Prosiding SNRT (Seminar Nasional Riset Terapan) ISSN 2341-5662 (Cetak)
Politeknik Negeri Banjarmasin, 9 November 2017 ISSN 2341-5670 (Online)

RANCANG BANGUN MIKROPROCESSOR SEDERHANA 8 BIT

Zaiyan Ahyadi[1], Sarifudin[2]

Politeknik Negeri Banjarmasin[1]
z.ahyadi@poliban.ac.id[1]
sarif@ poliban.ac.id[2]

ABSTRACT

Internal circuit and design flow of microprocessor has not been well known by students. The objective of this research is to show clearly how to design a simple microprocessor that function work correctly. The method is designing a 16 bit simple microprocessor using VHDL language. Design flow using Top Down approach. The method is used because it can simulate and verified soon to know the unit works functionally without designing the detail. The basic functional units of the processor are ALU, accumulator and register set, control unit and busses. In order to processor can works properly, RAM, Rom and I/O ports are added to the design. The processor comes with 16 basic instruction that can be divided into three kinds of instruction as follows, ALU instruction, data transfer, and branch instruction. Memory space of RAM is divided into three spaces as follows, bit addressable, port, and space for data. The simulation and verification has been done by Xilinx ISE program. The experiments results show the processor works for arithmetic's instructions in loop. Beside that the "Quiz Button" bit logics instruction that used in digital control worked correctly for the processor. The microprocessor of this research can be used for teaching students and for the basic microprocessor for the further research to develop addition feature to the processor.
***Keyword** : Simple Processor, Top Down Approach, VHDL*

ABSTRAK

Rangkaian internal dan alur desainnya suatu mikroprosessor belum dikenal luas oleh para mahasiswa. Penelitian ini bertujuan untuk memberikan paparan yang jelas bagaimana membuat suatu mikroprosessor sederhana yang berfungsi dengan benar. Metode penelitian adalah dengan merancang suatu mikroprosessor sederhana 16 bit dengan menggunakan bahasa VHDL. Alur desain adalah dengan pendekatan secara Top Down. Metode ini dipilih karena simulasi dan verifikasi dapat dilakukan dengan cepat untuk mengetahui unit bekerja secara fungsional tanpa mengetahui detilnya. Prosessor hasil rancangan mempunyai unit fungsional dasar berupa ALU, akumulator dan register set, control unit dan bus. Serta dilengkapi dengan RAM dan ROM serta port Input Output agar mikroprosessor dapat berfungsi. Mikroprosessor ini dilengkapi dengan 16 perintah dasar yang terbagi menjadi tiga jenis perintah, yaitu perintah dasar ALU, perintah transfer data dan perintah bercabang. Ruang memori RAM terbagi tiga yaitu ruang memori yang dapat dialamati secara bit, ruang yang digunakan untuk PORT I/O dan ruang untuk data. Simulasi dan verifikasi dilakukan dengan bantuan program Xilinx ISE. Hasil ujicoba memperlihatkan bahwa mikroprosesor dapat menjalankan seluruh perintah artimetika untuk penjumlahan berulang. Di samping itu program "tombol kuis" juga dapat berjalan dengan baik menggunakan perintah logika bit.Mikroprosessor hasil penelitian ini dapat digunakan sebagai dasar untuk pembelajaran mahasiswa maupun sebagai dasar penelitian untuk mengembangkan fitur lain dari suatu mikroprosessor.
Kata Kunci : Processor Sederhana, Metode Top Down, VHDL

PENDAHULUAN

Mengetahui rangkaian dan cara kerja mikroprosessor secara detil dibutuhkan oleh para teknisi digital maupun mahasiswa dalam bidang elektronika dan komputer untuk dapat memahami secara kompresensif bagaimana kerja mesin digital. Namun rangkaian internal mikroprosessor tidak pernah dipulikasikan karena merupakan Intelektual Property (IP) dari pabrik pembuatnya. Untuk meningkatkan pemahaman pembelajaran maka diperlukan suatu rangkaian internal suatu mikroprosessor sederhana namun detil, mulai dari desain, program pendukung, dan cara membuat mikroprosessor.

Mikroprosessor yang banyak dipakai sekarang ini kebanyakan adalah mikroprosessor 64 bit atau minimal 32 bit. Namun pada era awal mikroprosessor adalah 8 bit. Ukuran bit ini mengacu pada register set terutama register dan ALU yang lebarnya 8 bit. Mikroprsessor sekarang ini sudah begitu canggih dan kompleks, sehingga sebagai bahan pembelajaran mahasiswa akan sangat sulit untuk difahami. Mikroprosessor sederhana 16 bit menjadi alternatif yang tepat untuk dibuat sebagai bahan pembelajaran.

Field Programmable Gate Array (FPGA) adalah chip yang berisi kumpulan ribuan bahkan jutaan gerbang logika yang dapat digunakan untuk membuat berbagai macam rangkaian digital berbasiskan gerbang logika. Mikroprosessor merupakan mesin digital yang berarti juga dapat dibuat berbasiskan FPGA. Dengan kata lain dengan memiliki FPGA seseorang seperti mempunyai pabrik pembuat chip digital sendiri.

Perancangan system digital secara Top Down lebih popular dengan adanya FPGA. Pendekatan ini memungkinkan perancangan dilakukan dengan cepat karena dapat dilakukan simulasi atas fungsional kerja desain yang dirancang tanpa seluruh rancangan selesai. Pendekatan secara Top Down mendahulukan desain secara behavioral atau cara kerja bukan pada detil rangkaian. Hal ini berlawanan dengan pendekatan secara Bottom Up yang mendahulukan perancangan unit-unit terkecil yang kemudian digabungkan menjadi sebuah desain yang lebih besar.

Rumusan masalah pada penelitian ini adalah bagaimana membuat desain internal suatu chip mikroprosessor sederhana dengan menggunakan bahasa VHDL. Penelitian ini bertujuan untuk membuat desain prototipe chip mikroprosessor 8 bit yang secara fungsional bekerja dengan benar.

Penelitian yang Berhubungan

Meskipun desain internal mikroprosessor merupakan Intelektual Proprty (IP) yang sangat dirahasiakan oleh pbrik pembuatnya, namun secara fungsional orang dapat menirunya. Peneliti E. Ayeh dan kawan-kawan [1] mencoba mendesain mikroprosessor sederhana 8 bit yang dipetakan pada FPGA Spartan. Project Dalton pada University of California seperti yang ditunjukkan pada laman http://www.cs.ucr.edu/~dalton/i8051/i8051syn/[2], memperlihatkan desain VHDL mikrokontroller model 8051 yang dapat disintesis. Pada project ini 8051 tersusun menjadi entity ALU, Control Unit, decoder, Debug, memori ROM dan RAM. Keluarga 8051 adalah mikrokontroller yang sangat sangat terkenal 4 dasawarsa

terakhir. Juan, H.-P. dan kawan-kawan pada penelitiannya [3] memberikan paparan untuk membuat mikrorosessor RISC 8 bit dengan pendekatan Top Down.

METODE PENELITIAN

Penelitian ini menggunakan metode pembuatan prototype mikroprosessor sederhana 16 bit dengan menggunakan bahasa VHDL (VSIC Hardware Desciption Language). Perancangan dilakukan dengan menggunakan metode Top Down dimana pada tahap awal perancangan dilakukan secara umum dengan mendiskripsikan komponen secara *behavioral* tanpa melihat secara detail rangkaian internal dari komponen tersebut. Setelah perancangan selesai akan dilanjutkan pada tahapan simulasi. Setelah simulasi memperlihatkan hasil yang benar, maka perancangan dapat ulangi lagi dengan mendefinisikan komponen penyusun menjadi lebih detail secara struktural. Kemudian dilanjutkan dengan simulasi lagi, demikian seterusnya sampai pada tahapan seberapa detail perancangan yang ingin dicapai.

Processor yang dirancang dilengkapi dengan RAM dan ROM serta instruction set untuk dapat berfungsi dan disimulasikan. Penelitian ini menggunakan desain dasar mikroprosessor yang didapat peneliti pada situs web[4]. Dari desain dasar ini dilakukan beberapa modifikasi sehingga ddidapat fitur tambahan yang menjadikan hasil rancangan mempunyai kelebihan dibandingkan dengan desain dasar.

Adapun tahapan dalam perancangan mikroprosessor ini adalah sebagai berikut:

1. Menentukan set instruksi yang akan digunakan.
2. Menentukan unit-unit yang diperlukan agar seluruh set instruksi dapat bekerja dengan baik.
3. Menentukan tahapan (*stage*) dalam menjalankan instruksi
4. Menuliskan program VHDL secara *behavioral* untuk membuat unit-unit tersebut dalam satu kesatuan entiti.
5. Menjalankan simulasi terhadap modul pada langkah 3.
6. Memisahkan unit yang pada langkah 3 menjadi modul terpisah satu persatu.
7. Membuat modul *top entity* untuk modul-modul pada langkah 5.
8. Menjalankan simulasi untuk top entity.
9. Menjalankan *sintesis* untuk verifikasi modul.
10. Kembali ke langkah 6 sampai semua unit menjadi modul tersendiri.

Set Instruksi

Set instruksi merupakan kumpulan instruksi yang digunakan untuk menjalankan prosesor. Pada penelitian ini, instruksi dirancang selebar 16 bit yang terdiri dari 8 bit opcode dan 8 bit operand. Untuk instruksi yang tidak mempunyai operand, tetap saja harus dituliskan operand-nya dengan sembarang nilai. Keseragaman instruksi ini mempermudah proses decoding dan mempermudah pengubahan nilai register PC. Setiap instruksi akan menyebabkan perubahan PC sebesar 2 byte, namu karena prosessor ini adalah prosessor 8 bit, maka transfer isntruksi dari ROM ke Instruction Register dilakukan dengan dua tahap per 8 bit.

Tabel 1 memperlihatkan daftar set instruksi yang dapat dijalankan oleh mikroprosessor pada penelitian ini. Kolom pertama sebelah kiri merupakan

instruksi dalam heksadesimal. Kolom kedua memuat register mana yang bekerja untuk perintah tersebut. Dan kolom ketiga merupakan keterangan bagaimana cara kerja perintah tersebut. Instruksi terdiri dari 16 bit yang dinyatakan dengan 4 digit bilangan heksadesimal. Simbol score (-) menyatakan bahwa digit tersebut tidak berpengaruh sehingga dapat dituliskan digit apa saja dalam bilangan heksa decimal, untuk praktisnya lebih baik dituliskan 0. Sedangkan digit dengan huruf x atau y menyatakan dalam perintah sesungguhnya variabel tersebut bisa diganti bilangan heksadesimal yang menunjukkan alamat x atau y.

Tabel 1. Set instruksi

Instruksi (Hexadesimal)	Unit yang bekerja	Keterangan
10xx	Acc, RAM	Transfer data dari Accumulator ke memori xx
11xx	Acc, RAM	Transfer data dari memori alamat xx ke accumulator
20xx	Acc, RAM, ALU	Menjumlahkan data di akumulator dengan data di memori xx
21xx	Acc, RAM, ALU	Mengurangi Accumulator dengan data di memori alamat xx
22xx	Acc, RAM, ALU	Operasi AND bitwise data di akumulator dengan data di memori xx
23xx	Acc, RAM, ALU	Operasi OR bitwise data di akumulator dengan data di memori xx
30--	Acc, ALU	Mengubah data di akumulator menjadi negatifnya
31--		Halt
4-xx	PC	Mengubah data di PC menjadi xx
5-xx	Acc, PC	Loncat bersyarat Jika Acc = 0, ubah nilai PC menjadi xxx
60xx	Acc, PC, RAM, Indirect Reg	Transfer data dengan pengalamatan tak langsung Transfer data dari alamat yang ditunjukkan oleh data pada memori xx ke accumulator
61xx	Acc, PC, RAM, Indirect Reg	Transfer data dengan pengalamatan tak langsung Transfer data dari akumulator ke memori dengan alamat yang ditunjukkan oleh data pada alamat xx
8-xx	Acc	Mengubah nilai Akumulator menjadi xx
9-xx	Acc	Menjumlahkan data xx dengan akumulator

Unit fungsional yang diperlukan

Suatu mikroprosessor terdiri dari tiga unit fungsional yaitu Arithmatic Logic Unit (ALU), Register Set dan Control Unit. Ketiga unit tersebut dihubungkan oleh bus. Bus yang terdapat pada suatu mikroprosessor yaitu bus data, bus alamat dan bus control.

Berdasarkan instruksi yang telah direncanakan, akan didapatkan unit fungsional yang diperlukan untuk menjalankan seluruh isntruksi tersebut. Unit funsional tersebut adalah :

- Program Counter (PC)
  Register yang berfungsi untuk menyimpan alamat ROM tempat instruksi berikutnya yang akan dijalankan
- Instruction Register (IR)

Register yang berfungsi untuk menyimpan instruksi yang sedang dijalankan.
- Accumulator (Acc)
  Register utama yang selalu digunakan dalam setiap proses yang melibatkan ALU
- Arithmetic Logic Unit (ALU)
  Meruapakan unit yang berfungsi untuk melakukan operasai aritmetika dan logika. Dari daftar instruksi yang telah dirancang operasi tersebut adalah penjumlahan, pengurangan, operai bitwise OR, AND dan menjadikan bilangan negatip
- Control Unit (CU)
  Unit ini berfungsi untuk mengirimkan sinyal control ke unit lain pada saat yang tepat, sehingga tidak terjadi tabrakan aliran data. Unit ini yang mengatur *timing* kapan sinyal control untuk unit yang mana. Timing dapat dinyatakan dalam bentuk *stage* atau tahapan
- Indirect Address Register (IAR)
  Merupakan register yang menyimpan alamat tak langsung dalam pengaksesan RAM. Register ini berfungsi seperti fungsi pointer dalam pemrograman.
- Read Only memory (ROM)
  Memori untuk tempat penyimpanan program
- Random Access Memory (RAM)
  Memori untuk menyimpan data.

Menentukan Tahapan Instruksi

Mikroprosessor merupakan mesin digital singkron, yang berarti agar bekerja dengan baik memerlukan sinyal clock. Suatu instruksi mikroprosesor tidak selesai dalam satu tahap atau satu sinyal clock saja. Suatu mikroprosessor dasar memerlukan lima tahapan dalam menjalankan suatu instruksi[5], yaitu

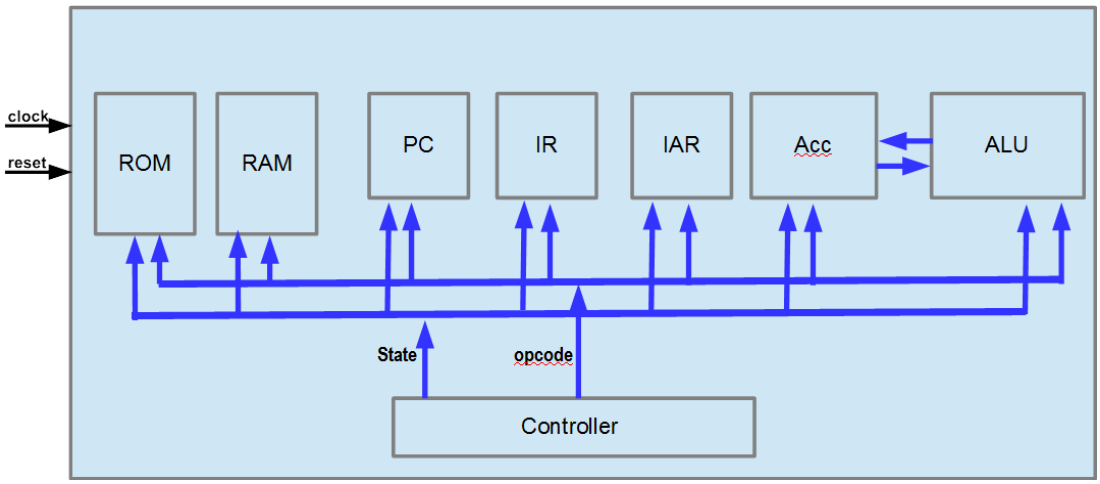
1. Instruction Fetch, yaitu mentransfer instruksi dari ROM ke IR
2. Instruction Decoding, yaitu menerjemahkan perintah (opcode) untuk kemudian dipilih proses yang tepat.
3. Operand Fetch, yaitu mentransfer data yang akan dioperasikan
4. Execution, yaitu menjalankan perintah opcode. Pada tahapan ini melibatkan ALU.
5. Write Back, mentransfer data hasil perhitunga ALU kembali ke memori.

Berdasarkan seluruh daftar instruksi dan keterangan cara kerja instruksi tersebut serta memperhatikan seluruh unit penyusunnya maka untuk mikroprosessor pada peneltian ini, suatu instruksi dijalankan dengan maksimal 5 tahap (stage). Tahap tersebut adalah :

1. Instruction fetch pertama.
   Pada tahapan ini terjadi transfer data dari ROM ke byte tinggi IR
2. Instruction fetch yang kedua.
   Pada tahapan ini terjadi transfer data dari ROM ke byte rendah IR
3. Instruction decoding
   Pada tahapan ini opcode (byte tinggi IR) dibaca untuk menentukan operasi yang harus dilakukan oleh prosessor

4. Operand Fetch
   Pada tahapan ini operan diambil dari memori ke dalam prosessor
5. Execution
   Pada tahapan ini operasi dalam ALU dilaksanakan. Untuk perintah transfer data tanpa melibatkan ALU tahapan ini tidak dilaksanakan.

Membuat Modul Behavioral



Gambar 3. Modul behavioral sistem mikroprosesor

Jika semua unit telah ditentukan dan didapatkan tahapan dalam menajalankan instruksi selanjutnya adalah membuat modul behavioral untuk keseluruhan system. Gambar 1 memperlihatkan unit penyusun berupa blok dalam modul behavioral system mikroprosessor yang dirancang. Pada modul behavioral keseluruhan system belum diperlukan nama seluruh sinyal untuk setiap unit. Tujuan utama modul behavioral adalah simulasi dapat berjalan dengan benar seusai rancangan. Verifikasi dengan simulasi dapat dilihat perubahan bentuk gelombang sinyal yang dideklarasikan pada modul. Modul behavioral juga tidak memetakan unit secara structural, sehingga unit-unit diimplementasikan sebagai process=proses yang berjalan secara *concurrent*. Setiap process menjadikan sinyal clock sebagai *sensitivity list* utama. Pada modul ini hubungan antara unit dapat dilakukan tanpa menggunakan sinyal bus alamat dan bus data.

Berikut struktur dasar modul VHDL yang ditunjukkan oleh kode 1, untuk keseluruhan sistem mikroprosessor yang dirancang. Tanda titik sebanya 3 buah menyatakan bahwa di lokasi tersebut terdapat beberapa baris program yang dihilangkan. Untuk tahap awal tidak perlu dituliskan seluruh proses unit secara langsung, tapi pelan-pelan sedikit demi sedikit unttuk dijalankan simulasi secara bertahap untuk menghindari kebanyakan *error* kode. Harus diperhatikan bahwa pada *process* suatu unit hanya boleh mengubah nilai sinyal yang dimiliki oleh unit itu saja, jangan sampai ikut mengubah sinyal yang dimili unit lain. Misalnya process untuk unit accumulator hanya boleh mengubah nilai register accumulator saja. Atau process untuk RAM hanya boleh mengubah nilai RAM saja.

Kode 1. Bentuk dasar program behavioral

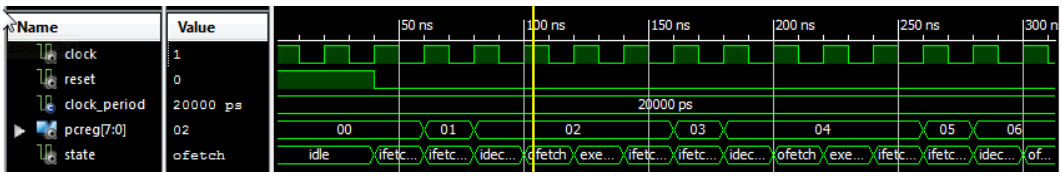
```
entity proc8 is
port (
      clock, reset: in STD_LOGIC
   );
end proc8;

architecture Behavioral of proc8 is
  --disini deklarasi semua sinyal yang diperlukan
signal state: state_type;
signal accReg: STD_LOGIC_VECTOR(7 downto 0);
signal irReg : STD_LOGIC_VECTOR(15 downto 0);
. . .

begin
   --disini dituliskan aliran sinyal yang concurrent
  signal state: state_type;
  signal accReg: STD_LOGIC_VECTOR(7 downto 0);

  --disini proses untuk unit ROM dan RAM
  process(reset) begin
  . . .

  end process;

  --disini proses untuk unit Instruction Register
  process(clk_ctrl) begin
  . . .
  end process;

  --disini proses untuk unit Program Counter
  process(clk_pc) begin
  . . .
  end process;

--disini proses untuk unit Accumulator
--disini proses untuk unit ALU
--disini proses untuk unit Control Unit
--disini proses untuk unit Indirect Address Register

end Behavioral;
```

Menjalankan simulasi

Simulasi untuk mdoul behavioral dilaksanakan secara bertahap, sedikit demi sedikit ditambahkan sinyal dan unit yang diperlukan. Pada system prosessor, ketika system baru ssaja dihidupkan PC langsung menunjuk alamat awal ROM (alamat 0) sehingga instruksi pertama yang dijalankan adalah isntruksi yang tersimpan dil lamat ini. Untuk selanjutnya alur program akan berjalan sesuai

dengan isntruksi yang telah dituliskan di dalam ROM. Untuk maka perlu disusun terlbeih dahulu sebuah program yang terdiri dari urutan instruksi yang telah dirancang, dan diketahui secara pengaruh pada sinyal-sinyal yang dideklarasikan pada modull VHDL.

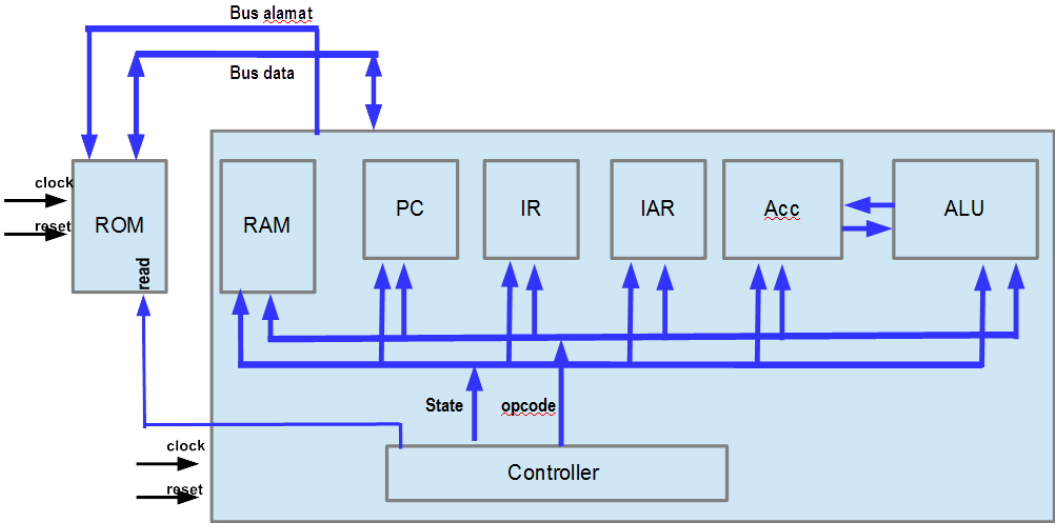
Gambar 2 memperlihatkan hasil simulasi untuk program sederhana. Dapat dilihat pada sinyal state, terlihat urutan stage dengan benar mulai dari *ifetch1, ifetch2, idecode, ofetch, dan execute* secara berluang-ulang. Ini menyatakan sinyal *timing* dari Unit Control dalam bentuk tahapan telah berjalan dengan benar. Demikian juga dengan PC, dapat dilihat bahwa PC bertambah setiap tahan instruction fetch, yang terdiri dari dua kali instruction fetch dalam satu siklus.



Gambar 4. Simulasi untuk unit Program Counter

Memisahkan Unit Menjadi Modul Terpisah

Pada tahapan ini unit-unit dalam satu buah modul VHDL secara *behavior* dipisahkan menjadi modul atau file VHDL yang berdiri sendiri. Untuk tahap pertama pilihan yang tepat adalah memisahkan memori RAM menjadi modul yang terpisah. menjadi dua modul yang berdiri sendiri (dihitung dengan modul asal). Untuk membuat kedua modul ini bekerja sama, perlu dibuat modul VHDL ketiga yang dinamakan dengan top entity. Entity ini melingkupi entity ROM dan entity prosessor.



Gambar 5. blok diagram unit ROM telah menjadi unit sendiri

Pada saat memisahkan unit inilah akan diperlukan sinyal penghubung antara satu modul dengan modul lainnya yaitu sinyal bus alamat dan bus data. Bus control tidak lagi berpa sinyal *state* melainkan menjadi sinyal control individual

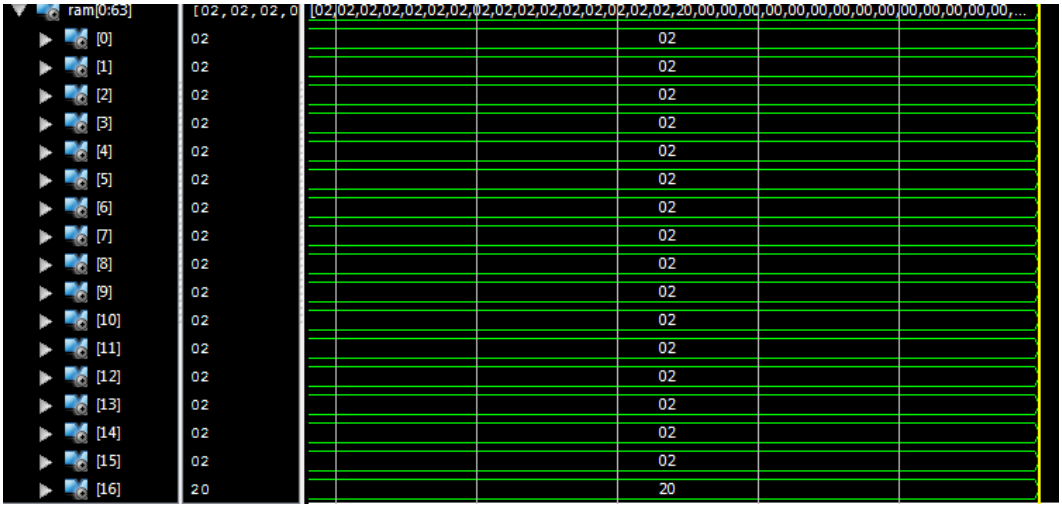
untuk unit tersebut. Contoh untuk ROM adalah sinyal read. Gambar 5 memperlihatkan satu unit (ROM) telah terpisah menjadi modul tersendiri.

HASIL DAN PEMBAHASAN

Untuk mengetahui hasil rancangan prosessor telah bekerja dengan baik harus dilakukan simulasi. Simulasi dilakukan terlebeih dahulu untuk setiap unit fungsional penyusun mikroprosessor. Pada penelitian ini program simulasi yang digunakan adalah program simulasi yan terinteggrasi dalam program Xilinx ISE versi 14. Untuk melakukan simulasi perlu dibuat entiti testbench untuk unit yang akan ditest.

Tabel 2. Urutan Perintah Simulasi System Mikroprosesor

Alamat ROM	Instruksi (Heksadesimal)	Keterangan
00	8000	Ubah nilai akumulator menjadi “00”.
02	1010	Simpan nilai akumulator ke alamat 10. Inisialisasi nilai awal penjumlahan = 0
04	1011	Simpan nilai akumulator ke alamat 11. Inisialisasi pointer menunjukan alamat 0
06	8010 (awal loop)	Ubah nilai akumulator menjadi “10”
08	3000	Negatipkan nilai akumulator
0a	2011	Tambahkan dengan data pada alamat 11
0c	500e	Jika hasil penjumlahan adalah 0 maka loncat ke alamat 20
0e	6011	Sum = sum + *pointer
10	2010	Tambahkan akumulator dengan data pada alamat 10
12	1010	Simpan nilai akumulator ke alamat 10
14	8001	Ubah nilai akumulator menjadi 1
16	2011	Tambahkan dengan data pada alamat 11
18	1011	Simpan hasilnya di alamat 11
1a	4006	Loncat ke alamat 06
1c	3300	halt



Gambar 6. Simulasi dengan Memori

Untuk simulasi system mikroprosessor lengkap dengan akses memori RAM dan dengan program yang tersimpan di dalam ROM dilakukan dengan membuat ROM beserta instruksi yang sudah tersimpan di dalamnya. Adapun instruksi tersebut merupakan instruksi berurutan sederhana yang menghitun penjumlahan bilangan yang tersimpand dalam RAM pada alamat 00h sampai dengan alamat 0eh yang hasilnya disimpan pada alamat 10h. Isi RAM dari alamt 00 sampai dengan 0e adalah 02h yang berarti nilai RAM pada alamat 10h adalah 32d atau 20h. Gambar simulasi sinyalnya dapat dilihat pada gambar 6. Dari hasil simulasi telihat bahwa isi RAM adalah benar 20h.

KESIMPULAN

Penelitian ini berhasil merancang mikroprosessor 8 bit, meskipun belum diimplementasikan pada FPGA namun telah dapat disimulasikan dengan hasil yang benar. Alur desain dipaparkan pada artikel ini dengan harapan dapat membantu para mahasiswa maupun peneliti lain yang ingin memulai penelitian dalam bidang desain mikroprosessor.

DAFTAR PUSTAKA

Ayeh, E.; Agbedanu, K.; Morita, Y.; Adamo, O.; Guturu, P., *"FPGA Implementation of an 8-bit Simple Processor",* IEEE 2008 IEEE Region 5 Conference - Kansas City, MO, USA

http://www.cs.ucr.edu/~dalton/i8051/i8051syn/

Juan, H.-P.; Holmes, N.D.; Bakshi, S.; Gajski, D.D.,*"Top-Down Modeling of RISC Processors in VHDL"* Proceedings of EURO-DAC 93 and EURO-VHDL 93- European Design Automation Conference

http://www.cse.wustl.edu/~dzar/class/260/hw/cpu.html

Mostafa Abd. El-Barr, Hesham El Rewini, *"Fundamental organization and architecture computer"*, Wiley InterScience, ISBN 0-471-46741-3,2005