BAB V
PENGUJIAN DAN ANALISIS

Bab ini membahas tentang pengujian per-blok dan pengujian keseluruhan dari sistem FPU. Pengujian per-blok dan pengujian keseluruhan sistem dilakukan guna mengetahui apakah tiap-tiap blok dan keseluruhan sistem sudah berjalan dengan baik dan benar.

Pengujian dilakukan dengan menggunakan software VHDL sebagai penguji secara terpisah, dimana untuk setiap blok akan digunakan software penguji tersendiri dengan *software* bagian yang diuji adalah tetap. Hal ini dilakukan guna mengetahui lebih detail apakah fungsi setiap blok telah berjalan. Pengujian yang dilakukan adalah pengujian terhadap:

1) Unit *Input-Output.*
2) *Operand Decoder.*
3) *Logical Right Shifter.*
4) *Arithmetic Unit.*
5) *Leading Zero Counter*
6) *Logical Left Shifter.*
7) *Exception Handler.*
8) *Floating Point Encoder.*
9) *Floating Point Unit*

Dalam seluruh pengujian akan digunakan komputer sebagai pemrogram dan pendukung pengujian dengan spesifikasi sebagai berikut:

- Notebook TOSHIBA Satellite L645D
- AMD Phenom™ II P820 Triple-Core Processor @1,80 GHz
- Memory 4 GBytes
- ATI Mobility Radeon HD 4200 Series
- Windows 7 Home Premium 64-bit
- *Software* Xilinx ISE Project Navigator 11.1
- *Software* Digilent Adept

5.1 Pengujian *Input-Output*

5.1.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah unit *input output* dapat menerima masukan dari pengguna dan menampilkannya pada *seven segment.*

5.1.2 Peralatan Pengujian

Peralatan yang digunakan dalam pengujian unit *input output* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

5.1.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian unit *input-output* ini adalah sebagai berikut:

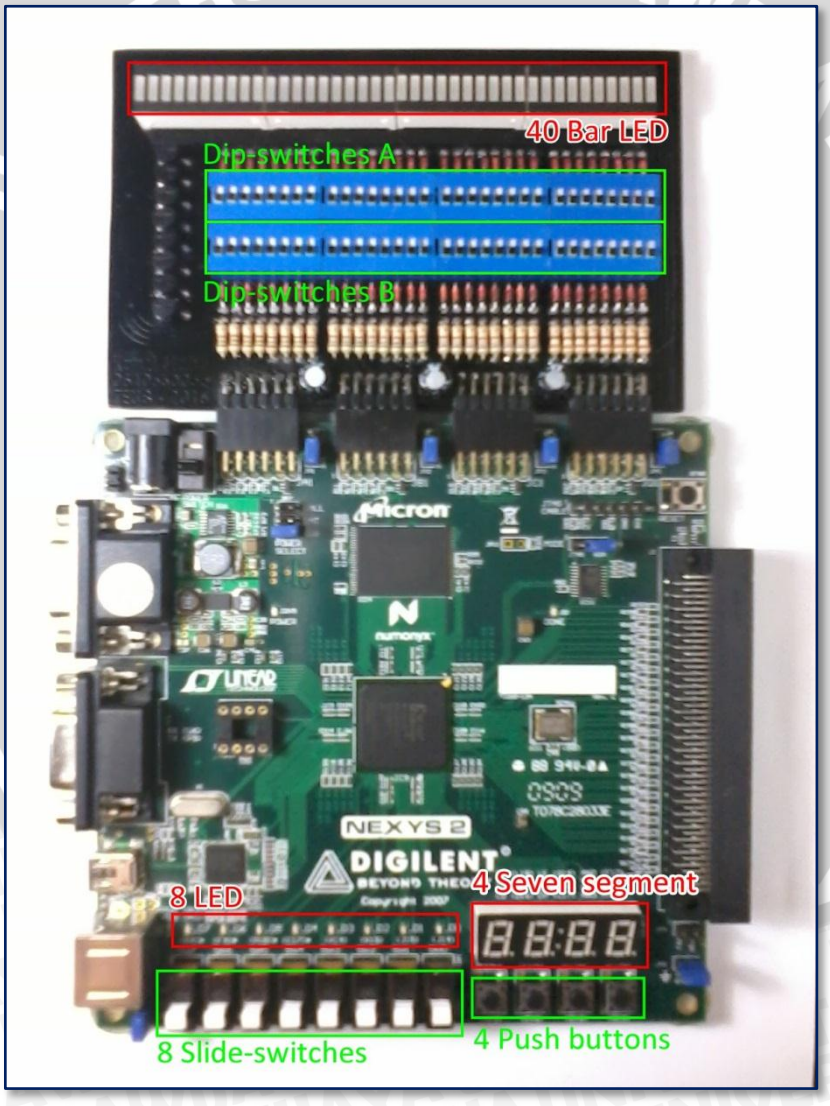
1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.1 (a).
3) Menetapkan 8 *slide-switches*, 32 *dip-switches A,* dan *32 dip-switches B* sebagai *input*.
4) Menetapkan 8 LED, 50 Bar-LED, dan 2 *seven segment display* sebagai *output.*

   Cara menetapkan *input* dan *output* adalah dengan menambahkan *script* dengan format *.ucf. Isi dari *script* tersebut adalah sebagai berikut (hanya berupa potongan *script*):

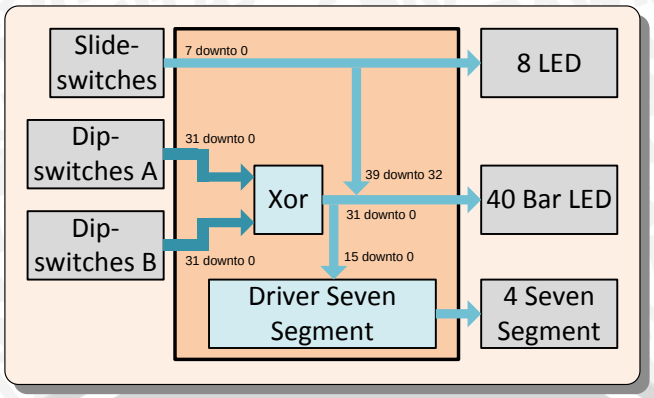
```
NET "Clock_50MHz" LOC = B8;
NET "Switches[0]" LOC = G18;
NET "Switches[1]" LOC = H18;
NET "Switches[2]" LOC = K18;
NET "Switches[3]" LOC = K17;
NET "Switches[4]" LOC = L14;
```

```
NET "Switches[5]" LOC = L13;
NET "Switches[6]" LOC = N17;
NET "Switches[7]" LOC = R17;
NET "Buttons[0]" LOC = B18;
NET "Buttons[1]" LOC = D18;
...
```

5) Membuat program yang sesuai dengan diagram blok pengujian *input output* yang ditunjukkan pada Gambar 5.1 (b).
6) Melakukan pengujian dengan cara memberi masukan melalui 8 *slide switches*, 32 *dip-switches A,* dan 32 *dip-switches B.*
7) Melihat hasil keluaran pada 8 LED, 40 Bar LED dan pada 4 *seven segment*.



(a)



(b)

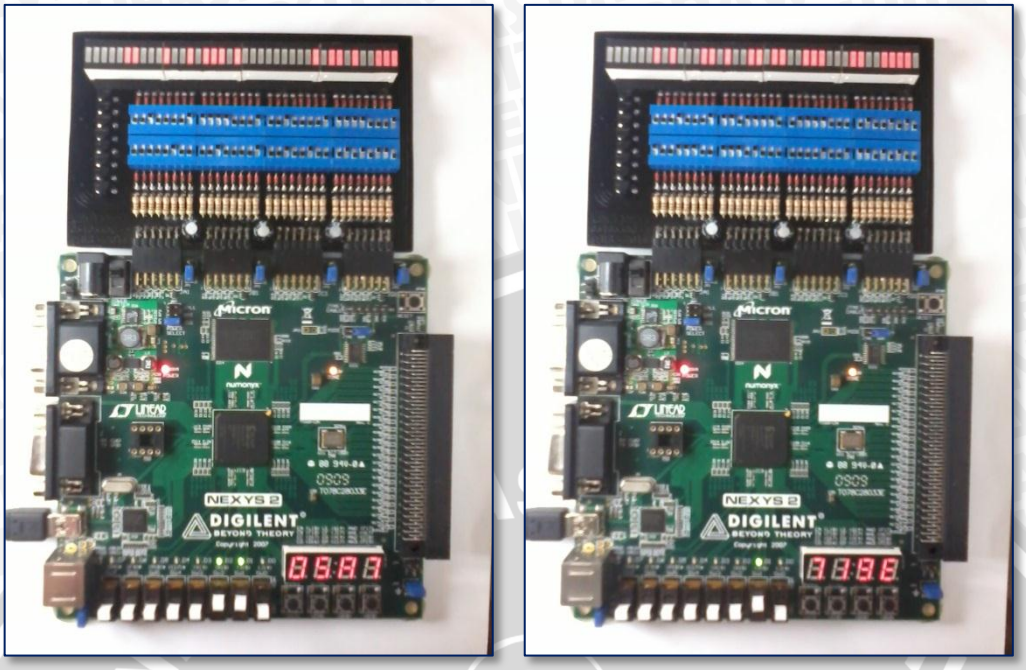
Gambar 5.1 (a) Peletakan *input output* SPARTAN XC3S500E. (b) Diagram blok pengujian *input output*.

5.1.4 Data Hasil Pengujian *Input Output*

Pengujian dilakukan sebanyak 8 kali, yaitu dengan mengubah kondisi semua masukan dan kemudian melihat hasil keluaran pada 8 LED, 40 Bar LED dan *seven segment.* Berdasarkan pengujian yang dilakukan, didapatkan hasil pengujian yang ditunjukkan dalam Tabel 5.1.

Tabel 5.1. Hasil Pengujian *Input-output*

No.	Kondisi *8 slide switch*	Kondisi *Dip-switches A*	Kondisi *Dip-switches B*	Keluaran 8 LED	Keluaran 40 Bar LED	Keluaran *Seven segment*
1.	00	00000000	00000000	00	0000000000	0000
2.	06	21F121F1	04212456	06	0625D005A7	05A7
3.	10	246579A7	0623302D	10	102246498A	498A
4.	63	1E44D118	1385C3DA	63	630DC112C2	12C2
5.	00	01148504	18121988	00	0019069C8C	9C8C
6.	55	01705298	24138392	55	552563D10A	D10A
7.	20	11DE784A	9CE809D4	20	208D36719E	719E
8.	13	178E5569	AD42C3C9	13	13BACC96A0	96A0



Gambar 5.2. (a) Hasil pengujian pada Tabel 5.1. no 2. (b) Hasil Pengujian pada Tabel 5.1 no 7.

5.1.5 Analisis Hasil Pengujian

Pada pengujian *input output* didapatkan hasil bahwa unit *input* telah dapat menerima masukan yang diberikan dan *output* telah dapat menampilkan hasil sesuai yang diinginkan.

Hal tersebut dibuktikan dengan sesuainya nyala 8 LED, 40 Bar LED dan nyala 4 *seven segment* berdasarkan kondisi masukan. Misalnya pada pengujian ke-2, Ketika kondisi *slide switches* adalah `35h`, nyala 8 LED menunjukkan `06h` dan 8 deret LED MSB dari 40 Bar LED juga menunjukkan nilai `06h`. Dan ketika kondisi *dip-switches A* adalah `21F121F1h` dan *dip-switches B* adalah `04212456h`, nyala 32 deret LED LSB dari 40 Bar LED menunjukkan `5D005A7h`, yang mana nilai tersebut adalah sesuai dengan hasil xor dari nilai `04212456h` dan `21F121F1h`. Dan keluaran *seven segment* juga menunjukkan hasil sesuai dengan yang diinginkan, yaitu menampilkan 4 digit pertama dari nilai 40 Bar LED, yaitu `05A7h`.

Begitu pula dengan pengujian ke-3, yaitu ketika kondisi 8 *slide-switches* diubah menjadi bernilai `10h`, maka kondisi nyala 8 LED juga menunjukkan nilai `10h`, dan 8 deret LED MSB dari 40 Bar LED pun menunjukkan nilai `10h`, Dan ketika kondisi *dip-switches A* adalah

246579A7h dan *dip-switches B* adalah 0623302Dh, nyala 32 deret LED LSB dari 40 Bar LED menunjukkan 2246498Ah, yang mana nilai tersebut adalah sesuai dengan hasil xor dari nilai 246579A7h dan 0623302Dh. Keluaran *seven segment*-pun menampilkan 4 digit pertama dari 40 Bar LED yaitu 302Dh.

Dalam pengujian *input output* ini tidak terdapat kesalahan atau hasil *error.* Hal ini dibutuhkan karena dalam sistem FPU tidak boleh terdapat kesalahan (*bug*) sedikitpun.

5.2 Pengujian *Operand Decoder*

5.2.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah *operand decoder* dapat menghasilkan keluaran berupa eksponen, mantissa dan *flag* (*Sign, Zero, Not a Number,* dan *Infinity*) sesuai dengan yang diharapkan ketika diberi masukan berupa operand.

5.2.2 Peralatan Pengujian

Peralatan yang digunakan dalam pengujian *operand decoder* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

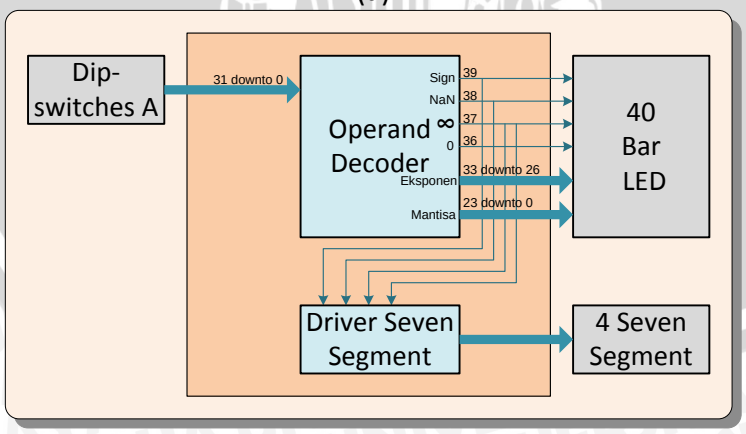
5.2.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian *operand decoder* ini adalah sebagai berikut:

1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.2 (a).

3) Menetapkan 32 bit *dip-switches A* sebagai masukan *operand decoder*.
4) Menetapkan 40 Bar LED sebagai keluaran *operand decoder*, dengan rincian: bit ke-39 sebagai *flag sign*, bit ke-38 sebagai *flag NaN*, bit ke-37 sebagai *flag infinity*, bit ke-36 sebagai *flag zero*, bit ke-33 sampai bit ke-26 sebagai eksponen dan bit ke-23 sampai bit ke-0 sebagai mantisa. Sedangkan sisanya akan diberi logika 0.
5) Menetapkan 4 *seven segment display* sebagai keluaran *operand decoder*, dengan rincian: digit ke-3 sebagai *flag sign*, digit ke-2 sebagai *flag NaN*, digit ke-1 sebagai *flag infinity*, dan digit ke-0 sebagai *flag zero*.
6) Membuat program yang sesuai dengan diagram blok pengujian *operand decoder* yang ditunjukkan dalam Gambar 5.2 (b).
7) Melakukan pengujian dengan memberi masukan dari *operand decoder* dengan cara mengubah kondisi 32 *dip-switches A*.
8) Mencatat *output* yang terdiri dari eksponen, mantisa, dan sinyal *flag* melalui tampilan 40 Bar LED.

(a)



(b)

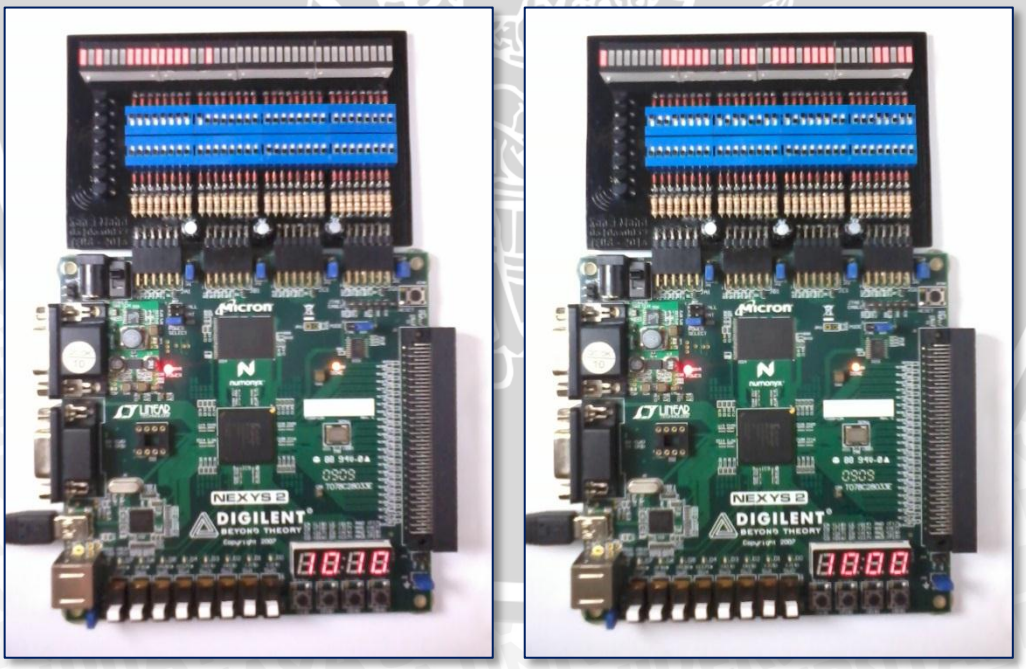
Gambar 5.3 Peletakan *input output* SPARTAN XC3S500E untuk pengujian *operand decoder*. (b) Diagram blok pengujian *operand decoder*.

5.2.4 Data Hasil Pengujian *Operand Decoder*

Pengujian dilakukan sebanyak 8 kali, yaitu memberi masukan *operand decoder* dengan cara mengubah kondisi 32 *dip-switches A* sesuai Tabel 5.2, dan kemudian melihat keluarannya yang berupa eksponen, mantisa dan sinyal *flag* melalui tampilan 40 Bar LED dan mencatatkan hasilnya kedalam Tabel 5.2. Berdasarkan pengujian yang dilakukan, didapatkan hasil pengujian yang ditunjukkan dalam Tabel 5.2.

Tabel 5.2. Hasil pengujian *Operand Decoder*

No	*Dip-switches A*	*Bar LED*						*Seven segment*
		39	*38*	*37*	*36*	*33 - 26*	*23 - 0*	
1.	00000000	0	0	0	1	01	000000	0001
2.	FF800000	1	0	1	0	FF	800000	1010
3.	FFFFFFFF	1	1	0	0	FF	FFFFFF	1100
4.	80630055	1	0	0	0	01	630055	1000
5.	24657981	0	0	0	0	48	E57981	0000
6.	86233030	1	0	0	0	0C	A33030	1000
7.	9FB3BC1B	1	0	0	0	3F	B3BC1B	1000
8.	721DFFC4	0	0	0	0	E4	9DFFC4	0000



Gambar 5.4. (a) Hasil pengujian pada Tabel 5.2 nomor 2. (b) Hasil Pengujian pada Tabel 5.2 nomor 7.

5.2.5 Analisis Hasil Pengujian *Operand Decoder*

Pada pengujian *operand decoder* didapatkan hasil bahwa unit *operand decoder* telah dapat menerima masukan yang diberikan dan dapat menampilkan hasil yang diinginkan.

Pada pengujian pertama, kondisi *dip-switches A* yang merepresentasikan masukan *operand decoder* adalah `00000000h` maka keluaran *operand decoder* bagian *flag zero* yang ditunjukkan oleh 40 Bar LED bit ke-36 akan bernilai 1, sedangkan keluaran *operand decoder* bagian *flag sign* bernilai 0, *flag NaN* bernilai 0, *flag infinity* bernilai 0, eksponen bernilai `0001h` dan mantisa bernilai `000000h`.

Pada pengujian kedua, masukan *operand decoder* adalah `FF800000h` yang setara dengan format data untuk nilai minus tak terhingga, maka keluaran *operand decoder* bagian *flag infinity* yang ditunjukkan oleh 40 Bar LED bit ke-37 menjadi berlogika 1 dan bagian *flag sign* yang ditunjukkan oleh 40 Bar LED bit ke-39 bernilai 1.

Pada pengujian ketiga, masukan *operand decoder* yang ditunjukkan oleh kondisi *dip-switches A* adalah `FFFFFFFFh` yang setara dengan format data untuk nilai *Not a Number*, maka keluaran *operand decoder* bagian *flag NaN* yang ditunjukkan oleh 40 Bar LED bit ke-38 menjadi bernilai 1.

Pada pengujian keempat, masukan *operand decoder* adalah `10630055h` keluaran eksponen yang ditunjukkan 40 Bar LED bit ke-33 sampai bit ke-26 adalah `00h` yang berarti data masukan tersebut adalah sebuah angka yang terdenormalisasi dengan nilai eksponen sebesar -126 bukan -127, karena itu keluaran eksponen dari *operand decoder* adalah `00h` yang merepresentasikan nilai -126 (`01h - bias`). Dan keluaran mantisa yang ditunjukkan 40 Bar LED bit ke-23 sampai bit ke-0 adalah `630055h`.

Pada pengujian kelima sampai kedelapan, masukan *operand decoder* adalah berupa angka ternormalisasi, artinya eksponen dari *operand* masukan setelah terbias adalah lebih dari 0 maka bit MSB dari mantisa yang ditunjuukan oleh 40 Bar LED bit ke-23 adalah bernilai 1.

Dalam pengujian yang dilakukan, *operand decoder* memberikan hasil keluaran sesuai dengan yang diharapkan dan tidak memberikan hasil *error*.

5.3 Pengujian *Logical Right Shifter*

5.3.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah *unit logical right shifter* dapat bekerja sesuai dengan spesifikasi dan fungsi yang dirancang. *Logical right shifter* harus dapat melakukan proses pergeseran bit kearah kanan atau kearah bit yang bernilai lebih kecil sebanyak yang diinginkan.

5.3.2 Peralatan Pengujian

Peralatan yang digunakan dalam pengujian *logical right shifter* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

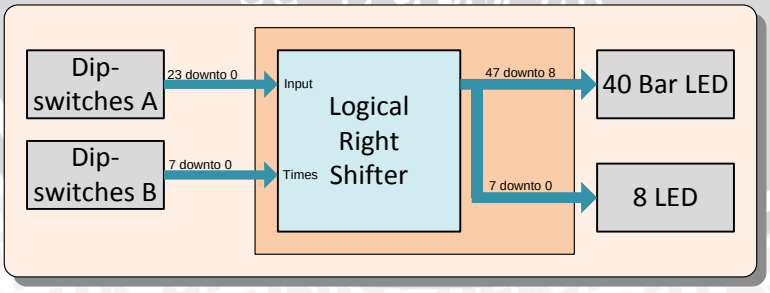
5.3.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian *logical right shifter* ini adalah sebagai berikut:

1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.5 (a).
3) Menetapkan *dip-switches A* bit ke-23 sampai bit ke-0 sebagai masukan data yang akan digeser oleh *logical right shifter*, dan menetapkan *dip-switches B* bit ke-7 sampai bit ke-0 sebagai masukan data *logical right shifter* yang menentukan banyaknya pergeseran yang akan dilakukan.
4) Menetapkan Bar LED sebagai keluaran *operand decoder* untuk bit ke-47 sampai bit ke-8 dan menetapkan LED sebagai keluaran *operand decoder* untuk bit ke-7 sampai bit ke-0.
5) Membuat program yang sesuai dengan diagram blok pengujian *logical right shifter* yang ditunjukkan dalam Gambar 5.5 (b).
6) Melakukan pengujian dengan cara memberikan masukan *logical right shifter* dengan cara mengubah kondisi *dip-switches A*.

7) Mencatat keluaran yang ditunjukkan oleh 40 Bar LED dan 8 LED.

(a)



(b)

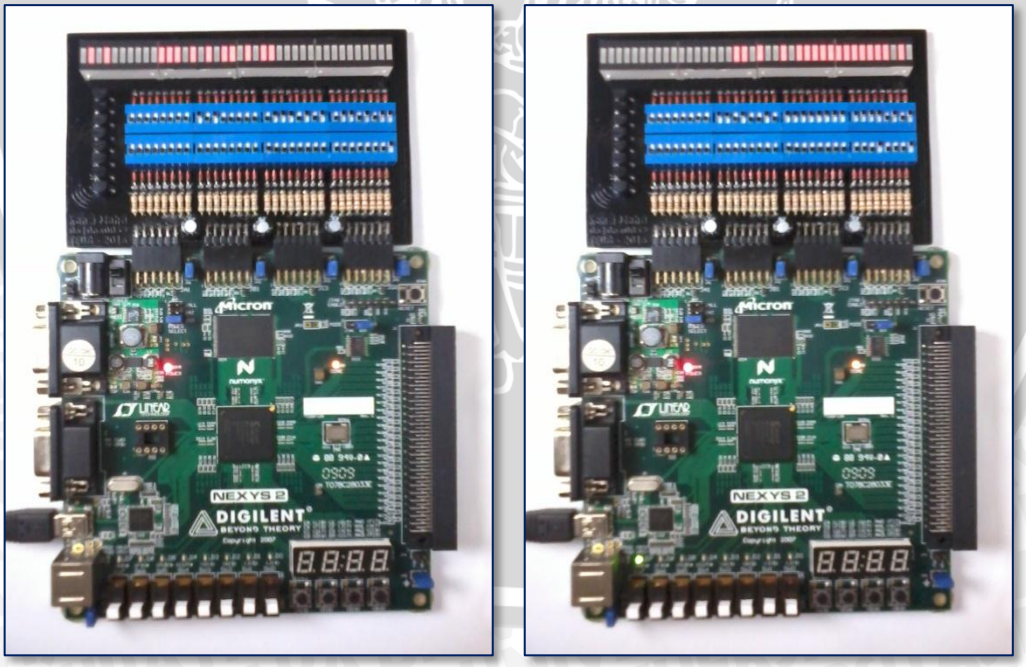
Gambar 5.5 Peletakan *input output* SPARTAN XC3S500E untuk pengujian *logical right shifter*. (b) Diagram blok pengujian *logical right shifter*.

5.3.4 Data Hasil Pengujian *Logical right shifter*

Pengujian dilakukan sebanyak 8 kali, yaitu dengan memberi masukan *logical right shifter* berdasarkan tabel 5.3, dengan cara mengubah kondisi dari *dip-switches A*. dan melihat hasil keluarannya melalui 40 Bar LED dan 8 LED kemudian mencatatkannya kedalam Tabel 5.3. Berdasarkan pengujian yang dilakukan didapatkan data hasil pengujian yang ditunjukkan dalam Tabel 5.3.

Tabel 5.3. Data hasil pengujian *Logical right shifter*

No	*Dip-switches A* 23 - 0	*Dip-switches B* 7 - 0	40 Bar LED 39 - 0	8 LED 7 - 0
1.	71A625	00	71A6250000	00
2.	A0756B	01	503AB58000	00
3.	80303A	04	080303A000	00
4.	B16728	0E	0002C59CA0	00
5.	D2FFF5	11	0000697FFA	80
6.	95B534	18	00000095B5	34
7.	D243E2	2F	0000000000	01
8.	E39C75	30	0000000000	00



Gambar 5.6. (a) Hasil pengujian pada Tabel 5.3 nomor 2. (b) Hasil Pengujian pada Tabel 5.3 nomor 5.

5.3.5 Analisis Hasil Pengujian

Dari data hasil pengujian *logical right shifter* dapat disimpulkan bahwa *logical right shifter* telah dapat menerima masukan yang diberikan dan mengeluarkan data yang diinginkan sesuai dengan spesifikasi dan fungsi yang dirancang.

Pada pengujian pertama, masukan *Logical right shifter* sebagai data yang akan digeser adalah 71A625h yang direpresentasikan oleh *dip-switches A* bit ke-23 sampai bit ke-0 dan besar pergeseran (*Times*) yang direpresentasikan oleh *dip-switches B* bit ke-7 sampai bit ke-0 adalah 00h. bilangan 71A625h sebenarnya merepresentasikan angka 0.11100011010011000100101010b, sehingga bila digeser sebanyak 00h maka keluaran *logical right shifter* dengan *bus* data 48 bit yang ditunjukkan oleh 40 Bar LED dan 8 LED adalah 71A625000000h yang merepresentasikan angka 0.11100011010011000100101000000000000000000000000b.

Pada pengujian keempat, masukan *logical right shifter* sebagai data masukan yang akan digeser adalah B16728h yang merepresentasikan angka 1.01100010110011100101000h, dan besar pergeseran yang diharapkan adalah 0Eh atau sebesar 14 dalam basis desimal maka keluaran dari *logical right shifter* dengan bus data 48 bit yang ditunjukkan oleh 40 Bar LED dan 8 LED adalah 0002C59CA000h yang merepresentasikan angka 0.00000000000001011000101100111001010000000000000b.

Pada pengujian ketujuh, data masukan yang akan digeser oleh *logical right shifter* adalah D243E2h yang merepresentasikan angka 1.10100100 100001111100010b, dan besar pergeseran yang ingin dilakukan adalah 2Fh, maka keluaran dari *logical right shifter* dengan bus data 48 bit adalah 000000000001h yang merepresentasikan angka 0.000000000000000000000000000000000000000000000001b.

Dalam pengujian, *logical right shifter* melakukan tugasnya dengan benar dan tidak memberikan hasil *error*.

5.4 Pengujian *Arithmetic Unit*

5.4.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah *arithmetic unit* dapat bekerja sesuai dengan spesifikasi dan fungsi yang dirancang. *Arithmetic unit* harus dapat melakukan berbagai perhitungan dasar untuk eksponen dan mantisa dengan benar.

5.4.2 Peralatan Pengujian

Peralatan yang digunakan untuk melakukan pengujian *arithmetic unit* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

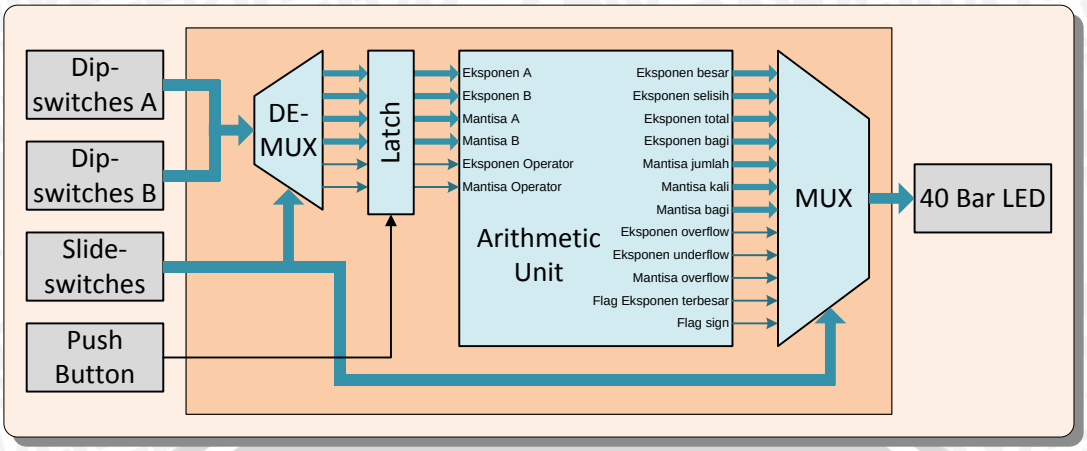
5.4.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian *logical right shifter* ini adalah sebagai berikut:

1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.7 (a).
3) Menetapkan *dip-switches A* dan *dip-switches B* sebagai masukan *Arithmetic unit* untuk eksponen A, eksponen B, Mantisa A, mantisa B, Eksponen operator dan mantisa operator yang terdemultiplekser berdasarkan masukan *slide switches.*
4) Menetapkan 40 Bar LED sebagai penampil semua keluaran *Arithmetic unit* yang termultiplekser berdasarkan masukan *slide switches.*
5) Membuat program yang sesuai dengan diagram blok pengujian *logical right shifter* yang ditunjukkan dalam Gambar 5.3 (b).

6) Melakukan pengujian dengan cara memberikan masukan *logical right shifter* dengan cara mengubah kondisi *dip-switches A, dip-switches B* dan *slide switches* kemudian menekan *push button* untuk menyimpan masukan di *latch*.
7) Mencatat keluaran satu persatu secara bergantian yang ditunjukkan oleh 40 Bar LED berdasarkan kondisi *slide switches*.

(a)



(b)

Gambar 5.7. Peletakan *input output* SPARTAN XC3S500E untuk pengujian *arithmetic unit*. (b) Diagram blok pengujian *arithmetic unit*.

5.4.4 Data Hasil Pengujian *Arithmetic Unit*

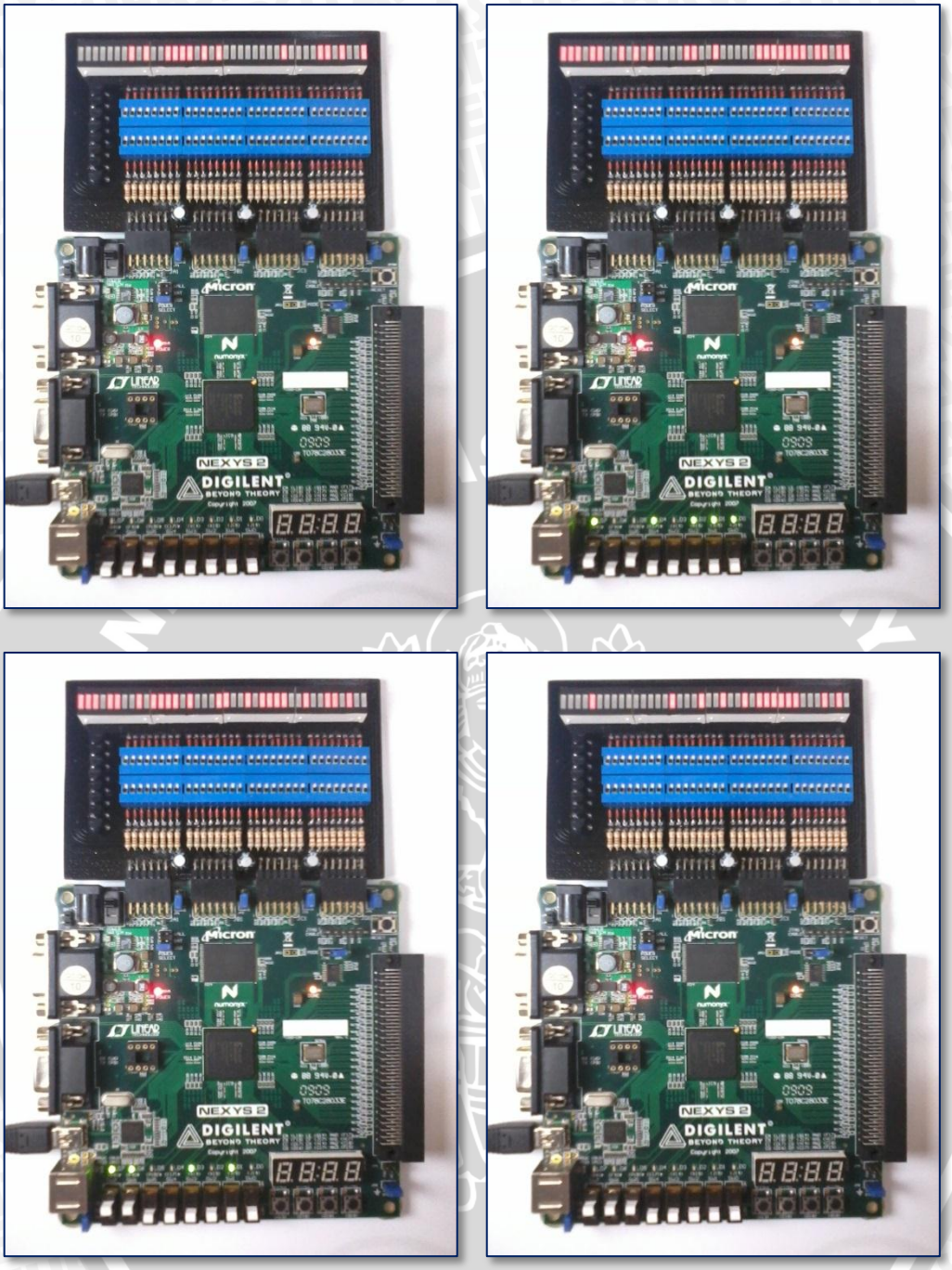
Pengujian dilakukan sebanyak 8 kali, yaitu dengan memberi masukan *arithmetic unit* berdasarkan tabel 5.4. Dan melihat hasil keluarannya melalui 40 Bar LED kemudian mencatatkannya kedalam Tabel 5.4. Berdasarkan pengujian yang dilakukan didapatkan data hasil pengujian yang ditunjukkan dalam Tabel 5.4.

Tabel 5.4. Data Hasil Pengujian *Arithmetic Unit*

No	E A	E B	Mantisa A	Mantisa B	EP	MP	EB	ES	ET	ED	Mantisa Jumlah	Mantisa Kali	Mantisa Bagi	EO	EU	MO	FE	FS
1.	20	13	000011DE784A	496233F0F7AF	0	0	20	0D	B4	32	496245CF6FF9	000004DF8563	0ED3820	0	1	0	0	0
2.	19	88	8484CC734A25	1279DCA33B4C	0	1	88	6F	22	03	720AEFD00ED9	09906AFE8B50	0E584E9	0	0	0	1	0
3.	21	F1	083848583834	000000000001	1	0	F1	D0	AF	14	083848583835	000000000000	1FFFFFF	0	1	0	1	0
4.	F3	F4	1902A0EEFCD7	B362ABB24F2F	1	1	F4	01	7E	23	9A600AC35258	1186799600E0	11D8938	0	0	0	1	1
5.	F3	F5	FEDCBA987654	EDCBA9876543	0	0	F5	02	69	20	ECA8641FDB97	ECBD197934CA	1125F8A	1	0	1	1	0
6.	80	9D	900000000000	000002000000	0	1	9D	1D	9E	16	8FFFFE000000	000001200000	1200000	0	0	0	1	0
7.	A6	73	3FFFFFC00000	000007E00000	1	0	A6	33	B2	13	400007A00000	000001BFFFF9	1249244	0	0	0	0	0
8.	46	19	1515370F99F6	9BC3B78A6F69	1	1	46	2D	AC	23	86AE807AD573	0CD3F6880F51	11532DD	0	0	0	0	1

Catatan:

EA : Masukan *Arithmetic Unit* untuk Eksponen A
EB : Masukan *Arithmetic Unit* untuk Eksponen B
Mantisa A : Masukan *Arithmetic Unit* untuk Mantisa A
Mantisa B : Masukan *Arithmetic Unit* untuk Mantisa B
EP : Masukan *Arithmetic Unit* untuk Eksponen Operator
MP : Masukan *Arithmetic Unit* untuk Mantisa Operator
EB : Keluaran *Arithmetic Unit* untuk Eksponen terbesar
ES : Keluaran *Arithmetic Unit* untuk Eksponen selisih
ET : Keluaran *Arithmetic Unit* untuk Eksponen total
ED : Keluaran *Arithmetic Unit* untuk Eksponen bagi
Mantisa Jumlah : Keluaran *Arithmetic Unit* untuk Mantisa jumlah
Mantisa Kali : Keluaran *Arithmetic Unit* untuk Mantisa kali
Mantisa Bagi : Keluaran *Arithmetic Unit* untuk Mantisa bagi
EO : Keluaran *Arithmetic Unit* untuk Eksponen overflow
EU : Keluaran *Arithmetic Unit* untuk Eksponen underflow
MO : Keluaran *Arithmetic Unit* untuk Mantisa overflow
FE : Keluaran *Arithmetic Unit* untuk Flag eksponen terbesar
FS : Keluaran *Arithmetic Unit* untuk Flag sign



Gambar 5.8. Hasil pengujian pada Tabel 5.4 nomor 3 untuk masing-masing keluaran.

5.4.5 Analisis Hasil Pengujian

Dari data hasil pengujian *logical right shifter* dapat disimpulkan bahwa *logical right shifter* telah dapat menerima masukan yang diberikan dan mengeluarkan data yang diinginkan sesuai dengan spesifikasi dan fungsi yang dirancang.

Misalnya pada pengujian keempat, masukan Eksponen A adalah F3h dan masukan Eksponen B adalah F5h. Maka keluaran *Eksponen Besar* untuk nilai eksponen terbesar adalah F5h, keluaran *Eksponen Selisih* untuk nilai selisih kedua eksponen adalah 01h, keluaran *Eksponen Calculate* untuk nilai Eksponen A dikurangi Eksponen B (Sebab masukan Eksponen Operator Berlogika 1) adalah 7Eh, dan karena Eksponen A lebih kecil dari Eksponen B maka keluaran *Flag Largest* bernilai 1. Karena hasil dari *Eksponen Calculate* masih dalam range 8 bit untuk bilangan positif maupun negatif maka keluaran *Eksponen overflow* dan keluaran *Eksponen underflow* berlogika 0.

Dan pada pengujian keempat, masukan Mantisa A adalah 1902A0EEFCD7h dan Masukan Mantisa B adalah B362ABB24F2Fh, dan karena masukan *Mantisa Operator* bernilai 1 maka keluaran *Mantisa Jumlah* setara dengan Mantisa A dikurangi Mantisa B yaitu bernilai 9A600AC35258h dan karena hasilnya negatif maka keluaran *Flag Sign* bernilai 1.

Perkalian dan pembagian hanya melihat 24 bit MSB dari masing-masing masukan Mantisa, 24 bit LSB tidak akan mempengaruhi keluaran Perkalian dan pembagian. Pada pengujian keempat maka keluaran *Mantisa Kali* sama dengan 1902A0h dikali B362ABh yaitu bernilai 1186799600E0h. Dan pada pengujian keempat, pembagian 1902A0h dengan B362ABh menghasilkan keluaran *Eksponen Bagi* sama dengan -3 yang direpresentasikan dengan nilai 23h (MSB menandakan negatif), serta menghasilkan keluaran *Mantisa Bagi* bernilai 11D8938h.

Pada pengujian yang dilakukan, *arithmetic unit* bekerja dengan baik dan benar serta tidak memberikan kesalahan pada hasil keluaran.

5.5 Pengujian *Leading Zero Counter*

5.5.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah unit *leading zero counter* dapat bekerja sesuai dengan spesifikasi dan fungsi yang dirancang. *Leading zero counter* harus dapat melakukan perhitungan jumlah nol yang mendahului angka satu pertama dalam sebuah deret angka biner.

5.5.2 Peralatan Pengujian

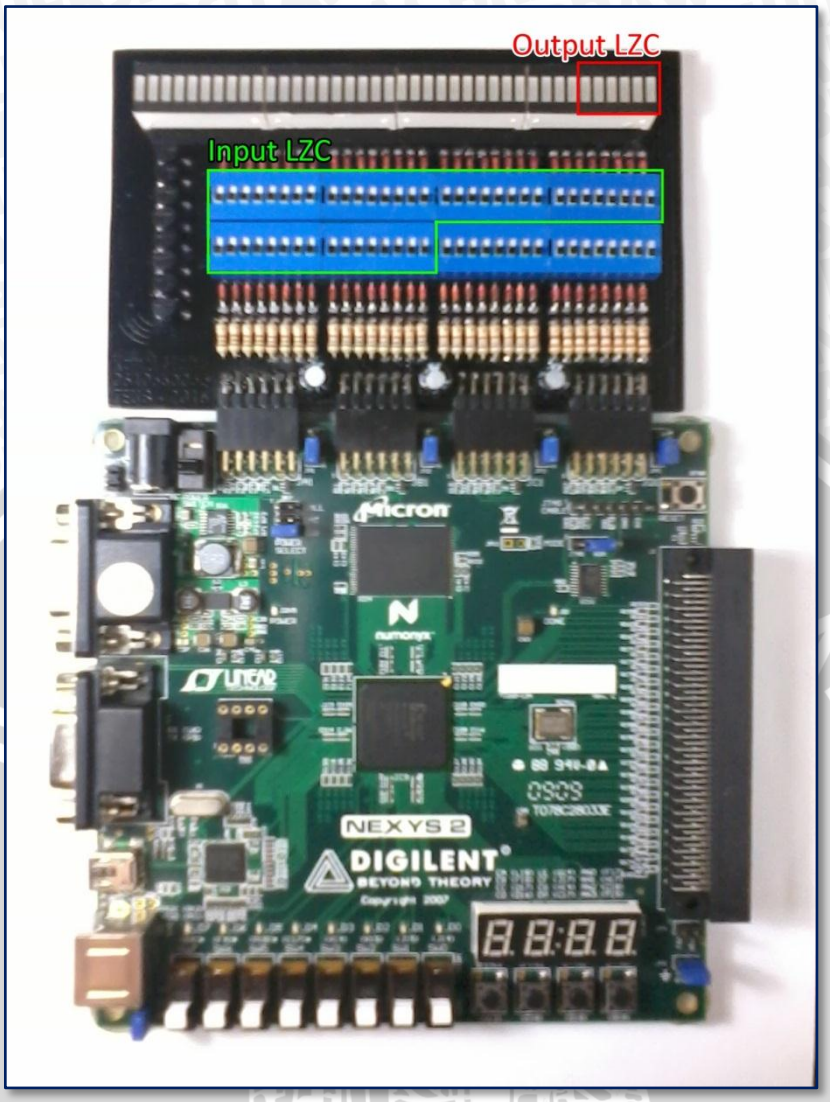
Peralatan yang digunakan dalam pengujian *leading zero counter* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

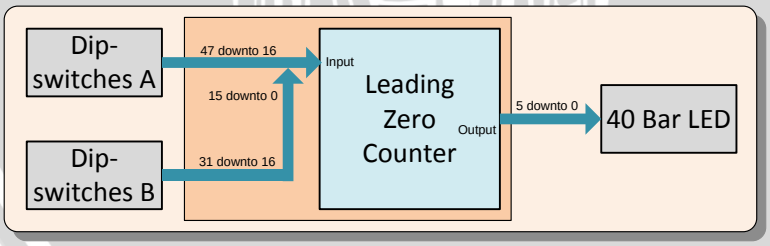
5.5.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian *logical left shifter* ini adalah sebagai berikut:

1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.9 (a).
3) Menetapkan *dip-switches A* sebagai masukan data *leading zero counter* untuk bit ke-47 sampai bit ke-16, dan *dip-switches B* bit ke-31 sampai bit ke-16 sebagai masukan *leading zero counter* untuk bit ke-15 sampai bit ke-0.
4) Menetapkan Bar LED bit ke-5 sampai bit ke-0 sebagai keluaran *leading zero counter*.
5) Membuat program yang sesuai dengan diagram blok pengujian *leading zero counter* yang ditunjukkan dalam Gambar 5.9 (b).
6) Melakukan pengujian dengan cara memberikan masukan *logical right shifter* dengan cara mengubah kondisi *dip-switches A* dan *dip-switches B* berdasarkan fungsinya sebagai masukan.
7) Mencatat keluaran yang ditunjukkan oleh 40 Bar LED.



(a)



(b)

Gambar 5.9 Peletakan *input output* SPARTAN XC3S500E untuk pengujian *leading zero counter*. (b) Diagram blok pengujian *leading zero counter*.

5.5.4 Data Hasil Pengujian *Leading Zero Counter*

Pengujian dilakukan sebanyak 8 kali, yaitu dengan memberi masukan *leading zero counter* berdasarkan tabel 5.5, dengan cara mengubah kondisi *dip-switches A* dan *dip-switches B* sesuai dengan ketentuannya sebagai masukan yang telah ditetapkan. Dan melihat hasil keluaran *leading zero counter* melalui 40 Bar LED pada bit ke-5 sampai bit ke-0, kemudian mencatatkannya kedalam tabel 5.5. Berdasarkan pengujian yang dilakukan, didapatkan data hasil pengujian *leading zero counter* yang ditunjukkan dalam Tabel 5.5.

Tabel 5.5. Data hasil pengujian *Leading Zero Counter*

No	*Dip-switches A* 31 - 0	*Dip-switches A* 31 - 16	40 Bar LED 5 - 0
1.	FFFFFFFF	FFFF	00
2.	80000000	0000	00
3.	30AB4582	3392	02
4.	01234567	89AB	07
5.	00000400	0000	15
6.	0000000A	BCDE	1C
7.	00000000	0001	2F
8.	00000000	0000	30

5.5.5 Analisis Hasil Pengujian

Dari data hasil pengujian *leading zero counter* dapat disimpulkan bahwa *leading zero counter* telah dapat menerima masukan yang diberikan dan mengeluarkan data yang diinginkan sesuai dengan spesifikasi dan fungsi yang dirancang.

Pada pengujian pertama, masukan *leading zero counter* yang ditunjukkan oleh kondisi *dip-switches A* dan *dip-switches* B bit ke-31 sampai bit ke-16 adalah FFFFFFFFFFFFh dan pada pengujian kedua, masukan *leading zero counter* adalah 800000000000h, keduanya tidak memiliki angka nol yang mendahului angka 1 pertama terhitung dari MSB, maka keluaran *leading zero counter* untuk keduanya adalah 00h.

Pada pengujian keempat masukan *leading zero counter* adalah 000004000000h, deret bit ini memiliki 21 deret nol sebelum angka 1

pertama terhitung dari MSB, maka keluaran *leading zero counter* yang ditunjukkan oleh 40 Bar LED bit ke-5 sampai bit ke-0 adalah `15h` atau berarti 21 dalam angka desimal.

Pada pengujian ketujuah masukan *leading zero counter* adalah `000000000001h`, deret bit ini memiliki 47 deret nol sebelum angka 1 pertama terhitung dari MSB, maka keluaran *leading zero counter* adalah `2Fh` atau berarti 47 dalam angka desimal.

Dalam pengujian, *leading zero counter* bekerja dengan baik dan benar serta tidak melakukan kesalahan.

5.6 Pengujian *Logical Left Shifter*

5.6.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah *unit logical left shifter* dapat bekerja sesuai dengan spesifikasi dan fungsi yang dirancang. *Logical left shifter* harus dapat melakukan proses pergeseran bit kearah kiri atau kearah bit yang bernilai lebih besar sebanyak yang diinginkan.

5.6.2 Peralatan Pengujian

Peralatan yang digunakan dalam pengujian *logical left shifter* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

5.6.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian *logical left shifter* ini adalah sebagai berikut:

1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.11 (a) .
3) Menetapkan *dip-switches A* sebagai masukan data yang akan digeser untuk bit ke-47 sampai bit ke-16, dan *dip-switches B* bit

ke-31 sampai bit ke-16 sebagai masukan data yang akan digeser untuk bit ke-15 sampai bit ke-0.

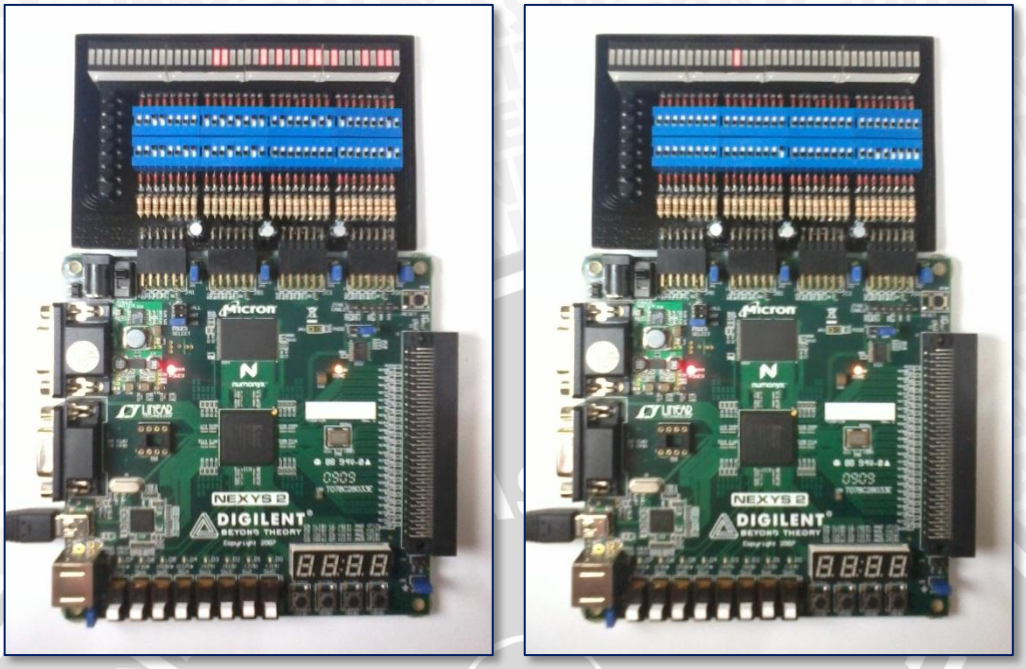
4) Menetapkan *dip-switches* bit ke-5 sampai bit ke-0 sebagai masukan *logical left shifter* untuk besar pergeseran yang akan dilakukan.
5) Menetapkan Bar LED bit ke-23 sampai bit ke-0 sebagai keluaran *logical left shifter*
6) Membuat program yang sesuai dengan diagram blok pengujian *logical right shifter* yang ditunjukkan dalam Gambar 5.11 (b).
7) Melakukan pengujian dengan cara memberikan masukan *logical right shifter* dengan cara mengubah kondisi *dip-switches A* dan *dip-switches B* berdasarkan fungsinya sebagai masukan.
8) Mencatat keluaran yang ditunjukkan oleh 40 Bar LED.

5.6.4 Data Hasil Pengujian *Logical left shifter*

Pengujian dilakukan sebanyak 8 kali, yaitu dengan memberi masukan *logical left shifter* berdasarkan Tabel 5.6, dengan cara mengubah kondisi *dip-switches A* dan *dip-switches B* sesuai dengan ketentuan sebagai masukan yang telah ditetapkan. Dan melihat hasil keluaran *logical left shifter* melalui 40 Bar LED pada bit ke-23 sampai bit ke-0, kemudian mencatatkannya kedalam Tabel 5.6. Berdasarkan pengujian yang dilakukan, didapatkan data hasil pengujuan yang ditunjukkan dalam Tabel 5.6.

Tabel 5.6. Data hasil pengujian *Logical left shifter*

No	*Dip-switches A* 31 - 0	*Dip-switches B* 31 - 16	*Dip-switches B* 5 - 0	40 Bar LED 23 - 0
1.	FFFFFFFF	FFFF	00	FFFFFF
2.	80000000	0000	00	800000
3.	30AB4582	3392	02	C2AD16
4.	01234567	89AB	07	91A2B3
5.	00000400	0000	15	800000
6.	0000000A	BCDE	1C	ABCDE0
7.	00000000	0001	2F	800000
8.	00000000	0000	30	000000



Gambar 5.12. (a) Hasil pengujian pada Tabel 5.6 nomor 3. (b) Hasil Pengujian pada Tabel 5.6 nomor 7.

5.6.5 Analisis Hasil Pengujian

Dari data hasil pengujian *logical left shifter* dapat disimpulkan bahwa *logical left shifter* telah dapat menerima masukan yang diberikan dan mengeluarkan data yang diingikan sesuai dengan spesifikasi dan fungsi yang dirancang.

Pada pengujian pertama masukan data yang akan digeser adalah `FFFFFFFFFFFFh` dan besar pergeseran yang diinginkan adalah `00h` maka keluaran *logical left shifter* adalah sama dengan 24 bit MSB dari masukan yaitu `FFFFFFh`.

Pada pengujian keempat masukan data yang akan digeser adalah `0123456789ABh` sesuai dengan kondisi *dip-switches A* dan *dip-switches B* bit ke-31 sampai bit ke-16. Besar pergeseran yang diinginkan adalah `07h` maka keluaran *logical left shifter* dengan bus data 24 bit adalah `91A2B3h`.

Pada pengujian kelima masukan data yang akan digeser adalah 000004000000h dan besar pergeseran yang diinginkan adalah `15h` maka keluaran *logical left shifter* yang ditunjukan oleh 40 Bar LED bit ke-23 sampai bit ke-0 adalah `800000h`.

Pada pengujian ketujuh masukan data yang akan digeser adalah `000000000001h`, besar pergeseran adalah `2Fh`, maka keluaran data setelah tergeser adalah `800000h`.

Dalam pengujian yang dilakukan, *logical left shifter* bekerja dengan benar dan tidak terdapat kesalahan logika.

5.7 Pengujian *Exception Handler*

5.7.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah *exception handler* berjalan sesuai dengan spesifikasi dan fungsi yang telah ditentukan, yaitu harus dapat memberikan keluaran jika terdapat perhitungan-perhitungan yang membutuhkan penanganan khusus. *Exception handler* bekerja berdasarkan masukan sinyal *flag* (*Sign, Not a Number, Zero, Infinity*) dan mode operasi (perkalian, pembagian, penjumlahan, dan pengurangan).

5.7.2 Peralatan Pengujian

Peralatan yang digunakan dalam pengujian *exception handler* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

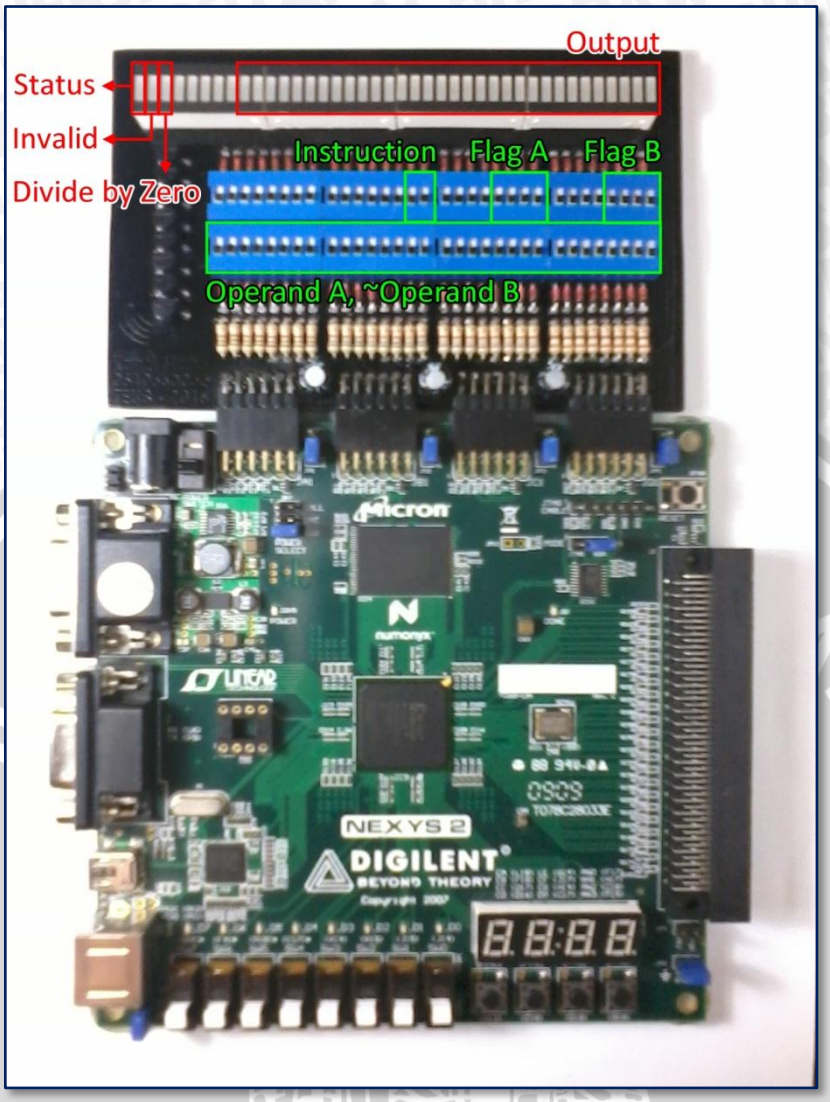
5.7.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian *exception handler* ini adalah sebagai berikut:

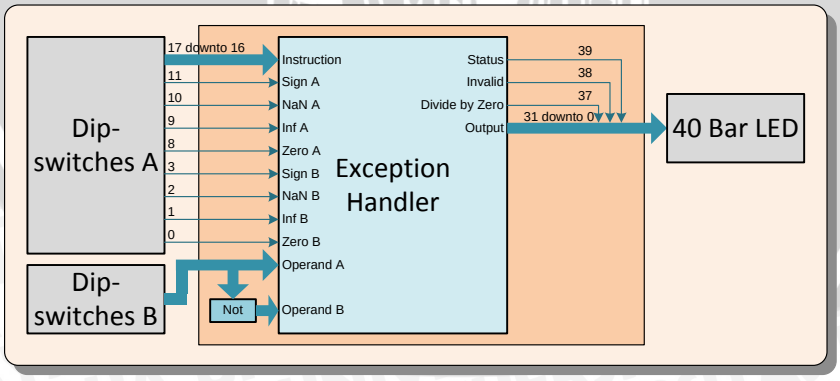
1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.13 (a).
3) Menetapkan *dip-switches A* sebagai masukan dengan ketentuan:
   - Bit ke-17 sampai bit ke-16 sebagai masukan instruksi.

- Bit ke-11 sebagai masukan *flag Sign A*.
- Bit ke-10 sebagai masukan *flag NaN A.*
- Bit ke-9 sebagai masukan *flag Infinity A.*
- Bit ke-8 sebagai masukan *flag Zero A*.
- Bit ke-3 sebagai masukan *flag Sign B*.
- Bit ke-2 sebagai masukan *flag NaN B*.
- Bit ke-1 sebagai masukan *flag Infinity B*.
- Bit ke-0 sebagai masukan *flag Zero B*.

4) Menetapkan *dip-switches B* sebagai masukan *operand A* sekaligus sebagai masukan *operand B* yang diinverter.
5) Menetapkan 40 Bar LED sebagai keluaran dengan ketentuan:
   - Bit ke-39 sebagai keluaran *status*.
   - Bit ke-38 sebagai keluaran *invalid*.
   - Bit ke-37 sebagai keluaran *divide by zero*.
   - Bit ke-31 sampai bit ke-0 sebagai keluaran *output*.
6) Membuat program sesuai dengan diagram blok pengujian *exception handler* yang ditunjukkan dalam Gambar 5.13 (b).
7) Melakukan pengujian dengan memberi masukan *exception handler* dengan cara mengubah kondisi *dip-switches A* dan *dip-switches B* sesuai dengan fungsi yang telah ditetapkan.
8) Mencatat *output* yang terdiri dari *status, invalid, divide by zero,* dan *output* melalui tampilan 40 Bar LED.



(a)



(b)

Gambar 5.13 Peletakan *input output* SPARTAN XC3S500E untuk pengujian *exception handler*. (b) Diagram blok pengujian *exception handler*.

5.7.4 Data Hasil Pengujian *Exception Handler*

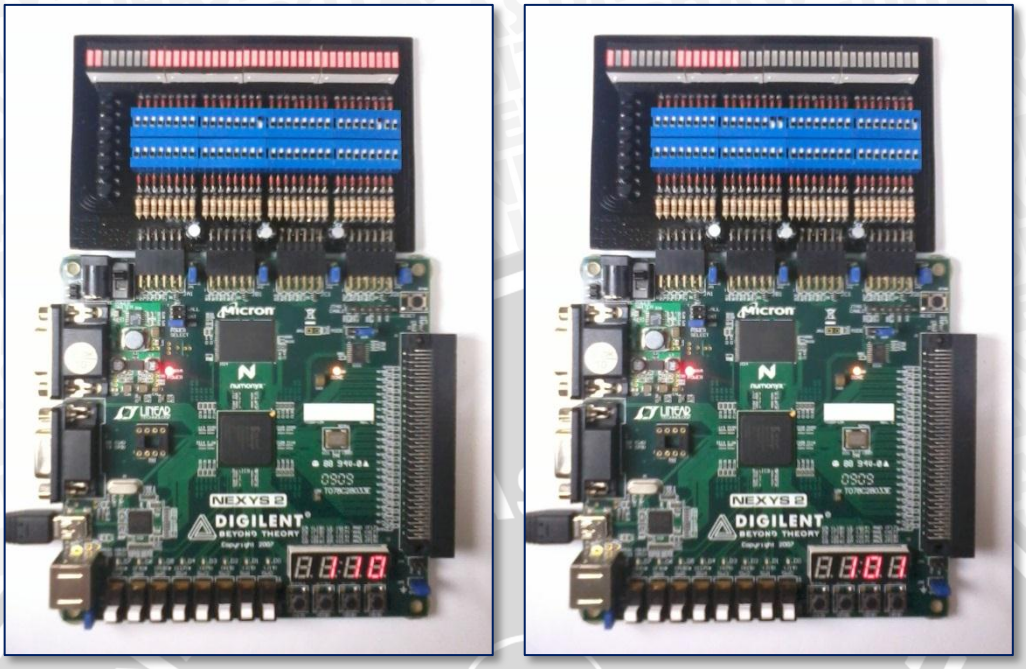
Pengujian dilakukan sebanyak 8 kali, yaitu memberi masukan *exception handler* dengan cara mengubah kondisi *dip-switches A* dan *dip-switches* B sesuai Tabel 5.7, dan kemudian melihat keluarannya yang berupa *status, invalid, divide by zero*, dan *output* melalui tampilan 40 Bar LED dan mencatatkan hasilnya kedalam Tabel 5.7. Berdasarkan pengujian yang dilakukan, didapatkan hasil pengujian yang ditunjukkan dalam Tabel 5.7.

Tabel 5.7. Hasil pengujian *Exception Handler*

No	Instr	Flag A	Flag B	Status	Invalid	Div by 0	Output
1.	00	0000	0000	0	0	0	FFFFFF
2.	00	1010	0010	1	1	0	FFFFFF
3.	01	0000	0100	1	1	0	FFFFFF
4.	01	1010	0010	1	0	0	FF8000
5.	10	0001	1010	1	1	0	FFFFFF
6.	10	1000	0001	1	0	0	800000
7.	11	0001	0001	1	1	0	FFFFFF
8.	11	0000	0001	1	0	1	7F8000

Keterangan:

- Data tertulis dalam bentuk angka biner, kecuali untuk kolom output yang tertulis dalam bentuk heksadesimal.
- Flag A terdiri dari urutan *Sign A, NaN A, Infinity A,* dan *Zero A*.
- Flag B terdiri dari urutan *Sign B, NaN B, Infinity B,* dan *Zero B*.



Gambar 5.14. (a) Hasil pengujian pada Tabel 5.7 nomor 3. (b) Hasil Pengujian pada Tabel 5.7 nomor 8.

5.2.5 Analisis Hasil Pengujian *Exception Handler*

Pada pengujian *exception handler* didapatkan hasil bahwa unit *exception handler* telah dapat menerima masukan yang diberikan dan dapat menampilkan hasil yang diinginkan.

Pada pengujian pertama, tidak ada *flag* (*Not a Number, Infinity, Zero*) yang aktif (berlogika 1) maka keluaran *status* berlogika 0, yang artinya tidak ada aktifitas perhitungan khusus. Dan keluaran lain diabaikan.

Pada pengujian kedua menunjukkan perhitungan minus *infinity* ditambah dengan *infinity* maka keluaran *invalid* sama dengan 1 dan *output* adalah *Not a Number* dalam bentuk IEEE 754 untuk bilangan *floating point* presisi tunggal.

Pada pengujian keempat menunjukkan perhitungan minus *infinity* dikurangi *infinity* maka kelurannya adalah minus *infinity* yang tersaji dalam bentuk standar IEEE 754 untuk bilangan *floating point* presisi tunggal.

Pada pengujian keenam menunjukkan perhitungan minus angka sembarang bukan termasuk angka khusus dikalikan dengan *zero* (nol) maka

keluarannya adalah nol yang tersaji dalam bentuk standar IEEE 754 untuk bilangan *floating point* presisi tunggal.

Pada pengujian kedelapan menunjukkan perhitungan angka sembarang bukan termasuk angka khusus dibagi dengan *zero* (nol) maka keluaran sinyal *divide by zero* berlogika 1 dan *output* bernilai tak berhingga yang tersaji dalam bentuk standar IEEE 754 untuk bilangan *floating point* presisi tunggal.

Dalam pengujian yang dilakukan, *exception handler* memberikan hasil keluaran sesuai dengan yang diharapkan dan tidak memberikan hasil *error*.

5.8 Pengujian *Floating Point Encoder*

5.8.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah *floating point encoder* berjalan sesuai dengan spesifikasi dan fungsi yang telah ditentukan, yaitu harus dapat memberikan keluaran data bilangan *floating point* standar IEEE 754 hasil dari perhitungan FPU. *Floating point encoder* bekerja berdasarkan data hasil normalisasi dan data dari unit *exception handler*.

5.8.2 Peralatan Pengujian

Peralatan yang digunakan dalam pengujian *floating point encoder* ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

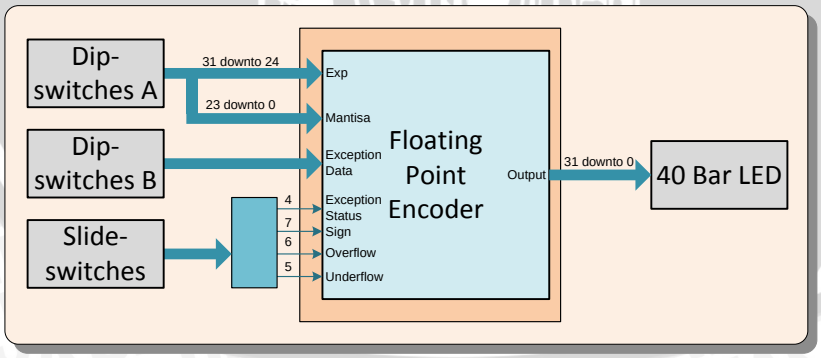
5.8.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian *floating point encoder* ini adalah sebagai berikut:

1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.15 (a).

3) Menetapkan *dip-switches A* sebagai masukan eksponen pada bit ke-31 sampai bit ke-24, dan sebagai masukan mantisa pada bit ke-23 sampai bit ke-0.
4) Menetapkan *dip-switches B* sebagai *exception data*.
5) Menetapkan *slide-switches* sebagai masukan dengan ketentuan:
   - Bit ke-7 sebagai masukan *Sign*.
   - Bit ke-6 sebagai masukan *Overflow.*
   - Bit ke-5 sebagai masukan *Underflow*.
   - Bit ke-4 sebagai masukan *Exception status*.
6) Menetapkan 40 Bar LED bit ke-31 sampai bit ke-0 sebagai keluaran *floating point encoder*.
7) Membuat program sesuai dengan diagram blok pengujian *exception handler* yang ditunjukkan dalam Gambar 5.15 (b).
8) Melakukan pengujian dengan memberi masukan *floating point encoder* dengan cara mengubah kondisi *dip-switches A* dan *dip-switches B* serta kondisi *slide-switches* sesuai dengan fungsi yang telah ditetapkan.
9) Mencatat *output* yang berupa data bilangan *floating point* standar IEEE 754 melalui tampilan 40 Bar LED.

(a)



(b)

Gambar 5.15 Peletakan *input output* SPARTAN XC3S500E untuk pengujian *floating point encoder*.
(b) Diagram blok pengujian *floating point encoder*.

5.8.4 Data Hasil Pengujian *Floating Point Encoder*

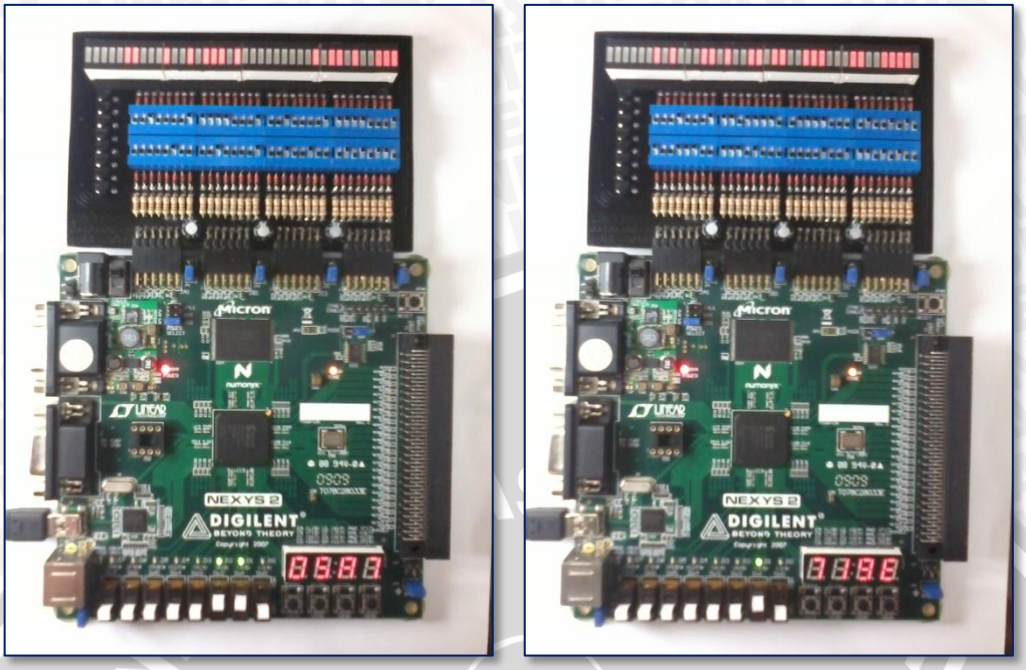
Pengujian dilakukan sebanyak 8 kali, yaitu memberi masukan *floating point encoder* dengan cara mengubah kondisi *dip-switches A*, *dip-switches* B, dan *slide-switches* sesuai Tabel 5.8, dan kemudian melihat keluarannya yang berupa bilangan *floating point* presisi tunggal standar IEEE 754 melalui tampilan 40 Bar LED dan mencatatkan hasilnya kedalam Tabel 5.8. Berdasarkan pengujian yang dilakukan, didapatkan hasil pengujian yang ditunjukkan dalam Tabel 5.8.

Tabel 5.8. Hasil pengujian *Floating point encoder*

No	Dip-switches A 31 - 24	Dip-switches A 23 - 0	Dip-switches B 31 - 0	Slide-switches 7 - 4	40 Bar LED 31 - 0
1.	00	000000	00000000	0000	00000000
2.	00	000000	7F800000	0001	7F800000
3.	00	000014	FFFFFFFF	0010	00000000
4.	F1	F73C51	FFFFFFFF	1100	FF800001
5.	21	D5681C	FFFFFFFF	1000	90D5681C
6.	24	EFF74C	FFFFFFFF	0000	126FF74C
7.	56	F4BEBE	FFFFFFFF	1000	AB74BEBE
8.	00	000000	80000000	0001	80000000

Keterangan:

- *Slide-switches* terdiri dari urutan masukan *Sign, Overflow, Underflow*, dan *Exception Status*.



Gambar 5.16. (a) Hasil pengujian pada Tabel 5.8 nomor 2. (b) Hasil Pengujian pada Tabel 5.8 nomor 6.

5.8.5 Analisis Hasil Pengujian *Floating Point Encoder*

Pada pengujian *floating point encoder* didapatkan hasil bahwa unit *floating point encoder* telah dapat menerima masukan yang diberikan dan dapat menampilkan hasil yang diinginkan yaitu berupa bilangan *floating point* presisi tunggal standar IEEE 754.

Pada pengujian kedua sinyal *exception status* yang ditunjukkan oleh *slide-switches* bit ke-4 bernilai 1 maka keluaran *floating point encoder* akan sama dengan nilai masukan *exception data* yang ditunjukkan oleh *dip-switches B* yaitu bernilai `7F800000h`.

Pada pengujian ketiga, sinyal masukan *underflow* berlogika 1 maka keluaran *floating point encoder* akan bernilai `00000000h`.

Pada pengujian keempat, sinyal masukan *overflow* berlogika 1 maka keluaran *floating point encoder* akan bernilai *Not a Number* atau `FF800001h`.

Pada pengujian ketujuh, sinyal masukan *sign* berlogika 1, sinyal masukan eksponen yang ditunjukkan *dip-switches A* bit ke-31 sampai bit ke-24 bernilai `56h`, dan sinyal masukan mantisa yang direpresentasikan oleh

kondisi *dip-switches A* bit ke-23 sampai bit ke-0 bernilai `F4BEBEh`. Maka gabungan ketiganya sesuai standar IEEE 754 untuk bilangan *floating point* presisi tunggal adalah `AB74BEBEh`.

Dalam pengujian yang dilakukan, *floating point encoder* memberikan hasil keluaran sesuai dengan yang diharapkan dan tidak memberikan hasil *error*.

5.9 Pengujian *Floating Point Unit*

5.8.1 Tujuan Pengujian

Pengujian ini dilakukan guna mengetahui apakah keseluruhan sistem *Floating Point Unit* termasuk didalamnya terdapat *control unit* dapat berjalan sesuai dengan spesifikasi dan fungsi yang telah ditentukan, yaitu harus dapat mengoperasikan dua buah bilangan *floating point* presisi tunggal standar IEEE 754 dengan operasi berupa penjumlahan, pengurangan, perkalian dan pembagian. Dan memberikan hasil operasi perhitungan dalam bentuk bilangan *floating point* presisi tunggal standar IEEE 754. FPU bekerja berdasarkan instruksi dan dua *operand* yang diberikan.

5.9.2 Peralatan Pengujian

Peralatan yang digunakan dalam pengujian FPU ini antara lain:

1) FPGA Xilinx 3E-500 FG320.
2) ZIFI-FPU Simulator sebagai unit input-output tambahan.
3) Komputer dengan dukungan USB 2.0 dengan daya 500 mA.
4) *Software* Xilinx ISE Project Navigator 11.1 untuk pemrogram.
5) *Software* Digilent Adept untuk antarmuka.

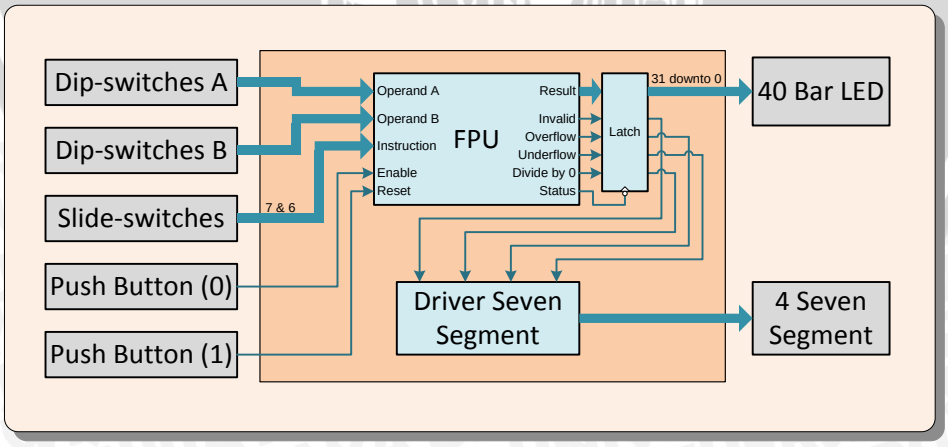
5.9.3 Prosedur Pengujian

Prosedur dalam melakukan pengujian FPU ini adalah sebagai berikut:

1) Menyiapkan peralatan yaitu FPGA dan Komputer sehingga siap untuk melakukan pemrograman.
2) Menentukan *input output* yang akan digunakan pada FPGA sesuai yang ditunjukkan pada Gambar 5.17 (a).
3) Menetapkan *dip-switches A* sebagai masukan *operand* A

4) Menetapkan *dip-switches B* sebagai masukan *operand* B
5) Menetapkan *slide-switches* bit ke-7 dan bit ke-6 sebagai masukan instruksi.
6) Menetapkan *push-button* bit ke-0 sebagai *enable* dan bit ke-1 sebagai *reset*.
7) Menetapkan 40 Bar LED bit ke-31 sampai bit ke-0 sebagai keluaran FPU.
8) Menetapkan 4 digit *seven-segment* sebagai keluaran berupa sinyal *invalid* pada digit ke-3, sinyal *divide by zero* pada digit ke-2, sinyal *overflow* pada digit ke-1, dan sinyal *underflow* pada digit ke-0.
9) Membuat program sesuai dengan diagram blok pengujian FPU yang ditunjukkan dalam Gambar 5.17 (b).
10) Melakukan pengujian dengan memberi masukan FPU dengan cara mengubah kondisi *dip-switches A* dan *dip-switches B* serta kondisi *slide-switches* bit ke-7 dan bit ke-6 sesuai dengan fungsi yang telah ditetapkan, kemudian menekan *push button* bit ke-0 sebagai sinyal *enable* agar FPU mulai melakukan perhitungan.
11) Mencatat *output* hasil perhitungan FPU yang berupa data bilangan *floating point* standar IEEE 754 melalui tampilan 40 Bar LED. Dan melihat sinyal khusus (*Invalid, Divide by zero, Overflow,* dan *Underflow*) melalui 4 digit *seven-segment*.

(a)



(b)

Gambar 5.17 Peletakan *input output* SPARTAN XC3S500E untuk pengujian keseluruhan sistem FPU (b) Diagram blok pengujian FPU.

5.9.4 Data Hasil Pengujian *Floating Point Unit*

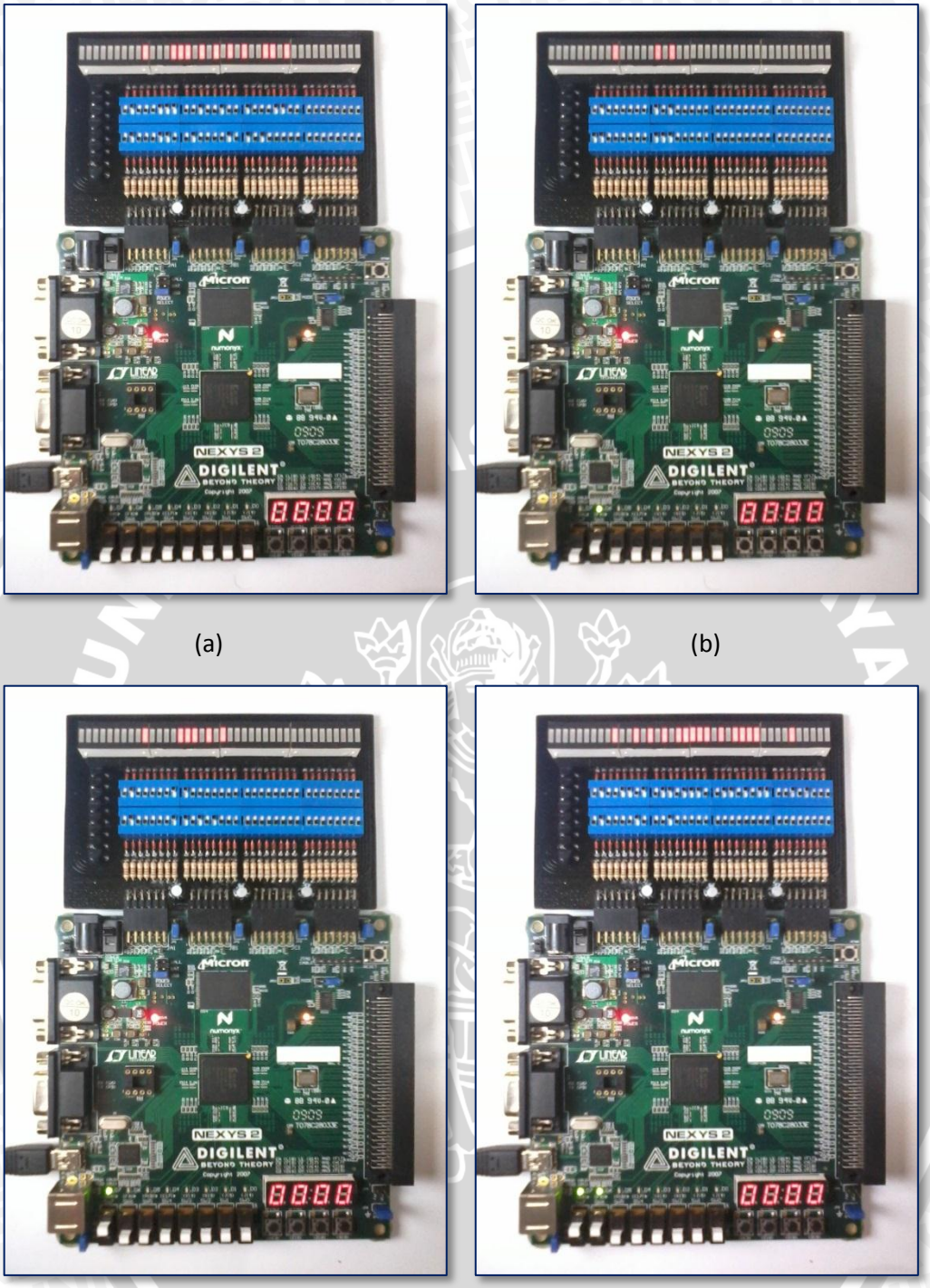
Pengujian dilakukan sebanyak 8 kali, yaitu memberi masukan FPU dengan cara mengubah kondisi *dip-switches A*, *dip-switches* B, dan *slide-switches* sesuai Tabel 5.9, dan kemudian melihat keluarannya yang berupa bilangan *floating point* presisi tunggal standar IEEE 754 melalui tampilan 40 Bar LED dan mencatatkan hasilnya kedalam Tabel 5.9. Berdasarkan pengujian yang dilakukan, didapatkan hasil pengujian yang ditunjukkan dalam Tabel 5.9.

Tabel 5.9. Hasil pengujian FPU

No	Instr	Op-A	Op-B	Result	Flag
1.	00	00000000	00000000	00000000	0000
2.	00	47248C00	4642A000	47553400	0000
3.	00	863A1788	000211A4	863A1745	0000
4.	00	7FEDCBA9	10630055	FFFFFFFF	1000
5.	00	00000001	00000001	00000002	0000
6.	00	007FFFFF	00000001	00800000	0000
7.	01	4E32D05E	4B3EBC20	4E2FD56D	0000
8.	01	7F800000	7F800000	FFFFFFFF	1000
9.	01	40A00000	C0E00000	41400000	0000
10.	01	3F800000	36EAE18A	3F7FFF8A	0000
11.	01	87654321	87654321	00000000	0000
12.	01	00000000	16072013	96072013	0000
13.	10	41800000	41A80000	43A80000	0000
14.	10	44F34000	3A102DE0	3F88FF97	0000
15.	10	000021F1	0A000028	00000000	0001
16.	10	7F800000	00000000	FFFFFFFF	1000
17.	10	80000001	70194624	A5994624	0000
18.	10	3E000000	41000000	3F800000	0000
19.	11	40400000	40A00000	3F199999	0000
20.	11	42F80000	0198724D	7F800001	0010
21.	11	4D6E6B28	42200000	4ABEBC20	0000
22.	11	C4FBA000	00000000	FF800000	0100
23.	11	80181288	000021F1	C3359056	0000
24.	11	2BAF31F6	2BAF31F6	3F800000	0000

Keterangan:

- Instr menunjukkan masukan instruksi yang disajikan dalam bentuk biner.
- Flag terdiri dari keluaran *Invalid, Divide by zero, Infinity*, dan *Underflow*.



(a) (b)

(c) (d)

Gambar 5.16. (a) Hasil pengujian pada Tabel 5.9 nomor 2. (b) Hasil Pengujian pada Tabel 5.9 nomor 7. (c) Hasil pengujian pada Tabel 5.9 nomor 9. (d) Hasil pengujian pada Tabel 5.9 nomor 15

5.9.5 Analisis Hasil Pengujian *Floating Point Unit*

Pada pengujian FPU didapatkan hasil bahwa FPU telah dapat melakukan perhitungan dua buah bilangan *floating point* presisi tunggal standar IEEE 754 dengan operasi perhitungan berupa penjumlahan, pengurangan, perkalian dan pembagian. Dan memberikan keluaran hasil operasi perhitungan dalam bentuk bilangan *floating point* presisi tunggal standar IEEE 754.

Analisis hasil pengujian akan dijabarkan dalam Tabel 5.10. Hasil dari perhitungan FPU akan dibandingkan dengan hasil perhitungan nilai yang setara dari *operand* dengan menggunakan *scientific calculator* yang sudah tersedia pada program standar windows7 64 bit.

Analisis hasil pengujian akan dijelaskan lebih lanjut dibawah ini, dengan mengacu data-data yang tertera pada Tabel 5.10.

Pengujian ke-1, menunjukkan FPU diberi dua masukan bilangan nol. Perhitungan yang dilakukan FPU setara dengan (+0)+(+0) dan hasilnya adalah +0. Perlu diingat bahwa format memori IEEE 754 mengenal bilangan 0 positif dan 0 negatif.

Pengujian ke-2, “hanyalah” perhitungan penjumlahan bilangan bulat dengan ketelitian 5 digit desimal. Ini merupakan contoh perhitungan yang dilakukan FPU secara sempurna.

Pengujian ke-3, terdapat perbedaan hasil perhitungan FPU dengan hasil perhitungan *calculator* windows7. Perbedaan hasil perhitungan mulai ada pada angka ke-7, ini disebabkan karena FPU memiliki ketelitian 32-bit atau sekitar 7 digit angka desimal.

Pengujian ke-4, salah satu masukan adalah *Not a Number*. Menjadi tugas *exception handler* untuk menangani perhitungan ini. Dan keluaran FPU adalah *Not a Number*. Perhitungan seperti ini tidak bisa kita jumpai ketika menggunakan *calculator* sebab kita tidak bisa memberikan masukan *Not a Number* kepada *calculator*.

Tabel 5.10. Analisis Hasil Pengujian FPU secara keseluruhan

No	Operasi	Operand A		Operand B		Hasil Perhitungan		Flag	Hasil *calculator*
		IEEE 754	Nilai Desimal	IEEE 754	Nilai Desimal	IEEE 754	Nilai Desimal		
1.	Jumlah	00000000	+0	00000000	+0	00000000	+0	-	0
2.	Jumlah	47248C00	42124 e0	4642A000	12456 e0	47553400	54580 e0	-	54580
3.	Jumlah	863A1788	-3.5000000e-35	000211A4	1.8999926e-40	863A1745	-3.4999808e-35	-	-3.499981000074e-28
4.	Jumlah	7FEDCBA9	Not a Number	10630055	4.4768112e-29	FFFFFFFF	Not a Number	Invalid	-
5.	Jumlah	00000001	1.401298…e-45	00000001	1.401298…e-45	00000002	2.802596…e-45	-	2.80259692864…e-45
6.	Jumlah	007FFFFF	1.175494…e-38	00000001	1.401298…e-45	00800000	1.175494…e-38	-	1.175494350822…e-38
7.	Kurang	4E32D05E	7.5000000e8	4B3EBC20	12500000	4E2FD56D	7.3749997e8	-	737500000
8.	Kurang	7F800000	Infinity	7F800000	Infinity	FFFFFFFF	Not a Number	-	-
9.	Kurang	40A00000	5.0000000	C0E00000	-7.0000000	41400000	12.000000	-	12
10.	Kurang	3F800000	1.0000000	36EAE18A	6.9999996e-6	3F7FFF8A	9.9999297e-1	-	0.9999930000004
11.	Kurang	87654321	-1.7247773e-34	87654321	-1.7247773e-34	00000000	+0	-	0
12.	Kurang	00000000	0	16072013	1.0915334e-25	96072013	-1.0915334e-25	-	-1.0915334e-25
13.	Kali	41800000	16.000000	41A80000	21.000000	43A80000	336.00000	-	336
14.	Kali	44F34000	1946.0000	3A102DE0	5.5000000e-4	3F88FF97	1.0703000	-	1.0703
15.	Kali	000021F1	1.2175882e-41	0A000028	6.1630052e-33	00000000	0	underflow	8.055929440805864e-74
16.	Kali	7F800000	Infinity	00000000	Zero	FFFFFFFF	Not a Number	Invalid	-
17.	Kali	80000001	-1.4012985e-45	70194624	1.8974401e+29	A5994624	-2.6588798e-16	-	-2.65887996596985e-16
18.	Kali	3E000000	1.2500000e-1	41000000	8.0000000	3F800000	1.0000000	-	1
19.	Bagi	40400000	3.0000000	40A00000	5.0000000	3F199999	5.9999996e-1	-	0.6
20.	Bagi	42F80000	124.00000	0198724D	5.5999995e-38	7F800001	Not a Number	overflow	2.21428591198981…e+39
21.	Bagi	4D6E6B28	2.5000000e8	42200000	40.000000	4ABEBC20	6250000.0	-	6250000
22.	Bagi	C4FBA000	-2013.0000	00000000	0.0000000	FF800000	Infinity	Div by 0	Cannot Divide by Zero
23.	Bagi	80181288	-2.2106997e-39	000021F1	1.2175882e-41	C3359056	-181.56381	-	-181.563824288047469…
24.	Bagi	2BAF31F6	1.2448365e-12	2BAF31F6	1.2448365e-12	3F800000	1.0000000	-	1

Pengujian ke-5, perhitungan penjumlahan antara dua nilai terkecil yang dapat diwakili oleh IEEE 754 presisi tunggal, yaitu penjumlahan antara 00000001h dengan 00000001h adalah 00000002h. Bentuk IEEE 754 00000001h sebenarnya mewakili nilai 1.401298464324817070923729583289916131280261941876515771757 07e-45 dan hasil perhitungannya secara desimal adalah 2.802596928649634141847459166579832262560523883753031543514 14e-45 merupakan nilai yang terwakili dengan sempurna oleh 00000002h. Namun pada *calculator* windows7 mode *scientific* hanya dapat memberikan masukan sebanyak 32 deret angka yaitu 1.401298464324817070923729583289 9e-45, sehingga hasil perhitungannya yaitu 2.802596928649634141847459166579 8e-45. Hasil perhitungan antara FPU dengan *calculator* sama untuk 32 deret angka pertama.

Pengujian ke-6, penjumlahan angka terdenormalisasi terbesar yang dengan angka terdenormalisasi terkecil yang dapar diwakili oleh IEEE 754, 007FFFFFh mewakili nilai 1.175494210692441075487029444849287348827052428745893333857 17e-38, dan hasil perhitungannya secara desimal adalah 1.175494350822287507968736537222245677818666555677208752150875e-38 merupakan nilai yang terwakili sempurna oleh 00800000h. Dalam pengujian ini hanya 32 deret angka yang bisa dibandingkan dengan *calculator*, dan hasil *calculator* untuk 32 deret angka pertama adalah sama dengan hasil perhitungan FPU yaitu 1.175494350822287507968736537222 2e-38.

Pengujian ke-7, terjadi perbedaan hasil antara FPU dengan *calculator*, meskipun hasil sebenarnya dari perhitungan (7.375e8) hanya memiliki 4 angka penting, namun ekspresi biner dari 7.375 membutuhkan 25 deret biner (1.01011111101010101011011) sehingga tidak bisa diekspresikan secara sempurna melalui IEEE 754 presisi tunggal yang hanya memiliki ketelitian mantisa sebesar 24 bit. Jadi perbedaan hasil perhitungan terjadi karena ketelitian yang dimiliki FPU.

Pengujian ke-8, kedua masukan adalah angka khusus yang bernilai *infinity*. Maka perhitungan ini akan ditangani oleh *exception handler*. Dimana hasil dari *Infinity* dikurangi *Infinity* tidak memiliki hasil eksak, maka

keluaran FPU akan menjadi *Not a Number* disertai sinyal *Invalid* menjadi aktif (berlogika 1). Pengujian ini tidak bisa dibandingkan dengan *calculator* sebab kita tidak bisa memberikan masukan *Infinity* kepada *calculator*.

Pengujian ke-9, merupakan perhitungan sangat sederhana. Angka masukan hanya angka satuan bulat dalam bentuk desimal. Merupakan contoh perhitungan sempurna yang dapat dilakukan oleh FPU.

Pengujian ke-10, adalah pengurangan dengan hasil sebenarnya (yang direpresentasikan oleh *calculator*) membutuhkan ketelitian 13 digit angka desimal. Merupakan hal yang tidak bisa dicapai FPU dan tidak bisa direpresentasikan dengan format memori IEEE 754 32 bit. Hal ini menyebabkan perbedaan keluaran antara FPU dengan *calculator*.

Pengujian ke-11, merupakan pengujian pengurangan dua angka yang sama. Dan hasil dari perhitungan dua angka yang sama oleh FPU adalah 0, hasil yang sama juga ditunjukkan oleh *calculator*.

Pengujian ke-12, pengujian bilangan 0 dikurangi dengan suatu bilangan positif, maka hasilnya adalah negatif dari bilangan pengurang. Dalam hal ini angka pengurang adalah 1.0915334e-25 maka hasil dari FPU adalah -1.0915334e-25. Hasil yang sama ditunjukkan oleh *calculator*, berarti bahwa FPU melakukan perhitungan ini dengan sempurna.

Pengujian ke-13, adalah perhitungan perkalian bilangan bulat dengan hasil 3 digit angka desimal. Merupakan contoh perhitungan perkalian yang dapat dilakukan FPU dengan sempurna, hal ini dibuktikan oleh hasil perhitungan yang sama antara FPU dengan *calculator*.

Pengujian ke-14, juga contoh perhitungan perkalian yang dapat dilakukan oleh FPU dengan sempurna, terbukti dari kesamaan hasil perhitungan antara FPU dengan *calculator*.

Pengujian ke-15, hasil perhitungan yang ditunjukkan *calculator* memiliki eksponen -74, eksponen ini lebih kecil dari yang bisa direpresentasikan oleh IEEE 754 *floating point* presisi tunggal.

Pengujian ke-16, adalah contoh perhitungan *invalid* dalam matematika dimana tidak ada hasil eksak dari perhitungan *Infinity* dikali dengan *Zero*. Perhitungan ini tidak bisa dibandingkan dengan *calculator*

sebab kita tidak bisa memberikan masukan *Infinity* sebagai masukan perhitungan dalam *calculator*.

Pengujian ke-17, perhitungan antara bilangan denormalisasi dengan bilangan dengan eksponen cukup besar untuk tidak terjadi *Underflow*. Ada perbedaan hasil perhitungan antara FPU dengan *calculator* yang terletak pada angka kedelapan. Perbedaan hasil perhitungan ini dikarenakan tingkat presisi yang dimiliki FPU.

Pengujian ke-18, merupakan perhitungan 2 angka berkomplemen dalam perkalian yaitu 0,125 dikali 8. Hasil FPU menunjukkan nilai 1 dan hasil yang sama ditunjukkan oleh *calculator.*

Pengujian ke-19, merupakan perhitungan yang relatif mudah dalam perhitungan desimal. Pembagian 3 dengan 5 menghasilkan bilangan rasional 0,6. Namun bilangan 0,6 akan menjadi irrasional ketika direpresentasikan dalam biner (0.1001 1001 1001 1001 1001 1001…), hal ini diakibatkan karena perbedaan titik ketelitian antara bilangan biner dengan bilangan desimal. Jadi bisa dikatakan bahwa perbedaan hasil perhitungan antara FPU dengan *calculator* bukan sepenuhnya kesalahan FPU.

Pengujian ke-20, hasil sebenarnya yang ditunjukkan *calculator* adalah 2,21428591198981357051906879634 54e+39 sebuah angka yang lebih besar dari jangkauan IEEE 754 presisi tunggal ($1{,}1754942 \times 10^{-38}$) maka keluaran FPU menjadi *Not a Number* disertai sinyal *Overflow* aktif.

Pengujian ke-21, adalah pengujian pembagian antara bilangan bulat dengan bilangan bulat dengan hasil yang bulat. Menunjukkan perhitungan yang sempurna yang dilakukan FPU, ini dibuktikan dari hasil yang sama antara FPU dengan *calculator*.

Pengujian ke-22, contoh perhitungan angka sembarang (bukan *Infinity*) dibagi dengan nol. Hasilnya adalah *Infinity* disertai dengan sinyal *Divide by Zero* aktif. Hasil ini sama dengan *calculator* yang menunjukkan hasil pesan “Cannot Divide by Zero”.

Pengujian ke-23, perhitungan pembagian dua angka terdenormalisasi hasil *calculator* adalah -181.563824288047469579616491027 1 dan hasil FPU adalah -181.56381. Perbedaan hasil perhitungan ini juga disebabkan oleh tingkat presisi yang dimiliki FPU.

Pengujian ke-24, merupakan pengujian dua angka yang sama yang mana hasil perhitungan sebenarnya yang ditunjukkan oleh *calculator* adalah 1, dan FPU menunjukkan hasil perhitungan yang sama.

Dalam pengujian yang dilakukan, FPU telah bekerja dengan benar dengan memberikan keluaran yang diharapkan dan tidak memberikan hasil *error*.