TRABAJO PRÁCTICO Nº 2

Análisis Dinámico de un Robot e implementación en FPGA

Alumnos:

- **Jorge, Asade.**
- **Rosende, Alejandro.**
- **Villafañe, Melisa.**

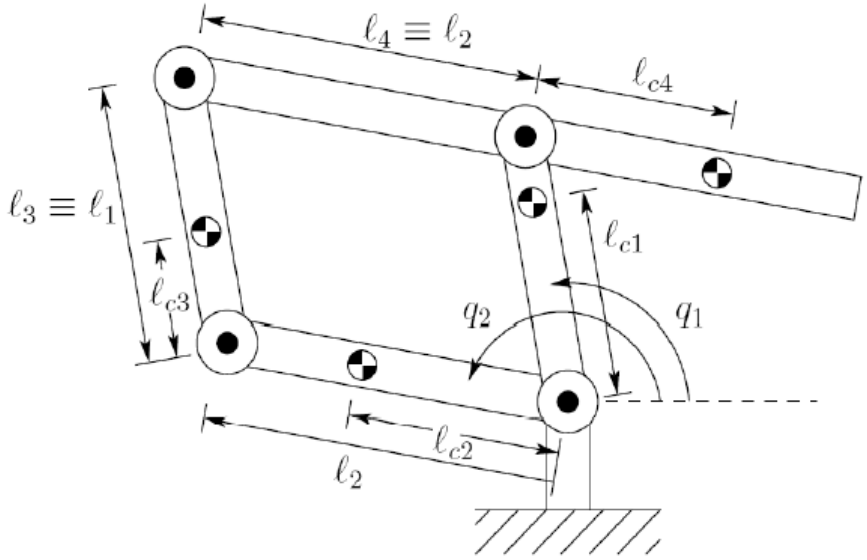
Profesor: Ing. H. Giannetta.

Análisis Dinámico de un Robot e implementación en FPGA.

Introducción sobre dinámica del robot.

Introducción

El diseño y control del manipulador **"Five Bar Linkage"** se divide en tres partes cinemática, dinámica y control. En este TP realizaremos el análisis dinámico de la estructura mecánica.



Este sistema presenta la ventaja de que eligiendo correctamente la longitud de los lados se simplifican las ecuaciones de forma que se puede obtener dos ecuaciones independientes entre si de q1 y q2.

Análisis dinámico

Datos:

- $l_1 = l_3 = 13.5cm$
- $l_2 = l_4 = 7.5cm$
- $l_{C1} = l_{C3} = 6.75cm$
- $l_{C2} = 3.75cm$
- $l_{C4} = 5.625cm$
- $m_1 = m_3 = 1.5kg$
- $m_2 = m_4 = 1kg$

Para desarrollar el análisis dinámico vamos a asumir:

$$l_1 = l_3 \wedge l_2 = l_4$$

Si bien las longitudes deben coincidir, los centros de masa no tienen esta limitación.

$$L_{C1} \neq L_{C3}$$

Para el análisis dinámico utilizamos el enfoque energético de Lagrange-Euler.

2. Ecuaciones de los centros de masa:

$$\begin{bmatrix} X_{C1} \\ Y_{C1} \end{bmatrix} = \begin{bmatrix} l_{C1}\cos(q_1) \\ l_{C1}sen(q_1) \end{bmatrix}$$

$$\begin{bmatrix} X_{C2} \\ Y_{C2} \end{bmatrix} = \begin{bmatrix} l_{C2}\cos(q_2) \\ l_{C2}sen(q_2) \end{bmatrix}$$

$$\begin{bmatrix} X_{C3} \\ Y_{C3} \end{bmatrix} = \begin{bmatrix} l_{C2}\cos(q_2) \\ l_{C2}sen(q_2) \end{bmatrix} + \begin{bmatrix} l_{C3}\cos(q_1) \\ l_{C3}sen(q_1) \end{bmatrix}$$

$$\begin{bmatrix} X_{C4} \\ Y_{C4} \end{bmatrix} = \begin{bmatrix} l_{C1}\cos(q_1) \\ l_{C1}sen(q_1) \end{bmatrix} + \begin{bmatrix} l_{C4}\cos(q_2 - \pi) \\ l_{C4}sen(q_2 - \pi) \end{bmatrix}$$

$$\begin{bmatrix} X_{C4} \\ Y_{C4} \end{bmatrix} = \begin{bmatrix} l_{C1}\cos(q_1) \\ l_{C1}sen(q_1) \end{bmatrix} - \begin{bmatrix} l_{C4}\cos(q_2) \\ l_{C4}sen(q_2) \end{bmatrix}$$

3. Calculo de las velocidades en función de los centros de masa:

$$v_{C1} = \begin{bmatrix} -l_{C1}\sin(q_1) & 0 \\ l_{C1}\cos(q_1) & 0 \end{bmatrix}\dot{q}$$

$$v_{C2} = \begin{bmatrix} 0 & -l_{C2}sen(q_2) \\ 0 & l_{C2}\cos(q_2) \end{bmatrix}\dot{q}$$

$$v_{C3} = \begin{bmatrix} -l_{C3}sen(q_1) & -l_{C2}sen(q_2) \\ l_{C3}\cos(q_1) & l_{C2}\cos(q_2) \end{bmatrix}$$

$$v_{C4} = \begin{bmatrix} -l_{C1}sen(q_1) & l_{C4}\cos(q_2) \\ l_{C1}\cos(q_1) & -l_{C4}sen(q_2) \end{bmatrix}$$

4. En las velocidades angulares de los cuatro enlaces es evidente que:
   $\omega_1 = \omega_3 = q_1 k, \omega_2 = \omega_4 = q_4 k$

5. La matriz inercia esta dada por:

$$D(q) = \sum_{i=1}^{4} m_1 J_{vc}^T J_{vc} + \begin{bmatrix} I_1 + I_3 & 0 \\ 0 & I_2 + I_4 \end{bmatrix}$$

6. Sustituyendo en la ecuación lo obtenido en el punto 2, obtenemos
   $d_{11}(q) = m_1 l_{C1}^2 + m_3 l_{C3}^2 + m_4 l_1^2 + I_1 + I_3$
   $d_{12}(q) = d_{21}(q) = (m_3 l_2 l_{C3} - m_4 l_1 l_{C4})\cos(q_2 - q_1)$
   $d_{22}(q) = m_2 l_{C2}^2 + m_3 l_2^2 + m_4 l_{C4}^2 + I_2 + I_4$
   $I_1 = I_3 = l_{C1}^2 \cdot m_1 = 0.0068 kg \cdot m^2$
   $I_2 = l_{C2}^2 \cdot m_2 = 0.0014 kg \cdot m^2$
   $I_4 = l_{C4}^2 \cdot m_4 = 0.0032 kg \cdot m^2$
   $d_{11}(q) = 0.0455 kg \cdot m^2$
   $d_{22}(q) = 0.013 kg \cdot m^2$

7. Considerando la condición
   $m_3 l_2 l_{C3} = m_4 l_1 l_{C4}$
   $m_3 l_2 l_{C3} = 75.9375 kg \cdot cm^2$
   $m_4 l_1 l_{C4} = 75.9375 kg \cdot cm^2$
   Esta condición se debe cumplir siempre a menos que se mantenga siempre la relación $q_1 + \frac{\pi}{2} = q_2$

8. Por lo tanto $d_{12}(q) = d_{21}(q) = 0$, es decir que la matriz es diagonal y constante. Como consecuencia las ecuaciones dinámicas no contendrán ni las fuerzas de Coriolis ni las fuerzas Centrífugas.

9. Entonces la energía potencial queda:

$$P = g\sum_{i=1}^{4} y_{Ci} = g \cdot sen(q_1) \cdot (m_1 l_{C1} + m_3 l_{C3} + m_4 l_1) + g \cdot sen(q_2) \cdot (m_2 l_{C2} + m_3 l_2 - m_4 l_{C4})$$

Reescribiendo:

$$\phi_1 = g \cdot \cos(q_1) \cdot (m_1 l_{C1} + m_3 l_{C3} + m_4 l_1) = 10\frac{m}{s^2} \cdot \cos(q_1) \cdot 0.3375 kg \cdot m = 3.375 N \cdot m \cdot \cos(q_1)$$

$$\phi_2 = g \cdot \cos(q_2) \cdot (m_2 l_{C2} + m_3 l_2 + m_4 l_{C4}) = 10\frac{m}{s^2} \cdot \cos(q_2) \cdot 0.20625 kg \cdot m = 2.0625 N \cdot m \cdot \cos(q_2)$$

10. Sabiendo que ϕ_1 depende solo de q_1 y que ϕ_2 depende solo de q_2 se puede reescribir:

$$d_{11}\ddot{q}_1 + \phi_1(q_1) = \tau_1$$

$$d_{22}\ddot{q}_2 + \phi_2(q_2) = \tau_2$$

$$\tau_1 = \ddot{q}_1 \cdot 0.0455 kg \cdot m^2 + 3.375 N \cdot m \cdot \cos(q_1)$$

$$\tau_2 = \ddot{q}_2 \cdot 0.013 kg \cdot m^2 + 2.0625 N \cdot m \cdot \cos(q_2)$$

11. Así es que se puede operar con q_1 y q_2 independientemente sin preocuparse por los ángulos. Considerando una trayectoria de tercer orden:

$$q_1 = q_0 + q_1 \cdot t + q_2 \cdot t^2 + q_3 \cdot t^3$$

$$q_2 = q_0 + q_1 \cdot t + q_2 \cdot t^2 + q_3 \cdot t^3 + \frac{\pi}{2}$$

$$\tau_1 = (\alpha + t \cdot q_3) \cdot 0.0455 kg \cdot m^2 + 3.375 N \cdot m \cdot \cos(q_1)$$

$$\tau_2 = (\alpha + t \cdot q_3) \cdot 0.013 kg \cdot m^2 + 2.0625 N \cdot m \cdot \cos(q_2)$$

Considerando el torque máximo:

$$\tau_1 = (\alpha + t \cdot q_3) \cdot 0.0455 kg \cdot m^2 + 3.375 N \cdot m$$

$$\tau_2 = (\alpha + t \cdot q_3) \cdot 0.013 kg \cdot m^2 + 2.0625 N \cdot m$$

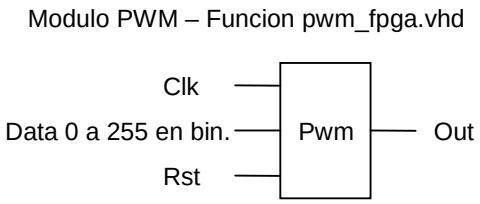
Se estimo un servo motores de Scheneider (Lexium 05) con las siguientes especificaciones:

- Potencia 0.4 to 6 kW - Alimentación 115 V to 480 V
- Torque Nominal 0.5 to 36 Nm
- Velocidad nominal 1500 to 8000 min –1
- Baja inercia

Resultados de la simulación

Código y esquemas

Utilizando la función – Bloque PWM (pwm_fpga.vhd)



Programa pwm_pretb.vhd

```
LIBRARY ieee;
use ieee.std_logic_1164.all;

ENTITY pwm_fpga_test_bench IS
END pwm_fpga_test_bench;

ARCHITECTURE arch_test_bench OF pwm_fpga_test_bench IS

COMPONENT pwm_fpga
PORT (
        clock: in std_logic;
        reset: in std_logic;
        data_value: in std_logic_vector(7 downto 0);
        pwm: out std_logic
);
END COMPONENT;

-- Internal signal declaration
SIGNAL sig_clock    : std_logic;
SIGNAL sig_reset1   : std_logic;
SIGNAL sig_reset2   : std_logic;
SIGNAL sig_reset3   : std_logic;
SIGNAL sig_data_value: std_logic_vector(7 downto 0);
SIGNAL sig_100      : std_logic_vector(7 downto 0);
SIGNAL sig_pwm1,sig_pwm2,sig_pwm3,sig_pwm4,sig_pwm5,sig_pwm6        : std_logic;
shared variable ENDSIM: boolean:=false;
  constant clk_period:TIME:=100 ns;
```

```
BEGIN
clk_gen: process

        BEGIN
                If ENDSIM = FALSE THEN
                        sig_clock <= '1';
                        wait for clk_period/2;
                        sig_clock <= '0';
                        wait for clk_period/2;
                else
                        wait;
                end if;
        end process;

-- Instancio los 6 pwm como modulos pwm_fpga

inst_pwm_fpga1 : pwm_fpga
PORT MAP(
                clock   => sig_clock,
                reset   => sig_reset1,
                data_value      => sig_data_value,
                pwm   => sig_pwm1
        );

inst_pwm_fpga2 : pwm_fpga
PORT MAP(
                clock   => sig_clock,
                reset   => sig_reset2,
                data_value      => sig_100,
                pwm   => sig_pwm2
        );
inst_pwm_fpga3 : pwm_fpga
PORT MAP(
                clock   => sig_clock,
                reset   => sig_reset2,
                data_value      => sig_data_value,
                pwm   => sig_pwm3
        );
inst_pwm_fpga4 : pwm_fpga
PORT MAP(
                clock   => sig_clock,
                reset   => sig_reset3,
                data_value      => sig_100,
                pwm   => sig_pwm4
        );
```

```
inst_pwm_fpga5 : pwm_fpga
PORT MAP(
                clock   => sig_clock,
                reset    => sig_reset3,
                data_value      => sig_data_value,
                pwm    => sig_pwm5
        );

inst_pwm_fpga6 : pwm_fpga
PORT MAP(
                clock   => sig_clock,
                reset    => sig_reset1,
                data_value      => sig_100,
                pwm    => sig_pwm6
        );

stimulus_process: PROCESS

  VARIABLE bit1,bit2,bit3,bit4,bit5,bit6,bit7,Aux  : integer;
  BEGIN
  bit1:=0;  -- Inicializo variables
  bit2:=0;
  bit3:=0;
  bit4:=0;
  bit5:=0;
  bit6:=0;
  bit7:=0;
  Aux:=0;
  sig_data_value <= "00000000";  -- Inicio variable de pwm
  sig_100 <= "00000001";    -- Inicio variable de 100%
        sig_reset1 <= '1';       -- todo en cero no da un duty del 100%, queda en 0
        sig_reset2 <= '1';       -- y no arranca
        sig_reset3 <= '1';
        wait for 100 ns;

        for I in 0 to 255 loop
        bit1:= bit1+1;   -- incremento del vector binario 0 a 255
        bit2:= bit2+1;
        bit3:= bit3+1;
        bit4:= bit4+1;
        bit5:= bit5+1;
        bit6:= bit6+1;
        bit7:= bit7+1;
        sig_data_value(0) <=  not sig_data_value(0);
        if(bit1 = 2) then
          sig_data_value(1) <=  not sig_data_value(1);
          bit1:=0;
          end if;
```

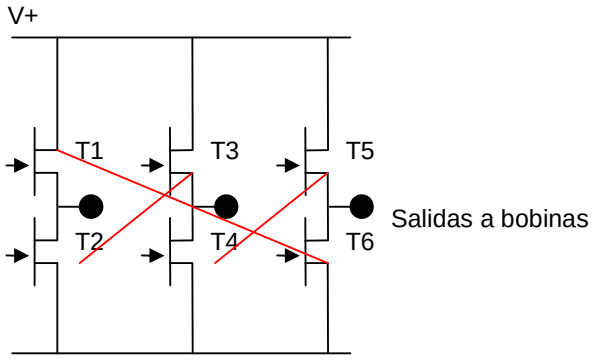
```
if(bit2 = 4) then
  sig_data_value(2) <=  not sig_data_value(2);
  bit2:=0;
  end if;
if(bit3 = 8) then
  sig_data_value(3) <=  not sig_data_value(3);
  bit3:=0;
  end if;
if(bit4 = 16) then
  sig_data_value(4) <=  not sig_data_value(4);
  bit4:=0;
  end if;
if(bit5 = 32) then
  sig_data_value(5) <=  not sig_data_value(5);
  bit5:=0;
  end if;
if(bit6 = 64) then
  sig_data_value(6) <=  not sig_data_value(6);
  bit6:=0;
  end if;
if(bit7 = 128) then
  sig_data_value(7) <=  not sig_data_value(7);
  bit7:=0;
end if;

if(Aux = 0) then      -- Habilito cada rama del puente
sig_reset1 <= '0';
sig_reset2 <= '1';
sig_reset3 <= '1';
end if;
if(Aux = 1) then
sig_reset1 <= '1';
sig_reset2 <= '0';
sig_reset3 <= '1';
end if;
if(Aux = 2) then
sig_reset1 <= '1';
sig_reset2 <= '1';
sig_reset3 <= '0';
end if;
Aux := Aux+1;
if(Aux = 3) then
  Aux:=0;
end if;
wait for 25600 ns;  -- espero por 256 clocks para que se cumpla el periodo
        -- del pwm
end loop;
sig_reset1 <= '1';   -- Reseteo cuando termino
```

```
        sig_reset2 <= '1';
        sig_reset3 <= '1';
wait;
 END PROCESS stimulus_process;
END arch_test_bench;
```

Esquema de conexión

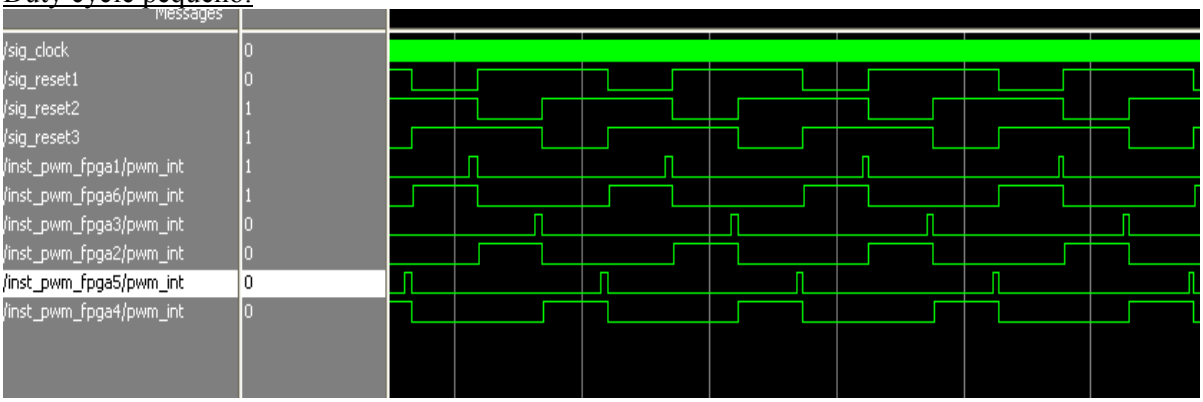
En el siguiente esquema podemos ver como es la circulación de corriente por los transistores. Vale la pena destacar que los transistores T1, T3 y T5 son los manejados por PWM, mientras que los T2, T4 y T6 actúan solamente como llaves (Pwm con 100% de duty cycle). Con color rojo se ven las líneas de conexión por donde cierran el circuito al motor y los puntos negros son los bornes de conexión para cada una de las bobinas del motor.



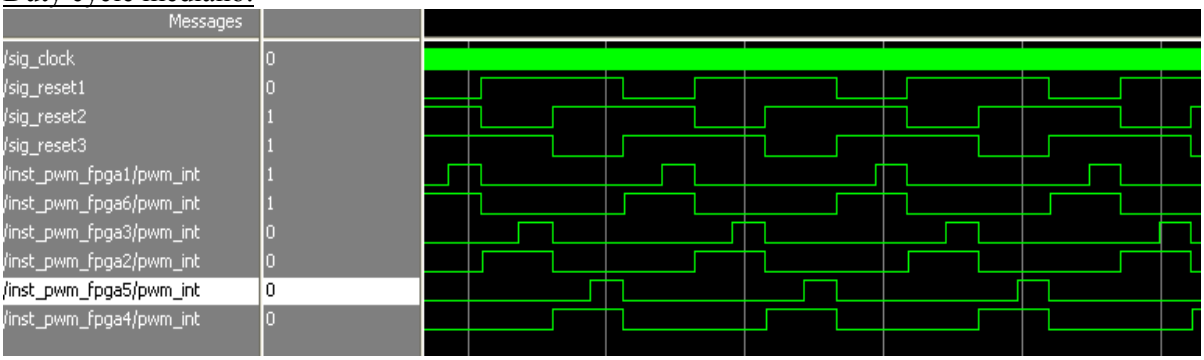
Resultados de la simulación

Como se puede apreciar en el código la simulación se realizo una rampa de valores de 0 a 255, esto hizo que no se permitiese ver todo el desplazamiento del duty cycle en la pantalla. Por lo tanto se tomaron 3 capturas, con el duty pequeño, mediano y grande. También se puede observar la habilitación de cada rama T2-T3 sig_reset2, T4-T5 sig_reset3 y T1-T6 sig_reset1.

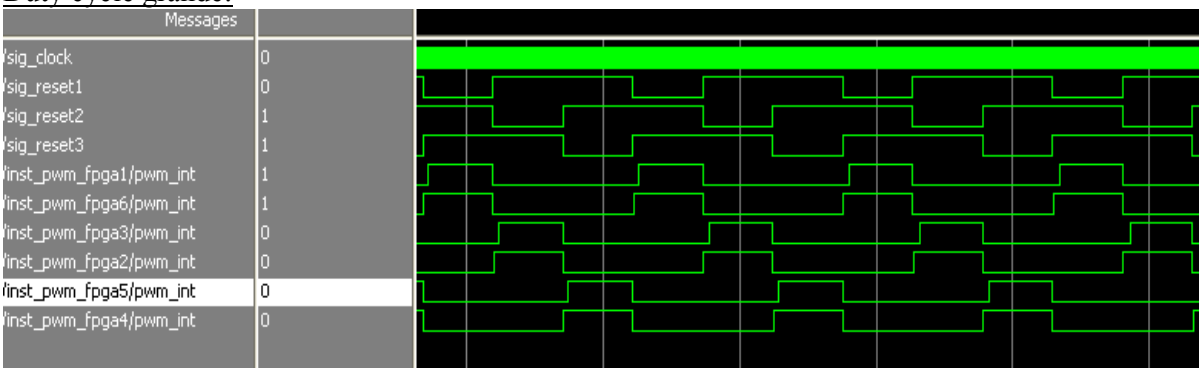
Duty cycle pequeño:



Duty cycle mediano:



Duty cycle grande:



Conclusiones finales.

Podemos concluir que el manipulador "Five Bar Linkage" es un sistema que presenta la ventaja de que al elegir correctamente la longitud de los lados que se puede operar q1 y q2 independientemente , sin preocuparse por las interacciones entre los dos ángulos. Además mediante las ecuaciones y valores de longitudes, masa y torque se puede determinar una aceleración máxima.

De la aplicación con el model Sim y VHDL, nos dimos cuenta es una herramienta de gran utilidad para programar y simular todo los que tenga que ver con FPGA, aunque no logramos extendernos con profundidad sobre él. Se logro realizar el trabajo práctico, aunque seguro que se puede optimizar el código.