AN021

Application Note 021 zu MODULAR-4

Digitaler PID-Regler auf MODULAR-4

Autor: M.H. AN021.DOC (7 Seiten)

1. Aufgabenstellung

In vielen Fällen der Prozessautomatisierung kommen Regler zum Einsatz. Die am meisten verbreiteten parameteroptimierten Regler haben P-, PI- oder PID-Verhalten.
Wünschenswert wäre es, solche analogen Regler durch Digitalrechner, wie z.B. MODULAR-4, zu ersetzen. Man kann dann auf Erfahrungen mit analogen Reglern zurückgreifen und im Prinzip die bekannten Einstellregeln für die Reglerparameter verwenden.

2. Diskretisieren der Differentialgleichungen kontinuierlicher PID-Regler

Der PID-Regler setzt sich aus einem proportionalen Anteil, einem integralen Anteil und einem differentiellen Anteil zusammen.

$$u(t) = K * \left[e(t) + 1/Ti * \int_0^t e(\tau)\, d\tau + Td * de(t)/dt \right]$$

(2.1) Gleichung des analogen PID-Reglers

Wobei nach DIN 19226 die einzelnen Parameter wie folgt bezeichnet werden:

K Verstärkungsfaktor
Ti Integrierzeit (Nachstellzeit)
Td Differenzierzeit (Vorhaltzeit)

Die Regler mit P-, PI-, PD-Verhalten etc., ergeben sich aus dem PID-Regler durch Weglassen der entsprechenden Anteile.
Für relativ kleine Abtastzeiten To (im Verhältnis zur Regelzeit) kann man diese Gleichung durch Diskretisieren direkt in eine Differenzengleichung umwandeln. Hierzu wird der Differentialquotient durch eine Differenz erster Ordnung und das Integral durch eine Summe ersetzt. Die kontinuierliche Integration kann dabei durch Rechteck- oder Trapezintegration angenähert werden. An dieser Stelle soll jedoch nur die Rechteckintegration betrachtet werden.

Dadurch ergibt sich:

$$u(k) = K * [e(k) + To/Ti \sum_{i=0}^{k-1} e(i) + Td/To\ (e(k) - e(k-1))]$$

(2.2) Digitaler nichtrekursiver PID-Algorithmus (To = Abtastzeit)

Dies ist eine nichtrekursive Form eines Regelalgorithmus. Denn zur Bildung der Summe müssen alle vergangenen Regelabweichungen e(k) gespeichert werden. Da der Wert u(k) der Stellgröße das Ergebnis ist, wird dieser Algorithmus auch "Stellungsalgorithmus" genannt. Dieser nicht rekursive Stellungsalgorithmus des PID-Reglers ist wegen des großen Speicherbedarfs in der Praxis ungeeignet. Zur Programmierung auf Digitalrechnern sind rekursive Algorithmen zweckmäßiger. Bei diesen Algorithmen wird der momentane Stellwert u(k) aus dem letzten Stellwert u(k-1) und aus Korrekturtermen berechnet. Zur Ableitung des rekursiven Algorithmus subtrahiert man von (2.2)

$$u(k-1) = K * [e(k-1) + To/Ti \sum_{i=0}^{k-2} e(i) + Td/To\ (e(k-1) - e(k-2))]$$

und erhält als PID-Regelalgorithmus

$$u(k) - u(k-1) = Q0 * e(k) + Q1 * e(k-1) + Q2 * e(k-2)$$

(2.3) Digitaler rekursiver PID-Algorithmus

mit den Koeffizienten

$$Q0 = K * [1 + Td/To]$$

$$Q1 = - K * [1 + 2Td/To - To/Ti]$$

$$Q2 = K * Td/To$$

Es wird also nur die momentane Änderung der Stellgröße

$$du(k) = u(k) - u(k-1)$$

berechnet. Deshalb wird dieser Algorithmus auch mit "Geschwindigkeitsalgorithmus" bezeichnet.
Ein Vorteil des Regelalgorithmus in rekursiver Form ist, daß beim Umschalten von Hand auf Automatikbetrieb keine besonderen Maßnahmen zum stoßfreien Umschalten erforderlich sind.

3. Genauigkeitsbetrachtungen

In Abhängigkeit von der vom System geforderten Gesamtgenauigkeit ist die Wortlänge der analog/digitalen bzw. der digital/analogen Wandlung festzulegen. Die berechneten Größen bzw. Zwischengrößen sind hinsichtlich der Genauigkeit unkritisch, müssen jedoch gegen Null und unendlich mit Grenzwerten abgefangen werden.

Vor der Abtastung ist wie bei jeder Signalverarbeitung ein Anti-Aliasing Filter notwendig. Die Steuerspannung ist wegen der Periodizität diskreter Folgen in der Bandbreite zu begrenzen.

4. PID-Regler für MODULAR-4

4.1. Echtzeit PID-Regler mit Programm 184

Zu dieser Application-Note ist ein Echtzeitprogramm von SORCUS Computer erhältlich, das den oben abgeleiteten Algorithmus benutzt. Die Reglerkoeffizienten Q0 bis Q2 werden dem Programm über Parameter übergeben. Alle drei Koeffizienten müssen dem Programm im Long-Integer-Format (32-Bit) übergeben werden. Um die Rechengenauigkeit zu erhöhen, müssen die Koeffizienten vor der Übergabe an das Echtzeitprogramm mit 1024 multipliziert werden.

Dieses Programm dient lediglich zur Demonstration, es ist jedoch voll funktionsfähig. Deshalb wird derzeit zum Einlesen der geregelten Größe nur das SPB-Modul M-AD16-3 unterstützt und zur Ausgabe der Stellgröße das SPB-Modul M-DA4-4. Beide Module haben eine Auflösung von 12-Bit.
Der Sollwert wird über Parameter eingestellt und darf zwischen 0 und 4095 liegen.

Start des Programms 184

Echtzeit PID-Regelung

a) Laden und Installieren des Programms 184
b) Initialisieren der Parameter
   - Sollwert einstellen
   - Abtastintervall einstellen
   - Regler Koeffizienten übergeben
c) PID-Regler durch Aufruf der Prozedur 6 starten. Der Regler ist unmittelbar nach Aufruf der Prozedur 6 (vom PC aus oder von einem anderen Echtzeitprogramm auf der Karte) aktiv.
   Durch Aufruf der Prozedur 8 kann das Programm abgebrochen werden.

4.2. Simulation

Es besteht die Möglichkeit, den Regler über den PC mit Eingangsdaten zu versorgen, und so das Verhalten des Reglers bei bestimmten Eingangssignalen (Sprungfunktion -> Sprungantwort des Reglers) zu analysieren. Die neue Stellgröße kann dazu vom PC abgeholt werden und evtl. grafisch dargestellt werden.

PID-Regler Simulation

a) Laden und Installieren des Programms 184
b) Initialisieren der Parameter
   - Regler Koeffizienten Q0 bis Q1 übergeben
   - ersten Eingangswert übergeben
c) Regler durch Aufruf der Prozedur 10 initialisieren
d) Ausgangswert des Reglers durch Aufruf der Prozedur 12 berechnen lassen. Der Status des Programms geht während der Berechnung auf "1" und nach der Berechnung und dem Eintragen des Ergebnisses in die Parameter auf "0".
e) Ergebnis auslesen (erst wenn Status des Programms wieder auf "0" gegangen ist). Evtl. nächsten Eingangswert übergeben und bei Punkt d) weitermachen.

4.3. Installierung

Programm	Task	Source	Daten	Anmerkung
184 (=B8H)	23	TIMER-B	keine	

Programm 184 muß unter Task 23 (Task 17H) installiert werden. Das Programm darf zur Zeit nur einmal auf einer MODULAR-4 Karte installiert werden.

4.4. Die Prozeduren

Nr.	Funktion
6	Diese Prozedur initialisiert und startet den Regler und den verwendeten Timer. Nach Aufruf dieser Prozedur sind alle eingestellten Parameter aktiv. Zum Abbrechen des Regelvorganges muß die Prozedur 12 benutzt werden.
8	Diese Prozedur bricht einen mit Prozedur 6 gestarteten Reglerlauf ab.
10	Diese Prozedur initialisiert den Regler. Diese Prozedur muß benutzt werden, wenn der Regler mit Daten vom PC gespeist wird (siehe Prozedur 12).
12	Prozedur 12 wird nur zur Reglersimulation benutzt. Über die entsprechenden Parameter wird ein Eingangswert an den Regelalgorithmus übergeben. Danach wird der neue Ausgangswert des Reglers in die dafür vorgesehenen Parameter geschrieben. Wenn diese Prozedur zum erstenmal benutzt wird, muß der Regler vorher mit der Prozedur 10 initialisiert werden.

4.5. Die Parameter

Nr.	Init	Funktion
0	0	Status: 0 = Programm bereit 1 = Regler läuft 2 = Programm abgebrochen
1	1	Steckplatz des Moduls, das die geregelte Größe (Istwert) einlist. Dieser Parameter wird zur Zeit nicht benutzt. Das Modul muß zur Zeit auf Steckplatz 1 stecken.
2	12	Typ des Moduls, das zum Einlesen benutzt wird. Dieser Parameter wird zur Zeit nicht benutzt. Es wird nur Typ 12 (M-AD16-3) unterstützt.
3	0	Kanalnummer, über den eingelesen wird. Dieser Parameter wird zur Zeit nicht benutzt. Es wird immer über Kanal 0 eingelesen.
4	2	Steckplatz des Moduls, das die Stellgröße ausgibt. Dieser Parameter wird zur Zeit nicht benutzt. Das Modul muß zur Zeit auf Steckplatz 2 stecken.
5	9	Typ des Moduls, das zur Ausgabe benutzt wird. Dieser Parameter wird zur Zeit nicht benutzt. Es wird nur Typ 9 (M-DA4-2) unterstützt.
7	255	Trigger, der den Regler aktiviert. Dieser Parameter wird zur Zeit nicht benutzt. Der Regler kann nur über Prozedur 6 aktiviert werden.
8	0	Triggerflanke. Wird zur Zeit nicht benutzt
9	250	TIMER-B Data
10	7	TIMER-B Code für Verteiler (1 bis 7, entspr. 4, 10, 16, 50, 64, 100, 200
11	200	TIMER-B Teiler (1...255)
12	1	TIMER-B Teiler (1...255) **Das Abtastintervall To ergibt sich zu:** **0,4 * PAR-9 * Vorteiler * PAR-11 * PAR-12 [µs]**
13	0	Niederwertigste-Byte von Q0*
14	0	... (Q0* = Q0 * 1024)
15	0	...
16	0	Höchstwertigste-Byte von Q0* Nr. Init Funktion
17	0	Niederwertigste-Byte von Q1*
18	0	... (Q1* = Q1 * 1024)
19	0	...
20	0	Höchstwertigste-Byte von Q1*
21	0	Niederwertigste-Byte von Q2*
22	0	... (Q2* = Q2 * 1024)
23	0	...
24	0	Höchstwertigste-Byte von Q2*
25	255	Low-Byte des Sollwertes
26	7	High-Byte des Sollwertes
27	0	Low-Byte des Eingangswertes des Reglers <1>
28	0	High-Byte des Eingangswertes des Reglers
29	0	Low-Byte des Ausgangswertes des Reglers <1>
30	0	High-Byte des Ausgangswertes des Reglers

<1> Wird nur für die Reglersimulation verwendet (Prozedur 12).