BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Deskripsi Masalah

Seperti yang sudah dipaparkan pada BAB I, permasalahan yang timbul adalah belum tersedianya perangkat lunak yang memanfaatkan metode Quine-McCluskey dalam menyederhanakan fungsi logika sehingga tersedia rangkaian gerbang logika yang lebih sederhana dari suatu fungsi. Fungsi Boolean yang tidak disederhanakan, akan menimbulkan kerumitan yang *relative* besar pada pengimplementasian gerbang logika. Dan apabila fungsi – fungsi tersebut dipertahankan, akan menambah kebutuhan memori yang cukup besar di dalam sistem komputer.

Penyederhanaan fungsi Boolean, dapat diselesaikan dengan beberapa metode yang telah baku yaitu: metode Aljabar, metode Peta Karnaugh, dan metode Quine-McCluskey. Metode Aljabar sudah jarang digunakan karena prosesnya yang mengandalkan *try & error*, maka rentan terjadi kesalahan.

Metode Peta Karnaugh dan metode Quine-McCluskey masing-masing memiliki langkah-langkah penyederhanaan yang berbeda dalam tingkat kerumitan dan ketelitiannya. Metode-metode penyederhanaan fungsi boolean ini dapat menghasilkan persamaan fungsi boolean yang sederhana untuk membentuk rangkaian logika dari persamaan sederhana yang didapat. Metode Peta Karnaugh memerlukan ketelitian yang cukup tinggi dalam proses penyederhanaan fungsi boolean. Untuk penyederhanaan jumlah variabel yang besar (yaitu 4, 5, 6 atau

bahkan lebih) akan lebih rumit karena pembentukan pola biner untuk menggambarkan sejumlah kotak berbentuk persegi panjang yang berisi minterm dimana setiap perubahan pola biner dari baris ke kolom sebelum dan sesudahnya hanya memiliki 1 (satu) buah perubahan saja. Pengelompokan angka '1' bersesuaian yang telah diisikan ke dalam peta akan membentuk banyak kemungkinan angka '1' yang bersebelahan untuk dijadikan kelompok, kondisi ini mempersulit proses penyederhanaan dan pembentukan rangkaian logikanya. Penyederhanaan menggunakan metode Quine-McCluskey memberikan hasil yang lebih pasti. Metode ini digunakan untuk mempresentasikan minimasi ekspresi fungsi boolean, dan menyediakan sebuah prosedur sistematis untuk membangun semua *Prime Implicant* dan kemudian mengambil sebuah set minimum dari *prime* yang ada. Metode tabulasi ini digunakan apabila fungsi yang akan disederhanakan memiliki variabel lebih dari enam.

Seperti yang telah dipaparkan pada BAB II, metode penyederhanaan yang disangka efektif untuk dikonversi ke dalam sistem komputer yaitu metode Quine-McCluskey atau sering disebut metode tabulasi. Metode ini mengubah sebuah fungsi Boolean menjadi sebuah himpunan prima, dimana sebanyak mungkin peubah dieliminasi (dihilangkan) secara maksimal, hingga didapat fungsi Boolean yang paling sederhana. Ini dapat dilakukan dengan melakukan perulangan penggunaan hukum komplemen, $a + a' = 1$.

Sebagai contoh, fungsi Boolean dengan empat peubah dalam bentuk SOP :

$f(a, b, c, d) = \sum(3, 11) = 3(0011, 1011) = a'b'cd + ab'cd$ dan

$f(a, b, c, d) = \sum(7, 11) = 3(0111, 1011) = a'b'cd + ab'cd.$

	a b c d		*a b c d*
	-------		-------
3	0 0 1 1	7	0 1 1 1
11	1 0 1 1	11	1 0 1 1
	-------		-------
BENTUK PRIMA -> (3,11)	- 0 1 1		?
	Contoh (a)		*Contoh (b)*

Pada contoh(a), kedua *minterm* tersebut dapat dikombinasikan menjadi sebuah bentuk prima yaitu (3,11), karena memiliki tepat satu perbedaan bit pada posisi bit nomor satu. Hasil kombinasi dalam bentuk prima (3,11) menyatakan bahwa peubah '*a*' telah dieleminasi. Hal ini sesuai dengan hukum komplemen, $a + a' = 1$. Pada contoh(b), kedua *minterm* tersebut tidak dapat dikombinasikan menjadi sebuah bentuk prima, karena memiliki dua perbedaan bit pada posisi bit nomor satu dan dua. Setiap kombinasi dari *minterm* yang dapat membentuk sebuah bentuk prima baru harus memiliki tepat satu perbedaan bit pada posisi yang sama. Metode Quine-McCluskey mereduksi sebuah fungsi yang bentuk ekspresi booleannya berbentuk *Sum Of Product* atau penjumlahan dari perkalian, untuk mendapatkan sebuah bentuk *prime implicant.*

Tahapan dari penyederhanaan fungsi boolean dengan menggunakan metode Quine-McCluskey jika dibuat kedalam bentuk skema penyederhanaan ditampilkan dalam diagram alir 3.1 berikut :

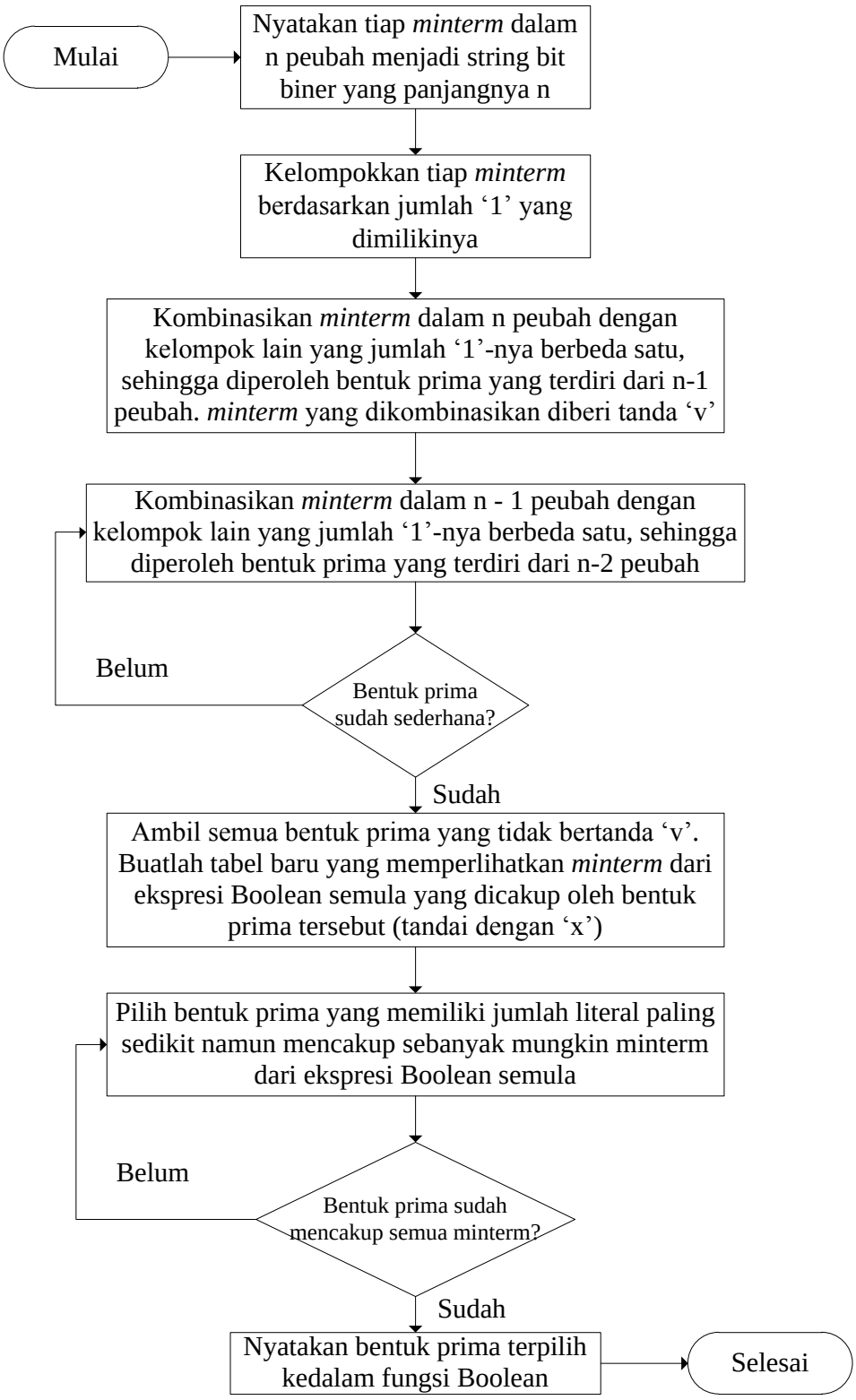


Diagram 3.1 Flowchart Skema Penyederhanaan

3.2 Langkah Penyelesaian Dengan Metode Quine-McCluskey

Secara umum, langkah – langkah metode Quine-McCluskey untuk menyederhanakan fungsi Boolean dalam bentuk SOP adalah sebagai berikut :

1. Nyatakan tiap *minterm* dalam *n* peubah menjadi *string bit* yang panjangnya *n*, yang dalam hal ini peubah komplemen dinyatakan dengan ‘0’, peubah yang bukan komplemen dengan ‘1’.
2. Kelompokkan tiap *minterm* berdasarkan jumlah ‘1’ yang dimilikinya.
3. Kombinasikan *minterm* dalam *n* peubah dengan kelompok lain yang jumlah ‘1’-nya berbeda satu, sehingga diperoleh bentuk prima (*prime-implicant*) yang terdiri dari *n* - 1 peubah. *Minterm* yang dikombinasikan diberi tanda “√”.
4. Kombinasikan *minterm* dalam *n* – 1 peubah dengan kelompok lain yang jumlah ‘1’-nya berbeda satu, sehingga diperoleh bentuk prima yang terdiri dari *n* – 2 peubah.
5. Teruskan langkah 4 sampai diperoleh bentuk prima yang sesederhanan mungkin.
6. Ambil semua bentuk prima yang tidak bertanda “√”. Buatlah tabel baru yang memperlihatkan *minterm* dari ekspresi Boolean semula yang dicakup oleh bentuk prima tersebut (tandai dengan “×”). Setiap *minterm* harus dicakup oleh paling sedikit satu buah bentuk prima.
7. Pilih bentuk prima yang memiliki jumlah literal paling sedikit namun mencakup sebanyak mungkin *minterm* dari ekspresi Boolean semula. Hal ini dapat dilakukan dengan cara berikut :

a. Tandai kolom – kolom yang mempunyai satu buah tanda "×" dengan tanda "*", lalu beri tanda "√" di sebelah kiri bentuk prima yang berasosiasi dengan tanda "*" tersebut. Bentuk prima ini telah dipilih untuk fungsi Boolean sederhana.
b. Untuk setiap bentuk prima yang telah ditandai dengan "√", beri tanda *minterm* yang dicakup oleh bentuk prima tersebut dengan tanda "√" (di baris bawah setelah "*").
c. Periksa apakah masih ada *minterm* yang belum dicakup oleh bentuk prima terpilih. Jika ada, pilih dari bentuk prima yang tersisa yang mencakup sebanyak mungkin *minterm* tersebut. Beri tanda "√" bentuk prima yang dipilih itu serta *minterm* yang dicakupnya.
d. Ulangi langkah c sampai seluruh *minterm* sudah dicakup oleh semua bentuk prima.

Langkah – langkah penyederhanaan metode Quine-McCluskey juga berlaku untuk penyederhanaan fungsi Boolean dalam bentuk POS. Dapat dilihat bahwa bentuk fungsi *output* selalu sama dengan bentuk fungsi *input,* artinya *input* dalam bentuk SOP akan menghasilkan *output* dalam bentuk SOP, dan demikian pula untuk bentuk POS.

3.3 Contoh Kasus

Agar lebih jelas, perhatikan contoh berikut. Bentuk *input* dalam bentuk SOP :

$f(w, x, y, z) = \sum (0, 1, 3, 4, 5, 6, 8, 10, 11, 15)$.

Langkah – langkah penyederhanaan fungsi Boolean dengan metode Quine-McCluskey adalah sebagai berikut:

LANGKAH-1 : Nyatakan tiap *minterm* dalam n peubah menjadi string bit biner yang panjangnya n. (Pada contoh ini, jumlah peubah adalah 4 sehingga n = 4)

0 = 0000

1 = 0001

3 = 0011

4 = 0100

5 = 0101

6 = 0110

8 = 1000

10 = 1010

11 = 1011

15 = 1111

LANGKAH-2 : Kelompokkan tiap *minterm* berdasarkan jumlah ‘1’ yang dimilikinya.

Hasil Penyelesaian Langkah - 2 :

```
----------------
 term   w x y z
----------------
 0      0 0 0 0  -> jumlah bit '1' = 0 buah
----------------
 1      0 0 0 1  -> jumlah bit '1' = 1 buah
 4      0 1 0 0
 8      1 0 0 0
----------------
 5      0 1 0 1  -> jumlah bit '1' = 2 buah
 6      0 1 1 0
 3      0 0 1 1
 10     1 0 1 0
----------------
 11     1 0 1 1  -> jumlah bit '1' = 3 buah
----------------
 15     1 1 1 1  -> jumlah bit '1' = 4 buah
----------------
```

LANGKAH-3 : Kombinasikan *minterm* dalam n peubah dengan kelompok lain yang jumlah ‘1’-nya berbeda satu, sehingga diperoleh bentuk prima (*prime-implicant*) yang terdiri dari n-1 peubah. *minterm* yang dikombinasikan diberi tanda ‘v’

LANGKAH-4 : Kombinasikan *minterm* dalam n - 1 peubah dengan kelompok lain yang jumlah ‘1’-nya berbeda satu, sehingga diperoleh bentuk prima yang terdiri dari n-2 peubah.

LANGKAH-5 : Teruskan langkah 4 sampai diperoleh bentuk prima yang sesederhana mungkin.

Hasil Penyelesaian Langkah-3, 4 dan 5 :

term	w	x	y	z	
0	0	0	0	0	v
1	0	0	0	1	v
4	0	1	0	0	v
8	1	0	0	0	v
5	0	1	0	1	v
6	0	1	1	0	v
3	0	0	1	1	v
10	1	0	1	0	v
11	1	0	1	1	v
15	1	1	1	1	v

Dikombinasikan menjadi :

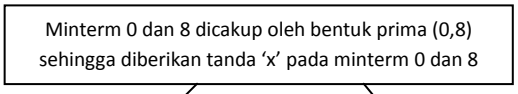
term	w	x	y	z	
0,1	0	0	0	-	v
0,4	0	-	0	0	v
0,8	-	0	0	0	
1,3	0	0	-	1	
1,5	0	-	0	1	v
4,5	0	1	0	-	v
4,6	0	1	-	0	
8,10	1	0	-	0	
3,11	-	0	1	1	
10,11	1	0	1	-	
11,15	1	-	1	1	

Dikombinasikan menjadi :

term	w	x	y	z
0,1,4,5	0	-	0	-
0,4,1,5	0	-	0	-

LANGKAH-6 : Ambil semua bentuk prima yang tidak bertanda ‘v’. Buatlah tabel baru yang memperlihatkan *minterm* dari ekspresi Boolean semula yang dicakup oleh bentuk prima tersebut (tandai dengan ‘x’).

Hasil Penyelesaian Langkah - 6 :

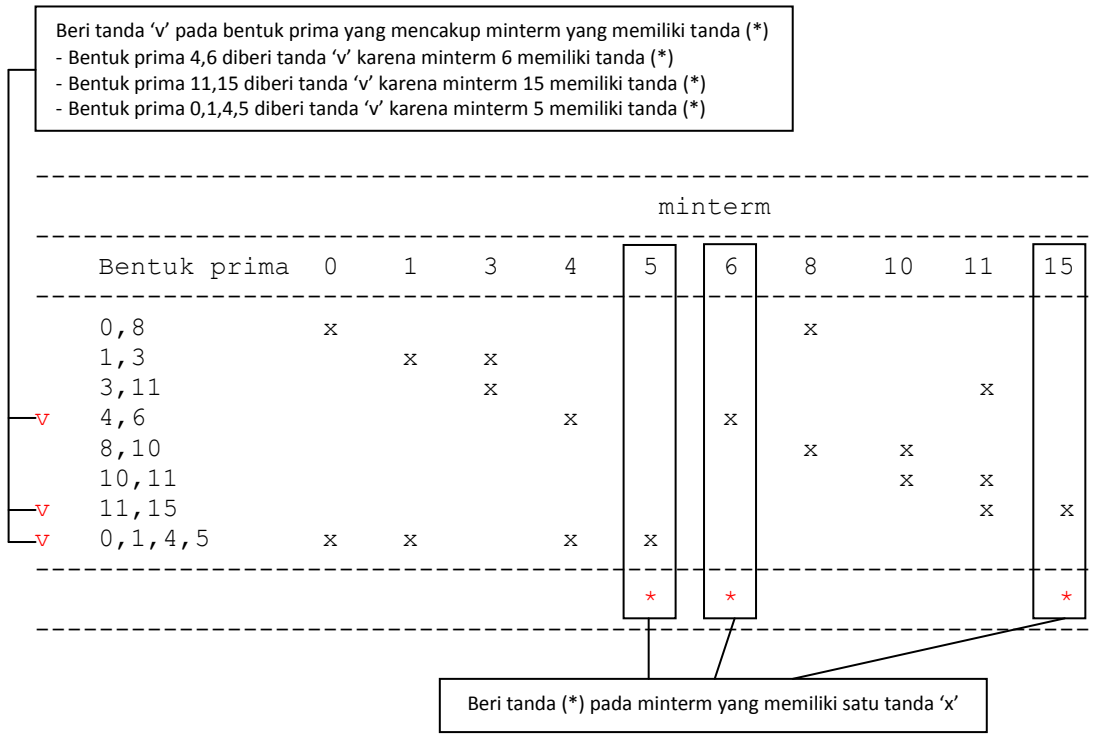


	minterm									
Bentuk prima	0	1	3	4	5	6	8	10	11	15
0,8	x						x			
1,3		x	x							
3,11			x						x	
4,6				x		x				
8,10							x	x		
10,11								x	x	
11,15									x	x
0,1,4,5	x	x		x	x					

LANGKAH-7 : Pilih bentuk prima yang memiliki jumlah literal paling sedikit namun mencakup sebanyak mungkin minterm dari ekspresi Boolean semula. Hal ini dapat dilakukan dengan cara berikut :

LANGKAH-7.A : Tandai kolom-kolom yang mempunyai satu buah tanda ‘x’ dengan tanda (*), lalu beri tanda ‘v’ di sebelah kiri bentuk prima yang mencakup minterm yang mempunyai tanda (*) tersebut. Bentuk prima ini telah dipilih untuk fungsi Boolean sederhana.

Hasil Penyelesaian Langkah-7 dan 7.A :



LANGKAH-7.B : Untuk setiap bentuk prima yang telah ditandai dengan ‘v’, beri tanda *minterm* yang dicakup oleh bentuk prima tersebut dengan tanda ‘v’ (di baris bawah setelah tanda (*)).

Hasil Penyelesaian Langkah-7.B :

		minterm								
	Bentuk prima	0	1	3	4	5	6	8	10	11	15
	0,8	x						x			
	1,3		x	x							
	3,11			x						x	
v	4,6				x		x				
	8,10							x	x		
	10,11								x	x	
v	11,15									x	x
v	0,1,4,5	x	x		x	x					
						*	*				*
		v	v		v	v	v			v	v

Minterm 0,1,4,5 diberi tanda ‘v’ karena dicakup oleh bentuk prima 0,1,4,5 yang memiliki tanda ‘v’

LANGKAH-7.C : Periksa apakah masih ada minterm yang belum memiliki tanda ‘v’ (artinya, belum dicakup oleh bentuk prima terpilih). Jika ada, pilih dari bentuk prima yang tersisa yang mencakup sebanyak mungkin minterm tersebut. Beri tanda ‘v’ untuk setiap bentuk prima yang dipilih itu serta minterm yang dicakupnya.

LANGKAH-7.D : Ulangi langkah 7.C sampai seluruh minterm sudah dicakup oleh bentuk prima

Hasil Penyelesaian Langkah - 7.C dan 7.D :

Sampai tahap ini, masih ada *minterm* yang belum tercakup dalam bentuk prima terpilih, yaitu 3, 8, 10. Untuk mencakup *minterm* tersebut, kita pilih bentuk prima (8,10) dan (1,3) karena mencakup *minterm* 3, 8 dan 10 sekaligus.

		minterm								
	Bentuk prima	0	1	3	4	5	6	8	10	11	15
	0,8	x						x			
v	1,3		x	x							
	3,11			x						x	
v	4,6				x		x				
v	8,10							x	x		
	10,11								x	x	
v	11,15									x	x
v	0,1,4,5	x	x		x	x					
						*	*				*
		v	v	v	v	v	v	v	v	v	v

Sekarang semua minterm sudah tercakup, bentuk prima yang memiliki tanda ‘v’ merupakan hasil minimisasi

Semua minterm sudah tercakup dalam bentuk prima terpilih. Bentuk prima yang terpilih adalah :

1, 3 yang bersesuaian dengan *term* *w'x'z*

(*minterm* 1 (0001) dan *minterm* 3 (0011) memiliki persamaan bit pada posisi 1, 2 dan 4 sehingga dapat dikatakan *minterm* 1 dan *minterm* 3 merupakan fungsi Boolean *w'x'z*).

4, 6 yang bersesuaian dengan *term* *w'xz'*

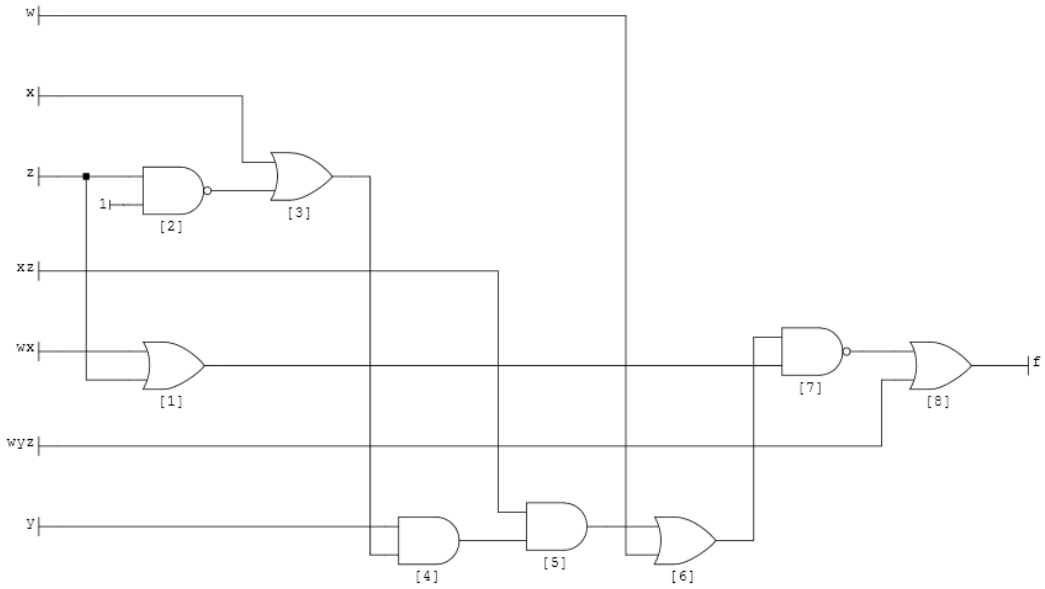
8, 10 yang bersesuaian dengan *term* *wx'z'*

11, 15 yang bersesuaian dengan *term* *wyz*

0, 1, 4, 5 yang bersesuaian dengan *term* *w'y'*

Dari semua langkah – langkah penyederhanaan, hasil minimisasi fungsi Boolean dalam bentuk SOP: $f(w, x, y, z) = \sum (0, 1, 3, 4, 5, 6, 8, 10, 11, 15)$

adalah $f(w, x, y, z) = w'x'z + w'xz' + wx'z' + wyz + w'y'$.



Misalkan *input* fungsi dalam bentuk POS dengan nomor – nomor *maxterm* yang sama dengan nomor – nomor *minterm* pada contoh sebelumnya.

$f(w, x, y, z) = \prod (0, 1, 3, 4, 5, 6, 8, 10, 11, 15)$

Langkah – langkah dan hasil penyederhanaan sama dengan penyederhanaan dalam bentuk SOP, yaitu menghasilkan bentuk prima terpilih (1,3), (4,6), (8,10), (11,15) dan (0,1,4,5). Perbedaannya hanya terletak pada bentuk *output*. Bentuk prima terpilih diubah ke bentuk POS (sama dengan bentuk *input*).

1, 3 bersesuaian dengan *term* $(w + x + z')$.

4, 6 yang bersesuaian dengan *term* $(w + x' + z)$.

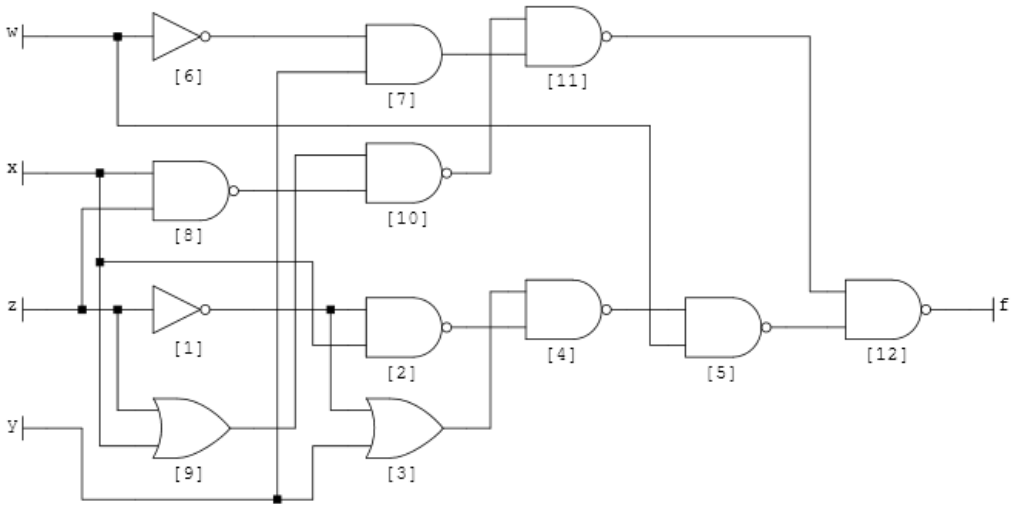
8, 10 yang bersesuaian dengan *term* $(w' + x + z)$.

11, 15 yang bersesuaian dengan *term* $(w' + y' + z')$.

0, 1, 4, 5 yang bersesuaian dengan *term* $(w + y)$.

Dengan demikian, hasil minimisasi fungsi Boolean dalam bentuk POS adalah

$f(w, x, y, z) = (w + x + z')(w + x' + z)(w' + x + z)(w' + y' + z')(w + y)$.



3.4 Perancangan Sistem

Dalam tahap perancangan proses terdapat beberapa proses yang dilakukan untuk mendapatkan sebuah hasil yang paling sederhana, yang kemudian dapat digunakan untuk penggambaran rangkaian logika digital yang biasa digunakan pada elektronika digital (seperti video game, oven gelombang mikro, bagian sistem kendaraan bermotor, dan berbagai alat ukur seperti multimeter dan ostiloskop).

Proses-proses yang dilakukan untuk mendapatkan hasil-hasil tersebut adalah proses penyederhanaan menggunakan metode Quine-McCluskey. Sebelum pelaksanaan proses penyederhanaan, terlebih dahulu didefinisikan kebutuhan-kebutuhan yang menjadi masukan, yaitu variabel dalam bentu SOP (Sum Of Product) atau POS (Product Of Sum) dan jumlah variabel masukan. Untuk masukan yang masih dalam bentuk persamaan (belum dipresentasikan kedalam bilangan biner), variabel komplemen (misal : x'y') ditandai dengan menggunakan tanda aksen (') dan variabel bukan komplemen (misal :x,y) tetap. Untuk menjelaskan proses-proses yang ada, digunakan bantuan Diagram Konteks yang ditunjukan pada gambar berikut ini :

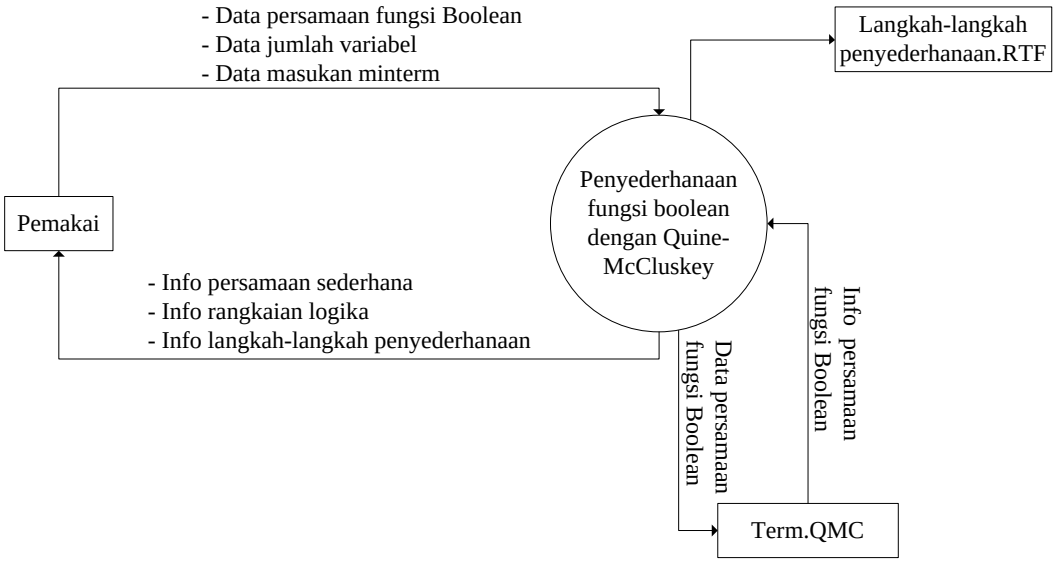


Diagram 3.2 Diagram Konteks

Diagram 3.2 memperlihatkan gambaran secara global dari proses penyederhanaan dengan menggunakan metode Quine-McCluskey, dimana diperlihatkan hubungan antara pemakai dengan proses penyederhanaan masukan dari pemakai berupa persamaan fungsi boolean dan jumlah variabelnya sedangkan dari proses akan menghasilkan keluaran berupa persamaan sederhana.

Berdasarkan diagram konteks pada gambar diagram 3.2, didapat suatu analisa Data Flow Diagram berikut ini:

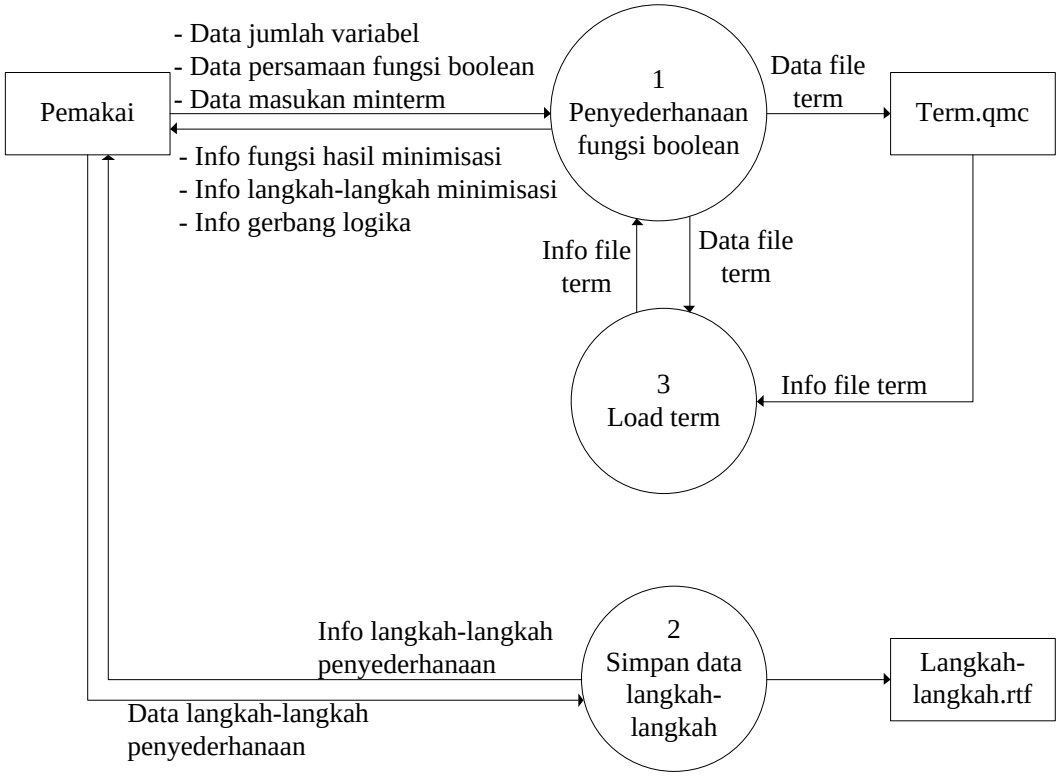


Diagram 3.3 DFD Level 1

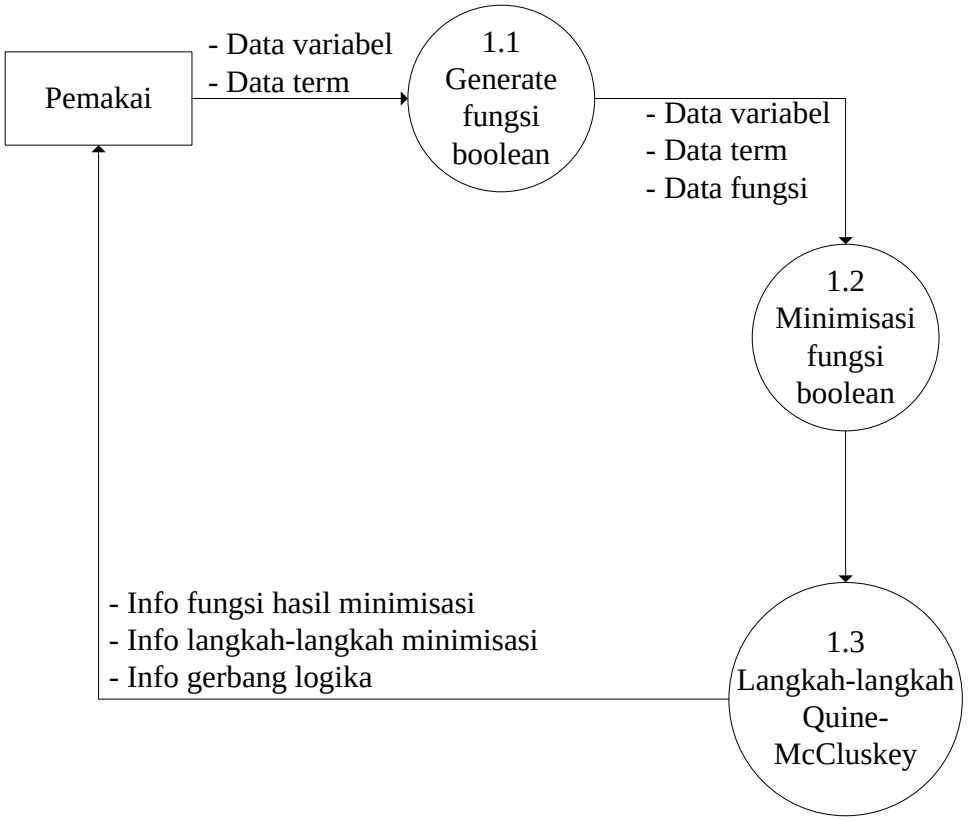


Diagram 3.4 DFD Level 2

Struktur menu yang dirancang pada perangkat lunak penyederhanaan fungsi Boolean dengan metode Quine-McCluskey ini adalah sebagai berikut:

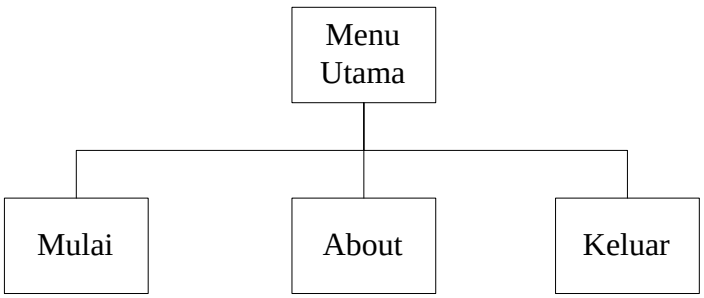
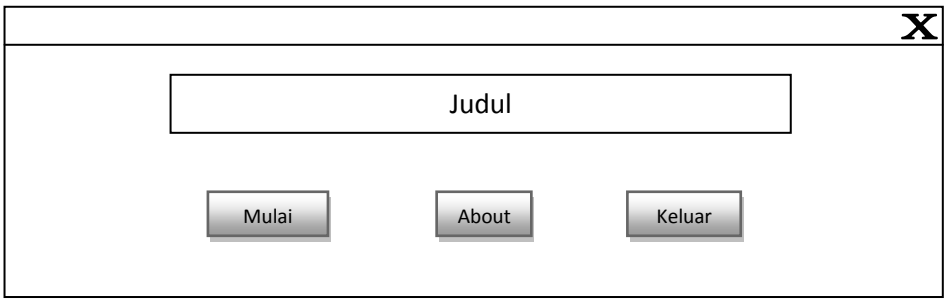


Diagram 3.5 Struktur menu

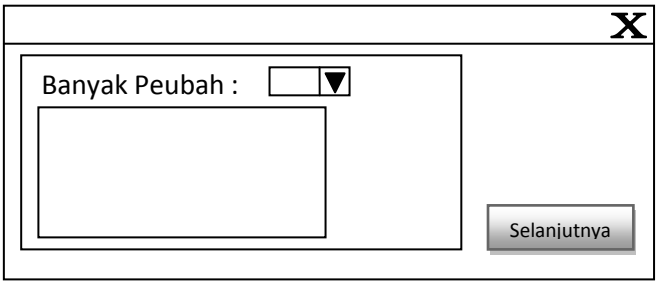
3.5 Perancangan Antarmuka

Tampilan antarmuka utama:



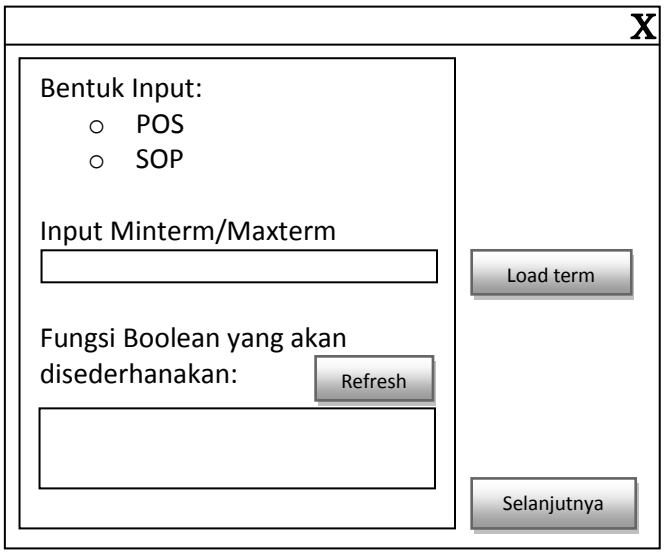
Gambar 3.6 Tampilan menu utama

Tampilan antarmuka input variabel:



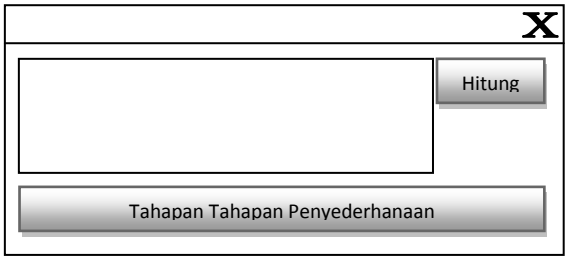
Gambar 3.7 Tampilan input variabel

Tampilan antarmuka input bentuk kanonik:



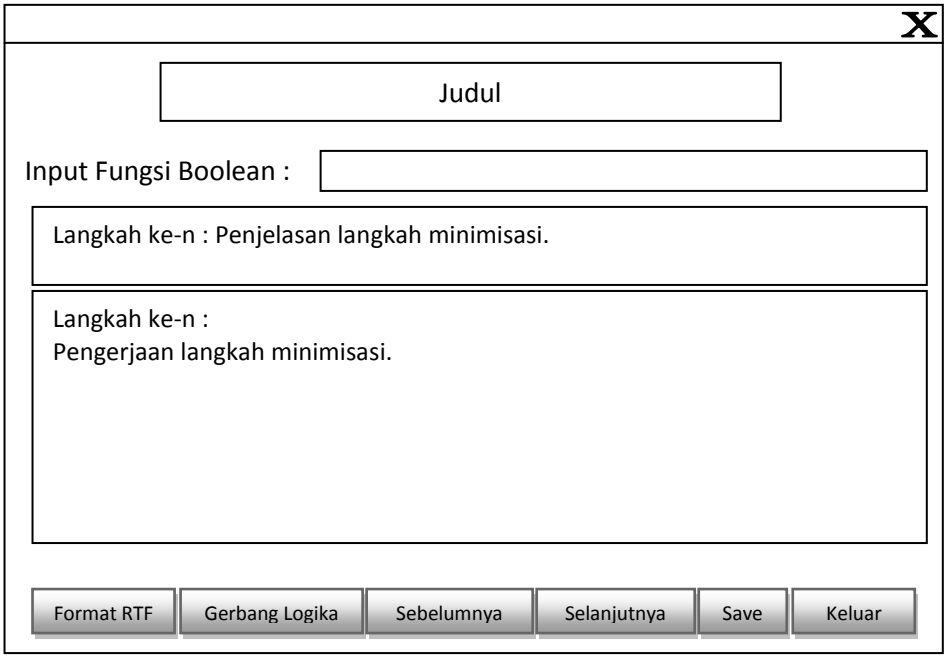
Gambar 3.8 Tampilan menentukan bentuk kanonik dan input term

Tampilan antarmuka hitung minimisasi:



Gambar 3.9 Tampilan hasil minimisasi

Tampilan antarmuka Tahapan-tahapan penyederhanaan:



Gambar 3.10 Tampilan langkah-langkah minimisasi