi dan mengunggah program, membuat, membuka, dan menyimpan sketsa, dan membuka monitor serial.

Gambar 2.5 Arduino IDE
Sumber *: (www.learn.adafruit.com, 2017*)

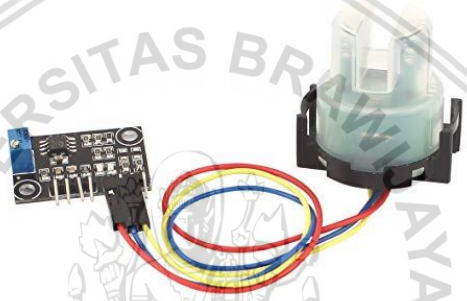
2.2.5 Sensor Suhu DS18B20

DS18B20 adalah sensor suhu digital seri terbaru dari Maxim IC (dulu yang buat adalah Dallas Semiconductor, lalu diambil alih oleh Maxim Integrated Products). Sensor ini mampu membaca suhu dengan ketelitian 9 hingga 12-bit, rentang -55°C hingga 125°C dengan ketelitian (+/-0.5°C). Setiap sensor yang diproduksi memiliki kode unik sebesar 64-Bit yang disematkan pada masing-masing chip, sehingga memungkinkan penggunaan sensor dalam jumlah besar hanya melalui satu kabel saja (single wire data bus/1-wire protocol).

Gambar 2.6 Sensor DS18B20
Sumber : (Maxim Integrated)

2.2.6 Sensor Kekeruhan Air

Turbidimeter adalah alat yang digunakan sebagai alat uji standar untuk mengetahui tingkat kekeruhan air. Keberadaan alat ini sebenarnya sudah umum dan mudah dicari. Namun, karena harganya relative mahal menjadikan alat ini hanya dimiliki oleh pihak–pihak tertentu. Untuk menguji apakah air yang kita punya mempunyai standar atau tidak harus pergi ke Laboratorium pengujian air minum, hal ini menyebabkan kurang efektif dan efisien. Dasar pembacaan kekeruhan air, pada sensor tersebut ada sejenis sensor sumber cahaya dan penangkap cahaya, yang kemudian dilewatkan ke bagian air yang akan di lakukan pengukuran atau pengecekan kekeruhan. Sensor ini bisa kita hubungkan ke perangkat pengolah instrument pengukuran seperti ke mikrokontroller ataupun ke arduino.



Gambar 2.7 Sensor Kekeruhan Air
Sumber : (*www.ge-mcs.com*, 2013)

2.2.7 Sensor Kadar Garam

Merupakan sebuah sensor yang digunakan untuk mengetahui kadar garam yang ada pada air.Sensor ini menggunakan elektroda *stainless steel*, dan memiliki *output* analog dengan *range* 0 – 5V.

Gambar 2.8 Sensor Kadar Garam
Sumber : (Depo Inovasi)

2.2.8 *Library* avr.sleep

Penggunaan instruksi *SLEEP* dapat memungkinkan aplikasi untuk mengurangi konsumsi daya secara signifikan. Perangkat AVR dapat dimasukkan ke mode *sleep* yang berbeda. Cara termudah adalah secara opsional mengatur mode sleep yang dikehendaki dengan menggunakan set_sleep_mode () (biasanya *default* ke mode diam dimana CPU ditidurkan tetapi semua timer perangkat masih berjalan), dan kemudian memanggil sleep_mode (). *Library* ini secara otomatis mengatur bit untuk mengaktifkan *sleep*, *goes to sleep*, dan membersihkan bit *enable sleep*.(*microchip.com*, 2016)

2.2.9 Interupsi

Interupsi adalah proses dalam sistem mikrokontroler yang menghentikan aliran program utama akibat terjadinya (event) trigger (pemicu) tertentu dari suatu sumber (vector) interupsi dan memaksa sistem mikrokontroler untuk mengeksekusi sub-rutin/fungsi/blok program layanan interupsi (interrupt service routine, ISR) hingga selesai (complete). Setelah program interupsi selesai dikerjakan, maka sistem mikrokontroler akan kembali melanjutkan program utama yang sebelumnya dihentikan. (*robotics-university.com*, 2015)

2.2.10 *Library Sleep Interrupt*

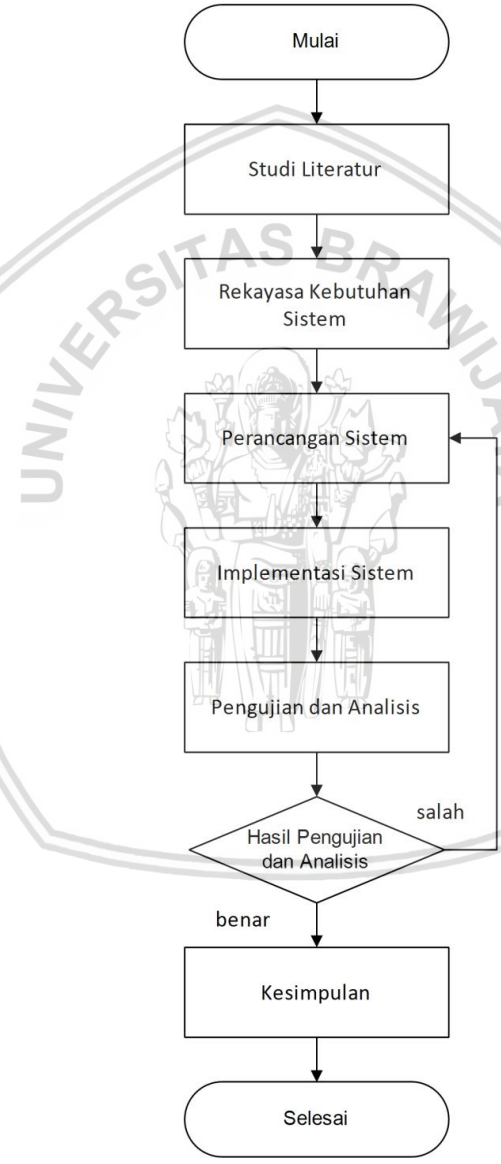
Sleep dibantu oleh interupsi. tanpa interupsi, hanya reset yang dapat membangunkan Arduino lagi. Untungnya interupsi dimasukkan sejak versi 0007 dari Arduino IDE.Pada bagian depan perangkat keras, Arduino dilengkapi dengan dua port interupsi: pin digital 2 dan 3. Jadi Arduino dapat merasakan pin-pin itu untuk suatu peristiwa untuk bangun dan melanjutkan eksekusi kode. Bahkan dimungkinkan untuk mengeksekusi kode khusus tergantung pada pin mana yang memicu untuk bangun (interupsi).

interrupt pada USART (port serial) juga akan membangunkan Arduino. Agar ini berfungsi, Arduino harus berada dalam mode POWER_MODE_IDLE, satu-satunya mode daya yang tidak menonaktifkan USART. Meskipun mode ini tidak memberikan penghematan daya yang besar, dapat menggunakan fungsi yang disediakan di *avr / power.h (power_adc_disable ()*, *power_spi_disable ()*, *power_timer0_disable ()*, *power_timer1_disable ()*, *power_timer2_disable ()*, *power_twi_disable ()*) untuk menonaktifkan modul perangkat keras untuk mencapai penghematan daya yang lebih besar. (*arduino.cc,* 2017)

BAB 3 METODOLOGI

Berikut ini merupakan suatu diagram yang menjelaskan bagaimana alur metode penelitian yang digunakan. Penelitian ini merupakan suatu jenis penelitian implementatif pada konteks penelitian pengembangan lanjut yang merupakan pengembangan terhadap sistem yang sudah pernah diteliti sebelumnya dengan menambah beberapa fitur baru. Tahapan-tahapan yang dilakukan mulai dari mencari literatur hingga menyelesaikan sistem. Diagram alir metode penelitian dapat dilihat pada Gambar 3.1 berikut ini :

.



Gambar 3.1 Alur Metodologi Penelitian

3.1 Studi Literatur

Studi literatur digunakan untuk memperoleh dan mempelajari referensi yang didapat. Referensi ini akan dijadikan sebagai panduan dalam melakukan penelitian. Referensi yang didapat berupa literatur dari berbagai bidang ilmu yang berhubungan dengan penelitian tentang "*Implementasi Hardware Redundancy Pada Sistem Akuisisi Data Sensor Dengan Menggunakan Metode Hot Standby Sparing"*, antara lain:

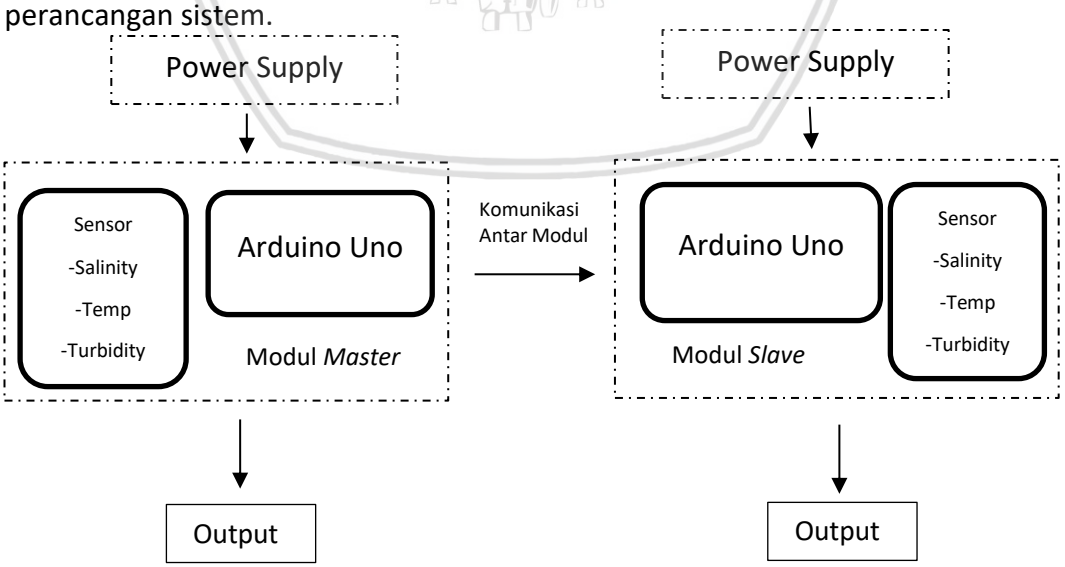
1. *Hardware Redundancy*
2. *Redundancy*
3. *Arduino*

3.2 Analisis Kebutuhan

Analisis kebutuhan dilakukan untuk mengetahui dan menentukan kebutuhan apa saja yang dibutuhkan dalam penelitian *"Implementasi Hardware Redundancy Pada Sistem Akuisisi Data Sensor Dengan Menggunakan Metode Hot Standby Sparing"*, baik kebutuhan perangkat keras dan perangkat lunak maupun kebutuhan fungsional dan non-fungsional.

3.3 Perancangan dan Implementasi

Pada tahap ini menerapkan semua tahapan dari studi literatur sampai pada tahap analisis kebutuhan dalam melakukan perancangan pengembangan penelitian *"Implementasi Hardware Redundancy Pada Sistem Akuisisi Data Sensor Dengan Menggunakan Metode Hot Standby Sparing"*. Hasil dari perancangan pengembangan kemudian direalisasikan dan diterapkan pada sistem sesuai dengan rancangan yang telah dibuat, kemudian dievaluasi kembali melalui bimbingan kepada dosen pembimbing penelitian. Berikut adalah diagram blok perancangan sistem.



Gambar 3.2 Diagram Blok Perancangan Sistem

Berdasarkan Gambar 3.2, sistem akan dirancang dengan menggunakan dua mikrokontroler yang dikomunikasikan secara serial. Komunikasi serial ini dilakukan secara *wired*. Pada kedua arduino akan dirancang sistem yang sama persis dapat menangkap hasil dari pembacaan sensor salinitas, suhu dan kekeruhan. Sehingga sistem akan dirancang dengan beberapa proses yaitu :

1. Modul *Master*.

   Pada proses ini akan didapatkan data dari sensor yang kemudian akan dikirimkan kepada *user* melalui serial monitor yang ada pada fitur aplikasi Arduino IDE.
2. Modul *Slave*.

   Pada proses ini, proses yang dilakukan sama dengan modul *master,* tetapi pada proses ini modul *slave* akan menunggu triger dari modul *master*, apabila triger bernilai 0 atau diasumsikan terjadi *fault* pada modul *master* maka modul *slave* akan mengambil alih pekerjaan untuk mengirimkan hasil dari pembacaan sensor kepada *user*.
3. Pada sisi komunikasi antar modul, tiap modul akan saling berkomunikasi menggunakan *wire* sebagai *interupt* yang dapat mengaktifkan modul *slave* ketika terdapat kesalahan pada modul *master*.

3.4 Pengujian

Pengujian dilakukan untuk menunjukkan bahwa perancangan pengembangan telah sesuai dengan apa yang dituangkan pada analisis kebutuhan sistem. Pengujian akan dilakukan dengan menggunakan skenario pengujian pada tambak buatan yang terbuat dari bak kecil berisi air, sistem akan diuji pada tambak buatan tersebut dengan cara membuat seolah-olah terjadi *fault* pada salah satu modul sehingga dapat menghasilkan output yang dapat digunkana sebagai bahan pertimbangan apakah sistem tersebut dapat bekerja sesuai dengan kebutuhan sistem.

3.5 Penutup

Tahap ini merupakan tahap akhir dari penelitian skripsi yaitu dengan memberikan kesimpulan terhadap hasil pengujian. Selain memberikan kesimpulan, saran juga dibuat sebagai rujukan untuk penelitian terkait selanjutnya agar lebih baik dan dapat lebih disempurnakan.

BAB 4 REKAYASA KEBUTUHAN SISTEM

Dalam rekayasa kebutuhan sistem berdasarkan sistem yang dibuat,tujuan utama yang ingin dicapai, yaitu :

Implementasi *Hardware* Redundansi Pada Sistem Akuisisi Data Sensor Dengan Menggunakan Metode *Hot Standby Sparing* merupakan sebuah rancangan sistem yang dirancang agar dapat meningkatkan *availability* dari sistem akuisisi data sensor.

Pada bab ini akan dijelaskan secara rinci mengenai rekayasa kebutuhan yang harus dipenuhi dalam proses perancangan hingga proses implementasi sistem. Kebutuhan yang dimaksud meliputi gambaran umum dari sistem, kebutuhan fungsional, kebutuhan perangkat keras dan kebutuhan perangkat lunak.

4.1 Gambaran Umum Sistem

Sistem yang akan dibangun adalah suatu sistem yang dapat merudundansi sistem akuisisi data dari sensor sehingga dapat meningkatkan *availability* dari sistem tersebut. Metode yang digunakan yaitu metode *Hot Standby Sparing* dimana metode tersebut akan dapat berjalan ketika mendapatkan suatu *interrupt* atau *input* yang dapat membangunkan sistem yang berada dalam keadaan *standby*.

Untuk *Output* dalam sistem dapat diketahui dengan menggunakan serial monitor yang terdapat pada fasilitas *software* Arduino sehingga dapat diketahui apakah modul *slave* sudah dapat meredundansi modul *master* atau gagal. Rekayasa pengujian dapat dilakukan dengan membuat *fault* salah satu atau semua sensor atau bahkan membuat modul *master* mati total.

4.1.1 Prespektif sistem

Sistem ini dapat dikatakan berjalan dengan benar apabila sesuai dengan yang di harapkan yaitu sistem dapat meredundansi modul primer sehingga sistem tersebut dapat tetap berjalan walaupun terdapat *fault* pada sistem primer tersebut.

4.1.2 Ruang lingkup

Ruang lingkup pada sistem ini yaitu redundansi *hardware* dari sistem akuisisi data sensor yang terdiri dari sensor suhu,sensor kadar garam dan sensor kekeruhan air. Ketiga sensor tersebut merupakan sensor yang berperan penting terhadap kualitas air sehingga digunakan penulis untuk dijadikan sebagai variabel utama sistem yang akan diredundansi karena bersifat *critical* sistem yang artinya merupakan suatu sistem penting bagi kepentingan monitoring kondisi air tambak.

4.1.3 Karakteristik pengguna

Karakteristik pengguna diperuntukkan bagi para peternak tambak yang membutuhkan sistem dengan nilai *availability* tinggi sehingga dapat diandalkan untuk mendapatkan hasil panen berkualitas baik.

4.1.4 Lingkungan operasi sistem

Lingkungan operasi sistem merupakan lingkungan yang mendukung kebutuhan sistem dimana terdapat tambak dengan kondisi air yang memiliki tingkat suhu, kadar garam dan kekeruhan yang mudah berubah-ubah.

4.1.5 Asumsi dan ketergantungan

Beberapa asumsi dan ketergantungan persyaratan sistem sebagai berikut:

1. Sensor dapat membaca data sesuai dengan fungsi sensor tersebut dan sistem dapat meredundansi sewaktu-waktu terjadi *fault* pada sistem utama.
2. Untuk dapat menyala diperlukan sumber daya yang cukup stabil antara 3V – 5V jika dibawah 3V maka *microcontroller* bisa mati dan jika diatas 5V maka komponen yang ada bisa rusak karena kelebihan muatan.

4.2 Kebutuhan Fungsional

Pada subbab ini akan dijelaskan kebutuhan fungsional dari sistem, kebutuhan fungsional dari sistem yang akan dibangun merupakan penjelasan dari fitur atau fungsi yang harus ada dan dapat diperoleh dengan sistem tersebut. Pada kebutuhan sistem meliputi aspek *input* dan *output* sistem, serta fungsi respon sistem terhadap i*nput* dan *output* dalam proses berjalannya sistem. Kebutuhan fungsional dari sistem adalah sebagai berikut:

1. Sistem *secondary* merupakan suatu sistem yang memiliki komponen sama persis dengan sistem primer hanya saja sistem *Secondary* tidak menggunakan IC 7408. Sistem *secondary* berfungsi untuk meredundansi sistem primer apabila terdapat *fault* pada sistem primer baik dari sisi sensor yang mengalami gangguan, modul yang *fail* atau bahkan gangguan power sistem primer.
2. Sistem primer merupakan sistem utama yang digunakan dalam sistem akuisisi data sensor. Dalam keadaan *fail* seperti power mengalami gangguan dan terdapat sensor mati, output dari sistem primer dapat digunakan sebagai *interrupt* sistem sekunder yang berada dalam keadaan *standby* sehingga sistem sekunder dapat berjalan dan mengabil alih pekerjaan sistem primer.

4.3 Kebutuhan perangkat keras

Kebutuhan perangkat keras merupakan kebutuhan yang harus dipenuhi yang berwujud perangkat keras. Pada penelitian ini dibutuhkan beberapa perangkat keras, kebutuhan perangkat keras yang digunakan adalah sebagai berikut:

1. *Microcontroller* Arduino UNO

Mikrocontroller merupakan sebuah perangkat keras atau otak dari suatu sistem yang berfungsi untuk mengeksekusi dan menjalankan program yang telah dibuat. Program yang dijalankan akan diwujudkan berupa keluaran atau aksi sesuai dengan program yang dibuat.

2. Sensor DS18B20

Sensor DS18B20 merupakan perangkat keras sensor yang berfungsi untuk menangkap data suhu pada air. Sensor ini dapat bekerja langsung dalam air karena sudah dilengkapi dengan pembungkus *stainless* kedap air.

3. Sensor *Salinity*

Sensor ini merupakan sensor yang digunakan untuk mendapatkan data kadar garam dari air. Sensor ini juga dapat bekerja untuk mengetahui kemurnian air dan kontaminasi air.

4. Sensor *Turbidity*

Sensor ini merupakan sensor yang digunakan untuk mengetahui seberapa keruh kondisi air. Sensor akan diletakan bersamaan dengan sensor suhu dan sensor kadar garam pada air yang akan diteliti.

5. PCB Matrix

PCB berguna untuk menjadi wadah berkumpulnya dari beberapa komponen yang digunakan oleh sistem. PCB yang digunakan dalam penelitian ini adalah PCB Matrix yang biasa digunakan untuk latihan dalam membuat berbagai macam elektronika.

6. Laptop

Sebagai pemberi kontrol pemrograman pada Arduino UNO melalui *software* IDE dan monitoring penulis pada serial monitor serta sebagai sumber power bagi kedua modul.

7. IC 7408

IC 7408 merupakan IC yang berfungsi sebagai gerbang logika yang akan digunakan sebagai *input interrupt* dari modul *master*.

4.4 Kebutuhan perangat lunak

Pada sub bab ini akan dijelaskan kebutuhan perangkat lunak yang dibutuhkan oleh sistem mencakup seluruh perangkat lunak yang diperlukan untuk mengimplementasikan sistem yang akan dibuat. Kebutuhan perangkat lunak merupakan kebutuhan yang harus dipenuhi dalam bentuk *software*. Kebutuhan tersebut adalah sebagai berikut:

1. ***Integrated development environment (IDE)***

Perangkat lunak *integrated development environment (IDE)* adalah sebuah perankat lunak yang di gunakan sebagai pengembangan atau memprogram di dalam lingkup Arduino. Arduino itu sendiri menggunakan bahasa C sehingga banyak yg menggunakan *software* tersebut yang di klaim mudah di gunakan.

2. ***Library avr sleep***

Library yang digunakan untuk menerapkan fungsi yang akan digunakan untuk membuat program *sleep* dari *microcontroller.*

3. ***Library avr power***

Library ini digunakan untuk memuat set fungsi yang akan digunakan membuat program mengaktifkan dan menonaktifkan fitur dari *microcontroller* seperti *ADC*, *timer*, dan lainnya.

4. ***Library* DallasTemperature dan OneWire**

Library ini digunakan untuk memuat set fungsi yang digunakan untuk membuat program yang dapat membaca nilai suhu sehingga dapat dimonitoring melalui serial monitor pada *software* Arduino IDE.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

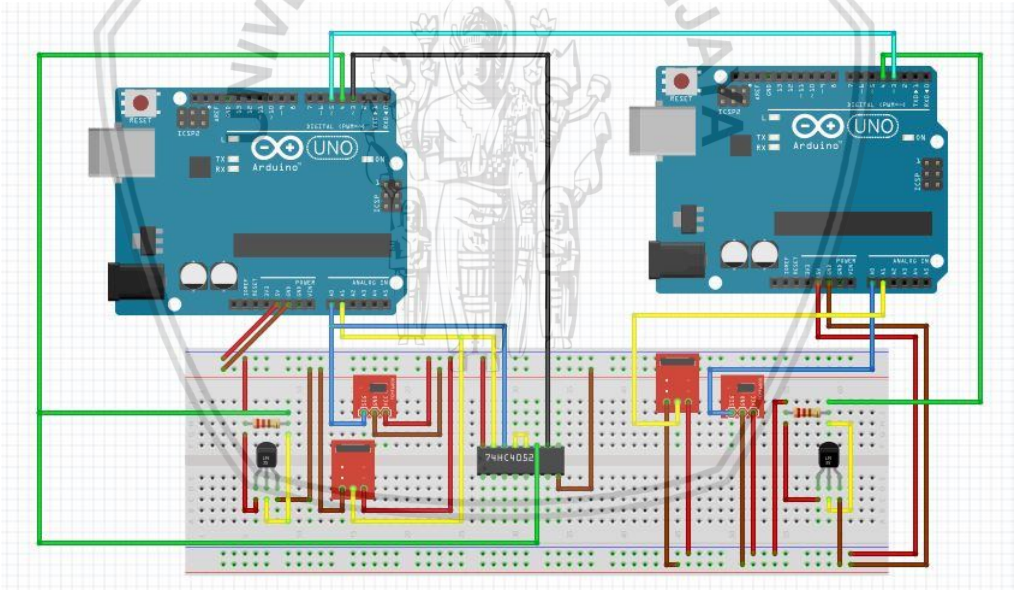
5.1 Perancangan

Pada bab ini akan dibahas perancangan dari Implementasi *Hardware Redundancy* Sistem Akuisisi Data Sensor Dengan Menggunakan Metode *Hot Standby Sparing*. Pada perancangan ini meliputi perancangan sistem secara keseluruhan, perancangan sistem utama yang digunakan sebagai *master* modul, sistem cadangan yang digunakan sebagai *slave* modul atau modul pengganti, perancangan komunikasi antara sensor-sensor dengan modul *master* serta perancangan komunikasi antara modul *master* dan modul *slave*.

5.1.1 Perancangan Sistem

5.1.1.1 Perancangan Perangkat Keras

Perancangan perangkat keras sistem secara keseluruhan merupakan suatu perancangan yang menjelaskan bagaimana sistem dirancang secara keseluruhan meliputi sistem utama, sistem slave serta komunikasinya secara umum. Berikut merupakan perancangan perangkat keras sistem seperti pada Gambar 5.1.

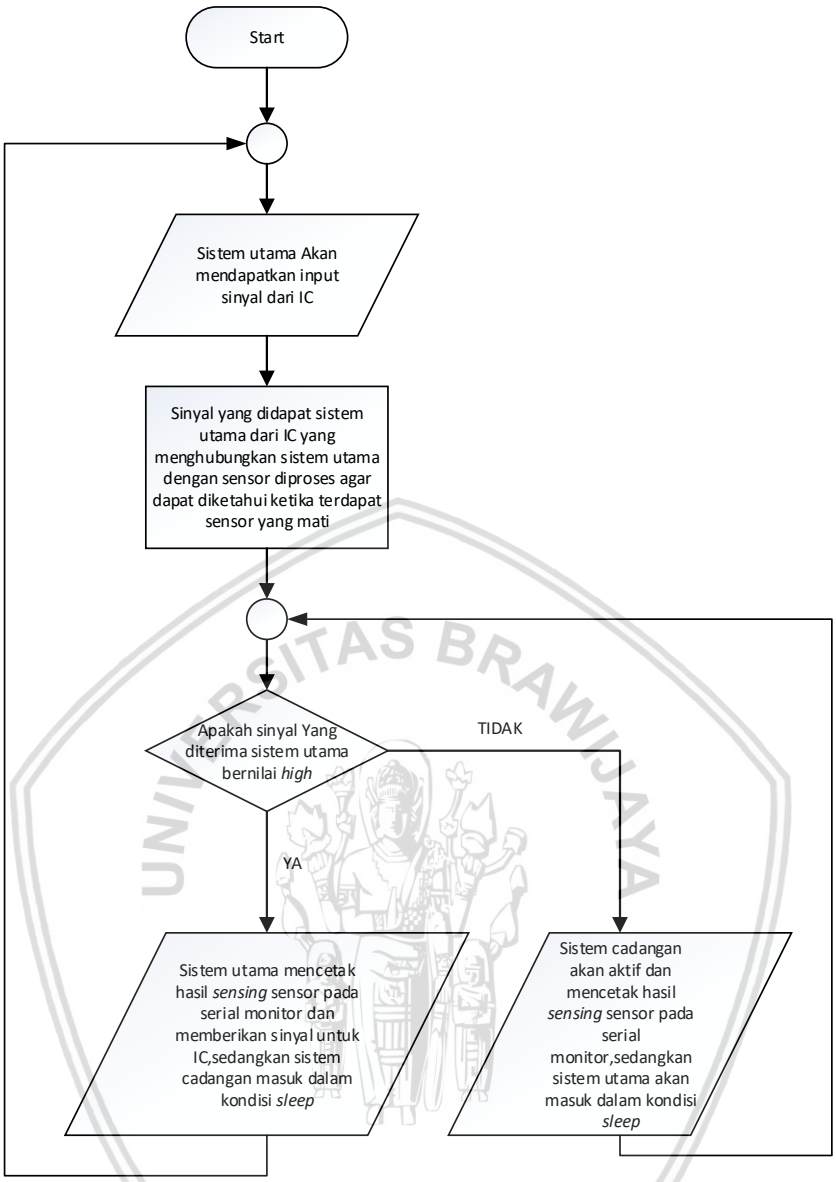


Gambar 5.1 Perancangan Sistem

Pada Gambar 5.1 dapat dilihat bahwa sistem dirancang untuk dapat saling terhubung, baik antara sensor dengan modul maupun modul dengan modul yang lainnya. Hubungan pin-pin untuk setiap komponen yang ada pada perancangan sistem dapat dilihat pada Tabel 5.1 dan Tabel 5.2.

5.1.1.2 Perancangan Perangkat Lunak

Agar lebih jelas bagaimana komunikasi perangkat lunak yang dirancang pada sistem, penulis memberikan gambar diagram blok perangkat lunak seperti yang dapat dilihat pada Gambar 5.2.



Gambar 5.2 Diagram Blok Perancangan Sistem

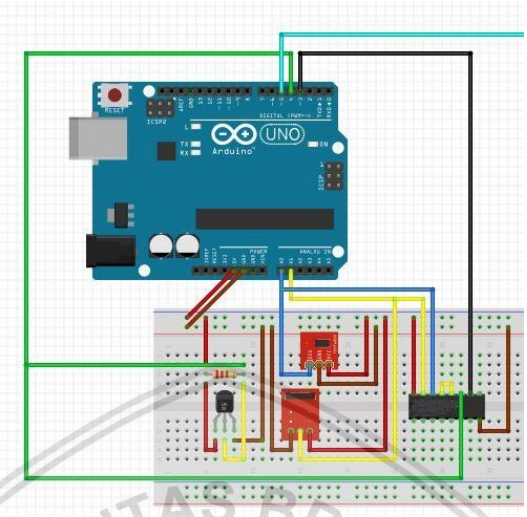
5.1.2 Perancangan Modul *Master*

5.1.2.1 Perancangan Perangkat Keras

Modul *Master* merupakan modul utama pada sistem.Pada tahap perancangan perangkat keras modul *master*, beberapa komponen utama yang digunakan ialah :

1. Arduino UNO R3
2. Sensor Suhu DS18B20
3. Sensor Kekeruhan Air
4. Sensor Kadar Garam
5. IC 7408

Berikut perancangan perangkat keras dari modul *master* yang menggambarkan hubungan antara komponen-komponen yang ada pada modul *master* seperti pada Gambar 5.3.



Gambar 5.3 Perancangan Modul *Master*

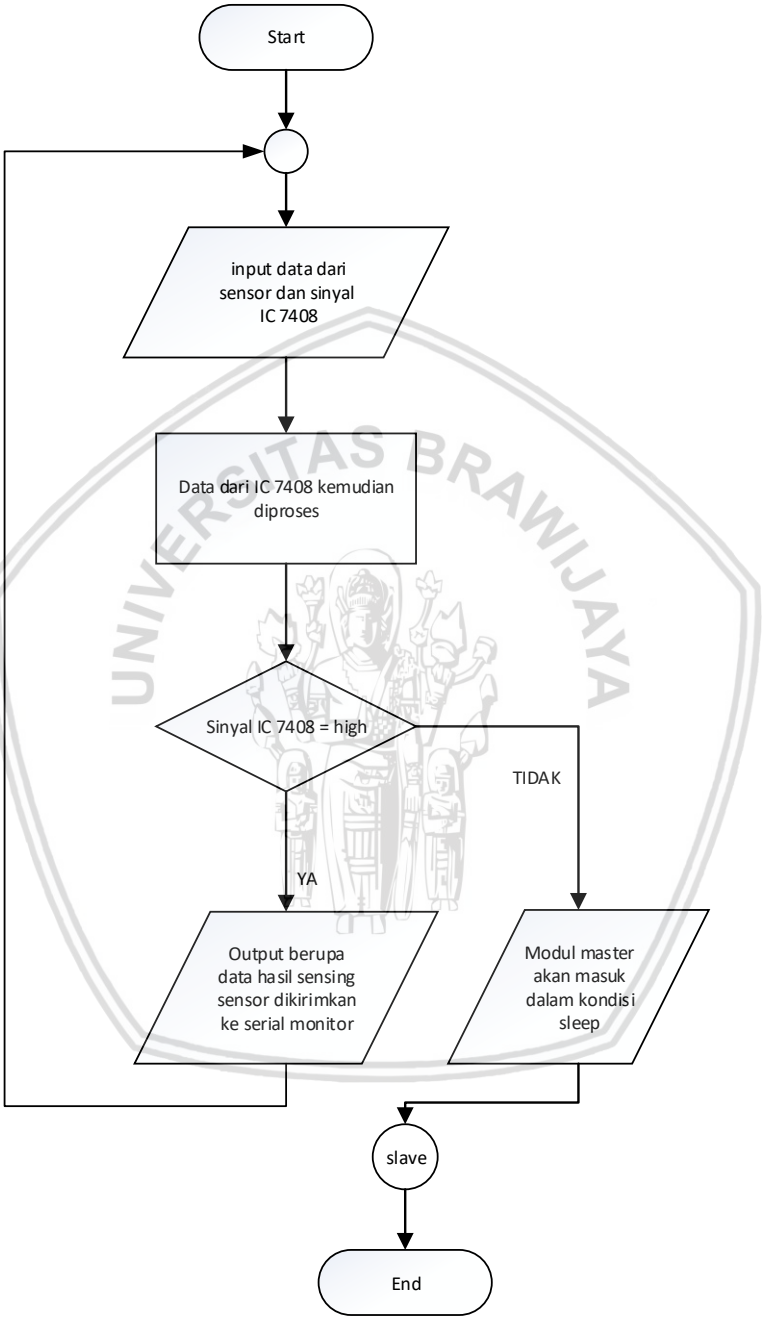
Gambar 5.3 merupakan rancangan dari modul yang akan digunakan sebagai *master*. Dapat dilihat dari Gambar 5.3, bahwa *output* dari sensor-sensor yang ada nantinya akan diterima oleh pin sensor serta *pin interrupt*. Pada pin sensor, data dari akuisisi sensor akan ditampilkan menggunakan serial monitor yang merupakan fasilitas dari aplikasi Arduino IDE, sedangkan pada *pin interrupt*, data sensor sebelumnya akan masuk atau dikirimkan ke IC 7408 yang merupakan IC gerbang logika AND untuk menghasilkan output 1 apabila seluruh sensor aktif dan 0 apabila ada salah satu atau lebih sensor tidak aktif. Sehingga *pin interrupt* hanya akan menerima nilai anatar 0 dan 1, apabila nilai input 0 maka modul akan masuk ke mode *sleep* dan apabila nilai input 1 maka modul akan aktif. Sedangkan *pin output* digunakan sebagai *input* untuk modul *slave* atau cadangan, apabila modul *master* sleep maka *pin output* akan berada pada kondisi *low* sehingga modul cadangan akan aktif dan sebaliknya. Agar lebih jelas tentang penempatan sensor dan berbagai komponen pada Arduino UNO yang digunakan sebagai modul *master*, maka penulis membuat sebuah Tabel 5.1 berikut:

Tabel 5.1 Table Koneksi Pin Arduino *Master* dengan Komponen dan Sensor

Pin Arduino UNO	Komponen & Sensor
Pin 3	*Interrupt* IC7408
Pin 4	Sensor Suhu
Pin 5	*Output* sebagai *Interrupt* Modul *Slave*
Pin A0	Sensor Kekeruhan Air
Pin A1	Sensor Kadar Garam

5.1.2.2 Perancangan Perangkat Lunak

Agar lebih jelas bagaimana komunikasi perangkat lunak yang dirancang pada modul *master*, penulis memberikan gambar *flowchart* perangkat lunak seperti yang dapat dilihat pada Gambar 5.4.



Gambar 5.4 *Flowchart* Modul *Master*

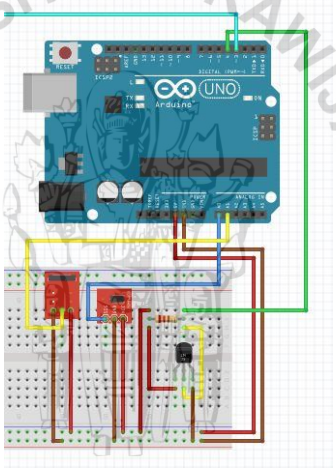
5.1.3 Perancangan Modul *Slave*

5.1.3.1 Perancangan Perangkat Keras

Modul *Slave* merupakan modul yang digunakan sebagai cadangan modul apabila terjadi *fault* pada modul *master*. Pada tahap perancangan perangkat keras modul *slave*, beberapa komponen yang digunakan ialah :

1. Arduino UNO R3
2. Sensor Suhu DS18B20
3. Sensor Kekeruhan Air
4. Sensor Kadar Garam

Berikut perancangan dari modul *master* yang menggambarkan hubungan antara komponen-komponen yang ada pada modul *master* seperti pada Gambar 5.5.



Gambar 5.5 Perancangan Modul *Slave*

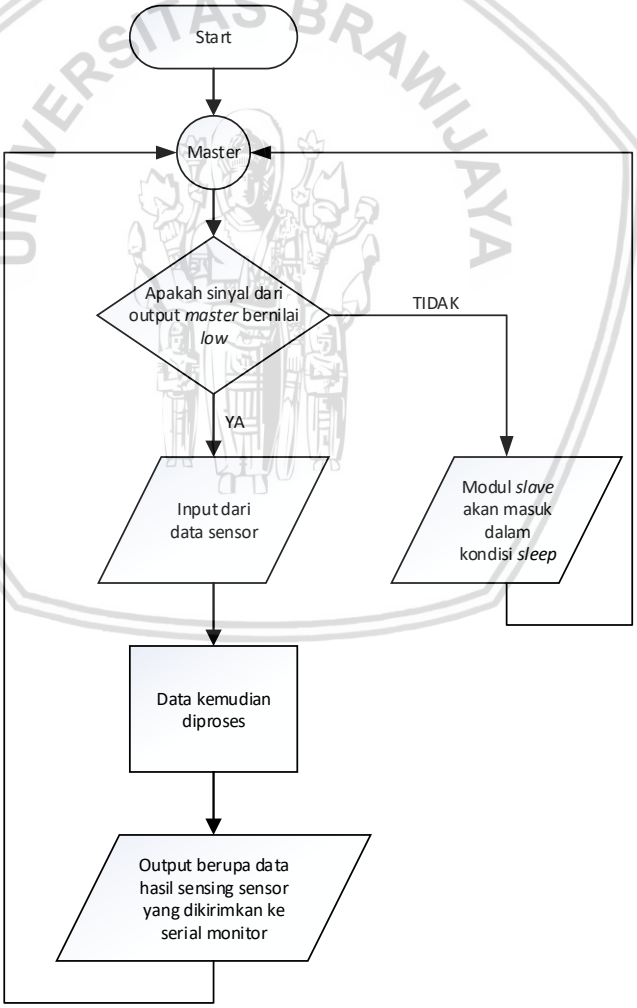
Pada Gambar 5.5, dapat dilihat bahwa perbedaan antara modul *master* dengan modul *slave* ialah pada *input pin interrupt*, pada modul *slave input pin interrupt* ialah dari modul *master*, apabila modul *master* dalam keadaan *sleep* maka modul *slave* akan *wake* dari mode *sleep* dan akan mengeksekusi program untuk mengakuisisi data dari sensor-sensor. Pada modul *slave*, permasalahan dibatasi pada kondisi modul akan tetap aktif walaupun terdapat *fault* dari salah satu atau beberapa sensor. Ini dikarenakan sebagai konsekuensi dari penentuan diantara kedua modul yang mana akan digunakan sebagai *master* dan *slave*, sehingga untuk menghindari adanya kondisi *race* ketika sistem pertama kali dijalankan atau dihidupkan, maka modul *slave* tidak menerima *interrupt* dari sensor-sensor melainkan dari modul *master*. Untuk memperjelas bagaimana penempatan sensor dan berbagai komponen pada Arduino UNO yang digunakan sebagai modul *slave*, maka penulis menjabarkan penempatannya dengan menggunakan Tabel 5.2 berikut :

Tabel 5.2 Table Koneksi Pin Arduino *Slave* dengan Komponen dan Sensor

Pin Arduino UNO	Komponen & Sensor
Pin 3	*Interrupt* dari Output Pin 5 Modul *Master*
Pin 4	Sensor Suhu
Pin A0	Sensor Kekeruhan Air
Pin A1	Sensor Kadar Garam

5.1.3.2 Perancangan Perangkat Lunak

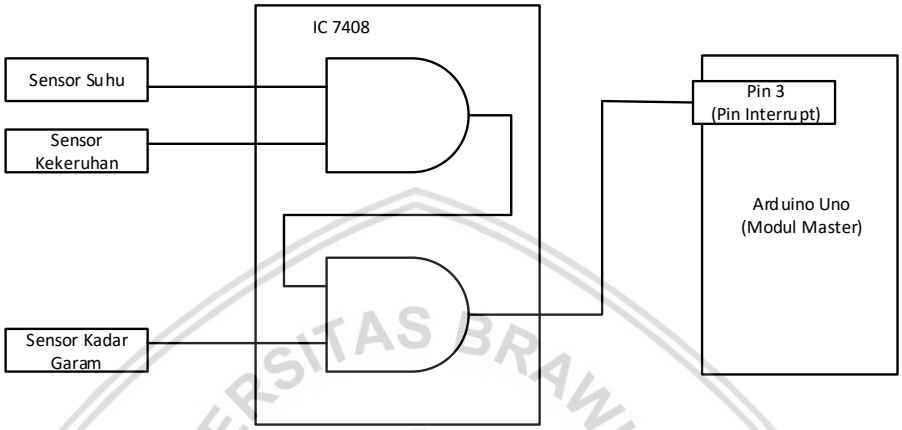
Agar perancangan lebih jelas dan terperinci, maka penulis membuat perancangan perangkat lunak modul *slave* dengan menggunakan *flowchart* sebagai berikut :



Gambar 5.6 *Flowchart* Modul *Slave*

5.1.4 Perancangan *Interrupt* pada Modul *Master*

Komunikasi antara sensor dan modul menggunakan media IC 7408 sebagai gerbang komunikasinya. IC 7408 merupakan IC yang digunakan sebagai gerbang logika *AND*, apabila ada salah satu saja *input* yang bernilai *low* maka IC akan mengirimkan *output low* ke modul *master* dan akan mengirimkan *output high* jika dan hanya jika semua sensor bernilai *high*. Berikut perancangan komunikasi antara sensor dan modul *master* seperti pada Gambar 5.7.



Gambar 5.7 Diagram Blok Interrupt pada Modul *Master*

Seperti sudah dijelaskan diagram blok yang ada pada Gambar 5.1, perancangan *Interrupt* pada modul *master* dapat dilihat pada Gambar 5.3. Pada Gambar 5.3 dijelaskan bagaimana komunikasi antara sensor dengan modul *master* dimana sebelum *output* dari sensor masuk ke modul *master*, telebih dahulu *output* masuk ke IC 7408 untuk deteksi apabila salah satu atau beberapa sensor ada yang mengalami *fault*. Tujuan penulis menggunakan IC 7408 ialah karena *output* yang akan dihasilkan IC hanya akan bernilai *high* apabila semua sensor diasumsikan aktif dan tidak terjadi *fault*. Berikut tabel kebenaran dari gerbang IC pada Gambar 5.3:

Tabel 5.3 Tabel Kebenaran Output IC 7408

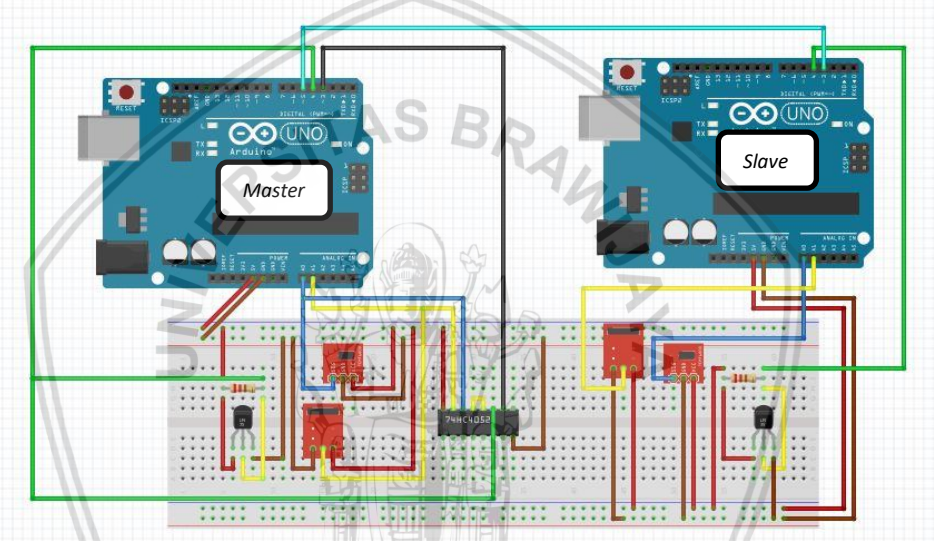
Sensor Suhu	Sensor Kekeruhan	Sensor Kadar Garam	Output
0	0	0	0
1	0	0	0
0	1	0	0
1	1	0	0
0	0	1	0
1	0	1	0
0	1	1	0
1	1	1	1

Dapat dilihat dari Tabel 5.3 bahwa output yang akan diberikan IC 7408 hanya akan bernilai *high* apabila seluruh sensor bernilai *high*, salah satu saja terdapat sensor yang *fault* maka output dari IC 7408 akan bernilai *low*.

5.1.5 Perancangan Redundansi Sistem

5.1.5.1 Perancangan Perangkat Keras

Pada tahap perancangan perangkat keras redundansi sistem ini bertujuan agar modul *slave* dapat meredundansi modul *master* apabila sewaktu-waktu terjadi *fault* yaitu sensor yang mati pada modul *master*, baik *fault* yang disebabkan karena ada salah satu atau beberapa sensor yang mati maupun terjadi *fault* seperti modul *master* tiba-tiba mati karena kendala power. Berikut perancangan dari komunikasi antara modul *master* dan *slave* :

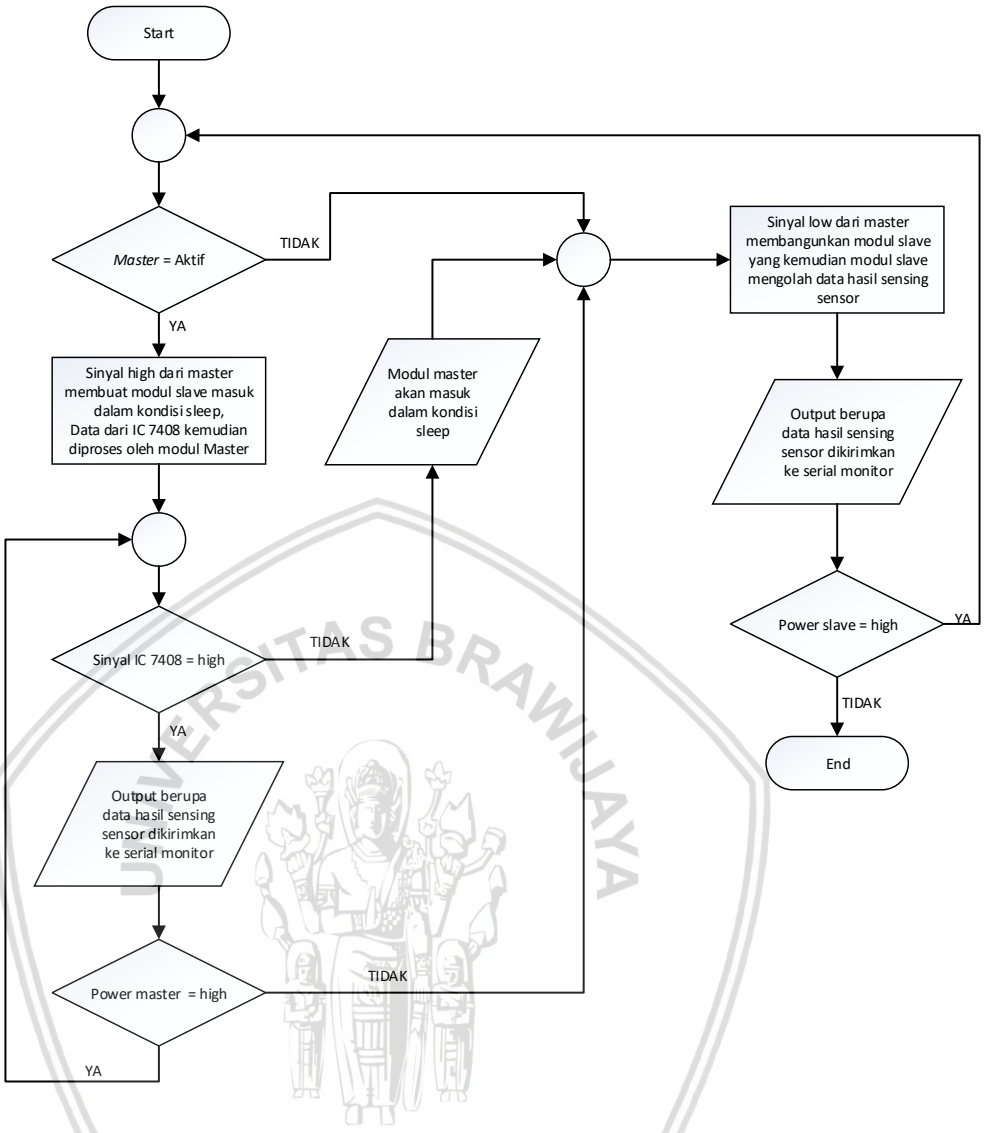


Gambar 5.8 Perancangan Komunikasi Antara Modul *Master* dan *Slave*

Gambar 5.8 diatas merupakan rangkaian komunikasi apabila modul *master* dan modul *slave* saling terhubung. Berdasarkan Gambar 5.8 dapat dilihat bahwa yang membedakan antara modul *master* dan modul *slave* ialah *interrupt* yang diterima masing-masing modul. Pada modul *master*, *pin interrupt* yaitu pin 3 menerima *interrupt* dari sensor sensor yang mengirimkan data yang sebelumnya sudah masuk kedalam IC 7408 terlebih dahulu seperti pada Gambar 5.7. Sedangkan pada modul *slave*, *pin interrupt* yaitu pin 3 menerima *interrupt* dari pin 5 yang ada pada modul *master*, apabila modul *master sleep* maka pin 5 akan bernilai *low* dan *pin interrupt* akan menerima nilai *low* yang akan menghidupkan atau *wake* modul *slave* dan sebaliknya ketika modul *master wake*, maka pin 5 akan bernilai *high* yang artinya modul *slave* akan menerima nilai *high* pada *pin interrupt* yang membuat modul *slave* masuk dalam kondisi *sleep*.

5.1.5.2 Perancangan Perangkat Lunak

Agar perancangan lebih jelas dan terperinci, maka penulis membuat perancangan perangkat lunak redundansi sistem dengan menggunakan *flowchart* sebagai berikut :



Gambar 5.9 *Flowchart* Redundansi Sistem

5.2 Implementasi

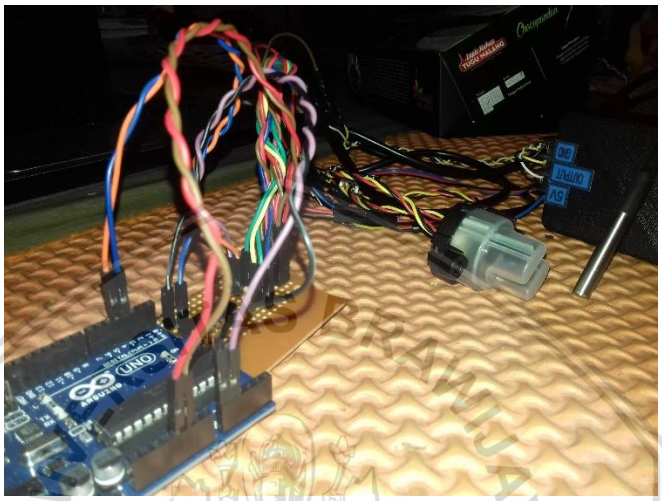
Pada implementasi sistem dilakukan saat semua proses pada tahap perancangan sistem telah selesai atau telah dipenuhi. Pembahasan tentang implementasi pada sub bab ini akan membahas mengenai implementasi *hardware* redundansi pada sistem akuisisi data sensor dengan menggunakan metode *hot standby sparing* baik dari implementasi modul *master* dan *slave*, implementasi *interrupt* pada modul *master* serta implementasi komunikasi antara modul *master* dan modul *slave*.

5.2.1 Implementasi Modul *Master*

5.2.1.1 Implementasi Perangkat Keras

Pada implementasi modul *master*,sensor akan dihubungkan ke IC 7408 dan ke pin input pada Arduino. Tujuan dari dihubungkannya data sensor ke IC 7408

ialah untuk deteksi kesalahan yang terjadi pada sensor, apabila ada salah satu atau beberapa sensor mengalami *fault* maka output yang dikeluarkan IC 7408 ialah bernilai *low*, nilai ini menjadi input dari pin *interrupt* dimana ketika *input* yang diterima bernilai *low* maka *interrupt* akan membuat Arduino masuk dalam keadaan *sleep* dan sebaliknya. Sedangkan tujuan dari dihubungkannya data sensor ke pin *input* data sensor ialah agar hasil data *input* sensor dapat diamati melalui serial monitor yang ada pada fitur aplikasi Arduino IDE. Hasil implementasi modul *master* dapat dilihat pada Gambar 5.10.



Gambar 5.10 Implementasi Modul *Master*

5.2.1.2 Implementasi Perangkat Lunak

Untuk implementasi perangkat lunak berupa kode pemrograman yang diupload pada modul *master*, penulis menjabarkan kode pemrograman secara lengkap pada Lampiran A laporan skripsi ini. Berikut penggalan program pada aplikasi Arduino IDE yang menjelaskan tentang fungsi-fungsi pin pada Arduino beserta proses pengolahan data hasil *sensing* sensor yang digunakan pada modul *master* seperti pada Gambar 5.11. dan Gambar 5.12.

```
void setup() {
  Serial.begin(9600);
  pinMode(4,INPUT);
  digitalWrite(4,HIGH);
  pinMode(A0,INPUT);
  analogWrite(A0,HIGH);
  pinMode(A1,INPUT);
  analogWrite(A1,HIGH);
  pinMode(3,INPUT);
  digitalWrite(3,HIGH);
  pinMode(5,OUTPUT);
  digitalWrite(5,HIGH);
}
```

Gambar 5.11 Potongan Program Fungsi Pin Modul *Master*

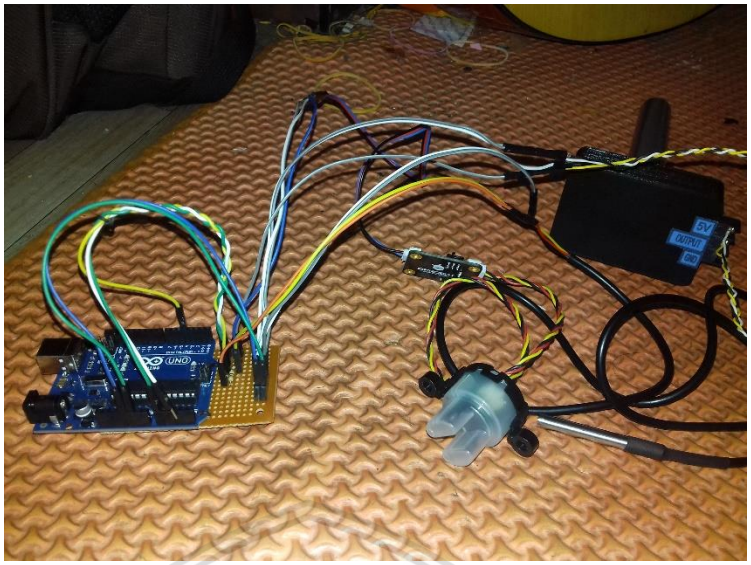
```
void loop() {
  sleepStatus = digitalRead(pinPin);
  if(sleepStatus == HIGH){
  suhuSekarang =ambilSuhu();
  int sensorValue = analogRead(A0);
  int sensorValue2 = analogRead(A1);
  float voltage2 = sensorValue2*(5.0/1024.0);
  float voltage = sensorValue*(5.0/1024.0);
  float kekeruhan = 100.00-(voltage/4.16)*100.00;
  Serial.println("=======================================");
  Serial.println("SUHU");
  Serial.print(suhuSekarang);
  Serial.println(" C");
  Serial.println("KEKERUHAN");
  Serial.print(voltage);
  Serial.print(" Volt");
  Serial.print("    ");
  Serial.print("Nilai kekeruhan =");
  Serial.print(kekeruhan);
  Serial.println("  NTU");
  Serial.println("KADAR GARAM");
  Serial.print(voltage2);
  Serial.println(" Volt");
  delay(1000);
  }if(sleepStatus == LOW){
  sleepSetup();
  }
```

Gambar 5.12 Potongan Program Pengolahan *Sensing* Sensor *Master*

5.2.2 Implementasi Modul *Slave*

5.2.2.1 Implementasi Perangkat Keras

Pada implementasi modul *slave*, sensor tidak dihubungkan ke IC 7408, data dari sensor langsung masuk ke pin input data sensor. Tujuannya tidak digunakan IC 7408 pada modul *slave* ialah agar input pin interrupt yang diterima modul *slave* nantinya terfokus hanya dari output modul *master*. Dalam kondisi seperti ini masalah dibatasi yaitu seluruh sensor pada modul *slave* dianggap aktif dan tak mengalami kesalahan sedikitpun, sehingga ketika melakukan pengujian sistem dapat diamati bagaimana sistem *slave* dapat meredundansi sistem *master*. Berikut hasil dari implementasi modul *slave* dapat dilihat pada Gambar 5.13.



Gambar 5.13 Implementasi Modul *Slave*

5.2.2.2 Implementasi Perangkat Lunak

Untuk implementasi perangkat lunak berupa kode pemrograman yang diupload pada modul *slave*, penulis menjabarkan kode pemrograman secara lengkap pada Lampiran B laporan skripsi ini. Berikut penggalan program pada aplikasi Arduino IDE yang menjelaskan tentang fungsi-fungsi pin pada Arduino UNO serta proses pengolahan data hasil *sensing* sensor yang digunakan pada modul *slave* seperti pada Gambar 5.14 dan 5.15.

```
void setup() {
  Serial.begin(9600);
  pinMode(4,INPUT);
  digitalWrite(4,HIGH);
  pinMode(A0,INPUT);
  analogWrite(A0,HIGH);
  pinMode(A1,INPUT);
  analogWrite(A1,HIGH);
  pinMode(3,INPUT);
  digitalWrite(3,LOW);
}
```

Gambar 5.14 Potongan Program Fungsi Pin Modul *Slave*

```
void loop() {
  sleepStatus = digitalRead(pinPin);
  if(sleepStatus == LOW){
  suhuSekarang =ambilSuhu();
  int sensorValue = analogRead(A0);
  int sensorValue2 = analogRead(A1);
  float voltage2 = sensorValue2*(5.0/1024.0);
  float voltage = sensorValue*(5.0/1024.0);
  float kekeruhan = 100.00-(voltage/4.16)*100.00;
  Serial.println("=======================================");
  Serial.println("SUHU");
  Serial.print(suhuSekarang);
  Serial.println(" C");
  Serial.println("KEKERUHAN");
  Serial.print(voltage);
  Serial.print(" Volt");
  Serial.print("    ");
  Serial.print("Nilai kekeruhan =");
  Serial.print(kekeruhan);
  Serial.println("  NTU");
  Serial.println("KADAR GARAM");
  Serial.print(voltage2);
  Serial.println(" Volt");
  delay(1000);
  }if(sleepStatus == HIGH){
  sleepSetup();
  }
```

Gambar 5.15 Potongan Program Pengolahan *Sensing* Sensor *Slave*

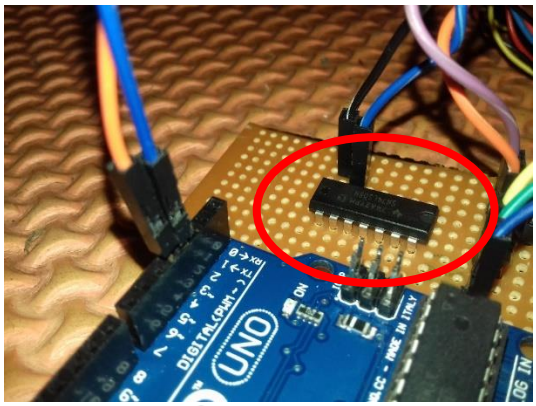
Dapat dilihat perbedaan pada Gambar 5.11 dengan Gambar 5.14,yaitu pada modul *slave* tidak menggunakan pin 5 karena pada modul *master interrupt* diterima dari input pin 3 yang asalnya dari koreksi kesalahan IC 7408 dan pada pin 5 modul *master* akan menghasilkan output sebagai *interrupt* pin 3 modul *slave*.

5.2.3 Implementasi *Interrupt* pada Modul *Master*

5.2.3.1 Implementasi Perangkat Keras

Pada implementasi perangkat keras komunikasi antara sensor dengan modul *master*, seperti sudah dijelaskan sebelumnya pada pembahasan tentang implementasi modul *master* bahwa output dari sensor akan masuk langsung ke *pin input* sensor dan juga masuk ke IC 7408. IC7408 merupakan sebuah IC gerbang logika *and*, digunakan gerbang logika *and* karena logika ini hanya akan menghasilkan *output high* jika dan hanya jika semua *input* dari *output* sensor yang masuk ke IC 7408 bernilai *high*, salah satu saja input bernilai *low*, maka output yang dikeluarkan IC 7408 akan bernilai *low* yang berarti akan mengaktifkan *pin interrupt* dan membuat modul *master* masuk dalam kondisi *sleep*. Sedangkan data sensor yang masuk langsung ke pin *input* sensor akan ditampilkan pada serial

monitor yang ada di aplikasi Arduino IDE. Berikut hasil dari implementasi perangkat keras *interrupt* pada modul *master* seperti pada Gambar 5.16.



Gambar 5.16 Implementasi *Interrupt* Modul *Master*

Pada Gambar 5.16 dapat dilihat pada lingkaran merah merupakan IC 7408 yang digunakan untuk seleksi apabila ada kesalahan dari salah satu atau beberapa sensor maka output dari ic tersebut akan bernilai *low* dan hanya akan bernilai *high* jika dan hanya jika seluruh sensor dalam keadaan baik, dalam hal ini tidak terjadi kesalahan sedikitpun. *Output* dari IC 7408 kemudian menjadi input pin *interrupt* yaitu pin 3, dimana ketika input *low* maka modul *master* akan masuk dalam kondisi *sleep* dan sebaliknya.

5.2.3.2 Implementasi Perangkat Lunak

Untuk implementasi perangkat lunak berupa kode pemrograman yang diupload pada modul *master* sebagai *interrupt*, penulis menjabarkan kode pemrograman secara lengkap pada Lampiran A laporan skripsi ini. Berikut potongan program interupsi pada modul *master* seperti dapat dilihat pada Gambar 5.17.

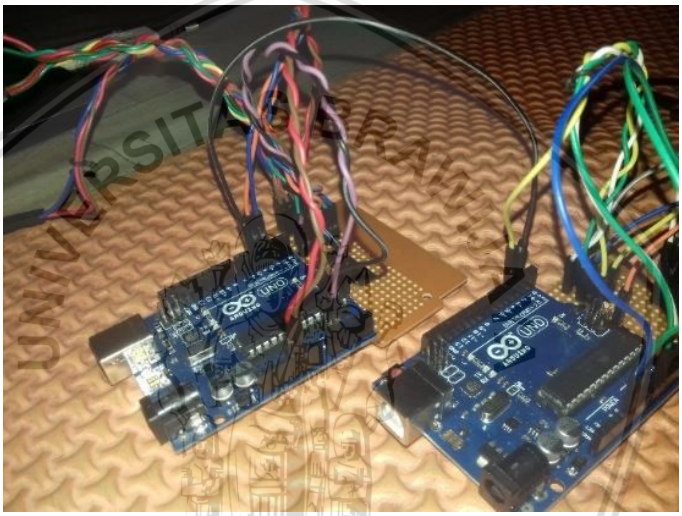
```
void sleepSetup(){
  sleep_enable();
  attachInterrupt(0,pinInterrupt,LOW);
  set_sleep_mode(SLEEP_MODE_PWR_DOWN);
  digitalWrite(5,LOW);
  sleep_cpu();
  Serial.println("terbangun");
  digitalWrite(5,HIGH);
}

void pinInterrupt(){
  sleep_disable();
  detachInterrupt(0);
}
```

Gambar 5.17 Potongan Program Interupsi Modul *Master*

5.2.4 Implementasi Redundansi Sistem

5.2.4.1 Implementasi Perangkat Keras

Pada implementasi perangkat keras redundansi sistem, implementasi redundansi ini bertujuan untuk mengiplementasikan bagaimana *hardware* redundansi dapat bekerja pada sistem akuisisi data sensor khususnya pada metode *hot standby spare*. Terdapat *pin output* yaitu pin 5 pada modul *master* yang digunakan sebagai *input* dari *pin interrupt* yang ada pada modul *slave*, ketika *pin output* modul *master* bernilai *low* karena terdapat *fault* pada salah satu atau beberapa sensor pada modul *master*, maka *pin interrupt* yang ada pada modul *slave* akan aktif dan membangunkan modul *slave* yang sebelumnya ada dalam kondisi *sleep*, begitupun sebaliknya. Berikut implementasi dari komunikasi antara modul *master* dan *slave* dapat dilihat pada Gambar 5.18.



Gambar 5.18 Implementasi Redundansi Sistem

Pada Gambar 5.18 dapat dilihat kabel berwarna hitam menghubungkan modul *master* dan *slave*, kabel tersebut berguna sebagai media untuk menyampaikan *output* pin 5 yang ada pada modul *master* sebagai input *intrrupt* pin 3 yang ada pada modul *slave*, apabila *interrupt* bernilai *low* maka modul *slave* akan aktif atau terbangun dari kondisi *sleep* dan sebaliknya.

5.2.4.2 Implementasi Perangkat Lunak

Untuk implementasi perangkat lunak berupa kode pemrograman yang diupload pada modul *slave* sebagai *interrupt* agar modul *slave* mampu meredundansi modul *master*, penulis menjabarkan kode pemrograman secara lengkap pada Lampiran B laporan skripsi ini. Berikut potongan program interupsi pada modul *slave* seperti dapat dilihat pada Gambar 5.19.

```
void sleepSetup(){
  sleep_enable();
  attachInterrupt(0,pinInterrupt,LOW);
  set_sleep_mode(SLEEP_MODE_PWR_DOWN);
  sleep_cpu();
  Serial.println("terbangun");
}

void pinInterrupt(){
  sleep_disable();
  detachInterrupt(0);
}
```

Gambar 5.19 Potongan Program Interupsi Modul *Slave*

BAB 6 PENGUJIAN DAN ANALISIS

Pengujian merupakan bab yang berisi hasil dari pengujian sistem berdasarkan dari implementasi sistem untuk melihat fungsional dari sistem. Dalam analisis sistem dapat dilihat apakah sistem sudah dapat berjalan sesuai dengan kebutuhan fungsional sistem sehingga sistem dapat dikatakan berjalan dengan baik atau tidak. Pengujian dan analisis akan dibagi menjadi dua sub bab, yaitu pengujian menghindari kondisi *race* dan pengujian membuat kesalahan tiap sensor, pada pengujian membuat kesalahan tiap sensor akan dilakukan tiga pengujian karena ada tiga sensor yang digunakan yaitu sensor suhu, kekeruhan dan kadar garam.

6.1 Pengujian Fungsi Modul *Master*

Pada pengujian fungsi modul *master* akan dijelaskan apa tujuan dari pengujian ini, bagaimana prosedur pengujiannya dan hasil analisis dari pengujian.

6.1.1 Tujuan Pengujian

Tujuan dari pengujian fungsi modul *master* adalah untuk memastikan bahwa modul *master* dapat berfungsi sesuai dengan tujuan dibuatnya modul *master* yaitu dapat memonitoring kondisi air.

6.1.2 Prosedur Pengujian

Dalam pengujian fungsi modul *master*, prosedur pengujiannya adalah sebagai berikut:

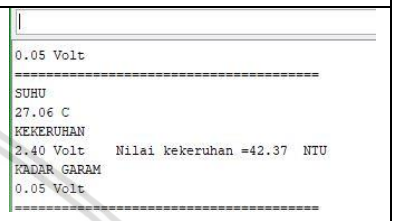
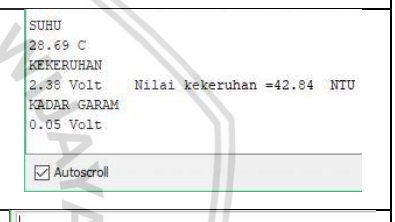
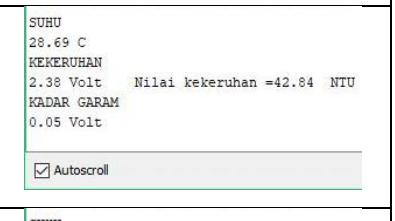
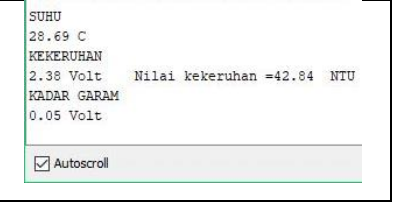
1. Menghubungkan modul *master* dengan Laptop.
2. Meng-*upload* program yang sudah dibuat pada aplikasi Arduino IDE.
3. Setelah selesai *upload*, buka serial monitor untuk mengamati hasil dari *sensing* sensor-sensor yang ada pada modul *master*.

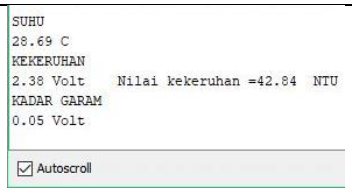
6.1.3 Hasil dan Analisis Pengujian

Dalam hasil dan analisis pengujian akan dijelaskan hasil dari pengujian serta memberikan analisis dari hasil pengujian fungsi modul *master*, hasil pengujian dapat dilihat pada Tabel 6.1.

Tabel 6.1 Pengujian Fungsi Modul *Master*

Nama Pengujian	Hasil Pengujian	Screenshoot Pengujian
Pengujian ke 1	Berhasil	0.05 Volt ========== SUHU 27.06 C KEKERUHAN 2.40 Volt Nilai kekeruhan =42.37 NTU KADAR GARAM 0.05 Volt ==========

Pengujian ke 2	Berhasi	
Pengujian ke 3	Berhasil	
Pengujian ke 4	Berhasil	
Pengujian ke 5	Berhasil	
Pengujian ke 6	Berhasil	
Pengujian ke 7	Berhasil	
Pengujian ke 8	Berhasil	
Pengujian ke 9	Berhasil	

Pengujian ke 10	Berhasil	

Dari hasil pengujian yang dilakukan sebanyak 10 kali pengujian fungsi modul *master*, dapat dilihat pada Tabel 6.1 bahwa modul *master* mampu memberikan data hasil *sensing* dari sensor yang dapat diamati melalui serial monitor. Hasil tersebut membuktikan bahwa modul *master* dapat berfungsi dengan baik.

6.2 Pengujian Fungsi Modul *Slave*

Pada pengujian fungsi modul *slave* akan dijelaskan apa tujuan dari pengujian ini, bagaimana prosedur pengujiannya dan hasil analisis dari pengujian.

6.2.1 Tujuan Pengujian

Tujuan dari pengujian fungsi modul *slave* adalah untuk membuktikan bahwa modul *slave* dapat berfungsi dengan baik dalam memonitoring kondisi air saat modul *master* mengalami masalah.

6.2.2 Prosedur Pengujian

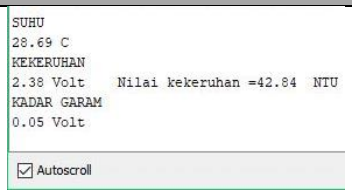
Dalam pengujian fungsi modul *slave*, prosedur pengujiannya adalah sebagai berikut:

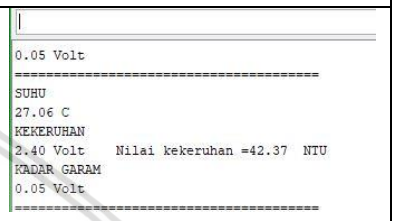
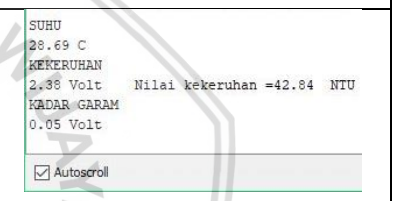
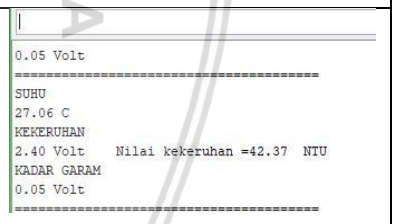
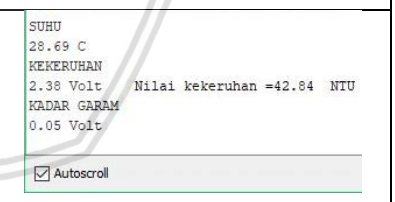
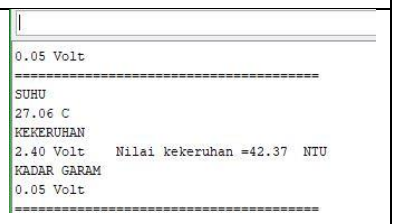
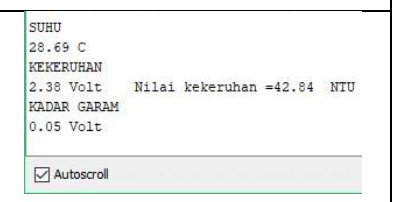
1. Menghubungkan modul *slave* dengan Laptop.
2. Meng-upload program yang sudah dibuat pada aplikasi Arduino IDE.
3. Setelah selesai *upload*, buka serial monitor untuk mengamati hasil dari *sensing* sensor-sensor yang ada pada modul *slave*.

6.2.3 Hasil dan Analisis Pengujian

Dalam hasil dan analisis pengujian akan dijelaskan hasil dari pengujian serta memberikan analisis dari hasil pengujian fungsi modul *slave* seperti yang dapat dilihat pada Tabel 6.2.

Tabel 6.2 Pengujian Fungsi Modul *Slave*

Nama Pengujian	Hasil Pengujian	Screenshoot Pengujian
Pengujian ke 1	Berhasil	

Pengujian ke 2	Berhasi	
Pengujian ke 3	Berhasil	
Pengujian ke 4	Berhasil	
Pengujian ke 5	Berhasil	
Pengujian ke 6	Berhasil	
Pengujian ke 7	Berhasil	
Pengujian ke 8	Berhasil	
Pengujian ke 9	Berhasil	

Pengujian ke 10	Berhasil	SUHU 28.69 C KEKERUHAN 2.38 Volt Nilai kekeruhan =42.84 NTU KADAR GARAM 0.05 Volt Autoscroll

Dari hasil pengujian fungsi modul *slave* yang dilakukan sebanyak 10 kali pengujian, dapat dilihat pada Tabel 6.2 bahwa modul *slave* mampu memberikan data hasil *sensing* dari sensor yang dapat diamati melalui serial monitor. Hasil tersebut membuktikan bahwa modul *slave* dapat berfungsi dengan baik.

6.3 Pengujian Menghindari Kondisi *Race*

Pada pengujian menghindari kondisi *race* akan dijelaskan apa tujuan dari pengujian ini, bagaimana prosedur pengujiannya dan hasil analisis dari pengujian.

6.3.1 Tujuan Pengujian

Tujuan dari pengujian menghindari kondisi *race* adalah untuk memastikan bahwa ketika sistem dinyalakan pertama kali baik modul *master* maupun modul *slave* secara bersamaan tidak akan terjadi kondisi *race* atau kedua modul tidak ada yang mengalah dan dalam kondisi aktif keduanya.

6.3.2 Prosedur Pengujian

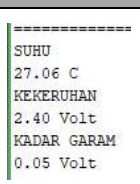
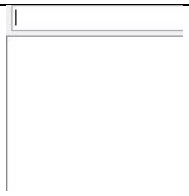
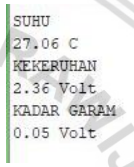
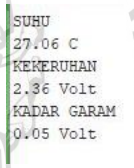
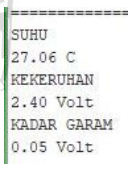
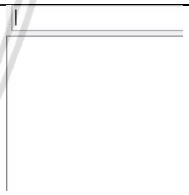
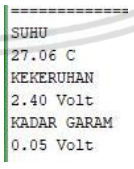
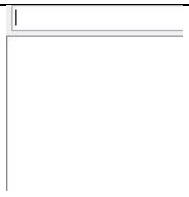
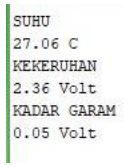
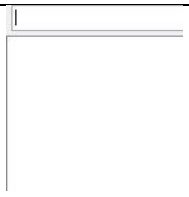
Dalam pengujian menghidari kondisi *race*, prosedur pengujiannya adalah sebagai berikut:

1. Menghubungkan modul *master* dengan Laptop.
2. Meng-*upload* program yang sudah dibuat pada aplikasi Arduino IDE.
3. Setelah selesai *upload* pada modul *master*,cabut modul *master* dari Laptop.
4. Hubungkan modul *slave* dengan Laptop.
5. Meng-*upload* program yang sudah dibuat pada aplikasi Arduino IDE.
6. Setelah selesai *upload* pada modul *slave*, cabut modul *slave* dari Laptop.
7. Pasang kabel penghubung modul *master* dan modul *slave.*
8. Hubungkan modul *master* dan modul *slave* pada sumber daya power secara bersamaan.
9. Melihat hasil melalui serial monitor pada masing masing modul.

6.3.3 Hasil dan Analisis Pengujian

Dalam hasil dan analisis pengujian akan dijelaskan hasil dari pengujian serta memberikan analisis dari hasil pengujian menghidari kondisi *race* seperti yang dapat dilihat pada Tabel 6.3.

Tabel 6.3 Pengujian Menghindari Kondisi *Race*

Nama Pegujian	Berhasil Menghindari Kondisi *Race*	Screenshoot	
		Modul *Master*	Modul *Slave*
Pengujian ke 1	Ya		
Pengujian ke 2	Ya		
Pengujian ke 3	Ya		
Pengujian ke 4	Ya		
Pengujian ke 5	Ya		
Pengujian ke 6	Ya		
Pengujian ke 7	Ya		

Pengujian ke 8	Ya	========== SUHU 27.06 C KEKERUHAN 2.40 Volt KADAR GARAM 0.05 Volt	
Pengujian ke 9	Ya	SUHU 27.06 C KEKERUHAN 2.36 Volt KADAR GARAM 0.05 Volt	
Pengujian ke 10	Ya	SUHU 27.06 C KEKERUHAN 2.36 Volt KADAR GARAM 0.05 Volt	

Dari hasil pengujian menghindari kondisi *race* yang dilakukan sebanyak 10 kali pengujian dapat dilihat dari Tabel 6.3 bahwa saat kedua modul tersebut dinyalakan bersama-sama, modul *master* akan langsung aktif dan menampilkan data dari hasil tangkapan data ketiga sensor sedangkan modul *slave* akan langsung masuk dalam kondisi *sleep* saat pertama kali dinyalakan. Hasil pengujian tersebut membuktikan bahwa sistem dapat menghindari kondisi *race* pada saat sistem dinyalakan bersamaan.

6.4 Pengujian Redundansi Sistem

Pada pengujian redundansi sistem akan dijelaskan apa tujuan dari pengujian rdundansi sistem, bagaimana prosedur pengujiannya dan hasil analisis dari pengujian yang telah dilakukan.

6.4.1 Tujuan Pengujian

Tujuan dari pengujian redundansi sistem ini adalah untuk memastikan bahwa modul *slave* mampu berjalan dengan baik untuk melakukan redundansi terhadap modul *master* ketika terdapat kesalahan dari salah satu atau beberapa sensor yang ada pada modul *master*.

6.4.2 Prosedur Pengujian

Dalam melakukan pengujian redundansi sistem, prosedur pengujiannya adalah sebagai berikut:

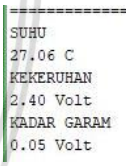
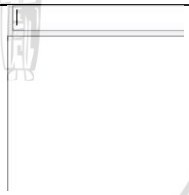
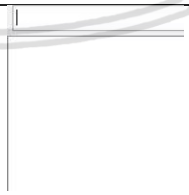
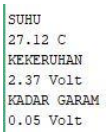
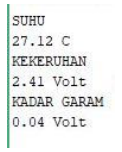
1. Menghubungkan kedua modul ke Laptop.
2. Setelah modul terhubung, buat kesalahan pada sensor-sensor yang ada pada modul *master.*
3. Mengamati apa yang terjadi baik pada modul *master* dan *slave* maupun mengamati melalui serial monitor yang ada pada aplikasi Arduino IDE.

6.4.3 Hasil dan Analisis Pengujian

Dalam hasil dan analisis pengujian akan dijelaskan hasil dari pengujian redundansi sistem serta memberikan analisis terhadap hasil pengujian tersebut. Hasil pengujian dan analisi akan dibagi menjadi tiga bagian sebagai berikut:

1. Pengujian saat sensor suhu mengalami masalah.

Tabel 6.4 Pengujian Redundansi Kesalahan Sensor Suhu

Nama Pegujian	Hasil Redundansi	Screenshoot	
		Modul *Master*	Modul *Slave*
Pengujian ke 1	Berhasil		
Pengujian ke 2	Berhasil		
Pengujian ke 3	Berhasil		
Pengujian ke 4	Berhasil		
Pengujian ke 5	Berhasil		
Pengujian ke 6	Berhasil		

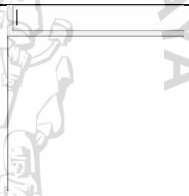
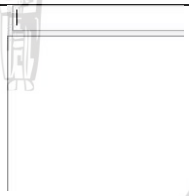
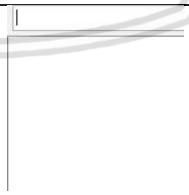
Pengujian ke 7	Berhasil		SUHU 27.12 C KEKERUHAN 2.41 Volt KADAR GARAM 0.05 Volt
Pengujian ke 8	Berhasil		SUHU 27.06 C KEKERUHAN 2.38 Volt KADAR GARAM 0.04 Volt
Pengujian ke 9	Berhasil		SUHU 27.06 C KEKERUHAN 2.36 Volt KADAR GARAM 0.05 Volt
Pengujian ke 10	Berhasil		SUHU 27.12 C KEKERUHAN 2.37 Volt KADAR GARAM 0.05 Volt

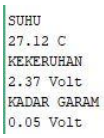
Pada hasil pengujian pertama yang dilakukan sebanyak 10 kali pengujian dapat dilihat dari Tabel 6.4 bahwa ketika modul *master* mengalami masalah pada sensor suhunya, maka modul *master* akan masuk dalam kondisi *sleep* dan tidak melakukan sensing sensor, sehingga modul *slave* akan aktif dan akan memberikan data hasil tangkapan sensor-sensor yang ditampilkan melalui serial monitor karena menerima sinyal *low* dari modul *master*.Dari hasil pengujian redundansi kerusakan sensor suhu, redundasi sistem dapat berjalan 100% ketika sensor suhu mengalami kerusakan.

2. Pengujian saat sensor kekeruhan mengalami masalah.

Tabel 6.5 Pengujian Redundansi Kesalahan Sensor Kekeruhan

Nama Pegujian	Hasil Redundansi	Screenshoot	
		Modul *Master*	Modul *Slave*
Pengujian ke 1	Berhasil		SUHU 27.12 C KEKERUHAN 2.41 Volt KADAR GARAM 0.05 Volt

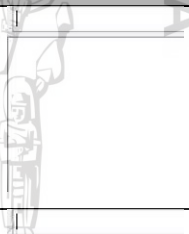
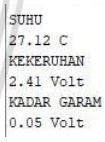
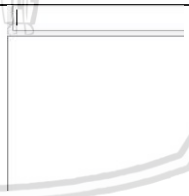
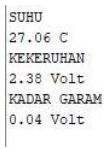
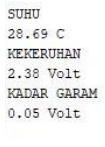
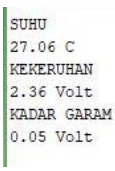
Pengujian ke 2	Berhasil		SUHU 27.06 C KEKERUHAN 2.38 Volt KADAR GARAM 0.04 Volt
Pengujian ke 3	Berhasil		========== SUHU 27.06 C KEKERUHAN 2.40 Volt KADAR GARAM 0.05 Volt
Pengujian ke 4	Berhasil		SUHU 27.06 C KEKERUHAN 2.36 Volt KADAR GARAM 0.05 Volt
Pengujian ke 5	Berhasil		SUHU 27.12 C KEKERUHAN 2.37 Volt KADAR GARAM 0.05 Volt
Pengujian ke 6	Berhasil		SUHU 27.06 C KEKERUHAN 2.38 Volt KADAR GARAM 0.04 Volt
Pengujian ke 7	Berhasil		SUHU 27.12 C KEKERUHAN 2.41 Volt KADAR GARAM 0.05 Volt
Pengujian ke 8	Berhasil		SUHU 27.06 C KEKERUHAN 2.38 Volt KADAR GARAM 0.04 Volt
Pengujian ke 9	Berhasil		SUHU 27.12 C KEKERUHAN 2.41 Volt KADAR GARAM 0.05 Volt

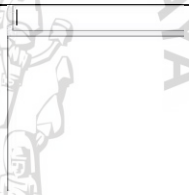
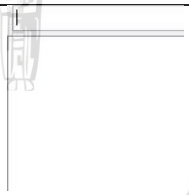
Pengujian ke 10	Berhasil		

Pada hasil pengujian yang dilakukan sebanyak 10 kali untuk menguji sistem redundansi ketika sensor kekeruhan terjadi kerusakan dapat dilihat dari Tabel 6.5 bahwa ketika terdapat masalah pada sensor kekeruhan yang terdapat pada modul *master*, maka modul *master* akan masuk dalam kondisi *sleep.* Sehingga modul *slave* akan aktif dan memberikan data hasil tangkapan sensor-sensor yang ditampilkan pada serial monitor karena menerima sinyal *low* dari modul *master.* Dari hasil pengujian redundansi kerusakan sensor kekeruhan, redundasi sistem dapat berjalan 100% ketika sensor kekeruhan mengalami kerusakan.

3. Pengujian saat sensor kadar garam mengalami masalah.

Tabel 6.6 Pengujian Redundansi Kerusakan Sensor Kadar Garam

Nama Pegujian	Hasil Redundansi	Screenshoot	
		Modul *Master*	Modul *Slave*
Pengujian ke 1	Berhasil		
Pengujian ke 2	Berhasil		
Pengujian ke 3	Berhasil		
Pengujian ke 4	Berhasil		

Pengujian ke 5	Berhasil		SUHU 28.69 C KEKERUHAN 2.38 Volt KADAR GARAM 0.05 Volt
Pengujian ke 6	Berhasil		SUHU 27.06 C KEKERUHAN 2.38 Volt KADAR GARAM 0.04 Volt
Pengujian ke 7	Berhasil		SUHU 27.12 C KEKERUHAN 2.41 Volt KADAR GARAM 0.05 Volt
Pengujian ke 8	Berhasil		SUHU 27.06 C KEKERUHAN 2.38 Volt KADAR GARAM 0.04 Volt
Pengujian ke 9	Berhasil		SUHU 28.69 C KEKERUHAN 2.38 Volt KADAR GARAM 0.05 Volt
Pengujian ke 10	Berhasil		SUHU 27.12 C KEKERUHAN 2.37 Volt KADAR GARAM 0.05 Volt

Pada hasil pengujian yang dilakukan sebanyak 10 kali untuk menguji redundansi sistem ketika sensor kadar garam mengalami masalah atau mati dapat dilihat dari Tabel 6.6 bahwa ketika terdapat masalah pada sensor kadar garam yang terdapat pada modul *master*, maka modul *master* akan masuk dalam kondisi *sleep.* Sehingga modul *slave* akan aktif dan memberikan data hasil tangkapan sensor-sensor yang ditampilkan pada serial monitor karena menerima sinyal *low* dari modul *master.* Dari hasil pengujian redundansi kerusakan sensor kadar garam, redundasi sistem dapat berjalan 100% ketika sensor kadar garam mengalami kerusakan.

Dari ketiga hasil pengujian diatas dapat dilihat bahwa ketika modul *master* mengalami masalah dari tiap sensor baik sensor suhu, sensor kekeruhan maupun sensor kadar garam maka modul *master* akan masuk dalam kondisi *sleep* dan

modul *slave* akan terbangun karena menerima sinyal *low* dari modul *master*. Hasil pengujian tersebut membuktikan bahwa modul *slave* mampu dengan baik dan memiliki nilai ketepatan sebesar 100% saat meredundansi modul *master* secara otomatis sehingga kesalahan pada modul *master* dapat diatasi dengan baik serta mampu meningkatkan ketersediaan dari sistem akuisisi data sensor yang dalam hal ini digunakan untuk memonitoring kondisi air.

BAB 7 PENUTUP

Bab penutup merupakan bab terakhir dalam dokumen skripsi ini yang akan menjelaskan kesimpulan yang merujuk pada latar belakang dan menjelaskan saran bagi perbaikan dan pengembangan sistem untuk sistem yang lebih baik.

7.1 Kesimpulan

Berdasarkan rumusan masalah yang diangkat di awal dan berdasarkan hasil dari pengujian dan implementasi yang telah dilakukan maka dapat ditarik beberapa poin kesimpulan, seperti yang dapat dilihat di bawah ini.

1. Sistem ini dapat mendeteksi adanya kegagalan pada sistem dengan memanfaatkan komponen IC 7408 yang digunakan untuk memproses output sensor, tujuan dari pemrosesan ini ialah agar output yang dihasilkan oleh IC 7408 dapat dijadikan sebagai input pin interupsi pada modul yang digunakan sebagai modul *master*. Kesalahan pada salah satu atau beberapa sensor dapat dikatahui ketika modul *master* masuk dalam kondisi *sleep* yang berarti modul mengalami masalah. Ini dikarenakan interupsi untuk *sleep* akan aktif ketika *output* yang dihasilkan oleh IC 7408 bernilai *low*, sehingga saat salah satu saja sensor bernilai *low* maka *output* IC 7408 akan bernilai *low* sesuai dengan karakteristik ic ini yaitu hanya ketika seluruh *input* bernilai *high* maka *output* akan dapat menghasilkan nilai *high*.
2. Ketika terdapat kesalahan pada salah satu atau beberapa sensor yang ada di modul *master*,sistem dapat terus berjalan karena terdapat modul yang digunakan sebagai *slave*.Modul *slave* merupakan modul cadangan yang memiliki fungsi sama persis dengan modul *master* yang mampu melakukan *sensing* dengan menggunakan sensor untuk memonitoring kondisi air serta dapat berfungsi sebagai redundansi modul *master* atau menggantikan pekerjaan yang awalnya dilakukan oleh modul *master* sehingga seolah-olah sistem ini tetap bekerja walaupun sebenarnya terdapat kesalahan pada sistem.Hal tersebut dibuktikan dengan melakukan pengujian sebanyak 10 kali untuk mengetahui apakah redundansi sistem sudah berjalan dengan baik dan menghasilkan data 100% sistem mampu meredundansi ketika modul *master* mengalami masalah.
3. Komunikasi modul *master* dan *slave* dilakukan dengan menggunakan media *wire* atau kabel.Modul *slave* akan menerima sinyal dari *output* modul *master*,ketika modul *master* mengalami kesalahan sistem maka modul akan masuk kedalam kondisi *sleep*, ini berarti modul *master* akan mengirimkan sinyal *low* kepada modul *slave* dan akan membangunkan modul *slave* yang pada awalnya berada dalam kondisi *sleep*. Hal ini membuktikan bahwa modul *slave* mampu meredundansi modul *master* yang mengalami masalah.

7.2 Saran

Beberapa saran dari penulis yang dapat diberikan sebagai bahan untuk memperbaiki dan mengembangkan sistem adalah sebagai berikut:

1. Memperhatikan waktu peralihan dari modul *master* ke modul *slave* agar tidak terlalu lama,sehingga dapat meningkatkan *availability* sistem.
2. Media komunikasi antara modul *master* dan *slave* kedepannya menggunakan media nirkabel sehingga dapat mengurangi penggunaan kabel yang berlebih dan agar sistem terlihat lebih rapi.
3. Menggunakan *board* mikrokontroler selain Arduino UNO untuk dapat dibandingkan performanya baik dari segi ketepatan,kecepatan dan efisiensi dari penerapan sistem.

DAFTAR PUSTAKA

Abidin, Z., dkk., 2014. *Energy efficient communication protocol for wireless microsensor networks*. In: Proc 33rd Hawaii International Conference on Sistem Sciences (pp. 1-10).

Agustiningsih, E., 2016. *Perancangan Perangkat Monitoring Kualitas Air Pada Kolam Budidaya Berbasis Web Localhost.* Universitas Maritim Raja Ali Haji ,Tanjungpinang.
Diakses 08 September 2017.

Dubrova, E., 2012. *Fault-Tolerant Design*. s.l.:s.n.

Anonim, 2018. https://www.arduino.cc/en/Guide/Introduction#, Diakses 20 Januari 2018

Losq, J., 1975. *Influence of fault-detection and switching mechanisms on the reliability of standby systems*. In: Digest 5th International Symposium on Fault-Tolerant Computing, (pp. 81–86).

National Instrument, 2008. *Redundant System Basic Consepts*. http://www.ni.com/white-paper/6874/en/. Diakses 16 Januari 2018.

Sambora, Y., 2016. *Monitoring Kualitas Air Pada Budidaya Udang Berbasis ATMEGA328 yang Terkonfigurasi Bluetooth HC-05*. UNY,Yogyakarta.