

FPGA를 이용한 DC Motor의 속도제어에 관한 연구

The study on Speed Control of DC Motor using FPGA

서 용 원, 김 윤 서, 양 오

청주대학교 정보통신공학부(전화:(043)229-8440, 팩스:(043)229-8461, E-mail : ds3dhy@chongju.ac.kr)

Abstract : 본 논문은 DC Motor의 속도 제어를 위해서는 속도를 결정해주는 PWM 출력과 Motor의 속도를 측정할 수 있는 고속카운터가 필요하며 설정한 값과 실제 출력되는 값을 동일하게 만들어주는 제어부분을 구현하여야 하며 시스템을 구성하기 위한 주변 I/O도 구성되어야 한다. 기존 마이크로프로세서로 구현을 하게 되면 PWM 출력과 제어 알고리즘에 대한 연산 및 주변 I/O에 대한 구현이 용이하겠지만 DC Motor의 Encoder에서 나오는 신호를 카운터하기에는 부족한 측면이 많으며 마이크로프로세서의 연산처리 과정에 따라 제어 알고리즘 연산에 소비되는 시간도 FPGA로 구현한 시스템보단 상대적으로 여유가 없다. 본 논문에서는 FPGA만을 이용하여 PWM, HSC, PID, 주변 I/O등을 하나의 Chip에 System On Chip화함으로 실제 시스템에 적용할 때 제어시스템의 소형화와 제어대상을 고속의 정확성있는 제어시스템을 연구 하였다.

Keywords : FPGA, VHDL, PID, PWM, HSC, DAC, DC servo Motor

I. 서 론

본 논문에서 DC Motor 제어 시스템을 구성하기 위해 시스템 유저가 PC를 통해 운전목표치인 SV(Set Point Variable : DC motor의 목표속도)를 설정할 수 있도록 하는 Data 통신 및 MUX block(1)과 Plant의 운전상태를 계속하는 HSC(고속카운터) block(2)에서 측정하여 Feedback한 PV(Process Variable : DC motor의 회전속도)값을 이용하여 운전목표치인 SV와 운전상태인 PV와의 오차인 편차 E를 구하여 편차 E가 있다면 E가 없도록 편차 E와 비례한 출력을 내는 비례동작(Proportional action : P 동작)과 편차 E의 적분에 해당하는 출력을 내는 적분동작(Integral action : I 동작)과 편차 E의 미분에 해당하는 출력을 내는 미분동작(Derivative action : D 동작)을 수행하여 SV와 PV가 일치하도록 하는 PID 제어 block(3)이 필요하며 PID 제어 block에서 나온 값에 해당하는 조작량 MV(Manipulative Variable : PWM duty)를 처리하는 PWM block(4)을 구현 하였으며 Scope로 SV, PV, E를 측정하기 위해 DAC block(5)을 구성하였다.

FPGA를 이용한 하나의 Chip으로 구현하기 위해 VHDL을 이용하여 각 Block을 구현하였으며 Block간의 유기적으로 Data의 교환을 통해 DC Motor가 PC에서 설정한 속도에 맞는 안정한 상태를 유지할 수 있도록 PID제어 알고리즘을 이용하여 실시간으로 제어할 수 있는 시스템에 대한 연구를 하였다. PID제어를 수행할 때 P gain, I gain, D gain은 PC에서 통신을 통해 변화시켜가며 최적 제어가 가능하게 하였다.

본 연구는 반도체설계교육센터(IDEC)의 지원에 의한 것입니다.

II. 본 론

1 System 구성

본 논문에서는 최대 속도가 3900 RPM이며 엔코더가 432 P/R를 출력하는 DC Motor 2개를 Coupling으로 연결하여 PWM에 의한 속도 조절과 계측을 수행하였으며 PWM은 H-bridge 회로를 이용하여 구현 하였다. Data 통신, Data MUX, PID제어, HSC, PWM 및 외부에 있는 4-CH의 8bit인 Analog Devices사의 DAC(digital/analog converter)인 AD7226을 제어하는 Block들은 Xilinx사의 XC2S100-5TQ144C를 이용하여 구현 하였다.

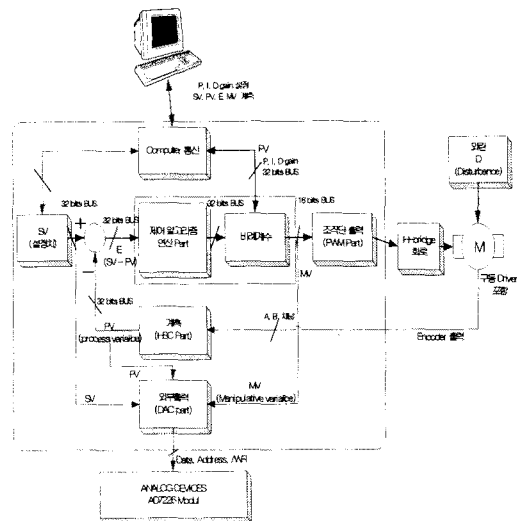


그림1. System 구성도

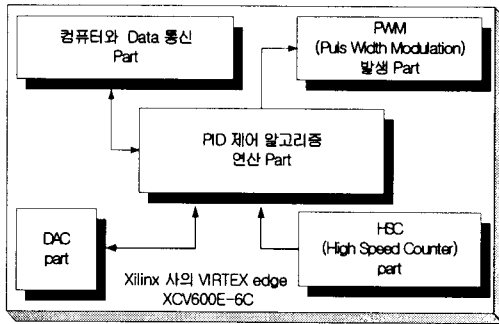


그림2. FPGA 의 내부 구성

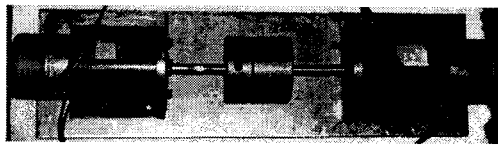


그림3. 2개의 DC Motor 연결

그림2 에서 보는 것처럼 크게 PID, HSC, PWM, Data communication and MUX, DAC 5개의 Block으로 구성되어 있으며 Plant로 사용하고 있는 Motor는 2개를 커플링으로 연결하여 한쪽은 속도를 증가하기 위해 PWM의 A,B,C,D 출력에 의한 H-bridge 회로와 연결하며 한쪽은 HSC의 ENC_A, ENC_B 입력에 연결되어 속도를 계측 할 수 있다. 그림 3에서 사용한 히타치사의 DC motor는 30V, 40W, 3900 RPM이며 432 P/R의 엔코더 출력을 갖는다.

2 PID 제어 Block

1700년 Watt의 증기기관에 사용한 Governor가 PID 제어의 기원이며 본 논문에서는 PID 제어를 Digital화하여 VHDL로 coding 하였다. SV값을 PC로부터 받고 HSC block을 통해 PV값을 얻어 편차를 구하고 PID action을 취해 MV 값을 얻어 PWM block 구동시켜 적절하게 H-bridge를 구동하는데 이러한 제어시스템의 핵심은 PID action을 취하는 부분이다. 편차 E와 비례한 출력을 내는 P action과 편차 E의 적분에 해당하는 출력을 내는 I action과 편차 E의 미분에 해당하는 출력을 내는 D action의 합으로 표현되며 PID의 기본 식은 식1은 다음과 같다.

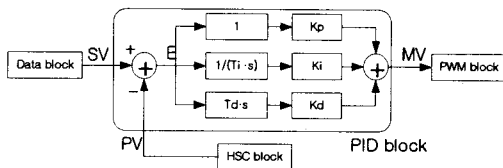


그림4. PID Block의 개념도

$$MV = K_p e(n) + K_i T_i \sum_{i=0}^n e(i) + K_d \frac{e(n) - e(n-1)}{T_d} \quad \text{--- 식1}$$

$$e(n) = SV - PV \quad \text{--- 식2}$$

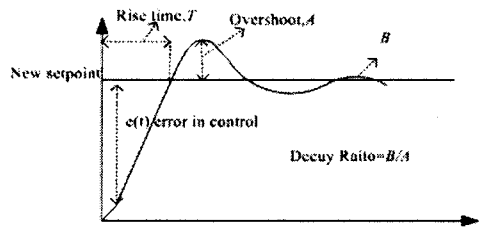


그림5. 과도상태

3 HSC Block

DC motor에서는 90°위상차의 Pulse 신호를 2 Channel에서 출력하며 속도가 증가 할수록 일정시간 동안 나오는 Pulse의 수는 증가하게 된다. 이렇게 출력되는 파형을 FPGA는 입력받아 1 clock Delay되는 신호를 만들고 A_1Q, A_2Q, B_1Q, B_2Q 신호를 만들고 NOT을 취해 A_1QN, A_2QN, B_1QN, B_2QN를 만들어 8개의 신호를 조합하여 카운트하며 회전방향도 계측한다.

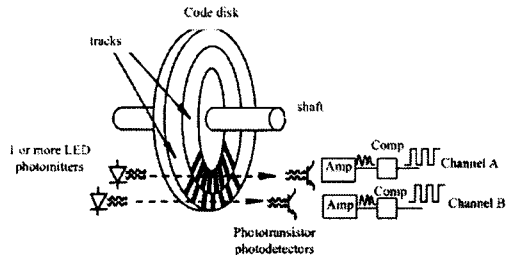
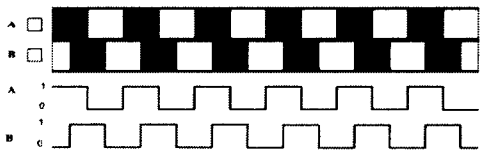


그림6. DC Motor Encoder



```
process (RSTB, SYS_CLK, ENC_AQ, ENC_BQ, A_1Q, B_1Q)
begin
```

```
if RSTB = '0' then
```

```
  A_1Q <= '0';
```

```
  A_2Q <= '0';
```

```
  B_1Q <= '0';
```

```
  B_2Q <= '0';
```

```
elsif SYS_CLK = '1' and SYS_CLK'event then
```

```
  A_1Q <= ENC_AQ;
```

```
  B_1Q <= ENC_BQ;
```

```
  A_2Q <= A_1Q;
```

```
  B_2Q <= B_1Q;
```

```
end if;
```

```
end process;
```

```
-- Encoder Pulse check
```

```
A_1QN <= not(A_1Q);
```

```
A_2QN <= not(A_2Q);
```

```
B_1QN <= not(B_1Q);
```

```
B_2QN <= not(B_2Q);
```

```
A1 <= not(A_1Q and A_2Q and B_1QN and B_2Q);
```

```

A2 <= not(A_1Q and A_2Q and B_1Q and B_2QN);
A3 <= not(A_1QN and A_2QN and B_1Q and B_2QN);
A4 <= not(A_1QN and A_2QN and B_1QN and B_2Q);
A5 <= not(A_1Q and A_2QN and B_1Q and B_2Q);
A6 <= not(A_1QN and A_2Q and B_1Q and B_2Q);
A7 <= not(A_1QN and A_2Q and B_1QN and B_2QN);
A8 <= not(A_1Q and A_2QN and B_1QN and B_2QN);

```

```

-- X_CW, X_CCW signal genertaion
process (A1, A2, A3, A4, A5, A6, A7, A8)
begin
  X_CW <= not(A2 and A4 and A6 and A8);
  X_CCW <= not(A1 and A3 and A5 and A7);
end process;

```

4 PWM Block

Pulse Width Modulation으로 삼각파 형태로 증감하는 카운터와 PID의 결과인 MV에 의해 결정된 Duty값에 의해 Ton 상태가 유지되는 Ts가 바뀐다.

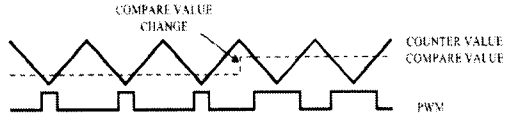


그림8. 삼각파에 의한 PWM 생성

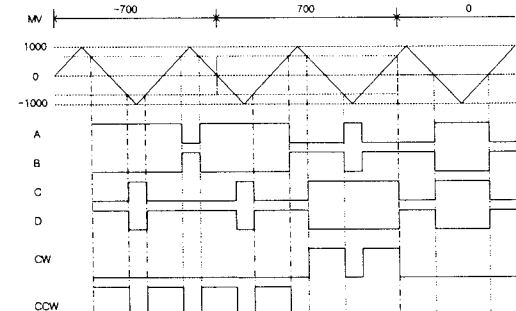


그림9 정·역방향의 DC Motor 구동을 위한 PWM

5 Data Block

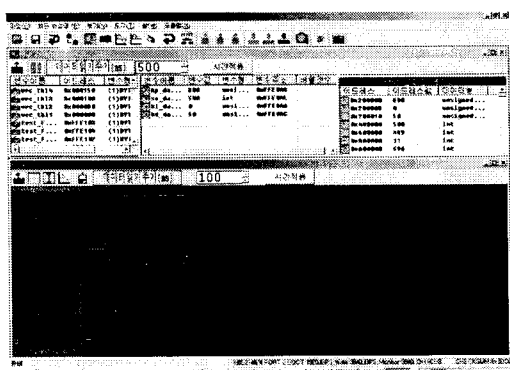


그림10 PC에서 SV, Gain설정과 PV, MV, E 계측 Data block에서는 PC에서 SV, Kp, Ki, Kd를 설정하

고 SV, PV, E, MV를 계측하고 그래프로 그릴 수 있도록 Data 통신과 MUX를 수행한다.

6 DAC Block

외부에서 SV, PV, E, MV를 오실로스코프로 계측하기 위하여 외부의 4 Channel DAC인 AD7226을 구동한다. Data Block에서는 최대 100ms 마다 계측을 하기 때문에 계측되는 값이나 그래프를 신뢰하기 힘들다. 본 논문에서는 500 MHz의 오실로스코프를 사용하여 DAC 출력값을 계측하였다.

7 H-bridge

DC Motor를 구동하기 위해서는 그림11 Q1과 Q3나 Q2와 Q4가 동시에 ON되면 되며 Q1과 Q4나 Q2와 Q3는 동시에 ON되면 회로는 파괴되므로 그림12에서와 같이 Dead time을 주어 시스템을 보호하여야 한다. 본 논문에서는 PWM block에서 Dead time을 주도록 하였다.

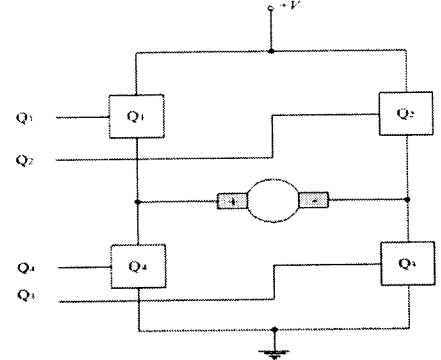


그림11 . DC motor와 H-bridge

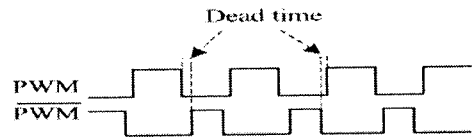


그림12 . Dead time을 준 파형

```

process(SYS_CLK, PPWM)
begin
  if PPWM = '1' then
    PPWM_cnt <= Dead_time;
    PQPWM <= '1';
  else
    if SYS_CLK = '1' and SYS_CLK'event then
      if PPWM_cnt > 0 then
        PPWM_cnt <= PPWM_cnt - 1;
        PQPWM <= '1';
      else
        PQPWM <= '0';
      end if;
    end if;
  end if;
end process;

```

8 시뮬레이션

Xilinx ISE, Synplify pro, Modelsim을 이용하여 VHDL Coding, 및 Synthesize, Implement, Simulation을 수행 하였다.

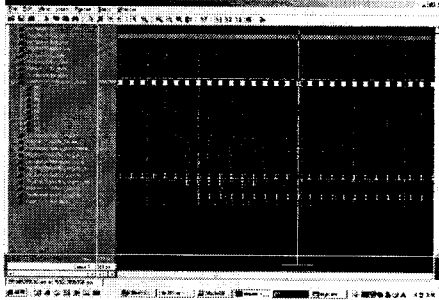


그림13. PID Block Simulation

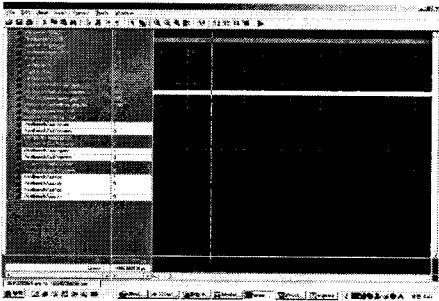


그림14. PWM Block Simulation

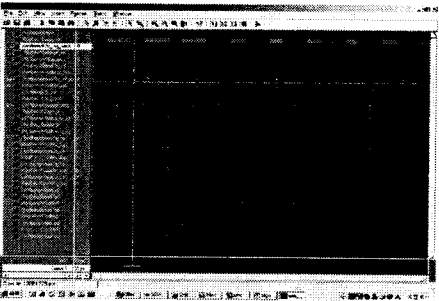


그림15. HSC Block Simulation

9 결과 파형

DAC를 이용하여 SV를 변화시킬 경우 PV의 변화를 계속하고 외부에 부하를 인가하였을 때 PV의 변화를 계속하였다. Lecroy LT334 500MHz를 이용한 결과 파형이다.

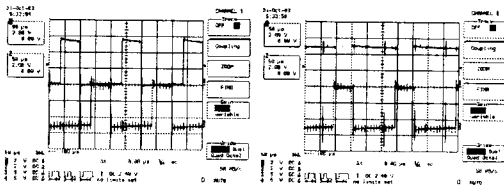
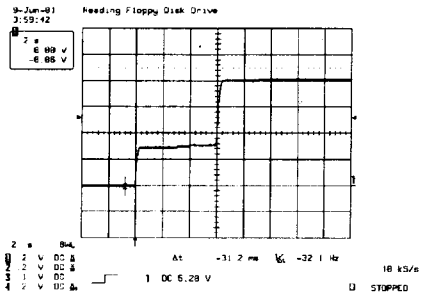
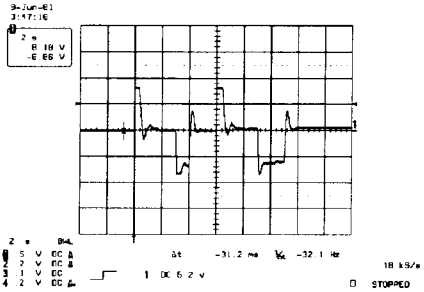


그림16. PWM과 Dead time 파형



(a) SV 변화에 따른 파형



(b) 외란에 의한 파형

그림17. PID 제어에 의한 SV, PV 파형

III. 결 론

FPGA를 이용하여 DC motor의 제어 시스템을 구성하여 속도제어를 구현할 결과 시스템의 규모가 Xilinx사의 FPGA 하나의 칩으로 구현이 가능하여 제어 시스템의 규모가 현저하게 작아 졌으며 DC motor의 Encoder에서 들어오는 고속의 펄스 신호를 4채배의 속도로 정확하게 카운트하며 정,역의 방향까지 계속함을 볼 수 있었으면 PWM도 4000 sampling이 가능하며 H-bridge 회로에서 필요한 Deadtime을 아주 용이하게 조절이 가능함을 보였다. 그리고 PID 제어부의 설계에서 연산부들이 하드웨어적으로 설계 되어져서 PID 제어에 필요한 연산인 1 clock에 수행됨을 알 수 있다.

참고문헌

- [1] 오성권, "퍼지모델 및 제어시스템", 내하출판사, 2002
- [2] 이재규, "C로 배우는 알고리즘", 세화출판사, 2003
- [3] XC2S100의 Datasheet, "www.xilinx.com", Xilinx
- [4] 양오, "AT89C51의 기초와 응용", 신화전산기획, 2002
- [5] 양오, "디지털시스템 설계 및 응용", 복두출판사, 2001
- [6] 김영철, 정연모, 조중휘, 홍윤식, "디지털 시스템 설계를 위한 VHDL", 홍릉과학출판사, 1999
- [7] 김윤서, 양오 "PD제어기와 신경망 제어기를 이용한 유도전동기의 속도제어", 대한전기학회학술대회, 2001