

Arithmatika Komputer

Pertemuan - 2

Oleh :

Riyanto Sigit, S.T, M.Kom

Nur Rosyid Mubtada'i S.Kom

Setiawardhana , S.T

Hero Yudo Martono, S.T

Politeknik Elektronika Negeri Surabaya - ITS

2005

? Mengapa belajar Arithmatika

- Mengerti bagian-bagin ALU
- Memahami representasi Integer
- Memahami cara operasi penambahan, pengurangan, perkalian dan pembagian dengan representasi Interger
- Memahami representasi Floating point
- Memahami cara penambahan, pengurangan, perkalian dan pembagian dengan representasi Floating Point

Kode Biner

- Data huruf akan dirubah menjadi kode ASCII
- Dri kode ASCII dirubah menjadi bilangan biner.
- Data gambar merupakan kumpulan dari angka-angka yang merupakan perwakilan dari warna masing-masing titik / pixel, dan angka tersebut yang akan dirubah dalam bentuk biner.
- Semua data direpresentsikan/dituliskan dalam bentuk 0 dan 1

? Proses dikodekan dalam Biner

- Sebagian besar operasi yang ada di dalam proses komputer adalah operasi aritmatika.
- Operasi aritmatika Apa saja ?
 - Penambahan
 - Pengurangan
 - Perkalian
 - Pembagian.

? Data yang bagaimana yang dioperasikan

- Adalah data yang berupa data angka.
- Data angka digolongkan menjadi
 - data bilangan bulat / integer
 - Data bilangan pecahan/float
- Pada bab ini akan dipelajari (?)
 - Data interger dan float di representasikan didalam bentuk biner
 - Cara agar data tersebut bisa di operasikan secara aritmatik

? Belajar ALU

Semua operasi aritmatik dilakukan oleh

ALU

ALU (Aritmatic Logic Unit)

- Merupakan bagian CPU yang berfungsi membentuk operasi - operasi aritmatika dan logika terhadap data.
- Semua proses ada disini ? .

? Semua Proses disini

- Semua komponen CPU lainnya dan komponen penyusun komputer secara keseluruhan berfungsi
 - Membawa data ke ALU untuk diproses
 - Mengambil lagi hasil proses dari ALU

Representasi Proses

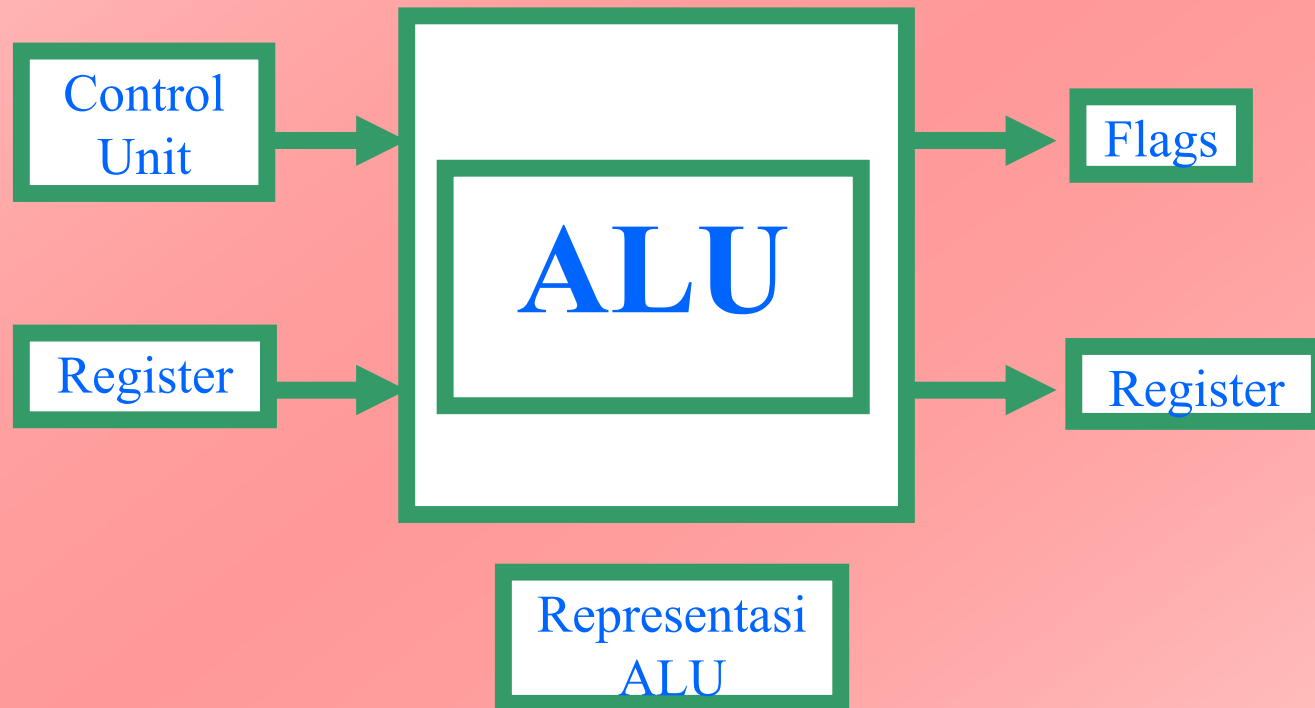
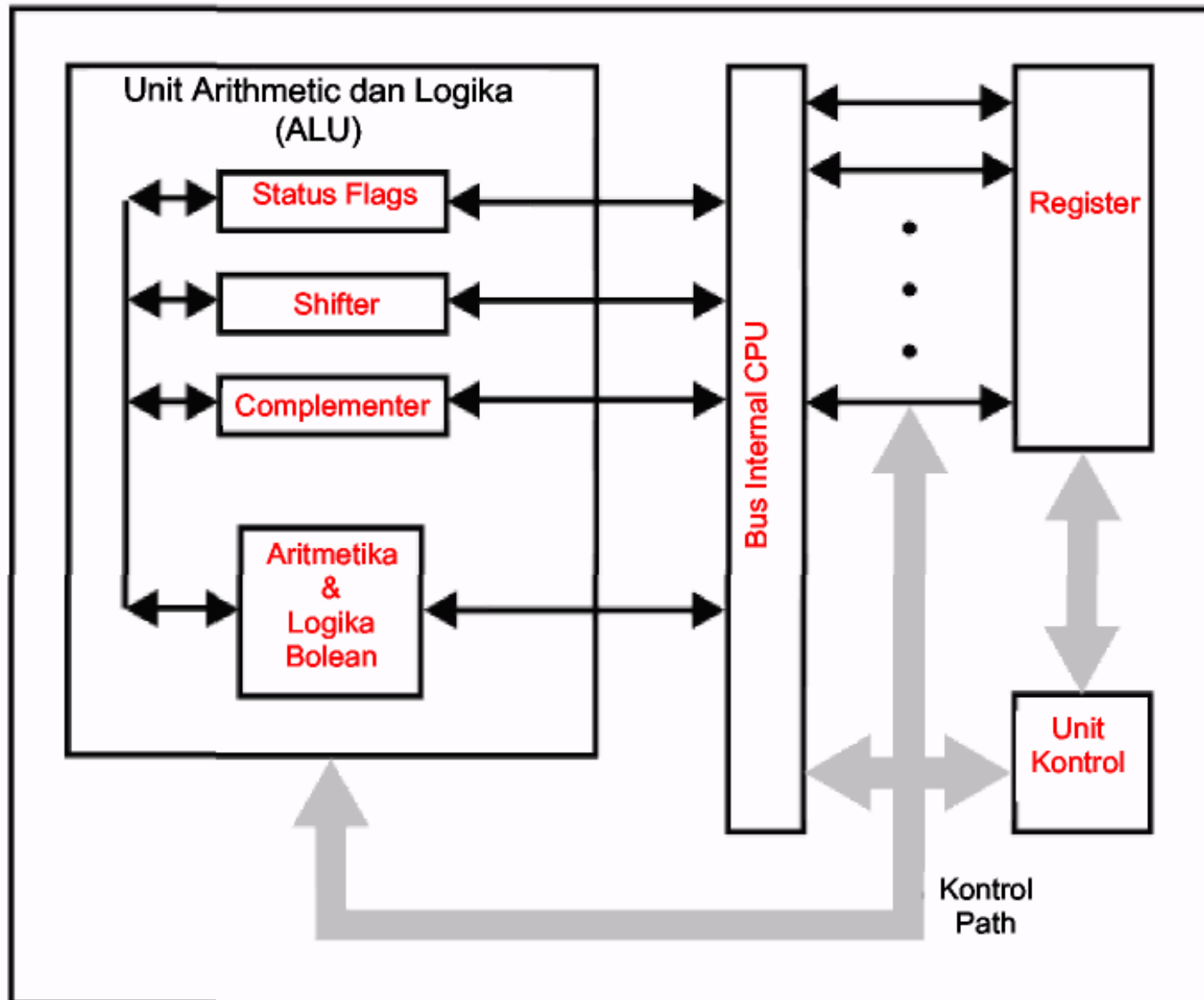


Diagram penyusun CPU dengan ALU di dalamnya



? Penjelasan Hubungan

- Hubungan interkoneksi ALU dengan
 - Register
 - Unit kontrol
 - Flags
- ❖ Kesemuanya melalui bus internal CPU

Register ? Flag ? Unit Kontrol ?

- *Register* adalah tempat penyimpanan data sementara dalam CPU selama proses eksekusi. Apabila terjadi proses eksekusi data dalam register dikirim ke ALU untuk diproses, hasil eksekusi nantinya diletakkan ke register kembali.
- *Flag* diset ALU sebagai hasil dari suatu operasi, misalnya: overflow flag, diset 1 bila hasil komputasi melampaui panjang register tempat flag disimpan.
- *Unit kontrol* akan menghasilkan sinyal yang akan mengontrol operasi ALU dan pemindahan data ke dan dari ALU

! Ayo kita mulai belajar Aritmatika

? Representasi Integer

- Sistem bilangan dengan radix yang berbeda
 - Biner
 - Oktat
 - Desimal
 - heksadesimal
- Topik : Biner dan operasi Aritmetikanya

Ada alasan mendasar kenapa bilangan biner dipilih untuk mekanisme representasi data komputer

?

? Jawabnya

- Komputer secara elektronika hanya mampu membaca dua kondisi sinyal
 - Ada sinyal atau ada tegangan
 - Tidak ada sinyal atau tidak ada arus listrik yang mengalir.
- Dua kondisi tersebut yang digunakan untuk merepresentasi bilangan dan kode - kode biner
 - Level tinggi (ada tegangan) sebagai representasi bilangan 1
 - Level rendah (tidak ada arus) sebagai representasi bilangan 0

? Representasi “Integer” oleh Biner

- Dalam sistem bilangan biner terdapat

empat macam sistem untuk merepresentasikan integer

? Representasi “Integer” oleh Biner

- Representasi unsigned interger
- Representasi nilai tanda (sign-magnitude).
- Representasi bias
- Representasi komplemen dua (two's complement)

! Mari kita Bahas satu persatu

Representasi

Biner terhadap Integer

Representasi Unsigned Integer

- Untuk keperluan penyimpanan dan pengolahan komputer diperlukan bilangan biner yang terdiri atas bilangan 0 dan 1.
- Suatu word 8 bit dapat digunakan untuk menyatakan bilangan desimal 0 hingga 255
- Contoh :
 - $0000\ 0000_2 = 0_{10}$
 - $0000\ 0001_2 = 1_{10}$
 - $1000\ 0000_2 = 128_{10}$
 - $1111\ 1111_2 = 255_{10}$

Formula - Representasi Unsigned Integer

- Formulasi umum dalam unsigned integer N adalah :

$$N = \sum_{i=0}^{n-1} 2^i a_i$$

a = bit ke i

➤ Kelemahan :

- Hanya dapat menyatakan bilangan positif saja.
- Sistem ini tidak bisa digunakan untuk menyatakan bilangan integer negatif

Representasi Nilai Tanda

- Berangkat dari kelemahan metode unsigned integer.
- Dikembangkan beberapa konvensi untuk menyatakan bilangan integer negatif
- Konvensi yang bagaimana ?

Konvensi - Representasi Nilai Tanda

- Perlakuan bit paling berarti (paling kiri) di dalam word sebagai bit tanda.
- Bila bit paling kiri adalah 0 maka bilangan tersebut positif
- Bila bit paling kiri adalah 1 maka bilangan tersebut negatif

Konvensi - Representasi Nilai Tanda

Contoh:

$$+21_{10} = 0\ 0010101_2$$

$$-21_{10} = 1\ 0010101_2$$

Formula - Representasi Nilai Tanda

- Formulasi umum dalam signed integer N

$$N = \sum_{i=0}^{n-2} 2^i a_i$$

Jika a ke n-1 sama dengan 0

Dan

$$N = - \sum_{i=0}^{n-2} 2^i a_i$$

Jika a ke n-1 sama dengan 1

Representasi Nilai Tanda

- Apa punya kelemahan ?
- Jawabnya : **YA**

? Kelemahan

- Masalah pada operasi aritmetika penjumlahan dan pengurangan yang memerlukan pertimbangan tanda maupun nilai bilangan
- Adanya representasi nilai ganda pada bilangan 0
 - $00000000_2 = 0_{10}$
 - $10000000_2 = 0_{10}$

Representasi bias

- Digunakan untuk menyatakan eksponen (bilangan pemangkat) pada representasi bilangan pecahan
 - dapat menyatakan bilangan bertanda, yaitu dengan mengurutkan bilangan negatif paling kecil yang dapat dijangkau sampai bilangan positif paling besar yang dapat dijangkau
- ❖ **Mengatasi permasalahan pada bilangan bertanda yaitu +0 dan -0**

Representasi bias

⊕ Contoh :

- $127_{10} = 11111111_2$

- $1_{10} = 10000000_2$

- $0_{10} = 01111111_2$

- $-1_{10} = 01111110_2$

- $-128_{10} = 00000000_2$

Formula - Representasi bias

- Formulasi umum dalam biased integer N

$$N = \sum_{i=0}^n 2^i a_i - B$$

- Jika menggunakan bilangan bias 8 bit maka b akan bernilai 127, nilai ini didapat 2 dipangkatkan dengan dari n jumlah bit dikurangi 1 hasilnya dikurangkan dengan 1

$$B = 2^{(n-1)} - 1$$

Representasi komplement 2 (two's complement)

- Merupakan perbaikan metode Nilai Tanda yang memiliki kekurangan pada operasi penjumlahan dan pengurangan serta representasi bilangan nol
- Bagaimana Sistemnya ?

Bilangan Negatif – 2's Comp

- Sistem bilangan dalam Komplemen Dua menggunakan bit paling berarti (paling kiri) sebagai bit tanda dan sisanya sebagai bit nilai seperti pada metode Nilai Tanda
- Berbeda untuk representasi bilangan negatifnya.
- Apa Perbedaannya ?
- Bilangan negatif dalam metode komplemen dua dibentuk dari
 1. komplemen satu dari bilangan biner semula (yang bertanda positif)
 2. Menambahkan 1 pada LSB-nya
 3. Diperolehlah bilangan negatifnya

Representasi komplement 2 (two's complement)

Contoh :

$$+21_{10} = 0001\ 0101_2$$

Bilangan negatifnya dibentuk dengan cara:

$$+21_{10} = 0001\ 0101_2$$

dibalik menjadi

$$= 1110\ 1010_2$$

ditambah dengan

1

menjadi

$$= 1110\ 1011_2 = -21_{10}$$

Formula - Representasi komplemen 2

- Formulasi umum dalam 2's komplement integer N

$$N = -2^{n-1} a_{n-1} + \sum_{i=0}^{n-2} 2^i a_i$$

- Untuk mengetahui nilai dalam sistem Komplemen Dua dengan cara seperti berikut:

Contoh – 2's Complement

- Menghitung bilangan 2's Complement 8 bit

-128	64	32	16	8	4	2	1

Contoh - 2's Complement

Misalkan bilangan 1010 1010 adalah

-128	64	32	16	8	4	2	1
1	0	1	0	1	0	1	0

$$\begin{aligned} &= -128*1+64*0+32*1+16*0+8*1+4*0+2*1+1*0 \\ &= -128+32+8+2 = - 86 \end{aligned}$$

2's Complement

- *Konversi Panjang Bit Berlainan :*

- Dalam metode Nilai Tanda dapat dilakukan seperti dibawah ini :

$$+3 = 0011 \quad (4 \text{ bit}) \quad -3 = 1011 \quad (4 \text{ bit})$$

$$+3 = 00000011 \quad (8 \text{ bit}) \quad -3 = 10000011 \quad (8 \text{ bit})$$

- Prosedur diatas tidak berlaku untuk integer negatif dalam Komplemen Dua.

- Dalam metode Komplemen Dua berlaku aturan:

- Pindahkan bit tanda ke posisi paling kiri yang baru

- Dan mengisinya dengan salinan - salinan bit tanda.

- Untuk bilangan positif diisi dengan 0

- untuk bilangan negatif diisi dengan 1.

Contoh:

$$+3 = 0011 \quad (4 \text{ bit}) \quad -3 = 1101 \quad (4 \text{ bit})$$

$$+3 = 00000011 \quad (8 \text{ bit}) \quad -3 = 11111101 \quad (8 \text{ bit})$$

2's Complement dan Bias

Desimal	Nilai-Tanda	Komplemen dua Dua	Bias
+7	0111	0111	1111
+6	0110	0110	1110
+5	0101	0101	1101
+4	0100	0100	1100
+3	0011	0011	1011
+2	0010	0010	1010
+1	0001	0001	1001
+0	0000	0000	1000
-0	1000	---	---
-1	1001	1111	0111
-2	1010	1110	0110
-3	1011	1101	0101
-4	1100	1100	0100
-5	1101	1011	0011
-6	1110	1010	0010
-7	1111	1001	0001
-8	---	1000	0000

Penjumlahan dan Pengurangan

- Penambahan pada complement dua ditunjukkan pada contoh slide berikutnya.
- Empat contoh pertama menunjukkan operasi yang berhasil.
- Bila hasil dari operasi adalah positif, kita akan mendapatkan bilangan positif dalam notasi biner yang asli.
- Bila hasil adalah negatif, kita akan mendapatkan bilangan negatif dalam bentuk komplek dua.
- Perlu di perhatikan bahwa, dalam keadaan tertentu, terdapat carry bit setengah ujung word.

Kemudian bit ini akan di abaikan

2.3. Aritmetika Integer

- Membahas operasi aritmetika (Sistem Komplemen Dua)
 - Penjumlahan
 - Pengurangan
 - Perkalian
 - Pembagian

Penjumlahan dan Pengurangan

a.	$(-6) + (+3)$			b.	$(+7) + (-7)$	
	1010	(-6)			0111	(+7)
	<u>0011</u>	(3)			<u>1001</u>	(-7)
	1101	(-3)			10000	(0)
c.	$(+2) + (+3)$			d.	$(-1) + (-5)$	
	0010	(+2)			1111	(-1)
	<u>0011</u>	(+3)			<u>1011</u>	(-5)
	0101	(+5)			1010	(-6)
e.	$(+6) + (+3)$			f.	$(-3) + (-6)$	
	0110	(+6)			1101	(-3)
	<u>0011</u>	(+3)			<u>1010</u>	(-6)
	1001	<i>overflow</i>			0111	<i>overflow</i>

Penjumlahan dan Pengurangan

- Pada sembarang keadaan, hasil operasi dapat lebih besar dari yang dapat di tampung ukuran word yang di gunakan.
- Overflow.
- Bila terjadi overflow, ALU harus membersihkan sinyal tentang keadaan ini sehingga tidak terdapat usaha untuk menggunakan hasil operasi tersebut

Overflow

- Untuk mendeteksi overflow gunakan aturan :
 - *Bila dua buah bilangan di tambahkan, dan keduanya positif atau keduanya negatif, maka overflow akan terjadi bila dan hanya bila memiliki tanda yang berlawanan*

ADDER

- Pada proses penambahan yang ada di ALU diselesaikan dengan *switch* elektronik.
- Pertambahan dari dua buah digit binari (*binary digit* atau *bit*) dilakukan oleh elemen ALU yang disebut *adder*

? Jenis Adder

- Half Adder ?

- Full Adder ?

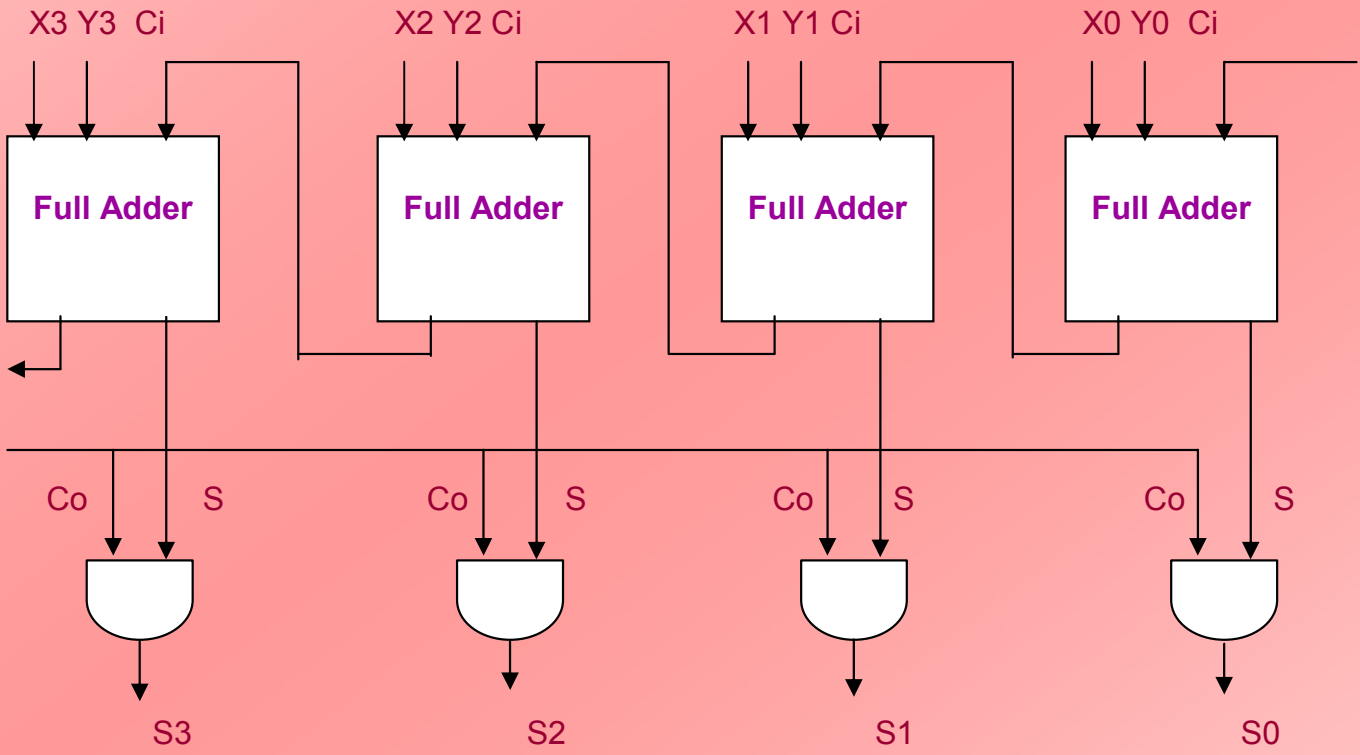
? Half Adder

- Fungsi dari *half adder* adalah menambahkan dua buah *binary digit* dengan hasil berupa pertambahan dan sebuah *carry of*.
- **Input** ada **2** macam yaitu X dan Y sedangkan outputnya berupa *Sum* dan *Carry of*
- Pada *half adder* hasil *carry of* **tidak ikut ditambahkan** pada perhitungan selanjutnya

? Full Adder

- Fungsi dari *full adder* adalah menambahkan dua buah *binary digit* serta *carry of* dari perhitungan sebelumnya dengan hasil berupa pertambahan dan sebuah *carry of*.
- **Input** ada **3** macam yaitu X, Y dan Ci (*carry of input* yang dihasilkan oleh pertambahan sebelumnya) sedangkan outputnya berupa *Sum* dan *Carry of output*
- Pada *full adder* hasil *carry of* **ikut ditambahkan** pada perhitungan selanjutnya

4-bit parallel binary adder menggunakan Full Adder



Penjelasan :

- Input terdiri dari bilangan binari 4 bit, yaitu yang pertama X_3, X_2, X_1 dan X_0 dan yang kedua adalah Y_3, Y_2, Y_1 dan Y_0 .
- Contoh, dua buah bilangan binari 4 bit, yang pertama adalah 1001 dan yang kedua adalah 0101

$$X_3 = 1 ; X_2 = 0 ; X_1 = 0 ; X_0 = 1$$

$$Y_3 = 0 ; Y_2 = 1 ; Y_1 = 0 ; Y_0 = 1$$

Proses Penambahan

- Proses pertambahan dimulai dari digit yang paling kanan
- Bagaimana Prosesnya ?

Urutan proses :

1. X_0 dan Y_0 , yang masing – masing bernilai 1, maka hasil pertambahan kedua bit tersebut adalah 0 dengan carry of output 1 dan carry of tersebut akan ditambahkan sebagai input (carry of input) untuk full adder berikutnya
2. X_1 bernilai 0 dan Y_1 bernilai 0 dan carry of input bernilai 1, maka hasil pertambahan adalah 1 dengan carry of output bernilai 0 untuk full adder berikutnya, yaitu bit X_2 dan Y_2
3. X_2 bernilai 0 dan Y_2 bernilai 1 dan carry of input bernilai 0, maka hasil pertambahan adalah 1 dengan carry of output bernilai 0 untuk full adder berikutnya, yaitu bit X_3 dan Y_3
4. X_3 bernilai 1 dan Y_3 bernilai 0 dan carry of input bernilai 0, maka hasil pertambahan adalah 1 dengan carry of output bernilai 0
5. Hasil akhir dari pertambahan adalah $S_3 = 1$; $S_2 = 1$; $S_1 = 1$ dan $S_0 = 0$ yaitu bilangan binari 1110

Proses Pengurangan

- Proses pengurangan dapat digunakan mesin penambahan, yaitu dengan mengasumsikan bahwa:

$$A-B = A+(-B)$$

? Cara mendapatkan Bil (-)

1. Ubahlah bit - bit menjadi komplemen satu, termasuk bit tandanya
2. Perlakukan hasil pengubahan komplemen satu sebagai unsign binary integer kemudian tambahkan 1 pada LSB-nya

Misalnya:

$$0101 = 5$$

di balik menjadi 1010

jika ditambah

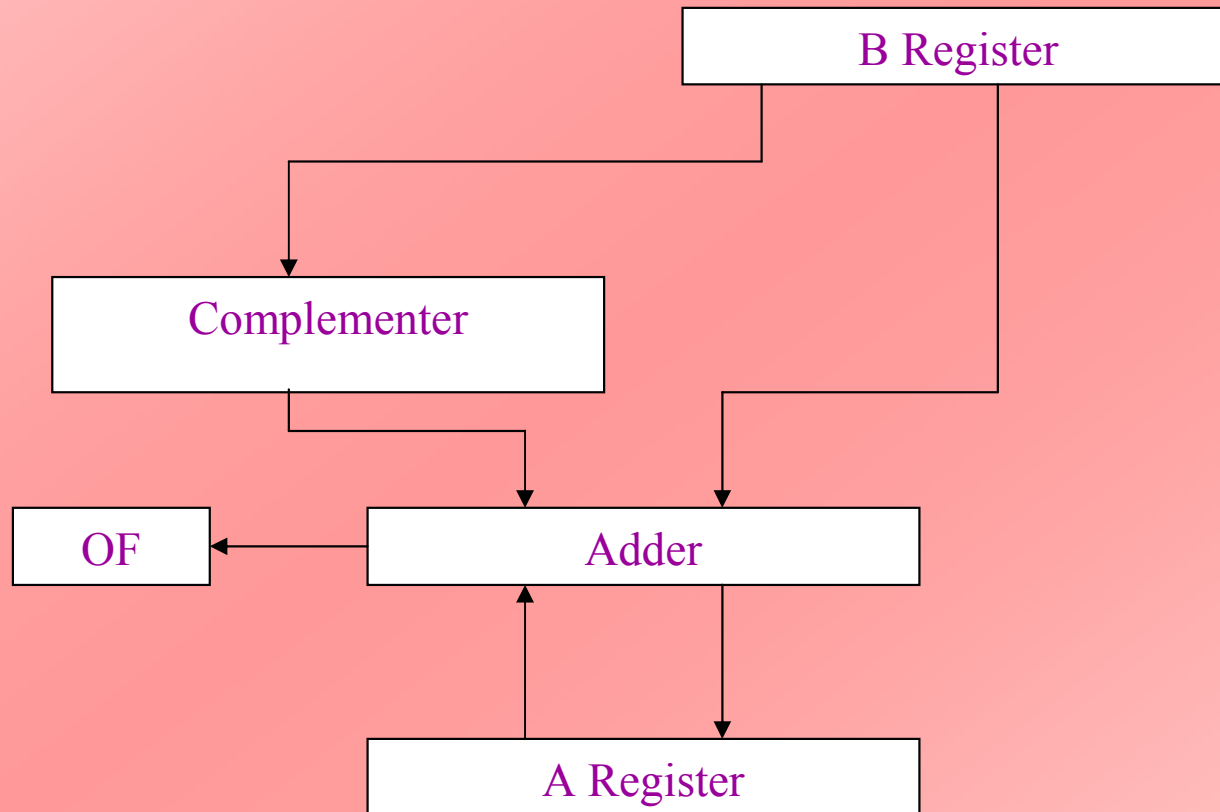
$$\begin{array}{r} 1010 \\ \underline{1+} \\ 1011 \end{array}$$

(-) to (+), (+) to (-)

- Demikian juga sebaliknya (negatif ke positif) dapat dilakukan dengan algoritma yang sama
- Tetapi cara ini terdapat dua anomali dalam sistem Komplemen Dua, yaitu pengubahan integer 0 dan - 128 seperti dijelaskan dibawah ini dengan contoh word 8 bit

0000 0000 = 0
di balik menjadi 1111 1111
jika ditambah 1+
sama dengan 10000 0000 over flow dapat di abaikan
1000 0000 = -128
di balik menjadi 0111 1111
jika ditambah 0000 0001+
sama dengan 1000 0000 sama dengan -128

Diagram Proses Pengurangan



Perkalian dan Pembagian

- Pada proses perkalian dapat dilakukan dengan melakukan penjumlahan berulang kali, misal: $2 * 4 = 2 + 2 + 2 + 2 = 8$.
- Cara Pendekatan ?

? Heuristic Method

- Menggunakan pendekatan perkalian yang dilakukan dengan pensil

$$\begin{array}{r} 1011 \\ \underline{1100} \times \\ 0000 \\ 0000 \\ 1011 \\ \underline{1011} \quad + \\ 10000100 \end{array}$$

multiplicand (11)
multiplier (12)
product (132)

Apa yang dapat anda simpulkan ?

=

Apa yang anda amati

Amati !

1. Perkalian meliputi pembentukan produk – produk parsial dan untuk memperoleh hasil akhir dengan menjumlahkan produk – produk parsial
2. Definisi produk parsial adalah multiplier bit sama dengan 0 maka produk parsialnya adalah 0, bila multiplier bit sama dengan 1 maka produk parsialnya sama dengan multiplikan
3. Terjadi penggeseran produk parsial satu bit ke kiri dari produk parsial sebelumnya
4. Perkalian dua buah integer biner n -bit akan menghasilkan bentuk produk yang panjangnya sampai dengan $2n$ -bit

Kesimpulan proses

- Terdapat penyederhanaan proses dari pada contoh penjumlahan diatas, yaitu
 - Dengan penambahan berjalan pada produk parsial daripada menunggu sampai akhir.
 - Hal ini akan menghemat proses dan tempat penyimpanan dalam prosesor.
- * Operasinya :
 - Apabila multiplier bernilai 1 terdapat operasi penambahan dan penggeseran,
 - Apabila multiplier bernilai 0 hanya terdapat operasi penggeseran saja

Penyederhanaan masalah (1011h * 1100h)

- Mula – mula Q diisi bit multiplier kemudian kontrol logika membaca bit multiplier satu per satu.
- Bila Q0 sama dengan 1, maka multiplikan ditambahkan ke register A dan hasilnya disimpan dalam register A.
- Selanjutnya seluruh bit yang berada dalam C, A, dan Q digeser ke kanan satu bit.
- Bila Q0 sama dengan 0, maka hanya terjadi penggeseran isi C, A, dan Q.
- Proses berulang hingga selesai perkalian dan hasil perkalian tersimpan pada register A dan register Q

Penjelasan

- Hasil akhir perkalian tersimpan pada register A dan register Q, dengan MSB pada bit paling kiri register A dan LSB pada bit paling kanan register Q.
- Implementasi hardware dari proses perkalian ini disajikan pada gambar diagram proses pergeseran
- Proses perkaliannya dijelaskan pada diagram alir

- Perkalian tersebut diatas tidak berlaku pada representasi bilangan two's komplement
- Bagaimana representasi dengan 2's Complement

?

- Untuk mengalikan bilangan dengan representasi 2's komplement menggunakan **algoritma booth** yang merupakan modifikasi dari algoritma diatas yaitu
 - Set register A dan register Q-1 dengan 0. Isikan multiplikan pada register M dan multiplier pada register Q. Set counter dengan nilai n-bit word
 - Perhatikan posisi bit – bit pada Q0 dan Q-1. Apabila :
 - $Q_0Q_{-1} = 11$ dan 00 , maka geser isi A, Q, Q-1 ke kanan 1 bit
 - $Q_0Q_{-1} = 10$, maka isi register A dikurangi isi register M dan hasilnya disimpan pada register A. Lakukan operasi penggeseran
 - $Q_0Q_{-1} = 01$, maka isi register A ditambah isi register M dan hasilnya disimpan pada register A. Lakukan operasi penggeseran. Penggeseran dilakukan satu bit ke kanan dan MSB diisi bit bernilai sama dengan bit yang digeser pada posisi MSB sebelumnya
 - Operasi selesai setelah siklus sama dengan jumlah n-bit word yang dikalikan. Hasil berada pada register A dan register Q

Contoh :

Perkalian Komplemen Dua antara 7
(0111) dan 3 (0011) :

dimana :

- Isi register M dengan 0111
- Isi register Q dengan 0011

Contoh :

Siklus	A	Q	Q-1	Keterangan	
0	0000	0011	0	Initial value	Posisi Q0Q-1 = 10, siklus berikutnya A – M, lalu geser.
1	1001 1100	0011 1001	0 1	A A – M Shift	Register A berisi 1001 berasal dari (A – M), yaitu 0000 – 0111. Setelah geser posisi Q0Q-1 = 11, jadi siklus berikutnya geser saja.
2	1110	0100	1	Shift	Posisi Q0Q-1 = 01, siklus berikutnya A + M, lalu geser.
3	0101 0010	0100 1010	1 0	A A + M Shift	Register A berisi 0101 berasal dari (A – M), yaitu 1110 + 0111. Setelah geser posisi Q0Q-1 = 00, jadi siklus berikutnya geser saja.
4	0001	0101	0	Shift	Siklus = n bit, stop!

Hasil perkalian integer 3 dan 7 adalah 0001 0101 = 21

Pembagian

- Pembagian pada unsigned binary seperti halnya pada sistem desimal
- Istilah dalam pembagian ?
 - Devidend adalah bilangan yang dibagi,
 - Divisor adalah bilangan pembagi,
 - Quotient adalah hasil pembagian,
 - Remainders adalah sisa pembagian,
 - Partial remainders adalah sisa pembagian parsial

Contoh :

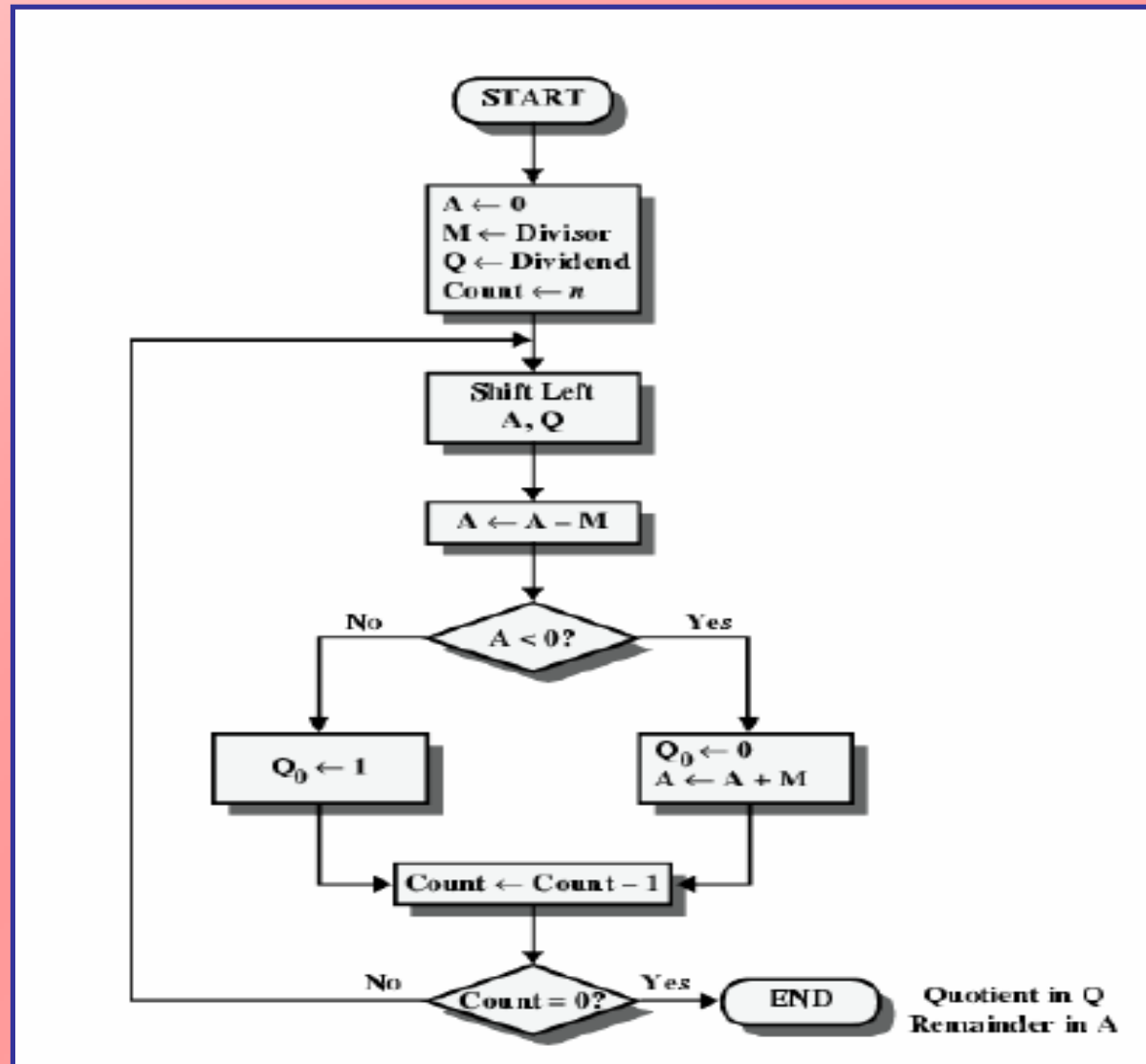
Desimal :

$$\begin{array}{r} 11 \\ 13 \overline{) 147} \\ \underline{143} \\ 4 \end{array}$$

Biner :

$$\begin{array}{r} 1101 \\ 1011 \overline{) 10010011} \\ \underline{1011} \\ 01110 \\ \underline{1011} \\ 001111 \\ \underline{1011} \\ 100 \end{array}$$

Diagram Alir



Penjelasan :

- Menjelaskan algoritma mesin bagi proses pembagian unsigned binary.
- Divisor diletakkan pada register M, sedangkan diviend pada register Q.
- Pada setiap langkah, register A dan register Q secara bersama – sama digeser ke kiri sejauh 1 bit.
- M dikurangkan dari A untuk menentukan apakah A dapat membagi partial remainders.
- Bila dapat dibagi, maka Q0 akan memperoleh bit bernilai 1.
- Bila tidak dapat dibagi, Q0 akan memperoleh bit bernilai 0 dan M harus ditambahkan kembali ke A untuk menyimpan nilai sebelumnya.
- Counter dikurangi 1 setiap siklus proses, dan proses berlanjut sebanyak n langkah.
- Pada akhirnya quotient akan berada pada register Q dan remainders berada pada register A

A	Q	M = 0011	A	Q	M = 0011
0000	1001	Initial Value	0000	0111	Initial Value
0000	1110	Shift	0000	1110	Shift
1101		Subtrac	1101		Add
0000	1110	Restore	0000	1110	Restore
0001	1100	Shift	0001	1100	Shift
1110		Subtrac	1110		Add
0001	1100	Restore	0001	1100	Restore
0011	1000	Shift	0011	1000	Shift
0000		Subtrac	0000		Add
0000	1001	Set $Q_0 = 1$	0000	1001	Set $Q_0 = 1$
0001	0010	Shift	0001	0010	Shift
1110		Substrac	1110		Add
0001	0010	Restore	0001	0010	Restore

(a) (7) : (3)

(a) (7) : (- 3)

Pembagian Komplemen Dua

? Alorithmanya

1. Muatkan divisor ke register M dan dividend ke register A dan Q. Dividend harus diekspresikan sebagai komplemen dua 2^n -bit
2. Geser A, Q ke kiri sejauh 1 bit
3. Bila M dan A memiliki tanda yang sama, lakukan $A \leftarrow A - M$. Bila tandanya berbeda, lakukan $A \leftarrow A + M$
4. Operasi akan berhasil bila tanda A sesudah dan sebelum operasi tetap
 1. Bila operasi berhasil atau ($A = 0$ AND $Q = 0$), maka set $Q_0 = 1$
 2. Bila operasi gagal, maka set $Q_0 = 0$ dan simpan nilai A sebelumnya
5. Ulangi langkah 2 sampai 4 sampai terdapat posisi bit di Q
6. Remainders akan berada di A. Bila tanda divisor dan dividend sama, maka quotient akan berada di Q, sedangkan bila tanda tidak sama maka quotient yang benar adalah komplemen dua dari Q

Bagaimana Formulasi nya ?

Formulasi secara umum

- $D = (Q \times V) + R$

Dimana :

- $D = \text{dividend}$ $V = \text{divisor}$
- $Q = \text{quotient}$ $R = \text{remainder}$

Pembagian 7 (0111) oleh 3 (0011)

A	Q	M =0011
0000	0111	Inisialisasi
0000	1110	Shift 1
1101		Subtract
0000	1110	Restore
0001	1100	Shift 2
1110		Subtract
0001	1100	Restore
0011	1000	Shift 3
0000		Subtract
0000	1001	Set Q0=1
0001	0010	Shift 4
1110		Subtract
0001	0010	Restore

Selamat Belajar

