한국산학기술학회논문지
Vol. 10, No. 12, pp. 3574-3580, 2009

IEC 61131-3 국제표준언어 기반 임베디드 모션제어기의 개발

김원호[1*]
[1]동의대학교 메카트로닉스공학과

Development of an Embedded Motion Controller based on the IEC 61131-3 International Standard Language

Won-Ho Kim[1*]
[1]**Department of Mechatronics Engineering, Dong-Eui University**

요 약 본 논문에서는 IEC 61131-3 국제표준언어에 근거한 PLC 프로그래밍 개발환경을 지원해 주는 임베디드 모션 제어기를 개발하였다. 이 모션제어기에는 모션시스템 개발자가 PC 개발환경은 물론, PLC 개발환경에서도 손쉽게 모션제어 프로그래밍이 가능하도록 IEC 61131-3 개발 도구인 CoDeSys가 탑재되었으며, 직선 및 원호보간 제어와 같은 다양한 모션제어 함수들이 PLCopen 표준에 부합되는 모션 펑션블록으로 구현되었다. 또한 실시간 운영체제하에서 이더넷 기반 원격제어 및 모션 프로그래머를 위한 모션 시뮬레이터가 구현되었다.

Abstract An embedded motion controller supporting the PLC programming environment based on the IEC 61131-3 International Standard Language was developed in this paper. In this developed motion controller, the CoDeSys, one of the IEC61131-3 development tools, was embedded in order to support that of PLC as well as the development environment of the PC, and the various function blocks based on PLCopen standard for motion control such as the linear and circular interpolation control were implemented. Moreover, the ethernet based remote control on real-time operating system and the motion simulator for a motion programmer were implemented.

Key Words : IEC 61131-3, CoDeSys, Motion Control, Embedded Control System, PLCopen, Remote Control

1. 서론

PLC(Programmable Logic Controller) 시스템[1]은 로직(Logic) 제어, 모션, HMI, SCADA, 프로그래밍, 디버깅을 위한 별도의 모듈을 가지고 있는데, 이 구성은 오늘날 산업현장에서 광범위하게 사용되고 있지만, 중앙 집중식 I/O를 사용할 경우, 아주 작은 시스템을 구성하더라도 복잡하고 고가의 배선공사 비용을 지불해야 하는 문제를 안고 있다. 이에 반해 PC 기반 시스템[2,3]은 PLC 시스템에 비해 제어 구성이 더욱 간편하고 유지보수가 훨씬 수월하다는 장점이 있다. 또한 네트워크 I/O를 통해 설치와 유지보수에 필요한 노동력을 절감할 수 있으며, 배선공사가 간단해 좀 더 진보된 구조를 가지고 있다.

모션제어시스템[4]을 구성할 때, PLC 기반 제어시스템은 로직 제어와 모션 제어를 복합적으로 사용할 수 있기 때문에 다양한 시스템에 적용이 가능하며, 개발자의 입장에서는 PC기반 제어에서 사용되는 C/C++ 언어보다 프로그래밍이 쉽다는 장점이 있지만, 통신 유니트과 모션제어 유니트의 가격이 매우 비싸므로 제품의 가격 경쟁력과 확장성이 떨어지며, PLC 제조사에 제한적이라는 단점이 있다. 반면 PC 기반 모션제어기는 ISA 또는 PCI 방식으로 기존의 PLC 시스템을 대체하여 산업용 표준 PC의 하드웨어 및 소프트웨어 기술을 적용한 시스템으로서, 하드웨어 및 소프트웨어에 대한 응용 기술들이 많이 개발되고 있고, 저가의 PC 플랫폼 구입이 용이하며, 개방형 구조로 특정 하드웨어 업체에 종속되지 않으므로 시스템

본 논문은 동의대학교 교내연구비에 의하여 연구되었음(과제번호 2007AA177)
*교신저자 : 김원호(kwh@deu.ac.kr)
접수일 09년 11월 03일 수정일 09년 12월 02일 게재확정일 09년 12월 16일

의 추가나 개선작업을 쉽게 할 수 있다는 장점이 있다.

최근들어 IEC61131-3을 적용한 PLC 프로그래밍 툴에 관한 연구[5-8]가 다양하게 진행되고 발표되었지만, 이 표준을 적용한 모션제어기 개발에 관한 연구는 발표된 적이 없다.

본 논문에서는 PLC의 장점을 살리면서 PC 기반 제어기의 장점을 활용하기 위하여, PLC 및 C 언어 프로그래밍 개발 환경을 동시에 지원하게 해주는 IEC61131-3 프로그래밍[9] 시스템인 CoDeSys를 탑재한 임베디드 모션제어기를 개발하였다.

3S-Smart Software[10]의 CoDeSys[11]는 Rockwell Automations의 ICS Triplex INC ISaGRAF[12]와 더불어 가장 널리 사용되어지고 있는 IEC 61131-3 개발 도구로서, PLC, PC 및 임베디드 기반의 산업용 컨트롤러 자동화 장치에 적용이 가능하며, 다양한 산업분야의 장비 개발에 CoDeSys가 활용되고 있다.

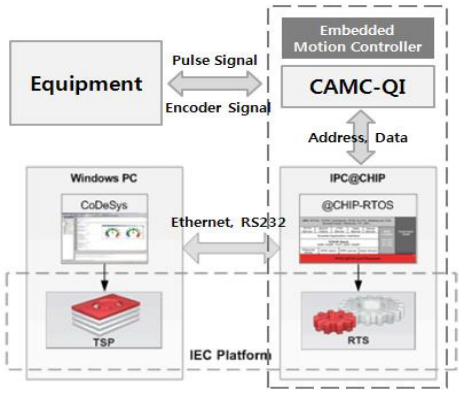
본 논문에서 개발된 제어기는 기존의 상용 모션제어기에 비해 소형이고 저가격이면서도 편리한 사용자 개발환경을 지원한다. 또한 모션제어에 요구되는 직선 및 원호보간의 모션제어 동작을 포함한 다양한 모션제어동작을 PLCopen 표준[13]에 부합되는 모션 펑션블록(Function Block)으로 구현하여 사용자가 쉽게 IEC61131-3 표준에 근거한 PLC 프로그램에 의해서 모션제어 프로그래밍이 가능하다. 또한 실시간 운영체제 환경을 지원하므로 이더넷기반 원격 감시제어가 가능하여 분산제어기로서의 적용이 가능하다. 개발된 임베디드 모션제어기를 2축 X-Y 플롯터에 적용하여 다양한 모션동작을 시험함으로써 그 성능을 검증하였다.

2. 모션제어기의 설계 및 구현

PC의 윈도우 상에서는 Codesys를 이용하여 개발한 임베디드 제어기로 모션구동을 시킬 수 있다. CoDeSys는 Controller Development System에서 나온 약성어로서, 임베디드 또는 PC기반의 산업용 컨트롤러 자동화 장치를 PLC 프로그래밍이 가능한 산업표준의 IEC61131-3 컨트롤러로 바꿔주는 역할을 한다. 그림 1은 개발한 모션제어 시스템 구성도를 나타낸 것이다. IEC 플랫폼은 BECK@IPC 모듈로 구성된 특정 하드웨어를 CoDeSys와 연결시키는 환경을 제공하는 역할을 하며 TSP(Target Support Package)와 RTS(Run Time System)로 구성되어 있다.

TSP는 CoDeSys 프로그래밍 시스템의 소프트웨어와 IPC@CHIP의 하드웨어에 대한 정보를 포함하고 있으며, 이것을 PC에 설치하면 CoDeSys에서 IPC@CHIP에 접근할 수 있다. RTS는 IPC@CHIP의 실시간 운영체제 환경에서 실행되는 응용 프로그램으로써 CoDeSys 프로그래밍 시스템과 IPC@CHIP 사이에서 통신처리를 담당하는 역할을 하며, CoDeSys의 IEC61131-3 환경에서 IPC@CHIP을 제어할 수 있도록 지원해준다.

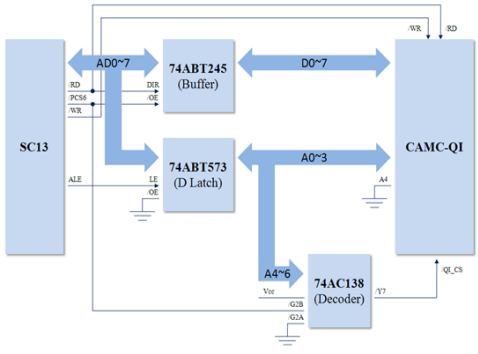
CoDeSys에서 실행시킨 PLC 프로그램은 이더넷을 통해서 임베디드 모션제어기에 전달되고 임베디드 모션제어기는 4축 보간 정밀모터제어칩인 CAMC-QI[14]의 제어에 의해 위치지령 만큼의 펄스신호를 출력시켜 장비를 구동시키게 된다.



[그림 1] 개발된 모션제어 시스템의 구성도

2.1 하드웨어 설계

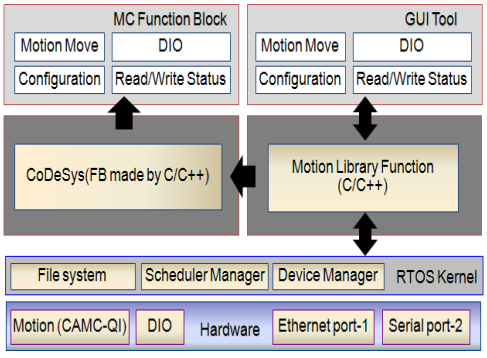
임베디드 모션제어기는 i80186 CPU를 내장한 IPC@CHIP인 SC13 모듈[15]과 CAMC-QI 칩을 사용하여 구성하였다. SC13은 어드레스 버스와 데이터 버스를 이용하여 CAMC-QI에 구동설정 및 구동명령을 내리면 CAMC-QI에서는 설정된 목표 속도와 목표 위치만큼의 펄스를 출력하여 모터를 구동시키게 된다.



[그림 2] 임베디드 모션제어기의 하드웨어 구성도

2.2 소프트웨어 설계

그림 3은 개발된 모션제어기의 소프트웨어 구성도이다.



[그림 3] 모션제어기의 소프트웨어 구성도

SC13에 실시간 운영체제(RTOS) 커널을 이식하여 실시간 동작이 가능하고 이더넷 통신을 지원한다. 모션함수는 Borland C++환경에서 구현되었으며, 구현된 모션함수에 대한 동작 확인은 호스트 PC에서 GUI를 통하여 확인이 가능하도록 설계하였다. 모션제어를 위한 PLCopen 펑션블록들은 Borland C++환경에서 구현된 모션함수를 바탕으로 CoDeSys환경에서 개발되었으며, 이 펑션블록들을 이용하여 파라미터 설정, 디지털 입출력(DIO)제어, 모션 구동 명령 프로그래밍이 가능하다. 모션 펑션블록은 내부적으로는 C언어로 개발되었으며, CoDeSys에서 ST언어를 사용하여 펑션블록의 입출력 변수를 선언하고 외부 라이브러리 형식의 확장자로 저장을 하면 C파일, 헤더파일, lib파일이 하나의 프로젝트로 생성된다. 생성된 C파일은 Borland C++ 환경에서 lib 파일을 통해 ST언어로 선언했던 입출력변수들을 포인터를 이용해서 접근이 가능하다.

2.2.1 C/C++기반 모션함수의 구현

개발 제어기는 CAMC-QI 칩을 이용하여 모션제어동작을 구현하였으므로, 모션제어관련 함수를 Borland C++ 환경에서 CAMC-QI 칩을 제어하는 모션 라이브러리로 구현하였다. 이렇게 함으로써 CoDeSys를 이용하여 펑션블록과 연결시키는 작업도 단순화시킬 수가 있다.

표 1은 Borland C++에서 구현한 모션구동관련 함수를 나타낸 것이다.

[표 1] 구현된 모션구동 관련 주요 함수

No	Function
1	SignalSetHomeLevel(int AxisNo, unsigned Level)
2	SignalSetLimit (int AxisNo, unsigned int StopMode, unsigned int PositiveLevel, unsigned int NegativeLevel)
3	SignalServoOn (long AxisNo, unsigned int OnOff)
4	SignalServoAlarmReset(int AxisNo, unsigned int OnOff)
5	MovePos(int AxisNo, float Pos, float dVel, float Accel, float Decel)
6	LineMove(int Coord, float *Pos, float Vel, float Accel, float Decel)

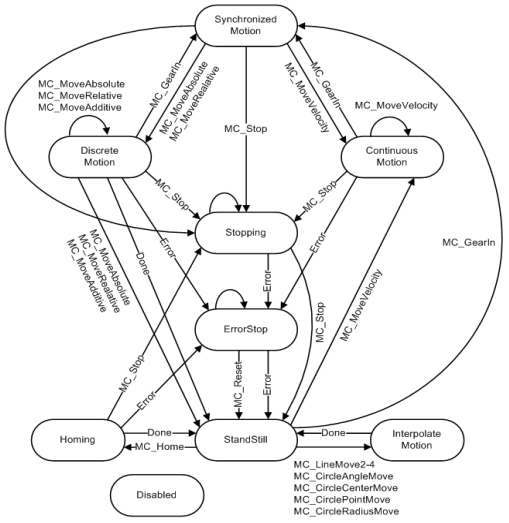
2.2.2 모션 펑션블록의 구현

표 2에 PLCopen 표준에 부합되도록 구현된 펑션블록을 나타내었다.

[표 2] 구현된 펑션블록

Function Block List	
MC_MoveAbsolute	Absolute Position Drive
MC_MoveRelative	Relative Position Drive
MC_LineMove2	Linear interpolation drive - 2 axis
MC_CircleCenterMove	Circular interpolation drive (start/center/end point)
MC_Stop	Stop Drive
MC_Reset	Reset if Error occur
MC_ServoOnOff	Turn On/Off Servo Drive

그림 4는 PLCopen에서 사용되고 있는 상태 천이도를 나타낸 것이다. 모션에 대한 모드 전환이 어떠한 펑션블록을 이용하여 전환이 되며, 에러가 발생 시 진행이 되는 순서를 상세히 나타내고 있으며, 이에 근거하여 펑션블록을 개발하였다. 펑션블록의 실행은 일반적으로 StandStill(정지) 상태로부터 시작되거나 원점구동명령(Homing)이 완료된 후 시작된다. 원점구동이 완료되면 다시 StandStill 상태로 전환된다.



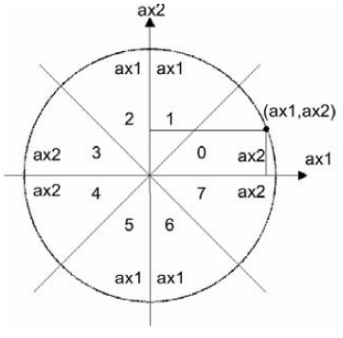
[그림 4] PLCopen 상태 천이도

(1) MC_LineMove

직선 보간을 수행하는 이 펑션블록은 현재좌표에 대한 종점좌표를 설정하고, 2축 직선보간 명령을 입력하면 실행된다. 종점좌표는 현재 위치에 대한 상대 값으로, 각각의 축의 출력 펄스수로 정한다.

(2) MC_CircleCenterMove

원호 보간은 현재 좌표에 대한 원호의 중심좌표 및 종점좌표를 설정하고, 원호보간 명령을 입력하는 것으로 실행된다. 그림 5에 0상한에서는 원호상을 이동하는 보간좌표(x,y)는 항상 Y축의 절대치가 X축의 절대치보다 작게 된다. 절대치의 값이 작은 축을 단축이라 하면 1,2,5,6 상한은 X축이 단축이 되고 0,3,4,7상한은 Y축이 단축이 된다. CAMC-QI는 내부적으로 단축은 그 상한 사이에 드라이브 펄스를 출력하고, 장축은 원호보간 연산결과에 따라 펄스를 출력하거나 출력하지 않도록 원호보간 알고리즘이 이미 구현되어있다[16].



[그림 5] 원호보간 연산의 0~7 상한과 단축

아래의 수식은 MC_CircleCenterMove 펑션블록 내부에 적용된 CAMC-QI를 이용한 알고리즘을 나타낸다. 원을 그리기 위해서는 각각의 분원에 대한 단축의 길이를 이용하여 전체 펄스 수를 계산하고 모션제어 칩 레지스터에 써넣어야 한다. 원에 대한 전체 펄스 수를 계산하기 위해서는 전체 라디안(구동 시작점에서 종점을 이루는 각) 값을 계산해야 한다. 그림 6과 같이 시작점과 종점이 서로 다른 분면에 위치할 때, 전체 라디안(radian)을 계산하는 방법은 달라지며 이에 대한 알고리즘은 아래와 같다.

① 현재위치의 원점에 대한 라디안 값 :

Case 1 : CenterY > 0

$$SRadian = acos(\frac{CenterX}{Radius}) + \pi \quad (1)$$

Case 2 : CenterY < 0

$$SRadian = acos(-\frac{CenterX}{Radius}) \quad (2)$$

Case 3 : CenterY = 0 and CenterX < 0

$$SRadian = 0 \quad (3)$$

Case 4 : CenterY = 0 and CenterX > 0

$$SRadian = \pi \quad (4)$$

② 종점위치의 원점에 대한 라디안 값 :

Case 1 : CenterY > EndY

$$ERadian = acos(-\frac{EndX - CenterX}{Radius}) + \pi \quad (5)$$

Case 2 : EndY > CenterY

$$ERadian = acos(\frac{EndX - CenterX}{Radius}) \quad (6)$$

Case 3 : CenterY = EndY and CenterX < EndX

$$ERadian = 0 \quad (7)$$

Case 4 : CenterY = EndY and CenterX > EndX

$$ERadian = \pi \quad (8)$$

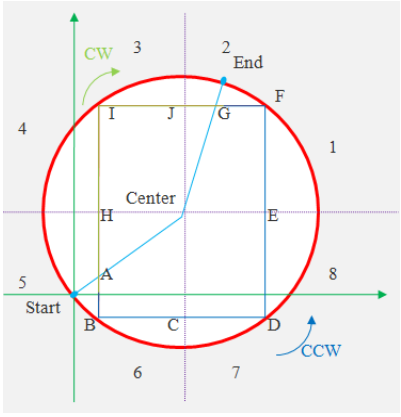
③ 구동방향이 CW이면:

```
case (SRadian > ERadian):
        TotalRadian=|SRadian-ERadian|
case (SRadian < ERadian):
        TotalRadian=360-|SRadian-ERadian|
case (SRadian == ERadian):
        TotalRadian=360
```

④ 구동방향이 CCW이면:

```
case (SRadian > ERadian):
        TotalRadian=|SRadian-ERadian|-360
case (SRadian < ERadian):
        TotalRadian=|SRadian-ERadian|
case (SRadian == ERadian):
        TotalRadian=360
```

8등분으로 나눈 분원에서 1,2,5,6 상한은 X축이 단축이 되므로 이 단축 길이의 합이 원호에 대한 총 펄스가 되고, 0,3,4,7상한은 Y축이 단축이 되므로 이 단축 길이의 합이 원호에 대한 총 펄스가 된다. 만약 시작점과 종점사이의 라디안이 45° 이하일 경우는 해당 분면에 대한 단축 길이의 합이 원호에 대한 펄스 값이 되고, 45° 이상일 경우에는 원을 그리는 방향에 따라 아래와 같이 계산이 이루어진다.



[그림 6] 원호 중심점 보간 설명하기 위한 그림

CCW 방향일 경우,

$$TotalPulse = AB + BC + CD + DE + EF + GF$$
$$= (\frac{R}{\sqrt{2}} - |Center\,Y|) + \frac{R}{\sqrt{2}} + \frac{R}{\sqrt{2}} + \frac{R}{\sqrt{2}} + \frac{R}{\sqrt{2}} + (\frac{R}{\sqrt{2}} - |EndX - Center\,X|) \quad (9)$$

CW 방향일 경우,

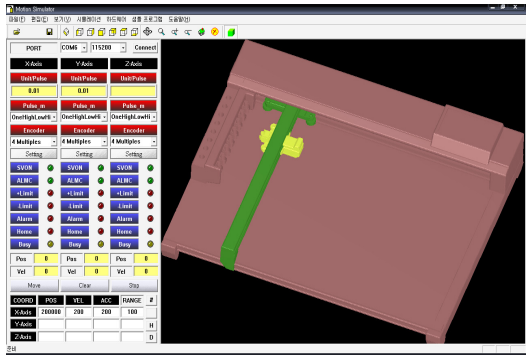
$$TotalPulse = AH + HI + IJ + JG \quad (10)$$
$$= |Center\,Y| + \frac{R}{\sqrt{2}} + \frac{R}{\sqrt{2}} + |EndX - Center\,X|$$

위의 공식들은 CAMC-QI의 원호 보간에 대한 레지스터에 데이터를 쓰기 위해 CAMC-QI 칩의 데이터 시트(data sheet)에 의해, 계산한 공식이며 "MC_CircleCenterMove" 펑션블록 내부에 C 언어로 구현하였다.

2.2.3 원격 감시제어 및 모션 시뮬레이터의 구현

호스트 PC에서 원격으로 개발된 모션제어기에 접속한 후, CoDeSys를 이용하여 C/C++ 언어나 PLC 언어로 프로그래밍한 실행파일을 전송하여 장비를 구동할 수도 있으며, 모션제어기에 연결된 장비의 현재 환경변수 상태, 구동상태 등을 감시할 수도 있도록 원격 감시제어 기능을 구현하였다. 또한 실제 구동할 장비가 없이도 모션 프로그래밍에 의한 장비의 구동을 가상으로 먼저 확인할 수 있는 모션 시뮬레이터를 구현하였다.

그림 7은 구현된 원격 감시제어 기능과 모션시뮬레이터를 이용하면 모션 프로그래밍에 의한 XY 플롯터의 동작을 가상으로 확인한 화면이다.

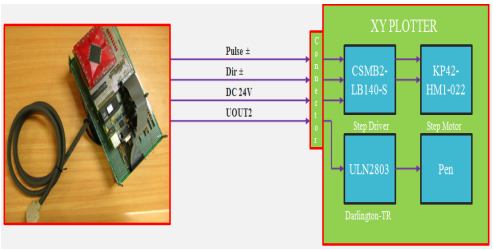


[그림 7] 원격 감시제어 기능과 모션시뮬레이터에 의한 XY 플롯터의 가상 동작 화면

3. 모션 펑션블록을 이용한 성능 실험

3.1 실험 환경

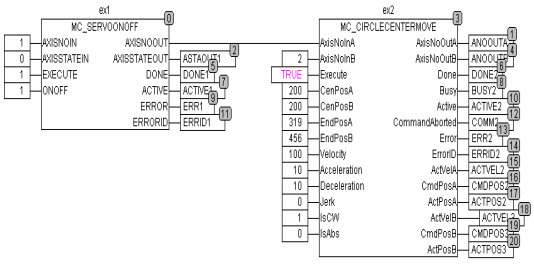
개발된 임베디드 모션제어기를 X-Y 플로터에 적용하여 상용 PC기반 모션제어기(PCI-N404) [17]와 성능을 비교하여 보았다. 아래의 그림 8은 임베디드 모션제어기와 X-Y 플로터 간의 연결도를 나타낸 것이다. 임베디드 모션제어기에서는 PULSE, DIR 핀을 통해서 X-Y 플로터 각 축에 해당하는 모터 드라이브로 입력이 되며 PELM, NELM 핀은 기구의 파손을 막기위해 각 축에 해당하는 기구적 한계부분에 연결된 센서와 연결되어 있다. ECA, ECB, ECZ 신호는 엔코더 A, B, Z상 신호로써 고속 포토커플러를 통하여 모션제어기의 CAMC-QI 칩과 연결된다.



[그림 8] 모션제어기와 X-Y 플롯터의 연결선도

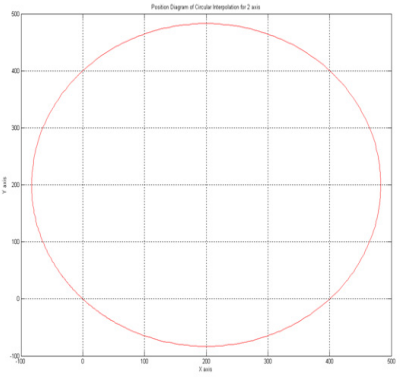
3.2 실험 결과

개발된 임베디드 모션제어기에 의해 표 2에 구현된 모든 펑션블록에 대해 실험을 실시하여 정상적인 동작을 확인하였다. 그림 9는 구현된 모션 펑션블록 중에서 "MC_CircleCenterMove"를 이용하여 모션 구동 실험을 한 것으로서, CoDeSys에서 구현된 모션 펑션블록을 이용하여 PLC 프로그래밍을 한 화면을 보여준다.



[그림 9] MC_CircleCenterMove 펑션블록의 실행 화면

CoDeSys의 "Single Cycle" 기능을 이용하여 모션구동이 실행되는 동안 X, Y 좌표에 대한 데이터를 획득하였고, 그림 10과 같이 원호보간 움직임에 대한 궤적을 확인하였다.



[그림 10] MC_CircleCenterMove 실행 결과

실험 결과, PLC 프로그래밍에 의해 동작하는 임베디드 모션제어기의 성능은 CAMC-QI를 사용하고 C-프로그래밍에 의해 동작하는 상용 PC기반 모션제어기의 성능과 같음을 확인하였다. 즉, 개발된 임베디드 모션제어기와 상용 PCI-N404 모션보드가 모두 CAMC-QI에 의해 좌우되므로 모션 구동 성능은 같지만, PCI-N404 보드로 모션 시스템을 구성할 때보다 개발된 임베디드 모션제어기로 모션시스템을 구성하면, 보다 소형이면서 저가격으로 시스템을 구성할 수 있을 뿐만 아니라, 사용자의 개발환경 측면에서도 장점을 가지게 된다.

4. 결론

본 논문에서는 IEC 61131-3 국제표준언어에 근거한 PLC 프로그래밍 환경을 지원해 주는 임베디드 모션제어기를 개발하였다. 본 논문에서 개발된 임베디드 모션제어기는 기존의 PLC 및 PC 기반 모션제어기에 비해 소형이면서 저가격이고, 사용자 입장에서는 C/C++ 프로그래밍 개발 환경은 물론, IEC 61131-3 표준의 PLC 프로그래밍 개발 환경도 지원한다는 장점이 있다. 또한 개발 제어기에는 실시간 운영체제 환경에서 이더넷 기반 원격제어 기능 및 모션 프로그래밍 개발자를 위한 모션 시뮬레이터 기능이 구현되었다.

개발된 제어기는 현재 펄스타입 모터제어칩인 CAMC-QI를 이용하여 모션제어 기능을 구현하였으므로 위치제어용 모션제어기로 개발되었다. 향후 본 논문에서 개발된 기술을 이용하여 CoDeSys를 DSP에 이식하고, 펄스 출력뿐만 아날로그 출력을 지원하는 DSP 기반 모션제어기를 개발하면, 저가의 PLC 개발환경을 지원하는 속도 및 토크제어용 고성능 모션제어기로 개발할 수 있을 것이다.

참고문헌

[1] John R. H. and Frederic D. H., "Programmable Logic Controllers : Programming Methods and Applications", Prentics Hall, 2002.

[2] 김원호,최성혁, “PC기반 제어기술과 활용”, 제어계측, pp.49-55, 1월, 2004.

[3] 김원호,최성혁, “산업용 PC, CompactPCI 기반 전자제어시스템”, 전자산업정보, pp. 42-44, 8월, 2003.

[4] Chip McDaniel, "Motion control system choices", AutomationDirect, April, 2005.

[5] 정구,심주현,이제필,이철수,“제어시스템 설계를 위한 IEC1131-3 기반의 제어로직생성기의 개발”,한국공작기계학회 춘계학술대회논문집, pp171-176, 2001.
[6] 이철수,정구,이제필,심주현, "IEC61131-3 표준을 따른 PC용 소프트웨어 PLC의 개발", 한국공작기계학회논문집, Vol.11, No. 1, pp. 61-69, 2월, 2002.
[7] 조영임, "IEC1131-3 표준언어처리를 위한 지능적 소프트웨어 PLC의 개발", 퍼지및지능시스템학회논문지, Vol.14, No. 2, pp. 207-215, 2004.
[8] 김태형,엄태진,장원지,홍대선,“윈도우 기반 PLC 구동 프로그램 개발”, 한국정밀공학회지, 1999.
[9] IEC, "Programming Industrial control systems using IEC 1131-3", 1995.
[10] http://www.3s-software.com
[11] Joel Ek,“CoDeSys on CCP XS for BROMMA CONQUIP",Master Thesis, Malardalen University, 2006.
[12] http://www.isagraf.com
[13] http://plcopen.org, "PLCopen Standardization in Industrial Control Programming".
[14] http://www.ajinextek.com, "CAMC-QI Hardware Chip User Manual".
[15] http://www.beck-ipc.com, "Information Sheet: IPC@CHIP@ SC13".
[16] 김원호,신진호,“고정밀 모션 제어기 ASIC용 개선된 DDA 보간 알고리즘에 관한 연구”, 한국정보기술학회 논문지, 제7권1호, 2월, 2009.
[17] http://www.ajinextek.com, "PCI-N404 Hardware User Manual".

김 원 호(Won-Ho Kim) **[정회원]**

- 1988년 2월 : 경북대학교 전자공학과 (공학석사)
- 1999년 8월 : 경북대학교 전자공학과 (공학박사)
- 1988년 6월 ~ 1993년 2월 : 한국전자통신연구소(ETRI) 연구원
- 1995년 9월 ~ 현재 : 동의대학교 메카트로닉스공학과 교수

<관심분야>
임베디드시스템, 정밀모션제어, 로봇제어시스템