Jurnal ELKOMIKA | Vol. 4 | No. 2 | Halaman 208 - 227
ISSN (p): 2338-8323 Juli - Desember 2016
ISSN (e): 2459-9638

Penerapan Metoda *Serial Peripheral Interface* (SPI) pada Rancang Bangun *Data Logger* berbasis SD *card*

RATNA SUSANA, MUHAMMAD ICHWAN, SAVERO AL PHARD

Institut Teknologi Nasional Bandung
Email : ratnassn@yahoo.com

ABSTRAK

Serial Peripheral Interface (SPI) adalah protokol komunikasi yang dapat digunakan sebagai interface komunikasi antara mikrokontroler dengan SD Card. Dengan menerapkan metoda SPI pada data logger berbasis SD Card, maka dapat diketahui karakteristik protokol komunikasi SPI antara mikrokontroler dengan SD Card. SD Card diformat dengan tipe FAT 16, dan data di dalam SD Card berupa sekumpulan paket data sensor yang diambil secara periodik dan disimpan dalam bentuk file dengan format.csv. Berdasarkan format paket data sensor yang dibuat, dapat dihitungwaktu perekaman data yang diperlukan agar kapasitas SD Card terisi penuh oleh data sensor. Hasil penelitian menunjukkan,bahwa metoda SPI yang diterapkan pada penelitian ini memiliki sifat akan melakukan pemeriksaan berulang pada pin MISO terhadap command yang dikirimkan oleh mikrokontroler melalui pin MOSI. Proses read/write data pada SD Card data logger memiliki keberhasilan 100%, karena SD Card telah terinisialisasi dalam mode SPI melalui perintah reset dan init SD Card. Komunikasi ini dapat dilakukan dengan menggunakan crystal 4 Mhz – 20 Mhz. Untuk pengujian konfigurasi SPI, hanya Independent Slave Configuration yang dapat digunakan pada komunikasi SPI dengan 2 SD card sebagai slave.

Kata kunci *: Serial Peripheral Interface (SPI), Data Logger, SD card, FAT16*

ABSTRACT

Serial Peripheral Interface (SPI) is a communication protocol that can be applied as a communication interface between microcontroller to SD Card. By implementing the SPI method to a data logger based on SD Card, it can be known the characteristics of the SPI communications protocol between microcontroller to SD Card. SD Card formatted in FAT 16 type, and data on the SD Card is the form of sensor data packets collection which be captured periodically and saved in .csv format file. Based on the sensor data packet format is created, it can be calculated recording time data required so that the SD Card capacity completely filled by the sensor data. Research results show, that the SPI method applied in this study has the properties will do repeated testing on MISO pin to the command sent by the microcontroller through the MOSI pin. The read / write data on the SD Card data logger has a 100% success, because the SD Card has been initialized in SPI mode through the reset and init SD Card command. This communication can be established using crystal 4 Mhz - 20 Mhz. At SPI configuration testing, only the Independent Slave Configuration can be used in SPI communication with 2 SD card as a slave.

Kata kunci *: Serial Peripheral Interface (SPI), Data Logger, SD card, FAT16*

1. PENDAHULUAN

Data logger merupakan perangkat elektronik yang terhubung dengan sensor dan berfungsi untuk mencatat data secara berkala **(Badhiye, dkk, 2011)**. Perangkat ini dapat diaplikasikan pada sistem-sistem yang memerlukan pencatatan ataupun perekaman data secara otomatis. Sesuai dengan fungsinya *data logger* dilengkapi dengan mikrokontroler dan memerlukan memori untuk menyimpan data. Memori yang digunakan dapat berupa memori internal di dalam mikrokontroler ataupun memori eksternal.

Beberapa *data logger* dirancang dan diimplementasikan menggunakan mikrokontroler yang terhubung dengan memori eksternal jenis SD *card*. SD *card* digunakan sebagai media penyimpanan data dari sejumlah parameter data hasil pengukuran**(Vojtech, 2011; Nhivekar, 2011; Rudi, 2013; Dutta, 2014)**. Agar mikrokontroler dengan SD *card* dapat saling berkomunikasi, dalam hal ini adalah melakukan proses *read/write* data, maka diperlukan suatu protokol komunikasi yang dapat menghubungkan keduanya. SD *card* terhubung dengan mikrokontroler melalui protokol komunikasi *Serial Peripheral Interface*(SPI)**(Vojtech, 2011; Nhivekar, 2011; Dutta, 2014)**. SPI merupakan protokol komunikasi dengan *interfacing* yang sederhana dan kecepatan yang dimilikinya masih memungkinkan untuk terjadinya komunikasi transfer data dengan mudah **(Choudhury,2014)**.

Penggunaan memori eksternal pada *data logger* menjadi sangat dibutuhkan ketika sistem yang dibangun memerlukan pencatatan dan perekaman data dalam jumlah besar. Salah satu memori eksternal yang biasa digunakan adalah SD *card*, seperti pada penelitian **(Vojtech, 2011; Nhivekar, 2011; Rudi, 2013; Dutta, 2014)**yangmembahas aplikasi *data logger* dengan dilengkapi SD Card. Hal penting yang perlu diperhatikan adalah bagaimana proses *read/write* data pada SD *card* dapat dilakukan. Atas dasar itulah penulis melakukan penelitian yang bertujuan untuk mengetahui karakteristik protokol komunikasi SPI pada *data logger* yang menggunakan SD *card*, sehingga proses *read/write* data pada SD *card* dapat dilakukan oleh mikrokontroler.

Berdasarkan tujuan penelitian tersebut, maka permasalahan yang dibahas adalah bagaimana menerapkan metoda SPI pada *data logger* berbasis SD *card*. Dari pembahasan tersebut dapat diketahui konfigurasi yang harus dilakukan antara mikrokontroler dengan SD *card*, cara penulisan (*write*) data pada SD *card*, cara pembacaan (*read*) data pada SD *card*, pengaruh *clock* pada komunikasi SPI dan pengaruh kapasitas SD *card* terhadap banyaknya data yang direkam. Untuk keperluan penelitian ini maka penulis merealisasikan sebuah *prototype data logger* yang dilengkapi SD *card*, selanjutnya data tersebut dikirimkan menuju *Personal Computer* (PC) pengguna sistem.

Serial Peripheral Interface (SPI) adalah salah satu protokol komunikasi serial *synchronous* kecepatan tinggi yang dimiliki oleh ATmega32, karena itulah *prototype data logger* dibangun dengan menggunakan mikrokontroler ATmega32. Dengan menggunakan AVR 8 bit ini, maka sistem dialokasikan untuk 8 data sensor analog yang disimulasikan dengan resistor variabel. Umumnya *data logger* bersifat *real time* dan data sensor yang disimpan dilengkapi dengan data waktu, untuk itu diperlukan komponen RTC untuk memberikan informasi waktu pengambilan data sensor. *Data logger* ini dilengkapi pula dengan modul Xbee Pro RF sebagai modul *wireless*, sehingga dapat berfungsi sebagai sistem telemetri dan dapat diaplikasikan pada sistem *Wireless Sensor Network* (WSN). Karena *data logger* yang dibangun dapat berfungsi sebagai sistem telemetri,maka pengambilan data dapat dilakukan dari jarak jauh

secara nirkabel dan dapat diletakkan pada tempat-tempat yang sulit dijangkau ataupun daerah-daerah rawan bencana.

Pada penelitian ini SD *card* yang terhubung dengan mikrokontroler diformat dengan tipe FAT16 yang merupakan salah satu tipe *file system* 16 bit. Sedangkan data yang disimpan di dalam SD *card* berupa sekumpulan paket data sensor yang diambil secara periodikdan disimpan dalam bentuk *file* dengan format .csv. Hasil penelitian ini selanjutnya dapat dijadikan sebagai dasar untuk menerapkan metoda SPI antara mikrokontroler dengan memori eksternal lainnya.

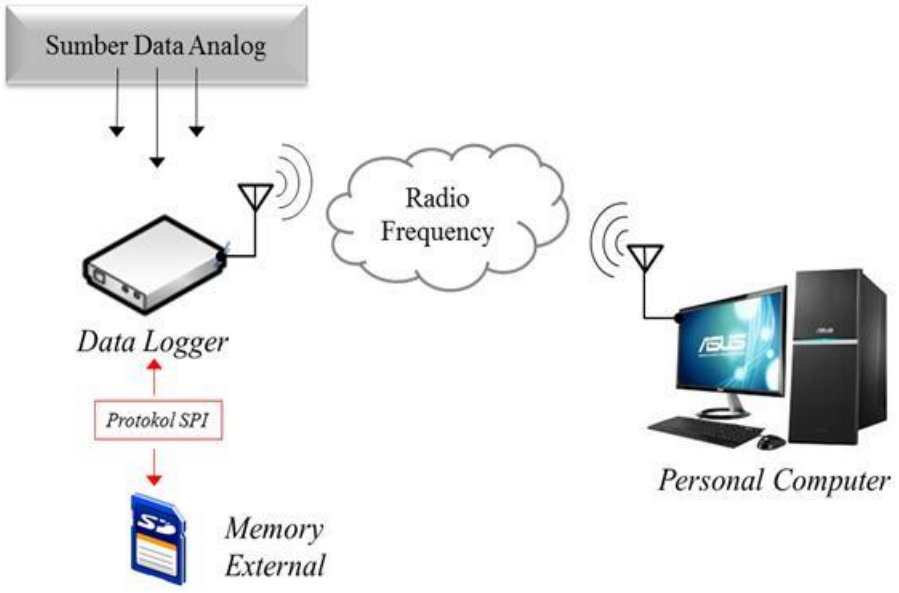
2. METODOLOGI PENELITIAN

2.1 Gambaran Umum Sistem

Pada penelitian ini, *data logger* yang dirancang bertujuan sebagai media pendukung penelitian mengenai karakteristik komunikasi SPI, sehingga *data logger* yang dibangun memiliki spesifikasi sistem sebagai berikut:

1. Menggunakan mikrokontroler sebagai pengendali.
2. Sumber data pada *data logger* berasal dari sensor analog. Jumlah pin analog yang dapat digunakan maksimum sebanyak 8 pin dengan resolusi 8-bit.
3. Data yang disimpan dilengkapi dengan informasi waktu.
4. Menggunakan media tanpa kabel (*wireless*) sebagai pengiriman data antara modul *data logger* dengan PC.
5. Pengambilan data dilakukan secara periodik.
6. Menggunakan memori eksternal sebagai media penyimpanan data.
7. Komunikasi antara mikrokontroler dan memori eksternal menggunakan komunikasi SPI.
8. Menggunakan catu daya 5 volt.

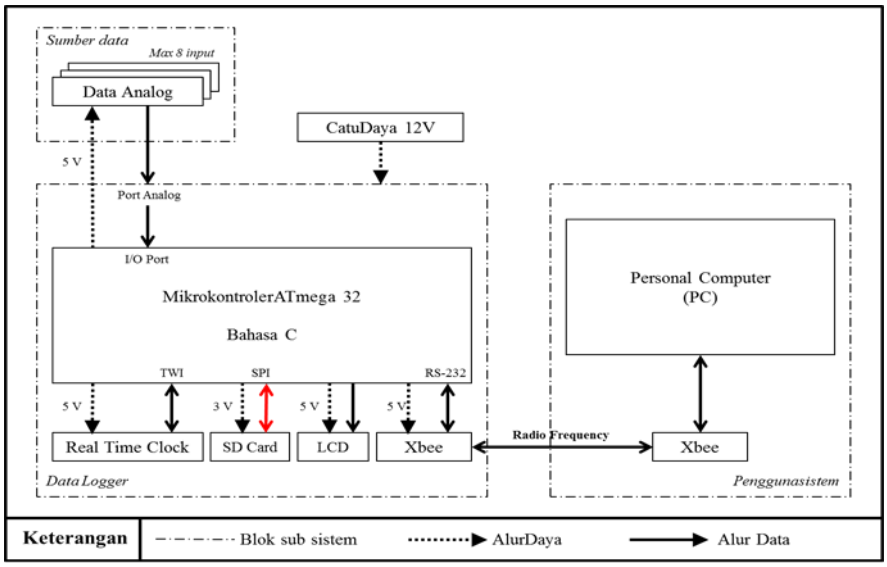
Berdasarkan spesifikasi sistem yang telah disusun, maka secara umum sistem dapat digambarkan seperti pada Gambar 1.



Gambar 1. Gambaran umum system

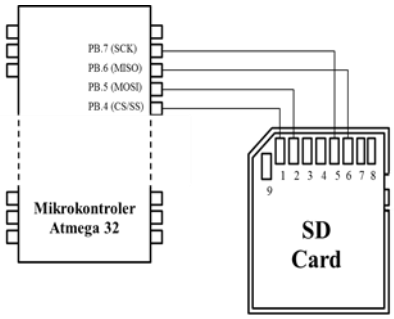
2.2 Perancangan dan Realisasi *Hardware*

Berdasarkan spesifikasi yang telah diuraikan maka perlu dirancang perangkat *data logger* dengan blok diagram pada Gambar 2.



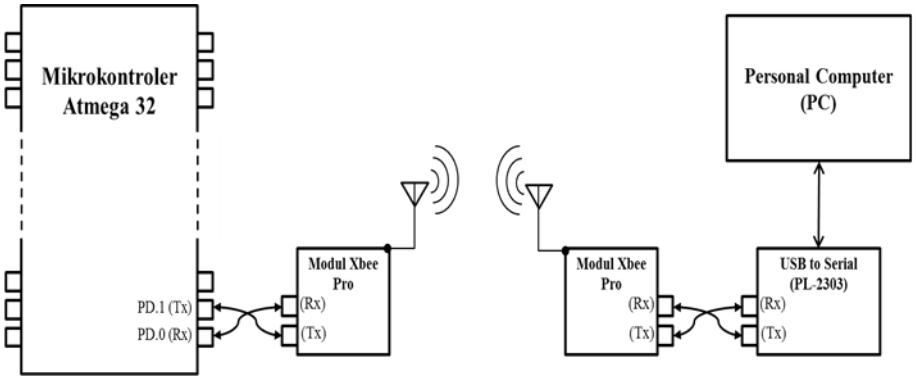
Gambar 2. Blok diagram sistem

Secara keseluruhan sistem terdiri dari 3 sub sistem, yaitu sumber data, *data logger*, dan pengguna sistem. Sumber data analog yang berasal dari 8 resistor variabel akan diolah oleh mikrokontroler untuk kemudian dicatat dan disimpan pada SD *card*. Data pada SD *card* dapat dibaca langsung pada *Personal Computer* (PC) secara otomatis, yaitu dengan mengirimkan data di dalam SD *card* ke PC secara nirkabel. Data-data hasil pengukuran dari 8 resistor variabel dan informasi waktu dari RTC akan disimpan pada SD *card* dalam bentuk *file* dengan format .csv. *Comma Separated Value* (CSV) merupakan suatu format *file* yang berupa nilai-nilai yang dipisahkan dengan titik koma (;). Konfigurasi rangkaian SD *card* dengan mikrokontroler dapat dilihat pada Gambar 3.



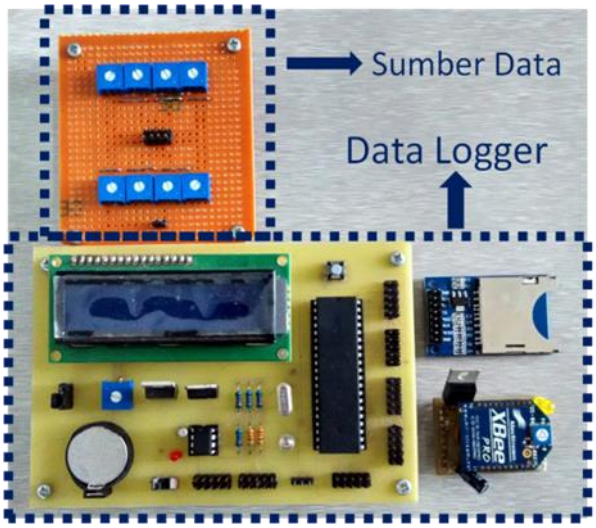
Gambar 3. Rangkaian mikrokontroler dan SD *card*

Alur pengiriman data dari mikrokontroler sampai ke PC melalui beberapa tahap. Pertama-tama data dari mikrokontroler akan dikirimkan secara serial ke modul RF (*XBee Pro*). Setelah itu pengiriman data antar modul RF *transmitter* dan modul RF*receiver* terjadi secara *wireless*. Setelah sampai pada modul RF *receiver,* data diteruskan ke PC menggunakan komunikasi serial. Selanjutnya data yang didapatkan akan ditampilkan pada PC.Sistem transmisi yang digunakan antara*data logger*dan PC terlihat seperti pada Gambar 4.



Gambar 4. Sistem Transmisi

Realisasi perangkat keras dilakukan dengan mencetak sistem minimum mikrokontroler ATmega32. Pada *board* ini terintegrasi pula dengan LCD, RTC, serta catu daya +5 Vdc dan +3 Vdc. Sedangkan hubungan antar mikrokontroler dengan resistor variable, modul SD *card* dan modul Xbee Pro menggunakan kabel *jumper*. Realisasi perangkat keras dapat dilihat pada Gambar 5.

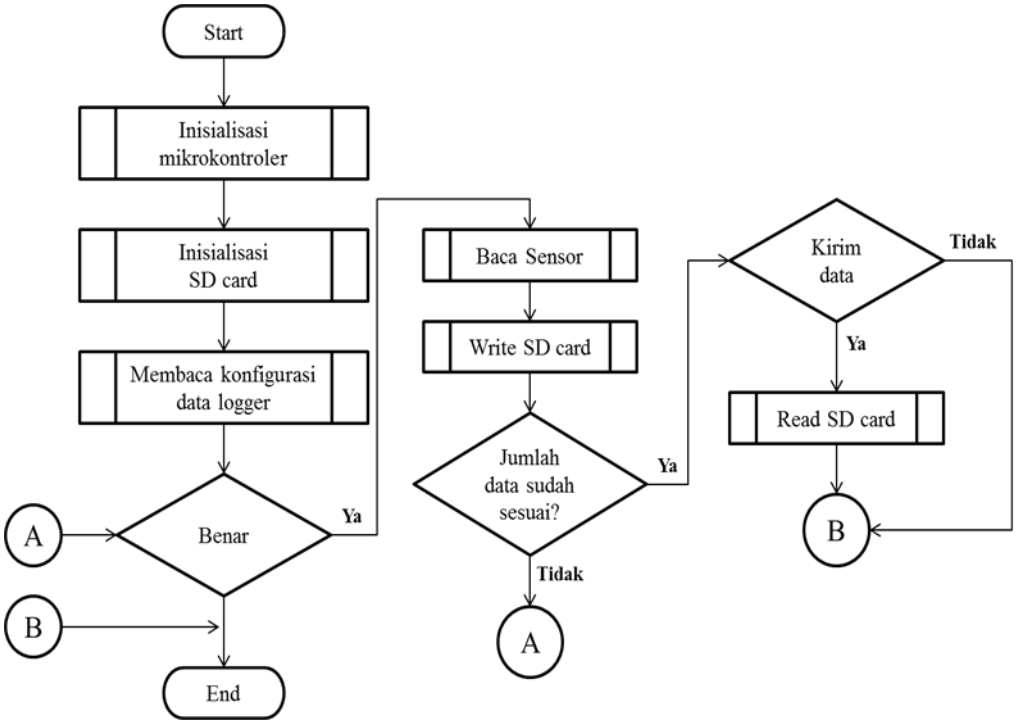


Gambar 5. Realisasi perangkat keras

2.3 Perancangan dan Realisasi *Software*

Perangkat lunak (*software*) yang digunakan pada penelitian ini ada 3, yaitu *software* pada pemrograman mikrokontroler menggunakan *CodeVision* AVR, *software* pada pemrograman Xbee Pro menggunakan X-CTU, dan *software* pada *Personal Computer* (PC) menggunakan Terminal.Program pada mikrokontroler terdiri dari beberapa sub program yaitu inisialisasi, membaca konfigurasi, membaca data dari sensor, menulis dan membaca media penyimpanan serta mengirimkan data ke PC.

Program yang akan ditanamkan pada mikrokontroler mengikuti diagram alir (*flowchart*) pemrograman untuk *data logger* seperti terlihat pada Gambar 6.



Gambar 6. *Flowchart* pemrograman pada mikrokontroler

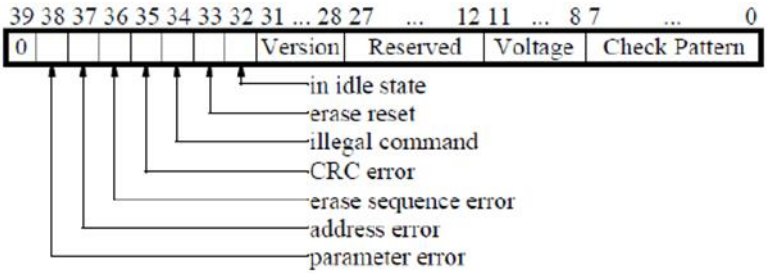
Cara berkomunikasi dengan SD *card* adalah dengan mengirimkan *command* kepada SD *card* dan menerima *response* dari SD *card*. SD *card command* yang terdiri dari 48-bit.

Tabel 1 merupakan penjelasan dari bit *command*.

Tabel 1.*Command* untuk SD *card*

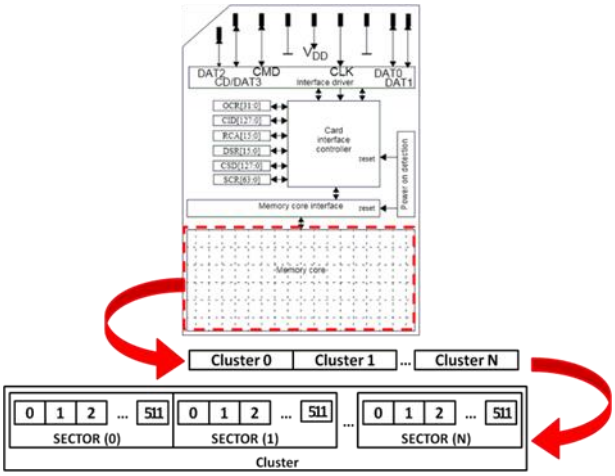
Bit Position	Width(bits)	Value	Name
47	1	0	Start bit
46	1	1(for command), 0(for response)	Transmission bit
45-40	6		Command Number
39-8	32		Argument
7-1	7		CRC
0	1	1	Stop bit

Gambar 7 merupakan penjelasan dari bit *response*.



Gambar 7. *Response* dari SD *card*

SD *card* memiliki arsitektur seperti ditunjukkan pada Gambar 8.



Gambar 8. Arsitektur SD *card*

Seluruh data *file* terletak pada *Memory core*. *Memory core*terdiri dari beberapa *cluster* dan setiap *cluster* berisi beberapa sektor. Masing-masing sektortersusun dari kumpulan *byte* dan setiap *byte* memiliki 8 bit lokasi memori. Format data SD *card* umumnya menggunakan format *File Allocation Table* (FAT). FAT12 digunakan untuk kapasitas lebih kecil dari 16 MB. FAT16 digunakan untuk kapasitas 32 MB hingga 2 GB. FAT32 digunakan untuk kapasitas di atas 2 GB (SDHC).

Pada penelitian ini digunakan SD *card* yang telah diformat dengan tipe FAT16. FAT16 adalah salah satu tipe *file* sistem 16-bit, yang berarti bahwa alamat *cluster* tidak dapat lebih besar dari 16-bit. Oleh karena itu jumlah maksimum *cluster* yang dapat dirujuk dengan sistem FAT16 adalah 2^{16} (65536) *cluster* dengan tiap *cluster* memiliki beberapa sektor yang masing-masing sektor memiliki kapasitas 512 *byte*.

Tabel 2. Isi partisi FAT16

Reserved Region + Boot Sector
File Allocation Table (FAT) Region
Root Directory
Data Region

Berikut ini adalah penjelasan mengenai cara menggunakan SD *card* dengan FAT16. SD *card* terbagi atas sektor-sektor dan tiap satu sektornya berisi 512 *byte*. Secara *default*, proses baca atau tulis selalu melibatkan satu sektor (512 *byte*).Hal pertama yang harus dilakukan adalah membaca parameter SD *card* dengan urutan langkah sebagai berikut:

1. Mengirimkan perintah *Reset* dan *Init* ke SD *card*.
2. Membaca *Master Boot Record* (berada di sektor 0) untuk mengetahui lokasi *Boot Sector*. Lalu *Boot Sector* dibaca secara keseluruhan.
3. Nilai pada alamat-alamat tertentu dari *Boot Sector* diambil dan dihitung sehingga didapat parameter antara lain: alamat FAT *Region*, alamat *Root Directory*, alamat *Data Region*, jumlah *sector* per *cluster*, tipe FAT, dan kapasitas SD *card*.

Setelah membaca parameter SD *card*, maka langkah selanjutnya adalah menulis data pada SD *card*. Urutan langkahnya adalah sebagai berikut:

1. Mengirimkan perintah *Reset* dan *Init* ke SD *card*.
2. Selanjutnyaprogram akan menulis Nama Kartu dan Nama *File* pada *Root Directory*. Nilai *Root Directory* akan dibaca untuk mengetahui posisi *sector* awal untuk *file*.Program akan menulis tabel FAT pada FAT *Region*.

3. PENGUJIAN DAN ANALISIS

3.1 Metoda Pengujian

Berdasarkan tujuan pada penelitian ini, metode pengujian yang dilakukan penulis yaitu pengujian mengenai karakteristik dari protokol komunikasi *Serial Peripheral Interface* (SPI) dan pengujian mengenai *data logger*. Beberapa tahap pengujian protokol komunikasi SPI, dilakukan dengan menggunakan *Logic Analyzer* untuk menampilkan sinyal dari beberapa pin yang akan diamati. Beberapa tahap pengujian lainnya dilakukan melalui pengamatan data pada PC.

Logic analyzer adalah perangkat elektronik yang dapat meng-*capture* beberapa sinyal dari sistem digital atau rangkaian digital. Selain dapat meng-*capture* sinyal, *Logic analyzer* juga dapat menampilkan data sinyal yang telah di-*capture* ke dalam PC. Gambar 8 adalah *Logic analyzer* yang digunakan dalam pengujian ini.

Gambar 9. *Logic Analyzer*

Logic analyzer pada pengujian ini menggunakan tipe Saleae Logic Analyzer yang memiliki 8 buah kanal. Alat ini digunakan untuk meng-*capture* dan menampilkan sinyal dari beberapa pin yang digunakan dalam komunikasi SPI yaitu *Slave Select* ($\overline{SS}$), *Master Input | Slave Output* (MISO), *Master Output | Slave Input* (MOSI), dan *Serial Clock* (SCK).

3.2 Pengujian *Serial Peripheral Interface* (SPI)

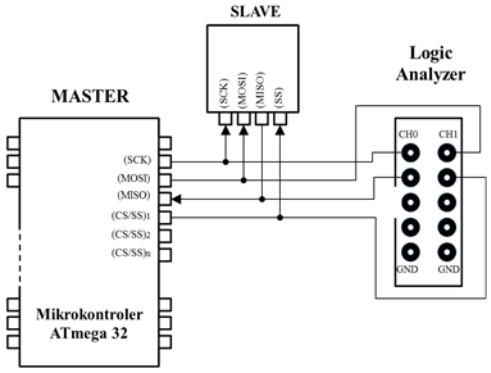
Sebelum melakukan pengujian sistem *data logger*, terlebih dahulu dilakukan pengujian protokol komunikasi *Serial Peripheral Interface* (SPI) pada komunikasi antar mikrokontroler dengan SD *card*. Pengujian ini bertujuan untuk mengetahui karakteristik dari protokol komunikasi SPI yang terjadi antar mikrokontroler dengan SD *card*.

3.2.1 Pengujian konfigurasi Mikrokontroler dan SD *card*

Pengujian ini bertujuan untuk mengetahui konfigurasi SPI yang dapat digunakan dalam komunikasi mikrokontroler dan SD *card*. Pada pengujian ini dilakukan inisialisasi SD *card* oleh mikrokontroler dengan program *reset* dan *init* SD *card*.

a. *One Slave Configuration*

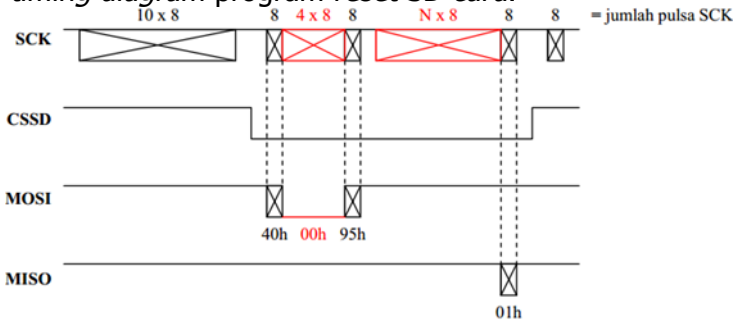
One slave configuration adalah konfigurasi 1 (satu) *Master* dengan 1 (satu)*Slave*. Pada pengujian ini mikrokontroler sebagai *Master* dan SD *card* sebagai *Slave.* SD *card* yang digunakan memiliki kapasitas sebesar 256 MB. Gambar 10 menunjukkan rangkaian *One Slave Configuration*.



Gambar10.*One Slave Configuration*

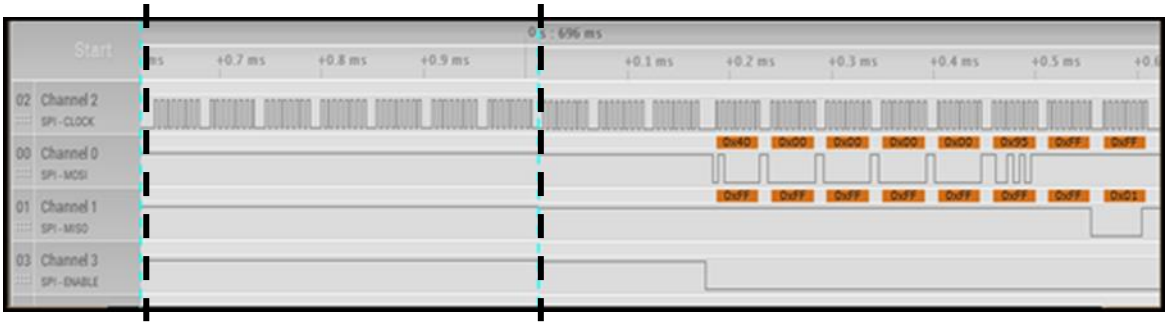
Reset* SD *card

Gambar 11 adalah *timing diagram* program *reset* SD *card*.



Gambar 11.*Timing diagram reset* SD *card*

Hasil pengujian dengan program *reset* dapat dilihat pada Gambar 12.

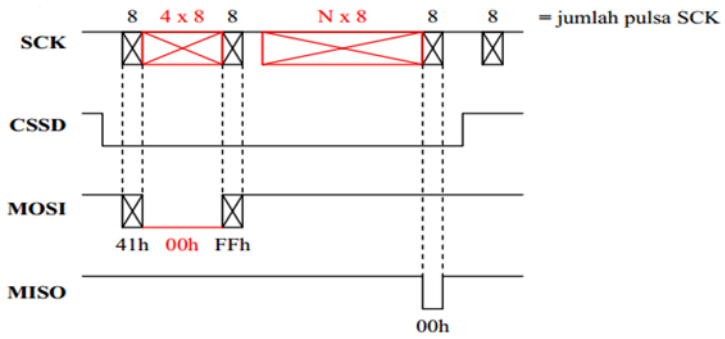


Gambar 12.*Timing diagram* hasil pengujian *reset* SD *card*

Gambar 12 menunjukkan bahwa sinyal MOSI dari mikrokontroler mengirimkan sinyal *resetcommand* (400000000095h) kepada SD *card*. Pin MISO yang bernilai 01h menyatakan bahwa perintah *reset* ini sudah diterima oleh SD *card*. Hasil pengujian ini telah sesuai dengan *Timing diagram reset* SD *card* pada Gambar 11, artinya SD *card* telah terinisialisasi ke dalam mode SPI.

Init* SD *card

Gambar 13 adalah *timing diagram* program *init* SD *card*.



Gambar 13.*Timing diagram init* SD *card*

Hasil pengujian dengan program *init* dapat dilihat pada Gambar 14.

Gambar 14.*Timing diagram* hasil pengujian *init* SD *card*

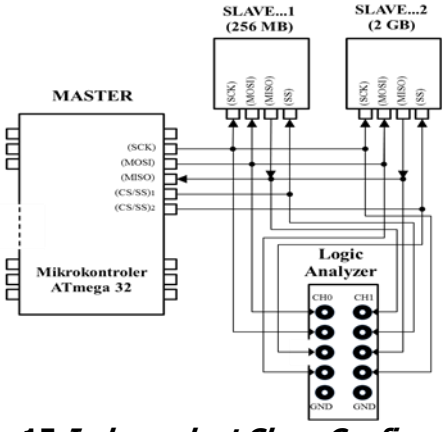
Gambar 14 menunjukkan bahwa sinyal MOSI pada mikrokontroler mengirimkan sinyal *init command* (4100000000FFh) kepada SD *card*. Pin MISO yang bernilai 00h menyatakan bahwa SD *card* telah terinisialisasi. Hasil pengujian ini telah sesuai dengan *timing diagraminit* SD *card* (Gambar 13), artinya SD *card* telah terinisialisasi dan siap untuk menerima *command* selanjutnya.

b. *Multiple Slaves Configuration*

Multiple Slaves Configuration adalah konfigurasi 1 (satu) *Master* dengan lebih dari 1 (satu) *Slave*. Pada pengujian ini mengkonfigurasikan 2 (dua) SD *card* dengan masing-masing memiliki kapasitas sebesar 256 MB dan 2 GB. Konfigurasi *Multiple Slaves* pada SPI terdapat 2 (dua) macam yaitu *Independent Slave Configuration* dan *Daisy Chain Configuratio.*

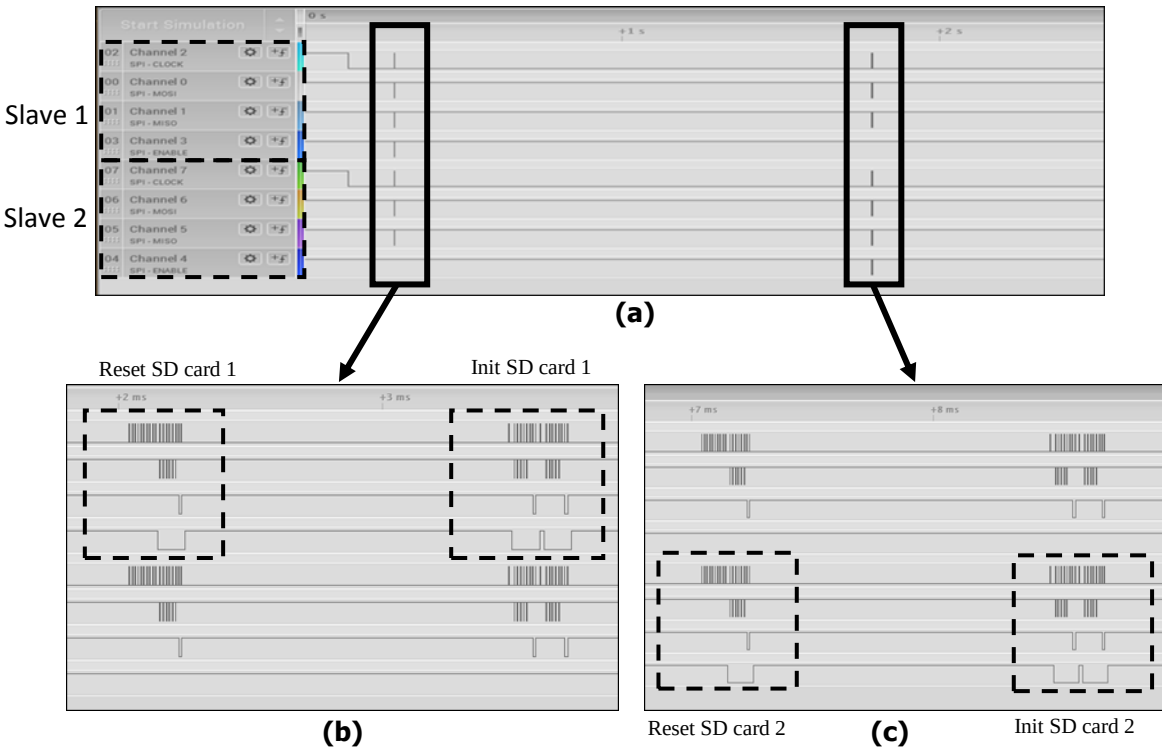
Independent Slave Configuration

Gambar 15 menunjukkan rangkaian *Independent Slave Configuration*.



Gambar 15.*Independent Slave Configuration*

Hasil pengujian *Independent Slave Configuration* dapat dilihat pada Gambar 16.

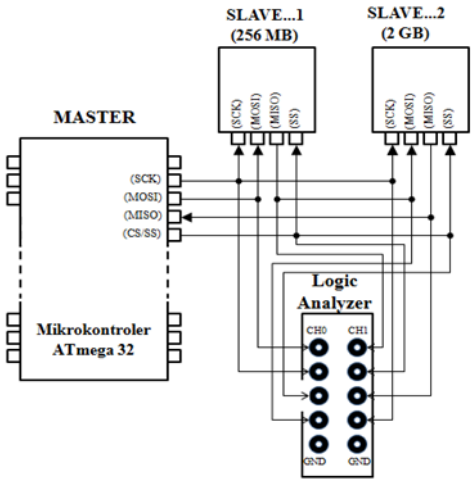


Gambar 16. (a) Hasil inisialisasi *Independent Slave Configuration*;(b) Hasil inisialisasi SD *card* 1; (c) Hasil inisialisasi SD *card* 2

Pada Gambar 16, mikrokontroler melakukan proses inisialisasi pada masing-masing *Slave* secara bergantian dengan cara memberi logika *low* pada pin CSSD SD *card* yang akan diakses. Dan masing-masing *Slave* telah terinisialisasi dan dapat digunakan sebagai media penyimpanan data.

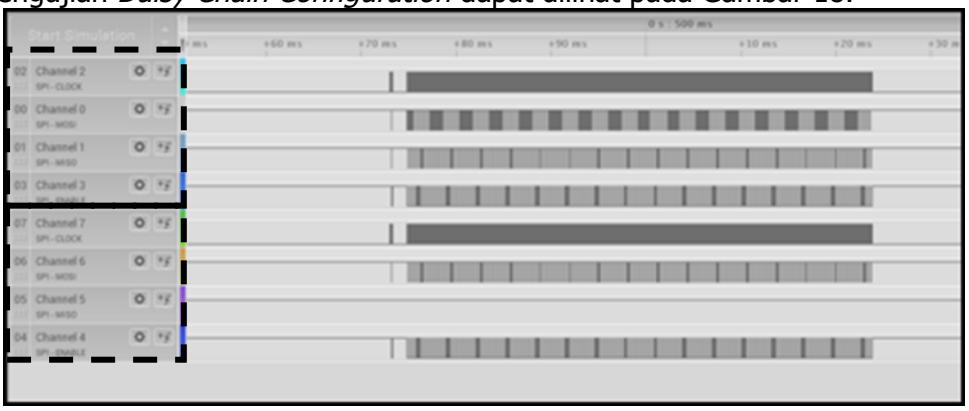
Daisy Chain Configuration

Gambar 17 menunjukkan rangkaian *Daisy Chain Configuration*.



Gambar 17.*Daisy Chain Configuration*

Hasil pengujian *Daisy Chain Configuration* dapat dilihat pada Gambar 18.



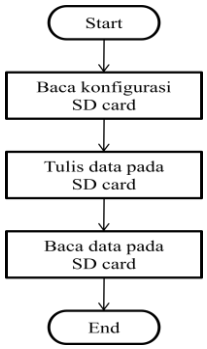
Gambar 18.Hasil inisialisasi *Daisy Chain Configuration*

Pada Gambar 18 terjadi pemeriksaan berulang dikarenakan sinyal MISO dari *Slave* 1 tidak terhubung dengan mikrokontroler melainkan terhubung ke MOSI pada *Slave* 2. Hubungan ini menyebabkan mikrokontroler tidak menerima informasi mengenai proses inisialisasi SD *card* sehingga terjadi pengulangan sampai batas maksimal.

Dari hasil pengujian *Multiple Slave Configuration* dapat diambil kesimpulan bahwa *Independent Slave Configuration* dapat digunakan untuk komunikasi SPI antar mikrokontroler dengan SD *card* sedangkan *Daisy Chain Configuration* tidak dapat digunakan untuk komunikasi SPI antar mikrokontroler dengan SD *card*.

3.2.2 Pengujian *Read / Write* data pada SD *card*

Pada pengujian *read/write* data SD *card*, dilakukan pemrograman pada mikrokontroler mengikuti *flowchart* pada Gambar 19.

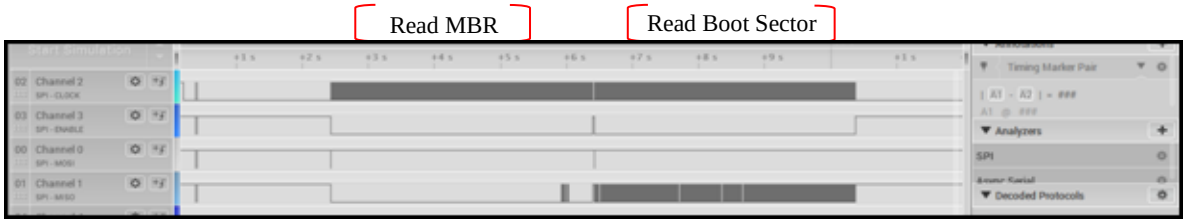


Gambar 19.*Flowchart* pengujian *read/write* data pada SD *card*

a. *One Slave Configuration*

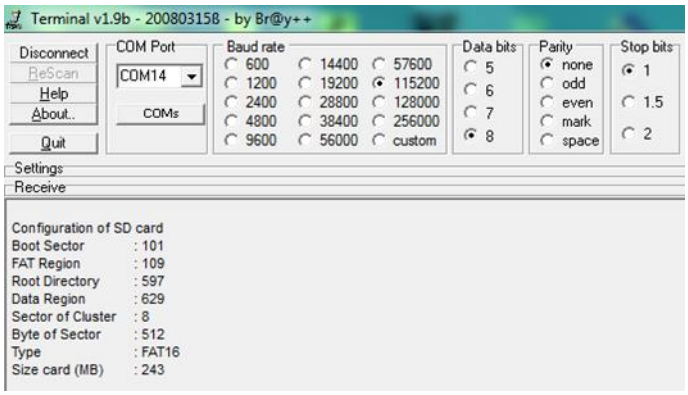
1) *Read* konfigurasi SD *card*

Sub program ini bertujuan untuk mengetahui konfigurasi mengenai SD *card* sehingga data yang ditulis dapat dibaca oleh mikrokontroler maupun PC. Gambar 20 adalah sinyal dari program *read* konfigurasi.



Gambar 20. *Timing diagram* proses *read* konfigurasi SD *card*

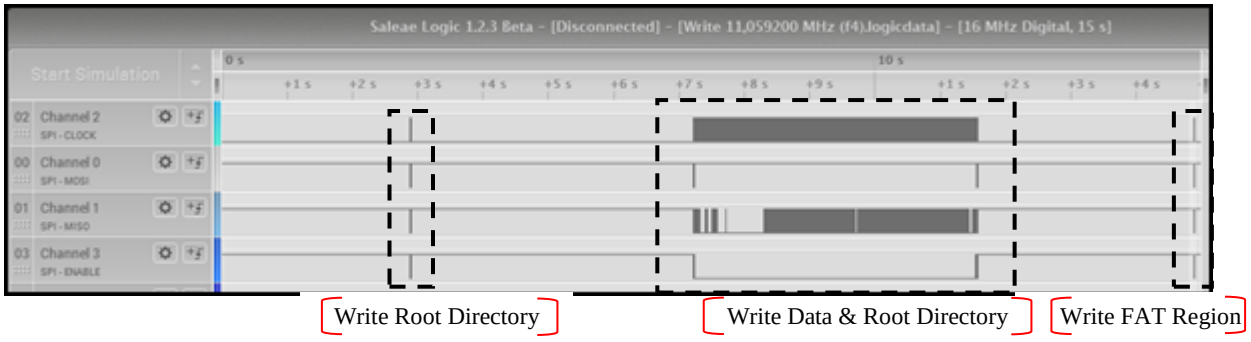
Gambar 21adalah gambar hasil pembacaan *Master Boot Record* dan *Boot Sector*.



Gambar 21. Tampilan beberapa parameter MBR dan *Boot Sector*

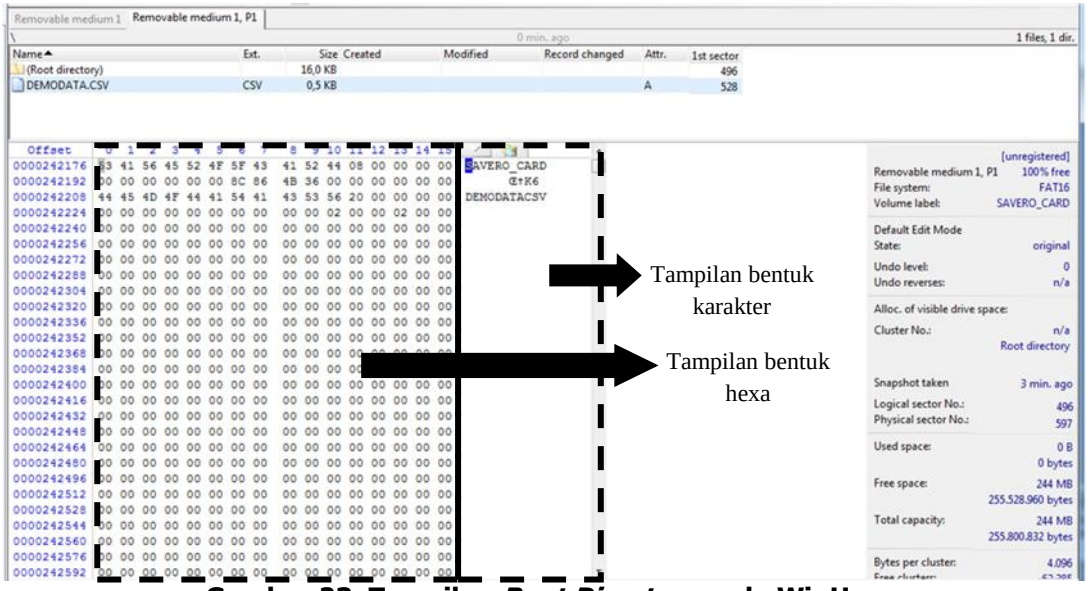
2) *Write* data pada SD *card*

Sub progam ini bertujuan untuk melakukan penulisan datapada SD *card* oleh mikrokontroler dengan komunikasi SPI.Gambar 22 merupakan *Timing diagram* dari program *write* data pada SD *card*.



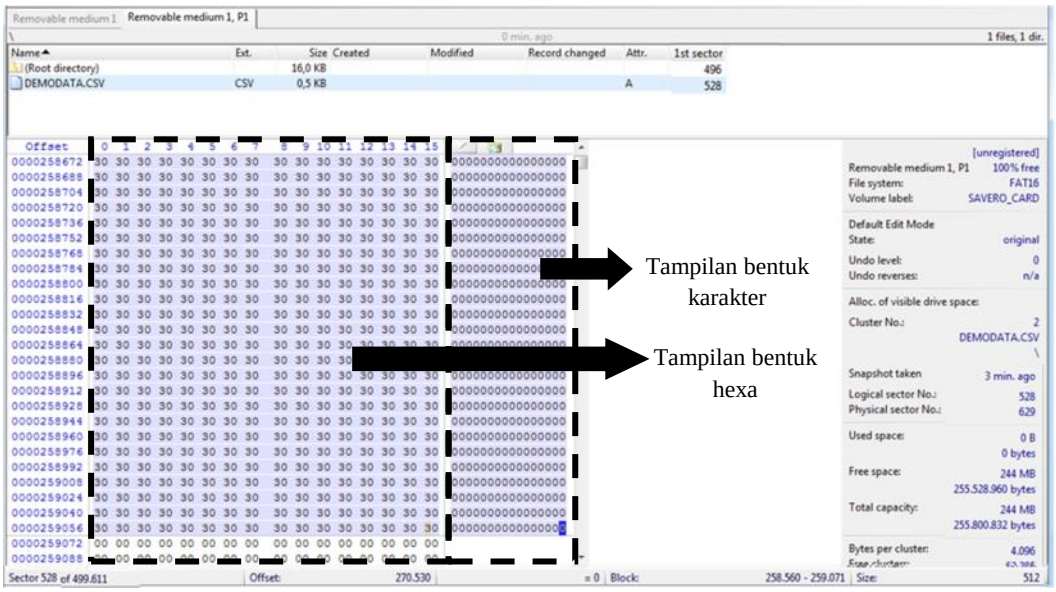
Gambar 22. *Timing diagram* proses *write* data pada SD *card*

Gambar 23 menunjukkan pembacaan *Root Directory* hasil penulisan pada *software* WinHex.



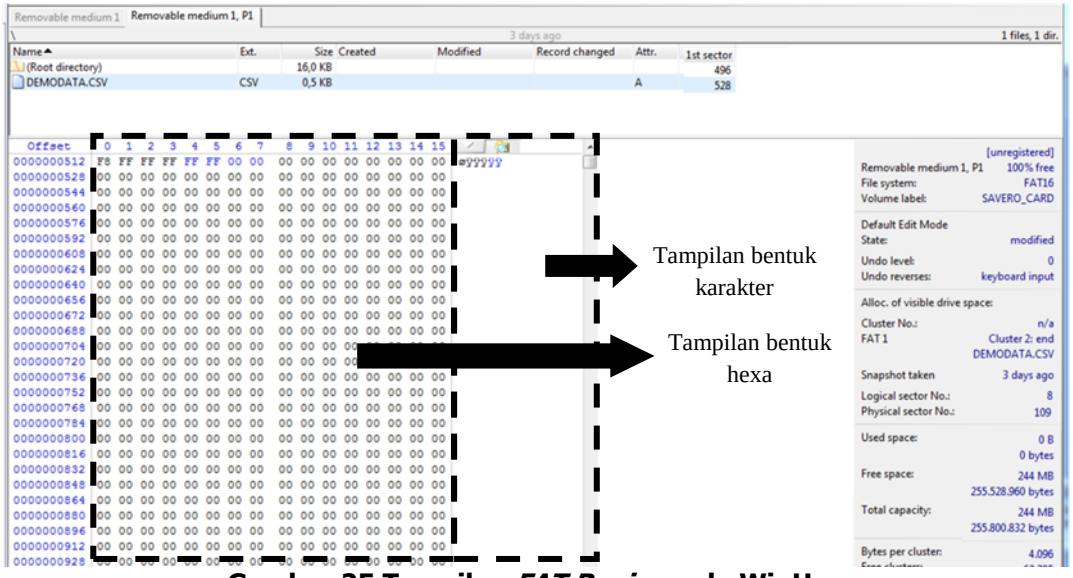
Gambar 23. Tampilan *Root Directory* pada WinHex

Gambar 24 menunjukkan pembacaan *Data Region* hasil penulisan pada *software* WinHex.



Gambar 24. Tampilan *Data Region* pada WinHex

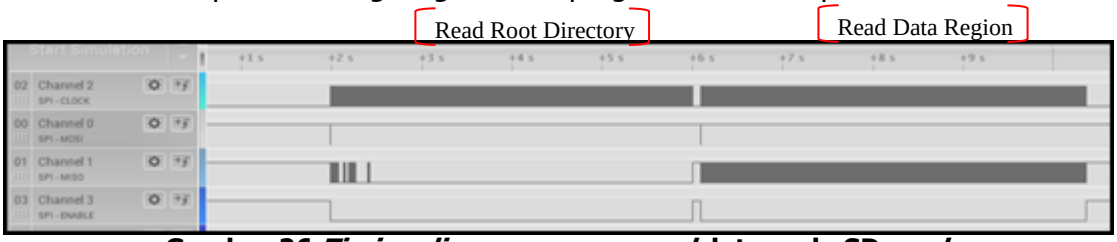
Gambar 25 menunjukkan pembacaan *FAT Region* hasil penulisan pada *software* WinHex.



Gambar 25.Tampilan *FAT Region* pada WinHex

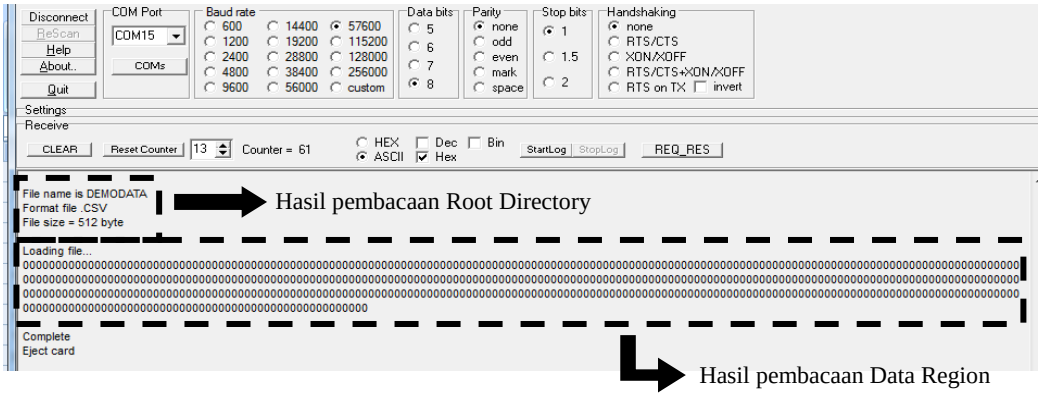
3) *Read* data pada SD *card*

Sub program ini bertujuan untuk melakukan pembacaan data sebanyak 512 *byte* pada SD *card* oleh mikrokontroler dengan komunikasi SPI dan mengirimkan data tersebut ke PC.Gambar 26 merupakan *Timing diagram* dari program *read* data pada SD *card*.

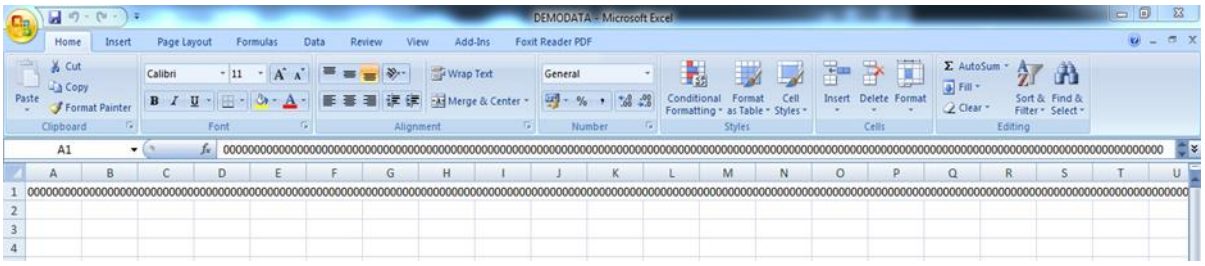


Gambar 26. *Timing diagram* proses *read* data pada SD *card*

Gambar 27 dan 28 merupakan perbandingan hasil pembacaan data pada aplikasi Terminal pada PC dan pembacaan*file* isi SD *card* secara langsung pada PC.



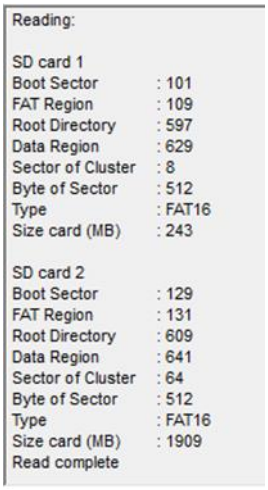
Gambar 27. Pembacaan pada Terminal



Gambar 28. Pembacaan pada Excel

b. *Multiple Slaves Configuration*

Pada pengujian *read/write Multiple Slaves Configuration*, hanya dilakukan dengan menggunakan*Independent Slave Configuration* dikarenakan *Daisy Chain Configuration* tidak dapat digunakan untuk komunikasi antara mikrokontroler dengan SD *card.* Gambar 29 merupakan hasil pembacaan MBR dan *Boot Sector* pada 2 SD *card*.



Gambar 29.Tampilan beberapa parameter MBR dan *Boot Sector*dari 2 SD *card*

Gambar 30 adalah hasil pembacaan data pada aplikasi Terminal pada PC.

Gambar 30.Pembacaan data 2 SD *card* pada Terminal

3.2.3 Pengujian nilai *clock*

Pengujian ini bertujuan mengetahui pengaruh nilai *clock* terhadap waktudalam komunikasi SPI antar mikrokontroler dengan SD *card* saat melakukan *read/write* data sebanyak 512 *byte*.Hasil pengujian mengenai nilai *clock* dapat dirangkum dalam bentuk tabel seperti pada tabel 3.

Tabel 3. Hasil pengujian nilai *clock*

Clock Frequency	Crystal 4 MHz			Crystal 8 MHz			Crystal 11.059200 MHz			Crystal 12 MHz			Crystal 16 MHz			Crystal 20 MHz		
	Avg. Freq.	Time (s)		Avg. Freq.	Time (s)		Avg. Freq.	Time (s)		Avg. Freq.	Time (s)		Avg. Freq.	Time (s)		Avg. Freq.	Time (s)	
		Write	Read		Write	Read		Write	Read		Write	Read		Write	Read		Write	Read
Crystal / 2	2000	12.05	7.636	4000	12.06	7.633	5529.6	12.02	7.649	6222	12.06	7.682	5529.6	12.1	7.723	5818	12.05	7.691
Crystal / 4	1000	12.05	7.64	2000	12.06	7.632	2800	12.01	7.648	3027	12.06	7.682	4000	12.11	7.722	4870	12.08	7.702
Crystal / 8	500	12.06	7.639	1000	12.04	7.632	1383	12.03	7.65	1493	12.07	7.682	2000	12.17	7.75	2489	12.05	7.677
Crystal / 16	250	12.08	7.639	500	12.06	7.632	691.2	12.04	7.651	751.7	12.06	7.682	1000	12.12	7.724	1244	12.06	7.682
Crystal / 32	125	12.09	7.64	250	12.09	7.633	345.6	12.06	7.653	375.8	12.08	7.684	500	12.16	7.756	625.7	12.08	7.683
Crystal / 64	62.53	12.13	7.647	125	12.09	7.635	172.8	12.11	7.657	187.6	12.13	7.687	250	12.19	7.758	312	12.1	7.684
Crystal / 128	31.25	12.22	7.649	62.53	12.23	7.645	86.42	12.21	7.663	93.72	12.22	7.694	125	12.2	7.766	156.2	12.15	7.689

Berdasarkan hasil pengujian pada Tabel 3 dapat ditarik kesimpulan bahwa semua *crystal* yang terdapat di pasaran dapat digunakan untuk komunikasi SPI antara mikrokontroler ATmega32 dengan SD *card* menggunakan format FAT16.

Setelah pengujian protokol komunikasi SPI dilakukan, maka didapat beberapa kesimpulan mengenai karakteristik dari protokol komunikasi SPI yang terjadi antar mikrokontroler dengan SD *card*. Beberapa kesimpulan tersebut adalah:

1. Agar *Master* dapat berkomunikasi dengan *Slave*, maka *Master* perlu memberi logika *low* pada pin CSSD. Apabila digunakan lebih dari 1 *Slave*, maka *Master* perlu member logika *low* pada pin CSSD dari *Slave* yang akan diajak berkomunikasi dan memberi logika *high* pada pin CSSD dari *Slave* lainnya.
2. Pada pegujian protokol komunikasi SPI antar mikrokontroler dengan SD *card* menggunakan beberapa *command*, yaitu *reset command* (CMD0), *init command* (CMD1), *read command* (CMD17),dan *write command* (CMD24).
3. Berdasarkan Tabel 3 dapat disimpulkan bahwa *crystal* 4 MHz - 20 MHz dapat digunakan untuk komunikasi SPI antara mikrokontroler dengan SD *card* dengan masing-masing nilai *clock*darisetiap*crystal*adalah *crystal*/2 - *crystal*/128.

3.3 Pengujian *Data Logger*

Pengujian *data logger* dilakukan agar dapat mengetahui bahwa sistem telah direalisasikan dengan benar dan dapat mendukung penelitian ini. Pengujian sistem *data logger* dilakukan dengan pengujian keseluruhan sistem *data logger* mengikuti diagram alir dari sistem *data logger* yang telah dirancang seperti pada Gambar 5. Pengujian ini meliputi pengiriman data secara *wireless* dan pengujian kapasitas data SD *card* yang dapat diisi berdasarkan waktu tertentu.
Pada pengujian mengenai *data logger*dilakukan perekaman data 8 (delapan) resistor variabel ke dalam SD *card* berkapasitas 2 GB. Perangkat *data logger* diletakkan berjarak 5 meter dari PC dan pengiriman data dilakukan secara *wireless*.Tabel4 dan 5 adalah penjelasan dari format satu paket datayang terdiri dari 59 *byte*.

Tabel 4.Format Paket Data Baris Pertama

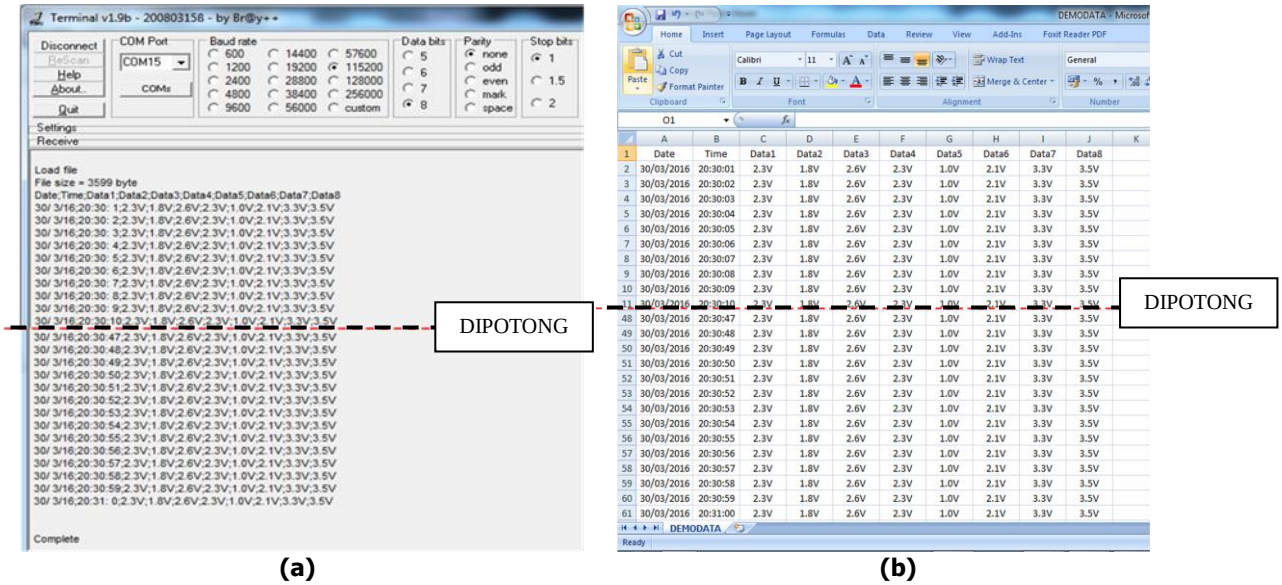
Baris pertama										
Keterangan	Date	Time	Data1	Data2	Data3	Data4	Data5	Data6	Data7	Data8
Jumlah byte	4 byte	4 byte	5 byte	5 byte	5 byte	5 byte	5 byte	5 byte	5 byte	5 byte

Tabel 5.Format Paket Data Baris Kedua sampai baris ke-n

Baris kedua hingga baris ke-n										
Format data	dd/mm/yy	hh:mm:ss	x.xV	x.xV	x.xV	x.xV	x.xV	x.xV	x.xV	x.xV
Jumlah byte	8 byte	8 byte	4 byte	4 byte	4 byte	4 byte	4 byte	4 byte	4 byte	4 byte

Note : karakter ';' digunakan untuk memisahkan antar kolom
karakter '\n' dan '\r' digunakan untuk mengakhiri baris

Gambar 31 merupakan hasil pengujian selama 1 menit dengan skala periodik pengambilan data setiap 1 detik.



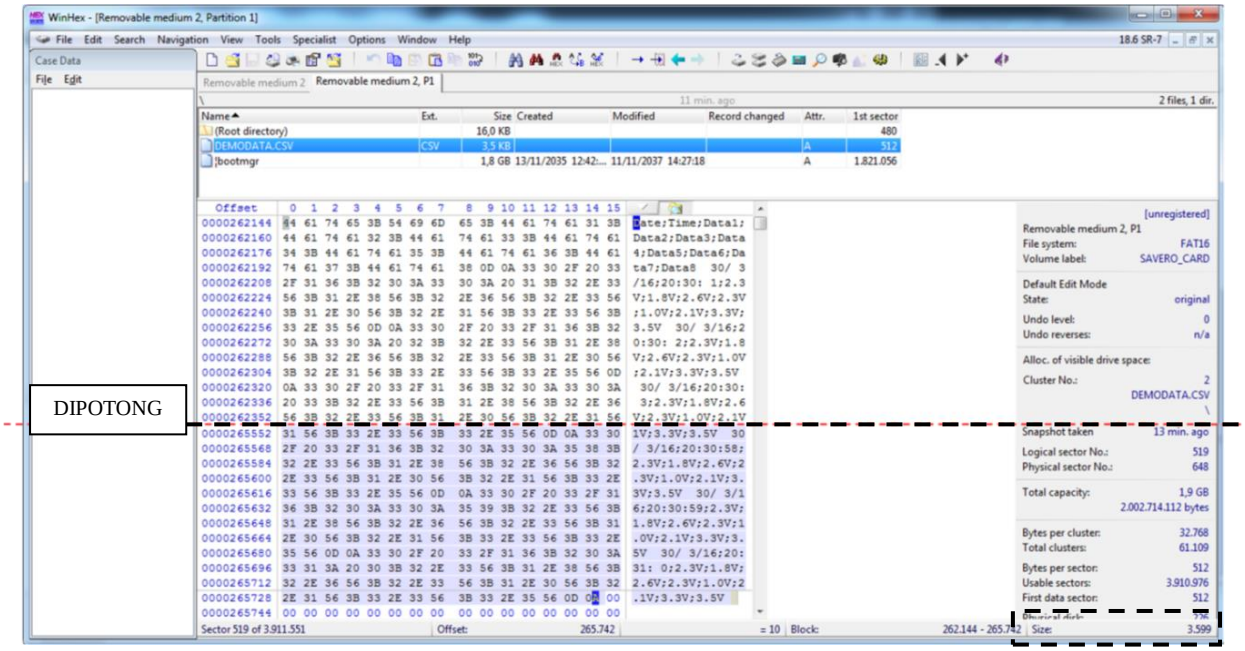
(a) **(b)**

Gambar 31.(a) Data yang diterima pada *software* Teminal secara *wireless*;(b) Data pada isi *file* SD *card* yang dibaca dengan *software* Excel

Dari kedua data yang telah didapat dilakukan verifikasi data guna mengetahui apakah data yang dikirim telah sesuai dengan data yang direkam. Hasil dari verifikasi kedua data tersebut

adalah tidak terjadi data yang hilang. Hal ini terbukti dari 61 data yang tersimpan dalam SD *card* semuanya dapat terbaca pula pada *software* Terminal yang menampung data yang dikirim secara *wireless*.

Selain verifikasi data, dilakukan pula pengecekan dengan *software* WinHex untuk melihat isi dari SD *card*. Gambar 32 menampilkan *window* dari *software* WinHex.



Gambar 32.Data yang dibaca pada *software* WinHex

Pada Gambar 32 dapat terlihat pada kotak (dengan garis putus-putus) mengenai ukuran dari *file* sebesar 3599 *byte*. Dari percobaan selama 1 menit dapat ditarik kesimpulan bahwa dalam perekaman data selama 1 menit dengan periodik pengambilan data setiap 1 detik menghasilkan data sebanyak 3599 *byte* atau 61 baris yang berisi 1 baris pertama mengenai *header* dan 60 baris selanjutnya mengenai data hasil perekaman.
Berdasarkan format paket data yang dibuat, dapat diperoleh persamaan untuk menentukan lama waktu perekaman data dan total *byte* data yang dapat direkam pada SD *card* selama waktu perekaman tersebut. Kedua persamaan ini memiliki hubungan satu dengan lainnya melalui jumlah data yang direkam.

$$Waktu\ perekaman\ data = Jumlah\ data\ x\ skala\ periodik \quad (1)$$

$$Total\ byte = (Jumlah\ data\ x\ paket\ data) +\ total\ byte\ baris\ pertama \quad (2)$$

Hasil perhitungan untuk total *byte* data yang direkam pada SD *card* melalui persamaaan 1 dan 2, selanjutnya akan dibandingkan dengan hasil pengujian yang didapat pada *software* WinHex.Berikut adalah contoh perhitungan untuk waktu perekaman 1 menit dengan skala periodik pengambilan data selama 1 detik.

- Perhitungan 1 menit dengan skala periodik 1 detik:

$$Jumlah\ data = \frac{Waktu\ perekaman\ data}{skala\ periodik}$$

$$Jumlah\ data = \frac{60}{1} = 60\ data$$

Setiap 1 paket data berisi 59 *byte* dan baris pertama berisi 59 *byte*, maka:

$$Total\ byte = (Jumlah\ data\ x\ paket\ data) + \ total\ byte\ baris\ pertama$$
$$Total\ byte = (60\ x\ 59)\ + 59 = 3599\ byte$$

Sehingga untuk waktu perekaman 1 menit dengan skala periodik 1 detik menghasilkan 61 baris dengan 3599*byte* dan telah sesuai dengan hasil yang didapat dari pengukuran.

Pengujian *data logger* dilakukan beberapa kali guna mengetahui apakah hasil perhitungan berdasarkan persamaan yang telah didapat sesuai dengan yang diukur. Pengujian ini menggunakan lama waktu yang beragam dengan skala periodik selama 1 detik dan 5 detik. Hasil pengujian dapat dilihat pada Tabel 6 dan 7.

Tabel 6.Perbandingan data yang direkam dengan perhitungan menggunakan periode 1 detik

Waktu	Jumlah data		Total byte	
	Hasil pengukuran	Hasil perhitungan	Hasil pengukuran	Hasil perhitungan
1 menit	61	61	3599 byte	3599 byte
3 menit	181	181	10679 byte	10679 byte
5 menit	301	301	17759 byte	17759 byte
10 menit	601	601	35459 byte	35459 byte
30 menit	1801	1801	106259 byte	106259 byte
1 jam	3601	3601	212459 byte	212459 byte
2 jam	7201	7201	424859 byte	424859 byte
3 jam	10801	10801	637259 byte	637259 byte

Tabel 7. Perbandingan data yang direkam dengan perhitungan menggunakan periode 5 detik

Waktu	Jumlah data		Total byte	
	Hasil pengukuran	Hasil perhitungan	Hasil pengukuran	Hasil perhitungan
1 menit	13	13	767 byte	767 byte
3 menit	37	37	2183 byte	2183 byte
5 menit	61	61	3599 byte	3599 byte
10 menit	121	121	7139 byte	7139 byte
30 menit	361	361	21299 byte	21299 byte
1 jam	721	721	42539 byte	42539 byte
2 jam	1441	1441	85019 byte	85019 byte
3 jam	2161	2161	127499 byte	127499 byte

4. KESIMPULAN

Berdasarkan perancangan, realisasi, dan hasil pengujian sistem yang telah dilakukan, maka penelitian ini dapat disimpulkan sebagai berikut:

1. Berdasarkan pengujian mengenai konfigurasi SPI, protokol komunikasi SPI antara mikrokontroler dengan SD *card* dapat menggunakan lebih dari satu *Slave* (dalam penelitian ini adalah SD *card*) dengan konfigurasi *Independent Slave* dengan cara memberikan logika *low* pada pin $\overline{SS}$ dari *Slave* yang akan diajak berkomunikasi dan memberikan logika *high* pada pin $\overline{SS}$ dari *Slave* yang lainnya seperti terlihat pada Gambar 16.
2. Berdasarkan pengujian mengenai proses *read/write* data pada SD *card*, proses ini telah berhasil dilakukan oleh mikrokontroler melalui 4 tahapan yang diawali inisialisasi SD

card, pembacaan konfigurasi SD *card*, penulisan data pada SD *card*, dan pembacaan data SD *card*.

3. Berdasarkan Tabel 3 dapat disimpulkan bahwa *crystal* 4 MHz - 20 MHz dapat digunakan untuk komunikasi SPI antara mikrokontroler dengan SD *card* dengan masing-masing nilai *clock* darisetiap *crystal* adalah *crystal*‖2 - *crystal*‖128.
4. Berdasarkan Tabel 6 dan 7 terlihat bahwa persamaan (1) dan (2) dapat digunakan untuk menghitung waktu perekaman data dan total *byte*, Sehingga persamaan (1) dan (2) dapat digunakan untuk melakukan perhitungan banyaknya data yang dapat direkam oleh SD *card* dengan kapasitas tertentu, seperti pada pengujian sistem *data logger* bahwa dengan skala periodik pengambilan data setiap 1 detik, SD *card* berkapasitas 2 GB mampu menampung data selama 1 tahun 27 hari 8 jam 11 menit 44 detik.

DAFTAR RUJUKAN

Badhiye, S.S., Chatur, P.N., Wakode, B.V. (2011). *Data Logger System: A Survey*. International Journal of Computer Technology and Electronics Engineering(IJCTEE): 24-26.

Choudhury, S., Singh G.K., Mehra R.M. (2014). *Design and Verification Serial Peripheral Interface (SPI) Protocol for Low Power Applications*. International Journal of Innovative Research in Science, Engineering and Tecgnology: 16750-16758. ISSN: 2319-8753.

Dutta, K., P., Rai, P., Pandey, R. K.. (2014). *Microcontroller Based Automatic Multichannel Temperature Monitoring System*. International Journal of Advanced Research in Electrical, Electronics and Instrument Engineering. Vol 3, Issue 9: 11771-11777 ISSN : 2278-8875.

Hartono, R.. (2013). *Perancangan Sistem Data Logger Temperature Baterai Berbasis Arduino Duemilanove*. Proyek Akhir.

Kriti, J., et al. (2015). *Design and Implementation of Microcontroller Based Speed Data Logger*. GE-INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH. VOL-3, ISSUE-5 IF-4.007: 139-145 ISSN: (2321-1717).

Nhivekar, G., S., Mudholker, R., R. (2011). *Data Logger and Remote Monitoring System for Multiple Paramater Measurement Application*. E-Journal of Science & Technology (e-JST). 55-62

SIMAK, V., HRBCEK, J., Juraj, Z. (2011). *Modular Approach During On-Board Unit Development*. Journal of Information, Control and Management Systems. Vol 9: 145-150

Setiono, A., et al. (2010). *Pembuatan Dan Uji Coba Data Logger Berbasis Mikrokontroler Atmega32 Untuk Monitoring Pergeseran Tanah*. Jurnal Fisika Himpunan Fisika Indonesia. Vol 10: 83-94. ISSN 0854-3046.