

Implementasi Sistem Kendali ON-OFF pada *Field Programmable Gate Array* (FPGA)

Yohanes B. Gunawan Sugiarta, Feni Isdaryani[#], Dianthika P. Andini, Rina Hikmawati

Jurusan Teknik Elektro, Politeknik Negeri Bandung
Jl. Gegerkalong Hilir, Kabupaten Bandung Barat, Indonesia
[#]feni.isdaryani@polban.ac.id

Abstrak

Field Programmable Gate Array (FPGA) adalah komponen semikonduktor yang didasarkan pada matriks blok logika yang dapat dikonfigurasi oleh pemakainya dan terhubung melalui interkoneksi yang *programmable*. FPGA dapat diprogram sesuai dengan aplikasi atau fungsi yang diperlukan oleh pengguna. Untuk menunjang ilmu pengetahuan di bidang teknik elektronika, maka perlu adanya media pembelajaran mengenai dasar penggunaan dan penerapan FPGA. Oleh karena itu, pada penelitian ini dibuatlah modul simulator kendali ON-OFF menggunakan FPGA. Terdapat dua pendekatan yang digunakan dalam membangun kendali ON-OFF, yaitu ON-OFF murni dan ON-OFF menggunakan waktu tunda, dimana waktu tunda ini didefinisikan menggunakan logika *counter*. Seluruh perancangan algoritma pengendali menggunakan deskripsi *VHSIC Hardware Description Language* (VHDL). Hasil implementasi menunjukkan bahwa respon dapat sesuai dengan kondisi yang diinginkan. Untuk kondisi ON-OFF dengan logika *counter*, motor DC dapat ON selama 5 detik selanjutnya OFF secara otomatis sesuai dengan *behaviour* yang diinginkan.

Kata kunci: FPGA, VHDL, ON-OFF, motor DC

Abstract

A *Field Programmable Gate Array* (FPGA) is a semiconductor component based on a matrix of user-configurable logic blocks connected through programmable interconnects. FPGA can be programmed according to the application or function required by the user. To support science in the field of electronic engineering, it is necessary to have learning media regarding the basic use and application of FPGA. Therefore, in this study, an ON-OFF control simulator module was made using an FPGA. There are two approaches used in building ON-OFF control, namely pure ON-OFF and ON-OFF using delay time, where this delay time is defined using counter logic. All control algorithm designs use the *VHSIC Hardware Description Language* (VHDL) description. The implementation results show that the response can be in accordance with the desired conditions. For the ON-OFF condition with counter logic, the DC motor can be ON for 5 seconds then OFF automatically according to the desired behaviour.

Keywords: FPGA, VHDL, ON-OFF, DC motor

I. PENDAHULUAN

Perkembangan teknologi di bidang elektronika sudah semakin meningkat, salah satunya adalah perkembangan pada komponen mikrokomputer. Sejarah mikrokomputer diawali dengan ditemukannya *Integrated Circuit* (IC), selanjutnya ditemukannya mikroprosesor. Dalam perancangan mikroprosesor sebagai pusat pengendali dan pengolahan data pada sistem komputer mikro, diperlukan komponen register unit, kontrol unit serta *Arithmetic and Logic Unit* (ALU). Kendala yang sering dihadapi dalam merancang mikroprosesor adalah mahal dan lamanya fabrikasi,

oleh karena itu dapat diimplementasikan ke dalam *Field Programmable Gate Array* (FPGA) [1].

FPGA adalah komponen semikonduktor yang didasarkan pada matriks blok logika yang dapat dikonfigurasi oleh pemakainya dan terhubung melalui interkoneksi yang *programmable*. FPGA dapat diprogram berdasarkan aplikasi atau fungsi sesuai dengan yang diinginkan. Interkoneksi dapat dengan mudah diprogram ulang, sehingga memungkinkan FPGA untuk mengakomodasi perubahan pada desain atau bahkan mendukung aplikasi baru selama masa pakainya. Kelebihan lain dari FPGA yaitu dapat dikonfigurasi oleh *end user* dan tidak memerlukan proses fabrikasi [2]. Karena

kemampuannya yang dapat diprogram ulang, FPGA banyak diterapkan pada bidang sistem kendali, *data centre*, *enterprise networking*, sampai pada *Automotive Advanced Driver Assistance Systems* (ADAS).

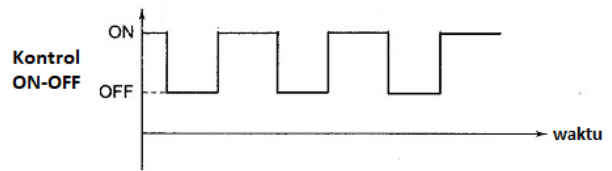
Pemrograman pada FPGA cukup menantang. Hal ini dikarenakan beberapa alasan, seperti FPGA merupakan perangkat keras digital, sehingga dibutuhkan pengetahuan dasar akan rangkaian digital agar dapat memahami dan menggunakannya [3]. Dikarenakan dasar penggunaan FPGA adalah pemrograman, maka diperlukan pengetahuan dasar akan pemrograman. Namun, hal ini tidaklah sederhana karena diperlukan pemahaman paralel akan rangkaian digital dan bahasa pemodelan dari FPGA tersebut. Pada [4] dilakukan penelitian terhadap ancaman keamanan dan mengidentifikasi gangguan tersembunyi yang dapat terjadi selama mengimplementasikan FPGA. Metode yang digunakan terdiri dari tiga teknik, yaitu meningkatkan *file* batasan pengguna, pemilihan desain replika secara acak dan perakitan *submodul runtime*. Keseluruhan implementasi pada FPGA merupakan kombinasi kerjasama antara *software* dan *hardware* yang digunakan juga bahasa deskripsi perilaku dari sistem. Saat ini semakin meningkat aplikasi penggunaan FPGA, berbagai struktur diterapkan untuk mendapatkan performa yang semakin baik. Seperti pada [5] melakukan penelitian untuk mencari *routing* yang paling efisien pada multi-FPGA, karena penggunaan multi-FPGA ini berkecepatan tinggi, murah, dan dapat digunakan pada aplikasi yang kompleks.

Untuk menunjang ilmu pengetahuan di bidang teknik elektronika, maka perlu adanya media pembelajaran mengenai dasar penggunaan FPGA dan penerapan sederhananya. Telah banyak penelitian mengenai desain dan implementasi FPGA pada bidang akademik seperti pada [6] yang mendesain purwarupa mikroprosesor berbasis FPGA ALTERA EPF10K10. Kemudian, pada [7] mengimplementasikan alat identifikasi odor menggunakan FPGA jenis Spartan 3E produksi Xilinx. Oleh karena itu, maka pada penelitian ini dirancang modul simulator kendali ON-OFF menggunakan FPGA. Dimana kendali ON-OFF dibangun berdasarkan dua pendekatan yaitu ON-OFF murni dan ON-OFF dengan waktu tunda.

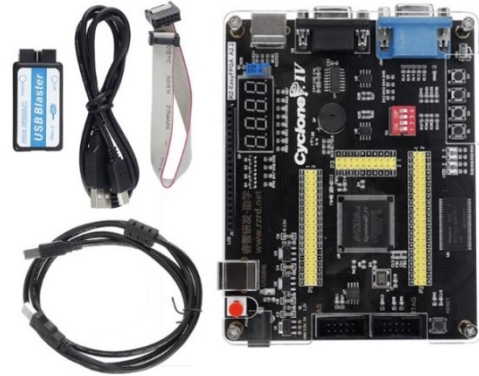
II. METODE PENELITIAN

A. Kontrol ON-OFF

Kontrol ON-OFF merupakan salah satu jenis pengendali yang paling sederhana. Prinsip kerja dari kontrol ON-OFF atau disebut juga dengan kendali dua posisi adalah jika *output* sistem berbeda



Gambar 1. Grafik respon kendali ON-OFF



Gambar 2. FPGA Cyclone IV EP4CE6E22C8N

dengan *input* yang diberikan, maka pengendali akan menghasilkan aksi kontrol (*manipulated value/ MV*) untuk merubah kondisi. Dalam kendali ON-OFF, hanya ada dua kondisi MV yaitu ON dan OFF (*high* dan *low*). Ilustrasi respon kendali ON-OFF terhadap waktu dapat dilihat pada Gambar 1. Kontrol ON-OFF masuk ke dalam kategori sistem lup terbuka, dimana *output* dari sistem tidak mempunyai pengaruh terhadap aksi kontrol. Dengan demikian, *output* dan *input* dari sistem saat *steady state* memiliki karakteristik yang statis.

B. Field Programmable Gate Array (FPGA)

Berdasarkan pada penelitian [8], FPGA dapat digunakan sebagai kontroler untuk mengatur kecepatan dengan mengatur *Pulse Width Modulation* (PWM), skema desain sirkuit terintegrasi *Space Vector PWM* (SVPWM), optimalisasi frekuensi *switch* yang dinamis, modulasi dinamis dengan variasi DC-link, linier dan strategi over-modulasi. Seluruh desain diimplementasikan dan diverifikasi pada satu Altera FPGA. Metode desain *Top-Down* baru pada FPGA diusulkan dalam makalah ini. Pada [9], FPGA diimplementasikan untuk mengatur kecepatan motor *Brushless DC* (BLDC) dengan teknik pengaturan PWM.

Adapun jenis-jenis FPGA sangat bermacam dan produsennya pun bervariasi. Seperti pada [7], FPGA yang digunakan adalah jenis Spartan 3E yang juga produksi Xilinx. Pada penelitian yang dilakukan oleh [6] menggunakan FPGA jenis Altera EPF10K10 produksi Intel. Pada penelitian ini akan digunakan FPGA jenis Altera Cyclone IV dengan spesifikasi yang

tertera pada [10]. Secara umum, FPGA jenis Altera Cyclone IV EP4CE6E22C8N berserta dengan set USB *blaster* dan kabel *power* yang digunakan dalam penelitian ini dapat dilihat pada Gambar 2.

C. Very High-Speed Integrated Circuit Hardware Description Language (VHDL)

VHSIC *Hardware Description Language* (VHDL) menjadi semakin populer sebagai cara untuk membuat rangkaian elektronik digital yang kompleks baik untuk simulasi dan sintesis. Rangkaian digital yang dirangkai menggunakan VHDL dapat dengan mudah disimulasikan, lebih mungkin untuk disintesis menjadi beberapa teknologi target, dan dapat diarsipkan untuk modifikasi dan penggunaan kembali nanti.

VHDL adalah bahasa pemrograman yang telah dirancang dan dioptimalkan untuk menggambarkan perilaku rangkaian dan sistem digital [11]. VHDL dapat memodelkan perilaku dan struktur sistem digital pada berbagai tingkat abstraksi, mulai dari tingkat sistem hingga gerbang logika, untuk entri desain, dokumentasi, dan tujuan verifikasi. Sejak 1987, VHDL telah distandarisasi oleh *Institute of Electrical and Electronics Engineers* (IEEE) sebagai IEEE Std 1076. Versi terbaru (per April 2020) diantaranya adalah IEEE Std 1076-2019. Untuk memodelkan sistem analog dan sinyal campuran, HDL standar IEEE berdasarkan VHDL yang disebut VHDL-AMS (secara resmi IEEE 1076.1) telah dikembangkan. VHDL dinamai sesuai dengan program Departemen Pertahanan Amerika Serikat yang membuatnya, *Very High-Speed Integrated Circuits Program* (VHSIC). Pada awal 1980-an, Program VHSIC mencari HDL baru untuk digunakan dalam desain sirkuit terintegrasi yang bertujuan untuk pengembangan. Contoh *behaviour* pada bahasa pemrograman VHDL dapat dilihat pada Gambar 3.

```

1  entity circuit_1 is
2      Port (a:in STD_LOGIC;
3           b:in STD_LOGIC;
4           out1:out STD_LOGIC);
5  end circuit_1;

-----

6  architecture Behavioral of circuit_1 is
7
8  begin
9      out1 <= (a and b);
10 end Behavioral;
```

Gambar 3. Contoh *behaviour* untuk bahasa pemrograman VHDL

Adapun contoh sederhana algoritma yang ditulis dalam VHDL dapat dilihat pada Gambar 4. Berdasarkan pada Gambar 4 yang menjelaskan logika pemrograman dengan menggunakan bahasa VHDL. Logika dasar yang digunakan pada implementasi pengendali ON-OFF yaitu dengan menggunakan logika AND GATE dimana *input* didapat dari *push button* dan *output* terhubung dengan *driver relay* untuk menggerakkan motor DC ketika input dari kedua *push button* yang bernilai *logic* “1”. Hasil pengukuran menunjukkan *logic* “1” dengan tegangan sebesar 4,5 V.

D. Perancangan Sistem

Standar desain FPGA menurut [12], terdiri dari empat tahap, yaitu *design entry*, *logic synthesis*, implementasi desain, dan konfigurasi perangkat. Perangkat yang dibutuhkan untuk mendesain FPGA direpresentasikan oleh HDL seperti yang digunakan dalam penelitian ini adalah *Verilog* HDL. Hal tersebut membuat tahap *design entry* pasti melibatkan pembuatan / perancangan *Verilog* HDL (VHDL). Penelitian ini bertujuan untuk membuat modul pembelajaran berbasis FPGA menggunakan bahasa *Verilog* HDL.

Adapun tahap dalam perancangan sistem kendali ON-OFF adalah pertama mengidentifikasi variabel yang akan dikontrol yaitu kecepatan motor DC dalam hal ini direpresentasikan oleh PWM. Selanjutnya menuliskan spesifikasi yang diinginkan adalah terdapat dua mode kecepatan putaran motor DC yaitu kecepatan normal dan cepat. Kemudian tahap selanjutnya adalah mengkonfigurasi sistem yang dibangun dalam struktur *lup* terbuka seperti tertera pada Gambar 5, selanjutnya adalah membangun kontroler ON-OFF dan ON-OFF dengan waktu tunda. Tahap terakhir adalah analisa performansi sistem.

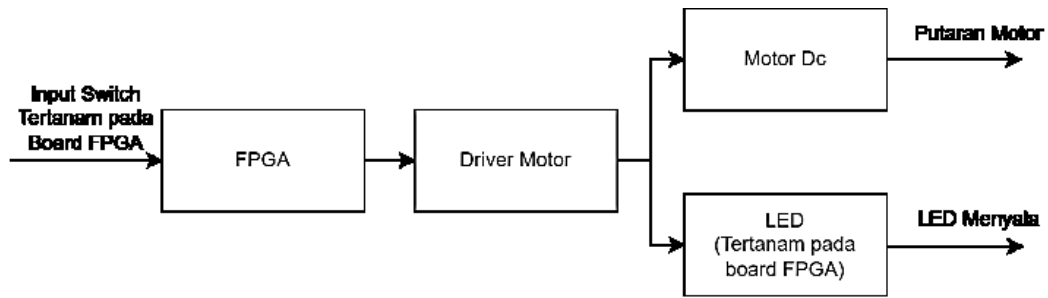
```

-- (this is a VHDL comment)
/*
   this is a block comment (VHDL-2008)
*/
-- import std_logic from the IEEE library
library IEEE;
use IEEE.std_logic_1164.all;

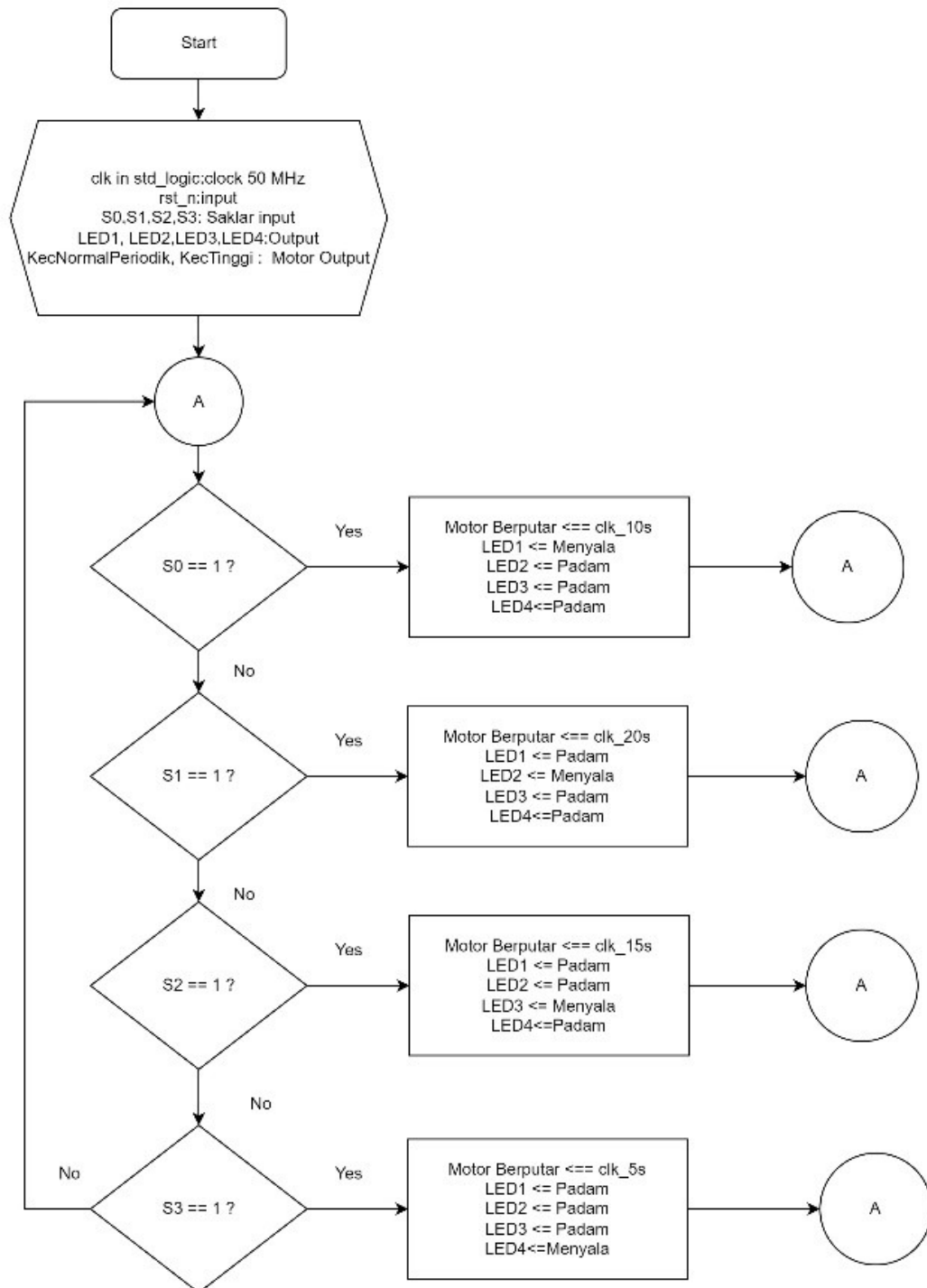
-- this is the entity
entity ANDGATE is
    port (
        I1 : in std_logic;
        I2 : in std_logic;
        O  : out std_logic);
end entity ANDGATE;

-- this is the architecture
architecture RTL of ANDGATE is
begin
    O <= I1 and I2;
end architecture RTL;
```

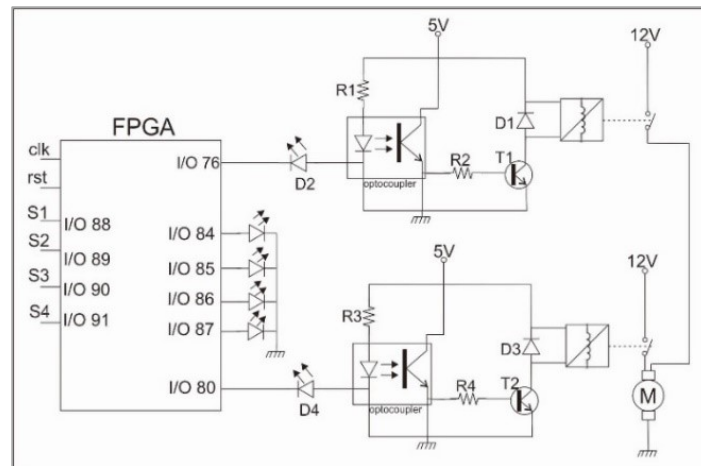
Gambar 4. Contoh sederhana pemrograman dalam VHDL



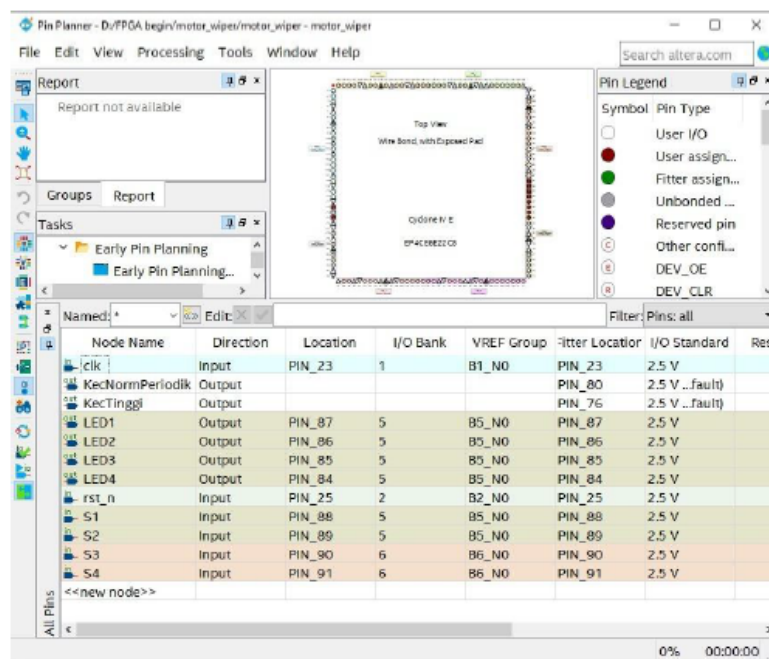
Gambar 5. Diagram blok pengendali ON-OFF motor



Gambar 6. Diagram alir VHDL untuk kendali ON-OFF dengan logika *counter* pada motor DC



Gambar 7. Rangkaian skematik sistem



Gambar 8. Konfigurasi *pin planner* pengendali ON-OFF motor DC pada Quartus

Tahap pertama agar dapat merancang VHDL diawali dengan menyatakan prinsip kerja sistem terlebih dahulu. Sistem yang dirancang terdiri dari 4 buah saklar (S1–S4), 4 buah LED, dan motor DC sebagai aktuator. Diagram blok kendali dari sistem ini menggunakan struktur *lup* terbuka seperti yang dapat dilihat pada Gambar 5.

Diagram alir dapat dilihat pada Gambar 6 dan skematik rangkaian dapat dilihat pada Gambar 7. Ketika saklar S1 ditekan maka LED1 ON dan kecepatan motor normal dan periodik, sedangkan ketika saklar S2 ditekan maka, LED2 dan *KecNormal*, merepresentasikan kecepatan normal pada motor, sedangkan ketika saklar S3 LED3 dan *KecTinggi* representasi kecepatan tinggi pada motor. Terakhir ketika saklar S4 ditekan maka, LED4 representasi untuk STOP/OFF semua kondisi

motor. Kendali akan terus melakukan penyeleksian penekanan pada saklar dan berulang. Frekuensi yang diperlukan adalah sebesar 50 MHz.

III. HASIL DAN PEMBAHASAN

Pengujian sistem dilakukan mejadi dua tahap yaitu, tahap simulasi dan tahap implementasi. Pada tahap simulasi, algoritma kendali ON-OFF untuk motor DC ditulis dalam deskripsi VHDL dan *pin planner* untuk penetapan setiap lokasi pin FPGA yang akan digunakan kemudian. Simulasi dijalankan dan dilihat responnya pada diagram waktu. Simulasi dilakukan menggunakan perangkat lunak Quartus. *Software* ini digunakan untuk membuat simulasi rangkaian logika secara digital dengan memanfaatkan bahasa deskripsi yaitu

VHDL. Pemilihan Quartus dikarenakan perangkat lunak ini karena jenis FPGA yang digunakan yaitu Altera Cyclone IV.

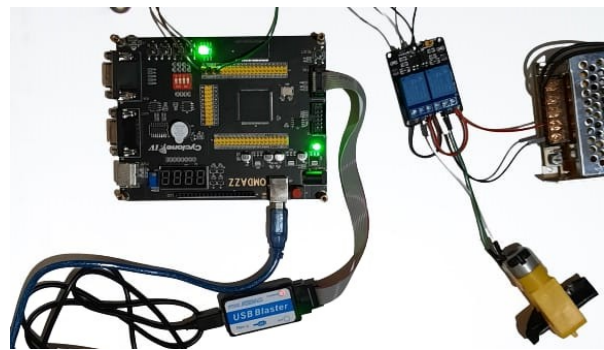
Perencanaan *pin planer* pada Quartus dilakukan untuk mengunduh program VHDL ke dalam *board* FPGA, dimana *pin planner* yang dibuat dapat dilihat pada Gambar 8. Penggunaan pin disesuaikan dengan *datasheet board* FPGA yang digunakan. Pada penelitian ini menggunakan *board* FPGA Altera. *Output* sistem terdiri dari 4 buah LED dan kecepatan motor. Untuk empat buah LED terhubung dengan pin 84 - pin 87, dimana LED tertanam *board* FPGA dan untuk *driver* motor terhubung dengan pin 77 dan 78, sedangkan untuk *input* berasal dari *push button* yang terhubung ke pin 88-91 pada *board* FPGA dan terakhir pin reset yang terhubung dengan pin 25.

Hasil realisasi desain kendali ON-OFF menggunakan *board* FPGA dapat dilihat pada Gambar 9. *Input* berasal dari *push button*, lalu *driver* berupa modul relai 5V yang terhubung dengan motor DC. Perlu diperhatikan bahwa sebelum *upload* program ke *board* FPGA, harus menghubungkan kabel USB JTAG dan kabel *power* ke *board* FPGA terlebih dahulu. Ketika program sudah di-*upload* maka program kendali ON-OFF sudah bisa dijalankan.

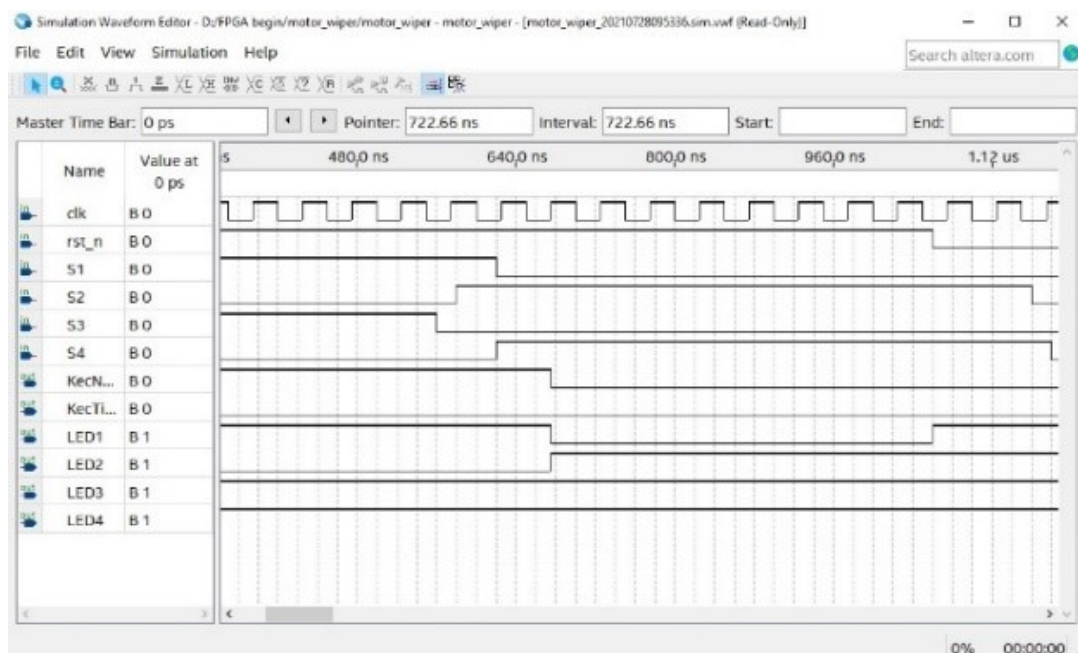
Perangkat lunak Quartus juga dapat menunjukkan *behaviour* dari desain algoritma yang dirancang. Sebelum desain dijalankan perlu terlebih membuat *waveform* yang diinginkan biasa disebut

dengan *test vectors*. Hal ini untuk merepresentasikan *input*. Saat dijalankan, FPGA akan menggunakan *test vector* ini ke dalam model algoritma yang didesain kemudian menyatakan respon (dalam hal ini adalah kondisi kecepatan motor, tergantung pada *input pushbutton* yang diberikan).

Adapun respon diagram waktu dari algoritma kendali ON-OFF ditunjukkan pada Gambar 10. *Pushbutton* adalah *input*-nya, dan respon motor DC direpresentasikan dengan LED juga. Jika *input* memiliki *logic* “1” maka *output* akan berlogika “1” (kondisi ON) dengan kecepatan normal, begitu pula sebaliknya. Waktu LED dan motor DC berlogika “1” tergantung pada seberapa lama *user* menekan kembali *push button* pada *board* FPGA.



Gambar 9. Implementasi pengendali ON-OFF motor DC menggunakan FPGA

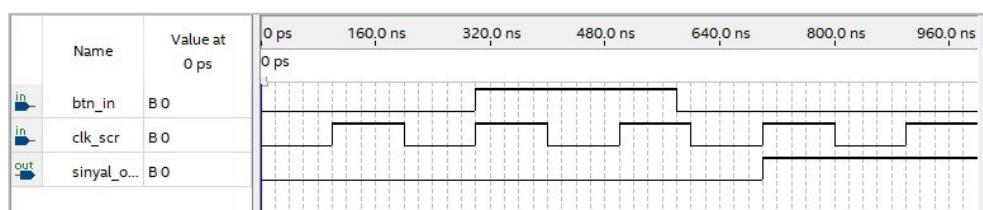


Gambar 10. Respon diagram waktu dari *software* Quartus

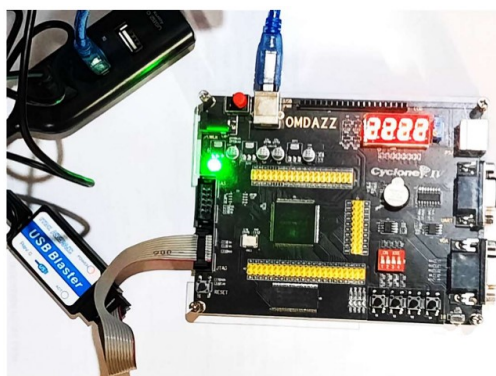
```

top_kendali.vhd
11 end top_kendali;
12
13 architecture behavioral of top_kendali is
14 -- Sinyal u/ debounce
15 signal delay1, delay2, delay3: std_logic;
16 signal bnc: std_logic;
17 -- Sinyal u/ PWM
18 signal btn: std_logic:= '0';
19 signal s_on: std_logic:= '0';
20 signal pwm_on: std_logic:= '0';
21 signal cnt: std_logic_vector(27 downto 0):= (others=>'0');
22 signal cnt_pwm: std_logic_vector(3 downto 0):= (others=>'0');
23
24 begin
25   sinyal_out<='1' when cnt_pwm<=8 and pwm_on='1' else '0';
26   bnc<=btn_in;
27   --Debounce
28   process(clk_scr)
29   begin
30     if(rising_edge(clk_scr))then
31       delay1<=bnc;
32       delay2<=delay1;
33       delay3<=delay2;
34     end if;
35   end process;
36   btn<=delay1 and delay2 and delay3;
37
38   --btn di-klik
39   process(btn,cnt)
40   begin
41     if(falling_edge(btn)) then
42       s_on<='1';
43       cnt<=cnt-1;
44     end if;
45   end process;
46 end architecture behavioral;

```

Gambar 11. VHDL untuk kendali ON-OFF dengan logika *counter* pada motor DCGambar 12. Diagram waktu untuk kendali ON-OFF motor DC dengan logika *counter*

Selanjutnya dilakukan pengujian kendali ON-OFF dengan *behaviour* yang lain, yaitu pada deskripsi VHDL ditambahkan logika *counter*. Penambahan logika *counter* ini bertujuan untuk membuat sistem dapat berubah kondisi dari ON ke OFF secara otomatis, sesuai dengan waktu tunda yang diinginkan. Cuplikan deskripsi VHDL dapat dilihat pada Gambar 11, dimana logika *counter* dideskripsikan dengan kode “cnt” yang menghitung mundur dari selama 5 detik. Perencanaan konfigurasi pin yang akan digunakan (*pin planner*) dapat dilihat pada Gambar 8. Sedangkan pada Gambar 12 menunjukkan respon diagram waktu, dan Gambar 13 menunjukkan cuplikan hasil pengujian kendali on-off dengan FPGA dan dapat dilihat bahwa LED menyala.

Gambar 13. Cuplikan hasil kendali ON-OFF motor DC dengan logika *counter*

Hasil perancangan kendali ON-OFF dengan menggunakan logika *counter* adalah ketika input *push button* berlogika “1” dan *clock* berlogika “1” maka *output* akan berlogika “1” setelah sinkronisasi dengan *clk logic* “1”. *Counter* digunakan untuk waktu tunda (*delay*) saat *output* berlogika “1”, dimana respon diagram waktu dapat dilihat pada Gambar 12.

Perancangan kendali ON-OFF pada VHDL dapat dilakukan dengan berbagai pendekatan. Dalam penelitian ini menggunakan logika *counter* dan ON-OFF murni. Pada kendali ON-OFF murni, dibutuhkan *logic input* sebagai penentu kondisi ON dan OFF. Dalam hal ini *logic input* direpresentasikan oleh *push button* yang tertanam pada *board* FPGA. Sedangkan untuk kendali ON-OFF menggunakan logika *counter*, dimana logika *counter* berfungsi untuk waktu tunda (*delay*) di atur selama 5 detik menghasilkan jeda waktu antara kondisi ON dan kondisi OFF dari motor, dan pengguna hanya perlu menekan *push button* satu kali untuk kondisi ON saja. Jika tidak diberi *counter* maka *output* ketika diberi *logic* “1” akan terus berada pada kondisi ON dan periode waktu OFF ditentukan oleh *input* ketika diberi *logic* “1” kembali atau saat pengguna menekan kembali *push button* untuk mematikan putaran motor DC.

IV. KESIMPULAN

Pada penelitian ini modul simulator kendali ON-OFF menggunakan FPGA telah direalisasikan. *Board* FPGA yang digunakan adalah Cyclone IV EP4CE6E22C8N dan VHDL dibuat menggunakan *software* Quartus. Desain kendali ON-OFF dibangun berdasarkan dua jenis logika yaitu ON-OFF murni dan ON-OFF menggunakan logika *counter*. Hasil implementasi pada putaran motor DC menunjukkan bahwa respon sistem dapat mengikuti kondisi ON-OFF yang ditentukan dengan baik. Adapun untuk kondisi ON-OFF murni, maka motor DC akan berputar jika *push button* ditekan dan akan OFF jika *push button* ditekan kembali. Sedangkan logika *counter* berfungsi untuk waktu tunda (*delay*) diatur selama 5 detik pada kondisi ON-OFF dari motor DC, sehingga pengguna tidak perlu menekan *push button* untuk mematikan putaran motor DC. Penelitian selanjutnya dapat menerapkan metode kendali yang lebih lanjut untuk mengendalikan *plant* yang lebih kompleks.

UCAPAN TERIMA KASIH

Ucapan terima kasih kami sampaikan kepada Pusat Penelitian dan Pengabdian Masyarakat Politeknik Negeri Bandung yang telah mendukung penelitian ini dalam bentuk penelitian peningkatan kapasitas Laboratorium Prodi Teknik Elektronika.

REFERENSI

- [1] G. McFarland, *Microprocessor Design: A Practical Guide from Design Planning to Manufacturing*, The McGraw-Hill Publishing Companies, Inc., 2006.
- [2] R. M. Pindoriya, A. K. Mishra, B. S. Rajpurohit and R. Kumar, "FPGA Based Digital Control Technique for BLDC Motor Drive," in *2018 IEEE Power & Energy Society General Meeting (PESGM)*, Portland, OR, USA, 2018.
- [3] A. E. Putra and A. M. Fajri, "Purwarupa Mikroprosesor Berbasis FPGA Altera EPF10K10 dengan Deskripsi VHDL," in *Seminar Nasional Informatika 2008*, Indonesia, 2008.
- [4] Z. Zhang, Q. Yu, L. Njilla, and C. Kamhoua, "FPGA-oriented moving target defense against security threats from malicious FPGA tools," in *2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST)*, Washington, DC, USA, 2018.
- [5] U. FAROOQ, I. BAIGI and B. A. ALZHRANI, "An Efficient Inter-FPGA Routing Exploration Environment for Multi-FPGA Systems," *IEEE Access*, vol. 6, pp. 56301 - 56310, 2018.
- [6] M. Hanhila, T. Mantere, and J. T. Alander, "FPGA-implementation of PID-controller by differential evolution optimization," *Open Engineering*, vol. 8, no. 1, pp. 395-402, 2018.
- [7] Y. Wang and U. Schaefer, "Real time simulation of a FPGA based space vector PWM controller," *SPEEDAM*, pp. 833-838, 2010.
- [8] D. F. Sari, M. Rivai, T. Mujiono and Tasripan, "Implementasi Teknologi Field Programmable Gate Array (FPGA) pada Alat Identifikasi ODOR," in *Seminar Nasional Informatika 2010*, Yogyakarta, Indonesia, 2010.
- [9] J. P. Deschamps, G. Sutter, and C. Enrique, *Guide to FPGA Implementation of Arithmetic Functions*, Springer Netherlands, 2012.
- [10] Intel, "Product Specifications - Cyclone® IV EP4CE6 FPGA," Intel, [Online]. Available: <https://ark.intel.com/content/www/us/en/ark/products/210472/cyclone-iv-ep4ce6-fpga.html>. [Accessed 24 Sept 2021].
- [11] D. Pellerin, "An Introduction to VHDL for Synthesis and Simulation," Accolade Design Automation, Inc., [Online]. Available: http://www.uco.es/~ffl/mumuj/h_intro.htm. [Accessed 24 Sept 2021].
- [12] G. Jagadeesh, S. K. Lam, and T. Srikanthan, "A Short Course on Implementing FPGA Based Digital Systems," in *2008 14th IEEE International Conference on Parallel and Distributed Systems*, Melbourne, Victoria, Australia, 2008.