

BAB II

LANDASAN TEORI

2.1 Sistem Bilangan

Sistem bilangan tidak lain adalah merupakan suatu sandi bagi masing-masing kuantitas yang bersangkutan. Setelah menghafalkan sandi, yang bersangkutan dapat mencacah. Dalam hal ini mengarah kepada aritmatika maupun bentuk-bentuk matematika yang lebih tinggi.

Sistem bilangan yang paling umum dan banyak digunakan pada saat ini adalah sistem desimal yang menggunakan sepuluh lambang bilangan yaitu 0, 1, 2, 3, 4, 5, 6, 7, 8, dan 9. Penerapan bilangan yang ingin dinyatakan hanya menggunakan kombinasi dari kesepuluh angka tersebut. Selain sistem desimal, terdapat sistem bilangan lain yang umum digunakan dalam bidang teknik digital yaitu sistem biner, sistem oktal dan sistem heksadesimal yang masing-masing memiliki basis 2, basis 8 dan basis 16. Basis dari bilangan menunjukkan banyaknya lambang bilangan yang digunakan dan biasanya dinyatakan dalam subscript, misalnya seperti 190_{10} , 1901_{10} , 100102_{10} , 4038_{10} , $1A6BF16_{16}$

2.1.1 Sistem Bilangan Desimal dan Biner

Sistem bilangan desimal adalah sistem bilangan yang biasa digunakan dalam kehidupan sehari-hari yaitu sistem bilangan dengan basis 10 yang simbol bilangannya adalah 0, 1, 2, 3, 4, 5, 6, 7, 8, dan 9, penggunaan 0 sampai 9 tidak merupakan keharusan. Bagaimanapun juga, karena sistem bilangan hanyalah

suatu sandi dapat menggunakan berapapun banyaknya lambang yang kita inginkan. Sistem bilangan biner adalah suatu sandi yang hanya menggunakan dua lambang dasar, lambang yang lazim digunakan untuk sistem biner ini adalah 0 dan 1.

Suatu bilangan dapat dikonversikan kedalam bentuk sistem bilangan yang lainnya, hal ini paling mudah dapat dipahami dengan cara melihat secara langsung dari contoh-contoh sebagai berikut:

1. Konversi desimal ke biner

$$\begin{array}{r}
 176 = \dots\dots\dots 2 \\
 176 : 2 = 88 \quad \text{sisa} \quad 0 \quad (\text{LSB}) \\
 88 : 2 = 44 \quad \text{sisa} \quad 0 \\
 44 : 2 = 22 \quad \text{sisa} \quad 0 \\
 22 : 2 = 11 \quad \text{sisa} \quad 0 \\
 11 : 2 = 5 \quad \text{sisa} \quad 1 \\
 5 : 2 = 2 \quad \text{sisa} \quad 1 \\
 2 : 2 = 1 \quad \text{sisa} \quad 0 \\
 1 : 2 = 0 \quad \text{sisa} \quad 1 \quad (\text{MSB})
 \end{array}$$

$$\text{Jadi } 176_{10} = 10110000_2$$

2. Konversi biner ke desimal

$$\begin{aligned}
 10101101_2 &= (1 \times 2^7) + (1 \times 2^5) + (1 \times 2^3) + (1 \times 2^2) + (1 \times 2^0) \\
 &= 128 + 32 + 8 + 4 + 1 \\
 &= 173
 \end{aligned}$$

$$\text{Jadi } 10101101_2 = 173_{10}$$

2.1.2 Sistem Bilangan Heksadesimal

Sistem bilangan heksadesimal mempunyai basis 16, walaupun dapat digunakan 16 buah lambang yang manapun, namun lazimnya digunakan 0 sampai 9 dan A sampai F, seperti terlihat pada Tabel 2.1. setelah mencapai 9 pada sistem heksadesimal pencacahan dilanjutkan dengan A, B, C, D, E, F.

Setelah kehabisan lambang dasar, dapat dibentuk kombinasi 2 angka, dengan mengambil angka kedua diikuti oleh angka pertama, kemudian angka kedua diikuti oleh angka kedua dan seterusnya.

Tabel 2.1 Angka-angka Heksadesimal

Desimal	Biner	Heksadesimal
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F

1. Konversi heksadesimal ke desimal

Untuk mengubah bilangan heksadesimal ke ekivalen desimalnya adalah dengan cara menjumlahkan hasil kali antara angka-angka heksadesimal bobot-bobotnya. Salah satu contohnya adalah sebagai berikut:

$$\begin{aligned} F8E6 &= F(16^3) + 8(16^2) + E(16^1) + 6(16^0) \\ &= 15(16^3) + 8(16^2) + 14(16^1) + 6(16^0) \\ &= 63718_{10} \end{aligned}$$

2. Konversi desimal ke heksadesimal

Suatu cara untuk melaksanakan perubahan desimal ke heksadesimal adalah mengubah dari desimal ke biner dan kemudian ke heksadesimal. Jika dikehendaki perubahan secara langsung, dapat digunakan metode *hex-dabble* (heksa-plus sisa). Gagasannya adalah membagi secara berturut-turut dengan 16 sambil menuliskan sisa-sisanya.

Berikut diperlihatkan contoh cara pengerjaannya yaitu mengubah 2479 ke heksa desimal:

$$2479 : 16 = 154 \text{ sisa } 15 = F \text{ (LSB)}$$

$$154 : 16 = 9 \text{ sisa } 10 = A$$

$$9 : 16 = 0 \text{ sisa } 9 = 9 \text{ (MSB)}$$

Jadi heksadesimal dari 2479 adalah 9AF

3. Konversi Heksadesimal ke Biner

Untuk mengubah bilangan heksadesimal ke bilangan biner adalah dengan mengubah masing-masing angka heksadesimal kedalam 4 bitnya dengan

menggunakan sandi pada Tabel 2.1. Sebagai contoh, berikut adalah perubahan 9AF ke biner:

9	A	F
1001	1010	1111

Jadi ekivalen biner dari 9AF adalah 1001 1010 1111

4. Konversi Biner ke Heksadesimal

Untuk mengubah dari biner ke heksadesimal juga dapat digunakan Tabel 2.8 sebagai contoh heksadesimal dari biner 1000 1100 adalah :

1000	1100
8	C

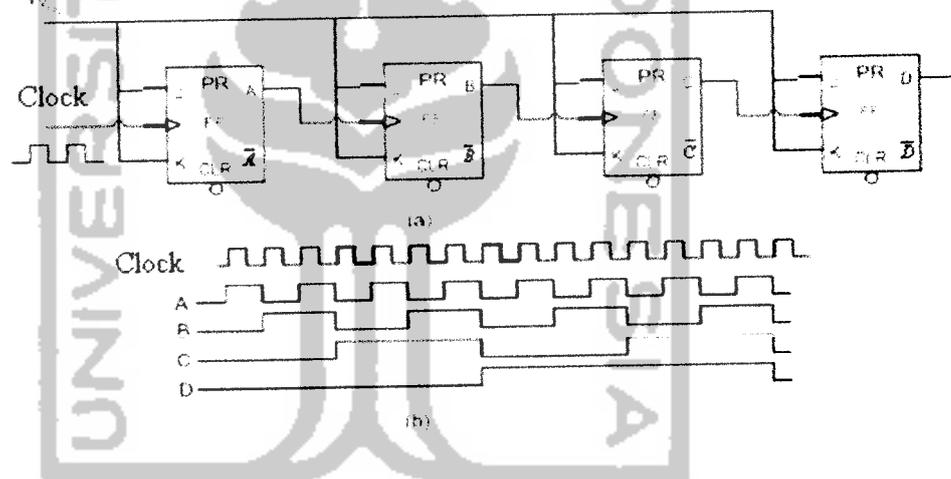
Jadi ekivalen heksadesimal dari biner 1000 1100 adalah 8C.

2.2 Teknik Pencacah

Pencacah merupakan salah satu sub sistem yang paling berguna dan paling banyak kemampuannya dalam suatu sistem digital. Pencacah yang di *drive* oleh sebuah *clock* dapat digunakan untuk mencacah banyaknya siklus *clock*. Karena pulsa *clock* terjadi pada selang waktu yang diketahui, pencacah dapat digunakan sebagai suatu instrumen untuk mengukur waktu (dan dengan demikian periode atau frekuensi).

2.2.1 Pencacah Biner 4-Bit

Flip-flop dapat dihubungkan untuk mendapatkan sebuah pencacah elektronik; suatu unit yang mencacah banyaknya picu masukan. Gambar 2.8 memperlihatkan empat buah flip-flop dalam gandengan. Suatu gelombang segi empat mendrive flip-flop A. Gelombang ini disebut *clock*. Keluaran flip-flop A mendrive flip-flop B, flip-flop B selanjutnya mendrive flip-flop C, yang selanjutnya mendrive flip-flop D. Semua masukan J dan K dihubungkan ke +Vcc. Ini berarti masing-masing flip-flop akan berubah keadaan (*toggle*) akibat peralihan negatif pada masukan *clock*-nya.

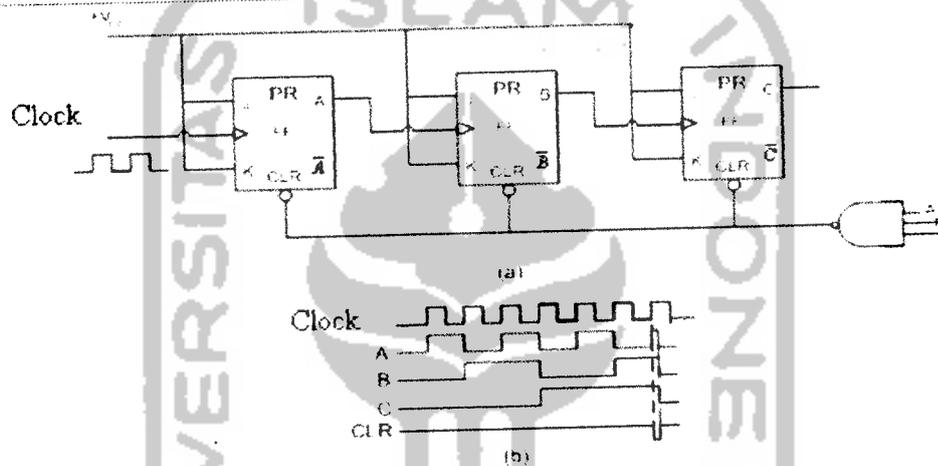


Gambar 2.1 (a) Pencacah empat bit; (b) Bentuk gelombang.

Bila keluaran suatu flip-flop lainnya, dinamakan *Ripple counter* atau pencacah tak serempak; flip-flop A harus berubah keadaan sebelum dapat memicu flip-flop B; B harus berubah sebelum dapat memicu C, dan seterusnya.

2.2.2 Pencacah Tak Serempak

Salah satu metode yang digunakan untuk membuat sebuah pencacah melompati cacahan-cacahan tertentu adalah dengan mencatu-balikkan suatu sinyal dari beberapa flip-flop tertentu ke flip-flop yang mendahuluinya. Diperhatikan pencacah yang terlihat pada Gambar 2.2, karena hanya sebuah cacahan yang harus dilompati, akan sangat memudahkan bila kita membuat pencacah melewati urutan dasarnya dan kemudian meresetnya satu cacahan lebih awal.



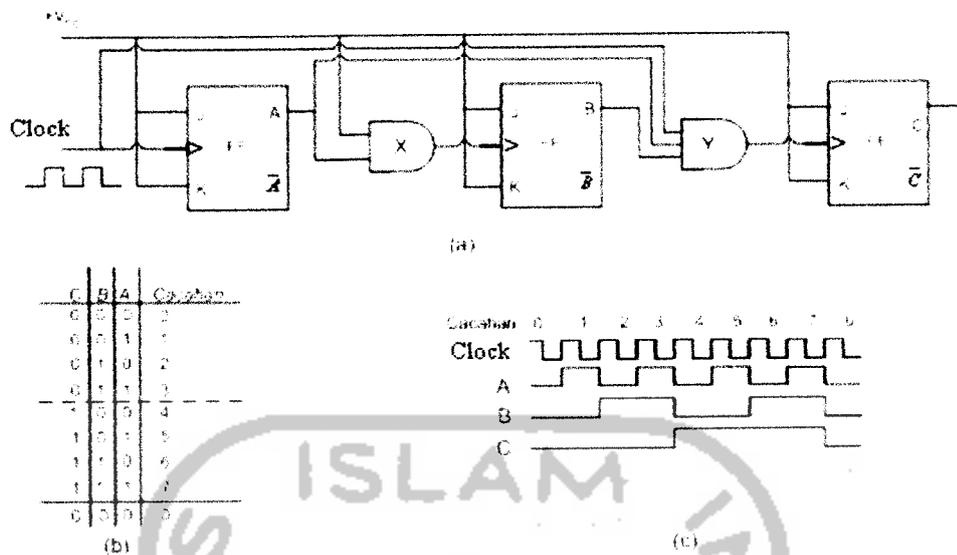
Gambar 2.2 (a) Pencacah Tak Serempak; (b) Bentuk gelombang

Jika A, B, dan C dihubungkan ke sebuah gerbang NAND seperti terlihat pada Gambar 2.2 keluaran gerbang NAND akan menjadi rendah hanya pada saat $A = 1$, $B = 1$, dan $C = 1$ (cacahan 7). Jika keluaran gerbang NAND ini dihubungkan ke CLR pada masing-masing flip-flop, semua flip-flop akan direset ke nol secara langsung ketika pencacah mencapai cacahan 7. Jenis pencacahan ini kadang-kadang disebut “pencacah permutasi” karena cacahan dasarnya telah dipermutasi (diubah).

2.2.3 Pencacah Paralel

Ripple Counter paling sederhana pembuatannya, namun terdapat suatu keterbatasan pada frekuensi kerja maksimumnya. Pada *ripple counter* waktu-waktu tunda ini bersifat aditif (ditambah), dan waktu penyelesaian keseluruhan bagi pencacah yang bersangkutan adalah sekitar waktu tunda dikalikan dengan banyaknya flip-flop seluruhnya. Keterbatasan kecepatan ini dapat ditanggulangi dengan penggunaan pencacah serempak atau paralel. Perbedaannya dalam hal ini adalah bahwa setiap flip-flop dipicu oleh *clock*. Maka, semuanya mengalami peralihan secara serempak.

Konstruksi suatu pencacah biner paralel diperlihatkan pada Gambar 2.3 bersama tabel kebenaran dan bentuk gelombangnya bagi urutan cacahan dasar. Flip-flop A pada gambar 2.3 berubah keadaan oleh setiap peralihan negatif pada masukan lonceng. Keluaran gerbang AND X menjadi tinggi bilamana *clock* adalah tinggi dan A adalah tinggi. Maka flip-flop B berubah keadaan pada setiap pulsa *clock* berikutnya. Keluaran gerbang AND Y menjadi tinggi setiap kali lonceng serta A dan B menjadi tinggi. Maka flip-flop C berubah keadaan oleh setiap pulsa *clock* keempat. Semuanya ini mewujudkan sebuah pencacah biner paralel mod-8



Gambar 2.3 Pencacah biner paralel mod-8 (a) Diagram logika; (b) Tabel kebenaran; (c) bentuk gelombang

Pencacah paralel yang terlihat pada Gambar 2.3 dapat digunakan sebagai dasar untuk membangun pencacah dengan modul lain.

2.2.4 Pencacah BCD

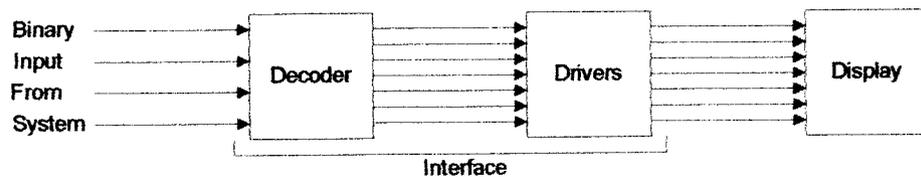
Untuk membangun pencacah-pencacah yang bukan mencacah dalam sandi biner langsung. Suatu bentuk lain yang sangat bermanfaat adalah pencacah dekade yang mencacah dalam desimal bersandi biner termodifikasi 2421. Pencacah ini diperlihatkan pada Gambar 2.4 operasi pencacah ini dapat dijelaskan sebagai berikut:

1. A harus berubah keadaan setiap kali pulsa lonceng menurun, dan dengan demikian masukan *clock* ke *flip-flop* A haruslah *clock* itu sendiri.

2. B harus berubah keadaan setiap kali A menurun kecuali pada peralihan dari cacahan 7 ke cacahan 8. Keluaran gerbang NAND X adalah rendah bilamana C dan \bar{D} keduanya tinggi. Hal ini mencegah B agar tidak menurun pada peralihan dari cacahan 7 ke cacahan 8, karena keluaran gerbang NAND X menahan masukan X ke flip-flop B tetap rendah sepanjang cacahan 4,5,6, dan 7. Dapat diperhatikan bahwa masukan J ke flip-flop B selalu TINGGI (benar), dan dengan demikian B dapat menjadi TINGGI pada peralihan dari cacahan 5 ke cacahan 6.
3. Masukan *clock* ke *flip-flop* C didrive oleh B, karena C harus berubah keadaan setiap kali B menurun.
4. D harus berubah keadaan setiap kali B dan C keduanya TINGGI dan A menurun. Jika masukan J dan K ke *flip-flop* D ditentukan oleh keluaran sebuah gerbang AND yang masukan-masukannya adalah B dan C, maka D dapat berubah keadaan bila B dan C keduanya TINGGI. Selanjutnya jika masukan lonceng ke *flip-flop* D didrive oleh A, D akan berubah keadaan pada peralihan dari 7 ke 8 dan dari 9 ke 0

Pencacah ini merupakan pencacah dekade, dan oleh sebab itu hanya memiliki 10 keadaan. Karena digunakan empat buah *flip-flop*, maka terdapat enam keadaan tak absah (diabaikan) yang harus dikaji. Pencacah BCD yang terlihat pada Gambar 2.4 mewakili salah satu metode untuk mewujudkan pencacah ini.

sebelum ke piranti tampilan. Dengan demikian, untai antar muka terdiri dari dua komponen dasar: pengawa-sandi dan penggerak.



Gambar 2.5 Sistem tampilan dasar

2.3 Aritmatik Biner

2.3.1 Penambahan Biner

Penambahan bilangan biner merupakan suatu tugas yang sangat sederhana. Aturan – aturan (tabel-tabel penambahan) untuk penambahan biner yang menggunakan dua bit, dipaparkan pada gambar 2.6. Aturan 4 mengatakan bahwa, dalam biner $1+1 = 10$ (desimal 2). Angka 1 dalam penambahan tersebut harus dibawa ke kolom berikutnya seperti dalam kolom penambahan desimal yang biasa.

			Jumlah	Bawaan keluar
Aturan 1	0	+	0 = 0	
Aturan 2	0	+	1 = 1	
Aturan 3	1	+	0 = 1	
Aturan 4	1	+	1 = 0	Dan bawaan 1 = 10

Simbol + berarti tambah

Gambar 2.6 Aturan untuk penambahn biner

Dua contoh persoalan penambahan biner diilustrasikan sebagai berikut :

$$\begin{array}{r}
 1 \ 0 \ 0 \ 4 \\
 +0 \ 1 \ 0 \ +2 \\
 \hline
 \text{(jumlah)} \ 1 \ 1 \ 0 \ 6 \ \text{(desimal)}
 \end{array}$$

$$\begin{array}{r}
 \begin{array}{c} \leftarrow \\ \leftarrow \\ \leftarrow \end{array} \\
 \begin{array}{r}
 1 \ 0 \ 1 \ 4 \\
 + \ 0 \ 1 \ 1 \ +2 \\
 \hline
 \text{(jumlah)} \ 1 \ 1 \ 0 \ 0 \ 6 \ \text{(desimal)}
 \end{array}
 \end{array}$$

2.3.2 Pengurangan Biner

Aturan-aturan untuk pengurangan biner dari dua bit diilustrasikan pada gambar 2.7. Angka sebelah atas dalam suatu persoalan pengurangan disebut *minumen* (yang dikurangi). Angka sebelah bawah disebut pengurang (*subtrahend*), dan jawabannya disebut selisih (*difference*). Perhatikan kolom 1 dari bilangan biner pada gambar 2.8, angka 1 dikurangkan dari 0. angka 1 dipinjam dari kolom 2 biner, sehingga tinggal 0 pada kolom tersebut. Pengurang 1 dikurangkan dari minuen 10 (desimal 2). Pengurang ini menghasilkan selisih 1 pada kolom 1. kolom 2 biner menggunakan aturan 1 (0-0) dan sama dengan 0. maka, aturan 2 adalah $0 - 1 = 1$ dengan pinjaman 1.

	Minuend		Subtrahend		Selisih	Pinjaman keluar
Aturan 1	0	-	0	=	0	
Aturan 2	0	-	1	=	1	Pinjam 1
Aturan 3	1	-	0	=	1	
Aturan 4	1	-	1	=	0	

Gambar 2.7 Aturan untuk pengurangan biner

	Biner	Desimal
	pinjam	
Minuend	10	10
	- 0	1
Subtrahend	- 0	1
Selisih	0 1	1

Gambar 2.8 Persoalan pengurangan biner yang menunjukkan suatu pinjaman

2.3.3 Perkalian Biner (*Binary Multiplication*)

Aturan untuk perkalian biner adalah sangat sederhana. Aturan – aturan ditunjukkan pada gambar 2.9. Perkalian biner hanyalah seperti perkalian desimal belaka. Gambar 2.10 mengilustrasikan suatu persoalan perkalian biner yang sederhana.

0	0	1	1
x 0	x 1	x 0	x 1
0	0	0	1

Gambar 2.9 Aturan – Aturan untuk perkalian Biner

a	13	B	1101	Multiplikan
	x 10		x 1010	Pengali
	00		0000	Hasil Parsial ke 1
	13		1101	Hasil Parsial ke 2
	130		0000	Hasil Parsial ke 3
			1101	Hasil Parsial ke 4
			1000010	Hasil

Gambar 2.10 Persoalan perkalian biner sederhana

Persoalan perkalian desimal pada sebelah kiri gambar 2.10 menunjukkan metode tradisional pengerjaan operasi ini dengan tangan. Pada sebelah kanan gambar 2.10, metode yang sama digunakan untuk bilangan biner. Biner 1101 (multiplikand, multiplicand) dikalikan dengan 1010 (pengali, multiplier). Prosedur tersebut ialah, mula-mula mengalikan bit 1 dari pengali (0) dengan multiplikand. Hasilnya berupa hasil parsial ke 1 yaitu 0000. kedua bit 2 (1) pada gambar 2.10 dikalikan dengan pengali. Hasilnya berupa hasil parsial ke 2, yaitu 1101. perlu diperhatikan bahwa hasil parsial ke 2 tersebut digeser satu posisi ke sebelah kiri dari hasil parsial ke 1. ketiga, bit 4 (0) dikalikan dengan pengali. Hasilnya adalah 0000, yang muncul sebagai hasil parsial ke 3. Keempat, bit 8 pada pengali, dikalikan dengan pengali, yang sama yaitu 1101. hasil ini dicatat pada posisi hasil parsial ke 4. Akhirnya, empat hasil parsial tersebut ditambahkan sehingga menghasilkan hasil akhir 10000010 desimal (130). Jika diubah kedalam bentuk heksadesimal maka memiliki hasil 82, yaitu:

$$\begin{aligned} 82 &= (8 \times 16^1) + (2 \times 16^0) \\ &= 128 + 2 \\ &= 130 \end{aligned}$$

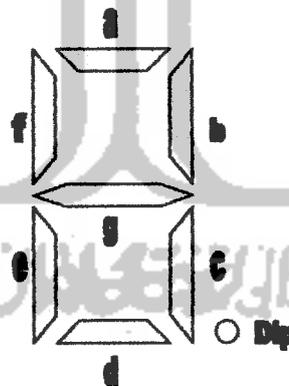
2.4 Tombol Masukan

Sebagai media untuk memasukkan data pada unit aritmatik ini digunakan tombol-tombol input yang keseluruhan berjumlah 16 tombol. 16 tombol tersebut terdiri atas 10 tombol untuk angka yaitu "0" s/d "9", 4 tombol untuk jenis operasi yaitu "+", "-", "x", 1 tombol "=" dan 1 tombol "clear". Semua tombol berupa

saklar push on tetapi bentuknya berbeda. Tombol bilangan berbentuk bulat sedangkan tombol operator, tombol “=” dan tombol “clear” berbentuk kotak. Tombol push on adalah tombol yang awalnya off, jika ditekan akan on, dan akan off kembali jika dilepas.

2.5 Penampil Tujuh Segmen

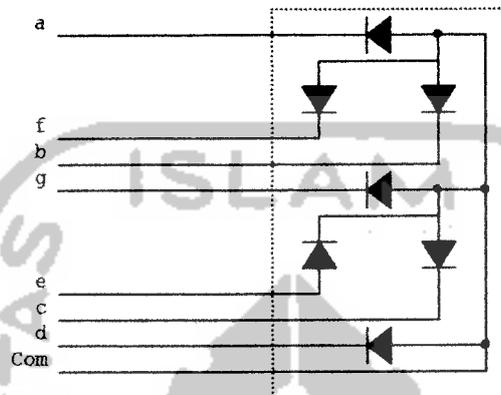
Penampil tujuh segmen ada 2 jenis, yaitu anoda bersama dan katoda bersama. Sebenarnya kedua jenis penampil tujuh segmen tersebut memiliki karakteristik yang hampir sama hanya berbeda penggabungan kaki-kaki led. Pada skripsi ini digunakan penampil tujuh segmen anoda bersama. Penampil tujuh segmen anoda bersama terdiri dari tujuh segmen terpisah yang diberi label a sampai g seperti yang disajikan pada gambar 2.11. tujuh segmen merupakan cacah segmen minimum yang diperlukan untuk menampilkan angka 0 sampai 9 seperti diilustrasikan pada gambar 2.12. rangkaian penampil tujuh segmen anoda bersama dapat dilihat pada gambar 2.13.



Gambar 2.11 Penampil tujuh segmen



Gambar 2.12 Tujuh segmen dalam digit desimal



Gambar 2.13 Rangkaian penampil tujuh segmen anoda bersama

2.6 Field Programmable Gate Array (FPGA)

Field-Programmable Gate Array (FPGA) adalah salah satu piranti yang termasuk dalam kelompok *programmable logic Devices*. Teknologi IC FPGA diperkenalkan pada tahun 1985 oleh perusahaan semikonduktor Xilinx. FPGA adalah sebuah konsep teknologi IC yang dapat diprogram dan dihapus seperti halnya RAM. FPGA kemudian berkembang pesat, baik dari segi kepadatan gerbang, kecepatan dan disertai dengan penurunan harga jual.

Penemuan FPGA telah membuat peningkatan yang pesat akan pembuatan prototype beberapa system digital. Salah satu produsen yang ada dipasaran adalah Xilinx. Disamping produsen lainnya Actel dan Altera. Prinsip dasar dari

pemrograman/pengkonfigurasi FPGA Xilinx adalah perubahan gambar rangkaian elektronika digital dari perangkat lunak Xilinx menjadi file aliran bit (bitstream) dan dikonfigurasi (didownload) ke dalam IC FPGA Xilinx tersebut sehingga IC tersebut terkonfigurasi secara perangkat keras yang dirancang dalam perangkat lunak Xilinx. FPGA produk xilinx sudah melewati beberapa generasi antara lain XC2000, XC3000 dan XC4000. tiap generasi memiliki sifat dan kemampuan yang berbeda. Perbedaan tersebut meliputi kecepatan, kapasitas gerbang logika.

2.6.1 FPGA keluarga Xilinx Spartan II

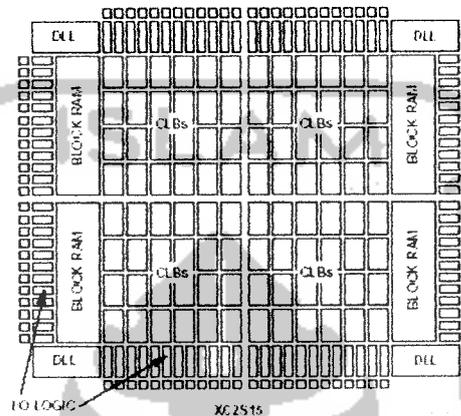
Spartan II merupakan salah satu keluarga FPGA yang dikeluarkan oleh Xilinx. Xilinx merupakan salah satu pabrik pembuat FPGA yang cukup terkenal. Keluarga Spartan II merupakan keluarga FPGA yang memiliki 15.000 sampai 200.000 gerbang. IC Xilinx ini dapat diprogram dan dihapus dengan waktu yang tidak terbatas serta murah harganya. Keluarga Spartan ini dapat diprogram dengan mudah menggunakan *Xilinx Development System* ataupun dengan *development system* lain yang dikembangkan oleh para pengguna.

Tabel 2.2 Data Keluarga Spartan II

Device	Logic Cells	System Gates (Logic and RAM)	CLB Array (R x C)	Total CLBs	Maximum Available User I/O ⁽¹⁾	Total Distributed RAM Bits	Total Block RAM Bits
XC2S15	432	15.000	8 x 12	96	86	6.144	16K
XC2S30	972	30.000	12 x 18	216	132	13.824	24K
XC2S50	1.728	50.000	16 x 24	384	176	24.576	32K
XC2S100	2.700	100.000	20 x 30	600	196	38.400	40K
XC2S150	3.888	150.000	24 x 36	864	260	55.296	48K
XC2S200	5.292	200.000	28 x 42	1.176	284	75.264	56K

2.6.2 Struktur Dasar Keluarga Spartan II

Suatu piranti FPGA terdiri atas *Configurable Logic Blocks* (CLB), unit input/output, empat buah *Delay-Locked Loops* (DLLs), unit RAM dan unit routing yang dapat diprogram secara otomatis penuh. Susunan dan letak masing-masing bagian tersebut dapat dilihat pada gambar 2.4.



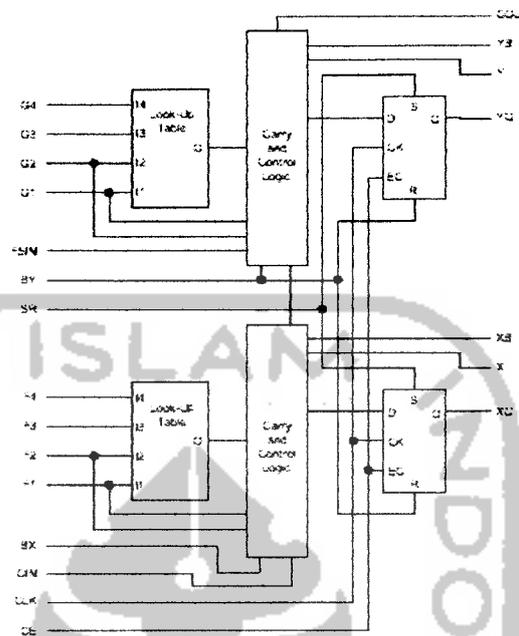
Gambar 2.14 Diagram Blok Dasar Keluarga Spartan II

2.6.3 Input/Output Blocks (IOB)

Input/Output blocks merupakan bagian dari FPGA yang berfungsi menghubungkan FPGA dengan piranti lain yang akan terkoneksi IOB keluarga Spartan II mampu bekerja pada berbagai macam standar I/O seperti TTL, CMOS dan PCI. Kemampuan untuk menyesuaikan dengan berbagai macam I/O didukung dengan kemampuan tiap *pad* I/O untuk ditambahi *pull-up* dan *pull-down resistor*.

Bagian *buffer* pada Spartan II IOB *input path* akan menghubungkan sinyal input yang masuk secara langsung dengan logika internal atau secara tidak langsung melalui input flip-flop optional. Sedangkan bagian *output path*

dalam dua *slices* yang identik. Tiap CLB ini juga memuat logika yang akan mengkombinasi generator pembangkit fungsi logika untuk 4 sampai 6 input.



Gambar 2.16 CLB pada Spartan II

Generator pembangkit fungsi diimplementasikan dalam sebuah *look-up tables* (LUT) 4 input. LUT ini juga dapat membangkitkan sebuah RAM 16 x 1 sinkron serta membangkitkan fungsi *shift register* 16 bit. Fungsi RAM yang dibangkitkan generator ini akan melengkapi *block RAM* yang dimiliki oleh Spartan II sehingga mampu menghasilkan unit penyimpan data yang handal.

2.6.5 Programmable Routing Matrix

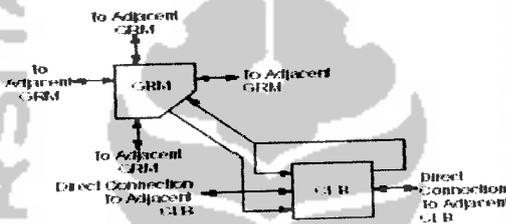
Programmable Routing Matrix merupakan cara sebuah FPGA melakukan *routing* menghubungkan CLB-CLB dan IOB-IOB yang digunakan dalam desain menjadi satu kesatuan sistem. *Routing* ini dilakukan secara otomatis penuh.

Namun untuk keperluan tertentu, optimasi jalur yang paling pendek dapat dilakukan *routing* manual.

Dalam keluarga Spartan II ada beberapa macam *routing* yang bisa digunakan yaitu:

1. *Local Routing*

Local Routing digunakan untuk mengimplementasikan hubungan antara LUT, flip-flop dan *General Routing Matrix* (GRM), antara jalur umpan balik internal CLB dengan LUT lain pada CLB yang sama untuk koneksi *high speed*, serta antara jalur-jalur langsung yang bisa dibuat untuk memperkecil *delay*.



Gambar 2.17 Struktur *Local Routing*

2. *General Purpose Routing*

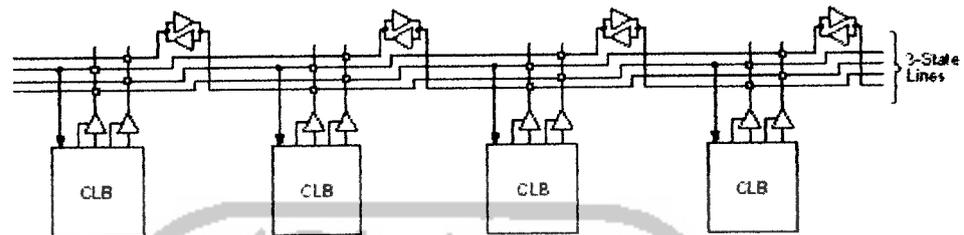
General Purpose Routing digunakan untuk melakukan koneksi vertikal dan horisontal antar kolom dan baris CLB yang digunakan.

3. *I/O Routing*

I/O Routing khusus digunakan untuk menghubungkan *array* pada CLB dengan IOB.

4. *Dedicated Routing*

Dedicated Routing digunakan untuk menghubungkan beberapa CLB, IOB ataupun LUT yang memerlukan perlakuan khusus untuk memaksimalkan performa



Gambar 2.18 Koneksi BUFT untuk *dedicated Horizontal Bus Line*

5. Global Routing

Global Routing digunakan untuk menghubungkan clock dengan bagian yang membutuhkan serta sinyal-sinyal dengan *fan-out* yang tinggi ke bagian lain.

2.6.6 Pemrograman FPGA

Keluarga Spartan II yang dikeluarkan oleh Xilinx dapat dengan mudah diprogram menggunakan *development system software* keluaran Xilinx yaitu Xilinx Foundation Series. Saat ini versi yang terbaru adalah versi 6.i3. *Software* memiliki kemampuan *place and route tools* (PAR) yang memungkinkan penggunanya menyusun IOB, LUT dan CLB menjadi sebuah kesatuan sistem.

FPGA dapat diprogram menggunakan metode *schematic* ataupun menggunakan *Hardware Description Language*, baik Verilog maupun VHDL. Masing-masing metode memiliki kelebihan dan kelemahannya sendiri.

Keluarga Spartan II dapat dioperasikan dalam 4 mode yaitu:

1. *Slave Serial mode*

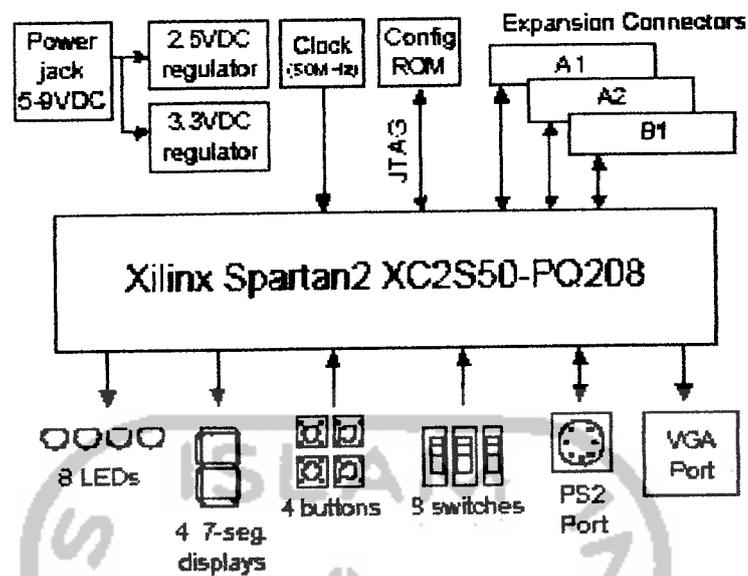
2. *Master Serial mode*
3. *Slave Parallel mode*
4. *Boundary Scan mode*

Mode yang paling mudah digunakan adalah *Boundary Scan Mode* dimana tidak diperlukan koneksi-koneksi khusus, cukup menggunakan kabel paralel yang dikoneksikan menggunakan JTAG, maka desain dapat dengan mudah diimplementasikan ke dalam FPGA.

2.7 Modul Pegasus

Modul pegasus dikeluarkan oleh pengembang ke 3, yaitu DIGILENT. Dengan IC FPGA utama yaitu, Xilinx Spartan2 XC2S50, dengan perangkat lunak Xilinx. Perangkat pendukung yang terdapat pada modul Pegasus diantaranya:

- a. 50k-gate Xilinx Spartan2 FPGA dengan 50k gerbang dasar dan 200MHz operasi *maximum*.
- b. XCF01S Xilinx Flash ROM.
- c. Berbagai macam I/O, termasuk didalamnya delapan LED, empat *seven-segment*, empat saklar *pushbutton*, dan delapan saklar geser.
- d. 50MHz *oscilator* dan satu *pin* untuk *oscilator* tambahan.
- e. PS/2 dan VGA *port*.
- f. *Pin* 96 I/O dibagi dalam 3 bagian/*port*, yaitu: A1, A2, dan B1. yang masing-masing terdiri dari 40 *pin*.
- g. Seluruh I/O *pin* sudah dilengkapi pengamanan.
- h. *Port* pemrogram mode JTAG



Gambar 2.19 Blok diagram Pegasus

Modul Pegasus sudah dapat dibilang sangat komplit karena hanya dengan menggunakan modul ini, sudah dapat mendownload *design* dan menjalankannya tanpa perlu menambah komponen lain, karena telah tersedia beberapa masukan dan keluaran.

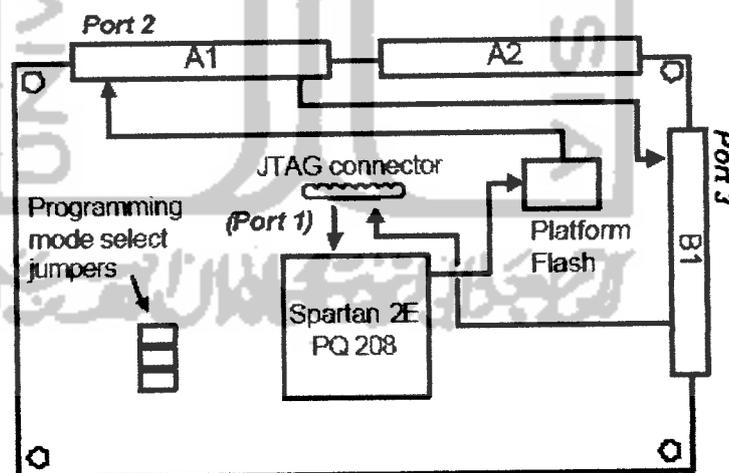
2.7.1 Port JTAG dan Mengkonfigurasi Modul

Pada modul Pegasus ini selain terdapat IC FPGA utama (Spartan2 XC2S50) juga terdapat IC FPGA *secondary* yaitu Spartan2 XCF01S yang berfungsi sebagai IC *Flash* ROM, sehingga apabila ada suatu modul yang dapat diprogram (terdapat IC FPGA) yang terhubung ke modul Pegasus ini, maka dapat dikonfigurasi melalui *port* JTAG (*port1*), dari modul utama.

Dengan menggunakan *port* JTAG dapat diketahui IC tipe, jenis dan dari keluarga FPGA yang ada pada modul utama maupun pada modul lain (tambahan)

yang terhubung ke modul utama melalui *port* tambahan secara *automatic scan chain*.

Scanning dengan menggunakan JTAG sangat mudah karena perangkat lunak Xilinx melakukannya secara otomatis, Xilinx mencari melalui *port* JTAG dan membaca IC FPGA utama, kemudian apabila tidak ada modul lain yang terhubung pada *port* JTAG tambahan *port2* (A1) atau *port3* (B1) maka *buffer* pada modul pegasus menghilangkan keberadaan *port* tambahan tersebut, sedangkan jika ada modul (memiliki IC FPGA) maka *buffer* akan menyatakan bahwa ada modul yang terhubung. Saat *scanning port* JTAG membaca FPGA utama, ke *flash* ROM (XCF01S), lalu membaca modul yang terhubung pada *port* JTAG tambahan, setelah terbaca IC FPGA *secondary* yang terhubung melalui *port* tambahan tersebut, maka dapat dikonfigurasi secara bersamaan melalui 1 kabel utama, baik itu *port* USB, Paralel maupun Ethernet.



Gambar 2.20 Aliran scan JTAG pada Pegasus

2.7.2 *Power Supply*

Modul Pegasus memerlukan 5Vdc *power supply*, sedangkan untuk masukan/keluaran dapat menggunakan *power* dari luar 3Vdc - 5Vdc. *power supply* ini juga dipergunakan untuk mengaktifkan 8 LED, 4 display *seven-segment*, sebagai masukan +5Vdc untuk saklar *pushbutton*, dan saklar geser, serta sumber *clock internal* dan *power port* tambahan.

2.7.3 *Oscillator*

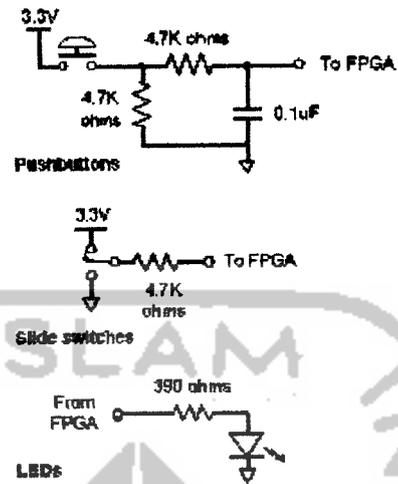
Modul Pegasus menyediakan 50MHz *Oscillator* sebagai clock utama untuk aplikasi-aplikasi yang akan dikonfigurasikan ke Spartan2 XC2S50. *Oscillator* tersebut terhubung langsung ke Spartan2 XC2S50 (*pin 77*).

2.7.4 *Saklar Pushbuttons, Saklar Geser, Indikator Led*

Empat saklar *pushbutton*, dan delapan saklar geser disediakan sebagai masukan. Saklar *pushbutton* dalam keadaan normal menghasilkan 0Vdc (*rendah*), dan akan berubah menjadi 3-5Vdc (*tinggi*) ketika saklar *pushbutton* ditekan. Saklar geser menghasilkan rendah (0Vdc) atau tinggi (3Vdc-5Vdc) secara tetap tergantung dari posisinya. Saklar *pushbutton* sebagai masukan menggunakan rangkaian RC sebagai pengaman dan agar menghasilkan masukan yang stabil, sedangkan saklar geser hanya terhubung dengan resistor secara series.

8 LED disediakan sebagai keluaran anoda LED terhubung ke *pin* keluaran melalui resistor 390 Ω , sedangkan katoda LED terhubung langsung ke *Ground*.

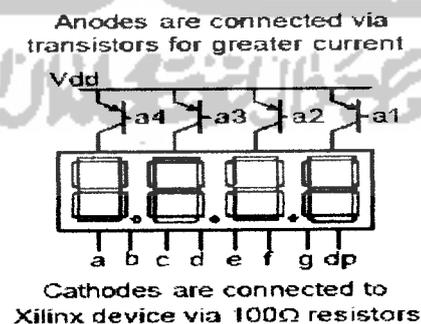
LED ke 9 digunakan sebagai *indikator power* FPGA, dan LED ke 10 digunakan sebagai *indikator* status pemrograman JTAG.



Gambar 2.21 Rangkaian saklar *pushbutton*, saklar geser, LED

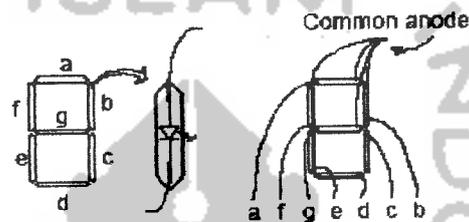
2.7.5 Seven-Segment

Pada modul Pegasus terdapat 4 digit *common anoda seven-segment*. *Display seven-segment* dikonfigurasi secara *multiplexer*, jadi hanya ada 7 masukan katoda untuk mengaktifkan 28 katoda pada 4 *digit seven segment*. 4 *digit* selektor *enable* berfungsi sebagai pengatur *digit* pada *seven-segment*.



Gambar 2.22 *Common anode seven-segment 4 digit*

Ketujuh anoda dari 4 *seven-segment* saling tersambung kedalam selektor “*common anode*”, *display* ini memiliki 4 selektor yang dinamakan AN0 - AN3, apabila ada sinyal/pulsa yang mengaktifkan selektor ini maka *digit* dari selektor tersebut akan *aktif*. Sinyal/pulsa yang digunakan adalah sinyal/pulsa rendah (0Vdc), Katoda dari setiap seven segment terhubung kedalam 7 keluaran, dan diberi nama CA - CG, dan akan *aktif* apabila ada sinyal/pulsa rendah (0Vdc).

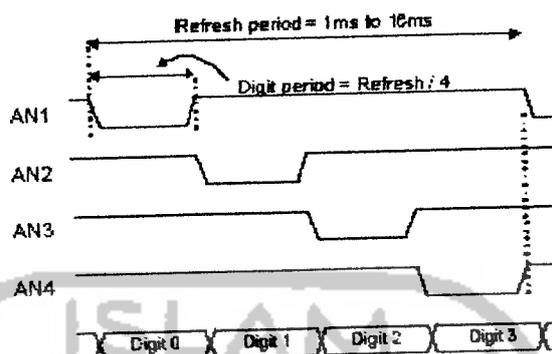


Gambar 2.23 *Common anode seven-segment 1 digit*

Dilihat dari diagram yang terlihat maka menghasilkan *display seven segment multiplexer*, yang mana bila anoda atau selektor (AN0 - AN3) di aktifkan maka *digit* tersebut yang akan *aktif* dan mendapatkan sinyal/pulsa katoda (CA - CG). Jika dilakukan secara terus-menerus dan bila benar maka keempat *digit* akan terlihat *aktif* seolah-olah semua *aktif* secara bersamaan. Jika diberi sinyal rendah (0Vdc) secara terus menerus 1ms - 16 ms (selector), maka akan terlihat semua *digit aktif*. Dengan syarat, bersamaan dengan *digit enable* data untuk *digit* tersebut harus ada. *Refresh frekuensi* berkisar 60Hz sampai 1KHz. Sebagai contoh dengan 16ms (62.5Hz) sebagai *refresh time* setiap *digit* akan mendapat waktu *aktif* selama $\frac{1}{4}$ dari *refresh time* (dalam hal ini 4ms), selektor harus sesuai dengan data yang akan dimasukkan, dengan kata lain saat *digit 1 aktif* hanya boleh data

untuk *digit 1* yang ada pada CA - CG. Begitu pula dengan *digit 2*, *digit 3* dan *digit*

4. hal ini dapat dilihat pada gambar 2.24.



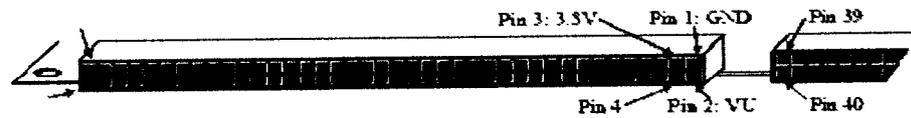
Gambar 2.24 Diagram waktu sinyal *seven-segment*

Tabel 2.3 Masukan katoda untuk *digit* desimal

Digit Shown	Cathode Signals						
	a	b	c	d	e	f	g
0	0	0	0	0	0	0	1
1	1	0	0	1	1	1	1
2	0	0	1	0	0	1	0
3	0	0	0	0	1	1	0
4	1	0	0	1	1	0	0
5	0	1	0	0	1	0	0
6	0	1	0	0	0	0	0
7	0	0	0	1	1	1	1
8	0	0	0	0	0	0	0
9	0	0	0	1	1	0	0

2.7.6 Port I/O Tambahan

Pada modul Pegasus terdapat 3 *port* tambahan (A1, A2, dan B1) dengan masing-masing *port* terdapat 40 *pin*. Masing-masing *port* memiliki GND pada *pin* 1, VU pada *pin*2, dan 3Vdc - 5Vdc pada *pin*3. untuk *pin* 4-35 merupakan *pin* sinyal I/O, dan *pin* 36-40 digunakan sebagai *port* JTAG tambahan, atau juga bisa digunakan sebagai *clock* tambahan.



Gambar 2.25 Pin penghubung tambahan

Tabel 2.4 Konfigurasi pin yang terdapat pada modul pegasus.

Pegasus Expansion Connector Pinout								
Connector B1			Connector A1			Connector A2		
Pin	Signal	B1	Pin	Signal	A1	Pin	Signal	A2
39	TDO	TDO	39	TDO	TDO	39	GCK0	GCK0
40	TDI	TDI	40	TDI	TDI	40	GND	GND
37	TMS	TMS	37	TMS	TMS	37	n/c	n/c
38	TCK	TCK	38	TCK	TCK	38	n/c	n/c
35	MB1-INIT	90	35	MA1-INIT	189	35	MA2-INT	138
36	GND	GND	36	GND	GND	36	Not used	n/c
33	MB1-WAIT	95	33	MA1-WAIT	192	33	MA2-WAIT	140
34	M1-RST	94	34	M1-RST	191	34	MA2-RST	139
31	MB1-DSTB	97	31	MA1-DSTB	194	31	MA2-DSTB	142
32	MB1-WRIT	96	32	MA1-WRIT	193	32	MA2-WRIT	141
29	MB1-DB7	99	29	MA1-DB7	190	29	MA2-DB7	147
30	MB1-ASTB	98	30	MA1-ASTB	195	30	MA2-ASTB	148
27	MB1-DB5	101	27	MA1-DB5	201	27	MA2-DB5	148
28	MB1-DB6	100	28	MA1-DB6	200	28	MA2-DB6	148
26	MB1-DB3	108	26	MA1-DB3	203	26	MA2-DB3	151
26	MB1-DB4	102	26	MA1-DB4	202	26	MA2-DB4	150
23	MB1-DB1	110	23	MA1-DB1	205	23	MA2-DB1	160
24	MB1-DB2	109	24	MA1-DB2	204	24	MA2-DB2	152
21	P-LS6CLK	112	21	LS6CLK	3	21	P-IO18	162
22	MB1-DB0	111	22	MA1-DB0	206	22	MA2-DB0	161
19	P1-DB7	114	19	DB7	5	19	P-IO16	164
20	P-CSA	113	20	CSA	4	20	P-IO17	163
17	P-DB6	119	17	DB6	7	17	P-IO14	168
18	P-OE	115	18	OE	6	18	P-IO15	165
15	P-DB5	121	15	DB5	9	15	P-IO12	168
16	P-WE	120	16	WE	8	16	P-IO13	167
13	P-DB4	123	13	DB4	14	13	P-IO10	173
14	P-ADR5	122	14	ADR5	10	14	P-IO11	172
11	P-DB3	126	11	DB3	16	11	P-IO8	175
12	P-ADR4	125	12	ADR4	15	12	P-IO9	174
9	P-DB2	129	9	DB2	18	9	P-IO6	178
10	P-ADR3	127	10	ADR3	17	10	P-IO7	176
7	P-DB1	133	7	DB1	21	7	P-IO4	180
8	P-ADR2	132	8	ADR2	20	8	P-IO5	179
5	P-DB0	135	5	DB0	23	5	P-IO2	187
6	P-ADR1	134	6	ADR1	22	6	P-IO3	181
3	VCC0	VCC0	3	VCC0	VCC0	3	VCC0	VCC0
4	P-ADR0	136	4	ADR0	24	4	P-IO1	188
1	GND	GND	1	GND	GND	1	GND	GND
2	VU	VU	2	VU	VU	2	VU	VU

Tabel 2.5 Port Accessory

Accessory Port Pinout					
Pin	Name	FPGA Pin	Pin	Name	FPGA Pin
1	AC0	P49	4	AC3	P47
2	AC1	P48	5	GND	-
3	AC2	P81	6	Vdd	-

Tabel 2.6 Pin FPGA XC2S50

Pegasus FPGA Pin Assignments					
Pin	Function	Pin	Function	Pin	Function
1	GND	53	VCCO	105	VCCO
2	TMS	54	MODE2	106	PROGRAM
3	LSBCLK	55	PB-I014	107	INITIO
4	LCSA	56	PB-I013	108	LMB1-DB3
5	LDB7	57	BTN2	109	LMB1-DB2
6	LDE	58	BTN1	110	LMB1-DB1
7	LDB6	59	BTND	111	LMB1-DB0
8	LWE	60	AND	112	LPB-LSBCLK
9	LDB5	61	CE	113	LPB-CSA
10	LADR5	62	CD	114	LPB-DB7
11	GND	63	DP	115	LPB-OE
12	VCCO	64	GND	116	GND
13	VCCINIT	65	VCCO	117	VCCO
14	LDB4	66	VCCINIT	118	VCCINIT
15	LADR4	67	CC	119	LPB-DB6
16	LDB3	68	CG	120	LPB-WE
17	LADR3	69	AN1	121	LPB-DB5
18	LDB2	70	CB	122	LPB-ADR5
19	GND	71	AN2	123	LPB-DB4
20	LADR2	72	GND	124	GND
21	LDB1	73	CF	125	LPB-ADR4
22	LADR1	74	CA	126	LPB-DB3
23	LDB0	75	AN3	127	LPB-ADR3
24	LADR0	76	VCCINIT	128	VCCINIT
25	GND	77	GCK1	129	LPB-DB2
26	VCCO	78	VCCO	130	VCCO
27	VS	79	GND	131	GND
28	VCCINIT	80	GCK0	132	LPB-ADR2
29	HS	81	SW7AC2	133	LPB-DB1
30	BLUE	82	SW5	134	LPB-ADR1
31	GRN	83	SW6	135	LPB-DB0
32	GND	84	SW4	136	LPB-ADR0
33	RED	85	GND	137	GND
34	PS2C	86	SW3	138	LMA2-INT
35	PS2D	87	SW2	139	LMA2-RESET
36	LD7	88	SW1	140	LMA2-WAIT
37	LD6	89	SW0	141	LMA2-WRITE
38	VCCINIT	90	LMB1-INT	142	LMA2-DSTB
39	VCCO	91	VCCINIT	143	VCCINIT
40	MC1-DB4	92	GND	144	VCCO
41	LD5	93	GND	145	GND
42	LD4	94	LMB1-RESET	146	LMA2-ASTB
43	LD3	95	LMB1-WAIT	147	LMA2-DB7
44	LD2	96	LMB1-WRITE	148	LMA2-DB6
45	LD1	97	LMB1-DSTB	149	LMA2-DB5
46	LDD	98	LMB1-ASTB	150	LMA2-DB4
47	AC3	99	LMB1-DB7	151	LMA2-DB3
48	AC1	100	LMB1-DB6	152	LMA2-DB2
49	AC0	101	LMB1-DB5	153	DIN/DIO
50	MODE1	102	LMB1-DB4	154	BTN3
51	GND	103	GND	155	CCLK
52	MODE0	104	DCNE	156	VCCO
				157	TDO
				158	GND
				159	TDI
				160	LMA2-DB1
				161	LMA2-DB0
				162	LPA-I018
				163	LPA-I017
				164	LPA-I016
				165	LPA-I015
				166	LPA-I014
				167	LPA-I013
				168	LPA-I012
				169	GND
				170	VCCO
				171	VCCINIT
				172	LPA-I011
				173	LPA-I010
				174	LPA-I09
				175	LPA-I08
				176	LPA-I07
				177	GND
				178	LPA-I06
				179	LPA-I05
				180	LPA-I04
				181	LPA-I03
				182	GCK2
				183	GND
				184	VCCO
				185	GCK3
				186	VCCINIT
				187	LPA-I02
				188	LPA-I01
				189	LMA1-INT
				190	GND
				191	LMA1-RESET
				192	LMA1-WAIT
				193	LMA1-WRITE
				194	LMA1-DSTB
				195	LMA1-ASTB
				196	VCCINIT
				197	VCCO
				198	GND
				199	LMA1-DB7
				200	LMA1-DB6
				201	LMA1-DB5
				202	LMA1-DB4
				203	LMA1-DB3
				204	LMA1-DB2
				205	LMA1-DB1
				206	LMA1-DB0
				207	TCK
				208	VCCO

Tabel 2.7 Konfigurasi Pin yang Terdapat pada Board Pegasus.

Pegasus Expansion Connector Pinout								
Connector B1			Connector A1			Connector A2		
Pin	Signal	B1	Pin	Signal	A1	Pin	Signal	A2
39	TDO	TDO	39	TDO	TDO	39	GCK0	GCK0
40	TDI	TDI	40	TDI	TDI	40	GND	GND
37	TMS	TMS	37	TMS	TMS	37	n/c	n/c
38	TCK	TCK	38	TCK	TCK	38	n/c	n/c
35	MA1-INT	30	35	MA1-INT	188	35	MA2-INT	138
38	GND	GND	38	GND	GND	38	Not used	n/c
33	MA1-WAIT	35	33	MA1-WAIT	192	33	MA2-WAIT	140
34	M1-RET	34	34	M1-RET	191	34	MA2-RET	139
31	MA1-DSTB	37	31	MA1-DSTB	194	31	MA2-DSTB	142
32	MA1-WRT	38	32	MA1-WRT	193	32	MA2-WRT	141
29	MA1-DB7	39	29	MA1-DB7	199	29	MA2-DB7	147
30	MA1-ASTB	33	30	MA1-ASTB	195	30	MA2-ASTB	148
27	MA1-DB5	101	27	MA1-DB5	201	27	MA2-DB5	149
28	MA1-DB6	100	28	MA1-DB6	200	28	MA2-DB6	148
25	MA1-DB3	108	25	MA1-DB3	203	25	MA2-DB3	151
28	MA1-DB4	102	28	MA1-DB4	202	28	MA2-DB4	150
23	MA1-DB1	110	23	MA1-DB1	205	23	MA2-DB1	160
24	MA1-DB2	109	24	MA1-DB2	204	24	MA2-DB2	152
21	P-LSBCLK	112	21	LSBCLK	3	21	P-IO13	162
22	MA1-DB0	111	22	MA1-DB0	208	22	MA2-DB0	161
19	P1-DB7	114	19	DB7	5	19	P-IO18	164
20	P-CSA	113	20	CSA	4	20	P-IO17	163
17	P-DB6	118	17	DB6	7	17	P-IO14	168
18	P-OE	115	18	OE	6	18	P-IO15	165
15	P-DB5	121	15	DB5	8	15	P-IO12	169
18	P-WE	120	18	WE	9	18	P-IO13	167
13	P-DB4	123	13	DB4	14	13	P-IO10	173
14	P-ADRE	122	14	ADRE	10	14	P-IO11	172
11	P-DB3	128	11	DB3	18	11	P-IO8	175
12	P-ADR4	125	12	ADR4	15	12	P-IO9	174
9	P-DB2	129	9	DB2	19	9	P-IO6	178
10	P-ADR3	127	10	ADR3	17	10	P-IO7	176
7	P-DB1	133	7	DB1	21	7	P-IO4	180
8	P-ADR2	132	8	ADR2	20	8	P-IO5	179
5	P-DB0	135	5	DB0	23	5	P-IO2	187
8	P-ADR1	134	8	ADR1	22	8	P-IO3	181
3	VCC0	VCC0	3	VCC0	VCC0	3	VCC0	VCC0
4	P-ADRC	136	4	ADRC	24	4	P-IO1	169
1	GND	GND	1	GND	GND	1	GND	GND
2	VU	VU	2	VU	VU	2	VU	VU

2.7 Sistem Digital

Suatu isyarat digital merupakan suatu gelombang yang mempunyai peralihan mendadak antara dua nilai besaran atau dengan kata lain hanya mengenal dua macam keadaan. Dua keadaan ini tidak dikenal besarannya, tetapi dikenal dengan keadaan tinggi rendah. Dengan bilangan biasa, dua macam keadaan ini dapat diberi tanda 0 (nol) jika rendah, dan tanda 1 (satu) jika tinggi. Dalam isyarat-isyarat listrik tanda nol dapat diartikan tidak ada tegangan, dan isyarat satu dapat diartikan ada tegangan. Dalam hal ini dimungkinkan satu tingkat isyarat dapat bernilai 0 dan tingkat isyarat lain bernilai 1. keadaan 0 dan 1 inilah yang dinamakan nalar (LOGIC).

Tabel berikut ini menunjukkan dua macam keadaan yang dimaksud dalam sistem digital:

Tabel 2.8 Tabel keadaan sistem digital

NALAR	Keadaan suatu sistem				
	1	ON	Tinggi	Ya	Naik
0	OFF	Rendah	Tidak	Turun	0 Volt