Pemodelan ALU pada Mikroprosesor MIPS menggunakan *Redstone* di *Minecraft*

Mazaya Zata Diny[1], Hilal Hudan Nuha[2], Andrian Rakhmatsyah [3]

[1,2,3]Fakultas Informatika, Universitas Telkom, Bandung
[1]mazayaazd@students.telkomuniversity.ac.id, [2]hilalnuha@telkomuniversity.ac.id,
[3]kangandrian@telkomuniversity.ac.id

Abstrak

Minecraft bisa digunakan sebagai cara yang menarik untuk menjelaskan dan memvisualisasikan aliran listrik dalam rangkaian digital. Di Indonesia, *Minecraft* masih belum banyak digunakan sebagai *game* yang memvisualisasikan elektronika digital dengan gerbang logika digital dimana interkoneksi masing-masing gerbang tersebut dapat dikonfigurasi antara satu dengan yang lainnya. Sementara itu, sumber daya yang terdapat pada *Minecraft* menyimpan potensi yang bisa memadukan antara kreativitas, *game* dan elektronika digital. Penelitian ini membahas tentang penggunakan Minecraft untuk pemodelan *Arithmatic Logical Unit* (ALU) pada Mikroprosesor. Pemilihan arsitektur ALU yang merupakan salah satu komponen utama di setiap arsitektur mikroprosesor yang akan dimodelkan adalah karena pertimbangan peranan ALU pada mikroprosesor yang penting untuk membentuk unit pengolahan data di komputer. ALU yang dibangun yaitu arsitektur logika (OR, NOR, AND, NAND, XOR, XNOR) dan arsitektur aritmatika (ADD, SUB). Hasil pemodelan menunjukkan hasil bahwa semua operasi standar ALU berhasil berjalan dengan baik.

Kata kunci : *Minecraft, Redstone, Arithmatic Logical Unit.*

Abstract

Minecraft can be used as an interesting way to explain and visualize electricity in a digital circuit. In Indonesia, Minecraft is still not widely used as a game that visualizes digital electronics with digital logic gates where interconnection of each gate can be configured with the other gate. Meanwhile, the Minecraft's resources holds potential that can combine creativity, game and digital electronics. This study discusses the use of Minecraft for Arithmatic Logical Unit (ALU) modeling on Microprocessors Choosing ALU architecture which is one of the main components in each microprocessor architecture that will be modeled. Due to the consideration of ALU's role as the important microprocessors to form data processing units on the computer. ALU is built on logic architecture (OR, NOR, AND, NAND, XOR, XNOR) and arithmetic architecture (ADD, SUB). The modeling results show that all standard ALU operations have been successful.

Keyword : *Minecraft, Redstone, Arithmatic Logical Unit.*

1. Pendahuluan

Latar Belakang

Game saat ini banyak digunakan sebagai media pembelajaran, informasi yang disampaikan akan lebih menarik dan mudah dipahami [1]. Salah satu cara menciptakan pembelajaran yang menyenangkan yaitu dengan mengemas *game* menjadi media pembelajaran kreatif. Pengembangan *game* yang signifikan mampu menciptakan infrastruktur virtual [2] untuk mensimulasikan suatu informasi yang rumit, seperti mensimulasikan mikroprosesor dengan *game* populer komputer, *Minecraft* [3].

Minecraft yaitu *game* simulasi berbasis blok, *multi-platform sandbox* [2]. Di tahun 2019 ini *Minecraft* telah terjual sebanyak 156 juta *copy* dan kini *Minecraft* dimainkan oleh 75 juta per bulannya. *Minecraft* menyediakan *physic engine* yang cukup canggih untuk analisis dan penelitian jaringan [4] dengan mengatur, menjelajahi dan membangun struktur tiga dimensi [5] diarea yang sangat luas. Pengguna dapat membuat dunianya sendiri menggunakan sumber daya yang terdapat pada *game* seperti blok, barang, dan alat [3].

Menggunakan *Minecraft* untuk pendidikan adalah konsep yang dapat mengajarkan proses menghitung dan logika [1]. Melalui sumber daya yang tersedia dapat mengimplementasikan *Redstone* [4] sebagai kunci utama untuk membuat suatu konsep berjalan. *Redstone* dikategorikan dalam komponen daya [2], dimana didalamnya terdapat sumber listrik, sehingga menciptakan suatu arus atau sinyal.

Di Indonesia, *Minecraft* masih belum banyak digunakan sebagai *game* yang memvisualisasikan ataupun mensimulasikan elektronika digital dengan gerbang-gerbang digital dimana interkoneksi masing-masing gerbang tersebut dapat dikonfigurasi antara satu dengan yang lainnya. Sementara itu, sumber daya yang terdapat pada *Minecraft* menjadi hal yang unik sekaligus menyenangkan dalam upaya memperpadukan antara kreatifitas, *game* dan elektronika digital.

Pemilihan *Minecraft* sebagai wadah penelitian ini bukan hanya karena kemampuan yang efisien dalam merancang rangkaian digital (terdiri dari gerbang-gerbang logika) sesuai keinginan dan kebutuhan, tetapi karena *Minecraft* adalah aplikasi *java* yang didokumentasikan dengan baik [4]. Pemrosesan arus atau sinyal digital dapat dilakukan dengan merancang desain arsitektur dengan tersusun padat, agar memungkinkan instruksi berjalan dengan baik, sehingga dapat melihat kapan terjadinya pengantaran arus ke berbagai rangkaian, dan kapan pula perubahan menyebabkan kehilangan kinerja arus [4]. Sehubungan dengan pemrograman, modifikasi keterampilan *Minecraft* memungkinkan adanya kemungkinan *visual coding* yang menarik, dengan metode seperti *coding blocks* di *CodeBlocks* [6]. Dan pemilihan arsitektur *Arithmatic Logical Unit* (ALU) pada *Microprosesor without Interlocked Pipeline Stages* (MIPS) yaitu salah satu komponen utama dalam mikroprosesor MIPS [7] sebagai arsitektur yang akan divisualisasikan, sebab mempertimbangkan peranan ALU pada mikroprosesor untuk membentuk fungsi-fungsi pengolahan data komputer.

Implementasi berbasis *Redstone* ini dibangun untuk melatih dan mengembangkan kreatifitas atau penalaran, kemampuan berpikir logis, mengembangkan logika matematika dengan bentuk, pengukuran geometris, analisis pola, atau pemetaan dalam literatur pengembangan keterampilan penalaran [6]. Dengan adanya implementasi berbasis *Redstone* ini dapat merancang, memahami dan mengungkapkan data dalam suatu grafik serta kepekaan terhadap keseimbangan, relasi, warna, garis, bentuk, dan ruang.

Topik dan Batasannya

Berdasarkan latar belakang yang sudah dijelaskan sebelumnya, maka masalah yang akan dijawab dalam pembangunan penelitian ini, anatara lain:

1. Bagaimana membangun sebuah rangkaian digital atau logika berbasis *Redstone* agar dapat mengolah dan menjalankan instruksi dari *Arithmatic Logical Unit* (ALU) di *Minecraft* ?
2. Berapa banyak sumber daya yang dibutuhkan untuk implementasi ALU di *Minecraft* ?

Sementara itu, batasan masalah yang digunakan dalam penelitian ini agar cakupan penelitian yang diambil sesuai dengan kebutuhan dan kemampuan penulis. Batasan dalam penelitian ini dibagi menjadi dua, yaitu batasan dalam mengolah dan menjalankan *Arithmatic Logical Unit* (ALU) dan batasan dalam pembangunan arsitektur. Batasan tersebut, antara lain:

- Batasan dalam mengolah dan menjalankan ALU:
  1. Masukan berupa biner
  2. Keluaran berupa biner
- Batasan dalam pembangunan arsitektur:
  1. Terdapat *Operation Code* (opcode) dan *decoder*.
  2. Arsitektur yang digunakan 32 bit.
  3. Arsitektur yang dimodelkan yaitu *single cycle*.
  4. *Design* rangkaian sesuai kreatifitas.
  5. Penggunaan *Redstone* sesuai kebutuhan.
  6. Operasi logika minimal terdiri dari AND, OR, XOR, NOT
  7. Operasi aritmatika minimal terdiri dari ADDER

Tujuan

Tujuan yang akan dicapai dalam penelitian ini, antara lain:

1. Untuk mengetahui apakah rangkaian digital menggunakan *Redstone* dapat berperilaku benar secara fisik dan dapat menunjukkan terjadinya pengantaran arus ke berbagai rangkaian, dan kapan pula perubahan menyebabkan kehilangan kinerja arus, dengan menggunakan *Operation Code* untuk menentukan data apa yang akan diproses oleh komponen tersebut.
2. Untuk mengetahui gerbang-gerbang logika pada rangkaian digital yang dibangun sudah relevan atau belum untuk mengatur instruksi dalam komunikasi diantara rangkaian lainnya, dengan melihat instruksi masukan yang diberikan dan keluaran yang dihasilkan sudah sesuai atau belum.

Organisasi Tulisan

Penulisan tugas akhir ini tersusun dalam beberapa bagian. Bagian pertama adalah pendahuluan, dimana pendahuluan menjelaskan mengenai latar belakang, rumusan masalah dan tujuan dari topik yang diambil dalam tugas akhir ini. Bagian dua adalah studi terkait, berisi penelitian sebelumnya terkait dengan topik yang diambil sebagai literatur dan acuan dalam membuat tugas akhir. Bagian ketiga merupakan sistem yang dibangun, dimana berisikan penjelasan dan gambaran alur kerja sistem, termasuk alur kerja *Redstone* terhadap rangkaian digital yang dibangun. Bagian keempat yaitu evaluasi, dimana bagian ini menjelaskan mengenai pengujian yang dilakukan, analisis terhadap hasil yang dicapai dan menjelaskan scenario yang dibangun untuk melakukan pengujian. Bagian kelima adalah kesimpulan dan saran yang memberikan kesimpulan mengenai penelitian yang sudah dilakukan dan memberikan saran untuk penelitian selanjutnya.

2. Studi Terkait

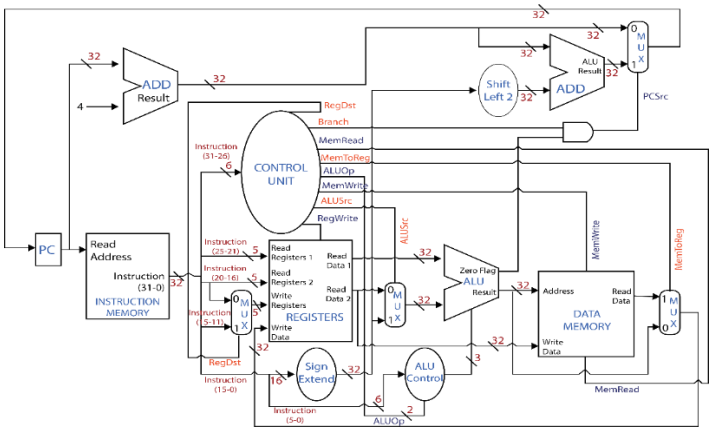
Gamification, dalam hal pendidikan, dapat bekerja jika konteksnya benar [1]. *MinecraftEDU*[1] sudah menunjukkan bahwa *Minecraft* merupakan bangunan yang bagus untuk sebuah permainan pendidikan [1], dikarenakan mekanika komputasi bawaan yang mengintegrasikan elemen logika, desain dan *scripting*. Menggunakan Minecraft sebagai simulator jaringan interaktif dapat dengan mudah dipahami. Untuk membangun sirkuit listrik yang paling dasar pada *Minecraft* memiliki tiga komponen penting yaitu [1]:

1. Generator (sumber),
2. Arus (koneksi),
3. Dan konsumen (beban).

Konsep utama dalam permainan ini adalah blok, berupa kotak terdiri dari 16 x 16 pixel. Sehingga seluruh area didalam *Minecraft* adalah matriks 3D yang diisi dengan berbagai jenis blok [1]. Permainan ini tidak memiliki alur pemain yang dibatasi, fokus utamanya adalah kreatifitas dalam membangun [5]. *Minecraft* didalamnya terdapat juga komponen daya yang disebut dengan *Redstone*. *Redstone* ini berupa energi yang dapat menyalurkan arus atau sinyal sehingga mampu membangun jaringan listrik didalamnya [3]. Terdapat 5 komponen dalam pengelompokan *Redstone* yaitu komponen *power*, *transmission*, *mechanism*, *cellular* dan *etc* [8]. Dimana kategori perkomponen ini dapat memudahkan dalam membangun implementasi visual *Arithmatic Logical Unit* (ALU) pada *Microprosesor without Interlocked Pipeline Stages* (MIPS) [7]. ALU adalah modul utama dalam *Central Processing Unit* (CPU) yang melakukan aritmatika seperti penjumlahan, pengurangan dari bilangan biner dan operasi logika seperti *INVERT, AND, OR, XOR* dan fungsi *Boolean* lainnya [9]. Desain utama yang diperlukan dalam pemodelan ALU yaitu [10]:

1. Full Adder
2. Blok Logika
3. Multiplexer

ALU yaitu sirkuit rangkaian kombinasional yang melakukan serangkaian operasi aritmatika dan logika dasar [10], dapat dikatakan sebagai otak setiap CPU seperti pada mikroprosesor, pemroresan sinyal digital dan lainnya [10]. Bagian penting dari unit aritmatika yaitu *Full Adder* [10]. Rangkaian *Binary Adder* yaitu salah satu unit dasar pada ALU yang digunakan untuk melakukan operasi aritmatika [9].



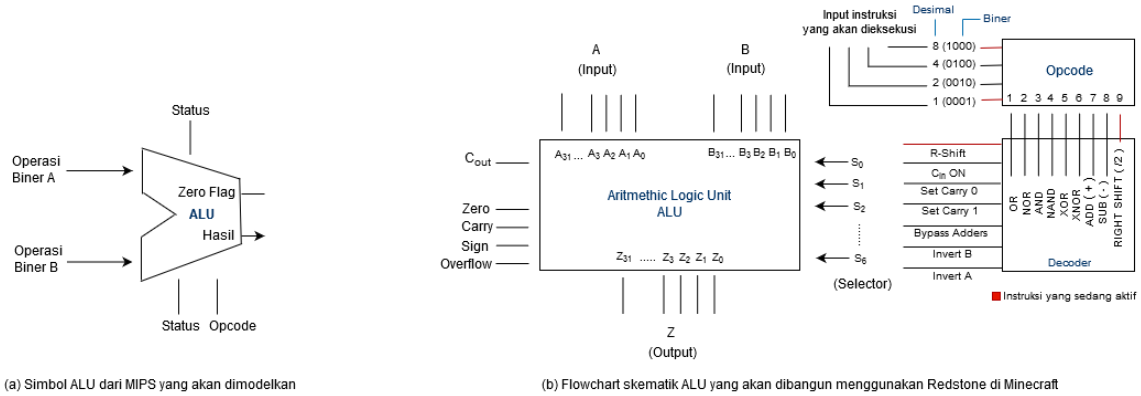
Gambar 1 *Single Cycle MIPS Prosessor*.

[1] https://education.minecraft.net/

MIPS yaitu arsitektur kumpulan instruksi *assembly programming* yang cocok untuk mengenal komputer dan mengetahui cara mikroprosesor bekerja [7]. Dalam pemrograman *assembly language* dari prosesor MIPS ini menekankan topik yang diperlukan untuk bit, pola bit, operasi pada pola bit, dan bagaimana pola bit mewakili instruksi dan data [11]. Mempertimbangkan kesederhanannya, MIPS cocok untuk sebagai lingkungan pengembangan interaktif yang ditujukan untuk penggunaan pendidikan [7]. Menggunakan *assembly language* sebuah instruksi penambahan dalam bahasa mesin diartikan dengan kode '10110011' yang dalam *assembly language* dapat dibuat dalam instruksi *'mnemonic'* yaitu *'ADD'*, sehingga mudah dalam pembangunan sketsa ALU pada arsitektur MIPS untuk implementasi di *Minecraft* [3].

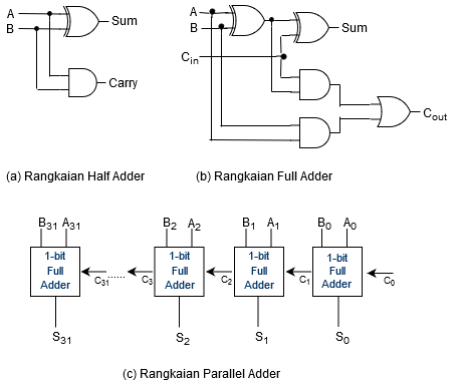
3. Sistem yang Dibangun

Gambaran umum rangkaian yang akan dibangun pada penelitian antara lain, sistem ALU yang dapat menerjemahkan instruksi-instruksi yang diberikan. Sistem ALU ini bekerja dan dimulai dengan melakukan penerjemahan masukkan *biner* dan instruksi yang akan dilakukan.. Alur proses sistem dapat dilihat pada gambar 2.



Gambar 2 Simbol dan *Flowchart* Skematik ALU yang akan dimodelkan menggunakan *Redstone* di *Minecraft*.

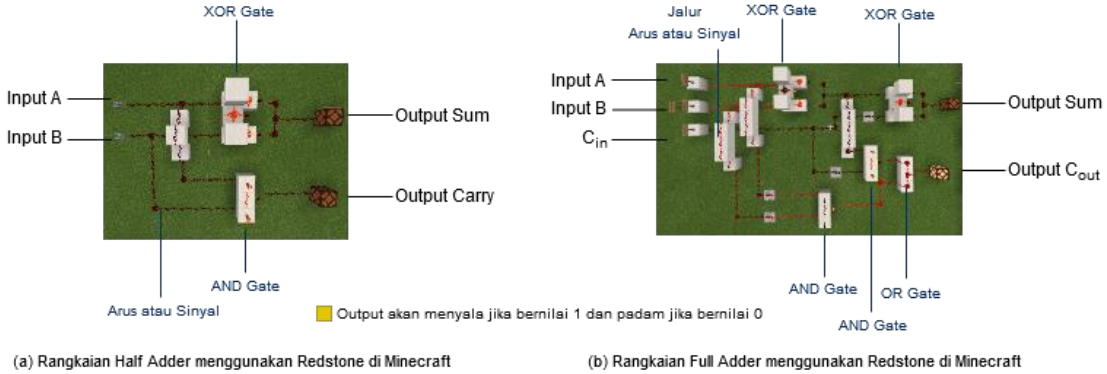
Proses pembuatan Pemodelan ALU menggunakan *Redstone* terlihat pada (gambar 2a) sebagai bagian yang akan dimodelkan dan pada (gambar 2b) proses bagaimana ALU akan dibangun menggunakan *Redstone* di *Minecraft*. Pemodelan akan menggunakan *Operation Code* sebagai bagian dari instruksi yang menentukan operasi yang akan dilakukan dan *Decoder* sebagai rangkaian logika yang ditugaskan untuk menerima *input-input* biner dan mengaktifkan salah satu *output*-nya sesuai dengan urutan biner tersebut. Pertama (gambar 2b) akan memasukkan instruksi dengan masukkan biner, lalu masukkan akan diteruskan ke *decoder* dan *decoder* akan menerima masukkan operasi apa yang dipilih dan mengaktifkannya. Setelah operasi berhasil diaktifkan dan ALU menerima instruksi tersebut maka selanjutnya memasukkan *input* A dan *input* B (gambar 2b) agar ALU mengeksekusi aritmatika atau logika sehingga menghasilkan *output*.



Gambar 3 Rangkaian *Adder* yang akan menjadi dasar dalam dibangunnya pemodelan ALU menggunakan *Redstone* di *Minecraft.*

ALU dalam mengoperasikan tugasnya melibatkan suatu sikuit khusus yang disebut dengan *Adder*. *Adder* akan digunakan untuk operasi aritmatika dan sebagai dasar dalam membangun ALU menggunakan *Redstone* di *Minecraft*. *Adder* juga disebut rangkaian kombinasional yang terbagi menjadi 3 jenis *Adder*, yaitu rangkaian *Adder*

dengan menjumlahkan 2-bit disebut *Half Adder* (gambar 3a), rangkaian Adder dengan menjumlahkan 3-bit disebut *Full Adder* (gambar 3b) dan rangkaian Adder dengan menjumlahkan banyak bit disebut *Parallel Adder* (gambar 3c). Rangkaian logika tersebut dibangun secara bertahap, dengan membangun *Half Adder* terlebih dahulu dan memastikan rangkain berjalan telah sesuai atau belum, lalu membangun *Half Adder* selanjutnya agar menjadi rangkaian *Full Adder* dan setelah rangkaian telah berjalan dengan baik, selanjutnya membangun *Parallel Adder* sebanyak 32-bit.



Gambar 4 Pemodelan Rangkaian *Adder* yang dibangun menggunakan *Redstone* di *Minecraft.*

Pada (gambar 4a dan 4b), melihatkan bagaimana *coding block* dilakukan yaitu dengan cara menyatukan keseimbangan, relasi, logika, ruang dan pengukuran geometris. Pada proses ini setiap rangkain logika akan diterjemahkan kedalam desain blok menggunakan *Redstone* dengan membuat kolaborasi dari beberapa gerbang sehingga menjadi rangkaian yang padat dan meminimalisir penggunaan sumber daya. Dengan menggunakan kombinasi bermacam-macam gerbang logika ini lah cara untuk membangun sebuah ALU didalam *Minecraft*. Untuk membangun ALU diperlukan gerbang logika *XOR*, *OR* dan *AND*, ketiga gerbang tersebut akan disatukan sehingga membentuk rangkaian *Adder*, lalu dari rangkaian *Adder* ini akan menjadi dasar dari ALU untuk menggabungkan fungsi-fungsi lainnya sehingga dapat menciptakan instruksi lainnya, seperti *NOR*, *XNOR* dan *NAND*.

Table 1 *Control Word* untuk Operasi ALU di *Operation Code*

Input	Operasi
0001	OR Gate
0010	NOR Gate
0011	AND Gate
0100	NAND Gate
0101	XOR Gate
0110	XNOR Gate
0111	A+ B (Penjumlahan)
1000	A – B (Pengurangan)
1001	B – A

Table 2 Set Instruksi ALU

	Bypass Adder (OR)	Set Carry 0 (XOR)	Set Carry 1	Cin ON	Invert A	Invert B
Zero	0	0	0	0	0	0
OR	1	1	0	0	0	0
NOR	1	1	1	0	0	0
AND	0	1	1	0	1	1
NAND	0	1	0	0	1	1
XOR	0	1	0	0	0	0
XNOR	0	1	1	0	0	0
A+ B	Default					
A – B	0	0	0	1	0	1
B – A	0	0	0	1	1	0

Dalam rangka memodelkan ALU dari MIPS menggunakan *Redstone* di *Minecraft* ini, desain ALU yang dibangun lebih padat dan menggunakan *decoder* untuk memudahkan dalam menyalakan instruksi yang akan dieksekusi. ALU *cascading* yaitu metode meneruskan hasil operasi untuk ALU lain dengan menghubungkan *output* dari ALU yang dihubungkan ke *input* ALU lain menggunakan jalur *bypass*. *Bypass Adder* diperlukan untuk melewatkan rangkaian *Adder* atau mengabaikan *Adder* agar logika *OR* dan *NOR* dapat menghasilkan keluaran sesuai tabel kebenaran. Untuk menjalankan instruksi *OR* perlu mengaktifkan *Bypass Adder* dan *Set carry 0* dengan memberikan nilai 1 sebagai indikator arus menyala.

Set Carry 0 digunakan untuk mempresentasikan tidak diperlukannya nilai *overflow* dalam digit selanjutnya atau bisa juga digunakan sebagai pemanggilan instruksi *XOR* untuk mengeksekusi nilai gerbang logika lainnya. Untuk nilai *Set Carry 1* mempresentasikan *overflow* dalam digit selanjutnya dari penjumlahan dengan banyak digit. Untuk rangkaian *Half Adder* secara sederhana tersusun atas kombinasi gerbang logika *XOR* dan *AND*, serta nilai *carry* tidak disertakan dalam penjumlahan. Dengan *input* A dan B melalui gerbang *XOR* menghasilkan output *Sum* maka hanya perlu mengaktifkan *Set Carry 0* dengan memberikan nilai 1, sementara input A dan input B yang melewati gerbang logika *AND* menghasilkan *output Carry*, maka perlu mengaktifkan *Set Carry 0* sebagai jalur pelaksanaan dan *Set Carry 1* untuk menghasilkan *output* 1 dengan *input*-an A dan B bernilai 1 menggunakan instruksi *invert A* dan *invert B*. Rangkaian *Full Adder* menjumlahkan bilangan *binary* dengan menyertakan nilai *carry* dalam penjumlahannya, terdapat *input*-an *Cin* didalamnya yang merupakan nilai bit *carry* dari langkah sebelumnya. *Cin* diperlukan untuk instruksi pengurangan, jika rangkaian *subtractor* memerlukan untuk meminjam bit jika bit yang dikurangi lebih kecil dari pada bit pengurang.

Invert A dan *Invert B* menggunakan komplemen dua *(two's complement)* dengan membalikkan nilai dalam proses negasinya semua bit juga akan dibalik. Instruksi subtractor memerlukan mengaktifkan inputan paling akhir ke salah satu invert untuk proses pengurangan.

4. Evaluasi

Untuk melakukan hasil dari pemodelan yang telah dilakukan guna membangun pemodelan ALU dari MIPS menggunakan *Redstone* di *Minecraft*, maka akan dilakukan analisis mengenai ALU menggunakan *Minecraft* . Implementasi ini digunakan untuk melihat rangkaian digital menggunakan *Redstone* dalam menerjemahkan dan menjalankan instruksi. Evaluasi yang kedua adalah melihat gerbang-gerbang logika sudah relevan atau belum dimodelkan menggunakan *Minecraft.* Evaluasi yang ketiga adalah banyaknya sumber daya yang dipakai dalam implementasi pemodelan ALU.

Tujuan dilakukan ketiga evaluasi tersebut adalah mengetahui bahwa *game Minecraft* dapat digunakan untuk belajar sambil bermain dan juga membantu dalam penalaran terhadap logika. Serta memungkinkan membantu dalam menentukan pembangunan arsitektur MIPS yang baik.

4.1 Analisis dan Hasil Pengujian

Untuk melakukan analisis pengujian diperlukan beberapa skenario yang memetakan pengujian yang dilakukan. Skenario ini disusun berdasarkan kebutuhan untuk melakukan evalusi pemodelan yang telah dibangun. Pengujian ini bertujuan untuk mengetahui bagaimana pemodelan ALU dari MIPS menggunakan *Redstone* di *Minecraft*.

4.1.1 Analisis dan Hasil Pengujian

Pengujian dilakukan menggunakan *Redstone* di *Minecraft* menggunakan diagram gambar dan diagram batang untuk melihat banyaknya sumber daya yang digunakan dalam pemodelan ALU. Pada penelitian ini, dilakukan beberapa tahapan skenario dari pengujian yang dilakukan:

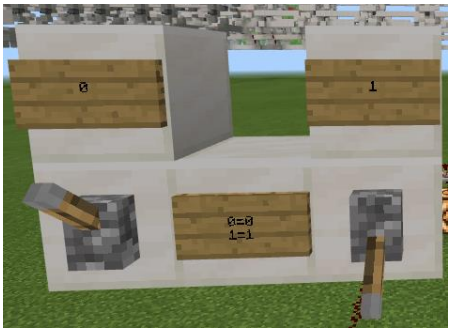
1. **Skenario 1: Melihat pada diagram gambar dari hasil pemodelan ALU menggunakan Redstone di Minecraft untuk melihat apakah rangkaian dapat berperilaku benar secara fisik.**
   Pada skenario ini, dari ALU yang sudah dibangun dari tabel kebenaran dan simbol rangkaian aritmatika dan logika untuk melihat pemodelan dari ALU menggunakan Redstone. Pengujian dilakukan pada sebuah gerbang-gerbang logika yang digabungkan menjadi satu bagian yaitu ALU. Diagram gambar dipilih karena visualisasi data yang akan digunakan dalam melakukan pengujian agar lebih jelas dalam proses penyampainnya.
2. **Skenario 2: Melihat pada diagram batang untuk mengetahui berapa banyak sumber daya yang digunakan.**
   Pada skenario ini, diagram batang dibuat berdasarkan jumlah sumber daya yang digunakan dalam memodelkan ALU. Terdapat 3 bagian yang akan dihitung pemakaian sumber dayanya yaitu *ALU, ALU Decoder* dan *Opcode.*

4.1.2 Analisis Hasil Pengujian

Berangkat dari skenario pengujian yang telah dibuat, dapat dilakukan analisis mengenai keluaran dari masing-masing skenario uji. Berikut merupakan hasil analisis yang telah dilakukan:

4.1.2.1 Hasil Pengujian Skenario 1: Melihat pada diagram gambar dari hasil pemodelan ALU menggunakan *Redstone* di *Minecraft* untuk melihat apakah rangkaian dapat berperilaku benar secara fisik.

Pada skenario pengujian 1, diambil beberapa hasil pemodelan untuk melihatkan gambaran secara umum pemodelan yang dilakukan dengan menggunakan *Redstone* di *Minecraft*. Berikut beberapa dasar pemodelan dan membangun ALU menggunakan *Redstone* di *Minecraft*:

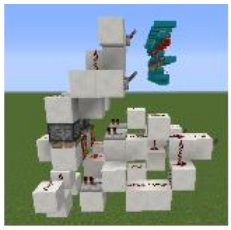


Gambar 5 Ketentuan nilai *input*-an pada *lever* atau saklar

Pada gambar 5, terdapat *lever* atau saklar sebagai masukan yang akan memberikan nilai untuk instruksi yang dijalankan. Dengan ketentuan, jika *lever* atau saklar pada posisi ke atas akan memberikan nilai 0 atau arus yang dihasilkan akan padam dan jika lever atau saklar pada posisi ke bawah akan memberikan nilai 1 atau arus yang dihasilkan akan menyala.

(a) Tampak depan rangkaian 1-bit Adder.

(b) Tampak samping kanan rangkaian 1-bit Adder.

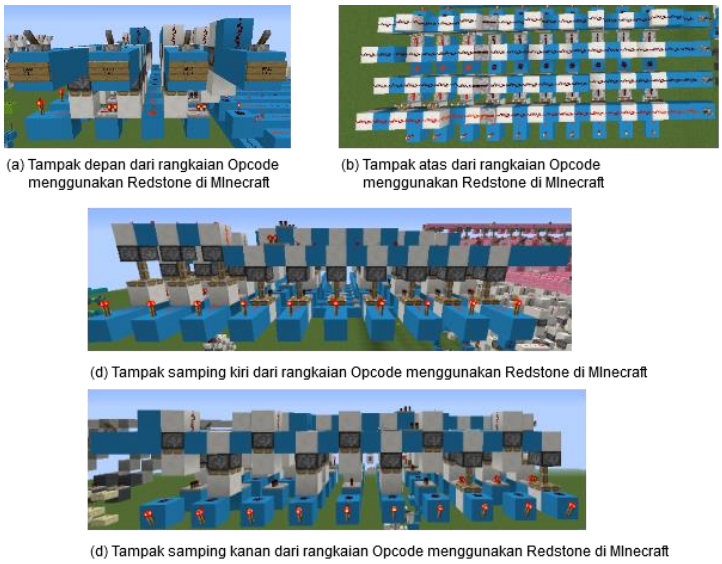


(c) Tampak samping kiri rangkaian 1-bit Adder.

(d) Tampak belakang rangkaian 1-bit Adder.

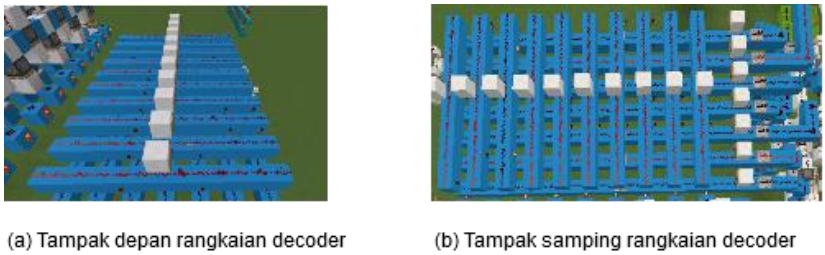
Gambar 6 Rangkain *Compact Full Adder*

Pada gambar 6, rangkain *Adder* yang akan digunakan dalam memodelkan ALU akan lebih sederhana dengan menggabungkan gerbang logika menjadi lebih padat. Dimana dengan menggunakan model ini sumber daya yang digunakan akan lebih sedikit dan arsitektur ALU yang dibangun lebih sederhana. Dapat dilihat dengan memodelkan arsitektur ALU yang lebih sederhana akan memudahkan dalam menyatukan instruksi-instruksi lainnya.


(a) Tampak depan dari rangkaian Opcode menggunakan Redstone di MInecraft
(b) Tampak atas dari rangkaian Opcode menggunakan Redstone di MInecraft
(d) Tampak samping kiri dari rangkaian Opcode menggunakan Redstone di MInecraft
(d) Tampak samping kanan dari rangkaian Opcode menggunakan Redstone di MInecraft

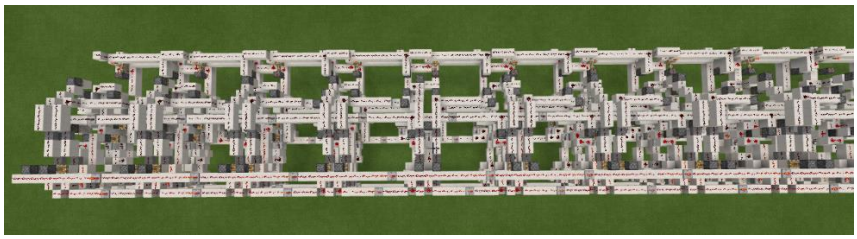
Gambar 7 Rangkaian *Operation Code* 4-bit

Pada gambar 7, *Operation Code (Opcode)* yaitu kode operasi berbentuk biner akan menentukan instruksi mana yang akan dilakukan atau diproses. Pada (gambar 7a) bagian *input*-an berupa biner, misalkan untuk menjalankan instruksi *AND*, pada tabel 1 logika *AND* terletak pada posisi ke-3 maka *Opcode* yang diperlukan dengan *input*-an 0011 untuk meneruskan instruksi pada *decoder*.


(a) Tampak depan rangkaian decoder
(b) Tampak samping rangkaian decoder

Gambar 8 Rangkaian *Decoder*

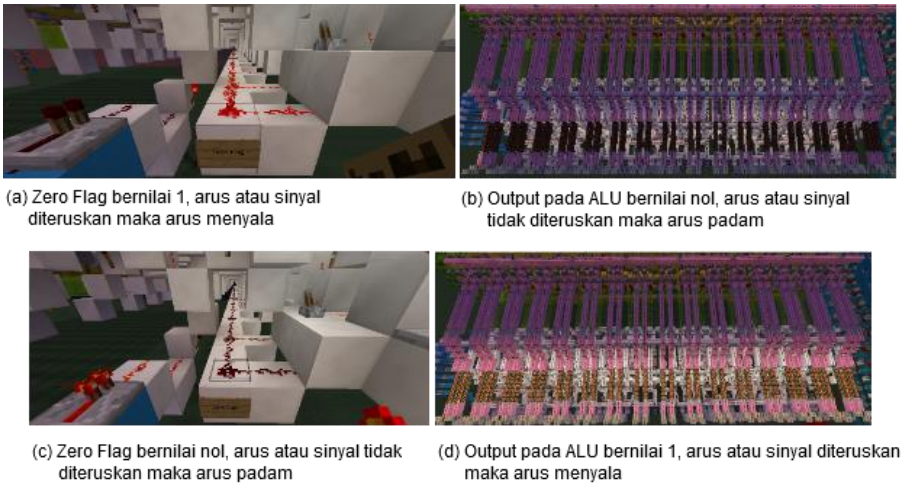
Pada gambar 8, evaluasi selanjutnya ini *Decoder* akan menerima instruksi yang telah dipilih oleh *Opcode* dan menyalakan instruksi yang telah terpilih. Pada tabel 2 fungsi yang akan diaktifkan akan bernilai 1. Dalam rangkain *Decoder* fungsi yang akan dijalankan atau bernilai 1 memerlukan *Redstone Torch* (obor) untuk mengaktifkan arus difungsi yang bernilai 1. Misalkan pada instruksi *AND* fungsi yang diperlukan yaitu *Set Carry 0*, *Set Carry 1*, *Invert A* dan *Invert B* akan bernilai 1 atau arus menyala. Maka diperlukan penempatan *Redstone Torch* (obor) dibagian baris fungsi *Set Carry 0*, *Set Carry 1*, *Invert A* dan *Invert B*. Sedangkan baris fungsi lainnya yang tidak diperlukan tidak perlu menempatan *Redstone Torch* (obor) untuk dapat menjalankan instruksi *AND*.



Gambar 9 Rangkain ALU 32-bit

Pada gambar 9, merupakan rangkaian ALU 32-bit yang telah dimodelkan menggunakan *Redstone* di *Minecraft*. Rangkaian tersebut terdiri dari logika *OR*, *NOR*, *AND*, *NAND*, *XOR*, *XNOR* dan aritmatika ADD dan SUB. Unit aritmatika dan logika akan berjalan ketika instruksi *Opcode* dan *Decoder* telah meneruskan arus atau sinyal ke dalam rangkain

ALU. Pada gambar 9 input-an masih berupa biner yang dilakukan secara manual dalam input-annya, misalkan ketika instruksi *ADD* diaktifkan dan ingin melakukan operasi penjumlahan, contohnya penjumlahan 4 + 3 maka perlu memasukkan input A berupa biner 0000000000000000 0000000000000100 dan *input*-an B berupa biner 0000000000000000 0000000000000110 (32-bit) maka keluaran yang akan dihasilkan berupa biner 0000000000000000 0000000000001110 ⟶ 7. Untuk contoh logikanya, misal instruksi *AND* dijalankan maka *AND* akan bernilai *true* jika input A dan input B masing-masing bernilai *true*, dengan menggunakan *Opcode* dan *Decoder* maka secara otomatis input A dan input B akan bernilai *true* sehingga hasil keluaran yang dihasilkan oleh ALU berupa biner 1111111111111111 1111111111111111 ⟶ *True*.


(a) Zero Flag bernilai 1, arus atau sinyal diteruskan maka arus menyala
(b) Output pada ALU bernilai nol, arus atau sinyal tidak diteruskan maka arus padam
(c) Zero Flag bernilai nol, arus atau sinyal tidak diteruskan maka arus padam
(d) Output pada ALU bernilai 1, arus atau sinyal diteruskan maka arus menyala

Gambar 10 Bagian Rangkaian pada ALU yang memperlihatkan *Zero Flag* bekerja

Pada gambar 12, *Zero Flag* digunakan sebagai indikator apakah nilai keluarannya nol atau bukan. Jika nilai keluarannya nol maka *Zero Flag* bernilai 1 dan sebaliknya bernilai nol. Dari (gambar 12b) melihatkan keluarannya berupa nol maka arus atau sinyal padam dan pada (gambar 12a) *Zero Flag* bernilai 1 maka arus atau sinyal menyala. Dan dari (gambar 12d) melihatkan keluarannya berupa 1 maka arus atau sinyal menyala dan pada (gambar 12c) Zero Flag bernilai nol maka arus atau sinyal padam.

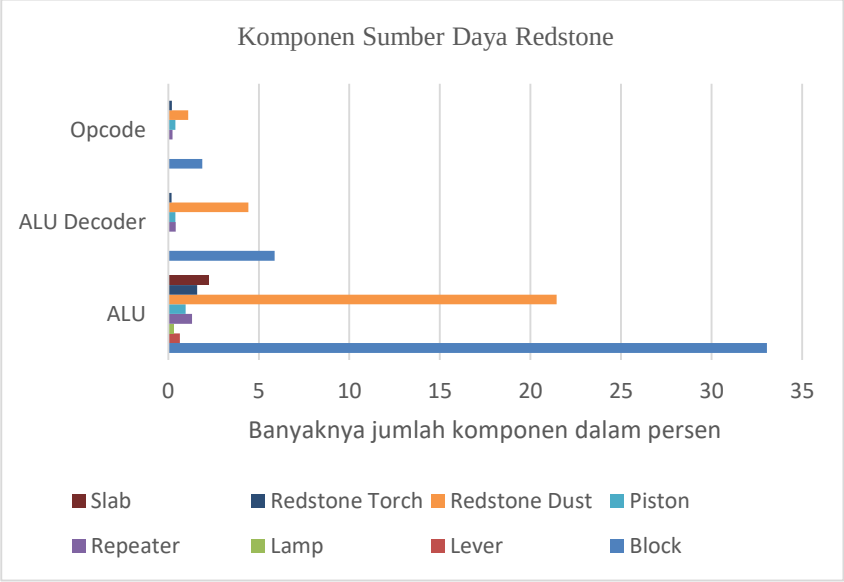
4.1.2.2 Hasil Pegujian Skenario 2: Melihat pada diagram batang untuk mengetahui berapa banyak sumber daya yang digunakan.

Diagram batang digunakan sebagai meninjau seberapa banyak sumber daya *Redstone* yang digunakan dalam membangun ALU 32-bit didalam *Minecraft*. Melalui diagram batang ini akan melihatkan banyaknya sumber daya Redstone yang diperlukan untuk memodelkan ALU 32-bit.

Table 3 Banyaknya Komponen Sumber Daya *Redstone* di *Minecraft* yang digunakan

Komponen Sumber Daya digunakan	Bentuk	Fungsi	ALU	ALU Decoder	Opcode
Blok		Blok digunakan untuk mendukung komponen *redstone* dan untuk mengirimkan daya.	3306	587	188
Lever		*Lever* dapat dikatakan seperti sakelar, dimana sinyal akan menyala jika ‘on’ dan akan padam jika ‘off’.	65	0	4

Lamp		*Redstone lamp* untuk memberikan cahaya.	32	0	0
Repeater		1. Dapat memperpanjang sinyal *redstone* namun komponen ini tidak dapat menyala dari *powered block*. 2. Membuat sinyal jadi lurus ke depan (tidak dapat balik ke belakang atau ke samping seperti *redstone dust*) 3. Dapat membuat sistem *loop*. 4. Dapat mengunci *redstone repeater* lainnya	131	42	24
Piston		Piston digunakan untuk memindahkan blok atau entitas. Piston terdiri dari dua jenis: piston biasa hanya mendorong blok, sementara piston yang lekat dapat mendorong dan menarik blok.	96	40	40
Redstone Dust		Berfungsi sebagai suatu lintasan bagi sinyal yang bekerja atau menyala.	2144	443	110
Redstone Torch		Dapat mengantarkan sinyal ke segala arah dengan menggunakan *activation block*.	160	19	20
Slab		*Slab* digunakan untuk mendukung komponen *redstone* dan untuk mengirimkan daya agar tidak sinyal tidak saling berbenturan.	224	0	0



Gambar 11 Banyaknya jumlah komponen sumber daya *Redstone* yang digunakan dalam membangun ALU di *Minecraft*

Pada tabel 3, melihatkan komponen sumber daya *Redstone* yang dibutuhkan dalam memodelkan ALU, dari banyaknya sumber daya yang tersedia dalam memodelkan ALU hanya memerlukan sumber daya seperti *Block* sebagai wadah untuk mendukung komponen *Redstone* lainnya dapat menjalakan tugasnya dengan baik. *Lever* untuk menentukan masukkan biner 0 atau 1. *Lamp* untuk memudahkan melihat hasil keluaran bernilai 0 = arus padam atau bernilai 1 = arus menyala. *Repeater* sebagai memperpanjang arus atau sinyal, dikarenakan *Redstone Dust* yang berperan sebagai arus atau sinyal hanya dapat aktif sepanjang 15 arus, maka dari itu untuk memperpanjang arus atau sinyal agar merata diperlukan *Repeater*. *Piston* sebagai mendorong dan menarik blok agar menahan atau melepaskan arus atau sinyal, misalnya salah satu fungsi tidak diperlukan maka dengan *piston* akan menahan arus atau sinyal agar tidak ikut dieksekusi, *Redstone Torch* sebagai penahan atau memberikan arus. *Slab* sebagai komponen pendukung agar dalam pengiriman arus atau sinyal tidak saling bebenturan untuk beberapa kondisi.

Pada gambar 13, dengan banyaknya jumlah komponen yang digunakan dalam memodelkan ALU menggunakan *Redstone* di *Minecraft* ini guna meninjau arsitektur yang dimodelkan sudah sederhana atau belum. Sangat penting dalam pemilihan bentuk desain yang akan dibangun karena semakin rendah level model yang akan dibangun maka semakin banyak sumber daya *Redstone* yang akan digunakan dan sangat membutuhkan waktu yang cukup lama dalam merancangnya.

4.2 Analisis Hasil Pengujian

Hasil pemodelan dari beberapa rangkaian gerbang-gerbang logika menunjukkan hasil pemodelan ALU dari MIPS dapat berperilaku benar secara fisik. Selain itu, ALU dirancang sesederhana mungkin guna mengoptimalkan ruang yang ada agar meminimalkan penggunakan sumber daya dan juga menghasilkan pemodelan sederhana namun tetap memiliki fungsionalitas yang sama.

Implementasi rangkaian ALU pada penelitian ini menggunakan *Redstone* di *Minecraft* melihatkan bahwa rangkaian digital dapat mengatur instruksi dalam komunikasi diantara rangkaian lainnya dan melihatkan masukkan dan keluaran dapat dibangun secara relevan.

Hasil pengujian dari dua skenario sebelumnya membuktikan bahwa rangkain ALU dapat dimodelkan menggunakan *Redstone* dan sumber daya yang dipakai memodelkan arsitektur dibangun dengan sederhana di *Minecraft*.

5. Kesimpulan dan Saran

7.1. Kesimpulan

Berdasarkan hasil dan analisis yang sudah dibangun sebelumnya dapat ditarik beberapa kesimpulan, berikut beberapa hal yang dapat disimpulkan:

1. Pemodelan ALU dapat dibangun di dalam *game Minecraft*, hanya saja membutuhkan waktu dan ketelitian yang cukup dalam implementasinya.
2. Pada rangkaian ALU dapat memperlihatkan bagaimana aritmatika dan logika diproses didalamnya.

7.2. Saran

Untuk mengembangkan kembali dan membuat penelitian ini menjadi lebih baik maka beberapa saran yang dapat dijadikan masukan dalam penelitian ini, antara lain:

1. Penelitian selanjutnya, diharapkan ALU dapat dibangun lebih baik lagi.
2. Penelitian selanjutnya, diharapkan dapat membangun pemodelan MIPS secara keseluruhan dan dengan masukkan integer.
3. Penelitian selanjutnya, diharapkan dapat melihatkan keseluruhan rangkaian arsitektur MIPS menjalankan instruksi-instruksi dari komponen satu dan lainnya.
4. Pada penelitian selanjutnya, diharapkan sumber daya yang dipakai lebih sedikit.
5. Pada penelitian selanjutnya, diharapkan dapat melihatkan performansi mikroprosesor MIPS berjalan sesuai konsep.

DAFTAR PUSTAKA

[1] J. Stuyts and J. Driesen, "Lumen: Educating youngsters about energy, using Minecraft," in *IEEE Power and Energy Society General Meeting (PESGM)*, Boston, MA, USA , 2016.

[2] T. Alstad, . J. . R. Duncan, S. Detlor, B. French, . H. Caswell, Z. Ouimet, Y. Khmelevsky , G. Hains, R. Bartlett and A. Needham, "Minecraft Computer Game Performance Analysis and Network Traffic Emulation by a Custom Bot," in *Science and Information Conference*, London, 2015.

[3] Y. Huh, G. . T. Duarte and M. . E. Zarki, "MineBike: Exergaming with Minecraft," in *International Conference on e-Health Networking, Applications and Services (Healthcom)*, Ostrava, Czech Republic, 2018.

[4] T. Alstad, J. R. Dunkin, . S. Detlor and B. , "Game Network Traffic Simulation by a Custom Bot," in *Annual IEEE Systems Conference (SysCon) Proceedings*, Vancouver, BC, Canada , 2015.

[5] C. Patrascu and S. Risi, "Artefacts: Minecraft meets Collaborative Interactive Evolution," in *IEEE Conference on Computational Intelligence and Games (CIG)*, Santorini, Greece , 2016.

[6] K. T. Foerster , "Teaching Spatial Geometry in a Virtual World: Using Minecraft in Mathematics in Grade 5/6," in *IEEE Global Engineering Education Conference (EDUCON)*, Athens, Greece , 2017.

[7] R. L. Uy and N. M. D. Kho , "MIPSers: MIPS Extension Release 6 Simulator," in *International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*, Manila, Philippines , 2017.

[8] J. B. M. Persson and S. McManus, "List of redstone components," Mojang, Microsoft Studios, 2015 Juli 29. [Online]. Available: https://minecraft.gamepedia.com/List_of_redstone_components. [Accessed 2019 Juli 11].

[9] N. Ravindran and R. M. Lourde, "An optimum VLSI design of a 16-BIT ALU," in *2015 International Conference on Information and Communication Technology Research (ICTRC)*, Abu Dhabi, United Arab Emirates , 2015.

[10] M. A. Ahmed, M. A. M. El-Bendary, F. Z. Amer and S. M. Singy, "Delay Optimization of 4-Bit ALU Designed in FS-GDI Technique," in *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, Aswan, Egypt, Egypt , 2019.

[11] L. Wang, J. Ruan and D. Zhang, "MIPS CPU test system for practice teaching," in *2017 12th International Conference on Computer Science and Education (ICCSE)*, Houston, TX, USA , 2017.

[12] S. . S. Omran and A. K. Abdul-abbas , "Design and implementation of 32-Bits MIPS processor to Perform QRD Based on FPGA," in *International Conference on Engineering Technology and their Applications (IICETA)*, Al-Najaf, Iraq , 2018.

[13] M. S. P. Ritpurkar , P. G. D. Korde and P. M. . N. Thakare, "Design and Simulation of 32-Bit RISC Architecture Based on MIPS using VHDL," in *International Conference on Advanced Computing and Communication Systems*, Coimbatore, 2015.

[14] A. Suzuki, R. Kobayashi and H. Shimada, "Instruction Rearrangement and Path Limitation for ALU Cascading," in *2016 International Conference On Advanced Informatics: Concepts, Theory And Application (ICAICTA)*, George Town, Malaysia , 2016.

[15] A. F. Kirichenko, I. V. Vernik, M. Y. Kamkar, J. Walter, M. Miller, L. R. Albu and O. A. Mukhanov, "ERSFQ 8-Bit Parallel Arithmetic Logic Unit," in *IEEE Transactions on Applied Superconductivity*, NY, USA, 2019.

[16] M. F. Tolba, A. H. Madian and A. G. Radwan, "FPGA realization of ALU for mobile GPU," in *2016 3rd International Conference on Advances in Computational Tools for Engineering Applications (ACTEA)*, Beirut, Lebanon , 2016.

[17] J. Bergensten, M. Persson and S. McManus, "List of redstone components," MOJANG, Microsoft Studios, 2015 Juli 29. [Online]. Available: https://minecraft.gamepedia.com/List_of_redstone_components. [Accessed 11 Juli 2019].