PERCOBAAN 6

INSTRUKSI PUTAR, GESER, DAN *ROUTINE* PERKALIAN

Oleh : Sumarna, Jurdik Fisika, FMIPA, UNY
E-mail : **sumarna@uny.ac.id**

Tujuan dari percobaan ini adalah untuk memberikan pengertian dan penggunaan mengenai perintah-perintah putar (rotate), serta geser (shift). Selain itu, agar memahami teknik-teknik merancang program yang menggunakan routine perkalian biner.

Data 9 bit (yang dibentuk oleh data 8 bit yang ada dalam register atau memori dan carry flag) dapat digeser 1 (satu) bit ke kiri atau ke kanan dengan perintah ROTATE (putar) atau SHIFT (geser). Karena terdapat banyak cara untuk menggeser bit data yang paling kanan atau paling kiri dengan arah ke luar atau ke dalam, maka terdapat banyak perintah dalam kelompok ROTATE atau SHIFT (ada 13 perintah yang berbeda), yaitu RLCA, RLA, RRCA, RRA, RLD, RRD, RLC s, RL s, RRC s, RR s, SLA s, SRA s, dan SRL s (dengan s = register atau lokasi di dalam memori). Mnemonik perintah-perintah tersebut mempunyai ciri sebagai berikut :

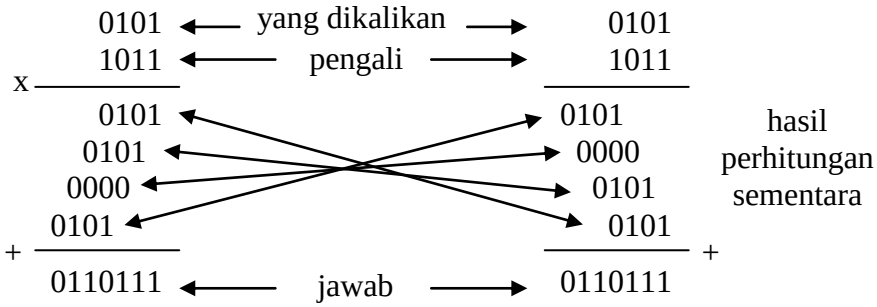
1. Jika huruf yang paling kiri perintah tersebut adalah “R”, maka perintah tersebut adalah ROTATE. Perintah ini dipakai untuk memutar data 9 bit (terdiri dari data 8 bit dan carry flag) ke kiri atau ke kanan 1 bit. Dalam perintah ROTATE **tidak ada** bit yang ke luar dari data.
2. Jika huruf yang paling kiri perintah tersebut adalah “S”, maka perintah tersebut adalah SHIFT. Perintah ini dipakai untuk menggeser semua bit data ke kiri atai ke kanan sejauh 1 bit. Bit yang telah digeser dan keluar dari satu sisi tidak akan masuk melalui sisi yang lain.
3. Jika huruf ke dua dari kiri adalah “R” berarti SHIFT ke kanan atau ROTATE ke kanan, misalnya RR, SRL, dan RRCA.
4. Jika huruf ke dua dari kiri adalah “L” berarti SHIFT ke kiri atau ROTATE ke kiri, misalnya RL, SLA, dan RLCA.
5. Arti huruf ke tiga dapat diringkas sebagai berikut :

a. Pada perintah ROTATE, huruf ke tiga C menyatakan rotasi pada data 8 bit tidak termasuk carry flag. Huruf ke tiga (atau huruf ke empat) “A” berarti perintah tersebut dioperasikan dengan akumulator, misalnya RLA, RRA, RLCA, dan RRCA. Huruf ke tiga “D” berarti operasi SHIFT pada bilangan desimal atau heksadesimal, misalnya RLD atau RRD. Perintah-perintah ini dipakai untuk merotasikan data pada memori yang ditunjuk oleh HL ke kiri atau ke kanan 1 (satu) digit (4 bit). Digit yang masuk dari arah kiri atau kanan berasal dari akumulator bit 0 s/d bit 3. Sedangkan digit yang keluar dari sisi yang lain dikirim k eakumulator bit 0 s/d bit 3.

b. Pada perintah SHIFT, huruf ke tiga “A” menyatakan “arithmetic shift”. Data biner yang digeser ke kiri berarti mengalikannya dengan 2, sedangkan data biner yang digeser ke kanan berarti membaginya dengan 2. Perintah yang termasuk kelompok ini adalah SLA dan SRA. Karena bit 7 dipakai sebagai “bit tanda” dan tanda data tidak beribah dalam operasi ini, maka bit paling kiti (bit 7) tidak berubah. Huruf ke tiga “L” berarti “logical shift”. Misalnya perintah SRL. Pada operasi ini, status “0” (nol) selalu dipindahkan ke bit 7 dari arah kiri.

Perkalian Biner :

Operasi perkalian bilangan biner tak bertanda dapat dilakukan dengan cara menggeser bilangan biner ke kiri atau dengan loop program penjumlahan (penjumlahan yang berulang). Contoh perkalian biner adalah sebagai berikut :



Dengan mencermati ilustrasi di atas terlihat bahwa satu bit bilangan pengali diperiksa. Jika bit tersebut adalah 1, maka bilangan yang dikalikan ditulis sama. Jika bit tersebut adalah 0, maka yang dituliskan adalah 0000. Posisi bialangan-bilangan yang ditulis (disimpan) tersebut diatur sedemikian rupa sehingga bit berorde paling kecil dari bilangan yang dikalilkan berada dalam satu garis dengan bit bilangan pengali yang sedang diperiksa. Pada contoh di atas baik bialngan yang dikalikan maupun bilangan pengali adalah data 4 (empat) bit. Jadi operasi pemeriksaan, penggeseran dan penambahan diulang sebanyak 4 kali. Untuk perkalian data 8 bit, operasi-operasi tersebut harus diulang 8 kali. Demikian pula untuk perklaian 16 bit, operasi harus diulang 16 kali. Pada contoh sebelah kiri, proses pemeriksaaan bit dimulai dari bit bilangan pengali yang berorde rendah. Pada contoh sebelah kanan, pemeriksaan bit dimulai dari bit berorde tinggi. Tetapi hasil perkalian dari kedua contoh di atas adalah sama.

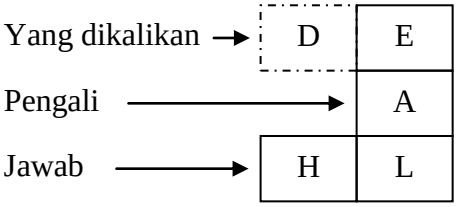
Percobaan 6.1 :

Kalikan data 8 bit pada register E dengan data 8 bit pada register A dan hasilnya disimpan pada pasangan register HL. Dengan menggunakan algoritma perhitungan pada contoh sebelah kanan, program dirancang sebagai berikut :

1. Proses pemeriksaan bit dimulai dari bit berorde tinggi. Bilangan pengali adalah 8 bit, sehingga jumlah loop sama dengan 8. Pada setiap pelaksanaan loop bit yang diperiksa dapat digeser ke carry flag dengan perintah RLCA. Kemudian sesuai dengan status carry flag dapat menentukan apa yang akan dikerjakan berikutnya.
2. Jika bit pertama yang diperiksa, hasilnya didapat dengan menggeser bilangan yang dikalikan ke kiri sebanyak (n – 1) bit, di mana n adalah jumlah bit pada bilangan pengali. Hasil berikutnya didapat dengan menggeser hasil perhitungan sementaranya ke kiri (n – 2) bit, (n – 3) bit, dan seterusnya. Pada contoh ini tidak diperlukan register lain untuk menyimpan hasil perhitungan sementaranya. Setiap hasil perhitungan sementara dapat dijumlahkan langsung ke pasangan register HL.

3. Dari uraian di atas, kita dapat melihat bahwa hasil perhitungan sementara harus digeser ke kiri (n – 1) bit, (n – 2) bit, (n – 3) bit, dan seterusnya. Karena pemeriksaan bit juga bergeser ke kiri dalam proses ini, kita bisa mendapatkan hasil perhitungan sementara baru dengan menambahkan tiap-tiap hasil perhitungan sementara ke hasil perhitungan sementara yang sebelumnya. Metode ini sangat efisien dan digunakan pada diagram alir (flowchart) program berikut.

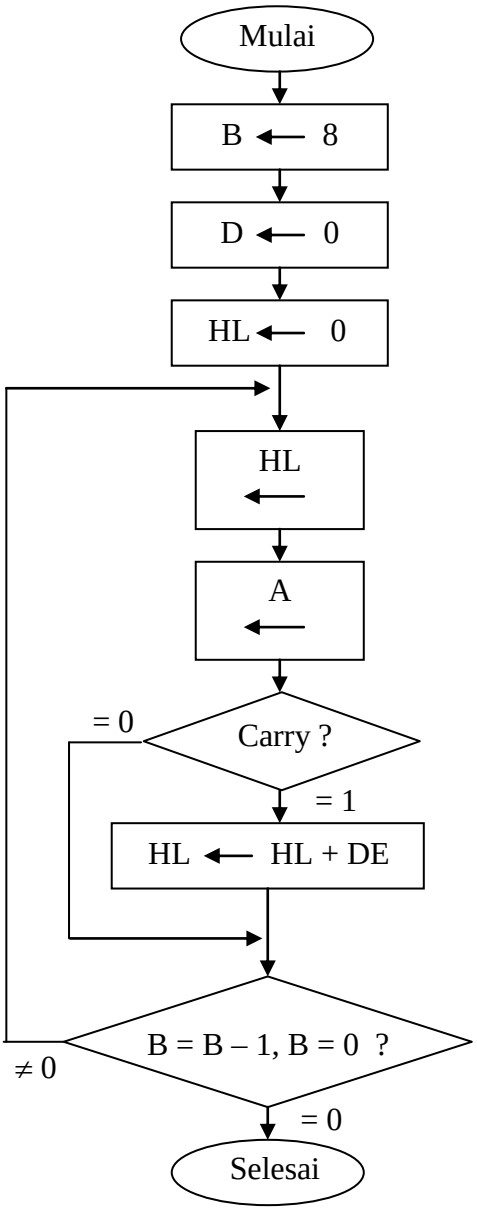
4. Program dan Diagram alir :



Bilangan pengali di E
Bilangan yang dikalikan di A
Hasil perhitungan di HL
Register yang berubah B, D, HL, F.

```
                Org     1800h

                LD      B,8
                LD      D,0
                LD      H,D
                LD      L,D
Loop :          ADD     HL,HL
                RLCA
                JRNC    Nadd
                ADD     HL,DE
Nadd :          DJNZ    Loop
                RET
```



Percobaan 6.2 :

Rancanglah program yang dapat digunakan untuk menggeser data 32 bit yang disimpan pada pasangan register-register HL dan DE yang saling berdekatan ke kanan 1 bit (membagi data dengan 2). Masukkan program tersebut ke MPF-1 dan pelajari hasilnya ! Cocokkan hasilnya dengan perhitungan secara manual !. Misalkan sebelum program dijalankan, isilah register HL dengan 0304h dan DE dengan 0102h. Setelah program dijalankan, apakah isi register HL = 0182h dan DE = 0081h ?. Perhatikan ilustrasi program berikut :

```
Org   1800h
SRA   H
RR    L
RR    D
RR    R
RST   38h
```

Kemudian ubahlah program itu sehingga dapat dipakai untuk menggeser data 32 bit ke kiri (mengalikannya dengan 2).

Percobaan 6.3 :

Rancangalah sebuah program untuk menggeser data 32 bit yang tersimpan di RAM pada alamat 1A00h s/d 1A03h, ke kiri 5 bit (atau mengalikannya dengan 20h). Masukkan program ke MPF-1 dan pelajari hasilnya. Tetapkanlah alamat 1810h sebagai alamat awal program ini. Cocokkan dengan hasil yang dikerjakan secara manual.

Percobaan 6.4 :

Dengan menggunakan perintah RLD, rancangalah sebuah program untuk menggeser data BCD yang tersimpan di RAM pada alamat 1A00h s/d 1A03h, ke kiri 4 bit. Masukkan program ke MPF-1 dan pelajari hasilnya. Gunakan alamat sebagai awal program adalah 1830h. Cocokan dengan hasil yang dikerjakan secara manual.

Percobaan 6.5 :

Program berikut ini dapat digunakan untuk mengalikan data 16 bit yang tersimpan pada pasangan register DE dengan isi register A. Masukkan program itu ke MPF-1 dan pelajari hasilnya. Cocokkan dengan hasil yang dikerjakan secara manual. Bandingkan dengan program berikut :

```
        Org     1800h

        LD      BC,800h
        LD      H,C
        LD      L,C
Loop :  ADD     HL,HL
        RLA
        JRNC    Nadd
        ADD     HL,DE
        ADC     A,C
Nadd :  DJNZ    Loop
        Rst     38h
```

Bandingkan program tersebut dengan program pada percobaan 6.1 di atas. Diskusikan kelebihan dan kekurangannya !.

Percobaan 6.6 :

Rancangalah sebuah program untuk mengalikan data 32 bit yang tersimpan di RAM pada alamat 1A00h s/d 1A03h dengan data 32 bit yang tersimpan di RAM pada alamat 1A04h s/d 1A07h. Hasilnya disimpan di RAM pada alamat 1A08h s/d 1A0Fh !. Masukkan program itu ke MPF-1 dan pelajari hasilnya. Cocokkan dengan hasil yang dikerjakan secara manual.