

Rapid Prototyping을 위한 실시간 제어시스템 개발에 관한 연구

강 문호, 정 경민, 박 윤창
선문대학교 공과대학 기계 및 제어공학부

Development of a Real-Time Control System for Rapid Prototyping

Kang, Moon-Ho, Jeong, Kyung-Min, Park, Yoon-Chang
Div. of Mechanical and Automatic Control Eng., Sun Moon Univ.

Abstract - In this research a real-time control system was developed without program codings during control system designing procedures. On the Simulink window control system is designed in the form of block diagrams, program codes are produced automatically with the real time workshop package, then C-compiler compiles the program codes. With this automatic real-time program producing mechanism rapid prototyping is realized. To show effectiveness of the proposed system designing scheme a DSP-based DC motor speed control system was constructed and PI and Fuzzy control methods were implemented.

1. 서 론

최근, 제어이론과 제어기계기술이 발전함에 따라 제어시스템이 더욱 복잡 다양화되어 미분 방정식이나 행렬식 등을 이용하여 제어시스템을 일일이 수동으로 해석 및 설계하는 것은 매우 어렵고 많은 시간이 소요되는 경우가 많다. 또한 해석방법에 있어서도 복잡한 계산뿐만이 아니라 다양한 그래프를 포함하는 사용자 인터페이스작업을 동반하는 경우가 많으며 제어시스템 설계과정에서 필수적으로 동일한 일의 반복작업이 포함되는 특성을 가진다. 따라서 이러한 문제를 해결하는 데에 있어서 컴퓨터의 사용이 필수적으로 되어 있다. 그러나 이러한 문제점들을 해결하기 위한 종래의 방법인 Fortran, C, C++ 등을 이용하여 직접 프로그래밍 하는 방식은 제어부 프로그램뿐만이 아니라 그 애플리케이션, 데이터의 입·출력, 컴퓨터와 제어기간의 통신을 담당하는 프로그래밍 등 부가적인 일에도 많은 시간을 들여야 하므로 용이한 작업이 아니다.

본 연구에서는 이러한 문제점을 해결하기 위해서 제어시스템 설계시 프로그램 코딩이 필요하지 않은 실시간 제어시스템을 개발한다. Simulink[1] 윈도우상에서 블록다이어그램 형태로 제어시스템을 설계하고, Real-time workshop(RTW)[2]을 통해 설계된 제어시스템 구현을 위한 C코드를 자동 생성시킨 후, TI사의 C컴파일러[3]를 이용하여 컴파일과 링크를 수행하고 실행파일을 생성한다. 생성된 실행파일은 노트북 PC의 프린터 포트를 통해 TMS320C31을 갖는 범용 DSP보드로 다운로드 되어져 제어작업을 수행한다. 또한 제어작업을 수행하는 도중에 RTW의 External Mode 기능을 이용하여 별도의 컴파일과 링크과정이 없이 실시간으로 제어파라미터를 변경할 수 있도록 하여 제어시스템의 해석 및 설계에 소요되는 시간을 최소화하였다. 제어작업중 발생되는 데이터들은 프린터 포트를 통해 업로드되어져 Simulink의 그래프상에서 확인할 수 있도록 하였다.

개발된 시스템의 유용성을 확인하기 위해서 DSP를 이용한 DC모터 속도제어 장치를 제작하고, 속도제어

알고리즘으로는 PI제어기법과 퍼지제어기법[4]을 선정하여 Simulink 윈도우에서 제어시스템을 설계하고 실험을 행한 후, 각각의 제어성능을 비교하였다.

2. 시스템 구성

2.1 소프트웨어 구성

제어시스템의 rapid prototyping을 위한 가장 중요한 요소는 시스템 설계시 필요한 제어알고리즘, 제어결과 분석을 위한 데이터의 처리 및 호스트(컴퓨터)와 플랜트(DSP)사이의 통신 등을 구현하기 위한 프로그램 코드를 자동으로 생성, 컴파일, 링크하므로서 기존에 수작업에 의한 처리와 비교하여 시스템 개발에 소요되는 시간을 대폭 줄이는 기능이다. 본 연구에서는 Math Work사의 Simulink와 Real Time Workshop(RTW)을 이용하여 시스템 설계시 필요한 코드들을 자동 생성한 후, 생성된 C코드들과 TI사의 C-컴파일러를 연계시켜 컴파일, 링크시킨 후 DSP에서 구동되는 실행파일을 생성한다. 이러한 일련의 과정을 보이면 Fig. 1과 같다.

시스템을 구성하는 제어알고리즘, 사용자가 작성한 I/O 인터페이스부, 인터럽트처리부 및 모듈실행부, 실험데이터 입·출력부 등이 각각 블록형태로 Simulink 윈도우상에서 설계되면, RTW는 전체시스템에 대한 C코드를 자동 생성한다. 이후, NMAKE가 생성된 코드들과 TI 컴파일러를 연계하여 DSP에서 구동될 수 있는 이진 코드를 생성하면, 이 코드들은 병렬통신 프로그램에 의해 PC로부터 DSP로 다운-로드되어 DSP를 탑재한 플랜트가 제어알고리즘에 따른 작업을 수행한다. 실험이 종료되면 실험결과 데이터는 병렬통신 프로그램에 의해 플랜트로부터 PC로 업-로드되어 Matlab의 Scope화면상에 디스플레이되고 분석된다.

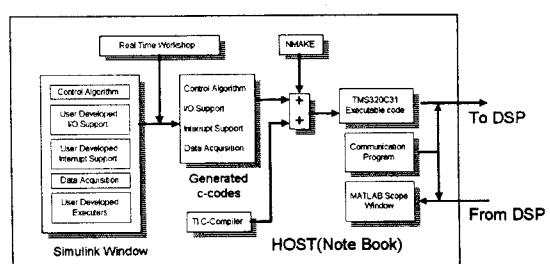


Fig.1 Automatic code generation, compile and link operations flow.

2.2 하드웨어 구성

실험에 사용된 하드웨어의 구성은 그림 2와 같다. Simulink, RTW와 C-컴파일러를 비롯한 시스템

설계에 필요한 프로그램들은 노트북에 탑재된다. RTW에 의해 C-코드들을 생성한 후 이를 컴파일, 링크시킨 후 발생되는 실행 이진 파일은 노트북의 프린터 포트를 통해 ADC, DAC 및 카운터 등의 I/O와 메모리, 주변 장치 등을 갖는 DSP(TMS 320C31)보드로 다운로드 된다. DSP는 제어알고리즘에 따라 제어입력을 계산하고 이 값은 DAC를 통해 아날로그값으로 변환되어 DC 모터 PWM 드라이브에 인가되어 DC 모터를 구동한다. DC 모터의 속도값은 타코메타에 의해 전압값으로 변환되어 DSP보드의 ADC로 귀환된다.

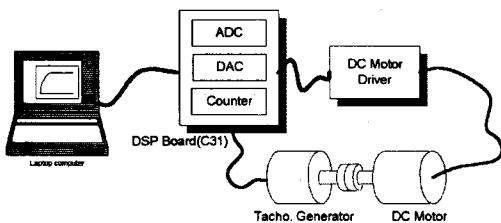
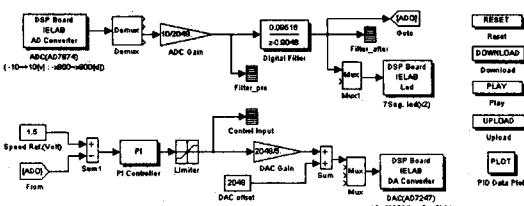


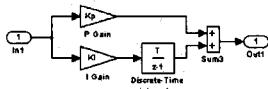
Fig. 2 Hardware configuration.

3. 실험

개발된 시스템의 유용성을 확인하기 위해서 제작된 DC모터 속도제어 장치에 PI제어기법과 퍼지제어기법을 적용한 후 실험을 통하여 각각의 제어성능을 비교하였다. Simulink 윈도우상에서 구성된 PI제어 시스템과 퍼지제어 시스템 블록도를 보이면 각각 그림 3, 4와 같다.

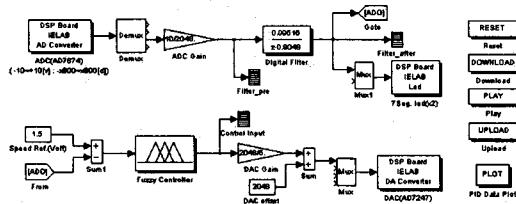


(a) PI speed control system.

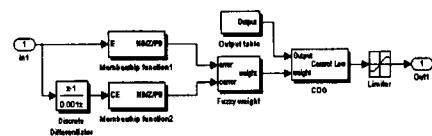


(b) PI controller in (a)

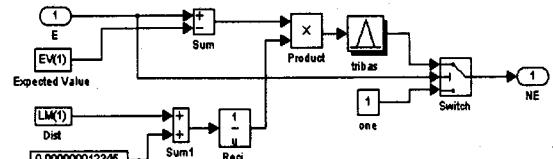
Fig. 3 Simulink model for a DC motor PI speed control system.



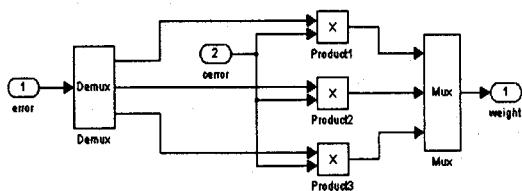
(a) Fuzzy speed control system.



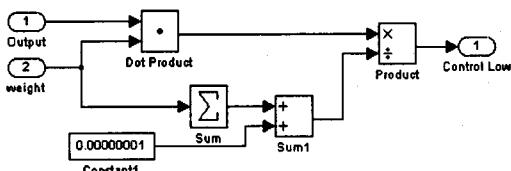
(b) Fuzzy controller in (a)



(C) Membership value of error calculator in the membership function1 block of (b)



(C) Weight of error calculator in the Fuzzy Weight block of (b)



(d) Control input calculator in the COG block of (b)

Fig. 4 Simulink model for a DC motor fuzzy speed control system.

실험 중 타코메타 출력단의 전압감쇄기(POT)에 잡음이 인가되어 정확한 속도정보를 검출하기 어려웠기 때문에 디지털 필터를 통해 POT 출력전압을 필터링한 후 제어기에 입력되는 속도정보로 사용하였다. 속도제어시 속도지령값은 1500(rpm)에 해당하는 1.5[v]로 설정하였고, PI 제어기 이득들과 퍼지제어기 파라미터들은 반복실험을 통해 최적값으로 설정하였다.

퍼지제어기는 속도오차와 속도오차의 미분을 이용하는 PD형으로 설정하였고, 속도오차와 오차미분에 대한 소속함수의 형태는 삼각형으로 설정하였으며, 퍼지제어시 필요한 속도오차의 미분값은 속도오차값을 제어주기인 1(ms)마다 미분하여 구했다. 퍼지제어기 설계시 사용된 속도오차와 속도오차 미분에 대한 소속함수의 형태, 입력 출력 퍼지-룰들을 보이면 각각 Fig.5와 Table 1과 같다. 직류전동기에 인가되는 최종 출력값은 무게중심법(COG)에 의해 산출하였다.

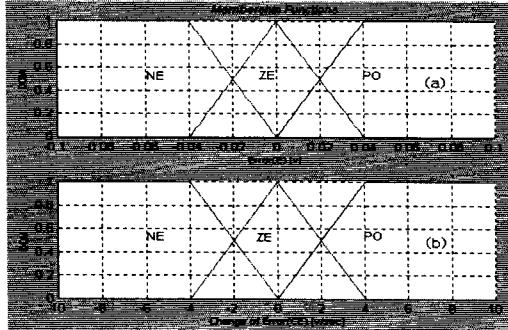


Fig. 5 Membership functions. ((a) Membership function for speed error: E, (b) Membership function for change of speed error: CE)

Table 1. Fuzzy rule base.

CE \ E	NE	ZE	PO
NE	0	0	0
ZE	0	0	5
PO	0	5	25

NE:Negative ZE:Zero PO:Positive

Fig. 6은 PI 제어에서 응답특성을 보인다. Fig. 6에서 (a)는 제어입력파형을 보이고, (b)는 약 16Hz의 밴드 폭을 갖는 디지털 필터(Fig.3, 4의 Digital filter 블록)에서 필터링되기 전의 POT 출력이고, (c)는 필터링 된 후의 POT 출력이다. 출력데이터 생성을 위한 출력데이터 변수 이름, 출력데이터 개수 등을 Fig.3에 보이는 scope 블록들을 통해 결정되며, 실험결과 데이터들은 실험 종료후 DSP보드로부터 PC로 엣-로드된 후 Matlab의 Plot함수를 이용하여 도시하였다.

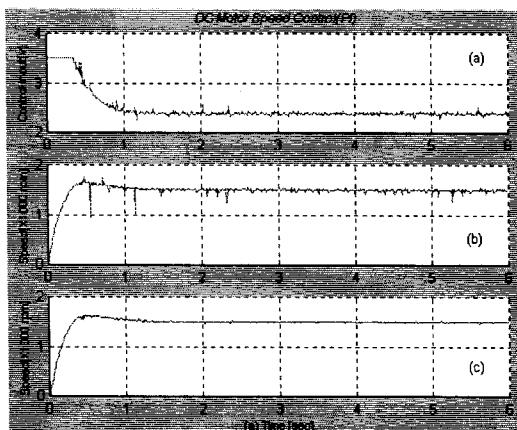


Fig. 6 PI speed control response. ((a) Control input, (b) Speed before filtering, (c) Speed after filtering)

Fig. 7은 퍼지제어시 응답특성을 보인다. (a)는 제어입력파형을 보이고, (b)는 필터링되기 전의 POT 출력이고, (c)는 필터링 된 후의 POT 출력이다. PI제어 결과와 비교하여 과도특성이 개선되지만 속도오차의 미분에 따라 측정잡음이 증가하여, 정상상태 속도의 채터링을 유발할 수 있다. Fig. 6의 경우와 마찬가지로 실험종료후 DSP보드로부터 PC로 엣-로드된 후 Matlab의 Plot함수를 이용하여 도시하였다.

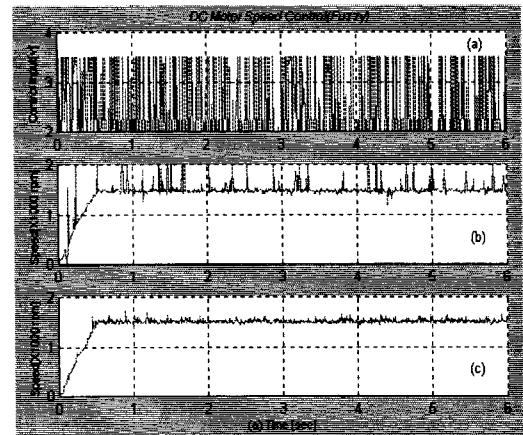


Fig. 7 Fuzzy Controller speed response. ((a) Control input, (b) Speed before filtering, (c) Speed after filtering)

4. 결 론

본 연구에서는 제어시스템 설계시 프로그램 코딩이 필요하지 않은 실시간 제어시스템을 개발하였다. Simulink 윈도우상에서 블록다이어그램 형태로 제어시스템이 설계되고, 설계된 제어시스템 구현을 위한 C코드의 생성 및 컴파일과 링크가 자동으로 이루어져 실시간 제어시스템 설계시 설계기간을 단축하고 신뢰성을 높힐 수 있는 rapid prototyping을 위한 실시간 제어시스템 설계 환경을 구축하고 TMS320C31을 갖는 범용 DSP보드와 DC 모터 속도제어 드라이브를 제작하였다.

제시된 시스템 설계방식의 유용성을 보이기 위해, 속도제어 알고리즘으로 PI제어기법과 퍼지제어기법을 선택하고 Simulink 윈도우상에서 DC 모터 속도제어 시스템을 설계한 후 실험을 통해 각각의 제어성능을 비교하였다.

(참 고 문 헌)

- [1] *Simulink, Dynamic System Simulation MATLAB*, the Math Works Inc., 1997.
- [2] *Real-Time Workshop, for Use with Simulink*, the Math Works Inc., 1997.
- [3] *TMS320C3X/4X Optimizing C Compiler User's Guide*, the Texas Instruments Inc., 1998.
- [4] L., A., Zadeh, "Fuzzy sets", *Information and Control*, vol.8, pp.338-353, 1965.