論文 한국공작기계학회지 제8권 제5호 1999. 10.
Journal of the Korean Society of Machine Tool Engineers

실시간 제어에 의한 개방형 CNC 소프트웨어 모듈의 설계 및 구현

이철수*, 이제필*

The Design and Implementation of Open Architecture CNC Software Module by a Real-time Control

Cheol-Soo Lee*, Je-Phil Lee*

Abstract

This paper describes the design and implementation of a PC(personal computer) based open architecture machine tool controller. The hardware of open architecture CNC has generally a motion control board on a PC for controlling a servo motor. But this paper describes open architecture hardware that consists of a PC, a counter board, a DAC board, and a DIO board only. This makes it easy to generate CNC software module in a hardware-independent way. The proposed open architecture CNC software runs on the MS-Windows NT. The paper describes a method of controlling servo motors using a real-time timer of MS-Windows NT and a commercial real-time operating system on the MS-Windows NT. An open and reconfigurable software module is made up of an object and an API(application programming interface). Using the object and the API, a new CNC system can be quickly configured to control different machine tools. The proposed open architecture CNC system is applied to 4-axis lettering center.

Key Words : Open Architecture CNC(개방형 구조의 CNC), Real-time Operating System(실시간 운영체제), Personal Computer(개인용 컴퓨터)

1.서 론

개방형 CNC 시스템은 현재 산업계의 관심이 되고 있다.[1~3] CNC 공작기계 사용자의 다양한 요구 때문에 CNC 공작기계 업체의 전통적인 폐쇄형 하드웨어와 소프트웨어 구조를 가지고는 신속히 대응하기가 어렵게 되었다. 즉, CNC 시스템 개발 업체만의 독특한 하드웨어를 구성하여 제작하는 경우 하드웨어에 대한 부담이 클 뿐만 아니라 소프트웨어 개발에 있어서도 많은 노력이 필요하였다. CNC 공작기계 업체는 하드웨어는 물론 소프트웨어

* 전남대학교 산업공학과

까지도 기계의 부품을 조립하듯 각각의 기능을 갖는 소프트웨어 모듈 즉, 객체(object) 또는 API(application programming interface)를 필요한 기능에 따라 조립하여 완성할 수 있는 CNC 시스템을 요구하게 되었다. 이러한 시스템은 제품의 다양성은 물론 제품 개발 기간을 단축할 수 있으며, 각 모듈들은 이미 검증된 것들이므로 시스템의 신뢰성도 간단한 검증만을 통하여 확인할 수 있는 장점을 가진다.

개방형 CNC 시스템을 구성하기 위해서는 대부분 하드웨어를 모듈화하고 메인 시스템과 각 모듈들을 직접 제작하거나 현재 상용화된 하드웨어를 사용한다. 특히, 메인 시스템으로 일반 개인용 컴퓨터(personal computer)를 사용하는 것은 다른 하드웨어 모듈을 선택할 수 있는 폭을 넓혀주며 소프트웨어를 제작하는 데에도 많은 장점을 갖는다.

본 논문에서는 PC를 기본 하드웨어 플랫폼으로 하고 기계와의 인터페이스 만을 위하여 하드웨어 모듈들을 사용한다. 따라서 서보 모터를 제어하거나 입출력 접점을 제어하기 위해서는 운영체제인 MS-Windows NT에서 제공되는 멀티태스킹(multitasking)과 실시간 타이머(real-time timer) 등의 기능을 이용하여 처리하는 방법을 제안한다. 또, 운영체제에서 제공되는 실시간 타이머는 약간의 오차를 가지므로 정밀한 가공이 필요한 경우에는 상업용 실시간 운영체제(real-time operating system)를 부가하여 정밀도는 높이도록 한다.

현재 CNC 시스템의 개방형 구조에 관한 많은 연구가 이루어지고 있다. 이러한 CNC 시스템의 개방화는 세계적으로 OSACA[1], OMAC[2], OSEC[3] 등의 컨소시엄이 조직되어 각자의 개방화 정도를 설정하고 기계에 적용하는 등의 개방화가 진행중이다.

OSACA 프로젝트는 OAC(open architecture control)와 관련된 하드웨어, 소프트웨어, 네트워크를 포함한 대부분의 표준화 문제를 다룬다.[4-5] HOAC-CNC(hierarchical open architecture control multi-processor CNC)은 ISA Bus를 통하여 하드웨어를 기능별로 모듈화 하고 각 모듈별 소프트웨어를 제작한다.[6] 정밀도나 생산성의 관점에서 제어 구조나 통신 네트워크가 제조 시스템의 성능에 미치는 영향을 분석한다.[7] 또 개방형 시스템 구조에서 가공정보를 데이터베이스화하고[8-9], 서보 모터 제어 부분을 개방형 구조로 설계하였다.[10] 이들 시스템은 PC에 이송 축의 제어를 위한 모션 제어 보드(motion control board)를 장착하여 제작하였다.[11-12] 현재 상용화된 모션 제어 보드로는 삼성전자[13]와 MEI(Motion Engineering Inc.)[14], Delta-Tau[15] 등에서 개발한 상품이 있는데, 이들은 DSP(digital signal processing)칩 또는 MC68030을 이용하였으며, 이송 축을 제어할 수 있는 API를 제공한다.

2. 제안된 개방형 CNC의 하드웨어 구성

제안된 개방형 CNC 하드웨어는 PC 플랫폼에 카운터 보드(counter board), DAC 보드(digital analog converter board), DIO 보드(digital input output board), LAN 보드(local area network board) 등으로 구성되었다. 이 시스템은 한 개의 CPU에서 NC 코드를 해석하여 이송 코드로 만들고, 이를 서보 모터에 지령하여 원하는 궤적을 이동하도록 한다. 따라서 고속의 CPU가 필요한데, 꾸준히 업그레이드 되는 PC 플랫폼은 CNC 시스템의 이러한 요구를 잘 만족시켜 주어 하드웨어 개발에 대한 부담을 줄여주는 장점을 갖고 있다.

Fig. 1에 나타난 것처럼 카운터 보드는 각 축의 서보 모터에 장착되어 있는 회전 엔코더(rotary encoder) 신호를 받아 서보 모터의 회전수를 계산함으로써 각 축의 현재 위치를 파악한다. DAC 보드는 CNC 시스템에서 서보 모터를 속도제어(velocity control) 하기 때문에 실행주기(sampling time) 동안의 각 축의 이송량을 아날로그 전압으로 변환하여 서보 드라이브(servo drive)에 지령함으로써 최종적으로 서보 모터의 운동을 제어한다. DIO 보드는 기계와의 디지털 접점 제어를 위하여 사용된다. 하지만 기계의 스위치나 센서 등에 직접 연결할 수는 없으므로 DIO 보드의 입력은 입력 단자대(digital input panel), 출력은 출력 단자대(relay actuator panel)를 통하여 각종 센서나 스위치 등에 연결된다.

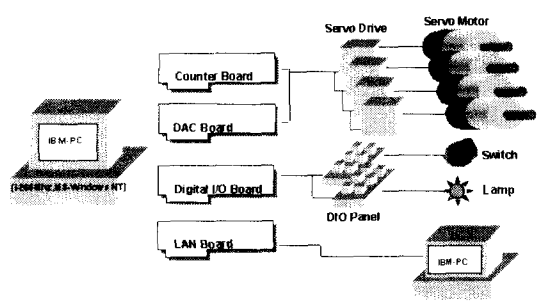


Fig. 1 Hardware architecture of an open CNC system

3. 개방형 CNC의 실시간 제어

3.1 실시간 시스템 및 필요성

CNC 소프트웨어의 운영체제는 내부 데이터의 흐름들을 원할히 처리하기 위하여 실시간 멀티태스킹 기능을 대부분 가지고 있다. 만약 CNC 소프트웨어가 실시간 멀티태스킹 운영체제를 가지고 있지 않으면, 공작기계의 응용작업에서 발생될 수 있는 많은 이벤트들의 실시간 처리와 공작기계에 필요한 다양한 동작을 구현하기 위해서는 하드웨어 인터럽트를 사용하여야 하므로 응용 프로그램의 작성을 어렵게 만든다.

본 논문에서는 실시간 멀티태스킹, 실시간 반응성(real-time responsiveness) 등의 문제를 해결하기 위하여 개방형 CNC 시스템의 운영체제로 MS-Windows NT 4.0을 사용하였다. PC 플랫폼의 CNC 시스템에서 MS-Windows NT를 운영체제로 선택하는 것은 많은 장점을 가진다. 첫째는 최신 기술이 적용되어 있으며, 주변 환경에 빠르게 적응한다는 점이다. 둘째는 다른 클라이언트들과 동일한 네트웍 프로토콜, 인터페이스와 API를 사용한다. 셋째는 개방적인 하드웨어 구조를 갖은 것이다. 넷째는 업무용 응용프로그램은 물론 Open GL과 같은 CNC의 소프트웨어를 구성하기에 적합한 다양한 도구가 존재한다. 다섯째는 소프트웨어 개발의 효율성이다. 목표가 되는 시스템 환경에서 직접 개발을 함으로써 개발 기간을 단축할 수 있다. 여섯째는 운영체제의 존속에 대한 믿음이다.

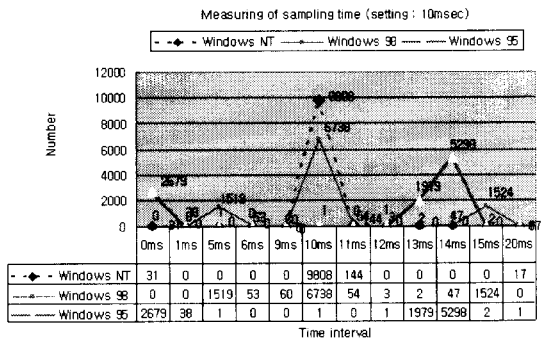
그러나 MS-Windows NT는 완벽한 선점 방식의 멀티태스킹과 타이머 또는 실시간 타이머로 이벤트를 처리할 수 있지만, 완벽한 실시간 타이머는 제공되지 않으므로 CNC 시스템에서 서보 모터를 제어하거나 기계를 제어하는 PLC(programmable logic controller)를 처리하기에는 적합하지 않다. 따라서 본 논문에서는 이에 대한 해결 방법을 제안하고, 보다 엄격한 실시간 반응성(hard real-time responsiveness)을 위하여 MS-Windows NT에 상업용 실시간 운영체제를 추가하여 처리한다.

3.2 실시간 제어를 위한 각 운영체제의 성능 실험

MS-Windows NT는 완벽한 실시간 타이머가 제공되지 않으므로 일정시간 간격으로 특정 기능을 연속적으로 수행해야 하는 경우에는 바람직한 결과를 얻지 못한다.

Fig. 2는 다음의 프로그램으로 MS-Windows NT와 98, 95에서 실시간 타이머의 정확도를 실험한 결과이다.[16] MS-Windows NT와 98, 95의 멀티태스킹은 프로세스(process)와 쓰레드(thread)의 개념으로 구현되는데, 보통 태스크는 한 개의 프로세스와 한 개의 쓰레드로 구성된다. 본 논문에서는 프로세스에는 "REAL_TIME", 쓰레드에는 "TIME_CRITICAL"로 최상의 우선순위를 설정하여 실시간 타이머를 동작시켰다.[17] 테스트한 시스템의 하드웨어 사양은 인텔 펜티엄 II 266 Mhz와 메인 메모리 128 MBytes이다. 10회 테스트하여 최악의 결과를 표시하였다.

```
// 프로세스와 쓰레드의 생성 및 우선순위 부여.
...
// 실시간 타이머(10 msec 간격) 생성과 소멸.
TIMECAPS    tc;
UINT        uTimerResolution;
If (timeGetDevCaps(&tc, sizeof(TIMECAPS)) == TIMERR_NOERROR) {
    UTimerResolution = min(max(tc.wPeriodMin, 1), tc.wPeriodMax);
    TimeBeginPeriod(UTimerResolution);
    ...
    TimeEndPeriod(UTimerResolution);
}
If (timeSetEvent(10, 1, PeriodicCallBack, NULL, TIME_PERIODIC) == NULL)
    Return          ERR_TIMER;
...
// 실시간 타이머의 이벤트 수집 및 시간 주기 계산.
void CALLBACK    PeriodicCallBack(UINT id, UINT msg, DWORD userData, DWORD dw1, DWORD dw2) {
    ...
    ProcessPeriodic(id, userData);
    ...
    if (fDestoryTimer)    // 타이머가 필요 없을 때.
        TimeKillEvent(id);
}
```



	0ms	1ms	5ms	6ms	9ms	10ms	11ms	12ms	13ms	14ms	15ms	20ms
Windows NT	31	0	0	0	0	9808	144	0	0	0	0	17
Windows 98	0	0	1519	53	60	6738	54	3	2	47	1524	0
Windows 95	2679	38	1	0	0	1	0	1	1979	5298	2	1

Fig. 2 Accuracy of an internal real-timer on MS-Windows NT, 98, 95

MS-Windows NT는 4가지의 경우만을 가지면서 설정치10msec에 집중되어 있고, MS-Windows 98은 10msec에 집중되어 있으면서 5, 15msec에도 상당한 분포를 보인다. MS-Windows 95는 13msec와 14msec에 많은 분포를 보인다. 이처럼 세가지 운영체제 모두 실시간 제어를 하기에는 부족함을 알 수 있다. 이것은 서보 모터의 운동 특성에 영향을 미쳐서 가공 정밀도를 떨어뜨릴 수 있다. 또, 기계의 입출력이 불규칙적으로 수행됨으로써 시스템의 안정성을 저해하는 요인으로 작용할 수 있다.

3.3 보완된 실시간 제어 방법

서보 모터를 제어하는 소프트웨어 모듈의 실행주기를 10 msec로 설정하고, 이를 수행하기 위하여 실시간 타이머의 이벤트는 MS-Windows NT에서 최소 설정치인 1msec 간격으로 발생시킨다. 그리고 발생되는 이벤트에 대하여 그 시점의 정확한 시간을 구하여 이미 설정된 허용오차 범위(예를 들면, 9.8msec이상)에 포함되는 경우에는 지정된 실행주기(10msec)를 사용하여 서보 모터를 제어한다.

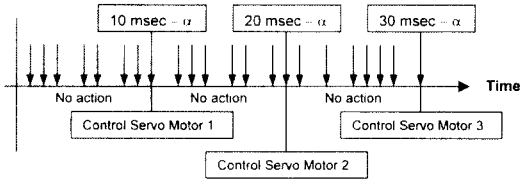


Fig. 3 Control of servo motor with compensated method

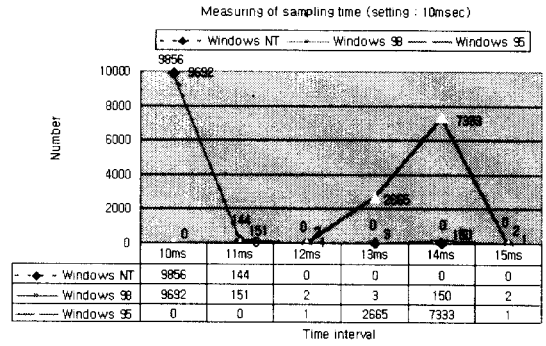
Fig. 3은 서보 모터를 일정한 간격으로 제어하기 위하여 제안된 방법이다. 설정된 1msec간격으로 이벤트가 발생하면 실행주기(10msec)의 허용오차(α) 범위를 만족하는 경우에만 서보 모터를 제어한다. 다음은 이를 제어하기 위한 프로그램이다.

```
// 실시간 타이머(1 msec 간격) 생성과 소멸.
If (timeSetEvent(1, 1, PeriodicCallBack, NULL,
TIME_PERIODIC) == NULL)
    Return          ERR_TIMER;
...
// 실시간 타이머의 이벤트 수집 및 시간 주기 계산.
if (ExactInterval-OldRealInterval ) (SamplingRate-
Alpha)) {// 범위에 들어 올때만 서보 모터를 제어한다.
    MC_ControlServoMotor(RealInterval);
        // 테스트시에는 데이터를 수집한다.
}
OldRealInterval = RealInerval;
ExactInterval + = SamplingRate;
```

그러나 이와 같은 방법도 실행주기가 일정하지 않음으로써 가공상의 오차가 발생된다. 식 (1)은 실행주기가 일정하지 않아 발생되는 가공오차를 계산하는 식이다.

$$E=F/(60\times 1000)\times |S-S_t| \qquad (1)$$

단, E : 가공오차량(mm),
F : 이송속도(mm/min),
S : 실행주기(m sec),
S_t : $_t$ 시점의실행주기(m sec).



	10ms	11ms	12ms	13ms	14ms	15ms
Windows NT	9856	144	0	0	0	0
Windows 98	9692	151	2	3	150	2
Windows 95	0	0	1	2665	7333	1

Fig. 4 Accuracy of an internal real-time timer with compensated method on MS-Windows NT, 98, 95

식 (1)에서 이송속도(F)가 커짐에 따라 이송경로의 오차도 커짐을 알 수 있다. 가공을 500mm/min으로 하고 실행주기의 최대 차이가 1msec라고 가정한다면, 가공오차는 약 0.008mm가 발생된다. 즉, Fig. 4에 MS-Windows NT에서 실행주기의 차이는 1 msec 정도이므로 축의 이송상의 오차는 약0.008이다. MS-Windows 98도 비슷한 결과를 보이지만 14msec에도 약간의 분포를 보이며, MS-Windows 95는 13msec와 14msec에 많은 분포를 보여 설정치 10msec에 대해 많은 오차가 생기는 것을 알 수 있다. 또 PC의 하드디스크나 콘솔(console)을 제어하는 경우에도 MS-Windows NT는 MS-Windows 98, 95보다는 안정적이었다.

MS-Windows NT에서 제공되는 실시간 타이머를 가공에 사용할 때, 일정한 범위 내에서만 항상 실행이 되면 큰 문제가 되지 않는다. 왜냐하면 가공오차는 정지 시에 발생되는데 이 때의 가공속도는 거의 0에 가깝기 때문이다. 식 (1)에서 이송속도가 작을 경우 가공상의 오차도 작음을 알 수 있다. 하지만 정밀 가공에서는 이런 오차도 문제가 될 수 있으므로 상업용 실시간 운영체제를 이용한다. 이러한 경우에 보통 실행주기의 오차는 최대 0.03msec 이하이므로[18], 비용이 문제가 되지 않는다면 실시간 운영체제를 MS-Windows NT에 부가하면 된다.

4. 제안된 개방형 CNC의 소프트웨어 구조

4.1 개요

제안된 개방형 CNC 소프트웨어는 운영체제 또는 실시간 운영체제에서 제공되는 멀티태스킹, 실시간 타이머 등을 이용하여 각 모듈을 동작 단위인 태스크로 만든다. 태스크는 크게 모션 코드 생성 태스크(해석과 경로생성 모듈), 서보 모터 제어 태스크, 입출력 제어 태스크, 통합 제어 태스크, 사용자 인터페이스 태스크 등으로 나누어진다.

Fig. 5는 개방형 CNC 소프트웨어의 구조를 나타낸 것이다. Win32 Process나 Real-time Process는 태스크를 나타낸 것이며, 제공되는 객체(object)나 API의 조합으로 구성된다. MS-Windows NT만으로 CNC 시스템을 구성할 때에는 Real-time Process영역의 태스크들도 Win32 Process 영역에서 수행된다. 실시간 운영체제를 사용할 경우에는 부가된 Real-time HAL(hardware abstraction layer)이 MS-Windows NT의 인터럽트보다 우월하여 고해상도 클럭과 타이머를 제공하여 실시간 제어가 가능하게 한다.[18] 서보 제어 태스크와 입출력 제어 태스크는 실시간 수행 영역에서 수행되는 것으로 Real-time HAL이 제공하는 실시간 타이머에 의해 정확한 실행주기를 가지고 소프트웨어 모듈이 반복된다. 다른 태스크들은 Windows API와 실시간 API를 이용하여 수행된다.

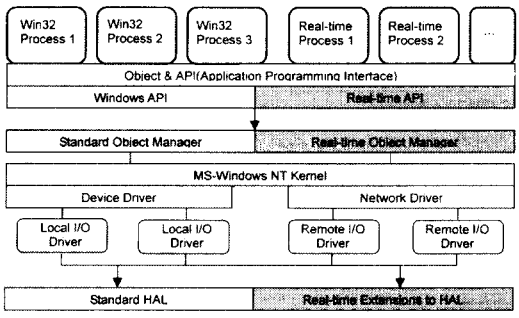


Fig. 5 Software architecture of an open CNC system

4.2 태스크와 공유메모리

각 태스크들은 한 개의 독립된 모듈로 동작하며, 다른 태스크들과도 상호 독립적이다.

Fig. 6은 각 태스크간의 데이터의 흐름을 표시한 것이다. 개방형 CNC 소프트웨어의 각 태스크들은 프로세스간 통신(interprocess communication)과 메시지 발생 등을 통하여 데이터와 이벤트를 교환한다. 각 태스크는 한 개의 프로세스로 존재하지만 실제 구현에서는 모션 코드 생성 태스크, 통합 제어 태스크 및 사용자 인터페이스 태스크는 한 개의 프로세스 내에 3개의 쓰레드로 존재한다. 각각을 프로세스로 설정하는 것보다는 단일 프로세스 내에 몇 개의 쓰레드로 처리하는 것이 시간이나 자원면에서 볼 때 낭비가 더 적다.[17]

내부 시스템 데이터는 모션 코드 생성 태스크와 통합 제어 태스크, 사용자 인터페이스 태스크의 공통변수를 포함하고 있으며, 모듈 제어 데이터는 이 세 태스크들의 순차적 수행이나 수행 오류를 처리한다.

제안된 개방형 CNC 시스템의 각 프로세스는 모두 SMAF(shared memory access function)를 통하여 데이터 교환이 이루어지기 때문에 각 태스크들은 모두 상호 독립적으로 동작할 수 있다. 데이터 교환이 일어나는 공유메모리는 메모리에 포인터를 이용하여 직접 접근이 가능한 파일매핑(file mapping) 방법을 사용한다.[17]

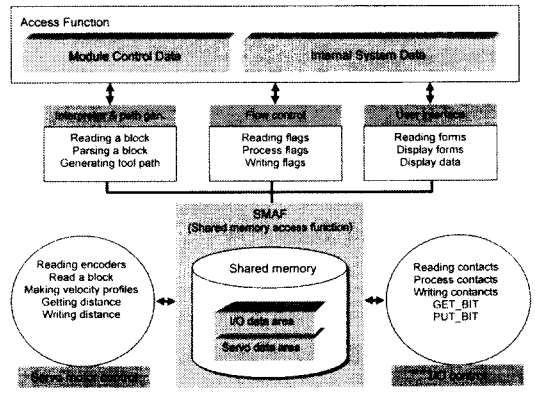


Fig. 6 Data flow between tasks

Table 1 Structure of a shared memory

ITEMS	EXPLANATION
Input Output Signal	Input signals, output signals
System Parameter	Starting address of each parameter, IO contact number, sampling time for servo control, sampling time for PLC 1, 2 level, assignment time for PLC 1,2 level, check of each task status, etc.
Machine Parameter	Number of used axis, axis type(axis, spindle, MPG, etc), port number of axis, Encoder value & Multi ply, encoder direction, voltage direction, max-min command voltage, PID gain & reference value, backlash value, overrun value, gear ratio, etc
Moving Condition Parameter	Homing velocity(rapid, dog, C-pulse detection), max velocity, override(rapid, cutting, spindle), software limit(+, -), axis interlock, positioning time, read & write pointer of continuous block, progressing bit of continuous block, etc.
Continuous Block Parameter	Distance & velocity & acc-dec time of each axis, vector velocity, vector acc-dec time, dwell time, angle of thread, pointer & file name of block, etc.
Monitering Variable	Command position, current position, remained distance, following error, command velocity, current velocity, dwell time, homing check, moving check of axis, positioning check, etc

Table 1은 공유메모리의 구조를 나타낸 것이다. 공유메모리는 크게 입출력 제어 영역과 서보 모터 제어 영역으로 구성된다. 입출력 영역은 기계와 관련된 입출력 접점과 다른 태스크와 주고 받는 상태 접점이 있으며, 접점의 개수는 시스템의 구성에 따라 변경이 가능하다. 입출력 제어 태스크에서는 각 접점에 대하여 매크로로 작성된 접근함수를 통하여 제어할 수 있다.

공유메모리의 대부분이 서보 모터 제어 태스크와 다른 태스크와의 데이터 교환을 위하여 존재한다. 시스템 파라미터는 서보 모터 제어 태스크와 입출력 제어 태스크의 초기화에 필요한 파라미터들이다. 기계 파라미터는 사용할 기계가 정해지면 결정되는 값들이며, 이송 조건 파라미터는 기계에서 축을 이송시킬 때 필요한 파라미터들을 설정한다.

연속블록 버퍼는 축 이송을 위한 모션코드를 담을 수 있는 영역으로 2개의 영역이 존재한다. 따라서 동시에 서로 다른 두개의 시스템을 제어하거나 기계를 이루는 축들과는 별도의 축을 제어할 수 있다. 지령된 블록은 플래그에 의해 기능이 결정되는데, 각 축의 이송량은 해당 축에 대한 이송량 또는 축의 방향으로 사용된다. 이송 시에 각 축의 이송속도를 각각의 축별로 수행하게 함으로써 비동기 이송에 대한 처리를 수행할 수 있다. 또 비동기 이송시에 각 축별로 가감속을 설정할 수 있어서 축의 유연한 움직임을 수행할 수 있다. 동기 이송은 벡터 이송속도와 벡터 가감속을 통하여 수행된다. 정지시간이나 나사 가공시의 각도 등은 해당 플래그가 설정되었을 때 수행된다. 현재 수행중인 블록의 해당 데이터 파일과 파일에서의 위치는 모니터링을 위하여 사용된다.

모니터링 변수들은 서보 모터를 제어할 때 다른 태스크에서 참조할 데이터를 제공한다. 단지 최종 사용자에게 알리기 위한 데이터도 있지만 다른 태스크와의 상호 신호를 주고받는 데이터도 포함하고 있다.

4.3 실시간 운영체제 영역의 태스크

실시간 운영체제 영역에서는 입출력 제어 태스크와 서보 모터 제어 태스크가 수행된다. 입출력 제어 태스크는 기계에 연결된 입출력 신호와 타 시스템과의 접점 신호를 일정한 주기로 주고 받으며, 결정된 로직을 수행하는 기능을 담당한다. 또 기계의 접점신호를 다루므로 다른 태스크들과 많은 이벤트들을 교환한다. 이러한 이벤트들간의 순차적인 수행이 매우 중요하며 이를 위한 접점이 별도로 사용된다.

서보 모터 제어 태스크는 CNC 시스템에서 최종적으로 축 이송을 담당하는 태스크로서, 다른 태스크로부터 지령된 이송량을 정해진 속도에 따라 이송한다.

Fig. 7은 서보 모터 제어 태스크의 흐름을 각 단계를 수행하는 하드웨어와 함께 표시하였다. 각 단계는 그 기능을 수행할 API가 제공되는데 필요에 따라 이 API를 순서에

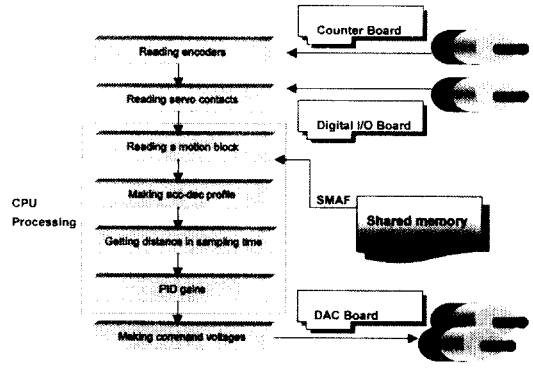


Fig. 7 Flow of servo motor control task

맞게 조립하거나 재정의함으로써 새로운 서보 모터 제어 태스크를 만들 수 있다.

서보 모터 제어 태스크는 일정한 제어 알고리즘이 실행 주기 간격으로 연속적으로 반복되므로 제어 알고리즘도 순차적으로 되어 있다. 제어 프로그램은 먼저 카운터 보드를 통하여 각 실행주기에 모터 위치를 파악하여 전 실행 주기에 지령한 위치와의 차이를 구한다. 공유메모리의 블록에 이송할 블록이 존재하면 이를 SMAF를 통하여 가져와 해당 블록에 대한 가감속 프로파일을 생성한다.

가감속 프로파일의 생성은 FIR 필터(finite impulse response filter)나 수학적인 계산을 통하여 구현할 수 있지만 본 논문에서는 FIR필터는 고속가공에는 적합하지 않으므로 수학적 계산법을 적용하였다. 식 (2)는 가감속 구간에서의 지령속도를 구한다. 각 실행주기당 축의 이송량은 식 (2)에서 구한 지령속도와 실행주기를 곱한 값이다.

$$f = F \cdot \{(t/T) - (1/2\pi) \cdot \sin(2\pi \cdot t/T)\} \qquad (2)$$

단, f: 지령속도,

F: 도달속도,

t: 가감속구간에서의 진행시간,

T: 가감속시간.

입력된 엔코터 값을 통하여 추종오차를 구하고 이를 바탕으로 PID게인값을 적용하여[19], 최종적으로 실행주기당 지령할 이송량을 구한다. 이를 DAC보드에 지령할 전압으로 변환하여 출력함으로써 각 실행주기당 서보 모터 제어 태스크의 역할을 종료한다.

4.4 비실시간 운영체제 영역의 태스크

비실시간 운영체제에서 수행되는 모듈은 경로 생성 태스크, 통합 제어 태스크, 사용자 인터페이스 태스크가 있다. 경로 생성 태스크는 크게 해석 모듈(interpreter module)과 경로 생성 모듈(trajectory generation module), 그리고 사용자 인터페이스로부터 메시지에 의한 직접 축 이송 등으로 구분된다. 해석 모듈은 작업자에 의해 작성된 NC 데이터나 CAM 시스템으로부터 나온 NC 데이터를 받아들여 이송될 공구의 움직임을 지령하거나 공작기계의 동작을 제어한다. 경로 생성 모듈은 입력된 공구 경로에 지정된 기능을 추가하여 새로운 공구 경로를 산출한다.

통합 제어 태스크는 각 프로세스들이 모두 모듈화 되어 있고 내부 함수들이 독립적으로 동작함으로 순차적인 진행이 필요한 경우나 사용자에 의한 이벤트 발생과 같은 경우를 처리한다. 따라서 통합 제어 태스크는 서보 모터 제어 태스크와 입출력 제어 태스크보다 낮지만 다른 태스크보다는 더 높은 우선순위를 갖는다. 통합 제어 태스크는 각각의 태스크에서 발생되는 이벤트들에 대한 즉각적인 반응을 수행하거나 발생된 이벤트에 대하여 다른 태스크들에게 전달하는 기능을 담당한다.

사용자 인터페이스 태스크는 좌표값이나 기계 파라미터, 입출력되는 명령과 같이 최종 사용자와 관련된 각 화면의 구성과 사용자의 요구에 따라 변경되는 화면간의 관계를 입력 받아 수행한다.

5.적용 사례

제안된 개방형 CNC 시스템을 타이어 사이드판의 4축 문자가공시스템에 적용하였다. 문자가공을 위한 개방형 CNC 시스템의 하드웨어는 인텔 펜티엄II 266MHz CPU에 4채널 24Bits 카운터 보드, 4채널 12Bits DAC & 24Bits DIO 보드, 48Bits DIO 보드, LAN 보드를 사용하였다. 제어하고자 하는 축의 수를 확장하는 경우에는 카운터 보드와 DAC 보드를 추가하면 된다. 그리고 운영체제는 MS-Windows NT 4.0과 문자가공의 특성상 실시간 운영체제를 추가하여 정밀한 제어를 하였다. Fig. 8은 PC플랫폼에 하드웨어 모듈이 장착되어 있는 개방형 CNC 시스템의 하드웨어 모습을 보이고 있다.

4축 문자가공시스템은 타이어 사이드판의 프로파일(profile)을 측정한 후, 2차원 평면에서 작성된 가공 형상을 측정된 프로파일에 눌러붙임 투영을 한다.[20] 그리고

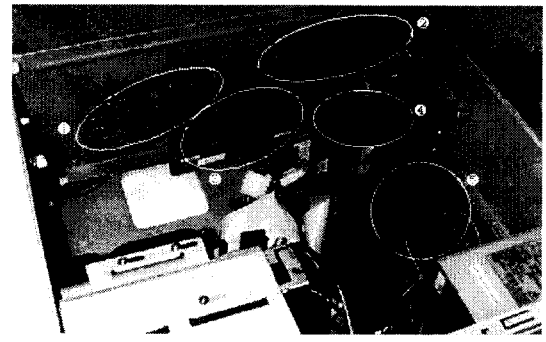
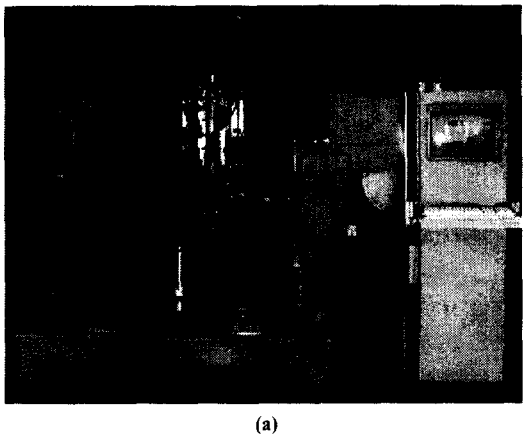


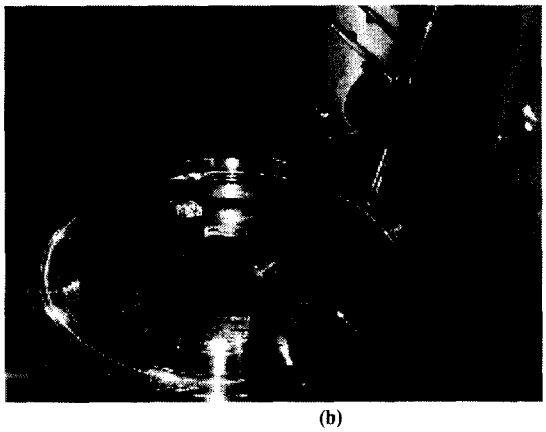
Fig. 8 Hardware of a proposed open CNC ①12bits DAC & 24bits DIO board, ②24bits counter board, ③48bits DIO board, ④LAN board, ⑤Intel pentium II 266MHz CPU

투영된 각 점들을 역기구학에 의하여 각 축의 이동량 또는 회전량을 산출하여 서보 모터를 동작시킨다.

4축 문자가공시스템은 기구적으로도 독특한 구조를 가지고 있는데, 4축 중 직선 축인 X, Z축과 회전축인 B, C축을 가지고 있다. 가공 중 공구의 자동교환을 위하여 30개의 포켓(pocket)을 가지고 있는 ATC(automatic tool changer)가 있다. 또 공구가 교환되면 자동으로 공구의 길이를 측정하는 센서가 있어 모든 작업을 자동화 하였다. ATC는 상업용 PLC에 위치결정모듈을 부착하여 서보 모터의 위치 결정[21]과 기계부의 접점 제어를 수행하였다.[22] 또 문자를 가공하는 시스템이므로 고주파 스핀들을 사용하는데, DIO의 출력접점을 통하여 스핀들의 RPM을 지령한다. 가공 시의 스핀들 RPM은 보통 약25000~40000RPM이다.



(a)



(b)

Fig. 9 4-axis lettering center installed proposed open CNC (a) Lettering center (b) Machining

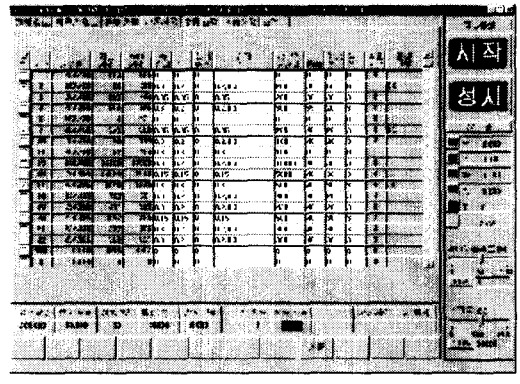


Fig. 10 Input screen for auto-machining

Fig. 9는 제안된 개방형 CNC에 의하여 개발된 4축 문자가공시스템이다. (a)는 시스템의 전체 모습을 나태낸 것인데, 기계에서 좌우, 상하로 이송되는 X, Z축이 있고 스핀들이 장착된 B축, 그리고 척(chuck)과 가공물을 올려놓은 C축이 있다. 오른쪽에 원반형으로 된 공구를 장착하는 ATC와 PC를 장착한 시스템이 있다. (b)는 가공하는 모습을 표시한 것으로 고속으로 회전하는 스핀들이 타이어 사이드판에 글자나 무늬를 새겨 넣는다.

Fig. 10은 타이어 사이드판 전체를 가공할 때의 가공 조건들을 표시한 화면이다. 작업자는 가공할 총 깊이, 스텝, 정삭여유, 공구가공거리, 스핀들 RPM, 가공속도, 접근속도를 입력한 후, 가공하고자 하는 영역만을 선택하면 자동적으로 필요한 공구개수와 ATC에 공구를 삽입해야 할 포켓번호를 알려준다. 따라서 작업자는 지정된 ATC의 해당 포켓에 공구를 끼우면 된다.

6. 결 론

본 논문에서는 개방형 CNC 시스템을 구현하기 위한 하드웨어 모듈과 소프트웨어 모듈에 대한 방법을 제시하였다.

개방형 CNC 시스템의 하드웨어 구성은 산업체에 널리 대중화된 PC 플랫폼에 하드웨어를 기능별로 모듈화 하여 장착하는데, 일반적으로 서보 모터를 제어하는 부분은 DSP칩 또는 MC68030을 이용한 별도의 하드웨어 모듈을 제작한다. 하지만 본 논문에서는 일반적인 용도로 사용되는 카운터 보드, DAC 보드, DIO 보드, LAN 보드 등을 장착하여 공작기계를 제어하는 방법을 제안하였다. 필요한 기능에 따라 한 개의 독립된 모듈로 구성하여 하드웨

어 종속성에서 탈피하여 원하는 소프트웨어를 구현할 수 있었다.

개방형 CNC 시스템은 MS-Windows NT환경에서 수행된다. MS-Windows NT의 부족한 실시간 타이머를 이용하여 제어할 수 있는 방법을 제안하였고, 보다 정밀한 가공을 위하여 상업용 실시간 운영체제를 부가하여 제어하는 방법도 제안하였다. 각 모듈에 대한 재사용을 위해 모든 기능을 객체(object) 그리고 API(application programming interface)로 제공하며, 각 태스크는 이것의 조립만으로 구성될 수 있도록 하였다.

이처럼 구성된 하드웨어와 소프트웨어를 타이어 사이드 금형의 문자가공시스템에 적용하여 실제 산업현장에서 사용하고 있다.

참 고 문 헌

(1) OSACA, http://www.osaca.org/

(2) OMAC, http://www.arcweb.com/omac/

(3) OSEC, http://202.234.194.3/OSEC/

(4) G.Pritsschow, Ch.Daniel, G.Junghans, W.Sperling, "Open System Controllers - A Challenge for the Future of the Machine Tool Industry", Annals of CIRP Vol.42/1, pp.449~452, 1993.

(5) Wolfgang Sperling, Peter Lutz, "Designing Applications for an OSACA Control", ASME Vol. 1, pp.91~95, 1997.

(6) Y.Altintas, W.K.Munasinghe, Department of Mechanical Engineering, "A Hierarchical Open-Architecture CNC System for Machine Tool", Annals of CIRP Vol.43/1, pp.349~354, 1994.

(7) Yoram Koren, Zbigniew, J.Pasek, A.Galip Ulsoy, Uri Benchetrit, "Real-Time Open Control Architectures and System performance", Annals of CIRP Vol.45/1, pp.377~380, 1996.

(8) Mamoru Mitsushi, Takaai Nagao, Hideyuki Okabe, Makoto Hashiguchi, Katsuya Tanaka, "An Open Architecture CNC CAD/CAM Machining System System with Data-Base Sharing and Mutual Information Feedback", Annals of CIRP Vol.46/1, pp.269~274, 1997.

(9) Kazuo Yamazaki, Yoshimaro Hanaki, Masahiko Mori, Kazusaka Tezuka, "Autonomously Proficient CNC Controller for High-Performance Machine Tools Based on an Open Architecture Concept", Annals of CIRP Vol.46/1, pp.275 ~278, 1997.

(10) D.Dumur, P.Boucher, J.Roder, "Advantages of an Open Architecture Structure for the Design of Predictive Controllers for Motor Drives", Annals of CIRP Vol.47/1, pp.291~294, 1998.

(11) T.E.Balley, D.D.A.Renton, M.A.Elbestawl, "PC Based Open Architecture Machine Tool Control", ASME Vol. 1, pp.39~48, 1997.

(12) Y.Altintas, N.A.Erol, "Open Architecture Modular Tool Kit for Motion and Machining Process Control", Annals of CIRP Vol.47/1, pp.295~300, 1998.

(13) 삼성전자, "FARA MMC - Series multi motion 제어기", 1996.

(14) Motion Engineering Inc., "DSP-Series, Motion Controller C Programming Manual", 1995.

(15) Delta Tau Data Systems, Inc., http://www.deltatau.com/

(16) 김종원, "윈도우 환경에서 시간측정하는법", 마이크로소프트웨어, pp.212~217, 1999/2.

(17) 한기용, "윈도우 프로그래밍", 도서출판 대림, 1997.

(18) http://www.vci.com/

(19) 崔明鎬, "DC 모터의 디지털 서보 제어기 구현 연구", 충남대학교 석사학위 논문, 1996.

(20) 이철수, 박광렬, "타이어 사이드판의 문자 가공을 위한 4축 가공 시스템", 산업공학 제11권 제2호, pp.65~77, 1998/7.

(21) LG산전, "LG Master-K시리즈 위치결정 모듈 매뉴얼", 1995.

(22) LG산전, "LG Master-K시리즈 프로그래머블 로직 컨트롤러 프로그래밍 매뉴얼", 1995.