

DESAIN DAN SIMULASI ARITHMETIC LOGIC UNIT DENGAN VHDL UNTUK PROCESSOR ELEMENT RISC ARSITEKTUR PARALEL PENGOLAHAN CITRA LAPCAM

W. Tri Hartono¹, E. Mozef. 2, Mushlihudin³

^{1,2}Jurusan Teknik Elektro, Politeknik Negeri Bandung
Jl. Gegerkalong Hilir Ds. Ciwaruga, Kotak Pos 6468 Bandung
rwtri_h@yahoo.com, wahyu@ic.ee.itb.ac.id

³Program Studi Teknik Elektro Universitas Ahmad Dahlan
Jl. Prof. Soepomo Janturan Yogyakarta 55164
udin@uad.c.id

Abstrak

Sebuah konsep baru arsitektur paralel untuk pengolahan citra LAPCAM (*Linear Array of Processor with Content Addressable Memory*), sedang dikembangkan. LAPCAM terdiri dari tiga komponen utama, yaitu *Processor Element (PE)*, *Memori MAM*, dan *Jaringan interkoneksi OAC*.

Arsitektur LAPCAM, mempunyai *feature* yang optimal antara jumlah *Processor Element (PE)* dan kecepatan eksekusi. Selain itu LAPCAM mempunyai jenis memori dan jaringan interkoneksi yang baru, *MAM (Multi-mode Access Memory)* dan *OAC (Orthogonal Addressable Crossbar)*. *Element* yang unik ini menjadikan LAPCAM sebagai suatu arsitektur paralel yang optimal untuk berbagai macam permasalahan pengolahan citra.

Dalam tulisan ini, akan dikupas secara rinci, disain dari sebuah *Arithmetic Logic Unit (ALU)* yang merupakan bagian terpenting dari *Processor Element (PE)*, dengan menggunakan *VHSIC Hardware Description Language (VHDL)*. Simulasi akan dikerjakan menggunakan *Software Max+plus II* dari ALTERA. Dari hasil simulasi diperlihatkan bahwa keluaran *WR_C* dan *WR_Z* hanya akan aktif pada instruksi-instruksi *ADD* dan *SUB* saja.

Kata kunci: Arsitektur paralel, Pengolahan citra, *Processor Element*, VHDL, ALTERA, LAPCAM

1. PENDAHULUAN

LAPCAM merupakan suatu disain baru dari arsitektur paralel untuk pemrosesan citra yang sedang dikembangkan saat ini [2][3][4][5].

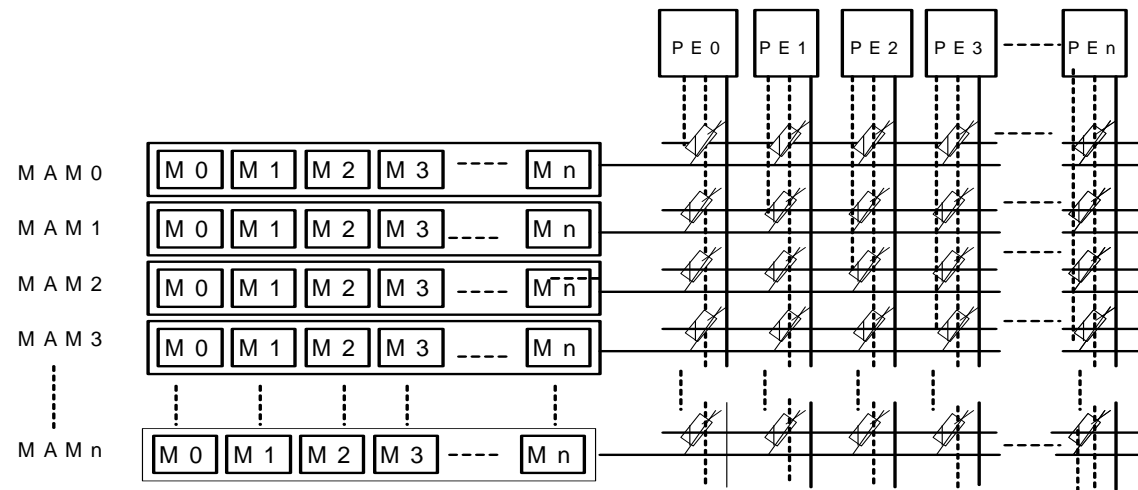
Arsitektur ini memiliki keunggulan-keunggulan bila dibandingkan dengan arsitektur sejenis, antara lain:

- Mempunyai *feature* yang optimal antara jumlah *Processor Element (PE)* dan kecepatan eksekusi. Dalam hal ini memungkinkan arsitektur LAPCAM dengan hanya 512 PE mempunyai performansi rata-rata sama atau bahkan lebih baik bila dibandingkan dengan arsitektur tipe 2d dengan 262144 PE.
- Arsitektur ini sangat memungkinkan direalisasi dengan teknologi aktual, dengan biaya yang relatif lebih murah bila dibandingkan dengan arsitektur jenis lain.

Secara umum struktur arsitektur LAPCAM

terdiri dari n PE, n *bank memory MAM*, dan $n \times n$ jaringan interkoneksi OAC, diperlihatkan pada Gambar 1 [2][3][4][5][7]. Sebuah *bank memory MAM* terdiri dari n sel. *Memory MAM* mempunyai 3 fungsi yang berbeda yaitu sebagai : *RAM (Random Access Memory)*, *CAM (Content Addressable Memory)*, dan *SM (Shift Memory)*

Pada disain arsitektur paralel yang optimal, aspek Prosesor, Memori, dan Interkoneksi memegang peranan sangat penting. Dalam konsep LAPCAM telah dibuktikan bahwa penggunaan memori MAM dan Interkoneksi OAC telah menjadikan LAPCAM sebagai sebuah arsitektur paralel yang optimal [1][2][3][4][5][7].



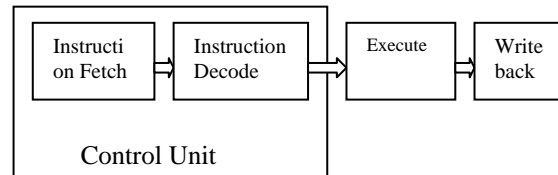
Gambar 1. Struktur Arsitektur LAPCAM, dengan n PE untuk pengolahan $n \times n$ pixel.

Processor Element (PE) untuk arsitektur paralel pengolahan citra LAPCAM di desain sebagai suatu prosesor sederhana yang mempunyai beberapa port dengan instruksi minimal, akan tetapi PE di desain untuk dapat bekerja dalam kecepatan tinggi. Dengan alasan ini maka sangat tepat untuk menggunakan arsitektur jenis RISC.

Format sederhana 8-bit data digunakan dengan alasan berikut :

1) untuk membuat desain sesederhana mungkin, dan 2) untuk dapat memanipulasi data yang biasa digunakan yaitu sebesar *256 pixel levels* [2][3][4][5] [7].

Konsep pemikiran RISC *Processor Element* didasarkan pada asumsi bahwa dengan *instruction set* dasar yang minimal yang dipadukan dengan *compiler* yang tepat akan menghasilkan unjuk kerja yang jauh lebih baik dibandingkan *Complex Instruction Set Computer* (CISC) dengan jumlah instruksi yang begitu banyak[3][4][5]. Dengan sederhananya operasi-operasi yang digunakan secara ideal, menja-dikan setiap instruksi hanya membutuhkan satu *cycle perocessor*[6].



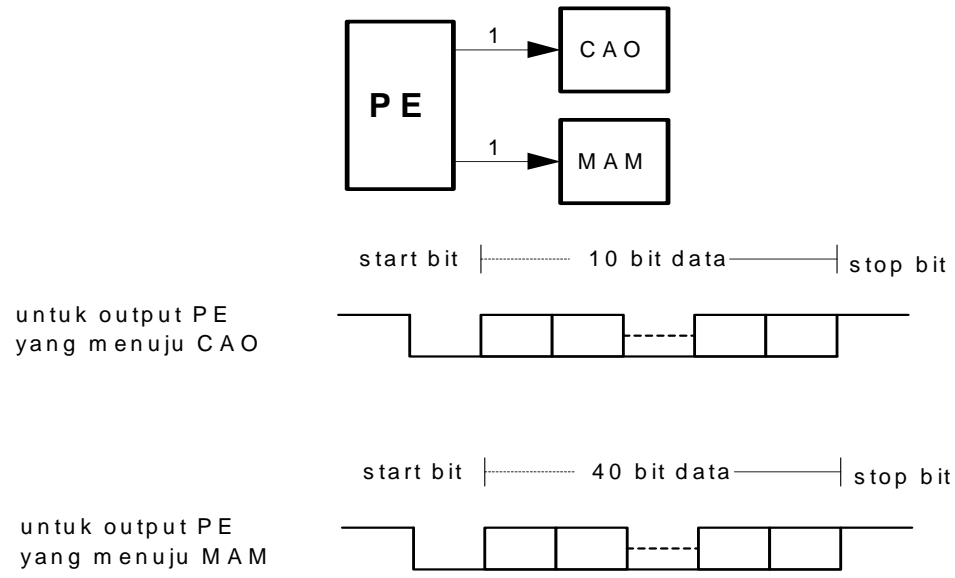
Gambar 2. RISC Processor Element Datapath

Basic data path suatu RISC *processor Element* ditunjukkan seperti pada gambar 2. *Instruction Decoder* men-load *instruction* yang ditunjuk oleh *program counter* (PC) dari memori prosesor. *Instruction Decoder* kemudian akan men-generate sinyal kontrol yang tepat untuk *Execution Unit*, yang kemudian melakukan suatu fungsi yang diinginkan (aritmetika, logika, dan sebagainya) pada data. Unit *Writeback* kemudian meng-update memori dengan nilai-nilai yang baru[6].

Prosesor Element (PE) merupakan mikro-prosesor 8-bit yang mempunyai kemampuan pengolahan data sebesar 8-bit. Arsitektur mikroprosesor ini terdiri dari ALU (*Arithmetic Logic Unit*) 8-bit, *register Accumulator* dengan lebar sebesar 8-bit dan 8-bit *Data RAM* sebanyak 32-buah. Selain itu, juga mempunyai kemampuan pengalamatan program memori, sebesar 512-byte.

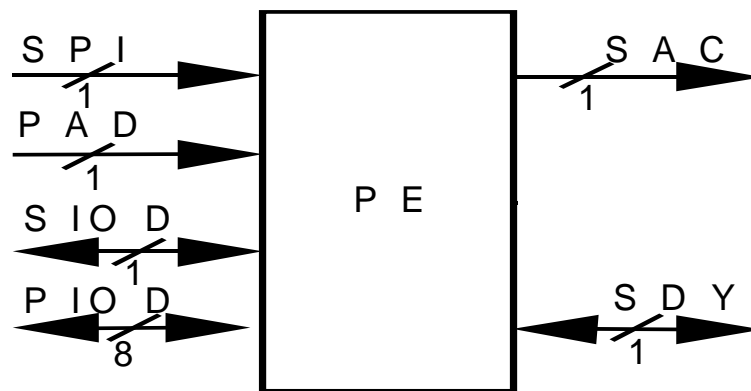
Protokol yang digunakan untuk mentransfer instruksi/data antara PE, MAM, dan OAC, berupa protokol *serial*, seperti pada Gambar 3, bit pertama merupakan start bit (active low), serta 1 bit terakhir digunakan untuk stop bit (active high).

Internal operation: untuk OAC 10 bit
 untuk MAM 5 x 8 bit
 Input/output data : 1 bit



Gambar 3 : Protokol Serial yang diterapkan antara PE-CAO dan PE-MAM

PE (Gambar 4) dirancang mempunyai enam port: SAC, SDY, SIOD, PIOD, PAD dan SPI. SAC merupakan *port serial output* 1-bit yang digunakan untuk mengaktifkan suatu rangkaian interkoneksi OAC.



Gambar 4. Processor Element (PE)

SDY adalah *serial input/output* 1 bit *bi-directional port* yang digunakan untuk mengirim instruksi serial 40-bit ke MAM atau menerima data serial 16-bit dari MAM. SIOD adalah *port input/output* serial 1 bit *bi-directional* untuk mengirim/menerima data ke/dari host PC. PIOD adalah *port parallel input/output bi-directional* 8-bit untuk mengirim/menerima data ke/dari *host* PC. PAD adalah *port serial input*. address 1-bit yang digunakan oleh *host* untuk men-select PE saat *host* ingin mengontrolnya. SPI adalah *port serial input* 1-bit yang digunakan untuk men-download program dari *host* (lihat tabel 1) [1][2][3][4][5][7].

PE mempunyai instruksi prosesor standar (contohnya data *transfer instructions*, *arithmetic*, *logical*, *flag*, *jump instructions etc*, lihat Tabel 3) dan *special instructions* (contohnya *MAM instructions* dan *OAC instructions*).

Tabel 1. Definisi Port

No	Port	size of Port/Direction	Protocol	Function
1	SAC	1/output	Serial	Untuk mengirim data *) ke OAC network.
2	SDY	1/ In/Out	Serial	Untuk mengirim data ke memori MAM via OAC network
3	SIOD	1/ In/Out	Serial	Untuk send/receive serial data to/from a host (PC)
4	SPI	1/ Input	Paralel	Untuk download program dari Host
5	PAD	1/ Input	Serial	Digunakan oleh Host untuk menseleksi PE saat Host ingin mengontrolnya

Keterangan: Data dalam tabel dapat mengandung data, address and/or instruksi seperti pada definisi.

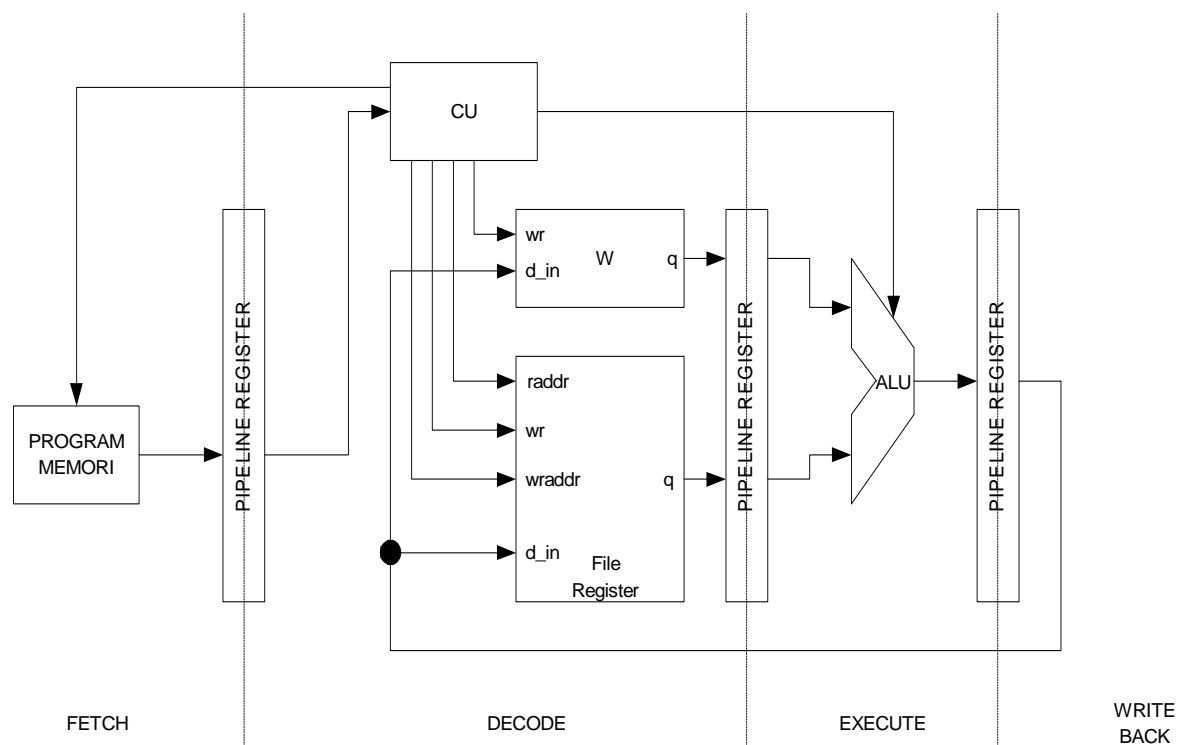
2. BLOCK DIAGRAM FUNGSIONAL PROCESSOR ELEMENT

Blok diagram arsitektur Processor Element (PE), dapat dilihat pada Gambar 5. Proses dimulai dari fetch, yang men-load program counter dengan alamat pada program memory.

Instruksi kemudian di fetch dari program memori dan dikirim ke instruction decoder, yang memuat operand dan bus function select dan mengenerate sinyal kontrol untuk operasi prosesor.

Tahap eksekusi melakukan suatu operasi atau interaksi dengan data memory, register, dan operasi aritmatika. Setelah tahap eksekusi hasilnya akan ditulis ke register-register dari prosesor, bila diperlukan.

3. BLOCK DIAGRAM ARITHMETIC LOGIC UNIT

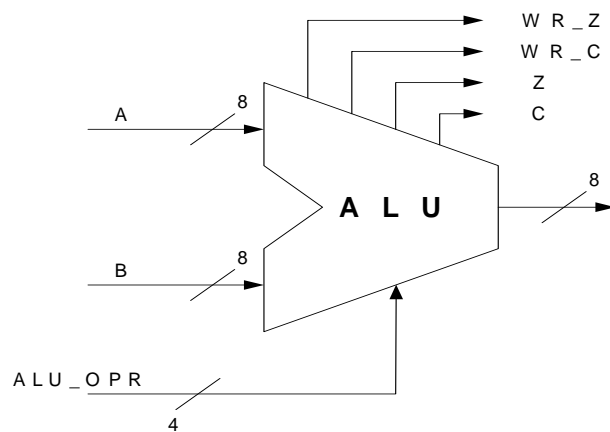


Gambar 5. Block Diagram Fungsional Processor Element

ALU merupakan komponen pada PE yang melakukan fungsi aritmetika (Penjumlahan, pengurangan), dan melakukan berbagai operasi dalam binary word. Gambar 6 menunjukkan diagram blok untuk 8-bit Arithmetic Logic Unit (ALU). ALU ini dapat melakukan sebanyak tujuh fungsi logika yaitu AND, OR, XOR, NOT, RL, RR, SWAP, empat fungsi aritmetika dalam dua

TELKOMNIKA Vol. 3, No. 1, April 2005 : 47 - 53

masukan 8-bit binary words yaitu ADD, SUB, DEC, INC DAN TIGA fungsi control CLC, CLZ, TEST ,lihat tabel 3. Fungsi-fungsi logika atau aritmetika yang dilakukan ditentukan oleh ALU_OPR, yang menseleksi jenis operasi yang akan dilakukan.



Gambar 6: Block Diagram Arithmetic Logic Unit

Dalam perancangan ini A ditetapkan sebagai input *binary word* 8-bit, yaitu A0-A7, dan B merupakan *input binary word* 8-bit B0-B7. Q merupakan *output binary word biner* 8-bit, Q0-Q7. Jenis operasi yang terjadi pada ALU, tergantung dari kombinasi bit pada *selector* ALU_OPR, maka ALU akan melakukan satu dari 14 fungsi logika, terhadap input A dan B dalam dua word, untuk lebih jelasnya perhatikan Tabel 2.

Tabel 2. Input dan Output Control Unit.

No	Nama	Lebar	Arah	Keterangan
1	A	8-bit	masukan	
2	B	8-bit	masukan	
3	Q	8-bit	keluaran	Hasil dari operasi
4	C	1-bit	keluaran	Keluaran carry flag
5	Z	1-bit	keluaran	Keluaran zero flag
6	wr_c	1-bit	keluaran	Sinyal tsb digunakan untuk menulis isi dari C ke register C
7	wr_z	1-bit	keluaran	Sinyal tsb digunakan untuk menulis isi dari Z ke register Z
8	ALU_opr	4-bit	keluaran	Sinyal ini digunakan untuk memilih operasi pada ALU.

4. SIMULASI

Simulasi *Arithmetic Logic Unit* dilakukan dengan menggunakan devais FLEX10K70RC240-4 dengan clock masukan sebesar 1/40ns=25 Mhz. Hasil dari simulasi tersebut dapat dilihat pada Gambar 7 [7].

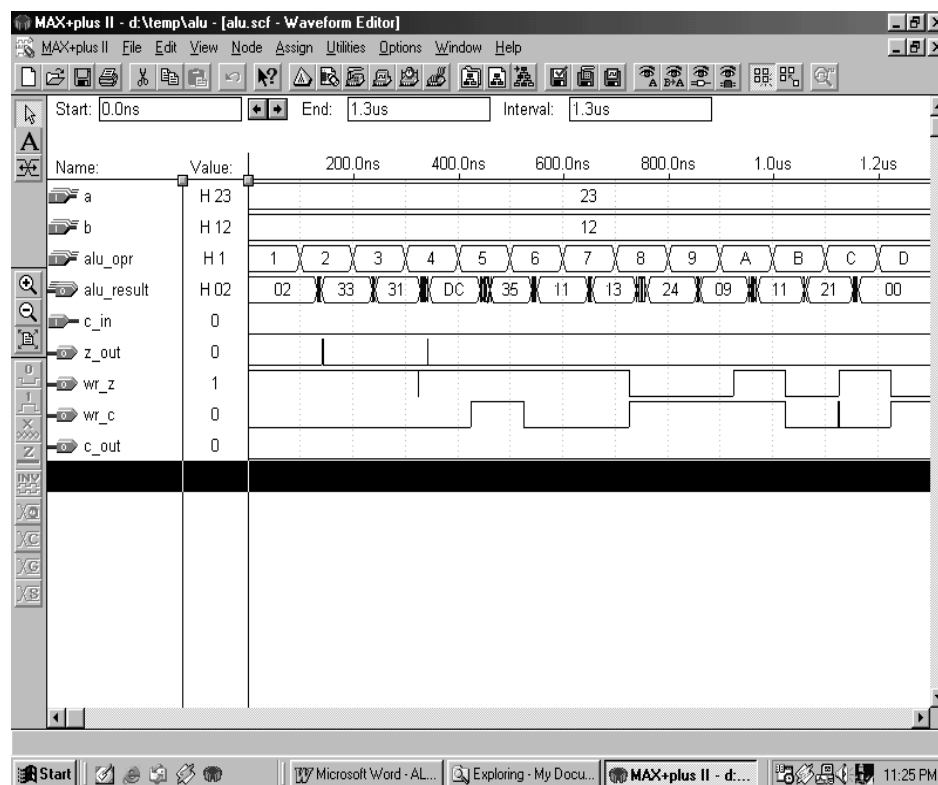
Tampak pada Gambar 7 hasil simulasi pada bagian ALU. Sinyal yang ditampilkan yaitu sinyal masukan **a** diberi nilai 23, sinyal **b** yang diberi nilai 12, sinyal alu_opr yang merupakan jenis operasi yang diberikan. Jenis-jenis operasi dapat dilihat di Tabel 3.

Untuk menguji komponen ALU, dilakukan pengujian sebagai berikut:

Masukan Alu **a** diberi nilai 23H (0010 0011)

Masukan Alu **b** diberi nilai 12H (0001 0010)

Control alu_opr yang merupakan selektor saat bernilai 1H (0001), yang berarti dipilih operasi logika AND (lihat tabel 3). Pada keluaran *alu_result* didapatkan nilai 02H (0000 0010), nilai ini sesuai dengan hasil operasi logika AND untuk nilai 23H dan 12H.



Gambar 7. Hasil simulasi File Register

Tabel 3. Operasi-operasi di Arithmetic Logic Unit

ALU_OPR	Fungsi	Logika
0001	AND	$Q = A \text{ AND } B$; $WR_Z = '1'$;
0010	OR	$Q = A \text{ OR } B$; $WR_Z = '1'$;
0011	XOR	$Q = A \text{ XOR } B$; $WR_Z = '1'$;
0100	NOT	$Q = \text{NOT } A$; $WR_Z = '1'$;
0101	ADD	$Q = A \text{ ADD } B$; $WR_Z = '1'$; $WR_C = '1'$;
0110	DEC	$Q = A - 1$; $WR_Z = '1'$;
0111	INC	$Q = A + 1$; $WR_Z = '1'$;
1000	RL	$Q = A \ll 1$; $WR_C = '1'$;
1001	RR	$Q = A \gg 1$; $WR_C = '1'$;
1010	SUB	$Q = A \text{ SUB } B$; $WR_Z = '1'$; $WR_C = '1'$;
1011	SWAP	$Q = B_{3-0}B_{7-4}$
1100	TEST	$Z = '1'$ WHEN $B = '0'$, ELSE '0'. $WR_Z = '1'$
1110	CLZ	$Z = '0'$; $WR_Z = '1'$;
1101	CLC	$C = '0'$; $WR_C = '1'$
1101	CLC	$C = '0'$; $WR_C = '1'$

Pada saat `alu_opr` bernilai 2H(0010), ini merupakan operasi logika *OR* (lihat tabel 3). Pada keluaran `alu_result` didapatkan nilai 33H (0011 0011), nilai ini sesuai dengan hasil operasi logika *OR* untuk nilai 23H dan 12H.

Pada saat `alu_opr` bernilai 5H(0101), ini merupakan operasi aritmetika penambahan *ADD* (lihat Tabel 3). Pada keluaran `alu_result` didapatkan nilai 35H (0011 0011), nilai ini sesuai dengan hasil operasi aritmetika *ADD* untuk nilai 23H dan 12H. Saat menjumlahkan (0010 0011 + 0011 0010), terdapat carry pada saat menjumlahkan 1 + 1 untuk bit ke 7, pada simulasi dapat dilihat bahwa nilai `carry_out` bernilai 1 (satu).

5. KESIMPULAN

1. Dalam Penelitian ini, telah berhasil disimulasikan ALU untuk PE berjenis RISC.
2. Karena merupakan logika kombinasional maka setiap operasi memerlukan delay/tundaan rata-rata sekitar 90ns jika diimplementasikan pada komponen 10K70RC240-4 dari ALTERA.
3. Dari hasil simulasi diperlihatkan bahwa keluaran `WR_C` dan `WR_Z` hanya akan aktif pada instruksi-instruksi *ADD* dan *SUB*.

DAFTAR PUSTAKA

- [1] E. Mozef, "**LAPCAM: An Optimal Parallel Architecture for Image Processing: Realization and Evaluation**" Presented at the ITSF seminar-29 January 2001
- [2] E. Mozef, S. Weber, J. Jaber, and E. Tisserand, "**Real-time connected component labeling on one-dimensional array processors based on Content-Addressable Memory: optimization and implementation**" UMIIST-IEEE 3rd International Workshop on Image and Signal Processing, Manchester, United Kingdom, Nov. 96, pp. 691-694.
- [3] E. Mozef, S. Weber, J. Jaber, and E. Tisserand, "**Design of linear array processors with Content-Addressable Memory for intermediate level vision**" ISCA-IEEE 9th International Conference on Parallel and Distributed Computing Systems, Dijon, France, Sept. 1996, pp. 585-588
- [4] E. Mozef, S. Weber, J. Jaber, and E. Tisserand, "**Parallel Architecture Dedicated to Connected Component Analysis**" IAPR-IEEE 13th International Conference on Pattern Recognition, Vienna, Austria, August 1996, pp. 699-703. (IEEE Computer Society Press)
- [5] E. Mozef, S. Weber, J. Jaber, and C. Bataille, "**LAPCAM, Linear Array of Processors using Content-Addressable Memories: A new design of machine vision for parallel image computations**" IAPR International Workshop on Machine Vision Applications, Tokyo, Japan, Nov. 96, pp. 166-169.
- [6] Hamblen, James O., Michael D. Furman, "**Rapid Prototyping of Digital System**" Kluwer Academic Publishers, 2000. Pp 1-6, pp.12-16
- [7] W. Tri Hartono, "**Perancangan dan Simulasi VHDL RISC Processor Element untuk Arsitektur Paralel Pengolahan Citra LAPCAM**" Tesis Magister Teknik Elektro, Institut Teknologi Bandung (ITB), Bandung, Juni 2002.