〈논 문〉 SAE NO. 98370059

# 전자제어의 Event 처리방법에 관한 연구

## A Study on the Event Processing for Electronic Control

이 종 숭*, 이 중 순**, 정 성 식***, 하 종 률***
J. S. Lee, J. S. Lee, S. S. Chung, J. Y. Ha

## ABSTRACT

For digital engine control timings, such as ingition and injection, are based on the crank shaft angle. Therefore, it is very important that the angle of the crank shaft can be detected with accuracy for optimal ignition timing. Sequential multi-point injection(MPI) systems that have independent injection events for each cylinder, are used to inject an accurate quantity of fuel, and to cope with varying engine status promptly. In this study the distributorless ignition(DLI)system that is direct ignition method, is used for accurate ignition and flexible ignition timing. A crankshaft position sensor has been installed such that it generates a number of pulses per crankshaft revolution to permit accurate detection of the crank shaft angle.

An event detecting algorithm has been developed, which detects the crank shaft pulses generated by the position sensor, and the software outputs the required control signals at given crank angle values. We clarified that the hardware method is the best way to increase the performance of the control system, because the event detecting duration $T_{(1+2)max}$ becomes zero.

주요기술용어 : Micro Processor Unit(마이크로프로세서 장치), Crank Angle pulse(크랭크각 펄스), Electronic Control Unit(전자제어장치), Distributorless Ignition System(배전기가 없는 점화장치), Direct Ignition(직접점화), Sequential Multi Point Injection System (순차적 다점 연료분사시스템)

* 정회원, 동아대학교 대학원
** 정회원, 동명대학 기계과
*** 정회원, 동아대학교 기계공학과

## 1. 서 론

기관제어에 디지털 기술이 도입된 후 많은 기능들이 단일 제어장치로 집적되고 있다. 초기 시스템은 연료와 점화제어를 위해 독립된 제어 시스템을 사용했으나, 디지털 하드웨어의 발전 및 알고리즘과 소프트웨어의 개선으로 현재는 집적 제어로 가는 추세이다. 기관의 동작에 필요한 연료량과 점화시기같은 제어요소는 각각의 센서로부터 데이터를 받아 들여야 하므로, 빠르고 정확한 제어를 하기 위해서는 집적된 시스템이 유리하다. 특히 반도체 기술의 발달로 높은 클럭 주파수에서 동작하는 MPU(Micro processor unit)가 개발되어, 기관의 정상적인 운전에 필요한 계산 시간을 줄여 빠른 제어를 가능하게 한다.

디지털 기관제어[1~4]에서 개선된 기술 가운데 연료분사시기 및 점화시기 등은 크랭크 각을 기준으로 한다. 순차적 다점 연료분사시스템[5](Sequential multi point injection system)에서는 각 실린더마다 독립된 연료분사기를 통하여 원하는 시기에 연료가 분사되어야 하므로 크랭크 축의 각도를 정밀하게 측정할 수 있다면 좀더 개선된 제어가 가능하다. 또한 기관의 회전수, 공연비 등에 따라 점화시기를 이동하거나, 기관이 요구하는 크랭크 각도에 점화를 행하기 위하여 배전기가 없는 점화장치인 DLI(Distributorless ignition system) 방식을 채택하여 ECU(Electronic control unit)에서 직접 점화(Direct ignition)를 하는 방식으로 변해 가고 있는 추세이다.

크랭크축의 각도를 정밀하게 검출하기 위하여는, 크랭크 축의 회전에 따른 신호(Event)를 많이 발생할 수 있는 센서를 부착하고, 발생된 신호를 전기적 신호로 변환하여 ECU로 입력하여야 한다. 입력된 신호를 이용하여 ECU는 크랭크 축의 회전각을 계산한 후에 기관의 정상적인 운전에 필요한 제어신호를 적절한 시기에 각종 엑츄에이터로 출력한다. 즉 정밀한 크랭크 축의 각도는 크랭크 회전에 따른 Event를 증가시켜야 하며, 증가된 Event를 검출하기 위하여 속도가 빠른 MPU를 사용하여야 한다. 이와 아울러 증가된 Event를 검출하고, 처리하기 위한 알고리즘과 소프트웨어도 동시에 개발되어야 한다.

본 연구는 기관의 전자제어에 있어서, 제어를 위한 가장 근본적이고 기초가 되는 크랭크 각도를 Event 개념으로 검출하는 알고리즘과 소프트웨어 및 하드웨어의 개선을 목적으로 소프트웨어만으로 검출하는 방법, 인터럽트를 이용한 방법, 하드웨어를 이용하는 방법들에 대하여 밝힌다.

## 2. 이론적 배경

### 2.1 Event와 각도변환

점화와 연료 분사시기와 같은 기관 제어 시스템의 출력 값들은 크랭크 각도를 기준으로 하고 있다. 시간 영역의 미분방정식을 크랭크 각 영역으로 변환하면 식(1)과 같다.

$$N = \frac{1}{6}\frac{d\theta}{dt} \quad [\text{rpm}] \tag{1}$$

N : 기관속도(rpm), $\theta$: 크랭크 각도(deg.)

크랭크 축의 회전에 따른 임의의 위치에서 Event를 이용한 정밀한 각도를 측정하기 위하여 크랭크 축에 다수의 Event를 발생하는 자기센서 또는 광전센서[4]를 부착하고, 크랭크축의 CA(Crank angle) pulse 수를 $P$ [pulse/revolution]라 하면 Event에 해당하는 크랭크의 회전각은

$$\theta_p = \frac{360}{P} \quad [\text{deg/event}] \text{ 이다.}$$

따라서, Event의 변화에 대한 크랭크축의 회전각은

$$\Delta\theta = \theta_p \Delta p \quad [\text{deg}] \tag{2}$$

이고, $p$ 는 크랭크축에서 발생되는 Event 개수이다. $\Delta p$ 는 Event의 변화량이며 이산적(Discrete)인 값으로서 ECU에서 검출하기 쉬운 Digital 신호이다. 크랭크각도 영역에서 Event 영역으로 변환하면, 식(1)과 (2)로부터 식(3)이 된다.

$$N = \frac{\theta_p}{6} \frac{\Delta p}{\Delta t} \text{ [rpm]} \quad (3)$$

본 연구에서는 크랭크 축의 회전에 대하여 Event를 이용한 각도를 측정하기 위한 방법을 제시하며, 펄스 수를 높인 센서의 디지털 출력을 Event로 하여, 크랭크 축의 회전에 대한 펄스의 상태변화를 검출하여 크랭크 각을 측정한다.

### 2.2 오 차

$\Delta p$ 는 크랭크 축의 회전에 따른 Event의 변화이므로 Event와 Event사이에서는 직접적인 각도를 계산할 수 없으며 간접적인 방법으로만 측정을 할 수 있다. 따라서, 오차를 감소시키기 위하여 크랭크 축의 1회전에 따른 Event 개수 ($P$)를 크게 하면 가능하지만, Event 개수를 무한정 크게 할 수 없으므로 Event에 따른 크랭크 각도는 오차가 발생할 수밖에 없으며, 이 오차는 Event를 검출하는 Sampling 주파수가 충분히 높을 때 $\theta_p$ 보다는 적다. 오차를 줄이기 위하여 Event를 증가시키면 ECU에서 Event신호의 처리에 많은 시간이 필요하게 되어 전체적인 전자제어 시스템[5]의 성능을 저하시키는 원인이 된다. 이는 빠른 속도의 MPU를 사용하여 개선할 수 있다.

## 3. 실험장치 및 방법

### 3.1 실험장치

본 실험에 사용된 기관은 4-Valve 2.0ℓ DOHC SI 기관을 개조한 단기통 기관을 사용하였다. 이 기관은 고속으로 운전을 행하기에는 어려움이 있어 1000rpm까지만 기관에 적용하여 실험을 하고, 그 이상의 속도에서는 기관회전에 대응하는 신호를 펄스발생기에서 발생하여 실험을 한다. 기관의 크랭크축에 엔코더를 부착하여 크랭크 1회전당 180개의 펄스를 발생하거나, 또는 펄스발생기를 이용하여 기관의 회전에 대응하는 신호를 발생하여 입력 인터페이스로 보낸다. 입력 인터페이스에서는 신호를 입력하여 컴퓨터에서 처리할 수 있도록 파형을 정형하고, 출력 인터페이스에서는 컴퓨터의 출력을 입력하여 드라이버에 필요한 신호로 변환하여 드라이버로 보낸다. 드라이버에서는 각각의 엑츄에이터에 필요한 신호로 완충하여 엑츄에이터를 동작시킨다. 모든 입력신호는 컴퓨터로 입력되며 컴퓨터는 이들 신호의 처리 및 제어에 필요한 연산을 수행한다. 제어 및 연산프로그램은 C 언어[7,8]와 어셈블리 언어[6,10]로 작성하였다.

### 3.2 Event 검출 알고리즘

Fig.2는 크랭크축의 회전에 따라 발생된 Event (Pulse)로서 입력신호를 정형한 것이다. 파형의 상부레벨을 High, 하부레벨을 Low, 이들의 변화를 Event라고 한다. Low에서 High로 변화하는 부분을 Positive edgy(Ⓐ점), 그 반대인 부분을 Negative edgy(Ⓑ점)라고 한다. Event를 검출하기 위하여 대부분 Edgy의 변화를 이용하며, CA pulse의 상태가 변화하는 것을 하나의 Event로 한다. 따라서 CA pulse의 상태 변화를 확인하기 위하여 이전의 상태를 항상 기억하고 있어야

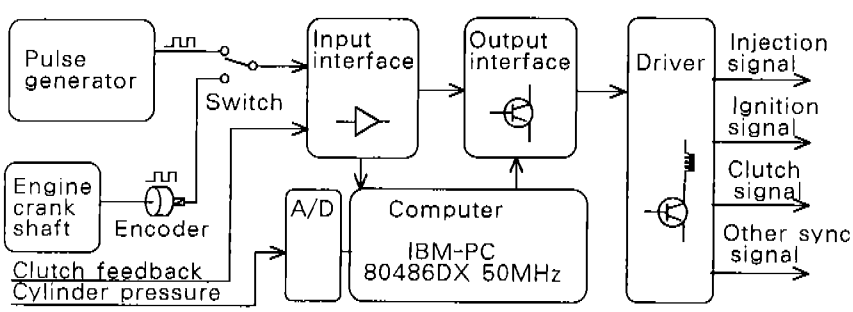



Fig.1 The schematic diagram of the experimental apparatus

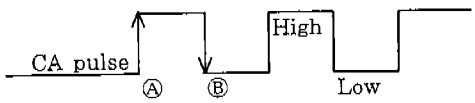



Fig.2 The crank angle pulse

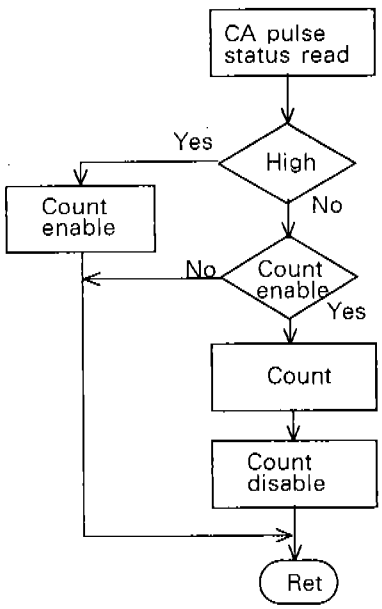



Fig.3 The flow chart of the event detection

하며 이러한 방식에서 보면

① Positive edgy 상태변화만 검출하는 방법

② Negative edgy 상태변화만 검출하는 방법

③ 2가지의 모든 상태의 변화를 검출하는 방법의 경우가 있으며, ③의 경우는 다른 경우보다 정밀도가 2배로 된다. ②의 경우에 대한 Flow chart는 Fig.3과 같이 구현할 수 있으며, 이를 구현하기 위하여 C 언어와 어셈블리 언어를 사용하였으며 어셈블리 언어로 코딩을 하면 Program 1과 같다.

본 실험에서는 Fig.4와 같이 점화신호, 연료분사시기, 연료분사기간을 미리 설정하여 정속운전을 행하고, 프로그램에서는 Event를 검출한 후 설정된 값에 따른 정확한 제어신호의 출력을 하는가에 중점을 두고있다. 따라서, 센서의 값들을 입력하여 과도기적 기관의 운전상태에 따른 최적의 동작조건을 추출하여 점화신호, 연료분사시기, 연료분사기간을 계산하는 부분은 본 연구에서 제외되었다.

3.2.1 제어 프로그램에서 직접 Event를 검출하는 방법

프로그램의 수행 중에 항상 Event가 발생되었는지를 주기적으로 검사(Sampling)하고 변화가 있을 경우에 이를 검출하여 처리하는 방법으로서 CA pulse를 검출하는 알고리즘 중 가장 간단하게 사용할 수 있지만, Event 발생의 확인을 주기적으로 하여야 하므로 시간적인 낭비가 많고, 각종 센서의 값들을 입력하여 동작조건을 추출하는 시간이 상대적으로 줄어들게 되므로 오차가 발생하기 쉽고, 이로 인하여 시스템의 전체적인 성능을 저하시킨다. 또한, 기관의 운전속도가 증가하여 Sampling 속도보다 더 빠른 Event가 발생하면 프로그램만으로는 처리가 불가능하다. 즉, Event 발생 주기를 $T_E$, Sampling 주기를 $T_S$라고 하면 $T_E > T_S$의 조건을 만족하여야 한다.

3.2.2 Interrupt[6,10,13]를 이용하는 방법

주 프로그램을 실행을 하는 동안에 Event 발생을 주기적으로 Sampling 하지 않고 Event가 발생되면 현재 실행하고 있는 프로그램을 즉각 중지하고 Event 발생에 대한 처리프로그램(Interrupt Handler)을 실행한다. 실행이 끝나면 다시 인터럽터 이전에 실행하던 프로그램을 계속 실행한다. Fig.5에 이들의 실행 구성을 보였다.

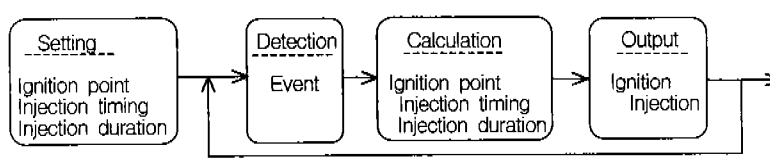



Fig.4 The program flow chart for the periodic sampling method

```
DETECT$EVENT    PROC        NEAR
                MOV     DX, PORT_ADDRESS
                IN      AX, DX                  ;READ DATA
                AND     AX, MASK_EVENT          ;MASK
                JNZ     STATUS_HIGH             ;IF STATUS HIGH
STATUS_LOW:     TEST    FLAG, COUNT_ENABLE
                JNZ     NO_COUNT
                INC     DATA_EVENT              ;ADD EVENT
                MOV     FLAG,COUNT_DISABLE      ;DISABLE COUNT
                JMP     RETURN
STATUS_HIGH:
NO_COUNT:       MOV     FLAG,COUNT_ENABLE       ;ENABLE COUNT
RETURN:         RET
DETECT$EVENT    ENDP
```

Program 1 The event detection subroutine

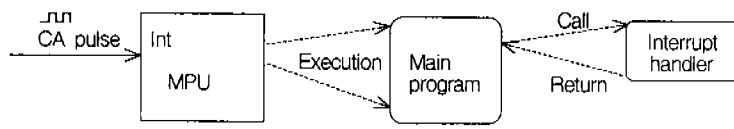



Fig.5 The program flow chart for the interrupt method

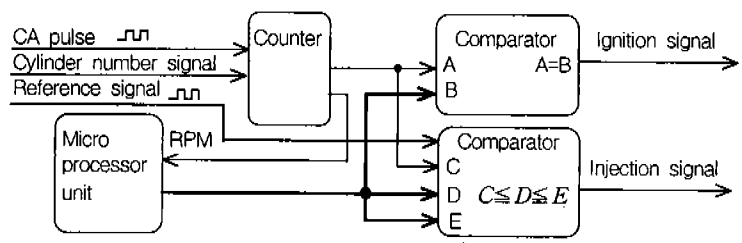



Fig.6 The block diagram with the hardware

이 경우는 CA pulse의 변화가 MPU에 인터럽터를 요구하면, MPU는 이를 인식하여 Main 프로그램을 잠시 중단하고 Event를 처리하는 프로그램을 실행한다. 이 프로그램이 완료되면 원래의 Main 프로그램으로 되돌아가서 실행을 계속하기 때문에 Event가 발생할 때만 처리를 하므로 시간의 낭비를 막을 수 있는 장점이 있다.

3.2.3 Hardware를 이용하는 방법

Event를 검출하기 위한 Interrupt 방식이나, 프로그램에서 주기적으로 Sampling하는 방법은 많은 Event가 발생하는 기관에서는 Event처리에 필요한 시간이 길어지므로, 기관의 최적운전 동작조건을 추출하는 시간이 상대적으로 짧아져서 전자제어 시스템의 성능을 저하시킨다.

특히, 정밀한 각도를 측정하고 제어하기 위해서는 크랭크 회전에 따른 Event를 많이 발생하여야 하므로, Event의 검출 및 설정된 각도에 제어신호를 보내는 부분의 처리는 하드웨어에 의존하는 방법이 더 효율적이다. 즉, MPU는 센서의 입력을 받아 최적의 운전동작조건을 계산하고 처리하여 하드웨어로 보내면 된다. 이 방법은 MPU의 부담이 줄어들어 효과적인 제어와

빠른 처리를 할 수 있다. Fig.6은 이들의 실행 구성을 보였다.

이의 구성을 보면, CA pulse를 입력받아 카운트하는 계수기가 있으며, 이는 입력된 카운터 수를 환산하여 각도 값으로 변환되도록 구성되어있다. 초기 조건을 설정해 주기 위하여 1번 실린더 검출신호 또는 실린더 번호 검출신호가 입력되어야 하고, 정확한 연료분사 기간의 제어를 위하여 기준신호를 동시에 입력한다. 이 때 MPU에서는 각종센서와 기관의 운전조건에 따른 최적 점화시기와 연료 분사시기 및 분사 량을 결정하여 비교기에 입력한다. 계수기의 카운트와 MPU에서 설정한 값이 조건에 적합할 때 점화 신호와 연료분사 신호를 비교기에서 직접 출력한다. 본 실험에서는 Timer와 계수기 기능을 함께 가지고 있는 8254[6,12] 소자를 사용하였다.

## 4. 실험결과 및 고찰

앞 절에서 언급한 것과 같이 1000rpm까지만 실제의 기관에 적용하여 실험을 하고, 그 이상의 속도에서는 기관회전에 대응하는 펄스를 펄스 제너레이터를 이용하여 실험을 하였다.

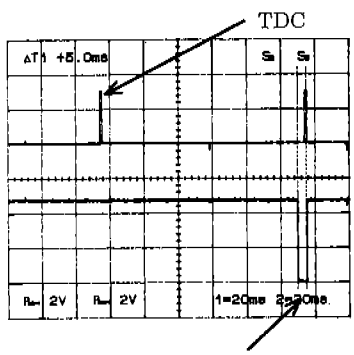
Event를 검출 3가지의 방법, 즉 직접 검출방법, 인터럽트를 이용한 방법, Hardware를 이용한 방법에서 미리 지정한 값에 제어신호를 출력하는 것이 1,000rpm까지 동일한 결과를 나타내므로, 별도의 구분 없이 하나의 그림에 3가지의 결과를 동시에 표현하였다. Fig.7과 Fig.8에 나타내었으며 점화시기, 연료분사시기 및 연료분사기간의 제어신호가 정확하게 출력되고 있음을 알 수 있다.

Fig.7은 TDC를 기준으로 점화신호가 BTDC 30deg.인 점(1000rpm에서 -4.99ms)에서 출력이 되고 있음을 나타내었다.

Fig.8은 TDC를 기준으로 연료분사신호가 ATDC 260 deg.인 점(1,000rpm에서 43.33ms)에서 출력되며, 연료분사기간은 연료분사시기를 기준으로 하여 설정된 시간 15ms동안 출력됨을 볼 수 있다.

본 연구에서는 Event를 검출($T_1$ 시간)하여 미리 설정된 점화시기, 연료분사시기·기간의 값에 정확하게 제어 신호를 출력($T_2$ 시간)하는 방법에 대한 것이며, 각종센서의 값을 읽어 기관의 최적운전조건을 추출($T_3$ 시간)하는 부분은 다루지 않았다. 따라서 MPU가 최적운전조건을 구하는 시간은 가상적인 값에 의존하여 실험을 하였으며 이들의 구성도를 Fig.9에 보였다.

컴퓨터(IBM-PC, 80486DX 50MHz)를 사용하여 Program.1(Negative edgy 상태변화만 검출하는 방법)을 어셈블리 언어로서 프로그램하였을 때 하나의 Event를 검출하는 시간 $T_1$은 1.00$\mu$s~1.44$\mu$s, 하나의 Event를 검출하여 비교·출력하는 시간 $T_2$는 2.24~4.72$\mu$s이다. 크랭크 축의 1회전당 Event를 검출하기 위한 최대시간은 식 (4)와 같다.

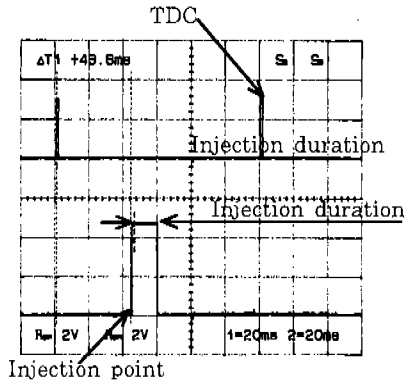



Ignition point

Setting values
- Injection duration : 15 ms
- Injection point : ATDC 260 deg
- Ignition point : BTDC 30 deg
- Speed of engine : 1000 rpm

1) The direct event detection.
2) The event detection with the interrupt.
3) The event detection with the hardware.

Fig.7 Ignition timing in each method




Setting values
Injection duration : 15 ms
Injection point : ATDC 260 deg
Ignition point : BTDC 30 deg
Speed of engine : 1000 rpm

1) The direct event detection.
2) The event detection with the interrupt.
3) The event detection with the hardware.

Fig.8 Injection timing and duration in each method

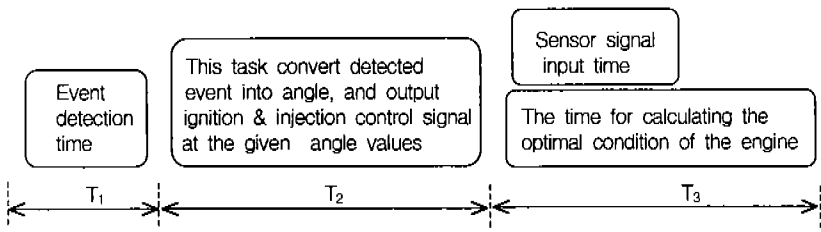



Fig.9 The time to be taken for the part of the program

$$T_{(1+2)max}=(T_{1max}+T_{2max})\times N_E \text{ [ms]} \quad (4)$$

$N_E$ : 크랭크축 1회전당 Event 개수.

식 (4)에서 알 수 있는 바와 같이 $N_E$가 증가할수록 정밀한 크랭크각도를 검출하여 제어할 수 있으나, Event를 검출하는 데에 많은 시간이 소요되어 늦은 속도의 MPU에서는 시스템의 성능을 저하시킨다. 따라서, 이의 개선을 위하여 속도가 빠른 MPU를 적용하고, 많은 Event를 빠른 시간에 검출하기 위한 검출 알고리즘과 소프트웨어의 개발이 동시에 이루어져야 한다.

본 시스템에서 Event를 검출하고 지정된 시간에 제어신호를 출력하는 시간이 시스템에 미치는 영향을 시간 개념으로 나타낸 것이 Fig.10 이다. 크랭크축의 1회전당 시간을 $T_C$라고 할 때, ECU가 최적운전조건을 추출하기 위하여 할당되는 시간은 식 (5)와 같이 된다.

$$T_3 = T_C - T_{(1+2)max} \quad (5)$$

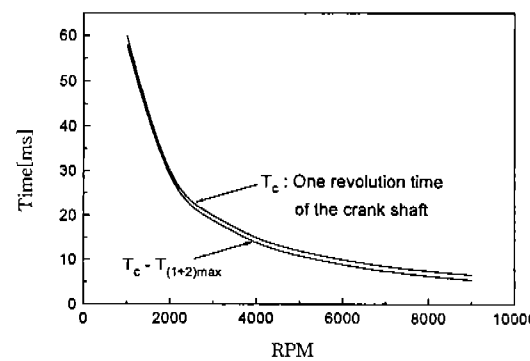



Fig.10 The relationship between for Tc and $T_3$

즉 $T_{(1+2)max}$ 값이 적을수록 제어 시스템이 기관의 최적운전조건을 계산하기 위한 부분에 많은 시간을 할당하므로 시스템이 안정되는 반면, 기관의 속도가 증가할수록 $T_C$가 적어지고 $T_{(1+2)max}$은 증가하여 $T_3$가 급속히 감소하므로 기관의 최적운전조건 추출 시간이 적어지게 되어 시스템이 불안정하게 된다.

즉, 제어 프로그램에서 직접 Event를 검출하는 방법은 일정한 주기로 Event를 검출하는 프로그램을 실행하므로 $T_{(1+2)max}$의 시간이 길어지고, 반면에 Interrupt를 이용하는 방법은 Event가 발생할 때마다 Event를 검출하는 프로그램을 실행하므로 $T_{(1+2)max}$의 시간이 직접 Event를 검출하는 방법보다는 짧아진다. 그러나, Hardware를 이용하는 방법은 Event를 Hardware적으로 처리하므로 $T_{(1+2)max}$의 시간이 0이 되어 최적의 상태를 나타낸다.

본 실험에서는 속도가 빠른 32Bit IBM-PC를 사용하였으므로, $T_{(1+2)max}$에 의한 $T_3$의 변화 폭은 적으나 속도가 늦은 MPU에서는 변화 폭이 크다, 따라서, 속도가 늦은 MPU에서는 $T_{(1+2)max}$의 시간을 감소하기 위하여 크랭크 회전에 따른 Event 수를 적게 해야만 하며, 이는 아래의 조건과 같이 나타낼 수 있다.

$T_3 \leq Tc - T_{(1+2)max}$의 조건일 때는 기관 행정에서 직접 제어가 가능하고,

$T_3 > Tc - T_{(1+2)max}$의 조건일 때는 다음의 기관 행정에서 제어가 가능하다.

## 5. 결 론

크랭크 각도를 Event 개념으로 검출하는 알고리즘 및 프로그램을 제어 프로그램에서 직접 Event를 검출하는 방법, Interrupt를 이용하는 방법, Hardware를 이용하는 방법으로 개발하였으며 이를 단기통 가시화 기관에서 적용하여 제어신호가 정확히 출력되었다.

최적운전조건을 구하는 시간에 비하여 Event 검출시간 $T_{(1+2)max}$가 상대적으로 짧을수록 우수한 시스템 특성을 가지며, $T_3 \leq Tc - T_{(1+2)max}$의 조건일 때는 현재의 행정에서 직접 제어가 가능하고, $T_3 > Tc - T_{(1+2)max}$의 조건일 때는 다음의 행정에서 제어가 가능하므로 후자인 경우는 성능의 저하를 가져온다, 따라서 속도가 빠른 MPU를 적용하여 $T_{(1+2)max}$를 짧게 하고 동시에 시스템에 적합한 Event검출 알고리즘과 프로그램이 적용되어야 한다.

본 연구에서는 Hardware를 이용한 방법이 가장 우수한 성능을 가지며 특히, Event 검출시간 $T_{(1+2)max}$이 0이 되어 제어시스템의 성능을 최대로 향상한다.

## 참 고 문 헌


1. 선우명호, 남평우, 김명준, "기관제어시스템 개발에 관한 연구" 한국자동차공학회 1996년도 추계학술대회 NO. 96380270, 1996.
2. 고상근, 선우명호, "저공해 엔진 전자제어 시스템 기술개발", G7 차세대자동차기술, 제2회, pp 179~182, 1994.
3. 한만배, 선우명호, "기관 기본 신호 및 Marker 신호 발생기 설계·제작에 관한 연구" 한국자동차공학회 1996년도 추계학술대회 NO. 96380269, 1996.
4. 김태훈, 조진호, "마이크로 컴퓨터를 이용한 가솔린 기관용 전자제어 장치의 개발에 관한 연구", 한국자동차공학회 논문집 제3권 제6호, pp. 224~237, 1995.
5. William B. Ribbens, 김수원 옮김, "자동차 전기공학" 청문각, pp. 209~247, 1996.
6. 차영배, "IBM PC 인터페이스응용", 통일출판사, pp. 369~418, pp. 259~316, 1996.
7. 허단, "다시 보는 어셈블리", 도서출판 한국컴퓨터 매거진, 1994.
8. 황희융, "TURBO C 2.0 라이브러리 매뉴얼", 교학사, pp. 43~104, pp. 627~656, 1995.
9. 임인건, "TURBO C 정복", 가남사, 1995.
10. 유정하 편저, "PC 시스템 프로그래밍", 정보문화사, pp. 43~104, pp. 627~656, 1991.
11. Nabajyoti Barkakati and Randall Hyde, "Microsoft Macro Assembler bible", Second Edition, pp.133~339.
12. Intel data book, "Peripheral Components", pp. 3-83~3-99, 1992.
13. M. Morris Mano, 김종상 역, "전자계산기 구조", 탑출판사, pp. 105~109, pp. 144~146, pp. 342~350, 1983.