PERANCANGAN *TRAFFIC LIGHT* MENGGUNAKAN MODUL *FIELD PROGRAMMABLE GATE ARRAY* XILINX NEXYS 3

Fatma Pandu Damayanti[*)], Munawar Agus Riyadi, dan Trias Andromeda

Program Studi Sarjana Departemen Teknik Elektro, Universitas Diponegoro
Jl. Prof. Sudharto, SH, Kampus UNDIP Tembalang, Semarang 50275, Indonesia

[*)]*E-mail: fatma.pandu@gmail.com*

Abstrak

Fungsi traffic light digunakan untuk mengatur perpindahan kendaraan di persimpangan jalan agar tidak terjadi kemacetan. Pada umumnya traffic light di Indonesia memiliki waktu yang tetap tanpa memperhatikan kemacetan. Pada penelitian ini, dikembangkan traffic light dengan masukan sensor limit switch. Sensor limit switch dipasang pada jarak tertentu di setiap lajur untuk mendeteksi antrian kendaraan yang padat. Masukan pada sensor limit switch diproses pada modul FPGA Nexys 3. Perancangan traffic light menghasilkan keluaran berupa LED dan seven segment yang bekerja dengan frekuensi 1 Hz. Seven segment yang digunakan terdiri dari 4 digit untuk menunjukkan lajur utara, selatan, barat dan timur. Setiap digit pada seven segment mengalami pergantian dengan menggunakan frekuensi 200 Hz. Masukan limit switch mempengaruhi lamanya waktu untuk LED warna hijau menyala. Jika nilai limit switch = 1, maka LED hijau pada lajur tertentu menyala selama 7 detik. Jika nilai limit switch = 0, maka LED hijau pada lajur tertentu menyala selama 5 detik. Hasil pengujian pada penelitian ini yaitu terjadi perpanjangan waktu jeda sebesar dua detik untuk LED hijau menyala jika nilai limit switch = 1. Masukan nilai limit switch hanya berpengaruh saat LED hijau menyala untuk lajur masing-masing.

Kata kunci: Traffic Light, FPGA, Xilinx, Nexys 3.

Abstract

Traffic light is used to set transferring of the vehicle at the road junction to avoid congestion. Traffic light in Indonesia has fixed time without regard to congestion. This research, advanced traffic light with limit switch sensor. The limit switch sensors were installed at each strip to detect heavy vehicles queue. The limit switch will be processed on FPGA Xilinx Nexys 3. Traffic light generates output, there are seven segments and LED used 1 Hz frequency. Seven segments used 4 digits to indicate lane North, South, East and West. Each digit on seven segments shown the turnover used 200 Hz frequency. Limit switch affect outputs on the length of time Green LED lights up. If the value of the limit switch = 1, the Green LED on certain lanes lights up for 7 seconds. If the value of the limit switch = 0, the Green LED on certain lanes light up for 5 seconds. The results on the research is take two seconds of extra time for Green LED lights up if the value of the limit switch = 1. The value of the limit switches only take effect when the Green LED is lit for each lane.

Keywords: Traffic Light, FPGA, Xilinx, Nexys 3.

1. Pendahuluan

Seiring perkembangan teknologi dalam dunia otomotif, maka layanan transportasi yang ditawarkan di Indonesia semakin meningkat. Populasi penduduk di Indonesia cukup tinggi. Dampaknya adalah bertambahnya jumlah transportasi, sedangkan fasilitas jalan terbatas. Hal ini menyebabkan kemacetan di beberapa tempat. Solusi pertama yaitu dengan adanya transportasi umum, masyarakat diharapkan menggunakannya secara optimal. Namun hanya sedikit penduduk yang menggunakan transportasi umum. Solusi kedua yaitu pembangunan *traffic light*. *Traffic light* berperan penting untuk menertibkan lalu lintas. Pengguna jalan dapat merasakan manfaat dari teknologi berupa sistem *traffic light* yang mengatur pergiliran kendaraan yang melalui persimpangan jalan. Sistem *traffic light* mengalami perkembangan dari desain dan teknologi, di antaranya menggunakan logika fuzzy[1], PLC[2], pengolahan citra digital[3], dan FPGA (*Field Programmable Gate Array*). Selain *traffic light*, FPGA sudah digunakan untuk beberapa penelitian sebelumnya. Penelitian tersebut yaitu perancangan *wearable controller*[4], perancangan implementasi *sha-1 password cracking*[5], perancangan sistem instrumentasi medis pengukur tiga tanda vital tubuh[6], *reconfigurable logic embedded architecture of support vector machine linear kernel*[7] dan *incremental high throughput network*

traffic classifier[8]. FPGA digunakan untuk penelitian sistem *embedded* pada *support vector machine* (SVM) *linear kernel*. Pada penelitian yang dilakukan Trias Andromeda dan tim [7], perancangan SVM dengan parameter linear kernel menggunakan FPGA jenis Altera Cyclone IV. Perancangan SVM ini menganalisis *number of features* dan *support vector* terhadap kinerja SVM *hardware*. *Number of features* menentukan frekuensi yang dapat beroperasi secara maksimal dan penggunaan jumlah *logic resource*. *Number of support vector* menentukan penggunaan jumlah *memory* di dalam *chip* yang digunakan selama sistem bekerja[7]. FPGA juga digunakan untuk perkembangan dalam bidang jaringan. FPGA dapat merealisasikan pengklarifikasi *network traffic* secara *online* dengan berdasarkan klasifikasi *incremental semi-supervised*. Dengan menggunakan FPGA, kinerja klasifikasi secara *online* dapat mencapai 1 Gbps *bitrate* tanpa adanya *dropping flow*[8].

Terdapat perancangan *traffic light* menggunakan FPGA menggunakan instumen tambahan, seperti perancangan *traffic light* menggunakan sensor *network* yang bisa menerima data tentang kecepatan kendaraan melalui wireless[9]. Selain itu perancangan *traffic light* menggunakan sensor *infrared*. Sensor ini berfungsi untuk mendeteksi adanya kendaraan darurat seperti pemadam kebakaran dan *ambulance* untuk didahulukan[10]. Perancangan *traffic light* ini menggunakan modul FPGA Xilinx Nexys 3 dan sensor *limit switch*. Sensor *limit switch* dipasang pada jarak tertentu. Sensor ini bekerja berdasarkan tekanan kendaraan yang berhenti pada batas jalan tertentu. Masukan sensor *limit switch* diperlukan untuk peningkatan pengaturan sistem *traffic light* yang baik. Pada perancangan *traffic light* oleh Shi Shuo dan tim [11], terdapat kontroller *traffic light* berupa *counter down* yang ditampilkan pada LED. *Counter down* ini bermaksud memberikan keakuratan waktu, sehingga di dalam penerapannya para pengendara lebih memperhatikan dan membatasi terjadinya kecelakaan. Pada penelitian ini, kontroller *traffic light* berupa *counter down* ditampilkan dalam *seven segment*.

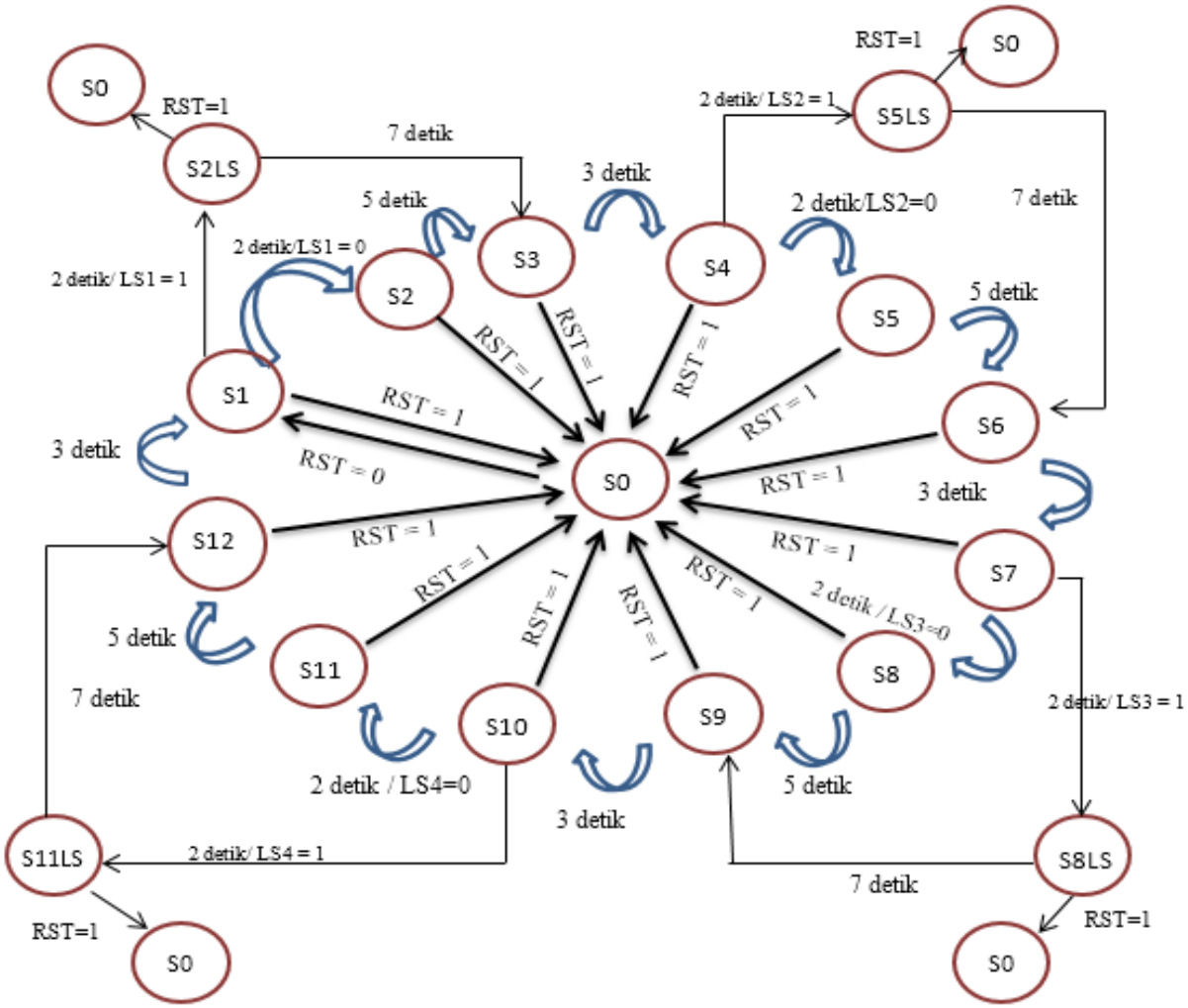
2. Metode

Perancangan *traffic light* pada penelitian ini menggunakan empat persimpangan jalan. Pada setiap persimpangan diberi 1 buah *limit switch* dan 3 buah LED berwarna merah, kuning dan hijau dengan jeda waktu tertentu. Jeda waktu berbeda jika sensor *limit switch* menyala. Jika sensor menyala, maka led hijau menyala lebih lama dengan waktu tertentu. *Traffic light* di setiap persimpangan akan disajikan ke dalam *truth table* pada **Tabel 1**. *Truth table* ini berfungsi untuk mempermudah langkah selanjutnya, yaitu pembuatan *state* FSM. Berikut ini *truth table* yang dirancang.

Tabel 1. *Truth table* perancangan *traffic light*

State	Time	Input		Traffic Light Unit			
		Reset	LS	Utara	Barat	Selatan	Timur
S0	3 s	1	xxxx	Red	Red	Red	Red
S1	2 s	0	xxxx	Red – Yellow	Red	Red	Red
S2	5 s	0	xxx0	Green	Red	Red	Red
S2LS	7 s	0	xxx1	Green	Red	Red	Red
S3	3 s	0	xxxx	Yellow	Red	Red	Red
S4	2 s	0	xxxx	Red	Red – Yellow	Red	Red
S5	5 s	0	xxx0	Red	Green	Red	Red
S5LS	7 s	0	xxx1	Red	Green	Red	Red
S6	3 s	0	xxxx	Red	Yellow	Red	Red
S7	2 s	0	xxxx	Red	Red	Red – Yellow	Red
S8	5 s	0	xxx0	Red	Red	Green	Red
S8LS	7 s	0	xxx1	Red	Red	Green	Red
S9	3 s	0	xxxx	Red	Red	Yellow	Red
S10	2 s	0	xxxx	Red	Red	Red	Red – Yellow
S11	5 s	0	xxx0	Red	Red	Red	Green
S11LS	7 s	0	xxx1	Red	Red	Red	Green
S12	3 s	0	xxxx	Red	Red	Red	Yellow

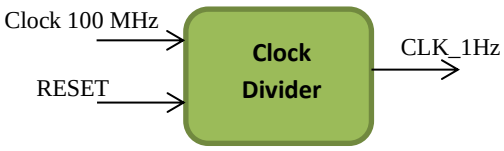
Pada **Tabel 1** terdapat tanda "x" yaitu *don't care*. *Don't care* adalah posisi dimana keadaan *input* tidak dihiraukan. LS merupakan *limit switch*. Setelah dibuatnya *truth table*, maka bisa dibuatkan *finite state machine*. *Input* yang digunakan yaitu *clock* dari modul FPGA Xilinx Nexys 3, dan *limit switch*. *Output* pada *state* yang dimaksud adalah LED, *seven segment* dan waktu jeda yang ditentukan. Keadaan *traffic light* ini terus berputar dari *state* 0 hingga *state* 12 dan kembali lagi ke *state* 1. Jika terdapat masukan *limit switch* dari salah satu persimpangan, maka LS menjadi '1' dan waktu jeda lampu hijau lebih lama. Jika jalanan kosong, maka semua *limit switch* LS = '0' dan tidak terdapat perubahan jeda waktu. Jika *reset* = 1, maka *state* kembali menuju *state* awal yaitu S0, kemudian berlanjut ke S1.



Gambar 1. Finite State Machine pada Perancangan Traffic Light

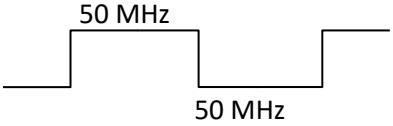
2.1. Perancangan *Clock Divider*

Perancangan *clock divider* ditujukan untuk pembuatan *clock* baru yaitu *clock* sebesar 1 Hz. Langkah pertama dalam perancangan *clock divider* yaitu menentukan diagram blok seperti pada **Gambar 2**. Diagram blok ini berupa masukan dan keluaran yang diperlukan dalam sistem *traffic light.* Masukan pada diagram blok yaitu berupa *clock* dalam modul FPGA sebesar 100 MHZ dan *reset*. *Reset* merupakan masukan yang berfungsi untuk mengubah *current state* menjadi *state* awal yang sudah ditentukan. Keluaran pada diagram blok yaitu *clock* sebesar 1 Hz. *Clock* 1 Hz berfungsi sebagai *clock* yang mengeksekusi proses *counter* dan setiap *state* yang ditampilkan pada Tabel 1.



Gambar 2. Diagram Blok *Clock Divider*

Clock pada modul dirancang dengan *duty cycle* 50 %. Dengan perbandingan *duty cycle* tersebut, diperoleh sinyal *high* dan *low* seperti pada **Gambar 3**. Perbandingan ini digunakan untuk menentukan *prescaler* pada perancangan.



Gambar 3. *Signal Clock* pada Modul FPGA Xilinx Nexys 3 dengan *Duty Cycle* 50%.

Clock divider bisa disebut dengan *prescaler* karena memiliki fungsi yang sama. Pada perancangan *clock divider*, sinyal *high* dan *low* memiliki *input frequency* sebesar 50 Mhz dan *output frequency* 1 Hz. Jumlah *prescaler* yang dirancang dapat ditentukan menggunakan **persamaan 1**.

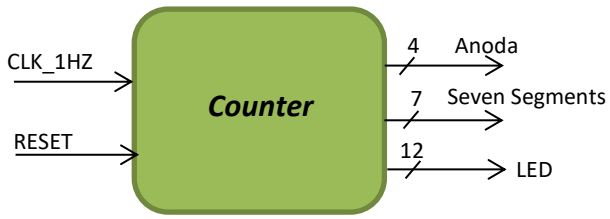
$$Prescaler = \frac{Input\ Frequency}{Output\ Frequency} \quad (1)$$

$$Prescaler = \frac{50.000.000}{1}$$
Prescaler = 50.000.00 *cycles*
Prescaler = 5 x 10^7 *cycles*

Jika *prescaler* sudah mencapai 5 x 10^7, maka clock 1 Hz mengalami *rising edge* dan menjadi sinyal *high* atau bernilai 1. Pada keadaan inilah, *clock* 1 Hz digunakan untuk mengeksekusi *current state* dan masukan *limit switch* menjadi *next state* yang sudah ditentukan. Saat *clock* 1 Hz bernilai 1, maka *prescaler* kembali dalam hitungan 0. Saat *prescaler* mencapai 5 x 10^7 maka *clock* 1 Hz mengalami *falling edge* dan menjadi sinyal *low* atau bernilai 0.

2.2. Perancangan *Counter*

Perancangan *counter* pada vhdl menggunakan proses yang berbeda dengan perancangan *clock divider*. Namun proses *clock divider* berkesinambungan dengan proses *counter*. Hal ini dikarenakan keluaran satu proses menjadi masukan pada proses lainnya. Pada proses *clock divider* dihasilkan *output* yaitu *clock* sebesar 1 Hz (CLK_1HZ). Clock 1 Hz yaitu *input* pada proses *counter* yang berfungsi untuk mengatur jalannya data baik dalam perhitungan *counter* maupun pergeseran LED. Pada perancangan *counter* diperoleh diagram blok seperti pada **Gambar 4**. Masukan pada diagram blok yaitu CLK_1HZ dan *reset*. Keluaran pada diagram blok yaitu empat bit anoda, tujuh bit *seven segment* dan 12 bit *LED*.



Gambar 4. Diagram Blok Perancangan *Counter*.

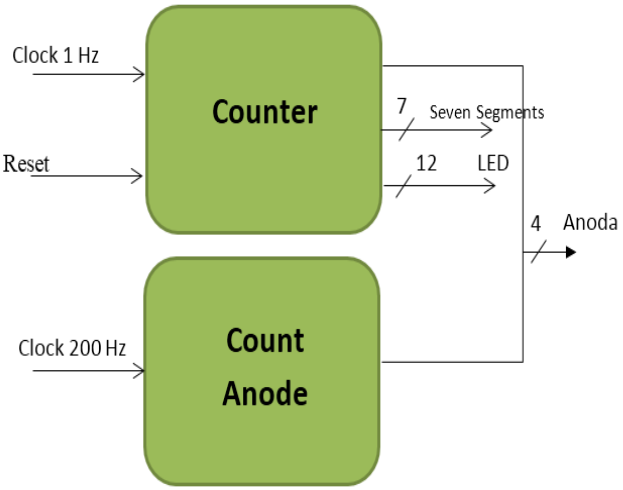
Counter yang dirancang menggunakan VHDL, dikonfigurasi dengan *seven segment* pada modul Xilinx Nexys 3. *Counter* hanya dihubungkan pada empat anoda *seven segment*. *Seven segment* yang digunakan yaitu ada pada modul FPGA Nexys 3.

2.2. Perancangan *Common Anode Seven Segment*

Pada perancangan *traffic light, time controller* ditunjukkan oleh *seven segment.* Pada modul FPGA Xilinx Nexys 3 terdapat empat digit *common anode seven segment*. Setiap digit *common anode* mewakili setiap lajur seperti pada **Gambar 5**. Prinsip kerja *common anode* ini dengan cara menyala bergantian dengan waktu jeda yang sangat cepat, sehingga seolah-olah semua *common anode* menyala secara bersamaan. Oleh karena itu, perancangan ini membutuhkan *clock* baru untuk mengeksekusi pergantian *common anode* pada *seven segment*. Untuk mendapatkan *clock* sebesar 200 Hz, maka *prescaler* dapat diperoleh menggunakan **persamaan 1**.

$$Prescaler = \frac{100.000.000}{200}$$
Prescaler = 500.000 *cycles*

Prescaler memiliki fungsi yang sama dengan *clock divider*. Fungsi *prescaler* yaitu membagi *clock* sumber menjadi *clock* baru. *Prescaler* pada perancangan vhdl dinamakan *signal counter*. *Prescaler* mengeksekusi *signal* count_anode. *Signal* count_anode menentukan *common anoda seven segment* yang aktif. *Common anode* pada modul FPGA jika diberi masukan '0' atau *active low*. Perancangan *common anode seven segment* bekerja berkesinambungan dengan proses perancangan yang lain yaitu perancangan *counter*. Hal ini dikarenakan *common anode* pada perancangan ini bekerja secara bersamaan dengan *common anode* pada proses perancangan *counter*.

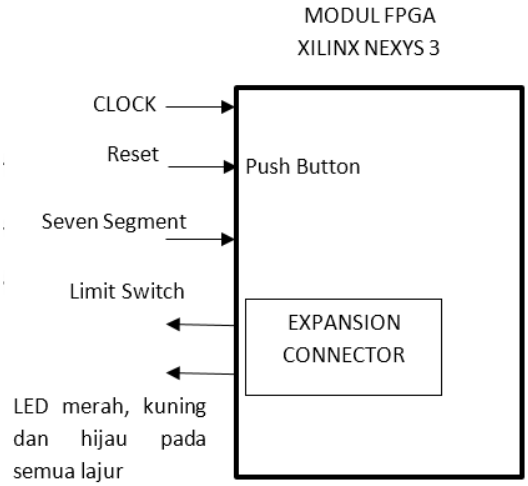


Gambar 5. Diagram Blok Perancangan *Common Anode Seven Segment.*

2.2. Perancangan *Hardware*

Hardware yang diperlukan dalam perancangan ini yaitu modul FPGA Xilinx Nexys 3 untuk membaca masukan sensor *limit switch*. Pada setiap komponen rangkaian diberi *current limiter* berupa resistor dengan nilai yang bervariasi. *Current limiter* ini berfungsi sebagai pembatas arus pada setiap komponen rangkaian agar tidak terjadi hubung singkat. (*short-circuit*). Sensor *limit switch* dihubungkan pada modul FPGA melalui *expansion connector*. *Reset* menggunakan salah satu *push button* yang terintegrasi pada modul FPGA. *Seven segment* yang digunakan terdapat dalam modul FPGA Nexys 3. Perancangan *clock divider* dan *common anode seven segment* menggunakan *clock* yang terdapat pada modul FPGA Xilinx Nexys 3 yaitu sebesar 100 MHz. Untuk keluaran LED warna merah, kuning dan hijau pada semua

lajur dihubungkan ke dalam modul FPGA menggunakan *jumper female-male* atau *male-male* melalui *expansion connector*. Perancangan *hardware* pada sistem *traffic light* ditampilkan pada **Gambar 6**.



Gambar 6. Perancangan *Hardware*.

3. Hasil dan Analisis

Perancangan *Traffic light* berhasil diimplementasikan ke dalam dengan menggunakan modul Xilinx Nexys 3. Hasil perancangan tedapat lima keadaan. Keadaan pertama yaitu ketika *limit switch* pada semua lajur bernilai 1, maka hasil perancangan *traffic light* tertera pada **Tabel 2**.

Tabel 2. Hasil *Traffic Light* Ketika *Limit Switch* Semua Lajur adalah 0.

State	Time	Input		Traffic Light Unit			
		Reset	LS	Utara	Barat	Selatan	Timur
S0	3 s	1	xxxx	Red	Red	Red	Red
S1	2 s	0	xxxx	Red – Yellow	Red	Red	Red
S2	5 s	0	xxx0	Green	Red	Red	Red
S3	3 s	0	xxxx	Yellow	Red	Red	Red
S4	2 s	0	xxxx	Red	Red – Yellow	Red	Red
S5	5 s	0	xxx0	Red	Green	Red	Red
S6	3 s	0	xxxx	Red	Yellow	Red	Red
S7	2 s	0	xxxx	Red	Red	Red – Yellow	Red
S8	5 s	0	xxx0	Red	Red	Green	Red
S9	3 s	0	xxxx	Red	Red	Yellow	Red
S10	2 s	0	xxxx	Red	Red	Red	Red – Yellow
S11	5 s	0	xxx0	Red	Red	Red	Green
S12	3 s	0	xxxx	Red	Red	Red	Yellow

Pada **Tabel 2** terlihat semua LED akan berjalan secara normal dan tidak ada perpanjangan jeda waktu pada LED hijau. Hal ini dibuktikan bahwa *state* 2, *state* 5, *state* 8, dan *state* 11 menghasilkan waktu jeda selama 5 detik untuk LED hijau menyala. Masukan *limit switch* dieksekusi pada saat memasuki *state* 2, *state* 5, *state* 8, dan *state* 11. Keadaan ini dapat digunakan ketika antrian kendaraan yang tidak padat. Keadaan ini juga dapat digunakan ketika lajur jalan tidak terdapat kendaraan yang melintas. Setelah semua *state* berjalan hingga *state* 12, maka *state* kembali ke keadaan *state* 1. Jika *reset* = 1, maka semua *state* yang sedang berjalan (*current state*) kembali menuju *state* 0. Hal ini sudah sesuai dengan tabel perancangan yaitu **Tabel 1**. Keadaan kedua yaitu ketika salah satu *limit switch* bernilai 1. Dalam keadaan ini, *limit switch* pada lajur utara jalan bernilai 1 dan *limit switch* pada lajur jalan lain bernilai 0, maka hasilnya akan tertera pada **Tabel 3**.

Tabel 3. Hasil *Traffic Light* Ketika *Limit Switch* Lajur Utara adalah 1.

State	Time	Input		Traffic Light Unit			
		Reset	LS	Utara	Barat	Selatan	Timur
S0	3 s	1	xxxx	Red	Red	Red	Red
S1	2 s	0	xxxx	Red – Yellow	Red	Red	Red
S2LS	7 s	0	xxx1	Green	Red	Red	Red
S3	3 s	0	xxxx	Yellow	Red	Red	Red
S4	2 s	0	xxxx	Red	Red – Yellow	Red	Red
S5	5 s	0	xxx0	Red	Green	Red	Red
S6	3 s	0	xxxx	Red	Yellow	Red	Red
S7	2 s	0	xxxx	Red	Red	Red – Yellow	Red
S8	5 s	0	xxx0	Red	Red	Green	Red
S9	3 s	0	xxxx	Red	Red	Yellow	Red
S10	2 s	0	xxxx	Red	Red	Red	Red – Yellow
S11	5 s	0	xxx0	Red	Red	Red	Green
S12	3 s	0	xxxx	Red	Red	Red	Yellow

Pada **Tabel 3** terlihat bahwa ada perbedaan waktu LED hijau untuk menyala pada *traffic light* lajur utara yaitu 7 detik. Ketika *limit switch* pada lajur utara bernilai 1, hasil hanya mempengaruhi waktu jeda LED hijau lajur utara. Hal ini dibuktikan pada lajur utara memberikan waktu selama 7 detik untuk LED hijau hidup. Pada lajur barat, selatan dan timur memberikan waktu selama 5 detik untuk LED hijau hidup. *Limit switch* lajur utara tidak berpengaruh pada lajur barat, timur dan selatan. Masukan *limit switch* dieksekusi pada saat memasuki *state* 2 *limit switch*, *state* 5, *state* 8, dan *state* 11. Setelah semua *state* berjalan hingga *state* 12, maka *state* kembali ke keadaan *state* 1. Jika *reset* = 1, maka semua *state* yang sedang berjalan (*current state*) kembali menuju *state* 0. Hal ini menunjukkan bahwa hasil pengujian sudah sesuai dengan tabel perancangan yaitu **Tabel 1**. Keadaan ketiga yaitu ketika dua *limit switch* bernilai 1 dan dua *limit switch* lainnya bernilai 0. Dalam keadaan ini, *limit switch* pada lajur barat dan selatan bernilai 1 dan pada lajur lainnya bernilai 0, maka hasilnya tertera pada **Tabel 4**.

Tabel 4. Hasil *Traffic Light* Ketika *Limit Switch* Lajur Barat dan Selatan adalah 1.

State	Time	Input		Traffic Light Unit			
		Reset	LS	Utara	Barat	Selatan	Timur
S0	3 s	1	xxxx	Red	Red	Red	Red
S1	2 s	0	xxxx	Red – Yellow	Red	Red	Red
S2	5 s	0	xxx0	Green	Red	Red	Red
S3	3 s	0	xxxx	Yellow	Red	Red	Red
S4	2 s	0	xxxx	Red	Red – Yellow	Red	Red
S5	5 s	0	xxx0	Red	Green	Red	Red
S5LS	7 s	0	xxx1	Red	Green	Red	Red
S6	3 s	0	xxxx	Red	Yellow	Red	Red
S7	2 s	0	xxxx	Red	Red	Red – Yellow	Red
S8	5 s	0	xxx0	Red	Red	Green	Red
S8LS	7 s	0	xxx1	Red	Red	Green	Red
S9	3 s	0	xxxx	Red	Red	Yellow	Red
S10	2 s	0	xxxx	Red	Red	Red	Red – Yellow
S11	5 s	0	xxx0	Red	Red	Red	Green
S12	3 s	0	xxxx	Red	Red	Red	Yellow

Pada **Tabel 4** terlihat bahwa ada perbedaan waktu LED hijau untuk menyala pada *traffic light* lajur barat dan selatan yaitu 7 detik. Pada lajur utara dan timur memberikan waktu selama 5 detik untuk LED hijau hidup. Ketika *limit switch* pada lajur barat dan selatan bernilai 1, hasil hanya mempengaruhi waktu jeda lampu hijau lajur barat dan selatan. *Limit switch* lajur barat dan selatan tidak berpengaruh pada lajur jalan utara dan timur. Keadaan keempat yaitu ketika tiga *limit switch* bernilai 1 dan satu *limit switch* lainnya bernilai 0. Dalam keadaan ini, *limit switch* pada lajur utara bernilai 0 dan *limit switch* pada lajur jalan lain bernilai 1, maka hasilnya akan tertera pada **Tabel 5.**

Tabel 5. Hasil *Traffic Light* Ketika *Limit Switch* Lajur Utara adalah 0.

State	Time	Input		Traffic Light Unit			
		Reset	LS	Utara	Barat	Selatan	Timur
S0	3 s	1	xxxx	Red	Red	Red	Red
S1	2 s	0	xxxx	Red – Yellow	Red	Red	Red
S2	5 s	0	xxx0	Green	Red	Red	Red
S3	3 s	0	xxxx	Yellow	Red	Red	Red
S4	2 s	0	xxxx	Red	Red – Yellow	Red	Red
S5LS	7 s	0	xxx1	Red	Green	Red	Red
S6	3 s	0	xxxx	Red	Yellow	Red	Red
S7	2 s	0	xxxx	Red	Red	Red – Yellow	Red
S8LS	7 s	0	xxx1	Red	Red	Green	Red
S9	3 s	0	xxxx	Red	Red	Yellow	Red
S10	2 s	0	xxxx	Red	Red	Red	Red – Yellow
S11LS	7 s	0	xxx1	Red	Red	Red	Green
S12	3 s	0	xxxx	Red	Red	Red	Yellow

Pada **Tabel 5** terlihat bahwa lajur utara memberikan waktu yaitu 5 detik untuk LED hijau hidup. Untuk lajur barat, selatan dan timur memberikan waktu 7 detik untuk LED hijau hidup. Ketika *limit switch* lajur barat, selatan dan timur adalah 0, maka tidak mempengaruhi waktu jeda LED hijau hidup pada lajur utara. Masukan *limit switch* dieksekusi pada saat memasuki *state* 2, *state* 5 *limit switch*, *state* 8 *limit switch*, dan *state* 11 *limit switch*. Setelah semua *state* berjalan hingga *state* 12, maka *state* kembali ke keadaan *state* 1. Jika *reset* = 1, maka semua *state* yang sedang berjalan (*current state*) kembali menuju *state* 0. Hal ini membuktikan bahwa hasil pengujian sudah sesuai dengan tabel perancangan yaitu **Tabel 1**.

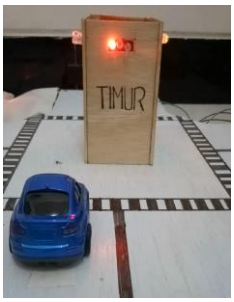
Keadaan kelima yaitu ketika *limit switch* pada semua lajur bernilai 1, maka hasilnya tertera pada **Tabel 6.** Pada **Tabel 6** terlihat bahwa ada perbedaan waktu LED hijau untuk menyala pada *traffic light* semua lajur jalan yaitu 7 detik. Kelima keadaan sudah membuktikan bahwa sesuai dengan tabel perancangan yaitu **Tabel 1**.

Tabel 6. Hasil *Traffic Light* Ketika *Limit Switch* Semua Lajur adalah 1.

State	Time	Input		Traffic Light Unit			
		Reset	LS	Utara	Barat	Selatan	Timur
S0	3 s	1	xxxx	Red	Red	Red	Red
S1	2 s	0	xxxx	Red – Yellow	Red	Red	Red
S2LS	7 s	0	xxx1	Green	Red	Red	Red
S3	3 s	0	xxxx	Yellow	Red	Red	Red
S4	2 s	0	xxxx	Red	Red – Yellow	Red	Red
S5LS	7 s	0	xxx1	Red	Green	Red	Red
S6	3 s	0	xxxx	Red	Yellow	Red	Red
S7	2 s	0	xxxx	Red	Red	Red – Yellow	Red
S8LS	7 s	0	xxx1	Red	Red	Green	Red
S9	3 s	0	xxxx	Red	Red	Yellow	Red
S10	2 s	0	xxxx	Red	Red	Red	Red – Yellow
S11LS	7 s	0	xxx1	Red	Red	Red	Green
S12	3 s	0	xxxx	Red	Red	Red	Yellow

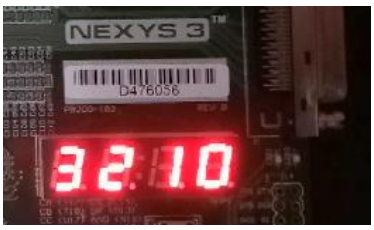
Kelima keadaan juga sudah sesuai dengan FSM pada **Gambar 1**. Dari perancangan FSM *traffic light*, keadaan *limit switch* = 0 adalah saat lajur jalan tidak terdapat kendaraan yang melintas atau kendaraan yang tidak memiliki antrian padat. Keadaan tersebut ditunjukkan pada **Gambar 5**. Keadaan *limit switch* = 1 adalah saat lajur jalan terdapat antrian kendaraan yang padat. Keadaan tersebut ditunjukkan pada **Gambar 6.** Ketika *limit switch* lajur utara bernilai 0, maka LED lajur utara mengalami transisi dari LED merah dan kuning menjadi LED hijau. Selain itu, digit 1 *seven segment* menunjukkan angka 5 yang ditampilkan pada **Gambar 7**. Saat *output state* yaitu LED hijau, maka cara kerja *counter* berubah menjadi *counter down* yang ditampilkan pada **Gambar 8**.Ketika *limit switch* lajur utara bernilai 1, maka LED lajur utara

mengalami transisi dari LED merah dan kuning menjadi LED hijau. Selain itu digit 1 *seven segment* menunjukkan angka 7 seperti yang ditampilkan pada **Gambar 9.**



Gambar 5. Keadaan saat *Limit Switch* = 0.

Gambar 6. Keadaan saat *Limit Switch* = 1.



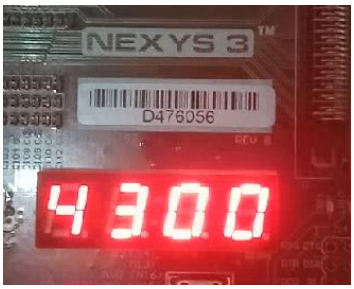
Gambar 7. Tampilan *Counter Seven Segment* pada Keadaan *Reset* atau Keadaan Awal.

Gambar 8. Tampilan *Counter Seven Segment* jika *Limit Switch* Lajur Utara = 0.

Gambar 9. Tampilan *Counter Down Seven Segment* pada Lajur Utara.

Setelah waktu jeda yang diberikan telah habis untuk LED hijau hidup, maka *state* berikutnya adalah *state* 3. Pada saat transisi menuju *state* 3, digit 1 dan digit 2 pada *seven segment* berubah menjadi 0. Pada saat *state* 3 sedang beroperasi, maka digit 1 tetap menunjukkan angka 0 hingga 3 detik dan digit 2 bekerja secara *counter up*. Hal ini dilakukan karena pada saat lajur utara menghasilkan *output* LED kuning. Dengan pertimbangan pengendara tidak memperhatikan lama waktu LED kuning menyala. Selain itu pengendara lebih fokus terhadap persiapan untuk berhenti sebelum LED berubah menjadi merah. Perubahan digit pada *seven segment* ditunjukkan pada **Gambar 11**.

Gambar 10. Tampilan *Counter Seven Segment* jika *Limit Switch* Lajur Utara = 1.



Gambar 11. Tampilan *Counter Seven Segment* awal pada Lajur Barat.

4. Kesimpulan

Rangkaian *traffic light* sudah berhasil direalisasikan dalam bentuk *hardware* menggunakan modul FPGA Xilinx Spartan-6 Nexys 3. Masukan sensor *limit switch* diproses pada saat memasuki *state* dimana *output*nya yaitu LED hijau hidup. Setelah memasuki *state* LED hijau, maka masukan sensor *limit switch* tidak berpengaruh pada

lamanya waktu jeda LED hijau. Masukan sensor *limit switch* bernilai 0 jika tidak terdapat kendaraan yang melintas di lajur jalan atau tidak terdapat antrian kendaraan yang padat. *limit switch* = 0, diperoleh *output* waktu jeda untuk LED warna hijau hidup dalam waktu 5 detik. Masukan sensor *limit switch* bernilai 1 jika terdapat antrian kendaraan yang padat. Saat *limit switch* = 1, diperoleh *output* waktu jeda untuk LED warna hijau hidup dalam waktu 7 detik. Masukan sensor *limit switch* yang bukan merupakan lajurnya, maka tidak mempengaruhi waktu jeda LED hijau pada lajur tersebut. Semakin kecil jumlah *prescaler*, maka frekuensi *output* yang diperoleh semakin kecil dan menghasilkan *clock* dengan waktu yang semakin lama. Untuk hasil yang lebih optimal, diperlukan adanya pengukuran tiap gerbang pada modul FPGA Xilinx Nexys 3. Untuk *traffic light* yang bersifat *adaptive*, diperlukan adanya instrument tambahan seperti sensor *infrared*.

Referensi

[1] Setianto; Liu Kin Men; Bambang Mukti Wibawa; Darmawan Hidayat, "Pengaturan Lampu Lalulintas Berbasis Fuzzy Logic," vol. 1, no. 2, pp. 16–20, 2017.

[2] D. Yildirim, "Traffic Light Control System for an Intersection Using S7-300 Plc", *Yeditepe University*, hal 1-12, 2011.

[3] D. A. Priutomo, I. R. Magdalena, and N. Andini, "Simulation and Analysis of System Smart Traffic Light Based on Digital Image Processing with Edge Detection and Segmentation," vol. 3, no. 1, pp. 478–485, 2016.

[4] A. N. Bawono, M. A. Riyadi, and T. Andromeda, "Perancangan Wearable Controller Menggunakan Modul Field Programmable Gate Array (Fpga) Xilinx Nexys 3 dan Accelerometer ADXL345."*Transient*, vol. 6, no.1, 2017.

[5] S. N. A. H. S. Zaim, "Perancangan Implementasi SHA-1 Password Cracking Berbasis FPGA," *Telemat. MKOM*, vol. 3, 2013.

[6] A. Rizal and M. A. Riyadi, "Perancangan Sistem Instrumentasi Medis Pengukur Tiga Tanda Vital Tubuh Menggunakan Core Fpga Xilinx Spartan-6."*Transient*, vol.4, no.3, 2015.

[7] J. Sirkunan, T. Andromeda, and M. N. Marsono, "Reconfigurable Logic Embedded Architecture of Support Vector Machine Linear Kernel," *IEEE*, pp. 1–5, 2017.

[8] H. R. Loo, A. Monemi, T. Andromeda, and M. N. Marsono, "Incremental High Throughput Network Traffic Classifier," *IEEE*, pp. 1–6, 2017.

[9] S. S. Chawla, S. Kamal, and N. Goel, "FPGA implementation of a traffic light system: Self adaptive with disruptions indicator," *12th IEEE Int. Conf. Electron. Energy, Environ. Commun. Comput. Control (E3-C3), INDICON 2015*, 2016.

[10] S. V Lahade, "Intelligent and Adaptive Traffic Light Controller (IA-TLC) using FPGA,"*IEEE*, pp.618-623, 2015.

[11] S. Shuo, H. Tian, and Y. Zhai, "Design of Intelligent Traffic Light Controller Based on VHDL," *2009 Second Int. Work. Knowl. Discov. Data Min.*, pp. 272–275, 2009.