Penerapan Aljabar Boolean pada Mikroprosesor

Muhammad Fariz Luthfan Wakan, 13517034
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517034@std.stei.itb.ac.id

***Abstrak* — Makalah ini dibuat untuk menjelaskan salah satu penerapan Aljabar Boolean dalam Mikroprosesor terutama *Arithmatics Logical Unit* (ALU). Mikroprosesor merupakan otak dalam sebuah sistem komputer. ALU merupakan komponen yang terdapat pada mikroprosesor yang bertugas untuk melakukan operasi aritmetika dan operasi logika. *Adder* merupakan salah satu rangkaian yang terdapat pada ALU. Makalah ini memodelkan berbagai macam rangkaian *Adder* menggunakan Gerbang Logika.**

Kata Kunci* — Aljabar Boolean, Mikroprosesor, *Arithmatics Logical Unit, Half Adder, Full Adder, Paralel Adder.

I. Pengantar

"Computers have the potential to do all kinds of amazing things. But the only thing that makes the computer smart, or useful, is you." – Bill Gates.

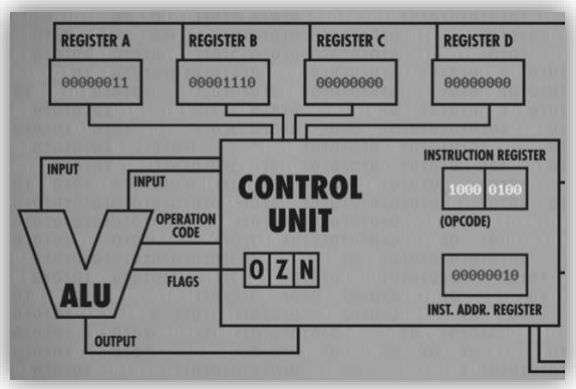
Virtual Reality, *Augmented Reality*, dan *Artificial Intelligence* merupakan teknologi dari hasil kreatifitas manusia yang membuktikan bahwa komputer dapat melakukan hal-hal yang luar biasa. Hanya dengan *HIGH* dan *LOW*, 1 dan 0, komputer dapat melakukan banyak hal, dari matematika sederhana sampai mensimulasikan dunia virtual.

Berbicara tentang komputer, terdapat banyak sekali komponen yang membuat komputer dapat melakukan berbagai hal. Salah satu komponen terpentingnya adalah prosesor.

Prosesor atau mikroprosesor merupakan IC (*Integrated Circuit*) yang digunakan sebagai otak dalam sebuah sistem komputer. Mikroprosesor terdiri dari beberapa rangkaian penting yang memiliki fungsi yang berbeda-beda. Berikut rangkaian yang terdapat pada mikroprosesor,

1. *Arithmatics Logical Unit* (ALU), bertugas untuk melakukan operasi aritmetika dan operasi logika.
2. *Control Unit* (CU), bertugas untuk memberikan arahan/kendali/kontrol terhadap operasi yang dilakukan di bagian ALU.
3. *Memory Unit* (MU) atau *registers*, penyimpanan data berkapasitas kecil dengan kecepatan yang sangat tinggi.

Cara kerja mikroprosesor yaitu menjalankan sederet instruksi atau perintah yang akan memberitahu mikroprosesor apa saja yang harus dilakukan dan dikerjakakan berdasarkan instruksi tersebut. Terdapat 3 tahap dalam satu siklus pada mikroprosesor dalam mengerjakan satu instruksi atau perintah. *Fetch phase*, *Control Unit* mengambil instruksi yang terdapat pada RAM. *Decode Phase*, *Control Unit* menerjemahkan perintah yang telah diambil. *Execute phase*, *Control Unit* melakukan eksekusi terhadap perintah yang telah diterjemahkan dan melanjutkannya ke ALU jika perintah yang diminta merupakan operasi aritmetika ataupun operasi logika.



Gambar 1. Gerbang logika Mikroprosesor

Komponen pada mikroprosesor terbentuk dari puluhan (bahkan ratusan) sirkuit atau rangkaian listrik. Aljabar Boolean sangat berperan penting dalam pembuatan komponen-komponen yang terdapat pada mikroprosesor. Aljabar Boolean digunakan untuk pemodelan gerbang logika, terutama penyederhanaan rangkaian yang terdapat pada mikroprosesor. Penyederhanaan rangkaian sangatlah penting dari berbagai aspek dalam pembuatan mikroprosesor.

Dari beberapa rangkaian yang ada, akan dibahas peran Aljabar Boolean terhadap rangkaian-rangkaian dasar yang membentuk mikroprosesor, rangkaian *Half Adder*; rangkaian *Full Adder*; dan rangkaian *Paralel Adder*.

II. Teori Aljabar Boolean, Gerbang Logika, dan Metode Peta Karnaugh

A. Aljabar Boolean

Aljabar Boolean pertama kali diperkenalkan oleh seorang matematikawan Inggris, George Boole. Aljabar Boolean merupakan cabang matematika yang digunakan untuk banyak hal pada saat ini, terutama menganalisis dan menyederhanakan Gerbang Logika pada rangkaian-rangkaian yang terdapat pada mikroprosesor. Aljabar Boolean setidaknya memiliki dua buah elemen yang menyatakan *False* atau *True* (0 dan 1), dan tiga buah operasi,

AND (•)	0	1
0	0	0
1	0	1

Tabel 1. Operasi AND

Operasi AND, dengan simbol '•'. Operasi AND akan menghasilkan nilai *True* jika kedua buah masukan adalah *True*. Pada tabel di atas, *True* dilambangkan oleh 1 dan *False* dilambangkan oleh 0.

OR (+)	0	1
0	0	1
1	1	1

Tabel 2. Operasi OR

Operasi OR, dengan simbol '+'. Operasi OR akan menghasilkan nilai *False* jika kedua buah masukan adalah *False*. Pada tabel di atas, *True* dilambangkan oleh 1 dan *False* dilambangkan oleh 0.

NOT (')	
0	1
1	0

Tabel 3. Operasi NOT

Operasi NOT, dengan simbol '''. Berbeda dengan dua operasi sebelumnya, operasi NOT merupakan operasi uner, operasi yang hanya menggunakan satu buah unsur. Sedangkan dua operasi sebelumnya merupakan operasi biner, operasi yang menggunakan dua buah unsur. Operasi NOT akan menghasilkan nilai yang berkebalikan dengan nilai masukannya, *True* jika masukan adalah *False*; *False* jika masukan adalah *True*. Pada tabel di atas, *True* dilambangkan oleh 1 dan *False* dilambangkan oleh 0.

Terdapat beberapa hukum yang berlaku pada Aljabar Boolean. Berikut hukum-hukum yang berlaku,

1. Hukum identitas, $a + 0 = a$; $a \bullet 1 = a$.
2. Hukum idempoten, $a + a = a$; $a \bullet a = a$.
3. Hukum komplemen, $a + a' = 1$; $aa' = 0$.
4. Hukum dominasi, $a + 1 = 1$; $a \bullet 0 = 0$.
5. Hukum involusi, $(a')' = a$.
6. Hukum penyerapan, $a + ab = a$; $a(a + b) = a$.
7. Hukum komutatif, $a + b = b + a$; $ab = ba$.
8. Hukum asosiatif, $a + (b + c) = (a + b) + c$; $a(bc) = (ab)c$.
9. Hukum distributif, $a + (bc) = (a + b)(a + c)$; $a(b + c) = ab + bc$.
10. Hukum De Morgan, $(a + b)' = a'b'$; $(ab)' = a' + b'$.
11. Hukum 0/1, $0' = 1$; $1' = 0$.

Ekspresi Boolean dalam Aljabar Boolean dapat diekspresikan dalam dua bentuk yang berbeda, penjumlahan dari hasil kali (*sum-of-product* atau SOP) dan perkalian dari hasil jumlah (*product-of-sum* atau POS). Penjumlahan dari hasil kali merupakan penjumlahan dari satu atau lebih *minterm* (hasil kali),

$$f(x, y) = x'y + xy'$$

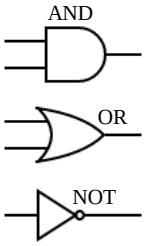
Ekspresi di atas terdiri dari dua *minterm*, $x'y$ dan xy'. Sedangkan, perkalian dari hasil jumlah merupakan perkalian dari satu atau lebih *maxterm* (hasil jumlah),

$$f(x, y) = (x + y')(x' + y)$$

Ekspresi di atas terdiri dari dua *maxterm*, $x + y'$ dan $x' + y$.

B. Gerbang Logika

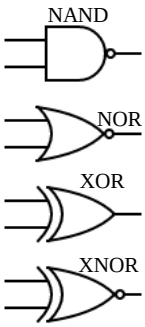
Gerbang logika merupakan suatu metode untuk menggambarkan fungsi Boolean. Setiap gerbang logika merepresentasikan sebuah operasi Boolean. Terdapat tiga macam gerbang logika dasar,



Gambar 2. Gerbang logika dasar

Setiap gerbang logika memiliki bentuk yang berbeda-beda dimana menjadi identitas dari gerbang logika tersebut. Sisi kiri gerbang logika adalah masukan dan sisi kanan gerbang logika merupakan keluaran dari operasi tersebut. Pada gerbang logika selain NOT, masukan dapat berupa dua sampai *n*.

Selain gerbang logika dasar (AND, OR, dan NOT), terdapat gerbang logika turunan,



Gambar 3. Gerbang logika turunan

Gerbang logika turunan merupakan kombinasi dari gerbang-gerbang dasar. NAND merupakan kombinasi dari AND dan NOT. NOR yang merupakan kombinasi dari OR dan NOT.

Begitu juga dengan XOR dan XNOR. X pada XOR dan XNOR adalah Eksklusif (*Exlucive*). Jika OR akan menghasilkan *False* jika kedua masukan adalah *False*. Pada *Exclusive* OR terdapat perbedaan dengan OR, *Exclusive* OR akan menghasilkan *True* jika salah satu masukan tetapi tidak keduanya bernilai *True*.

XOR	0	1
0	0	1
1	1	0

Tabel 4. Operasi XOR

C. Metode Peta Karnaugh

Metode Peta Karnaugh (atau *K-map*) merupakan metode untuk memetakan fungsi *sum-of-product* dalam bentuk tabular, dimana setiap elemen pada tabel tersebut merepresentasikan sebuah *minterm*. Tabel yang memetakan fungsi sum-of-product tersebut merupakan Peta Karnaugh. Dengan Peta Karnaugh, meminimalkan atau menyederhanakan fungsi Boolean menjadi lebih mudah.

Penggunaan Peta Karnaugh dalam meminimalkan atau menyederhanakan fungsi Boolean dilakukan dengan cara menggabungkan elemen-elemen bernilai 1 yang saling bersisian. Kelompok dari penggabungan tersebut membentuk pasangan (dua), kuad (empat), dan oktet (delapan). Berikut Peta Karnaugh yang terbentuk dari fungsi Boolean *f(w, x, y, z) = w'x'y'z' + w'x'y'z + w'x'yz + w'x'yz' + w'xy'z*.

wx/yz	00	01	11	10
00	1	1	1	1
01	0	1	0	0
11	0	0	0	0
10	0	0	0	0

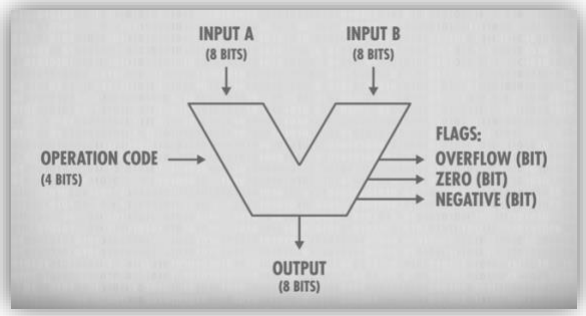
Tabel 5. Peta Karnaugh f(w, x, y, z)

Berdasarkan Peta Karnaugh di atas terdapat dua kelompok yang dapat dibentuk dari penggabungan elemen-elemen bernilai 1 yang saling bersisian, pasangan dan kuad. Pada kelompok yang membentuk pasangan, masukan pada peubah *x* tidak mempengaruhi hasil, sehingga *w'x'y'z* dan *w'xy'z* dapat diubah menjadi *w'y'z*. Pada kelompok yang membentuk kuad, masukan pada peubah *y* dan *z* tidak mempengaruhi hasil, sehingga *w'x'y'z'; w'x'y'z; w'x'yz; dan w'x'yz* dapat diubah menjadi *w'x'*. Dari penyederhanaan tersebut, didapat fungsi Boolean yang lebih sederhana, *f(w, x, y, z) = w'x' + w'y'z*.

III. ARITHMATICS LOGICAL UNIT

Arithmatics Logical Unit atau ALU merupakan komponen yang terdapat pada mikroprosesor yang bertanggungjawab dalam pemrosesan data, untuk melakukan operasi aritmetika dan operasi logika berdasarkan instruksi yang ditentukan. ALU terdiri dari dua bagian, unit aritmetika dan unit logika *boolean*.

Instruksi yang diberikan membuat ALU melakukan pemrosesan data lebih spesifik, operasi penjumlahan; operasi pengurangan; operasi perkalian; operasi pembagian; atau operasi lainnya.



Gambar 4. Gerbang logika Arithmatics Logical Unit (ALU)

Dalam menjalankan tugasnya untuk melakukan operasi aritmetika dan operasi logika, ALU melibatkan sebuah sirkuit khusus yang disebut dengan *Adder*.

Adder adalah rangkaian dari sekumpulan gerbang logika yang disusun sedemikian rupa sehingga dapat melakukan operasi penjumlahan dalam bentuk biner. Adder terdiri dari beberapa macam, *Half Adder; Full Adder;* dan *Paralel Adder*.

Half Adder merupakan rangkaian *Adder* yang paling sederhana, yang digunakan untuk menjulahkan dua buah bit. *Half Adder* memberikan hasil penjumlahan tak lengkap, sehingga terdapat rangkaian *Full Adder* dimana rangkaian *Full Adder* dapat menerima nilai *carry* dari hasil penjumlahan sebelumnya. *Paralel Adder* merupakan sekumpulan rangkaian *Full Adder* yang disusun sedemikian rupa sehingga dapat melakukan operasi penjumlahan secara lengkap.

Selain rangkaian *Adder*, terdapat juga sirkuit *Substractor*. Tidak hanya rangkaian *Substractor*, tetapi terdapat puluhan bahkan ratusan sirkuit pada ALU.

IV. PEMODELAN GERBANG LOGIKA

A. Half Adder

Dalam penjumlahan dua buah bit (satu bit dengan satu bit) harus menghasilkan satu buah bit pula. Karena bilangan biner terdiri dari 0 dan 1, maka penjumlahan 1 dengan 1 akan menghasilkan 0 (melebihi representasi satu buah bit). Sehingga dibutuhkan sebuah bit yang bernama *CARRY* dimana *CARRY* berfungsi untuk mengetahui apakah hasil dari penjumlahan dua buah bit menghasilkan nilai yang melimpah (*overflow*) atau tidak. Berikut informasi masukan dan keluaran berdasarkan kombinasi masukan dua buah bit,

Masukan	Keluaran

A	B	SUM	CARRY
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Tabel 6. Truth table untuk Half Adder

Berdasarkan informasi tabel di atas, dapat dibentuk Peta Karnaugh dengan membagi keluaran menjadi dua Peta Karnaugh yang berbeda, Peta Karnaugh untuk *SUM* dan Peta Karnaugh untuk *CARRY*. Pada kedua Peta Karnaugh, Kolom merepresentasikan nilai bit A dan baris merepresentasikan nilai bit B.

A/B	0	1
0	0	1
1	1	0

Tabel 7. Peta Karnaugh untuk SUM (Half Adder)

Fungsi yang terbentuk dari Peta Karnaugh di atas,

$$S(A, B) = A'B + AB'$$

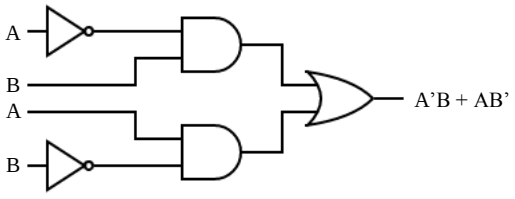
A/B	0	1
0	0	0
1	0	1

Tabel 8. Peta Karnaugh untuk CARRY (Half Adder)

Fungsi yang terbentuk dari Peta Karnaugh di atas,

$$C(A, B) = AB$$

Berdasarkan kedua fungsi yang terbentuk, *S(A, B)* dan *C(A, B)*, dapat dihasilkan gerbang logika sebagai berikut,



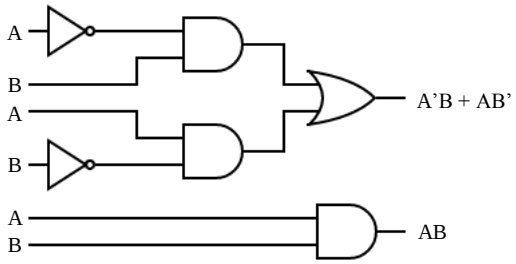
Gambar 5. Gerbang Logika S(A, B)

Pada fungsi S(A, B) terdapat dua gerbang NOT, dua gerbang AND, dan satu gerbang NOT. Gerbang AND pertama menghasilkan A'B dari masukan A' dan B. A' terbentuk setelah A melewati gerbang NOT. Gerbang AND kedua menghasilkan AB' dari masukan A dan B'. B' terbentuk setelah melewati gerbang NOT. Kedua hasil gerbang AND menjadi masukan dari gerbang OR dimana menghasilkan A'B + AB'.

Gambar 6. Gerbang Logika C(A, B)

Pada fungsi C(A, B) hanya terdapat satu gerbang AND dengan menghasilkan AB dari masukan A dan B.

Gerbang logika *Half Adder* didapat setelah menggabungkan kedua gerbang logika di atas. Berikut gerbang logika *Half Adder*,



Gambar 7. Gerbang Logika Half Adder

B. Full Adder

Mengisi kekurangan *Half Adder*, *Full Adder* memiliki tiga buah masukan, dua buah bit seperti *Half Adder* dan satu buah bit *CARRY*. Jika satu buah rangkaian Half Adder digabungkan dengan satu buah rangkaian *Full Adder*, rangkaian tersebut memiliki fungsi penjumlahan empat buah bit (dua bit dengan dua bit), karena *Full Adder* akan menerima *CARRY* yang diberikan *Half Adder*. Berikut informasi masukan dan keluaran rangkaian *Full Adder*,

Masukan			Keluaran	
A	B	$CARRY_{IN}$	SUM	$CARRY_{OUT}$
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1

1	0	1	0	1
1	1	1	1	1

Tabel 9. Truth table untuk Full Adder

Berdasarkan tabel di atas, dapat dibentuk Peta Karnaugh dengan membagi keluaran menjadi dua Peta Karnaugh yang berbeda, Peta Karnaugh untuk *SUM* dan Peta Karnaugh untuk $CARRY_{OUT}$. Pada kedua Peta Karnaugh, Kolom merepresentasikan nilai dua buah bit AB dan baris merepresentasikan nilai satu buah bit $CARRY_{OUT}$.

$AB/CARRY_{IN}$	00	01	11	10
0	0	1	0	1
1	1	0	1	0

Tabel 10. Peta Karnaugh untuk SUM (Full Adder)

Fungsi yang terbentuk dari Peta Karnaugh di atas,

$$S(A, B, CARRY_{IN}) = A'BCARRY_{IN}' + AB'CARRY_{IN}' + A'B'CARRY_{IN} + ABCARRY_{IN}$$

$AB/CARRY_{IN}$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Tabel 11. Peta Karnaugh untuk CARRY (Full Adder)

Fungsi yang terbentuk dari Peta Karnaugh di atas,

$$C(A, B, CARRY_{IN}) = ABCARRY_{IN}' + A'BCARRY_{IN} + ABCARRY_{IN} + AB'CARRY_{IN}$$

Fungsi C(A, B, $CARRY_{IN}$) dapat disederhanakan dengan mengunakan metode Karnaugh,

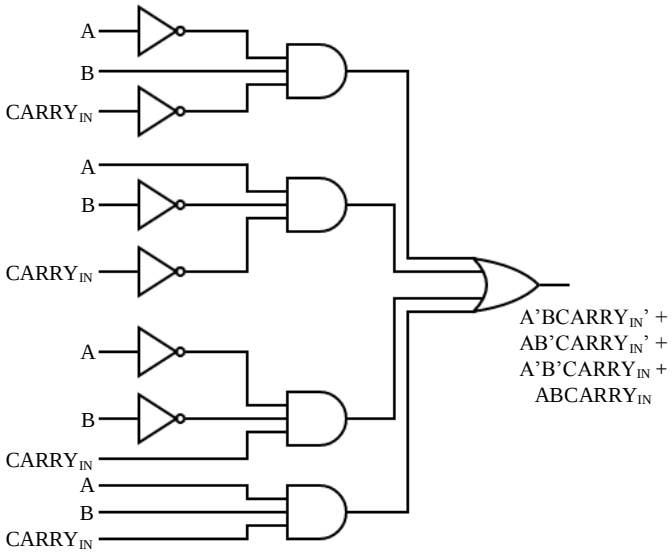
$AB/CARRY_{IN}$	00	01	11	10
0	0	0	1	0
1	0	1	1	1

Tabel 12. Minimalisasi Peta Karnaugh untuk CARRY (Full Adder)

Menimalisasi Peta Karnaugh dilakukan dengan mengelompokkan angka 1 yang bersebelahan membentuk 2, 4, 8, atau 16. Berdasarkan Peta Karnaugh di atas, kita dapat mengelompokkan angka 1 menjadi 3 kelompok, AB; $BCARRY_{IN}$; dan $ACARRY_{IN}$. Berikut fungsi C(A, B, $CARRY_{IN}$) dalam bentuk yang lebih sederhana,

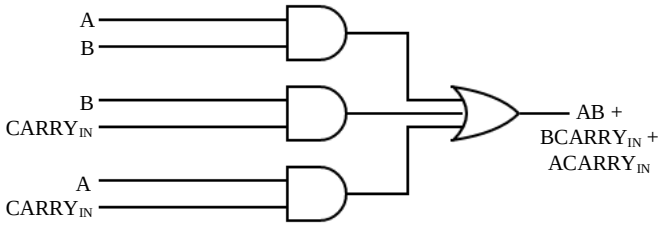
$$C(A, B, CARRY_{IN}) = AB + BCARRY_{IN} + ACARRY_{IN}$$

Berdasarkan kedua fungsi yang terbentuk, $S(A, B, CARRY_{IN})$ dan $C(A, B, CARRY_{IN})$, dapat dihasilkan gerbang logika sebagai berikut,



Gambar 8. Gerbang Logika $S(A, B, CARRY_{IN})$

Pada fungsi S(A, B, $CARRY_{IN}$) terdapat enam gerbang NOT, empat gerbang AND dan datu gerbang OR. Gerbang AND pertama menghasilkan A'B$CARRY_{IN}$' dari masukan A', B, dan $CARRY_{IN}$'. A' dan $CARRY_{IN}$' terbentuk setelah melewati gerbang NOT. Gerbang AND kedua menghasilkan AB'$CARRY_{IN}$' dari masukan A, B', dan $CARRY_{IN}$'. B' dan $CARRY_{IN}$' terbentuk setelah melewati gerbang NOT. Gerbang AND ketiga menghasilkan A'B'$CARRY_{IN}$ dari masukan A', B', dan $CARRY_{IN}$. A' dan B' terbentuk setelah melewati gerbang NOT. Gerbang AND terakhir menghasilkan $ABCARRY_{IN}$. Keempat hasil gerbang AND menjadi masukan dari gerbang OR dimana menghasilkan A'B$CARRY_{IN}$' + AB'$CARRY_{IN}$' + A'B'$CARRY_{IN}$ + $ABCARRY_{IN}$.

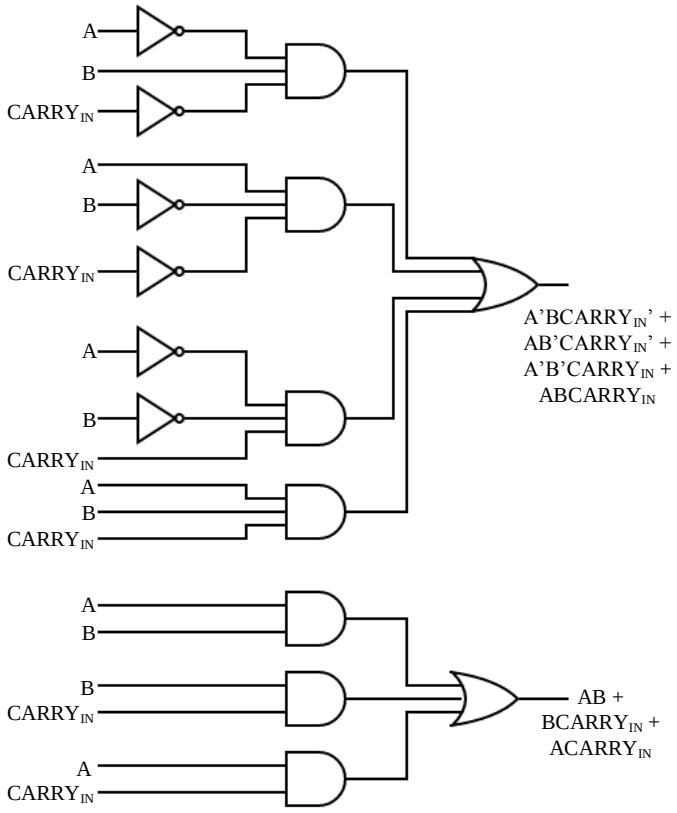


Gambar 9. Gerbang Logika $C(A, B, CARRY_{IN})$

Pada fungsi C(A, B, $CARRY_{IN}$) terdapat tiga gerbang AND, dan satu gerbang OR. Gerbang AND pertama menghasilkan AB dari

masukan A dan B. Gerbang AND kedua menghasilkan $BCARRY_{IN}$ dari masukan B dan $CARRY_{IN}$. Gerbang AND ketiga menghasilkan $ACARRY_{IN}$ dari masukan A dan $CARRY_{IN}$. Ketiga hasil gerbang AND menjadi masukan dari gerbang OR dimana menghasilkan AB + $BCARRY_{IN}$ + $ACARRY_{IN}$.

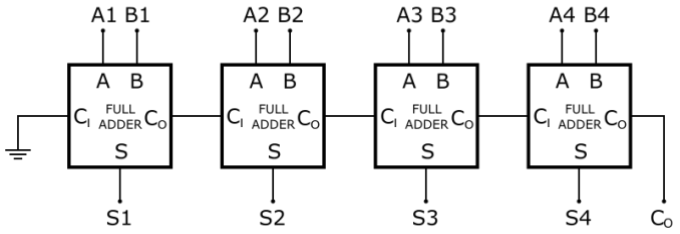
Gerbang logika *Full Adder* didapat setelah menggabungkan kedua gerbang logika di atas. Berikut gerbang logika *Full Adder*,



Gambar 10. Gerbang Logika Full Adder

C. Paralel Adder

Paralel Adder atau biasa disebut *Ripple Carry Adder* merupakan rangkaian penjumlahan dari beberapa bit. *Ripple Carry Adde*r merupakan implementasi dari rangkaian *Full Adder*. *Ripple Carry Adder* merupakan kumpulan dari rangkaian *Full Adder* yang disusun sedemikian rupa sehingga dapat melakukan operasi penjumlahan secara lengkap. Sehingga untuk penjumlahan antara 8-bit dengan 8-bit dibutuhkan 8 komponen *Full Adder* di dalam *Riple Carry Adder*. Dengan mengabstraksikan gerbang logika Full Adder, berikut gerbang logika *Ripple Carry Adder*,



Gambar 11. Gerbang Logika Ripple Carry Adder

Gambar di atas merupakan rangkaian penjumlahan delapan bit (empat bit dengan emapt bit), *4-bit Ripple Carry Adder*. A merupakan 4 bit pertama dan B merupakan 4 bit kedua. Hasil dari A dan B adalah S. Pada mikroprosesor, C_o yang terdapat pada paling akhir (paling kanan pada gambar) memberikan informasi bahwa hasil penjumlahan dari bit-bit tersebut menghasilkan nilai yang melimpah (*overflow*).

V. SIMPULAN

Laptop, ponsel pintar, dan berbagai macam perangkat lainnya, terdapat banyak sekali sirkuit atau rangkaian listrik di dalamnya, dimana sirkuit-sirkuit tersebut dibentuk menggunakan ilmu yang terdapat pada Aljabar Boolean khususnya gerbang logika. Aljabar Boolean membantu bagaimana suatu perangkat dapat berjalan dan bagaimana model gerbang logika pada perangkat tersebut.

Untuk memodelkan gerbang logika suatu perangkat, kita perlu memahami bagaimana perangkat tersebut bekerja. Dari pemahaman tersebut dapat dibentuk tabel kejujuran, sehingga di dapat fungsi Boolean dari perangkat tersebut. Penggunaan Peta Karnaugh dibutuhkan untuk menyederhanakan fungsi Boolean tersebut. Hasil dari penyederhanaan fungsi Boolean dapat divisualisasikan menggunakan konsep gerbang logika.

DAFTAR PUSTAKA

[1] Albert P. Malvino, *Digital Computer Electronics*. Illinois: Career Education, 1992.

[2] Rinaldi Munir, Matematika Diskrit. Bandung: Penerbit INFORMATIKA Bandung, 2010, ch 7.

[3] Carrie Anne, How Computers Calculate - the ALU: Crash Course Computer Science #5. https://www.youtube.com/watch?v=1I5ZMmrOfnA, diakses pada tanggal 1 Desember 2018.

[4] Carrie Anne, The Central Processing Unit (CPU): Crash Course Computer Science #7. https://www.youtube.com/watch?v=FZGugFqdr60, diakses pada tanggal 1 Desember 2018.

[5] Adi Catur Pamungkas, Pengertian Fungsi dan Cara Kerja Processor Lengkap Bagian – MasTekno. https://www.mastekno.com/id/pengertian-fungsi-dan-cara-kerja-processor, diakses pada tanggal 2 Desember 2018.

[6] Mochamad Alif Prayogo, Pengertian ALU, Fungsi ALU dan Rangkaian ALU pada CPU, Processor Komputer – MasTekno. https://www.mastekno.com/id/pengertian-alu-fungsi-cara-kerja-alu, diakses pada tanggal 2 Desember.

[7] Budi Kiswoyo, Rangkaian Adder atau Penjumlah. https://www.jalankatak.com/id/rangkaian-adder-atau-penjumlah, diakses pada tanggal 2 Desember.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 9 Desember 2018

Muhammad Fariz Luthfan Wakan
13517034