Perancangan Komponen Terprogram
Tugas Kuliah dan Praktikum PKT Project 3
Odemeter dua sumbu axis

Disusun oleh:

Nurul Akbar Arlan

(07111940000075)

Dosen:

Rudy Dikairono, ST., MT.

Bidang Studi Elektronika

Departemen Teknik Elektro

Institut Teknologi Sepuluh Nopember

Surabaya

2021

Tugas Kuliah dan Praktikum PKT Project 3
Odemeter dua sumbu axis

Rangkaian Digital VHDL

Pada bagian awal rangkaian yaitu inisialisasi port dan sinyal VHDL.

```
LIBRARY IEEE ;
USE IEEE.STD_LOGIC_1164.ALL;
USE IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.NUMERIC_STD.ALL;

ENTITY odometry_2_axis IS

    PORT    (rot0_a, rot0_b, rot1_a, rot1_b                    : in STD_LOGIC;
             loutx2, loutx1, loutx0, louty2, louty1, louty0    : out STD_LOGIC_VECTOR (6 downto 0));

END odometry_2_axis;

ARCHITECTURE logic OF odometry_2_axis IS

    SIGNAL a0prev        :  STD_LOGIC                       := rot0_a;
    SIGNAL a1prev        :  STD_LOGIC                       := rot1_a;
    SIGNAL counter0      :  INTEGER range -2191 to 2191     := 0;
    SIGNAL counter1      :  INTEGER range -2191 to 2191     := 0;
    SIGNAL posisitemp0   :  INTEGER range 0 to 999999       := 0;
    SIGNAL posisitemp1   :  INTEGER range 0 to 999999       := 0;
    SIGNAL posisitemp_x  :  INTEGER range 0 to 999999       := 0;
    SIGNAL posisitemp_y  :  INTEGER range 0 to 999999       := 0;
    SIGNAL posisi_x      :  STD_LOGIC_VECTOR (9 DOWNTO 0)   := "0000000000";
    SIGNAL posisi_y      :  STD_LOGIC_VECTOR (9 DOWNTO 0)   := "0000000000";
    SIGNAL bcdx0         :  STD_LOGIC_VECTOR (3 DOWNTO 0)   := "0000";
    SIGNAL bcdx1         :  STD_LOGIC_VECTOR (3 DOWNTO 0)   := "0000";
    SIGNAL bcdx2         :  STD_LOGIC_VECTOR (3 DOWNTO 0)   := "0000";
    SIGNAL bcdy0         :  STD_LOGIC_VECTOR (3 DOWNTO 0)   := "0000";
    SIGNAL bcdy1         :  STD_LOGIC_VECTOR (3 DOWNTO 0)   := "0000";
    SIGNAL bcdy2         :  STD_LOGIC_VECTOR (3 DOWNTO 0)   := "0000";
```

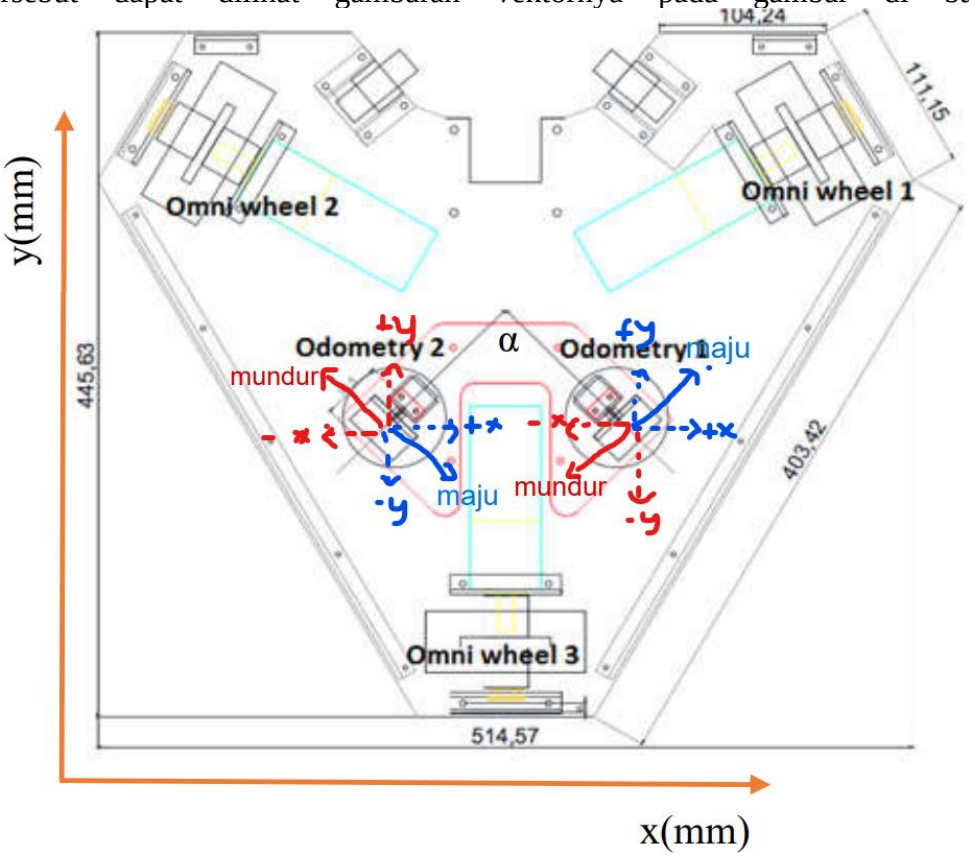
port terdiri dari empat input dan enam output, port input terdapat rout0_a dan rout0_b yang merupakan output dari rotary encoder odometry 1 dan rout1_a dan rout1_b yang merupakan output dari rotary encoder odometry 2. Sinyal sebanyak 16 yang terdiri dari a0prev dan a1prev yang berfungsi menyimpan nilai rot_a iterasi sebelumnya, counter0 dan counter1 befungsi menyimpan angka counter, posisitemp0 dan posisitemp1 berfungsi menyimpan nilai jarak tempuh odometry dalam bentuk integer, posisitemp_x dan posisitemp_y befungsi menyimpan nilai posisi x dan y dalam bentuk integer, posisi_x dan posisi_y berfungsi menyimpan nilai posisi dalam bentuk std_logic_vector, dan bcdx0 hingga bcdx2 dan bcdy0 hingga bcdy2 untuk menyimpan nilai posisi dalam bentuk binary coded decimal untuk seven segment.

```
    PROCESS (rot0_a, rot0_b, rot1_a, rot1_b)
    variable Zx    :  STD_LOGIC_VECTOR (21 DOWNTO 0);
    variable Zy    :  STD_LOGIC_VECTOR (21 DOWNTO 0);
    BEGIN
       --memastikan nilai zx kembali 0
       FOR i in 0 to 21 LOOP
          Zx(i) := '0';
       END LOOP;
       --memastikan nilai zy kembali 0
       FOR i in 0 to 21 LOOP
          Zy(i) := '0';
       END LOOP;
       --rotary encoder 0
       IF (rot0_a /= a0prev) THEN
          IF (rot0_b /= rot0_a) THEN
             IF (counter0 < 2191) THEN
                counter0 <= counter0 + 1;
             END IF;
          ELSIF (rot0_b = rot0_a) THEN
             IF (counter0 > -2191) THEN
                counter0 <= counter0 - 1;
             END IF;
          END IF;
       END IF;
       a0prev <= rot0_a;
       --rotary encoder 1
       IF (rot1_a /= a1prev) THEN
          IF (rot1_b /= rot1_a) THEN
             IF (counter1 < 2191) THEN
                counter1 <= counter1 + 1;
             END IF;
          ELSIF (rot1_b = rot1_a) THEN
             IF (counter1 > -2191) THEN
                counter1 <= counter1 - 1;
             END IF;
          END IF;
       END IF;
       a1prev <= rot1_a;
```

Selanjutnya adalah proses pertama dari rangkaian VHDL ini, sebelum ke proses pertama pada line 38 sampai 44 terdapat command untuk memastikan nilai Zx dan Zy kembali menjadi berisi 0, yang nantinya digunakan pada proses konversi biner ke bcd, kembali menjadi nol. Proses pertama dari rangkaian VHDL ini merupakan updown counter yang ditrigger oleh arah Gerakan counter tersebut, counter akan bekerja sebagai up counter ketika rotary encoder berputar searah jarum jam, sebaliknya ketika rotary encoder berputar berlawanan arah jarum jam maka counter akan bekerja sebagai down counter.

```
71      --hitung posisi
72      posisitemp0 <= abs(counter0) * 456 / 1000;
73      posisitemp1 <= abs(counter1) * 456 / 1000;
74      IF (counter0 >= 0) THEN
75         IF (counter1 >= 0) THEN
76            posisitemp_x <= (posisitemp0 * 707 / 1000) + (posisitemp1 * 707 / 1000);
77            posisitemp_y <= (posisitemp0 * 707 / 1000) - (posisitemp1 * 707 / 1000);
78         ELSE
79            posisitemp_x <= (posisitemp0 * 707 / 1000) - (posisitemp1 * 707 / 1000);
80            posisitemp_y <= (posisitemp0 * 707 / 1000) + (posisitemp1 * 707 / 1000);
81         END IF;
82      ELSE
83         IF (counter1 >= 0) THEN
84            posisitemp_x <= -(posisitemp0 * 707 / 1000) + (posisitemp1 * 707 / 1000);
85            posisitemp_y <= -(posisitemp0 * 707 / 1000) - (posisitemp1 * 707 / 1000);
86         ELSE
87            posisitemp_x <= -(posisitemp0 * 707 / 1000) - (posisitemp1 * 707 / 1000);
88            posisitemp_y <= -(posisitemp0 * 707 / 1000) + (posisitemp1 * 707 / 1000);
89         END IF;
90      END IF;
91      posisi_x <= std_logic_vector(to_unsigned(posisitemp_x, posisi_x'length));
92      posisi_y <= std_logic_vector(to_unsigned(posisitemp_y, posisi_y'length));
```

Kemudian perhitungan posisi dari kedua odometry, posisitemp0 dan posisitemp1 pada line 72 hingga 73 menyimpan nilai jarak yang dilalui oleh odometry 1 dan odometry 2 selanjutnya dari nilai jarak tersebut dapat dicari posisinya dengan cara, mengubahnya dalam bentuk vektor x maupun y, nilai vektor tersebut dapat dilihat gambaran vektornya pada gambar di bawah.



karena kedua odometry miring 45 derajat terhadap sumbu x maupun y maka dapat dicari nilai x-nya dengan dikalikan dengan sin(pi/4) yang bernilai sekitar 0.707. Lalu nilai x dan y masing-masing odometry dijumlahkan sehingga dapat nilai posisi x dan posisi y.

```
93          --convert ke bcd axis x
94          Zx(12 DOWNTO 3) := posisi_x;
95          FOR i IN 0 TO 6 LOOP
96             IF Zx(13 DOWNTO 10) > "0100" THEN
97                Zx(13 DOWNTO 10) := Zx(13 DOWNTO 10) + "0011";
98             END IF;
99             IF Zx(17 DOWNTO 14) > "0100" THEN
100               Zx(17 DOWNTO 14) := Zx(17 DOWNTO 14) + "0011";
101            END IF;
102            Zx(21 DOWNTO 1) := Zx(20 DOWNTO 0);
103         END LOOP;
104         bcdx0 <= Zx(13 DOWNTO 10);
105         bcdx1 <= Zx(17 DOWNTO 14);
106         bcdx2 <= Zx(21 DOWNTO 18);
107         --convert ke bcd axis y
108         Zy(12 DOWNTO 3) := posisi_y;
109         FOR i IN 0 TO 6 LOOP
110            IF Zy(13 DOWNTO 10) > "0100" THEN
111               Zy(13 DOWNTO 10) := Zy(13 DOWNTO 10) + "0011";
112            END IF;
113            IF Zy(17 DOWNTO 14) > "0100" THEN
114               Zy(17 DOWNTO 14) := Zy(17 DOWNTO 14) + "0011";
115            END IF;
116            Zy(21 DOWNTO 1) := Zy(20 DOWNTO 0);
117         END LOOP;
118         bcdy0 <= Zy(13 DOWNTO 10);
119         bcdy1 <= Zy(17 DOWNTO 14);
120         bcdy2 <= Zy(21 DOWNTO 18);
121      END PROCESS;
122
```

Proses ketiga adalah pengonversian nilai biner dari counter menjadi bcd(binary coded decimal) karena dibutuhkan oleh seven segment. Proses ini bekerja dengan cara angka biner digeser(shift right) dan dicuplik empat angka-empat angka, bila nilai dari keempat angka biner itu lebih besar dari 4 maka ditambahkan dengan 3, dan seterusnya hingga semua angka biner tercuplik.

```
123      --seven segment
124      loutx0 <= "0000001" when bcdx0 = "0000" else
125                "1001111" when bcdx0 = "0001" else
126                "0010010" when bcdx0 = "0010" else
127                "0000110" when bcdx0 = "0011" else
128                "1001100" when bcdx0 = "0100" else
129                "0100100" when bcdx0 = "0101" else
130                "0100000" when bcdx0 = "0110" else
131                "0001111" when bcdx0 = "0111" else
132                "0000000" when bcdx0 = "1000" else
133                "0000100" when bcdx0 = "1001" else
134                "0110000";
135
136      loutx1 <= "0000001" when bcdx1 = "0000" else
137                "1001111" when bcdx1 = "0001" else
138                "0010010" when bcdx1 = "0010" else
139                "0000110" when bcdx1 = "0011" else
140                "1001100" when bcdx1 = "0100" else
141                "0100100" when bcdx1 = "0101" else
142                "0100000" when bcdx1 = "0110" else
143                "0001111" when bcdx1 = "0111" else
144                "0000000" when bcdx1 = "1000" else
145                "0000100" when bcdx1 = "1001" else
146                "0110000";
147
148      loutx2 <= "0000001" when bcdx2 = "0000" else
149                "1001111" when bcdx2 = "0001" else
150                "0010010" when bcdx2 = "0010" else
151                "0000110" when bcdx2 = "0011" else
152                "1001100" when bcdx2 = "0100" else
153                "0100100" when bcdx2 = "0101" else
154                "0100000" when bcdx2 = "0110" else
155                "0001111" when bcdx2 = "0111" else
156                "0000000" when bcdx2 = "1000" else
157                "0000100" when bcdx2 = "1001" else
158                "0110000";
159
```

```
    louty0 <= "0000001" when bcdy0 = "0000" else
              "1001111" when bcdy0 = "0001" else
              "0010010" when bcdy0 = "0010" else
              "0000110" when bcdy0 = "0011" else
              "1001100" when bcdy0 = "0100" else
              "0100100" when bcdy0 = "0101" else
              "0100000" when bcdy0 = "0110" else
              "0001111" when bcdy0 = "0111" else
              "0000000" when bcdy0 = "1000" else
              "0000100" when bcdy0 = "1001" else
              "0110000";

    louty1 <= "0000001" when bcdy1 = "0000" else
              "1001111" when bcdy1 = "0001" else
              "0010010" when bcdy1 = "0010" else
              "0000110" when bcdy1 = "0011" else
              "1001100" when bcdy1 = "0100" else
              "0100100" when bcdy1 = "0101" else
              "0100000" when bcdy1 = "0110" else
              "0001111" when bcdy1 = "0111" else
              "0000000" when bcdy1 = "1000" else
              "0000100" when bcdy1 = "1001" else
              "0110000";

    louty2 <= "0000001" when bcdy2 = "0000" else
              "1001111" when bcdy2 = "0001" else
              "0010010" when bcdy2 = "0010" else
              "0000110" when bcdy2 = "0011" else
              "1001100" when bcdy2 = "0100" else
              "0100100" when bcdy2 = "0101" else
              "0100000" when bcdy2 = "0110" else
              "0001111" when bcdy2 = "0111" else
              "0000000" when bcdy2 = "1000" else
              "0000100" when bcdy2 = "1001" else
              "0110000";
END logic;
```

Dan proses terakhir adalah mengubah nilai bcd kedalam seven segment.

VHDL Testbench

Untuk testbench pertama yaitu robot bergerak searah sumbu x.

```
add wave rot0_a rot0_b rot1_a rot1_b loutx2 loutx1 loutx0 louty2 louty1 louty0
force rot0_a 0 0, 1 5ns, 0 15ns -repeat 20ns
force rot0_b 0 0, 1 10ns -repeat 20ns
force rot1_a 0 0, 1 5ns, 0 15ns -repeat 20ns
force rot1_b 0 0, 1 10ns -repeat 20ns
run 15510ns
```

sinyal rot0_a dan rot0_b berputar searah jarum jam, rot0_a diberi sinyal 0-1-1-0 dengan periode 5ns setiap nilai dan rot0_b diberi sinyal 0-0-1-1 dengan periode 5ns setiap nilai, diulang terus menerus hingga 15510ns, agar nilai count mencapai 1548 sehingga jarak yang ditempuh mencapai 706mm dan sinyal rot1_a dan rot1_b juga berputar searah jarum jam, rot1_a diberi sinyal 0-1-1-0 dengan periode 5ns setiap nilai dan rot1_b diberi sinyal 0-0-1-1 dengan periode 5ns setiap nilai, diulang terus menerus hingga 15510ns, agar nilai count mencapai 1548 sehingga jarak yang ditempuh mencapai 706mm.

Testbench selanjutnya yaitu robot bergerak searah sumbu y.

```
add wave rot0_a rot0_b rot1_a rot1_b loutx2 loutx1 loutx0 louty2 louty1 louty0
force rot0_a 0 0, 1 5ns, 0 15ns -repeat 20ns
force rot0_b 0 0, 1 10ns -repeat 20ns
force rot1_a 1 1, 0 5ns, 1 15ns -repeat 20ns
force rot1_b 0 0, 1 10ns -repeat 20ns
run 15510ns
```

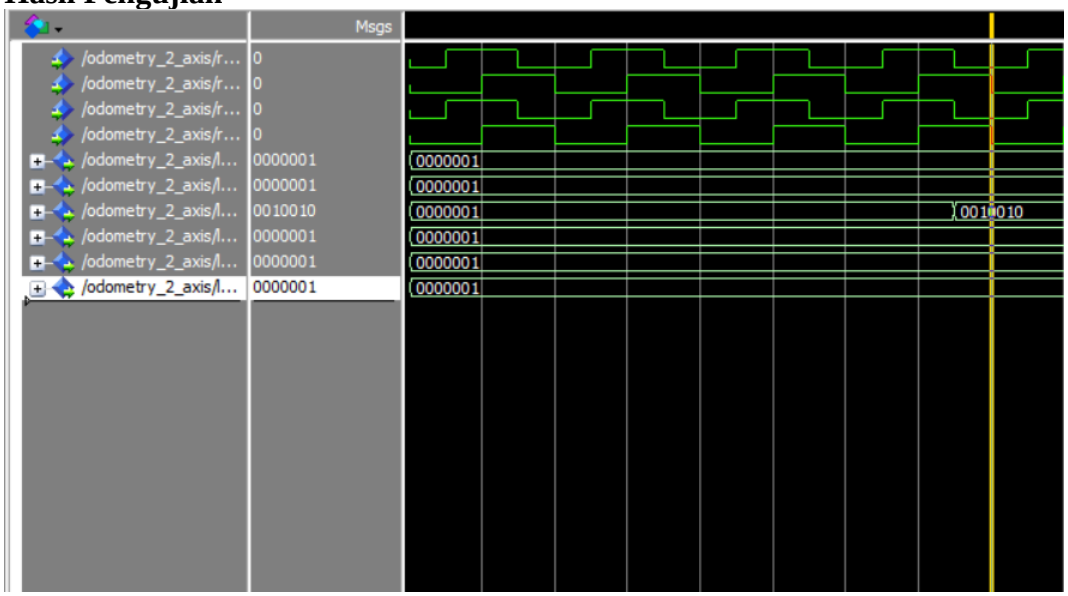
sinyal rot0_a dan rot0_b berputar searah jarum jam, rot0_a diberi sinyal 0-1-1-0 dengan periode 5ns setiap nilai dan rot0_b diberi sinyal 0-0-1-1 dengan periode 5ns setiap nilai, diulang terus menerus hingga 15510ns, agar nilai count mencapai 1548 sehingga jarak yang ditempuh mencapai 706mm dan sinyal rot1_a dan rot1_b juga berputar berlawanan arah jarum jam, rot1_a diberi sinyal 1-0-0-1 dengan periode 5ns setiap nilai dan rot1_b diberi sinyal 0-0-1-1 dengan periode 5ns setiap nilai, diulang terus menerus hingga 15510ns, agar nilai count mencapai 1548 sehingga jarak yang ditempuh mencapai 706mm.

Dan tesbench terakhir ketika robot bergerak 45 derajat terhadap sumbu x.

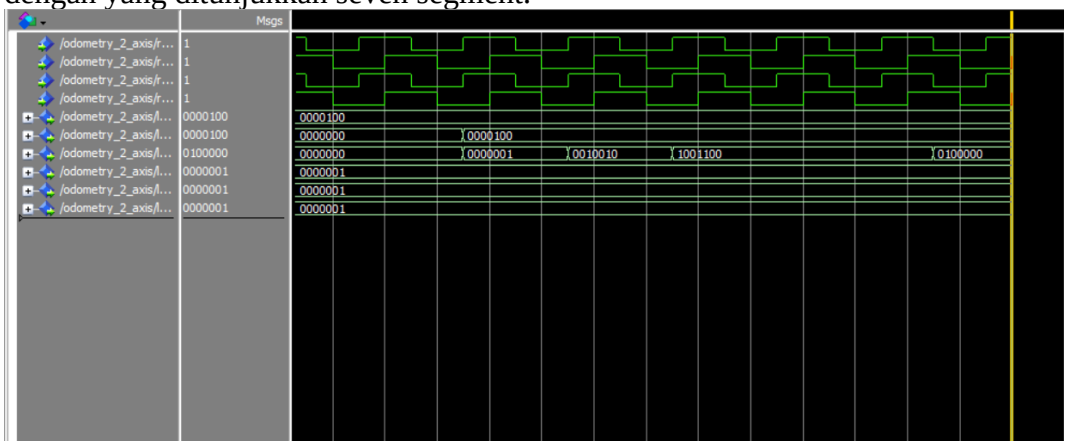
```
add wave rot0_a rot0_b rot1_a rot1_b loutx2 loutx1 loutx0 louty2 louty1 louty0
force rot0_a 0 0, 1 5ns, 0 15ns -repeat 20ns
force rot0_b 0 0, 1 10ns -repeat 20ns
force rot1_a 0 0
force rot1_b 0 0
run 21940ns
```

sinyal rot0_a dan rot0_b berputar searah jarum jam, rot0_a diberi sinyal 0-1-1-0 dengan periode 5ns setiap nilai dan rot0_b diberi sinyal 0-0-1-1 dengan periode 5ns setiap nilai, diulang terus menerus hingga 21940ns, agar nilai count mencapai 2191 sehingga jarak yang ditempuh mencapai 999mm dan sinyal rot1_a dan rot1_b juga tidak berputar sehingga diatur sinyal bernilai 0 hingga akhir.

Hasil Pengujian

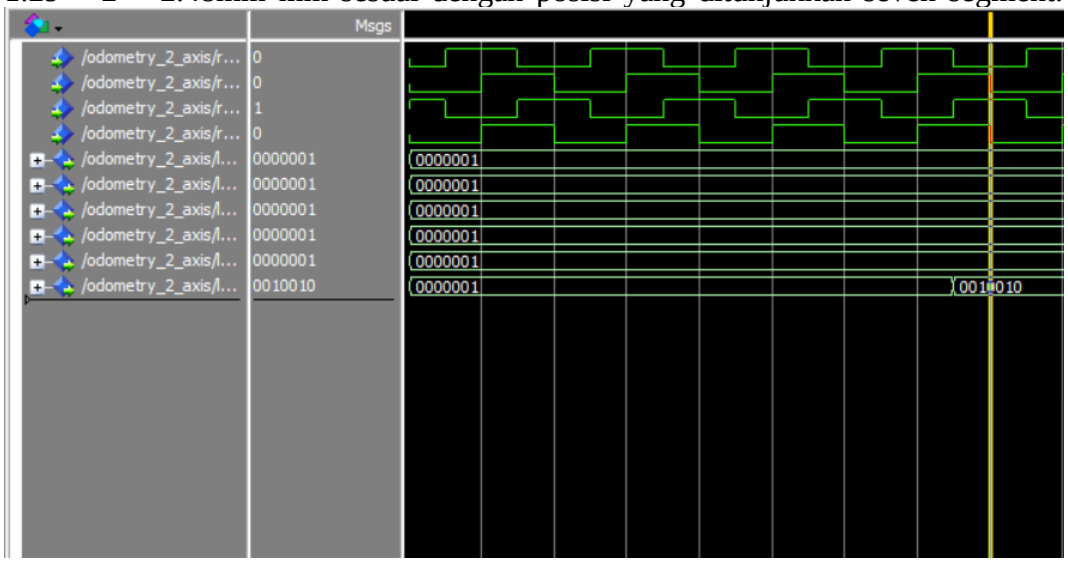


Untuk hasil pengujian testbench pertama terlihat bahwa ketika rangkaian diberi 4 count atau 2 pulsa seven segment menunjukkan posisi (2,0) mm terbukti dari hitungan 4 * 0.456 = 1.824 mm jarak yang ditempuh odometry, 1.824 * 0.707 = 1.29 mm jarak x yang ditempuh 1 odometry, sehingga jarak x yang ditempuh adalah 1.29 * 2 = 2.48 mm sesuai dengan posisi yang ditunjukkan seven segment. Lalu pada 15510ns seven segment menunjukkan angka (996,0) mm yang berarti rangkaian telah diberi 773 pulsa atau 1546 count, 1546 * 0.456 = 704.98 mm jarak yang ditempuh odometry, 704.98 * 0.707 = 490.42mm jarak x yang ditempuh odometry, sehingga jarak x yang ditempuh adalah 490.42 * 2 = 996.84 mm sesuai dengan yang ditunjukkan seven segment.

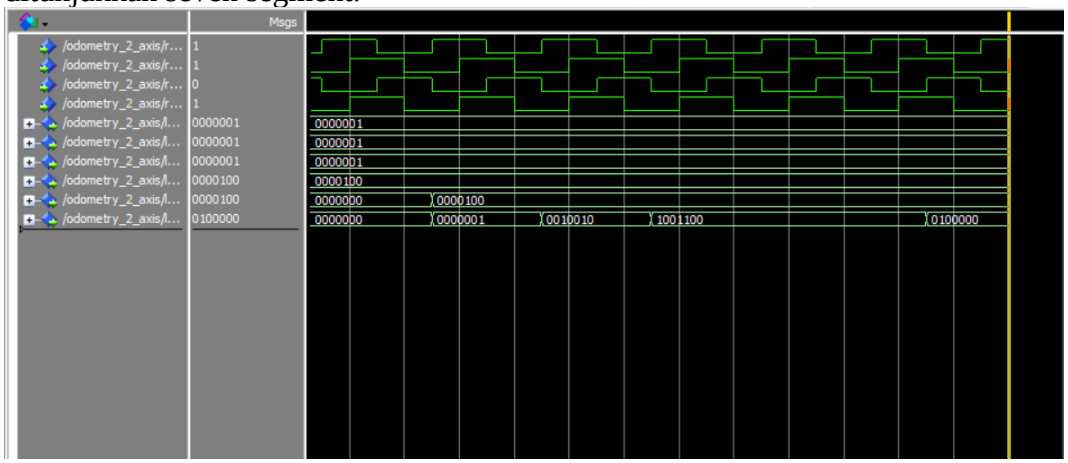


Lalu pada pengujian testbench kedua pada 80ns atau setelah count menyentuh 4 atau 2 pulsa seven segment menunjukkan posisi (0,2) mm terbukti dari hitungan 4 * 0.456 = 1.824 mm jarak yang ditempuh odometry, 1.824 * 0.707 = 1.29 mm jarak y yang ditempuh odometry, sehingga jarak y yang ditempuh adalah

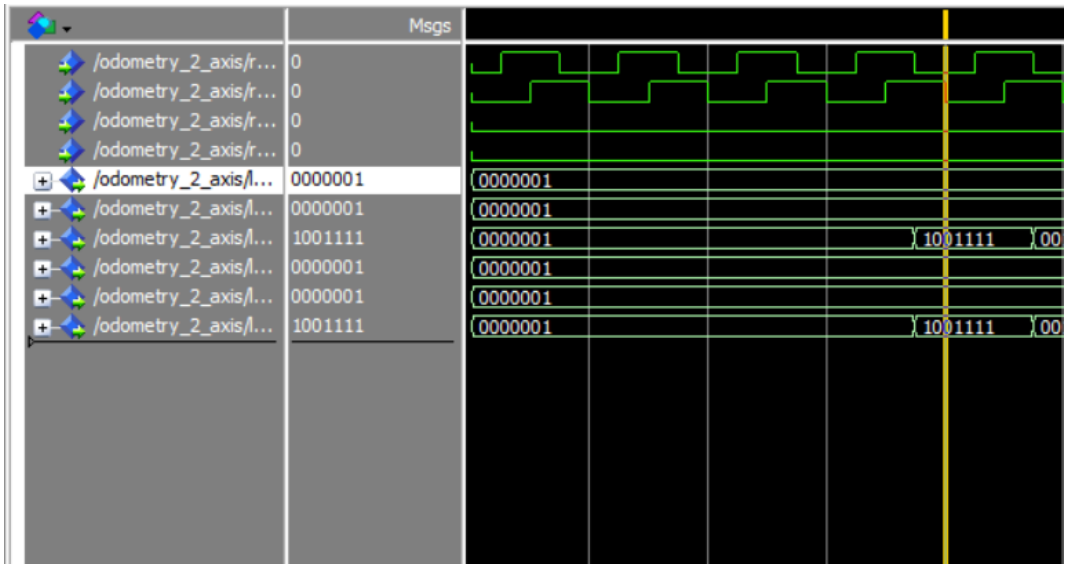
1.29 * 2 = 2.48mm mm sesuai dengan posisi yang ditunjukkan seven segment.



dan pada 15510ns seven segment menunjukkan (0,996) mm yang berarti rangkaian telah diberi 773 pulsa atau 1546 count, 1546 * 0.456 = 704.98 mm jarak yang ditempuh odometry, 704.98 * 0.707 = 490.42 mm jarak x yang ditempuh odometry, sehingga jarak x yang ditempuh adalah 490.42 * 2 = 996.84 mm sesuai dengan yang ditunjukkan seven segment.



Pengujian testbench terakhir ketika robot bergerak 45 derajat terhadap sumbu x, pada 80ns atau setelah count menyentuh 4 atau 2 pulsa seven segment menunjukkan posisi (1,1) mm terbukti dari hitungan 4 * 0.456 = 1.824 mm jarak yang ditempuh odometry 1, 1.824 * 0.707 = 1.29 mm jarak x dan y yang ditempuh odometry sesuai dengan posisi yang ditunjukkan seven segment.



dan pada 21930 ns seven segment menunjukkan posisis (706,706) mm, dari hitungan 2192 * 0.456 = 999.552 mm jarak yang ditempuh odometry 1, 999.552 * 0.707 = 706.68 mm jarak x dan y yang ditempuh odometry sesuai dengan posisi yang ditunjukkan seven segment.