PURWARUPA MIKROPROSESOR BERBASIS FPGA ALTERA EPF10K10 DENGAN DESKRIPSI VHDL

Agfianto Eko Putra[1], Arsyad Muhammad Fajri[2]
[1,2]) Program Studi Elektronika & Instrumentasi, Jurusan Fisika
Fakultas MIPA, Universitas Gadjah Mada, Sekip Utara, Yogyakarta 55281
Telp. (0274) 902382 – email: agfi@ugm.ac.id

Abstract
It has been designed and implemented an FPGA-based microprocessor prototype using Altera EPF10K10 and VHDL description then compiled and simulate using MAX+Plus II software. The microprocessor prototype is implementing using the Wizard A-01 development board and its assembly program stored in ROM. To decode and execute the instruction, it used Control Unit, which will send control signal to other components. The 16 instructions is implementing in this microprocessor prototype. This microprocessor prototype has 8-bit data bus and 4-bit address bus, implemented using 375 logic cells, operating at 14.72 MHz clock (maximum) and 3.68 MIPS.

***Keywords**:*
FPGA, microprocessor, VHDL, Altera, EPF10K10

1. PENDAHULUAN

Mikroprosesor telah berkembang menjadi bagian penting dalam dunia elektronika. Untuk menggunakan mikroprosesor, pengguna tinggal menyusun suatu program dengan memanfaatkan instruksi mesin (*assembly*) yang dimiliki oleh mikroprosesor tersebut. Adanya komponen register serta *Arithmetic and Logic Unit* (ALU) memungkinkan mikroprosesor dapat menjalankan berbagai macam komputasi, mulai dari komputasi sederhana hingga yang kompleks.

Menurut Turley (2002) kendala yang dihadapi dalam perancangan mikroprosesor adalah mahal dan lamanya fabrikasi. Untuk mengatasi kendala waktu dan biaya fabrikasi, maka sebuah mikroprosesor dapat diimplementasikan ke dalam *Field Programmable Array* (FPGA).

FPGA merupakan piranti yang dapat dikonfigurasi-ulang (*reconfigurable*). FPGA memiliki komponen kombinasional dan sekuensial dalam tiap sel logik-nya, sehingga memungkinkan dapat digunakan untuk implementasi rangkaian kombinasional maupun sekuensial. Melalui teknologi FPGA, implementasi rancangan sistem digital dapat dilakukan secara cepat (Maxfield, 2004).

Tujuan penelitian ini adalah merancang dan mengimplementasikan sebuah mikroprosesor sederhana (purwarupa mikroprosesor) dalam FPGA menggunakan deskripsi VHDL (*VHSIC Hardware Description Language*).

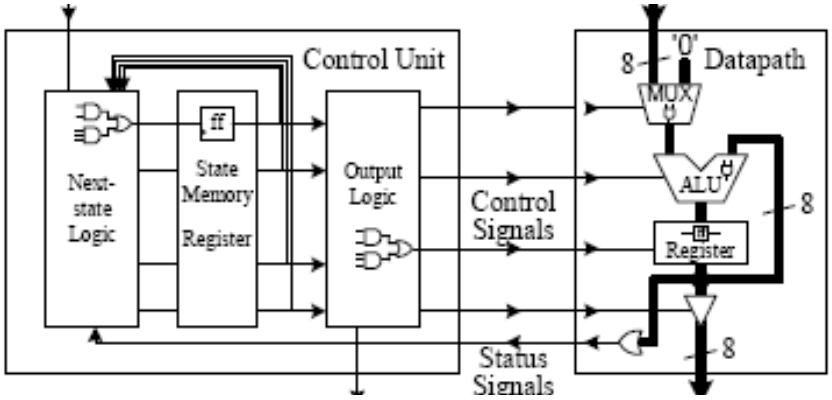
2. TINJAUAN PUSTAKA

2.1. MIKROPROSESOR

Menurut Hwang (2005), rangkaian digital dalam mikroprosesor dapat dibagi menjadi dua bagian yakni, *datapath* dan unit kontrol (*control unit*), sebagaimana ditunjukkan pada Gambar 1. *Datapath* berfungsi untuk menjalankan operasi pada data oleh mikroprosesor seperti penjumlahan dan operasi transfer data.

Di dalam *datapath*, terdapat ALU dan register yang dihubungkan oleh bus data. ALU berfungsi untuk melakukan operasi aritmatika dan logika pada data, sedangkan register-register dalam *datapath* berfungsi untuk menampung hasil operasi ALU, serta menampung data yang akan diolah oleh ALU (Hwang, 2005).

Control unit berfungsi untuk mengendalikan operasi pada *datapath* dan keseluruhan mikroprosesor. *Control unit* merupakan sebuah FSM (*Finite State machine*) yang bekerja dari satu kondisi (*state*) ke kondisi lainnya dengan sinkronisasi detak (Hwang, 2004).

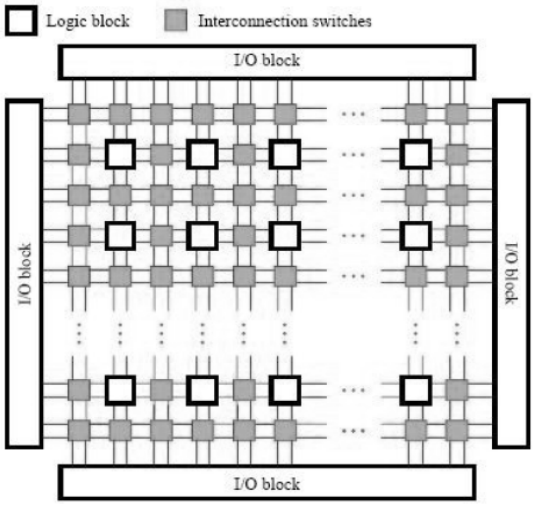


Gambar 1. Bagian dalam Mikroprosesor (Hwang, 2005)

2.2. FPGA (*Field Programmable Gate Array*)

Field Programmable Gate Array atau FPGA adalah sebuah rangkaian terpadu atau *Integrated Circuit* (IC) digital yang berisi sekumpulan blok logika dan blok interkoneksi yang dapat dikonfigurasi. FPGA dapat dikonfigurasi untuk menjalankan banyak sekali fungsi digital (Maxfield, 2004).

Pada Gambar 2 ditunjukkan struktur internal FPGA. FPGA memiliki 3 komponen penyusun yaitu blok logika, blok I/O dan blok koneksi. Blok-blok logika maupun hubungan antar blok dapat dikonfigurasi (Zeidman, 2004).



Gambar 2. Struktur Internal FPGA (Zeidman, 2004)

2.3. VHDL (*VHSIC Hardware Description Language*)

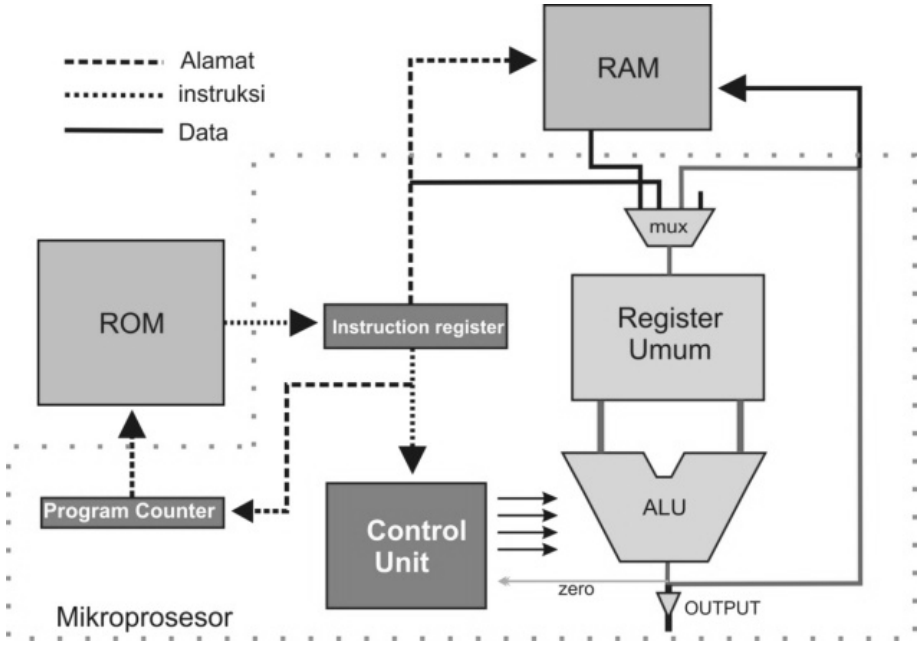
VHDL merupakan kepanjangan dari *VHSIC (Very High Speed IC) Hardware Description Language*. Sebuah rancangan VHDL terdiri atas dua bagian utama yakni, deklarasi entitas (*entity*) dan deklarasi arsitektur (*architecture*). Bagian deklarasi entitas, mendeskripsikan antarmuka rancangan, sedangkan bagian arsitektur berisi operasi internal dari rancangan (Chu, 2006).

3. METODOLOGI PENELITIAN DAN PERANCANGAN

3.1. ARSITEKTUR MIKROPROSESOR

Pada Gambar 3 ditunjukkan diagram blok sistem mikroprosesor yang dirancang. Masukannya berupa sederetan instruksi yang tersimpan dalam ROM. *Program Counter* (PC) menunjuk ke alamat ROM, tempat instruksi yang akan dieksekusi disimpan. Instruksi yang ditunjuk, akan dikirim ke *Instruction Register* (IR), kemudian instruksi diteruskan ke Unit Kontrol. Di dalam Unit Kontrol, instruksi akan diterjemahkan sehingga sinyal yang mengendalikan ALU dan Register Umum untuk melakukan operasi data. Dalam rancangan ini, mikroprosesor memiliki sebuah port keluaran yang terhubung ke keluaran ALU.

Mikroprosesor yang dirancang memiliki lebar jalur data 8-bit dan jalur alamat 4-bit. Dengan demikian, semua elemen yang terhubung ke jalur data memiliki lebar 8-bit. Sedangkan komponen yang terhubung ke jalur alamat yakni PC memiliki lebar data 4-bit.



Gambar 3 : Rancangan Mikroprosesor

Tabel 1. Set Instruksi Purwarupa Mikroprosesor

Instruksi	Format instruksi	Jumlah detak	Operasi
Tanpa operan			
NOP	0000 xxxx	2	
Satu Operan Register			
NOTL	0101 rr xx	4	rr ← NOT rr
MDK	1000 rr xx	4	rr ← rr + 1
MDN	1001 rr xx	4	rr ← rr – 1
KLR	1010 xx RR	4	Out ← RR
Dua Operan Register			
SLN	0001 rr RR	3	rr ←RR
ANDL	0011 rr RR	4	rr ← rr AND RR
ORL	0100 rr RR	4	rr ← rr OR RR
SUDO	0111 rr RR	4	rr ← rr – RR
TMB	0110 rr RR	4	rr ← rr + RR
Operan Data Langsung			
LD	0010 rr xx dddddddd	3	rr ← dddddddd
Operan Alamat			
LMP	1011 xxxx 0000aaaa	4	Pc ← aaaa
MTL	1100 xxRR 0000aaaa	4	M[aaaa] ← RR
LBZ	1101 xxxx 0000aaaa	4	Jika (z=0) maka pc←aaaa
MBC	1110 xxRR 0000aaaa	3	RR←M[aaaa]

Operasi yang dijalankan oleh mikroprosesor ditentukan oleh instruksi yang dieksekusi.Kumpulan instruksi yang dapat dijalankan oleh suatu mikroprosesor disebut set instruksi (He, 2002). Pada Tabel 1 ditunjukkan set instruksi yang diimplementasikan dalam rancangan mikroprosesor.

Untuk memudahkan pendekodean instruksi, format instruksi dibuat tetap. Empat bit pertama merupakan *opcode*, yang mewakili satu instruksi. Empat bit berikutnya mewakili dua bit operan register tujuan dan dua bit operan register sumber. Pada instruksi LD, format instruksi menjadi 16-bit. Delapan bit tambahan digunakan sebagai operan data langsung yang akan dimuat ke register. Format instruksi 16-bit juga digunakan pada operasi LMP, LBZ, MTL dan MBC yang membutuhkan operan alamat memori. Operan alamat yang digunakan unutuk menunjuk lokasi memori disimpan di 4-bit terakhir instruksi.

Mikroprosesor yang dirancang dibagi dalam sejumlah modul. Modul-modul tersebut menjalankan fungsi dari masing-masing komponen dalam mikroprosesor. Modul-modul yang telah dibuat selanjutnya digabung membentuk rancangan sebuah mikroprosesor.

3.2. ALU (*ARITHMETIC LOGIC UNIT*)

ALU atau *Arithmetic Logic Unit* berfungsi untuk menjalankan operasi aritmatika dan logika. Operasi ALU bergantung pada masukan ALU_sel mengikuti Tabel 2.

Tabel 2. Operasi ALU

ALU_sel	Operasi
000	Alu_out = alu_inb
001	Alu_out = Alu_ina AND alu_inb
010	Alu_out = Alu_ina OR alu_inb
011	Alu_out = NOT Alu_ina
100	Alu_out = Alu_ina + alu_inb
101	Alu_out = Alu_ina – alu_inb
110	Alu_out = Alu_ina + 1
111	Alu_out = Alu_ina -1

Dalam rancangan ALU, menggunakan deskripsi VHDL, digunakan pernyataan *case* untuk menentukan operasi yang dijalankan oleh ALU. Pernyataan *case* akan memeriksa masukan ALU_sel, kemudian menjalankan operasi yang sesuai. Misalnya, masukan ALU_sel adalah "001", maka rancangan akan menjalankan operasi logika AND terhadap dua buah masukan data. Deskripsi VHDL-nya sebagai berikut:

```
library IEEE;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;

entity alu is port (
      alu_sel: in std_logic_vector (2 downto 0);
      alu_inb: in std_logic_vector (7 downto 0);
      alu_ina: in std_logic_vector (7 downto 0);
      alu_out: out std_logic_vector (7 downto 0));
end alu;

architecture behavior of alu is
begin
      process(alu_sel,alu_ina,alu_inb)
      begin
        case alu_sel is
          when "001" => alu_out <= alu_ina AND alu_inb;
          when "010" => alu_out <= alu_ina OR alu_inb;
          when "011" => alu_out <= NOT alu_ina;
          when "100" => alu_out <= alu_ina + alu_inb;
          when "101" => alu_out <= alu_ina - alu_inb;
          when "110" => alu_out <= alu_ina + 1; --inc
          when "111" => alu_out <= alu_ina - 1; --dec
          when others => alu_out <= alu_inb;
```

```
        end case;
      end process;
end behavior;
```

3.3. REGISTER-REGISTER UMUM (*GENERAL REGISTERS*)

Register Umum berfungsi untuk menyimpan data sementara. Operasi yang dijalankan oleh ALU akan mengambil masukan dari Register Umum. Begitu juga data hasil operasi ALU akan disimpan kembali dalam Register Umum.

Modul Register Umum memiliki 4 buah register yakni **r0**, **r1**, **r2** dan **r3** yang masing-masing memiliki lebar 8-bit. Register Umum memiliki dua buah keluaran. Masing-masing keluaran berjalan secara paralel dan terhubung ke port masukan ALU.

3.4. MULTIPLEKSER

Multiplekser (atau mux) berfungsi untuk memilih masukan data ke register. Dalam rancangan ini, register dapat menerima masukan data dari 3 buah sumber yakni, data langsung, hasil operasi ALU serta dari memori. Keluaran Multiplekser terhubung ke modul Register Umum.

3.5. KENDALI KELUARAN

Kendali Keluaran digunakan sebagai "gerbang" bagi port keluaran mikroprosesor. Masukan dari Kendali Keluaran terhubung ke bus internal mikroprosesor. Keluaran-nya terhubung langsung ke port keluaran mikroprosesor.

3.6. PC (*PROGRAM COUNTER*)

Program counter atau PC berfungsi untuk menunjuk alamat instruksi yang akan dieksekusi. Saat dilakukan eksekusi program, isi PC akan dinaikkan satu setiap instruksi dijalankan, kecuali jika terdapat instruksi lompat. Jika instruksi lompat dieksekusi, maka PC akan diisi dengan nilai alamat yang ditunjuk oleh operan instruksi lompat yang bersangkutan.

3.7. IR (*INSTRUCTION REGISTER*)

Instruction register atau IR berfungsi untuk menyimpan instruksi yang akan dijalankan oleh mikroprosesor (Tanenbaum, 2001). Masukan IR terhubung ke ROM, untuk menerima masukan instruksi yang dijalankan. Keluaran IR terhubung ke Unit Kontrol, untuk diterjemahkan dan kemudian dieksekusi.

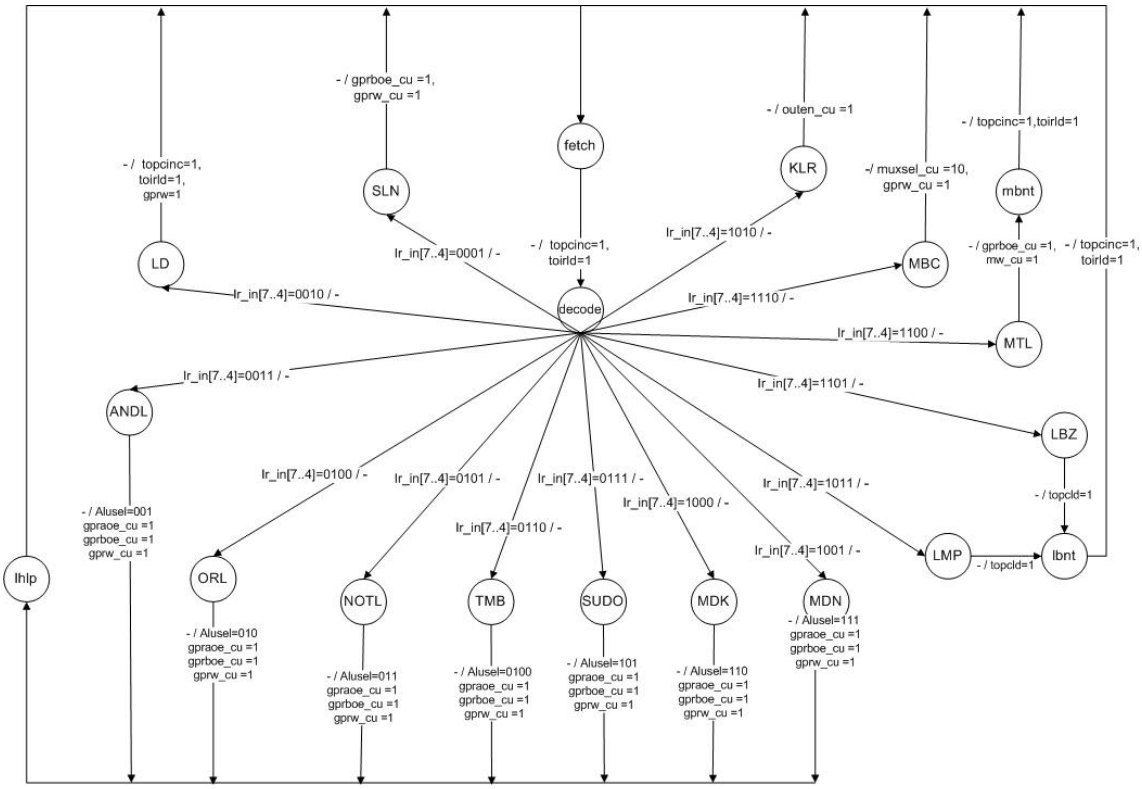
3.8. UNIT KONTROL

Unit Kontrol berfungsi untuk menerjemahkan instruksi dan mengengendalikan mikroprosesor (Rafiquzzaman, 2005). Unit Kontrol menghasilkan sinyal kendali untuk mengendalikan mikroprosesor. Unit Kontrol megendalikan mikroprosesor untuk menjalankan siklus ambil-terjemahkan-eksekusi atau *fetch-decode-execute*. Unit Kontrol menerima masukan dari IR, instruksi dari IR selanjutnya diterjemahkan dan dihasilkan sinyal kendali ke eleman lain dalam mikroprosesor. Kode VHDL untuk rancangan Unit Kontrol, dibuat sesuai dengan diagram kondisi yang ditunjukkan pada Gambar 4.

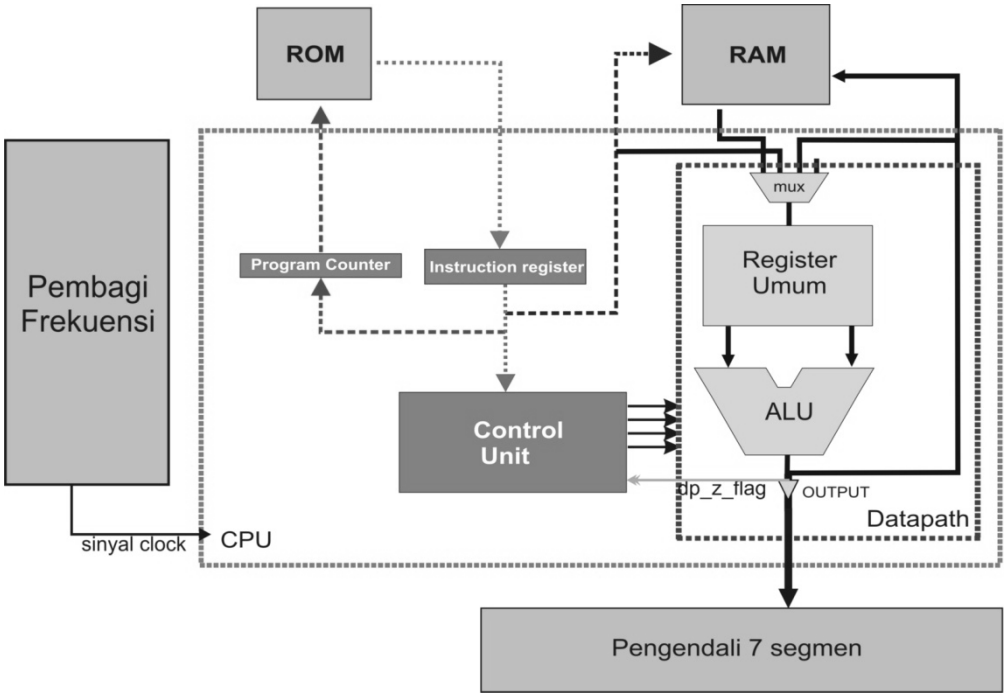
3.9. PENGGABUNGAN MODUL

Modul-modul yang telah dirancang, selanjutnya digabung ke dalam satu modul. Penggabungan modul-modul ini dilakukan dengan deskripsi VHDL dengan model struktural, untuk memudahkan penggabungan dan pemeriksaan terhadap hubungan antar modul. Modul-modul yang telah dirancang tidak langsung digabung dalam satu modul. Modul ALU, Multiplexer, Register Umum serta Kendali Keluaran digabung dalam modul *datapath*. Selanjutnya, *datapath* digabung dengan PC, IR serta control unit ke dalam modul CPU.

Modul modul yang telah dibuat digabung sesuai Gambar 5. Modul RAM, ROM, pengendali 7 segmen serta pembagi frekuensi digunakan untuk melakukan pengujian dan implementasi rancangan ke dalam papan pengembangan.



Gambar 4. Diagram kondisi Unit Kontrol



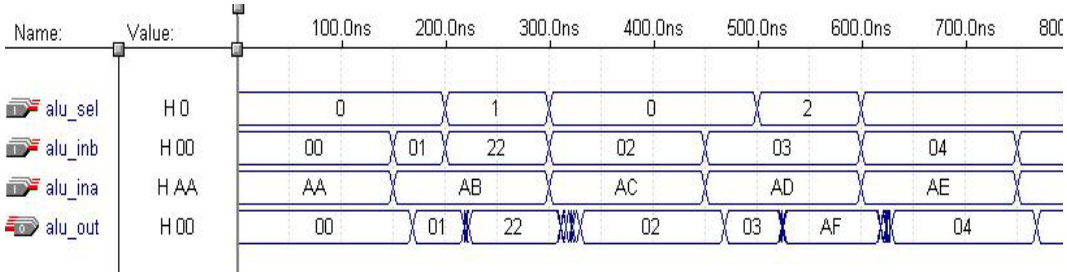
Gambar 5. Diagram penggabungan modul-modul

4. HASIL DAN PEMBAHASAN

4.1. ANALISIS SIMULASI

Untuk memeriksa kebenaran rangkaian rancangan mikroproesor, dilakukan simulasi fungsional dan pewaktuan. Simulasi fungsional tiap-tiap modul berjalan dengan baik karena saat dilakukan simulasi, tiap-tiap modul menunjukkan keluaran yang benar atau sesuai dengan yang diharapkan.

Dalam simulasi pewaktuan ditemukan adanya *glitch* pada rancangan. *Glitch* terjadi karena perbedaan tundaan rambatan pada jalur rancangan (Zeidman, 2004). *Glitch* muncul pada hampir setiap modul rancangan mikroprosesor ini saat dilakukan simulasi pewaktuan. *Glitch* yang muncul pada rancangan mikroprosesor berlangsung sangat singkat dan tidak mempengaruhi keluaran, sehingga masih dapat diterima. Pada Gambar 6 ditunjukkan contoh *glitch* yang terjadi saat simulasi pewaktuan ALU.



Gambar 6. *Glitch* pada simulasi pewaktuan ALU

4.2. ANALISIS PEWAKTUAN

Untuk melakukan analisis pewaktuan digunakan *Timing Analyzer* dari perangkat lunak Max+Plus II. Dengan *Timing Analyzer* dapat diperoleh tundaan rambatan terpanjang pada tiap-tiap jalur. Pada tabel 3 ditunjukkan tundaan rambatan pada rancangan mikroprosesor.

Tabel 3. Tundaan Rambatan

Tujuan	Tundaan rambatan dari sumber detak (ns)
addram (ke alamat RAM)	12.92
outram (ke data RAM)	55.04
Rata-rata ke RAM	38.85
ROM	13.2
Port Keluaran	60.3125

Selain diperoleh informasi tundaan rambatan, juga diperoleh informasi tundaan rambatan rancangan atau kecepatan tanggap rangkaian. Berdasarkan analisis dengan *Timing Analyzer*, diketahui mikroprosesor yang dirancang dapat diberi detak dengan frekuensi maksimal 14,72 Mhz.

Kinerja Mikroprosesor sering diukur dengan satuan MIPS (*Million Instruction per second*). MIPS menyatakan jumlah instruksi yang dapat dieksekusi dalam satu detik (Tanenbaum, 2001). Untuk memperoleh kecepatan mikroprosesor dalam satuan MIPS kecepatan maksimal mikroprosesor dibagi dengan jumlah detak tiap satu instruksi. Dalam rancangan ini satu instruksi rata-rata membutuhkan 4 detak. Sehingga kecepatan Mikroprosesor dalam satuan MIPS adalah:

$$\frac{14{,}72M}{4} = 3{,}68MIPS$$

Jadi kecepatan mikroprosesor dalam satuan MIPS adalah 3,68 MIPS, yang berarti Mikroprosesor dapat mengeksekusi 3,68 juta instruksi dalam 1 detik.

4.3. ANALISIS SUMBER DAYA FPGA

Sebagaimana ditunjukkan pada Tabel 4, rancangan Mikroprosesor yang telah dibuat membutuhkan 375 *Logic Element*. Modul yang paling banyak menggunakan sumber daya adalah Unit Kontrol, karena banyak menggunakan sumber daya untuk menjalankan fungsi-fungsi yang kompleks.

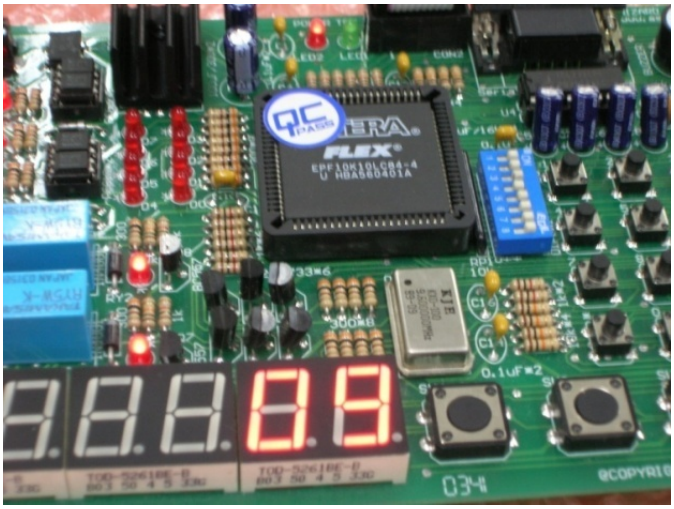
Tabel 4. Penggunaan Sumber Daya

Modul	LE atau flipflop	EAB atau embedded cell
ALU	93/0	0/0
Register Umum	82/32	0/0
Multiplexer	24/0	0/0
Kendali Keluaran	8/0	0/0
Unit Kontrol	146/44	0/0
Pc	9/4	0/0
Ir	8/8	0/0
Rom	0/0	8/1
Ram	0/0	8/1
CPU	**375/88**	**0/0**
Implementasi	453/111	16/2

4.4. PENGUJIAN DENGAN PROGRAM

Untuk melakukan pengujian dan analisis hasil implementasi, maka purwarupa Mikroprosesor yang telah dibuat harus diuji dengan menjalankan program (He, 2002). Program yang digunakan untuk melakukan ujicoba disimpan dalam file MIF. Selanjutnya program dimuat ke dalam ROM pada saat proses kompilasi. Pada gambar 7 ditunjukkan contoh tampilan hasil eksekusi program. Contoh salah satu dari beberapa program yang digunakan dalam uji coba ditunjukkan berikut (program penjumlahan register **r0** dan **r1**):

```
0: 00100000;  --   LD r0,5
1: 00000101;
2: 00100100;  --        LD r1,4
3: 00000100;
4: 01100001;  --        TMB r0,r1
5: 10100001;  --        KLR r1
6: 10100000;  --        KLR r0
7: 10110000;  --   LMP 5
8: 00000101;
```



Gambar 7 : Contoh tampilan hasil eksekusi pada papan pengembang

5. KESIMPULAN

Rancangan Mikroprosesor yang telah dibuat dengan VHDL membutuhkan 375 *logic element* pada FPGA Altera EPF10K10LC84-4. Rancangan Mikroprosesor yang telah dibuat dapat dioperasikan dengan frekuensi detak maksimal 14,72 Mhz dan dapat beroperasi hingga kecepatan maksimal 3,68 MIPS.

6. DAFTAR PUSTAKA

Chu, P. P., 2006, *RTL Hardware Design Using VHDL*, John Wiley & Sons Inc., New Jersey.

He, Y. Z., 2002, *Building A RISC Microcontroller in an FPGA*, Faculty of Electrical Engineering, Universiti Teknologi Malaysia.

Hwang, E.O., 2004, *Microprocessor Design Principles and Practices With VHDL*, Brooks / Cole, California.

Hwang, E. O., 2005, *Digital Logic and Microprocessor Design with VHDL*, Brooks / Cole, California.

Maxfield, C.., 2004, *The Design Warrior's Guide to FPGAs*. Mentor Graphics Corporation and Xilinx, Inc, USA.

Rafiquzzaman, M., 2005, *Fundamentals of Digital Logic and Microcomputer Design*. John Wiley & Sons, Inc., Hoboken, New Jersey.

Tanenbaum, A. S., 2001, *Organisasi Komputer Terstruktur*, Salemba Teknika, Jakarta.

Turley, J., 2002, *Design Your Own Microprocessor*, Circuit Cellar Magazine, http://www.circuitcellar.com/

Zeidman, B., 2004, Introduction to CPLD and FPGA Design, Embedded System Conference, San Fransisco.