BAB II

DASAR TEORI

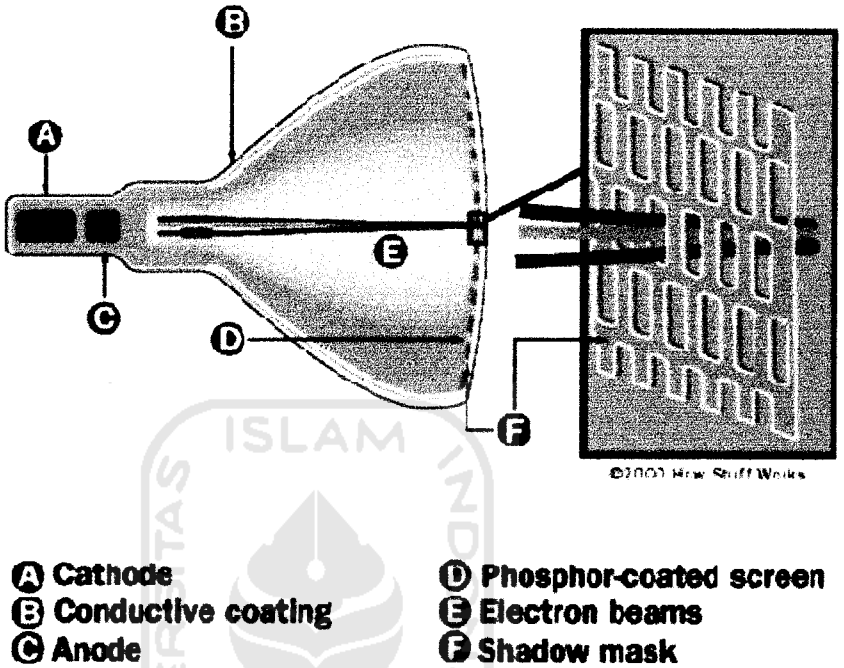
2.1 Monitor CRT (*Cathode RayTube*)

Prinsip kerja monitor konvensional, monitor CRT (Cathode Ray Tube), sama dengan prinsip kerja televisi yang berbasis CRT. Elektron ditembakkan dari belakang tabung gambar menuju bagian dalam tabung yang dilapis elemen yang terbuat dari bagian yang memiliki kemampuan untuk memancarkankan cahaya. Sinar elektron tersebut melewati serangkaian magnet yang membelok-belokkan sinar menuju bagian-bagian tertentu dari tabung bagian dalam.

Begitu sinar tersebut sampai ke bagian kaca tabung TV atau monitor, maka akan menyinari lapisan tersebut, menyebabkan tempat-tempat tertentu untuk menyala secara temporer. Setiap tempat tertentu mewakili pixel tertentu. Dengan mengontrol tegangan dari sinar tersebut, terciptalah teknologi yang mampu mengatur pixel-pixel tersebut untuk berpendar dengan intensitas cahaya tertentu. Dari pixel-pixel tersebut, dapat dibentuklah gambar.

Teorinya, untuk membentuk sebuah gambar, sinar tadi menyapu sebuah garis horizontal dari kiri ke kanan, menyebabkan pixel-pixel tadi menyala dengan intensitas cahaya sesuai dengan tegangan yang telah diatur. Proses tersebut terjadi pada semua garis horizontal yang ada pada pixel layar, dan ketika telah sampai

ujung, sinar tersebut akan mati sementara untuk mengulang proses yang sama untuk menghasilkan gambar yang berbeda.



Gambar 2.1 Bagian-bagian dari monitor CRT.

Sesaat sebelum elektron (*electron beams*) *tersebut* menyentuh *phosphor-coated screen*, mereka melalui sebuah *shadow mask* atau *aperture grille* yang terletak sepersekian inci di belakang layar, yang menyaring tembakan elektron tersebut agar mengenai fosfor yang tepat. Pada sebuah monitor CRT *shadow mask*, selembar metal yang memiliki lubang-lubang mengarahkan elektron yang ditembakkan pada lingkaran *phosphor-coated screen.* Pada monitor CRT *aperture grille* sinar diarahkan langsung melalui slot diantara kawat vertikal yang tipis. Pada kedua jenis monitor tersebut, ruang diantara lubang atau kawat tersebut (yang dikenal sebagai "dot pitch" pada jenis *shadow mask* dan "grille pitch" pada jenis *aperture grille*)

menentukan seberapa detail gambar yang dihasilkan oleh monitor: Secara garis besar, semakin kecil pitch, semakin presisi penempatan sinar tersebut, sehingga semakin jelas gambar yang ditampilkan.

Untuk menampilkan gambar suatu monitor memerlukan 5 signal input yaitu:

1. R (*red signal*).
2. G (*green signal*).
3. B (*blue signal*).
4. HS (*horizontal synchronization*).
5. VS (*vertical synchronization*).

2.1.1 R, G, B

Red, *Green* dan *Blue* adalah input yang dibutuhkan oleh monitor dimana pengiriman data dan penentuan warna adalah melalui R, G, B ini.

Tabel 2.1 Tabel warna 1 bit.

Color	Red	Green	Blue
Black	0	0	0
Blue	0	0	1
Green	0	1	0
Cyan	0	1	1
Red	1	0	0
Purple	1	0	1
Yellow	1	1	0
White	1	1	1

Tabel 2.1 adalah logika pembentuk warna yang akan kita gunakan untuk menentukan warna text yang akan kita tampilkan.

2.1.2 HS dan VS (*horizontal* dan *vertical synchronization*).

Sinyal dan *h_sync* dan *v_sync* dibutuhkan untuk mengaktifkan monitor saat *h_sync* dan *v_sync high* berarti monitor aktif sedangkan jika *h_sync* dan *v_sync* rendah monitor tidak aktif atau dalam keadaan *sleep mode* .

2.1.3 VGA Sistem Timing

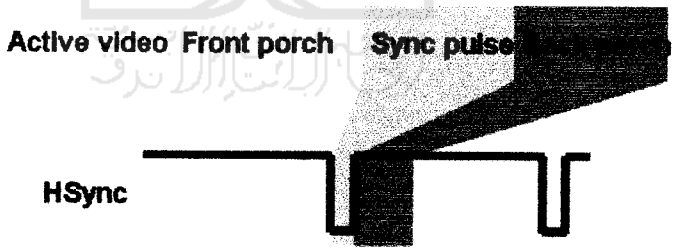
Dalam rangka untuk menampilkan gambar pada suatu monitor, gambar diletakkan pada suatu bingkai atau *frame*. Untuk itu dibutuhkan garis horisontal (pixel) dan garis vertical (line) untuk membangun suatu *frame*. Untuk menentukan nilai-nilai dari garis horisontal dan garis vertikal disebut VGA sistem *timing*.

VGA sistem *timing* dibutuhkan untuk menentukan mode dan pengaturan *Vsync* dan *Hsync*-nya. Jadi pada sebuah mode akan mempunyai frekuensi serta nilai horisontal dan vertikal *synchronization* agar monitor dapat bekerja pada mode tersebut. Dibawah ini adalah tabel sitem timing dari monitor CRT dimana untuk setiap mode mempunyai *clock, active video, front porch, sync pulse, back porch* yang berbeda-beda. Nilai-nilai dibawah ini bukanlah harga mati hanya sistem *timing* standar monitor. Monitor akan dapat bekerja pada suatu mode asalkan nilai *Vsync* dan *Hsync*-nya sesuai.

Tabel 2.2 Tabel sistem *timing* pada monitor

Format	Pixel Clock (MHz)	Horizontal (in Pixels)				Vertical (in Lines)			
		Active Video	Front Porch	Sync Pulse	Back Porch	Active Video	Front Porch	Sync Pulse	Back Porch
640x480, 60Hz	25.175	640	16	96	48	480	11	2	31
640x480, 72Hz	31.500	640	24	40	128	480	9	3	28
640x480, 75Hz	31.500	640	16	96	48	480	11	2	32
640x480, 85Hz	36.000	640	32	48	112	480	1	3	25
800x600, 56Hz	38.100	800	32	128	128	600	1	4	14
800x600, 60Hz	40.000	800	40	128	88	600	1	4	23
800x600, 72Hz	50.000	800	56	120	64	600	37	6	23
800x600, 75Hz	49.500	800	16	80	160	600	1	2	21
800x600, 85Hz	56.250	800	32	64	152	600	1	3	27
1024x768, 60Hz	65.000	1024	24	136	160	768	3	6	29
1024x768, 70Hz	75.000	1024	24	136	144	768	3	6	29
1024x768, 75Hz	78.750	1024	16	96	176	768	1	3	28
1024x768, 85Hz	94.500	1024	48	96	208	768	1	3	36

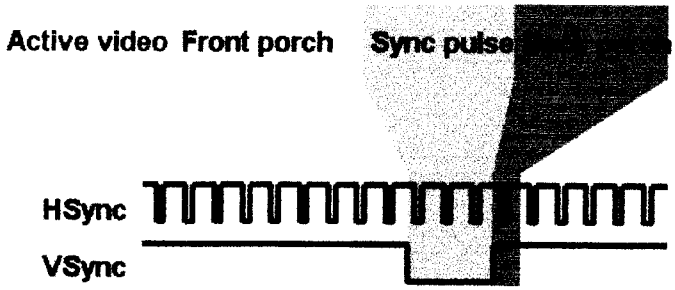
Sumber: Rick Ballantyne, Xilinx Inc.



Gambar 2.2 *Timing of a line.*

Masing-Masing garis video mulai dengan suatu daerah *active video*, di mana nilai RGB adalah *output* untuk masing-masing pixel dalam suatu baris. Daerah aktif diikuti oleh suatu daerah yang kosong, di mana pixel hitam dipancarkan. Pada

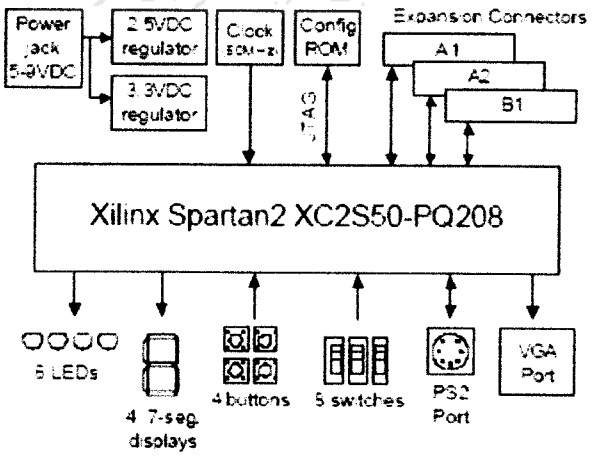
pertengahan interval yang kosong, suatu *sync pulse* horisontal dipancarkan. Di depan *sync pulse* terdapat *front porch* dan dibelakang *sync pulse* terdapat *back porch.*



Gambar 2.3 *Timing of a frame.*

Bingkai (*frame*) dibentuk dengan cara yang hampir sama seperti membentuk baris dari pixel-pixel. Permulaan suatu bingkai berisi semua baris yang akan ditampilkan pada layar, diikuti oleh *front porch.* Pada pertengahan interval Vsync yang kosong, suatu *sync pulse* vertikal dipancarkan. Dan dibelakang *sync pulse* terdapat *back porch.*

2.2 FPGA XC2S50



Gambar 2.4 Papan sirkuit Pegasus

FPGA (*Field Programable Gate Arrays*) adalah salah satu piranti yang termasuk dalam kelompok PLD (*Programable Logic Devices*). Media perancangan berkembang pesat semenjak ditemukannya PLD. Piranti PLD ini merupakan sebuah piranti yang dapat diprogram isinya dan dapat dihapus jika terjadi kesalahan. FPGA berbeda dari *general-purpose* mikroprosesor (misalnya intel) dalam hal fleksibilitas *logic*-nya. Karena pada FPGA tidak terbatas akan pemakaian register-register dan dapat mengimplementasikan rangkaian kombinasi maupun rangkaian sekuensial.

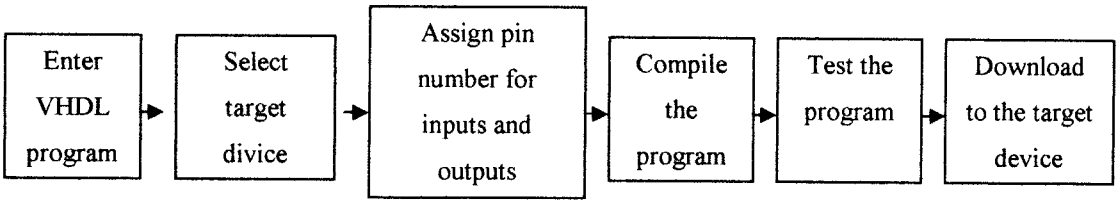
Keistimewaan piranti-piranti yang berbasis PLD adalah:

a) Mudah diprogram sehingga memudahkan perubahan desain jika terdapat kesalahan.
b) Bahasa pemrograman yang digunakan dapat lebih dari satu jenis, misalnya menggunakan *schematic* diagram, *state* diagram maupun *hardware description language* (HDL).
c) Harga tergolong murah untuk pembuatan sebuah *prototype*.

2.3 VHDL

VHDL merupakan singkatan dari *Very High Speed Integrated Circuit Hardware Description Language*. VHDL merupakan bahasa level tinggi yang dapat digunakan untuk mendeskripsikan dan mengoptimalkan watak sebuah sistem digital. VHDL merupakan bahasa pemodelan yang telah memiliki standar international sehingga mudah digunakan dimana saja. Hal ini cukup penting karena jika bahasa yang digunakan tidak standar bisa dimungkinkan adanya kesalahan deskripsi perangkat kerasnya. VHDL adalah salah satu jenis *hardware description language*

(HDL) yang digunakan untuk memprogram PLD. Langkah-langkah mengunakan *software* untuk memprogram PLD dapat dilihat pada Gambar 2.5.



Gambar 2.5 Blok diagram penggunaan software untuk memprogram PLD.

Program VHDL dimasukan, dan perangkat tujuan ditentukan serta pin dari perangkat dibuat sebelum program di *compile*. Setelah program di *compile* dan *software* menyadiakan simulasi desain untuk menyajikan kinerja yang sepatutnya, desain siap untuk di *download* keperangkat yang diinginkan.

VHDL dapat digunakan pada saat akan:

a) Implementasi sistem digital dengan struktur "*top-down*".
b) Meningkatkan produktivitas hasil. Karena menggunakan bahasa level tinggi sehingga desain dapat dilakukan dalam waktu yang lebih singkat jika dibandingkan dengan metode *schematic* diagram.
c) Membuat *prototype*.
d) Merancang sistem yang berubah-rubah.
e) Merancang sistem yanng selalu berkembang I/O dan struktur internalnya.

VHDL dapat digunakan untuk melakukan desain sebuah sistem sampai dengan implementasinya, dengan kemungkinan untuk mencampur beberapa level abstraksi bahasa. Berikut ini adalah level abstraksi bahasa mulai dari yang tertinggi sampai yang rendah:

a) Level abstraksi *behavior*

Bahasa yang mudah dipahami oleh manusia harus dispesifikasikan lagi menjadi bahasa deskripsi yang dapat disimulasikan. Dalam level ini sebuah sistem digital didefinisikan sebagai operasi yang dilakukan sistem terhadap waktu. Konsep waktu menjadi pembeda utama antara level ini dangan level yang lain. Dalam level *behavior* tunda wakru juga dapat dimasukan kedalam perhitungan proses. Level inilah yang paling mirip dengan bahasa pemrograman yang lazim digunaan pada komputer dan paling mudah untuk dipelajari oleh manusia.

b) Level abstraksi dataflow

Pada level ini sistem digital dideskripsikan dengan cara bagaimana data berpindah-pindah dalam sistem yang diinginkan. Karena hampir semua piranti digital sekarang menggunakan register, maka level abtraksi dataflow mendeskripsikan bagaimana data berpindah-pindah diantara register yang ada dalam sistem.

c) Level abstraksi struktur

Pada level ini sistem digital dideskripsikan dalam komponen-komponen penyusunnya dalam bentuk gerbang-gerbang dasar. Level ini membantu dalam membagi sebuah sistem yang besar menjadi bagian-bagian tertentu yang lebih kecil dan mudah diwujudkan. Semakin kompleks sebuah sistem maka semakin diperlukan struktur desain yang jelas agar kesalahan mudah dideteksi.

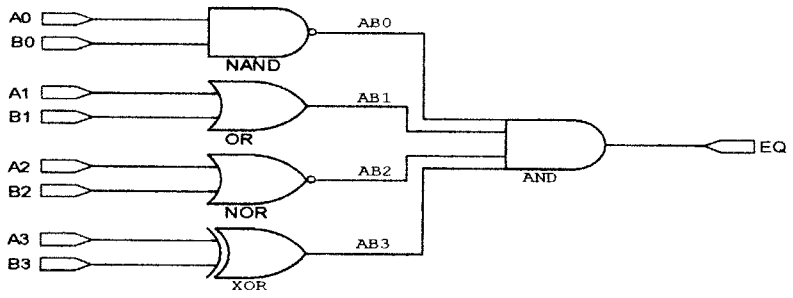
VHDL memiliki sintak-sintak tertentu yang harus dipatuhi agar sistem yang diinginkan dapat terwujud dengan baik. Aturan dasar untuk menulis sebuah kode VHDL adalah sebagai berikut:

1. Dalam sebuah desain VHDL minimal terdapat satu pasang *entity* dan *architecture*.
2. Entity mendeskripsikan unjuk erja sistem jika dilihat dari perspektif *input/output*. *Entity* tidak memuat deskripsi kerja (behaviour/dataflow/ strukture) dari sistem yang dibuat.
3. Architecture berisi deskripsi kerja dari sistem yang dibuat.
4. Sintaknya:

```
Entity <nama_rangkaian> is
port ([signal]<nama>:[<mode>]<indikasi_type>;
.....
[signal]<nama>:[<mode>]<indikasi_type>);
end <nama_rangkaian>;

architecture <nama_desain> of <nama_rangkaian> is
begin
<deskripsi_kerja_dari_sistem>;
end <nama_desain>;
```

Berikut ini adalah contoh dari code VHDL



Gambar 2.6 Contoh skema rangkaian digital.

Berdasarkan logika pada Gambar 2.6 maka dapat dibuat code VHDL sebagai berikut:

```
library ieee;
use ieee.std_logic_1164.all;
entity example is
   Port ( A0,A1,A2,A3,B0,B1,B2,B3 : in std_logic;
          EQ : out std_logic);
end example;
architecture Behavioral of example is
signal AB0,AB1,AB2,AB3:std_logic;
begin
AB0<=A0 nand B0;
AB1<=A1 or B1;
AB2<=A2 nor B2;
AB3<=A3 xor B3;
EQ<=AB0 and AB1 and AB2 and AB3;
end Behavioral;
```

2.4 *Board* 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 mudul Pegasus diantaranya:

a. 50k-gate Xilinx Spartan 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.

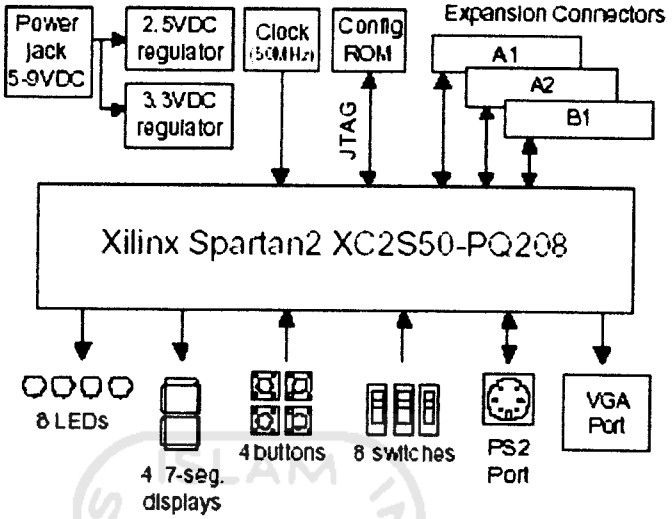
d. 50MHz *osilator* dan satu pin untuk *osilator* tambahan.

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 dilangkapi pengaman.

h. *Port* pemrograman mode JTAG



Gambar 2.7 Blok diagram Pegasus.

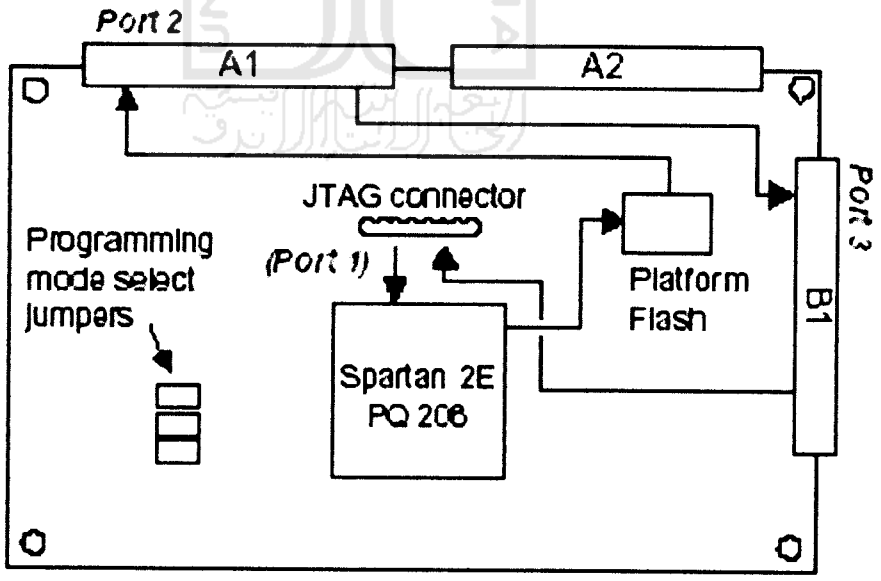
Modul Pegasus sudah dapat dibilang sangat komplit karena hanya dengan menggunakan mudul ini, sudah dapat mendownload *design* dan menjalankannya tanpa perlu menambah komponen lain, karena telah tersedia beberapa masukan dan keluaran.

2.4.1 *Port* JTAG dan pengkonvigurasian Modul

Pada modul Pegasus ini 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 kemodul Pegasus ini, maka dapat dikonfigurasikan melalui *port* JTAG (*port* 1),dari modul utama. Dengan menggunaka *port* JTAG dapat diketahui tipe IC, jenis dan dari keluarga FPGA yang ada pada modul utama maupun

pada modul lain (tambahan) yang terhubung kemodul utama melalui *port* tambahan secara *automatic scan chain.*

Scaning dengan menggunakan JTAG sangat mudah karena perangkat lunak Xilinx melakukanya secara otomatis, Xilinx mencari melalui *port* JTAG dan membaca IC FPGA utama, kemudian apabila tidak ada modul lain yang terhubung pada *port* JTAG tambahan *port*2 (A1) atau *port*3 (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 *scaning 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 tersebut, maka dapat dikonfigurasikan secara bersamaan melalui 1 kabel utama, baik itu *port* USB, Paralell maupun Ethernet.



Gambar 2.8 Aliran scan JTAG pada Pegasus.

2.4.2 *Power supply*

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

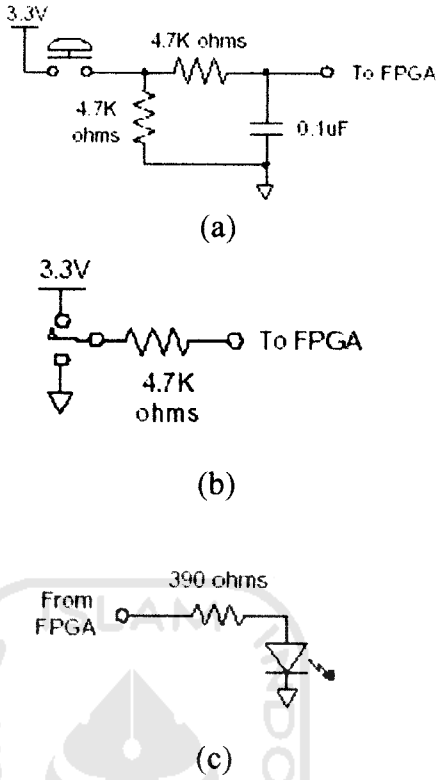
2.4.3 *Osilator*

Modul Pegasus menyediakan 50MHz *osilator* sebagai *clock* utama untuk aplikasi-aplikasi yang akan dikonfigurasikan ke Spartan2 XC2S50. *osilator* tersebut terhubung langsung ke Spartan2 XC2S50 (pin77).

2.4.4 Saklar *Pushbutton*, Saklar geser, *Indikator* LED

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

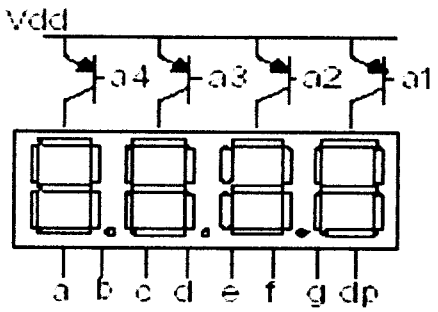
8 LED disediakan sebagai keluaran anoda LED terhubung ke *pin* keluaran melalui resistor 390Ω, sedangkan katoda LED terhubung langsung ke *groound*. LED ke 9 digunakan sebagai *indikator power* FPGA, dan LED ke 10 digunakan sebagai *indikator* status pemrograman JTAG.



Gambar 2.9 Rangkaian saklar *pushbutton* (a), Saklar geser (b), LED (c).

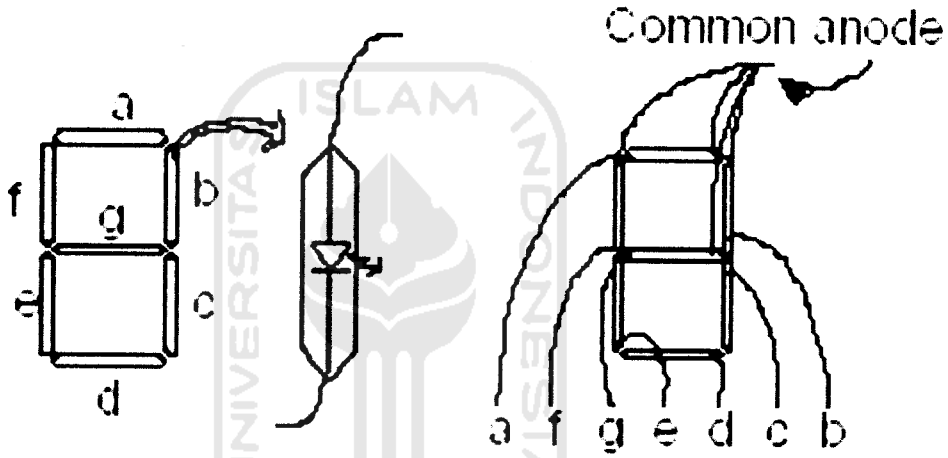
2.4.5 *Seven-Segment*

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

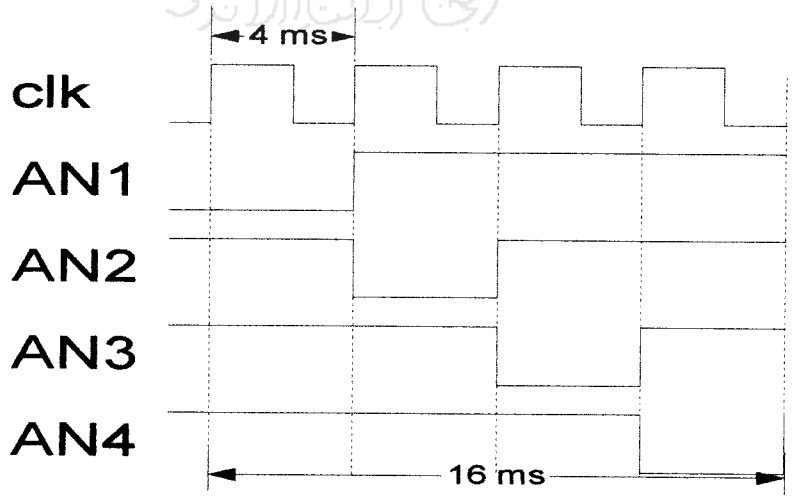


Gambar 2.10 *Seven-segment common* anoda 4 digit.

Ketujuh anoda dari 4 *seven-segment* saling tersambung kedalam *selector* "*common anode*", *display* ini memilki 4 selektor yang dinamakan AN1-AN4, apabila ada sinyal /pulsa yang mengaktifkan selektor ini maka *digit* dari *selector* 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.



Gambar 2.11 *-segment common* anoda 1 digit.

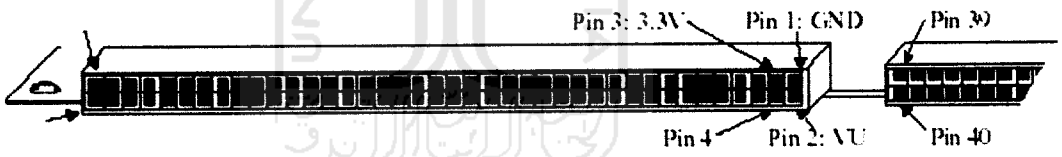


Gambar 2.12 Diagram waktu dari selektor AN.

Dilihat dari diagram yang terlihat maka menghasilkan *display seven-segment multiplexer*, yang mana bila anoda atau *seletor* (AN1-AN4) diaktifkan maka digit tersebut yang akan *aktif* dan mendapat sinyal/pulsa katoda (CA-CG). Jika dilakukan secara terus-menerus dan bila benar maka ke 4 *digit* akan terlihat seolah-olah semua aktif secara bersamaan.

2.4.6 *Port* I/O Tambahan

Pada modul Pegasus terdapat 3 *port* tambahab (A1, A2, dan B1) dengan masing-masing *port* terdapat 40 *pin*. Masing-masing *port* memilki GND dan *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 2.13 *Pin* penghubung tambahan.

Tabel 2.3 *Port accessory*

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

Setiap *pin* pada masing-masing I/O tersebut memiliki alamat sendiri-sendiri. Dan pengalamatan pin harus benar-benar sesuai dengan alamat *pin* yang ada jika tidak maka I/O tidak berfungsi pada *pin* yang digunakan. Alamat dari *pin* tersebut dapat dilihat pada Tabel 2.4.

Tabel 2.4 Konfigurasi *pin* yang terdapat pada modul Pegasus

PEGASUS *Expansion Conector Pinout*								
Conector **B1**			*Conector* **A1**			*Conector* **A2**		
Pin	***Signal***	***B1***	***Pin***	***Signal***	***A1***	***Pin***	***Signal***	***A2***
1	GND	GND	1	GND	GND	1	GND	GND
2	VU	VU	2	VU	VU	2	VU	VU
3	VCCO	VCCO	3	VCCO	VCCO	3	VCCO	VCCO
4	P-ADR0	136	4	ADR0	24	4	P-IO1	188
5	P-DB0	135	5	DB0	23	5	P-IO2	187
6	P-ADR1	134	6	ADR1	22	6	P-IO3	181
7	P-DB1	133	7	DB1	21	7	P-IO4	180
8	P-ADR2	132	8	ADR2	20	8	P-IO5	179
9	P-DB2	129	9	DB2	18	9	P-IO6	178
10	P-ADR3	127	10	ADR3	17	10	P-IO7	176
11	P-DB3	126	11	DB3	16	11	P-IO8	175
12	P-ADR4	125	12	ADR4	15	12	P-IO9	174
13	P-DB4	123	13	DB4	14	13	P-IO10	173
14	P-ADR5	122	14	ADR5	10	14	P-IO11	172
15	P-DB5	121	15	DB5	9	15	P-IO12	168
16	P-WE	120	16	WE	8	16	P-IO13	167
17	P-DB6	119	17	DB6	7	17	P-IO14	166
18	P-OE	115	18	OE	6	18	P-IO15	165
19	P1-DB7	114	19	DB7	5	19	P-IO16	164
20	P-CSA	113	20	CSA	4	20	P-IO1	163
21	P-LSBCLK	112	21	MA1-DB0	3	21	P-IO18	162
22	MB1-DB0	111	22	MA1-DB0	206	22	MA2-DB0	161
23	MB1-DB1	110	23	MA1-DB1	205	23	MA2-DB1	160
24	MB1-DB2	109	24	MA1-DB2	204	24	MA2-DB2	152
25	MB1-DB3	108	25	MA1-DB3	203	25	MA2-DB3	151
26	MB1-DB4	10	26	MA1-DB4	202	26	MA2-DB4	150
27	MB1-DB5	101	27	MA1-DB5	201	27	MA2-DB5	149
28	MB1-DB6	100	28	MA1-DB6	200	28	MA2-DB6	148
29	MB1-DB7	99	29	MA1-DB7	199	29	MA2-DB7	147
30	MB1-ASTB	98	30	MA1-ASTB	195	30	MA2-ASTB	146
31	MB1-DSTB	97	31	MA1-DSTB	194	31	MA2-DSTB	142
32	MB1-WRIT	96	32	MA1-WRIT	193	32	MA2-WRIT	141
33	MB1-WAIT	95	33	MA1-WAIT	192	33	MA2-WAIT	140
34	M1-RST	94	34	M1-RST	191	34	MA2-RST	139
35	MB1-INIT	90	35	MA1-INIT	189	35	MA2-INIT	138
36	GND	GND	36	GND	GND	36	Not used	n/c
37	TMS	TMS	37	TMS	TMS	37	n/c	n/c
38	TCK	TCK	38	TCK	TCK	38	n/c	n/c
39	TDO	TDO	39	TDO	TDO	39	GCK0	GCK0
40	TDI	TDI	40	TDI	TDI	40	GND	GND

Tabel 2.5 *Pin* FPGA XC2S50

PEGASUS FPGA *Pin Assigments*							
Pin	***Function***	***Pin***	***Function***	***Pin***	***Function***	***Pin***	***Function***
1	GND	53	VCCO	105	VCCO	157	TDO
2	TMS	54	MODE2	106	PROGRAM	158	GND
3	LLSBCLK	55	PB-IO14	107	INIT/IO	159	GND
4	LCSA	56	PB-IO13	108	LMB1-DB3	160	LMA2-DB1
5	LDB7	57	BTN2	109	LMB1-DB2	161	LMA2-DB0
6	LCE	58	BTN1	110	LMB1-DB1	162	LPA-IO18
7	LDB6	59	BTN0	111	LMB1-DB0	163	LPA-IO17
8	LWE	60	AN0	112	LPB-LSBCLK	164	LPA-IO16
9	LDB5	61	CE	113	LPB-CSA	165	LPA-IO15
10	LADR5	62	CD	114	LPB-DB7	166	LPA-IO14
11	GND	63	DP	115	LPB-OE	167	LPA-IO13
12	VCCO	64	GND	116	GND	168	LPA-IO12
13	VCCINIT	65	VCCO	117	VCCO	169	GND
14	LDB4	66	VCCINIT	118	VCCINIT	170	VCCO
15	LADR4	67	CC	119	LPB-DB6	171	VCCINIT
16	LDB3	68	CG	120	LPB-WE	172	LPA-IO11
17	LADR3	69	AN1	121	LPB-DB5	173	LPA-IO10
18	LDB2	70	CB	122	LPB-ADR5	174	LPA-IO9
19	GND	71	AN2	123	LPB-DB4	175	LPA-IO8
20	LADR2	72	GND	124	GND	176	LPA-IO7
21	LDB1	73	CF	125	LPB-ADR4	177	GND
22	LADR1	74	CA	126	LPB-DB3	178	LPA-IO6
23	LDB0	75	AN3	127	LPB-ADR3	179	LPA-IO5
24	LADR0	76	VCCINIT	128	VCCINIT	180	LPA-IO4
25	GND	77	GCK1	129	LPB-DB2	181	LPA-IO3
26	VCCO	78	VCCO	130	VCCO	182	GCK2
27	VS	79	GND	131	GND	183	GND
28	VCCINT	80	GCK0	132	LPB-ADR2	184	VCCO
29	HS	81	SW7/AC2	133	LPB-DB1	185	GCK3
30	BLUE	82	SW6	134	LPB-ADR1	186	VCCINIT
31	GRN	83	SW5	135	LPB-DB0	187	LPA-IO2
32	GND	84	SW4	136	LPB-ADR0	188	LPA-IO1
33	RED	85	GND	137	GND	189	LMA1-INT
34	PS2C	86	SW3	138	LMA2-INT	190	GND
35	PS2D	87	SW2	139	LMA2-RESET	191	LMA1-RESET
36	LD7	88	SW1	140	LMA2-WAIT	192	LMA1-WAIT
37	LD6	89	SW0	141	LMA2-WRITE	193	LMA1-WRITE
38	VCCINIT	90	LMB1-INT	142	LMA2-DSTB	194	LMA1-DSTB
39	VCCO	91	VCCINIT	143	VCCINIT	195	LMA1-ASTB
40	MC1-DB4	92	GND	144	VCCO	196	VCCINIT
41	LD5	93	GND	145	GND	197	VCCO
42	LD4	94	LMB1-RESET	146	LMA2-ASTB	198	GND
43	LD3	95	LMB1-WAIT	147	LMA2-DB7	199	LMA1-DB7
44	LD2	96	LMB1-WRITE	148	LMA2-DB6	200	LMA1-DB6
45	LD1	97	LMB1-DSTB	149	LMA2-DB5	201	LMA1-DB5
46	LD0	98	LMB1-ASTB	150	LMA2-DB4	202	LMA1-DB4
47	AC3	99	LMB1-DB7	151	LMA2-DB3	203	LMA1-DB3
48	AC1	100	LMB1-DB6	152	LMA2-DB2	204	LMA1-DB2
49	AC0	101	LMB1-DB5	153	DIN/D0/IO	205	LMA1-DB1
50	MODE1	102	LMB1-DB4	154	BTN3	206	LMA1-DB0
51	GND	103	GND	155	CCLK	207	TCK
52	MODE0	104	DONE	156	VCCO	208	VCCO

Pada modul Pegasus ada beberapa *pin* yang memiliki fungsi khusus dan pin tersebut juga dapat digunakan jika dibutuhkan. Dan tentu untuk menggunakannya harus dilakukan pemanggilan yang sesuai dengan alamat *pin* yang akan digunakan. Dan *pin* khusus tersebut memiliki alamat yang tertera sesuai pada Tabel 2.5.