

# EMS SD/MMC/FRAM *Application Note*

## Akses SD Card & FRAM Menggunakan MCS-51

Oleh: Tim IE

**S**ecure Digital (SD) atau MultiMedia Card (MMC) seringkali digunakan sebagai sarana penyimpanan data pada Personal Digital Assistant (PDA), kamera digital, dan telepon seluler (ponsel). AN ini akan menggunakan EMS SD/MMC/FRAM sebagai sarana penyimpanan data dengan menggunakan SD card dan FRAM yang tersedia.

Aplikasi ini terdiri dari 3 program:

1. DT-51™ Low Cost Micro System / Low Cost Nano System membaca parameter format data SD Card lalu mengirimkannya secara UART ke komputer.
2. Berdasar parameter yang sudah dibaca dan sudah dimasukkan ke program, DT-51™ Low Cost Micro System / Low Cost Nano System akan menulis sebuah file beserta isinya ke dalam SD Card.
3. DT-51™ Low Cost Micro System / Low Cost Nano System menulis data ke FRAM lalu membacanya dan mengirimkannya secara UART ke komputer.

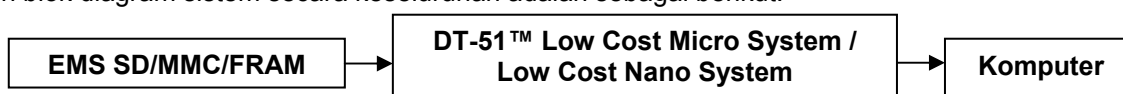
Bahasa pemrograman yang digunakan adalah bahasa BASIC (BASCOM-8051®).

Modul yang diperlukan:

- 1 DT-51™ Low Cost Micro System / Low Cost Nano System,
- 1 EMS SD/MMC/FRAM,
- 1 SD Card dengan format FAT16 (umumnya berkapasitas antara 32 MB hingga 2 GB)

**Pastikan SD card yang digunakan tidak menyimpan data penting atau terproteksi karena program pada AN ini akan menghapus/merusak isi sebelumnya.**

**A**dapun blok diagram sistem secara keseluruhan adalah sebagai berikut:



**Gambar 1**  
**Blok Diagram**

**H**ubungan antara modul-modul tersebut terdapat pada tabel berikut:

| DT-51™ Low Cost Micro System / Low Cost Nano System<br>Port 1 | EMS SD/MMC/FRAM<br>J2 |
|---|-----------------------|
| GND (pin 1)   | GND (pin 1)           |
| VCC (pin 2)   | VCC (pin 2)           |
| P1.0 (pin 3)*   | SCL (pin 3)           |
| P1.1 (pin 4)*   | SDA (pin 4)           |
| P1.2 (pin 5)*   | CD (pin 5)            |
| P1.3 (pin 6)*   | WP (pin 6)            |
| P1.4 (pin 7)*   | CSSD (pin 7)          |
| P1.5 (pin 8)*   | MOSI (pin 8)          |
| P1.6 (pin 9)*   | MISO (pin 9)          |
| P1.7 (pin 10)*  | SCK (pin 10)          |

\* Pin ini tidak mutlak dan dapat diganti pin lain tetapi harus mengubah program

**Tabel 1**

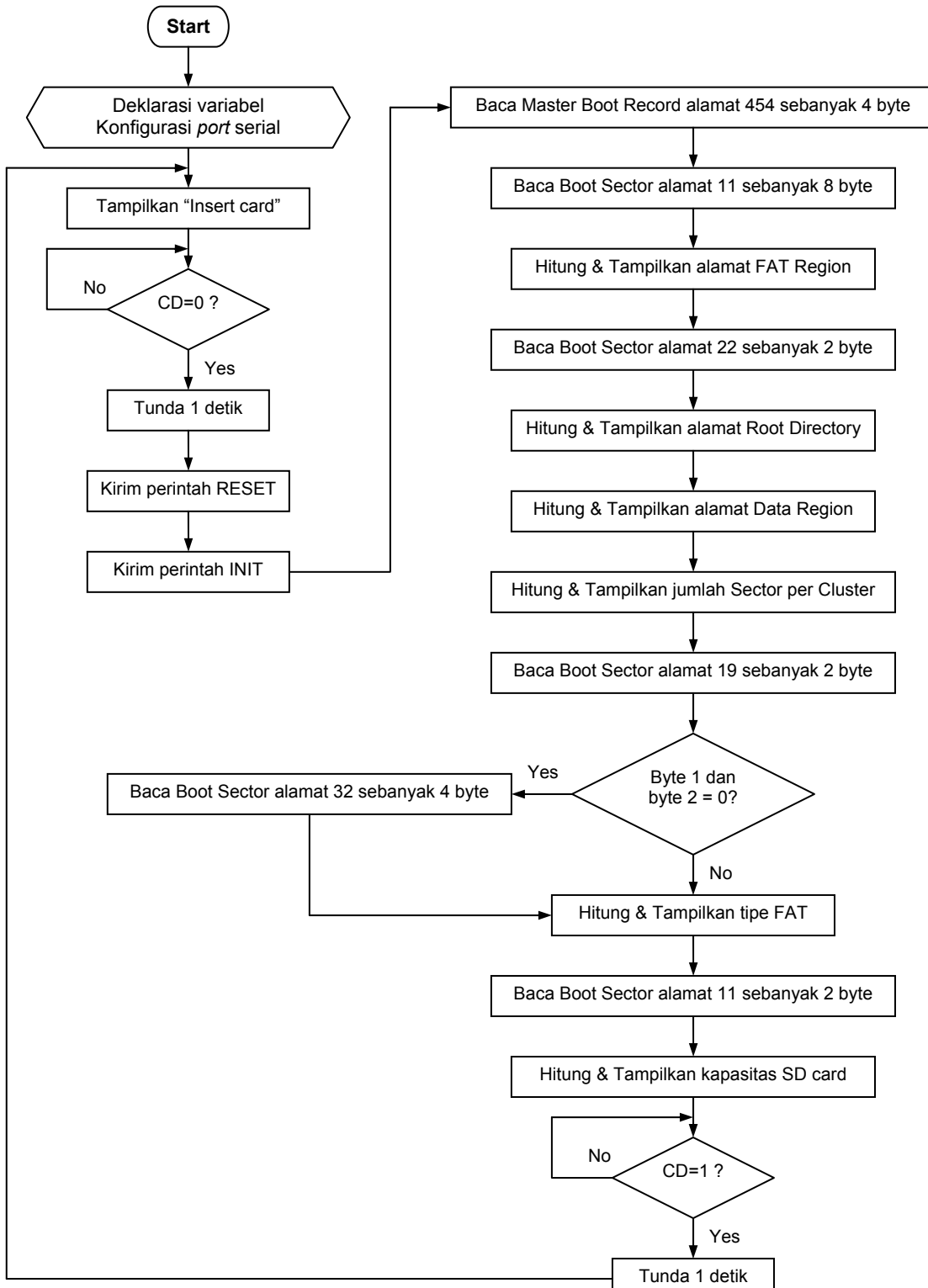
### Hubungan DT-51™ Low Cost Micro System / Low Cost Nano System dengan EMS SD/MMC/FRAM

Jumper J8 & J9 (DT-51™ Low Cost Micro System) atau jumper J3 & J4 (DT-51™ Low Cost Nano System) perlu diatur agar P3.0 dan P3.1 terhubung ke rangkaian UART RS-232 (posisi 1-2). Hubungkan DT-51™ Low Cost Series ke komputer menggunakan kabel serial yang tersedia. Pada komputer, gunakan program Terminal pada BASCOM-8051®, HyperTerminal®, Terminal®, atau program terminal lainnya dengan *baud rate* 9600, 8 bit data, 1

bit stop, tanpa bit *parity*, dan tanpa *flow control*. Khusus DT-51™ Low Cost Nano System, jumper J8 perlu dipasang untuk memberi resistor *pull-up* pada P1.0 dan P1.1. Setelah semua rangkaian dan catu daya terhubung dengan benar, programlah MCSread.HEX ke DT-51™ Low Cost Series menggunakan **DT-HiQ AT89S In System Programmer** atau divais *programmer* lain yang mendukung.

**Jika pemrograman ISP terganggu, coba lepas SD Card atau lepas modul EMS SD/MMC/FRAM atau ubah program dan pindahkan koneksi ke port lain.**

**F**lowchart dari program MCSread.BAS adalah sebagai berikut:



**Gambar 2**  
**Flowchart Program MCSread.BAS**

**C**ara kerja program secara garis besar adalah sebagai berikut:

1. Pertama program melakukan deklarasi variabel yang akan digunakan untuk menampung data dan parameter yang berhubungan dengan SD card. Program juga melakukan konfigurasi port serial pada *baud rate* 9600, 8 bit data, 1 bit stop, tanpa bit *parity*, dan tanpa *flow control*.
2. Program mengirimkan "Insert card" ke komputer lalu menunggu adanya kartu yang dimasukkan ke slot.
3. Setelah ada kartu yang dimasukkan ke slot (CD=0), program akan menunda 1 detik sebelum mengakses kartu. Hal ini bertujuan untuk memastikan bahwa kartu sudah tepat berada di tempatnya saat akan diakses. Nilai waktu 1 detik ini dapat diubah atau dihilangkan jika dirasa tidak diperlukan.
4. Langkah awal sebelum mengakses SD card untuk pertama kalinya adalah mengirimkan perintah Reset dan Init ke SD card.
5. Karena SD card yang digunakan memiliki format FAT16, maka parameter yang harus dibaca disesuaikan dengan format FAT16. Yang pertama harus dilakukan adalah membaca Master Boot Record (berada di sektor 0) agar lokasi Boot Sector dapat diketahui.
6. Lalu Boot Sector dibaca pada alamat-alamat tertentu sehingga didapat parameter antara lain: alamat FAT Region, alamat Root Directory, alamat Data Region, jumlah sector per cluster, tipe FAT, dan kapasitas SD card.
7. Program menunggu kartu dikeluarkan (CD=1).
8. Berikut ini tampilan yang muncul pada program terminal:

```
Insert card
Init - OK
FATregion: 59
RootDir: 81
DataRegion: 113
SectorCluster: 8
Type: FAT16
Size (MB): 30
```

9. Setelah kartu dikeluarkan, program kembali ke langkah nomor 2.

Setelah program MCSread berjalan normal dan menampilkan parameter seperti pada langkah 8, maka sesuaikan program MCSwrite dengan parameter tersebut. Contoh setelah baris program untuk deklarasi konstanta pada MCSwrite.BAS diganti:

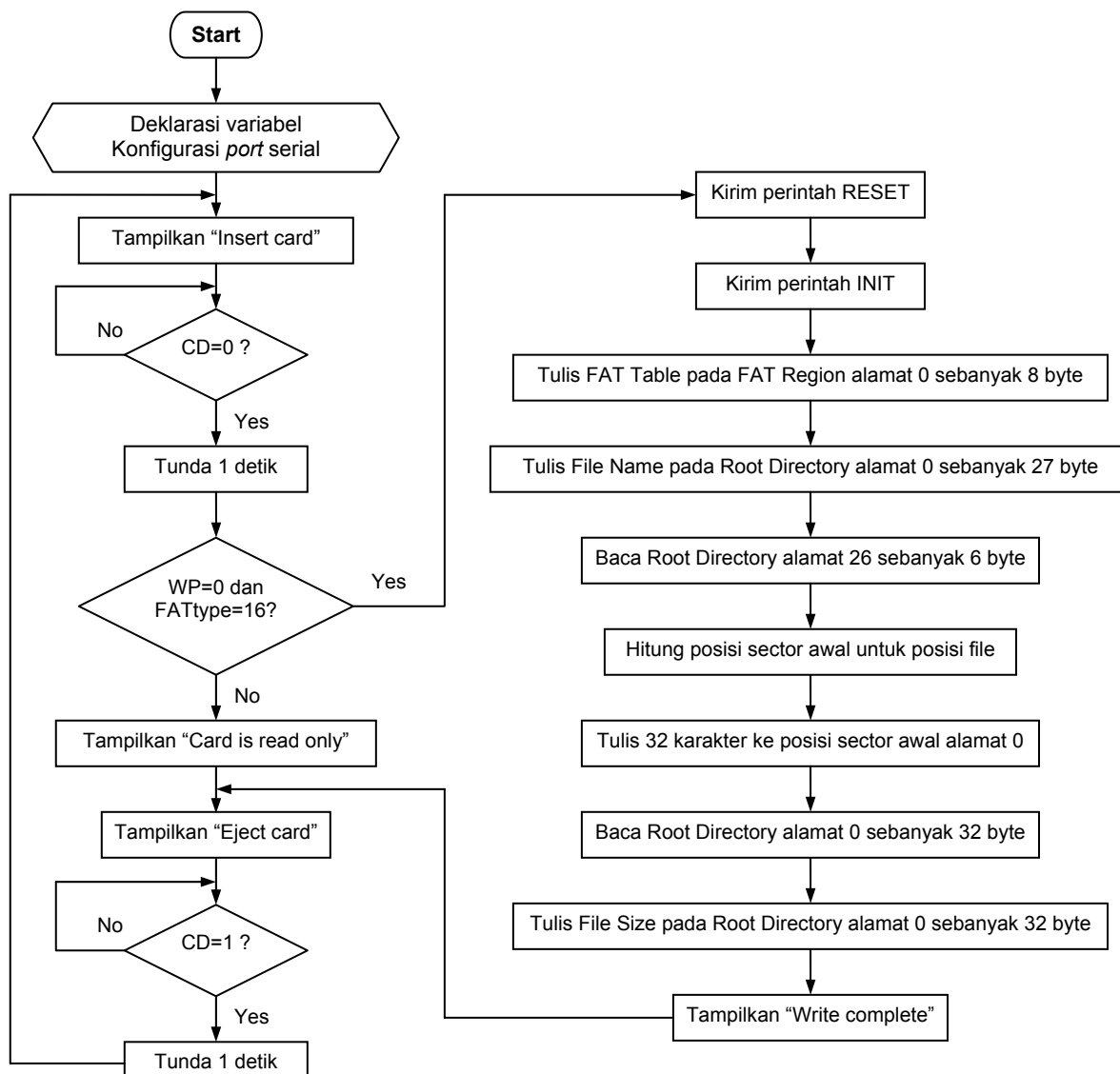
```
Const Fatregion = 59
Const Rootdir = 81
Const Dataregion = 113
Const Sectorcluster = 8
Const Fattype = 16
```

**Periksa dan pastikan bahwa nilai yang dituliskan sama dengan nilai yang telah dibaca oleh program MCSread.**

**Kesalahan atau perbedaan nilai dapat mengakibatkan format FAT16 pada SD card kacau dan tidak dapat diakses oleh card reader, kamera digital, PDA, dsb.**

Setelah itu lakukan *compile* terhadap MCSwrite.BAS dan program ke DT-51™ Low Cost Series.

**F**lowchart dari program MCSwrite.BAS adalah sebagai berikut:



**Gambar 3**  
**Flowchart Program MCSwrite.BAS**

**C**ara kerja program secara garis besar adalah sebagai berikut:

1. Pertama program melakukan deklarasi variabel yang akan digunakan untuk menampung data dan parameter yang berhubungan dengan SD card. Program juga melakukan konfigurasi port serial pada *baud rate* 9600, 8 bit data, 1 bit stop, tanpa bit *parity*, dan tanpa *flow control*.
2. Program mengirimkan "Insert card" ke komputer lalu menunggu adanya kartu yang dimasukkan ke slot (CD=0).
3. Jika kartu yang dimasukkan ke slot dalam posisi terkunci (WP=1), maka program akan menampilkan "Card is read only" dan "Eject card" lalu program menunggu SD card dikeluarkan (CD=1). Setelah SD card dikeluarkan, program kembali ke langkah nomor 2.
4. Jika kartu yang dimasukkan tidak terkunci dan tipe FAT yang dimasukkan adalah FAT16, program akan menunda 1 detik sebelum mengakses kartu. Hal ini bertujuan untuk memastikan bahwa kartu sudah tepat berada di tempatnya saat akan diakses. Nilai waktu 1 detik ini dapat diubah atau dihilangkan jika dirasa tidak diperlukan.
5. Langkah awal sebelum mengakses SD card untuk pertama kalinya adalah mengirimkan perintah Reset dan Init ke SD card.
6. Karena SD card yang digunakan memiliki format FAT16, maka proses menulis file harus disesuaikan dengan format FAT16. Program akan menulis tabel FAT pada FAT Region.
7. Program akan menulis Nama File pada Root Directory.

8. Root Directory akan dibaca untuk mengetahui posisi sector awal untuk file.
9. Isi file sebanyak 32 karakter dituliskan ke posisi sector awal mulai alamat 0.
10. Root Directory dibaca mulai alamat 0 sebanyak 32 byte.
11. Parameter ukuran file diubah lalu dituliskan kembali ke Root Directory mulai alamat 0 sebanyak 32 byte.
12. Program menampilkan pesan "Write Complete" lalu "Eject Card" dan menunggu kartu dikeluarkan.
13. Berikut ini tampilan yang muncul pada program terminal jika SD card tidak terkunci dan proses penulisan berhasil:

```

Insert card
Reset - OK
Init - OK
Writing:
FAT Table...
File Name...
File Contents...
File Size...
Write complete
Eject card

```

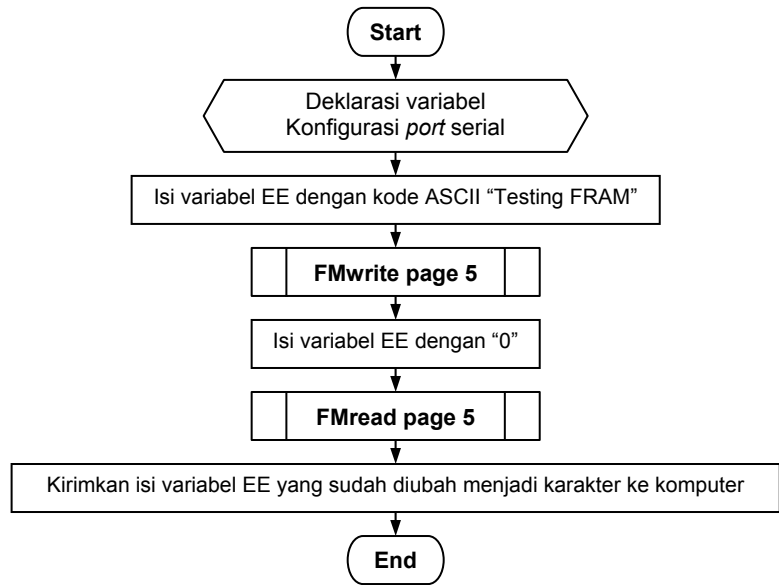
14. Setelah kartu dikeluarkan, program kembali ke langkah nomor 2.

Setelah proses tulis selesai, maka SD card dapat dilepas lalu dibaca pada komputer menggunakan card reader atau pada PDA. Di dalamnya akan terdapat 1 file bernama TESTINGS.TXT sebesar 32 byte yang berisi: "123456789;.<=>?@ABCDEFGHIJKLMN0P"

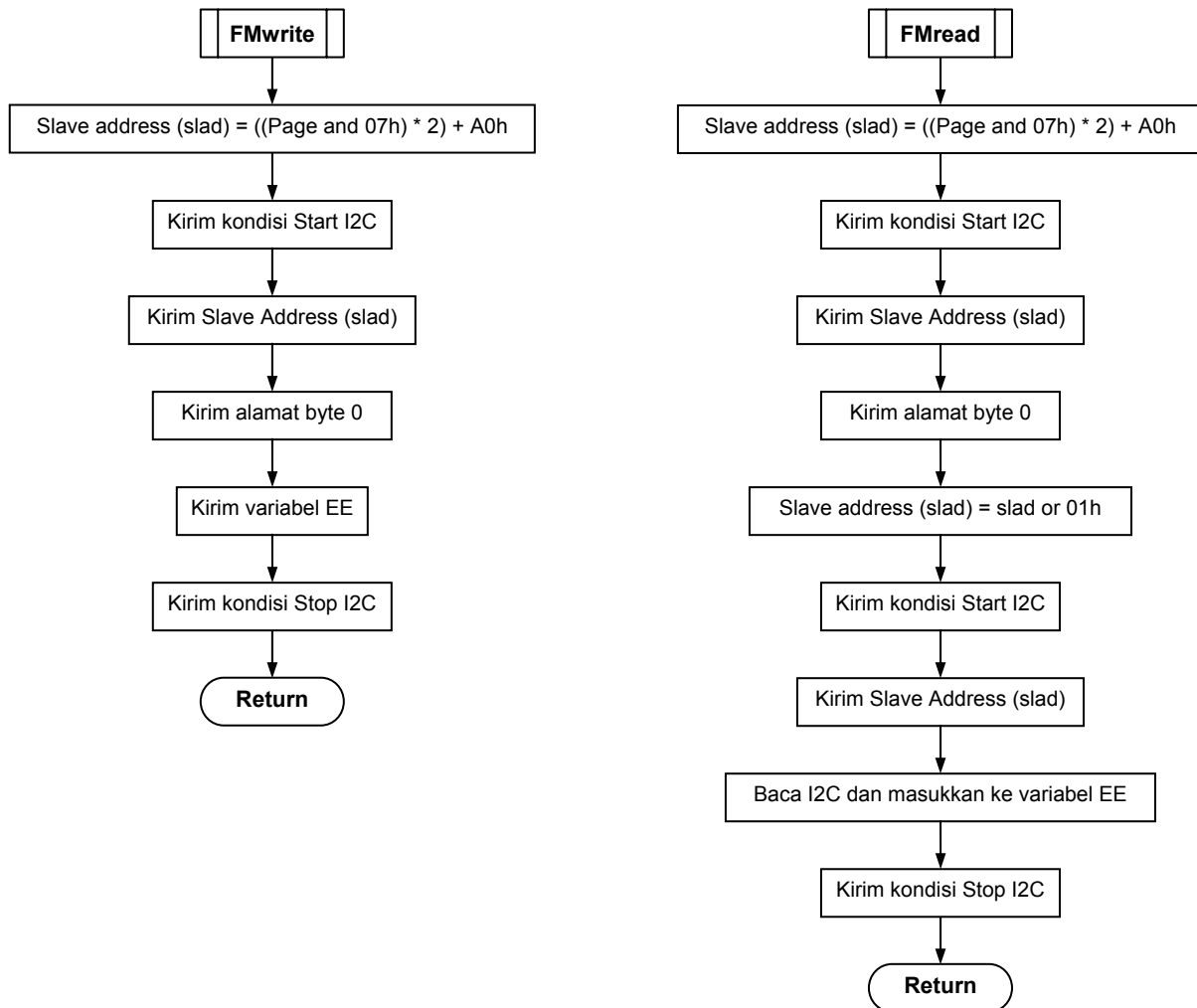
Akses ke SD card membutuhkan *buffer* sebanyak 512 byte. Karena keterbatasan RAM pada AT89S2051 dan AT89S51, terpaksa proses baca dan tulis dibatasi hanya 32 byte sehingga proses menjadi lebih rumit dan beberapa fitur harus dihilangkan. Caranya antara lain dengan hanya membaca/menulis alamat tertentu sebanyak byte tertentu, misalnya hanya mampu menulis File Name tapi tidak mampu menulis Card Name atau hanya membaca Boot Sector alamat tertentu pada satu saat sehingga harus membaca berulang-ulang dengan alamat yang berbeda.

Untuk itu, FRAM pada EMS SD/MMC/FRAM dapat digunakan sebagai *buffer*. Karena BASCOM-8051<sup>®</sup> versi demo membatasi ukuran program 4 KB, maka contoh program untuk akses FRAM diletakkan pada program terpisah. **Contoh penggunaan FRAM sebagai *buffer* dalam akses SD card terdapat pada AN SD Card Module - AVR**

**F**lowchart dari program FRAM.BAS adalah sebagai berikut:



**Gambar 4**  
**Flowchart Program FRAM.BAS**



**Gambar 5**  
**Flowchart Rutin FMread dan FMwrite**

**C**ara kerja program secara garis besar adalah sebagai berikut:

1. Pertama program melakukan deklarasi variabel yang akan digunakan untuk menampung data dari/ke FRAM. Program juga melakukan konfigurasi port serial pada *baud rate* 9600, 8 bit data, 1 bit stop, tanpa bit *parity*, dan tanpa *flow control*.
2. Program mengisi variabel EE dengan string "Testing FRAM". Pada listing program, variabel string ini harus diubah dulu ke kode ASCII agar dapat dimasukkan ke variabel EE.
3. Program akan menuliskan variabel EE ke dalam FRAM pada page 5 mulai alamat 0.
4. Semua isi variabel EE ditulis dengan "0".
5. FRAM page 5 dibaca lalu hasil pembacaan dimasukkan ke variabel EE.
6. Isi variabel EE diubah menjadi karakter lalu dikirimkan ke komputer.
7. Berikut ini tampilan yang muncul pada program terminal jika proses penulisan berhasil:

Testing FRAM

**C**ara kerja rutin FMwrite adalah sebagai berikut:

1. Pertama rutin akan menghitung alamat slave sesuai dengan parameter page (pager) yang diberikan saat memanggil rutin.
2. Lalu rutin akan mengirim kondisi Start I2C yang diikuti dengan alamat slave yang telah dihitung.
3. Rutin selalu menulis mulai dari alamat 0 untuk tiap page sehingga data yang dikirim berikutnya adalah 0.
4. Semua isi variabel EE sebanyak 16 byte dituliskan ke FRAM.
5. Setelah semua data terkirim, rutin mengakhiri dengan mengirim kondisi Stop I2C.

**C**ara kerja rutin FMread adalah sebagai berikut:

1. Pertama rutin akan menghitung alamat slave sesuai dengan parameter page (pager) yang diberikan saat memanggil rutin.
2. Lalu rutin akan mengirim kondisi Start I2C yang diikuti dengan alamat slave yang telah dihitung.
3. Rutin selalu membaca mulai dari alamat 0 untuk tiap page sehingga data yang dikirim berikutnya adalah 0.
4. Lalu kondisi Start I2C akan dikirim yang diikuti dengan alamat slave untuk proses baca (bit 0 = 1).
5. Data sebanyak 16 byte pada page FRAM tersebut akan dibaca dan disimpan ke dalam variabel EE.
6. Setelah semua data dibaca, rutin mengakhiri dengan mengirim kondisi Stop I2C.

**L**isting program terdapat pada *folder* **BAS51**.

**S**elamat berinovasi!

MCS-51 is a registered trademark of Intel Corporation.  
DT-51 is a trademark of Innovative Electronics.  
BASCOM-8051 is copyright by MCS Electronics.  
HyperTerminal is a copyright by Hilgraeve Inc.  
Terminal is a copyright by Bray++.